Design Science Research Methods and Patterns

Innovating Information and Communication Technology

Second Edition

Design Science Research Methods and Patterns

Innovating Information and Communication Technology

Second Edition

Vijay K. Vaishnavi
Georgia State University

William Kuechler, Jr.
University of Nevada at Reno

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business
AN AUERBACH BOOK

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

© 2015 by Taylor & Francis Group, LLC
CRC Press is an imprint of Taylor & Francis Group, an Informa business

No claim to original U.S. Government works
Version Date: 20150402

International Standard Book Number-13: 978-1-4987-1526-3 (eBook - PDF)

This book contains information obtained from authentic and highly regarded sources. Reasonable efforts have been made to publish reliable data and information, but the author and publisher cannot assume responsibility for the validity of all materials or the consequences of their use. The authors and publishers have attempted to trace the copyright holders of all material reproduced in this publication and apologize to copyright holders if permission to publish in this form has not been obtained. If any copyright material has not been acknowledged please write and let us know so we may rectify in any future reprint.

Except as permitted under U.S. Copyright Law, no part of this book may be reprinted, reproduced, transmitted, or utilized in any form by any electronic, mechanical, or other means, now known or hereafter invented, including photocopying, microfilming, and recording, or in any information storage or retrieval system, without written permission from the publishers.

For permission to photocopy or use material electronically from this work, please access www.copyright.com (http://www.copyright.com/) or contact the Copyright Clearance Center, Inc. (CCC), 222 Rosewood Drive, Danvers, MA 01923, 978-750-8400. CCC is a not-for-profit organization that provides licenses and registration for a variety of users. For organizations that have been granted a photocopy license by the CCC, a separate system of payment has been arranged.

Trademark Notice: Product or corporate names may be trademarks or registered trademarks, and are used only for identification and explanation without intent to infringe.

Visit the Taylor & Francis Web site at
http://www.taylorandfrancis.com

and the CRC Press Web site at
http://www.crcpress.com

To my family for their love and support

V.K.V.

"The professional schools will reassume their professional responsibilities just to the degree that they can discover a science of design [design science], a body of intellectually tough, analytic, partly formalizable, partly empirical, teachable doctrine about the design process."

Herbert Simon

Contents

PART II PATTERNS*

* The prefix M indicates that the pattern is a meta-level pattern, applicable to multiple stages in the research process. Meta-level patterns are explained in more detail at the end of the section "Using Patterns: The Design Science Research Cycle Revisited" in Chapter 6.

Foreword

I am a scholar in information systems: a frequent researcher, author, reviewer, conference program chair, and journal editor. In these roles, I could complain that Vaishnavi and Kuechler have left tracks across my desk for many years now. These tracks are not only the traces they have themselves left through their influences on my own work, but also through their influence on so many other scholars who cite their research, follow their methods, and emulate their ways of thinking about design science research. But of course such a complaint would be tongue-in-cheek, a brotherly jest meant to thinly conceal my admiration.

In the pages that follow, Vaishnavi and Kuechler continue to build their intellectual leadership in their application of design science research to the field of information and communication technology. At its genesis in the 1990s, they called it "improvement research." That label more clearly singled out their vision that researchers could not just improve their own knowledge through their research, but along the way they could build useful artifacts to directly improve the world being researched. Many of us found this idea inspiring. This reconceptualization of research was more than just a novel idea, it was a noble goal.

At that early stage, the need for *credentialing* grew in importance for improvement research. The idea of scholarly learning as a part of a process that builds artifacts was new, and we confronted some of our university colleagues who were justifiably suspicious about how the discovery of new knowledge might result from generative activities like the construction of IT artifacts. It was not sufficient to relabel "improvement research" as "design science research." At that time, this generative way of creating knowledge needed better scholarly credentials: rigorous methodology, strong theoretical grounds, and new ways of theorizing about the phenomena of interest.

In the first edition of this book, Vaishnavi and Kuechler delivered the early foundations for these first two credentials. Their ground-breaking methodology might best be described as *clean*: fundamental, complete, consistent, simple, and above all, practical. It is ideal for its purpose: think about the problem, think about

a good solution, *try it out*, and then think about what just happened. It is consistent with the scientific method in a way that would have deeply satisfied John Dewey. No wonder their methodology has been widely adopted by design science researchers around the world; and as a result it has helped to sustain myriad research discoveries through some of the toughest academic referee processes.

A second early foundation in their first edition was their inspired use of the concept of *patterns*. There are many kinds of theories described in this book: kernel theories, design theories, explanatory theories, predictive theories, mid-range theories, and so on. While they make no claims about any *grand* theories in design science research, patterns, if not indeed a grand theory of design science, can provide a suitable substitute until one comes along. The notion of patterns provides a tool for Vaishnavi and Kuechler to explain the many different kinds of things that happen in the research arena that comprises design science research. In their hands, patterns provide a pretty universal way to explain how people (like researchers) make sense of how other people assemble and make use of systems of computers, networks, and other technologies.

In this second edition, Vaishnavi and Kuechler move forward in refining these two major foundations. These ideas are updated in keeping with a decade of fast-paced development of the design science research community. But they have kept their methodology clean, and their application of patterns continues to serve well through the recent years of development. Both ideas have stood the test of time. This edition does provide more elaborate descriptions of the methodology, and additional *how-to* detail, as we should expect from additional years of experience.

Perhaps more importantly, Vaishnavi and Kuechler provide more explicit foundations for the third kind of credential: new ways of theorizing. Since we are dealing with a different way of researching, it should not be surprising that we must also deal with a different way of theorizing. Many design science researchers struggle not only to explain the theoretical contribution of their works, but also to convince critical readers that what they are explaining is justifiable as theory at all. With this new edition, Vaishnavi and Kuechler not only explain the nature of different kinds of theory in design science research, they step through a clear and detailed example of how to frame the theorizing aspects of the research activity within their methodology. This example (in Chapter 5) also shows how their methodology applies to more business-oriented design, such as business process design, as well as information and communication technology.

Vaishnavi and Kuechler, in expanding our understanding of the relationship between theory and design science research, offer a bolder view that breaks away from assumptions that design science research contributions need go no further than the consequent artifact. Together with the well-known design-theory kinds of theories, they centralize the kinds of explanatory and predictive theories that design scientists must invent in order to translate more basic natural- and social-science kernel theories into workable, problem-solving artifacts. Early thought in design science research regarded these "mid-range" theories as by-products of the

research (or worse, as waste output). But such theories are important to enable the new knowledge (generated in the research) to be applied to a "class" of artifacts, that is, generalized to a range of other experiences.

Their concept of this new type of design science theory captures an important reverse-flow of knowledge in this form of research: a knowledge flow that is actualized in the iterative aspect of their methodology. This theory flow models how the artifact informs each of the different kinds of theory at play in the research. The production (and re-production) of an artifact helps to revise the design theories: their "design-relevant" explanatory and predictive theories that are to a large extent "invented-here" in each instance of design science research. By centralizing this kind of mid-range theory in design science research, Vaishnavi and Kuechler move us a great deal closer to a complete understanding of and a more appropriate appreciation for, the value of new kinds of theory for the scientific community.

This new edition is not only a refreshed version of the previous edition, but has also been expanded to help bring us onto the current frontiers of design science research. As researchers, the authors are ever mindful of their goals toward not just science, but also *improvement*. True to their noble direction, they have put this thought into this second edition. The first edition was ground-breaking; nevertheless they have still managed to make this second edition a grand improvement.

Richard Baskerville

Preface

The need for this work became clear during the first author's teaching of a research seminar course on design science research (DSR) offered to doctoral students at Georgia State University over the last two decades. The course focuses on research whose purpose is the creation of knowledge that can be used for the development of or the improvement/innovation of information and communication technology (ICT) artifacts. This type of research is identical in technique and philosophy to that conducted by numerous other research communities including engineering and computer science, and yet it remains difficult to find good published material that can be used for teaching this type of research. Herbert Simon's book, *Sciences of the Artificial*, is a seminal work that has helped in realizing the uniqueness and importance of this type of research. The book, however, does not provide much guidance on how to perform such research. The launching of the yearly international conference on "Design Science Research in Information Systems and Technology (DESRIST)" in 2006 and the publishing of books/proceedings related to these conferences have generated additional DSR literature but this literature still falls short of providing a comprehensive guidance on the conduct of DSR. The first edition of this book and its major revision, update, and expansion—the current edition—aim to fulfill this need.

The objective of this second edition is to provide a comprehensive understanding of design science knowledge, particularly design theory—a desired form of such knowledge—and how it can be generated through DSR. A unique feature of the book is the presentation of DSR *patterns* to provide guidance on the conduct of DSR. The decision to use patterns to organize the knowledge on how DSR is conducted is based on our interest in patterns, the belief that patterns are an excellent mechanism for organizing and transmitting this type of knowledge, and the second author's positive experience with patterns in over 20 years of IT system design in industry. We firmly believe that over time we can find a set of patterns that can provide firm direction for conducting design science research in ICT, which will be useful particularly to a new researcher. In time, experienced design science

researchers will hopefully also find this a useful explication and codification of some of the techniques they have used. We trust that the 84 patterns presented in the book, 20 of them new to this edition, are a good start in this direction.

Use of the Book

The book can be used as a general book, text book, or a reference book for a DSR methods course in the field of ICT. Thus the book can be used at the doctoral level, master's level, and senior undergraduate level in the ICT field that includes information systems, information sciences, information technology, and computer science. The book will also be useful for students conducting research in the various field of engineering.

Acknowledgments

The ideas presented in this work have been largely shaped and influenced by the students in the doctoral level research seminar course that the first author has taught at Georgia State University. We would like to particularly mention the students in the 1996 offering of the course: Gayle Dixon-Randall, Paul Cule, David Gefen, Rich Klein, Bill Kuechler, Lynette Kvasney, George Littlejohn, and Linda Wallace; and the 1998 offering of the course: Ashley Bush, Gordon Depledge, Huoy Khoo, David Kuechler, Alisha Malloy, Amrit Tiwana, Rustam Vahidov, and Jie Yin. We would like to acknowledge their contribution to the patterns in this work and would like to thank them for their patience in learning the research process and tools through a systematic search for the desired knowledge.

Even before the first edition of the book was published, the course notes—the basis for the first edition—underwent thorough classroom use and testing. In this regard, acknowledgements are also due to the students who used these notes and offered their constructive suggestions: doctoral students at Georgia State University (2000–2007)—Punit Ahluwalia, Chad Anderson, Gayle Beyah, Andrew Burton-Jones, Lan Cao, Ricardo Checchi, Gloria Chen, Yide Chen, Sunyoung Cho, Karlene Cousins, Michael Cuellar, Yi Ding, Stephen Du, Alina, Dulipovici, Po-An Hsieh, Ghiyoung Im, Radhika Jain, Vijay Kasi, Jong Kim, Lei Li, Kannan Mohan, Nannette Napier, Chongwoo Park, Stacie Petter, Adriane Randolph, Benoit Raymond, Rebecca Rodecker, Robert Samsbury, Wei Shi, Ashish Singh, Sweta Sneha, Hirotoshi Takeda, Radu Vilas, Jijie Wang, Peng Xu, and Guanzhi Zheng; graduate level research seminar course students taught by the first author at the Indian Institute of Technology, Delhi, India (2004)—Abhishek Agrawal, Anish Bansal, Harsh Dand, Gaurav Gupta, Ben Imchen, Amit Khemka, Rishi Kumar, Shubham Markhand, Abhay Negi, and Ravi Rathore, and C. H. Shiva Shankar.

The first edition of the book has been used continuously as the text for a doctoral level DSR course taught by the first author in a seminar format. We gratefully acknowledge the suggestions and ideas for improvement provided by students who used the book: Sayed Almohri, Arun Aryal, Bryan Beckman, Langtao Chen, Tianjie Deng, Patrick Givens, Thomas Gregory, Mala Kaul, Hyung Koo Lee, Thomas Jong, Jong Seok Lee, Stefanie Markham, Eun Hee Park, Claudia Ruiz, Neetu Singh, Sumantra Sarkar, and Youyou Tao. The current edition has attempted to incorporate all the suggestions and feedback by the students. We are sure this work will be further improved by the contributions made by the current and future classes taking this course. While acknowledging the contributions to this work from the DSR seminar students, we take the responsibility for all errors or omissions.

Vijay Vaishnavi
William Kuechler Jr.

Chapter 1

Introduction

Until recently, it has been considered by many researchers to be impossible to *teach* research, at least in the same way that less complex skills such as reading or basic mathematics can be taught. This is because the practice of research is a complex activity requiring the extended use of several poorly understood cognitive activities such as creativity and intuition; research is at best a semi-structured activity. There are no algorithmic "recipes" for performing research and even the methodologies for research sometimes presented (including those in this book) are guidelines at best.

In the past, those wishing to become researchers were expected to serve an apprenticeship, frequently by way of graduate study at a university, usually under the close tutelage of a senior researcher in the field. During the course of the apprenticeship that extended over a period of years, the student researcher would gradually become "socialized" to the paradigmatic community in which they worked. If successful, the student was inculcated with an intimate and frequently tacit (i.e., internalized and largely unstated) understanding of the research field including:

- The important research questions
- The research methods considered by the community to be legitimate for exploring the research questions
- The prior research that provided the grounding of the field
- Knowledgeable colleagues
- Acceptable outlets for the research, including journals and conferences

This method of training researchers is still the dominant practice in many fields of research that are considered "paradigmatic"—areas that typically have a significant history (such as the hard sciences) and a dominant set of research questions,

method(s) for exploring them, and outlets for disseminating new knowledge. In contrast, information systems (IS) along with many other disciplines centered on information and communication technology (ICT) are currently multi-paradigmatic: They draw research questions, methodologies, and grounding philosophies from multiple fields that are loosely united under a common interest in *understanding the way in which human–computer systems are developed, produce and process information, and influence the organizations in which they are embedded*. We refer to these fields henceforward as ICT fields or disciplines.

It is because ICT is multi-paradigmatic that we felt the need to write this book. We believe researchers in ICT fields need a thorough grounding in each of the variety of research philosophies and techniques practiced in their field and it simply is not practical for any student to undertake a multi-year apprenticeship in each of the major ICT research paradigms. Moreover, design science research (DSR) as practiced in ICT fields is significantly different from the design-based research practiced in other fields such as architecture or industrial design; the need for and manner of validation of research results, for example, is more emphasized in IS, human–computer interface, and many branches of software engineering due to the grounding of those fields in management science, psychology, and other statistically based descriptive disciplines.

The reason DSR is applicable to ICT is due to some of the types of research questions that occur naturally to the field. Human–computer information producing and processing systems are by their nature complex and grounded in multiple disciplines. Questions frequently arise that have a sparse or non-existent theoretical background and exploring these is where DSR—exploring by creating—excels. Cultures at all technological levels have always had the ability to create artifacts that produce useful results without fully understanding how the artifacts work or without being able to elucidate the principles that contribute to the making of good (or better) examples of the artifacts. Bridges, boats, and waterwheels are just three examples of important artifacts that were produced, used, and highly valued thousands of years before the physical principles underlying them were understood in a manner that enabled methodical, consistent performance improvement. In our culture, information systems are frequently developed and used in a similar information vacuum: They do some useful work but we are not really sure how to make them better; they have significant effects on people and organizations, many unanticipated, and most poorly understood. Some schools of thought "instinctively" veer away from questions that lack a developed theoretical base to direct their experimentation. DSR, on the other hand, thrives in just this sort of theoretical terra incognita that many areas of ICT still remain.

Another reason we felt emboldened to write this book is that we felt the technique of the use of patterns—a formalized way of recording experience—would enable the written as opposed to the verbal and imitative communication of at least some of the concepts, techniques, and their subtle interrelationships that make up research praxis. Tutorials on research in any field are rare, and the use of *patterns*

in such a tutorial is unique as far as we know. However, the use of patterns to communicate contextually rich information will be familiar to many ICT fields such as software and computer engineering.

This book is structured as follows: Chapter 2 is an introduction to DSR in ICT that describes DSR in relation to other IS research paradigms with a longer history such as positivist and interpretivist research. IS is the specific ICT field of the authors, but the discussion is immediately applicable to ICT fields in general. Chapter 2 also relates DSR in ICT to DSR as practiced in other areas of intellectual exploration where it has a much longer history. A primary contribution of the chapter is the introduction of the DSR cycle that is developed as the general method used for the practice of DSR and an understanding of the expected outputs of DSR, particularly design theory. At the beginning of the patterns section (Part II) of the book, the DSR cycle is presented as a "road map" for the use of the patterns presented in the actual practice of DSR.

Chapters 1 and 2 are intended to give the readers an overview of and "feel for" DSR even if the paradigm is unfamiliar to them. Those coming to ICT research from management science or other business backgrounds will find much of the material on DSR new and are urged to read the introductory chapters carefully before proceeding to the patterns section (Part II). Those from a technical background such as engineering or physical science* will see many similarities to these areas of investigation, but will also, on careful reading, note significant differences between DSR as practiced in ICT and in other fields.

Chapter 3 places DSR in the historical context of ICT systems research and ICT artifact development and refinement. The DSR cycle is abstracted to become a framework for understanding the progress of entire fields of technological research and development over extended periods of time. Specifically for this book, Chapter 3 is intended to give readers new to DSR in ICT a sense of how syncretistic the DSR paradigm is. Unlike many academic fields where cross-discipline conceptualizations are novel and even discouraged (you will not get published) DSR actively draws from any field, the theory or practice of which gives insight into the working of or construction of a problem-solving artifact. Typical areas, from which DSR in ICT draws, as illustrated in the example in Chapter 3, are computer science, psychology, and human–computer interface—which itself is a highly derivative field.

Chapters 4 and 5 are new to the second edition and expand the treatment of *the development of theory in DSR*. In the 7 years since the publication of the first edition of this text, theory development in DSR—what it is, that is, the types of theory that are relevant to DSR, how to do it, and even whether or not theory building has a place in DSR—has been a large and ongoing part of the published community dialogue for the field. We strongly believe that DSR should strive to generate theory

* Other fields, such as education also utilize DSR (DSDE 1997); however, in practice few students with a background in education proceed on to graduate work in ICT fields.

and in Chapter 4 we present a taxonomy of DSR theory types (in the context of the IS discipline) that explains the different types of theories and relates them to traditional notions of theory from the hard sciences. In Chapter 5, an example is given of a DSR project and the development of both design theory and mid-range theory from that project.

Part II of the book contains the research patterns themselves. At the beginning of this section is a short chapter (Chapter 6) on "Using Patterns to Illuminate Research Practice." It begins by introducing the concept of patterns as it is used in this book. The qualifier "as used in this book" is necessary since, although patterns are used in many fields for many purposes, a precise general definition has proven elusive. The chapter then draws on concepts from the introductory chapters and outlines a methodology for the practice of DSR that is keyed to the patterns presented in the remainder of the book. The patterns are grouped by chapters with each chapter being applicable to one or more phases of the research methodology. To provide structure to the patterns and thereby guide their use, this new edition classifies all patterns, except those for *evaluation and validation* and *publishing*, according to their types such as *preliminaries* or *extrapolation*. In addition, each pattern includes a *motivation* section that answers why the pattern should be used.

New again to this edition is the introduction of concepts from TRIZ (Wikipedia-TRIZ 2014), the well-regarded problem solving and analysis method from the engineering discipline. TRIZ brings an algorithmic approach to the invention of new systems and the improvement of existing ones. Any relationships between patterns in the book and the TRIZ inventive principles have been documented. Some of the TRIZ inventive principles such as *nesting* and *intermedia*ry have been adapted and developed as new DSR patterns.

The book concludes with a third part, Part III, in which examples of published DSR, including some widely cited papers, are used as exemplars for their analysis. The analysis is carried out in terms of (1) knowledge contribution (type of knowledge contribution and assessment of any design theory advanced) and (2) the patterns that were used (or could have been used) in the reported research. The first part of the analysis is new to this edition; also, two additional examples have been included for analysis.

The authors have practiced DSR in the ICT fields of IS and computer science for much of their careers and have found it to be rewarding both as an intellectual practice and in terms of the research results obtained. Although this is not the place for an extended discussion of the history of ICT research, we feel safe in saying that the field is dynamic, multi-paradigmatic, and IS, in particular, generates much current DSR discussion as it transitions from a managerial to a technological focus (Iivari 2003). It is in the exploration of the technology of information and communications systems, better understanding how information systems do what they do and how to create systems with novel or better performance even in the absence of a strong theoretical grounding, that DSR is the paradigm of choice.

The book can be used as a general book, textbook, or reference book on DSR in ICT. As a general book, we recommend reading the first part of the book followed by a quick review of the rest of the book. As a textbook, we recommend reading the entire book and the actual use of patterns (Part II and Part III of the book) in carrying out a research project. As a reference book, we recommend reading the first part of the book, getting familiarity with the rest of the book, and then using the patterns on as-needed basis.

References

DSDE (1997). Special Issue of Design Studies on Design Education. *Design Studies* 18(3): 319–320.

Iivari, J. (2003). "The IS CORE VII: Towards Information Systems as a Science of Meta-Artifacts." *Communication of the AIS* 12 (October), Article 37.

Wikipedia-TRIZ (2014). "TRIZ." URL: http://en.wikipedia.org/wiki/TRIZ. (last accessed on January 29, 2015).

DESIGN SCIENCE RESEARCH METHODOLOGY

I

"It is like a voyage of discovery into unknown lands, seeking not for new territory but for new knowledge. It should appeal to those with a good sense of adventure."

Frederick Sanger

"Those who are enamored of practice without theory are like a pilot who goes into a ship without rudder or compass and never has any certainty where he is going. Practice should always be based on a sound knowledge of theory."

Leonardo da Vinci

The objective of the four chapters in this part of the book is to discuss design science research methodology for information and communication technology, and related topics. Here are the major highlights of these chapters: Chapter 2 discusses Design Science Research Cycle that models the general process followed in conducting design science research and the expected outputs of DSR, particularly design theory. Chapter 3 expands the Design Science Research Cycle to serve as a framework for understanding how research communities work together to advance a field of design science research. Chapters 4 and 5 focus on in-depth treatment of theory development in design science research (DSR). DSR theory is discussed in these chapters in the context of information systems but it is equally applicable to all the disciplines within information and communication technology.

Chapter 2

Introduction to Design Science Research in Information and Communication Technology*

Overview of Design Science Research

Research

Drawing heavily from Kuhn (1996; first published in 1962) and Lakatos (1978), research can be very generally defined as an *activity* that contributes to the *understanding* of a *phenomenon*. In the case of design science research (DSR), all or part of the phenomenon may be *created* as opposed to naturally occurring. The *phenomenon* is typically a *set of behaviors of some entity* (entities) that is found *interesting* by the researcher or by a group—a research community. *Understanding* in most western research communities is *valid (true) knowledge that may allow prediction* of the behavior of some aspect of the phenomenon. Thus, research must lead to contribution of knowledge—usually in the form of a theory—that is *new* and *valid (true)*.

* Adapted from the AIS design science research page developed and edited by the authors at http://desrist.org/design-research-in-information-systems/

For this contribution to be valued and accepted by a research community through its publication as research paper(s) or patent(s), it must also be something that is *interesting* to the research community (Gregor and Hevner 2013; Wilson 2002).

The set of activities a research community considers appropriate to the production of understanding (knowledge) are its research methods or techniques. Historically, some research communities have been observed to have nearly universal agreement on the phenomenon of interest and the research methods for investigating it; we term these *paradigmatic* communities. Other research communities are bound into a nominal community by overlap in sets of phenomena of interest and/or overlap in methods of investigation. We term these *pre-paradigmatic* or *multi-paradigmatic* research communities. Information and communication technology (ICT)-based disciplines such as *information systems* (IS) are excellent examples of multi-paradigmatic communities.

Design

Design means "to invent and bring into being" (Webster's Dictionary and Thesaurus 1992). Thus, design deals with creating some new artifact that does not exist. If the knowledge required for creating such an artifact already exists, then the design is *routine*; otherwise, it is *innovative*. Innovative design may call for the conduct of research (DSR) to fill the knowledge gaps and result in research publication(s) or patent(s).

Design Science and DSR

The design of artifacts is an activity that has been carried out for centuries. This activity is also what distinguishes the professions from the sciences. "Schools of architecture, business, education, law, and medicine, are all centrally concerned with the process of design" (Simon 1996—first edition published in 1969). However, in this century natural sciences almost drove out the design from professional school curricula in all professions, including business, with exceptions for management science, computer science, and chemical engineering—an activity that peaked two or three decades after the Second World War (Simon 1996).

Simon sets out a prescription for professional schools that include schools of business (in which most IS departments are housed): "... The professional schools will reassume their professional responsibilities just to the degree that they can discover a science of design [*design science*], a body of intellectually tough, analytic, partly formalizable, partly empirical teachable doctrine about the design process."

To bring the design activity into focus at an intellectual level, Simon (1996) makes a clear distinction between "natural science" and "science of the artificial" (also known as *design science*): A *natural science* is a body of knowledge about some class of things—objects or phenomenon—in the world (nature or society) that describes and explains how they behave and interact with each other. A *science of the*

artificial (*design science*), on the other hand, is a body of knowledge about the design of artificial (man-made) objects and phenomena—artifacts—designed to meet certain desired goals. Simon further frames the design of such artifacts in terms of an *inner environment*, an *outer environment*, and the *interface* between the two that meets certain desired *goals*. The outer environment is the total set of external forces and effects that act on the artifact. The inner environment is the set of components that make up the artifact and their relationships—the organization—of the artifact. The behavior of the artifact is constrained by both its organization and its outer environment. The bringing-to-be of an artifact, components, and their organization, which interfaces in a desired manner with its outer environment, is the design activity. The artifact is "structurally coupled" to its environment and many of the concepts of structural coupling that Varela (1988) and Maturana and Varela (1987) have developed for biological entities are applicable to designed artifacts.

In a perspective analogous to considering the design of artifacts as the crafting of an interface between the inner and outer environment, design can be thought of as a mapping from functional space—a functional requirement constituting a point in this multidimensional space—to attribute space, where an artifact satisfying the mapping constitutes a point in that space (Takeda et al. 1990). *Design science* then is knowledge in the form of constructs, techniques and methods, models, well-developed theory for performing this mapping—the know-how for creating artifacts that satisfy given sets of functional requirements. *DSR* is research that creates this type of missing knowledge using design, analysis, reflection, and abstraction.

Can Design Be Research?

The question we intend to answer in the affirmative is: can design (i.e., artifact creation) ever be considered an appropriate technique for conducting research in IS and other ICT-based disciplines so as to create design science knowledge? We will pursue this specific question in the following sections. For the remainder of this section, we discuss the question in the abstract—can design be research?—using as exemplars communities other than ICT where the question of whether or not design is a valid research technique has for many years been a resounding Yes!

Owen (1997) discusses the relation of design to research with reference to a conceptual map of disciplines (Figure 2.1) with two axes: symbolic/real and analytic/synthetic. The horizontal axis of the map position disciplines according to their defining activities: disciplines on the left side of the map are more concerned with exploration and *discovery*. Disciplines on the right side of the map are characterized more by invention and *making*. The map's vertical division, the symbolic/real axis, characterizes the nature of the subjects of interest to the disciplines—the nature of the phenomena that concerns the research community. Both axes are continua and no discipline is exclusively concerned with synthesis to the exclusion of analytic activities. Likewise, no activity is exclusively concerned with the real to the exclusion of the symbolic although the strong contrast along this axis between

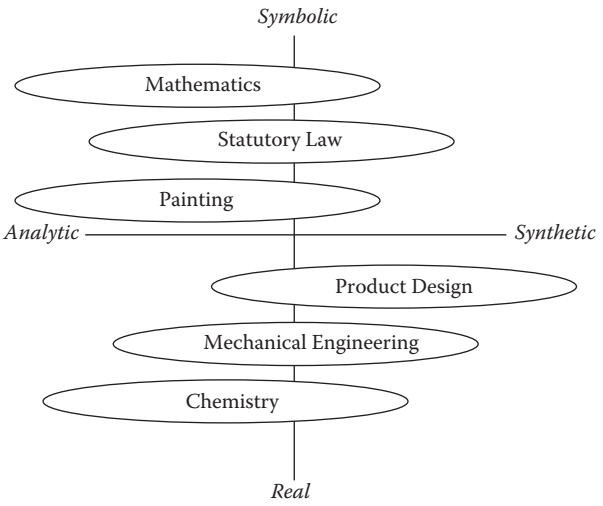


Figure 2.1 A conceptual map of disciplines. (From Owen, C., *Journal of the Japanese Society for the Science of Design*, 5(2): 36–45, 1997.)

the physical science of chemistry (real) and the abstract discipline of mathematics (symbolic) is strongly and accurately indicated in the diagram.

The disciplines that lie predominantly on the synthetic side of the map are either design disciplines or the design components of multi-paradigmatic disciplines. Design disciplines have a long history of building their knowledge base through making—construction (creation) of artifacts and evaluation of the artifacts' performance followed by reflection and abstraction. Architecture is a strongly construction-oriented discipline with a history extending over thousands of years. The architectural knowledge base consists of a pool of structural designs that effectively encourage a wide variety of human activities and has been accumulated largely through the post-hoc observation of successful constructions (Alexander 1964). Aeronautical engineering provides a more recent example. From the Montgolfier balloon through the First World War, the aeronautical engineering knowledge base was built almost exclusively by analyzing the results of intuitively guided designs—experimentation at essentially full scale.

Owen (1997) further presents a general model for generating and accumulating knowledge (Figure 2.2) that is helpful in understanding design disciplines and the DSR process: "Knowledge is generated and accumulated through action. Doing something and judging the results is the general model ... the process is shown as a cycle in which knowledge is used [creatively] to construct (create) works, and works are evaluated to build knowledge." In addition, reflection and abstraction play a role in the knowledge-building process. While knowledge building through construction is sometimes considered to lack rigor, the process is not unstructured. The channels

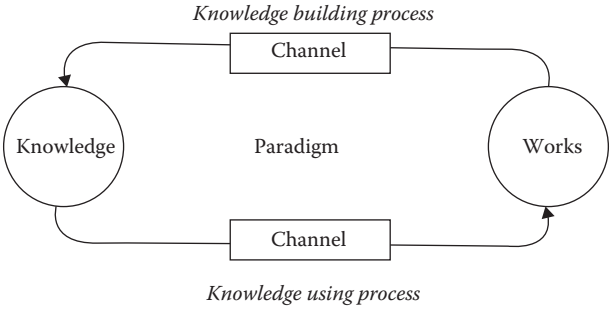


Figure 2.2 A general model for generating and accumulating knowledge. (From Owen, C., *Journal of the Japanese Society for the Science of Design,* 5(2): 36–45, 1997.)

in the diagram of the general model are the "systems of conventions and rules under which the discipline operates. They embody the measures and values that have been empirically developed as 'ways of knowing' as the discipline has matured. They may borrow from or emulate aspects of other disciplines' channels, but, in the end, they are special to the discipline and are products of its evolution" (Owen 1997).

DSR versus Design Research

DSR is a rapidly evolving field. Within the last decade even the most commonly accepted name for the field has changed—from "design research" (DR) to "DSR." As the DSR literature gained breadth and depth, researchers came to understand that the term "design research" had a long prior history as the study of design itself and designers—their methods, cognition, and education. DR is a broad area spanning all design fields, but importantly, does not have the defining feature of DSR: learning through building—artifact creation. IS design science researchers thus (in about 2005–2006, as a scan of the literature will show) widely began to add the distinguishing word "science" to the field designation. The distinction frequently expressed is that DR is research *into* or *about* design whereas DSR is primarily research *using design as a research method* or technique.

DSR when defined as learning through building is not unique to ICT. The fields of education, health care, computer science, and engineering also make extensive use of DSR. DSR in education, where curricula and learning programs are designed and empirically evaluated and in health care, where programs of treatment are designed and empirically evaluated—share the DSR-IS concern for rigorous evaluation and especially for the codification of design science knowledge as design theories to a greater degree than do the technical disciplines of computer science and engineering (Kuechler and Vaishnavi 2012); however, this concern is equally applicable to all ICT disciplines. More information on the history of DSR in IS, especially in North America is available in Kuechler and Vaishnavi (2008).

DSR versus Routine Design

A significant and valid question posed frequently to design science researchers is: How is your research different from a design effort; what makes your work research and not simply state-of-practice design?

We propose that DSR is distinguished from routine design by the *production of interesting (to a community) new and true knowledge*. In a typical *industry* design effort a new product (artifact) is produced, but in most cases, the more successful the project is considered to be, the less is learned. That is, it is generally desirable to produce a new product using state-of-practice application of state-of-practice techniques and readily available components. In fact, most product design efforts in industry are preceded by many meetings designed to "engineer the risk out of" the design effort. The risks that are identified in such meetings are the "we don't know how to do this yet" areas that are precisely the targets of DSR efforts. This is in no way meant to diminish the creativity that is essential to any design effort. We merely wish to point out that routine design is readily distinguished from DSR (within its community of interest) by the intellectual risk, the number of unknowns in the proposed design (missing knowledge).

Attempts at routine design can, however, also lead to DSR. To find out the missing knowledge in a new area of design, it is quite useful to attempt carrying out the design using existing knowledge. This gives the researcher a better feel for the extent of the missing knowledge and the challenges faced in filling the knowledge gaps.

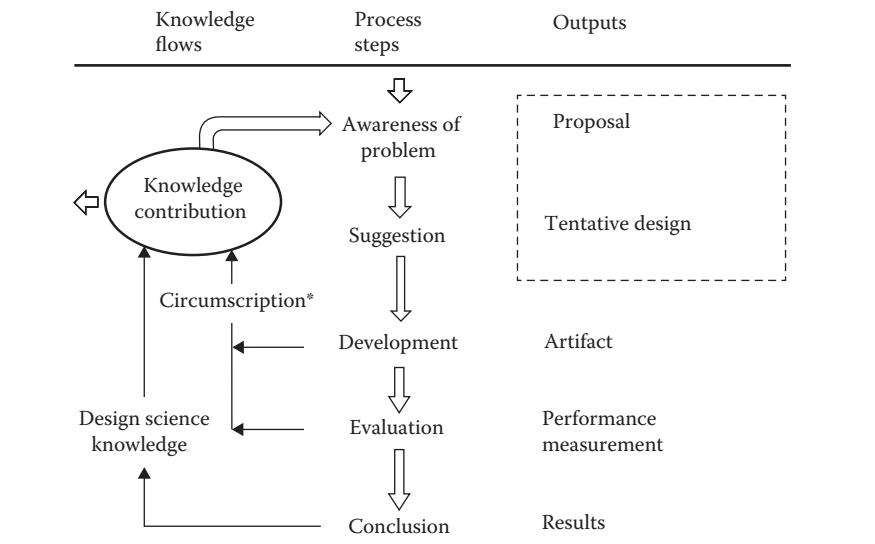
DSR Methodology

A DSR Process Model

In this section, a model of the general process followed by DSR in its multiplicity of as-practiced variants is described. This model is an adaptation of a computable design process model developed by Takeda et al. (1990). Even though the different phases in a design process and a DSR process are similar, the activities carried out within these phases are considerably different. Also, what makes the DSR process model different from the corresponding design process model is the fact that contribution of new (and true) knowledge needs to be a key focus of DSR. The research process model shown in Figure 2.3 can be interpreted as an elaboration of both the knowledge using process and the knowledge building process arrows in Figure 2.2. With respect to Figure 2.3, a typical DSR effort proceeds in phases that are discussed below.

Awareness of Problem

An awareness of an interesting research problem may come from multiple sources including new developments in industry or in a reference discipline. Reading in an



* Circumscription is discovery of constraint knowledge about theories gained through detection and analysis of contradications when things do not work according to theory (McCarthy, 1980)

Figure 2.3 Design science research (DSR) process model (DSR cycle).

allied discipline may also provide the opportunity for application of new findings to the researcher's field. The output of this phase is a proposal, formal or informal, for a new research effort.

Suggestion

The suggestion phase follows immediately behind the proposal and is intimately connected with it as the dotted line around proposal and tentative design (the output of the suggestion phase) in Figure 2.3 indicates. Indeed, in any formal proposal for DSR such as one to be made to the National Science Foundation or an industry sponsor, a tentative design and likely the performance of a prototype based on that design would be an integral part of the proposal. Moreover, if after investing considerable effort on an interesting problem a tentative design or at least the germ of an idea for problem solution does not present itself to the researcher, the idea (proposal) will be set aside. Suggestion is essentially a creative step wherein new functionality is envisioned based on a novel configuration of either existing or new and existing elements. The step has been criticized as introducing non-repeatability into the DSR method since human creativity is still a poorly understood cognitive process. However, the creative step has necessary analogues in all research methods; for example, in positivist research creativity is inherent

in the leap from curiosity about a phenomena to the development of appropriate constructs that operationalize the phenomena and an appropriate research design for their measurement.

Development

The tentative design is further developed and implemented in this phase. The techniques for implementation will, of course, vary depending on the artifact to be created. An algorithm may require the construction of a formal proof to show its correctness. An expert system embodying novel assumptions about human cognition in an area of interest will require software development, probably using a high-level package or tool. The implementation itself can be very pedestrian and need not involve novelty beyond the state-of-practice for the given artifact; the novelty is primarily in the design, not the construction of the artifact.

Evaluation

Once constructed, the artifact is evaluated according to criteria that are always implicit and frequently made explicit in the proposal (awareness of problem phase). Deviations from expectations, both quantitative and qualitative, are carefully noted and *must be tentatively explained*. That is, the evaluation phase contains an analytic sub-phase in which hypotheses are made about the behavior of the artifact. This phase exposes an epistemic fluidity that is in stark contrast to a strict interpretation of the positivist stance; see a later section on "Philosophical Grounding of DSR." At an equivalent point in positivist research, analysis either confirms or contradicts a hypothesis. Essentially, save for some consideration of future work as may be indicated by experimental results, the research effort is over. For the design science researcher, by contrast, things are just getting interesting! Rarely, in DSR, are initial hypothesis concerning behavior completely borne out. Instead, the evaluation phase results and additional information gained in the construction and running of the artifact are brought together and fed back to another round of suggestion (cf. the circumscription arrow of Figure 2.3). The explanatory hypotheses, which are quite broad, are rarely discarded, but rather are modified to be in accord with the new observations. This suggests a new design, frequently preceded by new library research in directions suggested by deviations from theoretical performance. (Design science researchers seem to share Allen Newell's conception (from cognitive science) of theories as complex, robust nomological networks (Newell 1973)). This conception has been observed by philosophers of science in many communities (Lakatos 1978), and working from it Newell suggested that theories are not like clay pigeons, to be blasted to bits with the Popperian shotgun of falsification. Rather, they should be treated like doctoral students. One corrects them when they err and is hopeful they can emend their flawed behavior and go on to be ever more useful and productive (Newell 1990).

Conclusion

This phase could be just the end of a research cycle or is the finale of a specific research effort. The finale of a research effort is typically the result of satisficing, that is, though there are still deviations in the behavior of the artifact from the (multiple) revised hypothetical predictions, the results are adjudged "good enough." Not only are the results of the effort consolidated and "written up" at this phase, but the knowledge gained in the effort is frequently categorized as either "firm"—facts that have been learned and can be repeatedly applied or behavior that can be repeatedly invoked—or as "loose ends"—anomalous behavior that defies explanation and may well serve as the subject of further research. Communication is very important in research (Hevner et al. 2004). Therefore, this phase, as a conclusion of a research effort indicated by the small leftward arrow coming out of knowledge contribution in Figure 2.3, needs to appropriately position the research being reported and make a strong case for its knowledge contribution (Gregor and Hevner 2013). Depending on the type of knowledge contribution and the state of knowledge in the area of research, the expectations on the nature and depth of knowledge contribution outputs can vary; see the next section ("Outputs of DSR").

Cognitive Processes Used in DSR

Figure 2.4 models the cognition that takes place during a DSR cycle. Both DSR and design (Takeda et al. 1990) use abduction, deduction, and circumscription but there is a difference in how these cognitive processes are used. In following the flow

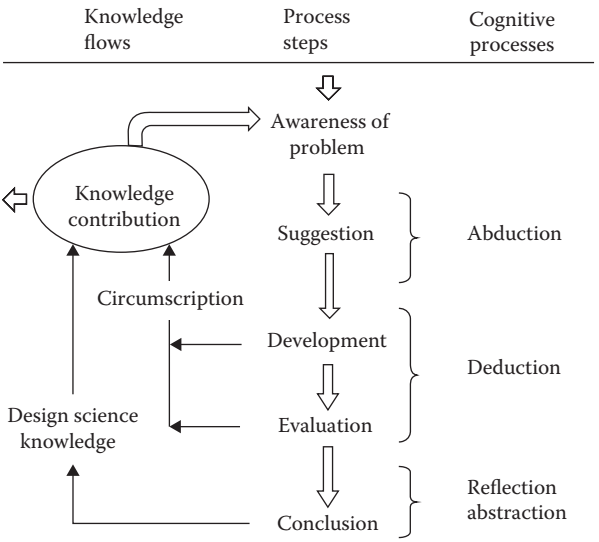


Figure 2.4 Cognition in the design science research cycle.

of creative effort through Figure 2.4, the types of new knowledge that arise from DSR activities and the reason that this knowledge is most readily found during such effort will become apparent.

In this model, the research begins with the *awareness of a problem*. DSR is sometimes called "improvement research" and this designation emphasizes the problem-solving/performance-improving nature of the activity. *Suggestions* for a problem solution are abductively drawn from the existing knowledge/theory base for the problem area (Peirce 1931). These suggestions may, however, be inadequate for the problem or suffer from significant knowledge gaps (which make the problem a research problem). Using existing knowledge, an attempt is made at creatively solving the problem. The solution—a tentative design—is used to implement an artifact in the next phase shown as *development* in the diagram. Partially or fully successful implementations are then *evaluated* according to a functional specification (sometimes implicit) during the *evaluation* stage. *Development*, *evaluation*, and further *suggestion* are frequently iteratively performed in the course of the research effort. The basis of the iteration, the flow from partial completion of the cycle back to *awareness of the problem*, is indicated by the *circumscription* arrow. *Conclusion* indicates the end of a research cycle or the termination of a specific DSR project.

Knowledge contribution resulting from new knowledge production is indicated in Figure 2.4 by the arrows labeled *circumscription* and *design science knowledge*. The *circumscription* process is especially important to understanding the DSR process because it generates *understanding that could only be gained from the specific act of construction*. Circumscription is a formal logical method (McCarthy 1980) that assumes that every fragment of knowledge is valid only in certain situations. Further, the applicability of knowledge can only be determined through the detection and analysis of contradictions—in common language, the design science researcher *learns or discovers* when things *do not* work "according to theory." This happens many times not due to a misunderstanding of the theory, but due to the necessarily incomplete nature of ANY knowledge base. The DSR process, when interrupted and forced back to *awareness of problem* in this way, contributes valuable *constraint knowledge* to the understanding of the always-incomplete-theories that abductively motivated the original research.

The creative cognitive processes of reflection and abstraction are used in the *conclusion* phase to make contributions of design science knowledge. At the conclusion of the research project, the overall contribution made by the research project to advance knowledge in the research area—preferably as a design theory (DT)—needs to be argued (see a later section on "Outputs of DSR").

Other DSR Process Models

There are a number of other excellent DSR process models—descriptions (and diagrams) of DSR process (cf. Peffers et al. 2008; Hevner et al. 2004; Purao 2002; Gregg et al. 2001; March and Smith 1995; Nunamaker et al. 1991). The model we

described above is similar to these models; its emphasis, however, is on a detailed process for generating design science knowledge.

The DSR methodology process model developed by Peffers et al. (2008) attempts to synthesize selected prior literature on the topic. This model, in comparison to the model shown in Figure 2.3, breaks the *awareness of problem* phase into two phases, *identify problem* and *motivate and define objectives of a solution*; merges the *suggestion and development* phases into a single phase, *design and development*; breaks the *evaluation* phase into two phases, *demonstration* and *evaluation*; and finally renames the *conclusion* phase as *communication*. A distinguishing feature of this model is identification of the fact that the research can get initiated from a variety of contexts—problem-centered initiation, objective-centered solution, design and development-centered initiation, client/context initiation—and start in a corresponding phase of the nominal process sequence shown.

Outputs of DSR

The output of a DSR project should be design science knowledge. To understand what form this knowledge contribution can take it is good to start with understanding the possible types of knowledge contribution of DSR.

Figure 2.5 shows a knowledge contribution framework for DSR (Gregor and Hevner 2013). In this framework, *invention* (inventing *new knowledge/solutions* for *new problems*), *improvement* (developing *new knowledge/solutions* for *known problems*), and *adaptation* (nontrivial or innovative adaptation of *known knowledge/solutions* for *new problems*) can all be types of knowledge contribution in DSR and a single research project can make more than one type of knowledge contribution.

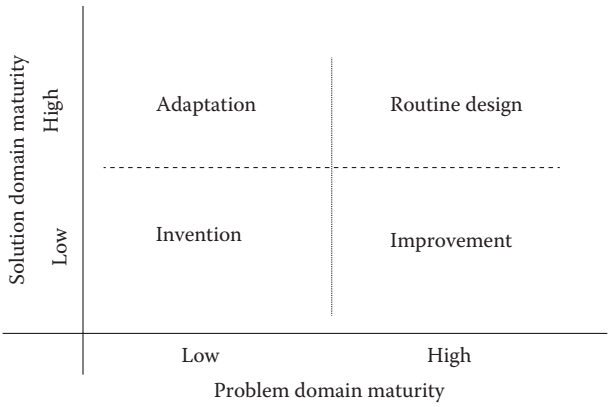


Figure 2.5 Design science research knowledge contribution framework. (Adapted from Gregor, S., and Hevner, A., *MIS Quarterly*, 37(2): 337–355, 2013.)

Routine design (applying *known knowledge/solutions* to *known problems*) by itself would seldom be considered as a research contribution. For knowledge contribution to be considered as a significant research contribution, it has to be judged as significant with respect to the current state of the knowledge in the research area (and be considered interesting).

Design science knowledge can be in the form of artifacts—constructs, models, frameworks, architectures, design principles, methods, and/or instantiations—and design theories (see Table 2.1). Instantiation is generally referred to as a *material* artifact while the other types of artifacts are referred to as *abstract* artifacts. A DT usually includes abstract artifacts and can also include instantiations. To get a better understanding of the different forms of knowledge contribution of DSR, we will now discuss the different forms of design science knowledge culminating in a detailed discussion of DT.

March and Smith (1995) in a widely cited paper, contrasting DSR with natural science research, propose four general outputs for DSR: *constructs*, *models*, *methods*, and *instantiations*. *Constructs* are the conceptual vocabulary of a problem/solution domain. Constructs arise during the conceptualization of the problem and are refined throughout the DSR cycle. Since a working design (artifact) consists of a large number of entities and their relationships, the construct set for a

Table 2.1 Potential Outputs of a Design Science Research Project

	Output	*Description*
1	Constructs	The conceptual vocabulary of a domain
2	Models	Sets of propositions or statements expressing relationships between constructs
3	Frameworks	Real or conceptual guides to serve as support or guide
4	Architectures	High-level structures of systems
5	Design principles	Core principles and concepts to guide design
6	Methods	Sets of steps used to perform tasks—how-to knowledge
7	Instantiations	Situated implementations in certain environments that do or do not operationalize constructs, models, methods, and other abstract artifacts; in the latter case such knowledge remains tacit
8	Design theories	A prescriptive set of statements on how to do something to achieve a certain objective. A theory usually includes other abstract artifacts such as constructs, models, frameworks, architectures, design principles, and methods

DSR experiment may be larger than the equivalent set for a descriptive (empirical) experiment.

A *model* is "a set of propositions or statements expressing relationships among constructs." March and Smith identify models with *problem and solution statements*. They are proposals for how things are or should be. Models differ from natural science theories primarily in intent: natural science has a traditional focus on truth whereas DSR focuses more on (situated) utility. Thus, a model is presented in terms of what it does and a theory described in terms of construct relationships. However, a theory can always be extrapolated to what can be done with the implicit knowledge and a set of entities and proposed relationships can always be expressed as a theoretical statement of how or why the output occurs.

A *method* is a set of steps (an algorithm or guideline) used to perform a task. "Methods are goal directed plans for manipulating constructs so that the solution statement model is realized" (March and Smith 1995). Implicit in a DSR method then is the problem and solution statement expressed in the construct vocabulary. In contrast to natural science research, a method may well be the object of the research program in DSR. Since the axiology of DSR (see the section on "Philosophical Grounding of DSR") stresses problem solving, a more effective way of accomplishing an end result—even or sometimes especially a familiar or previously achieved end result—is valued.

The final output from a DSR effort in March and Smith's explication is an *instantiation* that "operationalizes constructs, models and methods." It is the realization of the artifact in an environment. Emphasizing the proactive nature of DSR, they point out that an instantiation sometimes precedes a complete articulation of the conceptual vocabulary and the models (or theories) that it embodies. We emphasize this further by referring to the aeronautical engineering example given earlier in the section, "Can Design Be Research"; aircraft flew decades before a full understanding of how such flight was accomplished. And, it is unlikely the understanding would ever have occurred in the absence of the working artifacts. Thus, *situated implementation* may be a better phrase to capture the nature of this output.

Rossi and Sein (2003) and Purao (2002) set forth their own list of DSR outputs. All but one of these can be mapped directly to March and Smith's list. Their fifth output (*better theories—design theories*) is highly significant and merits inclusion in our general list of DSR outputs. We add to the list of outputs, additional abstract artifacts such as frameworks, architectures, and design principles.

Figure 2.6 presents a slightly different perspective on the outputs of DSR (Purao 2002; Gregg et al. 2001; Gregor and Hevner 2013). In this figure, the multiple outputs of DSR are classified by level of abstraction and generalization; outputs at higher levels are preferred since it reflects a more general advancement of knowledge in the area.

Explicitly the upper level of Figure 2.6 and implicitly the middle level are theories about the emergent properties of the embedded phenomenon. However, in any complex artifact, at either level of abstraction, multiple principles may be invoked

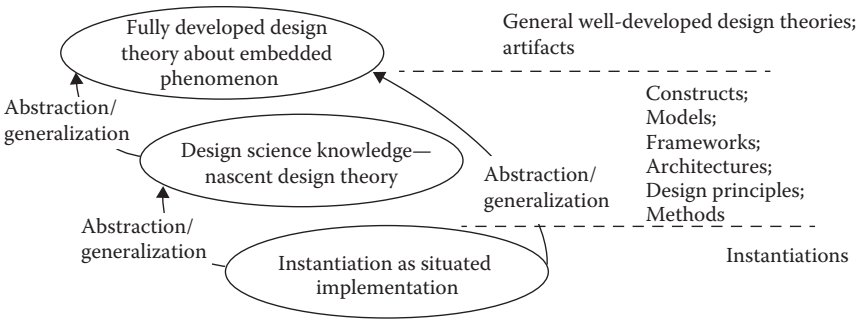


Figure 2.6 Design science knowledge hierarchy. (Adapted from Purao, S., GSU CIS Dept. Working Paper, 2002).

simultaneously to explain aspects of the artifacts behavior. In this sense, the behavior of the artifact in any single DSR project is over determined (Carroll and Kellogg 1989). This inevitable aspect of DSR has consequences discussed further in a later section on the "Philosophical Grounding of DSR."

Theory in DSR

It is helpful for understanding the different concepts of theory in DSR to first make concrete the traditional *natural sciences* conception of theory with an example. In high-school physics we learn that F = m*a. This, the second of Newton's laws of motion, began as a theory relating three well-defined constructs, force, mass, and acceleration, in a formal, mathematical way. Hypotheses could be derived from the theory of the form: if this theory is true, then if I apply force F to mass m, acceleration of that mass, a, should result. The theory is a formal statement from which formal implications of the type (if action A is performed on a system, then action B should result) can be derived.

In DSR, the phenomena of interest are *created* and so design theories have a different but analogous form to natural science theories. A DT is a set of prescriptive statements and outcome specification from which the implications can be drawn: If a system is constructed according to the (design) theoretical prescription, then that system will behave (or have outputs) as specified in the theory. In the following, we look at three conceptions of design science theory: fully developed DT, nascent DT, and design relevant explanatory/predictive theory (DREPT)—a type of theory that attempts to bridge between design and underlying natural phenomena.

A DT (fully developed or nascent) is a prescriptive type of theory, the fifth type of theory in Gregor's taxonomy of theories (Gregor 2006). Walls et al. (1992, 2004) provide one way of defining DT for IS and call it information systems design theory (ISDT). Gregor and Jones (2007) extend this work to provide a revised definition of ISDT. In the next section, the profile of a DT is discussed.

DT is the desired form of knowledge contribution from a DSR project. However, a well-developed and general DT in a research area may take years of effort by the research community in the area. It is thus more likely for a DSR project to contribute a nascent DT that is not so well developed; a nascent DT can be a preliminary contribution to a new DT or an incremental contribution to an existing broader DT in an area. Also, nascent DTs can vary in terms of their maturity and could be qualified with phrases such as "preliminary," "rudimentary," "reasonably developed," and so on.

Artifacts such as constructs, models, methods, and so on (see Table 2.1) are constituents of a DT but do not by themselves constitute a DT unless the other requirements of a DT (see Table 2.2), particularly those related to evaluation/validation and justificatory knowledge, are fulfilled. The knowledge contribution of DSR may be merely an instantiation with no or minimal contribution of abstract artifacts. This is possible in a situation where the knowledge contribution is of the invention type (see Figure 2.5). It is also possible that the knowledge contribution is an interesting partial or even an incomplete DT with potential for further work.

DT is about how to do something to meet a certain objective without fully answering why the prescribed actions should work. Design relevant explanatory/predictive theory—DREPT (Kuechler and Vaishnavi 2012)—is a new type of design-realm theory that augments the "how" part of a DT with explanatory information on "why" one should trust the design action to work. The explanatory information is provided using "kernel theory," established theory in natural, social, design, or mathematical sciences; the term "kernel theory" is used here with a broadened scope compared to its traditional use.

DSR can contribute to better theories (or theory building) in at least two distinct ways, both of which may be interpreted as analogous to experimental scientific investigation in the natural science sense. First, since the methodological construction of an artifact is an object of theorizing for many communities (e.g., how to build more maintainable software), the development phase of a DSR effort can be an experimental proof of method, an experimental exploration of method, or both.

Second, the artifact can expose relationships between its elements. It is tautological to say that an artifact functions as it does because the relationships between its elements enable certain behaviors and constrain others. However, if the relationships between artifact (or system) elements are less than fully understood and if the relationship is made more visible than previously during either the construction or evaluation phase of the artifact, then the understanding of the elements has been increased, potentially falsifying or elaborating on previously theorized relationships. (Theoretical relationships enter the DSR effort during the abductive reasoning phase of Figure 2.4.) For some types of research, artifact construction is highly valued precisely for its contribution to theory. Human-computer interface (HCI) researchers Carroll and Kellogg (1989) state that "... HCI artifacts themselves are perhaps the most effective medium for theory development in HCI." Walls et al.

Table 2.2 Profile of a Design Theory

Component	*Description*
Core components	
1. Purpose and scope	Provides a clear description of the purpose and scope of the new theory
2. Constructs	Describes all the existing or new entities or concepts relevant to the description of the theory
3. Knowledge of form and function	Includes the full description of models, frameworks, methods, and/or other abstract artifacts that form the body of the design science knowledge contribution
4. Abstraction and generalization	Is at such an abstract and general level that the artifacts resulting from the theory can change or be changed without affecting the theory
5. Evaluation and validation propositions	Has been evaluated for its truthfulness, i.e., assertions made based on the theory have been tested in an appropriate manner
6. Justificatory knowledge	Includes references to justificatory knowledge—tacit theory (informal experience-based insights and intuitions), kernel theory—that can provide a reasonable degree of justification of the theory
Additional components	
7. Principles of implementation	Describes the process for instantiating the theory
8. Expository instantiation	Includes an instantiation (possibly situated implementation) that can be used for exposition of the theory and/or for testing the theory

Source: Adapted from Gregor, S and Jones, D., *Journal of the Association for Information Systems (JAIS)*, 8(5): 312–335, 2007.

(1992) elaborate the theory building potential of design and construction in the specific context of IS.

Profile of a DT

Table 2.2 summarizes the profile of a DT, particularly one for the ICT field. Its structure is adapted from the structure of an IS DT provided by Gregor and Jones (2007), which itself is an extension of the structure provided by

Walls et al. (1992; 2004). Following are some explanatory comments about the DT profile components.

- **Purpose and Scope:** A DT like any theory should be new, interesting, and true. There should be enough information in this component of the theory to argue that the theory is new and interesting (to the relevant research and possibly practice community).
- **Constructs:** All the existing and new concepts and entities that are needed to fully understand the theory should be fully described.
- **Knowledge of Form and Function:** These form the body of the theory—the design science knowledge contribution—and thus should be described and explained in detail.
- **Abstraction and Generalization:** Generality and abstractness are the hallmarks of theory. A theory should cover a variety of ways the theory will get instantiated or changed, or even allow evolution, adaptation, or learning of the resulting artifacts without affecting the theory. In other words, a DT should have a degree of permanence and range of coverage so that one does not have to create a new version of the theory for each new situation.
- **Evaluation and Validation Propositions:** A theory, in addition to being novel and interesting, should be true. Thus, sufficient effort should be invested in evaluating and validating the theory propositions. The method of evaluation and validation can vary and can range from logical arguments to experimentation or mathematical proof.
- **Justificatory Knowledge:** In addition to trusting a theory based on its evaluation and validation, the researcher needs to provide some insights into why one should believe that the theory is likely to be true.

The other two components, principles of implementation and expository instantiation, may or may not be needed for a DT depending upon the nature of the theory and the state of the art in the area of research.

In Part III of this book, we use Table 2.2 for assessing the status of DT contributed by the discussed examples of DSR in ICT.

A Framework for Theory Development in DSR

Figure 2.7 shows a framework for theory development that extends the framework proposed by Kuechler and Vaishnavi (2012) for IS theory development (see Chapter 4, Figure 4.1), to one for DSR, particularly for DSR in ICT. The figure shows three paths for the development of artifacts with theory development ramifications. Arrow 1 represents the development of artifacts without any explicit development of theory. Arrow 2 indicates the use of existing justificatory knowledge (Gregor and Jones 2007) in the development of a DT and its instantiation into an artifact or the creation of an artifact with further refinement and development of

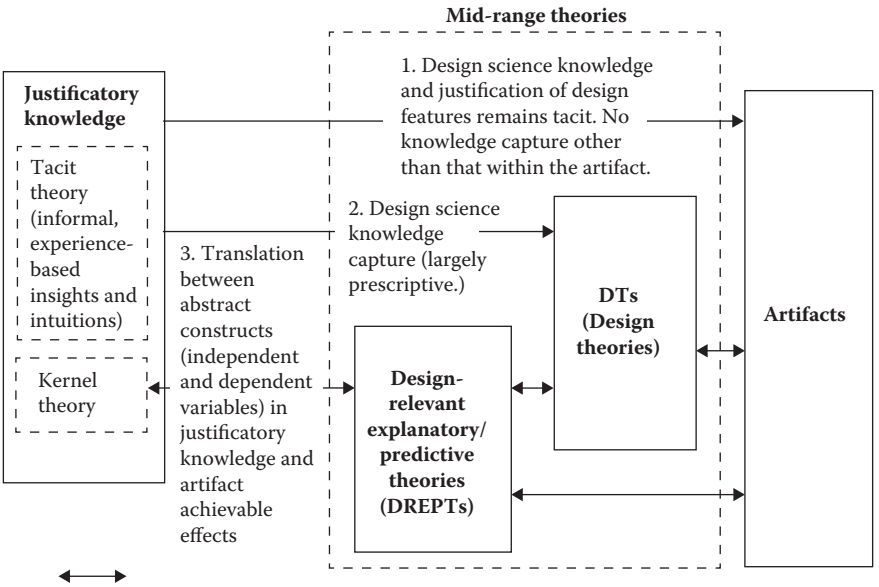


Figure 2.7 Framework for theory development in design science research.

DT from the artifacts using reflection and abstraction. Arrow 3 illustrates a strategy in which any relevant kernel theory (in terms of independent and dependent variables) from natural science, social science, design science, or mathematics is translated to artifact achievable effects in a design-relevant explanatory/predictive theory (DREPT), which after its evaluation through a created artifact can in turn lead to refinement and enrichment of the kernel theory. This path also provides a vehicle for not only showing how to design an artifact but also for understanding why the artifact should work. Chapter 5 (Kuechler and Vaishnavi 2008) provides a detailed example for the development of a DREPT.

Theory Development in DSR: A Brief Literature Review

An example of the rapid evolution of DSR is the recent attention directed to theory. One of the seminal DSR papers in IS, Nunamaker et al. (1991), alludes to theory and refinement of theory as an output from what they term the "engineering model" of IS research. Shortly thereafter, Walls et al. (1992) presented a conception of ISDT, a prescriptive encoding of design science knowledge

abstracted from a DSR-IS project; a number of widely cited IS papers have subsequently made use of ISDT, for example, Kasper (1996) and Markus et al. (2002). However, two influential papers subsequent to Walls et al. (1992), March and Smith (1995), and Hevner et al. (2004) do not explicitly mention theory and this has been interpreted by some in the field as suggesting that theory is *not* an output to be sought from DSR-IS. Yet more recent papers including Gregor (2006), Gregor and Jones (2007), Kuechler and Vaishnavi (2008), Arazy et al. (2010), Kuechler and Vaishnavi (2012), and Gregor and Hevner (2013) explicitly mention theory as a DSR project output and present methods for developing such theory during the course of DSR. Gregor (2006) provides a taxonomy of IS theory and proposes "Theory for Design and Action" as a type of IS theory (Type V Theory). Gregor and Jones (2007) build upon the work of Walls et al. (1992; 2004) and revise the structure for ISDT. Kuechler and Vaishnavi (2012) put forward a framework for theory development in DSR in the context of IS; see Figure 2.7. Niehaves et al. (2012) devised a framework for adding rigor to the translation between a DT and the corresponding design artifact. Gregor and Hevner (2013) stress contributions to knowledge as the expected output, which could be "partial theory, incomplete theory, or even some particularly interesting and perhaps surprising empirical generalization in the form of a new design artifact." Appendix 2A at the end of the chapter contains a section on selected recent papers concerning theory in DSR.

General Guidance on Expected Outputs from DSR

The general goal of DSR is to create or contribute to new and interesting design science knowledge in an area of interest—"a body of intellectually tough, analytic, partly formalizable, partly empirical teachable doctrine about the design process" (Simon 1996). The desired form of such knowledge is a DT along with artifacts such as constructs, models, methods, and instantiations (Table 2.1), and other abstract artifacts. The creation of a fully developed theory, however, cannot be expected from a single DSR project. It usually gets created as a community effort through multiple iterations of research, development, and practice, and many times includes active participation of the industry (see Chapter 3).

The creation of design science knowledge in an area usually begins as an invention type of knowledge contribution (see Figure 2.5) and is at the lowest level of abstraction/generalization according to Figure 2.6. This type of output is a situated implementation with possibly some work at the middle level of the abstraction/generalization framework shown in Figure 2.6, which may lead to the development of a nascent DT. It is very likely to have followed Path 1 or possibly Path 2 in the theory development framework shown in Figure 2.7. It is accepted as a design science knowledge contribution for the novelty and significance of the contribution from both problem definition and solution/knowledge development standpoints, and gets published or gets patented. Chen's work on the ER model (Chen 1976)

or the work of Agrawal et al. (1993) on data mining are examples of such research contributions that have spawned entire fields of research.

After the initial breakthrough type of research, DSR contributions in the area need to be improvement and/or adaptation types of knowledge contributions according to the knowledge contribution framework shown in Figure 2.5 and need to make progress on the level of abstraction/generalization of the research outputs (according to Figure 2.6). For such research, the creation of a general well-developed DT would be a goal but depending upon the state of knowledge in the area, a nascent DT can be an acceptable form of output as long as it is deemed to make significant contribution to the state of art in the research area. The research could follow Path 2 or Path 3 of the theory development framework (Figure 2.7). For improvement type of knowledge contribution, the research needs to produce a better solution according to some acceptable metric and for the adaptation type of knowledge contribution, the research needs to show the challenges and the non-trivial nature of adaptation of existing knowledge for a new problem or for a new version of an existing problem that usually manifests itself because of technology changes. In either case, the research needs to be deemed as making a significant and novel contribution and the outputs need to be at as high a level of abstraction and generalization as is possible.

In summary, to understand the expected outputs of a DSR project one needs to first assess the type of knowledge contribution being made with respect to the existing knowledge (see Figure 2.5). If the knowledge contribution can be argued to be significant but of the invention type, then it can even be at the lowest level of abstraction and generalization. If, on the other hand, the research does not make an original invention type of contribution but instead makes an improvement type of contribution and/or makes a novel use of existing knowledge in a new area (adaptation), then the research outputs need to be at higher levels of abstraction/generalization according to Figure 2.6. They should include at least a nascent DT, and one needs to argue how they are advancing the state of knowledge in the area.

Example of Community-Determined Outputs

Precisely what is obtained from a DSR effort is determined by (1) the phase of research on which reflection and analysis focuses (from Figure 2.3) and (2) the level of abstraction to which reflection and analysis generalize the knowledge contribution (see Figure 2.6). These factors in turn are strongly influenced by the community performing the research.

To illustrate the different outputs that are commonly seen as the desired result for DSR, consider the *same* artifact development as carried out by different ICT research sub-communities: database, software engineering, HCI, decision sciences, and IS cognitive researchers (IS cognitive research exchange—IS CORE): the construction of a data visualization interface for complex queries against large

relational databases. For all of the communities, the research is motivated by common *problem awareness*: A better interface can be developed that will allow users to more quickly and effectively obtain answers to questions about the performance of their business operations.

The theoretical impetus for the prospective improvement would vary between research communities. For the software engineering or database communities, the motivation could be new knowledge of faster access techniques or visual rendering techniques. For the decision sciences community and the HCI and cognitive research communities, the impetus could be new research in reference disciplines on visual impacts on cognition and/or on decision-making. The resulting artifact would be quite similar for all communities, as would the construction mechanics—the computer languages used in development, the deployment platforms, and so on. However, the stages of development on which observation and reflection is centered and the measures used to evaluate the resultant artifact (cf. Figure 2.3) would be considerably different for each community. Table 2.3 lists the communities that might construct a data visualization artifact, the primary perspective with which they would view the artifact and the different knowledge that would emerge from the research effort as a result of the differing perspectives.

Some explications of DSR in IS have stated that the primary focus is always on the finished artifact and how well it works rather than its component interactions, that is, *why* it works (Hevner et al. 2004), but more recent work (e.g., Gregor and Hevner 2013) and our example (case study) in a later section present a broader view. The apparent contradiction may simply be in how wide the net of *ICT Research* is cast and the selection of sub-communities it is considered to contain.

Table 2.3 Design Science Research Perspectives and Outputs by Community

Community	*Perspective*	*Knowledge Derived*
Human–computer interface; information systems cognitive research exchange science	Artifact as experimental apparatus	What database visualization interfaces reveal about the cognition of complex data relationships
Database; decision science software engineering	Artifact as focused design principle exploration	Principles for the construction of data visualization interfaces
Database; software engineering	Artifact as improved instance of a tool	A better data visualization interface for relational, business-oriented databases

Philosophical Grounding of DSR

Ontology is the study that describes the nature of reality, for example, what is real and what is not, what is fundamental and what is derivative.

Epistemology is the study that explores the nature of knowledge, for example, on what does knowledge depend and how can we be certain of what we know.

Axiology is the study of values: What values does an individual or group hold and why?

The definitions of these terms are worth reviewing because although assumptions about reality, knowledge, and value underlie any intellectual endeavor, they are *implicit* most of the time for most people, including researchers. Indeed, as historians and philosophers of science have noted, in "tightly" paradigmatic communities, people may conduct research for an entire career without considering the philosophical implications of their passively received areas of interest and research methods (Kuhn 1996—first published in 1962). It is typically only in multi-paradigmatic or pre-paradigmatic communities—such as IS—that researchers are forced to consider the most fundamental bases of the socially constructed realities (Berger and Luckman 1966; Searle 1995) in which they operate.

The contrasting ontological and epistemological assumptions implicit in natural science and social science research approaches have been authoritatively explicated in a number of widely cited works (Bunge 1984; Guba and Lincoln 1994). Gregg et al. (2001) add the meta-level assumptions of DSR (which they term the socio-technologist/developmentalist approach) to earlier work contrasting positivist and interpretive approaches to research. We have drawn from Gregg et al. in compiling Table 2.4 that summarizes the philosophical assumptions of those three "ways of knowing" and have added several insights from our combined 40+ years of DSR experience. Our first addition is the stress on *iterative circumscription* (cf. Figure 2.3) and how this essential part of the DSR methodology iteratively determines (or reveals) the reality and the knowledge that emerge from the research effort. The second addition to Table 2.4 is the row labeled axiology—the study of values. We believe it is the shared valuing of what researchers hope to find in the pursuit of their efforts that binds them into a community. Certainly the self and community valuation of their efforts and findings is a highly significant motivator for any researcher, and we were surprised to find how little stress this topic has received in the literature, especially given the significant differences in what each community values.

The metaphysical assumptions of DSR are unique. First, none of the ontology, epistemology, or axiology of the paradigm is derivable from any other. Second, ontological and epistemological viewpoints shift in DSR as the project runs through circumscription cycles depicted in Figure 2.3. This iteration is similar to but more radical than the hermeneutic processes used in some interpretive research.

Table 2.4 Philosophical Assumption of the Three Research Perspectives

	Research Perspective		
Basic Belief	*Positivist*	*Interpretive*	*Design*
Ontology	A single reality. Knowable, probabilistic	Multiple realities, socially constructed	Multiple, contextually situated alternative world-states. Socio-technologically enabled
Epistemology	Objective; dispassionate. Detached observer of truth	Subjective, i.e., values and knowledge emerge from the researcher-participant interaction	*Knowing through making*: objectively constrained construction within a context. Iterative circumscription reveals meaning
Methodology	Observation; quantitative, statistical	Participation; qualitative. Hermeneutical, dialectical	Developmental. Measure artifactual impacts on the composite system
Axiology: what is of value	Truth: universal and beautiful; prediction	Understanding: situated and description	Control; creation; problem-solving; progress (i.e., improvement); understanding

DSR by definition changes the state-of-the-world through the introduction of novel artifacts. Thus, design science researchers are comfortable with alternative world-states. The obvious contrast is with positivist ontology where a single, given composite socio-technical system is the typical unit of analysis; even the problem statement is subject to revision as a DSR effort proceeds. However, the multiple world-states of the design science researcher are not the same as the multiple realities of the interpretive researcher: many if not most design science researchers believe in a single, stable underlying physical reality that constrains the multiplicity of world-states. The abduction phase of DSR (Figure 2.4) in which physical laws are tentatively composed into a configuration that will produce an artifact with the intended problem-solving functionality virtually demands a natural-science-like belief in a single, fixed grounding reality.

Epistemologically, the design science researcher knows that a piece of information is factual and knows further what that information means through the process of development/circumscription. An artifact is developed. Its behavior is the result of interactions between components. Descriptions of the interactions are

information and to the degree the artifact behaves predictably the information is true. Its meaning is precisely the functionality it enables in the composite system (artifact and user). What it means is what it does. The design science researcher is thus a pragmatist (Peirce 1931). Venable (2006) has proposed letting utility theory be an appropriate form of a DT resulting from DSR, which makes utilitarian claims related, for example, to efficacy, effectiveness, efficiency, elegance, and ethicality (Checkland and Scholes 1999) for the created artifacts(s).There is also a flavor of instrumentalism (Hendry 2004) in DSR. The dependence on a predictably functioning artifact (instrument) gives DSR an epistemology that resembles that of natural-science research more closely than that of either positivist or interpretive research.

Axiologically, the design science researcher values creative manipulation and control of the environment in addition to (if not over) more traditional research values such as the pursuit of truth or understanding. Certainly the design science researcher must have a far higher tolerance for ambiguity than is generally acceptable in the positivist research stance. As many authors have pointed out, the end result of a DSR effort may be very poorly understood and still be considered a success by the community (Hevner et al. 2004). A practical or functional addition to an area body of knowledge, even as partial theory or incomplete theory (Gregor and Hevner 2013), codified and transmitted to the community where it can provide the basis for further exploration, may be all that is required of a successful project. Indeed, it is precisely in the exploration of "wicked problems" for which conflicting or sparse theoretical bases exist that DSR excels (March and Smith 1995; Carroll and Kellogg 1989).

Finally, the philosophical perspective of the design science researcher changes as progress is iteratively made through the phases of Figure 2.4. In some sense, it is as if the design science researcher creates a reality through constructive intervention, then reflectively becomes a positivist observer, recording the behavior of the system and comparing it to the predictions (theory) set out during the abductive phase. The observations are interpreted, become the basis for new theorizing, and a new abductive, interventionist cycle begins. In this sense, DSR is very similar to the action research methodology of the interpretive paradigm; however, the time frame of DSR construction is enormously foreshortened relative to the social group interactions typical of action research.

Bunge (1984) implies that DSR is most effective when its practitioners shift between pragmatic and critical realist perspectives, guided by a pragmatic assessment of progress in the DSR cycle. Purao (2002) presents a very rich elaboration on the perspective shifts that accompany any iterative DSR cycle. His analysis is grounded in semiotics and describes in detail how "the design science researcher arrives at an interpretation (understanding) of the phenomenon and the design of the artifact simultaneously."

An Example of ICT DSR

The example (case study) we have chosen to add detail and concreteness to the discussion of DSR philosophy and method in ICT is one from our joint experience. We make only two claims for this research: (1) it is a reasonable example as it comfortably encompasses all the points of the preceding discussion and (2) since it is our research we are privy to and able to present a multitude of details that are rarely written up and available in journal publications. We describe the research, from conception to the first publication drawn from it, in phases corresponding to those in Figure 2.3.

Smart Objects: A DSR Project

Awareness of Problem

In the mid-1980s, one of the senior project participants, Vijay, began actively seeking to extend his research from designing efficient data and file structures (a primarily computer science topic) to software engineering (an area with a significant IS component). In the course of a discussion with one of his colleagues at Georgia State University (GSU), he became aware of a situation that showed research promise: development of a computerized decision support system for nuclear reactors. Three Mile Island had brought national awareness to the problems associated with safe operation of a nuclear power plant, rule-based decision support systems were a current area of general IS interest, and the director of the research reactor at Georgia Tech was interested in developing a system to support its operations.

A doctoral student (Gary) was brought into the project to begin a preliminary support system development in the rule-based language, Prolog. Within a few weeks, it became apparent that a system to support the several thousand procedures found in a typical commercial power plant would be nearly impossible to develop in Prolog; and if developed, would be literally impossible to maintain. The higher-level expert system development packages available at the time (and currently) were more capable but still obviously inadequate. The difficulty of constructing and maintaining large expert systems was widely known at the time; however, the Prolog pilot project gave the research group significant insights they would not otherwise have had into the root causes of the problem: continuously changing requirements and the complexity inherent in several thousand rule-based interlocking procedures. Out of detailed analysis of the failed pilot system emerged the first *awareness of the problem* on which the research would focus: how to construct and continuously maintain a support system for the operation of a complex, hierarchical, procedure-driven environment.

Suggestion

There are many approaches to the problems of software system complexity and the research group discussed them over a period of months. Some of the alternatives that were discarded were the development of a new software development methodology specifically focused on operations support systems, automation of the maintenance function, and development of a high-level programming environment. New insights into the problem continued to emerge even as (and precisely because) potential solutions to the problem were considered. One key insight was that the system complexity resided primarily in control of the system, that is, although the individual procedures could be modeled straightforwardly, the procedure that should take precedence (control) over the others and where the results of that procedure should be routed depended in a highly complex fashion on past and present states of multiple procedures. Essential to the development of the system was the effective modeling of this complex control structure.

By this point Gary had decided to adopt the problem as his dissertation topic and under Vijay's direction began extensive research into various mechanisms for modeling (describing in a precise, formal way) control. As the realization grew that they were in effect seeking to describe the *semantics* of the system, his reading began to focus especially on some of the techniques to emerge from the area of semantic modeling.

During the alternating cycles of discussion, reading, and individual cogitation that characterize many DSR efforts, several software engineering concepts were brought together with a final key insight to yield the ultimately successful direction for the development. During one discussion, Vijay realized that the control information for the system was knowledge, identical in form to the domain knowledge in the procedures and could be modeled with rules, in the same way. However, since the execution of the individual procedures was independent of the control knowledge, the two types of rules could execute in different cycles, partitioning and greatly reducing the complexity of the overall system. Finally, the then relatively new concept of object orientation seemed to be the ideal approach to partitioning the total system knowledge into individual procedures. And if each "smart" object were further partitioned into a domain knowledge component and a control knowledge component, and the rules were stated in a high level English like syntax that was both executable and readable by domain experts...

Awareness of Problem Revisited

As noted in the general discussion of the DSR method, any of its phases may be spontaneously revisited from any of the other phases. Especially in the early stages of a project, this results in a conceptual fluidity that can be disconcerting to practitioners of less dynamic paradigms. Though it is difficult in retrospect to pinpoint exactly where in the process the change occurred, by the inception of the

development phase the problem statement had changed to a sub-goal implicit in the original problem statement: *how to effectively model operations support systems for complex, hierarchical, procedure-driven environments* with *control modeling* as the specific research problem. (This sort of "drilling down" into the problem or re-scoping the research at a more basic level occurs frequently in all research, but is effectively part of the method in DSR.)

Development

Although the development of a DSR artifact can be straightforward, that was not the case for smart objects. The construction was completely conceptual and involved the "discovery" through multiple thought and paper trials of the details of the novel entity that had been conceptualized at a high level in the suggestion phase, the "smart object."

For example, what (exactly) would the syntax be for the two types of rules, domain, and control? How (exactly) should the two rule evaluation cycles for each type of knowledge interleave? Should the two types of knowledge be permitted to interact? If so, how? Should control rules have the ability to "write" or "rescind" domain rules, a la Lisp? Or, vice versa?

