Theorizing in Design Science Research

Pries-Heje, Jan; Lee, Jong Seok; Baskerville, Richard

Published in:
Lecture Notes in Computer Science

Publication date:
2011

Document Version
Peer reviewed version

Citation for published version (APA):
Pries-Heje, J., Lee, J. S., & Baskerville, R. (2011). Theorizing in Design Science Research. *Lecture Notes in Computer Science*, *LNCS*(6629), 1-16.

Theorizing in Design Science Research

Jong Seok Lee[1], Jan Pries-Heje[2], and Richard Baskerville[1]

[1] Georgia State University, CIS Department, Atlanta, GA, 30303, USA
jlee@cis.gsu.edu, baskervile@acm.org
[2] Roskilde University, Department of Communication, Business & IT, Roskilde, Denmark
janph@ruc.dk

Abstract. Theory is a central element in research. Due to the importance of theory in research, considerable efforts have been made to better understand the process of theorizing, i.e., development of a theory. A review of the literature in this area suggests that two dominant theorizing approaches are anchored to deductive and inductive reasoning respectively. In contrast, an essential part of theorizing for design may involve abductive reasoning. The purpose of design theory is not to advance declarative logic regarding truth or falseness, but to guide learning and problem solving through the conceptualization of a design artifact. This paper critically examines the process of theorizing for design by developing an idealized design theorizing framework. The framework indicates that theorizing for design operates in two distinct domains: instance and abstract. Further, four key theorizing activities are identified in this framework: abstraction, solution search, de-abstraction, and registration. The framework provides grounds for building strong design theories in the design science paradigm by explicating the underlying theorizing process for design.

Keywords: Design Theory, Theorizing in Design Science Research.

1 Introduction

Design science research holds promise as a paradigm that can establish the relevance of academic information systems (IS) research for IS practice [1]. However, unless such research develops a solid contribution to theory, the paradigm loses its importance to academia [2]. While there is substantial work that describes design science theories [3-4], less is known about the process of creating theories in design science. If design theory is indeed a particular kind of theory, it follows that design theorizing may be a particular kind of theorizing. The purpose of this paper is to describe and illustrate an idealized process for theorizing in design science. Such theorizing processes are important in design science research if the paradigm is to maintain its contribution to the IS academic tradition while simultaneously making significant advances in IS practice.

The substantiation of a strong theoretical contribution is often regarded as *prima facie* evidence of high quality in scholarly work. However, definitions of "strong" theory, not to mention theory itself are so contentious among academics that it may be easier to exclude non-theory than it is to inclusively define theory [5]. Alternatively,

H. Jain, A.P. Sinha, and P. Vitharana (Eds.): DESRIST 2011, LNCS 6629, pp. 1–16, 2011.

a focus on the quality of the process of theorizing may be more meaningful than evaluating the quality of the theory under development [6]. Thus, the issues of the quality of the theory (as a product) are intertwined with the quality of the theorizing (as a process) because theorizing is critical in producing good theories; and necessarily to high quality research that contributes substantial theories. There are elaborations of inductive theorizing [7-10], deductive theorizing [11], and richer conceptions of theorizing as messy, human behavior [12]. Different research paradigms take on different theorizing approaches that lead to different types of theory, e.g., systematic, formal, or axiomatic [13].

Theory in design science research is deemed by many authorities to be so important that a distinct class of *design theory* is widely accepted [3-4]. These theories have specific components, for example, meta-requirements, a meta-design, a design method, testable product hypotheses, testable process hypotheses, etc. Design theories will typically encompass a design process for applying a meta-design for the purposes of instantiating a designed artifact. While there is an established body of work dedicated to explaining and understanding design theory and its components, there is a need for further examination of the process of theorizing for design. In general, the literature recognizes that theory and theorizing are intertwined, suggesting a need for more attention to design theorizing. Understanding the theorizing of design is important because it should help guide design science researchers to build stronger design theories.

Weick [12] recognizes that many discussions of design theorizing, like other theorizing processes, are rational idealizations of a disciplined form of imagination. The products of theorizing (theories) are social constructions that evolve from an ideation process of concurrent trials (conjectures) and errors (refutations). It is often a variant of other sense-making processes such as generalization, prediction, and problem solving. Unlike theory testing processes, theorizing is a search for plausibility rather than validity, and selecting one theory from among other imagined constructs may be because of its interest, believability, or beauty. Theorizing is rarely mechanistic, but is often a process characterized by an *"intuitive, blind, wasteful, serendipitous, creative quality"* (p. 519).

