# DESIGN SCIENCE RESEARCH IN ACTION – ANATOMY OF SUCCESS CRITICAL ACTIVITIES FOR RIGOR AND RELEVANCE

## Abstract

*Design Science Research (DSR) has reached a significant impact on scholar's research work around the globe in the information systems domain. DSR is an important IS research paradigm for creating descriptive and prescriptive knowledge concerning the artificial construction of today's reality in the interrelation between the social and the technological sub-system. Various prior research has decisively defined and structured DSR in order to derive rigorously relevant contributions in terms of frameworks and methodologies. This paper contributes to this discourse from a research in action point of view by investigating critical activities within the design science phases and when passing from one to the next DSR cycle. For that purpose we elaborate critical DSR activities and demonstrate their effective execution along four DSR in action examples to provide guidance and best practices for design science projects seeking for rigor and relevance.*

*Keywords: Design Science Research, DSR, Research Activities, Rigor, Relevance.*

## 1 Introduction

The design science approach originally goes back to engineering and has since gained significant attention in the domain of information systems (IS) research. Starting in the 60s and 70s scholars majorly focused on distinguishing the design science research (DSR) paradigm from positivist research approaches in natural science and social sciences (Orlikowski and Baroudi, 1991). It was then that Simon (1996) laid the foundation of the science of design in mathematics and defined designing as a search process within a closed solution space resulting in an optimized design or respective optimum. Many of the early research of IS was conducted using the design science paradigm, focusing on systems development approaches and methods such as the socio-technical approach (Bostrom and Heinen, 1977; Mumford, 1983) and the info-logical approach (Langefors, 1966; Sundgren, 1973; Lundeberg et al., 1978). Later on, researchers seemed to lose sight on design science until the beginning of the 1990s, when a variety of scholars revived design science in information systems: Walls et al. (1992) for example, broke new ground when they investigated design under the light of descriptive knowledge in information systems and formulated the information system design theory (ISDT). They concluded that rigor design science research in information systems must be informed by established behavioural theories. Since then, much work has been published trying to define the paradigmatic nature of information system research as a design science. Such research included the ontology of design science, especially including the place of an artefact in its context (Orlikowski and Iacono, 2001; Benbasat and Zmud, 2003; Iivari, 2007; Baskerville, 2008), the epistemology of design science, investigating first the nature of kernel theories and justificatory knowledge (Wall et al., 1992; Gregor and Jones, 2007) and second the outcome of design science in the form of a design theory (Gregor and Jones, 2007; Markus et al., 2002). Others focused on the methodology of design science by proposing particular scientific methods to create and evaluate designs (e.g., Hevner et al., 2004; Sein et al., 2011). By now, several DSR frameworks exist in literature that divide the design process into several phases by defining a set of milestones within the design process. Furthermore, they usually promote an iterative approach comprising several cycles of the design process (e.g., Takeda et al., 1990; Peffers et al., 2008; Sein et al., 2011). Even though, research slowly seems to converge towards a common understanding about which phases are essential for a design science process, there is only a small amount of publications actually addressing essential research activities necessary within each phase. This includes the inductive and deductive steps necessary to get from a practical problem to a set of design principles, the deductive actions to derive more concrete design decisions, the activities which lead to an instantiated artefact and finally methods leading to a comprehensive evaluation concept allowing generalizations and inductive conclusions about the underlying design principles and theories.

The following paper focuses on those specific activities of core DSR phases, which are critical to successfully achieve the descriptive and prescriptive epistemological goals by rigor and relevant DSR. This is done along four examples of design science works, to achieve a more tangible insight: Gass et al. (2011), Gurzick and Lutters (2009), Koppenhagen et al. (2011) and Zhang et al. (2011). These recent research works have been selected, because they transparently follow prominent DSR frameworks and/or methodologies and have completed one or many iterations providing relevant contributions.

The paper is structured in the following way: Section two gives an overview of existing methodological work that forms the basis of our research. Section three presents the high level activitiy exploration based on the closed loop epistemological system and the mapping to considerable DSR framworks phases. Section four highlights the reaseach activity elaboration and guidelines along core DSR phases and the four DSR examples. Section five concludes with the summary of the contributions and reflects upon limitations and future research opportunities.

