Opening up design science:
The challenge of designing for reuse and joint development

Georg von Krogh and Stefan Haefliger[1]

ETH Zurich, Kreuzplatz 5, 8032 Zurich, Switzerland

gvkrogh@ethz.ch, shaefliger@ethz.ch

September 26, 2010

Abstract

The purpose of this paper is to advance design science by developing a framework for research on reuse and the relationship between external IT artifacts and their users. A design science approach to IS research needs to grapple with the fact that a number of relevant, economically attractive, external IT artifacts cannot be designed from scratch nor meaningfully evaluated based on the current state of development, and so design science research will struggle with incomplete cycles of design, relevance, and rigor. We suggest a strategic research agenda that integrates the design of the relationship between an external IT artifact and the user by considering the impact artifacts exert on users. Three dimensions derived from adaptive structuration theory inform our framework on three levels of design granularity (middle management, top management, and entrepreneur): *agenda* considers the dynamic properties of technological objects, *adaptability* refers to the functional affordance of external artifacts in development, and *auspice* captures the symbolic expression and scope for interpretation. We derive implications for research design.

Keywords: Design Science, IT Artifact, Open Source Software, Adaptive Structuration, Inter-organizational Information Systems, Strategic Management of IT

[1] This project received support from the Swiss National Science Foundation (grant 100014_125513).

Introduction

Information systems (IS) pervade everyday organizational life. Managers and IS professionals build and evaluate IT artifacts, such as vocabulary, symbols, models, algorithms, procedures, and instantiations, tailored to organizational needs in order to solve problems that, until now, could not be addressed by information technology. The design science approach in IS established rigorous research guidelines that foster contributions to the problem-oriented, innovative, and effective creation, deployment, and evaluation of IT artifacts in organizations (Hevner et al., 2004; March and Storey, 2008; March and Smith, 1995). However, a growing number of IT artifacts are not only created outside the organization, they also extend beyond the organization in terms of both complexity and dynamics (Elbanna, 2010). As a result, any one designer's ability to fully understand and influence overall development remains limited. Working with systems and environments such as GNU Linux, Apache, or Mozilla, to name just a few of the largest and most popular open source (OS) families of programs, managers and designers face the challenge of using external IT artifacts—existing artifacts developed outside their organization. This design activity that encompasses relating to external IT artifacts is only partially understood: as reuse across organizations (Ravichandran and Rothenberger, 2003; Haefliger et al., 2008) and as community relations entertained by firms (Shah, 2006; Dahlander, 2007).

The emerging literature on reuse of external IT artifacts considers search and adaptation efforts (Bonaccorsi et al., 2006; Majchrzak et al., 2006; West, 2003), whereas the literature on community relations emphasizes evaluation and sharing of IT artifacts (Henkel, 2006; Dahlander and Wallin, 2007; Dahlander, 2007; Stuermer et al., 2009). These design activities, while effective in tackling the challenge of dealing with external artifacts, partially ignore the systematic context difference between the external developers and the designers and users within the adopting organization. We currently lack a comprehensive framework that could inform design science research on the use of externally developed IT artifacts, in particular adaptation and evaluation. Crucially, use occurs in a different context from development and designers must be made aware of the effects external IT artifacts can have on use within the organization (Ciborra, 1998). Starting with the search for an IT artifact and problem formulation all the way through the adoption, development and internal evaluation, understanding the effects of use and context, that may limit "degrees of freedom" in design, has become a priority for IS researchers and practitioners. A strategic perspective reinforces the urgency because successful information systems (e.g. for knowledge management) rely on accessible and well integrated IT artifacts (Butler et al., 2008; Massey et al., 2002), and integration refers to the everyday context of use in an organization.

The purpose of this paper is to advance design science by developing a framework for research on reuse and the relationship between external IT artifacts and their users. We seek to advance avenues for future research on IS through the design science approach by formulating and grounding a set of research questions. In the next section, we briefly introduce the literature on the design of IT artifacts, and show the importance of a research thrust on designing and relating to externally designed IT artifacts. Next, we develop a framework of research questions to guide future work in this particular area. Before concluding, we discuss the implications of our framework for research design and for the role and focus of the design science researcher.

