Proposal for Components of Method Design Theories

Increasing the Utility of Method Design Artefacts

DOI 10.1007/s12599-010-0120-x

The Authors

Dr. Philipp Offermann (✉)
Dipl.-Inform. Sören Blom
Dr. Udo Bub
Laboratories
Deutsche Telekom AG
Ernst-Reuter-Platz 7
10587 Berlin
Germany
philipp.offermann@telekom.de
soeren.blom@telekom.de
udo.bub@telekom.de

Dipl.-Ing. Olga Levina
Fachgebiet Systemanalyse und EDV
Technische Universität Berlin
Franklinstr. 28/29
10587 Berlin
Germany
olga.levina@sysedv.tu-berlin.de

Received: 2009-12-01
Accepted: 2010-07-03
Accepted after three revisions by Prof. Dr. Winter.
Published online: 2010-08-31

This article is also available in German in print and via http://www.wirtschaftsinformatik.de: Offermann P, Blom S, Levina O, Bub U (2010) Vorschlag für Komponenten von Methodendesigntheorien. Steigerung der Nutzbarkeit von Methodendesignartefakten. WIRTSCHAFTSINFORMATIK. doi: 10.1007/s11576-010-0239-x.

1 Introduction

The design of information systems (IS) artefacts is generally accepted as research in IS (Gregor 2006; Hevner et al. 2004; March and Smith 1995; Offermann et al. 2009) and the German counterpart "Wirtschaftsinformatik" (Gericke 2008; Wilde and Hess 2007). However, it is still unclear how design artefacts can be presented in a generalised form that provides a valuable contribution to the body of knowledge. In the scientific discipline, a body of knowledge together with conjectures, models and frameworks is encompassed by the word "theory" (Gregor 2006, p. 614). The trustworthiness of theories can be assessed through certain criteria, which differ depending on the epistemological point of view (Guba 1981). The fundamental question for this article is what the relationship is between design artefacts and their theories; and how design knowledge can be represented as theories in a way that facilitates evaluation (Gregor and Jones 2007; Walls et al. 1992).

The answers to these questions are relevant as in the current non-theory state researchers are confronted with several problems: Design artefacts are not comparable, scientific progress cannot be measured clearly (Aier and Fischer 2009a, 2009b), the purpose of design is not made explicit, and it remains unclear how the output can be and/or was evaluated. Problems are not limited to research as some problems also apply to industry: it remains unclear under which circumstances a design artefact can be used and which artefact yields the highest utility for a given problem, reducing acceptance.

Gregor and Jones (2007), building on Walls et al. (1992), proposed an anatomy of design theory. They realised the proposal of Walls et al. (2004, p. 56) to augment the structure of the original Walls et al. (1992) publication while remaining on the same level of abstraction. We believe that a specialisation of the design theory structure for each output type can render the structure even more usable, thereby further augmenting it. According to March and Smith (1995) there are four design theory output types: constructs, models, methods and instantiations. We have been and are still involved in several research projects addressing method engineering. From this background, in this methodological discussion we propose a refined design theory method structure. We do not focus on method engineering as in how to create methods or method theories, but instead suggest building blocks that give specific guidance on what to include and how to describe method design theories. By doing so we hope to address some of the aforementioned problems in a way that is applicable for researchers and practitioners.

The article is structured as follows. First, the nature of design science research output in general is discussed and the specialisation in form of methods introduced. Then, general components of design theory as outlined by Gregor and Jones (2007) are introduced. In Sect. 4, a specialisation of the components for methods and evaluation criteria are derived and examples presented. In Sect. 5, findings are discussed. Finally, a conclusion is drawn.

2 Methods as Design Artefacts

Design science is about designing artefacts related to IS. However, there is a discussion about what design science research output actually is (Alter 2002; Iivari 2002).

2.1 What is Design Science Research Output?

Vaishnavi and Kuechler (2007) identify the two perspectives on the design science research output from March and Smith (1995) and from Purao (2002). Hevner et al. (2004, p. 77) follow March and Smith (1995, p. 256) in that "IT artifacts are broadly defined as constructs (vocabulary and symbols), models (abstractions and representations), methods (algorithms and practises), and instantiations (implemented and prototype systems)".

Our understanding of the design science research output is based on Walls et al. (1992). Generally, an artefact design focuses both on some structure and on a

method to create something according to the structure. When focusing on method design, usually, the structure is taken as state-of-the-art. When evaluating a newly developed method, one would not attempt to prove the requirements resulting from the structure. Rather, one would show that the requirements are realised by using the method, taking the utility of the requirements for granted.

2.2 Method Design

We think that a design theory structure is more useful to research and practise if it is specialised for each aspect. In this paper, we present such a specialisation for methods.

According to Braun et al. (2005, p. 1296), a "method constitutes the basis for engineering-based procedure". They identify goal orientation, systematic approach, principles and repeatability as the fundamental defining attributes. They identify specification document, meta model, role, technique, activity/procedure model and tool as constituting elements (Braun et al. 2005). The Object Management Group (OMG) has standardised a software & systems process engineering meta-model (SPEM) (OMG 2008).