In a conceptual development such as this, the suggestion and construction phases blur because a successful design decision *is* an output product. The final deliverable (from this initial development) was a conceptual model consisting of: (1) a set of meta-level rules for implementing domain knowledge and control knowledge separately, but within a single structure, the "smart object," and (2) another set of meta-rules that described how the domain and control knowledge, once "modeled" as smart objects, would be interpreted (a virtual machine for executing the smart objects).

Evaluation

In a sense, evaluation takes place continuously in a design process (research or otherwise) since a large number of "micro-evaluations" take place at every design detail decision. Each decision is followed by a "thought experiment" in which that part of the design is mentally exercised by the designer. However, for the remainder of this section we will describe the "formal" evaluation that occurred after the design had stabilized.

In order to test the conceptual design, various operating environments were modeled and "hand-stepped" through the execution rules to determine that logically correct system behavior occurred at appropriate times in the simulation. The simulation that appeared in Gary's dissertation, the first publication to result from the research, was a grocery bagging "robot." This example had been popularized in a best-selling artificial intelligence textbook of the time and had the advantage of

being a familiar logic test bed to many external evaluators of the artifact. Exponents of other IS research paradigms may find the evaluation criteria simplistic, and wonder why, for example, modeling of the nuclear power plant operating environment was not the obvious choice. The answer is resources; the modeling and hand testing of even the grocery-bagging example occupied several man-months. During the evaluation, minor redesign of the artifact (the smart object conceptual model) occurred on several occasions, which is a common occurrence in DSR. By the end of the evaluation phase, the smart object model had successfully completed the simulation of numerous bagging exercises that included complex control situations and was adjudged a success by the design team.

Conclusion

The finale for the first research effort involving smart objects was the codification of the problem development, design basis in prior work, the design itself, and the results of the evaluation effort in Gary's dissertation (Buchanan 1991). The successful defense of the dissertation at GSU required careful consideration and judgment of the artifact and its performance by a committee made up primarily of other design science researchers. The core concepts were considered to have substantial merit, and Gary and Vijay produced several conference papers based on smart objects.

Epilogue

After Gary's graduation, Vijay and Gary collaborated on a paper based on the research project and submitted it to *IEEE Transactions on Knowledge and Data Engineering (TKDE)*. The paper was returned for substantial revisions. At this point, Gary's interest in the project waned, however a recently admitted GSU CIS doctoral student (Bill) found the concepts interesting enough to enter into the research group and continue the development effort. After 4 years, three conference papers on smart objects and related topics, and three major revisions, the TKDE paper was finally published as "A Data/Knowledge Paradigm for the Modeling and Design of Operations Support Systems" (Vaishnavi et al. 1997). By the time of acceptance, smart objects had been through several additional DSR cycles, each focusing on the refinement of a different aspect of the original design, or a critical support function for its use-in-practice such as the methodology developed for partitioning workflow IS into smart objects.

References

Agrawal, R. et al. (1993). "Mining Association Rules between Sets of Items in Large Databases," *Proceedings of the 1993 ACM SIGMOD Conference*, Washington, DC, 207–216.

Alexander, C. (1964). *Notes on the Synthesis of Form*. Cambridge, MA: Harvard University Press.

Arazy, O., Kumar, N., and Shapira, B. (2010). "A Theory-Driven Design Framework for Social Recommender Systems." *Journal of the Association for Information Systems (JAIS)* 11(9): 455–490.

Berger, P. and Luckman, T. (1966). *The Social Construction of Reality: A Treatise in the Sociology of Knowledge*. Garden City, NY: Doubleday.

Buchanan, G. (1991). *Modeling Operations Management Support Systems*. Unpublished Doctoral Dissertation, Atlanta, GA. College of Business Administration, Georgia State University.

Bunge, M. (1984). "Philosophical Inputs and Outputs of Technology." In *History and Philosophy of Technology*, Bugliarello, G. and Donner, D. (Eds.), Urbana, IL: University of Illinois Press, 263–281.

Carroll, J. and Kellogg, W. (1989). "Artifact as Theory Nexus: Hermeneutics Meets Theory-Based Design." In *Proceedings of CHI '89*. ACM Press.

Checkland, P. and Scholes, J. (1999). *Soft Systems Methodology in Action: A 30-Year Retrospective*. Chichester, UK: John Wiley & Sons.

Chen, P. (1976). "The Entity-Relationship Model: Toward a Unified View of Data." *ACM Transactions on Database Systems* 1(1): 9–37.

Gregg, D., Kulkarni, U., and Vinze, A. (2001). "Understanding the Philosophical Underpinnings of Software Engineering Research in Information Systems." *Information Systems Frontiers* 3(2): 169–183.

Gregor, S. (2006). "The Nature of Theory in Information Systems." *MISQ* 30(3): 611–642.

Gregor, S. and Hevner, A. (2013). "Positioning and Presenting Design Science Research for Maximum Impact." *MIS Quarterly* 37(2): 337–355.

Gregor, S. and Jones, D. (2007). "The Anatomy of a Design Theory." *Journal of the Association for Information Systems (JAIS)* 8(5): Article 19.

Guba, E. and Lincoln, Y. (1994). "Competing Paradigms in Qualitative Research." In *The Handbook of Qualitative Research*, N. Denzin and Y. Lincoln (Eds.): 105–117.

Hendry, R. (2004). "Are Realism and Instrumentalism Methodologically Different?" *Working paper*. Department of Philosophy, University of Durham, UK. Author e-mail: r.f.hendry@dur.ac.uk

Hevner, A., March, S., Park, J., and Ram, S. (2004). "Design Science in Information Systems Research." *MIS Quarterly* 28(1): 75–105.

Kuechler, W. and Vaishnavi, V. (2008). "On Theory Development in Design Science Research: Anatomy of a Research Project." *European Journal of Information Systems* 17(5): 1–23.

Kuechler, W. and Vaishnavi, V. (2008). "The Emergence of Design Research in Information Systems in North America." *Journal of Design Research* 7(1): 1–16.

Kuechler, W. and Vaishnavi, V. (2012). "A Framework for Theory Development in Design Science Research: Multiple Perspectives." *Journal of the Association for Information Systems (JAIS)* 13(6): 395–423.

Kuhn, T. (1962/96). *The Structure of Scientific Revolutions*. Chicago, IL: University of Chicago Press.

Lakatos, I. (1978). *The Methodology of Scientific Research Programmes*, J. Worral and G. Currie (Eds.), Cambridge, UK: Cambridge University Press.

March, S. and Smith, G. (1995). "Design and Natural Science Research on Information Technology." *Decision Support Systems* 15(4): 251–266.

Markus, M., Majchrzak, A., and Gasser, L. (2002). "A Design Theory for Systems that Support Emergent Knowledge Processes." *MIS Quarterly* 26(3): 179–212.

Maturana, H. and Varela, F. (1987). *The Tree of Knowledge: The Biological Roots of Human Understanding*. Boston, MA: New Science Library.

McCarthy, J. (1980). "Circumscription—A Form of Non-Monotonic Reasoning." *Artificial Intelligence* 13(1–2): 27–39.

Newell, A. (1973). "Production Systems: Models of Control Structures." In *Visual information Processing*, W.G. Chase (Ed.), New York: Academic Press, 463–526.

Newell, A. (1990). *Unified Theories of Cognition*. Cambridge, MS: Harvard University Press.

Niehaves, B., Ortbach, K., and Tavakoli, A. (2012). "On the Relationship between the IT Artifact and Design Theory: The Case of Virtual Social Facilitation," In *DESRIST 2012, LNCS 7286*, K. Peffers, M. Rothenberger, and B. Kuechler (Eds.), Springer-Verlag, Berlin, Heidelberg, 354–370.

Nunamaker, J., Chen, M., and Purdin, T. (1991). "System Development in Information Systems Research." *Journal of Management Information Systems* 7(3): 89–106.

Owen, C. (1997). "Understanding Design Research. Toward an Achievement of Balance." *Journal of the Japanese Society for the Science of Design* 5(2): 36–45.

Peffers, K., Tuunanen, T., Rothenberger, M., and Chatterjee, S. (2008). "A Design Science Research Methodology for Information Systems Research." *Journal of Management Information Systems* 24(3): 45–77.

Peirce, C.S. (1931–1935). *Collected Papers of Charles Sanders Peirce*, Harshorne, C. and Weiss, P. Eds. Vols. 1–6, Cambridge, MA: Harvard University Press.

Purao, S. (2002). "Design Research in the Technology of Information Systems: Truth or Dare." *Working Paper. GSU Department of CIS*. Atlanta, GA.

Rossi, M. and Sein, M. (2003). "Design Research Workshop: A Proactive Research Approach." Presentation delivered at IRIS 26, August 9–12, 2003.

Searle, J., Ed. (1995). *The Construction of Social Reality*. New York, NY: The Free Press.

Simon, H. (1996). *The Sciences of the Artificial*, Third Edition. Cambridge, MA: MIT Press.

Takeda, H., Veerkamp, P., Tomiyama, T., and Yoshikawam, H. (1990). "Modeling Design Processes." *AI Magazine* Winter: 37–48.

Vaishnavi, V. et al. (1997). "A Data/Knowledge Paradigm for the Modeling and Design of Operations Support Systems." *IEEE Transactions on Knowledge and Data Engineering* 9(2): 275–291.

Vaishnavi, V. and Kuechler, W. (2004/13). "Design Science Research in Information Systems" January 20, 2004; last updated October 23, 2013. URL: *http://www.desrist.org/design-research-in-information-systems/* (last accessed on January 28, 2015). [Pertti Jarvinen's critique of the 2004 version: http://www.desrist.org/desrist/content/Jarvinen/VaishnaviKuechlerRev.pdf (last accessed on January 28, 2015)]

Varela, F. (1988). "Structural Coupling and the Origin of Meaning in a Simple Cellular Automata." In *The Semiotics of Cellular Communication in the Immune System*, E. Scaraz, F. Celada, N. Michenson, and T. Tada (Eds.), New York, NY: Springer-Verlag.

Venable, J. (2006). "The Role of Theory and Theorising in Design Science Research." In *Proceedings of DESRIST 2006*, Claremont, CA.

Walls, J., Widmeyer, G., and El Sawy, O. (1992). "Building an Information System Design Theory for Vigilant EIS." *Information Systems Research* 3(1): 36–59.

Walls, J., Widmeyer, G., and El Sawy, O. (2004). "Assessing Information System Design Theory in Perspective: How Useful was our 1992 Initial Rendition." *Journal of Information Technology Theory and Application* 6(2): 43–58.

Wilson, J.R. (2002). "Responsible Authorship and Peer Review." *Science and Engineering Ethics* 8(2): 155–174.

Appendix 2A A Design Science Research Bibliography

General References on Design Science Research

Alexander, C. (1964). *Notes on the Synthesis of Form*. Cambridge, MA: Harvard University Press.

Carroll, J. and Kellogg, W. (1989). "Artifact as Theory Nexus: Hermeneutics Meets Theory-Based Design." In *Proceedings of CHI '89*. ACM Press.

Checkland, P. and Scholes, J. (1999). *Soft Systems Methodology in Action: A 30-Year Retrospective*. Chichester, UK: John Wiley & Sons.

Dasgupta, S. (1996). *Technology and Creativity*. New York, NY: Oxford University Press.

Gregg, D., Kulkarni, U., and Vinze, A. (2001). "Understanding the Philosophical Underpinnings of Software Engineering Research in Information Systems." *Information Systems Frontiers* 3(2): 169–183.

Gregor, S. and Hevner, A. (2013). "Positioning and Presenting Design Science Research for Maximum Impact." *MIS Quarterly* 37(2): 337–355.

Hevner, A., March, S., Park, J., and Ram, S. (2004). "Design Science in Information Systems Research." *MIS Quarterly* 28(1): 75–105.

Hevner, A. and March, S. (2003). "The Information Systems Research Cycle." *IT Systems Perspective* 36(11): 111–113.

Kuhn, T. (1962/96). *The Structure of Scientific Revolutions*. Chicago, IL: University of Chicago Press.

Lakatos, I. (1978). *The Methodology of Scientific Research Programmes*, J. Worral and G. Currie (Eds.), Cambridge, UK: Cambridge University Press.

March, S. and Smith, G. (1995). "Design and Natural Science Research on Information Technology." *Decision Support Systems* 15(4): 251–266.

Maturana, H. and Varela, F. (1987). *The Tree of Knowledge: The Biological Roots of Human Understanding*. Boston, New Science Library.

McCarthy, J. (1980). "Circumscription—A Form of Non-Monotonic Reasoning." *Artificial Intelligence* 13(1–2): 27–39.

McKay, J. and Marshall, P. (2005). "A Review of Design Science in Information Systems." In *Proceedings Australian Conference on Information Systems*, Sydney.

Owen, C. (1997). "Understanding Design Research. Toward an Achievement of Balance." *Journal of the Japanese Society for the Science of Design* 5(2): 36–45.

Orlikowski, W. and Iacono, C. (2001). "Desperately Seeking the "IT" in IT Research—A Call to Theorizing the IT Artifact." *Information Systems Research* 12(2): 121–134.

Peffers, K., Tuunanen, T., Rothenberger, M., and Chatterjee, S. (2008). "A Design Science Research Methodology for Information Systems Research." *Journal of Management Information Systems* 24(3): 45–77.

Peirce, C.S. *Collected Papers of Charles Sanders Peirce*, Vols. 1–6, Harshorne, C. and Weiss, P. (Eds.), Cambridge, MA: Harvard University Press (1931–1935).

Purao, S. (2002). "Design Research in the Technology of Information Systems: Truth or Dare." *Working Paper*. *GSU Department of CIS*. Atlanta, GA.

Rossi, M. and Sein, M. (2003). "Design Research Workshop: A Proactive Research Approach." Presentation delivered at IRIS 26, August 9–12, 2003.

Simon, H. (1996). *The Sciences of the Artificial*, Third Edition. Cambridge, MA: MIT Press.

Takeda, H., Veerkamp, P., Tomiyama, T., and Yoshikawam, H. (1990). "Modeling Design Processes." *AI Magazine* Winter: 37–48.

Varela, F. (1988). "Structural Coupling and the Origin of Meaning in a Simple Cellular Automata." In *The Semiotics of Cellular Communication in the Immune System*, E. Scaraz, F. Celada, N. Michenson, and T. Tada (Eds.), New York, NY: Springer-Verlag.

Walls, J., Widmeyer, G., and El Sawy, O. (1992). "Building an Information System Design Theory for Vigilant EIS." *Information Systems Research* 3(1): 36–59.

Wilson, J. R. (2002). "Responsible Authorship and Peer Review." *Science and Engineering Ethics* 8(2): 155–174.

Winter, R. (2008). "Design Science Research in Europe." *European Journal of Information Systems* 17(5): 470–475.

References on Philosophical Grounding of Design Science Research

Ackoff, R. (1962). "The Nature of Science and Methodology." In *Scientific Method: Optimizing Applied Research Decisions* (Chapter 2), R. Ackoff (Ed.), New York, NY: John Wiley.

Berger, P. and Luckman, T. (1966). *The Social Construction of Reality: A Treatise in the Sociology of Knowledge*. Garden City, NY: Doubleday.

Bunge, M. (1984). "Philosophical Inputs and Outputs of Technology." In *History and Philosophy of Technology*, Bugliarello, G. and Donner, D. (Eds.), Urbana, IL: University of Illinois Press, 263–281.

Gregor, S. (2006). "The Nature of Theory in Information Systems." *MISQ* 30(3): 611–642.

Gregor, S. and Jones, D. (2007). "The Anatomy of a Design Theory." *Journal of the Association for Information Systems (JAIS)* 8(5): Article 19.

Guba, E. and Lincoln, Y. (1994). "Competing Paradigms in Qualitative Research." In *The Handbook of Qualitative Research*, N. Denzin and Y. Lincoln (Eds.), Thousand Oaks, CA: Sage, 105–117.

Hendry, R. (2004). "Are Realism and Instrumentalism Methodologically Different?" *Working paper*. Department of Philosophy, University of Durham, UK. Author e-mail: r.f.hendry@dur.ac.uk

Latour, B. (1987). *Science in Action: How to Follow Scientists and Engineers through Society*. Cambridge, MA: Harvard University Press.

Markus, M.L. and Silver, M.S. (2008). "A Foundation for the Study of IT Effects: A New Look at DeSanctis and Poole's Concepts of Structural Features and Sprit." *Journal of the Association for Information Systems (JAIS)* 9(3/4): 609–632.

Niehaves, B. (2007). "On Epistemological Diversity in Design Science—New Vistas for a Design-Oriented IS Research." In *Proceedings of ICIS 2007*, Montreal, Quebec, Canada.

Saraswat, P. (1998). "A Historical Perspective on the Philosophical Foundations of Information Systems." Document online at the website of AIS SIG Philosophy, AIS: http://www.bauer.uh.edu/parks/fis/saraswat3.htm (last accessed on January 28, 2015)

Searle, J., Ed. (1995). *The Construction of Social Reality*. New York, NY: The Free Press.

References on Design Science Research Methodology

Fettke, P., Houy, C., and Loos, P. (2010). "On the Relevance of Design Knowledge for Design-Oriented Business and Information Systems Engineering." *Business and Information Systems Engineering* 2(6): 347–358.

Fettke, P., Houy, C., and Loos, P. (2010). "On the Relevance of Design Knowledge for Design-Oriented Business and Information Systems Engineering—Supplemental Considerations and further Application Examples." In *Publications of the Institute for Information Systems at the German Research Center for Artificial Intelligence* (DFKI) Vol. 191. URL: urn:nbn:de:bsz:291-scidok-34731 (last accessed on January 28, 2015)

Jarvinen, P. (2004). *On Research Methods*. Tiedekirjakauppa TAJU publisher, Helsinki, Finland. (Chapter 1: http://www.desrist.org/desrist/content/Jarvinen/Etodit041.pdf; Chapter 5: http://www.desrist.org/desrist/content/Jarvinen/ETODIT045.pdf) (last accessed on January 28, 2015)

Jarvinen, P. (2006). "On A Variety of Research Output Types": http://www.desrist.org/desrist/content/Jarvinen/Uddevalla.pdf (last accessed on January 28, 2015). *Department of Computer and Information Sciences working paper*, University of Tampere, Finland.

Jarvinen, P. (2006). "Research Questions Guiding Selection of an Appropriate Research Method": http://www.desrist.org/desrist/content/Jarvinen/Wien00normal.pdf (last accessed on January 28, 2015). *Department of Computer and Information Sciences working paper*, University of Tampere, Finland.

Jarvinen, P. (2007). "Action Research is Similar to Design Science." *Quantity and Quality* 41(1): 37–54.

Mingers, J. (2001). "Combining IS Research Methods: Towards a Pluralist Methodology." *Information Systems Research* 12(3): 240–259.

Newell, A. (1973). Production systems: Models of control structures. In *Visual information Processing*, W. G. Chase (Ed.), New York, NY: Academic Press, 463–526.

Newell, A. (1990). *Unified Theories of Cognition*. Cambridge, MA: Harvard University Press.

Nunamaker, J., Chen, M., and Purdin, T. (1991). "System Development in Information Systems Research." *Journal of Management Information Systems* 7(3): 89–106.

Shu, N. (1998). "Axiomatic Design Theory for Systems." *Research in Engineering Design* 10(4): 189–209.

Ulrich, F. (2006). "Towards a Pluralistic Conception of Research Methods in Information Systems Research," *Tel Aviv University, Department of Management, Research Report* available at https://www.econstor.eu/dspace/bitstream/10419/58156/1/715932098.pdf (last accessed on January 29, 2015).

Vaishnavi, V. and Kuechler, W. (2007). *Design Science Research Methods and Patterns*. Boca Raton, New York: Auerbach Publications.

Walls, J., Widmeyer, G., and El Sawy, O. (2004). "Assessing Information System Design Theory in Perspective: How Useful was our 1992 Initial Rendition." *Journal of Information Technology Theory and Application* 6(2): 43–58.

Zelkowitz, M. and Wallace, D. (1998). "Experimental Models for Validating Technology." *IEEE Computer* 31(5): 23–31.

References on Understanding Design Science Research in the Context of Information Systems Research

Adams, L. and Courtney, J. (2004) "Achieving Relevance in IS Research via the DAGS Framework." In *Proceeding of the 37th Hawaii International Conference on System Sciences*. IEEE Press.

Alter, S. (2003). "18 Reasons Why IT-Reliant Work Systems Should Replace 'The IT Artifact' as the Core Subject Matter of the IS Field." *Communication of the AIS* 12(October): 365–394.

Applegate, L. (1999). "Rigor and Relevance in MIS Research—Introduction." *MIS Quarterly* 23(1): 1–2.

Arnott, D. (2006). "Cognitive Biases and Decision Support Systems Development: A Design Science Approach." *Information Systems Journal* 16(1): 55–78.

Benbasat, I. and Zmud, R. (1999). "Empirical Research in Information Systems: The Practice of Relevance." *MIS Quarterly* 23(1): 3–16.

Brooks, F. (1996). "The Computer Scientist as Toolsmith II." *Communications of the ACM* 39(3): 61–68.

Caws, P. (1969). "The Structure of Discovery." *Science* 166 (December): 1375–1380.

Falconer, D. and Mackay, D. (1999). *Ontological Problems of Pluralist Research Methodologies*. In *Proceedings 5th AIS Conference on Information Systems*, Milwaukee, WI.

Fugetta, A. (1999). "Some Reflections on Software Engineering Research." *ACM SIGSOFT Software Engineering Notes* 24(1): 74–77.

Gehlert, A., Schermann, M., Pohl, K., and Krcmar, H. (2009). "Towards a Research Method for Theory-Driven Design Research." Paper presented at the Wirtschaftsinformatik 2009, Vienna, Austria. http://aisel.aisnet.org/wi2009/42/ (last accessed on January 28, 2015).

Germonprez, M., Hovorka, D., and Gal, U. (2011). "Secondary Design: A Case of Behavioral Design Science Research." *Journal of the Association for Information Systems (JAIS)* 12(10): 662–683.

Glass, R. (1999). "On Design." *IEEE Software* 16(2): 103–104.

Glass, R., Ramesh, V., and Vessey, I. (2004). "An Analysis of Research Computing Disciplines." *Communications of the ACM* 47(6): 89–94.

Hempel, C. (1966). *Philosophy of Natural Science*. Englewood Cliffs, NJ: Prentice Hall.

Hopcroft, J. (1987). "Computer Science: The Emergence of a Discipline." *Communications of the ACM* 30(3): 198–202.

Kleindorfer, G., O'Neill, L., and Ganeshan, R. (1998). "Validation in Simulation: Various Positions in the Philosophy of Science." *Management Science* 44(8): 1087–1099.

Kolfschoten, G. and Vreede, G. de (2009). "A Design Approach for Collaboration Processes: A Multi-Method Design Science Study in Collaboration Engineering." *Journal of Management Information Systems* 26(1): 225–256.

Iivari, J. (2003). "The IS CORE VII: Towards Information Systems as a Science of Meta-Artifacts." *Communication of the AIS* 12(October), Article 37.

Kuechler, W. and Vaishnavi, V. (2007). "Design [Science] Research in IS: A Work in Progress" in *Proceedings of 2nd International Conference on Design Science Research in Information Systems and Technology* (DESRIST '07): May 13–16, 2007, Pasadena, CA.

Kuechler, W.L. and Vaishnavi, V.K. (2008). "The Emergence of Design Research in Information Systems in North America." *Journal of Design Research* 7(1): 1–16.

Lee, A. (2000). "Systems Thinking, Design Science and Paradigms: Heeding Three Lessons from the Past to Resolve Three Dilemmas in the Present to Direct a Trajectory for Future Research in the Information Systems Field." Keynote Speech at the 11th International Conference on Information Management, Kaohsiung, Taiwan.

LeRouge, C. and Lisetti, C. (2005). "Triangulating Design Science, Behavioral Science, and Practice for Technological Advancement in Tele-Home Health." *International Journal of Healthcare Technology and Management* 7(5): 348–363.

March, S., Hevner, A., and Ram, S. (2000). "Research Commentary: An Agenda for Information Technology Research in Heterogeneous and Distributed Environments." *Information Systems Research* 11(4): 327–341.

Markus, M., Majchrzak, A., and Gasser, L. (2002). "A Design Theory for Systems that Support Emergent Knowledge Processes." *MIS Quarterly* 26(3): 179–212.

Morrison, J. and George, J. (1995). "Exploring the Software Engineering Component of MIS Research." *Communications of the ACM* 38(7): 80–91.

Newell, A. and Simon H. (1976). "Computer Science as Empirical Inquiry: Symbols and Search." *Communications of the ACM* 19(3): 113–126.

Norman, D. (1988). *The Design of Everyday Things*. New York, NY: Doubleday.

Parnas, D. (1998). "Successful Software Engineering Research." *ACM SIGSOFT Software Engineering Notes* 23(3): 64–68.

Peffers, K., Tuunanen, T., Gengler, C., Rossi, M., Hui, W., Virtanen, V., and Bragge, J. (2006). "The Design Science Research Process: A Model for Producing and Presenting Information Systems Research." In *Proceedings of DESRIST 2006*, Claremont, CA: 83–106.

Petroski, H. (1996). *Invention by Design: How Engineers Get from Thought to Thing*. Cambridge, MA: Harvard University Press.

Petter, S., Vaishnavi, V., and Hsieh, J. (2003). "Linking Theory with Practice: A Research Approach and Illustration of its Use in Software Project Management." *Working Paper,* Department of Computer Information Systems, Georgia State University.

Popper, K. (1980). "Science: Conjectures and Refutations." In *Introductory Readings in the Philosophy of Science*, R. Hollinger and A. Kline (Eds.), New York, NY: Prometheus Books, 29–34.

Robey, D. (1996). "Research Commentary: Diversity in Information Systems Research: Threat, Opportunity and Responsibility." *Information Systems Research* 7(4): 400–408.

Schon, D. (1983). *The Reflective Practitioner: How Professionals Think in Action*. New York, NY: Basic Books.

Tichy, W. (1998). "Should Computer Scientists Experiment More?" *IEEE Computer* 31(5): 32–40.

Tsichritzis, D. (1997). "The Dynamics of Innovation." In *Beyond Calculation: The Next Fifty Years of Computing*, P. Denning and R. Metcaffe (Eds.), New York, NY: Springer-Verlag: 259–265.

Truex, D. (2001). "Three Issues Concerning Relevance in IS Research: Epistemology, Audience and Method." *Communications of the AIS* 6:24.

Vaishnavi, V. and Kuechler, W. (2004/13). "Design Science Research in Information Systems" January 20, 2004; last updated October 23, 2013. URL: *http://www.desrist.org/design-research-in-information-systems/* (last accessed on January 28, 2015). [Pertti Jarvinen's critique of the 2004 version: http://www.desrist.org/desrist/content/Jarvinen/VaishnaviKuechlerRev.pdf (last accessed on January 28, 2015)]

Weber, R. (1987). "Toward a Theory of Artifacts: A Paradigmatic Base for Information Systems Research." *Journal of Information Systems*: 3–19.

Winograd, T. (1996). *Bringing Design to Software*. Reading, MA: Addison Wesley.

Winograd, T. (1997). "The Design of Interaction." In *Beyond Calculation: The Next Fifty Years of Computing*, P. Denning and R. Metcaffe (Eds.), New York, NY: Springer-Verlag, 149–162.

Yetim, F. (2011). "Bringing Discourse Ethics to Value Sensitive Design: Pathways toward a Deliberative Future." *AIS Transactions on Human Computer Interaction* 3(2): 133–155.

References on Theory and Theory Development in Design Science Research

Arazy, O., Kumar, N., and Shapira, B. (2010). "A Theory-Driven Design Framework for Social Recommender Systems." *Journal of the Association for Information Systems (JAIS)* 11(9): 455–490.

Baskerville, R. and Pries-Heje, J. (2010). "Explanatory Design Theory." *Business and Information Systems Engineering* 5: 271–282.

Germonprez, M., Hovorka, D., and Collopy, F. (2007). "A Theory of Tailorable Technology Design." *Journal of the Association for Information Systems* (JAIS) 8(6): 315–367.

Goldkuhl, G. (2004). "Design Theories in Information Systems—A Need for Multi-Grounding." *Journal of Information Technology Theory and Application* 6(2): 59–72.

Gregor, S. (2009). "Building Theory in the Sciences of the Artificial." In *Design Science Research in Information Systems and Technologies, Proceedings of DESRIST 2009*, Vaishnavi, V. and Purao, S. (Eds.), Philadelphia, PA: ACM.

Holstrom, J., Ketokivi, M., and Hameri, A. (2009). "Bridging Practice and Theory: A Design Science Approach." *Decision Sciences* 40(1): 65–87.

Houy, C., Fettke, P., and Loos, P. (2011). "On Theoretical Foundations of Empirical Business Process Management Research." In *Proceedings of the 2nd International Workshop on Empirical Research in Business Process Management (ER-BPM-11)*, University Blaise Pascal, Clermont-Ferrand, France.

Kuechler, W. and Vaishnavi, V. (2008). "On Theory Development in Design Science Research: Anatomy of a Research Project." *European Journal of Information Systems* 17(5): 1–23.

Kuechler, W., Park, E.H., and Vaishnavi, V. (2009). Formalizing Theory Development in IS Design Science Research: Learning from Qualitative Research. In *AMCIS '09*, San Francisco, CA, USA.

Kuechler, W. and Vaishnavi, V. (2012). "A Framework for Theory Development in Design Science Research: Multiple Perspectives." *Journal of the Association for Information Systems (JAIS)* 13(6): 395–423.

Lee, A. and Hubona, G. (2009). "A Scientific Basis for Rigor in Information Systems Research." *MIS Quarterly* 33(2): 237–262.

Müller, B. O., S. (2011). "The Artifact's Theory—a Grounded Theory Perspective on Design Science Research." In *Proceedings of the 10th Internationale Tagung Wirtschaftsinformatik*, Zurich, Switzerland: 1176–1186.

Niehaves, B., Ortbach, K., and Tavakoli, A. (2012). "On the Relationship between the IT Artifact and Design Theory: The Case of Virtual Social Facilitation," in *DESRIST 2012, LNCS 7286*, K. Peffers, M. Rothenberger, and B. Kuechler (Eds.), Springer-Verlag Berlin Heidelberg, 354–370.

Pries-Heje, J. and Baskerville, R. (2008). "The Design Theory Nexus." *MIS Quarterly* 32(4): 731–755.

van Aken, J. (2004). "Management Research Based on the Paradigm of the Design Sciences: The Quest for Field-Tested and Grounded Technological Rules." *Journal of Management Studies* 41(2): 219–246.

Venable, J. (2006). "The Role of Theory and Theorising in Design Science Research." In *Proceedings of DESRIST 2006*, Claremont, CA.

Weber, S., Beck, R., and Gregory, R. (2011). "Combining Design Science and Design Research Perspectives—Findings of Three Prototyping Projects." In *Proceedings of HICSS 2011*, IEEE Press.

References on Design and Design Science Research

Alturki, A., Gable, G., and Bandara, W. (2011). "A Design Science Research Roadmap," in *Service-Oriented Perspectives in Design Science Research*, H. Jain, A. Sinha and P. Vitharana (Eds.), Springer, Berlin/Heidelberg, 107–123.

Archer, L. (1984). "Systematic Method for Designers," in *Developments in Design Methodology*, N. Cross (Ed.), Chichester, Brisbane: Wiley, 57–82.

Au, Y. (2001). "Design Science I: The Role of Design Science in Electronic Commerce Research." *Communication of the AIS* 7(Article 1).

Baskerville, R. (2008). "What Design Science is Not." *European Journal of Information Systems* 17(5): 441–443.

Baskerville, R., Pries-Heje, J., and Venable, J. (2009). "Soft Design Science Methodology." In *Proceedings of Design Science Research in Information Systems and Technology (DESRIST)*, Vaishnavi, V. and Purao, S. (Eds.), Philadelphia, PA: ACM.

Baskerville, R., Lyytinen, K., Sambamurthy, V., and Straub, D. (2011). "A Response to the Design-Oriented Information Systems Research Memorandum," *European Journal of Information Systems* 20(11–15).

Bayazit, N. (2004). "Investigating Design: A Review of Forty Years of Design Research," *Design Issues* 20(1): 16–29.

Carlsson, S.A. (2006). "Towards an Information Systems Design Research Framework: A Critical Realist Perspective," In *Proceedings of the First International Conference on Design Science in Information Systems and Technology*, Claremont, 192–212.

Carlsson, S. (2007). "Developing Knowledge through IS Design Science Research: For Whom, What Type of Knowledge, and How." *Scandinavian Journal of Information Systems* 19(2): 75–85.

Chow, R. and Jonas, W. (2008). "Beyond Dualisms in Methodology: An Integrative Design Research Medium "MAPS" and some Reflections." In *Undisciplined! Design Research Society Conference*, Sheffield, UK, 1–18.

Cleven, A., Gubler, P., and Huner, K. (2009). "Design Alternatives for the Evaluation of Design Science Research Artifacts." In *Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology (DESRIST)*, Vaishnavi, V. and Purao, S. (Eds.), Philadelphia, PA: ACM.

Cleven, A., Wortmann, F., and Winter, R. (2010). "Process Performance Management–Identifying Stereotype Problem Situations as a Basis for Effective and Efficient Design Research." In *Global Perspectives on Design Science Research*, R. Winter, L. Zhao and S. Aier (Eds.), Springer, Berlin, 302–316.

Cole, R., Purao, S., Rossi, M., and Sein, M. (2005). "Being Proactive: Where Action Research Meets Design Research," In *International Conference on Information Systems (ICIS)*, Las Vegas, NV, USA.

Cross, N. (1982). "Designerly Ways of Knowing." *Design Studies* 3(4): 221–227.

Cross, N. (1993). "Science and Design Methodology: A Review." *Research in Engineering Design* 5(2): 63–69.

Cross, N. (2002). "Designerly Ways of Knowing: Design Discipline versus Design Science." *Design Issues* 17(3): 49–55.

Cross, N. (2002). "Design as a Discipline." In *The Inter-disciplinary Design Quandary Conference:* http://nelly.dmu.ac.uk/4dd//DDR3-Cross.html (last accessed on January 29, 2015).

Donnellan, B. and Helfert, M. (2010). "The IT-CMF: A Practical Application of Design Science." In *Global Perspectives on Design Science Research*, R. Winter, L. Zhao and S. Aier (Eds.), Springer, Berlin, 550–553.

Eekels, J., and Roozenburg, N. (1991). "A Methodological Comparison of the Structures of Scientific Research and Engineering Design: Their Similarities and Differences." *Design Studies* 12(4): 197–203.

Fischer, C., and Gregor, S. (2011). "Forms of Reasoning in the Design Science Research Process." In *Service-Oriented Perspectives in Design Science Research*, H. Jain, A. Sinha and P. Vitharana (Eds.), Springer Berlin/Heidelberg, 17–31.

Friedman, K. (2003). "Theory Construction in Design Research: Criteria, Approaches, and Methods." *Design Studies* 24(6): 507–522.

Gacenga, F., Cater-Steel, A., and Tan, W. (2011). "Towards a Framework and Contingency Theory for Performance Measurement: A Mixed-Method Approach." In *Proceedings of the 15th Pacific Asia Conference on Information Systems (PACIS)*, Brisbane, Australia.

Gill, T., and Hevner, A. (2011). "A Fitness-Utility Model for Design Science Research." In *Service-Oriented Perspectives in Design Science Research*, H. Jain, A. Sinha and P. Vitharana (Eds.), Springer Berlin/Heidelberg, 237–252.

Gregor, S. (2002). "Design Theory in Information Systems." *Australian Journal of Information Systems: Special Issue*, 14–22.

Hevner, A., Chatterjee, S., and Iivari, J. (2010). "Twelve Theses on Design Science Research in Information Systems." In *Design Research in Information Systems*, New York, NY: Springer, 43–62.

Hevner, A. and Chatterjee, S. (2010). *Design Research in Information Systems: Theory and Practice*, New York, NY: Springer, p. 320.

Hjalmarsson, A., Rudmark, D., and Lind, M. (2010). "When Designers Are Not in Control–Experiences from Using Action Research to Improve Researcher-Developer Collaboration in Design Science Research." In *Global Perspectives on Design Science Research*, R. Winter, L. Zhao and S. Aier (Eds.), Springer, Berlin, 1–15.

Jonas, W. (2007). "Research through DESIGN through Research: A Cybernetic Model of Designing Design Foundations." *Kybernetes* 36(9/10): 1362–1380.

Junglas, I., Niehaves, B., Spiekermann, S., Stahl, B.C., Weitzel, T., Winter, R., and Baskerville, R. (2010). "The Inflation of Academic Intellectual Capital: The Case for Design Science Research in Europe." *European Journal of Information Systems* 20: 1–6.

Kuechler, W. and Vaishnavi, V. (2008). "The Emergence of Design Research in Information Systems in North America." *Journal of Design Research* 7(1): 1–16.

Kuechler, B. and Vaishnavi, V. (2011). "Extending Prior Research with Design Science Research: Two Patterns for DSRIS Project Generation." In *Service-Oriented Perspectives in Design Science Research*, H. Jain, A. Sinha and P. Vitharana (Eds.), Springer Berlin/Heidelberg, 2011, 166–175.

Lohman, C., Fortuin, L., and Wouters, M. (2004). "Designing a Performance Measurement System: A Case Study." *European Journal of Operational Research* 156(2): 267.

McNaughton, B., Ray, P., and Lewis, L. (2010). "Designing an Evaluation Framework for IT Service Management." *Information & Management* 47(4): 219–225.

Nunamaker, J. and Chen, M. (1990). "Systems Development in Information Systems Research." In *System Sciences, Proceedings of the Twenty-Third Annual Hawaii International Conference*, IEEE Press, 631–640.

Offermann, P., Blom, S., Levina, O., and Bub, U. (2010). "Proposal for Components of Method Design Theories." *Business & Information Systems Engineering* 2(5): 295–304.

Offermann, P., Blom, S., Schönherr, M., and Bub, U. "Artifact Types in Information Systems Design Science—A Literature Review." In *Global Perspectives on Design Science Research*, 77–92.

Offermann, P., Levina, O., Schonherr, M., and Bub, U. (2009). "Outline of a Design Science Research Process." In *Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology*, Vaishnavi, V. and Purao, S. (Eds.), Philadelphia, PA: ACM, 1–11.

Osterle, H., Becker, J., Frank, U., Hess, T., Karagiannis, D., Krcmar, H., Loos, P., Mertens, P., Oberweis, A., and Sinz, E. (2011). "Memorandum on Design-Oriented Information Systems Research." *European Journal of Information Systems* 20, 7–10. doi:10.1057/ejis.2010.55; published online, December 7, 2010.

Patas, J., Milicevic, D., and Goeken, M. (2011). "Enhancing Design Science through Empirical Knowledge: Framework and Application." In *Service-Oriented Perspectives in Design Science Research*, H. Jain, A. Sinha and P. Vitharana (Eds.), Springer Berlin/Heidelberg, 32–46.

Piirainen, K. and Briggs, R. (2011). "Design Theory in Practice—Making Design Science Research More Transparent." In *Service-Oriented Perspectives in Design Science Research*, H. Jain, A. Sinha and P. Vitharana (Eds.), Springer Berlin/Heidelberg, 47–61.

Piirainen, K., Gonzalez, R., and Kolfschoten, G. (2010). "Quo Vadis, Design Science?–A Survey of Literature." In *Global Perspectives on Design Science Research*, R. Winter, L. Zhao and S. Aier (Eds.), Springer, Berlin, 93–108.

Pries-Heje, J., Baskerville, R., and Venable, J. (2008). "Strategies for Design Science Research Evaluation." In *Proceedings of the 16th European Conference on Information Systems (ECIS)*, Galway, Ireland, 255–266.

Roozenburg, N.F.M. and Eekels, J. (1995). *Product design: Fundamentals and Meth*ods. Wiley, Chichester, NY.

Samuel-Ojo, O., Shimabukuro, D., Chatterjee, S., Muthui, M., Babineau, T., Prasertsilp, P., Ewais, S., and Young, M. (2010) "Meta-analysis of Design Science Research within the IS Community: Trends, Patterns, and Outcomes." In *Global Perspectives on Design Science Research*, R. Winter, L. Zhao and S. Aier (Eds.), Springer, Berlin, 124–138.

Sein, M., Henfridsson, O., Purao, S., Rossi, M., and Lindgren, R. (2011). "Action Design Research." *MIS Quarterly* 35(1): 35–56.

Son, S., Weitzel, T., and Laurent, F. (2005). "Designing a Process-Oriented Framework for IT Performance Management Systems." *The Electronic Journal of Information Systems Evaluation* 8(3): 219–228.

Toleman, M. (1996). "The Design of the User Interface for Software Development Tools." *Doctoral Dissertation*, Department of Computer Science, University of Queensland, Australia.

Tovey, M. (1984). "Designing with both Halves of the Brain." *Design Studies* 5(4): 219–228.

Venable, J. (2010). "Design Science Research Post Hevner et al.: Criteria, Standards, Guidelines, and Expectations." In *Global Perspectives on Design Science Research*, R. Winter, L. Zhao and S. Aier (Eds.), Springer, Berlin, 109–123.

vom Brocke, J. and Lippe, S. (2010). "Taking a Project Management Perspective on Design Science Research." In *Global Perspectives on Design Science Research*, R. Winter, L. Zhao and S. Aier (Eds.), Springer, Berlin, 31–44.

Wieringa, R. (2010). "Relevance and Problem Choice in Design Science." In *Global Perspectives on Design Science Research*, R. Winter, L. Zhao and S. Aier (Eds.), Springer, Berlin, 61–76.

Zahedi, F. and Sinha, A. (2010). "Ontology Design for Strategies to Metrics Mapping." In *Global Perspectives on Design Science Research*, R. Winter, L. Zhao and S. Aier (Eds.), Springer, Berlin, 554–557.

Chapter 3

Aggregate Design Science Research Cycle as a Perspective on the Evolution of Computing Communities of Interest*

Introduction

The design science research cycle (DSRC) (Figures 2.3 and 2.4 in Chapter 2), in addition to being an empirically observed description of individual (or project team) design science research activity across multiple fields, is a powerful framework for understanding intellectual development at broader levels of human activity. In this chapter, the DSRC is expanded so that the actors are *communities*—of practice or research—and in each iteration through the cycle different communities, united only by a common interest in some aspect of a broadly useful artifact, for example, databases, pass information between each other via journals and other media, conferences, and social networks. From this perspective, information and communication technology (ICT) design science research projects can be seen not only to use the DSRC but also to participate in a broader, inter-group intellec-

* Adapted from the authors' article in *Computing Letters*, Vol. 1(3): 123–128, 2005.

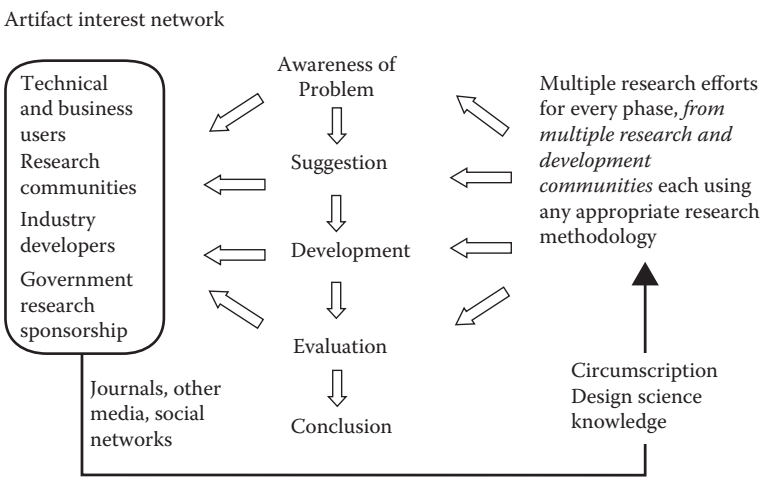


Figure 3.1 Aggregate design science research cycle.

tual conversation modeled by a collective version of the DSRC—the A(ggregate) DSRC (Figure 3.1).

Insight into the subject emphasis of a research community is valuable information both for the members of that community and its related areas, and for determining the degree of alignment between the research community and commercial applications of the research (Culnan 1987). Knowledge of the mechanisms by which a research community chooses to direct its resources is also of interest to the academic community in general, and to researchers in organizational behavior and dynamics, including those interested in group decision theory and concept diffusion through groups (Alavi et al. 1989). Research by several authors supports the commonsense observation that a *research community* is actually an aggregation of "invisible colleges" (Culnan 1987; Pfeffer et al. 1977), each with specific research directions, under a common "umbrella" heading. The common heading is an accurate gauge of general direction, but is always broad enough to support (and require) meaningful sub-topics "which tend to concentrate on examining common [highly specific] questions in common ways" (Pfeffer et al. 1977). Yet, despite interest in understanding research directions, no research that we are aware of has attempted to model the dynamics of an extended research community.

In an earlier work, we developed a cognitive model of design—the general design cycle (Takeda et al. 1990)—into a descriptive model of *design science research*, the *generation of knowledge through making* that typifies ICT and many engineering fields (Vaishnavi and Kuechler 2004/13). In this chapter, we first explain the general DSRC framework as a research approach to create an artifact. We then show how the general DSRC in aggregate form can be interpreted as a

framework for understanding how multiple streams of ICT research from varying disciplines converge to support the evolution of a complex computing artifact over time.

Design Science Research Cycle

Takeda et al. (1990) have analyzed the reasoning that occurs in the course of a general design cycle. Vaishnavi and Kuechler (Vaishnavi and Kuechler 2004/13) (see Chapter 2) have extended this analysis to explicate the knowledge generated in a design effort and apply the cycle specifically to design science research as illustrated in Figure 2.4 (in Chapter 2). In following the flow of creative effort through this diagram, the types of new knowledge that arise from design activities and the reason that this knowledge is most readily found during a design effort will become apparent.

Design science research is sometimes called "improvement research" and this designation emphasizes the problem-solving/performance-improving nature of the activity. In this model, all design begins with *awareness of problem*. The problem may be identified by a literature review, an experience in practice, or even conversations with colleagues. In the *awareness of problem* phase, the problem is not only identified but also defined. Explicit use of the DSRC prompts researchers to spend "more time defining the problem before deciding to build a tool" (Purao 2002). To properly define the problem, an initial literature review should attempt to (1) determine that the problem has not been previously solved and determine what, if any, research has been previously performed in the area, (2) determine that the problem is widespread and that the solution will be an interesting contribution to the practice and academic communities, and (3) define and scope the problem as appropriate for the resources available to the project. Gaps in current research should become readily apparent.

Suggestions for a problem solution are abductively drawn from the existing knowledge/theory base for the problem area (Pierce 1931) or developed creatively using an appropriate research methodology. Existing literature may be a sufficient guide to provide suggestions on the artifact to be developed; however, conducting an explanation research study* may also be helpful in identifying potential suggestions. The research conducted at the *suggestion* phase is used to create a tentative design for the artifact.

An attempt at implementing the artifact according to the suggested solution (or tentative design) is performed next. This stage is shown as *development* in the diagram. This phase of the DSRC is where most of the actual design takes place, which

* While design science research seeks to solve a problem or improve practice, explanation research aims to understand why a phenomenon occurs through the use of quantitative and/or qualitative research data collection and analysis.

is the effort required in synthesizing existing knowledge and a well-defined problem into an artifact for solving the problem. This is the only phase of the DSRC that requires a constructivist methodology. The artifact developed in this stage may be rather abstract in nature—such as constructs, models, or methods—or can be more tangible in the form of computing software or hardware (March and Smith 1995). In the development phase, the artifact's instantiation may be rather rudimentary as one focuses on design, rather than the implementation of the artifact (Vaishnavi and Kuechler 2004/13).

Partially or fully successful implementations are then evaluated according to the functional specification implicit or explicit in the suggestion (*evaluation* phase). After the development of an artifact, it is necessary to evaluate the artifact using empirical methods "to determine how well an artifact works" (Hevner et al. 2004). Researchers should evaluate their artifacts using methods and techniques similar to theory testing (March and Smith 1995), including action research, controlled experiments, simulation, or scenarios. The evaluation portion of the design science research approach does not signify a conclusion to research, but an opportunity to further refine the artifact through insight and suggestion (see the circumscription arrow in Figure 2.4).

Development, *evaluation*, and further *suggestion* are frequently iteratively performed in the course of the research effort. The basis of the iteration, the flow from partial completion of the cycle back to *awareness of problem*, is indicated by the circumscription arrow. *Conclusion* indicates termination of a specific design science research project. The cognitive underpinning of design science research (Hevner et al. 2004) is discussed fully in Vaishnavi and Kuechler (2004/13) (see Chapter 2).

Aggregate DSRC

The DSRC (see Figure 2.4) shows the design effort in-the-small, that is, as used for an individual design science research effort. Even at that level, an analysis of the cycle in use shows that each phase (*awareness of problem*, *suggestion*, etc.) comprises a sometimes brief but completely articulated research effort in itself. For example, "Example of Design Science Research" in Chapter 2 and Vaishnavi and Kuechler (2004/13) describe a design science research effort longitudinally and in that example the *awareness of problem* stage involved several months of field investigation in the area of interest; the *suggestion* stage likewise involved extensive library research and a pilot development program, and so on through all the phases. Though the ultimate intent of the process as a whole was the production of an artifact to support a specific type of operations environment, each stage of the cycle was a distinct research effort involving a methodology appropriate to gathering the information required at that stage of development. Frequently, the research phases did *not* use a constructivist methodology, but rather a meta-bibliographic

study or survey, a structured interview in a field setting, a small action research effort, or ethnography.

We propose that an aggregated form of the DSRC, which we term the aggregate design science research cycle (ADSRC), is an accurate depiction of a collective, longitudinal research stream in many areas of ICT research. The ADSRC shown in Figure 3.1 is an abstraction of the DSRC and includes (1) the aggregation of research and development efforts from multiple research programs in multiple communities into an *interest network* for the artifact and (2) the dissemination of knowledge and insights from the network back to individual research efforts. By collective research stream we mean the accumulated efforts of many researchers, which are considered to have a common focus. The focus can nearly always be taken to be the artifact produced by the development phase. The *artifact interest network* provides the only coordination available or needed to make a coherent stream from otherwise disjointed efforts contributed by different individuals or groups in different places at different times. The development phase of the ADSRC uses a constructivist methodology to create or enhance an artifact. Just as the DSRC uses different information gathering techniques in each of its phases, research efforts in any of the other phases of the ADSRC use any methodology appropriate for gathering the information required to motivate or evaluate the artifact. Indeed, the researchers involved in the other phases of an aggregate research stream (considered through the lens of the ADSRC) may be surprised to see themselves as part of what we consider in our framework as an extended development effort.

Exercising the ADSRC Framework: Concept Mapping 25 Years of Database Research

We confine our empirical support for the ADSRC as an explanatory framework to one area: database research. We have drawn from an earlier work in which we concept mapped (Eden 1992) over 500 papers from multiple communities on database research, use, and development written over a 25 year period (Kuechler and Beranek 1995); however, we believe the same technique applied to any significant ICT artifact (e.g., the WWW) would provide similar findings.

The first "database" papers were purely conceptual focusing on the obvious desirability (awareness of problem, from Figure 3.1) of having a data store detached from computer programs themselves. The vision of an integrated computer accessible data store that would permit queries to support decision making actually predates the use of computers in business (Bush 1945). More conceptual papers followed (suggestion), and prototypes of early databases were developed both in academic and industry settings (development) (Bachman 1972). The military has always been influential in database research, primarily through research grants and

early adoption of technologies. The earliest versions of the COBOL language contained advanced (for the time) information retrieval and manipulation commands largely due to the military influence.

Once databases began to be used in business, the *artifact network of interest* expanded from the computer science community and the development labs of large corporations to include business and the nascent management information systems (MIS) community, at that time housed in the management science colleges of larger universities. Papers concerned with technical and business use of databases and their implementation in business settings began to appear, which examined the usefulness and impact of databases in business (evaluation). The evaluation phase exposed a new set of problems inherent in isolated data structures (feedback from the artifact interest network to research programs; the long forward arrow in Figure 3.1) and a new round of conceptual papers began (*awareness of problem* and *suggestion*). A widely cited paper that influenced database use in business and indirectly set research agendas for computer science and information systems researchers for many years was Richard Nolan's 1973 Harvard Business Review paper, "Computer data bases: the future is now" (Nolan 1973). At a time when IS was still termed electronic data processing (EDP), Nolan's critique of information "silos" and call for organization-wide databases set out requirements for the database artifact that would take decades of technological research and development to realize.

Our meta-bibliographic study of the database area shows the same progression through the ADSRC phases in cycles for each of the major technical advances in databases: hierarchical, relational, and object oriented (multimedia). Moreover, as the importance (synonymous with *general use*) of the artifact increased, more communities became involved in the artifact network of interest. For example, the accounting community, over many years, has produced the evaluation phase research on the difficulties of audit and control (security) of databases, a topic that has led to major streams of research in the ICT fields. As early as 1971, papers appeared in social science journals, which foresaw the impact of databases on organizational work habits and on privacy issues (*awareness of problem*, *suggestion*, and *evaluation* phases of Figure 3.1) (Trystam 1971). The impact of these papers on technical research was indirect, providing the background that underlay continuous support for research in database security, transaction processing, and backup.

Using the ADSRC to Explain Coordination between Diverse Groups

A primary contribution of the ADSRC model is to expose the frequently invisible interaction and support that normally disparate communities provide to each other

through the artifact interest network. The interactions are complex and, between communities, indirect, which contributes to their invisibility. Various research groups are highly focused on frequently divergent goals, which make direct communication between groups unlikely. For example, MIS academics have an organizational rather than a technical focus and may publish research in management journals that are broadly influential on an artifact, yet will never be read by the technical researchers or developers whose artifacts are changed as a result. More concretely, in 20 years in industry the first author (of this chapter) does not recall reading a single accounting or MIS paper on the implementation of computing artifacts. However, the business people with whom he consulted drew their ideas on database implementation exclusively from these media. They communicated their (frequently inscrutable) requirements to him, and he communicated these in turn to the industry developers of the products he sold and installed.

Conclusion

We believe the ADSRC model reflects a sociological reality—it describes how computing research and development for a complex artifact actually originates and evolves over time. If each community in the ADSRC is considered as a node in a nomological network, it is readily apparent that the centroid of that network is the artifact produced by the development phase of the ADSRC.

More generally, the history of computing and information systems can be viewed as a loosely coupled nomological network of ADSRCs, each centered on a specific computing-related artifact. Within each ADSRC, the process can be observed to have occurred as follows:

- A problem reaches a level of critical interest within the research community and an artifact is produced.
- The artifact is then investigated by researchers from differing backgrounds, interests, goals, and research traditions, all seeking to understand some aspect of this new phenomenon.
- When a sufficient body of research has accumulated, we propose that the accumulated research centered on this new artifact invariably, although without conscious coordination, takes the form we have described as an ADSRC.
- If the artifact is sufficiently interesting or commercially or culturally significant, research in all phases of its ADSRC continues, operating as a self-organizing complex system with the artifact as its primary attractor (Mikhailov and Calenbuhr 2002).

We have shown through a meta-bibliographic study of the database literature over 25 years that the ADSRC model applies to this artifact, and proposed

that a similar analysis of any computing artifact will demonstrate the interaction between communities described by the ADSRC as its evolution is traced. The model is currently incomplete; the actual dynamics of inter-group communication lie within the artifact interest network, a black box in our model. Various aspects of the diffusion of information within and across groups have been explored by sociologists, IS, and other organizational researchers (Alavi et al. 1989) and philosophers of science (Kuhn 1962/96). However, other significant artifact coordination mechanisms have never been studied; large corporations such as IBM and NCR have always served as information clearing houses for business, industry, and academic research, yet we did not find any studies on the mechanism or effect of this important coordination nexus. Incorporating new work in these areas and prior findings into the ADSRC model is an interesting area for future research.

References

Alavi, M., Carlson, P., and Brooke, G. (1989). "The Ecology of MIS Research: A Twenty Year Review." *Proceedings of the 10th International Conference on Information Systems*, 363–375.

Bachman, C. (1972). "The Evolution of Storage Structures." *Communications of ACM* 15(7): 628–634.

Bush, V. (1945). "As We May Think." *Atlantic Monthly* 176(1): 101–108.

Culnan, M. (1987). "Mapping the Intellectual Structure of MIS: 1980–1985: A Co-Citation Analysis." *MIS Quarterly* 11(3): 341–353.

Eden, C. (1992). "On the Nature of Cognitive Maps." *Journal of Management Science* 29(3): 261–265.

Hevner, A., March, S. Park, J., and Ram, S. (2004). "Design Science in Information Systems Research." *MIS Quarterly* 28(1): 75–105.

Kuechler, W. and Beranek, M. (1995). "The Intellectual Structure of Database Research: A 25 Year Perspective." In *Proceedings of the 4th Georgia Research In Information Systems Conference*, Columbus, GA.

Kuhn, T. (1962/96). *The Structure of Scientific Revolutions* (3rd edition), Chicago: The University of Chicago Press.

March, S. and Smith, G. (1995). "Design and Natural Science Research on Information Technology." *Decision Support Systems* 15(4): 251–266.

Mikhailov, A. and Calenbuhr, V. (2002). *From Cells to Societies: Models of Complex Coherent Action*, New York: Springer.

Nolan, R. (1973). "Computer Data Bases: The Future is Now." *Harvard Business Review* 6(2): 98–114.

Pfeffer, J., Leong, A., and Strehl, K. (1977). "Paradigm Development and Particularism: Journal Publication in Three Scientific Disciplines." *Social Forces* 55(4): 938–951.

Pierce, C.S. (1931). *Collected Papers,* eds. C. Harshorne and P. Weiss, Cambridge: Harvard University Press.

Purao, S. (2002). Truth or Dare: Design Research in Information Technology, Working Paper, http://purao.ist.psu.edu/working-papers/dare-purao.pdf. (last accessed on January 29, 2015).

Takeda, H., Veerkamp, P., Tomiyama, T., and Yoshikawam, H. (1990). "Modeling Design Processes." *AI Magazine* 11(4): 37–48.

Trystam, J. (1971). "From Automatic Documentation to the Data Bank." *International Social Science Journal* 23(2): 285–293.

Vaishnavi, V. and Kuechler, W. (2004/13). "Design Science Research in Information Systems." January 20, 2004; last updated October 23, 2013. URL: *http://www.desrist.org/design-research-in-information-systems/* (last accessed on January 29, 2015).

Chapter 4

A Framework for Theory Development in Design Science Research: Multiple Perspectives*

Introduction

This chapter is the first of two chapters that delve into the recent developments in design science research (DSR) while focusing on information systems. It presents and discusses a framework for theory development in design science research in information systems (IS), which is readily extendible to DSR and in particular to DSR in ICT. This chapter stays close to the *JAIS* journal article source from which it is adapted; for example, it uses the term information systems design theory (ISDT) instead of Design Theory (DT), which is a formulation of design theories for information systems provided by Walls et al. (1992, 2004).

Historically it has taken over 350 years, from Francis Bacon (1994/1620) to Karl Popper (1989), to define and refine the current Western scientific notion of theory. Shelves of books (Dubin 1978, is an example well known to IS PhD students) and reams of academic publications have been written in the attempt to define theory and evolve methods for its creation. Recently, a widely cited paper was published

* Adapted from the authors' article in *Journal of the Association for Information Systems (JAIS)*, 13(6): 395–423, June 2012.

just on the single aspect of *identifying types of theory* in IS (Gregor 2006). We give this prelude to provide perspective on the limitations of any single work in the area of theory development. Our goal is certainly not to define theorizing in design science research in IS (DSR-IS) but rather to move a step toward that goal by describing a framework for theory development in DSR-IS and providing some simple illustrations of its use.

Design Science Research in IS (DSR-IS) Defined

In this chapter, we use the term DSR-IS to indicate IS research that uses artifact design and construction (learning through building) to generate new knowledge and insights into a class of problems.* Our framework for theory development in DSR-IS makes minimal requirements on the method by which DSR-IS is accomplished. It requires three general activities: (1) construction of an artifact where construction is informed either by practice-based insight or theory, (2) gathering of data on the functional performance of the artifact (i.e., evaluation), and (3) reflection on the construction process and on the implications the gathered data (from activity (2)) have for the artifact informing insight(s) or theory(s). The definition of IS artifact used in this discussion is the ensemble view from Orlikowski and Iacono (2001) and is very broad, including software, composite systems of software, users and use processes, and IS-related organizational methodologies and interventions.

Within the framework developed here, there are three potential outputs of a DSR-IS project: (1) an artifact, (2) an information systems design theory (ISDT), and (3) design-related explanatory/predictive theory (DREPT) (see Table 4.1). An artifact, in the broad definition given earlier, is mandatory. ISDT and DREPT are optional; the development of either or both for a DSR-IS project depends on the details of the project and the intention of the researchers. Within the DSR-IS community, several structures for ISDT have been proposed, as explained in more detail in a following section of this chapter. In each case, however, an ISDT captures design information on the class of artifacts of which the specific artifact created in the DSR-IS project is an expository instantiation. An *expository instantiation* is a part of the Gregor and Jones (2007) structure for an ISDT and serves as both a proof of concept of the ISDT and in many cases as the most readily comprehensible illustration of the ISDT (e.g., screen shots of a prototype). Many recent published examples of DSR-IS (Jones and Gregor 2006; Hall et al. 2003; Markus et al. 2002) have included an ISDT and we believe this to be an increasing trend.

A formal definition of DREPT is a novel contribution of this chapter and is developed in later sections of the chapter. However, the equivalent of DREPT (or

* The study of the act of design itself and of designers is design research (DR—without the word "science"). We consider both to be valued directions in IS research; however, DR is tangential to the subject of this chapter.

Table 4.1 An Acronym Quick Reference for Non-Design-Science Researchers

Acronym	*Expansion*	*Definition*
DSR-IS	Design science research in information systems	A research methodology in the information systems discipline in which new knowledge is produced by the construction and evaluation of "artifacts," broadly defined as *software, composite systems of software, users and use processes, and IS-related organizational methodologies and interventions*. Key elements distinguishing DSR-IS from behavioral IS research are the ability to explore new, as yet un-theorized areas, constructivist rather than statistical methods, and, as suggested in this chapter, the ability to build as well as test theory.
ISDT	Information systems design theory	As initially introduced by Walls et al. (1992, 2004), an ISDT is a set of primarily prescriptive statements describing how a class of artifacts should behave (meta-requirements) and how they can be constructed. Recently, suggestions have been put forth for expanding the scope of "information systems design theory" to include more "justificatory knowledge," that is information indicating why the artifact behaves as it does (Gregor and Jones 2007).
DREPT	Design relevant explanatory/ predictive theory	A type of theory suggested in this chapter (and the journal paper it is adapted from) that augments the "how" information content of the traditional ISDT statement with explanatory information explaining why the artifact has the effects it does. The explanatory information may borrow theoretical information from the natural, social, or design sciences. DREPT is similar to but more formally stated than the "justificatory knowledge" proposed as an addition to ISDT.

its functionality) has been informally described in several papers (Gregor and Jones 2007, Kuechler and Vaishnavi 2008b, Sein et al. 2011). DREPT explains how and why the artifact functions as it does; specifically it explains how novel artifact design features have the effects they do.

The three general activities of DSR-IS methods required by our framework are consistent with all the most widely cited discussions of such methods (Nunamaker et al. 1991; Walls et al. 1992; March and Smith 1995, Vaishnavi and Kuechler

2004/13; Hevner et al. 2004; Peffers et al. 2007; Gregor and Jones 2007). We note that March and Smith (1995), Hevner et al. (2004), and Peffers et al. (2007) are moot on general activity (3), that is, they do not explicitly mention reflection but certainly do not preclude it. In fact, the activity of reflection simply adds an additional, compatible step to any published DSR-IS methodology.

Recently, a new method for design science research in IS was proposed by Sein et al. (2011), *action design research*. Action design research, as we understand it, differs from "traditional design science research" by requiring on-organizational-site artifact implementation and evaluation so that the artifact emerges from both designer/researcher vision and interaction of the artifact and its designers with the organizational environment. The emergence takes place in the building, intervention, and evaluation phase of the research, during which "the problem and the artifact are continually evaluated." Though novel in some aspects, action design research explicitly prescribes all three of the general activities our framework for theory development requires. In fact, formalization of learning as Sein et al. (2011) define it is exactly our general activity (3) mentioned earlier: "These outcomes [from artifact implementation] can be characterized as design principles [ISDT] and with further reflection, as refinements to theories that contributed to the initial design [DREPT]" (Sein et al. 2011, p. 44).

As DSR-IS has matured the stress has increasingly shifted from the artifact itself to the abstracted requirements and methods for its design as primary deliverable from a DSR-IS effort (Gregor and Jones 2007; Kuechler and Vaishnavi 2008a). The most commonly understood form for this prescriptive information is termed a design theory (ISDT—information systems design theory) for the *class* of artifacts of which the specific artifact in the DSR-IS project is an instantiation (Walls et al. 1992, 2004). Walls et al. suggested a specific format for an ISDT, and many DSR-IS exemplars (Markus et al. 2002; Hall et al. 2003; Jones and Gregor 2006) have followed this definition to varying degrees. A template indicating the components of an ISDT is shown in Table 4.2.*

The ISDT of Table 4.2 is broadly divided into description of the functionality of a class of artifacts—the meta-requirements and meta-design of the design product—and the techniques for creation of an instance of the class—the design method of the design process. Both design product and design process may specify kernel theories, typically defined as "natural science theories from other disciplines" (March and Smith 1995) that suggest either the meta-requirements or the construction process. Following more recent published design science examples (Sein et al. 2011; Arnott 2006; Iverson et al. 2004), we have broadened the scope of kernel theories to include social, mathematical, and design science theories as well as natural

* As discussed in Chapter 2 (Outputs of Design Science Research), not every design science research project produces a full, formal theory (ISDT). Nascent or partial design theories are actually more common results (along with a functioning, evaluated artifact). In this chapter, for clarity, we use only fully developed ISDT as a contrast to explanatory design theories.

Table 4.2 Content Categories of Information System Design Theory (ISDT)

	ISDT Components
Design Science Research Artifact	1. Meta-requirements
	2. Meta-design
	3. Theories that inform or suggest the solution (from any field)
	4. Artifact evaluation criteria and measures
Artifact Design Process	1. Design method
	2. Theories that inform or suggest the design method
	3. Design process evaluation criteria and measures

Source: Adapted from Walls, J.G. et al. *Journal of Information Technology Theory and Application*, 6(2): 43–58, 2004.

science (e.g., physics and psychology) theories. The format of the ISDT content as logical statements—of functionality, design specifications, methods, and tests for these (the design hypotheses)—makes it apparent that an ISDT is by its nature and intent prescriptive.* An ISDT is similar to what is called a *model* in computer science and some engineering disciplines (Evbuowan et al. 1996); it provides high level definition of the functioning of an artifact to achieve a design goal and direction toward its construction, but it does not describe how the artifact works or by what mechanism(s) the meta requirements and design method achieve the design goal.

Although kernel theories are suggested components of an ISDT, the Walls et al. (1992, 2004) ISDT framework is moot as to how the kernel theory relates to or suggests the prescribed design. A more abstract type of design-relevant explanatory-predictive theory (DREPT) is required to capture that knowledge. We propose that in many cases this information can be as valuable to the cumulative work of IS researchers in an area as the artifact or the ISDT itself. Beyond content, we will show that DREPT is a useful formalism in a framework for theorizing in DSR-IS; it provides a logical step that bridges the conceptual distance between kernel theory constructs and artifact features. This conceptual distance is a near synonym for the

* To avoid a common misunderstanding, we note a difference between the terms prescriptive and normative. A prescription suggests action in a given circumstance in order to achieve an effect. Prescription does not imply logical completeness, that is, it makes no claim that it is the only action available in the circumstance to achieve the effect, nor does it imply, as a normative statement does, an imperative, an ought, that suggests this action and only this action is appropriate for achieving the effect in the given circumstance.

"creative leap" in DSR-IS, from theory to artifact, which has proven confounding to those from research traditions more centered in formal logic who attempt to understand DSR-IS.

A Knowledge Representation Perspective on the Framework

As the logical entry to our framework for theory development in DSR-IS, we explicitly represent ISDT and DREPT as *knowledge representations*, each capturing a different sort of design-related knowledge; this is illustrated in Figure 4.1. In a later section, we discuss theory generation with respect to specific phases in the progression of a design science research project; for now we note simply that under our framework there is an evolution and translation of DSR-IS knowledge from general explanatory or predictive constructs to the design features of an IS/IT artifact. The explanatory or predictive knowledge may originate in kernel theories or in experience-based insights (evidence-based justification) and intuitions, but it always exists; to suggest otherwise is to imply design is a random process. Note that ISDTs and DREPTs are mid-range theories, conceptual intermediaries between the highly abstract space of potential problem solutions suggested by kernel theories or insights and the concrete problem solution of the implemented artifact. The arrows of Figure 4.1 represent *logical progression*—from highly abstract notions through their progressive concretization to the artifacts themselves. As we will discuss in a later section, the actual *process* of knowledge translation and development in DSR-IS may also proceed inductively (from right to left in Figure 4.1) as well as deductively/abductively (from left to right in Figure 4.1) or along both paths.

Arrow 1 in Figure 4.1 represents design science research in IS that is *pre-ISDT* and as it is still occasionally seen in fields such as engineering and computer science. An artifact is designed and implemented as a solution to a problem addressable through technology. However, in an *Arrow-1-type* presentation of design science research, the artifact stands or falls on its own merits; there is no discussion of *how* the artifact features achieve the desired effects or even of the design techniques used in its construction. This method of doing design science research works best for truly groundbreaking innovations where the artifact presented is a singular, immediately useful contribution. Entity-relationship modeling for databases (Chen 1976) is the classic example of this sort of design science research.

Arrow 2 in Figure 4.1 is indicative of the current state of DSR-IS. The seminal insight in the development of an ISDT by Walls et al. (1992) was that the work done in creation of a problem-solving artifact would be much more valuable to both research and practice by formally codifying the design effort. Without this codification (in an ISDT), the requirements for the artifact and the method of its design would have to be explicated from a description of the artifact and/or from

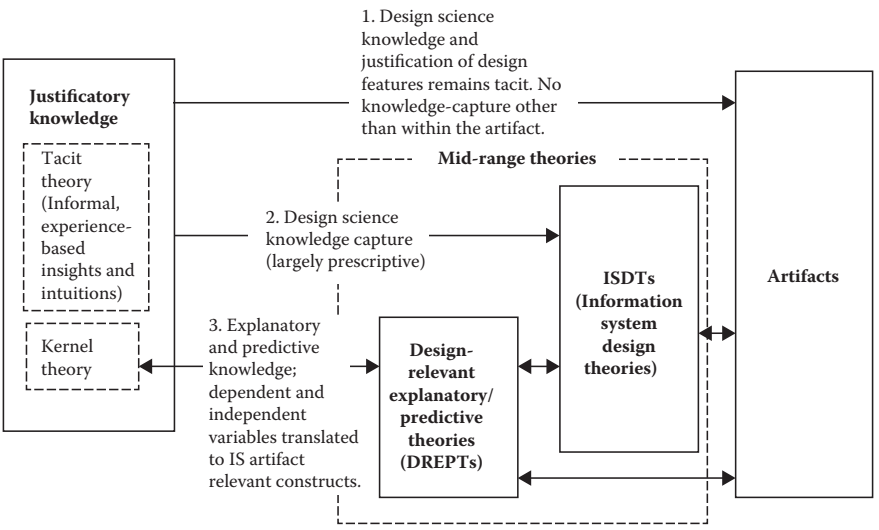


Figure 4.1 Information systems design theories and design-relevant explanatory/ predictive theories as mid-range knowledge representations in design science research.

observation of its working. For practitioners, such a level of effort is frequently too great to attempt. Even for researchers with a dedicated interest in the research area, deconstructing artifact performance to understand design principles and requirements is a difficult and time-consuming duplication of effort. A second valuable principle in the Walls et al. (1992, 2004) conception of an ISDT is abstraction: rather than codifying the design for a specific artifact implementation, an ISDT captures "meta" requirements and a "meta" design that are applicable to a *class* of artifacts. The specific implementation resulting from the DSR-IS project is just one example of the class.

Arrow 3 of Figure 4.1 illustrates the knowledge translation in the course of DSR-IS using a second mid-level representation, design-relevant explanatory/predictive theory (DREPT). DREPT extends the two insights that motivated ISDT. First, it captures knowledge generated during a design science research project that is *not* captured in an ISDT: the translation of highly abstract constructs from natural, social, or design science fields to the realm of artifact-achievable effects. By linking effects, the causes of which are explained by the kernel theory, with design features, the DREPT explains *how* and *why* a design based on the DREPT achieves its desirable novelty. ISDT, by contrast, is almost exclusively concerned with *implementation:* what and how to build (meta-requirements and meta-design). Kernel

theories are mentioned as having a relationship to the design, but no guidance on their refinement to design principles is given.

Second, DREPT is more abstract than ISDT and is therefore more broadly applicable. A DREPT captures knowledge that can be useful in the design of multiple classes of artifacts related by a common desirable *effect* (e.g., increased user attention to displayed information). Design science researchers typically pursue *knowledge in and of a field—explanatory and predictive understanding of the field*—as well as seek artifacts to provide concrete solutions to problems in that field. DREPT can provide the means to build such an explanatory and predictive knowledge base.

To clarify the translation from kernel theories to DREPT (Arrow 3 in Figure 4.1), consider a concrete example of a DREPT and the way in which it is developed. The example is a design science research project to improve *recommender systems*, such as the one used by Amazon to suggest book choices.* In the course of the project, during the researching of prior work on the subject, a number of potentially useful theories from social psychology, consumer research, and related fields are identified. These theories are broad explanatory theories (kernel theories) of *advice taking* and *trustworthiness of information* and they have no relationship to recommender systems or to technology in any form. The contribution they may have to enhanced recommender system design is thus "gut level" and highly nebulous. Therefore, before proceeding to design the recommender system (artifact), the researchers feel it would be beneficial to derive, from the kernel theories they have identified, a *technology-focused theory of advice taking* to guide their design efforts. This mid-range theory maps kernel theory *causes* of increased trustworthiness of information—typically information from specific sources—to specific information types available in a networked technology, such as e-relationship information. The theory further maps kernel theory *effects*—increased trust in information supplied—to technology artifact effects, such as increased usage of information supplied by recommender or other technological systems. By these mappings, the researchers have created from broad, technology-free theory, a design-related mid-range theory that can explain *why* a designed technological artifact that makes use of certain types of information for advising user choice has the *effect* of increased believability. Note that this mid-range theory, a DREPT as we have defined it, has the characteristic we suggest for such theory of being broad enough to directly assist in the design of multiple classes of artifacts, such as online advertising systems or dating services such as e-Harmony in addition to assisting in the design of recommender systems. We give two more detailed examples of the translation of kernel theory to DREPT in the last third of this chapter.

* This example draws heavily from an actual example of kernel theory refinement in Arazy et al. (2010).

Extending Knowledge Capture in DSR-IS: Alternative Approaches

Gregor and Jones (2007), working from seminal discussions of theory and design that have taken place over 2,500 years, revisit Walls et al.'s ISDT definition. Their analysis provides both a valuable "unpacking" of the theory components of Table 4.2 and suggests extensions, such as *artifact mutability* and *justificatory knowledge*, which bring the Walls et al. (2004) definition of ISDT closer to traditional conceptions of *descriptive* or *explicating* theory as contrasted to its current highly *prescriptive* form. We view Gregor and Jones (2007) as a call to DSR-IS researchers to capture more of the knowledge generated in a DSR-IS effort than is possible in the Walls et al. (1992, 2004) definition of an ISDT. This is precisely our position; certainly our efforts are complementary to Gregor and Jones (2007).

However, while Gregor and Jones (2007) propose extending knowledge capture by extending the definition of an ISDT, we propose keeping the Walls et al. definition of ISDT as almost purely prescriptive and propose extending knowledge capture with the addition of DREPT. We justify this on several grounds, first with the observation that the ISDT of Walls et al. (1992; 2004) is currently more familiar to the majority of DSR-IS practitioners (March and Storey 2008) and since that ISDT definition guided many past exemplars of DSR-IS (Kasper 1996; Markus et al. 2002; Adomavicius et al. 2008; Pries-Heje and Baskerville 2008), it allows us to make highly informative cross-study comparisons. Second, we propose that two distinct modes of knowledge capture intended for quite different purposes—a prescriptive ISDT to capture low-level (construction) design science knowledge and DREPT to capture artifact-relevant explanatory-predictive knowledge—are more comprehensible and do not overburden the already "busy," multifaceted Walls et al. information representation. Baskerville and Pries-Heje (2010) directly question the "layering of complexity" in expressions of design theory like that in Walls et al. (2004) ISDT. Indeed, Walls et al. in their 2004 reflection on ISDT (cited in Gregor and Jones 2007) explore the possibility that the ISDT as they originally conceived it (Table 4.2) was too unwieldy and ad hoc to be widely accepted and used.

The difference between the Gregor and Jones (2007) approach and our approach to capturing design science research knowledge is probably less than what may appear initially. The Gregor and Jones (2007) discussion of the *justificatory knowledge* component they propose to add to ISDT (pp. 327–328) includes much of the explanatory knowledge we propose be captured in DREPT. They describe it as "... knowledge of how material objects behave so as judge their capabilities for a design," and liken it to Walls et al.'s *kernel theories*. However, for Gregor and Jones, beyond merely being referenced, these theories should be logically linked, as much as is possible, to designed attributes via a discussion of why the artifact functions as it does. It is evident from their discussion that

we and Gregor and Jones (2007) share a strongly held belief that the knowledge presented about a DSR-IS project should explain *how* and *why* the design works in addition to explaining *how* to replicate it. Our DREPT may simply be a more formal version of their *justificatory knowledge* ISDT component. However, as we will show, the formality of DREPT is useful in our framework for theorizing in DSR-IS.

