Explanatory Design Theory

The paper demonstrates how design theories are explanatory. Design theories deliver functional explanations with a simple and elegant structure explaining generalized solution components by the related generalized requirements. Examples of design theory drawing from IS as well as other design-related fields to show how design theory can be both simple and complete. Analyses of notable design find that design theory consists of two parts: a design practice theory and an explanatory design theory.

DOI 10.1007/s12599-010-0118-4

The Authors

Prof. Richard Baskerville Ph.D.
Georgia State University
35 Broad Street NW
Atlanta, GA 30302
USA
baskerville@acm.org

Prof. Jan Pries-Heje Ph.D. (✉)
Roskilde University
Universitetsvej 1
4000 Roskilde
Denmark
janph@ruc.dk

Received: 2009-12-01
Accepted: 2010-07-03
Accepted after three revisions by Prof. Dr. Winter.
Published online: 2010-09-01

This article is also available in German in print and via http://www.wirtschaftsinformatik.de: Baskerville R, Pries-Heje J (2010) Erklärende Designtheorie. WIRTSCHAFTSINFORMATIK. doi: 10.1007/s11576-010-0237-z.

1 Introduction

As the millennium turned, it was becoming notable that the research discipline of information systems was drifting away from its center on information technologies and becoming too strongly anchored to behavioral and managerial aspects (Orlikowski and Iacono 2001). Design science broadens our scholarly interest beyond the explanation of existing phenomena because it creates valuable utility through the construction of innovative organizational and technical artifacts. In the design science research paradigm, the information systems discipline should seek not only to develop and build theories, but also technological artifacts that lend utility to theory (Hevner et al. 2004). The elevation of design to an equal footing with science is important because there can be an institutional perception that science occupies the higher intellectual ground over engineering or management.

The notion of a *science of design* entails the notion of a *theory of design* (Simon 1996). It is difficult to imagine science without theory, so it becomes incumbent to distinguish design theory that inhabits design science. Accordingly, scholars in information systems have explored and explained what constitutes a design theory.

However, the specific characteristics of design theory seem rather elaborate and overly complicated. Walls et al. (2004) specify seven components including kernel theories, hypotheses, method, etc. Gregor and Jones (2007) specify eight components including artifact mutability, expository instantiation, etc. Further, the emergence of these notions about design theory denies many important characteristics of normal theory. For example, design theory does not explain or predict, but rather informs: "Truth informs design and utility informs theory" (Hevner et al. 2004, p. 80).

It is not clear that layering such complexity into design theories serves the best interests of advancing design science research as a discipline on the level of other sciences. It violates one of the oldest principles of scholarship, the fourteenth century Ockham's Razor (1964; in Latin): "Pluralitas non est ponenda sine necessitate", which can be translated to: "entities should not be multiplied unnecessarily." We seek the simplest possible delineation of a design theory.

It also seems problematic to create such distinct and specialized forms of theory, especially when these theories must entail other theories (such as kernel or justificatory theory). Besides failing to satisfy our expectations for other forms of scientific theory, it creates complicated rationalizations. Authors of studies can be forced to fit their less structured design theories into these complicated frameworks and then appear to be superficial, as if using the frameworks like a "cloak of theoretical legitimacy" (Walls et al. 2004, p. 55).

In this paper we propose that a simplified notion of design theory has more in common with "normal" scientific theories, and is indeed descriptive and offers a particular kind of explanation. We demonstrate that separating the current notion of design theory into component parts yields an explanatory part and a practice part. We then show how existing design work in architecture, finance, management, cognitive psychology, computer science, and information systems can yield explanations.

2 Design Theory

There is a rich literature treating information systems design theory. There is not complete agreement about the characteristics and components of design theories, and of course there is no proof or evidence. Rather these exist as shared assumptions about design theory. In this section we identify a number of these shared assumptions, and discuss the problems and issues in the delineation of design theory. These issues lead to a proposition to partition design theory into an explanatory part and a practice part. We will also treat the generaliz-

ability of the explanatory part of design theory.

2.1 Assumed Characteristics of Design Theory

Design theory has been defined in various ways emphasizing various assumptions. For example, design theory is assumed by many to be *prescriptive*. Walls et al. (1992, p. 37) defined it as, "a design theory is a prescriptive theory based on theoretical underpinnings which says how a design process can be carried out in a way which is both effective and feasible". Prescriptive research, which focuses on improving things, stands in contrast to the descriptive research, which focuses on understanding things (March and Smith 1995).

Design theory is assumed by many to be *practical*. Goldkuhl emphasized this nature of design theories, "Design theories consist of knowledge of a practical character; i.e., for practical purposes" (Goldkuhl 2004, p. 61). Van Aken adapts these notions into the more general realm of management, defining management theory as a design science form; "prescription-driven research and to be used largely in an instrumental way to design solutions for management problems" (van Aken 2004, p. 221).

Design theory is assumed by many to be a *basis for action*. Gregor and Jones (2007, p. 313) find it is a type of theory that determines actions. "The distinguishing attribute of theories for design and action is that they focus on 'how to do something'. They give explicit prescriptions on how to design and develop an artifact, whether it is a technological product or a managerial intervention."

Design theory is assumed by many to be *principles-based*. Marcus et al. (2002, p. 182) emphasized the role principles in their definition of design theory components: "(1) a set of user requirements derived from kernel theory, (2) principles governing the development process, and (3) principles governing the design of a system (i.e., specifying and implementing its features)". While noting "some feeling against design principles as theory", Gregor and Jones (2007, p. 314) include principles of form, function, and implementation among their design theory components.

Design theory is assumed by many to be a *dualist construct*. Design theories regard both design as a product and design as a process. It is the natural outcome of a term that is both a noun and a verb. Simon (1996, p. 131) regards a "theory of design" with two essential components: "the shape of the design and the shape and organization of the design process". Walls et al. (1992, p. 43) divide the components of an Information Systems Design Theory into two classes: "Design Product" and "Design Process". Hevner et al. (2004, p. 83) distinguish guidelines for "Design as an Artifact" and for "Design as a Search Process". Gregor and Jones (2007) separate a design theory's "Principles of form and function" from the "Principles of Implementation".

2.2 Issues in Design Theory

There are also assumptions about what is *not* design theory. For many, design theory does not correspond to the notion of scientific theory as known in the natural sciences. "Natural science includes traditional research in physical, biological, social, and behavioral domains... Such research is aimed at understanding reality. ... Design science attempts to create things that serve human purposes. It is technology-oriented. ... Rather than producing general theoretical knowledge, design scientists produce and apply knowledge of tasks or situations in order to create effective artifacts" (March and Smith 1995, p. 253). In fact, for some, design science should not produce theory. "Design science products are of four types, constructs, models, methods, and implementations. ... Notably absent from this list are theories, the ultimate products of natural science research" (March and Smith 1995, pp. 253–254).

So much effort has been expended in delineating the non-science characteristics of design theory that it leads to questions about whether design theory can even exist. Hooker, for example, finds this assumption space so contradictory to common notions of theory that the entire construct of design theory is impossible. Hooker (2004, p. 2) points out that a theory is "an explanatory account of the way things are". The properties of theories include making "the world intelligible" and "a lawlike (or 'nomic') character". Treating design as theoretical is complicated because design is a practice in which a functional description passes into a physical description of an artifact. If design theory is a theory of practice, Hooker reasons, then it is fundamentally the same psycho-social theory that applies to any other field of practice. In other words, theories about the practical behavior of designers will not differ from theories about practical behavior of biologists. In a similar vein, the design process component of design theory causes many to struggle over whether design science differs in any significant way from the sociological methods of action research (Cole et al. 2005; Järvinen 2007).