"Method engineering is the engineering discipline to design, construct and adapt methods, techniques and tools for the development of information systems" (Brinkkemper 1996, p. 276). Method specification has been discussed e.g. by Greiffenberg (2004) and Eberlein and Jiang (2007). As methods have to be adapted to a context, most authors propose situational method engineering. There is usually a differentiation between method adaptation and method composition (Bucher et al. 2007). In adaptation, a generic method is adapted to project specific needs (Karlsson and Ågerfalk 2004; ter Hofstede and Verhoef 1997). This process is also called tailoring (Basili and Rombach 1987; Fitzgerald et al. 2000). In composition, method fragments, sometimes called chunks, are composed according to needs (Brinkkember et al. 1999; Plihon and Rolland 1997; Ralyté et al. 2003).

After designing a new method it should be evaluated (March and Smith 1995; Offermann et al. 2009; Peffers et al. 2008). Several method evaluation techniques have been proposed. (Kitchenham et al. 1995; Siau and Rossi 1998; Song and Osterweil 1992). Moody (2003) proposes a method evaluation model, based on the Technology Acceptance Model. According to Goldkuhl (2004), multi-grounding should be used, combining different evaluation techniques.

3 Components of Design Theory

Before we refine the components of a method design theory, we want to give an overview of our understanding of the term theory and what the current state of discourse regarding design theory in IS research is.

3.1 Theories in Information Systems

The term theory in IS research has been discussed, amongst others, by Cushing (1990), Davis et al. (1989), Walsham (1995). We follow Gregor (2006, 2009), who reviewed theory approaches in IS research and the related disciplines and offers the following theory definition: "Theory is [...] a generalised body of knowledge, with a set of connected statements expressing general relationships among constructs that refer to entities of different types, both real-world and theoretical" (Gregor 2009). This view is based on Walls et al. (1992) who state that the "purpose of a theory is prediction and/or explanation of a phenomenon" (Walls et al. 1992, p. 38), which in turn draw from Dubin (1978). Both Walls et al. (1992) and Gregor (2006) accept this view of theory which is grounded in social sciences without discussing whether it can be directly mapped to design science. Furthermore, Gregor (2002, 2006) identifies five types of theory in IS research: analysis, explanation, prediction, explanation and prediction, design and action.

3.2 Design Theories

Information Systems design theories (ISDT) are theories for design and action (Gregor 2006; Jones et al. 2003). Venable (2006) supports a broad view on a design theory, stating that an appropriate form of a design theory is a so-called utility theory that "makes an assertion that a particular type or class of technology [...] has [...] utility [...] in solving or improving a problematic situation". There have been many proposals on design theory components, e.g. Iivari et al. (2000–2001), Markus et al. (2002). Based on Gregor (2006, p. 620), Gregor and Jones (2007) identified eight components of a design theory for IS research, which are the basis for our considerations in this paper.

Any artefact is designed with an intention of use in mind. The *purpose and scope* component captures this intention. The purpose describes "what the system is for, the set of meta-requirements or goals that specify the artefact" (Gregor and Jones 2007, p. 322). The scope defines the range of situations in which the artefact can be used as intended in order to achieve the stated goals. This component allows categorising, comparison and extension of the theory under analysis.

Constructs are the building block of any theory. Gregor and Jones (2007, p. 325) call them "the entities of interest in the theory", but do not describe the possible sources or range of the constructs. Constructs can be "represented by words, [...] but mathematical symbols or parts of a diagram can also be used" (Gregor and Jones 2007, p. 325).

Principles of form and function are defined as "principles that define the structure, organisation, and functioning of the design product or design method" (Gregor and Jones 2007, p. 325). It is the "abstract 'blueprint' or architecture" of the artefact (Gregor and Jones 2007, p. 322). When applied to a design method, the "shape and features" (Gregor and Jones 2007, p. 326) of the method represent its principles of form and function.

Gregor and Jones (2007) make the case that any designed artefact is subject to evolution over time and that the designer might foresee a certain bandwidth of change during designing it, which they refer to as "*artifact mutability*". It is as "[t]he changes in state of the artifact anticipated in the theory, that is, what degree of artifact change is encompassed by the theory" (Gregor and Jones 2007, p. 322).

Testable propositions are statements about the theory that can be proven or falsified. The formulation of the propositions can be defined rather broadly (Gregor and Jones 2007, p. 327).

Justificatory knowledge is related to the "kernel theories" of Walls et al. (1992) and "micro-theories" of Simon (1981). It is knowledge that connects "goals, shapes, processes and materials of the design theory" (Gregor and Jones 2007, p. 327). The authors state that natural, social, as well as design science type of theories are possible justificatory knowledge (Gregor and Jones 2007, p. 327).

Principles of implementation are seen by Gregor and Jones (2007, p. 323) as an optional component of a design theory. They relate to the process and the means by which the design is realised (Gregor and Jones 2007, p. 328).

Expository instantiation contains a realistic implementation, i.e. a physical realisation of the design (Gregor and Jones 2007, p. 329). This optional component is introduced to allow the exposition or representation of a theory, as the artefact itself already has some representational power.

3.3 Criteria for Theory Evaluation

Criteria for evaluating theories depend on the epistemology of the research paradigm. For positivism, validity, reliability, generalisability and objectivity are used, for interpretivism credibility, dependability, transferability and confirmability are used respectively (Guba 1981; Travis 1999). For design science research, no consensus exists about which epistemology and ontology should be assumed. The criterion of *utility* that a design yields in application is considered relevant. It can be clearly measured through a utility metric, and would fit under a positivistic paradigm, whereas the act of design contains interpretative elements, especially the application in new or different situations, akin to interpretivism. It is beyond the scope of this paper to develop a genuine epistomology and ontology of design science research. For this paper we see validity/credibility as the truth of the utility statement of the design (i.e. the correctness of the statement "applying the method to a situation in scope will yield the claimed utility"). Objectivity/confirmability will be interpreted as the degree to which others are able to confirm the utility statement. Generalisability will be the assumption that the utility statement holds true for all possible instances within the scope, whereas transferability is the assumption that in out-of-scope situations that are nevertheless similar to those in the scope, the utility statement would hold to a certain degree.