Arazy et al. (2010) suggest an approach to theory development in DSR-IS that has parallels to but does not duplicate our framework. They share our belief (as do Gregor and Jones) that kernel theories are at such a high level of abstraction that their relationship to design and suggestions for design are frequently difficult to discern. Further, they share our understanding that the Walls et al. (1992, 2004) ISDT framework is inadequate to explicate design-related knowledge from kernel theories. To bridge the gap, to effect the linkage between kernel theories and design, Arazy et al. propose the development of what they term *applied theories*. Applied theories, in their understanding, are derived from kernel theories but address two major issues in linking kernel theory and design. First, the narrow scope (typically) of kernel theories frequently necessitates the use of multiple kernel theory frameworks to explain a single design feature. Second, the granularity of constructs in kernel theories is (frequently) inappropriate to design because the constructs do not map easily to design goals. Arazy et al. clarify this by saying "Although we cannot expect to find a direct one-to-one mapping (between constructs and design goals) the correspondence should be clear."

To address these problems, Arazy et al.'s applied theory is constructed by first identifying significant, potentially improvable facets of a problem-solving artifact. Second, themes or unifying factors from prior (high level theoretical) research that grounds each of the artifacts' facets are determined. The factors then become the constructs for the applied theory. Since the factors were identified from theory frameworks chosen for their relevance to artifact facets, they are at the appropriate granularity and map easily to artifact goals. The constructs of the original kernel theory frameworks are now the internal structure of each *applied theory* factor and generate specific, testable hypotheses concerning the applied framework.

The primary similarity between the theory refinement approach set out in Arazy et al. (2010) and the one developed in this chapter lies in the *explicit attempt to strongly link kernel theory constructs with design facets*. The approaches differ in that: (1) the applied theory of Arazy et al. remains in the area or discipline of the kernel theories from which it was derived, whereas the DREPT we propose is firmly in the design domain; (2) their approach is by definition a priori; it takes place prior to any design effort while our framework accommodates both a priori theorizing and reflective, inductive, after the fact theory development; (3) the multiple perspectives developed for our framework give it a more solid grounding in design science and an extended rationale; (4) Arazy et al.

(2010) limit themselves to "applied *behavioral* theory"* whereas our framework explicitly accommodates kernel theories from behavioral, physical (natural), or design sciences.

Structure of the Remainder of the Chapter

In the next section, we present the notion of mid-range theory from other fields of study and then elaborate on the nature and value of DREPT and ISDT from the perspective of Gregor's (2006) taxonomy of IS theory (a typological view of our framework). We then present an epistemological perspective of the framework and relate it to specific activities, in the design science research cycle. Throughout these discussions, abstract points are related to aspects of concrete projects (Vaishnavi and Kuechler 2004/13; Kuechler and Vaishnavi 2008b; Kasper 1996). To demonstrate the utility of the framework, we extend several published examples of DSR-IS with proposals for mid-range theories that derive logically from the constructs in the kernel theories and the design theories of the examples. The place of theory development in DSR-IS (and in IS in general) is still debated among DSR-IS researchers, and in the conclusion of the chapter we briefly discuss some of the cultural issues that figure in the debate, in addition to summarizing our contribution.

Mid-Range Theory in DSR-IS

Thomas Merton, the influential social scientist who introduced the concept of mid-range theory, gave the following definition for mid-range theories: "... theories that lie between the minor but necessary working hypotheses that evolve in abundance during day-to-day research and the all-inclusive systematic efforts to develop a unified theory..." (Merton 1968). Gregor (2006) in her exposition of theory in IS uses the degree of "generalization" of a theory as one taxonomic principle; mid-range theories have the characteristics of being "moderately abstract and limited in scope." Gregor also notes that one of the characteristics of mid-range theory, highly valued in all sciences, is that it easily leads to testable hypotheses. This is true for both ISDT and DREPT in their respective realms of artifact construction and artifact-effect understanding.

Unlike the management, medical, sociology, and even engineering literatures where mid-range theories are frequently specifically titled as such (i.e., "A mid-range theory of _____"), a search through IS literature databases reveals only two papers that explicitly present their findings as mid-range theory: Nelson et al. (2000) and

* We feel the approach set out in Arazy et al. (2010) is quite general even though they limit themselves in the paper to "applied *behavioral* theories."

Kuechler and Vaishnavi (2008b). However, mid-range theories are common in IS. In fact, though we cannot digress to do so here, a logical case can be made that most IS theories* are mid-range since IS is an applied discipline with a history of drawing from more fundamental disciplines. An example familiar to most IS researchers is the theory of cognitive fit (Shaft and Vessey 2006). Cognitive fit theory is essentially the specialization to the technology domain of aspects of multiple theories from cognitive science. Its constructs are constrained and specialized relative to those of the originating or suggesting theories and because cognitive fit is constrained to the technology domain, specialized claims concerning that domain can be proposed. The translation/specialization of constructs from a general theory to a more tightly scoped domain is one key principle of our framework for theory construction.

Most discussions about mid-range theory also make a distinction between substantive theory and formal theory (Gregor 2006; Bourgeois 1979; Merton 1957). Substantive theory is "developed for a specific area of inquiry, such as delinquent gangs, strikes, divorce..." (Gregor 2006). In DSR-IS, substantive theory could be induced from performance data on an artifact(s) when it is operated in a specific context. Formal theory, in contrast, has explanatory power across specific areas and operates in that sense at a higher level of abstraction. DREPT, as we conceive it (Figure 4.1), is formal theory in this sense.

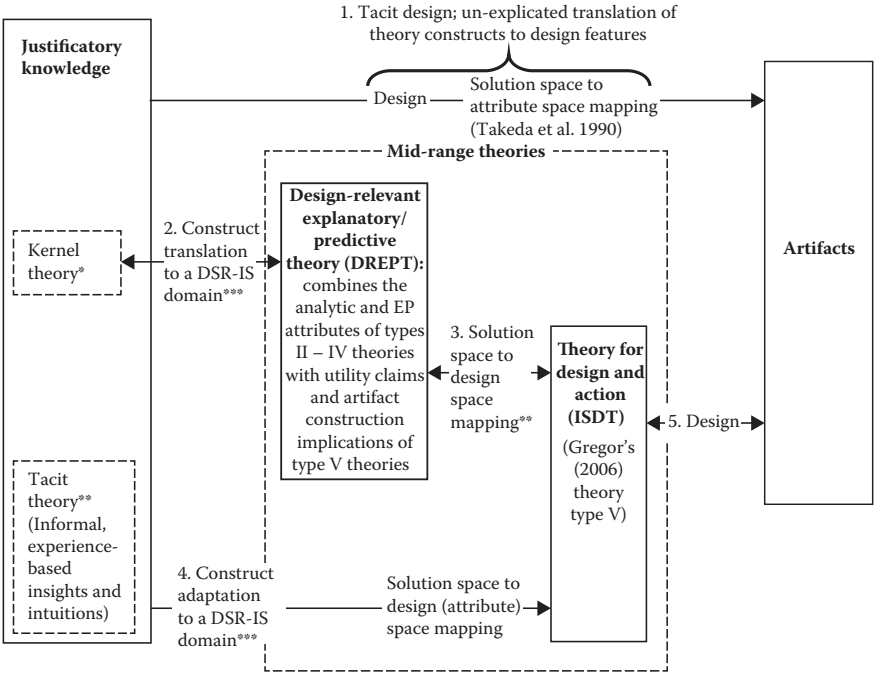
In some early discussions of mid-range theory from qualitative fields (Bourgeois 1979), mid-range theory building was suggested to be an inductive exercise: induction from field data gave substantive theory; induction and pattern matching from substantive theories yielded formal theory. However, in many fields—management, medicine, and engineering, for example—mid-range theory building has a deductive character, moving explicitly from a general explanatory/predictive theory to a specialized, environment-constrained mid-range theory expression (Nolan and Grant 1992; Hitt and Ireland 1994; Stone and McKenry 1998). Both deductively derived and inductively derived mid-range theories in all fields have the function of "linking" conceptual levels (Raab and Goodyear 1984). However, deductively derived mid-range theory typically has a distinct "starting point" in a broadly based theory of a certain phenomenon; that general theory is then made more specific to accommodate empirical data taken from a sub-range of the phenomenon covered by the general theory. Our framework specifically accommodates this type of theory building in DSR-IS where kernel theories provide the general constructs.

* Note the distinction between theories used in IS and IS theories. The theories most widely used for research in IS are from outside the IS domain—see http://istheory.byu.edu/wiki/Main_Page (last accessed on January 29, 2015). IS theories are those developed specifically for use in IS.

Typological Perspective of the Framework

As discussed in the introduction, the difference between ISDT and DREPT is in essence the difference between *how* to construct an artifact (ISDT) and *how/why* the artifact features have the desired effects (DREPT). The following discussion illustrates those differences.

Figure 4.2 provides a typological perspective of the framework essentials and illustrates the relationships between kernel theories, ISDT, and DREPT within Gregor's (2006) typology of IS theory. As in Figure 4.1, the arrows in Figure 4.2 represent logical progression—from highly abstract notions through their progressive



Kernel Theory: Social, mathematical, and design science theories as well as natural science (e.g., physics, psychology) theories

Legend: * Includes Gregor's (2006) theory types —II: Theory for explaining; III: Theory for predicting; IV: Theory for explaining and predicting (EP theory).

** Includes Informal *Experiential insight and Intuitions into a problem domain.*

*** The logical and cognitive processes used in these *theory refining* transitions are: reasoning by analogy; abduction; deduction; triangulation of perspectives . See Appendix 4C–*Theory Building Techniques in Design Science Research.*

Figure 4.2 Relationships between Gregor's (2006) theory types, DREPT, ISDT, and DSR-IS artifacts. (Adapted from Gregor, S., *MIS Quarterly* 30(3): 611–642, 2006; Venable, J., *DESRIST*, 2006.)

concretization to the physical artifact itself.* On the left of Figure 4.2 are theory types, II through IV, which are not constrained to design and action; they are theory as it is traditionally understood in the physical and social sciences. To these we have added an additional theory type—tacit theory—for completeness in the DSR-IS context, insights or evidence/experience-based justifications for pursuing a novel design. This theory type is informal and is frequently not explicitly stated but is very important to DSR-IS in that such theories provide design science research with the ability to explore areas where formal theory is sparse or non-existent.† The construction and operation of an artifact designed with tacit grounding yields data on which substantive theory can be based, allowing formal understanding to be bootstrapped from field-based evidence and intuitions.

At the far right of Figure 4.2 lie the most concrete results of a DSR-IS project, the functioning, testable, observable artifact. Between the more abstract theory types on the left and the artifact are the two types of mid-range DSR-IS theories that make explicit the knowledge that is implicit in the artifact. The first type of mid-range theory is design theory (ISDT), which is also termed by Gregor (2006) as type V: "theory for design and action." In this chapter we assume that type V theory has the expression shown in Table 4.2, which is described in detail in Walls et al. (2004). Venable (2006) makes a strong case that in addition to design information, design theory (ISDT) necessarily makes utility claims; these are of the form: if you construct an artifact according to *this design specification* and follow *this design process*, *this useful result* will ensue.

The second type of information systems design theory in Figure 4.2 is what we have termed DREPT. As implied by the figure, DREPT constructs are type II–IV theory constructs constrained to (specialized for) the design domain. They too make utility claims, but at a more abstract level than type V theory. The utility claim of a DREPT is: *this useful effect* is realized by *these lower level phenomena*, and so any artifact that implements these phenomena will achieve this effect. We are now in a position to more formally define DREPT. As implied in the earlier discussion, we envision DREPT in DSR-IS *not* as generalized type V theory (theory for action) but rather as:

(a) Type II–IV (explanatory/predictive) theory (using Gregor's typology)
(b) Derived from a highly abstract covering theory (kernel theory) that originated in a non-design domain or tacit theory
(c) That in which the kernel theory constructs have been translated into a technology domain

* We wish to emphasize that the progressive concretization of theory constructs that is diagrammed in Figure 4.2 does not represent the workflow or activity flow of a DSR-IS project. DSR-IS methodologies instead follow roughly the flow of Figure 4.4. See also Vaishnavi and Kuechler (2008b) and Peffers et al. (2007). We are indebted to Shirley Gregor (personal communication) for helping us to clarify this difference.

† Hevner et al. (2004, p. 99) state: "the existing knowledge base is often insufficient for design purposes and designers must rely on intuition, experience and trial-and-error methods."

The translation from kernel theory dependent variable (DV)/independent variable (IV) to DREPT theory DV/IV is covered in the following and several examples are given in a subsequent section of this chapter. Here, it is important to note that with the DV and IV of the theory statement clearly in the technology domain, DREPT comes very close to having an action-implication aspect per Gregor's (2006) type V theory.* That is, a design science researcher exploring the DV set forth in a DREPT can much more readily make associations between the DV and technology artifact design parameters than when working from kernel theory.

Epistemological Perspective of the Framework

The relationship of theory levels to the semantics of each level and to the iterative practice of DSR-IS is illustrated in Figure 4.3. Figure 4.3 draws heavily from Goldkuhl's (2004) epistemological examination of knowledge generated in the course of DSR-IS.

We interpret the heading "Explanatory statement" in Figure 4.3 to include Gregor's (2006) theory types II, III, and IV: explanation, prediction, and explanation/prediction, respectively. The heading "Prescriptive statement" corresponds to Gregor's (2006) type V theory for "design and action." The shaded boxes are our addition to relate epistemological statements to the DSR-IS terminology we use throughout this chapter. The dashed arrows also are our additions to illustrate relationships between knowledge levels and DSR-IS activities.

Figure 4.4 illustrates the reasoning that takes place in an idealized design science research cycle. New knowledge production is indicated in Figure 4.4 by the arrows labeled circumscription and design science knowledge. The circumscription process is especially important to understanding DSR-IS because it generates understanding that could only be gained from the specific act of construction. Circumscription is a formal logical method (McCarthy 1980) that assumes that every fragment of knowledge is valid only in certain situations, and validity can frequently not be predicted from theoretical considerations in advance. The

* An explanatory/predictive theory (Gregor's types II–IV) traditionally has the form: IF A (B, C, . . .) THEN D (E, F, . . .) (Lee and Hubona 2009). The output of design science research (following Bunge (1984)) is a technological rule: a chunk of general knowledge, linking an intervention or artifact with a desired outcome of performance in a certain field of application (van Aken 2004). In DSR-IS, we term a specific format for these technological rules as ISDT. The logical format of this technological rule is: "IF YOU WANT TO ACHIEVE Y IN SITUATION Z, THEN SOMETHING LIKE ACTION X WILL HELP." van Aken continues, "'Something like action X' means that the prescription is to be used as a design exemplar... The indeterminate nature of a heuristic technological rule makes it impossible to prove its effects conclusively, but it can be tested in context, which in turn can lead to sufficient supporting evidence."

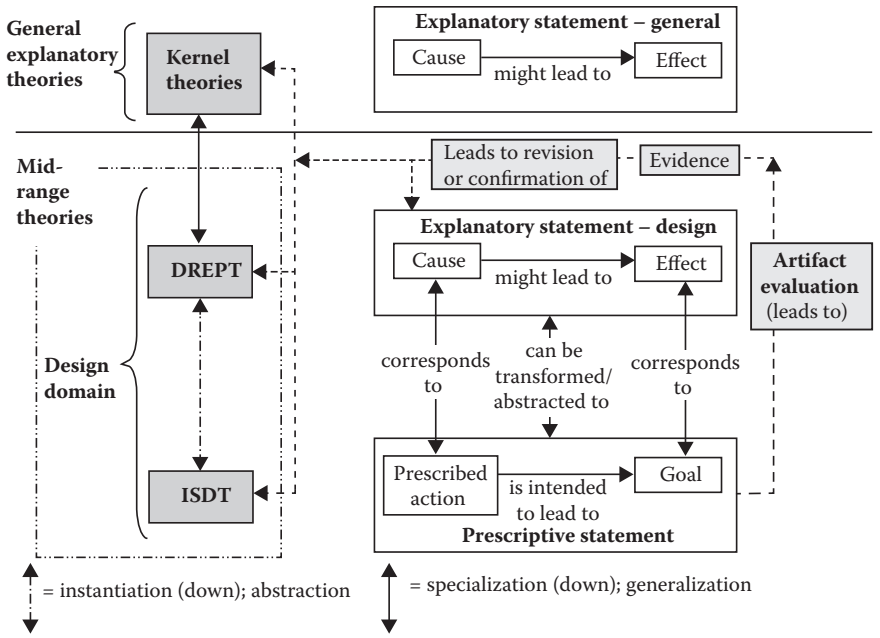


Figure 4.3 Relationships between Kernel Theory, Mid-Range Theory, Design Theory, and DSR-IS. (Modified from Goldkuhl, G., *Journal of Information Technology Theory and Application* 6(2): 59–72, 2004.)

knowledge has to be used—in this case as part of a working design—in order to clarify the implications of the theory in a given circumstance. This is not due to a misunderstanding of the theory, but due to the necessarily incomplete nature of any knowledge base. The design process, when interrupted and forced back into an earlier phase in this way, contributes valuable constraint knowledge to the understanding of the always-incomplete-theories that abductively motivated the original design.

Figure 4.3 is especially helpful in grounding levels of theory in DSR-IS project activities or phases that are illustrated in Figure 4.4. In a DSR-IS project, the overarching context is one of a business problem situation (Hevner et al. 2004). The <goal> of the project (ISDT level in Figure 4.3) is typically the development of a technological solution to all or an aspect of the problem. The problem and potential solution are set out, at least in functional terms, in the *awareness of problem* phase (Figure 4.4). Following awareness of problem, possible solutions to the problem—ways of achieving the goal—are researched and preliminarily evaluated during a *suggestion* phase (Vaishnavi and Kuechler 2004/13; Peffers et al. 2007). General explanatory statements of the form "<cause> might lead to <effect>" (see Figure 4.3) are derived from kernel theories; the <cause> in the explanatory statement suggests, by some train of reasoning, a <prescribed action> that might have an ameliorating

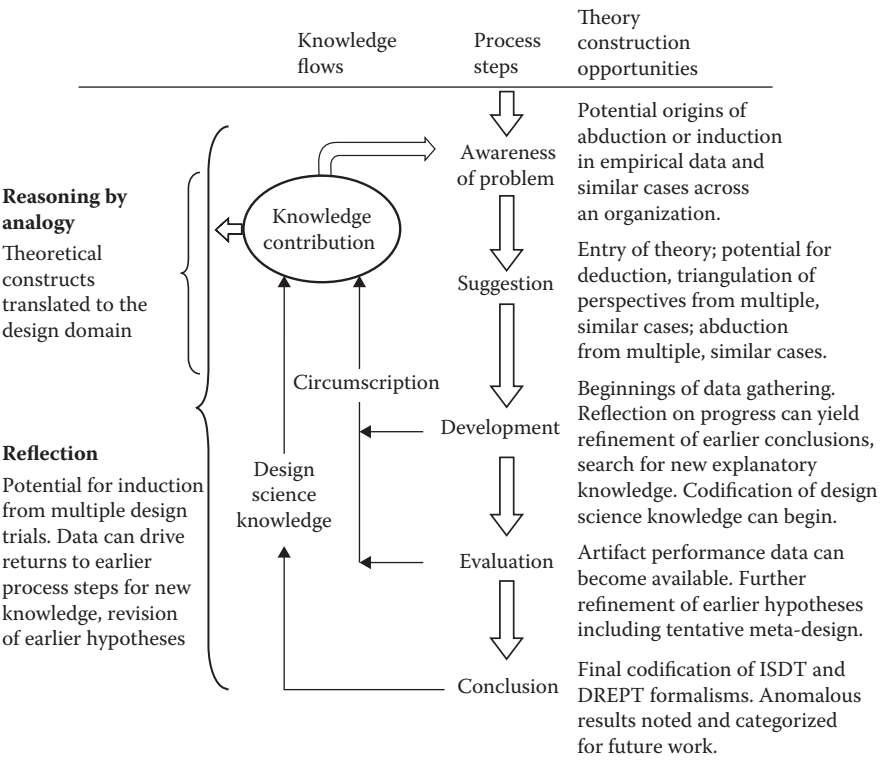


Figure 4.4 Reasoning in the design science research cycle.

effect on the problem situation. The cognitive mechanism we propose, that is most used during the conceptual translation from the theoretical domains to the design domain (the solid arrow in Figure 4.3 representing *specialization/generalization*), is analogical reasoning (Keane 1997; Gentner 1983); we discuss this further as a theory development technique in Appendix 4C. The same specialization arrow in Figure 4.3 marks the concretization* of DVs and IVs from kernel theory to artifact features and effects in DREPT.

In the development phase (Figure 4.4), DSR-IS project activity now becomes more concrete and specific; reasoning has moved from a high-level solution search to a design problem: what artifact might produce the <prescribed action> by what means (what *design features*) and thus help effect the <goal>? This reasoning is

* As constructs are specialized they are constrained by the context of the domain and become more concrete. For example, an explanatory kernel theory from psychology might contain the construct: "the number of visual field elements to which the subject can attend." When moved into the domain of the design of an information presentation screen, the construct may become "the number of screen icons to which the subject can attend."

represented by the dashed arrow in the left pane of Figure 4.3 labeled *instantiation/abstraction* and by the two arrows from *explanatory statements* to *prescriptive statements* in the right pane. During an *implementation phase* of the project (Vaishnavi and Kuechler 2004/13; Peffers et al. 2007) tentative solution artifacts are designed and constructed; this phase is frequently advised by another set of (design) kernel theories, which ground the implementation of similar artifacts. Validation of the artifact generates information that is used to assess the correctness of the entire reasoning/circumscription chain—the dashed lines and the heavy arrows of Figure 4.3—completing a DSR-IS suggestion-construction-evaluation cycle (Nunamaker et al. 1991; March and Smith 1995; Hevner et al. 2004). A DSR-IS project typically consists of many such cycles. Data on the effectiveness of the solution artifact—"evidence" in Figure 4.3—may cause the theoretical statements on the left hand side of Figure 4.3 to be revised or even abandoned and replaced by a new derivation from a new kernel theory(s).

The chain of reasoning between explanatory theory and the design domain, the mapping between kernel theories and design theories, between solution space (kernel theories) and attribute feature space (design—ISDT) is frequently *not* obvious; yet following the traditional method of DSR-IS, it is never explicated or even consciously noted. With our additions to Figure 4.3, it is clear that this logical bridge between explanation and prescription must always exist and if explicated can be captured in DREPT.

Theory Construction in DSR-IS: Two Published Examples

In the prior two sections, we have presented a typological perspective and an epistemological perspective on our DSR-IS theory building framework. We first discussed the types of theory pertinent to DSR-IS and the relationships between the DV and IVs of the different levels of theory. We then used an epistemological approach to explore the semantics of the different types of theory and related levels of theory to the information flows to and from different phases in the design science research cycle. In the following discussion, we apply our framework for theory development to two published examples of DSR-IS. We fully develop a DREPT for both papers to illustrate the use of the theory development framework and to demonstrate the potential value of this level of theory for DSR-IS. The DREPTs we propose are based on the constructs set out in the respective authors' analyses of their kernel theories and the statements of their ISDTs. These DREPTs capture plausible chains of reasoning from the kernel constructs—the *explanatory statements* of Figure 4.3—to the design theory injunctions—the *prescriptive statements* of Figure 4.3. An ISDT is explicitly developed in each of our published examples and after developing the DREPT we show how each ISDT constitutes a concretization of DV and IVs as predicted by our framework. Since DREPT

is explanatory theory, it generates hypotheses that can be tested by constructing artifacts according to the hypotheses and then evaluating the artifacts. We propose that every DREPT include testable hypotheses to aid future researchers, just as evaluation and validation are included in the Walls et al. (1992, 2004) ISDT formalism (Table 4.3). We have derived several testable hypotheses for each of our examples.

The tabular approach illustrated in Table 4.3 makes the specialization of theory constructs obvious and is used to concretely illustrate both DREPT and ISDT development for our two published examples. Note that Table 4.3 combines *analogical reasoning* and *deduction* (see Appendix 4C).

Kasper 1996

Our first DREPT "retrodiction" example is based on Kasper (1996), which was one of the first IS design science research efforts to explicitly develop an ISDT; the design theory specified design criteria for a decision support system (DSS) exhibiting improved calibration (roughly: the ratio of confidence in a decision to its correctness). Kasper does a very thorough job of setting forth the kernel theories from

Table 4.3 Logical Form and Semantics for Kernel to Design-related Explanatory/Predictive Theory (DREPT) Mapping

Non-Information System Kernel Construct/ Proposition	*Mapping*	*DREPT Construct/ Proposition*	*Semantics*
X (construct)	➔	Y (construct)	Construct (or concept) X from outside IS maps (is analogous) to DREPT construct Y
B (action)	➔	D (action)	Action B from a descriptive theory outside IS is analogous to IS artifact-achievable action D
C (general effect)	➔	E (artifact-induced effect)	An effect described in a theory from outside IS is analogous to the more restricted effect caused by the use of the designed artifact
B acting on X causes C	➔	Do D to Y to get result E	Thus, since performing B on X causes general effect C, then an artifact performing activity D on Y will yield effect E

which his ISDT derives. The kernel theories derive from three streams, the first two from psychology and the third from behavioral decision making:

1. Mental representation (mental models)
2. Problem solving
3. Calibration in decision making

Since no artifact is constructed and validated in the course of this work, there can be no demonstration of induction from artifact evaluation data for theory development. However, Kasper explicitly uses several of the methods we have proposed for theory development in Appendix 4C and we have attempted a hermeneutic extension of the narrative ratiocination of the paper to a "likely" DREPT.

For both examples, our hermeneutic proceeded as follows:

- We carefully read the paper itself.
- We carefully read the kernel theory papers.
- We identified constructs and propositions in the kernel theories.
- We identified constructs and proposition in the ISDT.
- We attempted to enter into the author's logic as he described what to him are the most salient features of the ISDT and the kernel theories.
- Using analogical reasoning, abduction, deduction, triangulation of perspectives, or some combination of these (see Appendix 4C), we set out probable mappings from kernel constructs and propositions to ISDT constructs and propositions.
- We reconstructed the logic that justifies the mappings.

Kernel Theory Constructs and Propositions

From problem solving and behavioral decision making literature (constructs are in bold):

- As the proportion of **inference** (conscious, abstract reasoning) in one's **mental representation** of a problem increases, the likelihood of **miscalibration** increases (Waggenaar 1988).
- The proportion of inference to **memory** (direct recall of information) used to formulate a mental representation is one determinant of **problem novelty**. The more novel the problem the more inference is required (Kaufmann 1985).

From problem solving and mental representation literature:

- The **locus of symbolic representation** of a problem shifts from linguistic to visual to exploratory as problem novelty increases (see the explanation in Appendix 4B) (Kaufmann 1985).

ISDT Constructs and Propositions

- A DSS exhibits design properties of **expressiveness**, **visibility** and **inquirability** (Davis and Kottermann 1994).
- Design method: "... the locus of the DSS design process needed to produce a specific DSS whose users are perfectly calibrated varies with problem novelty from **expressiveness** to **visibility** to **inquirability**" (Kasper 1996, p. 226).

Two different theories from psychology are used to give different perspectives on the construct of DSS calibration. Waggenaar (1988) proposes a relationship between abstraction in mental representation and miscalibration. Kaufmann (1985) proposes the abstraction level of a mental representation of a problem increases with problem novelty. Kasper's interpretation of this diagram in terms of DSS attributes is explained in Appendix 4B. A third perspective empirically investigates the effects of specific DSS design features (Davis and Kottermann 1994). The result of this *triangulation of perspectives* (see Appendix 4C) on the construct of calibration is the ISDT for a DSS exhibiting improved calibration.

Note the very liberal use of analogical reasoning, the mechanism that proposes similarity between the constructs in different domains, in this case between kernel theories and artifact design attributes. It is quite apparent (on *reflection*) that what Kasper has done is map expressiveness to linguistic mental attributes of a problem, visibility to visual mental attributes of a problem, and inquirability to what Kauffman in his kernel theory terms the exploratory aspect of a mental problem representation. However, it is misleading to construe the surface apparentness as obviousness. In point of fact there is *no obvious valid reason* to suppose the ISDT concept of *expressiveness*—which is an aspect of a computer artifact interface—is related in any way to the construct *linguistic representation* from Kauffman's kernel theory—which is an aspect of an internal mental representation of a problem. The same is true for the other two mappings; they are hard-won insights on Kasper's part, highly unlikely without the serendipitous confluence of the three kernel theory papers used in the triangulation of perspectives.

To see the tenuousness of the analogy, the reader is invited to scan Kauffman (1985—one of the three kernel theory references for Kasper's paper) and try to imagine themselves making the linkage between the concepts in that paper and any aspect of DSS without first having seen the analogy in Kasper's paper. Note that Kauffman (1985) is technology neutral and makes reference to no artifact whatsoever. In fact, the reasoning necessary to justify the mappings and to justify the manipulation of ISDT constructs toward the goal of improved calibration constitutes a sophisticated DREPT, which, once explicitly set forth and validated, is valuable far beyond the DSS context. The mappings of the DREPT, a *theory of computer-mediated problem representation effectiveness*, their proposed (testable) justification, and a narrative statement of the theory are shown in Table 4.4.

Table 4.4 Theory of Computer-Mediated Problem Representation Effectiveness

Kernel Construct/ Proposition		*IS Mid-Range Construct/ Proposition*	*Semantics*
Linguistic representation	➔	Expressiveness	This mental problem representation aspect suggests an analogous aspect in a computer system interface.
Visual imagery	➔	Visibility	As earlier.
Exploratory reasoning	➔	Inquirability	As earlier.
Problem novelty	➔	Problem novelty	Problem novelty maintains a common meaning in both kernel theories and the ISDT.
The more novel the problem the more abstract the mental representation	➔	Matching mental representations with interface analogues (above) leading to better decisions	A more verbally expressive interface matches (corresponds in some beneficial way) with the language component of a system user's mental problem representation; likewise the visual aspects of an interface correspond with the visual imagery component of a mental problem representation. The level of dialectics (the amount of conscious thought required for response) of the interface corresponds with the degree of abstraction of the mental problem representation.
Narrative Statement			
From the earlier kernel theory propositions, we infer that the more novel the problem the more the mental representation of the problem shifts from concrete—linguistic and visual representations, many drawn from memory—to abstract—reasoned relationships outside prior experience. The more exploratory reasoning (abstraction) figures in a problem representation, the greater the likelihood of miscalibration, and the greater the likelihood of ineffective decisions.			

(*Continued*)

Table 4.4 Theory of Computer-Mediated Problem Representation Effectiveness (*Continued*)

Kernel Construct/ Proposition		*IS Mid-Range Construct/ Proposition*	*Semantics*
Assisting the user with a decision is the final goal or an important intermediate goal of many decision support systems. System interfaces possess attributes of expressiveness, visibility, and inquirability. If the novelty of the decision situation is known, then *altering the system interface attributes will allow the computer interface (display/interaction) representation of the problem to match the users' internal representation and this will result in better decisions and more effective systems. Specifically, the interface should shift emphasis from expressiveness to visibility to inquirability as problem novelty increases.*			

The text in the narrative statement of the theory in Table 4.4 that we have italicized states the key assumptions of the theory. As stated, they lead easily to hypotheses testable through the construction and evaluation of artifacts that embody the propositions. We have diagrammed the construct mappings and the assumptions of the DREPT in a model that readily suggests empirical validation efforts (Figure 4.5). Two testable hypotheses taken directly from the model are as follows:

H1. When the problem represented in a DSS is highly novel, decision effectiveness will be increased by increasing the DSS interface inquirability (as that term is defined earlier).
H2. When the problem represented in a DSS is well understood (low novelty), a simple, language-based interface is optimal for decision effectiveness.

Validating the DREPT would also provide partial validation for one of its kernel theories, Kauffman's (1985) theory of symbolic mental problem representation. Modifications to the DREPT as would likely be necessary in the course of several design-build-evaluate cycles would, depending on the nature and extent of the modifications, reflect back on and propose modifications to the kernel theory. This cycle of empirical reasoning is shown in Figure 4.3 as the dashed lines leading from prescription (artifact) to evidence (validation information) back to description (kernel theories). The cycle is discussed in some length relative to another DSR-IS project in Kuechler and Vaishnavi (2008b). We note that the kernel theory validation of Kauffmann (1985) is a matter of considerable significance in this case (Kasper 1996) since Kauffmann's theory was published with little if any empirical validation. The same situation is likely in any DSR-IS effort where the kernel theories are taken from the most current literature of another field. Note also that the DREPT not only has value for researchers directly following in the area of

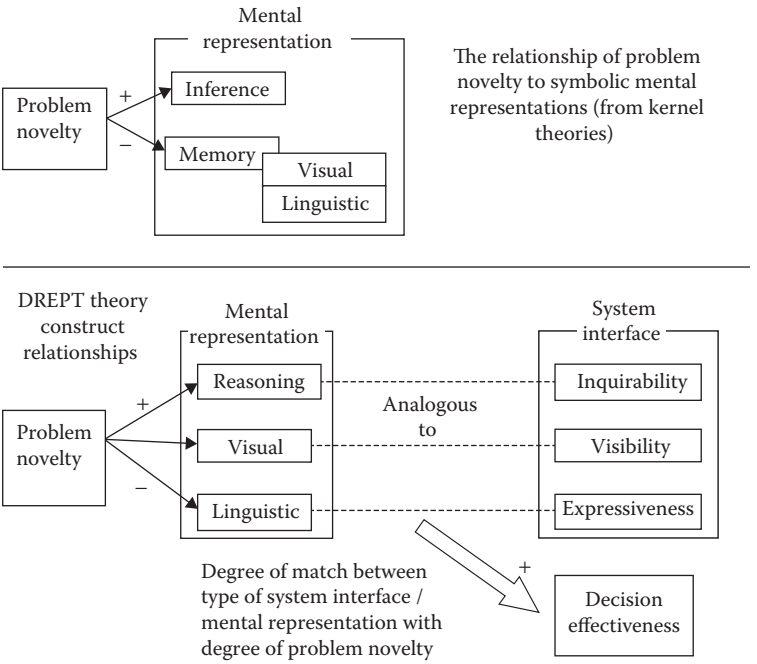


Figure 4.5 A model of the theory of computer-mediated problem representation effectiveness.

DSS calibration but also has obvious value for any IT artifact during the use of which decisions are made on novel information; search engine interfaces are a good example of this type of artifact, as are spreadsheet add-on modules for contingent data analysis (what-iffing).

Arnott 2006

The same DREPT extrapolation technique that we applied to Kasper (1996) can be applied to any published example of DSR-IS. Our second example is Arnott (2006), a paper also chosen in part due to its use of kernel theories from psychology; our familiarity with the area greatly aids in hermeneutic interpretation. For this case, the mapping from kernel theories to design theory is more obvious than that in Kasper (1996). However, we feel useful and valuable knowledge gained in the development and evaluation of the final artifact remains tacit in Arnott (2006) as written. We use three of the theory-building techniques we have proposed in Appendix 4C to construct a broad DREPT with a wider range of applications that makes this knowledge explicit, demonstrating the generality of the techniques in capturing knowledge otherwise lost in a DSR-IS project.

Arnott, like Kasper (1996), is concerned with remedying shortcomings in DSS; specifically, Arnott makes the case that the development of DSS should—in fact, must—be evolutionary, yet there exists no methodology specifically for this task. The artifact produced by Arnott's DSR-IS effort is an evolutionary DSS development methodology that focuses specifically on the types of bias that can occur in decision making and produces a DSS that minimizes those types of bias that might occur in the specific decision area for which the DSS is being produced.

Arnott (2006) explicitly uses a slightly modified version of the DSR-IS development cycle shown in Figure 4.4 and it is convenient to discuss the paper's grounding literature with reference to that illustration. For the awareness of problem phase Arnott draws from IS development literature to show that DSS should be developed during the course of an evolutionary process and that the results of DSS usage have frequently been less beneficial than anticipated. During the suggestion phase of the DSR-IS effort, Arnott turns to multiple areas outside IS for a better understanding of decision-making cognition. Psychology and cognitive science supply taxonomy of biases—predispositions to deviation from rational decision making—that could be countered by a properly designed DSS. Behavioral decision making and management science literature suggest decision process models that can help to overcome different types of bias. For Arnott, as for Kasper (1996), triangulation of perspectives (see Appendix 4C) is used; multiple theoretical frameworks with emphases different from each other and from Arnott's emphasis are dissected for relevant foci and these are "reassembled" into a coherent whole focused on Arnott's design issues. We suggest the triangulation of perspectives is inevitable whenever multiple kernel theories form the basis for the artifact design grounding.

During the development phase, Arnott designs an evolutionary DSS development methodology that focuses on identifying likely biases for the types of decisions the DSS will support, and incorporates debiasing techniques into the DSS design. This DSS development methodology is used in a single DSS development effort and during the evaluation phase, the effectiveness of the development effort is studied, guided by case analysis literature from management science (Yin 1994).

Arnott determines the result of his design effort, the evolutionary, debiasing DSS development methodology, to be conditionally successful. We agree; in addition, however, Arnott's presentation of the grounding and results of his DSR-IS project, subject to the techniques of *triangulation of perspectives* applied to his kernel theories combined with *induction* from his case observations, can be used to construct a *theory of the decision debiasing effects of participation in iterative system development*. *Induction* from the data collected in multiple phases of the case observation strengthened and added additional richness to the theorizing process. The DREPT suggests both immediate follow-up studies in the area of DSS construction methods and also more general empirical studies of IS construction processes as decision

and problem reframing techniques (in addition of course to being ways of building a system). The kernel theory propositions and case observations that underlie the theory are as follows.

Kernel Theory Constructs and Propositions

From psychology and cognitive science (constructs are in bold):

- Humans are subject to **biases** in decision making. A number of biases have been investigated and grouped into taxonomies that suggest which types of biases are most likely with which types of decisions (many surveyed in Fischoff 1982).

From cognitive science and behavioral decision making (constructs are in bold):

- The **cognitive process** of coming to understand a decision—investigating the decision and its options—can significantly reframe the decision (**reframing**) (Keren 1990).
- A **structure modifying task** wherein the user can manipulate the internal structure of a (decision) task is a known debiasing technique (Klayman and Brown 1993).

Empirical Observations of the DSR-IS Artifact in Operation (Constructs are Bolded)

- In the course of an extended, complex, **iterative DSS development** effort, the decision that was the focus of the DSS was significantly reframed.
- Both the primary **decision maker** and the **organizational personnel** who interacted with the decision maker in the context of the decision participated in the insights from the problem reframing.
- The reframing resulted from **participation in the DSS development process** more than from any specific debiasing focus or DSS development method step.

While the risks involved in generalizing from a single case are well known, the case that provided the observations on which this theory is based were obtained over many months and were partially confirmed by having been observed during multiple iterations of the development methodology. Induction was used to lift the level of abstraction for the theory from *DSS development* to *IS system development*, and our familiarity with most of Arnott's kernel theories, including those concerning IS development methods leads us to feel this abstraction is justified. The constructs and propositions of the theory are detailed in Table 4.5.

Table 4.5 Theory of the Decision Debiasing Effects of Information System (IS) Development Participation

Kernel Construct/ Proposition		*IS Mid-Range Construct/ Proposition*	*Semantics*
Bias	➔	Bias	Bias maintains a common meaning in both kernel theories and the information systems design theory—*predisposition to deviations from rational decision behavior.*
Structure modifying task	➔	IS decision support system (DSS) development process	An IS development process can be a (decision) structure modifying task.
Debiasing	➔	Reframing	Reframing can eliminate or minimize bias.
Decision maker	➔	IS system user/ customer as decision maker	In the context of the theory, the decision maker is strongly concerned with the IS system.
Modifying the structure of the decision task can minimize or eliminate bias	➔	Participating in the development of the IS (broader than the DSS only context of the Arnott's project) can result in decision task structure modification and/or substantial reframing of the problem statement	IS development efforts in which the participating domain personnel are those who make decisions based on IS output can have their understanding of the contexts of the decisions, and of the entire IS, including inputs and outputs, reframed by their participation.

Narrative Statement

From the kernel theory propositions earlier, we infer that processes that change the structure, both internal (mental) and external, of a decision task can decrease the bias associated with the decision.

Information systems development, especially iterative development efforts that involve viewing the context, inputs, and products of a system from multiple perspectives, can result in changed perceptions of the decisions that provide input to the system or are made as a result of final or intermediate outputs of the system. *Specifically, bias can be minimized for some of these decisions resulting in better systems and the effective revision of the processes providing input to or requiring output from the systems.*

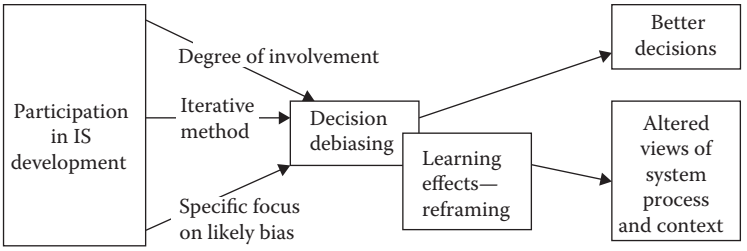


Figure 4.6 A model of the theory of the decision debiasing effects of participation in system development.

The original direction and final result of Arnott's (2006) research is certainly valuable in itself. However, a *theory of the debiasing effects of participation in system construction* that we constructed from the published results of the paper with very little hermeneutic interpretation is potentially even more valuable. It is at a significantly higher level than a *design theory for an iterative, bias focused DSS development methodology*, the artifact developed in the paper. The DREPT also has a significant explanatory component, proposing as it does that many of the learning effects seen in systems development are due to the debiasing of long-standing, unchallenged organizational decisions. The theory is also applicable to iterative process reengineering made with or without IT support. A model of the construct relationships from which DSR-IS experiments to confirm or disconfirm the theory can be readily developed is shown in Figure 4.6.

Two testable hypotheses taken directly from the model are as follows:

H1. A high degree of participation in an IS development effort will lead to more insightful views of the system process and context. Reframing of the originally stated problem situation for the system will mediate this effect.

H2. When system development participants focus on possible biases from organizational assumptions to implicit system decisions, better development decisions will result. As for H1, reframing of the originally stated problem situation for the system will mediate this effect. (Thus, development methods that explicitly analyze the organizational assumptions that underlie the situation the system addresses will yield superior results.)

We suggest the theory can also be meaningfully investigated from a study of prior published cases of IS development that provide detail on the learning effects that occurred with respect to the system and its organizational context in the course of development.

Discussion and Conclusions

We wish to point out that the emphasis of the post-hoc theory derivations we provided earlier was on exercising the DSR-IS theory development framework more than on the specific theoretical statements we derived. We realize that the nature of hermeneutic explication and the brief space we have for the explanation of our reasoning may make the specifics of our DREPT statements questionable to some. However, it should be noted that through the use of our framework we have produced logically sound theoretical statements with clearly defined DV and IVs and obvious design implications. And this is precisely the sort of testable-through-artifact-construction theory we hoped our framework would enable. An example of use of an early version of the framework in an actual DSR-IS project is developed in Kuechler and Vaishnavi (2008b). The kernel theory and DREPT propositions from that project are presented as Appendix 4A.

In the course of developing a framework for theory development in DSR-IS and illustrating its use, we have necessarily touched on a wide variety of topics in limited depth. Working from a generalized notion of mid-range theory as applied to DSR-IS, we have expanded on Gregor's typology of theory in DSR-IS to include both theory for design and action (ISDT: type V theory) *and* a design-relevant mid-range explanatory/predictive theory: DREPT. From that expansion we proposed a hierarchy of theory in DSR-IS arranged according to the level of abstraction of theoretical constructs. The framework was further explicated by an epistemological perspective that stressed the framework principle that transitioning between theory levels from kernel theory to artifact corresponds to increasing specialization (concretization) of IV and DVs. The framework by itself is useful for understanding design theories, their relations to kernel theories, and the roles each plays in the process of design. When combined with traditional theory development techniques—ways of thinking about design for design theory development (Appendix 4C)—the framework is useful in directing the construction of design theory as well.

We exercised our framework using two published examples of DSR-IS. For both examples we used our framework to formally explicate the ISDT that was developed in each paper. We additionally developed a DREPT for each to demonstrate the potential that that type of theory can have for DSR-IS. As a mid-range theory, each DREPT is generalizable to the design of any artifact utilizing the phenomena of its grounding kernel theory(s) to achieve artifact effects.

In the first of the examples, Kasper (1996), the transition from kernel theories in psychology to an ISDT for a DSS exhibiting improved calibration seemed straightforward; however, closer examination showed that there was no logical reason or published empirical basis for the mapping from kernel theory constructs to ISDT constructs. We developed post hoc a sophisticated and potentially very valuable DREPT to make that mapping logic explicit. We believe a broad gap between

kernel theory constructs and design features is not unusual and that it increases with the amount of intuition required by the researcher to reason from kernel theory to ISDT.

In our second example, Arnott (2006), the gap from kernel theories to ISDT was substantially less; the mapping from kernel theory constructs to ISDT constructs was more transparent. However, by combining the techniques of *induction* and *triangulation of perspectives* we were able to develop a DREPT that made explicit the substantial amount of information that was produced in Arnott's DSR-IS project yet remained tacit. By reflecting on the theories that underlay the artifact and by using *induction* on the data produced by the artifact evaluation effort, we were able to propose a DREPT with very broad application—almost any IS development project—and correspondingly significant potential value.

As mentioned previously, this chapter encompasses diverse material. We believe the breadth is required to ground the novel aspects of our framework within the context of theory development in DSR-IS, which is relatively new itself. In much of the seminal DSR-IS literature theorizing beyond design models (ISDT) is conspicuously absent in descriptions of the outputs from design science research. Some of the reasons for this are historical and are described in Kuechler and Vaishnavi (2008a); significantly many prominent design science researchers in IS were trained in computer science and engineering disciplines and carried over a very pragmatic focus on the artifact-as-contribution. It is only in some of the newer publications on DSR-IS (Venable 2006; Gregor and Jones 2007; Kuechler and Vaishnavi 2008b, Kuechler et al. 2009) that explanatory/predictive theory has been mentioned as a possible contribution from a DSR-IS project. We have tried to motivate the development of DREPT within DSR-IS from two standpoints: (1) as a new but potentially valuable means of capturing design science knowledge that would otherwise remain tacit in the design artifact and process and (2) as a formalism in our DSR-IS theory development framework that helps to explain the nature of design theory and its relationship to both the artifact and kernel theories. William James once described philosophy as "an unusually stubborn attempt to think clearly and consistently." We hope to persuade the field that theory development in DSR-IS be considered not as a complication or distraction but rather as "an unusually stubborn attempt to capture knowledge clearly and consistently for the benefit of practice and cumulative research efforts."

References

Adomavicius, G., Bockstedt, J.C., Gupta, A., and Kauffman, R.J. (2008). "Making Sense of Technology Trends in the Information Technology Landscape: A Design Science Approach." *MIS Quarterly* 32(4): 779–809.

Arazy, O., Kumar, N., and Shapira, B. (2010). "A Theory-Driven Design Framework for Social Recommender Systems." *Journal of the Association for Information Systems (JAIS)* 11(9): 455–490.

Arnott, D. (2006). "Cognitive Biases and Decision Support Systems Development: A Design Science Approach." *Information Systems Journal* 16(1): 55–78.

Bacon, F. (1994/1620). *Novum Organum*, P. Urback and J. Gibson, (Eds.), Trans, Chicago, IL: Open Court. (Original work published in 1620.)

Baskerville, R. and Pries-Heje, J. (2010). "Explanatory Design Theory." *Business and Information Systems Engineering* 5: 271–282.

Bourgeois, L. (1979). "Toward a Method of Middle-Range Theorizing." *Academy of Management Review* 4(3): 443–447.

Bunge, M. (1984). "Philosophical Inputs and Outputs of Technology." In Bugliarello and D. Donne (Eds.) *History and Philosophy of Technology*. Urbana, IL: University of Illinois Press, 263–281.

Chen, P. (1976). "The Entity-Relationship Model—Toward a Unified View of Data." *ACM Transactions on Database Systems* 1(1): 9–36.

Craig, E. (Ed.) (2000). *Concise Routledge Encyclopedia of philosophy*, London, New York, NY: Routledge.

Davis, F. and Kottermann, J. (1994). "User Perceptions of Decision Support Effectiveness: Two Production Planning Experiments." *Decision Sciences* 25(1): 57–78.

Dubin, R. (1978). *Theory Building*. New York, NY: The Free Press.

Evbuowan, N., Sivaloganathan, S., and Jebb, A. (1996). *A Survey of Design Philosophies, Model Methods and Systems*. *Proceedings of the Institute of Mechanical Engineers* 210: B4, 301–320.

Feyerabend, P. (1993). *Against Method*. London, Verso.

Fischoff, B. (1982). "Debiasing." In P. Slovic and A. Tversky (Eds.), *Judgment under Uncertainty: Heuristics and Biases*, NY: Cambridge University Press, 422–444.

Gentner, D. (1983). "Structure-Mapping: A Theoretical Framework for Analogy." *Cognitive Science* 7: 155–170.

Goldkuhl, G. (2004). "Design Theories in Information Systems—A Need for Multi-Grounding." *Journal of Information Technology Theory and Application* 6(2): 59–72.

Gregor, S. (2006). "The Nature of Theory in Information Systems." *MIS Quarterly* 30(3): 611–642.

Gregor, S. and Jones, D. (2007). "The Anatomy of a Design Theory." *Journal of the Association for Information Systems* 8(5): 312–335.

Hall, D., Paradice, D., and Courtney, J. (2003). "Building a Theoretical Foundation for a Learning-Oriented Management System." *Journal of Information Technology Theory and Application* 5(2): 63–85.

Hevner, A.R., March, S.T., Jinsoo, P., and Ram, S. (2004). "Design Science in Information Systems Research." *MIS Quarterly* 28(1): 75–105.

Hitt, M. and Ireland, R. (1994). "A Mid-Range Theory of the Interactive Effects of International and Product Diversification on Innovation and Performance." *Journal of Management* 20(2): 297–326.

Jones, D. and Gregor, S. (2006). The Formulation of an Information Systems Design Theory for E-learning. *DESRIST 2006*. Claremont, CA.

Kasper, G.M. (1996). "A Theory of Decision Support System Design for User Calibration." *Information Systems Research* 7(2): 215–232.

Kaufmann, G. (1985). "A Theory of Symbolic Representation in Problem Solving." *Journal of Mental Imagery* 9(2): 51–70.

Keane, M. (1997). "What Makes an Analogy Difficult? The Effects of Order and Causal Structure in Analogical Mapping." *Journal of Experimental Psychology: Learning, Memory and Cognition* 123: 946–967.

Keren, G. (1990). Cognitive Aids and Debiasing Methods: Can Cognitive Pills Cure Cognitive Ills? In J. Fabre and M. Gonzalez (Eds.), *Cognitive Biases*, Amsterdam, North Holland: Elsevier, 523–555.

Klayman, J. and Brown, K. (1993). "Debias the ENVIRONMENT Instead of the Judge: An Alternative Approach to Reducing Error in Diagnostic (and other) Judgement." *Cognition* 49(1–2): 97–122.

Kuechler, W. and Vaishnavi, V. (2008a). "The Emergence of Design Research in Information Systems in North America." *Journal of Design Research* 7(1): 1–16.

Kuechler, B. and Vaishnavi, V. (2008b). "On Theory Development in Design Science Research: Anatomy of a Research Project." *European Journal of Information Systems* 17(5): 489–504.

Kuechler, W., Park, E., and Vaishnavi, V. (2009). Formalizing Theory Development in IS Design Science Research: Learning from Qualitative Research. *AMCIS '09*. San Francisco, CA.

Latour, B. (1987). *Science in Action: How to Follow Scientists and Engineers through Society*. Cambridge, MA: Harvard University Press.

Lee, A. and Hubona, G. (2009). "A Scientific Basis for Rigor in Information Systems Research." *MIS Quarterly* 33(2): 237–262.

March, S.T. and Smith, G.F. (1995). "Design and Natural Science Research on Information Technology." *Decision Support Systems* 15(4): 251–266.

March, S.T. and Storey, V.C. (2008). "Design Science in the Information Systems Discipline: an Introduction to the Special Issue on Design Science Research." *MIS Quarterly* 32(4): 725–730.

Markus, M.L., Majchrzak, A., and Gasser, L. (2002). "A Design Theory for Systems that Support Emergent Knowledge Processes." *MIS Quarterly* 26(3): 179–212.

McCarthy, J. (1980). "Circumscription—A Form of Non-Monotonic Reasoning," *Artificial Intelligence* 13, 27–39.

Merton, R.K. (1957). *Social Theory and Social Structure*. Glencoe, Scotland: The Free Press.

Merton, R.K. (1968). *Social Theory and Social Structure*. New York, NY: The Free Press.

Nelson, K., Nadkarni, S., Narayanan, V., and Ghods, M. (2000). "Understanding Software Operations Support Expertise: A Revealed Causal Mapping Approach." *MIS Quarterly* 24(3): 475–507.

Nolan, M. and G. Grant (1992). "Mid-Range Theory Building and the Nursing Theory-Practice Gap: A Respite Care Case Study." *Journal of Advanced Nursing* 17(2): 217–223.

Nunamaker, J., Chen, M., and Purdin, T. (1991). "Systems Development in Information Systems Research." *Journal of Management Information Systems* 7(3): 89–106.

Orlikowski, W. and Iacono, C. (2001). "Desperately Seeking the "IT" in IT Research—A Call to Theorizing the IT Artifact." *Information Systems Research* 12(2): 121–134.

Pedersen, P., Methlie, L. and Thorbjornsen, H. (2002). "Understanding Mobile Commerce End-User Adoption: A Triangulation Perspective and Suggestions for an Exploratory Service Evaluation Framework." *35th HICSS*, Big Island, HI, USA, IEEE Computer Society.

Peffers, K., Tuunanen, T., Rothenberger, M., and Chatterjee, S. (2007). "A Design Science Research Methodology for Information Systems Research." *Journal of Management Information Systems* 24(3): 45–77.

Popper, K. (1989). *Science: Conjectures and Refutations: The Growth of Scientific Knowledge*. New York, NY: Routledge.

Pries-Heje, J. and Baskerville, R. (2008). "The Design Theory Nexus." *MIS Quarterly* 32(4): 731–755.

Raab, L. and A. Goodyear (1984). "Middle-Range Theory in Archaology: A Critical Review of Origins and Applications." *American Antiquity* 49(2): 255–268.

Reymen, I., Hammer, D., Kroes, P., van Aken, J., Dorst, C., Bax, M., and Basten, T. (2006). "A Domain-Independent Descriptive Design Model and its Application to Structured Reflection on Design Processes." *Research in Engineering Design* 16: 147–173.

Sein, M., Henfridsson, O., Purao, S., Rossi, M., and Lindgren, R. (2011). "Action Design Research." *MIS Quarterly* 35(1): 35–56.

Schon, D. (1983). *The Reflective Practitioner: How Professionals Think in Action*. New York, NY: Basic Books.

Shaft, T. and I. Vessey (2006). "The Role of Cognitive Fit in the Relationship between Software Comprehension and Modification." *MIS Quarterly* 30(1): 29–55.

Stone, G. and P. McKenry (1998). "Nonresidential Father Involvement: A Test of a Mid-Range Theory." *Journal of Genetic Psychology* 159(3): 313–336.

Takeda, H., Veerkamp, P., Tomiyama, T., and Yoshikawam, H. (1990). "Modeling Design Processes." *AI Magazine* 11(4): 37–48.

Vaishnavi, V. and Kuechler, W. (2004/13). "Design Research in Information Systems," last updated October 23, 2013. URL: *http://www.desrist.org/design-research-in-information-systems/* (last accessed on January 28, 2015).

Venable, J. (2006). "The Role of Theory and Theorizing in Design Science Research." *DESRIST 2006*, Claremont, CA.

Waggenaar, W. (1988). "Calibration and the Effects of Knowledge and Reconstruction in Retrieval from Memory." *Cognition* 28(3): 277–296.

Walls, J.G., Widmeyer, G.R., and El Sawy, O.A. (1992). "Building an Information System Design Theory for Vigilant EIS." *Information Systems Research* 3(1): 36–59.

Walls, J.G., Widmeyer, G.R., and El Sawy, O.A. (2004). "Assessing Information System Design Theory in Perspective: How Useful was our 1992 Initial Rendition." *Journal of Information Technology Theory and Application* 6(2): 43–58.

Yin, R.K. (1994). *Case Study Research: Design and Methods*. Newbury Park, CA: Sage Publications.

Wikipedia-Abduction (2014). "Abductive Reasoning." URL: http://en.wikipedia.org/wiki/Abductive_reasoning (last accessed, January 28, 2015).

Appendix 4A Kernel Theory and DREPT Propositions for a DSR Project

This appendix summarizes the kernel theory and DREPT propositions for Kuechler and Vaishnavi (2008b).

Theoretical Constructs

Table 4A.1 Theoretical Constructs for Kernel and Mid-Range Theories

Construct	*Definition*
Mental model	The internal, cognitive model (in this case, of business processes) that contains the information about the model elements and their relationships
Modes of cognition	Modes of perceiving information that determine the types of information most readily acquired and the strength of relationships between information elements as mental models are formed
Surface understanding (of processes)	Understanding of the "mechanics" of process elements—flows, actors, and decisions at an algorithmic level—excluding domain or context information
Deep understanding (of processes)	Surface understanding combined with knowledge of the context in which the process operates and the interactions, actual and potential, between the process and its environment
Soft context information	Organizational, cultural, or political information about the actors or environment of a process that is difficult to capture in conventional process notations but that is frequently critical to the success of the process. In a medical informatics context, e.g., the aversion of many older MDs to information technology is one example of soft context
Narrative (sometimes termed *text*)	Information in language form
Micro-rationales	Small concise narrative segments relating process details or context not found in diagrammatic representations, usually woven into a coherent "story" about the process
Salience	In this context, the term denotes the degree of attention and significance given to different information elements of a conceptual model

Kernel Theory Propositions

From the modal cognition literature:

- The cognitive model formed from information about a situation can be made more receptive to social or "soft" information by varying the mode of information presentation from abstract-propositional (numeric) to narrative (textual).
- (Note that a proposition of exactly this form can likely *not* be found in the literature. We have presented our interpretation, which at this point is quite informed. We have taken no liberties with matters of fact, but have "repackaged" conclusions from the kernel literature to concisely state what was of interest to us. The restatement also makes it easier to follow our development from one theory level to the next.)

From the multi-media comprehension literature:

- Richer cognitive models of physical processes that demonstrate greater transfer learning (across domains) result from mixed-media presentations of the processes, that is, text + illustrations, than from text or illustrations alone.

DREPT Propositions

(A theory of grammatical element salience in conceptual modeling)

1. In systems design, a conceptual model can be used to concisely represent one or more important aspects of the system.
2. A system always operates in a context. Usually the grammar(s) for the conceptual model(s) of the system is optimized for the representation of a narrow range of system constructs. Specifically, these grammars are not well suited to representing organizational context information, especially when they are graphical in form.
3. Organizational context information can be expressed in narrative (language) form.
4. Virtually all business systems are artificial—they are designed and there are reasons called *design rationale* that describe why they are as they are. Design rationale also can be expressed in narrative form.
5. When conventional (narrowly focused) conceptual models for processes are linked in a designer's mental model to expressions of critical organizational context and design rationale, better design decisions are achievable.
6. Computer-based conceptual model design and display artifacts can be built that force attentional links between conventional conceptual model element displays and narrative information displays of organizational context and

design rationale so as to facilitate the construction in the user of the artifact of mental models that link context information with the information captured by the conventional conceptual model.

7. The strongest and most useful overall mental model (conventional conceptual model and narrative components) will be produced when the narrative components are woven into a coherent (by basic literary standards) *story* rather than presented as separate, intelligible but logically unconnected text components. (This is one of the distinguishing features between a dual grammar conceptual model and a simple annotated conceptual model graphic display).

Appendix 4B Kaufmann's Diagrammatic Representation of the Change in Modes of Mental Representation with Problem Novelty and Kasper's Interpretation of Kaufmann's Diagram in Terms of DSS Attributes

Kaufmann's (1985) research suggests that there are three mental representational modes for tasks to be performed: linguistic, visual, and abstract, possibly interlinked questioning activities. All three modes are always involved in any task representation, but as the novelty of a task or problem increases, the usefulness of the representation shifts from linguistic to visual to questioning/exploration.

Kasper (1996) very directly maps from Kaufmann's (1985) representational modes to his DSS design criteria. As the degree of novelty of a problem increases, Kasper suggests that DSS design criteria importance should shift from linguistic/expressive to visual/visibility to exploration/inquirability.

See Kasper (1996) for a visual mapping of these attributes.

Appendix 4C Theory Building Techniques in Design Science Research

This appendix presents some of the techniques of theory development we feel are most applicable to design theory construction. We cannot hope to do justice to this huge subject in an appendix; instead we briefly describe the techniques we have used in past DSR-IS projects and that we have used in the examples in this chapter.

Table 4C.1 provides an overview of "ways of thinking about design" for design theory development; for each, the name, a brief description of the technique and references to more extensive information about the technique are provided. We have termed these techniques *"ways of thinking about design" that are useful for theory development* rather than "rules of logical derivation" because design, both its practice and, necessarily, in theorizing about it, is resistant to and in some ways

Table 4C.1 "Ways of Thinking about Design" for Design Theory Development

Technique	*Description*	*References*
Deduction	Allows deriving *b* from *a* only where *b* is a formal consequence of *a*. In other words, deduction is the process of deriving the consequences of what is assumed. Given the truth of the assumptions, a valid deduction guarantees the truth of the conclusion.	Craig 2000; Wikipedia-Abduction 2014
Induction	Allows inferring *b* from *a*, where *b* does not follow necessarily from *a*. *a* might give us very good reason to accept *b*, but it does not ensure that *b*.	Craig 2000; Wikipedia-Abduction 2014
Abduction	Allows inferring *a* as an explanation of *b*. Because of this, abduction allows the precondition *a* to be inferred from the consequence *b*. In other words, *b* exists and can be observed; *a* (or *a1*+ *a2*+ …) is the most parsimonious and therefor most likely explanation of *b*.	Craig 2000; Wikipedia-Abduction 2014
Triangulation of perspectives	Creation of a novel viewpoint on a problem by extracting individual "element foci" from multiple solution approaches to similar problems and combining these into a coherent viewpoint.	Pedersen et al. 2002
Circumscription	A rule of conjecture that allows "jumping to certain conclusions." Semi-formally: "The objects that can be shown to have a certain property P by reasoning from certain facts A are [considered for a given train of reasoning to be] all the objects that satisfy P" (McCarthy 1980). Without circumscription reasoning about the real world encounters the "qualification problem" whereby an intractable number of possibilities need be considered before making a logically defensible decision.	McCarthy 1980

(*Continued*)

Table 4C.1 "Ways of Thinking about Design" for Design Theory Development (*continued*)

Technique	*Description*	*References*
Analogical reasoning	Analogical reasoning is a mode of cognition in which the similarities between new and understood concepts are compared and the comparison used to gain understanding of the new concept. Analogical reasoning is a form of inductive reasoning in that it attempts to provide understanding of what is likely to be true, rather than a deductive proof of truth (or fact).	Gentner 1983; Keane 1997
Reflection	"The action or process of thinking carefully or deeply about a particular subject." (Oxford English Dictionary online 2010) "Reflection on a design process is thus defined as a combination of reflection on the perceived design situation and reflection on the remembered design activities." (Reymen et al. 2006)	Reymen et al. 2006; Schon 1983

antithetical to the methods of traditional single-valued Aristotelian logic. In fact, some of the techniques traditionally used and even promoted in design in all fields are *logical fallacies* when dissected by the axioms of first-order predicate logic.* The methods and techniques are used (or, logicians might say, misused) because they work; by thinking about design and design science knowledge (design theory) in these ways, designers for centuries have achieved useful, effective designs and have captured design science knowledge in ways that can be effectively transmitted between designers and across generations (Latour 1987; Feyerabend 1993).

* Abduction, for example, is equivalent to the logical fallacy of affirming the consequent.

Chapter 5

On Theory Development in Design Science Research: Anatomy of a Research Project*

> Theories are practical because they allow knowledge to be accumulated in a systematic manner and this accumulated knowledge enlightens professional practice. (Gregor 2006)

Introduction

In this chapter, we describe an information systems design science research project that aims to create a (prescriptive) design theory for a class of artifacts. Several phases of the project are informed by kernel theory (frequently theory from other fields that intends to explain or predict phenomena of interest) and the project in turn refines that theory into a mid-range design science research in information systems (DSR-IS) theory (Merton 1968; Markus and Lee 2000) that is more directly applicable to information systems development. The chapter is illustrative rather than prescriptive: there are few "shoulds" or "oughts," but rather a

* Adapted from the authors' article in *European Journal on Information Systems* (*EJIS*), 17(5): October 2008, pp. 489–504.

demonstration of the productive relationship that can be developed between design science research, with its principal stress on design theory, and kernel theory. In order for the chapter to serve as an "existence proof" of the potentially close relationship between design science research and kernel theory, it must accomplish two things: First it must demonstrate the pedigree of the project as a true act of design science research; we have tried to do this without being overly pedantic. Second, it must demonstrate the relationships between mid-range DSR-IS theory, the kernel theory from which it was refined, and the research conducted in betterment of IS artifact design.

In the next section, we provide a brief overview of the variant viewpoints on the role of theory in DSR-IS. This is followed by a section that outlines an in-progress DSR-IS project and its kernel theory. It sets out details of the research design and demonstrates the potential of the research artifact for refining applicable kernel theory into mid-range DSR-IS theory. In a separate section we summarize theory development in the project to date. The chapter's conclusion abstracts from our specific research project to a general discussion of the potential of DSR-IS for theory development. Beyond that, we propose that "kernel theories" from other fields are often so narrowly derived as to be more suggestive than useful as given, and that refinement of the theory in the act of development is required to give the theory direct applicability to IS design efforts (Carroll and Kellogg 1989).

Theory in DSR-IS: What Does It Mean?

A number of positions have been stated with respect to the use and development of theory in DSR-IS. Classifying these positions is made more difficult by the different meanings attached to the term "theory" by different writers. Gregor (2006) sets forth the taxonomy of five different types of theory in use within the field of information systems: (1) theory for analyzing, (2) theory for explaining, (3) theory for predicting, (4) theory for explaining and predicting, and (5) theory for design and action. She notes, and we strongly concur, that in DSR-IS writings and discussions of theory, attributes of the types in her taxonomy are frequently blended. In fact, as Gregor states, Iivari's (1986) three category taxonomy of theory—conceptual, descriptive, and prescriptive—spans her categorization. In the hopes of simplifying matters for this chapter, we have chosen to use a two-category taxonomy, very similar to that expressed in Walls et al. (1992, 2004) and Nunamaker et al. (1991). In addition to having a long history in the DSR-IS foundational literature, the two category taxonomy we use accords well with the distinction between explanation and prescription, which is at the heart of many philosophies of design:

1. "Kernel theories" frequently originate outside the IS discipline and suggest novel techniques or approaches to IS design problems. The term and meaning are derived directly from Walls et al. (1992, 2004); many kernel theories are "natural science" or "behavioral science" theories of Gregor's (2006) "explain" and "predict" type; in this book we broaden the concept of kernel theories to also include mathematical and design sciences.
2. "Design theories" give explicit prescriptions for "how to do something" and correspond almost exactly to the "design theories" of Walls et al. (2004, 1992) and Gregor's (2006) "design and action" theory type. As discussed in Chapter 2 in the section "Outputs of Design Science Research," the theory contribution of a DSR effort is frequently partial or nascent design theory rather than a formal, fully developed information systems design theory (ISDT).

The DSR-IS project we describe in this chapter uses and *refines* kernel theory as it aims to create a design theory for a new class of artifacts. Refinement of theory in DSR-IS is somewhat unusual and a brief overview of the positions set out for the use of theory (of any type) within DSR-IS will situate our approach. Table 5.1 shows some of the influential writing on DSR-IS and the actions and uses each paper proposes for each of the two types of theories. Table 5.1 is far from complete. A fuller treatment of the literature on theory in DSR-IS might begin with Venable (2006a), Gregor and Jones (2007), and Kuechler and Vaishnavi (2008).

A majority of the papers that discuss theory in the context of DSR-IS understand *design theory* as a prescriptive statement that is a significant, perhaps the most significant, output of the research effort. Many of these papers also discuss kernel theories, but a majority of them consider this type of theory to be only advisory to the design effort. To the best of our knowledge, only Venable (2006a), Vaishnavi and Kuechler (2004/13), and Simon (1996) (in our interpretation) discuss the position taken in this book, that kernel theories can both *inform* DSR-IS efforts and can in turn be *refined and developed* by DSR-IS. Figure 5.1 (Venable 2006a) shows the relationships between DSR-IS activities and theory development that we assume to exist in the discussions of our example project.

Mid-range IS theories were not discussed in the preceding section on theory in prior DSR-IS literature because they receive no mention in that literature. Based on a search of IS literature databases, we believe this chapter (and the journal paper it is adapted from) to be one of the first discussions of mid-range theories in the context of DSR-IS. In fact, while figuring prominently in the fields of sociology (where the idea originated), health care, and management, discussion of mid-range theory seems absent from IS literature save for Nelson et al. (2000) and the editor's introduction to the issue containing that paper (Markus and Lee 2000). Merton's (1968) original description of mid-range theories is that they are explanatory theories but

Table 5.1 Kernel and Design Theories in Design Science Research in Information Systems (DSR-IS) Literature

Discussion	*Kernel Theory Conception*	*Design Theory Conception*
Nunamaker et al. 1991	Kernel theories advise design solutions; possibility of refinement or development	DSR-IS research creates design theories
Walls et al. 1992, 2004	Kernel theories advise design solutions, govern design requirements	DSR-IS research creates design theories—design theory is the primary output of DSR-IS research
March and Smith 1995	Seems to relegate kernel theory refinement to natural science. "Rather than posing theories, design scientists strive to create models, methods and implementations that are innovative and valuable"	Our interpretation is that March and Smith's use of the terms "model" and "method"—specified as desirable outputs for DSR—span the meaning of the term "prescriptive design theory," at least in the fairly narrow meaning given to ISDT in Walls et al. (1992). See the discussions of research outputs in Section 3.1 of their paper
Simon 1996	Kernel theories advise design solutions; possibility of refinement or development	DSR-IS research creates design theories; prescriptive design theories can revitalize b-schools
Orlikowski and Iacono 2001	Posed as a possible distraction to full attention to the IT artifact itself	Seem to use the term "design theory" in a broader sense than just prescriptive "models"—explanatory theories of and about design as well as theories of artifact construction
Goldkuhl 2004	Kernel theories provide theoretical grounding for the artifact (highly desirable)	"Design theory is considered as practical knowledge used to support design activities"

(*Continued*)

Table 5.1 Kernel and Design Theories in Design Science Research in Information Systems (DSR-IS) Literature (*Continued*)

Discussion	*Kernel Theory Conception*	*Design Theory Conception*
Hevner et al. 2004	"... results from reference disciplines provide foundational theories..." (p. 80). Seems to relegate foundational theory refinement to *behavioral* IS research	"Prescriptive theories" (for artifact construction) are outputs of DSR-IS (p. 77)
Vaishnavi and Kuechler 2004/13	Stress that one of the significant attributes of DSR-IS is the ability to proceed in the absence of a theoretical basis; otherwise, as Venable 2006a	Operational principles (for artifact construction) (Dasgupta 1996, Purao 2002) can emerge at multiple levels
Venable 2006a	Termed solution space and problem theories, advise IS design at multiple levels; refinement or development of theories possible and beneficial	Termed utility theories, can emerge from a DSR-IS effort at multiple levels

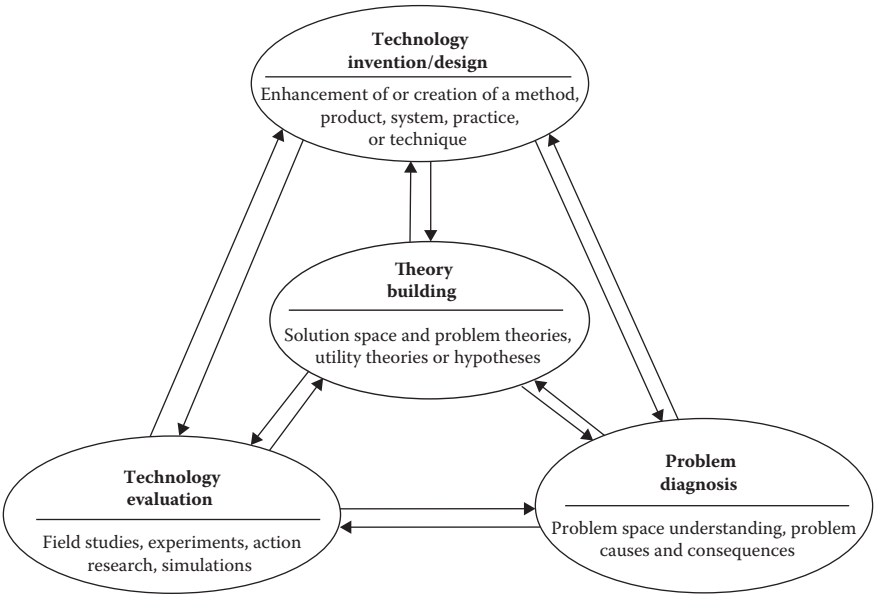


Figure 5.1 An activity framework for design science research. (From Venable, J., *Proceedings DESRIST,* 2006a.)

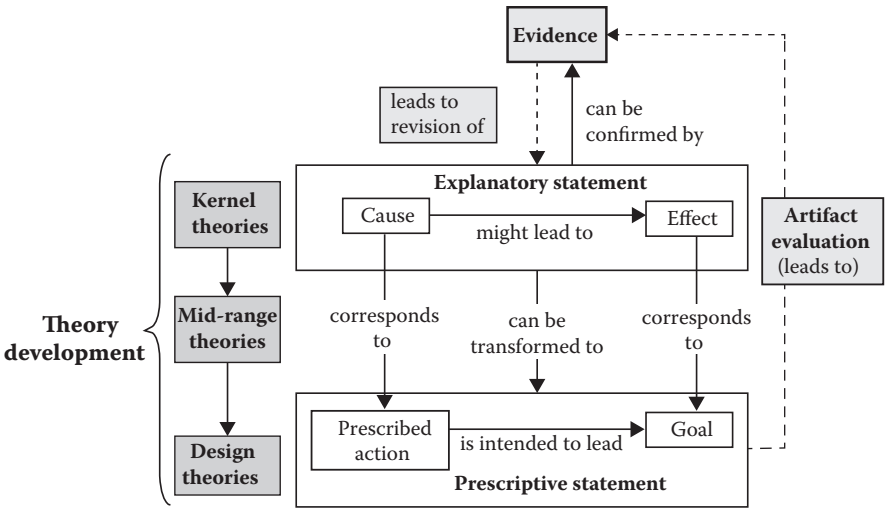


Figure 5.2 Relationships between kernel theory, mid-range theory and design theory, and the design process. (Modified from Goldkuhl, G. *Journal of Information Technology Theory and Application,* 6(2): 59–72, 2004.)

of a restricted scope and as such more readily suggesting actions for specific effects in applied fields. Gregor (2006), in a discussion of the breadth and focus of theories in IS, describes mid-range theories as leading to easily testable hypotheses. Note that kernel theories can be mid-range theories, albeit from different disciplines.

Elaboration on the relationship between design theories, kernel theories, mid-range theories, and the DSR-IS process is shown in Figure 5.2. The basis for Figure 5.2 is Goldkuhl's (2004) graphical clarification of the logical relationships between prescription and explanation in the design process. To that starting point, we have added the text highlighted in gray and the relationships specified by dotted lines. *Explanation* has been identified with *kernel theories*; note that kernel theories inform both the effect we seek in the artifact (the "goal") as well as suggesting the "prescribed action." *Prescription* has been identified with *design theories*, and we have added two relationships: (1) the loop from artifact to evidence that takes place during the evaluation of the artifact, and (2) the effect of this evidence on the explanatory statements that "can be revised to accord with" the observations or logically demonstrated behaviors of the artifact that take place during evaluation—observations that expose the theories in situ (Venable 2006b).

A final addition to the figure, *mid-range theories*, is depicted as a conceptual bridge between high-level explanatory kernel theories and highly prescriptive design theories. Through the praxis of the DSR project, new empirical knowledge and knowledge from kernel theories is translated from the kernel domain to become unique

IS theories.* The *evidence* coming from the design and evaluation of the artifact refines the kernel theories. The environment of the design evaluation more tightly scopes the original theory(s). The net result is a mid-range theory that, because of its tighter scope and additional information content, is much more easily extrapolated to design prescription than the kernel theories from which it was derived.

In the next section, we first elaborate on the phases of a design project during which the relationships shown in Figure 5.2 actually take place, and then describe the concrete design prescriptions and goals suggested by the kernel theory—by way of mid-range theory—for our project.

Theory-Refining DSR-IS Project

The activities of many design science research projects group naturally into phases such as those illustrated in Figure 5.3, which is similar to but more granular and directive in its description of project phases than Figure 5.1. However, just as in Figure 5.1, all research phases are potential opportunities for the development and refinement of kernel theory, mid-range theory, and design theory.

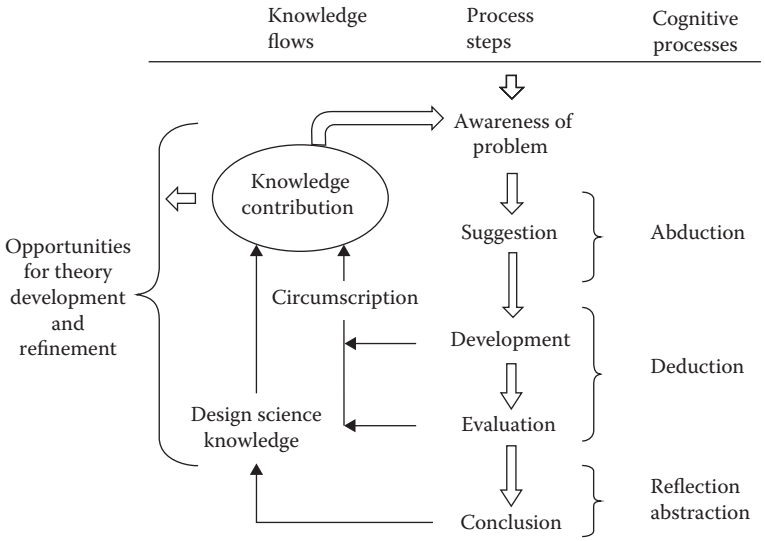


Figure 5.3 Cognition in the design science research cycle.