March and Smith, together with Hooker, claim that theorizing has a natural science intent, and does not belong in design science. March and Smith take the position "IT research should be concerned both with utility, as a design science, and with theory, as a natural science" (March and Smith 1995, p. 255). Both works concede theory and theorizing to the natural sciences alone using a narrow, natural science viewpoint on theory. Deciding whether, under the assumptions above, design theory is a legitimate type or class of theory would first require us to delineate the criteria for qualifying something as a theory. This puzzle is itself so problematic that management scholars have sidestepped the issue entirely, choosing instead to try defining only what "theory is not" (Sutton and Staw 1995). In other words, deciding how design theory differs from psycho-social theory on the one hand, and action theory on the other hand, would first require us to define what is *not* psycho-social theory and what is *not* action theory. As Hooker's arguments detail, it is hard to imagine any criterion that would characterize the application of psycho-social theory to the design community of practice as different from its application to any other community of practice.

2.3 Explanations in Science and Design Science

Like Hooker, most views of theory will at least admit that one form of theory can be an explanatory account of reality. In the philosophy of science, we can find four types or patterns of explanations: deductive, probabilistic, functional, and genetic. Using Nagel's definitions (Nagel 1961), deductive explanations operate where the conclusions are logically necessary outcomes of the premises. Probabilistic explanations operate where conclusions about a member of a class are the outcome of statistical premises about the class. Deductive explanations are common in the natural sciences, and probabilistic explanations

are common in the social sciences. Genetic explanations operate where conclusions about a phenomenon are the outcomes of the historical evolution of this phenomenon. Genetic explanations can detail how a system is the step-by-step outcome of previous generations of systems. Functional explanations, also called teleological explanations, are centered in design science research. These explanations indicate "one or more functions (or even dysfunctions) that a unit performs in maintaining or realizing certain traits of a system to which the unit belongs" (Nagel 1961, p. 23). Functional explanations are common in biology and the study of human affairs. Functional explanations operate by showing why a role or action of a system is necessary to bring about some goal. Simon (1996) explains how functional explanations serve design science (and other sciences of the artificial) by explaining a system's inner environment as a necessary consequence of the need for the system to function in its outer environment. He uses the example of the chronometer for a ship that reacts to the pitching of the ship (outer environment) by maintaining an invariant relation of the hands on the dial (Simon 1996, p. 8). "An important fact about this type of explanation is that it demands an understanding mainly of the outer environment" (p. 7).

Relationships in functional explanations use such language as "in order that", "for the sake of", etc. There is often a reference to "a future state or event" in terms of which the existence of a construct becomes intelligible (Nagel 1961, p. 25). Accordingly, functional explanations can take two forms, which we will call perpetual and conditional. A *perpetual functional explanation* is given for features or components that are present in all systems of a certain kind, regardless of timing or condition. A *conditional functional explanation* is given for features or components occurring upon a stated condition or time.

Theories in design science provide more than just prescriptions about how to design and construct artifacts for the purpose of achieving some goal. Design theories also serve to provide functional explanations of why designs and artifacts have certain attributes and features (Walls et al. 1992). The design theory explains the attributes and features of the design and artifact (found in Simon's "inner environment") by their necessity in achieving the purpose and goal of the artifact (found in Simon's "outer environment"). Design theories serve not only prescriptive purposes, but also serve functionally descriptive purposes as well.

2.4 Partitioning Design Theory

The management discipline's debate about theory does provide one helpful key to understanding this collision between design theory and other theory disciplines. Karl Weick (1995), writing in response to Sutton and Staw, argued that the focus on *theory* should be distinguished from *theorizing*. In other words, the value of these two elements, the process and the product, was more recognizable when examined separately. For our purposes, the contradictions that challenge the assumption space of design theory fundamentally arise in the dualist construct.

If we separate the theory of the design product from the theory of the design process (i.e., the design from its designing), Hooker's questions about the value of the theory soften dramatically. Hooker notes that fundamentally a design for something is an incompletely described abstraction of reality. Referring only to this incomplete description, and not the process for creating it, he admits, "To begin with, all science abstracts certain features of an object and more or less ignores the rest. In fact, the sciences are defined and distinguished partially by the level and type of abstraction they employ. So, if the proposal is that design science focuses only on certain features of an object – namely, those that belong to the designer's incomplete description of it – then it would seem to be no different in principle from any other sort of science" (Hooker 2004, p. 11).

Walls et al. (2004, p. 50) regarded this admission as an opportunity for design theory to exist on Hooker's terms. "This statement is consistent with our definition of meta-requirements and meta-design, which deal with a class of information system rather than a specific instance of one".

Removing the dualistic assumption from the premises of design theory separates the theoretical component about design practice from the theoretical component about the design artifact. This removal means that there are two types of design theories. One type of design theory, *design practice theory*, prescribes in a practical way how to design something. The second type of design theory prescribes principles that relate requirements to an incomplete description of an object. The nature of the requirements *explains* the incomplete description in terms of the requirements. This type of explanation is consistent with the definitions of functional and teleological scientific explanations. We will designate this type of design theory as *explanatory design theory*.

The incomplete description is the design artifact, not the instantiation of the design. Moreover, the design is incomplete because it describes a class of design problems, not a single specific design problem. The value of an explanatory design theory lies in its ability to explain a range of phenomena rather than a specific instance of a problem. This general explanation means that an explanatory design theory explains why a generalized set of requirements is satisfied by a generalized set of object features. The explanation is embodied in the relationship between a requirement and a feature. For example, Walls et al. (1992) describe a generalized set of requirements for a Vigilant Executive Information System (VEIS), and how these are satisfied by the features of their generalized design for VEISs. This theory has explanatory value and is generalizable across a range of VEIS applications. Similarly Markus et al. (2002) describe a generalized set of requirements for emergent knowledge management systems (EKMS), and how these are satisfied by the features of their generalized design for EKMSs. This theory has explanatory value and is generalizable across a range of EKMS applications.

By eliminating the dualist assumption within design theory, and acknowledging two separate and distinct theory components, the explanatory value of the design product aspect becomes apparent. For the purposes of this paper, we will continue to develop the explanatory design theory and postpone design practice theory for future research.