2 Deductive, Inductive, and Abductive Theorizing

Theorizing refers to the process of constructing a theory [12]. It is often described as interim struggles in which patterns that explain the relation of one property with another are searched and proposed, and the truth of such proposed patterns is examined through experience [6, 14]. Also, theorizing may be a form of *disciplined imagination* in which concurrent trial-and-error thinking is iterated through imaginary experiments [12]. Weick [12] suggests that theorizing largely consists of three components: problem statements, thought trials, and selection criteria. It is a process that involves concurrency and iteration in each of these components. Kaplan [15] made a distinction between knowledge growth by extension and knowledge growth by intension. Knowledge growth by extension concerns exploring new areas by applying the existing knowledge in one area to adjacent areas, whereas knowledge growth by intention concerns seeking more complete knowledge that operates within

a single area. The assumption underlying these two theorizing strategies is the intellectual reasoning method; knowledge growth by extension corresponds to inductive reasoning in which new knowledge is explored, whereas knowledge growth by intention corresponds to deductive reasoning in which existing knowledge is refined and tested. Inductive and deductive reasoning have been two dominant theorizing approaches in many research disciplines.

The origin of deductive reasoning dates back to ancient philosophy; Plato denied the validity of inductive sense making from experience, and asserted that only logical deduction is a valid method for developing theory, i.e., the hypothetico-deductive method. Deductive theorizing involves deducing a conclusion from a general premise, i.e., a known theory, to a specific instance (i.e., an observation). For instance, (a) premise: failure to incorporate user requirements leads to low user satisfaction, (b) instance: a system has failed to incorporate user requirements, (c) conclusion: the users of this system have low satisfaction. At the heart of deductive reasoning is *falsification* which suggests that a theory can only be shown to be wrong, but never be proven to be right [11]. Theorists using a deductive approach deduce hypotheses from general knowledge and attempt to falsify them in a variety of settings; thus, a surviving theory is deemed to become more complete.

In contrast, Aristotle recognized inductive reasoning as a valid method for generating knowledge, proceeding from particulars to generals. Bacon later conceptualized inductive reasoning by arguing that a theory can be inductively developed through discovering essential nature of observations. Inductive theorizing involves drawing a conclusion from specific instances. For instance, (a) instance: every system that failed to incorporate user requirements has resulted in low user satisfaction, (b) conclusion: failure to incorporate user requirements leads to low user satisfaction. Inductive theorizing is recognized a valid theorizing method by modern researchers [7-9, 16].

While the literature on deductive and inductive reasoning crystallizes two contrasting ways in which researchers can approach theorizing, Weick [12] criticizes such methodical views on theorizing, claiming that the process of theorizing is depicted as *mechanistic* when in fact it is intuitive and creative thinking process. Weick further argues that theorizing should be seen as sense making in that it involves a searching process where explanatory relationships are sought in concepts observed in the real world [17-18]. Theorizing may go beyond just a mechanistic approach based on deductive or inductive reasoning when it indeed involves making sense out of a phenomenon in a complex and open system. Furthermore, the product that comes out of theorizing may not always be a singular truth, but rather a situated truth that explains the given phenomenon well enough per human's intuition and creativity.

Simon [19] associated design logic with imperative logic, contrasting this with the declarative logic that inhabits both inductive and deductive reasoning. Recognizing that imperative logic is complicated by value judgments, Simon used the term *satisficing* to refer to the fact that the optimal solution is difficult to obtain, and *"figures of merit permit comparison between designs in terms of 'better' and 'worse' but seldom provide a judgment of 'best'"* (p. 138). Neither deductive nor inductive theorizing seems to correspond with Simon's description of optimal solution, but rather a discovery process of trial-and-error searching through declarative space. This search process echoes Weick's sense-making theorizing concept.

In the management field, design thinking has been adopted as a new concept, and it refers to *"the designer's sensibility and methods to match people's needs with what is technologically feasible and what a viable business strategy can convert into customer value and market opportunity."* [20, p. 2]. Martin [21] argues that design thinking relies on abductive reasoning in which sense making of an observation occurs through drawing inference to the best explanation. Peirce [22] argues that *"abduction is, after all, nothing but guessing"* (p. 137) in that its goal is to derive a possible conclusion in terms of what can be possibly true as opposed to declarative logic whose goal is to determine a proposition to be true or false. Abductive reasoning involves drawing a possible precondition from a specific consequence. For instance, one might conclude that (b) failure to incorporate user requirements leads to low user satisfaction from the specific instance that (a) a newly developed system did not lead to high user satisfaction. Such reasoning is considered a fallacy in deductive logic (affirming the consequent), but is acceptable in abduction. Such a conclusion is an acceptable explanation in abduction because (1) it is one of many possible explanations for instance, (2) it is useful in understanding the phenomena, and (3) it can serve as a basis for solving the problem. Comparison of three reasoning approaches is shown in Table 1.