* The concept of design relevant explanatory/predictive theory (DREPT) is introduced in (Kuechler and Vaishnavi 2012; see Chapter 4) but has its origin in the EJIS (2008) work discussed in this chapter. The "mid-range theories" discussed in this chapter are actually DREPTs but, letting the chapter stay true to its EJIS (2008) journal source, we will not be using the term "DREPT" in the rest of this chapter.

Background: Awareness of Problem

According to guidelines in Hevner et al. (2004), a design science research project seeks a solution to a real-world problem of interest to practice. This was certainly true of our project that originated in the continued interest of the industry advisory board of one of the authors' (IS) department in business processes—specifically in courses and research to support business process (BP) design, change, and management. After reviewing several cases supplied by the advisory board, it became obvious that even though the initiation and high level design of many business processes is performed by non-IT personnel, the steps of the design process and the associated problems are very similar to those found in IS design. The problem that became the focus of our DSR-IS effort was the sub-optimal design of business processes due to the lack of incorporation of soft context information into the final designs.

Soft context information is our term for information about the operational context of a system or process that has two characteristics:

1. It is frequently social or organizational information that is difficult to capture objectively with common specification notations such as data flow diagram (DFD), business process model notation (BPMN), or unified modeling language (UML).
2. (It) "... serve(s) as selection criteria for choosing among myriads of decisions. Errors of omission (of this information) are among the most expensive and difficult to correct once the information system has been completed" (Mylopoulos et al. 1992).

We chose "soft context" information as an umbrella term for the contextual information referred to in the literature by (unfortunately) multiple terms including *context information* (Gause 2005), *soft constraints* (Stefansen and Borch 2008), *non-functional requirements* (Cysneiros et al. 2001), and *requirements perspectives* (Nissen et al. 1996). An example of soft context information from the pool of process scenarios we prepared for our artifact evaluation is given in Appendix 5A.

With further investigation, we saw that not only were the activities such as requirements gathering and project management similar in IS and BP design, but also that the tools were similar. Many BP design efforts are supported by BP design software that represents the design in a graphic notation, frequently the emerging standard: BPMN (BP modeling notation). Sub-optimal design of IS due to a lack of incorporation of soft context information is a problem that has been researched in both information systems and computer science (Mylopolous et al. 1992). Many of the approaches to solving the problem in IS/CS have focused on the use of graphic notations to represent the soft context information for the project. Possibly the most widely known form of this type of notation is the Ishikawa diagram used in multiple fields to represent quality (a decidedly soft constraint) issues. One of the most common notations in the computer science subfield of requirements engineering is

the i* model (Yu 1995; Yu and Mylopoulos 1994). An excellent example of its use in representing soft context information is given for an air traffic management case study in Maiden et al. (2005). i* is a formalization of "influence diagrams" used in many fields to represent webs of interrelated qualitative influences in an environment. Examples of influence diagrams and an example of the i* notation from Maiden et al. (2005) are given in Appendix 5B. Other notations sometimes used to represent soft context are *hierarchical AND/OR graphs* (Cysneiros et al. 2001) and graphic representations of *contribution structures* (Gotel and Finkelstein 1995); examples of these notations are also given in Appendix 5B.

None of the suggestions from research to date have been widely adopted in industry (Davies et al. 2006; Lethbridge et al. 2003), and as a glance at Appendix 5B will show, the formal notations proposed would be highly complex for most real-world processes and would require some training in first-order predicate logic to be developed or understood. This creates a formidable barrier to their use by business persons in process design. Most significantly, the creation of graphic representations of soft context information presumes the information has been previously *noted* and *understood as significant* by project analysts—an assumption that our problem statement indicates is not the case. However, prior research in the IS/CS domain did help to refine our problem statement to a design research question: How could BP modeling notations be enhanced to make soft-context information more salient and more likely to be incorporated in final BP designs?

Suggestion

In this phase of a design science research project, various approaches to the problem, informed by prior research on related issues, are worked out as "thought experiments" to explore the feasibility of each approach (Vaishnavi and Kuechler 2007, p. 20; pp. 132–133; p. 139). It was at this point that "kernel theories" entered our design process. First, we reviewed the IS research on conceptual modeling and adopted the concepts and vocabulary from earlier research on design notation (Wand and Weber 2002). Instead of speaking of process drawings, we started referring to *conceptual model scripts* expressed in a *notational grammar*. We also became familiar with research guidelines for assessing the effectiveness of different conceptual models (Parsons and Cole 2005).

Then, as we reviewed prior approaches to the problem of soft-context "leakage" from system designs we saw that all of them focused on capturing soft-context information in some form of graphic notation. Intuitively, it seemed that this effort might be misdirected. Based on 20+ years of IS industry development experience we wondered if the real problem was not the capture and representation of soft-context information—in most cases the information was available in the original requirements notes—but rather in making that information more immediately available and especially more salient to the designer. Further, as we thought through different soft-information representations of our own, it seemed that a graphic representation of soft or contextual information was the wrong approach. We began to build the

position that the highly qualitative, sometimes political, frequently ambiguous nature of soft information was best captured by textual narrative rather than graphics.

At this point, hoping to better understand why some concepts are more salient than others, we began to investigate problem-solving cognition and came upon our "kernel theory"—actually a related set of theories from cognitive, educational, and social psychology that described and explained how varying the presentation of information could enhance or diminish information salience and thus problem-solving capabilities. One of our key papers, Zukier and Pepitone (1984) describe how the "base rate problem" made famous by Tversky and Kaheneman (1981) and originally viewed as a "flaw" in human reasoning could be eliminated by reframing the problem. When the same information that people ignored when presented as numeric abstractions was presented as part of a story, the information was correctly incorporated into the solution of the problem. Another researcher exploring cognitive mechanisms involved in solving word problems effectively duplicated Zukier and Pepitone's results and showed the importance of contextual information, especially intentional information, on eliminating "framing issues" in problem solving (Kuechler and Vaishnavi 2006; Jou et al. 1996).

In consideration of these experimental results, we came to believe that a possible means to make soft-goal information more salient to designers would be to induce, by means of a novel conceptual modeling grammar(s), a mode of cognition that psychologists term "narrative thinking." The alternative mode of cognition, "propositional thinking" tends to ignore problem irregularities (such as soft information!) and has been shown to be promoted by attention to abstract information presentations such as numeric and diagrammatic representations (Zukier 1990; Zukier and Pepitone 1984). For convenience, we refer to the web of more granular theories that underpin narrative and propositional thinking as *modal cognition theory* (Zukier 1986) and we refer to the research support for this kernel theory henceforward as the "narrative versus propositional thinking" literature.

Further investigation revealed a parallel development in educational psychology that was also concerned with improving the mental models formed during the presentation of descriptive information: multi-media comprehension. This subfield of educational technology has both theoretical and empirical branches that illustrate the relation between theoretical and prescriptive statements (Goldkuhl 2004; Figure 5.2) in yet another domain. The theoretical work in this field proposes high-level explanatory statements concerning learning from computer-mediated information presentations: text combined with various graphics that illustrate the concepts contained in the text. The results of low-level experiments in this literature provided support for broad explanatory statements that confirmed the cognitive effects from the narrative versus propositional thinking literature and provided further vocabulary and high-level constructs for the project (Mayer and Jackson 2005).

In the prescriptive branch, educational technology design papers sought to transition from theoretical statements of multi-media cognition to specific techniques for the most effective presentation of different types of material—laws of rectilinear motion, for example. These papers prototyped mixed narrative and graphic presentation

techniques and evaluated the resulting cognitive models. In Seufert et al. (2007), several display techniques were used in the context of understanding the physiological effects of vitamin C. First, hyperlinked text and an illustration were displayed simultaneously. When the hyperlinks were clicked, an arrow appeared at the appropriate portion of the illustration. In a second study, four different representations of related material—text, graphs, tables, and a chemical formula—were used. Subjects could move between the presentations, but only one representation was on-screen at a time. In each case, understanding was measured by a post-session objective quiz. In Lewalter (2003), the information content was the phenomena of gravitational lensing and the presentation techniques were text and static illustration or text and animated illustration; both learning and learning strategies were assessed in this study. While not directly applicable due to the different media content and artifact intent, this literature influenced both our grammar design and the design of the presentation software.

The "kernel theories" we had adopted suggested directions for a design solution to our research problem, but having been taken from social, cognitive, and educational psychology they gave no specific prescriptions as to how the information could be used in the context of IS/BP modeling. First, the experimental results that grounded the theories were obtained in carefully controlled laboratory situations. To be useful in a working IS design, the effects shown for narrative thinking would have to be demonstrated to be robust enough to give meaningful results in a far more complex environment. Second, the modes of presentation are different from our design environment than for the prior research in narrative versus propositional thinking. Prior research used (1) narrative expression of information and (2) numeric/narrative presentation as the two treatments in its experiments. Third, the kernel literature has yet to resolve some of its theoretical conflicts. Much of the recent literature in multimedia comprehension is involved with testing the net effect of two conflicting cognitive mechanisms, each with its own experimental support: cognitive load theory and coherence formation theory (Mayer and Jackson 2005). Cognitive load theory predicts better learning from leaner presentations. Coherence formation theories predict better and deeper learning and more skill transference from richer (greater information content) presentations. The not uncommon conflict of results from grounding (kernel) literatures is still more evidence of the need to generate mid-range theory and its attendant constructs from kernel theory for DSR-IS projects.

Our design attempts to induce "narrative thinking" by incorporating textual representation of soft information into a *graphic* design notation via a software artifact. Thus, whether our final artifact is successful or not in achieving its design goals, its development will *of necessity* yield a substantial amount of information about the extensibility, limits, and conditions of use of our kernel theories. When appropriately formulated and presented, this new information forms the grounding of *a theory of grammatical element salience in conceptual modeling (GESCM)*, a mid-level DSR-IS theory with two characteristics: (1) the power to explain salience in the context of conceptual modeling and (2) far greater facility for extrapolation to specific design criteria than the kernel theories from which it was derived.

Development

It is at the development phase of a design research project that the tentative direction(s) for artifact generation explored in the suggestion phase are made concrete through construction and iterative refinement (Vaishnavi and Kuechler 2007). Two interrelated artifacts emerged from the suggestion phase: (1) a novel dual grammar conceptual modeling technique and (2) a software modeling tool for the presentation of the process models (scripts).

The initial design for the conceptual modeling technique was derived from the statement of modal cognition theory: the mode of cognition termed "narrative thinking" gives rise to "story-like" mental models that both readily incorporate and make salient non-regular information such as soft context. Therefore, a BP model that stimulated narrative thinking could improve process designs. However, a large part of the "design problem" of this research—the mapping from suggestion to a workable artifact—was to develop a modeling technique that maintained the conciseness and precision of graphic representations while simultaneously promoting a mental model that kept soft context salient. We decided to develop a dual grammar process modeling technique that used BPMN for the graphic representation combined with textual process context descriptions and "micro-rationale" narratives; these concisely explained and gave context to the graphics by being integrally linked to related, small portions of the BPMN diagram.

The initial design for the software presentation artifact (essentially process modeling and documentation software) was informed by empirical studies of programmers in action as well as our kernel theory. From theoretical considerations, we believed appropriately presented narrative about a graphical model of a process could enhance the formation of the mental model of the process. However, empirical studies of programmers have shown that diagrammatic representations of systems become the dominant documentation for a system during the later phases of design. The narrative requirements documents that contain the soft-goal information are rarely consulted (Davies et al. 2006; Lethbridge et al. 2003). The failure of many designs to incorporate soft context information is de facto evidence that graphic representations also disproportionately influence *initial* cognitive model formation of the systems. Thus, the design of the presentation software focused on how to insure that the process description narrative and especially the micro-rationales were attended to so that they could have the desired effect. Since prior process modeling software was available to serve as an example, prototyping of the presentation software proceeded fairly rapidly using web-development technologies.

Micro-rationales are our term for short, concise statements of design rationale (Canfora et al. 2000). They are linked to small, coherent portions of a process design (and associated graphic representation) and describe *why* the process segment was designed as it was. By definition, rationale statements are at a level of abstraction above the mechanical description of the process; our BP micro-rationales were at a business evaluative level or social/cultural organizational level. In order to best help

induce a narrative mode of thought they were expressed as complete, syntactically correct sentences, and were woven into a longer "story" or textual description of how the process as a whole functioned. Micro-rationales and process description text are the first grammar of our hybrid modeling notation; BPMN graphical constructs are the second.

Our initial prototype naively assumed that if we presented a BPMN process diagram with some of its graphic elements set up as readily discernable hyperlinks to textual process description and micro rationales (which would display on the other side of the screen), then users would seek all available information and pursue the links. We were wrong. The majority of our pilot study subjects attempted to answer questions about the operation of the process without viewing any of the narrative components (working from the diagram only) *even though they had been instructed in the use of the links and advised of their value, that is, they were responsible for causing the display of information that was not available in the diagram*. Using rollovers in place of links was equally unsuccessful. While these results were fascinating in themselves, we truly wished to test our primary hypotheses—that narrative mode thinking could be induced by a presentation artifact and that it would result in superior reasoning about process designs—and so we ultimately designed the display software to *force* a sequential viewing of process text description and micro-rationales followed by their related process diagram "slices." Screenshots of the final prototype and additional description are given in Appendix 5C.

When appropriately articulated, the design constructs presented briefly in the preceding two paragraphs—dual grammar modeling scripts, presentation technique, and empirical knowledge of user (designer) notation viewing preferences—are available for incorporation into the GESCM theory.

Prototyping the modeling technique and testing the software required content. We required cases that were concise enough to be used in an evaluation session of reasonable duration, did not require uncommon domain information on the part of the user, were realistic, and contained mission-critical soft context requirements. The construction of such cases and the associated narrative and graphic descriptions of them occupied a significant amount of time. Eventually, we entered the pilot phase of our evaluation with three cases derived from real-world process implementations (see Appendix 5A).

Evaluation

In a DSR-IS project, the research process frequently iterates between *development* and *evaluation* phases rather than flowing in waterfall fashion from one phase into the next (Kuechler et al. 2005). Hevner et al. (2004, p. 89) term this iteration the "generate/test" cycle. The evaluation of our artifacts, as for most DSR-IS that deals with human–artifact interaction, took the form of an experiment.

Iteration between *development* (design) and *evaluation* (experiment) is one significant difference between design science research and "natural science" or

theory-driven "behavioral science" experimentation. In natural science research, the experimental procedure and apparatus are (ideally) constructed in such a way as to minimize confounds that might interfere with clear interpretation of the results; theory is either supported or disconfirmed. In design science research, both the artifact and the experimental setting are intentionally complex (and thus confounded) in order to develop methods and artifacts that are useful in practice. Due to the confounded nature of the observations gained in the evaluation phase of a DSR-IS effort, it is difficult if not impossible to disconfirm a theory. However, as noted by other researchers, the relation of a *designed* artifact to theory is *extension* and *refinement* of the theory rather than disconfirmation (Carroll and Kellogg 1989). This fundamental difference encourages the iteration between design and evaluation that would be considered improper "fishing for data" in a natural science experiment.

Though not the focus of this chapter, a brief description of the experimental design (evaluation framework) is necessary to understand the evaluation process.

MBA and MSIS students with more than 5 years of work experience were chosen as subjects. We evaluated the modeling technique and presentation software using the presence or absence of the treatment. Process designs were presented to subjects using either graphical display and separate "design notes" (no treatment) or using the linked dual grammar model (treatment). Each subject was presented with two versions of a process design: original and changed. The changed process eliminated one or more critical soft constraints. The subjects were to determine whether or not the changed process "is effective for the company." Subjects were trained to "think aloud" as they reasoned through answering the question and their concurrent verbal protocols were recorded. The software, in addition to presenting the process design models, tracked the information the subjects chose to view.

Both presentations make available identical information at very similar levels of convenience-of-access. We have followed guidelines for cognitive model experimentation set out in Parsons and Cole (2005) to the degree possible. We have striven to approximate *naturalistic evaluation* of the artifact (Venable 2006a) and believe the external validity of the experiment is strengthened by the nature of the subjects and procedure. Ninety percent of our MSIS student subjects are full-time IT professionals, many with over 15 years of industry experience. We have endeavored to make the experimental procedure realistic by attempting to emulate the "Hey, Ralph, can you take a quick look at this and tell me what you think?" task that in our experience is quite common in industry.

In the course of our study, we cycled between development and evaluation phases of the DSR-IS process numerous times in order to:

- Reprogram the software to force reference to the descriptive text and micro-rationales during treatment (we thought we had done so in the initial design but subjects are exceptionally devious at frustrating experimental expectations).
- Reprogram the software to eliminate display "quirks" that had become transparent to us but were distracting to subjects.

- Redefine process description narrative and micro-rationales to be clearer and to supply broader context. Again, things that were pellucid to us were shown by our pilot studies to need elaboration or rewording to be equally clear to our subjects.
- Rewrite the modeling scripts (as a result of the above refinements).

In fact, on two separate occasions when we believed ourselves to be through with our pilot study and thought we had begun the full experiment, it was necessary to make such significant changes to our prototype and our assumptions that we had to declare the results to that point part of the pilot, recruit more subjects, and begin "the actual experiment" again.

Using terminology from Walls et al. (1992), the goals of the development derive from the *meta-requirements* for the artifact. Our evaluation measurements then test the hypotheses that our *meta-design* has realized those goals. (We discuss design theory development more fully in the next section). The primary goal for the project was to improve the understanding of and reasoning about process models. In addition to better general understanding, we sought the specific improvement of increased salience during process modeling of critical "soft context" information about the process that is difficult to capture in existing process modeling languages and thus is frequently overlooked.

Our evaluation observations were of two types: (1) observations of understanding—the net effect of the artifact. Analysis of this data will tell us the degree to which the design goals had been achieved. (2) Observations of behavior—we will analyze this data in an attempt to understand how the net effects came to be and why they were as they were.

We discuss our observations of understanding first; these also fell under two different classifications: (1) tests for surface understanding of the process—its mechanics, its flow, and the isolated functioning of its activities—and (2) tests for deeper understanding, which includes the interaction of the process with the critical organizational context in which it operates. In educational psychology what we term *deeper understanding* is sometimes called *transfer learning* (Cook et al. 2007).

Surface understanding was operationalized as objective questions about the process, for example, what flowed from activity A to activity B under what decision conditions. Deeper understanding was operationalized as (1) the ability to assess the acceptability of changes to the process in the context described, (2) the ability to construct acceptable alternative changes to the process; changes that accomplished the same goal as the change presented in the session, and did not conflict with the soft or hard constraints presented in the process narrative, and (3) the ability to mentally simulate the performance of the original and/or changed processes under new conditions suggested to the subject after they had been presented with both original and changed processes and had formed mental models of them. Further, we measured both types of understanding with short-term (in session) and long-term (1 week) tests.

We assessed behavior in three different ways: (1) We recorded what the subjects viewed by programming our presentation software to store the information objects subjects chose to view as described earlier. This information will tell us the amount of time subjects spent on each type of information, graphic or textual, the order in which they viewed information, and so on. (2) We trained the subjects to speak aloud as they sought to understand the processes that were presented to them and recorded their verbal protocols. We will code these protocols to understand the different ways in which subjects form cognitive models of processes under the two experimental treatments (Vans and von Mayrhauser 1999). (3) We asked questions about their confidence in their answers to questions under the two treatments and about their information preferences—graphics or narrative—in differing business situations.

As of this writing we have completed data gathering, have transcribed the protocols, and have almost completed preliminary coding of the protocols. We have not begun formal analysis and so cannot claim statistical significance; however, preliminary observations have been encouraging. We have seen evidence of the development of different cognitive models in treated and untreated groups both in analysis of the verbal protocols and in better confidence, richer mental simulations, and objectively better correctness-of-answer scores for the treatment group. The pilot findings that drove redesign of the preliminary artifact are available also for incorporation into the still nascent GESCM theory.

Theory Development

The following discussion consolidates the theory development that has taken place during the DSR project to date. To maintain the focus of this chapter on theory development rather than the actual artifacts, we confine our discussion to theory statements concerning the display artifact only. Equally rigorous development for the dual grammar conceptual model can also be presented.

First we present the constructs used to express our theoretical propositions (see Table 5.2). We use a table format since all of the constructs have been discussed at previous points in this chapter. Second, we state and discuss propositions from our kernel theories: *modal cognition theory* and *multi-media comprehension* that seemed to have relevance to our design project. We then state and discuss the foremost proposition of the mid-range theory informed by our kernel theory, a *theory of GESCM*. We use the term "informed" to make it very clear that the link from kernel to mid-range theory is *not* one of logical deduction or other rigorous, formal procedure, but rather is due to what has been termed the "hypothetical/deductive" method (Baldwin and Yadav 1995). The hypothetical/deductive method is the introspective explication of the results of the cognitive process of analogical reasoning (Gentner 1983) from one domain to another, which we believe to be the basis of the kernel–mid-range inferences, followed by formal statement of these results. Lastly, we state

and discuss the tentative propositions of a design theory for the display artifacts: the *process model presentation software*.

Theoretical Constructs

Table 5.2 Theoretical Constructs for Kernel and Mid-Range Theories

Construct	*Definition*
Mental model	The internal, cognitive model (in this case, of business processes) that contains the information about the model elements and their relationships
Modes of cognition	Modes of perceiving information that determine the types of information most readily acquired and the strength of relationships between information elements as mental models are formed
Surface understanding (of processes)	Understanding of the "mechanics" of process elements—flows, actors, and decisions at an algorithmic level—excluding domain or context information
Deep understanding (of processes)	Surface understanding combined with knowledge of the context in which the process operates and the interactions, actual and potential, between the process and its environment
Soft context information	Organizational, cultural, or political information about the actors or environment of a process that is difficult to capture in conventional process notations but that is frequently critical to the success of the process. In a medical informatics context, e.g., the aversion of many older MDs to information technology is one example of soft context
Narrative (sometimes termed *text*)	Information in language form
Micro-rationales	Small concise narrative segments relating process details or context not found in diagrammatic representations, usually woven into a coherent "story" about the process
Salience	In this context, the term denotes the degree of attention and significance given to different information elements of a conceptual model

Kernel Theory Propositions

- From the modal cognition literature:
  - The cognitive model formed from information about a situation can be made more receptive to social or "soft" information by varying the mode of information presentation from abstract-propositional (numeric) to narrative (textual).
  - (Note that a proposition of exactly this form can likely *not* be found in the literature. We have presented our interpretation, which at this point is quite informed. We have taken no liberties with matters of fact, but have "repackaged" conclusions from the kernel literature to concisely state what was of interest to us. The restatement also makes it easier to follow our development from one theory level to the next.)
- From the multi-media comprehension literature:
  - Richer cognitive models of physical processes that demonstrate greater transfer learning (across domains) result from mixed-media presentations of the processes, that is, text + illustrations, than from text or illustrations alone.

Mid-Range Theory Propositions

(A theory of GESCM)

1. In systems design a conceptual model can be used to concisely represent one or more important aspects of the system.
2. A system always operates in a context. Usually the grammar(s) for the conceptual model(s) of the system is optimized for the representation of a narrow range of system constructs. Specifically, these grammars are not well suited to representing organizational context information, especially when they are graphical in form.
3. Organizational context information can be expressed in narrative (language) form.
4. Virtually all business systems are artificial—they are designed and there are reasons called *design rationale* that describe why they are as they are. Design rationale also can be expressed in narrative form.
5. When conventional (narrowly focused) conceptual models for processes are linked in a designer's mental model to expressions of critical organizational context and design rationale, better design decisions are achievable.
6. Computer-based conceptual model design and display artifacts can be built that force attentional links between conventional conceptual model element displays and narrative information displays of organizational context and design rationale so as to facilitate the construction in the user of the artifact of

mental models that link context information with the information captured by the conventional conceptual model.

7. The strongest and most useful overall mental model (conventional conceptual model and narrative components) will be produced when the narrative components are woven into a coherent (by basic literary standards) *story* rather than presented as separate, intelligible but logically unconnected text components. (This is one of the distinguishing features between a dual grammar conceptual model and a simple annotated conceptual model graphic display.)

Note the conceptual "leap" from kernel theory propositions to the primary propositions of GESCM. No existing research from the kernel fields allows us to draw the preceding propositions (6) or (7) as *conclusions*. They are at best inductions and need to be tested. However, the propositions are much closer to the information systems design domain than any of the kernel theories and *immediately suggest testable hypotheses where the tests are in the form of the evaluation of artifacts designed in accordance with the propositions.*

Design Theory Propositions

In setting out the design theory (see Table 5.3) derived from our mid-range theory statement, we continue to use the concepts—and for this section even the presentation format—from Walls et al. (1992, 2004).

In developing our mid-range theory from our kernel theories, we descended a level of abstraction; alternatively stated, the mid-range theory became more concrete. The kernel theories dealt with general cognitive abilities. GESCM applies these theories inductively to the more concrete realm of computer-mediated conceptual models. Transitioning from mid-range to design theories, we become still more concrete. At the ISDT level the statements are scoped to computer software for presenting graphic process models and related textual design rationale and context information. We believe this is still at a meta-level appropriate for an ISDT, that is, it applies to a class of process model presentation artifacts and leaves the graphic portion of the grammar and many other important design features unspecified.

Conclusions

The research project described in this chapter is an example of design science research that can yield not only a prescriptive design theory for a class of artifacts, but can also refine and extend the kernel theory that suggests the novelty in the artifact design approach. The novel information from artifact design and evaluation that we have captured and articulated forms the basis of a mid-range theory, *a theory of*

Table 5.3 Design Theory for Cognitively Enhanced Process Model Presentation Software

		Theory Component	*Description*
Design Product	1.	Meta-requirements	Multiple types of process information: graphic representations of process mechanics, narrative representation of organizational context, and narrative representations of design rationale are presented to the user in a manner that induces linkages in the overall mental model of the process
	2.	Meta-design	Graphic process representation components are displayed in logical sequence with linked narrative *necessarily* displayed before the subsequent or prior graphic component can be displayed
	3.	Theories that inform or suggest the solution (from any field)	Modal cognition theory + multi-media comprehension theory
	4.	Artifact evaluation criteria and measures	Users will develop richer cognitive models of business processes leading to better (re)design decisions
Design Process	1.	Design method	*
	2.	Theories that inform or suggest the design method	*
	3.	Design process evaluation and measures	* The design process for the display software did not seem to us to be outside the state-of-practice for sophisticated educational or www-commercial software

Source: Format taken from Walls, J. et al., *Journal of Information Technology Theory and Application*, 6(2): 43–58, 2004.

** Walls et al. (1992, 2004) define a complete ISDT as possessing both a product and a process component. However, after much reflection we are unable to see that the process by which we designed our display artifact was novel in any meaningful way.*

GESCM. The research meets the guidelines for design science research in IS set out in Hevner et al. (2004) and also follows one of the artifact evaluation approaches suggested in that paper: a controlled experiment.

With reference to Figure 5.3, kernel theories from outside IS entered the design science research process at two points. Theories of "narrative thinking," a mode of cognition receptive to unpatterned information, led to a novel design approach to a conceptual modeling grammar in the suggestion phase. Theories of multi-media comprehension from educational psychology informed both the grammar design at the suggestion phase and the design of the software artifact in the development phase. Since the evaluation of both research artifacts is accomplished with a controlled experiment, refinement of the kernel theories into the GESCM theory—as embedded in the artifacts—will be both statistically valid and rigorous within the limits of the design science paradigm. The paradigm necessarily introduces confounds into the interpretation of results, however, it also produces extension and refinement of the theories in the event of either success *or* lack of success of the artifacts.

If the artifacts are successful, they will ground the new mid-range GESCM theory and further experimentation in DSR-IS projects can extend and refine the theory. The GESCM theory is much more readily adoptable into future DSR-IS projects than were the kernel theories from which it was derived. If the artifacts are unsuccessful, they will suggest *limitations* to the kernel theories that were not obvious in the original theory statements. For example, lack of significant results for the artifacts in this project would suggest the induction of narrative thinking is more difficult when graphical representations supply much of the information on a problem than when the information is supplied solely by narrative and numeric representations as it was in the kernel theory experiments.

The DSR-IS project presented in this chapter is not unique in its ability to refine and extend kernel theory into mid-range DSR-IS theory. In fact, we believe along with other authors (Venable 2006a; Carroll and Kellogg 1989) that artifact design projects are the best possible opportunities for refining theory from other fields for use in IS. The nature of different research paradigms—natural and behavioral science versus design science—makes it unlikely that theory from outside design science will be readily adaptable to artifact construction. Natural and behavioral science experiments take place in much more restricted environments than those for design science artifact evaluation and typically use different levels of analysis than DSR-IS. Thus, almost all DSR-IS projects using kernel theories inevitably refine and extend those theories. It is our hope that this theory refinement and extension can come to be widely acknowledged as a potential part of and benefit of the DSR-IS process. Such acknowledgement would encourage the articulation, theoretic formulation, and publication of DSR-IS mid-range theories to the enhancement of all areas of IS research.

References

Baldwin, D. and Yadav, S. (1995). "The Process of Research Investigations in Artificial Intelligence—An Unified View." *IEEE Transactions on Systems, Man and Cybernetics* 25(5): 852–861.

Canfora, G. Casazza, G., and De Luca, A. (2000). "A Design Rationale Based Environment for Cooperative Maintenance." *International Journal of Software Engineering & Knowledge Engineering* 10(5): 627–646.

Carroll, J. and Kellogg, W. (1989). "Artifact as Theory Nexus: Hermeneutics Meets Theory-Based Design." In *Proceedings of CHI'89*, ACM Press.

Cook, D., Holder, L., and Youngblood, C. (2007). "Graph-Based Analysis of Human Transfer Learning Using a Game Testbed." *IEEE Transactions on Knowledge and Data Engineering* 19(11): 1465–1478.

Cysneiros, L., Leite, J., and Nito, J. (2001). "A Framework for Integrating Non-Functional Requirements into Conceptual Models." *Requirements Engineering* 6(2): 97–115.

Dasgupta, S. (1996). *Technology and Creativity*. New York, NY: Oxford University Press.

Davies, I., Green, P., Rosemann, M., Indulska, M., and Gallo, S. (2006). "How Do Practitioners Use Conceptual Modeling in Practice?" *Data & Knowledge Engineering* 58(3): 358–380.

Fickas, S. and Helm, R. (1992). "Knowledge Representation and Reasoning in the Design of Composite Systems." *IEEE Transactions on Software Engineering* 18(6): 470–482.

Gause, D. (2005). "Why Context Matters—And What Can We Do about It?" *IEEE Software* 22(5): 13–15.

Gentner, D. (1983). "Structure-Mapping: A Theoretical Framework for Analogy." *Cognitive Science* 7(2): 155–170.

Goldkuhl, G. (2004). "Design Theories in Information Systems—A Need for Multi-Grounding." *Journal of Information Technology Theory and Application* 6(2): 59–72.

Gotel, O. and Finkelstein, A. (1995). *Contribution structures [Requirements artifacts]*. Second IEEE International Symposium on Requirements Engineering (RE'95), IEEE Computer Society.

Gregor, S. (2006). "The Nature of Theory in Information Systems." *MISQ* 30(3): 611–642.

Gregor, S. and Jones, D. (2007). "The Anatomy of a Design Theory." *Journal of the Association for Information Systems (JAIS)* 8(5): Article 19.

Hevner, A., March, S., Park, J., and Ram, S. (2004). "Design Science in Information Systems Research." *MIS Quarterly* 28(1): 75–105.

Iivari, J. (1986). "Dimensions of Information Systems Design: A Framework for a Long-range Research Program." *Information Systems Frontiers* 11(2): 185–197.

Jou, J., Shanteau, J., and Harris, R. (1996). "An Information Processing View of Framing Effects: The Role of Causal Schemas in Decision Making." *Memory and Cognition* 24(1): 1–15.

Kuechler, W. and Vaishnavi, V. (2006). "So, Talk to me: The Effect of Explicit Goals on the Comprehension of Business Process Narratives." *MIS Quarterly* 30(4): 961–996.

Kuechler, W.L. and Vaishnavi, V.K. (2008). "The Emergence of Design Research in Information Systems in North America." *Journal of Design Research* 7(1): 1–16.

Kuechler, W. and Vaishnavi, V. (2012). "A Framework for Theory Development in Design Science Research: Multiple Perspectives." *Journal of the Association for Information Systems (JAIS)* 13(6): 395–423.

Kuechler, W., Vaishnavi, V., and Petter, S. (2005). "The Aggregate General Design Cycle as a Perspective on the Evolution of Computing Communities of Interest." *Computing Letters* 1(3): 123–128.

Lethbridge, T., Singer, J., and Forward, A. (2003). "How Software Engineers Use Documentation: The State of the Practice." *IEEE Software* 20(6): 35–39.

Lewalter, D. (2003). "Cognitive Strategies for Learning from Static and Dynamic Visuals." *Learning and Instruction* 13(2): 177–189.

Maiden, N., Manning, S., Jones, S., and Greenwood, J. (2005). "Generating Requirements from Systems Models Using Patterns: A Case Study." *Requirements Engineering* 10(4): 276–288.

March, S. and Smith, G. (1995). "Design and Natural Science Research on Information Technology." *Decision Support Systems* 15(4): 251–266.

Markus, L. and Lee, A. (2000). "Foreward: Special Issue on Intensive Research." *MIS Quarterly* 24(3): 473–474.

Mayer, R. and Jackson, J. (2005). "The Case for Coherence in Scientific Explanations: Quantitative Details Can Hurt Qualitative Understanding." *Journal of Experimental Psychology: Applied* 11(1): 13–18.

Merton, R. (1968). *Social Theory and Social Structure*. New York, NY: Free Press.

Mylopoulos, J., Chung, L., and Nixon, B. (1992). "Representing and Using Nonfunctional Requirements: A Process-Oriented Approach." *IEEE Transactions on Software Engineering* 18(6): 483–497.

Nelson, K., Nadkarni, S., Narayanan, V., and Ghods, M. (2000). "Understanding Software Operations Support Expertise: A Revealed Causal Mapping Approach." *MIS Quarterly* 24(3): 475–507.

Nissen, H.W., Jeusfeld, M., Jarke, M., Zemanek, G., and Huber, H. (1996). "Managing Multiple Requirements Perspectives with Metamodels." *IEEE Software* 13(2): 37–48.

Nunamaker, J., Chen, M., and Purdin, T. (1991). "Systems Development in Information Systems Research." *Journal of Management Information Systems* 7(3): 89–106.

Orlikowski, W. and Iacono, C. (2001). "Desperately Seeking the "IT" in IT Research—A Call to Theorizing the IT Artifact." *Information Systems Research* 12(2): 121–134.

Parsons, J. and Cole, L. (2005). "What do the pictures mean? Guidelines for Experimental Evaluation of Representation Fidelity in Diagramatical Conceptual Modeling Techniques." *Data & Knowledge Engineering* 55: 327–342.

Purao, S. (2002). "Truth or Dare: Design Research in Information Technology." *GSU CIS Department Working Paper*, 2002.

Seufert, T., Janen, I., and Bruken, R. (2007). "The Impact of Intrinsic Cognitive Load on the Effectiveness of Graphical Help for Coherence Formation." *Computers in Human Behavior* 23(3): 1055–1071.

Simon, A. (1996). *The Sciences of the Artificial*, Third Edition. Cambridge, MA: MIT Press.

Stefansen, C. and Borch, S. (2008). "Using Soft Constraints to Guide Users in Flexible Business Process Management Systems." *International Journal of Business Process Integration and Management* 3(1): 26–35.

Tversky, A. and Kehneman, D. (1981). "The Framing of Decisions and the Psychology of Choice." *Science* 211(4481): 453–458.

Vaishnavi, V. and Kuechler, W. (2004/13). "Design Science Research in Information Systems" January 20, 2004; last updated October 23, 2013. URL: http://www.desrist.org/design-research-in-information-systems/ (last accessed on January 28, 2015)

Vaishnavi, V. and Kuechler, W. (2007). *Design Science Research Methods and Patterns: Innovating Information and Communication Technology*. New York, NY: Auerbach.

Vans, A. and Von Mayrhauser, A. (1999). "Program Understanding Behavior during Corrective Maintenance of Large Scale Software." *International Journal of Human-Computer Studies* 51(1): 31–70.

Venable, J. (2006a). "The Role of Theory and Theorizing in Design Science Research." In *Proceedings DESRIST 2006*, Claremont, CA.

Venable, J. (2006b). "A Framework for Design Science Research Activities." In *Proceedings of the 2006 Information Resource Management Association Conference*. Washington, DC, USA, 24–26 May 2006.

Walls, J., Widmeyer, G., and El Sawy, O. (1992). "Building an Information System Design Theory for Vigilant EIS." *Information Systems Research* 3(1): 36–59.

Walls, J., Widmeyer, G., and El Sawy, O. (2004). "Assessing Information System Design Theory in Perspective: How Useful was our 1992 Initial Rendition." *Journal of Information Technology Theory and Application* 6(2): 43–58.

Wand, Y. and Weber, R. (2002). "Information Systems and Conceptual Modeling—A Research Agenda." *Information Systems Research* 13(4).

Yu, E. (1995). "Models for supporting the redesign of organizational work." In *The Proceedings of the Conference on Organizational Computing Systems*, ACM, 225–236.

Yu, E. and Mylopoulos, J. (1994). *Understanding "Why" in Software Process Modeling. 16th International Conference on Software Engineering*, Sorrento, Italy.

Zukier, H. and Pepitone, A. (1984). "Social Roles and Strategies in Prediction: Some Determinants of the Use of Base-Rate Information." *Journal of Personality and Social Psychology* 47(2): 349–360.

Zukier, H. (1986). The paradigmatic and narrative modes in goal-guided inference. *Handbook of Motivation and Cognition: Foundations of Social Behavior*. R.M. Sorrentino and E.T. Higgins (Eds.). New York, NY: Guilford.

Zukier, H. (1990). Aspects of Narrative Thinking. *The Legacy of Solomon Asch: Essays in Cognition and Social Psychology*. I. Rock (Ed.). Hillsdale, NJ: Lawrence Earlbaum and Associates, 195–209.

Appendix 5A A Process Change Scenario Illustrating "Soft Context Information" (A True Story)

Note that this scenario describes the revision of a significant organizational process that involves both information technology and non-automated process actions. The overall process is sometimes referred to as a "composite system" (Fickas and Helm 1992). The mission critical "soft context" information for this particular process revision is shown in bold italics in the scenario description in the following text.

A medium sized U.S. university made an administrative decision to transition from paper-based student course evaluations to a web-based system. One of the university IT department's senior analysts gathered requirements for the system and was placed in charge of the project. The analyst was told the primary

driver for the new system was the high cost of processing the paper forms. The analyst was also cautioned during interviews with several administrators that ***the system needed to generate very near the number of evaluations per course that the current system produced or the results would not be accepted.*** Not uncommonly this soft context information was never translated into a composite system requirement. A web-based system was developed that, when used, generated exactly the information required by the faculty and administration at a fraction of the cost per response. Unfortunately, the students saw no reason to take on the additional work of entering information into the system at a very busy time in the semester, and the system did not generate enough results to be usable. Several "obvious" paths to greater use, such as requiring the students to enter evaluation information before grades would be issued for them, are politically unpalatable at the university. After several semesters of unsuccessful attempts to exhort students to greater system use, the university is on the verge of abandoning the system.

Appendix 5B System Quality Representation

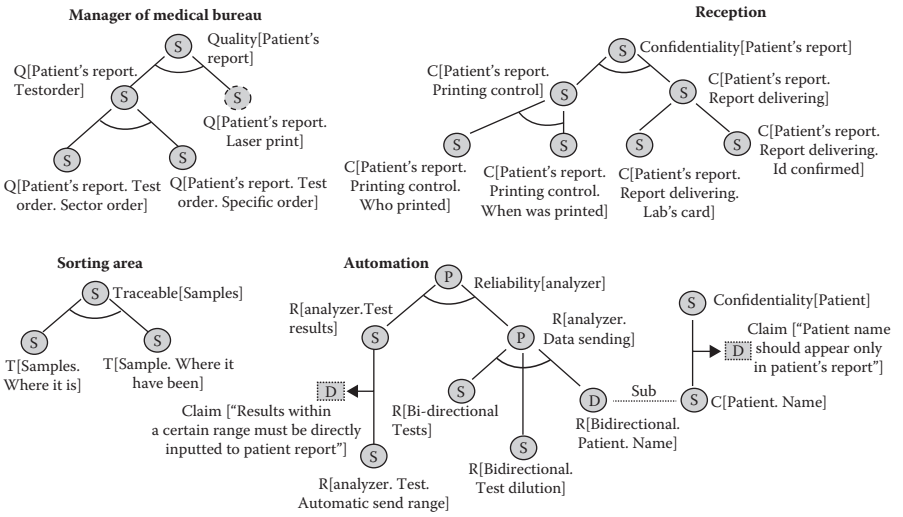


Figure 5B.1 AND/OR graphs used to represent system quality. (From Cysneiros, L., et al. *Requirements Engineering*, 6(2): 97–115, 2001.)

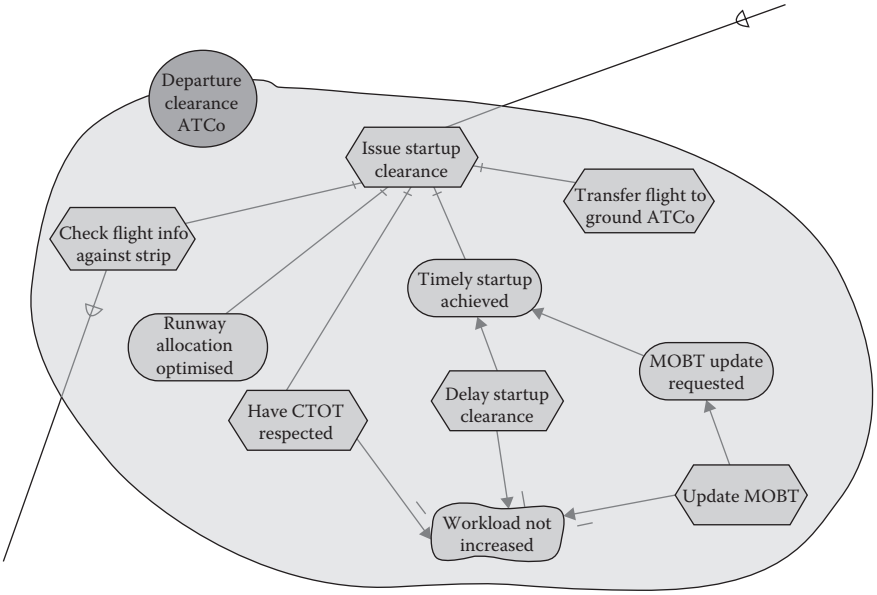


Figure 5B.2 i* graphs used to represent system context for an air traffic control system. (A very small portion of the total graph, from Maiden, N., et al. *Requirements Engineering*, 10: 276–288, 2005.)

Appendix 5C Sample Process Graph "Slices" and Associated Text Description and Micro-rationale as used in our Evaluation Prototype

With reference to the preceding diagram, the prototype works as follows for the *treatment* session:

In the actual prototype, the screen is wide enough to display a 50-character wide text section on the left of the screen and the full diagram on the right of the screen. Initially, instructions that are displayed on the left and only slice zero—the swim lane names and the graphic heading—are visible. The subject must click on the text to view the next information segment. Information segments alternate between narrative—descriptive text and micro-rationales—and the next sequential graphic slice. Text segments are displayed in sequential positions down the text display portion of the screen. Each piece of information, whether text or graphic, fades from view in 9 seconds. The subject must click on the information to make it reappear for 9 seconds. The only exception to this is the initial display of the graphic associated with a given text segment. That is, on clicking a text segment, the associated graphic is displayed and both are visible. However, after clicking on the associated graphic slice, both the graphic and its associated text disappear,

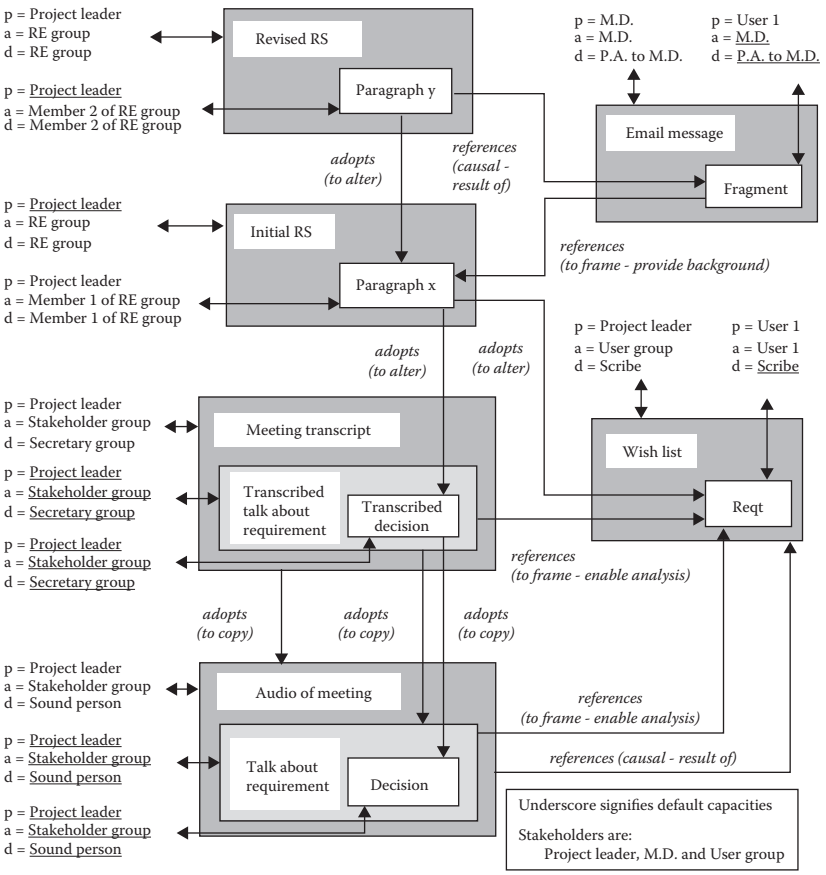


Figure 5B.3 Connectivity structures. (From Gotel, O. and Finkelstein, A., *Contribution structures [Requirements artifacts],* 1995.)

and the next text segment appears. The prototype records the time and object for every mouse click. During final data analysis, the click traces will augment coded transcriptions of the concurrent verbal protocols that were recorded as the subjects proceeded through the process display.

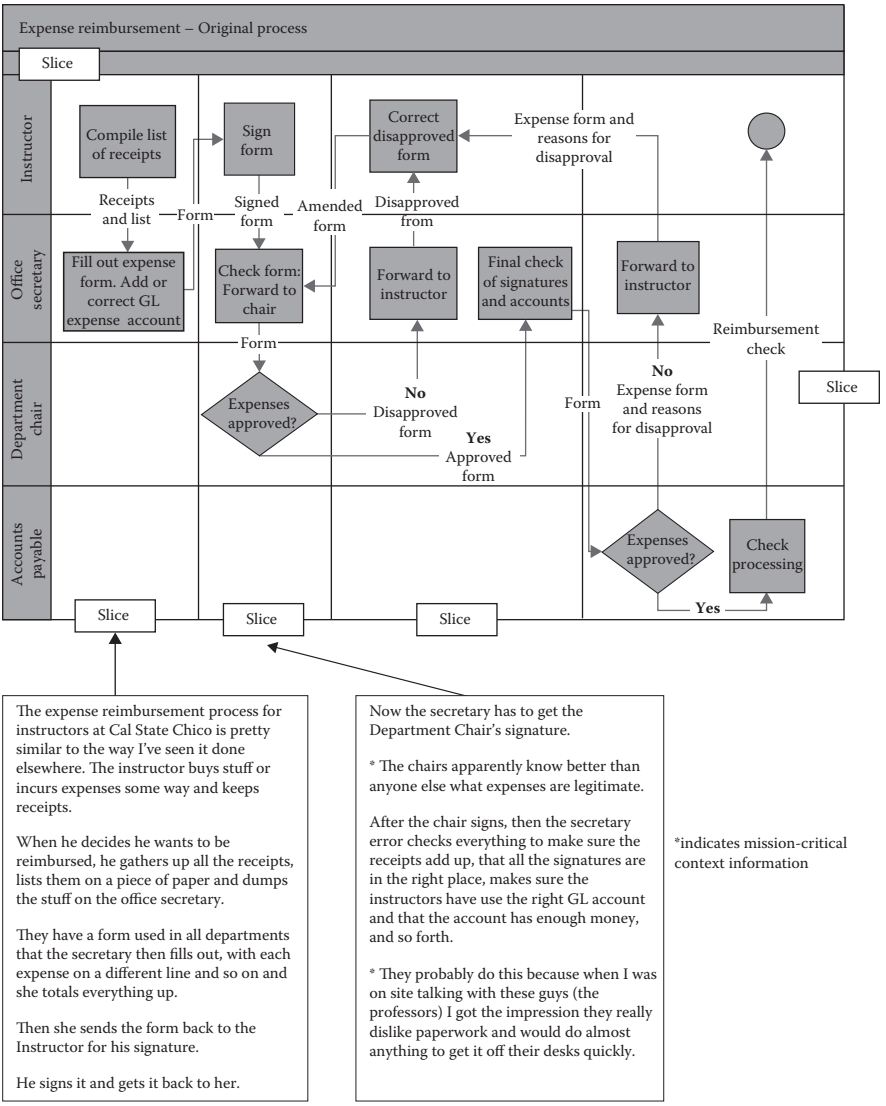


Figure 5C.1 Descriptive text and micro-rationales for slices 1 and 2.

PATTERNS II

"Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice."

Christopher Alexander

This part, constituting a major portion of the book, focuses on how design science researchers in information and communication technology actually conduct their research. It presents the results of a multiyear effort in mining patterns for conducting research. The book is the first published work to use patterns for encapsulating research process knowledge. The mining of patterns for this type of knowledge is an ongoing process and is incomplete by its very nature.

The part is divided into seven chapters. The first of these chapters provides a brief introduction on patterns and how to use the patterns for conducting design science research. The rest of the chapters in this part present 84 patterns divided into the different phases of a design science research project.

Chapter 6

Using Patterns to Illuminate Research Practice

Introduction

In this chapter, we describe patterns from several perspectives: first their historical origins as a means of communicating architectural design themes, and then as they are used in this book to describe aspects of the *art* of design science research. The general design science research cycle is then revisited (from previous chapters) adapted specifically for use as a framework for understanding design science research projects. This is followed by briefly discussing how the patterns in this book were mined, a discussion of the related TRIZ (Wikipedia-TRIZ 2014) approach used mainly in engineering, and outlining the structure used in presenting the patterns in this book. The chapter concludes with an extended case study of the use of patterns in the design science research project—development of the smart object paradigm—recounted in narrative form in Chapter 2.

Patterns, Then, and Now

Patterns, as we use the term here, is a communication technique developed and first used by Christopher Alexander (1964) to communicate a *way of building* structures to his architecture students. Alexander's intent was not to communicate facts about structures—how to calculate the loading on a specific type of stair, for

example—but rather to convey the much more subtle skill (or art) of constructing structures whose components flowed gracefully and meaningfully into one another to create a coherent design. Some of the problems inherent in trying to communicate this type of knowledge are visible in the word usage that describes the result: what does it mean for components to flow? Even more, what does it mean for them to flow gracefully and meaningfully? Finally, what is a "coherent" design? A time honored answer to such questions is: Let me show you an example. In a literal sense, patterns are a language-based way to communicate let-me-show-you-an-example. They are similar to but shorter and more structured than the case studies used to communicate similarly subtle and impossible-to-precisely-pin-down knowledge in business classes.

Patterns are also frequently defined as "a solution to a problem in a recurring context." However, the context for the type of problems patterns best address is never identical and so patterns are typically goal based rather than strictly algorithmic. A pattern demonstrates a way to or general technique for approaching a class or type of problems that are abstractly similar to other problems even though they have never occurred before in exactly the same way. Patterns are almost never presented as a set of strict rules because precision always limits applicability.* At this point, our ability to describe patterns with more words has been exhausted and so we too now fall back on examples.

As mentioned, research is at best a semi-structured activity. This is true in part because the nature of the activity is to explore the unknown, that is, the unstructured. A common problem when pursuing research in an interesting but new-to-you area is to become overwhelmed by the new information you have gathered which, by definition, is only generally applicable to an area not well understood by anyone and especially not by you. Place yourself in that problem; most researchers, however new, have had this experience—try to recall that feeling as vividly as possible. Now turn to the pattern on p. 187 of this book: *Structuring an Ill-Structured Problem*. Does it provide any assistance to you? Does it provide at least a high-level ordering-principle? For further assistance with the same problem, read carefully the pattern on p. 178 of this book, *Complex System Analysis*. Notice that the patterns focus your attention on a specific aspect of a situation without being task specific. They use phrases such as "analyze the structure..." without specifying what they mean by analysis or structure. In this way they are able to focus your vast store of tacit knowledge (or "common sense").

* A textbook in artificial intelligence from the second author's graduate study (Firebaugh 1988) contained a "quantitative" version of the precision-limits-applicability truism that is applicable here: Generality * Utility = C (a constant). In other words, you can make a concept or artifact more immediately useful only by making it more specific and thus limiting its generality. Conversely, the general applicability of a concept or artifact can be increased only by abstracting it and decreasing its utility in any specific context.

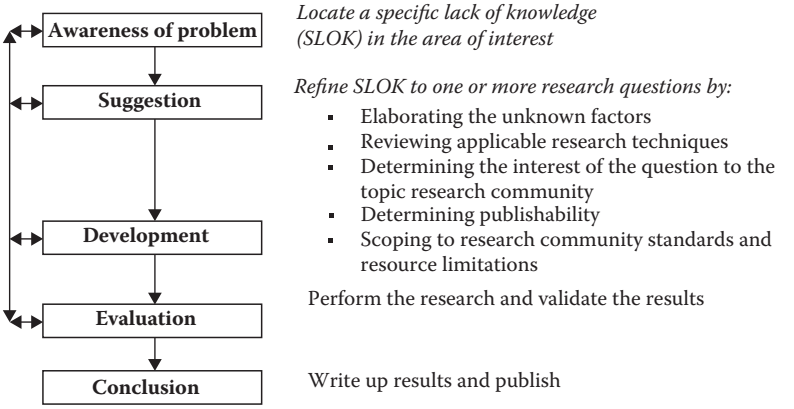


Figure 6.1 An abstraction of the design science research cycle.

Using Patterns: The Design Science Research Cycle Revisited

The arrows out of and into every phase of Figure 6.1, an abstraction of the design science research cycle discussed in Chapter 2, indicate that the method is indefinitely iterative. At any phase prior to (a satisfactory) conclusion, it is possible and sometimes necessary to return to an earlier phase. This is the nature of creative thought in general and designs in particular; examples of iteration between phases are given in the extended description of the authors' design science research project given in Chapter 2 ("An Example of ICT Design Science Research"). As also discussed in Chapter 2, the iterative nature of the method makes possible the generation of circumscription knowledge not possible without iteration.

The methodology and the patterns are quite general; however, they make several assumptions that we now make explicit:

1. Interest in the area of investigation. While many of the patterns are intended to help narrow the scope of research (*research domain identification*, *problem formulation, research topic identification*) or align it more closely with a community of research or practice (*understanding research community*, *research conversation, industry/practice awareness*), we assume you have chosen an area in which you already have a general interest.
2. A desire to publish. This assumption follows closely from the assumption of genuine interest in an area. We assume the research is intended to produce new knowledge that you and some community will feel it is valuable and

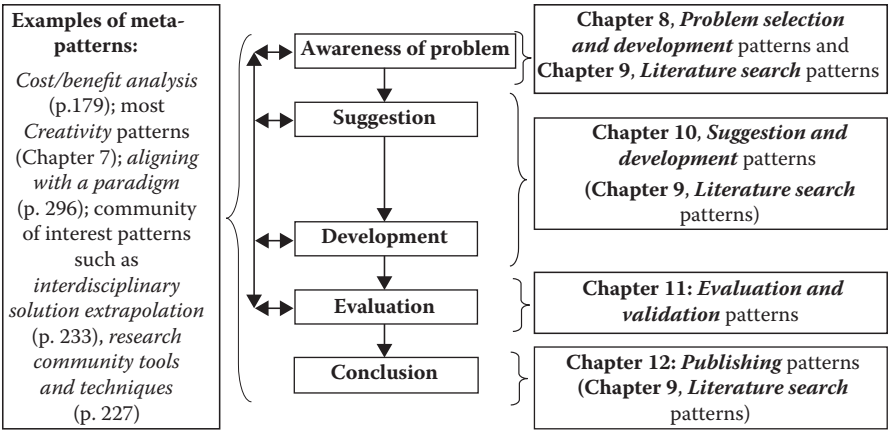


Figure 6.2 Patterns applicable at various phases of the design science research cycle.

interesting and that you will wish to share the knowledge through a research publication (or patent*).

Figure 6.1 has been modified as Figure 6.2 in which the pattern categories applicable to each phase of the methodology are indicated by their name or the number of the chapter in which they are found adjacent to each phase. If you are reading the CD ROM of the online version of this book, each of the pattern designations is a link to the page describing the pattern or the beginning of the applicable chapter.

The categorization scheme we have chosen for the patterns corresponds to the major activities of a design science research project when the project is viewed as a work process. The categories are used as the chapter headings of the current part (Part II) of the book in which the patterns are described in detail:

- Creativity (Chapter 7)
- Problem selection and development (Chapter 8)
- Literature search (Chapter 9)
- Suggestion and development (Chapter 10)
- Evaluation and validation (Chapter 11)
- Publishing (Chapter 12)

* The patterns in this book focus on design science research publications as an outcome of such research but many of them are also relevant to patents. Also, there is usually a research publication associated with a patent.

Most of the *creativity* patterns (Chapter 7) are applicable to all the phases of the research project. The patterns in Chapters 8 and 9 are applicable to the *awareness of problem* phase of the project. The patterns in Chapter 10 are applicable to two phases of the research project, *suggestion* and *development*. Patterns in Chapters 11 and 12 are applicable to the *evaluation* and *conclusion* phases of the project, respectively. In addition, some of the patterns in Chapter 9 are also useful in the *suggestion* and *conclusion* phases of the research project.

Note also that some of the pattern categories appear adjacent to multiple methodology phases and that some patterns are shown as applicable across the methodology as a whole. This is because different patterns operate at different levels of abstraction. For example, most of the patterns on creativity (Chapter 7) are applicable at any point in the research process when progress is stalled for want of ideas. By contrast, the pattern *familiarization with new area* (p. 200) is applicable primarily at the *awareness of problem* stage or the *suggestion* stage. We will refer henceforward to the most broadly applicable patterns as *meta-level* patterns. Due to its generality (as opposed to its level), such a pattern may be applicable at multiple stages in the methodology. This is the strength of patterns, but it can be confusing: some patterns are context independent, and it is up to the user to supply the context and details of the usage suggested by the pattern.

Mining of Design Science Research Patterns*

The design science research patterns described in this book are the result of mining such patterns from the authors' research experiences as well as from the literature; mining of these patterns was carried over a number of years as part of the design science research methods course taught by the first author. Some of the sources of the patterns have been discussed in the book as examples for the patterns and as pattern usage exemplars, described in the third part of the book. These empirical sources also serve to validate the patterns. For patterns that are adaptations of TRIZ inventive principles (see Appendix 6A) additional validation is provided by the fact that the principles are based on the study of about 40,000 patents (Altschuller 2005; Wikipedia-TRIZ 2014).

Problem-Solving Patterns in Engineering: The TRIZ Approach

This book deals with patterns for design science research whose goal is to aid the researcher in guiding the conduct of such research—creation of new knowledge. There has been a similar quest mainly in the field of engineering for mechanizing

* The patterns have been mined specifically for design science research but many of them are applicable to other types of research as well.

the creation of patents leading to the creation of the TRIZ approach. TRIZ is the Russian acronym for the "Theory of Inventive Problem Solving." G.S. Altschuller and his colleagues developed TRIZ between 1946 and 1985 by studying and analyzing successful patents and patent applications, which continues as an international activity. The TRIZ approach is well established in academia as well as in industry (Altschullar 2005; Gericke 2009). An important component of the TRIZ approach is a set of 40 "inventive principles" that are believed to virtually account for all inventive solutions in the patents studied (Gericke 2009). The 40 inventive principles in TRIZ, described in the appendix (Appendix 6A), are similar to patterns except for their focus—mostly engineering but also manufacturing, management, and so on—and their rather terse descriptions. They are well founded in empirical data since they are derived from about 40,000 reviewed and analyzed patents of inventions (Altschuller 2005; Wikipedia-TRIZ 2014).

Gericke (2009) compares the DSR patterns (Vaishnavi and Kuechler 2007) with the TRIZ inventive principles and proposes certain new DSR patterns that could result from transferring TRIZ inventive principles to the design science research domain. Building on this work, we discuss the connection of existing (Vaishnavi and Kuechler 2007) or new DSR patterns that are related to or can be considered to be adaptations of certain TRIZ principles. This connection of DSR patterns to TRIZ principles provides additional verification to the patterns since the TRIZ principles are empirically based on the study of existing patents. The appendix identifies the TRIZ principles with such connections to DSR patterns; the explanation of such connections is provided in the description of the respective DSR patterns.

Pattern Structure

Following and adapting the convention used for describing patterns, the patterns in this book will be structured as follows:

<<name>> (name of the pattern)
Type (type of the pattern,* i.e., what can this pattern be classified under)
Intent (purpose of the pattern)
Motivation (*why* one should be interested in considering this pattern)
Context/applicability (*when* can this pattern be useful or applicable)
Description (*what* is the "how-to" information contained in the pattern)
Notes (any applicable additional information)
Consequences (what can one expect as the result of the use of the pattern)
Usage example(s) (brief description of any pattern usage example(s))

* Not provided for evaluation and validation or publishing patterns. The implicit type of evaluation and validation patterns is Evaluation and Validation. Similarly the implicit type of publishing patterns is Publishing.

Connection to TRIZ inventive principles (adaptation of or relationship with TRIZ inventive principles, if applicable)
Related pattern(s)* (partial list of patterns that are related to this pattern and could be used along with the pattern)
Reference(s) (any relevant reference(s)

Pattern Usage in the Development of the Smart Object Paradigm

A case illustrating the use of patterns in an actual design science research project is presented in the following to make the concepts concrete and situate them in their use-context by showing the actions that resulted from their application. Since it is the case with which we have the most familiarity, we will revisit the smart object project, discussed in "An Example of ICT Design Science Research" (Chapter 2) to illustrate the general design science research cycle itself.

Please review in Chapter 2 the section entitled "An Example of ICT Design Science Research" prior to reading the pattern use discussion. That section gives the story line for the case and covers many details that are not repeated here. The use of patterns described later in the text follows the narrative in the cited section and follows the general design science research cycle of Figures 6.1 and 6.2. The designed artifact in this case is the research project itself. The multiple goals for the project are as follows:

1. Determine a problem interesting to one or more design science research communities.
2. Scope the problem to available resources while maintaining its "interestingness."
3. Solve the problem (improve the problem situation) with a designed artifact, that is, design and implement the artifact.
4. Evaluate the artifact.
5. Publish the results of the study.

The problems that arose in the pursuit of these goals and the patterns that were used to approach and overcome those problems are discussed in the following.

Pre-awareness of Problem

The broadest context in which to consider the case and the application of patterns is the academic setting in which it took place. There is an intellectual restlessness in any PhD granting institution (in this case Georgia State University) and it arises from two

* Other patterns of the same type are obviously related to the pattern and so are not listed among its related patterns. Any differentiating information among patterns within a type cluster is provided when discussing the cluster.

sources: the high native levels of interest in certain subject areas in the people who self-select to such environments and the pragmatic search for *interesting* problems to be *solved* and their explication *published*. (Publication influences compensation in the academic environment of many countries including the United States.) The following discussion exposes the meta-level use of research patterns that are operable *even before the problem identification and (problem) awareness stages* of the general design science research methodology. Indeed, these patterns are constantly applicable in the academic environment and literally shape the awareness stage (Searle 1995).

First, consider the adjectives *interesting* and *published* when applied to the word *problem*. At any point in time, design science research (or any broad type of research) is found interesting and publishable by only a limited number of communities of interest and their journals. Further, design science research itself is applicable only to a certain class of problem domains (defined, somewhat circularly, by the communities that use the paradigm).

The patterns *aligning with a paradigm* and *research domain identification* were first applied in this case *unconsciously* as part of the environmental scanning mode of the researchers. The principal actors were an experienced design science researcher and an apprentice design science researcher. The domains of interest and possible publication outlets were generally known, and the various paradigms applicable to design science research were also familiar at a high level and this information served as a preconscious filter, selecting for conscious attention only design science research opportunities (Gladwell 2005). For example, it would never have occurred to either Vijay (or later to Gary) to pursue an opportunity for research on *organizational structural change following IT deployment* since problems from this area do not "fit" the design science research paradigm nor are they one of the paradigm's research domains.

However, for someone new to design science research, *aligning with a paradigm* and *research domain identification* will be a valuable guide for becoming familiar with design science research methods, problems, and journals. *Aligning with a paradigm, research conversation,* and *research domain identification* each involve extensive, reflective *reading* from the work of a research community and/or direct observation of its work and then consideration of a course of action in light of the values revealed for that community. The patterns explicate and direct the time-honored advice to *become familiar with your research community*. In this case, familiarization with a community had been largely accomplished over a prior period of years and resulted in the *selective perception* (Kunda 1987) used (pre-consciously) to scan the environment for interesting problem domains.

Awareness of Problem

The opportunity that passed the filter put in place by the meta-level use of the two patterns discussed earlier and that initiated this case was a chance to investigate the command and control difficulties that arose in a critically complex environment, nuclear reactors. Following an invitation from a colleague at Georgia Tech, Vijay

and several other Georgia State faculty made several tours of the GT research reactor. Between tours, the following patterns were applied:

- The meta-level patterns discussed earlier, *aligning with a paradigm, research conversation*, and *research domain identification*, were re-applied at a more concrete level. First, it was necessary to establish that the problem set presented by nuclear reactor command and control was amenable to exploration by design science research (*aligning with a paradigm*). This seemed to be so; IT systems with embedded control paradigms and expert system modules were in use, and improvement of these systems constituted a partial solution of the overall problem. Such improvement was definitely part of the DSR paradigm. Notice how the general research direction (IT control system improvement) arose from the use of a very general pattern, very early in the case.
- Next, it was necessary to identify the research domains and specific research conversations implied by the problem domain. This information was identified through focused library research guided by the patterns: *research topic identification*, *problem formulation*, *understanding research community*, and *research conversation*. (Note: Many of these patterns will be used again later as the research problem and its solution become more focused.)

After tours of the reactor facility and follow-up question sessions of reactor personnel by phone and meetings, Vijay had amassed a substantial quantity of information. The patterns *research topic identification*, *complex system analysis*, *problem formulation*, *understanding research community*, *research conversation*, and *research domain identification* were applied to attempt to identify a more specific and more tightly scoped problem. However, there was need to move to the next stage, suggestion, to develop a preliminary solution and to see its shortcomings before a well-scoped research problem could be defined. This shows the need for iteration between the different phases of the research. The use of all patterns in this phase of research is summarized in Table 6.1.

Suggestion

The *suggestion* phase of design science research involves utilizing information gained in scanning the literature and using *brainstorming* pattern to investigate potential avenues of approach to the problem. Following application of the patterns *industry/practice awareness*, *problem space tools and techniques*, and *research community tools and techniques*, it initially seemed that an approach widely successful in other, superficially familiar environments, expert system design, might prove useful in this case. The second major actor in the case, Gary, a doctoral student at Georgia State, was brought into the project to develop an expert system in PROLOG for use in controlling the reactor. However, the application of several other patterns uncovered problems with the first-pass solution.

Table 6.1 Pattern Application during the Awareness of Problem Phase of Research

Patterns Utilized	*Actions Generated*
Aligning with a paradigm (p. 296), research conversation (p. 182), and research domain identification (p. 171)	Using these patterns, a design research opportunity emerged from a serendipitous site visit to an interesting (of and about designed artifacts) site
Research topic identification (p. 173); complex system analysis (p. 178); problem formulation (p. 167); understanding research community (p. 201); research conversation (p. 182); research domain identification (p. 171)	Using these patterns, opportunities for IT-related improvement of the operation of the site were investigated and a preliminary problem determined. The appropriate research community—complex control systems design—was identified
Industry/practice awareness (p. 203); research conversation (p. 182); solution-scope mismatch (p. 185); being visionary (p. 174); problem formulation (p. 167), redefining research problem (p. 169); research topic identification (p. 173)	When applied to what had been discovered of the problem domain given the effort expended to date, these patterns suggested that the domain was ill defined and simply determining a properly scoped ("doable") problem would be challenging. This phase of the project was revisited after developing a preliminary solution in the suggestion phase and a more tightly defined research problem formulated
Bridging research communities (p. 194); research domain identification (p. 171); understanding research community (p. 201); research conversation (p. 182)	Three distinct but interrelated research communities were identified and the literature for the research communities was revisited in a focused manner via the application of these patterns

- *Cost-benefit analysis* is another pattern that can be applied at multiple levels. In this case, after several weeks of development the slow rate of progress on the project became apparent. More detailed analysis of the problem and application of the *solution-scope mismatch* pattern showed that the available methods of expert system development were inadequate to a problem of this scale.
- *Means-ends analysis* showed that the primary problem was not in modeling the control rules, but rather in determining when many apparently similar rules should be applied. The patterns *general solution principle* and *abstracting concepts* helped Vijay and Gary proceed toward a solution that was broader, more generally applicable, and more elegant than would have been likely otherwise. These patterns advise rethinking potential solutions to encompass

more and more aspects of the problem. In this specific case, the patterns caused Vijay and Gary to reject more ad hoc solutions such as the development of an expert system design tool and focus instead on conceiving something that better modeled complex command structures in general.

The problem awareness phase was revisited and using the patterns industry/practice awareness, research conversation, solution-scope mismatch, being visionary, redefining research problem, problem formulation, and research topic identification, the research problem was redefined. At this stage the general problem—how to model, construct, and continuously maintain a support system for the operation of a complex, hierarchical procedure-driven environment with control modeling as the specific research problem—became explicit and remained the focus of the project through its completion. Three interrelated research areas—software engineering, database systems, and knowledge-based systems—were identified as areas that have dealt with modeling complex systems. The problem of bridging these areas was identified using the *bridging research communities'* pattern. The literature for these areas was revisited and analyzed. Table 6.1 includes the additional patterns used in the problem awareness phase.

By this point in the project, both Vijay and Gary felt they were "on to something," that is, they felt the problem was interesting and potentially solvable. This was an intuitive feeling, one of the aspects of research in general that cannot be completely captured with patterns, but that can be partially validated though the use of patterns such as *research conversation* that lead to actions that demonstrate alignment with a community for both problem and solution domains.

Means-ends analysis suggested that rule-based control systems were theoretically appropriate but practically unmanageable with existing techniques. Scrutinizing the results from the effort to date using *cost-benefit analysis*—only a general solution direction had emerged at this point—indicated that even a partial solution to the problem would likely involve considerable work. At this point, Gary decided to pursue the problem as his dissertation topic.

The pattern, *research conversation*, and other literature search patterns were used continuously throughout this phase. This uncovered an interesting in-use technique that seemed promising: frame-based knowledge representation in which multiple, similar aspects of a domain and rules for responding to them were encapsulated in a *frame*. Through the use of the pattern, *sketching solution*, a frame-based approach was investigated as a thought experiment. The use of *complex system analysis* followed by *means-ends analysis* found this approach also lacking; frames, as understood at that time, suffered from the same maintenance problems as simple rule-based systems.

In the course of an extended "gestation period" (also in the authors' experience, a facet of all research), the patterns brainstorming, different perspectives, integrating techniques, embedding concepts and techniques, and combining partial solutions were applied repeatedly to investigate different approaches to the problem. The approaches were evaluated using the patterns sketching solution and

means-ends analysis. After many iterations through solution proposal and solution evaluation spanning several months, the application of object-oriented programming techniques seemed promising; if rules could be encapsulated in objects, then the inheritance and especially the reuse capabilities of the object-oriented (OO) paradigm could possibly be leveraged to ameliorate the problem of rule maintenance that characterized the problem domain. More thought experiments guided by the patterns sketching solution and means-ends analysis showed this approach, a direct result of applying the patterns integrating techniques and combining partial solutions, to be promising. Having identified an approach to the problem, the next phase of the research cycle, development, was initiated. Table 6.2 summarizes the patterns applied and resulting actions during the suggestion phase.

Development

Development involves in-depth exploration, development and assessment of a solution direction. It requires repeated suggestion of methods for *specifically how* to accomplish the solution, turning a solution direction into a solution-in-fact. It is necessarily iterative for almost all non-trivial problems because large, imperfectly understood problems (by definition, the "interesting" problems for researchers) are multi-faceted with each facet typically explored and tentatively solved in turn. However, all facets must integrate into a coherent whole if they are to provide an acceptable solution and so backtracking to reassess a prior partial solution that impedes the solution of another facet is common.

Since the chosen approach was a synthesis of rule-based and object-oriented programming (from the patterns, *integrating techniques* and *elegant design*), the preliminary design step seemed obvious: substitute rules for programmed methods in a novel, object-oriented language. The two patterns key to this design stage, *sketching solution* and *means-ends analysis*, when applied to this facet of the solution showed it to be feasible but inadequate. The ability to inherit from previously defined objects offered some improvement over frames in maintaining large rule sets, but was still, as shown by thought experiments that "walked through" the use of these novel constructs, insufficient to manage the scale of the maintenance problem. The solution direction still seemed promising however, especially since application of the pattern *hierarchical design* showed that the overall complexity of the environment could be partitioned and modeled—and potentially controlled—through this approach.

An aspect of design (and design science research) that is shared by any creative endeavor is the manner in which a continuing focus on a problem changes the perception of the issue. It becomes clearer and better articulated even as, or perhaps because, multiple solutions have been attempted and have been discarded. Each attempt broadens the conceptual vocabulary that can be used in the problem description. Application of the pattern *using human roles* was natural to the project at this point since it is a common technique of object-oriented design—describe in detail what a human would do to solve the problem and then use the *being visionary* pattern

Table 6.2 Pattern Application during the Suggestion Phase of Research

Patterns Utilized	*Actions Generated*
Industry/practice awareness (p. 203); problem space tools and techniques (p. 226); research community tools and techniques (p. 227)	When applied to what had been discovered of the problem domain given the effort expended to date, these patterns suggested that the domain was ill defined and simply determining a properly scoped ("doable") problem would be challenging
Brainstorming (p. 156); research conversation (p. 182); complex system analysis (p. 178)	These patterns were used to cycle through potentially interesting aspects of the total problem space using ongoing research conversations to suggest approaches
Cost-benefit analysis (p. 179); solution-scope mismatch (p. 185); means/ends analysis (p. 236); general solution principle (p. 253); abstracting concepts (p. 249)	These patterns made clear the shortcomings of the preliminary solution and helped the more general and more interesting (to the research community) problem to emerge
Sketching solution (p. 264); research conversation (p. 182); complex system analysis (p. 178); means/ends analysis (p. 236)	These patterns permitted (relatively) rapid development of and evaluation of approaches to the general problem of control of a complex, rapidly evolving environment
Brainstorming (p. 156); different perspectives (p. 229); integrating techniques (p. 269); embedding concepts and techniques (p. 268); combining partial solutions (p. 266); sketching solution (p. 264); means/ends analysis (p. 236)	These patterns were responsible for the synthesis of rule-based systems with object-oriented concepts and the evaluation of this combined approach.

to conceive an automated approach to the human activities. The current, human approach was to use experience and judgment to select from the huge rule set an immediately applicable subset for different, superficially similar situations. At some point, the OO conception of "letting the objects direct themselves" became prominent and the most distinguishing feature of smart objects started to emerge: the use of a "judgment" or meta-level within and among objects containing rule sets to automate the selection of appropriate lower level rules. Essentially, higher level rules would simulate human intervention to automate the contextual selection of lower level rule sets to be activated. The details of how the higher and lower level rules would interact

was far from clear at this point; however, the broad applicability of the functional specification of this capability was powerful and elegant (*elegant design*).

Although the use of meta-level rules to guide rule-set selection had been suggested in the literature (continuing application of the patterns *understanding research community*, *interdisciplinary solution extrapolation*, and *research conversation*), the application and expansion of OO techniques provided a superior partitioning of and execution scheme for high-level rules than any technique yet discovered by Vijay and Gary in their literature search. This tentative conclusion of the superiority of the new technique was, of course, immediately subjected to the patterns *sketching solution* and *means-ends analysis*. Use of the *approaches for building theory* and *hypothetical/deductive approach* patterns helped in developing and formalizing the theory developed. Table 6.3 summarizes the patterns applied and resulting actions during the development phase.

Table 6.3 Pattern Application during the Development Phase of Research

Patterns Utilized	*Actions Generated*
Integrating techniques (p. 269); elegant design (p. 251)	Suggested the synthesis of object-oriented and rule-based programming (smart objects) as a concrete means of solving the research problem
Sketching solution (p. 264); means/ends analysis (p. 236)	Use of these patterns (1) developed the smart object synthesis more explicitly and (2) determined that the development was leading toward results at an acceptable pace
Hierarchical design (p. 262)	This pattern suggested still more elaboration of the smart object concept
Using human roles (p. 247); being visionary (p. 174)	When combined with hierarchical design, these patterns resulted in the conceptualization of one of the key aspects of the final smart object paradigm, the incorporation of a meta-level of supervisory rules to simulate the human intervention required by conventional solutions of the research problem
Elegant design (p. 251); research conversation (p. 182); sketching solution (p. 264); means/ends analysis (p. 236); interdisciplinary solution extrapolation (p. 233); hypothetical/deductive approach (p. 221)	Application of these patterns (1) confirmed that the smart object paradigm was a unique contribution and (2) that it did in fact provide a solution to the general research problem, and in creating and presenting the theory developed

Following the generation of a specific approach to the problem (a meta-level rule interpreter as a common part of all smart objects) came months of even lower level implementation work to articulate the general constructs into software design modules capable of being implemented in an existing OO language.