2.5 Describing Explanatory Design Theory

Simon's original work focused on imperative logics rather than theory, but his theory of design was partly anchored to the General Problem Solver (GPS) (Simon 1996, p. 122). This problem solving software was designed with two elements. One of these elements was based on the differences between the present situation

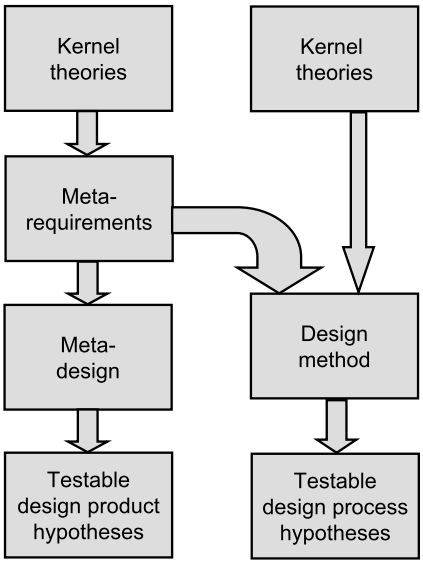

Fig. 1 Design theory according to Walls et al. (1992, 2004)

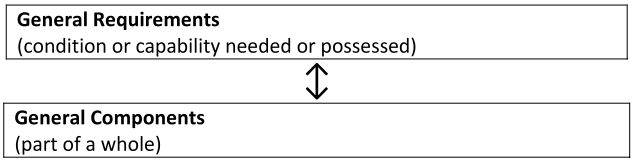

Fig. 2 Explanatory design theory

and the desired objects in some future situation. The second element was the set of actions that changed the objects or situations in order to remove the differences. In the language of current design theory, the differences can be regarded as the requirements. The necessary actions that change the objects or situations can be regarded as the components of the solution. Simon implemented these two key notions in declarative logic as the "utility function" (requirements) and the "command variables" (the components of the solution).

These two key elements in an explanatory design theory, the requirements and the components, and their embodied relationships that explain the solution, can be found in many key works in design theory. Walls et al. (1992) and Gregor and Jones (2007) have the carefully delineated structures that are widely cited, and parsimonious. Gregor and Jones map the "meta-requirements" of Walls et al. onto their element called "purpose and scope". In terms of the components of the generalized solution, they map the "meta-description" of Walls et al. onto their "Principles of form and function".

The various other elements attributed to design theory become peripheral when explanatory design theory is extracted from the dualist structures. For example, the full structure of Walls et al. is illustrated in **Fig. 1**. The elements that embody the explanatory design theory are the meta-requirements and the meta-design (the components). Walls et al. use the notion of "meta-" to emphasize the abstract nature of the design theory. The kernel theories, design method, and testable design hypotheses relate to design practice theory and are unnecessary to explanatory design theory. The kernel theories, which Gregor and Jones call "justificatory knowledge," are separate background theories that form the assumption space for the explanatory design theory. The hypotheses are deduced from the theory, and while possibly important for testing, are not essential to the theory itself. In any case, the hypotheses are at least optional. Hypotheses arise in a version of science that subscribes to a natural science, hypothetical-deductive mode of logic, such as that described by Nagel (1961). These are not always fundamental to diverse forms of social science.

One common element in all of these descriptions of the explanatory component of design theory is the notion of generalized requirements for the class of artifacts under consideration. For Simon, this is the utility function. Another common element is the generalized components that satisfy these general requirements. These two elements, general requirements and general components, together with the relationships between these elements, form the essence of a design theory. We have shown this graphically in **Fig. 2**. This explanatory design theory is a general design solution to a class of problems that relates a set of general components to a set of general requirements. We explained above what we mean by generalized. We use the term 'requirement' in a sense similar to that of the IEEE standard glossary (IEEE Std 610.12.-1990), that is: (1) a condition or capability needed by a user to solve a problem or achieve an objective; (2) a condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document; (3) a documented representation of a condition or capability as in (1) or (2). The standard does not define conditions or capabilities, but the Oxford Advanced Learner's dictionary (Wehmeier 2000, p. 255) defines the term 'condition' as, "the state that something is in", as "the circumstances or condition …", or as "the physical situation that affects how something happens". Further we use the term 'capability' as "the ability or qualities required in the whole set of general components necessary to do something". As a whole, the set of general components provide a generalized solution to the general requirements.

The IEEE standard glossary goes on to define component as, "one of the parts that make up a system". A component may be more than just hardware or software and may be subdivided into other components. For example, a requirement for organizational memory support may be answered by components such as 'Knowledge acquisition', 'Knowledge retention', 'Knowledge maintenance', 'Knowledge Search' and 'Knowledge Retrieval'; all five parts of a whole justified by the general requirement. Another example is a general requirement for ensuring profit without too much risk when investing. This requirement can be answered by portfolio thinking which involves two major component parts namely diversification of investments and combination of different stocks and options whose returns are not correlated.

The definitions of general requirements and general components must be circular. Requirements specify (and explain) the reasons for components. Components are justified by requirements.

The presence of conditions along with capabilities relates to Nagel's distinction between the two forms of functional explanations (which we called conditional functional explanations and perpetual functional explanations). Where unconditional capabilities appear in the general requirements, the explanatory design theory is delivering a perpetual functional explanation. When conditions appear in the general requirements, the explanatory design theory is delivering conditional functional explanations. Conditional explanations approximate an

IF…THEN… kind of logic. Explanatory design theory requirements will usually have at least one capability. If the explanation is not perpetual in nature, then we would expect one or more conditions to be present.

2.6 Explanatory and Constructive Design Theory

Taking an explanatory view of design theory does not eliminate the important role of design theory for prescribing, together with the design practice component, the construction of an artifact. This combined operation provides a constructive theory. The explanatory design theory explains *why* a component is being constructed into an artifact. The design practice component explains *how* to construct the artifact. Unlike theories that yield purely descriptive deductive or probabilistic explanations, design theories provide elements of a prescriptive theory of artifact construction and a descriptive theory for functionally explaining the artifact's intended features and behavior. This constructive role of explanatory design theories is similar to Dietz's (2006) construction theory that links requirements (functional system properties) to components (constructional system properties).

Together, an explanatory design theory and its related design practice theory explain "why" and "how" to construct an artifact. Once an artifact has been constructed, the explanatory design theory component continues to enact an explanatory role, explain an artifact's functions teleologically. In this way, design theories are more powerful and broader in scope than simple descriptive theories, because they are both prescriptive and descriptive in nature.

The usefulness of explanatory design theory is comparatively diverse and powerful. By focusing on the essential elements of design theory, we can discover how it is a constructive theory, prescriptive on the one hand, while remaining available as an explanatory theory, descriptive on the other hand. Because design theory applies as both constructive theory and explanatory theory, it serves roles before, during, and after artifact construction.

Because explanatory design theory has a role both in the explanation and construction of design artifacts, it still satisfies most of the assumed characteristics of design theory described in Sect. 2.1 above. It is both practical and a basis for action because it explains why each component is necessary for an artifact in terms of the artifact's own requirements. It is principles-based in a most straightforward way, because it operates purely on the relationship between two kinds of constructs (requirements and components). However, explanatory design theory is no longer a dualist construct because the process for construction (the design practice theory) has been removed as a separate and distinct category of theory.

2.7 Generalizability of Explanatory Design Theory

The generalizability of an explanatory design theory operates similarly to other kinds of explanatory theory. It will depend on the nature of its expression. It can be stated with more-or-less generality (scope) depending on the level of abstraction. This critical degree of an incomplete description is the degree of its abstract expression. The more abstract the expression of a theory, i.e., the more general its statements, then the more generalizable are its claims. For example, Walls et al. expressed their theory as a general description of a "vigilant executive information system". They did *not* choose to advance a broader theory, one that might have been more generally expressed as a "vigilant information system". They reasoned that the theory was anchored to experience with EISs, and they chose to limit the generality (and scope) of their claims accordingly.