Design and relate: an overview of the literature

Design science was originally conceived within engineering and computer science and aimed at problem solving in these areas (Simon, 1996). Today, design science is pervasive in several academic disciplines that build artifacts, such as mechanical or medical engineering, biotechnology, construction engineering, and architecture. In IS, design science evolved into a coherent body of theory and research on design and action (Gregor, 2006). It opened vast opportunities for predicting and observing the interaction between researchers, designers, users, organizations, and the evolving artifact (Cross, 2007; Markus, et al. 2002; Hevner et al., 2004; Banker and Kauffman, 2004; Gregor, 2006). Hevner and colleagues (2004) developed a foundational design science approach to IS research consisting of two activities, the initial development of artifacts and their subsequent justification and evaluation. They based their study on business needs originating with people, organizations, and technology, as well as theoretical foundations and research methods. More specifically, Hevner (2007) posited three cycles in design science: the relevance cycle that connects design science research and the problem environment through the specification of requirements and field testing; the design cycle that connects building and evaluating artifacts; and the rigor cycle that connects design science research and developing knowledge bases. Hevner et al. (2004) distilled the practical aspects of design science into seven pivotal IS guidelines: 1) create an artifact that addresses an organizational problem; 2) ensure the problem is relevant to business; 3) evaluate the utility of the design in view of the needs or problems it is created to address; 4) contribute to academic and practical knowledge through the new artifact, methods, or foundations; 5) use rigorous methods when creating and evaluating the artifact; 6) search for an effective artifact using available means to reach a desired end, within the (legal) constrains set by the problem environment; 7) communicate design outcomes to managers and academics.

As these guidelines show, one undisputed advantage of the design science approach is the intertwined nature of artifact design and the process of researching it. The distinctions between "research and design" or "observing and doing" become increasingly blurred, to the potential benefit of practice and academia alike. Design science helps researchers and managers engage in constructive dialogues: researchers to identify the most relevant and pressing research problems, and the academic IS discipline to contribute to practically useful knowledge, novel theories, and tested methodologies (Hevner, 2007). Fundamentally, design science frees the IS field of excessive technological determinism, or the simplistic view that technology is determined by rules or laws beyond human control (Hickman, 1998). It also helps IS researchers to add "truth value" to artifacts and recommendations by specifying their effectiveness and efficiency in specific situations (Iivari, 2007: 46-47). An important assumption for design science to work, however, is that context along the dimensions of people, organization, and technology is known, potentially understood or, to some limited extent, controllable by the researcher, much like an attempt to identify and unilaterally control a complex set of variables in quasi-experiments. Inside firms, the design science approach to IS research still very much relies on a notion of a cyclical process that starts with problem formulation and ends with successful implementation (Hevner, 2007; March and Storey, 2008). As Hevner suggested (2007: 89), "Good design science research often begins by identifying and representing opportunities and problems in an actual application environment." Hevner then proceeds to clarify how the application context not only provides requirements for research, but also specifies acceptance criteria for the final evaluation of the research outcome. Under such conditions, design science is rational, rigorous, and useful. Yet, with the advent of external IT artifacts—where collaborative development across organizational boundaries engages widely distributed populations of designers and users and integrates a large variety of technologies—new forms and contingencies raise an important challenge to conventional design science in IS: the context defined along multiple dimensions becomes increasingly dynamic and problematic to identify, understand, and control unilaterally. Garud, Jain, and Tuertscher (2008) alluded to this challenge when elaborating on the design approach taken by developers of collaborative efforts such as Wikipedia and GNU Linux. What the authors call "designing for incompleteness" is a cycle of evolving artifact designs that opens up questions and options for re-design.