## 2 State of the Art in Literature

The next paragraph gives an overview of major contributions to design science methodology in information systems. The presented work does not claim to be complete. However, we understand our own research as an extension of the results presented in these papers:

Takeda et al. (1990) built a design process model based on the general design theory (GDT) and resulting in a descriptive and cogitative model of design processes. The GDT is a mathematical formulation of the design process which represents an abstract theory about knowledge in a design process. According to GDT, a design process is characterized as a mapping from the function space to the attribute space, both of which are defined by the entity concept set. The function space comprises a topology of functions which are mapped to a topology of attributes in the attribute space. Therefore, a design specification, e.g., a set of requirements, is represented by a point in the function space, while a concrete design solution represents a point in the attribute space. Under the condition of ideal knowledge, a design solution is immediately obtained once the specifications are formulated (Yoshikawa, 1981; Tomiyama and Yoshikawa, 1987). Takeda et al. (1990) adjusted some underlying assumptions of the GDT. First, they interpret design not just as a simple mapping, but further include the thought of design as a constant refinement process. Second, they account for the fact that the concept of function is not always objectively formalized. Third, they replace the assumption of ideal knowledge with the concept of real knowledge which accounts for physical constraints. Building on these three adjustments, they propose a descriptive design process model that regards design as an evolutionary process which transfers the design specification to a design solution by gradually adding more attributes and refining existing ones. The current state of design solution is described by a meta-model which incorporates context independent solution entities. In order to evaluate the current state of the meta-model, a context specific instantiation has to be derived to see if the design model fulfils the specifications. The results of the evaluation are used to refine the meta-model. In order to determine how the descriptive model manifests itself in practical design projects, they conducted an experiment with designers who were asked to develop a solution design for a particular problem. The analysis of the experiment resulted in the general design cycle defining five major phases during the design process. These phases are, first, the awareness of the problem in which the object under consideration is compared with the design specifications. Second, the suggestion step, in which a concept is developed to solve the problem. Third, the development processes leading to context specific instantiations of the solution concept. Fourth, the evaluation of the solution candidates by testing them in various ways and fifth, the conclusion step, to decide which candidate to adopt, modifying the meta-model respectively (Takeda et al., 1990). Nunamaker et al. (1990) propose an approach that combines elements of the social science and engineering to a research methodology for systems development. They stress the importance of a strong foundation of research on an existing knowledge base and emphasize the fact that the outcomes of research must contribute to this knowledge base. In general, they state that system development research can be characterized by the three stages of concept, develop and impact. In detail, these three stages unfold in a system development process, first in the form of a conceptual model including the investigation of systems functionalities and requirements and their formulation in the form of a research question, second, the development of system architecture, third, the design of the system itself, fourth, the instantiation of the design in the form of prototypes and fifth the evaluation of the prototypes. The work of Walls et al. (1992) emphasized the need for a IS design theory. They define a design theory as prescriptive knowledge which brings together explanatory, predictive and normative aspects. A design theory prescribes a path leading to a more effective design. Their formal definition of a design theory includes both aspects of the product design as well as the design process. Components of a design theory which refer to the design of the product itself are kernel theories, meta-requirements defining a class of goals to which the theory applies, meta-design describing a class of artefacts which meet the meta-requirements and testable hypotheses about the design of the product. Components of the design process are kernel theories, design methods and testable design process hypotheses. An important aspect is the use of the term kernel theory which refers to the behavioural theory that forms the core of a design theory. Another important aspect is the awareness that design itself does not incorporate truth value, instead only hypotheses about the expected impact of a design can be tested and if necessary refuted (Walls et al., 1990). Often cited is the work of Hevner et al. (2004). Similar to Nunamaker et al. (1990), they analyze the interaction between the design and its practical and academic environment. According to Hevner et al. (2004), a pending problem in design science research is the discrepancy of rigor and relevance, design research and routine design. In order to address this discrepancy they propose seven guidelines to conduct design science research. These guidelines emphasize that design science must be motivated by a practical problem, which, however, is addressed by rigor methodology, including an evaluation according to scientific standards and also the communication of results to a community (Hevner, 2004). The framework of Peffers et al. (2008) defines six phases similar to the original design science cycle. They add one important point to the cognitive model of design research. While previous research, e.g., Takeda et al. (1990), incorporated only one phase for the deduction of a solution, Peffer et al. (2008) distinguish between two abstraction levels of the solution concept. First, on a rather abstract level, the objectives of a solution which outline the design by the desired impact, and second, more concrete, the design decision addressing particular design characteristics of the artefact (Pfeffer et al., 2008). Sein et al. 2011 extended the design science research methodology by adding action research elements to the process model. Previous work usually speaks of an instantiation of a solution in a meaningful context for evaluation. Sein et al. (2011) distinguish between several contexts, including researchers, practitioners and end-users.

## 3 Exploring Core DSR Phases and Activities

To identify core DSR phases and first high level critical activities within design science research, we looked at it from two perspectives: 1) along the epistemological loop of relevant and rigor DSR in the IS domain, 2) based on related work in the field of design science methodologies and frameworks.