In addition, the level of generality in the expression of an explanatory design theory is related to the completeness of its description of reality (Hooker 2004). The more abstract the notions in the theory, then the more incomplete is the description of reality. For example, because expression of the theory of "vigilant *executive* information systems" is more specific, it permits a more complete description of the more specific solution. Consequently, generality, completeness of description, and abstraction are dimensions of the same human decision. It occurs by leaving out some parts of a particular while retaining other parts, and arriving at general names and general ideas. The creation of the abstract is an individual person's conceptualization process. The process of "abstracting the universal from the particular" can be traced back to Aristotle and arrives in the philosophy of science through Aquinas, Hobbes and Locke (Walmsley 2000, p. 396). It refers to the way in which a person, within his or her intellect, forms an "idea" or "notion" of material phenomena.

While the choice of generality in the expression of an explanatory design theory lies in the mind doing the theorizing, its evidence and testing can be more objective. Explanatory design theories, like all theories, are necessarily tentative. Their prescriptive nature makes them certainly falsifiable. The credibility of the generalizations are dependent on the credibility assigned to the evidence, arguments, and background theories used to develop them. Because of the greater scope in its claims, the validity of a more generally stated theory is subject to wider scrutiny and possible denial. For example, the falsification scope of an explanatory design theory of vigilant *executive* information systems is narrower than the falsification scope of a vigilant information system. Such falsification would involve instantiating the theory with an artifact, and discovering that the resulting artifact did not satisfy the requirements. In such a setting, the explanatory design theory would no longer explain the components in the design product, and the theory consequently fails.

Like other kinds of explanatory theories, more general and more specific theories about phenomena can be seen as hierarchical, with specific theories inheriting the relationships between constructs of more general theories. If the assumptions are consistent, a general theory of a vigilant information systems should inhabit a more specific theory of vigilant *executive* information systems. These notions scale up and down with abstraction. If the assumptions are consistent, a more specific theory of vigilant *executive decision support* systems should reflect a more general theory of vigilant executive information systems.

Because explanatory theories express relationships between the constructs of requirements and components, it becomes necessary to specify and construct an instantiation in order to make validity checks. Goodman (1955) suggests such operations are beyond the theory itself, but are rather outside operations on (or with) the theory. He uses the term *projection* for operations in which a theory is projected into an instance. For us, this task of projection is a fundamental role of design practice theory. Constructively, the design practice theory projects the explanatory design theory into an instance.

3 Explanatory Design Theories in Reference Disciplines

Explanatory design theories are present in the reference disciplines of information systems. Reference disciplines mentioned in Baskerville and Myers (2002) – citing many authors – are engineering, computer science, cybernetic systems theory, mathematics, management science, behavioral decision theory, systems science, political science, psychology, sociology, accounting, finance, management, architecture, economics and anthropology. This is not an exhaustive list; there may be more.

While it is not possible to cover them all, we can 'span the field' by examining a careful selection of design theories. We selected design theories that range from highly behavioral to highly natural-science-oriented disciplines. We also conditioned our selection to range from a unit of analysis set as the society or the organization and ranging to a unit set as the individual. We present five examples out of the host of theories in which it is possible to detect the explanatory design theories in the literature on design. The examples we have taken arise from architecture (patterns), software engineering (faked rationale), portfolio theory (finance), organizational structure (management), and product design (everyday things). In **Table 1** we have illustrated how these examples together cover a very large field and diverse units of analysis.

The presence of multiple requirements and multiple components creates an expectation that components will map to requirements discretely and vice-versa. Sometimes such mapping can be simple and indisputable, but usually such mapping is made problematic by system or problem complexity and its own historicity. Requirements traceability regards the degree to which a particular component can map to a particular requirement. While conceivable, it is beyond the scope of the present paper. In the examples that follow, we will not attempt to unravel the unpublished requirements trace, but will instead follow the model of the original publications and treat the relationships between the requirements and components as holistic.

Table 1 The five examples and the field they span

Field	Discipline	What?	Unit of analysis
Architecture	Building and Engineering	Patterns and patlets	Buildings (&/) practices
Software Engineering	Computer Science	How to fake rationality	Documents
Portfolio Theory	Finance	Minimizing risks and maximizing profit	Assets
Organizational structure	Management	Designing effective organizations	Organizations
Product Design	Cognitive psychology	Designing everyday things	Things

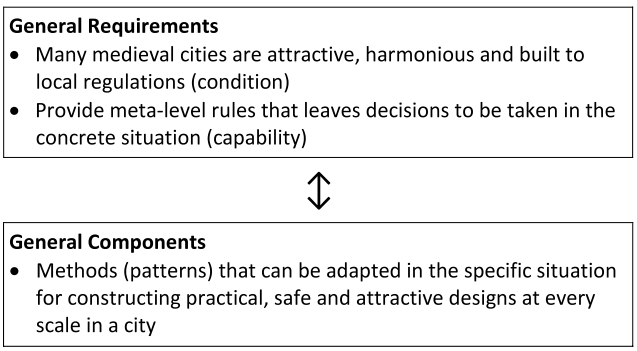

Fig. 3 Pattern as explanatory design theory

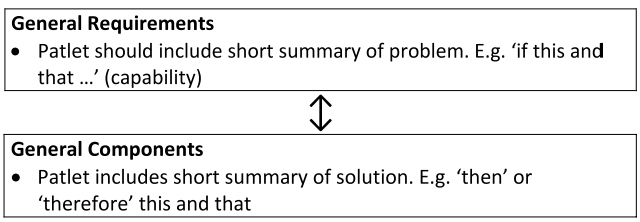

Fig. 4 Patlet as explanatory design theory

3.1 Patterns

The idea of capturing architectural design ideas for reuse in an archetypical form was pioneered under the name *patterns*. Christopher Alexander (1964) opened his book "Notes on the synthesis of form" with this statement: "These notes are about the process of design: the process of inventing things which display new physical order, organization, form, in response to function." In this statement we clearly can recognize the same elements as in our explanatory design theory. See **Fig. 3**. The presence of a condition suggests this theory delivers conditional functional explanations.

Some years later Alexander et al. (1977) constructed a pattern language. In computer science this may be termed as *a generative grammar*. It describes a vocabulary of interacting design patterns. The book describes exact methods for constructing practical, safe and attractive designs at every scale, from entire regions, through cities, gardens, buildings, and down to the doorknob of a door in the building. The pattern language provides rules and pictures, but leaves decisions to be taken from the precise environment of the project.

The idea of patterns was brought into software development and adopted especially by the object-oriented programming community (Coplien and Harrison 2005). In this community the explanation of a *design pattern* lay in its role as a general reusable solution to a commonly occurring problem. In 1995 the "Gang of Four" (Gamma et al. 1995) presented a book that rose to a monumental role for object-oriented software development.