4 Derivation of the Method Design Theory Components

In order to perform a method-specific specialisation of the design theory structure as proposed by Gregor and Jones (2007), we consider each of the eight components in turn and discuss what constitutes a specialisation specific to methods. We followed a hermeneutical approach to identify the refinements (Winograd and Flores 1986). Based on the individual pre-understanding regarding design theories and methods, the authors of this paper identified what would qualify as a method-specific specialisation of the theory elements of Gregor and Jones (2007). The different interpretations were then discussed and unified to outline their scientific role. To ground the resulting view in the discipline, each element was aligned with existing literature.

We then discuss how each element supports the criteria for theory evaluation. **Table 1** provides a summary of the results, including evaluation criteria derived from the method-specific specialisations in combination with the relevant theory evaluation criteria. Finally, we illustrate good specifications for each component by citing examples fulfilling the evaluation criteria. An overview of the examples used is given in **Table 2**.

4.1 Purpose and Scope

4.1.1 Method-specific Refinement

Purpose statements about methods contain three elements: The kind of output that the application of a method instance is supposed to produce (e.g. a particular kind of software), properties that the output bears (e.g. a certain degree of maintainability) and statements about the method itself (e.g. its efficiency). The scope of a method can be described from an "external" and "internal" perspective. The external perspective is concerned with the question of when and where a method can be applied. Preconditions to successful application include the type of product to be produced, but can also include properties of the organisation and team. Aier and Fischer (2009a) call this the "Anwendungsbereich" (application area/range) of a method, which consists of "project type" and "context" as conceptualised by Bucher et al. (2007).

The "internal" perspective deals with the scope of methods in terms of how much a method covers compared to the entirety that could be described by methods in principal. Cockburn (2007) proposes that methods can be characterised through "lifecycle coverage", "role coverage", and "activity coverage" (Cockburn 2007, p. 155). The lifecycle coverage describes which general phases this methods offers guidance for, the role coverage describes which roles are addressed through the method and the activity coverage lists the activities which are available for application in a project based on the method. In addition to Cockburn's structure, Iivari et al. (1998) discusses the term "approach" in distinction to the term "methodology" (synonymous to "method" in this paper). In Iivari et al. (2000–2001), the idea is further developed and the layer "paradigm" added. Each method belongs to an approach, which in turn belongs to a paradigm. A method contains techniques. For these, Oei and Falkenberg (1994) discuss how they can be compared based on their meta-model.

We have argued that a well-defined scope facilitates comparability between theories. This requires that the to-be-compared theories refer to the same spectrum of which they claim to cover a certain part. An agreeable common reference could be the Generalised Enterprise-Reference Architecture and Methodologies life-cycle model (ISO 2000, pp. 16–18). It defines seven phases from the identification to decommissioning of an artefact. For roles and activities we are not aware of any standard that is as widely accepted. For approaches in information systems development, the work of Iivari et al. (1998, 2000–2001) can be a starting point.

4.1.2 Scientific Role

Defining purpose and scope supports generalisability. The scope needs to be clearly defined in order to be able to generalise the validity of the utility statement from a limited number of supporting evidence to other cases. The scope defines the "population" of cases, in which applying the method is supposed to yield the claimed utility. Especially when concerning the external scope this can be a challenge as not all dimensions relevant to the utility of a method may be known. However, the differentiation between an instantiation of a method within the scope and purpose and the transfer of a method to a different scope and/or purpose is crucial to know if the utility statement is valid or has to be newly verified. At the same time the sufficiently rich description of purpose and intended scope can help to make the transfer to new settings easier.

Table 1 Summary of method design theory components

Component	Method-specific elements	Evaluation criteria
Purpose and scope	■ Project type ■ Project context ■ Lifecycle coverage ■ Role coverage ■ Activity coverage	■ Given a problem, is the purpose described in a way to tell if the method solves the problem? ■ Is the product of the method clear? ■ Given a problem context, is the scope described in a way to tell if the context falls within the scope? ■ Are all relevant dimensions specified within the scope? ■ Are the covered lifecycle phases, roles and activities specified?
Constructs	■ Method specific constructs (as defined by a meta-model, such as SPEM) ■ Output-specific constructs ■ Concepts in the application context	Are terms and concepts presented to ■ describe the method? ■ describe the resulting structure? ■ describe the application context? ■ describe the purpose and scope? ■ create the testable propositions? Are the concepts introduced comprehensively in one location and derived systematically?
Principles of form and function	■ Description of the method (according to the meta-model selected above)	■ Is the method described extensively enough to be useful and transferable? ■ For each possible role, is it clear which activities have to be done in what order?
Artefact mutability	■ Foreseeable changes (which part of the method, method base or method instance, kind of change) ■ Optional: support for change offered by method design paradigm (i.e. situational, method rationale)	■ Is the method described extensively enough to understand the functioning of each component? ■ Are conditions for changes described? ■ Does it become clear, which parts change and what they change into? ■ Are method tailoring mechanisms defined?
Testable propositions	■ Utility statement about theory ■ Optional: truth statements about match between method design and requirements of method output	■ Are testable propositions concerning the method utility given? ■ Are the propositions sufficiently well operationalised to allow testing by other scientists?
Justificatory knowledge	■ Design theories that this method is based on ■ Theories about the application context ■ Other aspects of interest	Are theories presented ■ for key method features not covered by testable propositions? ■ to support the testable propositions? ■ regarding the method product? ■ regarding the method setting?
Principles of implementation	■ Tailoring/assembly advice ■ Advice regarding introduction into real-life setting	■ Is the tailoring/assembly advice comprehensive with regard to the combinatorial possibility? ■ Does the advice for introducing the method cover different settings within the scope, or is at least clear for which it applies?
Expository instantiation	■ (Fictional) example of an instantiated method ■ Report on a real-life case study in which an instantiated method has been enacted	■ Does the example cover a case within the scope and is it covering the main concepts of the method? ■ Is the example specific enough to be illustrative but not too idiosyncratic to be incomprehensible?