Table 1. Three Theorizing Approaches

	Deductive	**Inductive**	**Abductive**
Purpose	Declarative	Declarative	Post Hoc Ergo Propter Hoc (i.e., "after this, therefore because of this")
Operating Ground	Closed System	Open System	Open System
Logic	Deriving an explanation for a given instance from the existing body of knowledge	Inferring a general conclusion from a specific instance	Inferring satisficing explanation for a specific consequence

The abductive reasoning approach is useful for design theorizing, because the purpose of design theory is to enable search for a *satisficing* solution for a given design problem. Its purpose is not to derive a hypothesis from the existing body of knowledge and test it in a closed system (deductive theorizing); nor does it intend to infer a conclusion from an observation in an open system (inductive theorizing). Consistent with this, Gregor [23] argues that deductive reasoning alone is insufficient in addressing design problems, because for most design problems there exist a range of potential solutions rather than a single standout solution. Deductive and inductive reasoning are certainly applicable and useful for design theorizing, but abductive reasoning may be more important and more common among researchers. Quite possibly, deductive and inductive claims may often be useful as rhetorical vehicles, *post-hoc* rationalizations of messy design theorizing processes, that explain why the design theories that proceed from design science research ought to be accepted as scientifically valid [24].

3 Design Theorizing

There is a substantial body of literature concerning the definition of design theory and what constitutes a design theory. Walls et al [4] define an information system design theory (ISDT) as *"a prescriptive theory which integrates normative and descriptive theories into design paths intended to produce more effective information systems"* (p. 36). In the IS design science research community, design theories are believed to be *prescriptive, practical, basis for action, principles-based, and dualist constructs* [3, 25]. Although design theory is generally believed to be practical, it is argued by many that design theory needs to be grounded in relevant reference theories, e.g., kernel theories [4, 26]. Further, Walls et al [4] elaborate seven components that a design theory should have, including meta-requirements, meta-design, design method, kernel theories, etc. Focusing on the dualistic assumption of design theory, Baskerville and Pries-Heje [25] present a simplified view of design theory that consists of two parts: design practice theory concerning *"the theoretical component about design practice"* and explanatory design theory concerning *"the theoretical component about the design artifact"* (p. 273).

With respect to theorizing for design, Walls et al [4] implicitly discuss how a design theory can emerge by showing the relationships between the components of design theory. Also, Gregor and Jones [3] discuss the relations between different types of theory, implying that theory for design and action can by guided by other types of theory, such as theory for explaining and prediction, theory for predicting, etc. More recently, Gregor [23] presents a high-level framework along with seven principles for design theory development by drawing on distinct characteristics of design science research. While these seminal essays have significantly enhanced our understanding of design theory development, what appears to be missing is a granular understanding of design theorizing process, i.e., what are specific elements and activities involved in the design theorizing process? While we recognize that theory building is a highly creative, thought process that cannot be easily captured in an explicit manner, development of an idealized process for design theorizing can aid both design science researchers and designers in solving so called *wicked design problems* [27].

One reason as to why theorizing for design is not well understood (besides its intuitive and creative nature) may have something to do with lack of consensus on what constitutes a theoretical contribution in design science research. Motivated by this issue, Aier and Fischer [28] present a set of six criteria that can be used to evaluate progress in design theories. Further, Keuchler and Vaishnaive [29] suggest that developing a design theory is inextricably bound to refinement and extension of kernel theories, and what may emerge in this theory refinement process is in fact mid-range theory that is particularly useful for constructing information systems artifact. However, a closer examination of design theorizing process which operates within the human mind is warranted to reveal how theorizing for design actually unfolds. Further, the role that theories play may vary across different design science research projects, i.e., kernel theories, mid-range theories, *post-hoc* rationalizations of design theorizing processes, etc. Although we do not discuss this issue explicitly in this paper, development of an idealized theorizing process may lead to a more differentiated discussion of the necessity of theory and theorizing process in design science research.

4 A Design Theorizing Framework

The need for a focus on design theorizing suggests the potential value of a framework to aid understanding of how we theorize for design and what key activities are involved in developing a design theory. The purpose of the framework is not to prescribe a mechanical method that a researcher can follow to theorize for design, but rather to identify and organize the essential activities in the theorizing process (see Figure 1).