Ten years later Jim Coplien and Neil Harrison (2005) presented nearly 100 Organizational Patterns. At the end of this book the authors explain *patlets* (p. 349): "A patlet is a short summary of the problem and solution for a pattern. Patlets are often used as an

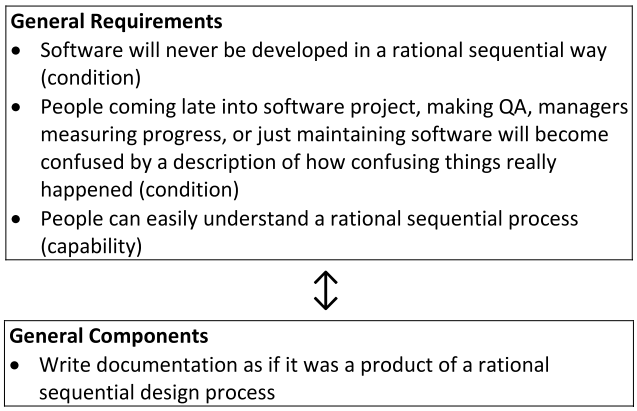

Fig. 5 How and why to fake a rational design process – as explanatory design theory

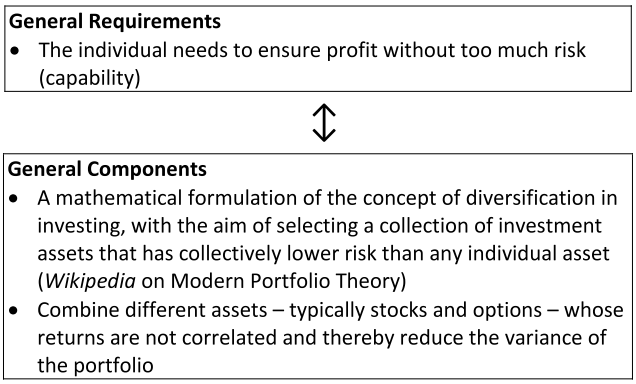

Fig. 6 Portfolio Theory – as explanatory design theory

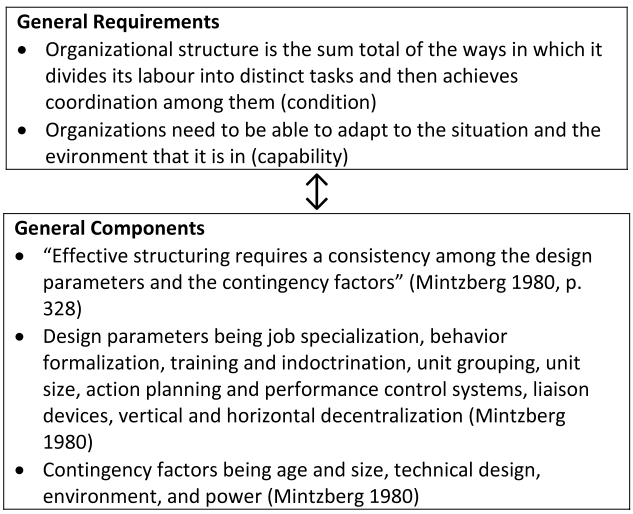

Fig. 7 Effective organizational design – as explanatory design theory

aid to discovering patterns in order to solve a particular problem at hand".

See **Fig. 4** for a representation of a Patlet as an explanatory design theory. The absence of conditions suggests this theory delivers perpetual functional explanations. An example patlet is called Size the Organization (p. 352). It says: "*If* an organization is too large, communications break down, and *if* it is too small, it can't achieve its goals or easily overcome the difficulties of adding more people. *Therefore*, start projects with a critical mass of about 10 people". Patlets are quite consistent with explanatory design theory.

3.2 Faking a Rational Design Process

In "A Rational Design Process: How and Why to Fake It" Parnas and Clements (1986) claim that "… to many observers, the usual process of designing software appears quite irrational …". That is because programmers "make a long sequence of design decisions with no clear statement of why they do things the way they do". The authors argue that this will never change: "… the picture of the software designer deriving his design in a rational, error-free way from a statement of requirements is quite unrealistic. No system has ever been developed in that way, and probably never will". However, they explain that we should write the documentation of a software system in a way equal to what "… we would have produced if we had followed the ideal process". We can translate this theory of the design process to a generalized picture similar to **Fig. 5**. This theory is oriented toward conditional functional explanations.

3.3 Portfolio Theory

When investing money, increasing return is often associated with increasing risk. Diversification of portfolio holdings will sometimes develop high returns while comparatively reducing risk. Portfolio theory explains why pairing assets with opposite risk profiles means that every investment in the portfolio does not go down (or for that matter up) at the same time. This idea and the mathematical expression and implementation of it brought Markowitz (1952) the Nobel Prize in 1990 (for this and two other contributions to economics). In **Fig. 6** the essence of portfolio theory is captured. This theory is oriented toward perpetual functional explanations.

3.4 Organizational Structure

Mintzberg's classic "Structure in Fives" (Mintzberg 1980, 1983) advanced an organizational theory for designing effective organizations. This contingency theory explains effective organizational design with five organizational parts, five coordinating mechanisms, five types of decentralization and so on. Mintzberg's design theory translates into a generalized form in **Fig. 7**. This theory should

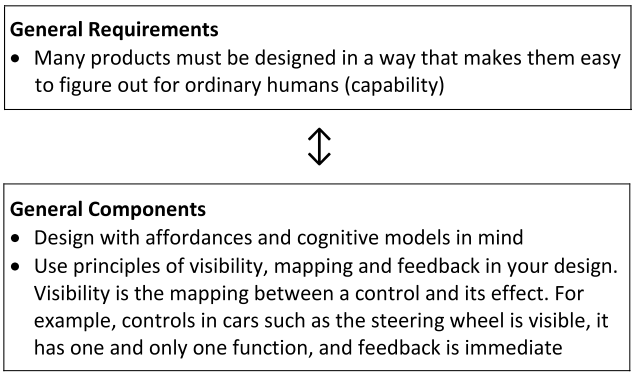

Fig. 8 The design of everyday things – as explanatory design theory

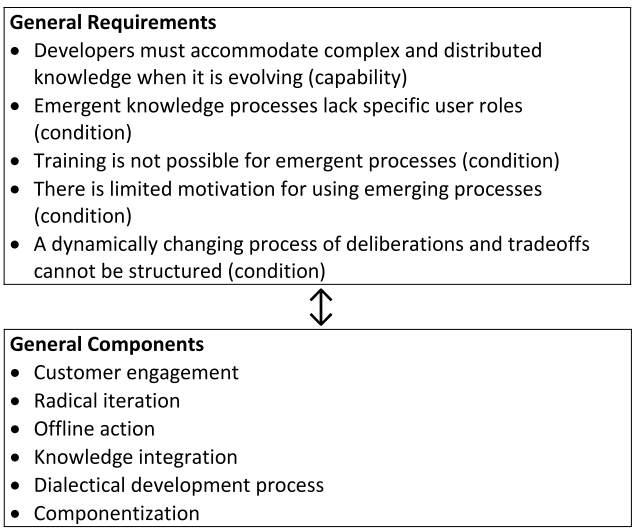

Fig. 9 Supporting emergent knowledge (Markus et al. 2002) – as explanatory design theory

deliver conditional functional explanations.

3.5 The Design of Everyday Things

… is the title of one of most influential books on usability (Norman 1988). Norman begins with the observation that even smartest among us can feel inept as we fail to figure out which light switch to turn on, or whether to push, pull, or slide a door in a building. Norman's work shows how we are not to blame. Instead he explains how the fault lies in poor product design that ignores both the needs of users and basic principles of cognitive psychology.

In his opening chapter, Norman explains a conceptual design model based on making "affordances" available. He advocates the use of visibility (pp. 17–23), as well as a principle of mapping (pp. 23–27), and a principle of feedback (pp. 27–29). He further details many examples of this model in operation. For example he emphasizes to "Use technology to make visible what would otherwise be invisible, thus improving feedback and the ability to keep control" (p. 192). Norman's design theory translates into a generalized form similar to **Fig. 8**. This theory should provide perpetual functional explanations.