4.1.3 Examples

The AAMM provides a proactive, evolutionary and demand-driven management of IS architecture. The attributes specify the characteristics and advantages of the method rather than the purpose. The purpose is the management of IS architectures, but the output of the method remains unclear. An example for an output specification can be found for SOAM: "The method [...] facilitates the design of an SOA system that adheres to the SOA design principles" (Offermann and Bub 2009a). CADI presents a good scope

Table 2 Overview of example methods used

Name of method	Abbreviation	Purpose and scope	Reference
Method for application architecture management	AAMM	Management of application architecture in companies	Hafner (2005), Hafner and Winter (2005)
Service-oriented architecture method	SOAM	Design of SOA-systems incl. legacy system integration for business information system architects	Offermann and Bub (2009a, 2009b), Offermann (2009)
Component-oriented architecture design and implementation	CADI	Analysis and design for component-oriented software systems based on business requirement	Stojanović (2005)
Service Design Method	SDM	Support IT providers in the design and description of their IT services	Abeck et al. (2005)
Methodology for Testing Intrusion Detection Systems (IDS)	IDSTM	Strategies and procedures for testing intrusion detection systems	Puketza et al. (1996)

statement: "Potential users [...] are all the actors in the development process, including business analysts, system architects and software developers" (Stojanović 2005, p. 11).

4.2 Constructs

4.2.1 Method-Specific Refinement

Constructs can either be derived from the method itself, the structure of the output it produces or the context in which a method enactment takes place. "Method" itself is naturally a relevant concept as much as the elements that constitute methods in general. Several method meta-models exist (cf. the discussion by Braun et al. 2005) and in most cases it will be the most comprehensive and yet pragmatic approach to model a method according to one particular meta-model. Other constructs might stem from the product or the application context of the method, e.g. the approaches from Iivari et al. (1998, 2000–2001). Such constructs might refer product classes (e.g. "middleware" or "service-oriented") or meta-requirements of the structure or the method. Relevant constructs are likely to be drawn from justificatory knowledge.

While our aim is not to offer a comprehensive and fixed ontology of possible method constructs, we can further delineate the possible concepts within a particular design theory by highlighting interrelation between concepts and other method elements. Firstly, the scope and goals will include terms that need to be introduced as constructs. The principles of form and function that describe the method can introduce further constructs and testable propositions about the theory are required to be built using constructs. The resulting harmonised method description will thus allow increased comparability, understanding and comprehension of the developed method.

4.2.2 Scientific Role

The constructs do not directly support a theory evaluation criterion. They form the basis for the description of all other components.

4.2.3 Examples

The AAMM extensively introduces constructs about the application context in Sect. 6.1, stretching over 39 pages. They are defined in a structured manner and are consequently used and reference in the method description. SOAM explicitly introduces method specific constructs: "Methods [...] describe a way to transform an initial state into a target state. [...] Activities explain what has to be done. [...] Activities produce results, but may also use existing results as inputs. [...] Techniques support the generation of results [...]. Finally, a meta-model for the results can be specified for clarity and consistency." Constructs relating to the resulting structure and the application context are introduced in a Sect. titled "Service-oriented architecture" in Offermann and Bub (2009a).

4.3 Principles of Form and Function

4.3.1 Method-Specific Refinement

As discussed in Sect. 3, the principles of form and function are the abstract blueprint or architecture of the artefact. If the artefact is a system, the notion of "architecture" is clearly defined either as "enterprise architecture" (e.g. Aier 2007) or "software architecture" (e.g. Bass et al. 2003) and this definition allows for a well-understood separation between architecture and the detailed design of method outputs. For methods, we are not aware of such a clear distinction, but consider situational method engineering to offer a parallel. Here, a general method is described which then can either be adapted or composed to the specific need. The general method represents the superset of individual solutions, similar to the differentiation between architecture and detailed design. Regardless, the principles of form and function require that each element (e.g. roles, activities, phases) of the method is described as defined by the meta-model that was chosen to define the internal constructs, together with all relations between elements. Justificatory knowledge should be referenced wherever possible.

4.3.2 Scientific Role

The principles of form and function are the "sine qua non" of utility. Without an appropriate description, the method can not be applied in any setting. The principles of implementation are supporting the manifestation of utility, but if the original method is insufficient or insufficiently described, advice on its implementation cannot remedy the flaws. The principles of form and function are also a supporting factor of the method's validity for the same reasons, as the utility claim can only be tested if one is reproducing the method appropriately.