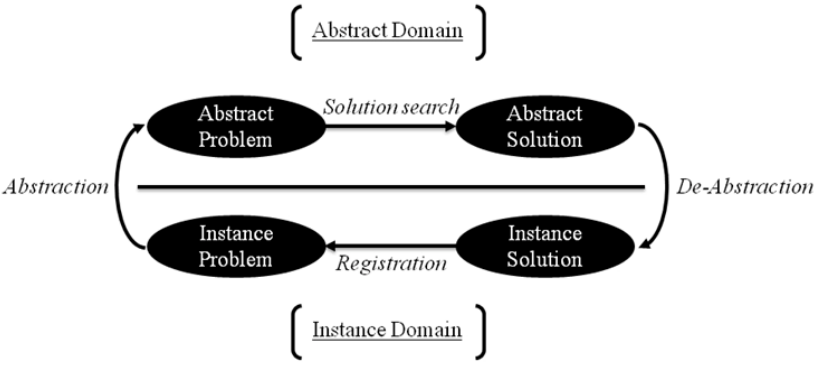

Fig. 1. Design Theorizing Framework

4.1 Theorizing Domains

A key underlying assumption of this framework is that theorizing for design operates in two distinct domains: an abstract domain and an instance domain. In the abstract domain, a solution search process occurs in which an abstract solution is searched for an abstract problem. Simon [19] uses the highway design example to illustrate that solution search process operates at the conceptual level in which specific construction plans, such as particular locations, are not specified. The abstract domain is generalized, operating at a theoretical with a "class of problems", a "class of goals", or a "class of artifacts" [4, p. 42] rather than particulars. In contrast, the instance domain refers to where an instance (particular) solution is applied to address an instance (particular) problem. Further, the two theorizing domains operate on their own independent ground; specifically, there are fewer constraints on the development of abstract solution search process. For instance, the abstract solution is not constrained to or restrained by an instance problem. Following Weick's notion of disciplined imagination, operations within the abstract domain often build on a basis that is explicit, novel, and interesting in a way that *"stands out in* [one's] *attention in contrast to the web of routinely taken-for-granted propositions"* [30, p. 311]. In contrast, an instance domain may not be as novel, and interesting as an abstract domain; most interesting theoretical insights are discovered when researchers think independently from their observations/data [6, 31-32] .

These two theorizing domains highlight the notable duality in design theories. Most pronunciations of design theory elements demonstrate two fundamental parts: a design practice theory and an explanatory design theory [25]. This duality can be drawn from Simon's original work [19] and is very clearly represented in the Walls et al. framing of design theory [4] to include such elements as design methods, meta-requirements, and meta-designs. In the transition from the abstract domain to the instance domain we can locate design practice theory and design methods. Design practice theory mainly concerns bringing a proposed design artifact to life; it emerges by moving from abstract domain to instance domain, i.e., development of an instance solution based on an abstract solution. Design methods operate similarly. In the abstract domain, we can locate explanatory design theory, meta-requirements, and meta-designs. Explanatory design theory concerns *"principles that relate requirements to an incomplete description of an object"* [25, p.273]. Therefore, explanatory design theory emerges out of abstract solution search process where a search for the right set of command variables takes place, and abstract requirements of a design artifact are identified [19]. Walls et al. [4] used the term 'meta-requirements' to show how the functional requirements and basic features that constitute a class of design artifacts are abstract.

4.2 Theorizing Activities

There are four activities in the theorizing framework, each represented by an arrow in Figure 1. These activities are abstraction, solution search, de-abstraction, and registration. Given that all four of these activities may take place as human thought, it may be possible that these occur not cyclically (as represented in Figure 1), or in the order implied by the arrows, but perhaps may arise simultaneously. In terms of activities, please recognize that the framework is an idealization to aid in understanding and comprehending what can be involved in design theorizing. Each activity is described below.

Abstraction. A theory is said to have generalizability when it is applicable across different settings that go beyond a specific setting in which it was tested [33-34]. Generalizability of a theory is a concern to most theorists, as theories that fail to produce generalized inferences are not considered a strong theory, or not a theory at all [5-6, 30]. Design theory is no exception. A strong design theory should show applicability across widely different settings, and address a broad class of design problems. Theory is said to arise from identifying the key links between data and prescriptions (or propositions) by discarding detailed information, and the abstraction is a process of deriving key concepts observed in a specific instance [6]. In design theorizing, abstraction can be realized when a researcher derives common concepts or ideas from an instance problem by removing details pertaining to the context of the instance problem; by doing so, a broad set of problems can be identified. This process of abstraction essentially involves *reflective judgment* where unknown universals for given particulars are sought [35]. When people recognize a problem which cannot be solved intuitively, they rely on their cognitive faculties to distinguish between the peculiarities and the essential conditions for the problem [35-37]. During this process, reflective judgments are called for to understand the problem at a more

universal level. Most design problems cannot be solved intuitively or with certainty. It is this uncertain nature of design problems that calls for reflective judgment to decide which essential conditions are applicable to a broader class of problems than just the one at hand. For example, a system user may express his or her frustration for being unable to locate documents effectively in a knowledge management system (an instance problem). Abstraction can be achieved by extracting the key concepts related in the problem, such as user frustration and systems search functionality (an abstract problem).

Solution search. Simon [19] describes how the solution search process (i.e., a goal-seeking system) communicates with the outside environment through two channels: the afferent (the world of the senses) and the efferent (the motor world). The afferent is a sensory world where outside environment is perceived by regarding its state, and the efferent is the world where actions are taken. These two worlds operate at the abstract level; the problem environment is recognized through stored memory information in the human mind, and any particular actions are imaginary [19]. The attainability of goals is determined by making associations between elements of the imagined environment with the elements of the imagined actions. These associations are *"between particular changes in states of the world and particular actions that will bring these changes about"* (p. 141). Thus, theorizing for the solution search concerns understanding relationships between the afferent and efferent, and how the afferent responds to the changes made by actions in the efferent. This process is highly iterative, and requires searching for the right set of actions in the efferent (e.g., creating components of a design artifact) that will bring sufficient changes in the afferent (e.g., solving the requirements of the problem). Thus, an important element in this process is theorizing for the generalized components and the generalized requirements of a design artifact. Each component of the imagined artifacts (the efferent) would have to be theorized individually and collectively in the context of the afferent. The functional explanation of the imagined design artifacts proceeds from this theorizing process, i.e., an explanatory design theory [25].