### 3.1 Closed Loop Epistemological System of Relevant and Rigor DSR

Based on Hevner et al. (2004) and Sein et al. (2011) Figure 1 depicts the epistemological loop of practical relevance and rigorously conducted design research. An actual research project can be motivated by inspiring and informing practice and by the existing theory base. In the first case, problems or research issues are identified from the practical socio-technical system and inductively aggregated to classes of problems and related potential classes of solutions. In the second case, existing theories (e.g., kernel theories) suggest potentials for further research or provide guidance to apply scientific knowledge to the socio-technical problem space. Therefore, in a deductive effort, classes of problems subject to research and related possible solution classes can be derived. The combined result of the practical induction and the theoretical deduction forms a class of problems and respective solutions as basis of design science research projects - the ‘deductive/inductive initialization’ phase. Instantiated from the class of problems/solutions and as part of the practical induction, requirements are explored to potentially address practical problems. Informed by the problems and solutions space, as well as by the relevant practice and justificatory knowledge, design principles are identified, which are high level responses to the identified key requirements and, therefore, are instantiations of the class of potential solutions – the ‘design principles induction’ phase. The software artefact is a central instantiation in DSR projects of the problem and solution classes and is composed out of design decisions, converting generic abstract design principles into tangible features, architecture components, user interface elements, use-cases, scenarios, etc. We use the term artefact synonymously to software or IT artefacts and, therefore, refer to it as a technical sub-system which supports processes that are not processes per se (Pries-Heje et al., 2008). An artefact is developed and applied in the social context - the ‘artefact build’ phase - which is followed by its ‘evaluation’ phase. From there, the artefact evaluation results are further formalized to a design theory (Walls et al., 1992) dependent on the grade of generalizability to prescriptively inform other DSR projects also addressing the respective class of problems. Design Principles and potential design theories would be a significant enhancement or addition to the existing scientific body of knowledge. Consequently, we argue that design principles are essential and design theories, dependent on the grade of generalizability, ample contributions of DSR projects. From the relevance perspective, a specific artefact version, after one-to-many DSR cycle iterations, can be intervened as an application system into the social sub-system and as part of the socio-technical practice, the application system could be a potential source for (a) class(es) of issues. The epistemological loop of DSR closes here and its results could be used to inform subsequent research loops.

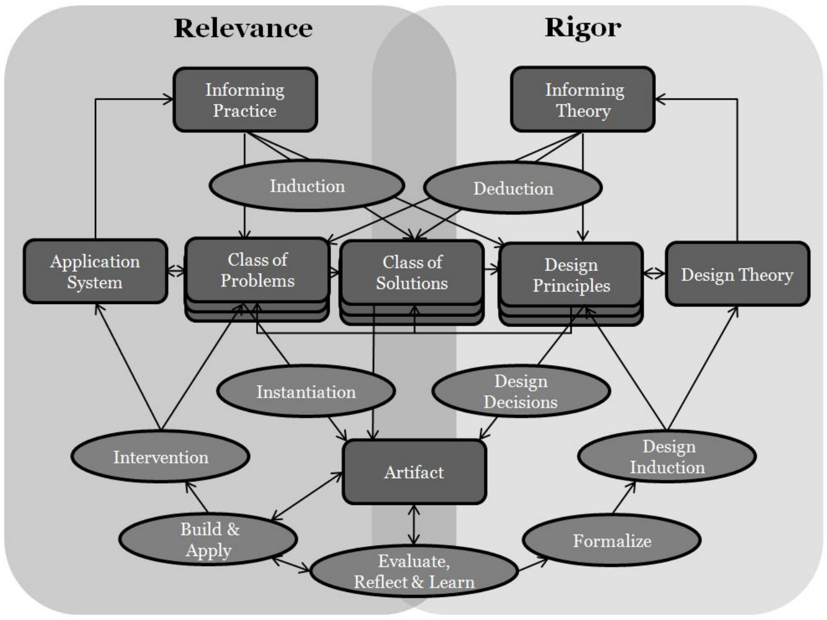

*Figure 1. Closed loop epistemological system of relevant and rigor DSR.*

When mapping to prominent DSR frameworks a great fit between the core phases and respective framework phases was intended. Especially to the framework of Nunamaker et al. (1990), a mapping can be done close to all phases, excluding design principles related activities. We perceive also a great fit to Hevner et al. (2004). In general, design principles induction seems to have a rather low mapping potential compared to the other activities. This leads to the assumptions that this phase and respective activities are rather new explications and have not been well formalized as of today or have not been explicitly detailed and reflected on. Table 1 summarizes the mappings between DSR framework phases and core phases (based on Peffers et al., 2008).