Evaluation

As discussed in "An Example of ICT Design Science Research" (Chapter 2), the micro-evaluation of aspects of a design takes place almost constantly during the *suggestion* and *development* phases of the research cycle. However, in the *evaluation* phase of the cycle the goal is a macro-evaluation/validation of the entire designed artifact.

The smart object concept had many *theoretical* benefits and had been proven feasible in the prior phase of the research cycle. Now it was time to empirically explore whether or not the design actually realized the theoretical benefits claimed for it. Design science research is sometimes criticized for its lack of empirical validation. In this case, Gary and Vijay were sensitive to this criticism and spent considerable time, guided by multiple patterns, to find an evaluation process for smart objects that was both rigorous enough to demonstrate the value of the concept and yet achievable with the resources available.

The pattern *technological approach exemplars* led to another review of the problem domain literature, this time focused on discovering what validation techniques were used by the chosen research community. This information would not absolutely constrain the direction taken, but would definitely influence it; it is widely understood that straying beyond the techniques commonly employed by a research community increases the difficulty of publishing in that community (Murray 2005). It was discovered that literally all the techniques of evaluation and validation explored by the patterns of Chapter 11 had been applied to different published design science research efforts and were acceptable to the design science research community. Thus, for this case the actual choice of validation technique and its scope would have to be determined by the specifics of the project and by still another application of the pattern: *cost-benefit analysis*.

A consideration of the pattern *mathematical proofs* made it apparent that formal proof was not applicable to smart objects. The artifact was a conceptual design from a potentially infinite design space and no approach to optimization has yet been developed. Likewise, preliminary use of the pattern *using metrics* showed metrics to be inapplicable since no formal metrics existed for evaluating the control of large, complex systems. Further, *experimentation*, involving by definition the comparison of two or more control techniques when applied to the same environment, was eliminated as impractical given the resources available for the project. The reasoning involved in this decision merits further discussion:

As was discussed in "An Example of ICT Design Science Research" (Chapter 2), researchers from other paradigms sometimes find the degree of validation applied to designed science research artifacts simplistic. This is due to the lack of understanding

Table 6.4. Pattern Application during the Evaluation Phase of Research

Patterns Utilized	*Actions Generated*
Technological approach exemplars (p. 245)	This pattern guided the researchers toward validation techniques that were acceptable to the research community
Mathematical proofs (p. 289); using metrics (p. 291)	These patterns were used in a via negativa—a preliminary application demonstrated that the smart object paradigm was not amenable to these validation techniques
Technological approach exemplars (p. 245); demonstration (p. 283); simulation (p. 290); logical reasoning (p. 287)	Technological approach exemplars served as a meta-level pattern suggesting a validation strategy that incorporated the synergistic application of multiple patterns in validating the research
Cost-benefit analysis (p. 179)	This pattern was applied specifically to the validation strategy as it emerged to ensure that resources were not exceeded. An artful balance was called for in creating a validation that satisfied technological approach exemplars (and thus made publication easier) and the eternal problem of resource limitations

of the extraordinary difficulty of "full-scale" validation of a complex artifact, understanding that typically comes only from experience or long and close observation of the design science research process.

In this case a "full-scale" test of the smart object paradigm as applied to a nuclear reactor environment would have involved man-years of effort, broken out as follows: First a smart object interpreter or compiler would have needed to have been constructed and tested; the effort for this process alone was known from experience to be in on the order of man-years. Next, a full-scale control system would have needed to have been designed and programmed in the new interpreter/compiler. Finally, the control system would have needed to have been installed at the nuclear reactor facility, the full staff trained in its use, and the system operated over a period of time with extensive measurements and observations taken of all processes. Even a cursory application of the *cost-benefit analysis* pattern indicated that the amount of effort required for a full-scale test of the smart object paradigm could not be justified for a single dissertation and the possibility of a few publications.

Fortunately the research communities of interest in this case were all design science research oriented and were amenable to more modest forms of validation than full scale testing. Though some forms of validation had been ruled out by the nature of the project, use of *technological approach exemplars* led to the conjunction

of the use of the evaluation patterns *demonstration*, *simulation*, and *logical reasoning*. This yielded a validation strategy that was essentially an extended demonstration of the operation of smart objects at a logical level. The strategy had two stages: first the functional logic description of a smart object execution engine was reasoned to have the attributes claimed for the smart object design. The second stage involved an extended "walk through" or detailed step-by-step explication of the logical operation of a smart object design for a simple robot executing the task of bagging groceries. Successful operation of this exercise would, at least for the research community consisting of Gary's dissertation committee, constitute proof of concept of a novel, useful advance in knowledge. Table 6.4 summarizes the patterns applied and resulting actions during the *evaluation* phase.

Conclusion

For this case, the artifact that emerged from the general design science research cycle was an academic *research project*. Thus, the traditional goal for the *conclusion* phase of the design science research cycle for this case is dissemination of the results of the project through published papers. The primary problems encountered in publishing are convincing the members of a research community who have been selected to review the papers sent to a particular venue that the results are (1) interesting, (2) novel, and (3) accessible to the community, that is, well and clearly articulated.

To a great degree the issue of *interest* depends on how well directed the research effort was by the patterns *research domain identification*, *research conversation*, and more generally *industry/practice awareness*. Research communities as communities of interest by definition have a highly focused awareness. Topics outside the traditional core interests for a research community are frequently rejected as either uninteresting or inappropriate.

Both *novelty and significance* in publication are addressed by the pattern of the same name. This pattern suggests ways to increase the salience of a contribution to the research community. The pattern *aligning with a paradigm* also makes suggestions on how to make a research presentation appear consonant with the problems found interesting and the techniques found acceptable to a given research community.

At the end of this phase, Gary's dissertation (Buchanan 1991) had been composed and successfully defended, and papers based on the dissertation had been accepted by two conferences, Buchanan et al. (1990) and Vaishnavi et al. (1993), and a related paper in a third conference (Kuechler et al. 1995). The choice of conferences was strongly directed by "community alignment" patterns mentioned earlier. All papers made extensive *use of examples* to make concrete and more understandable the novel abstraction of smart objects. A fourth publication, a submission to the journal *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, was also generated in this phase (Vaishnavi et al. 1997). The choice of journal was made only after much deliberation guided by the publishing patterns in Chapter 12 of this book, and several "walk throughs" during which preliminary sketches of a paper

Table 6.5 Pattern Application during the Conclusion Phase of Research

Patterns Utilized	*Actions Generated*
Research domain identification (p. 171); research conversation (p. 182); industry/ practice awareness (p. 203)	These patterns, when successfully applied at earlier phases in the research cycle, align the research effort with the terminology and practice of a research community and make publication of both conference and journal papers easier.
Aligning with a paradigm (p. 296)	In this phase of the research cycle, this pattern was invoked again to determine exactly which one of several similar paradigms (almost but not quite equated with specific journals) to choose to submit results to.
Style exemplars (p. 301); novelty and significance (p. 299)	These patterns assist in focusing on a specific journal and in making salient in the presentation of research results the novelty and significance of the research. Unless the reviewers perceived both attributes in the research, it will be difficult to publish.
Writing conference papers (p. 303); use of examples (p. 302)	Application of these patterns assists in successful conference paper preparation.
Writing journal papers (p. 305); use of examples (p. 302)	Application of these patterns assists in successful journal paper preparation.

were made and evaluated according to the paradigm(s) apparent in a given journal's editorial statement and in exemplar papers published in recent issues of the journal. The ultimate choice of TKDE was made when the pattern *style exemplars* led to an understanding that the results of the smart object project could be structured very similarly to several published exemplars in that journal. Table 6.5 summarizes the patterns applied and resulting actions during the conclusion phase.

This concludes the smart object paradigm use case; however, as stated in "An Example of ICT Design Science Research" (Chapter 2), several other research projects based on the smart object paradigm were conducted and resulted in a stream of published work extending over a period of 8+ years.

Practice, Practice, Practice

While the use of the patterns in this book will assist in the solution of many of the problems encountered in a design science research effort, patterns are *not*

rules and the method we have outlined for their use is simply a guideline from the experience of many design science researchers. There is still no other way to fully understand design science research than to conduct it. We suggest the reader review "An Example of ICT Design Science Research" (Chapter 2), the textual description of the smart object project, and then, guided by the appropriate patterns and ongoing reference to this chapter—the "how to" section of this book—plunge into the *awareness of problem* phase for their own design science research project. Good luck!

References

Alexander, C. (1964). *Notes on the Synthesis of Form*. Cambridge, MA: Harvard University Press.

Altschuller, G.S. (2005). *The Innovation Algorithm: TRIZ, Systematic Innovation and Technical Creativity*. Worcester, MA: Technical Innovation Center. ISBN 0-9640740-4-4.

Buchanan, G., Vaishnavi, V., and Nevins, A. (1990). "Modeling Operations Management Support Systems." *Proceedings of the IEEE International Symposium on Man, Machine and Cybernetics*, 134–136.

Buchanan, G. (1991). *Modeling Operations Management Support Systems*. Unpublished Doctoral Dissertation, Atlanta, GA: College of Business Administration, Georgia State University.

Firebaugh, M.W. (1988). *Artificial Intelligence*. Boston, MA: Boyd & Fraser Publishing Company.

Gericke, A. (2009). "Problem Solving Patterns in Design Science Research—Learning from Engineering." *ECIS 2009 Proceedings*.

Gladwell, M. (2005). *Blink: The Power of Thinking Without Thinking.* New York, NY: Little Brown and Company.

Kuechler, W.L., Lim, N., and Vaishnavi, V.K. (1995). "A Smart Object Approach to Hybrid Knowledge Representation and Reasoning Strategies." *Proceedings of the 28th Hawaiian International Conference on Systems Sciences*, 33–41.

Kunda, Z. (1987). "Motivated Inference: Self-serving Generation and Evaluation of Causal Theories." *Journal of Personal and Social Psychology* 53, 669–679.

Murray, R. (2005). *Writing for Academic Journals*. Maidenhead, NY: Open University Press.

Searle, J. (1995). *The Construction of Social Reality*. New York, NY: Free Press.

Vaishnavi, V., Buchanan, G., and Nevins, A. (1993). "Smart Objects: A Tool for Building Intelligent Support Systems." *Proceedings of the 26th Hawaiian International Conference on Systems Sciences*, 93–102.

Vaishnavi, V., Buchanan, G., and Kuechler, W. (1997). "A Data/Knowledge Paradigm for the Modeling and Design of Operations Support Systems." *IEEE Transactions on Knowledge and Data Engineering* 9(2): 275–291.

Vaishnavi, V. and Kuechler, W. (2007). *Design Science Research Methods and Patterns*, Boca Raton, New York: Auerbach Publications.

Wikipedia-TRIZ (2014). "TRIZ." URL: http://en.wikipedia.org/wiki/TRIZ (last accessed on January 29, 2015).

Appendix 6A The TRIZ Inventive Principles

Table 6A.1 The 40 TRIZ Inventive Principles

No.	*TRIZ Inventive Principle*	*Description*
P1	Segmentation[a]	Divide an object into independent parts.
P2	Removal/ extraction	Remove the interfering part or property from an object or extract only the necessary part or property from an object.
P3	Local quality[a]	Change the structure of an object from homogeneous to heterogeneous. Make each part of an object carry out a different function. Each part of an object should function under conditions that are most suitable for its operation.
P4	Asymmetry	Change the shape of an object from symmetrical to asymmetrical or increase its degree of asymmetry if it is already asymmetrical.
P5	Merging/ consolidation	Bring closer or merge identical or similar parts; assemble identical or similar parts to perform parallel operations. Make operations contiguous or parallel; bring them together in time.
P6	Universality	Make a part or object perform multiple functions so as to eliminate the need for other parts or objects.
P7	Nesting[a]	Place each object inside another object recursively.
P8	Counterweight	In order to compensate for the weight of an object, merge it with another object to provide a lift or make the object interact with the environment to utilize aerodynamic, hydrodynamic, and other forces.
P9	Counter action[b]	Replace an action with both harmful and beneficial effects with counter actions to control harmful effects. Create beforehand stresses in an object that will oppose known undesirable working stresses later on.
P10	Prior action[b]	Perform beforehand the required change of an object either partially or fully. Pre-arrange objects such that they can come into play from the most convenient place and without losing time for their delivery.

(*Continued*)

Table 6A.1 The 40 TRIZ Inventive Principles (*continued*)

No.	*TRIZ Inventive Principle*	*Description*
P11	Cushioning in advance[b]	Prepare emergency means in advance to compensate for relatively low reliability of an object.
P12	Equipotentiality	Limit potential changes in a potential field (e.g., in a gravitational field, change operating conditions to eliminate the need for raising or lowering objects).
P13	Doing the opposite	Invert the actions(s) used to solve a problem (e.g., heat an object instead of cooling it). Make movable parts (or the external environment) fixed and the fixed parts movable. Turn the object or process upside down.
P14	Spheroidality, curvature	Use curvilinear parts, surfaces, or forms instead of rectilinear ones. Go from linear to rotary motion; use centrifugal forces.
P15	Dynamics	Divide an object into parts that are capable of movement relative to each other. If an object or process is rigid or inflexible, make it movable or adaptive.
P16	Partial or excessive action[a]	If it is hard to fully solve a problem using a method, then the problem may be easier to solve by using more or less the same method.
P17	Another dimension[a]	Move an object in two- or three-dimensional space. Use a multi-storey arrangement instead of a single-storey arrangement. Tilt or reorient the object, lay it on its side. Use "another side" of a given area.
P18	Mechanical vibration	Make an object oscillate or vibrate; increase its frequency (even up to the ultrasonic level). Use an object's resonant frequency. Use piezoelectric vibrators instead of mechanical ones. Use combined ultrasonic and electromagnetic field oscillations.
P19	Periodic action[a]	Use periodic or pulsating actions instead of continuous ones. If an action is already periodic, change the periodic magnitude or frequency. Use pauses between periods to perform a different action.
P20	Continuity of useful action[a]	Carry on useful action continuously; make all parts of an object work at full load, all the time. Eliminate all idle or intermittent actions or work.

(*Continued*)

Table 6A.1 The 40 TRIZ Inventive Principles (*continued*)

No.	*TRIZ Inventive Principle*	*Description*
P21	Rushing through[b]	Conduct a process or certain stages of the process (e.g., destructible, harmful, or hazardous operations) at high speed.
P22	Converting harm into benefit[a]	Use harmful factors (particularly, harmful effects of the environment or surroundings) to achieve a beneficial effect.
P23	Feedback	Introduce feedback (referring back, cross-checking) to improve a process or action.
P24	Intermediary[a]	Use an intermediary carrier article or intermediary process.
P25	Self-service[b]	Make an object serve itself by letting it perform auxiliary helpful functions. Use waste resources, energy, or substances.
P26	Copying	Instead of an unavailable, expensive, or fragile object, use simpler and inexpensive copies. Replace an object or process with optical copies; if visible optical copies are already used, move to infrared or ultraviolet copies.
P27	Cheap short-living objects[b]	Replace an expensive object with a multiple of inexpensive objects with certain qualities such as service life.
P28	Mechanical substitution	Replace a mechanical means with a sensory (optical, acoustic, taste, or smell) means. Change from static to movable fields, from unstructured fields to those having structure.
P29	Pneumatics/ hydraulics	Use gas and liquid parts of an object instead of solid parts (e.g., inflatable, filled with liquids, air cushion, hydrostatic, hydro-reactive).
P30	Flexible shells/ thin films	Use flexible shells and thin films instead of three-dimensional structures.
P31	Porous materials	Make an object porous or add porous elements such as inserts, coatings, and so on. If an object is already porous, use the pores to introduce a useful substance or function.

(*Continued*)

Table 6A.1 The 40 TRIZ Inventive Principles (*continued*)

No.	*TRIZ Inventive Principle*	*Description*
P32	Color changes[b]	Change the color or transparency of an object or its external environment.
P33	Homogeneity	Make objects interact with a given object of the same material (or material with identical properties).
P34	Discarding/ recovering	Discard portions of an object that have fulfilled their functions (by dissolving, evaporating, and so on.) or modify them directly during operations. Conversely, restore consumable parts of an object directly during operation.
P35	Parameter changes	Change an object's parameters such as its physical state (e.g., to gas, liquid, or solid state), concentration or consistency, degree of flexibility, temperature.
P36	Phase transitions	Utilize phenomena occurring during phase transitions (e.g., volume changes, loss or absorption of heat, and so on.).
P37	Thermal expansion	Use thermal expansion (or contraction) of materials. In case of thermal expansion, use multiple materials with different coefficients of thermal expansion.
P38	Strong oxidants[b]	Replace common air with oxygen-enriched air, pure oxygen, ozonized oxygen, or ionized oxygen. Replace ozonized (or ionized oxygen) with ozone.
P39	Inert atmosphere	Replace a normal environment with an inert one. Add neutral parts or inert additives to an object.
P40	Composite materials	Change from uniform to composite (multiple) materials.

Source: Adapted from Altschuller, G.S., *The Innovation Algorithm: TRIZ, Systematic Innovation and Technical Creativity*, 2005; Gericke, A., *ECIS*, 2009.

Note: The principles in shaded rows do not seem to have applicability to DSR.

[a] There is a DSR pattern(s) that is or can be considered to be an adaptation of the principle.

[b] There is a DSR pattern(s) that is related to the principle.

Chapter 7

Creativity Patterns

Creativity is an integral part of all intellectual endeavors and is critically needed in all areas of research. However, it is a "soft" and poorly understood cognitive skill and is universally acknowledged to be difficult or impossible to teach! Fortunately, most individuals are amply creative if only the skill can be enhanced, focused, and directed; directing attention is precisely where patterns excel.

Use the following seven patterns to grow and harness your creative energies for the conduct of your research:

Enhancement Type

Utilization Type

The patterns have been classified according to two types: *enhancement* and *utilization*. The *enhancement* type patterns help in enhancing our creative abilities. The *utilization* type patterns help in utilizing the creativity abilities that we already have. The patterns within each type are listed in alphabetic order. The patterns of the same type are obviously related and so are not listed as related pattern(s) for a particular pattern.

All the *utilization* patterns in this chapter are meta-level patterns; they are applicable at any point in a research effort when you are facing a problem or situation that needs an inventive solution. In this and subsequent chapters, meta-level patterns are indicated by the superscript ^{M} preceding the pattern name.

Enhancement Type Patterns

- Meditation
- Stimulating creativity

Meditation describes how we can gradually increase our creative abilities. *Stimulating creativity* describes the conditions that seem to stimulate creativity.

Meditation

Type

Enhancement

Intent

Gradually unlock the creativity that is already a part of you.

Motivation

To enhance our creativity, we need to strengthen our connection with our unconscious mind so that we can utilize it for enhancing our creativity. This is similar to the need for regular physical exercise to strengthen our physical body. With regular meditation, we can gradually remove any hindrances to tapping into our reservoir of creativity.

Context/Applicability

You would like to gradually enhance your creative abilities so that they can be used in research when needed.

Description

1. Sit down in a relaxed position and close your eyes.
2. Feel the location and environment, and start relaxing.
3. Do not control your thoughts. Let the thoughts come to your mind freely but do not pay any attention to them. Gradually the thoughts will start diminishing.
4. The gap between thoughts will gradually expand, eventually reaching a state of thoughtlessness—the state of meditation.

Notes

1. One needs to practice meditation regularly for a long time to learn it.
2. There are many techniques for meditation including the ones that focus on one's breath and the constant repetition of a word—*mantra*. The above technique taps into a person's inner desire to seek true happiness. This technique may not be the most appropriate technique for you and so experimenting with other techniques may be needed.

Consequences

The regular practice of meditation will help in removing the hindrances in tapping your creativity.

Stimulating Creativity

Type

Enhancement

Intent

Create conditions for stimulating your creativity, which may otherwise remain dormant.

Motivation

In addition to working on our long-term growth in the creativity dimension (using the *mediation* pattern), we would like to know the conditions that can be used as operational principles to enhance our creative abilities.

Context/Applicability

You would like to realize the full potential of using your creativity in the pursuit of your research.

Description

Ladd (1987) lists the following conditions that seem to stimulate unconscious mental processes:

1. *Doubt:* Having or developing a trait for doubting the validity of assumptions that we routinely make and venturing to resolve the doubts is helpful in creating the need for new ideas.
2. *Venturesome Attitude:* A degree of research entrepreneurship is needed for a person to delve into the unknown. One needs to be ready to take risks and not to be afraid of mistakes.
3. *Tolerance for Uncertainty:* New ideas or insights are many times fragmentary and even contradictory. One thus needs to have tolerance for uncertainty in order to nurture the creation of new ideas.
4. *Diversity:* A creative idea is often a connection between ideas or concepts that were not connected before. Diversity of interests and experiences is thus helpful to growth and the productive use of the unconscious mind.
5. *Thorough Preparation:* Thorough preparation is one of the stages of inventive process (see *Stages of Inventive Process* pattern). It is, however, not enough to think hard on the problem to fulfill this step in the inventive process. One needs to do whatever one can possibly do consciously to make progress in the solution of the problem. This includes the proper formulation of the problem. One cannot expect a solution from the unconscious mind when the problem is either not formulated at all or is formulated poorly.
6. *Tension:* An intense desire to find a solution is a strong stimulus to the unconscious mind. It is thus helpful to reach a state of tension, where finding a solution is utterly important.
7. *Temporary Abandonment:* This corresponds to the stage of incubation in the inventive process (see *Stages of Inventive Process* pattern). Developing the habit of consciously abandoning a problem when one is burning to find the solution is thus very important for tapping the unconscious energies. This habit needs to be learned for becoming creative.
8. *Writing:* Writing is often considered as the laborious chore that must be conducted after the fun of invention is over. The process of writing itself can, however, be a source of new ideas and a way of communicating with the unconscious mind. Writing clarifies ideas and leads to new ideas. It is thus useful too in the creative process.
9. *Exchange with Colleagues:* Exchange of ideas with colleagues needs the verbalization of ideas. That itself is helpful in the creative process because it leads to the translation of thoughts from the unconscious mind to the consciousness mind.
10. *Freedom from Distraction:* It takes effort to "start the engine of the unconscious mind" and once it has started, one can be in a productive mood. It is thus useful to have stretches of time that are free from other distractions.

11. *Sensitivity to Similarities:* The ability to see analogies and to see similarities between seemingly dissimilar things is a tool that can promote the creative association of ideas and concepts. The ability to abstract out the differences to see the similarities at a certain level of abstraction is also a useful trait for creativity.
12. *Capturing Intuitions:* A productive inventive process needs a two-way flow of messages between the conscious mind and the unconscious mind. Intuitions are the messages that the unconscious mind sends to the conscious mind. These intuitions need to be captured as and when they occur. Capturing intuitions and using them also makes one better receptive to the unconscious mind, which promotes more intuitions.
13. *Combinations:* A rich variety of conditions listed earlier is stimulating to the unconscious processes. It is thus useful to intersperse writing with temporary abandonment, and so on, to provide a rich environment for creative processes.

Consequence

The use and promotion of conditions discussed in this pattern can, over a period of time, can improve your creativity.

Related Pattern(s)

Periodic Work (p. 158) is a complementary pattern.

Utilization Type Patterns

- [M]Brainstorming
- [M]Changing attitude
- [M]Periodic work
- [M]Stages of inventive process
- [M]Wild combinations

Brainstorming describes the process of coming up with new ideas or concepts. The *changing attitude* pattern provides a way to invoke the use of creativity. *Periodic work* shows how to give the unconscious mind time to maximize its contribution. The *stages of inventive process* pattern describes the stages that one goes through in the inventive process. *Wild combinations* describe the consideration of combination of solution elements that may not be logically related.

[M]Brainstorming

Type

Enhancement

Intent

Generate a new idea or concept by first generating a large number of ideas that then are evaluated for their merit.

Motivation

While attempting to come up with a new idea individually or in a group, we often find it difficult to come up with a novel idea. This is because we are too judgmental and kill ideas even before they are born. Therefore, a process is needed that can overcome this limitation.

Context/Applicability

This pattern can be used at an individual level or applied by a group of people who would collectively like to generate a novel idea. The premise is that new ideas need to emerge and be nurtured to assess their value before they get killed through our sense of discrimination.

Description

Brainstorming (originally developed by an advertising executive in 1939 (Osborne 1963)) is a process for generating a novel idea that can be used by an individual or a group of people. The process is divided into two distinct phases:

1. Generate and record as many ideas as is possible. All ideas and particularly the dumb ideas are very welcome. This is an attempt to tap our unconscious resources to create ideas that would normally get killed before they are born because of our individual or societal sense of "goodness."
2. Evaluate the ideas that have been generated in the first phase for their appropriateness and usefulness.

The separation of the two phases and the deliberate welcoming of "all" ideas is key to the success of the process.

Note

Some research studies (Diehl and Stroebe 1987, 1991) suggest that group brainstorming could generate fewer ideas than individuals of the group working alone.

Several factors can contribute to a loss of effectiveness in group brainstorming (Wikipedia-Brainstorming 2015).

Consequences

The pattern is routinely applied by groups of people in organizational settings to generate new ideas for products. The pattern can also be applied to generate new research ideas by a group of researchers or even by an individual researcher. Individual brainstorming has been shown to be superior to group brainstorming for creative writing (Furnham and Yazdanpanahi 1995; Wikipedia-Brainstorming 2015); this may also be applicable to the use of brainstorming in research.

Usage Example(s)

Vaishnavi et al. (1997) utilize brainstorming to envision the attribute set for an ideal operations support system; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

[M]Changing Attitude

Type

Utilization

Intent

Spur the use of creativity in the research work.

Motivation

Changing attitude—looking at a problem or solution in a different light—is a way to fire up our creative energies. We need to utilize this and other similar techniques to get us out of our usual and constrained thinking modes.

Context/Applicability

You are stuck at a certain stage of your research. You would like to utilize your creativity to make progress.

Description

Look at the issue/problem in a completely different manner, from a different angle. This may mean changing your attitude toward prior research or certain approach

to research. This, along with the use of other creativity patterns, may spur your creativity and you may see the problem in a different light.

Consequences

The pattern can help you in taking a break from the current direction of thinking and open up a new direction.

Connection to TRIZ Inventive Principles

This pattern is related to the TRIZ Inventive Principle (Altschuller 2005; Gericke 2009), P32: *Color changes* (see Table 6A.1 in Chapter 6), which suggests changing the color or transparency of an object or its external environment. The pattern, correspondingly, suggests looking at the research problem or its solution from a new angle.

Related Pattern(s)

The *suggestion and development* pattern, *[M]different perspectives* (p. 229), is a complementary pattern.

[M]Periodic Work

Type

Utilization

Intent

Schedule the research work to maximize the use of creativity.

Motivation

In order for creativity to play a role in our research project, we need to utilize the work and contribution of our unconscious mind on the project. For that we need to give the unconscious mind time to work on the project. This would imply that we do not consciously work on the same project continuously but work in a periodic manner.

Context/Applicability

You have started working on a research project that will need the full use of your creativity. You would like to schedule the work on the project in such a manner that the use of your creativity in the project is facilitated.

Description

Schedule your time for doing different types of work so that you do not have to do the same type of work continuously. This way you will provide your unconscious mind time to contribute to the research work when you are not doing the work consciously. This could be done simply by scheduling your time such that your work on the current phase of the project is periodic instead of continuous.

Consequences

The work on the research project becomes a collaborative effort between your conscious mind and your unconscious mind.

Connection to TRIZ Inventive Principles

This pattern is an adaptation of the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P19: *periodic action* (see Table 6A.1 in Chapter 6), to the design science research domain; P19 suggests the performance of periodic or pulsating action instead of a continuous one. The connection provides an indirect validation of the pattern since the TRIZ principles are derived from the study and analysis of a vast number of patents in the engineering domain.

Related Pattern(s)

Stimulating creativity (p. 153) is a complementary pattern.

ᴹ*Stages of Inventive Process*

Type

Utilization

Intent

Understand and apply the creative (inventive) process.

Motivation

When we are stuck with some problem and need to utilize our creativity for solution of the problem, we need to know the process that we should follow to actively involve our unconscious mind for the creative solution of the problem. This pattern describes such a process. A crucial step of the process is the communication of the problem to our unconscious mind.

Context/Applicability

You are at the solution development or some other stage of your research where conscious logical thinking is not sufficient to make progress. This pattern will assist in tapping into the unconscious creative processes.

Description

The inventive process consists of six stages (Hadamard 1954; Ladd 1987; Wallas 1926):

- Interest
- Preparation
- Incubation
- Illumination
- Verification
- Exploitation

Interest: You need to have a strong interest in solving the problem if you would like to tap into your unconscious creative energies. There are two reasons for this. First, you cannot devote the time and energy needed for solving the problem unless you have a strong interest in the problem. Second, you are unlikely to be able to enlist the support of your unconscious mind in the solution process unless you have a strong interest in the solution of your problem.

Preparation: There is no direct way of communicating with your unconscious mind. Preparation is a necessary stage for "communicating" the problem to your unconscious mind and involves the use of all the conscious means that you have available for attempting to solve the problem. Hard work and a degree of physical tiredness and frustration seem to stimulate unconscious energies.

Incubation: This is a stage of unconscious mental activity. In this stage you need to "sleep over" the problem. You should refrain from consciously thinking over the problem.

Illumination: This is the stage in which the unconscious mind communicates with the conscious mind. Just as the preparation stage is a vehicle for sending messages from the conscious mind to the unconscious mind, the illumination stage involves messages being sent from the unconscious mind to the conscious mind. This stage has also been called "sudden enlightenment or comprehension" (Beveridge 1957).

Verification: This stage involves conscious voluntary activity just as the preparation stage. The activities in this stage include the expression of the solution in precise terms and the testing of the validity of the solution offered by the illumination stage using logic and existing knowledge.

Exploitation: This is the last stage in which the work of the previous stages is put to productive use.

Even though there is a progression from one stage to the other as discussed above, the stages do not necessarily follow each other in a strict sequence. One may iterate through one of more of these stages before moving on to the next stage. For example, the verification stage may show the inadequacy of the solution made available by the illumination stage. This serves as preparation for going back to the incubation stage.

Consequences

The practice in the use of this pattern will help you to harness your creative energies in the solution of any problem that you may be facing while conducting your research.

Usage Example(s)

Hadamard (1954) describes the use of the inventive process by Henri Poincare, a famous mathematician, in the invention of Fuchsian functions. Poincare had found one class of such functions. He knew that these functions constituted only a special case and his problem was to find the most general form of such functions. He applied persistent conscious effort (preparation stage) that helped him to define the problem better but the solution he was seeking still evaded him. He eventually found the solution unexpectedly while serving in the army.

ᴹ*Wild Combinations*

Type

Utilization

Intent

Find an unconventional solution to a problem by considering a wild combination of ideas.

Motivation

Through our upbringing, we have developed a strong sense of discrimination at an individual and societal level. This is very useful to us from both individual and societal standpoints; it prevents us from committing mistakes. This sense of discrimination is, however, detrimental to letting us consider unusual and unconventional ideas. This pattern encourages us to let such ideas be born and considered.

Context/Applicability

You are trying to solve a problem or improve an existing solution to a problem. Logical/conventional ideas do not seem to lead to the desired solution. You are now trying to see if the unconventional and wild use of ideas can break the impasse.

Description

1. Combine existing ideas in a wild and unconventional way to produce a large collection of ideas. The trick is to lower your discriminatory guard so that you can think of novel ways of combining ideas possibly from seemingly unrelated fields.
2. Select the best of these combined ideas. The number of combined ideas can be very large and it may not be possible to find the more promising of these ideas using conscious logical thinking. The unconscious mind needs to be tapped to find at least the promising ideas that then can be analyzed further at the conscious level. An extended experience in the area helps one in developing an aesthetic sense that guides the selection of the promising ideas using the unconscious mind.

Consequences

The use of this pattern can help you in moving out of the conventional mold and in thinking of novel combination of ideas for solving a problem.

Usage Example(s)

1. The invention of the IBM typewriter was the result of thinking in an unconventional way. In a standard typewriter prior to the introduction of the IBM Selectric, the keys were stationary and the paper moved. The novelty of the IBM typewriter seems to have been the opposite—to "let the stationary things move and the moving things be stationary," which was a radical change in the standard typing process. The result was a much faster typewriter.
2. Genetic algorithms are the result of the rather wild combination of the idea of Darwinian evolution with that of mathematical optimization. This has resulted in a novel class of algorithms that can be used in optimizing a function without requiring any knowledge about the nature of the function.
3. Codd (1983) combines the use of relational set theory and predicate logic with that of data modeling to make a bold departure from conventional thinking; also see Chapter 13, p. 336.

References

Altschuller, G.S. (2005). *The Innovation Algorithm: TRIZ, Systematic Innovation and Technical Creativity*. Worcester, MA: Technical Innovation Center. ISBN 0-9640740-4-4.

Beveridge, W. (1957). *The Art of Scientific Investigation*. Vintage Books (Random House), Third Edition, New York.

Codd, E. (1983). "A Relational Model of Data for Large Shared Data Banks." *Communications of the ACM*, 13(6): 77–387. Reprinted in *Communications of the ACM, 25th Anniversary Issue*, 26(1), 64–69.

Diehl, M. and Stroebe, W. (1987). "Productivity Loss in Brainstorming Groups: Toward the Solution of a Riddle." *Journal of Personality and Social Psychology* 53(3): 497–509.

Diehl, M. and Stroebe, W. (1991). "Productivity Loss in Idea-Generating Groups: Tracking Down the Blocking Effect." *Journal of Personality and Social Psychology* 61(3): 392–403.

Furnham, A. and Yazdanpanahi, T. (1995). "Personality Differences and Group versus Individual Brainstorming." *Personality and Individuality Differences* 19(1): 73–80.

Gericke, A. (2009). "Problem Solving Patterns in Design Science Research—Learning from Engineering." *ECIS 2009 Proceedings*.

Hadamard, J. (1954). *The Psychology of Invention in the Mathematical Field*. New York, NY: Dover Publications, Inc., 1–64.

Ladd, G. (1987). *Imagination in Research*. Ames, Iowa: Iowa State University Press.

Osborne, A.F. (1963). *Applied Imagination: Principles and Procedures of Creative Problem Solving*, Third Revised Edition. New York, NY: Charles Scribner's Sons.

Vaishnavi, V., Buchanan, G., and Kuechler, W. (1997). "A Data/Knowledge Paradigm for the Modeling and Design of Operations Support Systems." *IEEE Transactions on Knowledge and Data Engineering* 9(2): 275–291.

Wallas, G. (1926). *The Art of Thought*. New York, NY: Harcourt Brace.

Wikipedia-Brainstorming (2015). "Brainstorming." URL: http://en.wikipedia.org/wiki/Brainstorming (last accessed on January 29, 2015).

Chapter 8

Problem Selection and Development Patterns

The patterns in this chapter are applicable to the *awareness of problem* phase of the research (see Figure 6.2 in Chapter 6): You intend to pursue a design science research effort, but have not yet identified and developed a research problem that you can work on.

Use the following 18 patterns to help you in identifying and developing your research problem:

Preliminaries Type

- Problem formulation (p. 167)
- [M]Redefining research problem (p. 169)
- [M]Research domain identification (p. 171)
- Research topic identification (p. 173)

Visionary Type

- [M]Being visionary (p. 174)

Extrapolation Type

- Interdisciplinary problem extrapolation (p. 176)

Analysis Type

- [M]Complex system analysis (p. 178)
- [M]Cost-benefit analysis (p. 179)
- Leveraging expertise (p. 181)
- [M]Research conversation (p. 182)
- Research offshoots (p. 184)
- [M]Solution-scope mismatch (p. 185)
- Structuring an ill-structured problem (p. 187)
- [M]Questioning constraints (p. 188)

Generalization Type

- [M]Abstraction (p. 190)

Exploration Type

- Experimentation and exploration (p. 191)

Segmentation Type

- Hierarchical decomposition (p. 193)

Combination Type

- Bridging research communities (p. 194)

The above patterns are classified into eight types: *preliminaries*, *visionary*, *extrapolation*, *analysis*, *generalization*, *exploration*, *segmentation*, and *combination* and are presented in alphabetic order within each type. Patterns of the same type are obviously related and any differentiating information will be discussed at the time the patterns of the type are presented.

The names used for the types are generally self-explanatory. The *extrapolation* type name is used in the sense of extrapolation of ideas—to infer (not deduce) from

something known. In terms of types of expected knowledge contribution (see the design science research knowledge contribution framework shown in Figure 2.5—Chapter 2), the *being visionary* pattern is particularly suited to seeking an *invention* type of knowledge contribution and the *interdisciplinary problem extrapolation* pattern is particularly suited to seeking an *adaptation* type of knowledge contribution.

Meta-level patterns are indicated by the superscript ^{M} preceding the pattern name. The *redefining research problem* and *research domain identification* (meta-level) patterns gets used in the *problem awareness* phase but may be revisited after the project is in the *suggestion* phase or even in the *development* phase. The patterns *being visionary*, *questioning constraints*, *cost-benefit analysis*, and *abstraction*, while strongly identified with the *problem awareness* phase, are in fact applicable at any point in the research program. The pattern *research conversation* is also most naturally used in the *problem awareness* phase of a project, but may also be revisited whenever the research interests of different communities require detailed investigation, for example, in the *literature search*, *evaluation*, and *conclusion* phases. The patterns *complex system analysis* and *solution-scope mismatch* are also useful in the *suggestion and development* stage of the research.

This chapter will guide you in using patterns to help you in systematically identifying and developing a research problem that best suits your circumstances and needs and that is likely to be pursued successfully.

Preliminaries Type Patterns

- Problem formulation
- ^{M}Redefining research problem
- ^{M}Research domain identification
- Research topic identification

The above four patterns are generally useful in the preliminary stage of a research project. Any research problem needs to be well formulated and so the *problem formulation* pattern is generally useful. The extent to which the other two patterns, *research domain identification* and *research topic identification*, get used depends on whether you are new to research or moving to a new research domain. The *redefining research problem* pattern may get used in the preliminary stage of the research project but it may get revisited even after moving to the *suggestion* phase of the research in order to refine the project and make it doable.

Problem Formulation

Type

Preliminaries

Intent

Identify a specific research problem along with interesting research questions/issues.

Motivation

Identifying and defining a research problem along with corresponding research questions is a very important preliminary step in a research project. In many instances this step needs to be revisited as we gain familiarity with the research problem and make some headway in understanding the real knowledge gap that needs to be addressed in the research.

Context/Applicability

You have identified a research domain (see *research domain identification* pattern). You may have identified a set of problems in the research domain (see *research topic identification* pattern). Now you want to find specific research issues that you can work on, identify your research objectives based on their importance and existing research, and create a problem statement.

Description

Literature search (see *literature search* pattern, *familiarization with new area*) is a major task in this activity. The identification of goals, the inner environment, and the outer environment (see the section "Design Science and Design Science Research" in Chapter 2) can be useful in understanding the area where the research contribution is needed. Additionally, some understanding of the research community using the *literature search* pattern, *understanding research community*, use of or informal creation of a framework (see *literature search* pattern, *framework development*), and awareness of practice and industry (see *literature search* pattern, *industry/practice awareness*) is useful in this activity.

The ability to induce valid, focused, and interesting research questions from the information gained from the use of the patterns mentioned earlier is the most important and useful activity. This requires the use of creativity (see *creativity* patterns, Chapter 7).

Consequences

The use of this pattern should lead to a research problem that is interesting to you and to the research community. How good and significant the research problem and the research questions are depends on how creatively you use the available information.

Usage Example(s)

1. Abbasi and Chen (2008) first argue for an expanded computer-mediated communication system (CMC) and then scope their problem by focusing on a subset of issues for such systems; also see Chapter 13, p. 320.
2. The problem is identified based on the observed needs at CERN (Berners-Lee and Cailliau 1990). It is clearly stated and scoped; also see Chapter 13, p. 325.
3. Datta (1998) poses a new research problem. He justifies the value of the problem to practice as well as the approach used for its solution; also see Chapter 13, p. 340.
4. McLaren et al. (2011) use this pattern in conjunction with the *research conversation* and *research topic identification* patterns to define and scope their work; also see Chapter 13, p. 352.
5. The research problem (Purao et al. 2003) is identified from the literature in information systems and software engineering literature and the proposed solution approach is delineated from the prior naïve approaches; also see Chapter 13, p. 356.
6. Vaishnavi et al. (1997) use this pattern in the initial formulation of their research problem as well as after the problem is redefined in the *suggestion* phase; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Related Pattern(s)

The pattern uses the literature search patterns: *familiarization with new area* (p. 200), *[M]understanding research community* (p. 201), *[M]framework development* (p. 205), and *[M]industry/practice awareness* (p. 203).

[M]Redefining Research Problem

Type

Preliminaries

Intent

Redefine the research problem after reviewing the proposed research and its scope.

Motivation

The scope of the proposed research can be too broad or too narrow. In the former case, the problem may not be solvable, may not be solvable within the available

resources, or the problem may be too vague or too broad to make a meaningful knowledge contribution possible. In the latter case, the problem may look more like routine design instead of design science research or the expected knowledge contribution may lack generality. The reviewing of the scope of the proposed research and redefining the research problem is quite commonly done before the problem gets fully refined and well focused.

Context/Applicability

The pattern needs to be used at the beginning of the research project but is also reused as the *problem awareness* phase is often revisited even after moving to the *suggestion* phase of the research.

Description

1. Review the current formulation of the research problem and examine its nature and scope.
2. Does the problem look too ambitious? Is the problem solvable or solvable within the available resources? Is the scope too vague or broad to promise any meaningful or significant knowledge contribution? If the answer to any of these questions is in the positive then start working on narrowing the scope of the research problem or refining the research objective to look less daunting.
3. Does the problem look more like routine design than design science research? Does the research problem seek to produce an "artifact as situated implementation" (see Figure 2.6)? If the answer to either of the questions is in the affirmative then find if the problem solution promises an invention type of knowledge contribution (see the knowledge contribution framework shown in Figure 2.5). If so, then the problem may only need to be reviewed for refinement; otherwise, work on broadening the scope of the problem.

Consequences

Using the pattern, possibly more than once during the project, you will be able to settle on a refined and well-scoped research problem that promises interesting knowledge contribution.

Usage Example(s)

1. The field of "approximate algorithms" (see Wikipedia-Approximation (2014)) deals with approximate solutions of optimization problems that are unlikely to have exact efficient solutions. The field got started with the need for finding reasonable but sub-optimal solutions to such problems as the trav-

eling salesman problem for which a polynomial time solution does not exist and is unlikely to be ever found—a case of redefining the original optimization problems.

2. Simon (1956) introduced the term "satisficing" as a way to redefine the problem of finding an optimal decision by one that finds the best available decision in circumstances in which an optimal decision cannot be determined; see Simon (1996) (pp. 27–29).
3. In the work described by Vaishnavi et al. (1997), the research problem had to be redefined several times to create a well-scoped problem; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

[M]Research Domain Identification

Type

Preliminaries

Intent

Identify a research domain as a starting point for research problem development and conduct of research.

Motivation

We may be knowledgeable about a number of domains and need to select one of them for conducting research. The right selection of the research domain is important since it can shape our long-term research productivity and career.

Context/Applicability

You are new to research or intend to start working in a new research domain. The use of this pattern would not be needed if the research domain and/or topic are suggested naturally by such factors as membership in a highly paradigmatic research community. A research topic can also emerge from reading a research paper or attending a conference; in this case too there would not be any need for the use of this pattern.

Description

Your interest should be the primary criterion for choosing the research domain. This is because high-quality research requires your full involvement at both the conscious and unconscious levels. It requires the use of your physical as well as

creative energies. A sustained commitment to the research domain is difficult if you do not have an innate interest in the domain.

A close second criterion for choosing the domain should be the availability of resources for the conduct of the research. The resources that may be needed include access to the relevant literature, research community through conferences, newsletters, and so on, and collaborators, colleagues, or mentors.

You will gain more details on the research domain and a research topic as you progress further in developing your research problem; these details in turn can necessitate a review of the research domain decision. Thus, you should move on to identifying the research problem area in the chosen domain if the criteria discussed earlier have been satisfied, *before* you form an emotional attachment with the domain or a topic within the domain.

Notes

1. How a research domain or topic was chosen by the author is usually not of general interest to the reader of a publication and thus this information is not included in the presentation of the research.
2. The breadth of the research domain depends on the maturity of the domain. When the domain is relatively new, it can include a large number of topics and issues. As the domain matures, it bifurcates into specialized domains. One test for the existence of a research domain is the existence of a research community with outlets of communication such as conferences, newsletters, special interest groups of professional associations, and journals; there may only be a few of such outlets for a young research domain. Examples of current research domains are software metrics, electronic commerce technology, and database systems.
3. The next pattern, *research topic identification*, would be the next natural choice of a pattern to use for the further development of your research problem.

Consequences

A consequence of the use of this pattern is that you will have a good start in your further development of the research problem or a research program. There is greater likelihood that you will enjoy working in the research domain and succeeding in it.

Usage Example(s)

Vaishnavi et al. (1997) use this pattern along with other patterns related to the *bridging research communities* pattern for the identification and analysis of the research communities that have done work related to their research problem; also see "An

Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Research Topic Identification

Type

Preliminaries

Intent

Identify a research topic along with a general set of research questions and issues that are of interest to you and to the relevant paradigmatic community.

Motivation

Choosing an appropriate research topic in a research domain can be a daunting task. We need to identify it in spite of a huge amount of literature in the domain and the topic should meet certain criteria such as being interesting, addressing a knowledge gap, and promising reasonable assurance of success.

Context/Applicability

You have identified a research domain that you want to conduct research in but you do not yet have a research topic. You would like to identify a research topic along with a general set of research questions and issues that are interesting to you as well as the research community, and for which adequate resources are potentially available.

Description

Use the following steps along with your creativity (see *creativity patterns*, Chapter 7) to come up with a research topic along with a set of research questions and issues:

1. Familiarize yourself with the research domain (see *literature search* pattern, *familiarization with new area*).
2. Casually understand the relevant research community (see *literature search* pattern, *understanding research community*).
3. Use an existing framework to understand the work conducted in the area. If such a framework does not exist, it may be useful to develop at least an informal framework to provide some structure to the literature (see *literature search* pattern, *framework development*).
4. It may be useful to become aware of the state of art in practice and industry (see *literature search* pattern, *industry/practice awareness*).

Consequences

The pattern will help you in identifying a set of research questions and issues associated with a research topic that are of interest to you, the relevant research community, and/or to the practitioner community. The development of your research topic or program should be guided by this set but should not be limited by the set. Interesting research in many cases comes through "stirring the pot" or seeing the research area in new and novel ways.

Usage Example(s)

1. McLaren et al. (2011) use this pattern along with the "classic" definition of DSR as a solution to a business problem, to define and scope their research effort; also see Chapter 13, p. 352.
2. Vaishnavi et al. (1997) use this pattern in defining a research problem in the *problem awareness* phase of their research and after this phase was revisited in the *suggestion* phase of the research; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Related Pattern(s)

The pattern is using the following *literature search* patterns: *familiarization with new area* (p. 200), *[M]understanding research community* (p. 201), *[M]framework development* (p. 205), and *[M]industry/practice awareness* (p. 203).

Visionary Type Pattern

The *being visionary* pattern is typically invoked when the research project is expected to yield an invention type of knowledge contribution (see the knowledge contribution framework shown in Figure 2.5). It is also useful for seeking to do something distinct or different.

[M]Being Visionary

Type

Visionary

Intent

Envision an improvement to a situation or problem even if the present solution is acceptable.

Motivation

Significant progress in any research field starts with a vision for "what ought to be" as against "what currently is." Radical progress in a field gets initiated and sometimes fulfilled through a bold vision.

Context/Applicability

You are familiar with a problem or situation. You are not satisfied with the current solution or situation and can envision an improvement that you think to be feasible to perform.

Description

Identify the key features, criteria, or attributes of the current situation or the current best solution to a problem. Analyze the current situation and describe the ideal or desired set of features, values for attributes and criteria, and relevant qualitative aspects. In other words, create a "vision" for an improvement of the current situation. Critically review the "gap" between the current situation and the desired situation. The analysis need not be exhaustive but should examine if there are any major hurdles in bridging the gap. If the gap seems to be too large and infeasible to cover, make the gap smaller, that is, revise your vision and make it more realistic. You may want to increase the gap once you have gained confidence in covering the smaller gap.

Consequences

The consequences depend upon how bold, relevant, valuable, and compelling your vision is. If the vision is strong, you may become a pioneer in your field. On the other hand, if the vision is weak, then the result of fulfilling the vision will also be weak. A danger of trying to fulfill a strong vision is that you may not be successful in realizing the vision. Even if you are not successful in fulfilling the vision, you will learn much about the problem. Thus, you can hardly lose from being visionary.

Usage Example(s)

1. In his book, Ackoff (1978) lists a number of features that he would like to see in telephone communication systems. Since then research has made available most of such features, for example, caller ID.
2. Bentley and Saxe (1979) generalize the perfectly balanced binary search tree into a multidimensional search tree to organize a set of k-vectors (vectors of size k); perfectly balanced binary trees are used for organizing 1-vectors. The performance of the new data structure is shown to be $\lfloor \log_2 n \rfloor + k$ for the search

operation. This performance is optimal in any comparison-based model. The data structure does not, however, support update (insert and delete) operations efficiently. AVL-trees (Adel'son-Velskij and Landis 1962), on the other hand, are dynamic versions of the perfectly balanced binary trees. Vaishnavi envisioned whether one could have a dynamic version of the multidimensional search tree proposed by Bentley and Saxe with a performance of $O(\log_2 n) + k$. Specifically, it was envisioned that there exists a multidimensional version of the AVL-tree with the desired performance. The vision was fulfilled by Vaishnavi (1984) and later with a number of other similar data structures.

3. In Berners-Lee and Cailliau (1990), the authors identify their concerns with how information is accessed and envision the concept of "web of information nodes"; also see Chapter 13, p. 325.
4. Chen (1976) identifies problems with the existing data models and envisions a solution to the problems; also see Chapter 13, p. 328.
5. Codd (1970) shows his dissatisfaction with the existing data models and envisions an improvement that would ensure data independence; also see Chapter 13, p. 336.
6. A new vision for system resource management was developed in Denning (1968) that moved away from the prevailing approaches of managing the processor and memory resources separately; also see Chapter 13, p. 345.
7. Vaishnavi et al. (1997) envision the attributes of an ideal operations management support system; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Related Pattern(s)

The *creativity* pattern, [M]*brainstorming* (p. 156), can be used by this pattern.

Extrapolation Type Pattern

Use of the *interdisciplinary problem extrapolation* pattern is useful when one is seeking an *adaptation* type of knowledge contribution (see the knowledge contribution framework shown in Figure 2.5).

Interdisciplinary Problem Extrapolation

Type

Extrapolation

Intent

Extrapolate research in one area to create an interesting research problem in a different area.

Motivation

Interesting and productive research problems can be developed by extrapolating research in one area to a different area. A prerequisite for doing so is our familiarity with a number of different research areas.

Context/Applicability

You are familiar with an interesting piece of research in a certain area and think that a similar research in a different area would be interesting.

Description

1. Do not confine your readings to your own specialty alone. At least skim through the research in other areas.
2. While skimming through the research in the other areas ask yourself whether the type of research conducted in the other area would be interesting in your own area.
3. If the answer to the question is in the positive then formulate a problem using the benefit of the research conducted in the other area.

Consequences

The pattern can help you in identifying interesting solvable research problems. However, you should be careful in questioning the relevance of the problem and how the problem is formulated to your own area. If the problem cannot be extrapolated entirely, it may still be possible to adapt some portion of the problem for your area of research.

Usage Example(s)

Datta (1998) extrapolates the problem of software process discovery to that of discovery of business processes and the use of grammar discovery to reveal process maps; also see Chapter 13, p. 340.

Analysis Type Patterns

- [M]Complex system analysis
- [M]Cost-benefit analysis

- Leveraging expertise
- [M]Research conversation
- Research offshoots
- [M]Solution-scope mismatch
- Structuring an ill-structured problem
- [M]Questioning constraints

Analysis is an important way to define and develop a research problem. The *cost-benefit analysis* pattern gets used to assess the benefits relative to the cost of a project and to determine the project's feasibility. Leveraging the *expertise* pattern is useful in best utilizing the expertise of the researcher or that of the members of the research team. The rest of the patterns offer different methods of identifying and defining a research problem.

[M]Complex System Analysis

Type

Analysis

Intent

Analyze a complex system to find areas where research is needed to improve the performance or effectiveness of the system.

Motivation

Analysis of existing complex systems can provide a good vehicle for formulating an interesting research problem that is also relevant to practice.

Context/Applicability

You are familiar with or have access to a complex system. You are interested in conducting research that can improve the performance or effectiveness of the complex system.

Description

Be alert for deficiencies and problem areas while conducting the following analysis of the complex system:

1. Analyze the static structure of the complex system. Find out what the subsystems of the system are and how they are related to each other and apply the same analysis to the subsystems recursively. Most often, you will find the system to be a hierarchic system or a "nearly decomposable system."

(The difference between a nearly decomposable system and a hierarchic system is that while the interactions between the subsystems in the former are weak compared to those within the subsystems, such interactions are not negligible.)

2. Analyze the dynamic behavior of the system and study how this behavior is produced.
3. Study the evolution of the system. Complex systems usually are the result of a long process of evolution from a relatively simple system.
4. Attempt a preferably simple representation of the system. The representation of complex systems need not be complex.

Consequences

You will get a deeper understanding of the complex system and how it manages its complexity. You will also be able to see problem areas that can be starting points for formulating a research problem of relevance to the complex system.

Usage Example(s)

In Vaishnavi et al. (1997), the research problem is identified in the process of analyzing a complex operations environment and its modeling; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

ᴹCost-Benefit Analysis

Type

Analysis

Intent

Use cost-benefit analysis to determine if the planned expenditure of resources is justified by the expected research benefits.*

* When an early version of this book was used to teach a class on design science research to graduate students at the Indian Institute of Technology in Delhi, the students expressed surprise that a cost-benefit analysis would be applied to research. We believe this is one of the most important and widely applicable patterns in the book since our experience is that resources, especially time, are always limited. Initial appraisal of the cost-benefit of a research project and reappraisal whenever the project seems "stalled," especially in its early phases, is crucial at many points in a researcher's career. These include finishing a PhD program and, in the United States at least, the pre-tenure (assistant professor) phase of a job at a university.

Motivation

Cost-benefit analysis needs to be conducted before the start of a research project especially when the project is large and/or external funds are sought.

Context/Applicability

You are planning to commit to the expenditure of a large amount of resources for your research project. Such resources can be physical equipment such as computers and software or human resources such as the subjects in an experiment, and time, which is always a scarce resource in any environment, business, or academic. Determining whether the planned cost justifies the research benefits is required when developing a research proposal for a doctoral degree or a research grant. Most research in industry requires a cost-benefit analysis before any resources are committed to the project.

Description

This pattern suggests an analysis of the planned major cost and its expected benefits before you plunge into actual implementation of the plan:

1. Analyze and estimate the expected cost in human and physical resources. Confirm that these resources are available or are likely to be available.
2. Analyze the expected research benefits, that is, the expected research findings and results that can result from the planned expenditure.
3. Explore alternative less expensive strategies for carrying out the research. Make a convincing case for the expected benefits outweighing the costs.
4. Develop a detailed plan with milestones so that you can confirm that the expected research benefits are materializing as the project proceeds. Even after starting the project, monitor the costs and benefits. Scale down or even cancel a planned expenditure if the benefits do not justify the expenditure.

Consequences

This pattern will help you in exploring all the alternatives before you plunge into conducting your research especially when major expenditure in physical and/or human resources are needed. It will also lead you to analyze the planned cost and to see if the expenditure is feasible.

Usage Example(s)

1. Detailed estimates of human and physical resources are made in Berners-Lee and Cailliau (1990). An attempt is made to reduce the cost of the project without affecting the research benefits; also see Chapter 13, p. 325.

2. Vaishnavi et al. (1997) use this meta-level pattern in the *suggestion/development* phases of their research while iterating with different possible solution approaches to their research problem; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Leveraging Expertise

Type

Analysis

Intent

Select a research problem to pursue that can leverage your or your team's strengths and expertise.

Motivation

Individual or collective expertise (in case of team projects) is a principal resource that can be used for the successful completion of a research project. We would like to make the best use of this resource.

Context/Applicability

You (or your team) have a number of research areas and/or problems that you are generally interested in. You do not have the time or resources to develop completely new areas of expertise to aid your research. You would like to choose a research topic that has the best chances of successful completion based on your current strengths.

Description

Your strengths, expertise, and interest are very important determinants of success for your (or your team's) research project. To leverage your expertise:

1. Identify your strengths and the areas of your expertise. Find what areas you are most comfortable in and what areas interest you most. Ask yourself if there is a particular type of experience that gives you some unique strength in a certain type of research project.
2. Choose your research topic or project such that it is either utilizing your unique expertise or strength, or builds on them. If you are working as a team, choose the research project such that it utilizes the collective (preferably overlapping) strengths of the individual members of the team.

Consequences

Based on this pattern you (or your team) will pursue research that utilizes your current expertise and strengths. This is a conservative approach. The down side of this approach is that you (or your team) will not develop expertise and interest in new research domains.

Usage Example(s)

1. A person who has worked in the software industry for many years before pursuing research has unique insights in the software development area. This person can bring his/her expertise and strengths to bear on a research project in the software development area as against a research area that does not deal with software development.
2. Choobineh and Lo (2005) leverage the expertise and insight gained earlier through writing a survey of the area of database design support systems; see also Chapter 13, p. 332.
3. Purao et al. (2003) are leveraging their prior research experience in addressing the problem; see also Chapter 13, p. 356.

[M]Research Conversation

Type

Analysis

Intent

Analyze the literature to find opportunities for research or to "position" your research.

Motivation

A research conversation takes place through research publications and presentations. Getting intimately familiar with such conversations is a powerful tool to find opportunities for research or for positioning already completed research.

Context/Applicability

You are new to a research area and would like to conduct research that can be published relatively easily. Alternatively, you have a research idea and you would like to position it best with respect to the ongoing "research conversations."

Description

1. Identify the research field of relevance and familiarize yourself with the work being conducted in the area using the *literature search* pattern, *familiarization with new area*.
2. Understand the intellectual structure of the research community using the *literature search* pattern, *understanding research community*.
3. If there is a fairly extensive literature in the research area, utilize an existing framework to understand the ongoing research in the area or at least informally develop a framework. The *literature search* pattern, *framework development*, will be useful.
4. Identify the current "puzzles" and research gaps that may be of interest to you.
5. Scan the literature to find the conversations currently going on with respect to the identified research puzzles and the problems/issues that still need to be addressed.

Consequences

You will become more closely linked to the research community and the current research paradigms being followed by the community. This will help you in getting yourself and your work accepted by the community.

Usage Example(s)

1. Abbasi and Chen (2008) utilize their expertise in positioning their research within the computer-mediated communication analysis area; also see Chapter 13, p. 320.
2. The literature review in Chen (1976) shows the research conversation going on in the data modeling area and the author's attempt to position his research with respect to this conversation; also see Chapter 13, p. 328.
3. Using their prior survey work in the field, Choobineh and Lo (2005) are able to identify and join a research conversation on automated database design support systems; see also Chapter 13, p. 332.
4. Codd (1970) shows a good understanding of the research in data modeling and positions his contribution with respect to this research; also see Chapter 13, p. 336.
5. Denning (1968) provides a good analysis of the existing literature on resource allocation and positions his contribution in the context of this analysis; also see Chapter 13, p. 345.
6. Hoare (1978) demonstrates his awareness of the existing research problems in parallel programming and positions the reported work with respect to these problems; also see Chapter 13, p. 348.

7. McLaren et al. (2011) use this pattern to identify both the problem area and the novelty contributed by their solution artifact; also see Chapter 13, p. 352.
8. Vaishnavi et al. (1997) use this meta-level pattern in the *publishing* stage of their research to identify a journal that their work could best fit in; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.
9. The ongoing research conversations in the relevant journals revealed that no algorithm existed for constructing optimal multi-way search trees (Vaishnavi et al. 1980); also see Chapter 13, p. 360.

Related Pattern(s)

This pattern uses the *literature search* patterns: *familiarization with new area* (p. 200), *[M]understanding research community* (p. 201), and *[M]framework development* (p. 205).

Research Offshoots

Type

Analysis

Intent

Find research problems that have resulted from a recent seminal research contribution.

Motivation

A good vehicle for finding interesting and doable research problems in a research area is to identify recent seminal published research paper(s) in that area. Such work can be a rich source for new research problems.

Context/Applicability

While examining recent literature in your research area, you have found a seminal research paper that solves an existing research problem. The research contribution reported in the paper is significant either because the research problem solved is significant or the approach used in the solution of the problem is novel or significant.

Description

A significant research contribution usually opens up a new segment of research. The solution to old problems gives rise to new research problems. Critically review the research paper that solves the existing problem. While examining the paper, try to answer the following questions:

1. Does the paper address all the issues of the problem? Are there issues that still remain unresolved?
2. Has the most general version of the problem been solved?
3. Has the solution to the problem made certain assumptions about the problem? How reasonable are these assumptions? Has the solution weakened or removed certain constraints for the solution of the problem? Are these constraints important?

A positive answer to one or more of the above questions will lead to the identification of a gap in knowledge that needs to be filled.

Consequences

The new research gaps identified are likely to be less significant than the research gap addressed by the research that you have examined. This pattern is more useful to identifying relatively small research problems that you can work on rather than a broad stream of research. If the examined research paper has opened up a broad area of research, then there may be scope for identifying a wide area of research that you can work on.

[M]Solution-Scope Mismatch

Type

Analysis

Intent

Determine whether existing solution(s) to a problem can be used when the scope of the problem is expanded or a more complex version of the problem is considered.

Motivation

A published research article would generally be presented in the most general form and covering the largest scope possible. However, the business environment of the solved problem or the technology used may have changed. In either case, the published solution can be examined in the context of an expanded or changed scope. This may result in an interesting research problem.

Context/Applicability

There exists a good or reasonable solution for a research problem. You can think of a more complex version of the problem or one with expanded or changed scope that is worth solving, which has either not been addressed so far in the literature or the available solution is not reasonable. The existing solution technique for the smaller scope problem can be applied to the new problem.

Description

Apply the existing solution technique to the larger problem and analyze the solution. If the solution is acceptable under these conditions then you have solved the problem using the existing technique. This may be a research contribution if the new problem is important and the application of the solution technique is nontrivial. If, on the other hand, an analysis of the solution shows that it is not a good solution, then you may have discovered a research problem worth solving. If you can think of a set of such problems with varying complexity, then it is useful to apply the preceding steps to all these problems. This will provide you with more information on how the existing solution technique works as the scope of the problem expands. This information will be useful in your exploration of a better solution technique to the set of problems and in making your solution more general.

If there exists more than one solution technique to the limited scope problem that can be applied to the larger scope problem, then the above steps need to be applied using all such solution techniques. If the application of any of the existing solution techniques does not lead to a reasonable solution to the more general problem(s), then you have made a case for generalizing the existing solution technique(s) or finding a new solution technique. It is better to try generalizing the existing solution technique(s) before trying to come up with a different solution technique.

Consequences

If the application of existing solution technique(s) leads to an acceptable solution for the expanded problem(s), then the work may not be as productive as in the case when the solution is less than acceptable. However, in either case, it is a good investment of your time. In the former case, you have shown that the existing solution can be extended to the more complex problem and hence there is no need of coming up with a generalized or different solution technique. In the latter case, you have found an interesting research problem that is worth solving.

Usage Example(s)

1. Abbasi and Chen (2008) discuss the limitations of existing systems and the need to overcome these limitations; also see Chapter 13, p. 320.
2. The limitations of the existing data models to support data independence are demonstrated in Codd (1970); also see Chapter 13, p. 336.
3. While analyzing the background literature, Denning (1968) discusses the merits of the least recently used selection policy for memory management when there is only one process and the weaknesses of the policy when there are many processes; also see Chapter 13, p. 345.
4. McLaren et al. (2011) use this pattern to identify knowledge gaps in existing solutions to the problem the authors have defined; also see Chapter 13, p. 352.
5. Vaishnavi et al. (1997) discuss a mismatch that existed between the available tools and what was needed for constructing and maintaining a support system for complex operations environments; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.
6. An efficient algorithm existed for organizing data in the primary storage in optimal fashion (Knuth 1971) but no such algorithm existed for disk storage (Vaishnavi et al. 1980); also see Chapter 13, p. 360.

Related Pattern(s)

The suggested approach is following the *suggestion and development* pattern, *easy solution first* (p. 234).

Structuring an Ill-Structured Problem

Type

Analysis

Intent

Provide some structure to an ill-structured problem.

Motivation

Practice many times presents ill-structured situations. These, when structured, can result in research problems that will also have relevance.

Context/Applicability

You are familiar with a problem usually driven from practice. The problem does not have a clear objective or constraint. There could also be a source of uncertainty in the problem.

Description

1. Identify the key objectives. If there are multiple objectives, prioritize them and drop those objectives that are not as important. Make sure that the objectives being considered are not in conflict with each other.
2. Analyze the nature of objectives. Try to quantify qualitative objectives into quantitative ones.
3. Analyze the constraints for their relevance and drop those constraints that do not seem to be relevant.
4. Attempt to state the problem in precise formal terms.

Consequences

Some structure and preciseness will be introduced into the problem that originally was ill structured and ill defined. The solution of the problem may still need heuristics or human judgment but it is always better to refine and formalize the problem as much as possible. This opens the possibility of using algorithmic or optimization techniques to parts of the problem, which is preferable to the use of "softer techniques."

Usage Example(s)

1. Given the difficulty of automating the discovery of complete process descriptions from actual process event traces, Datta (1998) decomposes the total problem into components and demonstrates that a process activity graph is an important component of the discovery of AS-IS processes; also see Chapter 13, p. 340.
2. The research (Purao et al. 2003) generates a structured approach to the complex problem of expert performance in conceptual design from the machine learning literature; also see Chapter 13, p. 356.

[M]Questioning Constraints

Type

Analysis

Intent

Identify a gap in research by questioning constraints that may be explicitly or implicitly imposed on a research problem by the research community.

Motivation

Design science research is conducted assuming a number of stated and unstated constraints related to the environments of the research problem and the technology being used. The stated assumptions obviously can change and need to be re-examined. The unstated assumptions may stem from the current micro-paradigm under which the research has been conducted. Questioning both types of such assumptions can lead to good research problems.

Context/Applicability

You are starting to work in a new field and thus able to look at the field afresh without being burdened by the prevailing assumptions and constraints.

You are aware of some new technology or other developments in the field or related fields that can impact the prevailing constraints used in the current research paradigm being followed in the field.

Description

1. Conduct a quick study of the field to find out the constraints that are part of the current research paradigm being followed in the solution of the research problem.
2. Take a fresh "outside" view of the field to find out whether the constraints are valid. The invalidity of the constraints may also be determined by technology and other developments that have taken place in the field or related fields. Form a list of constraints that you can argue to be unnecessary.
3. Identify gaps in research by analyzing whether the existing solution to the problem holds when the unnecessary constraints are removed.

Consequences

You will either have a better appreciation of the constraints imposed to the research problem or you will have identified a new research problem or problems. The identified research problem(s) may, however, be too difficult to solve. On the other hand, you may be able to open a new research direction or even a new research paradigm.

Usage Example(s)

There was an implicit constraint in the research community that there is no difference between how data is presented and how it is represented. The work reported in Codd (1970) is largely the result of questioning this constraint; also see Chapter 13, p. 336.

Generalization Type Pattern

The *abstraction* pattern is useful in moving the research output of a research project to a higher level of abstraction (see the abstraction hierarchy of the DSR outputs shown in Figure 2.6). This pattern can be useful particularly when seeking *improvement* or *adaptation* types of knowledge contribution (see the knowledge contribution framework shown in Figure 2.5).

ᴹAbstraction

Type

Generalization

Intent

Abstract a research problem from its many concrete instances and state the research issues and questions.

Motivation

Design science research problems are initially posed and solved in their different concrete instances. This can, for example, happen in different types of industries that are beset with their own concrete versions of a problem. Using abstraction to define and develop a more general abstract problem from its concrete instances can be rewarding to both the researcher and the research field. This needs to be done as the research problem area matures to justify claims of knowledge contribution.

Context/Applicability

You are aware of many concrete problems from experience or literature that seem to be somewhat similar. You want to formulate a deeper research problem from the concrete instances.

Description

1. Use your abstraction and creative abilities to think of the underlying issues that give result to the concrete problems.
2. Informally model the underlying phenomenon to see whether the solution of the identified underlying problem will solve the observed concrete problems.
3. Check if the underlying problem can also lead to other concrete problems that you may not have experienced or discovered before.
4. Define the underlying problem and frame research questions that need to be answered in order to solve the underlying problem.

Consequences

You are able to move from a level of development or ad hoc problem solution to a broader research level. You are able to frame research issues and questions that are significant and have a broad impact.

Usage Example(s)

Hoare (1978) abstracts the problem of communication between processes and their synchronization to that of finding a simple way for sequential processes to communicate with each other; also see Chapter 13, p. 348.

Exploration Type Pattern

The *experimentation and exploration* pattern provides guidance in probing an area to find an interesting research problem.

Experimentation and Exploration

Type

Exploration

Intent

Explore a new area and the research problems in the area through experimentation.

Motivation

Most design science research uses experimentation to explore a phenomenon that is not well understood. It is through experimentation that the intricacies of the phenomenon can be understood and research problems can be developed.

Context/Applicability

You are working in a research area that is not fully understood. In this area, experiments or prototypes can be built to understand the phenomenon being researched or to test a theory or design principle being developed in the research.

Description

In an area that is not fully understood, experimentation (Tichy, 1998) that can proceed through prototyping is an excellent way of gaining familiarity with the area and understanding the real issues that needed to be addressed. The experimentation reveals complexities of the area and helps in discovering useful areas of investigation. The following steps provide a general guidance for following this approach:

1. If the area has been investigated before, then form a prototype that incorporates the current knowledge of the area. If the area has not been investigated before, then build a prototype that incorporates your best hypotheses in the area being investigated.
2. Observe the prototype (experiment) in action and make a systematic record of the performance of the various parameters of interest under varying conditions of execution.
3. Use the knowledge gained through observations to identify the problems and issues that need to be researched.

Consequences

You may uncover new areas of research through the use of this pattern. It is also possible that the pattern does not lead to the discovery of completely new or innovative problems of research. In either case, the pattern should increase your understanding and knowledge of the area; this will help you in understanding and isolating the different research issues that are important for the area of research.

Usage Example(s)

The work of Choobineh and Lo (2005) is in the context of an existing published system, which provides information for comparison and evaluation; also see Chapter 13, p. 332.

Segmentation Type Pattern

The *hierarchical decomposition* pattern gives form to an age-old technique to manage complexity.

Hierarchical Decomposition

Type

Segmentation

Intent

Hierarchically decompose a research problem to manage the complexity of solving the problem.

Motivation

Hierarchical decomposition is a time-tested method to manage complexity posed by a complex problem. This technique can be used to define a research problem that is interesting as well as manageable. Decomposability of the problem, however, is a key requirement for the use of this technique.

Context/Applicability

You have identified a research problem. The problem seems, however, to be too complex and unlike anything that you have seen before. You may also be unfamiliar with the problem domain.

This pattern assumes that the problem is decomposable. This means that it is possible to decompose the problem into smaller problems such that the solution to the smaller problems can be composed into the solution of the bigger problem. Not all problems, however, are decomposable. An example of a non-decomposable problem would be one in which any solution to one part of a problem can change some aspect of another part of the problem.

Description

Hierarchical decomposition is a standard technique for managing complexity. Here are the guiding steps:

1. Decompose the problem into subparts.
2. Formulate the problem into the problems for solving each of the parts and the problem of combining the solutions for the parts to form the solution for the entire problem.

3. If the parts of the problem are still complex then repeat the process for each part.
4. Depending on the complexity of the problem, choose one or more parts of the problem at some level of decomposition to be your research problem.
5. If the resources permit, move to a higher level problem in the hierarchical decomposition after the lower level problems are solved.

Consequences

The pattern lets you concentrate on a relatively smaller problem at any one time.

Combination Type Pattern

The *bridging research communities* pattern provides guidance on defining a research problem that attempts to bridge different but related research areas.

Bridging Research Communities

Type

Combination

Intent

Identify a problem that attempts to bridge the gap between two interrelated but distinct research areas (communities).

Motivation

Just as interdisciplinary problems provide a fruitful avenue for research, problems that bridge interrelated but distinct research areas (communities) within the same field are attractive since such research has better impact and reaches a broader audience.

Context/Applicability

- You want your research to have a significant impact and have a broad audience.
- You have identified two or more research communities that have some overlap in issues that they address.
- You are either familiar with the overlapping research communities or are willing to learn about these communities and their research. Alternatively, you may be working in a team where different members of the team have expertise in the knowledge areas of the research communities.

The pattern offers significant benefits but there are also pitfalls. Therefore, you need to use your judgment before the use of this pattern for the identification and development of a research problem. Here are the benefits and pitfalls of the use of this pattern.

Benefits

- The results of your research will be of interest to a broader audience.
- There is likelihood of some novelty in your research because of its interdisciplinary nature, which increases its significance.
- You can also improve the quality of the research by picking and adapting the strong approaches used in the research areas involved.

Pitfalls

- The involved research communities may use different terminologies and it may be difficult to use a single terminology that satisfies all of these communities.
- There may be a difference in how the overlapping research communities see the research issues and the assumptions that are deemed to be reasonable.
- A result of the preceding two points is that the publication of your research that bridges the research communities may be difficult or time consuming. This is because it is unlikely for the editor of a research journal to find referees that are well versed in all the involved areas. The editor may choose different referees specializing in different areas. In this case, there is the possibility of the different referees not agreeing on the format or the contents of the reported results.

Description

1. Select two or at most three distinct but interrelated research communities that have distinct approaches or insights to address certain common issues.
2. If you are not familiar with all the research communities and their literature, then spend time to gain such familiarity. The *literature search* pattern, *familiarization with new area* or *understanding research community,* may be useful in this regard. A better solution is to form a team of researchers who have expertise in the research conducted in the research communities. In this case too, some understanding of the research communities by all the members of the team is needed.
3. Identify approaches or insights provided by the research communities for some common problems or issues that have the potential of being combined in a complementary fashion.
4. See if the above process can be extended to other related research communities.

Consequences

The application of this pattern should help you in identifying an important research project that is of interest to a number of research communities. A successful bridging of the research communities through your research has the potential of a broad impact. Even if you do not execute the research project, the use of this pattern can provide you with some new insights that you may be able to use in your subsequent research.

Usage Example(s)

1. The work in Datta (1998) draws heavily from the literature of the research communities of workflow management and business process reengineering as well as grammar discovery as previously applied to software process discovery; also see Chapter 13, p. 340.
2. In Fraser et al. (1991), the authors use this pattern by developing techniques that enable informal and formal methods to be used together. Specifically, the authors provide a means to make use of the strengths of structured analysis in capturing user requirements, and the strengths of Vienna Development Method (VDM) in assuring specification completeness, through a translation mechanism. While the benefits of exploiting the strengths of each community's approach would seem obvious, the authors had to address the comments of the referees some of which were divergent in their suggestions. Since the publication of the paper, the fusing of informal and formal specifications has been pursued by a number of other researchers and tools have been developed that incorporate the fusion.
3. The work in Purao et al. (2003) draws heavily from multiple communities—software engineering, machine learning, human learning, and cognition; also see Chapter 13, p. 356.
4. Vaishnavi et al. (1997) bridge the research communities of data modeling, knowledge representation, and software engineering; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Related Pattern(s)

The following patterns can be useful in the use of the current pattern: [M]*research domain identification* (p. 171); the *literature search* patterns: *familiarization with new area* (p. 200); [M]*understanding research community* (p. 201); and [M]*research conversation* (p. 182).

References

Abbasi, A. and Chen, H. (2008). "CyberGate: A Design Framework and System for Text Analysis of Computer-Mediated Communication," *MIS Quarterly* 32(4): 811–838.

Ackoff, A. (1978). *The Art of Problem Solving*, Wiley, New York.

Adel'son-Velskij, G. and Landis, Y. (1962). "An Algorithm for the Organization of Information." *Soviet Mathematik Doklady* 3, 1259–1263.

Bentley, J. and Saxe, J. (1979). "Algorithms on Vector Sets," *SIGACT News* 11(2): 36–39.

Berners-Lee, T. and Cailliau, R. (1990). "WorldWideWeb: Proposal for a Hypertext Project." URL: http://www.w3.org/Proposal.html (last accessed on January 29, 2015).

Chen, P. (1976). "The Entity-Relationship Model: Toward a Unified View of Data." *ACM Transactions on Database Systems* 1(1): 9–37.

Choobineh, J. and Lo, A. (2005). "CABSYDD: Case-Based System for Database Design." *JMIS* 21(3): 281–314.

Codd, E. (1970). "A Relational Model of Data for Large Shared Data Banks." *Communications of the ACM* 13(6): 377–387. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 64–69, January, 1983.

Datta, A. (1998). "Automating the Discovery of AS-IS Business Process Models: Probabilistic and Algorithmic Approaches." *Information Systems Research* 9(3): 275–301.

Denning, P. (1968). "The Working Set Model for Program Behavior." *Communications of the ACM* 11(5): 323–333. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 43–48, January, 1983.

Fraser, M., Kuldeep, K., and Vaishnavi, V. (1991). "Informal and Formal Requirements Specification Languages: Bridging the Gap." *IEEE Transactions on Systems* 17(5): 454–466.

Hoare, C. (1978). "Communicating Sequential Processes." *Communications of the ACM* 21(8): 666–677. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 100–106, January 1983.

Knuth, D. (1971). "Optimum Binary Search Trees." *Acta Informatica* 1(1): 14–25.

McLaren, T., Head, M., Yuan., Y., and Chan, Y. (2011). "A Multilevel Model for Measuring Fit between a Firm's Competitive Strategies and Information Systems Capabilities." *MIS Quarterly* 35(4): 909–929.