4.3.3 Examples

Comprehensive method descriptions often stretch over many pages. The AAMM needs 148 and SOAM 50 pages. While the appropriate length depends on the internal scope of the method, e.g. lifecycle phases covered, thick method descriptions tend to be rather long. This poses problems for researchers as scientific publications are often limited in length.

The SDM is structured along the roles involved in the process. "(1) [The method is] initiated by the service manager [...]. The service manager's main focus is [...] (3) The service module manager fills the templates [...] (4) The service module catalogue owner collects [...] (5) The service manager accesses the finished [...] catalogue [...]. (6) As the final activity [...] by the service catalogue owner [...]" (Abeck et al. 2005).

4.4 Artefact Mutability

4.4.1 Method-Specific Refinement

For methods, artefact change can either happen on the level of method design (i.e. the artefact described in a design theory) or on an instance of a method (the "method in action"), used by a particular group of people in a particular company. With these conceptual differentiations in mind, we can ask what artefact mutability means for methods. Mutability of the first kind implies that the designer can foresee changes of the method itself. If a situative method engineering approach is followed this could manifest in method chunks that could be added or altered for new usage scenarios. Mutability of the second kind would imply that possible changes in the instantiation of the method are foreseen in the original method design and are described there. We argue that both types of mutability fall under the "artefact mutability" as introduced by Gregor and Jones (2007).

Situational method engineering offers a conceptual blueprint for combining both levels: Experience from earlier projects can be included in the method base over time as each individual instance is adapted "just-in-time" from the then-current method base; it always reflects all changes to the method base up to that point. The practise of altering a method instance while using it is sometimes referred to as "dynamic method tailoring" (Karlsson 2008) or "evolutionary method engineering" (Rossi et al. 2004; Rossi et al. 2000). The latter suggest documenting the "method rationale", which they understand to be the design decisions that lead to a particular method instance.

We propose that the author of a method design theory should describe which elements of the proposed method might change in which ways, and on which level this occurs (method or instance). The author might additionally refer to a method paradigm that supports incorporation of change in general in order to illustrate how the change will be performed.

4.4.2 Scientific Role

Specifying artefact mutability supports generalisability. Adaptations of the principles of form and function should be discussed to enable the user to perform an adaptation to the peculiarities of a situation. Understanding how and which changes the method designers could envision is also possibly helpful for transferability, as intentions for change are expressed.

4.4.3 Examples

SOAM specifies some phases and activities as being optional. However, it is not clear when to exclude which activity and how to tailor the method to certain project situations such as a Greenfield or a workflow-only approach.

The IDSTM states that it might be simplified for a limited scope: "The procedures are designed for testing an IDS that monitors a network of computers, although some of the procedures can be directly applied to an IDS that only monitors a single computer." (Puketza et al. 1996, p. 724). While the scope limitation is clear, the adaptation is not. The specific procedures applicable still would have to be determined.

4.5 Testable Propositions

4.5.1 Method-Specific Refinement

For comprehensive guidance regarding testable propositions, we believe that three points have to be discussed:

- What the statements are about (the method itself or the results produced?),
- what is asserted by the statement, "utility" or "truth" (the latter in terms of the design fulfilling requirements), and
- how complete the collection of testable propositions must be.

A possible framework for statements about the method itself can be found in Moody (2003), who describes a method evaluation model, based on the Technology Acceptance Model (Davis 1989).

Regarding "utility" vs. "truth", Venable (2006) argues that design theories are solely concerned with assessing "utility". Such statements are broad in scope, as they relate a complete class of solutions to a complete class of problems. The second view is offered by Walls et al. (1992) who state that a "design process hypothesis" must be able to verify that the artefact produced by a process is consistent with the defined meta-requirements. The statements would refer to individual elements of solution and problem and therefore be more fine-grained than the utility statements of Venable. We argue that both types of propositions are useful, but that at least one utility statement must be made. The utility statement is the basis for considering the design theory: if the suggested solution does not solve the problem it was designed for or if it is not better than known approaches, the contribution is not significant.

Finally, it needs to be discussed whether the author of the theory can and should provide all possible propositions. Gregor and Jones (2007) see testable propositions as a required part of design theories, which justifies the assumption that they believe in an author's capability to provide sufficiently complete and meaningful propositions. It is easily imaginable, though, that an author does not see all possibly interesting propositions. The credibility of a theory might still depend on unstated propositions and it would be inconsequent to require that the author's propositions are the only "binding" ones. Because of this, we believe that the author's propositions should be viewed as an attempt to improve credibility for the most relevant claims and for those that are least established for an existing body of knowledge.

4.5.2 Scientific Role

The testable propositions support the validity of the method. The validity mainly concerns the utility statement; a method is valid if it is useful in respect to the purpose. To be objective/confirmable, the hypotheses also have to be operationalised in a way reproducible by a

different researcher. Some additional hypothesis might be of qualitative nature, requiring credibility rather than validity.

4.5.3 Examples

For SOAM, it should be expected that the method is better than other methods of the same application domain, or at the least equivalent. While this might be desirable, the hypothesis is very generic as "better" cannot easily be measured. It is detailed that "Using a questionnaire, 'perceived ease of use', 'perceived usefulness' and 'intention to use' can be evaluated" (Offermann and Bub 2009a). These constructs are then detailed by specific questions in a questionnaire. Still, the testable hypotheses are quite broad. It might be helpful to also provide testable hypotheses for smaller parts of the methods, such as activities or techniques. Also, these smaller parts might have a stronger grounding in justificatory knowledge.