| | *Deductive / Inductive Initialization* | *Design Principles Induction* | *Artefact Build* | *Evaluation* |
|---|---|---|---|---|
| Taketa et al. (1990) | Problem Enumeration, Awareness of Problem | n.a. | Development | Evaluate and Conclusion |
| Nunamaker et al. (1990) | Construct a Conceptual Framework | n.a. | Built the (Prototype) System | Observe & Evaluate the System |
| Walls et al. (1992) | Meta Requirements, Kernel Theories | Design Method, Meta Design | Testable Design | Test Process/Product Hypothesis |
| Hevner et al. (2004) | Important and Relevant Problems | Foundations Constructs, Models, Methods | Develop/Build | Justify/Evaluate |

| | | | | |
|---|---|---|---|---|
| Peffers et al. (2008) | Problem Identification and Motivation | n.a. | Demonstration | Evaluation |
| Sein et al. (2011) | Problem Formulation | Practice-inspired Research | Building, Intervention | Evaluation, Formalization of Learning |

*Table 1. Activity Mapping to DSR Framework Phases (based on Peffers et al., 2008)*

Based on those considerations we focus our elaboration on critical activities of core DSR phases, as they are located in the heart of the epistemological loop, of actual DSR frameworks and methodologies as well as they play major roles in many DSR projects and allocate significant amounts of resources. Therefore, the critical activities within the following four core DSR phases are a potential source of efficiency in research projects: **1) deductive/inductive initialization**, **2) design principles induction**, **3) artefact build**, **4) evaluation**.

## 4 Activity Elaboration in DSR Project Phases

### 4.1 Deductive/Inductive Initialization

The inductive initialization of DSR projects from the informing practice, based on related justificatory knowledge, deserves specific attention as the motivation for starting research activities. Practical problems are often rather implicit and rarely made explicit, e.g., in a way that research topics are selected along defined criteria and relevance measures. Very often, research motivation is grounded in the practical background and knowledge of the researchers or the socio-technical system they are imbedded in. This does not imply something objectionable and in many cases it also makes sense to utilize expert knowledge of the researcher, in particular in interpretative or critical philosophical settings of the intended research (Gregor and Jones, 2007). Nevertheless, we argue for additional sources of inductive research initializations, for instance research tendering, where research cases could be submitted, evaluated and decided on in a structured process along defined research criteria for relevance and applicability to a scientific discourse. Such an argument could be built along the following criteria examples: grade of innovation, potential to generalize, socio-technical impact, follow-up research to an introduced application system, executability in a DSR project (staffing, knowhow, budgeting, facilities, involvement of practitioners, etc.). The deductive initialization of DSR projects can be motivated by research gaps, recommended further research or deductions of theories: kernel theories from natural or social science, mid-range theories, theories in use and design theories (Kuechler and Vaishnavi, 2008). Most prominently, theories are applied to testable product design hypothesis and more seldom to testable design process hypothesis (Wall et al., 1992). Table 2 provides an overview of the motivation and initialization approaches of the four related DSR works.

| | *Deductive / Inductive Initialization* |
|---|---|
| Gass et al. (2011) | • Research motivated by observation that end-users consume an increasing number of cloud services<br>• A general problem definition was induced from the observation that the availability of new services potentially increases pre-existing interoperability issues since the same data is scattered over even more locations |
| Gurzick and Lutters (2009) | • Research motivated by the dynamics of how online communities evolve, and at the same time, the need for more novel synthesis of earlier online communities' guidelines, which are more descriptive, to prescriptive design theories to guide online community designs<br>• Comprehensive study of existing literature around online communities providing the first descriptive guidelines<br>• Each of the explored eight guidelines is motivated by theories |

|  |  |
|---|---|
| Koppenhagen et al. (2011) | • Initial awareness of class of problems arose from the current field observations in the e-procurement and e-marketplace space<br>• Research primarily motivated inductively by practical issues in the area of business-to-business collaboration, e-marketplaces, system interoperability, etc.<br>• Perceived timely issue and high relevance to practice, e.g., in terms of cost intensive system integration and lack of combining structured data with unstructured user activity |
| Zhang et al. (2011) | • Inductive motivation upon both literature reviews and industrial practice in the area of collections of digital information that demonstrates what a person knows and can do<br>• Open source and recent commercial ePortfolio offerings often lack effectiveness and efficiency due to design problems and missing incorporation of collaboration features<br>• Deduction from kernel theories that inform the definition of a ePortfolio design theory along the design science framework for ISDT (Walls et al., 1992) |