4 Information Systems Explanatory Design Theories

Like design theories in our reference disciplines, prominent design theories in information systems will also adapt to this form. Since explanatory design theories are, by definition, a subset of the seminal Walls et al. (1992) model, theories developed using this more elaborate model easily distill to explanatory design theories. The simpler explanatory design theory model also fits other published design theories that do not carefully follow the seminal framework. We will briefly examine seven design theories in the information systems literature and explain how each can be represented as explanatory design theories.

Markus et al. (2002) propose a design theory for systems that support emergent knowledge processes. This paper relates an explanatory design theory that addresses the dual problems embodied by (1) knowledge processes (which are complicated human activities) and (2) dynamic settings (where knowledge processes have to change continuously to match an evolving context). These requirements explain why the design solution involves support for rapidly iterative user involvement. The explanatory design theory (p. 206) is summarized as follows in **Fig. 9**. This theory is oriented toward conditional functional explanations.

Walls et al. (1992) propose a design theory for vigilant executive information systems. In that paper, we find an explanatory design theory that addresses the variety and evolution of structuring issues at the executive level of organizations. While this problem is approximated by a sense-and-respond requirement, it operates at a very high conceptual level, and the issues must be translated into an actionable level. This requirement explains why the design solution involves general templates that trap the issues into an explicit sense-and-respond framework that maximizes useful information about the issues for executives, and provides a cohesive track to responsible actions. The explanatory design theory can be summarized as follows (adapted from meta-requirements **Table 9**, p. 51 and from meta-design, **Table 11**, p. 54) in **Fig. 10**. This theory should provide conditional functional explanations.

Brohman et al. (2009) proposed a design theory for strategic network-based customer service systems. The requirement involves deciding which services to deliver to a customer base where each and every customer may seem to want distinctly different services. This requirement explains why, for efficiency, vendors need to find a set of services that have value to the largest possible group of customers, without driving them out of the customer network to other vendors. Accordingly, the design solution involves a process of discovering the ideal set of services for a customer network. In this paper, we find an explanatory de-

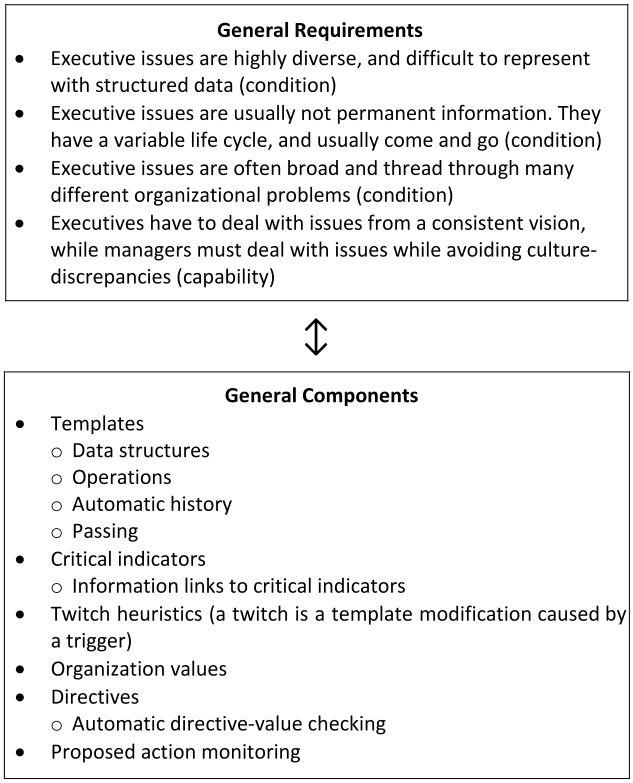

Fig. 10 A design theory for vigilant EIS (Walls et al. 1992) – as explanatory design theory

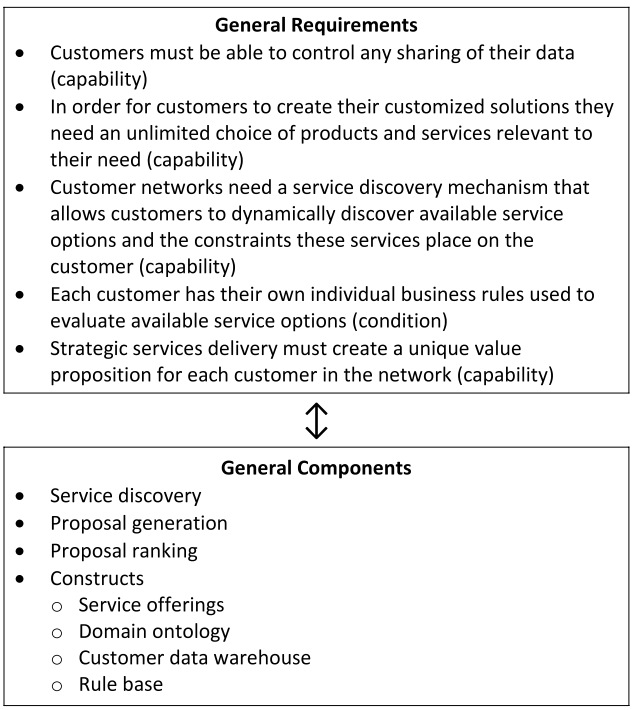

Fig. 11 A design theory for strategic network-based customer service – as explanatory design theory

sign theory that is summarized as follows in **Fig. 11**. This theory is oriented toward conditional functional explanations.

Ngai et al. (2009) propose a design theory for RFID-based healthcare management systems. The design theory formulates the requirements as a problem of information errors that endanger patient safety. These requirements explain why a general solution is needed that involves using RFID technology as the basis for patient identification, location, tracking, medication/monitoring, and drug inventory management. In this paper, we find an explanatory design theory that is summarized as shown in **Fig. 12**. This theory should provide conditional functional explanations.

Stein and Zwass (1995) discuss the need for more concise organizational memory support systems, and propose a design theory. Post-industrial organizations demand better decision making, innovation, and information acquisition/distribution. These three demands increase the usage of information and communications technology. Use of such technology leaves a trace record of organizational processes, rationale, context, outcomes, etc. The fundamental requirement is the need to organize and extract these organizational "memories" from the technology. This requirement explains why the solution needs an integrated system that manages organizational memory. In this paper, we find an explanatory design theory that is summarized as shown in **Fig. 13**. This theory is oriented toward perpetual functional explanations.

Hall et al. (2003) propose a design theory for learning-oriented knowledge management systems. Much of the literature on knowledge management systems makes explicit conceptual links to organizational learning. But implementation of knowledge management in information technology rarely goes beyond managing the organization's store of knowledge. The basic requirement lies in the organizational learning aspect – the expansion of the store – which needs better treatment. This requirement explains why we need a system that explicitly supports organizational learning that expands the organizations knowledge store. In this paper, we find an explanatory design theory that is summarized as in **Fig. 14**. This theory should provide conditional functional explanations.