Today, there is a growing environment populated with large and complex IT artifacts that have been in development over many years (the GNU Linux operating system, for example, began development in 1991) involving communities of thousands of individuals and organizations worldwide (von Hippel and von Krogh, 2003). The artifacts are often instantiations, (e.g. OS software licensed code), but may equally well include a vast repertoire of alternative algorithms, symbols, design methods, vocabulary, or procedures. In the light of this phenomenon, we propose two important issues to be considered for design science: reuse and community relations. First, the global search space for OS-related artifacts is virtually unconstrained. Although software repositories such as Freshmeat or SourceForge (which list more than 120,000 projects and 1.2 million developers producing a host of IT artifacts) aid firms and users conduct searches for effective artifacts, the information problem is overwhelming: depending on the constraints set by the problem environment, one does not know if an even more effective artifact could be available "out there," or if some individual or firm is working on solving the exact same problem at this very moment. Haefliger et al. (2008) found that OS software developers sometimes use software repositories for searches, but that they rely more strongly on their contacts with other trusted developers when identifying effective and efficient artifacts. It should also be noted that the reuse literature in IS focuses on cost savings through internal reuse programs (Frakes and Isoda, 1994; Kim and Stohr, 1998; Cybulski et al., 1998; Lynex and Layzell, 1998) without formulating methods to decide which external IT artifacts to reuse in order to design new or better information systems.

Second, current and future individuals and organizations involved in the design of the external IT artifact may be unknown to the firm, its designers, and users, as will their problems and business needs. Yet, an increasing number of managers and designers in firms see great promise in OS-related IT artifacts developed by global communities, and have started to adapt these to internal needs, contribute to their collective development, and build relationships with the communities (Bonaccorsi et al., 2006; Henkel, 2006; Dahlander and Wallin, 2006). Additionally, a design science approach to IS research needs to answer the question of how best to relate to the community to jointly build new and effective IT artifacts. OS software and related artifacts often evolve without a clear "roadmap," and the boundaries of their future functionality may be highly uncertain. For example, in very large systems, such as the Debian GNU Linux Distribution, continuously changing groups of committers with privileged access to the software development add or remove software components at a high pace. Their decisions to remove or add components are often based on the intensity of certain components' use in the community (Maillart et al., 2008). Any single organization is but one (possibly important) contributor to the community's project. Given the importance of the nature of each contribution to the collective design agenda, design science needs methods to determine what to contribute for joint development.

One response to these and other challenges would be to limit the application of design science to the stable and well-defined organizational context where it first originated, and then create new theories and research approaches for these novel and evolving phenomena. We believe this would be the wrong response, as it would fail to realize the strong potential that design science in IS holds for the phenomenon of OS software. Rather, we argue that designing the relationship between the artifact and the user is a vital, if not central, aspect of design when external IT artifacts are developed by global communities. This calls for a new research framework with a set of questions to be tackled by design science researchers. We turn to this point next.

Toward a new research framework for design science

A design science approach to IS research needs to grapple with the fact that a number of relevant external IT artifacts cannot be designed from scratch nor meaningfully evaluated based on the current state of development, and so design science research will struggle with incomplete cycles of design, relevance, and rigor. Three observations lead to a new research agenda that integrates the design of the relationship between an external IT artifact and the user.

1. Economically attractive, complex, and large IT artifacts are publicly available and can be used for free.

2. Their complexity may prevent members of one organization from fully knowing or understanding the context of the artifact, including the dimensions of people, organization, and technology involved in its design.
3. The artifact is being developed by a global community, which may or may not be open to artifact modifications and suggestions for improvement as part of the design process. The sustainable development and maintenance of the artifact remain outside the full control of any one organization.

These observations do not imply that contributors to collective innovation processes are not open to influence, but that joining or even influencing the development agenda can be difficult to achieve in a manner satisfactory to the user (O'Mahony and Ferraro, 2007; Spaeth et al., 2008). The first observation justifies economic interest in external IT artifacts. The second points to the challenge of reuse because, with only a partial understanding of context, reuse decisions become complex design matters. The third observation draws attention to community organization (Sawhney and Prandelli, 2001; Lee and Cole, 2003) and, thus, the importance of community relations by anyone wishing to invest in, use, and control external IT artifacts.