4.6 Justificatory Knowledge

4.6.1 Method-Specific Refinement

The different aspects of the design theory, i.e. the method itself, the product and the application context all provide possible anchors for theory. The selection of appropriate theories is highly situation specific and can not be canonised. The general discussion in Gregor and Jones (2007) regarding admissible kinds of knowledge applies.

4.6.2 Scientific Role

Justificatory knowledge supports transferability and validity. By presenting a justification for the method design, these justifications can be re-evaluated when transferring the method to a different purpose and/or scope. Justificatory knowledge supporting testable hypotheses supports their validity.

4.6.3 Examples

The AAMM discusses the following approaches: Enterprise Architecture Management by IBM; The Open Group Architecture Framework (TOGAF) by 'The Open Group'; Management of IT-Architectures following Dern; IT-Architecture Engineering following Krüger and Seelmann-Eggebert. However, this knowledge is used rather to differentiate the new method than to justify parts of the new method. SOAM, on the other hand, explains what existing methods the activity is based on for each activity.

4.7 Principles of Implementation

4.7.1 Method-Specific Refinement

Implementing a method means to bring a generic method design into action for a specific situation. This encompasses two aspects:

- The adaptation of a general method for a particular situation, and
- the introduction of the adapted method into the organisation that is supposed to work using the method.

The first item closely relates to method adaptation in situative method engineering. Advice on the process of how and where a method can be altered can be considered to fall under "principles of implementation". The second item subsumes a variety of aspects, from teaching potential method users as well as implementing tool-support or altering organisational policies. Both items can be optionally provided for method design theories. In the case of situative method engineering, guidance on adaptation is mandatory.

4.7.2 Scientific Role

Principles of implementation concerning implementation within the scope will support the utility claim, while a discussion about adaptation and implementation outside the scope will support transferability.

4.7.3 Examples

The IDSTM offers the following principle of implementation: "Often, in the course of testing an IDS, it may be necessary to repeat a particular test. For example, a test can be repeated to determine why (or why not) the IDS failed the test. Repeating the execution of a sequential test script is accomplished by simply running the script again with the same input." (Puketza et al. 1996, p. 722).

4.8 Expository Instantiation

4.8.1 Method-Specific Refinement

The expository instantiation is a "real-life", exemplar application of the design theory. For methods, instantiation can, in principle, mean two things:

- The derivation of a specific method description for a particular context and situation, and
- a report on the execution of such a derived specific method in the context and situation it was made for.

Gregor and Jones (2007) seem to hint at the second option. We argue that both types of instantiation can be illustrative for the design, whereas the second type is more informative and therefore preferable.

4.8.2 Scientific Role

Expository instantiations support generalisability, transferability and validity. An exemplar application is useful when creating a new instance of the method (within the scope, or transferred to a new scope), as an example often is easier to understand than a theoretical description, and can illustrate the intentions of the method designer beyond descriptive elements of the method. If the example is a real-life application of the method, it can serve as a data point for the validity of the utility and even if it is fictional, it can support other researchers in understanding what the method designer would consider a fair and appropriate example for an evaluation scenario.

4.8.3 Examples

The AAMM is derived through the generalisation of instances. The resulting method itself is not instantiated again. SOAM, on the other hand, "was used in four companies: Vattenfall Europe, Bosch und Siemens Hausgeräte (B/S/H), Prevent DEV and Ideal Lebensversicherung" (Offermann and Bub 2009b). Presenting the implementation in different companies is very helpful as each company has different requirements; implicitly, principles of implementation are presented.

5 Discussion

In the previous section we have introduced a model for structuring method design theories and have shown for exemplary method publications how to evaluate whether they contain the information necessary for a method design theory. We have borrowed the structure of design theories from a more general

Abstract

Philipp Offermann, Sören Blom, Olga Levina, Udo Bub

Proposal for Components of Method Design Theories

Increasing the Utility of Method Design Artefacts

Gregor and Jones have proposed components for design theories, building on theory concepts from behavioural sciences and prior publications. Their design theory structure addresses IT artefacts in general, not specific to any type, such as constructs, models, methods or instantiations. Their work is an important contribution to the academic discussion of design theories. The authors are building on this and believe that specialised design theory structures for different types of artefacts further increases utility, usability and acceptance of the components for both academia and practise. They have analysed each of the components published by Gregor and Jones and proposed refinements specific to method design artefacts wherever applicable. For each component, they derive evaluation criteria and present examples of method publications fulfilling the criteria. They argue that by presenting method design theories according to this structure the contribution of method design artefacts to the body of knowledge will increase.

Keywords: Methods, Method engineering, Method construction, Theory, Design theory, Methodology

paper from Gregor and Jones (2007). Accordingly, the selection of components that are part of the theory are only as complete and appropriate as one considers those in the original paper to be. We then refined each component in turn towards becoming more method specific. The outcome of this refinement is grounded in literature; that is, we did not suggest refined concepts that are unknown in method design. Finally, we identified the role of each component in supporting theory evaluation criteria. We feel that this aspect is important to clarify how a design theory differs from merely a design, but have also admitted that this discussion requires further thought. Clarifying whether design science has a unique epistemological view (which is our opinion), would allow for a better justification of the points made in this paper. We will dedicate future work to this task.