Kasper (1996) proposes a design theory for user calibration in decision support systems. The requirement is the need to prevent decision-makers from miscalibrations. Decision-makers are most frequently overconfident about the quality of their decisions. Such self assessments are typically higher than objective

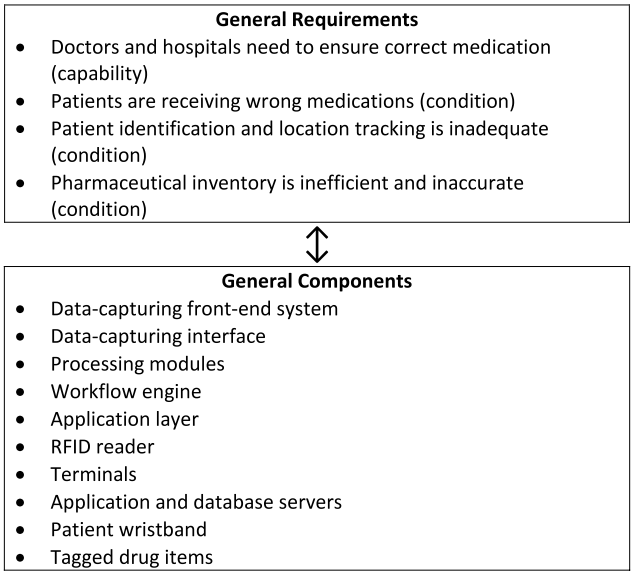

Fig. 12 A design theory for healthcare management with RFIDs – as explanatory design theory

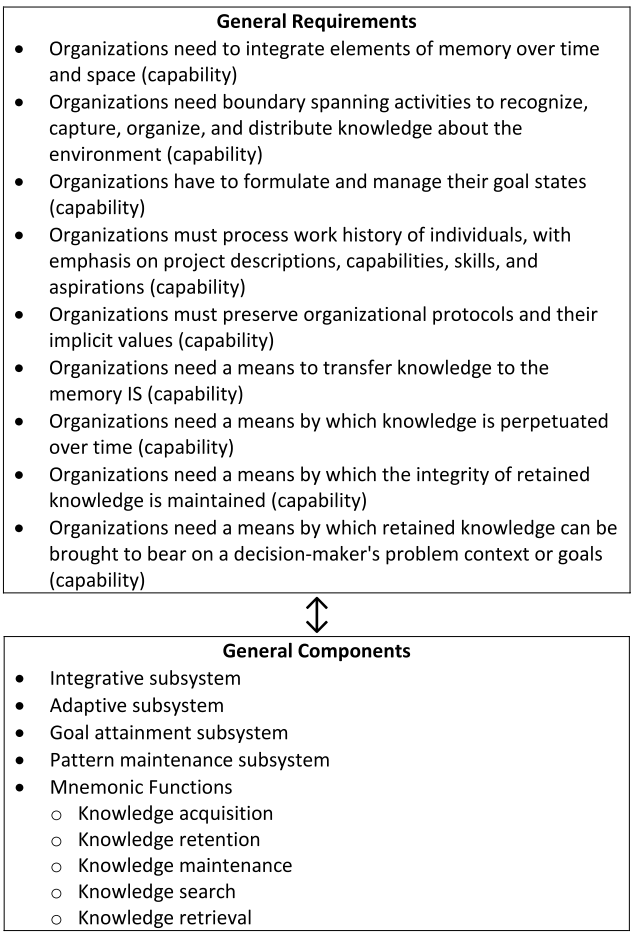

Fig. 13 A design theory for concise organizational memory support – as explanatory design theory

measures. This mis-calibration by decision makers leads to overconfidence in the chosen course of action and subsequently to disaster. This requirement explains why a decision support system has to provide true symbolic representation to enable the decision maker's mental model to be calibrated against reality. In this paper, we find an explanatory design theory that is summarized as in **Fig. 15**. This theory is oriented toward perpetual functional explanations.

5 Discussion and Conclusion

Design, design research, and design science are of growing prominence in information systems. There seems to be an important need for a more precise and practical definition of a design theory. The current notion of design theory, one that involves a dualist engagement of both design process and product, limits the acceptability of design theory as scientific theory. By separating design theory into design practice theory and explanatory design theory, we discover that design theory harbors a more fundamental, descriptive form of theory that does explain how design features achieve design requirements in a generalized form.

We have argued that an explanatory design theory provides functional or teleological explanations as opposed to positivist deductive or probabilistic explanations. This functional objective, when coupled with the constructive role of these theories, make it less relevant to expect the theory to deliver the "best" or the "most optimal" design. Instead the focus is on satisfying a need or solving a problem. A better understanding of the broad value of design theories opens the need for much further research on the topic. For example, how do we evaluate explanatory design theories? How can we ensure that the set of general requirements and general components of an explanatory design theory is complete? Is it sensible to consider evaluating the quality of a design theory, and if so, what attributes define this quality?

Based on a study of notable design writings in architecture, patterns and patlets, organizational design, portfolio theory, user interface design, and computer science, we have shown that only two elements are essentially necessary for a complete design theory. These elements embody a general design solution to a

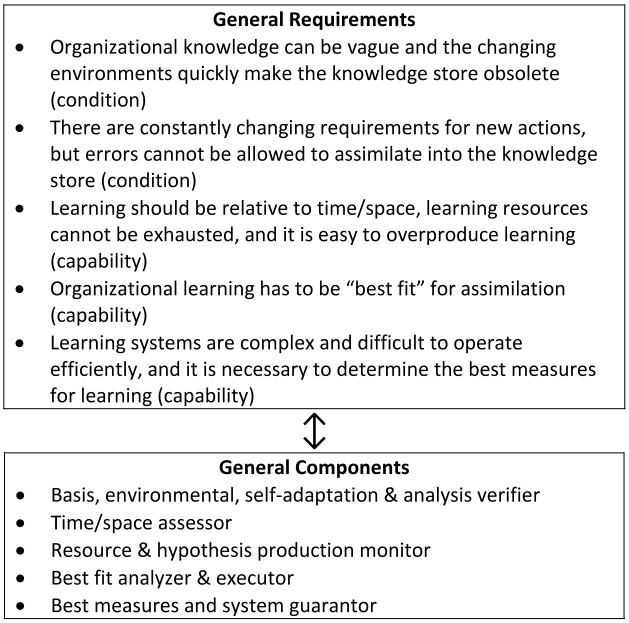

Fig. 14 A design theory for learning-oriented knowledge management systems (Hall et al. 2003)

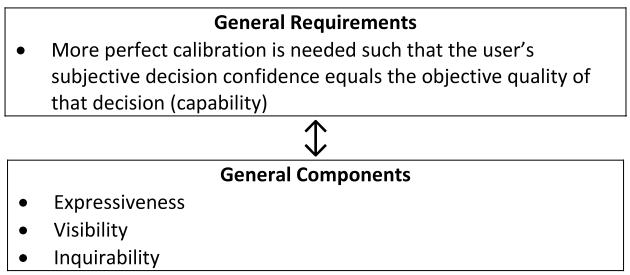

Fig. 15 A design theory for user calibration in DSS – as explanatory design theory

class of problems that relates a set of *general components* to a set of *general requirements*. In making this reduction we considered several alternatives. First we could have kept the notion of kernel theories (Walls et al. 1992, 2004). However, it is unclear exactly what kernel theories contribute, and why such separate theories should be integrated with explanatory design theories. This question has recently been discussed (Hovorka 2010) and criticized. Another reduction would involve retaining the distinction between process and product (Walls et al. 1992). But such elaborations violate Ockhams razor. We also considered including the distinction between constructs, models, methods, and instantiations (March and Smith 1995). However, this distinction does not simplify the theory, but actually complicates it further. Using such distinctions means analyzing the different types of components for the different types of requirements.