*Table 2. Deductive/Inductive initialization of related DSR works*

## 4.2 Design Principles Induction

Design Principles (DPs) are generic, high-level representations of the class of solutions addressing a class of problems. Other than requirements, DPs are not directly implementable characteristics of a potential solution or artefact, but describe a solution from its generic perspectives. DPs imply a great potential for innovative approaches and generalizations and are in their emergence a very important element which should contribute not only to a practical issue resolution, but also enrich and enhance the scientific knowledge. Researchers should seek for innovative design principles which even break common practices and theory assumptions and utilize cross discipline scientific knowledge to reach more advanced epistemological levels than possible in practical requirement realization in application development. The DSR control cycle of trial and error proposed by many DSR frameworks leads, when conducted rigorously, to the right conclusions in terms of valid, conditionally valid or not valid design principles to achieve certain goals. Researchers should be encouraged to suggest innovated design principles, but at the same time, make sure that a transparent relation between key requirements and design principles can be established and tested. DPs are one of the main transmitters between relevance and rigor in DSR projects. Without adequate DPs, a relevant, while at the same time rigorous, DSR would not be possible. Design principles induction starts with the deduction of requirements. Walls et al. (1992, p. 43) describe in great detail the branching of requirements from kernel theories as description of “[...] the class of goals to which the theory applies”. We focus here on the practice deduction of requirements. Requirements in IS research are defined as documented physical and functional needs that a particular product or service must fulfil. It is a statement that identifies a necessary attribute, capability, characteristic, or quality of a system for it to have value and utility to a user.

To distil adequate design principles we recommend to first cluster the array of requirements and then derive key requirements. This can be done via logical content aggregation in the research team and/or with iterative evaluation steps within the socio-technical environment using qualitative or quantitative data collection and analysis methods (e.g., observations, interviews, regression analysis). The final step is to combine, link and interpret key requirements to emergent design principles. For instance, in the DSR work of Koppenhagen et al. (2011) the research team clustered seven key requirements and combined/assigned them first to five design principles. After further iterations, the team realized that two DPs are higher level super-ordinates to two other DPs; consequently the team reduced the DPs to the remaining three which still satisfied all key requirements. The accurate derivation of DPs by informing theories can be retraced in the paper of Zhang et al. (2011): they motivate their DPs as independent variables in the hypothesis model by the kernel theory of communities of practice (CoP) (Lave and Wenger, 1991). Table 3 provides an overview of the design principle induction approaches of the four related DSR works.

| | *Design Principle Induction* |
|---|---|
| Gass et al. (2011) | • The explorative study was used to model a use-case, comprising the requirements<br>• Synthesis of seven design principles from the persona and use-case<br>• Motivated by the kernel theory of task technology fit (TTF); an impact on dependent variable end-user performance was assumed<br>• The design principles are considered as independent variables in hypothesis model TTF and are expected to influence user performance (not in original paper) |
| Gurzick and Lutters (2009) | • Synthesis, extension and detailing of preliminary design guidelines (requirements) along the DSR process of building and evaluating the artefact<br>• Final design guidelines can be interpreted as design principles according to the definitions above, e.g., fluid sense of purpose according to the changing needs of the community, online communities should be, at least in part, constructed by its members, etc. |
| Koppenhagen et al. (2011) | • Initial requirements deducted from qualitative pre-study with unstructured interviews<br>• The original requirements list have been condensed by combining several requirements to seven key requirements which have been initially abstracted to five design principles<br>• After multiple evaluations, the original five have been reduced to three super ordinate design principles: prevent document exchange, enable collaboration on networked/shared business object types, ensure seamless interaction flow |
| Zhang et al. (2011) | • Key-requirements (meta-requirements) have been assigned to ISDT levels and emerged along defined design methods, which are grounded in the respective process kernel theories<br>• Requirements (meta-design) have been derived and n:1 assigned to meta-requirements – therefore meta-requirements are collections or categorizations of meta-design elements; some examples of requirements are service-oriented application, central digital repository of multimedia, templates, galleries, and other resources, drag-and-drop implementation, etc.<br>• Key requirements (meta-requirements) can also be interpreted as dependent variables in their hypothesis model<br>• Design principles instantiated as independent variables in hypothesis model, motivated by the kernel theory of communities of practice (CoP) (Lave and Wenger, 1991)<br>• DPs proposed to have a positive effect to support the key requirements (meta-requirements) – in this respect the DPs elaborated are: knowledge synthesis and diffusion, annotation, blog connection and group collaboration |