De-Abstraction. During the solution search, proposed solutions or design artifacts may be theoretical, abstract concepts; these are imagined, generalized problems and solutions. Thus, in order for these to be tested in a specific setting, the generalized, abstract concepts need to be narrowed and instantiated for a particularized setting and a particularized artifact. This de-abstraction involves adding details pertaining to a specific context in which the solution will be applied, and all the details of the instance solution become articulated. De-abstraction essentially requires *deterministic judgment* in which we can subsume given particulars under known universals [35]. De-abstraction is a realization process that may still be partly imaginary; potentially a thought experiment within a design theory is tried as the basis for an imaginary artifact within the designer's mind. Obviously, it may also become partly (or wholly) materialized as an instantiated artifact in reality.

Registration. Whether the design artifact resulting from the de-abstraction process is imaginary or material, the design has to further try this outcome against an instance of the problem setting to verify that the instance outcome has potential to serve the needs of an instance of the problem. Like the de-abstraction outcome, this problem instance

may also be wholly imaginary or more-or-less material. Consequently this "trial" may be more-or-less a thought experiment or more-or-less material and empirical. Registering the theory means trying an instance of the solution against an instance of the problem, and adjusting the theory to more exactly correspond to the requirements of the instance. Such adjustments can lead to further abstraction activity as theory adjustments may have more general effects. Because the registration activity is part of the theorizing process, it may be (or may not be) independent of the design science research evaluation process. Evaluation, in design science research, is usually regarded as a validation or proof process where a design theory is shown empirically to stand in terms of how well a resulting artifact performs or to what degree it works as intended [38-40]. While conceptually the evaluation ought to occur after the theorizing process has matured to completion, it could be regarded as a final registration activity in which it is shown empirically that no further adjustment is required.

4.3 Theorizing Threshold

There is no universal starting point in the design theorizing framework. We believe that theorizing begins when some stimulation threshold is exceeded that drives the processes of disciplined imagination and abduction to start one or more of the design theorizing activities. Intuitively, we might think that theorizing always commences with a recognition of an instance problem, and proceeds in the following order; identification of an abstract problem, development of an abstract solution, particularizing an instance of this solution, and registering it to the originating instance problem (this order is indicated by the arrows in the framework). This order would reflect the ideal essence of design science research whose aim is to achieve a clearly stated goal, i.e., bring an intended change to the real world through creation of a new design theory and its resulting artifact [4, 41]. However, reflecting on our own research experience indicates that such an origin for these activities and such an order may be idealizations. It is not the only way that theorizing can take place. As a human sense-making process, theorizing can be messy.

We returned to two of our own design science research projects and reflected on the theorizing process that emerged in these cases. We chose cases that appear in published research to enable interested readers to examine the process and the results of the theorizing more carefully. We selected one case in which the design theorizing threshold was first crossed in the instance domain, and one case in which the design theorizing threshold was crossed in the abstract domain.

Case 1: Crossing the threshold in the instance domain and theorizing the design theory nexus. This work developed a method for constructing decision systems along with instantiations for organizational change decision-making and user involvement decision-making. The theory in this work centered on a conceptual structure called a design theory nexus as a means for addressing the "wicked problem" of multi-criteria decision-making. The instantiations included an IT artifact based on spreadsheet software, used empirically in organizations to help decision makers determine what organizational change approach or what user involvement approach to adopt. A subjective evaluation of the artifacts by participants was positive in terms of their satisfaction in use, and their intention to adopt the outcome artifact results.

In the projects associated with this theory, the theorizing threshold was first crossed in the instance domain, when researchers began conducting search conferences (a form of action research) in an organization to unearth a possible organizational change approach. The results led to a taxonomy of change methods (reported in [42]), which was still very much in the instance domain. Theorizing moved to an abstract domain when the problem was generalized into a form later recognized as multi-criteria decision making (an abstract problem), and the concept of a theory nexus [43] adapted as an abstract solution. The theorizing process returned to the instance domain where it was registered against an organizational change instance. The process went through further abstraction-and-instance cycles that included development of the instance solution for selecting a user involvement approach in IT-developing projects. The theorizing result of these further cycles clarified the aspects of the theory in spanning both single-criterion and multi-criteria decision settings (the user involvement approach proved to be single-criterion - for further case details see [27]).

Case 2: Crossing the threshold in the abstract domain and theorizing a software process improvement design theory. This work developed a design theory that generalized alternative, competing models for improving software organizations. The theory proposed a universal 4-stage model that explained how software process improvement generally progressed in organizations. Each stage was elaborated with a conceptual model of its possible elements. These models were then instantiated with examples (specimens) of published software process and organizational improvement methods such as Six Sigma, CMMI, Balanced Scorecard etceteras. The validation was anchored to the evidence used to instantiate the published examples within the models.