The simplified explanatory model of a design theory brings both strengths and weaknesses when compared with more elaborate models. For example, compared to Walls et al. (1992), explanatory design theory does not enforce the hypothetico-deductive model of the natural sciences. Consequently it admits softer sociological approaches to scholarly research. This feature will be seen as a weakness by some research communities and as strength by others. The complex and elaborate design theory definitions lead scholars to criticize design science that fails to demonstrate "required" elements such as testable hypotheses. From the perspective of explanatory design theory, such demonstrations add unnecessarily complex requirements for completeness. Again – to support our argument – we draw on Ockham's Razor (Lidwell et al. 2003, pp. 142–143); when given a choice between functionally equivalent designs, the simplest design is pre-

Abstract

Richard Baskerville, Jan Pries-Heje

Explanatory Design Theory

Design, design research, and design science have received increasing attention lately. This has led to a more scientific focus on design that then has made it timely to reconsider our definitions of the design theory concept. Many scholars in Information Systems assume a design theory requires a complex and elaborate structure. While this structure has appeal for its completeness and complexity, it has led scholars to criticize simplicity and elegance in design science theories that fail to demonstrate the "required" elements. Such criticisms lead to questions about whether design theory can be considered theory at all.

Based on a study of notable design writing in architecture, finance, management, cognitive psychology, computer science as well as information systems and the philosophy of science, the authors demonstrate that design theory consists of two parts: a design practice theory and an explanatory design theory. An explanatory design theory provides a functional explanation as to why a solution has certain components in terms of the requirements stated in the design. For explanatory design theory, only two elements are essentially necessary for a complete design theory: requirements and solution components. The argument is logical as well as empirical; the authors give examples of design theory drawing from IS as well as other design-related fields show how design theory can be both simple and complete. The paper concludes with a proposal for explanatory design theory.

Keywords: Design theory, Design science, Design research, Research method

ferred. Ockham's razor compels the simplest framework for design theory: explanatory design theory.

References

Alexander C (1964) Notes on the synthesis of form. Harvard University Press, Cambridge

Alexander C, Ishikawa S, Silverstein M, Jacobson M, Fiksdahl-King I, Angel S (1977) A pattern language: towns, buildings, construction. Oxford University Press, New York

Baskerville R, Myers M (2002) Information systems as a reference discipline. MIS Quarterly 26(1):1–14

Brohman M, Piccoli G, Martin P, Zulkernine F, Parasuraman A, Watson R (2009) A design theory approach to building strategic network-based customer service systems. Decision Sciences 40(3):403–430

Cole R, Purao S, Rossi M, Sein MK (2005) Being proactive: where action research meets design research. In: Avison D, Galletta D, DeGross JI (eds) Proc 26th international conference on information systems. Association for Information Systems, Las Vegas, pp 325–336

Coplien JO, Harrison NB (2005) Organizational patterns of agile software development. Pearson Prentice Hall, Upper Saddle River

Dietz JLG (2006) Enterprise ontology: theory and methodology. Springer, Heidelberg

Gamma E, Helm R, Johnson R, Vlissides J (1995) Design patterns: elements of reusable object-oriented software. Addison-Wesley, Reading

Goldkuhl G (2004) Design theories in information systems – a need for multi-grounding. JITTA: Journal of Information Technology Theory and Application 6(2):59–72

Goodman N (1955) Fact, fiction, & forecast. Harvard University Press, Cambridge

Gregor S, Jones D (2007) The anatomy of a design theory. Journal of the Association for Information Systems 8(5):312–335

Hall D, Paradice D, Courtney JF (2003) Building a theoretical foundation for a learning-oriented knowledge management system. JITTA: Journal of Information Technology Theory and Application 5(2):63–84

Hevner AR, March ST, Park J, Ram S (2004) Design science in information systems research. MIS Quarterly 28(1):75–105

Hooker JN (2004) Is design theory possible? Journal of Information Technology Theory and Application 5(2):73–82

Hovorka D (2010) Incommensurability and multi-paradigm grounding in design science research: implications for creating knowledge. In: Pries-Heje J, Venable J, Bunker D, Russo NL, DeGross JI (eds) Human benefit through the diffusion of information systems design science research. IFIP AICT, vol 318. Springer, Berlin, pp 13–27

Järvinen P (2007) Action research is similar to design science. Quality and Quantity 41(1):37–54

Kasper GM (1996) A theory of decision support system design for user calibration. Information Systems Research 7(2):215–232

Lidwell W, Holden K, Butler J (2003) Universal principles of design. Rockport Publishers, Gloucester

March ST, Smith GF (1995) Design and natural science research on information technology. Decision Support Systems 15(4):251–266

Markowitz HM (1952) Portfolio selection. The Journal of Finance 7(1):77–91

Markus ML, Majchrzak A, Gasser L (2002) A design theory for systems that support emergent knowledge processes. MIS Quarterly 26(3):179–212

Mintzberg H (1980) Structure in 5's: a synthesis of the research on organization design. Management Science 26(3):322–341

Mintzberg H (1983) Structure in fives: designing effective organizations. Prentice Hall, Englewood Cliffs

Nagel E (1961) The structure of science: problems in scientific explanation. Routledge & Kegan, London

Ngai E, Poon J, Suk F, Ng C (2009) Design of an RFID-based healthcare management system using an information system design theory. Information Systems Frontiers 11(4):405–417

Norman DA (1988) The design of everyday things, 2002nd edn. Basic Books, New York

Ockham W (1964) Philosophical writings: a selection. Translated by Boettner P. Bobbs-Merril, Indianapolis

Orlikowski WJ, Iacono CS (2001) Research commentary: desperately seeking "IT" in IT research – a call to theorizing the IT artifact. Information Systems Research 12(2):121–134

Parnas DL, Clements PC (1986) A rational design process: how and why to fake it. IEEE Transactions on Software Engineering 12(2):251–257

Simon HA (1996) The sciences of the artificial, 3rd edn. MIT Press, Cambridge

Stein EW Zwass V (1995) Actualizing organizational memory with information systems. Information Systems Research 6(2):85–117

Sutton RI, Staw BM (1995) What theory is not. Administrative Science Quarterly 40(3):371–384

van Aken JE (2004) Management research based on the paradigm of the design sciences: the quest for field-tested and grounded technological rules. The Journal of Management Studies 41(2):219–246

Walls JG, Widmeyer GR, El Sawy OA (1992) Building an information system design theory for vigilant EIS. Information Systems Research 3(1):36–59

Walls JG, Widmeyer GR, El Sawy OA (2004) Assessing information system design theory in perspective: how useful was our 1992 initial rendition? JITTA: Journal of Information Technology Theory and Application 6(2):43–58

Walmsley J (2000) The development of Lockean abstraction. British Journal for the History of Philosophy 8(3):395–418

Wehmeier S (ed) (2000) Oxford advanced learner's dictionary, 6th edn. Oxford University Press, Oxford

Weick KE (1995) What theory is not, theorizing is. Administrative Science Quarterly 40(3):385–390