Reuse and community relations are positions on two dimensions that need to be taken into consideration when designing the relationship between an external artifact and the user. Reuse can mean adopt-as-is or adapt-to-fit, a choice with respective trade-offs. A purely reactive approach to reuse (adopt-as-is) may lead to unsatisfactory results in terms of fit with organizational needs. A focus on adaptation (adapt-to-fit), on the other hand, may miss the advantages of external maintenance and future improvements. The approach to community relations implies similar trade-offs: founding and nurturing a community is costly but rewarding in terms of influence over the artifact development agenda (Spaeth et al., 2010), whereas remote participation may yield no influence at all. Community relations can mean foundation or sponsorship by a firm—such as IBM's decision to release the Eclipse software development platform under an OS license and help build a foundation to manage it—or remote participation, evidenced in the roughly 170 member organizations contributing to the Eclipse platform. A statement in a recent keynote address by Dan Frye, Vice President of OS software at IBM, shows how delicate this trade-off is for Eclipse:

> "For IBM, one of the hardest lessons it had to learn was one about control. Mainly, there is none. There is nothing that we can do to control individuals or communities, and if you try, you make things worse. What you need is influence. It goes back to the most important lesson, which is to give back to the community and develop expertise. You'll find that if your developers are working with a community, that over time they'll develop influence and that influence will allow you to get things done." (Quoted in Kerner, 2010.)

While the trade-off may prevent firms and other users from engaging in and developing community relations, research by Joachim Henkel (2006) shows that reciprocity in community relations can offer benefits for the participant firm, such as external bug fixes and software improvements, and an enhanced technical reputation.

The trade-offs apparent in reuse and community relations are contingent upon the way designers and users in organizations understand and relate to external contexts, technologies, and community organization. Existing artifacts are known to impact users (Orlikowski, 1992; DeSanctis and Poole, 1994)—an insight that may be fruitfully combined with a design science approach that views artifacts as the creative outcome resulting from a precise understanding of the gaps between the current and desired states of organizational information systems. We propose that a future design science approach should consider more closely the relationship between a given and adaptable artifact and the user. Adaptive structuration theory (AST) is particularly useful to help devise a framework toward this end (DeSanctis and Poole, 1994; Markus and Silver, 2008).[2] AST builds on the work by sociologist Anthony Giddens, and originates from an effort to combine various institutional and decision making theories in prior IS research. AST examines technological and organizational change from the types of structure offered by advanced technologies, and the structures that emerge in human action as people interact with these technologies. The theory is relevant to our argument because it aims at improving the design and implementation of new technologies (DeSanctis and Poole, 1994). Correspondingly, we suggest the effects of IS on users should be considered along the three dimensions identified by Markus

2 March and Smith (1995) suggested that adaptive structuration theory is a prominent example of an approach to theorizing about the "why" and "how" effects in IS design. For example, an issue that connects the two theories is why particular artifacts (constructs, models, methods, and instantiations) work. It should be noted, however, that Markus et al. (2002) elected to talk about "design theory" when describing knowledge management systems linking theory and instantiations, rather than design science, which focuses more on design as an activity. For more on this, see Gregor (2006).

and Silver (2008) in their later critique and advancement of AST: technological object, functional affordances, and symbolic expression.

Acknowledging that a global community may continuously develop artifacts, the dimensions of AST take on a dynamic aspect ranging from current to future activities and states, which needs to be considered by designers. The characteristics of the technological object can be subsumed under "agenda," consisting of realized, planned, or evolving technology. The functional affordances can be termed "adaptability," since they result from an informed perception of functions by designers ready to invest considerable resources to adapt a given functionality. Third, "auspice" is the promise of symbolic expression that covers the interpretative scope of designers and users who choose to work with a "foreign" external IT artifact developed by a global community. In ancient Rome, an auspice referred to patterns as "signs from the gods," such as a flock of bird or the appetite of chickens, interpreted by an augur. Analogously, designers read patterns in the development of an evolving complex artifact as well as the many weak and strong signals emerging from the behavior of the community. Patterns in development might include software components removed from or added to the OS software repository, implementations of algorithms, choice of programming language, API specifications, waiting times for bug fixing, release histories, or evolving documentation of code. The behavior of the community may include messages posted to mailing lists, thread lengths of discussions, the developers or users who choose to participate in certain discussions, the types of mailing list used (e.g. technical topics versus general lists), frequently asked questions, helping behavior, etc. The community often cultivates and communicates specific values that might contradict organizational values and otherwise interfere with working routines, rhythms, and interaction patterns in the organization. Within this structuring process, the role of the designer can be considered at different levels of granularity: from middle management to top management, and from an industry perspective, where designers are entrepreneurs (Steyaert, 2007) using given artifacts and combining them with their own perception of markets, user needs, and technological visions. Table 1 presents a number of critical design issues that risk remaining invisible when ignoring the relationship between external IT artifacts and users.