Like case 1, this work developed an initial framework for comparing and contrasting alternative models for improving software organizations. However, this theorizing threshold was crossed initially in the abstract domain as a search for universals in software process improvement. There were no software improvement instances driving this search, just a scholarly curiosity (details of the initial framework are reported in [44]). Unlike case 1, both the abstract and the instances were thought processes; more imaginary than empirical. The instances drawn into the cycles of theorizing were published methods and frameworks. While real to a certain extent, these instances were registered completely through conceptual argumentation rather than field experiments. As the process went through further abstraction-and-instance cycles, the theory was reframed within a body of technological rules as well as the process models (this design theory is elaborated in [45]).

4.4 Discussion

In Case 1, the problem instance and its immediate solution were developed first as a more-or-less un-theorized design, a classification of major organizational change approaches according to their central feature. Such designs have been described as the result of pre-theory in research [46]. In Case 1, the abstract problem and abstract solution were identified later in the research. Our theorizing for Case 1 occurred in the following order: instance problem – instance solution – abstract problem – abstract solution. In contrast, in Case 2 theorizing began with recognition of an

abstract problem; that is, a problem was recognized in a researchers' mind at the abstract level (a set of personal/professional experiences may have led to problem recognition at the abstract level, but an instance problem was not yet clearly defined). Our theorizing for Case 2 occurred in the following order: abstract problem – abstract solution – instance solution – instance problem. The fact that theorizing began with identification of an abstract problem independently from any instance problem supports our proposition that theorizing operates in two distinct domains. In Case 2, the search for an abstract solution began without any constraints imposed by an initiating recognition of an instance problem. Importantly in this case, de-abstraction played more significant role than abstraction, as an abstract solution was applied to develop an instance solution and an abstract problem was applied to identify an instance problem.

The two cases are summarized in Table 2. In both cases it is difficult to find any explicit observations that unveil the abstraction process. This is consistent with its nature as a cognitive process of reflective judgment. More detailed research notes would be required; something akin to a personal diary would be necessary to explicate the abstraction activities in these cases. Still these experiences indicate support for the proposition of two distinct theorizing domains, and they show how theorizing can begin at multiple points in the theorizing framework.

Table 2. Examples of Theorizing Thresholds

	Case 1 Organizational Change Nexus	**Case 1 User involvement Nexus**	**Case 2 Software Process Improvement Framework**	**Case 2 Design Theory for Software Process Management**
Abstract problem	Difficulty of choosing among many organizational change approaches	How and when to have user participation in an IT project	What software process improvement approach to use out of many different	Difficulty of designing quality management policies
Abstract solution	Ten generalized change strategies and a way to choose among them	Set of methods and techniques for deciding user participation in IT project	Framework for comparing and contrasting normative models for improving software organizations	Design rules for quality management
Instance problem	Design of organizational change initiatives in two companies	User participation in IT project management in ten companies	Recommending software process improvement approach in a concrete organization	Myriad available quality management fragments

Table 2. (*continued*)

Instance solution	Calculating fit between organizations and change strategies based on query	Applying technological rules in a field study in ten companies	An instantiation strategy of the framework	Design rules based on stages of process improvement models
Abstraction	Problem generalized into a form recognized as multi-criteria decision making	A second iteration realized that this instance could also be abstracted as multi-criteria	Not observed (because this case began with an abstract problem)	From simple rule of thumb in framework to technological rules forming a design theory
Solution search	Realized that it was a form of the concept of a theory nexus	Realized that a theory nexus would be an abstract solu-tion	Identifying 4 universal stages when selecting improvement model	Aligning the technological rules and realizing that the first stage of the four was dominant
De-abstraction	Developed into a spreadsheet and an intervention with manage-ment in an organization	Originally developed into specific decision making tool	Made into course material and taught in several professional courses	Made into course material for Graduate & Executive Master level teaching
Registration	Tested in two companies – and later in many	Evaluated with in 10 companies with many project managers	Applied by participants in professional courses	Applied by MPF-participants
Reference	[42]	[27]	[44]	[47]

This empirical evaluation using the past research projects reveals a need for future research. While we believe that the framework proposed in this paper provides a solid conceptualization of design theorizing, further empirical investigation is warranted to critically evaluate and improve the proposed framework. Given that theorizing is a thought process, and can be messy, methods that enable a close examination of human thinking, such as protocol analysis [48-49] and thought experiment [50], may provide a useful means to evaluate and improve the proposed framework. Further, a more systematic analysis of design science research publications is warranted to exploit the different roles that theory play in design science research, and to assess how the proposed framework for theorizing process can be applied and adapted in different cases.