Additionally, we looked at examples of academically published method designs to see whether parts of our proposals can be found in practise and if so, to illustrate the evaluation criteria. While we have found individual aspects in different method publications, we have not found one that covers all or almost all aspects. One very practical issue that we have identified as a possible cause is limitation of space. A complete method description, i.e. principles of form and function, can be very long in itself. It might be more feasible to prepare a practitioner publication containing only the designed method, and to publish the method design theory separately. In the light that some design theory components cannot be filled in the beginning or even at all, a life-time approach may be taken. Theories would be proposed "bare bones" and then elements would be added as the theory becomes better understood and supporting knowledge is created around it.

6 Conclusion

In this paper we have realised one of the recommendations of Walls et al. (2004), namely to propose a specialised version of the design theory structure for methods. We proposed a method description structure built on the design theory structure proposed by Gregor and Jones (2007). We also proposed evaluation criteria for each component. Our idea was that such a specialisation with method-specific evaluation criteria would be more readily applicable for proposing and evaluating method designs as design theories, providing benefits to research and practise. The proposed scientific description structure enables transparent and grounded presentation of the method, therefore supporting communication and deeper understanding of the intellectual core. We were able to offer method-specific concepts for most components of the structure (cf. **Table 1**). In current method publications, missing or incomplete descriptions of the methods leave actual benefits open and evaluations on shaky ground. Using the proposed evaluation criteria aims at reducing these problems. Thereby, research would benefit from a more cumulative approach to method engineering. For practitioners, the usage of the structure would make it clearer when and where a method can be applied, how it compares to other methods and what measurable benefits can be expected.

Still, some issues remain. It remains to be discussed how method design theories should be published. Also, to create a consistent body of knowledge of IT methods, individually published design theories would have to be aggregated first and then made comparable, serving as a major facilitator of scientific progress. To achieve this, a consensus of how to define scope and purpose would have to be reached. With a consensus on how to describe method design theories technical standardisation could follow, which would make referencing, search and comparison even easier.

References

Abeck S, Link S, Mayerl C, Mehl O, Vogel T (2005) A system supported method to design IT services. In: IEEE conference on integrated management (IM 2005), Nice

Aier S (2007) Integrationstechnologien als Basis einer nachhaltigen Unternehmensarchitektur – Abhängigkeiten zwischen Organisation und Informationstechnologie. GITO, Berlin

Aier S, Fischer C (2009a) Dokumentation und Fortschrittsbestimmung von Methoden zur Gestaltung soziotechnischer Systeme am Beispiel einer Methode zum Service Engineering. In: Hansen HR, Karagiannis D, Fill H-G (eds) 9. Internationale Tagung Wirtschaftsinformatik – Band 1, Wien

Aier S, Fischer C (2009b) Scientific progress of design research artefacts. In: 17th European conference on information systems, Verona

Alter S (2002) Sidestepping the IT artifact, scrapping the IS silo, and laying claim to "systems in organizations". Communications of the Association for Information Systems 12:494–526

Basili VR, Rombach HD (1987) Tailoring the software process to project goals and environments. In: Proceedings of the 9th international conference on software engineering, Monterey

Bass L, Clements P, Kazman R (2003) Software architecture in practise, 2nd edn. Addison-Wesley, Boston

Braun C, Wortmann F, Hafner M, Winter R (2005) Method construction – a core approach to organizational engineering. In: Proceedings of the 2005 ACM symposium on applied computing, Santa Fe

Brinkkemper S (1996) Method engineering: engineering of information systems development methods and tools. Information and Software Technology 38:275–280

Brinkkember S, Saeki M, Harmsen F (1999) Meta-modelling based assembly techniques for situational method engineering. Information Systems 24(3):209–228

Bucher T, Klesse M, Kurpjuweit S, Winter R (2007) Situational method engineering – on the differentiation of "context" and "project type". In: Situational method engineering: fundamentals and experiences. Springer, Boston, pp 33–48

Cockburn A (2007) Agile software development. Addison-Wesley, Upper Saddle River

Cushing BE (1990) Frameworks, paradigms and scientific research in management information systems. Journal of Information Systems 4(2):38–59

Davis FD (1989) Perceived usefulness, perceived ease of use, and user acceptance of information technology. MIS Quarterly 13(3):319–340

Davis FD, Bagozzi R, Warshaw P (1989) User acceptance of computer technology: a comparison of two theoretical models. Management Science 35(8):982–1003

Dubin R (1978) Theory building. The Free Press, New York

Eberlein A, Jiang L (2007) Description of a process development methodology. Software Process Improvement and Practice 12:101–118

Fitzgerald B, Russo N, O'Kane T (2000) An empirical study of system development method tailoring in practise. In: Hansen HR, Bichler M, Mahrer H (eds) Proceedings of the eighth European conference on information systems, Vienna

Gericke A (2008) Konstruktionsforschung und Artefaktkonstruktion in der gestaltungsorientierten Wirtschaftsinformatik: Ein Literaturüberblick. http://www.alexandria.unisg.ch/publications/49921. Accessed 2009-10-19

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

Gregor S (2002) A theory of theories in information systems. In: Gregor S, Hart D (eds) Information systems foundations: building the theoretical base. Australian National University, Canberra, pp 1–20

Gregor S (2006) The nature of theory in information systems. MIS Quarterly 30(3):611–642

Gregor S (2009) Building theory in the sciences of the artificial. In: DESRIST'09, Malvern

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

Greiffenberg S (2004) Methodenentwicklung in Wirtschaft und Verwaltung. Dr. Kovač, Hamburg

Guba EG (1981) Criteria for assessing the trustworthiness of naturalistic inquiries. Educational Communications and Technology Journal 29(2):75–91