*Table 3. Design principles induction of related DSR works*

## 4.3 Artefact Build

An artefact in the IS discipline is a tangible socio-technological instance in a DSR project which can be experienced, discussed, tested, evaluated, changed, improved, extended, etc. Artefacts appear in the IS discipline in various forms: e.g., models such as UML and class diagrams, software semi-products or prototypes, wireframes, visual designs or mock-ups. They imply the implicit nature of unfinished or premature which shall motivate further iterative enhancements or specifications. As an intermediate activity, design decisions (DDs) need to be made explicit as instantiations of design principles and therefore a one-to-many deduction from DPs to DDs needs to be possible and traceable. Of course a design decision can support multiple DPs but must support at least one. Types of DDs are, for example: use-cases, process instantiations, features, functions, architecture concepts and elements, user interface patterns and elements, system integration patterns, graphical representations, interaction patterns, etc. When building an artefact, the interrelation between artefact elements, for instance a specific user interface with all it facets (controls, colours, buttons, entry fields, output sections, etc.) should be traceable to design decisions and therefore indirectly relate to design principles. For the build process we suggest user-centric design (Davis, 1989; Constantine, 1999), agile development methodologies (Truex et al., 1999) with evolutionary/rapid prototyping methods (Crinnion, 1992), looping between persona definitions (in case of user-centric research, description of personalities using the artefact or later the respective application system), use-case definitions (business process, scenario, step sequence), storylines (central themes of usage, for instance, including user interaction steps), wireframe designs (e.g., manual UI drawings, sketches, interaction drafts), visual designs (e.g., static screen layouts of actual screens) and finally the development of an artefact concept version (ACV) and a artefact prototype version (APV). The latter ACV and APV are motivated by the highly iterative DSR approach and there are many arguments when and how to apply an artefact in the real world socio-technological target setting (e.g., Sein et al., 2011; Pries-Heje et al., 2008). We argue for an early intervention of an artefact version directly after the first iteration of creating an artefact based on defined design decisions and principles. For instance, when first wireframes or visual designs are built they should be applied, discussed and validated outside the research team in a setting which allows for an open, unbiased, early feedback. To further distinguish the levels of artefact maturity when applied in the social setting (e.g., test persons, business user, stakeholder, manager), we propose an artefact concept version (ACV) followed by an artefact prototype version (APV). The ACV is a static but interactive artefact version along one, narrow use-case to test the overall design concepts (UI elements, design decisions and principles, etc.) before conducting the next research stages. As DSR is perceived to be rather resource intensive, this helps to focus the efforts early in the right directions and to achieve reasonable research contributions at the end. The APV is an advanced artefact version which behaves like a software application system, is more flexible regarding the user interactions and focuses on the implementation of multiple use-cases. We also recommend building the APV already on the technical target platform and architecture to be able to measure the impact of those elements on the overall utility of the artefact too. Nevertheless, the APV is still restricted in terms of usage and applicability in multiple contexts such as holistic business scenarios, but the test and evaluation results bear great potential for validity and utility of design decisions and therefore design principles. Both ACV and APV are well applicable for qualitative and quantitative empirical evaluation methods to gather indications for the next steps in the actual DSR project and knowledge creation being applicable for generalization. Table 4 provides an overview on the artefact build activities of the four related DSR works.

| | *Artefact Build* |
|---|---|
| Gass et al. (2011) | • Design decision expressed as conceptual models of UML use-case, system architecture and data model<br>• First iteration artefact instantiated as fully functional web-based integration service<br>• The artefact was implemented as part of a scientific pre-study |
| Gurzick and Lutters (2009) | • Detailed description of all design decisions (design components) and mapping to design principles (design guidelines)<br>• First iteration artefact, a media-rich online community developed with the goal of stimulating teenagers to rethink their idea about learning and education - as much as possible created in accordance with existing preliminary guidelines<br>• Developed by a multidisciplinary team of researchers, with members from Psychology, History, Economics, Imaging Research and Information Systems |
| Koppenhagen et al. (2011) | • Transparent design decision process for architectural and data object concept to user interface design decisions; multiple iteration within research team, subject matter experts and the socio-technical context<br>• Used agile development methodology in the artefact build process of persona, use-cases, visual designs, concept modelling, ACV and APV<br>• Artefacts applied in various enterprises of high-tech and chemical industry with subject matter experts, designers, engineers, business professionals, IT personal for qualitative evaluation to proof design decisions and principles |
| Zhang et al. (2011) | • Along the defined kernel theories for design process and the applied design methods, the original artefact was extended by additional design decisions (features) of the enhanced artefact version<br>• Applied kernel theories in build/apply cycles of new artefact: emergent development, evolutionary prototyping and usability testing theory (Dumas and Redish, 1999) |

*Table 4. Artefact Build of related DSR works*

## 4.4 Evaluation

Many activities of the evaluation phase have been elaborated in various previous sections of this paper as mere ex-post artefact evaluation. However, current research suggests a combination of ex-ante (evaluation of design principles and partially design decisions) and ex-post evaluations (Pries-Heje et al., 2008). In addition, a continuous evaluation during activities to derive design principles and build the artefact seems most appropriate. For instance, to further reduce the risk of bias, multiple researchers should be involved in a DSR project, even separated into research teams; one team responsible for building and enhancing the artefact, one for the ongoing evaluation within the socio-technical context, e.g., with focus groups (Hevner and Chatterjee, 2010). While this approach is unquestionably resource intensive and is only applicable in larger DSR setups, it offers the potential to decrease the bias in evaluations using qualitative methods. Table 5 provides an overview of the evaluation activities of the four related DSR works.