5 Conclusion

In this paper we described and illustrated an idealized theorizing process for design theories. In this process design theorizing operates across two distinct domains; the instance domain encompasses an instantiated solution that is applied to solve an instantiated problem, and the abstract domain encompasses an abstract solution that is devised to solve an abstract problem. We identified four theorizing activities in this process and discussed the role of each activity in developing a design theory. Three conclusions emerge as our contribution.

First, design theorizing necessitates making connections between an abstract domain and an instance domain: abstraction and de-abstraction. Abstraction concerns *reflective judgment* where search for unknown universals for given particulars takes place, whereas de-abstraction concerns *deterministic judgment* where given particulars are subsumed under known universals. Through the process of de-abstraction, design practice theory that involves the instantiation of a proposed design artifact may emerge. Any particular stand-alone design solution that lacks connections with an abstract class of design solutions (and/or a class of design problems) is incomplete as a theory. Particular design solutions may be close to design theories, but the level of abstraction needs to be raised to a class basis that is explicit, novel, and interesting.

Second, a review of the literature on theorizing reveals two dominant theorizing approaches (deductive and inductive) that have been adopted in different research paradigms. While these two theorizing approaches are a useful reasoning tool for theory development, theorizing for design often necessitates adoption of a line of reasoning that is essential for problem solving, i.e., abductive reasoning. Theorizing in design science is abductive because it seeks an imperative logic (rather than declarative) in order to address design problem through the conceptualization of a design artifact. This theorizing process provides a good example of disciplined imagination involving intuitive and creative thinking processes. The adoption of abductive reasoning for design theorizing enables the search for a *satisficing* solution for a given design problem. Further, through the activity of abstract solution search, functional explanations (explanatory design theory) that identification the reasons for meta-requirements result.

Third, reflections on the authors' own prior design science research projects reveal that there is no universal starting point with which design theorizing commences; any origin or ordering in theorizing activities indicated in Figure 1 would be an idealization. A review of two cases shows how the theorizing threshold can be first crossed in either the instance domain or the abstract domain.

Theory is an important and central element in research. Different research paradigms take on different approaches for theorizing. If design theory is a particular kind of theory, it follows that design theorizing may be a particular kind of theorizing. An essential part of design theorizing may involve abductive reasoning because there is a purpose aimed at guiding learning and problem solving. The framework proposed in this paper is aimed toward building strong design theories in the design science paradigm by providing an idealized theorizing process for design science research.

References

1. Winter, R.: Design science research in Europe. European Journal of Information Systems 17, 470–475 (2008)
2. Baskerville, R., Lyytinen, K., Sambamurthy, V., Straub, D.: A response to the design-oriented information systems research memorandum. European Journal of Information Systems 20, 11–15 (2011)
3. Gregor, S., Jones, D.: The anatomy of a design theory. Journal of the Association for Information Systems 8, 312–335 (2007)
4. Walls, J.G., Widmeyer, G.R., El Sawy, O.A.: Building an Information System Design Theory for Vigilant EIS. Information Systems Research 3, 36–59 (1992)
5. Sutton, R.I., Staw, B.M.: What Theory is Not. Administrative Science Quarterly 40, 371–384 (1995)
6. Weick, K.E.: What Theory is Not, Theorizing Is. Administrative Science Quarterly 40, 385–390 (1995)
7. Dubin, R.: Theory Building in Applied Areas. In: Dunnette, M.D. (ed.) Handbook of Industrial and Organizational Psychology. Rand McNally, Chicago (1976)
8. Locke, E.A.: The Case for Inductive Theory Building. Journal of Management 33, 867–890 (2007)
9. Eisenhardt, K.M.: Building Theories from Case Study Research. Academy of Management Review 14, 532–550 (1989)
10. Glaser, B.G., Strauss, A.: The Discovery of Grounded Theory: Strategies for Qualitative Research. Wiedenfeld and Nicholson, London (1967)
11. Popper, K.: The Logic of Scientific Discovery. Unwin Hyman, London (1980)
12. Weick, K.E.: Theory Construction As Disciplined Imagination. Academy of Management. The Academy of Management Review 14, 516–531 (1989)
13. Freese, L.: Formal Theorizing. Annual Review of Sociology 6, 187–212 (1980)
14. Homans, G.C.: Contemporary Theory in Sociology. In: Faris, R.E.L. (ed.) Handbook of Modern Sociology, pp. 951–977. Rand McNally, Chicago (1964)
15. Kaplan, A.: The conduct of inquiry. Chandler, San Francisco (1964)
16. Alvesson, M., Karreman, D.A.N.: Constructing Mystery: Empirical Matters In Theory Development. Academy of Management Review 32, 1265–1281 (2007)
17. Gergen, K.J.: Correspondence versus autonomy in the language of understanding human action. In: Fiske, D.W., Shweder, R.A. (eds.) Metatheory in Social Science, pp. 136–162. University of Chicago Press, Chicago (1986)
18. Henshel, R.L.: Sociology and prediction. The American Sociologist 6, 213–220 (1971)
19. Simon, H.: The Sciences of the Artificial. MIT Press, Cambridge (1996)
20. Brown, T.: Design Thinking. Harvard Business Review, 1–10 (June 2008)
21. Martin, R.: The Design of Business. Harvard Business Press, Boston (2009)
22. Peirce, C.S.: The Logic of Drawing History from Ancient Documents. In: Burks, A.W. (ed.) Collected Papers of Charles Sanders Peirce, Volumes VII and VIII: Science and Philosophy and Reviews, Correspondence and Bibliography, vol. 7. Harvard University Press, Cambridge (1958)
23. Gregor, S.: Building Theory in the Sciences of the Artificial. In: Vaishnavi, V., Purao, S. (eds.) Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology. ACM, New York (2009)
24. Parnas, D., Clements, P.: A rational design process: How and why to fake it. IEEE Transactions on Software Engineering SE 12, 251–257 (1986)