Parnas, D. (1998). "Successful Software Engineering Research." *Software Engineering Notes* 23(3): 64–68.

Purao, S., Storey, V., and Han, T. (2003). "Improving Analysis Pattern Reuse in Conceptual Design: Augmenting Automated Processes with Supervised Learning." *ISR* 14(3): 269–290.

Simon, H. (1996). *The Sciences of the Artificial*, Third Edition. The MIT Press, Cambridge, MA.

Simon, H.A. (1956). "Rational Choice and the Structure of the Environment." *Psychological Review* 63(2): 129–138.

Tichy, W. (1998). "Should Computer Scientists Experiment more?" *IEEE Computer* 31(5): 32–40.

Vaishnavi, V. (1984). "Multidimensional Height-Balanced Trees." *IEEE Transactions on Computers* C-33(4): 334–343.

Vaishnavi, V., Buchanan, G., and Kuechler, W. (1997). "A Data/Knowledge Paradigm for the Modeling and Design of Operations Support Systems." *IEEE Transactions on Knowledge and Data Engineering* 9(2): 275–291.

Vaishnavi, V., Kriegel, H., and Wood, D. (1980). "Optimum Multiway Search Trees." *Acta Informatica* 14(2): 119–133.

Wikipedia-Approximation (2014). "Approximation Algorithm." URL: http://en.wikipedia.org/wiki/Approximation_algorithm (last accessed on January 29, 2015).

Chapter 9

Literature Search Patterns

The patterns in this chapter are applicable to the *awareness of problem* phase of the research (see Figure 6.2 in Chapter 6). You conduct a literature search to understand a research area and/or to position the research ideas or approaches that you may be considering. Meta-level patterns are indicated by the superscript [M] preceding the pattern name.

The pattern *industry/practice awareness* may be revisited whenever a detailed investigation of industry techniques in an area may be beneficial, in the *evaluation* and *conclusion* phases, for example. The patterns *understanding research community* and *framework development* may be revisited in the *suggestion* and *development* phases of the research.

Use the following four patterns to conduct the literature search.

Preliminaries Type

- Familiarization with new area (p. 200)
- [M]Understanding research community (p. 201)

Analysis Type

- [M]Industry/practice awareness (p. 203)

Modeling Type

- [M]Framework development (p. 205)

The patterns in this chapter are categorized into three types: *preliminaries*, *analysis*, and *modeling. Preliminaries*-type patterns are useful to get a better understanding of a research area. The *analysis*-type pattern lets one analyze the state of the art from a certain perspective. The *modeling*-type pattern is useful to get a big picture of the research in the area and to identify the knowledge gaps.

This chapter will guide you in using patterns that can help you in an effective literature search.

Preliminaries Type Patterns

- Familiarization with new area
- [M]Understanding research community

The above two patterns of *preliminaries* type are interlinked patterns that are useful for a new researcher or one who is starting work in a new research area.

Familiarization with New Area

Type

Preliminaries

Intent

Get familiar with a new research area.

Motivation

Before starting to work in a new research area, familiarization with the area is needed to ensure success and to avoid pitfalls.

Context/Applicability

You are either new to research or are exploring working in a new research area. You have identified the domain of research (see *problem selection and development* pattern, *research domain identification*). You are now interested in familiarizing yourself with the domain so that you can possibly find a set of research questions or problems that are of interest to you and the relevant research community.

Description

Familiarize yourself with the research literature and the research community in the domain by:

- Using the Internet resources such as World Wide Web search tools
- Reading literature in the area
- Attending conferences
- Talking to people working in the area
- Casual understanding of the selected research community (see *understanding research community* pattern)
- If the research community is not highly paradigmatic or if the literature is not well organized, then use the *understanding research community* pattern to get a deeper understanding of the research community.
- If the literature in the area is extensive and no good published survey is available, then use the *framework development* pattern to get a better idea of the work in the area and its structure.

Consequences

The pattern will help you getting conversant with the research area to get started on research in the area. You may find that you are not prepared to work in the research area because of inadequate knowledge of the area and the time it will take to acquire that knowledge. The familiarity with the area will also give you a better idea of your level of interest in the research area; you may decide not to pursue research in the area based on this information.

Related Pattern(s)

The pattern uses the [M]*framework development* pattern (p. 205) (in addition to [M]*understanding research community*).

[M]*Understanding Research Community*

Type

Preliminaries

Intent

Understand how the community organizes its "intellectual structure" and gains "acceptance" by the community.

Motivation

Each research community is like a social tribe that needs to be fully understood by a new researcher in the community. Understanding the research community helps in acceptance by and assimilation in the community and thus ensures better success in the research work.

Context/Applicability

You are new to the research community. You would like to understand the community. This would help you to gain acceptance by the community and to become able to influence the community. Understanding the community and your acceptance by the community would also help you to report your research in a way that is acceptable to the editors and reviewers of journals of the community.

Description

1. Use the World Wide Web, conference proceedings, books, and journals to gain knowledge and understanding of:
   - The history, foundation, paradigm, and culture of the community
   - The hot issues, shared beliefs, shared values, and tacit knowledge of the community
   - The research techniques, procedures, protocols, and tools that the community has accepted as their standard for working on the research issues
   - The vocabulary used by the community and the level of abstraction and explanation used to communicate research ideas and results
2. Use the understanding gained to know what the intellectual boundaries of the community are. Stay within this boundary unless you want to enhance the community by extending these boundaries. This is usually an activity for mature researchers.
3. Retain your individuality and creativeness to pursue issues and research directions that influence and enhance the research community.

Consequences

The use of the pattern can help in your assimilation in the community. There is, however, a danger that you may get overly assimilated, which may prevent you from offering novel and creative research directions and solutions. Thus, you should try to maintain your individuality and creativity while using the pattern to gain understanding of the community and your acceptance by the community.

Usage Example(s)

1. Abbasi and Chen (2008) show their understanding of the computer-mediated communication (CMC) research community through their analysis of the research community's vocabulary, publication outlets, and so on; also see Chapter 13, p. 320.
2. Chen's paper (1976) shows a good understanding of the data modeling research community; also see Chapter 13, p. 328.
3. Choobineh and Lo (2005) show their understanding of the research community gained through earlier survey work; also see Chapter 13, p. 332.
4. Denning's paper (1968) shows a good understanding of the research community and its intellectual structure; also see Chapter 13, p. 345.
5. McLaren et al. (2011) devote considerable time to understanding prior research in their area before defining requirements for their artifact; also see Chapter 13, p. 352.
6. Vaishnavi et al. (1997) use this pattern along with other patterns related to the *bridging research communities* pattern to understand the different research communities relevant to their research problem; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Analysis Type Pattern

The *industry/practice awareness* pattern provides a way to analyze the state of the art in industry and practice.

[M]*Industry/Practice Awareness*

Type

Analysis

Intent

Maintain awareness of the developments in industry and practice.

Motivation

To conduct research that is relevant and well grounded, a thorough awareness of industry and the state of practice is needed.

Context/Applicability

You want to find research topics that are of relevance and interest to practice and industry. Alternatively, you want to find applications of your research to industry.

Description

Use the following strategies to remain abreast of the current practice in industry:

1. Use the same systems and tools as used in the industry. For example, use the programming languages, database systems, design methodologies, and other systems and tools used currently in practice. This will help you in experiencing firsthand the problems and issues faced in practice.
2. Read professional and trade magazines to remain aware of the developments in practice.
3. Accept a visiting assignment in an industrial organization of relevance to your research domain. Participate or observe the actual work being done and abstract the issues and problems arising from this work.

Consequences

To obtain the desired benefits of this pattern, you need to be able to identify the problems faced in practice and to be able to abstract them into research problems that are of general interest. In other words, you need to be careful not to identify yourself too closely with the actual work that you are observing or participating in or with the compromises being made in carrying out the work.

Usage Example(s)

1. Abbasi and Chen (2008) show their understanding of capabilities of commercial CMC systems through the features they incorporate in their research artifact; also see Chapter 13, p. 320.
2. Research on the World Wide Web (Berners-Lee and Cailliau 1990) was conducted at CERN and was motivated by problems faced at CERN in linking and accessing information; also see Chapter 13, p. 325.
3. Codd (1970) shows a good awareness of the available commercial database management systems and their limitations; also see Chapter 13, p. 336.
4. Datta (1998) stresses the real-world aspects of the problem addressed by citing from general non-technical citations from workflow and process management; also see Chapter 13, p. 340.
5. McLaren et al. (2011) thoroughly study existing practice solutions to the problem they address; also see Chapter 13, p. 352.

6. The research (Purao et al. 2003) is motivated by the well-known industry problem of facilitating the reuse of design components; also see Chapter 13, p. 356.
7. The research problem in Vaishnavi et al. (1997) was identified through attempting to model a real-world operations support system using Prolog; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Modeling Type Pattern

The *framework development* pattern uses modeling to abstract and categorize the literature in an area to get a better grasp of the area.

ᴹ*Framework Development*

Type

Modeling

Intent

Develop a framework for a research area that organizes the literature of the area and identifies gaps in knowledge that need to be filled.

Motivation

A framework for a research area greatly helps in getting a big picture of the area and making sense of its existing literature.

Context/Applicability

There is a fairly extensive body of published work in the research area. However, a good recent survey of this work is not available. You would like to get a good understanding of the research conducted in the area and its structure.

Description

Use morphological analysis (Zwicky 1967) to form structures (morphologies) of existing information in the subject area. Use the analysis to derive a classification scheme that can serve as a framework for understanding the existing work in the area as well as for exposing the areas that have not received adequate attention.

The development of a good framework is a creative task (see Creativity Patterns, Chapter 7) but the following steps can serve as a guideline:

1. Collect the entire literature or a good sample of the literature to form the literature base.
2. Analyze key ideas and currently known dimensions and parameters in the literature base.
3. Analyze and abstract this information to form a tentative classification scheme.
4. Populate the classification scheme with the literature in the literature base.
5. Examine the contents of the literature in the different categories of the classification scheme to see if the classification scheme needs to be revised.
6. Abstract the concepts of the classification scheme to derive its dimensions.
7. Examine and abstract the relationships between the different dimensions to form an initial version of the framework.

Consequences

The pattern should provide a framework for organizing the literature in the research area. A good framework should help in providing new insights into the research domain and identifying important gaps in the existing research. A good framework can be very useful in surveying a research area and can be a contribution by itself.

Usage Example(s)

1. Abbasi and Chen (2008) formally structure the existing literature in the CMC area to show its knowledge gaps; also see Chapter 13, p. 320.
2. Chen (1976) describes a framework for multilevel views of data and introduces the entity-relationship model using this framework; also see Chapter 13, p. 328.
3. The work in Datta (1998) draws from literature from multiple fields to investigate a problem not currently addressed. The author develops a framework to provide an intellectual structure for the problem addressed; also see Chapter 13, p. 340.
4. McLaren et al. (2011) develop a taxonomy of existing solutions to their problem prior to proceeding with development of their artifact; also see Chapter 13, p. 352.
5. Purao et al. (2003) position their research approach through the development of a framework of machine learning techniques; also see Chapter 13, p. 356.

References

Abbasi, A. and Chen, H. (2008). "CyberGate: A Design Framework and System for Text Analysis of Computer-Mediated Communication." *MIS Quarterly* 32(4): 811–838.

Berners-Lee, T. and Cailliau, R. (1990). "WorldWideWeb: Proposal for a Hypertext Project," URL: http://www.w3.org/Proposal.html (last accessed on January 29, 2015).

Chen, P. (1976). "The Entity-Relationship Model: Toward a Unified View of Data." *ACM Transactions on Database Systems* 1(1): 9–37.

Choobineh, J. and Lo, A. (2005). "CABSYDD: Case-Based System for Database Design." *JMIS* 21(3): 281–314.

Datta, A. (1998). "Automating the Discovery of AS-IS Business Process Models: Probabilistic and Algorithmic Approaches." *Information Systems Research* 9(3): 275–301.

Denning, P. (1968). "The Working Set Model for Program Behavior." *Communications of the ACM* 11(5): 323–333. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 43–48, January, 1983.

McLaren, T., Head, M., Yuan, Y., and Chan, Y. (2011). "A Multilevel Model for Measuring Fit between a Firm's Competitive Strategies and Information Systems Capabilities." *MIS Quarterly* 35(4): 909–929.

Purao, S., Storey, V., and Han, T. (2003). "Improving Analysis Pattern Reuse in Conceptual Design: Augmenting Automated Processes with Supervised Learning." *ISR* 14(3): 269–290.

Vaishnavi, V., Buchanan, G., and Kuechler, W. (1997). "A Data/Knowledge Paradigm for the Modeling and Design of Operations Support Systems." *IEEE Transactions on Knowledge and Data Engineering* 9(2): 275–291.

Zwicky, F. (1967). "The Morphological Approach to Discovery, Invention, Research, and Construction," in F. Zwicky and A.G. Wilson, (Eds.), *New Methods of Thought and Procedure*, New York, NY: Springer-Verlag.

Chapter 10

Suggestion and Development Patterns

The patterns in this chapter are applicable to the *suggestion* as well as *development* phases of the research (see Figure 6.2 in Chapter 6). They can assist in determining the strategies that can be employed to develop a solution to your research problem and in generating knowledge that is of general value.

Suggestion and development patterns are normally employed after you have developed your research problem to a reasonable level. You would now like to proceed to develop a solution and associated theory. You would like to know the different approaches and techniques that you can use to guide your research.

With reference to Figure 6.2, note that the patterns for this chapter are applicable to two phases of the methodology, *suggestion* and *development*. This is due to the fact that in practice iterations between suggestion and development occur many times in the course of a typical design science research (DSR) project. Following an initial *suggestion* phase, a project proceeds to *development*; upon further investigation the development appropriate to the initial suggestion may well prove to be impractical or require resources in excess of those available or may provide information on a new suggestion that is more interesting or more practically implemented. A real-world example of this type of iteration is given in Chapter 2, "An Example of ICT Design Science Research."

The close coupling of the *suggestion* and *development* phases have led us to combine the patterns applicable to those phases rather than attempting to separate them according to some arbitrary criteria.

The following 41 patterns can guide you in different aspects of solution and theory development:

Theory Type

Preliminaries Type

Visionary Type

Extrapolation Type

Analysis Type

Exploration Type

Modeling Type

Generalization Type

Segmentation Type

Combination Type

Development Type

Collaboration Type

- Provocation (p. 274)
- Research process adaptation (p. 275)
- [M]Utilizing expertise (p. 276)

The patterns are classified into 12 types, theory, preliminaries, visionary, extrapolation, analysis, exploration, modeling, generalization, segmentation, combination, development, and collaboration, and are discussed in alphabetic order within each type. Any differentiating information between the related patterns of a certain type will be provided when the patterns of the type are discussed.

The type names are generally self-explanatory. As for the *problem selection and development* patterns, the word "extrapolation" in *extrapolation* type is used in the sense of extrapolation of ideas. In terms of the knowledge contribution framework shown in Figure 2.5 (Chapter 2), the visionary type patterns would be particularly used for an *invention* type of knowledge contribution. For *adaptation* type of knowledge contribution (see Figure 2.5), the patterns that suggest to be particularly useful are the *interdisciplinary solution extrapolation* pattern and the first three *modeling* type patterns. In *improvement* type of knowledge contribution (see Figure 2.5), there is greater need for abstraction and theory development. Thus, the *theory* type and *generalization* type patterns would have higher relevance. Patterns of this type could also be useful for an *adaptation* type of knowledge contribution.

Meta-level patterns in this (and other) chapters are indicated by the superscript [M] preceding the pattern name. The patterns *problem space tools and techniques*, *research community tools and techniques*, *interdisciplinary solution extrapolation*, and *technological approach exemplars* are most naturally used at the development stage of a project, but may also be revisited at the validation stage, when seeking an evaluation technique for research results that will be acceptable to the research community associated with the target journal. *Different perspectives*, *means/ends analysis*, *exploring generalizability*, *sketching solution*, *embedding concepts and techniques*, and *utilizing expertise* while strongly identified with suggestion and development are in fact applicable at any point in the research program.

Research is a creative process and the solution/theory development stage is a particularly creative stage of the research. The patterns listed above can thus only guide you in your solution and theory development. They cannot by themselves generate the solution and the associated theory.

Theory Type Patterns

- Approaches for building theory
- Building design-related explanatory/predictive theory (DREPT)
- Expanding design theories (DTs) with design and measurement models

- Hermeneutical/inductive (H/I) approach
- Hypothetical/deductive (H/D) approach
- Iterative prototyping

The above patterns provide general guidance for developing theory. A well-developed theory is the desired form of output from a DSR project. However, the research output may be a partial or a nascent design theory (along with artifacts such as constructs, models, methods, and instantiations), or even just an artifact in the form of a situated implementation.

The first pattern, *approaches for building theory*, provides a quick overview of four approaches for building theory described next, *DREPT*, *H/I approach*, *H/D approach*, and *iterative prototyping*. The pattern *expanding DTs with design and measurement models* provides guidance on strengthening DTs through a formal framework for linking independent and dependent variables of a design theory with their operationalization in the corresponding implemented artifact.

The different approaches for building theory are very different from each other. Their applicability will be determined by the nature of the research project and familiarity of the researcher with the different approaches. Among the patterns for the different approaches, the *H/D approach* is likely to be most used. This is because it is a relatively simpler approach and is also somewhat similar to the research approach used in natural sciences. *Iterative prototyping* also is an equally straightforward approach. However, it is generally related to system development types of research projects and its multiple iterations are a fundamental part of the approach. The *H/I approach* is difficult to follow unless an artifact with prototyping design documentation already exists that can be experimented on for generating theory. *Building DREPT* (a new approach) is potentially perhaps the strongest pattern but its use requires maturity.

Approaches for Building Theory

Type

Theory

Intent

Get a general understanding of the different approaches for building theory.

Motivation

Since building theory is a goal of DSR, an appropriate direction needs to be selected for building the theory. This needs an understanding of the different approaches for building theory.

Some DSR practitioners question the necessity of deriving theory from a DSR project. However, building theory in the sense we intend it—the capture of knowledge generated in the project—is a desirable goal for all research including DSR. The nature of such theory or even whether building theory is a realistic goal for a given DSR project depends upon the maturity of the existing state of the art in the problem area. Except for the case when the expected knowledge contribution is of the *invention* type (see the DSR knowledge contribution framework—Figure 2.5—in Chapter 2), development of theory in some form would be a goal of the research. In that case, you would like to identify an approach for building theory while solving the research problem.

Context/Applicability

You have developed a DSR problem and would like to now get started on developing the solution for the problem and the associated theory. You would like to get a general guidance on how theory can be developed.

Description

Table 10.1 describes four general approaches for developing theory; see Baldwin and Yadav (1995) for related work.

Table 10.1 Approaches for Developing Theory

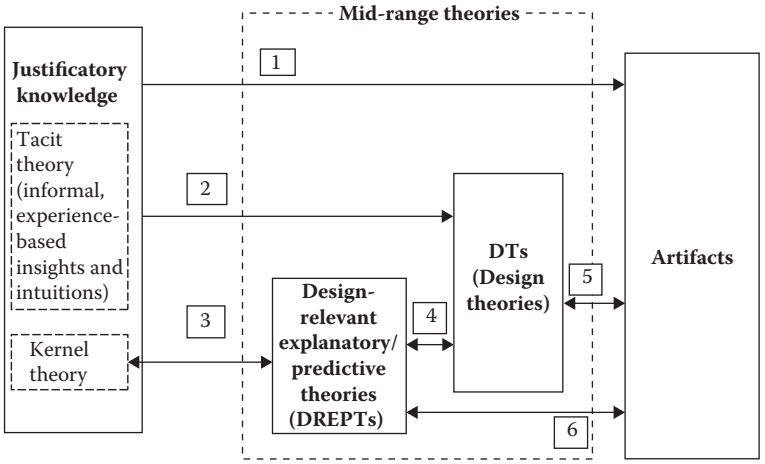
Hypothetical/ Deductive (H/D)	*Hermeneutical/ Inductive (H/I)*	*Iterative Prototyping*	*Design Related Explanatory/ Predictive (DREPT)*
Use intuition logical inference, analysis of prior work, justificatory knowledge, and/or results of theory testing to develop a solution and associated theory.	A prototype and extensive design documentation are created by you or are available to you. Create nascent theory and gain understanding of the mechanisms by which the artifact functions through observation of artifact operation and analysis of design documentation.	1. Build a prototype based on an initial solution and theory. 2. Test the prototype to evaluate the solution. 3. Based on the evidence gathered, revise the solution/ theory and modify the prototype to reflect the revised solution/theory. 4. Iterate through Steps 2 and 3.	Identify a relevant kernel theory and translate its knowledge into design-relevant constructs.

Distinguishing the Four Approaches to Theory Development

Figure 10.1 is a revision of Figure 2.7. It shows all the theory *types* that are shown in Figure 2.7 but for clarity omits the arrow captions that indicate the type of *knowledge* captured for each theory type. The intent of Figure 10.1 is rather to illustrate the different theory development *approaches*—procedural pathways to theory development—rather than knowledge capture.

With reference to Table 10.1, the H/D approach is in essence the development of an artifact as experimental apparatus or proof of concept. Using intuition or working from a theoretical basis or an empirical extension of prior work, the researcher tacitly or explicitly makes the statement: if principles $\{x_n\}$ are true then artifact constructed according to design tenets $\{y_n\}$ will exhibit behaviors $\{z_n\}$. In the H/D approach, the validation of the artifact consists in observations, analysis, and/or measurements that confirm the set of hypothesized propositions or behaviors, $\{z_n\}$. In a strict, positivist interpretation of the H/D approach, the theoretical principles $\{x_n\}$ are either proven or disproven and the project ends. In DSR, the approach is typically more exploratory, and the derivation of design tenets from principles and the deduction of expected behaviors from design tenets might both be questioned, potentially leading to modifications of $\{x_n\}$, $\{y_n\}$, or $\{z_n\}$ and thus increased knowledge.

If the principles guiding the project were design principles from tacit theory or prior design science research projects, then Path 1 in Figure 10.1 would be followed. If the principles were from a non-design area, then they would



Kernel theory: Social, mathematical, and design science theories as well as natural science (e.g., physics, psychology) theories

Figure 10.1 Theory development framework: four general approaches.

necessarily need to be translated to design implications, following either Paths 2 and 5, if design implications were the sole intermediary; Paths 3 and 6, if explanatory implications were the sole intermediary; or Paths 3, 4, and 5, if both explanatory implications and design implications were derived from the source theories.

Again with reference to Table 10.1, the H/I approach is in essence the DSR analog of the "grounded theory" qualitative research method from the social sciences. Just as in grounded theory, the phenomenon—in DSR, the artifact and prototyping design documentation—are analyzed/observed without any (or minimal) theoretical bias, with the intent to induce a theory from the analysis and observations. A significant difference from grounded theory is that in DSR the artifact is developed instead of occurring naturally, introducing a possible source of bias. Methods for minimizing such bias are given in the description section of the pattern for this approach. Assuming an artifact along with its design documentation already exist, then with reference to Figure 10.1, Path 5 in the backward direction would be taken if the primary interest were in design theory, Path 6 in the backward direction would be followed if the primary interest were in explanatory theory, and Paths 5 and 4 would be taken if both types of theory were to be induced (all paths taken in the backward direction).

Iterative prototyping (from Table 10.1) is a third approach to theory construction in DSR. In this approach, an artifact is developed from an intuitive or theoretical basis, just as in the H/D approach; however, the intent of this approach from the outset is to iterate between theory modification based on artifact analysis or observation and artifact modification based on implications of the emended theory. New knowledge comes both from the final products and from the iterative process itself. This approach requires considerable maturity as it contains no built-in stopping rule. In practice, however, a DSR project is begun with a focused interest in either design science knowledge or artifact performance/behavior and this will guide the project. Moreover, the pragmatic demands of publishing or simple resource limitations serve to constrain any DSR project. With reference to Figure 10.1, the paths followed in this approach are identical to those for the H/D approach; however, they are followed in *both* directions, a complete cycle constituting one iteration.

The *DREPT* approach (from Table 10.1) follows Path 3-4-5 or Path 3-6, possibly in the reverse direction as well. This approach is further described in the *building DREPT* pattern.

Consequences

Based on the research area and the research problem, you can assess the suitability of the approaches suggested by this pattern. It is possible that you may need a combination of these approaches or a completely different approach not suggested by this pattern.

Related Pattern(s)

Experimentation (p. 284) (*evaluation and validation* pattern) uses the first three approaches but for the purpose of evaluation and validation.

Building Design-Related Explanatory/ Predictive Theory (DREPT)

Type

Theory

Intent

Provide general guidance on developing design-related explanatory/predictive theory as output of your research.

Motivation

Design-related explanatory/predictive theory constitutes not only the "how" knowledge but also the "why" knowledge—why the prescriptive knowledge resulting from the research should work. This makes it a particularly desirable form of knowledge contribution for incrementally advancing a technology, for example, search engines. This type of contribution has the additional good attribute of being at a higher level of abstraction in the knowledge outputs hierarchy shown in Figure 2.6.

Context/Applicability

For one-time DSR projects, especially in organizational settings, for example, the field design of a method for the efficient merger of two IT departments, the additional effort of developing DREPT may not be justified. Where multiple projects to advance the same technology are envisioned, DREPT can increase the effectiveness of and potentially speed up future development.

Description

A published method for the development of DREPT is given in Kuechler and Vaishnavi (2012)—see Chapter 4; the method assumes that kernel theories suggest the development direction and specifies rigorous derivation of mid-range design theory from the dependent and independent variables of the kernel theory. This method generally follows a H/D approach and takes Paths 3 and 6 or Paths 3, 4, and 5 in the theory development framework shown in Figure 10.1.

There is no reason that DREPT could not be developed using either the *hermenutic/inductive* approach or the *iterative prototyping* approach; however, neither of these alternatives has been explored in the literature.

Consequences

The result of developing DREPT is a better understanding of what forces or phenomena in the inner environment of the artifact are responsible for the effects it exhibits. This can be a significant benefit in suggesting new approaches to the original problem.

Usage Example(s)

1. Kuechler and Vaishnavi (2008)—see Chapter 5—provide a detailed usage example for the use of this pattern.
2. Kuechler and Vaishnavi (2012) provide detailed explication of DREPT for Arazy et al. (2010), Arnott (2006), and Kasper (1996).

Expanding Design Theories (DTs) with Design and Measurement Models

Type

Theory

Intent

Provide guidance on how to fully explore the implications of a design theory for an artifact—with possible modification of the artifact itself—to more accurately link the theory with artifact implementation.

Motivation

A design theory is usually tested in an indirect manner—by evaluating a design artifact that is using the theory. However, there can be serious questions on whether the testing conducted on the design artifact truly tests the design theory. An element of rigor is needed so that the results from artifact testing can be used to judge the validity of the design theory.

Context/Applicability

The artifact that has been implemented in a DSR project is not performing as expected. The theory or insights motivating the design could be wrong; however,

assuming due dilligence in the literature and design phases of the project, it is more likely that design or measurement mappings are flawed.

Description

Enacting this pattern requires the development of an *inner model*, an *outer model*, a *measurement model*, and a *design model* as well as the identification of *measurement items* and *design items*. These terms are defined in Niehaves et al. (2012) along with an example of their explication for an ISDT, formulation of design theory for information systems provided by Walls et al. (1992, 2004). Note that the definitions of inner and outer model used here are not equivalent to Simon's (1996) definitions for similar terms—"inner environment" and "outer environment." The relationships between models is shown in Figure 10.2.

Note

The pattern exercises the forward and reverse paths between ISDTs and Artifacts on Path 5 of the DSR theory development framework shown in Figure 10.1.

Consequences

The use of this pattern could concievably salvage what might appear to be a failed project. In any event, use of this pattern will result in greater understanding of the complex relationship between a design theory and an artifact designed in accordance with that theory.

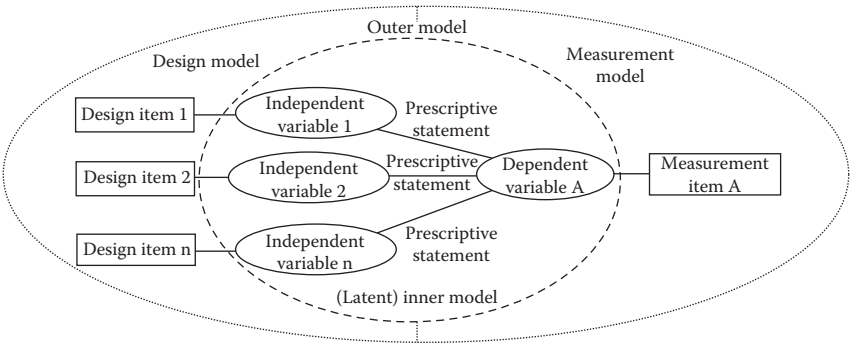


Figure 10.2 Design theory framework. (From Niehaves, B., et al., *7th International Conference on Design Science Research in Information Systems and Technology (DESRIST)*, 354–368, 2012.)

Usage Example(s)

Niehaves et al. (2012) give a well-commented example in the context of an artifact for measuring *virtual social facilitation*. They argue that it is sometimes not fully appreciated that an ISDT metadesign can give rise to an infinite number of artifact implementations. At the validation phase of a DSR project, it may be found that the performance of the artifact may not match theoretically predicted expectations. This may be due to ineffective implementation details rather than theoretical issues; rather, the mapping from possibility space—the meta design—to the actual artifact feature implementing the design may be ineffective. By explicating a measurement model and a design model (Nehaves et al. 2012), the nature of the mappings becomes clear and alternative mappings may be implemented in construction of the artifact to better result.

Hermeneutical/Inductive (H/I) Approach

Type

Theory

Intent

Get an understanding of the H/I approach to building theory and gain guidance in developing theory using this perspective.

Motivation

Prototypes are commonly developed as part of a DSR project. However, they are seldom used for building theory; they are mostly used for demonstration purposes. Proper use of the hermeneutical/inductive approach in building or analyzing a prototype can result in the development of theory in the research area.

Context/Applicability

You are planning to develop a prototype or have access to a prototype and its design documentation and feel the best understanding of the operation of the artifact would be through observation of its operation (a hermeneutical/inductive approach). You would like to ensure that the theory is developed without bias.

Description

Planning, documentation, data collection, and a conscious effort for removing any bias are key features of this approach:

1. If you are constructing the prototype, fully document your design decisions and assumptions. Articulate the reasons behind the decisions and the reasons for rejecting any alternate choices.
2. Separate your roles as prototype builder (if constructing a prototype), observer, and theory builder. Document the design process. Collect data on prototype behavior while varying design features and other parameters of the prototype.
3. To induce a theory:
   a. Write a case study narrative describing the prototyping (experiment) and the data obtained on the prototype behavior.
   b. Seek relationships between prototype design features, parameters, and the results of prototype behavior.
   c. Generalize these relationships.
4. Verify your theory by considering the possibility of alternative theories explaining the data and any contradictory evidence to your theory.

Note

The pattern exercises in the reverse direction paths 5, 6, or {5, 4} of the DSR theory development framework are shown in Figure 10.1. (See the description section of the pattern, *approaches to theory building*.)

Consequences

This pattern presents a systematic approach for developing theory using the hermeneutical/inductive approach. The approach, however, requires considerable effort and time. The conduct of a single research project may only provide an initial set of propositions; the development of theory may require the conduct of many projects possibly using different approaches for theory development (see the pattern *approaches for building theory*). Alternatively, the use of the pattern *iterative prototyping* may be more suitable to your type of research problem.

Hypothetical/Deductive (H/D) Approach

Type

Theory

Intent

Provide an outline of the H/D approach for developing theory in DSR and provide guidance in developing theory using this perspective.

Motivation

This approach generally follows the research model used in the physical sciences and thus draws upon a rich methodolgy knowledge base. The pattern is useful in the third phase of a DSR project where logical inference and deduction must be used to validate the claims of the research (see Figure 2.4).

Context/Applicability

The solution to a DSR problem is suggested by some justificatory knowledge such as kernel or tacit theory (informal experience-based insights and intuitions) from ICT or another discipline. Moreover, you may be familiar with the H/D approach from other positivist fields of study and respect its rigor and defined process.

Description

This approach to develop design principles works from theoretical principles—conjured up from kernel or tacit theory (from informal experience-based insights and intuitions). The implications of a high-level theory or principles for design of an artifact are fully explored and mapped, resulting in an artifact. The actual or expected performance of the artifact is then compared to the theoretical implications (hypotheses) that drove the design. Typically this research approach in DSR is less formal than in some other fields and in the event that artifact performance differs from that hypothesized, both hypotheses and design mappings can be reconsidered. See expecially the pattern *expanding DTs with design and measurement models*. See also the description section of the pattern *approaches to theory building.*

Note

The pattern exercises paths 1, {2, 5}, {3, 6}, or {3, 4, 5} of the DSR theory development framework shown in Figure 10.1.

Consequences

When the DSR project is using theory as its basis, the theory-directed artifact will have considerable explanatory power that can serve as a very productive basis for follow-up projects. When the pattern is used in this manner following the techniques of Kuechler and Vaishnavi (2008), mid-level theories formally derived from the kernel theory may be produced. Even if the approach is based on insights and intuitions, the validity of the design theory can be argued.

Usage Example(s)

1. Abbasi and Chen (2008) are explicit in their intent to develop a design theory for building computer-mediated communication text analysis systems and clearly uses the H/D approach for doing so; also see Chapter 13, p. 320.
2. The research reported in Chen (1976) uses the H/D approach to build a design theory for modeling and designing databases based on the concepts of entities and relationships; also see Chapter 13, p. 328.
3. Choobineh and Lo (2005) use this approach for incremental design theory development in the area of automated conceptual database design; also see Chapter 13, p. 332.
4. Codd (1970) uses this approach to develop the relational model and its associated theory; also see Chapter 13, p. 336.
5. Datta (1998) uses this approach to develop procedures for modeling AS-IS business processes and associated design theory; also see Chapter 13, p. 340.
6. Denning (1968) uses this research approach for developing an initial form of design theory for the management of computer system resources; also see Chapter 13, p. 345.
7. Hoare (1978) uses the approach for developing a basic form of design theory for designing programming languages for effectively using a multiprocessor machine for executing a single task; also see Chapter 13, p. 348.
8. McLaren et al. (2011) combine this approach with iterative prototyping to develop design theory; also see Chapter 13, p. 352.
9. Purao et al. (2003) use this approach to make an incremental addition to design theory for the automation of conceptual design of systems; also see Chapter 13, p. 356.
10. The research described in Vaishnavi et al. (1997) uses the H/D approach to building a design theory for modeling operations support systems; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.
11. Vaishnavi et al. (1980) use this approach to develop a design theory for constructing optimal multi-way search trees; also see Chapter 13, p. 360.

Iterative Prototyping

Type

Theory

Intent

Develop theory in an incremental iterative fashion that addresses your research problem.

Motivation

This is a useful approach especially for design science projects where the goal is to develop a theory for designing a certain class of systems. We may have minimal theory for designing these types of systems. It is through iterative prototyping, observations, evaluations, and refinement that a theory can be incrementally developed.

Context/Applicability

Your research problem is complex. It is not practical to develop theory at a single point in time. It is an incremental approach in which theory is developed iteratively and your problem development and prototype design is carried out to facilitate theory development.

Description

Frame precise research questions

Instead of asking how a system that is more capable than an existing system can be built, ask why certain architecture can do what other architectures cannot do. The reformulated question can guide you to what needs to be documented and what kind of data needs to be collected.

Decide whether you want to validate or invalidate a theory

The decision will affect the requirements of the prototype that you would like to design.

Construct a theory that addresses the problem

A theory generally is a set of propositions that identify units, states of units, and laws or beliefs about the interaction of units in order to explain, predict, and describe observations within some boundary. It includes new ideas and concepts, conceptual frameworks, new methods, and models (e.g., mathematical models, simulation models, and data models). Direct your prototype design and development effort to validate or invalidate the theory.

Construct a design based on the theory

What design flows from the theory? How best can the prototype-design validate or invalidate the theory?

Develop a prototype based on the design

Do not deviate from the chosen design in developing the prototype.

Evaluate the results

Does the data obtained from exercising the prototype support your theory and solve the research problem? Keep a log of your results. Both positive and negative evidence in support of the theory is valuable for getting a better understanding of the research problem.

Refine the problem, theories, and design based on these results

If the design and the resulting prototype validate your theory, then you have achieved your objective. Otherwise, the work already done will have provided to you a better understanding of the implications of your theory and its true applicability to the research problem. Use this improved understanding to revise the problem, theory, and design to correct the deficiencies.

Consequences

This use of this pattern will help you in iteratively improving your understanding of the research problem and in generating a theory that best addresses the problem. Depending upon the state of the art in your problem domain, an understanding of why an artifact does not work as expected can be valuable research information.

Usage Example(s)

1. McLaren et al. (2011) explicitly choose a prototyping approach, knowing that they will come to greater understanding of the problem through iterative construction of a solution; also see Chapter 13, p. 352.

Related Pattern(s)

Suggestion and development pattern, *empirical refinement* (p. 272), is similar but its focus is on solution and system development.

Preliminaries Type Patterns

- [M]Problem space tools and techniques
- [M]Research community tools and techniques

The difference between the above two patterns is that while the first pattern looks at what tools and techniques will be useful purely from the problem space standpoint, the second pattern focuses on the tools and techniques being used by the relevant research community.

[M]*Problem Space Tools and Techniques*

Type

Preliminaries

Intent

Identify tools and techniques applicable to the problem space.

Motivation

It is useful to independently examine the problem space and identify tools and techniques that can be used, without getting biased by the research space or the relevant research community. This can help in expanding the choice for the tools and techniques that can be used for the solution of the research problem.

Context/Applicability

You have identified and developed a research problem. You would like to evaluate the problem space for the tools and techniques that can be used to obtain new knowledge. You would like to be guided by the nature of the phenomenon rather than the traditions of the relevant research community.

Description

1. Study the nature of the phenomenon relevant to the research questions—the problem space.
2. Utilize your general knowledge of tools and techniques to see what tools and techniques can possibly be used to obtain knowledge relevant to the research questions.
3. See if there is a promising tool or technique that has been overlooked by the research community.

4. Revisit the *problem identification and development* phase to see if the research problem needs to be refocused to better utilize the identified tools and techniques.

Consequences

You will choose tools and techniques that you think are appropriate to the solution of your research problem without being directly influenced by the traditions of the relevant research community. The pattern will provide opportunity for the use of applicable techniques that have not been used so far.

Usage Example(s)

1. The final design of the research artifact in Abbasi and Chen (2008) uses many techniques from the general (largely academic) problem space; also see Chapter 13, p. 320.
2. The research reported in Chen (1976) departs from the prevailing research culture and uses graphics for data modeling; also see Chapter 13, p. 328.
3. Machine-learning techniques are used to instantiate theories of expert cognition in conceptual design (Purao et al. 2003); also see Chapter 13, p. 356.
4. Vaishnavi et al. (1997) abstract relevant concepts from the problem domain and incorporate them in their smart object paradigm framework; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Connection to TRIZ Inventive Principles

The pattern is related to the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P10: *prior action* (see Table 6A.1 in Chapter 6), which suggests performing beforehand the full or partial required change of an object. The pattern, like the inventive principle, is suggesting an action that would be useful at the start of a research project; however, unlike the inventive principle, the suggested action is optional.

ᴹ*Research Community Tools and Techniques*

Type

Preliminaries

Intent

Identify the tools and techniques that the relevant research community uses for solving problems similar to your research problem.

Motivation

There is no point in "reinventing the wheel." Therefore, familiarity with the tools and techniques already being used by the research community is useful. This ensures that any research tool or technique relevant to the research problem is not overlooked.

Context/Applicability

You have developed a research problem. You may have independently identified tools and techniques based on the nature of the problem (see the pattern, *problem space tools and techniques*). You would like to identify the tools and techniques commonly used by the relevant research community for solving similar problems so that your research benefits from past work.

Description

1. Use literature search (and the corresponding patterns) to find similar problems in the literature.
2. Find out the tools and techniques that have been used in such problems and assess their effectiveness through the knowledge that has been generated by the use of these techniques.

Consequences

You will gain knowledge about the research tools and techniques that have been used by other researchers for solving similar problems. This will help you in making an informed decision for choosing tools and techniques for solving your research problem.

Usage Example(s)

1. Berners-Lee and Cailliau (1990) propose to use prototyping as the vehicle for conducting research; prototyping is commonly used for conducting similar types of research; also see Chapter 13, p. 325.
2. Choobineh and Lo (2005) use the commonly used research techniques in the field—prototype building followed by experimentation; also see Chapter 13, p. 332.
3. Vaishnavi et al. (1997) built their solutions on top of existing models and concepts; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Connection to TRIZ Inventive Principles

Like the *problem space tools and techniques* pattern, this pattern is also related to the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P10: *prior action* (see Table 6A.1 in Chapter 6). The pattern too suggests useful but optional preliminary action—identifying tools and techniques being used by the relevant research community.

Visionary Type Patterns

- [M]Different perspectives
- Ideas repository
- Pursuing spontaneous ideas

The above three patterns present complementary approaches for invoking and/or utilizing creativity for coming up with new research solutions and approaches. These approaches are particularly useful for the *invention* type of knowledge contribution (see Figure 2.5) but can also be useful for the other two types, *adaptation* and *improvement*.

[M]Different Perspectives

Type

Visionary

Intent

Look at your research problem from different perspectives.

Motivation

The research problem is difficult or an impasse has been reached in attempts to solve the problem. Looking at the problem from a different perspective may be all that it will take to open up a new dimension for the solution of the problem.

Context/Applicability

There is not an obvious approach to the solution of the research problem. You would like to look at the problem in a new way to help find a solution.

Description

Look at the problem from different and unorthodox ways. For example, if the research question is how to make a system more reliable, ask how to prevent it from

being unreliable or less reliable. This may require the use of your creativity (See *creativity* patterns, Chapter 7). By looking at a problem from a novel perspective, an interesting solution may emerge for the problem.

Consequences

This pattern can lead to a novel solution when "using the beaten track" approaches do not work or do not lead to good solutions.

Usage Example(s)

1. Chen (1976) uses the framework presented in his paper to provide a new perspective on data modeling; also see Chapter 13, p. 328.
2. Codd (1970) provides a new perspective on data modeling by raising it to a higher level of abstraction; also see Chapter 13, p. 336.
3. Denning (1968) provides two new perspectives on the research problem: initiating the development of analytical models for program behavior and the use of a unified approach for process scheduling and core memory management; also see Chapter 13, p. 345.
4. Vaishnavi et al. (1997) use this pattern to come up with a novel general solution; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Connection to TRIZ Inventive Principles

This existing DSR pattern (Vaishnavi and Kuechler 2007) can be considered to be an adaptation of the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P17: *another dimension* (see Table 6A.1 in Chapter 6), to the DSR domain; P17 suggests moving an object in two- or three-dimensional space. The connection provides an indirect validation of the pattern since the TRIZ principles are derived from the study and analysis of a vast number of patents in the engineering domain.

Related Pattern(s)

Creativity pattern, [M]*changing attitude* (p. 157), is a complementary pattern.

Ideas Repository

Type

Visionary

Intent

Create a repository of ideas generated over multiple research projects.

Motivation

Novel ideas need to be valued since they drive DSR. Ideas get generated while carrying out a research project but some of them are not relevant or immediately useful. However, proper recording of all such ideas in multiple projects can turn out to be a useful resource in research.

Context/Applicability

While working on your research project you and/or your team members may develop ideas, all of which do not get applied to the current project. You would like to preserve these ideas for future projects.

Description

1. Create a repository for storing ideas in a structured fashion, for example, with key words as attributes so that the ideas can be retrieved easily.
2. Make it a habit to store all ideas developed during each project.
3. Whenever you feel stuck at some time in a research project, consult the ideas repository to see if some solution path gets suggested.

Consequences

Use of the pattern can gradually increase your research and problem-solving prowess.

Connection to TRIZ Inventive Principles

This pattern is related to the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P25: *self-service* (see Table 6A.1 in Chapter 6), which suggests an object serving itself by performing auxiliary helpful functions. This pattern suggests a type of self-service—the creation of an idea repository that the researcher can use for current and future projects.

The pattern is an elaboration of the pattern "re-use ideas"—identified by Gericke (2009) as a proposed DSR pattern that can be transferred from the TRIZ inventive principle "self-service."

Pursuing Spontaneous Ideas

Type

Visionary

Intent

Pursue a spontaneous idea or train of thought in solving the research problem.

Motivation

Spontaneous ideas generated during research need to be immediately followed and further developed. Being part of the creative process they are fragile and fleeting in nature; they need to be nurtured.

Context/Applicability

While consciously working vigorously on your research project, your unconscious mind is also working and can present spontaneous fleeting ideas related to the project. They need to be developed immediately lest they get lost.

Description

1. Whenever you get any spontaneous research idea while working on your project, capture the idea.
2. Develop the idea as much as possible and see if it can be applied to the current project. If it is applicable, pursue the use of the idea to a logical conclusion.
3. If the idea does not seem to be applicable, then store the developed idea in your ideas repository; see the pattern *ideas repository.*

Consequences

Using the pattern, you are leveraging your creativity to make you a better researcher.

Connection to TRIZ Inventive Principles

The pattern is related to the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P21: *rushing through* (see Table 6A.1 in Chapter 6), which suggests the conduct of a process at high speed. The pattern is similarly suggesting quick perusal of a spontaneous idea for the solution of the research problem at hand.

The pattern is an elaboration of the pattern "idea tracking"—identified by Gericke (2009) as a DSR pattern that can be transferred from the corresponding TRIZ inventive principle "rushing through."

Extrapolation Type Pattern

Interdisciplinary solution extrapolation, the only pattern of this type, is obviously relevant to the *adaptation* type of knowledge contribution (see Figure 2.5). However, its use requires broad and rich research interests and background as well as creative abilities particularly in the areas of abstraction and analogies.

ᴹ*Interdisciplinary Solution Extrapolation*

Type

Extrapolation

Intent

Explore the possibility that a solution or solution approach to a problem in one discipline or domain can be applied in or adapted to a different domain.

Motivation

Nontrivial extrapolation of a solution approach from one discipline to another can be a rich and fruitful avenue for finding a significant and interesting solution for a research problem. The type of knowledge contribution made using this approach—*adaptation*—is a separate category in the DSR knowledge contribution framework shown in Figure 2.5.

Context/Applicability

You are aware of a significant solution or solution approach to a problem or a class of problems in a domain different from that of your research problem. You have a hunch that there is some similarity between the problems in the two domains. (Virtually all successful researchers admit to following hunches.)

Description

1. Critically examine the problem in the other domain for which there exists a significant solution.
2. Abstract the problem in the other domain and your own research problem to see if there is any relationship with the two problems at the conceptual level. The relationship may not be obvious and you may need to use your creative abilities to see the relationship (see *creativity* patterns, Chapter 7).
3. If you have found a relationship, attempt to translate or adapt the solution in the other domain to provide a solution to your research problem.

Consequences

Creative application of the pattern can lead to a solution to your problem but the extrapolation should be non-trivial and significant to be judged as research contribution. Clever translation of knowledge in one domain to a different domain can also lead to significant new insights.

Usage Example(s)

1. The research reported in Datta (1998) develops its solution using published work in multiple fields; process modeling, workflow management, computer science (finite state machines); also see Chapter 13, p. 340.
2. Purao et al. (2003) extrapolate the use of machine-learning techniques from their traditional use in fields such as information retrieval to the reuse of conceptual design; also see Chapter 13, p. 356.
3. The smart object paradigm (Vaishnavi et al. 1997) fuses together concepts from databases, software engineering, artificial intelligence, and operating systems; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Analysis Type Patterns

- Easy solution first
- [M]Means/ends analysis

The above two patterns use analysis in some form to develop a solution to the research problem but they do not share any common attributes. *Easy solution first* asks for using an easy solution and analyzing its applicability and the resulting solution. *Means/ends analysis* suggests iterative analysis of the gap between the current state and the desired state in research problem solution and finding ways of bridging the gap.

Easy Solution First

Type

Analysis

Intent

Try an easy solution first.

Motivation

There is no point in reinventing the wheel. Studying and using the easy solution can be an educating and often rewarding experience.

Context/Applicability

You have a research problem for which there seems to be a relatively simple solution. You are not sure whether the simple solution will constitute a significant contribution.

Description

Good research never attempts to make the simple complex. You should therefore not even try to make your solution complex or make it look complex when there seems to be a simple solution to the research problem.

In many cases, the seemingly simple solutions turn out to be rather complex or do not turn out to be appropriate solutions for the problem. Trying the simpler solution first will help you to determine at a small cost the complexity of the problem and its appropriateness for further research efforts.

1. If there seems to be a simple solution to the research problem, use the solution to solve the problem and evaluate the solution.
2. If the simple solution works and provides a reasonable solution, then you need not pursue the problem any further. Depending upon the importance of the problem and whether the solution is nontrivial, the solution may be worth reporting as a research note or paper. The solution will also deepen your understanding of the problem area and help you in coming up with new research questions that may be worth pursuing.
3. If the solution does not work or leads to a solution that is not reasonable, then you have a better foundation for trying a reasonable solution for the problem. At this point, you can utilize your familiarity with the easy solution to see if it can be applied or extended in a certain way for the solution of the problem. If this approach does not work then you need to try a different solution technique.

Consequences

The use of this pattern will provide you with a better confidence that you have not tried to come up with a complex solution while there existed an equally good simple solution for the problem. This information will also help you in motivating your solution at the time of reporting your research.

Usage Example(s)

1. In order to provide a proof of concept, the project proposal (Berners-Lee and Cailliau 1990) attempts a simple solution instead of an elegant solution that would be more complex; also see Chapter 13, p. 325.
2. In Fraser and Vaishnavi (1997), the authors address the problem of having a model that can be used to assess the maturity of a software development organization in incorporating formal specifications in its development process. There already existed a well-known model for measuring the general maturity of an organization called the capability maturity model (CMM) (Paulk et al. 1995). The model was, however, more general than the one the authors were seeking. An organization could be at high maturity level for incorporating formal specifications but at a lower maturity level according to CMM.

   A relatively simple approach to the problem was to adapt the already known maturity model, CMM, to maturity in using formal specifications. Rather than coming up with a new maturity measurement model, the authors tried the simple approach first. Using the simple approach, the authors came up with a model that essentially projected CMM to the use of formal specifications. The resulting model was interesting but not significant by itself. The authors, however, built upon the simple model to construct a stronger model that also suggests strategies for moving to a higher maturity level.
3. Vaishnavi et al. (1980) utilize this pattern by first trying the available solution for a similar problem for binary search trees; also see Chapter 13, p. 360.

Connection to TRIZ Inventive Principles

The pattern is related to the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P11: *cushioning in advance* (see Table 6A.1 in Chapter 6), which suggests preparation of emergency means in advance to compensate for relatively low reliability of an object. In the DSR domain, the current pattern is similarly suggesting to pursue any simple solution to the research problem first to avoid any serious research effort that may not be fruitful because of the availability of a simple solution.

The pattern is similar to "rough solution first"—identified by Gericke (2009) as a DSR pattern that can be transferred from the TRIZ inventive principle "beforehand cushioning."

[M]Means/Ends Analysis

Type

Analysis

Intent

Use means/ends analysis to reach a desired solution state.

Motivation

Knowing precisely the starting and desired solution state of a problem can itself be an important resource. There should be a way to leverage this resource to obtain a solution.

Context/Applicability

You understand and can define preferably in a quantitative manner the desired solution state of the research problem. Similarly, you can define the initial problem state and the states that may be reached in attempts at solving the problem.

Description

This pattern prescribes a process that successively finds the means for narrowing the gap between the end and start states:

1. Precisely describe the desired solution state (end state) and the problem state (start state). Analyze the difference between the two states.
2. Look for methods that can be employed in narrowing the difference between the two states.
3. Employ the most promising method and observe the state that has resulted using the method. If the gap between the end state and the resulting state has narrowed, then the use of the method has been successful. Otherwise, use an alternative method.
4. If there still is a gap between the end state and the state resulting from the use of the method, then treat the resulting state as the new start state and repeat steps 2–4. Otherwise, you have found a solution to your problem.

Consequences

The advantage of the use of the pattern is that it lets you focus on the goal that your research should achieve. This makes your research focused and spurs your creativity (see *creativity* patterns, Chapter 7). The disadvantage is that at some point in the process you may reach a blind alley; at that point you may not be able to find a method that reduces the gap between the end and start states. It may also lead you to a solution that is not direct or elegant.

Usage Example(s)

Vaishnavi et al. (1997) use this meta-level pattern along with the *sketching solution* pattern in developing their solution to their research problem; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Related Pattern(s)

[M]*Sketching solution* (p. 264) can be useful to this pattern.

Exploration Type Patterns

- Exploring the use of crowdsourcing
- [M]Exploring generalizability
- Proactive assessment
- Simulation and exploration

All of the above patterns share the nature of exploration as an attribute but they serve different needs and purposes. Simulation and exploration is a specialized but rich research approach. This approach is extensively used in areas such as communication networks where it is not feasible to be experimenting on real-life environments. Exploring the use of crowdsourcing can be useful for very complex problems or to achieve efficiencies not possible within conventional research environments. Proactive assessment is a pro-active kind of approach that can be used before a solution or approach is fully developed. Exploring generalizability is useful in assessing the generality of a research problem solution while working in the suggestion phase and/or the evaluation phase of research. It is useful to enhance the quality of your research contribution.

Exploring the Use of Crowdsourcing

Type

Exploration

Intent

Explore the use of crowdsourcing for the entire research problem or some aspect of it.

Motivation

Crowdsourcing has become an important method for using collective human intelligence for the solution of difficult or seemingly unsolvable problems.

Context/Applicability

Your research problem is difficult or is such that it either needs collective human intelligence in some aspect of its solution or will greatly benefit from the use of collective human intelligence.

Description

1. Examine your research problem to find if the use of collective human intelligence is critical to the solution of the entire problem or some aspects of it.
2. If you find that the use of collective human intelligence can be beneficial but is not a critical need, then postpone the use of collective human intelligence for future work.
3. If you find that the problem is not solvable using conventional methods—not using collective intelligence—then identify the area(s) of the problem that critically needs the use of collective intelligence. These areas, for example, could include data collection or evaluation.
4. Examine the available cases for successful use of crowdsourcing and see if your project can utilize the existing work.
5. If the existing work on crowdsourcing does not fit with your needs for the project, then see if any of any existing work can be adapted for your use or sketch a research prototype for that purpose.

Consequences

The use of this pattern provides an additional approach for solving a seemingly unsolvable research problem or for making the solution meet real-life needs.

Usage Example(s)

1. Khan (2014) gives a tutorial on "incredible immediacy, affordability and plethora of existing crowdsourcing services for the purpose of eliciting user requirements, evaluating prototypes, understanding context of use or generating new design ideas."
2. Malone (2012) provides an overview of the new kind of collective intelligence that is enabled by the Internet and its potential.

Connection to TRIZ Inventive Principles

The pattern is related to the DSR domain the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P16: *partial or excessive action* (see Table 6A.1 in Chapter 6),

which says that if it is difficult to fully solve a problem using a method then the problem may be easier to solve using more or less of the same method; this principle has been identified by Gericke (2009) as one that can be transferred to DSR. The current pattern is similarly suggesting the use of "partial or excessive action" by utilizing collective human intelligence for solving some aspect of a research problem by making it possible to achieve results by investing less individual effort and excessive overall effort.

[M]Exploring Generalizability

Type

Exploration

Intent

After developing a research solution in a certain environment, change the environment to explore the generalizability of the solution.

Motivation

The generalizability of a research solution increases its value and demonstrates a higher degree of knowledge contribution. It can also point to ways of abstracting the solution so as to lead to a design principle or theory.

Context/Applicability

You have developed a solution to a research problem in a certain environment. You would like to explore the generalizability of the solution. This will make it possible for you to claim a higher degree of knowledge contribution. The pattern can also be used during the evaluation phase of your research.

Description

1. After you have developed a solution for the research problem you are working on, spend time on checking the generalizability of the solution and the possibility of its abstraction.
2. Change the environment in which the solution is working to a different environment and see if the solution still works. The environment could mean things such as the application area of the solution or the operating software/hardware environment needed for the solution.
3. Checking the working of the solution in multiple environments will give you a good understanding of the generalizability of the solution.

4. If the solution is found to be generalizable then see if a more abstract form of the solution can be developed. If you are successful in such effort then you could claim a more valuable knowledge contribution such as design principles or theory. Even if you are not successful in abstraction of the solution, the generalization claim itself can make your contribution stronger.

Consequences

Using this pattern you can make the best of your current research results. Without exploring generalizability you can undermine your research and not make it competitive.

Connection to TRIZ Inventive Principles

The pattern is related to the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P38: *strong oxidants* (see Table 6A.1 in Chapter 6), which asks for replacing common air with oxygen-enriched air, pure oxygen, ozonized oxygen, or ionized oxygen. In the DSR domain, the current pattern is similarly asking for the replacement of the environment of the developed research solution with other environments to explore the solution's generalizability.

Proactive Assessment for Side Effects

Type

Exploration

Intent

Proactively assess the potential side effects of a research approach or solution.

Motivation

Early assessment of a tentative solution to a research problem for negative side effects of the developed solution can help in coming up with a better solution and avoiding wasteful effort in developing the solution.

Context/Applicability

You are at the suggestion phase of the DSR project (see the design science research process model shown in Figure 2.3). You would like to assess a solution for any negative side effects before development of the solution.

Description

1. Explore the possible side effects of the tentative solution when it is fully developed and used.
2. Determine which of these side effects are undesirable. Assess the severity of these side effects.
3. Explore how the solution can be modified to avoid the negative side effects.
4. If you find a modified solution that does not have negative side effects, then develop it further. On the other hand, if you find that the solution cannot be modified to avoid the undesirable side effects then stop pursuing the solution any further and think of a different approach or solution.

Consequences

By proactively assessing a tentative solution for negative consequences, you can make necessary modifications to the solution at an early stage or are at least be saved from potentially wasteful solution development work.

Connection to TRIZ Inventive Principles

The pattern is related to the DSR domain the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P9: *counter action* (see Table 6A.1 in Chapter 6), which asks for counter actions to control the harmful effects of an action that has both beneficial and harmful effects. The current pattern similarly asks for proactive assessment of the developed solution or artifact for any negative effects or side-effects so that they can be counteracted.

The pattern is an elaboration of the pattern "side effect evaluation"—identified by Gericke (2009) as a DSR pattern that can be transferred from the TRIZ inventive principle "preliminary anti-action."

Simulation and Exploration

Type

Exploration

Intent

Understand and predict the behavior of a designed system.

Motivation

The research problem phenomenon is too complex to be fully or accurately modeled. At the same time, it is impossible or infeasible to actually implement a solution for the problem.

Context/Applicability

As part of a solution approach, you have designed a system or would like to explore alternative designs for the system. The system and its design are complex such that you cannot fully understand or predict the behavior of the system without actually implementing the design and building the system. The actual building of the system is, however, infeasible or cumbersome. You would like to understand or predict the behavior of the designed system without having to build the system.

Description

Simulation (Navidi 2006) is a way of imitating the "inner" and the "outer environments" (Simon 1996) in the small, implementing the design using the imitated inner and outer environments, and observing the behavior of the imitated system to understand and predict the behavior of the actual system. Digital computers and simulation languages have greatly facilitated simulation. Use the following steps to conduct simulation:

1. Identify or create objects (parts) that imitate the objects used in the real-life system.
2. Use your design to organize the parts into a system that imitates the desired system. The organization must not violate any organization principles of the inner environment of the real-life system.
3. Subject the imitated system to a range of environments that imitate the outer environment of the real-life system.
4. Observe the behavior of the imitated system to understand or predict the behavior of the real-life system.

Consequences

This pattern will provide you with new knowledge under the following two situations:

1. You fully understand the inner environment but do not fully understand or cannot analyze the system behavioral implications of the known organization principles used in the design.
2. The natural laws governing the inner environment are not fully known. However, abstract properties and laws governing the inner environment are known. The simulation helps in understanding or predicting abstract behavioral properties of the real-life system. Even in the first situation, the understanding of the inner environment and the prediction of behavior are at a certain level of abstraction.

Usage Example(s)

1. You have designed a motor vehicle. It is not possible to fully understand or predict the behavior of the vehicle under varying driving conditions. You simulate the motor vehicle to understand and predict the behavior of the vehicle under a variety of driving conditions that mimic actual driving conditions.
2. You want to design the layout of a bank in terms of the number of tellers, dimensions of the bank, and so on. You simulate the layout using different abstract components that mimic actual components relevant to the design. You implement the simulation on a computer and observe the lengths of lines that will be formed in front of the tellers using a variety of distribution patterns for the arrival of customers.

Modeling Type Patterns

- Modeling existing solutions
- [M]Technological approach exemplars
- Using surrogates
- Using human roles

The first two patterns listed above utilize modeling to build on or utilize existing successful ideas and solutions. The last two patterns also use modeling but for different purposes. Using surrogates in effect models real-life actors or situations that cannot be used in the research, without significant loss to the external validity of the research results. The last pattern, *using human roles*, attempts to model complex human information processing abilities to automate them using computing power.

Modeling Existing Solutions

Type

Modeling

Intent

Model existing solutions to similar problems to develop a solution approach.

Motivation

In many instances, there exist research problems similar to the current problem that have already been solved. It is useful to identify such problems and to extend or adapt their solutions to the solution of the current problem.

Context/Applicability

You would like to find the best approach to the solution of your problem based on the existing solutions for similar problems.

Description (see Simon (1996))

1. Identify problems that are "similar" to your research problem. This requires the ability to see analogies and to abstract problems and solutions.
2. Learn the solution approaches, concepts, and principles used for solving the similar problems.
3. Apply the gained knowledge to the solution of your problem. This may require modifying or adapting the solution possibly using other research patterns.

Consequences

This pattern lets one learn from other problems and their solutions. This can provide useful insights and even a useful solution approach. The risk in using this pattern is that it may hinder finding a unique approach that is not used for the solution of the other similar problems.

Usage Example(s)

1. McLaren et al. (2011) use aspects of prior solutions to their problem as a basis for their improved solution; also see Chapter 13, p. 352.
2. In Vaishnavi et al. (1980), the existing solution for the problem for binary search trees is used as a basis for the solution of the corresponding problem for multi-way search trees; also see Chapter 13, p. 360.

Related Pattern(s)

Ideas repository (p. 230) can be useful to this pattern.

ᴹ*Technological Approach Exemplars*

Type

Modeling

Intent

Use known exemplars to aid in carrying out the current research project.

Motivation

After we have a general idea of the approach to be used in some phase of our research, the approach needs to be operationalized. Unless we want to use some unorthodox method of operationalization, we can operationalize the approach following one or more exemplars of published research that uses the same or similar approach.

Context/Applicability

You have general ideas on how your research problem can be solved but are not sure how these ideas can be operationalized. There exist exemplars in the literature that show how others have solved similar types of problems.

Description

Exemplars are low-level paradigms (Kuhn 1996) or patterns that can be used for the solution of your problem. Here are some steps to serve as a guideline:

1. Find articles that can generally serve as exemplars for the solution of your research problem.
2. Select one or more articles that closely relate to your problem and seem to be influential.
3. Analyze the selected articles to mine a paradigm or pattern that you can use for the conduct of your research.
4. Instantiate the paradigm in terms of your research problem and its requirements.

Consequences

The pattern can help the researcher in gaining tacit and operational knowledge for the conduct of research. It can also serve as a "safe" method for producing knowledge that will be accepted by a paradigmatic research community relatively easily. The disadvantage of the use of the pattern is that it reinforces conformity and may not encourage the conduct of a research in a novel or unorthodox manner.

Usage Example(s)

Vaishnavi et al. (1997) use this meta-level pattern in the evaluation phase of their research to decide what validation techniques they should use in their research; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Using Human Roles

Type

Modeling

Intent

Use human roles for ideas and concepts.

Motivation

Automation of human activity or behavior is a major focus of interest in DSR. The entire field of artificial intelligence is about creating software that mimics some aspect of human intelligence. As such it is quite natural to study human roles for this purpose.

Context/Applicability

Your research is attempting to develop concepts, methods, and so on, in order to automate an activity that is currently performed by human beings. You would like to study and utilize human roles for performing the activity to get ideas and inspiration.

Description

1. Clearly define the activity that your research is targeting to automate.
2. Identify a task activity and a human role for performing the activity that closely resembles the activity of interest to your research.
3. Closely observe the performance of the activity by one or more human beings (subjects). Use audiovisual methods to record the performance of the activity along with verbal protocols that the subject(s) may provide.
4. Analyze the observations and protocols.
5. Use the analysis to aid in the development of concepts, models, and methods that can be used to automate the activity.

Consequences

The use of the pattern can provide useful insights and ideas that can be used to develop the desired solution.

Usage Example(s)

Vaishnavi et al. (1997) describe the study of human supervisory tasks in nuclear power plants to formulate them for automation through meta-level rules; also see

"An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Using Surrogates

Type

Modeling

Intent

Use surrogates to aid your research.

Motivation

Research needs to have external validity, that is, the results of the research should be valid for the phenomenon for which the results are claimed to apply. This would imply that the research should be conducted using the actual phenomenon. In many cases this is not possible or feasible. In such situations, surrogates can be used in place of elements of the phenomenon as long as it can be argued that the surrogates mimic or approximate the actual elements.

Context/Applicability

You are trying to establish a result for something that is either abstract or something that is difficult or costly to work with directly. You would like to explore the use of a surrogate for the subject of your result. Examples of surrogates are structured analysis (surrogate for an informal requirements specification language), students in a graduate programming class (surrogate for programmers), a commercial software package (surrogate for a component of a research prototype), and so on.

Description

1. Analyze the nature of the subject for which you are considering to use a surrogate.
2. Analyze the essential requirements of the subject to serve the intended research purpose.
3. See if the subject or some component of the subject can be substituted by a surrogate that is easier to handle or obtain. Make sure that the surrogate does not violate any research assumptions.
4. Use the surrogate in your research instead of the actual subject.

Consequences

If a suitable surrogate is found, the research may benefit in terms of time, effort, and/or cost. Finding a suitable surrogate, however, may be difficult. Additional care needs to be taken to make sure that the use of the surrogate does not bias the research results.

Usage Example(s)

In Fraser et al. (1991), the authors use structured analysis as a surrogate for an informal requirements specifications language and vienna development method as a surrogate for a formal requirements specifications language.

Connection to TRIZ Inventive Principles

This pattern is related to the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P27: *cheap short-living objects* (see Table 6A.1 in Chapter 6), which suggests replacing an expensive object with a multiple of inexpensive objects with certain qualities such as service life. The pattern is similarly replacing by a surrogate a subject that is either abstract or difficult/costly to work with directly.

Generalization Type Patterns

- Abstracting concepts
- Elegant design
- General solution principle
- Reaching the root

All the above patterns call for the use of generalization/abstraction abilities and are useful in making a significant research contribution. The fourth pattern, reaching the root, however, stands out as one that can lead to you even to an *invention* type of knowledge contribution (see Figure 2.5).

Abstracting Concepts

Type

Generalization

Intent

Abstract concepts from existing solutions to generalize the solutions and to theorize.

Motivation

Abstraction is a very important part of building theory in an area of DSR. In this type of research, a research area generally gets born as a result of an *invention* type of knowledge contribution (see Figure 2.5) and such contribution is usually at the lowest level of abstraction hierarchy of DSR outputs shown in Figure 2.6. As research in the area progresses, the research outputs are expected to be more abstract resulting eventually in a theory for the research area.

Context/Applicability

Solutions to specific instances or special cases of your research problem are available in the literature. You would like to abstract these solutions to form a general solution that will have wider applicability and impact.

Description

Use creativity (see *creativity* patterns, Chapter 7) and the following steps as a guide to develop abstract concepts from existing solutions to specific instances of a general problem in order to develop a solution to the general problem:

1. Analyze and understand the solutions to the special cases of the general problem and the underlying concepts behind these solutions.
2. Generalize the underlying concepts to more abstract but simple general concepts that encompass the underlying concepts in existing solutions.
3. Test the general concepts for their applicability to the solution of the special cases of the general problem. The resulting solution should be as good as the original solutions to the special cases of the general problem. If the solution does not cover all the special cases or does not lead to solutions that are comparable to original existing solutions, then modify the abstractions and/or the level of abstraction.
4. Use the abstract concepts to develop a solution to the general problem.

Consequences

The pattern lets one capitalize on previous work and learn from it to develop a solution to a general problem. The pattern contributes to theory by developing general concepts and other constructs that have general applicability. If successfully applied, the pattern can lead to contributions that have a broad impact.

Usage Example(s)

1. The research in Datta (1998) develops its solution by abstracting the prior work on software process modeling via grammar discovery; also see Chapter 13, p. 340.
2. Vaishnavi et al. (1997) use this pattern to derive the broad specifications of the smart object model; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Elegant Design

Type

Generalization

Intent

Design an artifact that is general and can be described functionally.

Motivation

DSR strives to create solutions that have general applicability and are not plagued by "ifs" and "buts."

Context/Applicability

Your research involves creating an artifact, that is, something that does not exist in nature but must be created. You would like to construct a general design for the artifact, one that can be completely described in functional terms, that is, the properties of the artifact in terms of what it does rather than the details of the construction and organization of the artifact.

Description

First, cast the design problem in the framework of the *sciences of the artificial* artifact (Simon 1996); see "Design Science and Design Science Research" in Chapter 2. View your artifact or the intended artifact as an interface between a given inner environment and an outer environment while meeting a set of desired goals.

The ideal generality of the artifact is achieved when the artifact is independent of the outer environment, that is, the artifact will function even when the outer environment is changed. The ideal in descriptive simplicity is achieved when one

can describe or predict the behavior of the artifact without having to describe how the artifact is constructed or organized using the inner environment.

Ideally, one would like the artifact to be independent of both the outer and inner environments. This would mean that the way the artifact is designed is such that one does not have to describe its inner or outer environment. Even though the ideal may not be achieved, it would be good to let the artifact approximate this ideal. This would constitute a reasonably elegant design.

Consequences

The pattern provides useful insights into generality and simplicity, which make the design of an artifact elegant even if the ideals set by the pattern are not fully realized. Complete external and inner environment independence is an ideal, but the principle is a useful metric for evaluating the elegance of possible designs.

Usage Example(s)

1. The proposed system (Berners-Lee and Cailliau 1990) is general and is described functionally; also see Chapter 13, p. 325.
2. The relational data model (Codd 1970) is a general model that can be functionally described; also see Chapter 13, p. 336.
3. The working set model (Denning 1968) is an elegant model that is general and can be described simply; it can be described in terms of its properties; also see Chapter 13, p. 345.
4. Hoare (1978) proposes a rich language for parallel processing, Communicating Sequential Processes (CSP), which is both simple and general; also see Chapter 13, p. 348.
5. The principle of data hiding (Parnas 1998) makes the design of software module independent of the internal changes in the design of another module, X, on which it depends. This in turn makes the design more general—the module implementation of X can be changed without affecting the design of the module. The principle also improves the descriptive simplicity of the module because the module must be described in terms of what it does rather than its implementation.
6. Consider the design of a watch (Simon 1996). A poorly designed watch would only keep accurate time if it was not moved; a better design would work regardless of movement, but fail if you got it wet; and an even better design would work perfectly even if you went scuba diving with it.

   One can describe a watch by simply saying that it keeps time; one does not have to describe the parts of the clock and how they are organized to say what the clock does. The design of the clock does not depend to a large extent on the exact material that is used for building the clock.

A poorly designed clock would require the user to unscrew the back and manually adjust springs and gears. A better design would only require the user to routinely wind the clock; and an even better design would require nothing of the user at all—the clock would simply tell accurate time.

7. A system modeled using the smart object model (Vaishnavi et al. 1997) has the characteristics of an elegant design as does the smart object model itself; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

General Solution Principle

Type

Generalization

Intent

Construct a general solution for a class of problems.

Motivation

Generalization along with abstraction is hallmark of how DSR in an area progresses. Finding a general solution principle can be a key to this progress.

Context/Applicability

You are trying to develop a general solution for solving a class of problems. You can find a general concept that is common to all the problems in the class.

Description

1. Find a general concept or principle that explains and unifies your class of problems.
2. Find a general problem-solving technique that is appropriate to the problems in the class.
3. Integrate the general concept or principle identified in Step 1 into the problem-solving technique resulting in a general technique for solving your class of problems.
4. Use the generalized technique to develop a general solution for the class of problems.
5. Tune the general technique to specific problems in the class of problems to take advantage of special restrictions or constraints.

Consequences

The use of the pattern can lead to interesting and useful solution to entire classes of problems. The use of the technique, however, may be difficult, as it requires conceptualizing general concepts and principles behind a class of problems and then integrating these concepts into a general solution technique. Dynamic programming technique is particularly amenable to this integration of concepts through its optimality principle.

Usage Example(s)

1. In Chen (1976), the author shows that the entity-relationship model generalizes the prevailing data models, the network model, the relational model, and the entity-set model; also see Chapter 13, p. 328.
2. Denning (1968) develops the working set model as a general model for program behavior that can be used for processor and memory allocation as well as for balancing processor and memory demands; also see Chapter 13, p. 345.
3. CSP (Hoare 1978) can be used to represent solutions for a number of problems related to communication and synchronization of processes; also see Chapter 13, p. 348.
4. The proposed prototype design (Purao et al. 2003) is general in that it can be used in multiple modes; also see Chapter 13, p. 356.
5. Vaishnavi et al. (1997) present a general solution for a class of problems—supporting complex operations environments—that can be instantiated to particular solutions; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.
6. In Vaishnavi et al. (1980), starting with an instance of the research problem for multi-way search trees, the authors developed a general solution principle for a class of problems; also see Chapter 13, p. 360.

Reaching the Root

Type

Generalization

Intent

Create novel concepts by abstracting seminal concrete ideas and reaching the root of those ideas.

Motivation

If we browse the usually huge research literature in any area, we are not likely to find many novel ideas. If we do find any seminal novel idea we should "grab it" and attempt to go to the root of that idea. The root of such idea can be a source for new knowledge.

Context/Applicability

While working on a general research theme or problem you are seeking to come up with new knowledge that is not an end in itself but can be the basis for a new line of thinking.

Description

1. Browse literature related to your research problem. It may be better to include research in related areas as well.
2. See if you come across any new seminal idea presented in a concrete setting that you have not come across before. If the idea is really novel, then it is more likely to be proposed in a concrete setting. (This is like any *invention* type of knowledge contribution (see Figure 2.5) being more likely to be an artifact as situated implementation (see Figure 2.6)).
3. First get fully familiar with the new seminal concrete idea to the extent that it starts looking like your own idea. At this point, this is a major resource that can become a source for new ideas.
4. Spend time on abstracting the idea—going to the "root" of the idea. (Every concrete idea has some abstract root. It is very likely that the inventor of the idea is unfamiliar with this "root" since otherwise the idea would have been presented in an abstract form.) In addition to conscious thinking, you will need to utilize all your creative energies.
5. Once you have grasped the root of the idea, see if you can instantiate the idea in different concrete settings (different from the original concrete setting). Each of these instantiations can be research contributions.
6. Try to present the root of the idea in an abstract form as a new design theory.

Consequences

Successful use of this pattern can distinguish your work from that of others and even lead to some fame!

Usage Example(s)

As a post-doctoral fellow trying to enter a new research area—data structures—the first author (of this book) came across a technical report (Willard 1978) related to

range search trees. The report was not well written and it was very difficult to fully understand; the paper was later published in an expanded form (Willard 1985). But the content of the report looked very different from anything he had seen before. He unconsciously started using this pattern and spent many days abstracting the ideas in the report to come to the "root" of these ideas. He finally realized that the root of the idea is the concept of "layering." He then started exploring if the root idea can be instantiated for other applications. This gave rise to two papers—Vaishnavi (1982) and Vaishnavi and Wood (1982). These papers, however, did not take the final step of the pattern—creating a theory out of the root idea. This was done by Chazelle and Guibas (1986) and its dynamic version was provided by Mehlhorn and Naher (1990).

Segmentation Type Patterns

- Asymmetric focus
- Building blocks
- Divide and conquer with balancing
- Emerging tasks
- Hierarchical design
- [M]Sketching solution
- Static and dynamic parts

All of the above patterns use segmentation in some form to manage complexity. Other than the *emerging tasks* pattern and to some extent the *sketching solution* pattern, the rest are closely related and need to be differentiated.

The *asymmetric focus* pattern complements both the *building blocks* and the *sketching solution* patterns. It combines the ideas of segmentation in the *building blocks* pattern with the idea of giving higher priority to critical components of the research problem; the identification of the critical components is facilitated by the use of the *sketching solution* pattern.

The *divide and conquer with balancing* pattern is a special case of the "building blocks" pattern; in this pattern, the sub-problems need to be identical to the parent problem and also need to be of equal or near equal sizes.

Hierarchical design pattern is related to the *divide and conquer with balancing* pattern. On the one hand, the latter pattern deals with a research problem rather than a complex system as is the case for the *hierarchical design* pattern. On the other hand, the subsystems in the *hierarchical design* pattern need not be identical to the parent system. In the *divide and conquer with balancing* pattern, the sub-problems must be identical to the original problem and on top of that the sub-problems need to be of equal or near equal sizes. The *hierarchical design* pattern is also related to the

building blocks pattern; it is a special case of the latter pattern, which deals with a research problem instead of a system.

The *static and dynamic parts* pattern is slightly different in that its focus is on separation of static and dynamic parts of a research problem and dealing with them separately.

Asymmetric Focus

Type

Segmentation

Intent

Put varying possibly asymmetric focus on the research sub-problems.

Motivation

The amount of effort or focus we put on the sub-problems of a large research problem should be driven by the importance of the sub-problems to contribute to the solution of the entire problem and the degree to which they are critical for demonstrating knowledge contribution.

Context/Applicability

The research problem is large and complex and hence the problem is divided into sub-problems.

Description

1. Divide the problem into sub-problems.
2. Assess the importance of the sub-problems with respect to their importance for the solution of the entire problem.
3. Pursue the solution of the important sub-problems to the depth and detail required to demonstrate the overall problem solution and knowledge contribution.