Hafner MJ (2005) Entwicklung einer Methode für das Management der Informationssystemarchitektur im Unternehmen. Difo-Druck, Bamberg

Hafner M, Winter R (2005) Vorgehensmodell für das Management der unternehmensweiten Applikationsarchitektur. In: Ferstl O (ed) Wirtschaftsinformatik 2005: eEconomy – eGovernment – eSociety. Bamberg

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

Iivari J (2002) The IS core – VII: Towards information systems as a science of meta-artifacts. Communications of the Association for Information Systems 12:568–581

Iivari J, Hirschheim R, Klein HK (1998) A paradigmatic analysis contrasting information systems development approaches and methodologies. Information Systems Research 9(2):164–193

Iivari J, Hirschheim R, Klein HK (2000–2001) A dynamic framework for classifying information systems development methodologies and approaches. Journal of Management Information Systems 17(3):179–218

ISO (2000) Industrial automation systems – requirements for enterprise-reference architectures and methodologies. ISO 15704

Jones D, Gregor S, Lynch T (2003) An information systems design theory for web-based education. In: IASTED international symposium on web-based education, Rhodes

Karlsson F (2008) A wiki-based approach to method tailoring. In: Proceedings of the 3rd international conference on the pragmatic web: innovating the interactive society, Uppsala

Karlsson F, Ågerfalk PJ (2004) Method configuration: adapting to situational characteristics while creating reusable assets. Information and Software Technology 46:619–633

Kitchenham B, Pickard L, Pfleeger SL (1995) Case studies for method and tool evaluation. IEEE Software 12(4):52–62

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

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

Moody DL (2003) The method evaluation model: a theoretical model for validating information systems design methods. In: Ciborra CU, Mercurio R, de Marco M, Martinez M, Carignani A (eds) Proceedings of the eleventh European conference on information systems, Naples

Oei H, Falkenberg E (1994) Harmonisation of information systems modelling and specification techniques. In: Verrijn-Stuart AA, Olle TW (eds) Methods and associated tools for the information systems life Cycle. Elsevier Science, North-Holland, Amsterdam, pp 151–168

Offermann P (2009) Eine Methode zur Konzeption betrieblicher Software mit einer Serviceorientierten Architektur. GITO, Berlin

Offermann P, Bub U (2009a) Empirical comparison of methods for information systems development according to SOA. In: 17th European conference on information systems, Verona

Offermann P, Bub U (2009b) A Method for information systems development according to SOA. In: AMCIS 2009 proceedings, Paper 108

Offermann P, Levina O, Schönherr M, Bub U (2009) Outline of a design science research process. In: Proceedings of the 4th international conference on design science research in information systems and technology, Philadelphia

OMG (2008) Software & systems process engineering meta-model specification. http://www.omg.org/cgi-bin/doc?formal/08-04-01.pdf. Accessed 2009-09-25

Peffers K, Tuunanen T, Rothenberger MA, Chatterjee S (2008) A design science research methodology for information systems research. Journal of Management Information Systems 24(3):45–77

Plihon V, Rolland C (1997) Using a generic approach to support the construction of methods. In: Database and expert systems applications. Springer, Berlin, pp 663–672

Puketza NJ, Zhang K, Chung M, Mukherjee B, Olsson RA (1996) A methodology for testing intrusion detection systems. IEEE Transactions on Software Engineering 22(10):719–729

Purao S (2002) Design research in the technology of information systems: truth or dare. GSU Department of CIS Working Paper

Ralyté J, Deneckère R, Rolland C (2003) Towards a generic model for situational method engineering. In: Advanced information systems engineering. Springer, Berlin

Rossi M, Tolvanen J-P, Ramesh B, Lyytinen K, Kaipala J (2000) Method rationale in method engineering. In: Proceedings of the 33rd Hawaii international conference on system sciences, Hawaii

Rossi M, Ramesh B, Lyytinen K, Tolvanen J-P (2004) Managing evolutionary method engineering by method rationale. Journal of the Association for Information Systems 5(9):356–391

Siau K, Rossi M (1998) Evaluation of information modeling methods – a review. In: Proc 31st annual Hawaii international conference on system sciences, Hawaii

Simon H (1981) Sciences of the artificial, 2nd edn. MIT Press, Cambridge

Song X, Osterweil LJ (1992) Toward objective, systematic design-method comparisons. IEEE Software 9(3):43–53

Stojanović Z (2005) A method for cComponent-based and service-oriented software systems engineering. Delft University of Technology

ter Hofstede AHM, Verhoef TF (1997) On the feasibility of situational method engineering. Information Systems 22(6/7):401–422

Travis J (1999) Exploring the constructs of evaluative criteria for interpretivist research. In: Proc. 10th Australasian conference on information systems

Vaishnavi VK, Kuechler W (2007) Design science research methods and patterns: innovating information and communication technology. Auerbach

Venable JR (2006) The role of theory and theorising in design science research. In: DESRIST 2006, Claremont

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? Journal of Information Technology Theory and Application 6(2):43–58

Walsham G (1995) Interpretative case studies in IS research: nature and method. European Journal of Information Systems 4(2):74–81

Wilde T, Hess T (2007) Forschungsmethoden der Wirtschaftsinformatik – Eine empirische Untersuchung. WIRTSCHAFTSINFORMATIK 49(4):280–287

Winograd T, Flores F (1986) Understanding computers and cognition: a new foundation for design. Ablex Publishing, Norwood