	Design granularity		
	Middle management (group level)	Top management (firm level)	Entrepreneur (industry level)
Agenda (dynamic properties of technological objects)	Design the adoption: How can global community dynamics be understood to anticipate relevant technologcial changes?	Design a reuse strategy: How do external artifacts complement the firm's sourcing strategy?	Design new business models: How do new technological developments give rise to new markets?
Adaptability (functional affordance of external artifact in development)	Design the contribution: How can a match between the external artifact and internal user needs be achieved? How to reliably evaluate security, reliability, sustainability?	Design community relations: How can future internal needs be matched with current functions identified?	Design for market demand: How can functional needs be matched with market demand?
Auspice (symbolic expression and scope for interpretation)	Design for goodwill: How can the usefullness be communicated and explored given diverse internal needs? How to evaluate and deal with incoherences in the system?	Design for openness: How can we overcome the NIH syndrome and, e.g. promote acceptance of FLOSS, or diversity in IS? How to foster creativity?	Design an environment for collaboration: How can community work translate to entrepreneurial activity and lead to spin-offs?

Table 1: Understanding the relationship between the external IT artifact and the user: a research framework for advancing design science in IS

Agenda. As designers, middle management need to anticipate the planned, realized, and evolving characteristics of the external IT artifact, to understand and evaluate its potential utility for their organizational problem, and ultimately for the business relevance of the artifact. In a global community the number of components that constitute the artifact may grow exponentially, satisfying some user's personal needs (von Hippel, 2001). In the process, some components may become increasingly peripheral, whereas others may take on an important role in the overall design, channeling the efforts of a greater part of the community toward upgrading and ensuring compatibility with other components. Nokia's development of the OS software platform Maemo is a case in point. While the platform includes OS software like GNU Linux, GTK, and Gnome, in addition to many other user developed instantiations (e.g. mapping software or star gazer), Nokia has chosen to keep some software proprietary, including user experience-related software components. However, for Nokia, the design of an optimal user experience needs to be kept consistent with the functionality of these evolving artifacts. Thus, Nokia's designers are required to interact carefully with, learn about, and monitor the realized, planned, and evolving technology development and other activities of the community.

In a context where the dimensions of people, organization, and technology are in flux, the organization's overall strategic direction is an important consideration for researchers who do IS design science. A changing context can alter organizational problems, undermine their business relevance, and diminish the utility of a design. The development agenda also includes top management designing an overall reuse strategy aimed at externally developed IT artifacts, and asks how the realized, planned, and evolving artifact complements other internal strategic assets (artifacts) and their sourcing. The reuse strategy defines a scope for design and action throughout the organization, and provides direction for middle-level managers and other designers and users to identify relevant communities and technologies. Moreover, the reuse strategy also defines how externally and internally developed artifacts relate, including make-or-buy decisions at the level of artifacts, such as software components and other instantiations. Ravichandran and Rothenberger (2003) discuss this for component markets, where the difference is that components are commercial and protected by copyright.

At the industry level, the development agenda may inspire the design of new business models. A study of new entrants in the Italian software industry (Bonaccorsi et al., 2006) finds that entrepreneurship emerges on the fringes of OS software development, giving rise to new markets for products and services. Here, users design business models that involve community support, packaging, combining, selling artifacts, or providing services to other OS software users (see, for example, Fitzgerald 2006). Interestingly, the fact that artifacts rapidly evolve is in itself an entrepreneurial opportunity. Examples of entrepreneurial firms that developed new business models based on OS software include Red Hat and Suse. In a world where Linux and related software may contain numerous releases, Red Hat adds value to the user of OS software by securing coherent and updated versioning of the software (Red Hat Linux) in combination with other artifacts. To summarize the discussion this far, the overarching research question regarding the effect of the agenda is: how can the link between existing technology and organizational needs be designed so that the use of the artifact can sustain, evolve, and grow in conjunction with the external development?