Consequences

Using the pattern, the research effort gets used more efficiently. For example, actual instantiation or implementation of a sub-problem solution may not be needed to demonstrate the overall solution from a knowledge contribution standpoint.

Usage Example(s)

In Petter and Vaishnavi (2008), the authors develop the experience exchange model for organizations to reuse experiential knowledge in the form of narratives. They, however, only provide a proof-of-concept instantiation of a critical component of the model, the experience exchange library.

Connection to TRIZ Inventive Principles

This pattern is related to the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P4: *asymmetry* (see Table 6A.1 in Chapter 6), which suggests replacing the shape of an object from symmetrical to asymmetrical. The pattern is similarly putting an asymmetric focus on certain research sub-problems instead of paying equal attention to them.

Building Blocks

Type

Segmentation

Intent

Divide the given complex research problem into smaller problems that can form the building blocks for solving the original problem.

Motivation

The research problem is complex but it can be broken down into smaller problems. To manage complexity, it is better to solve the smaller problems first to get a solution for the original bigger problem.

Context/Applicability

The problem is large or complex. It is difficult to fully understand or solve the entire problem. The problem can, however, be decomposed into smaller independent or nearly independent problems that are less complex.

Description

1. Decompose the problem into smaller problems, building blocks
2. Continue decomposing each of the resulting problems until they are understandable and amenable to finding a solution.

3. Solve each of the problems at the lowest level of decomposition.
4. Recursively assemble the solution to smaller problems to find the solution to the parent problems until the original problem is solved.

Consequences

The pattern, if applicable, is very useful for managing complexity and error. It is easier to test the correctness of a solution of a building block than that of the entire problem. It is also relatively simple to modify or change the solution to a simple building block.

Usage Example(s)

The problem solution offered by Abbasi and Chen (2008) is an excellent example of the use of this pattern along with that of *combining partial solutions* (p. 266); also see Chapter 13, p. 312.

Connection to TRIZ Inventive Principles

This existing DSR pattern (Vaishnavi and Kuechler 2007) can be considered to be an adaptation of the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P1: *segmentation* (see Table 6A.1 in Chapter 6), to the DSR domain; P1 suggests dividing an object into independent parts. The connection provides an indirect validation of the pattern since the TRIZ principles are derived from the study and analysis of a vast number of patents in the engineering domain.

Divide and Conquer with Balancing

Type

Segmentation

Intent

Manage complexity by dividing the problem into identical smaller problems.

Motivation

An algorithm needs to be designed for the solution of the problem that can be decomposed into smaller identical problems. Dividing the problem into smaller problems of nearly same size can result in an efficient recursive solution.

Context/Applicability

You are trying to solve a complex research problem. The problem can be divided into a set of similarly sized, smaller but identical problems. The solutions of the smaller-sized problems can be combined into the solution of the original problem.

Description

1. Divide the problem into identical but smaller problems of equal or nearly equal sizes. Preferably, the number of such smaller-sized problems should be two.
2. Examine the smaller problems and see if they can be solved.
3. If the smaller problems are solvable, then combine the solution of the smaller-sized problems into the solution for the original problem.
4. If the smaller problems are still complex, then recursively apply steps 1–3 to get the solution for each of the problems and then combine these solutions to form the solution for the original problem.

Consequences

The technique, if applicable, is an excellent technique for managing complexity.

Usage Example(s)

The pattern has been used with success in the design of a large number of efficient algorithms and data structures. Examples of such algorithms and data structures are binary search algorithm, dynamic data and file structures such as B-trees, and efficient data structures for multidimensional and spatial data such as k-d trees and quad trees (Samet 1989).

Connection to TRIZ Inventive Principles

This existing DSR pattern (Vaishnavi and Kuechler 2007) also can be considered to be an adaptation of the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P1: *segmentation* (see Table 6A.1 in Chapter 6), to the DSR domain; P1 suggests dividing an object into independent parts. The connection provides an indirect validation of the pattern since the TRIZ principles are derived from the study and analysis of a vast number of patents in the engineering domain.

Emerging Tasks

Type

Segmentation

Intent

Identify and carry out the next doable task that can contribute to the solution of the research problem and let the succeeding tasks emerge.

Motivation

We are stuck with a complex non-decomposable research problem. Instead of doing nothing it is better to try "chipping" the problem in some manner. This may lead to a solution or may lead to a deeper understanding of the problem to aid a creative solution.

Context/Applicability

The research problem is large or complex. It is not possible to break up the problem into sub-problems; see *hierarchical design* pattern and *building blocks* pattern. You may not be able to find all the tasks that can contribute to the solution of your problem but you may be able to find the next such doable task.

Description

This pattern uses an incremental and iterative approach along with creativity (see *creativity* patterns, Chapter 7) for the solution of the problem:

1. Instead of thinking about the solution to the entire problem, think about finding a doable task that can contribute to the solution of the problem.
2. While this task is being conducted, see if one or more tasks emerge as the next task. (As you are engaged in performing the first task, you may be unconsciously engaged in finding the next task that can help in solving the problem. Moreover, performance of the current task will provide previously unavailable information to assist in a more complete analysis and may also shed light on the solution of the next part.)
3. Continue this process until the complete solution is found.

Consequences

This task helps in a situation where you are overwhelmed by the complexity or difficulty of the research problem. It lets the use of an incremental approach for the solution of the problem and the use of creativity. The work in finding a task, especially the first task, serves as a vehicle for the "preparation" stage of the creative process (see the *creativity* pattern, *stages of inventive process*). Thus, carrying out the task may provide insights for the solution of the problem. The progress in finding the complete solution may be slow but it will be continuous.

Hierarchical Design

Type

Segmentation

Intent

Design a complex system using the divide and conquer strategy.

Motivation

A complex system needs to be designed. Hierarchical design of the system can manage complexity but first we need to check if this type of design can be used.

Context/Applicability

Your research involves designing a complex system. The system is decomposable or nearly decomposable, which means that the system can be decomposed into subsystems such that the interactions *between* subsystems are weaker than the interactions *within* subsystems.

Description

This pattern designs a system by designing its subsystems and the interactions between the subsystems. By properly designing the subsystems and the interactions between them, you create an artifact that satisfies the desired purpose.

Follow the following steps in designing the system:

1. Divide the system into subsystems (each subsystem should be significantly smaller than the original system).
2. For each subsystem, explore if there is an existing design that can be used. If there exists such a design then use the design.
3. If any subsystem can be designed without further decomposition then design it; otherwise, use the procedure recursively for designing the subsystem.
4. Design the interactions between the subsystems such that the overall system meets the desired objectives.

Use the following guidelines for decomposing a system into its subsystems in the above procedure:

- Reduce the number of interconnections and interactions between the subsystems.

- Reduce the dependency between subsystems. Subsystem A may require input from subsystem B, but ideally it should be capable of operating to at least some degree even if subsystem B fails.

Consequences

Applying this pattern will produce a design that consists of a hierarchical arrangement of subsystems, with each subsystem being reasonably independent of the others. The main advantage of using this pattern is a significant reduction in the complexity of designing the system.

Usage Example(s)

1. The overall proposed system (Berners-Lee and Cailliau 1990) is based on the design of a browser and a server and interaction between the two; also see Chapter 13, p. 325.
2. Consider a system that consists of 10 subsystems, each of which interacts with all the other subsystems (Simon 1996). There are a total of 45 interactions between subsystems. Overall, a total of 55 items (10 systems and 45 interactions) need to be designed. Next, consider a system that consists of 100 subsystems each of which interacts with all the other subsystems. In this case, there are 4950 interactions to be designed, which means that a total of 5050 systems and interactions have to be designed. This means that a system that has 10 times as many subsystems is nearly 100 times as complex! The cause of this rapidly growing complexity is the growth in the number of interactions as the number of subsystems increases.

   The solution to the growth in complexity is to reduce the number of interactions by designing hierarchically. By dividing the system into subsystems, each of which consists of a relatively small number of subsystems, the number of interactions is reduced. For example, we could design the 100-part system to consist of 10 subsystems, each of which consisted of 10 parts.

   There are several advantages to this approach. The system is simpler because the number of connections is dramatically reduced. The system is easier to understand since we can understand it in "chunks" of 10 items at a time, rather than having to understand all 100 parts at once. The system is easier to modify since we can often change the design of a single subsystem without necessarily impacting the other subsystems. In addition, the system is easier to debug since we can diagnose "hierarchically"—checking each subsystem rather than each individual part.

3. Vaishnavi et al. (1997) designed the smart object paradigm in a hierarchal manner separating its logical and architectural views, and separating the paradigm from its instantiation; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Connection to TRIZ Inventive Principles

This existing DSR pattern (Vaishnavi and Kuechler 2007) yet again can be considered to be an adaptation of the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P1: *segmentation* (see Table 6A.1 in Chapter 6), to the DSR domain. P1 suggests dividing an object into independent parts. The connection provides an indirect validation of the pattern since the TRIZ principles are derived from the study and analysis of a vast number of patents in the engineering domain.

ᴹSketching Solution

Type

Segmentation

Intent

Sketch a solution to a complex research problem (or the design of a complex system).

Motivation

Sketching a solution is a good idea for any complex problem. It does not by itself constitute a solution but provides a quick blueprint for a solution approach. The sketch can be used to focus on critical portions that need greater attention.

Context/Applicability

There is danger in overlooking or not giving enough priority to the solution of a critical component of the solution. If the solution to the critical component cannot be found then any effort invested in solving the other components would be wasted.

Description

1. Sketch a solution of the problem involving the use of building blocks and their respective solutions.
2. Verify that the entire problem can be solved if the solution to the identified building blocks is found. Check if there is any missing building block.

3. Identify the critical components (building blocks) whose solution is either critical to the solution of the entire problem or that seem to be difficult problems to solve. Use this information to prioritize the problem components that need to be solved first.

Consequence

This pattern complements the *building blocks* and *hierarchical design* patterns. Its use ensures that your efforts are directed at solving the right sub-problems and in the right order to be most productive in the solution of the complete problem.

Usage Example(s)

1. Berners-Lee and Cailliau (1990) provide an outline of their solution in the proposal; also see Chapter 13, p. 325.
2. Vaishnavi et al. (1997) use this pattern to sketch the use of partial solutions and identify the need for the concept of a monitor; also see Chapter 13, p. 312.

Static and Dynamic Parts

Type

Segmentation

Intent

Separate the static and dynamic parts of the research problem and solve them separately.

Motivation

If the static and dynamic parts of the problem can be separated, then it can lead to better management of the complexity of the problem solution.

Context/Applicability

You are trying to solve a research problem that has time-dependent components. To manage the complexity of the problem, you would like to separate the static parts of the problem from its dynamic parts. It should be possible to separate the static and dynamic portions of the problem.

Description

1. Separate the static and dynamic components of your problem.
2. Find separate basic solutions for the static and dynamic portion of the problem.
3. Combine the two types of solution in an innovative manner to form a seamless overall solution.

Consequences

The pattern lets one concentrate on the static and dynamic portions of the problem separately. The dynamic portion of the problem may be more difficult and may need greater attention. The pattern helps in doing so by separating the static and dynamic issues.

Combination Type Patterns

- Combining partial solutions
- [M]Embedding concepts and techniques
- Integrating techniques

Both *embedding concepts and techniques* and *integrating techniques* attempt to combine techniques but the approach of the former is to embed them in the solution being developed instead of integrating different techniques. The *combining partial solutions* pattern uses solutions at a higher granularity level.

Combining Partial Solutions

Type

Combination

Intent

Find and combine partial solutions to parts of the research problem to form the entire solution.

Motivation

Research should build on existing work and reuse any relevant ideas, concepts, or partial solutions. This makes us focus on the novel or innovative portions of the solution.

Context/Applicability

You cannot find a similar problem for which a solution exists, which you can possibly adapt or modify for your problem solution. There may, however, exist partial solutions that may be relevant to some parts of your problem.

Description

Identify existing solutions that satisfy some of the requirements for the solution of your problem.

1. Select those solutions that are best suited to your problem.
2. Extract concepts and ideas from the chosen solutions that seem to be promising for the solution of your problem.
3. Based on the "mined" concepts and ideas form a tentative solution for your problem.
4. Modify and refine the solution to best suit your problem.

Consequences

The pattern is useful when other techniques for developing a solution do not work. The pattern may be difficult to use because it requires the ability to "mine" ideas from a number of existing solutions and putting them to use in innovative ways.

Usage Example(s)

1. The problem solution offered by Abbasi and Chen (2008) is an excellent example of the use of the building blocks (p. 258) pattern along with this pattern; also see Chapter 13, p. 320.
2. The proposed project (Berners-Lee and Cailliau 1990) builds on the use of hypertext and HTML; also see Chapter 13, p. 325.
3. In solving the problem addressed in Datta (1998), the author draws heavily from the work done on using grammar discovery as a means to software process discovery; also see Chapter 13, p. 340.
4. McLaren et al. (2011) use aspects of prior solutions to their problem as a basis for their improved solution; also see Chapter 13, p. 352.
5. Vaishnavi et al. (1997) bring together concepts from semantic data modeling, rule-based inferencing models, and object-oriented design models into the smart object model; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Related Pattern(s)

Building blocks (p. 258) and *[M]sketching solution* (p. 264) can be useful to this pattern.

[M]Embedding Concepts and Techniques

Type

Combination

Intent

Create concepts and techniques that embed other concepts and techniques.

Motivation

New knowledge usually does not grow as an island but embeds existing knowledge—concepts, techniques—to make it more interesting and meaningful.

Context/Applicability

You need to come up with new concepts and/or techniques as part of your research contribution.

Description

1. Knowledge that is well connected and structured is generally more valuable than unrelated or unstructured knowledge.
2. Create new abstract concepts on top of other concepts including already existing concepts.
3. Seek to grow new knowledge by utilizing and embedding existing knowledge.

Consequences

Using the pattern, knowledge contribution is likely to be better understood and appreciated.

Usage Example(s)

1. Abbasi and Chen (2008) build their research artifact, CyberGate system, from modules containing prior partial solutions; also see Chapter 13, p. 320.
2. In Vaishnavi et al. (1997), the smart object concept builds on top of a number of existing concepts; also see Chapter 13, p. 312.

Connection to TRIZ Inventive Principles

The pattern is an adaptation of the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P7: *nesting* (see Table 6A.1 in Chapter 6), to the DSR domain; P7 suggests placing each object inside another object recursively. The connection provides an indirect validation of the pattern since the TRIZ principles are derived from the study and analysis of a vast number of patents in the engineering domain.

Related Pattern(s)

Hierarchical design (p. 262) can be useful to this pattern.

Integrating Techniques

Type

Combination

Intent

Integrate existing techniques (that include models and solutions) in areas of their respective strengths.

Motivation

Integration of existing techniques is a useful but often challenging approach for developing solution to a research problem.

Context/Applicability

You are working on a research problem for which there does not exist a single technique that can provide a desirable solution. However, there exist multiple techniques that have non-overlapping strengths and weaknesses in their use for solving the problem.

Description

1. Analyze the strengths and weaknesses of each of the techniques in relation to the requirements for the solution of your problem.
2. Design an informal framework (see the *literature search* pattern, *framework development*) that can incorporate the available techniques in the solution of the problem in such a manner that the techniques are used in only those areas where they have strengths for the solution of the problem.

3. Check to see that all aspects of the problem are covered. Fill in any gaps in the solution of the problem.
4. Think of ways of integrating the techniques in the solution of the problem. This may require the creation of new constructs or concepts (see *creativity* patterns, Chapter 7).
5. Think of ways of making the integrated technique conceptually simple and elegant without sacrificing its effectiveness for the solution of the problem.

Consequences

The use of this pattern can lead to useful and significant techniques, models, or solutions. In certain cases, the contribution can cross discipline or paradigm boundary, which is good for the advancement of knowledge but can also make it harder to communicate the results.

Usage Example(s)

1. Abbasi and Chen (2008) develop a synthetic solution using this pattern; also see Chapter 13, p. 320.
2. Work from multiple fields—process modeling, workflow management, computer science—is synthesized in Datta (1998) to provide three novel approaches to real-world process discovery; also see Chapter 13, p. 340.
3. CSP (Hoare 1978) abstracts and integrates a number of ideas for expressing parallel computations; also see Chapter 13, p. 348.
4. The smart object paradigm (Vaishnavi et al. 1997) integrates techniques from data modeling, knowledge representation, and object modeling areas; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Related Pattern(s)

The *literature search* pattern, [M]*framework development* (p. 205), is used in this pattern.

Development Type Patterns

- Continuous work
- Empirical refinement

The above two patterns have development as a common attribute. Continuous work is useful when the research project has already moved to the development phase of the research. Empirical refinement is useful in the iterative development of a system as part of the research problem solution.

Continuous Work

Type

Development

Intent

Work on the research project continuously with minimal interruptions.

Motivation

In the initial stages of solution development—suggestion phase—more time needs to be provided to the unconscious mind to come up with creative ideas and thus it is better not to work on the project in a continuous manner. However, when a solution approach or a tentative solution has been developed, the project benefits from continuous work.

Context/Applicability

The research has moved into the *development* phase (see the design science research process model shown in Figure 2.3), where a tentative solution or research approach needs to be fully developed.

Description

1. Block time such that solution development work can be carried out at a regular pace.
2. Minimize interruption to the work on the project.
3. Work on the project in a disciplined and systematic manner, documenting each step of the work.

Consequences

You will see noticeable and good progress on the research project. As a result, the solution approach being followed gets fully realized.

Connection to TRIZ Inventive Principles

The pattern is an adaptation of the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P20: *continuity of useful action* (see Table 6A.1 in Chapter 6), to the DSR domain. The connection provides an indirect validation of the pattern since the TRIZ principles are derived from the study and analysis of a vast number of patents in the engineering domain.

The pattern is also an elaboration of the pattern "continuous construction process"—identified by Gericke (2009) as a DSR pattern that can be transferred from the TRIZ inventive principle "continuity of useful action."

Empirical Refinement

Type

Development

Intent

Develop a solution to the research problem through iterations of system development, empirical observation, and refinement.

Motivation

When a theory for developing a new type of system is not available, a useful approach is this approach of empirical refinement—building needed knowledge and theory through iterative development, observation, and refinement.

Context/Applicability

Use of system development as a research process is appropriate to the research problem. The research involves designing a complex system in an area where either no theory exists or only fragments of theory are available to guide the design.

Description (see Figure 10.3)

1. Based on the current state of knowledge in the area, define system constructs and construct a conceptual framework, develop a system architecture, analyze and design a system based on the architecture, and build a prototype system based on the design.

   Follow these steps iteratively until an acceptable solution and an understanding of the underlying phenomenon is reached.
2. Observe the behavior of the constructed system under realistic conditions. Collect data that documents the behavior, deficiencies, and other interesting attributes.
3. Use the data collected in Step 2 to get a better understanding of the underlying phenomena and issues. Use this understanding to improve the conceptual framework (along with its constructs) and system architecture to remove the deficiencies. Redesign the system and modify the prototype to reflect the new architecture and conceptual framework.

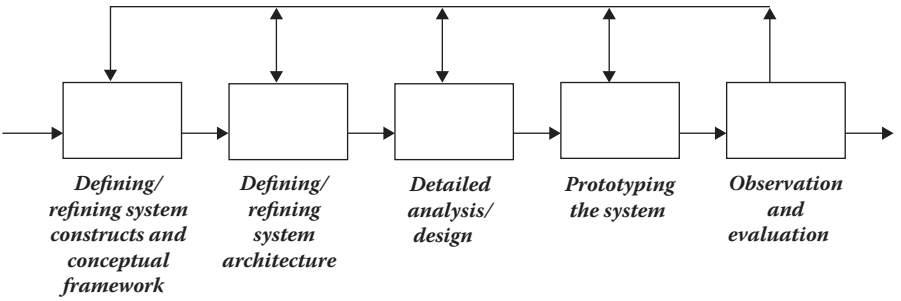


Figure 10.3 Empirical refinement. (Adapted from Nunamaker et al., *Journal of Management Information Systems* 7(3): 89–106, 1991.)

Consequences

You will get a better understanding of the problem domain and appropriate solutions. You may still not have a fully developed theory, but you will be able to develop better systems and a theory will emerge over time.

Usage Example(s)

1. The research conducted in designing the first time-sharing operating system reflects this pattern. There was little understanding of how such a system should be designed and what demands would be placed on it by its users. Starting with an initial rudimentary design that had severe deficiencies, the design was improved through successive cycles of empirical observation and design until an acceptable system was developed. The successive building process itself helped in building a theory.
2. The research in artificial intelligence has generally followed the strategy suggested by this pattern. For example, the progress in developing an acceptable theorem proving system was the result of iterative searching for heuristics and refining the system using the new heuristics.
3. Plans for future work in Berners-Lee and Cailliau (1990) indicate plans for refinement and empirical observation; also see Chapter 13, p. 325.
4. Empirical observation and refinement are planned for the future work of Purao et al. (2003); also see Chapter 13, p. 356.

Related Pattern(s)

Iterative prototyping (p. 223) is similar to this pattern but its focus is for theory development.

Collaboration Type Patterns

- Provocation
- Research process adaptation
- [M]Utilizing expertise

All of the above three patterns are useful for collaboration type of research but otherwise address different issues in collaborative team research. The *provocation* pattern attempts to invoke creativity in a team environment. The *research process adaptation* pattern guides how the team strengths and differences can be used in an optimal fashion. The last pattern, *utilizing expertise*, focusses on expertise to make sure that the team has all the needed expertise or can seek it from outside.

Provocation

Type

Collaboration

Intent

Use provocation to spur creativity within a research team.

Motivation

Provocation can be a helpful tool in making creativity energies flow in a research team.

Context/Applicability

A team is working on the research problem. Creativity needs to be used to develop a novel solution or solution approach.

Description

1. Have a "provocative" session of the research team. Use provocation to get the creative energies flowing within the research team. Let some team members provoke other members by making provocative assumptions about the research problem, suggesting provocative ideas, and drawing provocative conclusions.
2. This will let the team move to an out-of-the-box thinking mode and come up with creative suggestions.
3. The session needs to be conducted in a structured fashion with one member acting as the facilitator and observer. This team member should help in collecting ideas from the session and getting the team back into a saner mode.

Consequences

The team starts producing creative ideas related to the research problem at hand.

Connection to TRIZ Inventive Principles

The pattern is an adaptation of the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P22: *converting harm into benefit* (see Table 6A.1 in Chapter 6), to the DSR domain; P22 suggests using harmful factors, particularly those of the environment of surroundings, to achieve a beneficial effect. The connection provides an indirect validation of the pattern since the TRIZ principles are derived from the study and analysis of a vast number of patents in the engineering domain.

The pattern is an elaboration of the pattern "provocation"—identified by Gericke (2009) as a DSR pattern that can be transferred from the TRIZ inventive principle, "convert harm into benefit."

Related Pattern(s)

Creativity patterns: [M]*brainstorming* (p. 156) and [M]*wild combinations* (p. 161) are useful to this pattern.

Research Process Adaptation

Type

Collaboration

Intent

Adapt the research process to the research team and to any contingency aspects that develop or are discovered.

Motivation

A large research project may not progress exactly according to book. The research team itself may have its own peculiar strengths and weaknesses. Contingencies may develop or may be discovered. The research project should have ways of handling and even taking advantage of any such factors.

Context/Applicability

The research project is large and complex and is being carried out as a team effort. The research is in the *development* phase (see the design science research process model shown in Figure 2.3). You are the leader of the research team.

Description

1. As a leader of the research team, get familiar with and study the research team as well as the environment in which the research is to be carried out.
2. Identify characteristics of the research team such as the culture of the team and the individual strengths and weaknesses of the team members.
3. Adapt the research process to fully adapt to the nature of the research team and to the strengths/weaknesses of its members.
4. Contemplate environmental as well as other situations that may affect how the research is carried out.
5. Develop contingency plans to deal with any such situations and implement them when needed.

Consequences

The research process has been adapted to the research team and the team has confidence in dealing with any contingencies.

Usage Example(s)

The testing of the proof-of-concept artifact in the research reported in Chaturvedi et al. (2011) occurred in a war zone. Because of the uncertainties in a war situation, there was no guarantee that the testing would take place as planned. The research team would have been well served to have contingency testing plans to deal with such a situation.

Connection to TRIZ Inventive Principles

The pattern is an adaptation of the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P3: *local quality* (see Table 6A.1 in Chapter 6), to the DSR domain; P3 suggests letting each part of an object carry out a different function and operate under conditions most suitable to it. The connection provides an indirect validation of the pattern since the TRIZ principles are derived from the study and analysis of a vast number of patents in the engineering domain.

The pattern is an elaboration of the pattern "construction process adaptation"—identified by Gericke (2009) as a DSR pattern that can be transferred from the TRIZ inventive principle "local quality."

[M]Utilizing Expertise

Type

Collaboration

Intent

Utilize the expertise existing within the team or seek from outside any missing expertise needed for the project.

Motivation

Any research project needs specific expertise in certain areas during the different phases of the research. Such expertise within or outside the team needs to be fully utilized.

Context/Applicability

The research project is large or complex. It needs technical and other types of expertise to fully accomplish its task.

Description

1. Find out what are all the areas of expertise needed for the completion of the research project.
2. Check if the team members have the needed expertise.
3. If there are gaps in the available expertise within the team, then find out if such expertise can be sought outside the team.
4. If the needed expertise is not available even outside the team, then the suitability of the research project is in question and the project may need to be scrapped.
5. If the needed expertise is available outside the team, then find out the extent to which such expertise will need to be utilized in the project. If the available outside expertise is needed to be used extensively, then explore expansion of the team to include an outside member with such expertise.
6. Carry out the project with full utilization of the needed expertise within or outside the team.

Consequences

Full utilization of the expertise needed for the research project will ensure that the quality of the work carried out is high.

Usage Example(s)

A published article seldom discusses the issues of expertise needed for completion of the reported research work. One can, however, discern such information from the nature of the work and the backgrounds of the authors. The work reported in

Chaturvedi et al. (2011) seems to have needed expertise on simulation of strategic decision-making, which only one of the authors, Paul Drnevich, seems to have. It is possible that this person was included in the team to provide this expertise.

Connection to TRIZ Inventive Principles

The pattern is an adaptation of the TRIZ inventive principle (Altschuller 2005; Gericke 2009), P24: *intermediary* (see Table 6A.1 in Chapter 6), to the DSR domain; P24 suggests using an intermediary carrier article or intermediary process. The connection provides an indirect validation of the pattern since the TRIZ principles are derived from the study and analysis of a vast number of patents in the engineering domain.

The pattern is also an elaboration of the pattern "intermediary"—identified by Gericke (2009) as a DSR pattern that can be transferred from the corresponding TRIZ inventive principle (Principle 24).

References

Abbasi, A. and Chen, H. (2008). "CyberGate: A Design Framework and System for Text Analysis of Computer-Mediated Communication." *MIS Quarterly* 32(4): 811–838.

Altschuller, G.S. (2005). *The Innovation Algorithm: TRIZ, Systematic Innovation and Technical Creativity*. Worcester, MA: Technical Innovation Center. ISBN 0-9640740-4-4.

Arazy, O., Kumar, N., and Shapira, B. (2010). "A Theory-Driven Design Framework for Social Recommender Systems." *Journal of the Association for Information Systems (JAIS)* 11(9): 455–490.

Arnott, D. (2006). "Cognitive Biases and Decision Support Systems Development: A Design Science Approach." *Information Systems Journal* 16: 55–78.

Baldwin, D. and Yadav, S. (1995). "The Process of Research Investigations in Artificial Intelligence—An Unified View." *IEEE Transactions on Systems, Man, and Cybernetics* 25(5): 852–861.

Berners-Lee, T. and Cailliau, R. (1990). "WorldWideWeb: Proposal for a Hypertext Project." URL: http://www.w3.org/Proposal.html (last accessed on January 29, 2015).

Chaturvedi, A.R., Dolk, D.R., and Drnevish, P.L. (2011). "Design Principles for Virtual Worlds." *MIS Quarterly* 35(3): 673–684.

Chazelle, B. and Guibas, L. (1986). "Fractional Cascading: I, A Data Structuring Technique; II, Applications." *Algorithmica* 1: 133–191.

Chen, P. (1976). "The Entity-Relationship Model: Toward a Unified View of Data." *ACM Transactions on Database Systems* 1(1): 9–37.

Choobineh, J. and Lo, A. (2005). "CABSYDD: Case-Based System for Database Design." *JMIS* 21(3): 281–314.

Codd, E.F. (1970). "A Relational Model of Data for Large Shared Data Bank." *Communications of the ACM* 13(6): 377–387. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 64–69, January, 1983.

Datta, A. (1998). "Automating the Discovery of AS-IS Business Process Models: Probabilistic and Algorithmic Approaches." *Information Systems Research* 9(3): 275–301.

Denning, P. (1968). "The Working Set Model for Program Behavior." *Communications of the ACM* 11(5): 323–333. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 43–48, January, 1983.

Fraser, M. and Vaishnavi, V. (1997). "A Formal Specification Maturity Model." *Communications of the ACM* 40(12): 95–103.

Fraser, M., Kumar, K., and Vaishnavi, V. (1991). "Informal and Formal Requirements Specification Languages: Bridging the Gap." *IEEE Transactions on Systems* 17: 454–466.

Gericke, A. (2009). "Problem Solving Patterns in Design Science Research—Learning from Engineering." *ECIS 2009 Proceedings*.

Hoare, C. (1978). "Communicating Sequential Processes." *Communications of the ACM* 21(8): 666–677. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 100–106, January 1983.

Kasper, G. (1996). "A Theory of Decision Support System Design for User Calibration." *Information Systems Research* 7(2): 215–232.

Khan, V.J. (2014). "Harvesting Crowdsourcing for User & Design Research." URL: http://2014.hci.international/t23 (last accessed on January 29, 2015).

Kuechler, W. and Vaishnavi, V. (2008). "On Theory Development in Design Science Research: Anatomy of a Research Project." *European Journal of Information Systems* 17(5): 1–23.

Kuechler, W. and Vaishnavi, V. (2012). "A Framework for Theory Development in Design Science Research: Multiple Perspectives." *Journal of the Association for Information Systems (JAIS)* 13(6): 395–423.

Kuhn, T. (1962/96). *The Structure of Scientific Revolutions.* Third edition, Chicago, IL: The University of Chicago Press.

Malone (2012). "Collective Intelligence". URL: http://edge.org/conversation/collective-intelligence (last accessed on January 29, 2015).

McLaren, T., Head, M., Yuan, Y., and Chan, Y. (2011). "A Multilevel Model for Measuring Fit between a Firm's Competitive Strategies and Information Systems Capabilities." *MIS Quarterly* 35(4): 909–929.

Mehlhorn, K. and Naher, S. (1990). "Dynamic Fractional Cascading." *Algorithmica* 5(1–4): 215–241.

Navidi, W. (2006). *Statistics for Engineers and Scientists*, Boston, Mass: McGraw Hill.

Niehaves, B., Ortbach, K., and Tavakoli, A. (2012). "On the Relationship between the IT Artifact and Design Theory: The Case of Virtual Social Facilitation." *7th International Conference on Design Science Research in Information Systems and Technology (DESRIST)*, Las Vegas, NV, Springer, 354–368.

Nunamaker, J., Chen, M., and Purdin, T. (1991). "Systems Development in Information Systems Research." *Journal of Management Information Systems* 7(3): 89–106.

Parnas, D. (1998). "Successful Software Engineering Research." *Software Engineering Notes* 23(3): 64–68.

Paulk, M., Weber, C., Curtis, B., and Chrissis, M. (1995). *Capability Maturity Model: Guidelines for Improving the Software Process.* Reading, MA: Addison-Wesley Publishing Company.

Petter, S. and Vaishnavi, V.K. (2008). "Facilitating Experience Reuse among Software Project Managers." *Information Sciences: An International Journal* 178(7): April, 1783–1802.

Purao, S., Storey, V., and Han, T. (2003). "Improving Analysis Pattern Reuse in Conceptual Design: Augmenting Automated Processes with Supervised Learning." *ISR* 14(3): 269–290.

Samet, H. (1989). *The Design and Analysis of Spatial Data Structures.* Addison-Wesley Publishing Company.

Simon, H. (1996). *The Sciences of the Artificial*, Third Edition. Cambridge, MA: MIT Press.

Vaishnavi, V. and Kuechler, W. (2007). *Design Science Research Methods and Patterns*. Boca Raton-New York: Auerbach Publications.

Vaishnavi, V.K. (1982). "Computing Point Enclosures." *IEEE Transactions on Computers* C-31(1): 22–29.

Vaishnavi, V.K. and Wood, D. (1982). "Rectilinear Line Segment Intersection, Layered Segment Trees, and Dynamization." *Journal of Algorithms* 3(2): 160–176.

Vaishnavi, V., Buchanan, G., and Kuechler, W. (1997). "A Data/Knowledge Paradigm for the Modeling and Design of Operations Support Systems." *IEEE Transactions on Knowledge and Data Engineering* 9(2): 275–291.

Vaishnavi, V., Kriegel, H., and Wood, D. (1980). "Optimum Multiway Search Trees." *Acta Informatica* 14(2): 119–133.

Willard, D.E. (1985). "New Data Structures for Orthogonal Queries." *SIAM Journal on Computing* 14(1): 232–253.

Willard, D.E. (1978). *New Data Structures for Orthogonal Range Queries*, Technical Report, Harvard University.

Chapter 11

Evaluation and Validation Patterns

The patterns in this chapter are applicable to the *evaluation* phase of the research (see Figure 6.2 in Chapter 6). You have developed a solution that you think to be correct and you have a hypothesized a number of claims about your solution. You would like to evaluate and validate your solution and the claims about the solution that will be acceptable to the research community.

The following seven patterns appearing in alphabetic order provide vehicles for the evaluation and validation of your solution:

The above related patterns vary in terms of their appropriateness and the strength with which they can establish the validity of a solution. The *demonstration* pattern provides a weaker form of validation. It may however be appropriate if your solution is novel and solves a problem for which no solution exists such as for the *invention* type of knowledge contribution (see Figure 2.5). On the other extreme, the *mathematical proofs* pattern provides the strongest form of validation but a mathematical proof needs to be thoroughly checked to make sure that it does not have any flaw.

The strength of the *logical reasoning* pattern depends on the strength and preciseness of its arguments and assumptions. It is generally an alternative or supplement to the use of *demonstration*, *experimentation*, *mathematical proofs*, and *simulation* patterns. *Experimentation* and *simulation* patterns are useful when the problem is complex and not amenable to a mathematical proof. The use of the *using metrics* pattern is valuable in *experimentation*, *simulation*, and *mathematical proofs* patterns. It helps in quantifying the claims about the solution. The *benchmarking* pattern is a weaker form of the *using metrics* pattern and is useful along with the *experimentation* and *simulation* patterns; it is used when suitable metrics are not available.

The use of one or more of the preceding patterns can help you in convincing yourself and the research community of the validity and value of your solution. This in turn is very important for publishing your results.

Benchmarking

Intent

Use an available benchmark to show that your solution has reasonable performance or is better than some other available solution.

Motivation

Benchmarking provides a vehicle for the objective evaluation of a solution or comparison of different solutions (Tichy 1998). This makes it easy to verify that a claimed solution really solves a problem or to show that a certain solution is better than other existing solutions.

Context/Applicability

There is no established metric available that you can use to measure the performance of your solution (see *using metrics* pattern). You would like to show that the performance of your solution is reasonable or better than some available solution. The research community has, however, developed a benchmark for evaluating solutions to your class of problems. If no benchmark is available, you can create a test scenario or a class of such scenarios that you can use to evaluate your solution as well as any other available solution.

Description

1. Identify the benchmark that you can use to evaluate and validate your solution. If no benchmark is available then you may create your own benchmark. In this case, however, you need to establish that the new benchmark has some independent validity and is not biased toward your solution.

2. Use the benchmark to show the merit of your solution. If a solution to the research problem does not exist, then you need to show that your solution meets the criteria specified in the benchmark for a reasonable solution to the problem. If there is an existing solution to the problem, then you need to show using the benchmark that your solution is a better solution to the problem than the existing solution(s).

Consequences

Using a benchmark, preferably one established by an appropriate research community, provides a quick vehicle to establish the value of your solution.

Usage Example(s)

McLaren et al. (2011) actually develop a benchmark for evaluation of their artifact, in addition to using existing benchmarks from practice; also see Chapter 13, p. 352.

Demonstration

Intent

Demonstrate that your solution is realizable and valid.

Motivation

Instantiation of a solution for a research problem is needed to show that the solution is realizable. Careful testing of the solution can demonstrate the validity of the solution.

Context/Applicability

You have designed the solution for a problem. The problem or the solution is such that it is not possible to mathematically validate the solution. You would still like to demonstrate that the solution is realizable and works for a set of predefined situations. The pattern is particularly relevant when the demonstration of a solution itself would be considered a knowledge contribution.

Description

1. Construct (instantiate) the solution. This may mean construction of a prototype for the solution. Construction of the solution will show that the solution is realizable.

2. Demonstrate that the constructed solution is valid for a set of predefined situations. These situations should be predefined and not created to suit the solution. They should be constructed to exercise the problem variations.

Consequences

The demonstration of the solution may show inadequacies of the solution. On the other hand, it may show that the solution is feasible and acceptable. Exhaustive testing of the solution will increase the confidence in the solution. If the test situations are designed properly, then the construction of the solution and its testing for these situations can demonstrate the validity of the solution.

Usage Example(s)

1. Berners-Lee and Cailliau (1990) propose to demonstrate the solution through a prototype; also see Chapter 13, p. 325.
2. Chen (1976) demonstrates the use of the entity-relationship model for data base design and the use of the proposed diagrammatic technique with the use of an example; also see Chapter 13, p. 328.
3. The developed system (Choobineh and Lo 2005) is validated through an expert evaluation of a demonstration of the system by two expert designers; also see Chapter 13, p. 332.
4. The various attributes of the new model (Codd 1970) are demonstrated through an example; also see Chapter 13, p. 336.
5. Datta (1998) provides a walkthrough of a simple case to show the merits of the process activity graphs (PAGs) relative to the metrics used; also see Chapter 13, p. 340.
6. The versatility of Communicating Sequential Processes (CSP) (Hoare 1978) is shown by using the language for expressing the solutions to many classical programming problems; also see Chapter 13, p. 348.
7. The proposed solution (Purao et al. 2003) is demonstrated through the construction and exercise of a prototype; also see Chapter 13, p. 356.
8. Vaishnavi et al. (1997) use demonstration through examples and cases as a vehicle for evaluation; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Experimentation

Intent

Use experimentation to validate or reject a set of hypotheses associated with the claims about your solution.

Motivation

In many cases, it is not possible to mathematically prove the performance and/or other claims for a solution. Experimentation is the next best vehicle for generating evidence to support or reject claims made with respect to a solution (Zelkowitz and Wallace 1998).

Context/Applicability

You have developed a set of hypotheses related to the claims about your solution (usually a system). You cannot prove these hypotheses mathematically. You need to generate data from the system and then use this data to validate or reject your hypotheses.

Description

The nature of experiment for the validation of hypotheses depends on the type of experiment. The types are in turn dependent on the approach used in developing the solution. Table 11.1 outlines the different types of developmental contexts and motivations, and the corresponding validity criteria for hypotheses testing; see Baldwin and Yadav (1995) for related work.

Hypothetical/deductive experimentation involves constructing a prototype for the sole purpose of testing a set of kernel- or tacit theory-based hypotheses. There is, however, a danger of some bias in the creation of the prototype. It may not be possible to completely eliminate the bias and so stating the bias that may affect the results is important.

One can try to minimize the bias by use of the *hermeneutic/inductive* approach in which an artifact and its documentation already exist or the artifact is constructed without reference to theory; rather, the construction is guided by intuition and prior experience. If a prototype is built as part of the project then understanding of the mechanisms by which the artifact functions, and possibly theory, can be built inductively by observing the prototype during construction and operation. If the prototype and construction documentation are pre-existing, nascent theory and the mechanisms by which the artifact functions can be induced through observation of the prototype and analysis of the documentation. Validation, in this instance, is assurance of external and internal validity of the observations made during artifact construction.

In *iterative prototyping* development, the artifact is constructed as a solution for a problem, frequently inside an organization, and validation consists of observations and measurements of the artifact as a solution to the problem. Construction is typically iterative as the artifact is field tested and revised (Sein et al. 2011).

In the approach we have labeled *improvement over prior systems*, the intent is to construct an artifact with improved performance on a specific task or related set of tasks. Other similar artifacts (systems) have been constructed in the past, and

Table 11.1 Approaches for Experimentation

Hypothetical/ Deductive	*Hermeneutical/ Inductive*	*Iterative Prototyping*	*Improvement over Prior Systems*
Use intuition, results of past experiments, and a literature review to build the system with the intent of testing a set of hypotheses. Testing the system under varying environments is the experiment. *Artifact results are compared to theoretically derived hypotheses.*	Build the system without prior theory. Develop hypotheses inductively from observing the prototype and analyzing the prototyping documentation. Developing the system is the experiment. *Analyze the prototyping documentation to qualitatively accept or reject the hypotheses.* *The artifact is a proof of concept. Validation of successful operation is required.*	Build a prototype based on an initial set of hypotheses. As the prototyping progresses, you will get a deeper knowledge of the problem. Use this knowledge to modify the hypotheses and the prototype guided by the revised hypotheses. Developing the prototype is the experiment. *Use documentary evidence from the prototype to accept or reject the hypotheses.* *The artifact is used as it is iteratively constructed. Validation criteria are task performance in the field.*	Develop a solution and hypotheses from previously developed systems. Observing past systems is the experiment. *For building a better "mousetrap," validation requires the performance of the artifact to be compared against prior artifacts performing the same or similar functions.*

validation, in this case, consists in task performance comparison of the new artifact with prior artifacts.

Note that the *developmental contexts and motivations* discussed earlier are not mutually exclusive. For example, refinement or improvement of an artifact for which previous versions exist can be driven by theoretical developments in some field (*hypothetical/deductive*). In such cases, the validity criterion for both environments can be demonstrated; however, it is more common to position the artifact (and anticipate publication) within a specific community where one validation type will predominate.

In general, an experiment must satisfy the following criteria that have a bearing on the confidence or generality of the results established by the experiment:

Construct Validity: The surrogates for constructs that cannot be readily observed in the experiment must be valid substitutes.

Internal Validity: The experiment must not involve constructs that influence the observed behavior other than those that are part of the hypotheses.

External Validity: If the results of the experiment are supposed to be general but are tested in a simulated limited environment, one should be able to argue that the results are generalizable.

Reliability: The experiment should be replicable.

Consequences

The pattern will help in establishing results associated with the solution of the research problem in situations where collecting and analyzing data is the only feasible method of validation.

Usage Example(s)

1. Abbasi and Chen (2008) use carefully controlled experiments for validation of their research artifact performance; also see Chapter 13, p. 320.
2. An experiment is used to verify the effectiveness of the proposed system and its improved performance over prior tools (Choobineh and Lo 2005); also see Chapter 13, p. 332.
3. McLaren et al. (2011) validate their artifact using experimentation; also see Chapter 13, p. 352.
4. A formal experiment is conducted to evaluate the performance of the constructed prototype (Purao et al. 2003); also see Chapter 13, p. 356.

Related Pattern(s)

The *suggestion and development* pattern, *approaches for building theory* (p. 213), uses the first three approaches of this pattern but its focus is on theory development.

Logical Reasoning

Intent

Use logical reasoning to argue the validity of your solution.

Motivation

The use of logical reasoning is generally useful for any type of solution since it provides an insight into why the solution should work or have any claimed attribute. It is particularly useful to support the results obtained from experimentation or simulation.

Context/Applicability

It is not possible to use a formal mathematical proof to establish the validity of your solution. The problem may be too complex or it may not be possible to cast the problem and the solution criteria in a formal framework. The constructs and assumptions of the problem are, however, precise enough that a logical argument can be built for the hypothesized claims about the solution. This pattern could serve as a supplement or alternative to the experimental evaluation and validation of the solution. It can also be used to supplement the approaches of demonstration, mathematical proofs, or simulation.

Description

This is usually a weaker form of validating a solution than either using a mathematical proof or using experimental validation unless it is used to supplement other techniques. The steps for this form of validation are as follows:

1. Identify assumptions ("axioms") related to your research problem that are either known to be true or can be argued to be valid assumptions possibly using empirical data.
2. Identify rules ("deduction rules") related to your problem or solution that are either known to be true or can be argued to be valid possibly with the aid of empirical data.
3. Build a logical path from the assumptions (axioms) to the claims that you are making about the solution (hypotheses) using the deduction rules that you have identified.

Consequences

On one extreme, when the axioms, deduction rules, and the claims about the solution can be stated precisely and there is no vagueness in showing that the claims follow logically from the axioms, the technique is a mathematical proof for validation (see *mathematical proofs* pattern). On the other extreme, the axioms, the deduction rules, or the logical argumentation may be vague; in this case the pattern does not serve much value for validation. In this case, one should try the experimental method (*experimentation* pattern) or a simulation method (*simulation* pattern). There may, however, be a middle ground where this pattern may provide a reasonable support for the validation of the proposed solution.

Usage Example(s)

1. Chen (1976) uses logical reasoning to show that the E-R model is a generalization of the three existing data models; also see Chapter 13, p. 328.
2. Datta (1998) argues for the reasonableness of the metrics he uses for the evaluation of the strategies he proposes; also see Chapter 13, p. 340.
3. Denning (1968) uses logical reasoning to argue the usefulness of the entity set model and the correctness of its founding assumptions; also see Chapter 13, p. 345.
4. In Fraser and Vaishnavi (1997), the authors build a logical argument for showing why a certain strategy should have a potential to result in a certain maturity level of an organization for incorporating formal specifications in its software development process. This provides an internal validation of the proposed model.
5. Hoare (1978) uses logical reasoning to motivate CSP and its contribution; also see Chapter 13, p. 348.
6. Vaishnavi et al. (1997) provide logical reasons to convince the reader that their paper is making a significant contribution; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Mathematical Proofs

Intent

Prove mathematically the claims that you are making about the solution that you have developed for the research problem.

Motivation

Use of mathematical proofs to prove the claims being made about a solution to a research problem is always preferred since it is a strong vehicle for establishing any claim. However, it cannot be used in all problem situations because of the nature of a problem and/or the claims needed to be established.

Context/Applicability

The hypothesized claims for your solution can be expressed quantitatively or formally and the essential aspects of the problem and the solution can be expressed formally in a closed logical system.

Description

1. Precisely express the hypothesized claims about your solution quantitatively or formally.
2. Cast the claim to be proven as a *theorem* in a well-defined closed formal logical system.
3. Prove any auxiliary results (lemmas) that may aid in proving your theorem about the hypothesized claims about the solution.
4. Prove the claims theorem possibly using the already proven lemmas.

Consequences

This pattern provides the strongest form of validation of the claims that you have made about your solution. This validation is even stronger than experimental validation (see *experimentation* pattern). Great care however needs to be taken to making sure that there is no flaw in the mathematical proofs. Mathematical proofs can be long and/or complex; they need to be thoroughly examined for their correctness.

Usage Example(s)

Vaishnavi et al. (1980) use mathematical proofs to show the correctness and (time and space) complexity of their proposed algorithms; also see Chapter 13, p. 360.

Simulation

Intent

Use simulation to evaluate and validate your solution to the research problem.

Motivation

Simulation generally involves modeling a phenomenon on a computer. When the modeling is done correctly, it provides an efficient and useful technique for testing the performance of a solution related to the phenomenon. It is the obvious choice when it is not possible or feasible to use the actual phenomenon for experimentation (Kleindorfer et al. 1998; Simon 1996).

Context/Applicability

The research problem is complex such that your solution cannot be mathematically proven to be valid. The evaluation and validation of your solution in the real-life setting is either not feasible or is costly. However, the problem and its solution can be accurately modeled on a computer.

Description (see Navidi (2006))

1. Develop the conceptual model of the problem and its solution that will be simulated on a computer. This will involve deciding what entities and their interactions need to be captured in the simulation whose purpose is to evaluate the performance of your solution to the problem and to test its validity.
2. Develop an initial suite of test data that can exercise the model. This must take into account the goals of the solution (artifact) and the outer environment in which the solution must operate. This will involve modeling the outer environment.
3. Select a simulation package that is specifically designed for your problem domain. This will involve the least amount of programming. If such a package is not available, then choose a general programming language such as C++ or Java and model the problem, solution, and the outer environmental constructs in the language.
4. Run the simulation program for the developed test suite. Collect performance data and analyze it to evaluate your solution. If the performance does not meet your expectations, then the solution may need to be revisited and revised. Otherwise, test the solution over a wide range of conditions. Test the solution on extreme conditions to verify the range of outer environmental conditions over which your solution is valid.
5. Argue that your testing is representative of the real-life situations for which the solution is supposed to work. Argue that the data analysis supports the validity of the hypotheses regarding your solution.

Consequences

This pattern, if applicable, provides a reasonable and cost-effective way of evaluating and validating a solution. The alternative of testing the solution in real-life settings may be both costly and time consuming or may not even be feasible.

Usage Example(s)

Vaishnavi et al. (1997) exercise their model using multiple versions of the grocery bagging example; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Using Metrics

Intent

Use established metrics to aid the validation of your solution to the research problem.

Motivation

The use of metrics is an accepted method for establishing the performance or other claims being made about a solution to a research problem (March and Smith 1995). It provides a way to quantitatively make comparison of the solution to existing solutions.

Context/Applicability

Established metrics exist in the literature that you can use to evaluate the performance of your solution and to prove or argue the correctness of the hypotheses that you have made regarding the performance of your solution. If metrics are not available to measure the performance of your solution, then you will try using metrics available for a similar problem or even proposing new metrics.

Description

1. Determine whether or not there exist established metrics that are appropriate to measure the performance of your solution and to compare it with the performance of previous solutions, if they exist. If such metrics do not exist, determine whether or not metrics exist for measuring the performance of problems similar to your problem or alternatively propose new metrics. In such cases, you need to argue that the use of the chosen metrics is a reasonable way of evaluating and validating your solution.
2. Analyze or measure your solution using the chosen metrics. This may involve mathematical proofs, experimental measurements, or simulation (see the patterns *experimentation*, *simulation*, and *mathematical proofs*)
3. Show that your solution has the hypothesized performance according to the chosen metrics.

Consequences

This pattern allows you to possibly validate your solution in a way that is already accepted by the research community. This makes easier the acceptance of your solution by the research community.

Usage Example(s)

1. Using O-notation for expressing the running time or storage usage of an algorithm (originally proposed by Knuth) has become an accepted way for theoretically estimating the performance of an algorithm or for comparing the performance of two algorithms. The metric provides an indication of performance but only for sufficiently large sizes of input data and only indicates

how the running time (or storage use) will increase as the size of the input data increases. The metric has, however, been well established and accepted by the algorithm analysis and design science research community.

2. Datta (1998) proposes and uses metrics for the evaluation of the proposed strategies; also see Chapter 13, p. 340.
3. Vaishnavi et al. (1980) use the well-accepted metric of the O-notation to specify the performance of their algorithm; also see Chapter 13, p. 360.

References

Abbasi, A., and Chen, H. (2008). "CyberGate: A Design Framework and System for Text Analysis of Computer-Mediated Communication." *MIS Quarterly* 32(4): 811–838.

Baldwin, D. and Yadav, S. (1995). "The Process of Research Investigations in Artificial Intelligence—A Unified View." *IEEE Transactions on Systems, Man, and Cybernetics* 25(5): 852–861.

Berners-Lee, T. and Cailliau, R. (1990). "WorldWideWeb: Proposal for a Hypertext Project." URL: http://www.w3.org/Proposal.html (last accessed on January 29, 2015).

Chen, P. (1976). "The Entity-Relationship Model: Toward a Unified View of Data." *ACM Transactions on Database Systems* 1(1): 9–37.

Choobineh, J. and Lo, A. (2005). "CABSYDD: Case-Based System for Database Design." *JMIS* 21(3): 281–314.

Codd, E. F. (1970). "A Relational Model of Data for Large Shared Data Banks." *Communications of the ACM* *13*(6): 377–387. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 64–69, January, 1983.

Datta, A. (1998). "Automating the Discovery of AS-IS Business Process Models: Probabilistic and Algorithmic Approaches." *Information Systems Research* 9(3): 275–301.

Denning, P. (1968). "The Working Set Model for Program Behavior." *Communications of the ACM* 11(5): 323–333. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 43–48, January, 1983.

Fraser, M. and Vaishnavi, V. (1997). "A Formal Specification Maturity Model." *Communications of the ACM* 40(12): 95–103.

Hoare, C. (1978). "Communicating Sequential Processes." *Communications of the ACM* 21(8): 666–677. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 100–106, January, 1983.

Kleindorfer, G., O'Neill, L., and Ganeshan, R. (1998). "Validation in Simulation: Various Positions in the Philosophy of Science." *Management Science* 44(8): 1087–1099.

March, S and Smith, G. (1995). "Design and Natural Science Research on Information Technology." *Decision Support Systems* 15(4): 251–266.

McLaren, T., Head, M., Yuan, Y., and Chan, Y. (2011). "A Multilevel Model for Measuring Fit between a Firm's Competitive Strategies and Information Systems Capabilities." *MIS Quarterly* 35(4): 909–929.

Navidi, W. (2006). *Statistics for Engineers and Scientists.* Boston, Mass: McGraw Hill. (This book provides an excellent treatment of simulation.)

Purao, S., Storey, V., and Han, T. (2003). "Improving Analysis Pattern Reuse in Conceptual Design: Augmenting Automated Processes with Supervised Learning." *ISR* 14(3): 269–290.

Sein, M., Henfridsson, O., Purao, S., Rossi. M., and Lindgren, R. (2011). "Action Design Research." *MIS Quarterly* 35(1): 35–56.

Simon, H. (1996). *The Sciences of the Artificial*. Third Edition. Cambridge, MA: MIT Press.

Tichy, W. (1998). "Should Computer Scientists Experiment More?" *IEEE Computer* 31(5): 32–40.

Vaishnavi, V., Buchanan, G., and Kuechler, W. (1997). "A Data/Knowledge Paradigm for the Modeling and Design of Operations Support Systems." *IEEE Transactions on Knowledge and Data Engineering* 9(2): 275–291.

Vaishnavi, V., Kriegel, H., and Wood, D. (1980). "Optimum Multiway Search Trees." *Acta Informatica* 14(2): 119–133.

Zelkowitz, M. and Wallace, D. (1998). "Experimental Models for Validating Technology." *IEEE Computer* 31(5): 23–31.

Chapter 12

Publishing Patterns

The patterns in this chapter are applicable to the *conclusion* phase of the research (see Figure 6.2 in Chapter 6). You have either completed a research project or have obtained significant results while conducting your research. You would like to write a paper to report your results.

The following seven patterns presented in alphabetic order provide guidelines for publication:

The *conference and journal submissions* pattern provides general guidelines for submitting papers to conferences and journals, and for deciding whether to write a paper for a conference or a journal.

Writing conference papers and *writing journal papers* provide guidelines on how to write papers for conferences and journals, respectively.

The other four patterns, *style exemplars*, *aligning with a paradigm*, *novelty and significance*, and *use of examples*, provide guidelines that can be used to increase the chances of acceptance for your paper; they are particularly useful for journal submissions but can also be useful for conference submissions.

Writing research papers for publication is an art. These patterns are an attempt at a brief exposition of this art. The use of the patterns can increase the chances of success for your writing efforts.

As in the previous chapters, meta-level patterns are indicated by the superscript ^{M} preceding the pattern name. The patterns in this chapter, *aligning with a paradigm* and *style exemplars*, while strongly identified with publication may in fact also be used at the beginning of a project. Locating an exemplar paper describing research on a closely related topic at the beginning of a project can suggest development methods and validation techniques. Determining the paradigm with which the research problem is most closely associated can also suggest research methods, validation techniques, and allied literature at an early point (just after preliminary problem identification) in the research program.

^{M}Aligning with a Paradigm

Intent

Write your paper in such a way that it aligns with a research paradigm shared by the publication outlet.

Motivation

Acceptance of research by the research community is a social process. The way people in the research community understand and react to a new research paper is heavily affected by the prevailing research paradigms (Kuhn 1962/96). These paradigms contribute to shared symbols, beliefs, research puzzles, analogies, and metaphors, which in turn determine the importance of research questions and whether or not the explanations provided in a paper are acceptable.

Writing a research paper in such a way that it aligns with the prevailing research paradigm(s) increases the chances of acceptance of the paper by the research community. The alignment can be in terms of the research issues raised, the approach for addressing the research issues, and the way the research is presented. Not all research needs to or should follow the existing paradigms; never departing from the prevailing paradigms would be detrimental to the advancement of a field. However, it takes a greater effort to get acceptance of a paper by the research community if the paper significantly departs from the prevailing research paradigms.

Context/Applicability

You have identified a publication outlet such as a journal or a conference for which you would like to write a paper to report your research. You would like to do it in such a way that the paper is well received by the research community.

Description

The following steps can help in understanding the prevailing research paradigms and in writing your paper in a way that it aligns with such paradigms:

- Take time to fully comprehend the prevailing research paradigms in the area of your research. The *literature search* pattern, *understanding research community*, can be useful in this task.
- Relate your research problem to the research issues that the research community already understands.
- Use the community's shared symbols and beliefs in writing your paper.
- Find exemplar papers that you can use to model your paper after. See the pattern *style exemplars*.

Consequences

The use of this pattern will maximize the chances of acceptance of your paper. You need not always use this pattern. You may choose to write a paper in a way that departs from the prevailing paradigms but you should make a conscious decision to that effect. In such a case, the paper needs to educate the reader about the presented concepts and at least relate them to what the reader is expected to already know.

Usage Example(s)

1. The way the research is discussed and presented in Abbasi and Chen (2008) reflects the way such research is dealt with in *MIS Quarterly*; also see Chapter 13, p. 320.
2. Hoare (1978) presents communicating sequential processes (CSP) in a manner that aligns it with the shared symbols and beliefs of the research community that deals with formal treatment of parallel programming; also see Chapter 13, p. 348.
3. Vaishnavi et al. (1997) review the literature to align the reported work with respect to existing paradigms and also use this meta-level pattern in the awareness of problem phase of the research; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Related Pattern(s)

The *literature search* pattern, [M]*understanding research community* (p. 201), can be useful to this pattern.

Conference and Journal Submissions

Intent

Make a judicious choice of targeting your research work to a conference or a journal.

Motivation

Conferences and journals serve distinctly different purposes. Understanding this distinction can help in making our writing efforts become productive and effective.

Conferences provide an avenue for sharing our work with peers to receive useful feedback and criticism. The conference proceedings also provide a quicker way to get our research results published. Journals on the other hand are archival in nature and generally serve the purpose of detailed and complete exposition of our research work and results.

Some conferences may not publish their proceedings or their proceedings may not be refereed. Such conferences mainly serve the purpose of sharing the work and interaction with peers.

Context/Applicability

You have completed your research to solve a certain research problem. Alternatively, your research is ongoing but you have obtained certain results that you would like to report in the form of a paper for a conference or a journal.

Description

Consider writing a paper for a conference when:

- You have obtained some interesting results that you would like to share with the research community without delay.
- You have not yet fully worked out and tested your solution to the research problem.
- You would like to get feedback from the conference to guide your further research.

Consider submitting to a journal when:

- You have fully worked out the solution and validated it.
- Your contribution to knowledge is such that it is worth archiving in a journal.

Targeting your research work to a conference or a journal need not be an either/or choice. Actually it is quite useful and common to target the initial results of your

work to a conference and to submit the completed and in-depth treatment of the work to a journal.

There is a wide variety in the standards for conferences and journals. You should carefully choose the conference that best fits the type and quality of your research. For conferences, you should examine prior conference proceedings and call for papers to find if a certain conference is a suitable outlet for your work. For journals, examination of past papers and editorial policies can guide your selection.

Consequences

Conferences and journals have different purposes. By making a judicious choice of what work at what stage should be submitted to what conference or journal, you can let the conference and journal submissions play a synergistic role in advancing your research.

Novelty and Significance

Intent

Make your paper show novelty and significance.

Motivation

For a research paper to be acceptable for a good conference or a journal, it must report knowledge contribution that is new, valid (true), and interesting (Gregor and Hevner 2013; Wilson 2002). To be perceived as interesting, it must have significance with respect to the current state of the art in the corresponding research area.

Context/Applicability

You have conducted research and the knowledge contribution of the work has novelty and significance. You would like to write the paper in such a way that the reviewers of the paper clearly see the novelty and significance.

Description

A paper submitted to a journal or conference needs to be written for the reviewers of the paper as well as for the general readers of the paper after it is published. It is the reviewers (referees) who decide whether to accept or reject the paper for publication. The reviewers are likely to be less familiar with your specific research problem

than you are. It is your responsibility to show clearly the novelty and significance of your research so that the paper is not rejected on those grounds. Here are some guidelines in this regard:

- Place your research in the context of the existing literature showing novelty and significance. Show clearly the knowledge gaps in the existing literature. Discuss the importance of these gaps. Discuss how your reported research fills these gaps. The introduction is usually the section to establish the novelty and significance of your research. The significance of the reported research should also be highlighted in the concluding section of the paper.
- Discuss the potential limitations of your research and topics of future research in the concluding section. This helps to prevent any false impression about the contribution of the reported research.

Consequences

The use of this pattern can help the reviewers of the paper to have a better understanding of the novelty and contribution of the paper. This in turn can improve the chances of acceptance of the paper for publication.

Usage Example(s)

1. The way Abbasi and Chen (2008) formulate the research problem—using *research conversations* and *solution/scope mismatch* patterns—makes obvious the novelty and significance of the reported research; also see Chapter 13, p. 320.
2. Codd (1970) shows the novelty and significance of his work by discussing his work in the context of problems in existing data models and their significance; also see Chapter 13, p. 336.
3. Denning (1968) shows novelty and significance by contrasting the proposed model with existing models and by showing how this work initiates a new direction of research in system resource allocation; also see Chapter 13, p. 345.
4. Hoare (1978) demonstrates its novelty by comparing the reported research with existing research in the area. It shows its significance by showing that a small number of concepts—input, output, and concurrency—can be regarded as primitive concepts of parallel programming; also see Chapter 13, p. 348.
5. McLaren et al. (2011) explicitly develop both the novelty and significance of their artifact in their paper; also see Chapter 13, p. 352.
6. Novelty of the approach used and the significance of the problem addressed are stressed throughout (Purao et al. 2003); also see Chapter 13, p. 356.

7. Vaishnavi et al. (1997) show the novelty and significance of the reported work by discussing its strengths and limitations in the context of the existing literature; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.
8. Since the work reported in Vaishnavi et al. (1980) is solving a problem for multi-way search trees that is similar to the one previously solved by Knuth, the authors carefully distinguish the two problems and also show that a simple generalization of Knuth's solution is not an efficient solution to the problem; also see Chapter 13, p. 360.

[M]Style Exemplars

Intent

Use a style exemplar to increase the chances of success for the acceptance of your paper.

Motivation

Style exemplars from published research serve as a vehicle for knowing how research similar to our work (or proposed work) is conducted, evaluated, and written. Using appropriate exemplars to guide the work and how it is reported can ensure that the work is well received by the research community.

Context/Applicability

You are trying to write a paper to report research that is of good quality. However, the quality of the reported research by itself does not guarantee publication success. You would like to write the paper in such a way that it is well received by the referees while it is being reviewed and by the audience after it gets published.

Description

1. Find an exemplar paper in the journal that is close to the contents of your intended paper. Ideally, the authors of the paper should be well established and recognized by the research community.
2. Use the paper as a model to guide the writing of your paper. Use the notation and style of the paper to the extent possible and adapt it minimally if needed.

Consequences

The use of this pattern will help you to write your paper in a manner that is likely to be well received by the referees of the paper and thus be accepted by them. It also helps to better understanding of the paper by the readers, because it lets them understand the paper in the context of notation and style that is likely to be already familiar to them.

Usage Example(s)

1. Vaishnavi et al. (1997) use this meta-level pattern in the evaluation phase of their research to model the validation portion of their research after existing papers that also used demonstration for validation; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.
2. Vaishnavi et al. (1980) model their paper after the one written by Knuth for the same journal and for solving a similar problem for binary search trees; also see Chapter 13, p. 360.

Use of Examples

Intent

Use concrete examples to provide a better understanding of your research.

Motivation

The readability of a paper can make all the difference in whether the paper is accepted and how well it is liked after it is published. Examples play a large role in improving readability of a paper.

Context/Applicability

Your research solves a research problem that has applications to a problem or a class of problems. You would like to write the paper in a way that will make the paper better readable, which in turn can help in improving the chances of its acceptance.

Description

By staying at a level that is too general or abstract, the readers may not be able to fully understand the reported research or its benefits. Use a running example or a number of related examples to provide concrete illustrations of your research and its benefits:

- Use each example to illustrate a distinct aspect of your research and its benefits.
- Describe the purpose of each example and how it is achieving its purpose.
- Use graphics, where applicable, to improve the message of an example.

Consequences

The use of the pattern can improve the readability of your paper. The readability of the paper can also be beneficial in the paper review process.

Usage Example(s)

1. Abbasi and Chen (2008) use examples from actual computer-mediated communication internal to Enron to illustrate the capabilities of CyberGate; also see Chapter 13, p. 320.
2. Chen (1976) uses a running example to illustrate the proposed model and diagrammatic technique; also see Chapter 13, p. 328.
3. Codd (1970) contains a parts-projects-suppliers example to illustrate the relational model and its benefits; also see Chapter 13, p. 336.
4. Hoare (1978) uses a number of well-known examples to demonstrate the use of the concepts in CSP; also see Chapter 13, p. 348.
5. A running example is used to illustrate the proposed model and diagrammatic technique (Purao et al. 2003); also see Chapter 13, p. 356.
6. Vaishnavi et al. (1997) contain a number of examples to illustrate concepts related to the smart object model and to show their novelty and significance; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

Writing Conference Papers

Intent

Write a conference paper.

Motivation

The way a conference paper is written and even titled is very different from how it is done for a journal paper. A paper accepted for a conference will fit with the theme of the conference or the conference track for which it is submitted. It will also be written in such a manner that it will appeal to the conference attendees or potential attendees.

Context/Applicability

You have decided to write a paper for a conference and have chosen the conference that best suits your intended paper. You would like to know how best to write the paper so that it has the best chance for acceptance (Johnson et al. 1993).

Description

- Carefully study the call for papers to understand the focus of the conference. Identify the topic or track that your paper can fit in. Choose a writing style that best suits the focus of the conference, the chosen topic or track, and the expected audience for the conference.
- Focus on a single idea that you intend to write about in the paper. Fully develop the idea and support it with evidence. The idea should be of potential interest to the audience and should generate discussion. The topic of the paper should be such that it will add to the value of the conference for the conference attendees.
- The conference format will not allow you to do any major revision of the paper in response to the reviewers' comments. Therefore, the paper needs to be crisp and polished, and needs to meet the specified length restriction for the paper.
- The paper will be judged on such criteria as originality, technical quality, presentation quality, and contribution/potential impact. Make sure that the paper can score well on such criteria.
- How a paper should be written also depends on the type of the paper. Here are some examples:
  - A *theory* paper should have a clear focus, should clearly state the theory, which also must be of interest to the expected conference attendees, should relate the work with existing literature, and should provide evidence in support of the theory and show that the theory has been tested.
  - A *methods* paper should clearly provide the goals of the paper, should be focused, should tie the work to related literature, and should defend the proposed method.
  - An *experience* paper should focus on a single topic, should present relevant facts of an experiential nature, and should advance the state of knowledge.

Consequences

A successful conference paper can help you in getting timely feedback on your research ideas, can help you in socializing with members of your research community, and can provide you with new insights and ideas for further development of your research ideas. In certain cases, a conference paper can also evolve into a journal paper.

Writing Journal Papers

Intent

Write a journal paper.

Motivation

The acceptance of a paper in a journal is determined to a large degree by the fit of the paper with the theme and scope of the journal as well as by the degree to which it meets the quality expectations of the journal for its published papers.

Context/Applicability

You have decided to write a paper for a journal and have chosen the journal that best suits your intended paper. You would like to know how best to write the paper so that it has the best chance for acceptance.

Description

- The journals vary widely in quality, acceptance rates, type of research published, and writing style. Study carefully the editorial policies of the journal and its past papers to write in a way that will be acceptable to the journal.
- Choose an exemplar paper from the journal that closely matches the intended content of your paper. Use this paper as a model to guide the writing of your paper. See the pattern *style exemplars*.
- If possible, align your paper with a research paradigm that the journal papers share. See the pattern *aligning with a paradigm.*
- The paper is expected to have novelty and significance. Write the paper in such a way that the novelty and significance of the paper is clearly shown. See the pattern *novelty and significance*.
- The acceptance of the paper is based on the report of the referees for the paper. You have to make a case to the referees that the paper merits publication in the journal. Write the paper in such a way that it makes this case to the referees.
- Use examples or preferably a running example that makes the contribution of the paper more understandable. See the pattern *use of examples*.

Consequences

The paper may be accepted by the journal without any revision or after a minor revision. It is, however, more likely that the paper will be rejected or rejected with suggestions for a major revision. This is a normal iterative process for journal publications in most cases. In the case of a definite rejection, consider rewriting the paper for a different journal.

Usage Example(s)

1. The paper by Abbasi and Chen (2008) is a good example of writing a paper to ensure its success in acceptance by the journal *MIS Quarterly*; also see p. 320.
2. Vaishnavi et al. (1997) closely follow the principles of writing a paper that has archival value by relating their work to the existing literature and by showing its novelty and significance; also see "An Example of ICT Design Science Research" in Chapter 2, "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6, and Chapter 13, p. 312.

References

Abbasi, A. and Chen, H. (2008). "CyberGate: A Design Framework and System for Text Analysis of Computer-Mediated Communication." *MIS Quarterly* 32(4): 811–838.

Chen, P. (1976). "The Entity-Relationship Model: Toward a Unified View of Data." *ACM Transactions on Database Systems* 1(1): 9–37.

Codd, E.F. (1970). "A Relational Model of Data for Large Shared Data Banks." *Communications of the ACM* 13(6): 377–387. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 64–69, January, 1983.

Denning, P. (1968). "The Working Set Model for Program Behavior." *Communications of the ACM* 11(5): 323–333. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 43–48, January, 1983.

Gregor, S. and Hevner, A. (2013). "Positioning and Presenting Design Science Research for Maximum Impact." *MIS Quarterly* 37(2): 337–355.

Hoare, C. (1978). "Communicating Sequential Processes." *Communications of the ACM* 21(8): 666–677. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 100–106, January, 1983.

Johnson, R.E., Beck, K., Booch, G., Cook, W., Gabriel, R., and Wirfs-Brock, R. (1993). "How to Get a Paper Accepted at OOPSLA." In *OOPSLA Proceedings*.

Knuth, D.E. (1971). "Optimum Binary Search Trees." *Acta Informatica* 1(1): 14–25.

Kuhn, T. (1962/96). *The Structure of Scientific Revolutions*. Third edition, Chicago, IL: The University of Chicago Press.

McLaren, T., Head, M., Yuan, Y., and Chan, Y. (2011). "A Multilevel Model for Measuring Fit between a Firm's Competitive Strategies and Information Systems Capabilities." *MIS Quarterly* 35(4): 909–929.

Purao, S., Storey, V., and Han, T. (2003). "Improving Analysis Pattern Reuse in Conceptual Design: Augmenting Automated Processes with Supervised Learning." *ISR* 14(3): 269–290.

Vaishnavi, V., Buchanan, G., and Kuechler, W. (1997). "A Data/Knowledge Paradigm for the Modeling and Design of Operations Support Systems." *IEEE Transactions on Knowledge and Data Engineering* 9(2): 275–291.

Vaishnavi, V., Kriegel, H., and Wood, D. (1980). "Optimum Multiway Search Trees." *Acta Informatica* 14(2): 119–133.

Wilson, J.R. (2002). "Responsible Authorship and Peer Review." *Science and Engineering Ethics* 8(2): 155–174.

III KNOWLEDGE CONTRIBUTION & RESEARCH PATTERNS USAGE ANALYSIS

"The road to learning by precept is long, but by example short and effective."

Seneca the Younger

This final part of the book consisting of one chapter, Chapter 13, is devoted to an analysis of papers reporting design research to illustrate, through example, two important areas: (1) Knowledge contribution in this type of research. (2) Use of design science research patterns. Each of the papers is first analyzed to see what type of knowledge contribution is made and the state of the design science knowledge, expressed as design theory, and advanced by the reported research. This is followed by analysis of the paper to elaborate on the possible research patterns used in different phases of the research.

Chapter 13

Knowledge Contribution and Patterns Usage Analysis of Design Science Research Exemplars

Introduction

This chapter is useful for several modes of learning about design science research (DSR)—pertaining to both knowledge contribution and the process used in conducting such research.

Analysis Examples

Published DSR works, many of which have been highly influential in their areas, are analyzed in terms of their new knowledge contribution and the patterns used in the performance of the research. Some of them are from the information systems (IS) area while others are from the related field of computer science. In either case, they are exemplars of *learning and investigation through artifact creation*, the most fundamental characteristic of DSR.

The following published work—the smart objects running example (case study) of this book—is one of the sources analyzed for mining patterns and also serves as the first illustrative exemplar for knowledge contribution in DSR and the use of patterns (also see "Pattern Usage in the Development of the Smart Object Paradigm" in Chapter 6): "Smart Objects: A Data/Knowledge Paradigm for the Modeling and Design of Operations Support Systems."

The following are the other 11 research works that will be used as exemplars for design science knowledge contribution and research patterns usage. Their analysis will be carried out in alphabetic order of their authors.

- Abassi et al.: "CyberGate: A Design Framework and System for Text Analysis of Computer-Mediated Communication" (p. 320)
- Barners-Lee et al.: "World Wide Web: Proposal for Hypertext Project" (p. 325)
- Chen: "The Entity-Relationship Model: Toward a Unified View of Data" (p. 328)
- Choobineh et al.: "CABSYDD: Case-Based System for Database Design" (p. 332)
- Codd: "A Relational Model of Data for Large Shared Data Banks" (p. 336)
- Datta: "Automating the Discovery of AS-IS Business Process Models: Probabilistic and Algorithmic Approaches" (p. 340)
- Denning: "The Working Set Model for Program Behavior" (p. 345)
- Hoare: "Communicating Sequential Processes" (p. 348)
- McLaren et al.: "A Multilevel Model for Measuring Fit between a Firm's Competitive Strategies and Information Systems Capabilities" (p. 352)
- Purao et al.: "Improving Analysis Pattern Reuse in Conceptual Design: Augmenting Automated Processes with Supervised Learning" (p. 356)
- Vaishnavi et al.: "Optimum Multiway Search Trees" (p. 360)

In general, the example papers are selected to be representative of research in ICT, particularly in IS and computer science. The following papers have appeared among milestones of research—Selected Papers (1958–1982) in *Communications of the ACM*:

- "A Relational Model of Data for Large Shared Data Banks"
- "The Working Set Model for Program Behavior"
- "Communicating Sequential Processes"

The paper "World Wide Web: Proposal for Hypertext Project" is actually a short proposal but is included because of the Web revolution that the proposal has dawned. There are two papers that may seem to have somewhat similar themes: "CABSYDD: Case-Based System for Database Design" and "Improving Analysis Pattern Reuse in Conceptual Design: Augmenting Automated Processes with Supervised Learning." The similarity though is only at a very general level. The former addresses automation

of database design using cases and case-based reasoning. The latter focuses on semi-automation of conceptual design of systems.

Just as the first paper, "A Data/Knowledge Paradigm for the Modeling and Design of Operations Support Systems," analyzed in this chapter, analysis of the work reported in the last paper, "Optimum Multiway Search Trees," benefits from one or both authors' personal involvement in the research. This helps providing insights that would not be possible through just analysis of a paper reporting research conducted by a third party.

Knowledge Contribution Analysis

The new knowledge contribution of the work reported in a paper is examined for its type as well as content. The type of knowledge contribution is discussed with respect to the knowledge contribution framework discussed in Figure 2.5 (Gregor and Hevner 2013). In this framework, knowledge contribution is classified into four categories: *invention* (inventing new knowledge/solutions to new problems), *improvement* (developing new knowledge/solutions for known problems), *adaptation* (adapting known knowledge/solutions in a nontrivial or innovative manner to solve new problems), and *routine design* (applying known knowledge/solutions to known problems). Out of these, the last category, routine design, by itself does not generally constitute research.

The design science knowledge content of the contribution is analyzed to determine whether it can be represented as a design theory and the status of such a theory (if any). This analysis is done with respect to the profile shown in Table 2.2 for a design theory. The profile has six core components—*purpose and scope*, *constructs*, *knowledge of form and function*, *abstraction and generalization*, *evaluation and validation propositions*, *justificatory knowledge*—and two optional components—*principles of implementation* and *expository instantiation* (Gregor and Jones 2007).

Pattern Usage Analysis

As regards DSR patterns, the chapter can be scanned for examples of patterns the reader may wish to investigate further. The patterns for a particular analysis are grouped into the classifications of patterns used in prior chapters. After identifying patterns of interest, the papers using those patterns can be read in detail to see how, in actual practice in a research context, the patterns were executed. Alternatively, one or more of the analyzed papers can be read in full—possibly chosen for the reader's interest in or knowledge of a certain area—and then the pattern analyses can be followed on a second reading of the research paper.

It is not significant to the learning process if the reader has no familiarity with the actual detailed processes that occurred in the research effort described. As we have mentioned at other points in the book, published descriptions of research usually focus on the results of the research, not the process. What is important is identification of the patterns and processes that *are applicable or might have been at work*

in the research effort. These patterns can be identified by a hermeneutic* reading of the paper while (1) continually referring to the general methodology of DSR (Figure 2.3) as the overall activity flow that is most likely to have occurred and (2) referring to the patterns applicable to each of the general DSR methodology phases.

For example, the patterns used to identify and refine a problem area are frequently visible (between the lines) in the introduction section of the paper and sometimes the literature review section. With research papers, authors are frequently at some pain to justify the contribution of their literature to a research area, and the patterns used both to align with a research community and to define and refine their problem area can sometimes be identified in the conclusion sections of the paper as well as the sections just mentioned. Similarly, the patterns used to arrive at a successful validation effort can frequently be detected in the discussion sessions of many research papers.