25. Baskerville, R., Pries-Heje, J.: Explanatory Design Theory. Business & Information Systems Engineering 2, 271–282 (2010)
26. Goldkuhl, G.: Design Theories in Information Systems - a need for multi-grounding. Journal of Information Technology Theory and Application 6, 59–72 (2004)
27. Pries-Heje, J., Baskerville, R.: The Design Theory Nexus. MIS Quarterly 32, 731–755 (2008)
28. Aier, S., Fischer, C.: Criteria of progress for information systems design theories. Information Systems and E-Business Management 9, 133–172 (2011)
29. Kuechler, B., Vaishnavi, V.: On the theory development in design science research: Anatomy of a research project. European Journal of Information Systems 17, 489–504 (2008)
30. Davis, M.S.: That's Interesting!: Towards a Phenomenology of Sociology and a Sociology of Phenomenology. Philosophy of the Social Sciences 1, 309–344 (1971)
31. Bacharach, S.B.: Organizational theories. Academy of Management Review 14, 496–515 (1989)
32. Root-Bernstein, R.S.: Discovering: Inventing and Solving Problems at the Frontiers of Scientific Knowledge. Harvard University Press, Cambridge (1989)
33. Lee, A.S., Baskerville, R.L.: Generalizing Generalizability in Information Systems Research. Information Systems Research 14, 221–243 (2003)
34. Yin, R.: Case Study Research: Design and Methods. Sage Publications, Thousand Oaks (1994)
35. Kant, I.: Critique of the Power of Judgment. Cambridge University Press, Cambridge (2000)
36. Dewey, J.: How We Think: A Restatement of the Relation of Reflective Thinking to the Educative Process. Heath, Lexington (1933)
37. Dewey, J.: Logic: The Theory of Inquiry. Holt, Rinehart, & Winston, Troy, MO (1938)
38. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Quarterly 28, 75–105 (2004)
39. March, S.T., Smith, G.F.: Design and natural science research on information technology. Decision Support Systems 15, 251–266 (1995)
40. Vaishnavi, V.K., Kuechler, W.: Research patterns: Improving and innovating information & communication technology. Auerbach Publications, New York (2007)
41. Gregor, S.: The nature of theory in information systems. MIS Quarterly 30, 611–642 (2006)
42. Pries-Heje, J., Vinter, O.: A Framework for Selecting Change Strategies in IT Organizations. In: Münch, J., Vierimaa, M. (eds.) PROFES 2006. LNCS, vol. 4034, pp. 408–414. Springer, Heidelberg (2006)
43. Carroll, J.M., Kellogg, W.A.: Artifact as theory-nexus: hermeneutics meets theory-based design. In: ACM SIGCHI Bulletin–Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Wings for the Mind, vol. 20, pp. 7–14 (1989)
44. Pries-Heje, J., Baskerville, R.: Improving Software Organizations: An Analysis of Diverse Normative Models. In: Messnarz, R. (ed.) Proceedings of European Software Process Improvement 2003, pp. X.23–X.39. Verlag der Teknischen Universität Graz, Graz (2003)
45. Pries-Heje, J., Baskerville, R.: A design theory for managing software process improvement. In: Vaishanvi, V., Purao, S. (eds.) Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology, pp. 1–2. ACM, New York (2009)
46. Aaen, I.: Essence: facilitating software innovation. European Journal of Information Systems 17, 543–553 (2008)

47. Baskerville, R.L., Pries-Heje, J.: Design and management. In: Simonsen, J., Bærenholdt, J.O., Büscher, M., Scheuer, J.D. (eds.) Design Research: Synergies from Interdisciplinary Perspectives, pp. 63–78. Routledge, London (2010)
48. Isenberg, D.J.: Thinking and Managing: A Verbal Protocol Analysis of Managerial Problem Solving. The Academy of Management Journal 29, 599–775 (1986)
49. Ericsson, K.A., Simon, H.A.: Protocol Analysis: Verbal Reports as Data. The MIT Press, Cambridge (1993)
50. McAllister, J.W.: The evidential significance of thought experiment in science. Studies In History and Philosophy of Science Part A 27, 233–250 (1995)