| | *Evaluation* |
|---|---|
| Gass et al. (2001) | • Evaluation scenario was derived base on initially defined persona and use-case, for two design principles: seamless integration with web services and with local services in a lab-experiment with end-performance measurement for a particular set of tasks<br>• The different conditions included the availability or the absence of the integration service for the predefined tasks |
| Gurzick and Lutters (2009) | • Use the possibility of the artefact for collection of usage data and pre-/post surveys with design-oriented questions conducted online with the community members<br>• Results of usage data collections and surveys proved the validity of DDs and DPs |
| Koppenhagen et al. (2011) | • Early evaluation of design principles and decisions via persona descriptions, use-cases, concept models, visual designs, ACV and later (planned) APV<br>• Qualitative field studies for evaluations; data gathering via un-structured, semi-structured interviews, transcription and coding, semi-experimental setup, semi-quantitative pre-study |
| Zhang et al. (2011) | • Transparent evaluation, reflection and learning along the selected process kernel theories for design methods<br>• Test of design product as well as design process hypothesis with various statistical methods like linear regression, online surveys for pre- and post-test, scatter plots etc. |

*Table 5. Evaluation in related DSR works*

## 5 Conclusions

Our research showed that the identified critical activities in DSR are deeply embedded in the core of DSR frameworks and methodologies. They play a connecting role in the closed loop epistemological system of DSR. Furthermore, they have a strong impact on relevance and rigor of actual DSR project contributions. For instance, the four investigated DSR works cover those activities. However, from our studies we realize a great diversity in terms of the ontology used, and how, and to which extent those activities were conducted. For instance, the research initialization may be inductively motivated by informing practical problems, deductively from informing theories or from both. Nevertheless, we argue for the following common patterns: 1) the awareness of the deductive and inductive nature of activities and their proper execution is a central part of the epistemological process throughout the DSR cycles - independent of which DSR framework or methodologies are applied. 2) A coherent semantic flow from one to the next research activity is imperative. This can be supported by making the interrelation between predecessor to successor semantic elements very explicit - e.g., from requirements to key requirements, from key requirements to design principles, from design principles to design decisions, to the artefact and from the evaluation results to potential design theory hypothesis. 3) Evaluations should be conducted during many activities to verify requirements, key requirements, design principles, design decisions and artefact versions. 4) For the inductive as well as for the deductive steps qualitative or quantitative rigorous research methods must be applied for data gathering and analysis in context of the evaluations. 5) Agile development and prototyping methods can be used in the artefact build activities. They are in particular recommended for larger DSR projects which include many participants in different roles and responsibilities, in cases where continuous evaluations are intended and in the context of the utilization of specific DSR evaluation and development teams.

In order to be able to correctly assess the implications of our research, it is necessary to reflect our study's limitations: the sampling of the DSR projects, frameworks and methodologies were limited by the need to focus the elaboration on exemplary, but still few, examples. While results provide first indications, a larger sample would help to better understand relevant contingencies. Furthermore, we concentrated on critical activities in four core research phases and omitted, for instance, the important activities of generalization and potential design theory induction, as this would have significantly extended the length of the elaboration. We therefore would like to recommend those missing important activities for further research.

With our elaborations of critical DSR activities we intend to share our experiences in extensively conducting DSR, try to make tangible recommendations and provide guidelines how to efficiently execute design science research, in particular, to scholars who are rather unfamiliar to this important paradigm in the IS research field.

## References

Baskerville, R. (2008). What design science is not. European Journal of Information Systems, 17, 441-443.

Benbasat, I. and Zmud, R.W. (2003). The identity crisis within the discipline: Defining and communicating the discipline's core properties. MIS Quarterly, 27(2), 183-194.

Bostrom, R.P. and Heinen, J.S. (1977). MIS problems and failures: A socio-technical perspective: Part I: The causes. MIS Quarterly, 1(3), 17-32.

Constantine, L. (1999): Software for use: a practical guide to the models and methods of usagecentered design, Reading Mass.: Addison Wesley.

Crinnion, J. (1992). Evolutionary Systems Development (Software Science and Engineering), Springer.

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. MIS Quarterly, 13(3), 319-339.

Dumas, J.F. and J.C. Redish (1999). A Practical Guide to Usability Testing. Bristol, England, Intellect Books.

Gass, O. and Maedche, A. (2011). Enabling End-user-driven Data Interoperability – A Design Science Research Project. In Proceedings of the AMCIS 2011, Paper 221.