Smart Objects: A Data/Knowledge Paradigm for the Modeling and Design of Operations Support Systems

Source

Vaishnavi, V., Buchanan, G., and Kuechler, W. (1997). "A Data/Knowledge Paradigm for the Modeling and Design of Operations Support Systems." *IEEE Transactions on Knowledge and Data Engineering* 9(2): 275–291.

Knowledge Contribution

Contribution Type

The type of knowledge contribution (see Figure 2.5) of this work is *invention*, since the research is inventing a *new solution* and associated knowledge for a *new problem*. The problem addressed is that of modeling and design of a new class of systems, operations support systems, identified by the authors. The solution does utilize in a synergistic manner existing knowledge in three research areas of data modeling,

* Hermeneutics means the interpretation and understanding of social events by analyzing their meanings to the human participants and their culture. It differs from other interpretative techniques in that it emphasizes the importance of the content as well as the form of any given social behavior. The central principle of hermeneutics is that it is only possible to grasp the meaning of an action or statement by relating it to the whole discourse or world view from which it originates: for instance, putting a piece of paper in a box might be considered a meaningless action unless put in the context of democratic elections, and the action of putting a ballot paper in a box. One can frequently find reference to the "hermeneutic circle," that is, relating the whole to the part and the part to the whole (excerpted from the Wikipedia entry for hermeneutics).

software engineering, and knowledge representation but the solution offered to the core problem of control modeling in such systems is new.

Status of Design Theory

The following is an assessment of the contributed design science knowledge as a design theory with respect to the profile shown in Table 2.2:

Core Components

- **Purpose and Scope:** Modeling and design of operations support systems that can be used for managing large, complex operations environments such as manufacturing plants, military operations, and large power generation facilities. Control modeling is the specific research problem addressed.
- **Constructs:** Such as operations support systems, smart objects, control modeling concept.
- **Knowledge of Form and Function:** Smart object paradigm and its instantiation—smart object language (SOL).
- **Abstraction and Generalization:** The smart object paradigm is at a high level of abstraction and generality. It can be instantiated into multiple languages, and the smart objects in any of the instantiations can be implemented in different ways while still following the proposed paradigm. Also, the instantiated language can be changed while still complying with the paradigm.
- **Evaluation and Validation Propositions:** SOL—an instantiation of the contributed paradigm—can be used to develop executable models for operations support systems, particularly their complex dynamic control behavior. The validity of the proposition is demonstrated by using it to model part of an operations support system for a nuclear power plant and also for modeling a grocery bagging example. The grocery bagging example is used to exercise different aspects of SOL.
- **Justificatory Knowledge:** The proposed paradigm fuses concepts drawn from well-established design theories from the fields of data modeling, software engineering, and knowledge representation.

Optional Components

- **Principles of Implementation:** Principles of instantiating the proposed paradigm and using the instantiation for developing an operations support system are implicitly provided by defining the SOL and demonstrating its use.
- **Expository Instantiation:** An instantiation of the smart object paradigm—SOL—is provided but the SOL is not implemented.

Conclusion on Status of Design Theory

The reported research contributes a nascent design theory for the modeling and design of operations support systems. With further work and participation by the research community, a robust and fully developed design theory for such systems can be developed.

*Research Patterns Usage**

Problem Selection and Development Patterns (Awareness of Problem Phase)

Aligning with a Paradigm

The smart objects project began with the recognition of the problem of control of a complex environment as amenable to a DSR solution. The authors understood the DSR paradigm from years of research or practical experience in the DSR field and proceeded more deeply in their initial investigations only after having identified the nuclear power control modeling problem as a DSR opportunity, implicitly utilizing this meta-level (publishing) pattern.

Solution Scope Mismatch

The research problem was identified while attempting to develop a support system for a nuclear reactor using the rule-base language, Prolog. It was soon realized that it would be nearly impossible to develop such a system in Prolog and to maintain it to support the thousands of procedures typically needed in a commercial nuclear power plant. This led in turn to the realization that the current tools were not fully capable of constructing and continuously maintaining a support system for the operation of a complex environment. This meta-level pattern was also used in the suggestion/development phases of the research while attempting to find an appropriate solution to the research problem.

Being Visionary; Brainstorming

The authors analyzed the best-available solutions (from software design, databases, and knowledge models) with respect to the problem of modeling complex systems and found them to be not fully suitable. They then envisioned an improvement to the situation by coming up with a set of attributes that they felt were essential to any conceptual model of operations support systems. The attributes of this

* For the current example, as also for the last example—"Optimum Multiway Search Trees," the personal experience of one or both authors in conducting the research is utilized and so past tense is sometimes used in discussing the use of patterns.

process that distinguish it from routine design per se is that the authors knew, as they were developing the attribute set, that no existing technology could meet the requirements. This type of envisioning is sometimes termed "blue sky" design and effectively merges with the actions for the pattern *brainstorming*. These meta-level patterns were also used in the suggestion/development phases of the research in attempting to find a novel solution.

Redefining Research Problem; Problem Formulation; Research Topic Identification

The patterns *problem formulation* and *research topic identification* were used in the initial definition of the problem and the research topic. But it was only after the research moved into the *suggestion* phase that the research problem was redefined and a more specific and tightly scoped problem and topic were identified. Only then was the problem of *effectively modeling operations support systems for complex, hierarchical, procedure driven environments* with *control modeling* identified as the specific research topic.

Bridging Research Communities

The researchers identified three distinct but interrelated research communities—software engineering, database systems, and knowledge-based systems—that have developed distinct approaches to addressing the problem of modeling complex systems, none of which was adequate to the problem by itself. After familiarizing themselves with concepts from object-oriented design, semantic data modeling, active database system modeling, and rule-based knowledge modeling, they identified attributes in each of these areas that were essential for the design of complex systems and synthesized them in a complementary manner to be included in the SOM. The identification of and analysis of the three research communities required use of the following patterns related to *bridging research communities*:

- Research domain identification
- Understanding research community
- Research conversation
  (This meta-level pattern was also used in the suggestion/development phases of the research.)

Complex System Analysis

The authors analyzed the systems dealing with the management of complex operations environments, which they termed *operations support systems*. As no traditional class of ISs had the capability to address the depth of interactive, global support required for managing operations environments, they began by identifying the functionality required to improve the effectiveness of systems for managing operations environments. This

meta-level pattern was also used in the suggestion phase of the research to analyze an initial solution based on the use of frames for knowledge representation.

Literature Search Patterns (Awareness of Problem Phase)

Industry/Practice Awareness

This meta-level pattern was used both in the *problem awareness* and *suggestion* phases of the research. The authors identified problems faced in practice and abstracted them into research problems when they attempted to model a complex operations environment with Prolog; see also "An Example of ICT DSR" in Chapter 2. Using this "hands-on" approach, they increased their awareness of developments and problems in industry and practice and also experienced them first hand.

Suggestion and Development Patterns (Suggestion/Development Phases)

Hypothetical/Deductive Approach

The authors used the hypothetical/deductive approach to building theory by using intuition, results of past experimentation, literature review of approaches in different research communities such as data/knowledge models, and so on, to develop the smart object paradigm and its instantiation—SOL; SOL was evaluated for its ability to develop executable models of operations support systems, particularly their complex dynamic control behavior.

Problem Space Tools and Techniques; Research Community Tools and Techniques

The authors analyzed existing tools and techniques, abstracted relevant concepts, and incorporated those concepts into the smart object paradigm framework.

Abstracting Concepts

The attribute set required by the model (see earlier discussion of *being visionary*) was derived by abstracting general control principles from multiple examples of operations control environments and multiple partial solutions to the problem (see also the use of *combining partial solutions*).

Elegant Design

The artifact (here, the instantiation of smart object model (SOM)) is designed to be general and could be defined in functional terms. The model underlying the

artifact is independent of its outer and inner environments and so can be used to manage any operations environment. The authors also mention that the paradigm has proven richer than anticipated as it could be used in applications beyond its original intent.

Hierarchical Design

In developing SOM, Vaishnavi et al. decomposed the problem into subparts. Firstly, the smart object paradigm framework was described. Secondly, the logical and architectural views were reviewed. Then the steps to transition from the smart object paradigm to a working OSS were defined. The authors then decomposed the model into its conceptual attributes and its functional attributes. The problem of defining the conceptual attributes and functional attributes was further decomposed into sub-problems. For example, conceptual attributes were decomposed into sub-attributes of knowledge associated with operations, adaptive inferencing, structural relations between operations, and so on. These attributes were further decomposed into lower level problems. The complexity of the problem was both defined and appropriately handled by this approach.

Combining Partial Solutions; Sketching Solution

The authors found that while semantic data models, rule-based inferencing models, and object-oriented design provided partial solutions for operations support systems, they did not address all the desired attributes, particularly control abstraction. Through use of the pattern *sketching solution*, they identified the need to combine the strengths of the partial solutions augmented with the concept of a monitor to form the complete solution.

Embedding Concepts and Techniques

The concept of smart objects and associated techniques build on top of a number of existing concepts including those from active data models, object-oriented design, and rule-based logic engines.

Interdisciplinary Solution Extrapolation

The smart object paradigm fuses together concepts from databases, software engineering, artificial intelligence, and operating systems. It uses the general object-oriented structure from software engineering to manage complexity, semantic data modeling concepts from data bases, and production systems from artificial intelligence (AI), along with the operating systems concept of using a stack to monitor the status of an object.

General Solution Principle

The authors identify a general problem—the support of complex, large operations environments. They develop a general solution—the OSS framework—that can be instantiated for specific situations. The general solution is so broad that it can be called a paradigm. At various stages in developing this solution, the following meta-level patterns were used:

- Different perspectives
- Means-ends analysis
- Cost-benefit analysis
- Integrating techniques

The concept of a SOM draws conceptual modeling techniques from semantic data modeling, production systems, and the object modeling areas. It integrates into the model functionality from data modeling and knowledge engineering areas. It additionally introduces the concept of a "monitor" that helps in integrating the various techniques and creating a model that meets the desired requirements.

Using Human Roles

When the authors were surveying nuclear power plants and other complex operations environments, one of the primary shortcomings of existing attempts at computer control was that they were partial and required large amounts of human assistance. The authors analyzed the role played by human judgment in these environments and determined that much of it could be assumed by a meta-level of rules performing the human supervisory tasks.

Evaluation and Validation Patterns (Evaluation Phase)

Technological Approach Exemplars

The pattern *technological approach exemplars* led to a review of the problem domain literature focused on discovering what validation techniques were used by the chosen research community. This information did not absolutely constrain the direction taken, but definitely influenced it; it is widely understood that straying beyond the techniques commonly employed by a research community increases the difficulty of publishing in that community.

Demonstration

The authors evaluate the modeling ability of the SOM by demonstrating its use for a part of an operations support system for the nuclear power plant, which motivated

the entire work. They also use the widely understood grocery bagging example from AI to show the power of the model.

Simulation

The demonstration of the SOM using the grocery bagging example is extensive enough to be considered a simulation. Every aspect of the model is exercised in some manner in the demonstration.

Logical Reasoning

The authors do not provide any mathematical proofs but provide logical arguments to substantiate that the presented model is better able to model complex environments. They also argue that the model is conceptually consistent and maintainable. They present their case in the background of the existing literature in the related areas. They make the case that the presented work draws from the existing knowledge base and in turn contributes new knowledge.

Cost-Benefit Analysis

This meta-level pattern was used to determine the best strategy to use for the evaluation of the research.

Publishing Patterns (Conclusion Phase)

Aligning with a Paradigm

The paper does an extensive literature review to motivate the work and to align and relate it to the existing paradigms. They also discuss and illustrate the presented concepts in the light of the existing literature showing their novelty and significance.

Research Conversation

Closely related to the pattern *aligning with a paradigm*, this meta-level pattern was used to more specifically position the paper by identifying a journal that contained an ongoing research conversation that this research could logically enter into. Research conversation in this sense refers to multiple papers in multiple issues of the same journal cumulatively approaching a comprehensive solution to a large problem by presenting solutions to various aspects of the problem.

Writing Journal Papers

This paper was written at the conclusion of an extensive, four-year research program. The results of the research were solid enough and had been previewed and accepted at several conferences that the chances for publication of a well-written journal paper were good. The conference papers, in turn, productively used the *writing conference papers* pattern.

Novelty and Significance

The paper demonstrates novelty and significance by showing that the existing models drawn from a number of areas do not provide a total solution to the problem of modeling complex operations support systems and showing how the presented work fills an important knowledge gap. By placing the work in the context of the existing literature and showing its similarities and differences with existing models along with discussing the limitations of the work, they bring out the novel and significant aspects of the work.

Use of Examples

The paper contains a number of examples related to a subsystem of a prototype for an operations support system for a nuclear power plant to illustrate the concepts as well as to demonstrate the modeling capability of the SOM.

Style Exemplars

While the authors did not use any single paper as a template for the presentation of their ideas, they did search the target journal for, and found, multiple papers that presented novel, well-developed theoretical solutions to complex problems. These papers also relied on demonstration for validation, just as their paper, and provided style guidance in the writing of the paper.

CyberGate: A Design Framework and System for Text Analysis of Computer-Mediated Communication

Source

Abassi, A. and Chen, H. (2008). "CyberGate: A Design Framework and System for Text Analysis of Computer-Mediated Communication." *MIS Quarterly* 32(4): 811–838.

Knowledge Contribution

Contribution Type

This research can be classified (see Figure 2.5) as *improvement*. This is because it addresses the *existing problem* of designing computer-mediated communication (CMC) systems and develops a *new solution* and associated knowledge—a framework—that removes the deficiency of current systems that do not effectively capture text-based content in CMC systems, and then develops an instantiation of the framework—CyberGate.

Status of Design Theory

The following is an assessment of the contributed design science knowledge as a design theory with respect to the profile shown in Table 2.2.

Core Components

- **Purpose and Scope:** The research proposes a design framework for CMC text analysis systems and also an instantiation of the framework called CyberGate. The framework includes all three language meta-functions described by systemic functional linguistic theory (SFLT)—ideational, textual, and interpersonal.
- **Constructs:** Concepts from SFLT—ideational, textual, and interpersonal meta-functions—and their operationalization in CyberGate.
- **Knowledge of Form and Function:** Concept of CMC text analysis systems, a design framework for CMC text analysis systems.
- **Abstraction and Generalization:** The research proposes a framework, which defines a minimally complete CMC text analysis system. The framework can be instantiated in different ways and even an instantiated system can be changed as long as it complies with the underlying design framework.
- **Evaluation and Validation Propositions:** The CyberGate system (and hence its underlying design framework) can significantly improve text analysis capabilities over those provided by existing systems. A number of experiments have been conducted, which support the proposition.
- **Justificatory Knowledge:** Kernel theory—SFLT.

Optional Components

- **Principles of Implementation:** Guidelines for the choice of features, feature selection, and visualization techniques that CMC text analysis systems should use.
- **Expository Instantiation:** A system called CyberGate is developed and used in the experiments.

Conclusion on Status of Design Theory

Based on the preceding evidence, the research contributes a reasonably developed nascent design theory. The CMC research community can build upon this work to create a well-developed and general design theory for CMC systems.

Research Patterns Usage

Problem Selection and Development Patterns (Awareness of Problem Phase)

Research Conversation

The researchers expertly position their research within the existing research area of computer-mediated communication (CMC) text analysis by demonstrating an intimate familiarity with the literature and problems associated with this area (see also *literature search* patterns).

Solution Scope Mismatch

The researchers develop the significance of their research via a thorough investigation of existing systems in the area followed by development of the limitations of these systems. The need to go beyond existing systems and the benefits of doing so are developed, explaining and justifying the significance of their research.

Problem Formulation

Once the significance of an expanded CMC text analysis system has been established, the authors scope the problem to a specific attack on a subset of the issues that have been raised.

Literature Search Patterns (Awareness of Problem Phase)

Understanding Research Community

This pattern extends the *research conversation* pattern by a specific analysis of the CMC text analysis research community's vocabulary, publication outlets, and the structuring of research papers in that area.

Industry Practice Awareness

The authors have broad understanding of commercial CMC text analysis systems capabilities. This information is used not in the literature search portion of the paper per se but rather in the incorporation of features into the meta-design of the artifact; see pp. 823–824 of the paper.

Framework Development

The use of this pattern extends the knowledge gained by using the *problem selection and development* pattern, *research conversation*, this time by formally structuring the prior literature to expose knowledge gaps; see Table 1, Previous CMC Systems, on p. 814 of the article.

Suggestion and Development Patterns (Suggestion/Development Phases)

Hypothetical/Deductive Approach

The paper is a very clear exposition of the use of hypothetical/deductive approach to theory building. The authors build their design theory completely *in advance* of artifact construction. From SFLT principles, they develop a design framework for computer-mediated communication (CMC) text analysis systems, instantiate the framework as CyberGate, and experimentally test CyberGate for its text analysis capabilities.

Problem Space Tools and Techniques

The final design of the artifact utilizes multiple techniques from the general (largely academic) problem space. Note that several artifact features are derived from commercial products but that the final solution is a unique synthesis of techniques that lie outside what the research community had done until the date of this work.

Integrating Techniques

The problem the authors develop is sufficiently broad that no single technique from the research area could address it completely.

Building Blocks; Combining Partial Solutions

The building blocks pattern decomposes a complex problem into components, each of which is amenable to a simpler solution. The *combining partial solutions* pattern identifies those problem components for which a technique from the research area already exists. The highly synthetic solution to the problem the authors of this paper propose is an excellent example of the combined use of these two patterns. Note that the process of problem identification used by the authors ideally presages the patterns used here for solution development.

Embedding Concepts and Techniques

In utilizing this pattern, the solution for a problem—in this case, the design for a computer-mediated communication (CMC) text analysis system—is constructed

by extending existing concepts with new knowledge or knowledge from outside current practice. Abbasi and Chen show the desirability of a CMC text analysis system that incorporates analysis of content. They then construct a meta-design for such a system by embedding content analysis concepts into prior feature analysis designs. The process extends the concepts rather than using them as self-contained units as would result from the pattern *building blocks*.

Evaluation and Validation Patterns (Evaluation Phase)

Experimentation

Validation of artifact performance was via experiments (carefully controlled comparisons). An industry standard program for feature characterization was used as a baseline. For those features that were novel to CyberGate and not found in any current text analysis program, the performance of the artifact was compared with that of human coders. This is a relatively strong validation.

Publishing Patterns (Conclusion Phase)

Aligning with a Paradigm

In a meaningful way, the publishing pattern, *aligning with a paradigm*, must have begun during the *awareness of problem* phase when the pattern *research conversation* was used. A research community conversation is almost always conducted through publication outlets that are familiar with the type of research being discussed and the standard manner(s) of its presentation. Also, while the general paradigm, DSR, has seen acceptance at an increasing number of journals, each journal has a somewhat different manner of presentation. The authors obviously familiarized themselves with the presentation of DSR in the publication journal, *MIS Quarterly*.

Novelty and Significance

This pattern brings attention to the need to make the novel elements and the significance of a research project explicit. Just as in the preceding pattern, *aligning with a paradigm*, this pattern is actually begun by the application of the *research conversation* pattern in the *awareness of problem* phase. A large part of becoming familiar with a research conversation is becoming aware of the research questions that are meaningful for the chosen research community. The likelihood of novelty of the project was increased greatly by the pattern *solution/scope mismatch* that was applied also in the *awareness of problem* phase.

Use of Examples

Publishing is a form of communication, and like any communication of sophisticated concepts, is usually enhanced with concrete examples. In the instance of

CyberGate, the authors gained access to a large amount of computer-mediated communication internal to Enron, prior and during that company's notorious financial and legal problems. The communications served as a very sophisticated test case to illustrate CyberGate's capabilities in a widely understood context.

Writing Journal Papers

The paper is written in a manner that it will be acceptable to *MIS Quarterly*. The paper seems to take great pains to make sure that it follows the format and style of DSR papers that are currently acceptable to the journal.

World Wide Web: Proposal for Hypertext Project

Source

Berners-Lee, T. and Cailliau, R. (1990). "WorldWideWeb: Proposal for a Hypertext Project." URL: http://www.w3.org/Proposal.html (last accessed on January 29, 2015).

Note: The preceding source is not a research publication in a refereed journal or the proceedings of a conference; it is a proposal that was submitted by Berners-Lee and Cailliau to CERN (The European Center for Nuclear Research). CERN is the world's largest particle physics research center where scientists conduct experiments using particle accelerators and detectors to study the smallest constituents of matter in order to answer questions about the origins of matter and the universe. The reason we have included the proposal is its importance. It was the seed that led to the creation of World Wide Web (WWW). The website for CERN—http://public.web.cern.ch/Public/Welcome.html (last accessed on January 29, 2015) rightly paraphrases the introduction of CERN with the phrase "… where the Web was born!"

CERN is "its own sort of United Nations of the scientific world" where 6500 scientists from 80 countries work together (URL: http://www.exploratorium.edu/origins/cern/place/index.html—last accessed on January 29, 2015). The proposal was written to solve the problem of linking together different kinds of information—"reports, experiment data, personnel data, electronic mail address lists, computer documentation, experiment documentation, and many other sets of data." It proposes the novel but simple concept of using hypertext to provide a single user interface to access different classes of information stored at remote systems using networks. The proposal is rather short but is quite specific and concrete. It includes information already available on hypertext concepts and provides proposed work information on applications, scope, requirements analysis, architecture, building blocks, project phases, and resources required. The proposal also outlines future work.

The reason this rather limited proposal became the beginning of the now exponentially significant WWW is the simplicity of the proposal and the fact that it elegantly addressed a highly significant problem that existed in the large community of scientific and academic computer users.

Knowledge Contribution

Contribution Type

This proposal outlines extending *known solutions* (hypertext) to solve a *new problem* (creation of WWW). Even though the proposed use of the existing knowledge is very innovative and has dawned a revolution, it must be classified as *adaptation* type of knowledge contribution (see Figure 2.5).

Status of Design Theory

The proposal provides information on purpose and scope, and includes constructs such as WWW, but the design science knowledge in the proposal is not a design theory. Multiple research communities have been working on the concept of the proposal that is now known as the WWW. However, there is still not a well-developed design theory of the WWW.

Research Patterns Usage

Problem Selection and Development Patterns (Awareness of Problem Phase)

Problem Formulation

The problem is formulated based on the observed needs of CERN to utilize the available HyperText technology to integrate together information within the organization through a common interface thus overcoming a major problem of not being able to look up existing information because of incompatibilities of platforms and tools. The problem is stated in a way that makes it sound like a development problem instead of a major research problem. It is the solution approach that has made the solution a major advance.

Cost-Benefit Analysis

As is expected from a good proposal, it first sketches the benefits of the proposed work. The stated benefits were the ability to access information of various kinds such as reports, notes, data bases, documentation, online help, all of which had been created and stored autonomously, using a common user interface and hyperlinks. The needed resources—people (system architects, hyper-librarians, software

engineers), workstations, software, computer support, office area—are then outlined. The project is divided into two phases—the first phase lasting three months and the second phase lasting six months.

Being Visionary

The proposal envisions a radical departure from the existing environment in which data and information were not available in a timely fashion leading to frustration, wasted time, and obsolete answers.

Literature Search Patterns (Awareness of Problem Phase)

Industry/Practice Awareness

The work is obviously strongly tied to practice and its awareness. It is motivated by the identification of productivity impediments that needed to be removed. This is a good example of research advance that resulted from a bold attempt to solve a real problem in practice.

Suggestion and Development Patterns (Suggestion/Development Phases)

Research Community Tools and Techniques

The authors of the proposal are obviously aware of the tools and techniques used in this type of research. Prototyping is correctly selected as the appropriate technique to demonstrate the proposed concept and the feasibility of its implementation.

Empirical Refinement

The project focuses on the essential aspects of the project, which is ambitious but doable. They allude to the fact that completion of the two phases of the project will provide "an extremely useful set of tools," which would be further enhanced in future and would be studied for its use and abuse at CERN. Both of these observations were extremely prescient given the subsequent rise of the WWW and the research and development efforts that it gave rise to.

Easy Solution First

The project attempts to implement a simple scheme that provides a basic protocol for requesting diverse types of human readable information stored in different types of servers on a network using HyperText to serve as a single user-interface. This way the project focuses on the essential idea instead of complex issues and

enhancements such as the use of fancy multimedia or the use of sophisticated network authorization systems.

Elegant Design

The proposed design has all the characteristics of elegance. It is general in terms of the types of data files and the types of servers, display devices, and browsers used.

Hierarchical Design

The overall system is divided into two building blocks, browsers and servers, and how the two can be linked together. Design issues for each of the components are identified and solutions proposed.

Sketching Solution

The authors provide a succinct outline of their proposed solution within a proposal that is six pages long. The solution sketch clearly brings out the central concept of the solution as well as the areas that need to be the focus of the project.

Combining Partial Solutions

The contribution of the project is not in proposing a new technology but the concept of a simple protocol that forms the glue for utilizing the technologies of HyperText and HTML for linking together diverse types of information on different types of servers connected through a network. HyperText is the main underlying technologies used in the solution.

Evaluation and Validation Patterns (Evaluation Phase)

Demonstration

The authors propose to demonstrate their concept through a carefully designed prototype that demonstrates the generality as well as the feasibility of its implementation. This is quite appropriate for the objectives and non-objectives listed in the proposal and in a situation where a novel concept is being proposed for the first time.

Entity-Relationship Model—Toward a Unified View of Data

Source

Chen, P. (1976). "The Entity-Relationship Model: Toward a Unified View of Data." *ACM Transactions on Database Systems* 1(1): 9–37.

Knowledge Contribution

Contribution Type

The type of knowledge contribution (see Figure 2.5) of this research work is *invention*. It addresses the *new problem* of conceptual modeling of data for the purpose of designing databases. It invents the *new solution* and associated knowledge of modeling entities and relationships between entities and introduces a new diagrammatic technique—entity-relationship diagram—as a modeling technique. This research has dawned the field of conceptual modeling in the database area.

Status of Design Theory

The following is an assessment of the contributed design science knowledge as a design theory with respect to the profile shown in Table 2.2.

Core Components

- **Purpose and Scope:** The research generalizes how data is modeled for the design of databases. It proposes to capture the semantics of data by modeling data from the standpoint of the problem domain where the need to design databases arises.
- **Constructs:** Entities, relationships, and their graphical representations.
- **Knowledge of Form and Function**: The ER model and methods for converting an ER model into an equivalent relational, network, or entity set model.
- **Abstraction and Generalization:** An ER model of data is at a higher level of abstraction than even the corresponding relational model. Thus, the derived relational model or its internal representation of data can be changed while still complying with the overall ER model.
- **Evaluation and Validation Propositions:** (1) The ER model is a generalization of the three other existing data models—network model, relational model, and the entity set model. (2) The ER model can be used for designing databases. The truth of the first proposition is demonstrated through the presentation of methods for converting an ER model into the corresponding models at the lower level. The validity of the second proposition is shown through demonstration using an example of database design. The paper implicitly makes a claim for a third proposition—that the model has cognitive and/or practical utility. However, no work is reported on the testing of this proposition.
- **Justificatory Knowledge:** The model is based on set theory and relational theory. These theories provide justification for the ER model and its concepts.

Optional Components

- **Principles of Implementation:** The research does not propose the direct design and implementation of databases. Instead, it shows how an ER model can be converted to one of the other existing models such as the relational model, which in turn can be used for the design and implementation of a database.
- **Expository Instantiation:** The reported research demonstrates through an example the method for converting an ER model into an equivalent network, relational, or an entity-set model.

Conclusion on Status of Design Theory

The preceding analysis indicates that the contribution is providing rudiments of a nascent design theory. This paper has given rise to an entire research community on conceptual modeling and we now have broad design theory for the area.

Research Patterns Usage

Problem Selection and Development Patterns (Awareness of Problem Phase)

Research Conversation

The paper shows the author's awareness of the research conversations going on in the database community with respect to the prevailing data models and their strengths and weaknesses. The author identifies a knowledge gap from an analysis of the existing literature.

Being Visionary

Chen shows awareness of the literature on existing data models and their strengths and limitations. The network model can provide a natural representational view of data but its capability to achieve data independence between how it is represented and its use in applications had been challenged. The relational model provides a high degree of data independence but may not capture important semantic information about the domain being modeled. The entity set model also provides a high degree of data independence but introduces a degree of artificiality by treating everything, including a value, as an entity. Chen envisions a model that generalizes these models while modeling data at a conceptual level.

Literature Search Patterns (Awareness of Problem Phase)

Understanding Research Community

The author shows a good understanding of the literature discussing the differences between the network and the relational model as well as attempts at reducing the differences between the two models.

Framework Development

The author extends an existing framework for a deeper understanding of the existing literature. The framework contains four levels that range from the conceptual level (information existing in people's minds) to the physical level (access-path dependent data structures).

Suggestion and Development Patterns (Suggestion/Development Phases)

Hypothetical/Deductive Approach

The research is using this approach to develop a design theory for the conceptual modeling of databases. Based on intuition and understanding of the existing literature and its shortcomings, it is hypothesizing that the new model based on the concepts of entities and relationships generalizes the existing data models and can be used for modeling and designing a database and then demonstrating the validity of the hypotheses.

Problem Space Tools and Techniques

Chen specifically addresses the problem of modeling data at a conceptual level using graphics to represent the model. The graphics he proposes to use have been around for a long time.

Different Perspectives

Aided by a framework, the author is able to present a new perspective on data modeling that was not addressed by the existing models. This perspective is that of conceptual modeling that is critical to understanding data and its relationships from the problem perspective.

General Solution Principle

Chen shows that the new model that he proposes is a generalization of existing data models. He shows that the three existing data models in the literature can be derived from the entity-relationship model.

Evaluation and Validation Patterns (Evaluation Phase)

Demonstration

Chen uses parts of an example drawn from the manufacturing domain to demonstrate the new model along with a diagrammatic technique and its use in database design. Even though the presented work on evaluation and validation may look to be sparse, the work in this area seems to be adequate given the objective of the research—to define a new data model and to show that it generalizes or extends the existing data models. The significance of the contribution is demonstrated by the fact that literally hundreds of studies have since been performed on this model showing its cognitive and practical utility.

Logical Reasoning

The existing data models—relational, hierarchical, network, and entity set—are analyzed and it is shown through logical arguments that they can be derived from the entity-relationship model.

Publishing Patterns (Conclusion Phase)

Use of Examples

The author uses parts of a running example from the manufacturing domain to illustrate the use of the new model and to enhance the readability of the paper. The example deals with entities such as employees, departments, projects, suppliers, and parts that the reader can easily relate to.

Case-Based Database Design Support System

Source

Choobineh, J. and Lo, A. (2005). "CABSYDD: Case-Based System for Database Design." *JMIS* 21(3): 281–314.

Knowledge Contribution

Contribution Type

The type of knowledge contribution (see Figure 2.5) by this research is *improvement*. It addresses the *known problem* of developing automated techniques and tools for conceptual database design. It fills a knowledge gap by developing a *new solution* and associated knowledge for the problem that uses case-based reasoning to utilize stored and indexed cases of conceptual database design. The solution involves using

the case-based reasoning approach along with the use of expert system development technology for the creation of cases when no similar cases already exist for the guidance of a new conceptual database design problem. In addition, the instantiation of the approach involves a customized indexing method and the use of a learning algorithm.

Status of Design Theory

The following is an assessment of the contributed design science knowledge as a design theory with respect to the profile shown in Table 2.2.

Core Components

- **Purpose and Scope:** The purpose and scope is limited to investigating a new approach and its efficacy for developing automated tools and techniques for conceptual database design.
- **Constructs:** The research does not propose any new constructs but shows how to implement constructs—the extended entity-relationship model and associated concepts to facilitate case-based reasoning.
- **Knowledge of Form and Function:** An indexing and learning method; discovery and construction of rules for case-based searching and retrieval. Adaptation and learning for conceptual database design.
- **Abstraction and Generalization:** The proposed approach has learning as a component of the approach and thus provides a degree of generality to the approach. The system resulting from use of the approach can evolve using learning.
- **Evaluation and Validation Propositions:** A system designed following the proposed approach will perform better and will be preferred over a design that uses first principles (i.e., one that does not use a case-based reasoning system). An experiment was conducted using a system instantiating the proposed approach and another system that did not follow the approach. The experiment supported the proposition and also resulted in fewer errors by the subjects.
- **Justificatory Knowledge:** The justificatory knowledge comes from existing knowledge on design theories in the areas of case-based reasoning and expert systems. The novelty of the contributed solution is in showing how such knowledge can be used.

Optional Components

- **Principles of Implementation:** The principles of implementation are not explicitly spelled out but may be implicit in the details provided for the implementation of CABSYDD that follows the proposed approach.

- **Expository Instantiation:** CABSYDD is the expository implantation of the approach and is used for experimentation.

Conclusion on Status of Design Theory

The preceding analysis shows that the contribution is a nascent design theory that makes an incremental addition to the general design theory for the area. Additional work by the research community can lead to a more robust and general theory.

Research Patterns Usage

Problem Selection and Development Patterns (Awareness of Problem Phase)

Leveraging Expertise

The authors had worked together before on survey research in the same field (database design support systems). Beginning research in a new field with a survey paper to become familiar with the field and possibly determine gaps in the literature is a very productive strategy.

Research Conversation

Their prior survey work in the field has allowed the authors to identify a research conversation—automated database design support systems—in which to participate. This positions them in a paradigmatic community as researchers who are familiar with the problems and techniques of exploration of the problems in this area, who perceive the problems as important, and whose prior research provides grounding for the current research. Assuming reasonable novelty for the new contribution, publication is easier than is usually the case for research in new fields or on problems not previously identified.

Experimentation and Exploration

The authors chose to frame their "problem"—more effective database design tools—in the context of an existing, published system: NAICS. This contributes to the understanding of the problem and the prior contribution by the research community as well as provides a firm point of comparison for later evaluation and validation.

Literature Search Patterns (Awareness of Problem Phase)

Note that there is no formal literature search section in this paper. Instead, supporting citations are introduced as the approach to the problem (well known within the

community addressed) is developed. In the concluding section, the contribution of the paper is compared against other published contributions. Placing the comparison at the end of the paper instead of contrasting the approach to prior work at the beginning of the paper is unusual, but works well for an incremental contribution to an acknowledged difficult problem.

Understanding Research Community

A thorough understanding of the research community the authors are addressing has come from their prior survey work in the area. Notice how patterns in various sections of research development (*problem selection and development*, *literature search*, etc.) interleave, as would be expected when the patterns form a true "pattern language."

Suggestion and Development Patterns (Suggestion/Development Phases)

Hypothetical/Deductive Approach

The authors hypothesize that an automated case-based approach to conceptual database design would be more effective than an automated approach that is based only on first principles of database design. They test the hypothesis through prototyping and experimentation resulting in incremental contribution to a general design theory in the problem area.

Research Community Tools and Techniques

The previous survey of the field performed by the authors has given them an overview of the primary techniques in use: prototype building followed by experimental validation.

Evaluation and Validation Patterns (Evaluation Phase)

Demonstration

Like many of the problems addressed in DSR, the optimum design of a database support system is complex; even the concept of optimal is subject to contextual interpretation. For this reason, strong methods of proof are not widely applicable and demonstration and empirical verification are common. Note the authors' careful delineation of *validation* and *verification* in the section titled "Evaluation of the Systems." They *validated* the system by expert evaluation of a demonstration of the system by two expert database designers. Verification is a separate step involving a different but related pattern (in the following text).

Experimentation

The effectiveness of the system is *verified* by the experiment. To demonstrate the improved performance of their advance over an automated system based only on first principles, the authors conducted an experiment involving analysis of the performance of 31 students' use of a case-based and first principles-based design prototype. The results showed that the system resulting from the use of the new approach is preferred as well as results in fewer errors.

Relational Model of Data for Large Shared Data Banks

Source

Codd, E.F. (1970). "A Relational Model of Data for Large Shared Data Banks," *Communications of the ACM* 13(6): 377–387. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 64–69, January, 1983.

Additional Source

Codd, E.F. (1982). "The Relational Database: A Practical Foundation for Productivity (The 1981 Turing Award Lecture)," *Communications of the ACM* 25(2): 109–117.

Knowledge Contribution

Contribution Type

The type of knowledge contribution (see Figure 2.5) made by this research work is *adaptation*. The work innovatively adapts the *known solution* (knowledge) of mathematical relational theory and predicate calculus to solve a *new problem*. The problem of designing databases is not new but Codd is proposing the *new problem* of coming up with a technology to increase the productivity of data processing professionals in implementing application programs dealing with large data banks.

Status of Design Theory

The following is an assessment of the contributed design science knowledge as a design theory with respect to the profile shown in Table 2.2.

Core Components

- **Purpose and Scope:** The research addresses the problem of modeling and designing databases at a logical level so as to increase the productivity of programmers for developing application programs dealing with a large amount of data. The research does not concern the design of file structures for storing data.
- **Constructs:** Concepts such as relations (commonly known as tables), relationships between relations, functional dependency, and normal form.
- **Knowledge of Form and Function:** The relational database model; a simple normal form concept and method for a relation to achieve that form.
- **Abstraction and Generalization:** A central contribution of the relational model is to make significant progress toward data independence: separating the logical and the physical aspects of database management. This indicates a degree of abstractness of the relational model. Thus, simple changes in the base tables and any changes in how the data is physically stored will not affect the overall relational model and thus applications developed using the relational model will also not be affected.
- **Evaluation and Validation Propositions:** There are two types of such propositions. The first type has to do with the ability of the relational model to design any database. This is demonstrated using a real-life example. The other has to do with the performance of a relational database management system and whether it will perform as well as a non-relational database management system. This type of proposition has been left for future work.
- **Justificatory Knowledge:** The research utilizes the kernel theory of set and relational theory. It could have also utilized the knowledge about human cognitive processes.

Optional Components

- **Principles of Implementation:** The normalization procedure is introduced to provide guidance on designing a relational database.
- **Expository Instantiation:** An illustrative example is used to show how a relational database can be designed.

Conclusion on Status of Design Theory

The design science knowledge contributed by this research, based on the preceding evidence, forms essentials of a nascent design theory. It has taken years of collaborative work involving academia and industry to develop a well-developed relational database theory and technology. This is illustrative of what generally happens in DSR; see Chapter 3 for discussion of how computing communities of interest collaborate in the development of a technology.

Research Patterns Usage

Creativity Patterns

Wild Combinations

Codd makes a bold departure from conventional thinking. He sees a major gap in knowledge dealing with the problem of data independence—independence between use of data in application programs and their representation in data banks—and in proposing a solution that uses relational theory and predicate calculus.

Problem Selection and Development Patterns (Awareness of Problem Phase)

Research Conversation

Codd shows a good understanding of the existing research in the area. He recognizes that in the database systems that were being developed, the data representation characteristics could not be changed without impairing some application program. He also realizes that the existing data models were cluttered with physical representational properties such as ordering, indexing, and access path dependencies.

Solution/Scope Mismatch

Codd realizes that the database systems using the existing data models—network and hierarchical—were able to support application programs but only as long as the stored data characteristics were not changed or the structure of the files used in storing the data was not changed. This gave rise to the research problem of handling data independence and consistency.

Being Visionary

Codd analyzes the existing models, network and hierarchical, for representing data and envisions a solution that would address the problem of data dependence and inconsistency. The knowledge gap between the existing situation and the envisioned situation is identified as the research problem.

Questioning Constraints

Codd questions the constraint imposed by the database research community of not making a distinction between the logical view of data and its physical representation. This, he claims, opens up degrees of freedom for how data can be logically represented.

Literature Search Patterns (Awareness of Problem Phase)

Industry/Practice Awareness

Codd was working at IBM and was keenly aware of the existing database management systems and their limitations.

Suggestion and Development Patterns (Suggestion/Development Phases)

Hypothetical/Deductive Approach

Codd uses the hypothetical/deductive approach to theory development. He critically reviews the network and hierarchical models, and uses intuition and his extensive background in mathematical modeling to develop the relational model and essentials of its associated theory.

Elegant Design

Codd creates an artifact, the relational model, which can be functionally described as supporting data independence and consistency instead of the details of its construction. The artifact therefore has the characteristics of an elegant design.

Different Perspectives

Codd provides a different perspective on data modeling by making a distinction between physical data modeling and logical data modeling, which is at a higher level of abstraction.

Evaluation and Validation Patterns (Evaluation Phase)

Demonstration

Codd does not develop a prototype because his goal was to demonstrate the new concepts that he proposes at a theoretical level. Instead, he uses the example of a data bank containing data about parts, projects, and suppliers to demonstrate that the solution he proposes is realizable and valid.

Publishing Patterns (Conclusion Phase)

Novelty and Significance

Codd wrote his paper in such a way as to clearly show the novelty and significance of his research. In the introduction of the paper, he positions his research with respect to prior research on data modeling and shows a gap in the existing knowledge on

data independence. In the concluding section of the paper, he states the many questions raised in the paper but left unanswered such as the linguistic details of the needed data languages and their implementation. This opening up of a new field points to the novelty and significance of the work.

Use of Examples

Codd uses an elaborate example of a data bank (database) containing data about parts, projects, and suppliers to illustrate how the relational model can be used and to demonstrate how the model achieves data independence. The example makes the paper better understandable and also convinces the reader about the significance of the research.

Automating the Discovery of AS-IS Business Process Models: Probabilistic and Algorithmic Approaches

Source

Datta, A. (1998). "Automating the Discovery of AS-IS Business Process Models: Probabilistic and Algorithmic Approaches." *Information Systems Research* 9(3): 275–301.

Knowledge Contribution

Contribution Type

The type of knowledge contributed (see Figure 2.5) by this research is *adaptation*. The research addresses a *new problem*—developing strategies for systematic and automatic extraction of AS-IS business process models. To develop such strategies, the research innovatively adapts *known solutions* (knowledge) in the areas of process modeling and grammar discovery.

Status of Design Theory

The following is an assessment of the contributed design science knowledge as a design theory with respect to the profile shown in Table 2.2.

Core Components

- **Purpose and Scope:** The purpose of this work is to develop procedures for automating the discovery of existing real-life AS-IS business processes.

- **Constructs:** Concepts of events, states, and activities are used to define the process activity graph, which in turn is used to model an AS-IS business process.
- **Knowledge of Form and Function:** Three systematic strategies (procedures) for extracting AS-IS process models based on stochastic modeling and finite state machine synthesis methodologies.
- **Abstraction and Generalization:** The proposed procedures for the discovery of AS-IS business process models are based on methods and techniques adopted from existing literature in process modeling and grammar discovery. These AS-IS procedures have a degree of abstractness since they can improve with improvement of the underlying methods and techniques.
- **Evaluation and Validation Propositions:** The testable proposition of the research is that the developed procedures will work reasonably well for automated discovery of real-life AS-IS business processes. The validity of the proposition has been demonstrated with a case-study that applied the developed procedures to large real-world problems.
- **Justificatory Knowledge:** The work is based on kernel (design science) theories drawn from work on process modeling and grammar discovery. The research shares the basic intuition behind the proposed process discovery strategies.

Optional Components

- **Principles of Implementation:** No implementation guidelines are provided in this work.
- **Expository Instantiation:** The proposed procedures have been implemented and then used in a case study.

Conclusion on Status of Design Theory

The preceding analysis indicates that the design science knowledge contributed by this work constitutes a preliminary nascent design theory. This work can be developed further to create a deeper and general design theory on the discovery of AS-IS business process models.

Research Patterns Usage

Problem Selection and Development Patterns (Awareness of Problem Phase)

Note: The author has posed a problem toward which no prior work has been directed. For this reason more effort is taken to justify the value to practice of the problem and the approach to its solution than would be necessary for the presentation of

research that incrementally advances the solution to a previously researched (and acknowledged as important) problem.

Problem Formulation

The paper begins by identifying a previously unarticulated problem. The problem is inferred from the literature on workflow management, business process reengineering, and organizational management, where the assumption has been made that AS-IS processes are known. The paper first develops the case that in practice AS-IS processes are frequently not known and are expensive to determine. Then the concept of a process activity graph (PAG) is defined carefully as a partial but extremely important part of the solution of the problem. An entire section (Section 3) is devoted to defining and defending the utility of a PAG.

Bridging Research Communities

The research draws heavily from the communities of workflow management, business process reengineering, and grammar discovery as previously applied to software process discovery. The author acknowledges the degree to which prior research in software process discovery informs the presented research.

Structuring an Ill-Structured Problem

The problem of automated discovery of complete process descriptions from actual process event traces is extremely difficult. The paper approaches this by decomposing the total problem into components and demonstrating that a PAG (which the research presented in the paper is able to discover) is a vital and necessary component of a total automated discovery of AS-IS processes. (Speculating from experience, the authors of this book wonder if perhaps Section 3 of the paper, a careful development of and defense of the PAG as a necessary component of a complete process description, was not necessitated by reviewer comments of an earlier draft of the paper. In other words, the research the authors present automates the discovery of PAGs from process event traces. This is not a complete solution to the problem of automated discovery of business processes, and early reviews of the paper may have required the authors to defend the significance of their contribution. In so doing, a more structured view of the overall problem is introduced. Whether or not Section 3 was actually so motivated, the after-the-fact defense of a research contribution in response to reviewer comments is quite common and frequently has a constructive result.)

Interdisciplinary Problem Extrapolation

Work from the related area of software process discovery on the use of grammar discovery to reveal processes maps from event traces strongly informs this research.

Literature Search Patterns (Awareness of Problem Phase)

Note that there is no explicit literature review section for this paper. Instead, the relevant and supporting literature is introduced into the discussions of the appropriate sections. In Section 3, citations are introduced to support the adequacy of the PAG for modeling business processes. In Section 4, the process discovery strategies presented in the paper are grounded in prior cited work on grammar discovery. In Section 6, the basis for the algorithmic model of process discovery introduced in the paper (one of two novel discovery methods) is grounded in the literature.

Framework Development

The literature supporting the research draws from multiple fields, none of which address the exact problem for which the research proposes a partial solution. Thus, it is necessary for the author to create an intellectual structure for the work, more carefully developing the point of departure for the research than would be necessary for a previously researched problem.

Industry/Practice Awareness

Throughout Sections 1 and 2, the author repeatedly stresses the real-world aspects of the problem addressed by the research, bolstered by frequent general citations from workflow and process management, that is, citations not directly supporting the technical aspects of the research contribution.

Suggestion and Development Patterns (Suggestion/Development Phases)

Note: There is quite a bit of synthesis in this paper, as would be expected when the research contribution is directed toward a novel problem.

Hypothetical/Deductive Approach

The research hypothesizes that the existing theories in process modeling and grammar discovery can be adapted to develop strategies for systematic and automatic extraction of AS-IS business process models. Following this hypothesis, it develops appropriate procedures and evaluates the hypothesis using a case study.

Combining Partial Solutions

The research draws heavily from prior work in software process discovery using grammar discovery. The problem addressed is, however, sufficiently different than

additional techniques such as Markov chain modeling and finite state machine synthesis need to be incorporated into the final solution.

Interdisciplinary Solution Extrapolation

As previously mentioned, work from multiple fields: process modeling, workflow management, computer science (finite state machines), and operations research (Markov chain modeling) is recognized as a necessary component of the research solution.

Abstracting Concepts

The research hinges on the author's ability to recognize a basis for a solution to the problem the research addresses in the prior work on software process modeling via grammar discovery. The prior work is abstracted to a general approach to analogous problems.

Integrating Techniques

Work from the multiple fields previously mentioned not only grounds and supports the approach but is drawn into a complex synthesis to provide three novel approaches to real-world process discovery from event traces.

Evaluation and Validation Patterns (Evaluation Phase)

Note: the problem addressed is complex and does not lend itself to closed form solutions. Further, the author's probabilistic approach in itself precludes formal proofs of correctness. Thus, the research contribution is partially validated with reasoning and a case walkthrough (demonstration).

Using Metrics

The author develops and uses metrics for evaluation of the process discovery strategies.

Logical Reasoning

The metrics introduced by the author themselves could be problematic; however, the author relies on "self-evident reasonableness" as a validation of the metrics. The way in which these metrics are potentially satisfied by the strategies is discussed.

Demonstration

In Section 8, the metrics introduced in the prior section are shown to be satisfied in a walkthrough of a simple case to which they have been applied. The merits of the PAGs generated by the different strategies relative to the metrics are discussed.

Working Set Model for Program Behavior

Source

Denning, P.J. (1968). "The Working Set Model for Program Behavior." *Communications of the ACM* 11(5): 323–333. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 43–48, January 1983.

Additional Source

Denning, P.J. (1980). "Working Sets Past and Present." *IEEE Trans. on Software Engineering* 6(1), 64–84, January 1980.

Knowledge Contribution

Contribution Type

The type of knowledge contribution (see Figure 2.5) of this research is *improvement*. It develops a *new solution*—based on the concept of the working set—to a *known problem*; the problem of multi-programmed computer memory management.

Status of Design Theory

The following is an assessment of the contributed design science knowledge as a design theory with respect to the profile shown in Table 2.2.

Core Components

- **Purpose and Scope:** The purpose of this work is the development of a model for a general treatment of resource allocation for a computer system.
- **Constructs:** The working set and associated concepts such as processor demand, memory demand, and system demand.
- **Knowledge of Form and Function:** The working set model, method for implementing memory management using working sets.
- **Abstraction and Generalization:** The contribution of this work is at a conceptual level. The concept of using the working set as the basis of computer

memory management can be implemented in multiple ways. Thus, the conceptual contribution allows for different ways of implementing the concept.

- **Evaluation and Validation Propositions:** Working set memory management can be "tuned" for near-optimum memory management. The paper provides only logical arguments to support this proposition. The proposition has been substantiated through years of theoretical and experimental work carried out after the publication of the paper.
- **Justificatory Knowledge:** Prior work in the area of operating systems including work on simulations of the RCA Spectra 70/46 time-sharing operating system that used concepts similar to working set and memory balance.

Optional Components

- **Principles of Implementation:** The reported research does not provide any implementation guidelines.
- **Expository instantiation:** None

Conclusion on Status of Design Theory

The contributed design science knowledge includes the core components of a design theory at a conceptual level and thus the work must be considered as a preliminary version of a nascent design theory. It is only after years of work by the research community after the publication of this work that the working set concept has led to efficient methods for measuring a program's intrinsic memory demand, has assisted in understanding and in modeling of program behavior, and has been used as the basis of optimal multi-programmed memory management (Denning 1980). Thus, the research eventually led to a broader design theory for the area.

Research Patterns Usage

Problem Selection and Development Patterns (Awareness of Problem Phase)

Research Conversation

Denning analyzes the research conversations going on in the operating systems community through conference and journal papers. He finds that the resource allocation problem for multi-programmed computers (in which multiple programs execute at the same time) has progressed independently for allocating core memory and for process scheduling. He reasons that the absence of a general treatment of resource allocation is due to a "lack of an adequate model for program behavior" and proceeds to fill this knowledge gap.

Solution-Scope Mismatch

Denning analyzed a number of existing memory management algorithms and found that while they worked well in particular constrained situations, they did not do as well in the general situation. For example, the first-in/first-out strategy works well when the programs exhibit a sequential instruction fetch pattern. Similarly, the least-recently-used page selection strategy works well in a single process situation but not in a multi-process situation. He set out to address the problem in the most general situation.

Being Visionary

Denning envisions an approach in which the management of system resources—memory allocation and process scheduling—is addressed through a uniform approach in which the operating system balances processor and memory demands against available resources based an analysis of program behavior.

Literature Search Patterns (Awareness of Problem Phase)

Understanding Research Community

Denning develops his research problem based on an in-depth understanding and analysis of the operating systems research community. He also credits the working set concept to a number of reports associated with the pioneering research that was done at MIT under the auspices of Project MAC.

Suggestion and Development Patterns (Suggestion/Development Phases)

Hypothetical/Deductive Approach

Based on prior work in the operating systems area and intuitions, Denning hypothesizes that the concept of working set can be used as a basis for modeling resource allocation for a computer system. He uses logical arguments to argue for the validity of the hypothesis.

Elegant Design

The central concept of the working set model is the working set of pages associated with a process, defined to be the collection of its most recently used pages. The model is general and can be described in terms of its properties.

Different Perspectives

Denning, while understanding and building on the existing literature, provides a different perspective on what should be done to solve the problem. He initiates an analytical approach for examining the properties of the proposed working set model. He also shows that a computation's processor demand and its memory demand in a multi-programmed environment (where multiple programs are executing at the same time) are the manifestations of the same ongoing computation activity.

General Solution Principle

Denning develops a number of basic properties that must hold for resource allocation in computer systems and develops the working set model as an approach for solving the problem. He then expands on this work to show that the model can be used for balancing the processor and memory demands of a program.

Evaluation and Validation Patterns (Evaluation Phase)

Logical Reasoning

Denning uses logical arguments to show the weaknesses of the existing solutions, to show the reasonableness of the assumptions he makes, and to show how the working set model can be useful as a basis for memory management.

Publishing Patterns (Conclusion Phase)

Novelty and Significance

Denning wrote his paper in such a way as to clearly show the novelty and significance of his research. In the introduction section, he positions his research with respect to prior research on memory management by showing the gap in existing knowledge to be the lack of a unified approach to balancing memory and processor demand. Denning also positions his research as commencement of a stream of research on resource allocation based on the working set model.

Communicating Sequential Processes

Source

Hoare, C. (1978). "Communicating Sequential Processes," *Communications of the ACM* 21(8): 666–677. Reprinted in *Communications of the ACM, 25th Anniversary Issue* 26(1): 100–106, January 1983.

Knowledge Contribution

Contribution Type

This research makes an *improvement* type of knowledge contribution (see Figure 2.5). It is addressing a *known problem*—effectively using a multi-processor machine for executing a single task. The reported research provides a *new solution*—a minimal set of primitives in a programming language that include input, output, and a command based on the notion of Dijkstra's guarded command.

Status of Design Theory

The following is an assessment of the contributed design science knowledge as a design theory with respect to the profile shown in Table 2.2.

Core Components

- **Purpose and Scope:** Design of a simple solution—a minimal set of programming primitives that can be used to write programs—to effectively use a multi-processor machine for executing a single task.
- **Constructs:** Programming concepts such as the concept of a nondeterministic guard, along with a list of events and actions; when any of the events occurs, the corresponding action is taken.
- **Knowledge of Form and Function:** Communicating sequential process (CSP) programming language to highlight the basic primitives that should underlie a programming language used for effectively programming a multi-processor machine for executing a single task.
- **Abstraction and Generalization:** The contribution of this research is at a conceptual level—what programming primitives are minimally needed to handle multi-processor programming. As such, the contributed concepts can be instantiated in multiple ways in programming languages.
- **Evaluation and Validation Propositions:** The minimal set of programming language primitives illustrated in CSP is sufficient to write programs in a multi-processor environment. This is done by writing programs in CSP for a reference set of problems used for demonstrating the versatility of a programming language.
- **Justificatory Knowledge:** Existing design theory related to the design of programming languages.

Optional Components

- **Principles of Implementation:** The research paper does not provide any implementation guidelines.
- **Expository Instantiation:** Definition of CSP programming language and illustration of its use for writing programs.

Conclusion on Status of Design Theory

The design science knowledge contribution has the needed information to be considered as essentials of a nascent design theory. The research contribution—design theory—has over time proved to be very useful for utilizing the very highly parallel machines now being designed and used.

Research Patterns Usage

Problem Selection and Development Patterns (Awareness of Problem Phase)

Research Conversation

Hoare shows his awareness of the literature on computer programming and high level programming languages. He cites literature for methods that have been suggested for using a multi-processor computer to execute a single task effectively. He proposes to synthesize the available literature into a simple solution.

Abstraction

Hoare abstracts the problem of effectively using a multi-processor machine for executing a single task to that of finding a few abstract concepts that should underlie the design of a programming language used for the purpose. He suggests input, output, and concurrency (parallel composition of communicating sequential processes) as fundamental abstract concepts that should underlie any programming language for writing programs that effectively use a multi-processor machine.

Suggestion and Development Patterns (Suggestion/Development Phases)

Hypothetical/Deductive Approach

Hoare uses this approach to develop a basic form of a design theory for designing a programming language using which programs for effectively using a multi-processor machine can be written for executing a single task. He hypothesizes a simple solution that uses a minimal set of primitives and demonstrates the sufficiency of the minimal set by using it to write a reference set of programs.

Elegant Design

Hoare designs a simple programming language with a few primitive concepts that can be used for writing any program that effectively uses parallel processing. Note that parsimony of constructs is a general research principle (cf. Occam's Razor)

across all research methods. It leads to elegant empirical research designs as well as strong and elegant DSR contributions.

General Solution Principle

Hoare shows the generality of his proposed language, CSP, by demonstrating that constructs such as monitors and procedures, and solutions to famous programming problems such as the dining philosophers problem can be modeled using CSP.

Integrating Techniques

CSP adapts and integrates available concepts in the existing literature such as Dijkstra's guarded command and parbegin.

Evaluation and Validation Patterns (Evaluation Phase)

Demonstration

Hoare demonstrates the versatility and generality of CSP by demonstrating how CSP can be used to express solutions to many programming problems that have previously been used in the literature to illustrate the use of various programming language features.

Logical Reasoning

Hoare provides clear reasoning for the motivation of CSP and why a few underlying primitive concepts of CSP are enough to model the many elaborate constructs that were being used in programming languages.

Publishing Patterns (Conclusion Phase)

Aligning with a Paradigm

The work is clearly positioned in the programming and programming languages literature with respect to shared symbols and beliefs of the research community. It uses the well-accepted Backus-Naur form (BNF) notation for specifying CSP and builds on the published work of Dijkstra.

Novelty and Significance

Hoare examines the existing programming literature to show that the operations of input and output were not well understood in a formal sense. He also shows the lack of agreement in choosing from different available solutions for expressing a program that

can be effectively run on a multi-processor machine. He then proposes a simple solution, CSP. The paper thus clearly shows the novelty and significance of its contribution.

Use of Examples

The paper uses a number of well-known examples such as the dining philosophers problem to make the paper more readable as well as to demonstrate its contribution (cf. the use of the "grocery bagging" example to illustrate and validate the smart object paradigm, Chapter 2, "An Example of ICT DSR").

Multilevel Model for Measuring Fit between a Firm's Competitive Strategies and Information Systems Capabilities

Source

McLaren, T., Head, M., Yuan, Y., and Chan, Y. (2011). "A Multilevel Model for Measuring Fit between a Firm's Competitive Strategies and Information Systems Capabilities." *MIS Quarterly* 35(4): 909–929.

Knowledge Contribution

Contribution Type

The type of knowledge contribution (see Figure 2.5) made by this research work can be classified as *improvement*. The research addresses a *known problem*—need for a more fine-grained model for assessing the fit or misfit between a firm's competitive strategies and IS capabilities. The research develops a *new solution* for the problem—a multilevel strategic fit measurement model that can be used to measure the strategic fit of a firm's information systems at both an overall and detailed level.

Status of Design Theory

The following is an assessment of the contributed design science knowledge as a design theory with respect to the profile shown in Table 2.2.

Core Components (McLaren et al. 2011)

- **Purpose and Scope:** The design and evaluation of a multilevel strategic fit measurement model that assesses how well a firm's realized IS capabilities support the firm's realized competitive strategies.
- **Constructs:** Classification of approaches for strategic alignment measurement; conceptualization of the concept: Strategic fit of a firm

- **Knowledge of Form and Function:** The multilevel strategic fit measurement model comprising seven steps. The seven steps start with identifying the set of IS capabilities to be measured according to the type of IS and end with checking for corroboration of the overall and detailed strategic fit of the firm's IS assessed using interviews and archival documents.
- **Abstraction and Generalization:** The contributed measurement model is general in that the measurement instruments generated from the model can be changed according to the type of IS and needs of the firm.
- **Evaluation and Validation Propositions:** (1) The overall assessment of a firm's IS can be used for explaining or predicting the relationship between the firm's strategic fit and its organizational performance. (2) The overall assessment of strategic fit of a firm's IS determined by the measurement model can be used to prescribe how the firm's IS capabilities can be improved to realize its realized competitive strategies. The research assessed the validity of these propositions by evaluating an example instance of the measurement model using a multiple case study in a real-world context. Evidence from the evaluation corroborated the evaluation and validation propositions.
- **Justificatory Knowledge:** Each step of the proposed measurement model is grounded in existing IS theory.

Optional Components

- **Principles of Implementation:** The work does not provide guidelines on implementation of the measurement model.
- **Expository Instantiation:** The measurement model has been instantiated and used for testing of the model.

Conclusion on Status of Design Theory

The preceding analysis indicates that the research contribution qualifies for consideration as a preliminary form of a nascent design theory. This work can be used to create a more general design theory in the area.

Research Patterns Usage

Problem Selection and Development Patterns (Awareness of Problem Phase)

Research Topic Identification

The identification of the problem for this research fits the "classic" DSR model in that it emerges naturally from a business problem, evaluating the fit between an

organization's IS and its strategic business goals. Existing methods of fit assessment are determined to be inadequate. (As part of a pattern language, this pattern is combined here (as it frequently is) with the *problem formulation* pattern, discussed later in the text. The pattern *research conversation* is also used to help determine the novelty and value of the problem to the research community.)

Problem Formulation

Once the general problem is identified, various aspects of the problem (sub-problems) are developed. In this case, the goals for a more granular IS fit assessment method constitute the problem formulation.

Solution-Scope Mismatch

Through a thorough evaluation of existing procedures for IS fit evaluation (see the pattern *industry and practice awareness*), it becomes apparent that no existing fit evaluation method meets the business need the researchers are addressing. In an iterative fashion, the output from this pattern feeds back to *problem formulation* to provide additional definition.

Research Conversation

As discussed earlier, on encountering the initial problem, the researchers embark on an extensive search of prior research and industry practice in the area. This ensures that the research is novel (no reinventing the wheel) and helps position the research in contrast to prior developments in the area. In this case, the presented research is the first to make use of DSR in the development of an IS fit assessment method.

Literature Search Patterns (Awareness of Problem Phase)

Industry and Practice Awareness

The researchers evaluate all existing tools thoroughly in order to determine their applicability and/or shortcomings.

Understanding Research Community

As a result of the extensive literature search into prior methods of IS-strategy fit assessment, the researchers become aware of the traditional approaches to research in this area—conceptualization of an instrument on theoretical grounds, followed by survey validation—and the limitations of that approach. This leads them to pursue a DSR methodology instead.

Framework Development

In the process of applying the two prior (immediately preceding) patterns, the researchers develop a detailed typology of existing IS strategic goal fit assessment methods along with the strengths and weaknesses of each. This is a research contribution in itself, strengthening the paper, and also providing a framework for the development of the artifact.

Suggestion and Development Patterns (Suggestion/Development Phases)

Iterative Prototyping; Hypothetical/Deductive Approach

This is an interesting example that first uses *iterative prototyping* and then uses *hypothetical/deductive approach* for theory development. Due to the complexity of the problem and the subtlety of measurement of fit, the authors deliberately invoke first an iterative, prototyping approach. A fit assessment method is designed and tested against a specially developed test case, the shortcomings noted, and the prototype revised. Once the measurement model is developed, it is the hypothetical/deductive approach that is followed in validating the measurement model using real-world cases.

Modeling Existing Solutions

Knowledge gained about the current state of the art as determined by use of the *industry and practice awareness* pattern constituted the starting point of the solution developed by the researchers.

Combining Partial Solutions

Multiple methods existed for determining IS-strategic goal alignment. While none was complete, several offered good practices, concepts, or techniques that the authors integrate into their solution design.

Evaluation and Validation Patterns (Evaluation Phase)

Experimentation

The type of experiment performed corresponds most closely to the hypothetical/deductive method. Following development of the fit assessment method through iterative prototyping, the method is validated through use in five real-world organizations (cases). The cases deliberately are not part of the prototyping (development) assessment of the artifact as this would have introduced bias.

Benchmarking

Two benchmarks are used in evaluating the fit assessment method developed in this research. First, the "Euclidian distance" metric is used to produce a numeric value against which other methods are compared. Due to the expanded capabilities of the newly developed artifact, this benchmark, though used in previous research, is found to be not as useful in comparison with artifacts with identical functionality.

The second benchmark used is "acceptance by the organization" as determined by a qualitative analysis of interviews with organizational users of the fit assessment method and those who have reviewed its output.

Publishing Patterns (Conclusion phase)

Novelty and Significance

The researchers put forth significant effort during the enactment of the patterns problem formulation, *industry and practice awareness*, and *research conversation* to assure that the research problem is significant and that it has not been adequately addressed by prior efforts in the area. Additionally, the researchers note that all prior work has used a theory-based method of development rather than an empirically based DSR approach and point out that this constitutes valuable novelty in and of itself.

Improving Analysis Pattern Reuse in Conceptual Design: Augmenting Automated Processes with Supervised Learning

Source

Purao, S., Storey, V., and Han, T. (2003). "Improving Analysis Pattern Reuse in Conceptual Design: Augmenting Automated Processes with Supervised Learning." *ISR* 14(3): 269–290.

Knowledge Contribution

Contribution Type

The type of knowledge contribution (see Figure 2.5) of this research work is *improvement*. It addresses an *existing problem*—developing a semi-automated methodology and associated knowledge for the conceptual design of systems based on reuse of systems analysis patterns. The research is developing a *new solution* for the problem that augments the naïve approach for conceptual design with supervised learning mechanisms.

Status of Design Theory

The following is an assessment of the contributed design science knowledge as a design theory with respect to the profile shown in Table 2.2.

Core Components

- **Purpose and Scope:** The purpose and scope of this research is to develop a semi-automated methodology for the conceptual design of systems that is based on the reuse of analysis patterns and incorporates learning mechanisms into the naïve approach to conceptual design.
- **Constructs:** Systems analysis patterns, supervised learning mechanisms.
- **Knowledge of Form and Function:** A semi-automated methodology for the conceptual design of systems.
- **Abstraction and Generalization:** The proposed methodology uses supervised learning mechanisms for reuse of analysis patterns, which provides a degree of generality to the methodology. The methodology can improve with the use of better learning mechanisms.
- **Evaluation and Validation Propositions:** The empirical evaluation compares designs produced by the augmented approach proposed by this research with those produced by the naïve approach. The evaluation and validation propositions are: The design produced by the augmented approach contains fewer errors as compared to that produced by the naïve approach and this remains true for different problem sizes and for different domains. The overall results of the empirical evaluation strongly supported the evaluation and validation propositions.
- **Justificatory Knowledge:** The research work draws from existing design theory dealing with systems analysis patterns, conceptual design with reuse of analysis patterns, and use of learning for improving analysis pattern reuse.

Optional Components

- **Principles of Implementation:** The reported work does not provide any guidelines for implementation.
- **Expository Instantiation:** A prototype is developed using Java™, which can be used both as instantiation of the naïve approach for conceptual design (based on reusing analysis patterns) and the proposed approach that augments it with learning mechanisms.

Conclusion on Status of Design Theory

The preceding analysis implies that the design science knowledge produced by the research work is a nascent design theory—an incremental addition to a design

theory for this area. Further work and participation by the research community can result in a broader design theory for automating the conceptual design of systems.

Research Patterns Usage

Problem Selection and Development Patterns (Awareness of Problem Phase)

Problem Formulation

The problem is identified in the literature from ISs and software engineering, and from unanswered questions from the authors' prior research in related areas. The problem is clearly stated and scoped. The difference between the approach presented in the paper and prior (naïve) approaches is clearly delineated and used to help define the problem.

Leveraging Expertise

The problems and approaches in the area were familiar to several of the authors from their prior research.

Bridging Research Communities

The research draws heavily from the communities of software engineering, machine learning, and human learning and cognition.

Structuring an Ill-Structured Problem

In drawing heavily from the well-researched machine-learning community, a structured approach is generated to the complex and not well understood problem of duplicating expert performance in conceptual design.

Literature Search Patterns (Awareness of Problem Phase)

Industry/Practice Awareness

The research is motivated by the longstanding industry problem of facilitating reuse of design components.

Framework Development

The approach to the problem begins with the development of a framework of machine-learning techniques.

Suggestion and Development Patterns (Suggestion/Development Phases)

Hypothetical/Deductive Approach

The research uses this approach to make an incremental addition to the design theory for automating the conceptual design of systems. Using relevant existing design theories, it hypothesizes that augmenting the naïve approach for using systems analysis patterns in at least semi-automating the conceptual design of systems will work better than not using the learning mechanisms. The authors carry out this approach and test the theory using experimentation.

Problem Space Tools and Techniques

One of the prominent activities of this pattern is to "see if there is a promising tool or technique that has been overlooked by the research community." Inclusion of machine learning to instantiate theories of expert cognition in the design area exemplifies that approach.

Interdisciplinary Solution Extrapolation

Use of machine-learning techniques to enhance information retrieval has been explored in multiple fields including Web search. Here, that general solution technique is applied to conceptual design reuse.

Empirical Refinement

Plans for future work indicate plans for refinement and empirical observation.

General Solution Principle

The prototype design-assist mechanism is very general, capable of enhanced and naïve modes, and of trained or untrained modes within the broader enhanced mode.

Evaluation and Validation Patterns (Evaluation Phase)

Demonstration

The paper demonstrates the solution through construction and exercise of a prototype. The demonstration proceeds through a proof-of-concept feasibility study.

Experimentation

Following the feasibility demonstration, a formal experiment is conducted to evaluate the performance of the prototype. (Authors' note: The construction of a prototype followed by both proof-of-concept and formal experimental validation is rare for the type of complex artifact found in this paper and in the ICT design research communities in general.)

Publishing Patterns (Conclusion Phase)

Use of Examples

The paper uses a running example to illustrate the proposed model and diagrammatic technique. The training of the machine-learning modules in the proof-of-concept phase introduces the cases and databases used in the later experiment.

Novelty and Significance

Beginning with the abstract, the paper stresses the novelty of its approach in solving a significant problem. This theme is reinforced throughout the paper.

Optimum Multiway Search Trees

Source

Vaishnavi, V., Kriegel, H., and Wood, D. (1980). "Optimum Multiway Search Trees." *Acta Informatica* 14: 119–133.

Knowledge Contribution

Contribution Type

The knowledge contribution made by this research work is of *adaptation* type (see Figure 2.5). The research addresses the *new problem* of designing as an efficient algorithm for constructing an optimal multiway search tree. It makes an innovative use and adaptation of the general principles of dynamic programming as well as knowledge about a *known solution* (algorithm) for constructing an optimal binary search tree.

Status of Design Theory

The following is an assessment of the contributed design science knowledge as a design theory with respect to the profile shown in Table 2.2.

Core Components

- **Purpose and Scope:** Design of an optimal algorithm for constructing a static multiway search tree, given information on the frequencies of accessing different keys in the tree as well as the frequencies of accessing keys not in the tree. Optimal multiway search trees are useful for storing data in the secondary storage.
- **Constructs:** A new general optimality principle for multiway search trees, which can be tuned for specific related applications; a generalized search cost measure that includes both node visit cost and key comparison cost.
- **Knowledge of Form and Function:** An efficient algorithm for constructing an optimal multiway search tree based on the new generalized optimality principle and its use with dynamic programming; additional efficient algorithms for variations of the basic problem.
- **Abstraction and Generalization:** The basic contribution of the research paper is the new generalized optimality principle for multiway search trees. This principle is used in the solution of the basic problem, construction of a node access cost optimal t-ary search tree. But using the general nature of the optimality principle and the generalized tree search cost measure, a number of other efficient algorithms for search trees can be realized, such as for those with a height or a structure restriction. Some of these variations have been explored in the paper, but there is potential for utilizing the general principles for efficient algorithms for additional related applications.
- **Evaluation and Validation Propositions:** (1) Each of the designed algorithms constructs an optimal multiway search tree with the appropriate definition of tree search cost and restrictions on the height and structure of the tree. (2) Propositions on the time and space complexity of the designed algorithms. Both of these propositions have been mathematically proven.
- **Justificatory Knowledge:** Principles of dynamic programming; research community standard metrics for assessing the space usage and running time of algorithms; knowledge on analysis of space and time complexity of algorithms.

Optional Components

- **Principles of Implementation:** No guidelines are provided for the implementation of the algorithms.
- **Expository Instantiation:** All the algorithms are specified in pseudo-code.

Conclusion on Status of Design Theory

The design science knowledge contributed by this research is a reasonably developed nascent design theory for the chosen problem scope.

*Research Patterns Usage**

Problem Selection and Development Patterns (Awareness of Problem Phase)

Research Conversation

An analysis of the literature revealed that while an efficient algorithm existed for constructing optimal binary search trees, there did not exist any such algorithm for constructing multiway search trees that are used for storing data in the secondary storage. The resulting literature fitted well with the ongoing research conversations in the area.

Solution-Scope Mismatch

Knuth (1971) published an $O(n^2)$ time solution for constructing an optimal binary search tree. This was the only polynomial time algorithm for the problem and was a reasonably good solution. However, binary search trees are useful for storing data only in the primary storage; they are not useful when the data is very large and needs to be stored in the secondary storage such as disk storage. For disk storage, one should use a k-ary search tree, $k > = 3$, with the value of k depending on disk page size and other considerations. Thus, the efficient construction of an optimal k-ary search tree was an interesting research problem. The problem had not been addressed in the literature.

Suggestion and Development Patterns (Suggestion/Development Phases)

Hypothetical/Deductive Approach

After failing to develop an efficient algorithm by a straightforward extension of the dynamic programming-based work of Knuth (1971) for binary search trees, intuition was used to develop a different type of optimality principle for a dynamic programming-based algorithm. It was hypothesized that the use of this principle would lead to an efficient algorithm for multiway search trees and related structures. Such an algorithm was developed and the related correctness and efficiency-related results proven.

Easy Solution First

Instead of trying a new solution technique, Vaishnavi et al. first considered a straightforward application of the solution technique proposed by Knuth (1971).

* As for the first example analyzed in this chapter, the pattern analysis for this example benefits from the personal knowledge of the first author of the book in conducting the research; this explains the use of the past tense in the pattern analysis.

This approach led to an $O(n^{k+1})$ algorithm with a possible improvement to $O(n^k)$. This was not a feasible solution because k can be as large as 500. This gave rise to a research problem that was important and for which extension of an existing technique did not lead to a reasonable solution. Before trying a completely different technique, an attempt was made to apply the dynamic programming technique in a different manner. A novel optimality principle was discovered that was not a simple generalization of the corresponding principle for the binary search tree case. This gave rise to a reasonable algorithm that could also be "tuned" to other such problems with additional constraints.

Modeling Existing Solutions

An existing solution for binary search trees based on dynamic programming was modeled and then modified to develop a solution for the corresponding problem for multiway search trees.

General Solution Principle

A number of basic results that must hold for any optimal multiway search tree were first developed. The authors then identified the dynamic programming technique as an approach for constructing optimal search trees with a number of different additional constraints. Using the general basic results, they developed an optimality principle that could be integrated to the dynamic programming technique to give result to a general solution for a class of problems. They finally tuned the solution to a number of specific instances of the class to improve their solutions.

Evaluation and Validation Patterns (Evaluation Phase)

Using Metrics

The authors analyzed their proposed algorithm and proved that an optimal k-ary search tree can be constructed in $O(n^3 k)$ time, which can be reduced to $O(n^2 k)$ time for a special case of the problem. There was no previously published solution to the problem and the solution provided by the authors had a reasonable polynomial time performance. This showed that the solution is reasonably efficient.

Mathematical Proofs

In this paper, the authors proved that the proposed algorithm would indeed construct an optimal multiway search tree. They also proved the claimed time-complexity of the proposed algorithms.

Publishing Patterns (Conclusion Phase)

Style Exemplars

The work was motivated by a 1971 paper by Knuth published in *Acta Informatica*. Knuth (1971) gave an efficient algorithm for constructing optimal binary search trees that are useful for organizing data in the primary storage. The authors posed to themselves a similar problem for multiway search trees, which are used for organizing data in the secondary storage. Knuth was well regarded in the field. The authors chose to write their paper for *Acta Informatica* and used Knuth's paper as a style exemplar for writing the paper. The paper was accepted without any revision.

Novelty and Significance

The authors develop their research problem in the context of the existing literature showing its novelty and significance. They differentiate the problem of constructing an optimal multiway search tree from that of constructing an optimal binary search tree and discuss the importance of the former problem. They also discuss why an efficient algorithm for the problem does not follow from any existing work including that of Knuth's (1971) work for constructing an optimal binary search tree.

Conclusion

This chapter concludes the book with exemplars showing knowledge contribution in DSR particularly as design theory and the usage of research patterns discussed in the book. Using 12 exemplars, we have shown how the type of each research work can be determined and how the assessment of the work as a design theory contribution can be carried out. Neither the type of the knowledge contribution nor the information needed to conduct its assessment as a design theory was explicitly provided in the research publications. Substantial analysis was required to determine these contributions. We suggest that this type of information should be provided and discussed in the research papers. This will make it easier to judge the research contributions and their relationship to similar research, which in turn will advance DSR.

We sincerely hope that this chapter will serve as the commencement of the reader's interest in or pursuit of DSR. An excellent way of proceeding from this chapter in a research methods course would be to immediately choose examples of DSR from IS or related fields other than those analyzed in this book and proceed with an analysis on these papers similar to the analyses in this chapter.

Additional insights into DSR can be gained by interviewing or even casually speaking with researchers in the midst of a current DSR project. The patterns and the general DSR methodology of Figure 2.3 can be the basis for formal or informal discussions with researchers. Ultimately, a full understanding of the DSR method

can only be obtained through participation in a project using this methodology. However, a technique that approximates the performance of research and is more amenable to a research methods course is the preparation of a detailed proposal for a DSR project. This technique has been used with success for a number of years in courses in the IS PhD program at Georgia State University.

References

Gregor, S. and Hevner, A. (2013). "Positioning and Presenting Design Science Research for Maximum Impact." *MIS Quarterly* 37(2): 337–355.

Gregor, S. and Jones, D. (2007). "The Anatomy of a Design Theory." *Journal of the Association for Information Systems (JAIS)* 8(5): Article 19.

Knuth, D.E. (1971). "Optimum Binary Search Trees," *Acta Informatica* 1, 14–25.

Index

D

N

O

P

Q

R

S

T

U

V

W

Y