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

Gurzick, D. and Lutters, W.G. (2009). Towards a design theory for online communities. In Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology (DESRIST'09), 1-20.

Hevner, A. and Chatterjee, S. (2010). Design Research in Information Systems, Integrated Series in Information Systems 22, Springer Science&Business Media, 121-143.

Hevner, A. R., March, S. T., Park, J., and Ram, S., (2004). Design Science In Information Systems Research. MIS Quarterly, 28(1), 75-105.

Iivari, J. (2007). A paradigmatic analysis of Information Systems as a design science. Scandinavian Journal of Information Systems, 19(2), 39-64.

Koppenhagen, N., Katz, N., Mueller, B., and Maedche, A. (2011).  How Do Procurement Networks Become Social? Design Principles Evaluation in a Heterogeneous Environment of Structured and

Unstructured Interactions. In Proceedings of the Thirty Second International Conference on Information Systems, Shanghai, China, 1-19.

Kuechler, B. and Vaishnavi, V. (2008). On Theory Development in Design Science Research: Anatomy of a Research Project. In Vaishnavi, V. and Baskerville R. (Eds.), Proceedings of the Third International Conference on Design Science Research in Information Systems and Technology, Atlanta: Georgia State University, 1-15.

Langefors, B. (1966). Theoretical Analysis of Information Systems, Studentlitteratur, Sweden, Lund.

Lave, J. and Wenger E. (1991). Situated Learning: Legitimate Peripheral Participation, Cambridge: Cambridge University Press.

Lundeberg, M., Goldkuhl, G., and Nilsson, A. (1978). Systemering, Studentlitteratur, Sweden, Lund.

Ma, M. (2005). IT Design for Sustaining Virtual Communities: An Identity-Based Approach. Doctoral Thesis. University of Maryland.

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

Mumford, E (1983). Designing Human Systems for New Technology, The ETHICS method, Manchester Business School, Manchester.

Nunamaker, J.F., Chen, M., and Purdin, T.D.M., (1990). Systems development in information systems research. Journal of Management Information Systems, 7(3), 89-106.

Orlikowski, W.J. and Iacono, C.S. (2001). Research commentary: Desperately seeking the ”IT” in IT research – A call theorizing the IT artefact”, Information Systems Research, 12(2), 121-134.

Orlikowski, W.J., and Baroudi, J.J. (1991). Studying Information Technology in Organizations: Research Approaches and Assumptions. Information Systems Research, 2(1), 1-28.

Peffers, K., Tuunanen, T., Rothenberger, M.A., and Chatterjee, S. (2008). A Design Science Research Methodology for Information Systems Research, Journal of Management Information Systems, 24(3), 45-78.

Pries-Heje, J., Baskerville, R., and Venable, J.R. (2008). Strategies for Design Science Research Evaluation. Proceedings of the 16th European Conference on Information Systems, Paper 87.

Sein, M. K., Henfridsson, O., Purao, S., Rossi, M., and Lindgren, R., (2011). Action Design Research. MIS Quarterly, 35(1), 37-56.

Simon, H. A. (1996). The Science of Design: Creating the Artificial. In The Science of the Artificial (3rd ed.). Cambridge, Mass.: MIT Press, 111-138.

Sundgren, B. (1973). An Infological Approach to Data Bases, Ph.D. Diss., Skriftserie utgiven an statistiska centralbyrån, Nummer 7, Statistiska Centralbyrån, Stockholm.

Takeda, H., Veerkamp, P., Tomiyama, T., and Yoshikawam, H. (1990). Modeling design processes.

Tomiyama, T., and Yoshikawa, H. (1987). Extended General Design Theory. In Design Theory for CAD, Proceedings of the IFIP Working Group 5.2 Working Conference, Yoshikawa, H. and E. A. Warman, E.A: (Eds.), 95–124. Amsterdam: North-Holland.

Truex, D., Baskerville, R., and Klein, H. (1999). Growing systems in emergent organizations. Communications of the ACM, 42, 117-123.

Walls, J., Widmeyer, G., and El Sawy, O., (1992). Building an information system design theory for vigilant EIS. Information Systems Research, 3(1), 36–59.

Yoshikawa, H. (1981). General Design Theory and a CAD System. In Man-Machine Communication in CAD/CAM, Proceedings of the IFIP Working Group 5.2–5.3 Working Conference 1980 (Tokyo), Sata, T. and Warman, E.A. (Eds), Amsterdam: North-Holland, 35-53.

Zhang, X., Olfman, L., and Firpo, D. (2011). An Information Systems Design Theory for Collaborative ePortfolio Systems. In Proceedings of System Sciences (HICSS), 2011 44th Hawaii International Conference, 1-10.