Adaptability. Adaptability requires an understanding of the potential match between the user's requirements and the functional affordances, defined as "the possibilities for goal-oriented action afforded to specified user groups by technical objects" (Markus and Silver, 2008: 622). From a dynamic perspective, middle managers are challenged to find a match and design an adaptation strategy that takes into account the specific needs of the organization regarding scale, security, reliability, stability, and more. Here, make-or-buy decisions as part of the adaptation strategy become an important object of research. What are the costs and benefits involved in adapting an external IT artifact, and at what point do net benefits of adaptation offset the cost of developing a targeted artifact inside the firm?

For top management, these dynamics imply the choice of a community of external developers or the foundation and nurturing of sub-communities to design sustainable collaboration with competitors and volunteers (Spaeth et al., 2010). Top management involvement is essential here because engagement with the community to safeguard functional affordances represents potential risks including loss of reputation, loss of trade secrets, spillover of technically sensitive information, the use of internal labor for external IT artifact development of no relevance to organizational problems, unintended or intended breach of commercial and OS software licenses, etc. While prior work has suggested investigating the roles and activities of top management by design science methods (Van Aken, 2004), little is known about their involvement in design science and how the reuse of external IT artifacts opens a new research space.

An entrepreneur's challenge is to identify and build the match between the functional affordances of the existing artifact and a new segment of users willing to pay for a specific service or complementary product. The entrepreneur holds or creates knowledge about the evolving artifacts, their goal orientation, and the existing and potential needs of users. Through involvement in the design, the entrepreneur learns about the affordances of the IT artifact and starts to relate it to new users (Steyaert, 2007). This process may result in new matches between functionality and existing demand, but it may also, through a succession of commitments, result in the creation of entirely new markets (Sarasvathy and Dew, 2005). As Sarasvathy and Dew (2005: 559) put it: "Entrepreneurs do not 'leave it' to differences in tastes or behavior to build markets. They work very hard to make tastes cohere and to concurrently embody them into particular transformations in real artifacts." The overarching research question regarding the adaptability is: how can the functional affordances of the artifact be identified and (or) created in order to meet the needs of users when both technology and requirements continually evolve?

Auspice. The symbolic expression of an IT artifact corresponds to a dynamic scope for interpretation. Auspice is a promise in need of design. Management is challenged to communicate the usefulness of a need and convey a sense of motivation among users that triggers their creativity. For example, Hevner (2007) highlights the importance of creativity and flow in problem solving for a successful design. Yet, the use of an evolving, external IT artifact is fraught with difficulties: it is "not invented here," it may come with bugs, it may be poorly documented, it does not perfectly fit internal routines and practices, it may be abandoned by the original developers, and its inner workings may never be fully understood by users and designers in the organization. Appropriation by internal designers and users, their acceptance and enthusiasm for integrating the artifact with the information architecture of the organization, can be designed, and our framework specifies the issues involved. Middle management designs for goodwill toward a specific external IT artifact. The successful use of IT artifacts depends on organizational changes in routines and business processes (Brynjolfsson and Hitt, 1996), which include complex relationships between management levels and organizational units (Mata et al., 1995). Auspice describes the relationship between the external artifact and the user, the scope for interpretation and, hence, communication and flexibility in promoting the artifact's advantages. At the middle-management level, auspice also involves transforming internal processes to accommodate for and actively use the external IT artifact. These middle management tasks can be daunting, as any external IT artifact may be viewed with skepticism and mistrust, based on justified criticism.

For top management, designing openness becomes a strategic issue: why should the firm adopt external IT artifacts at all? Previous research has shown that IT assimilation crucially depends on top management (Armstrong and Sambamurthy, 1999) prioritizing and carrying out specific organizational initiatives. Top management may indeed have significant influence over whether external IT artifacts are generally positively received, and future research needs to substantiate the most effective strategies. In the case of Nokia's Maemo platform, significant communication was needed by top management on the direction of the company's collaboration with the community of volunteer developers contributing a variety of artifacts. A good starting point for research could be some of the world's largest technology companies, which frequently interact with and contribute to OS software communities. For example, companies such as Red Hat, IBM, Novell, Intel, and Oracle are top contributors to the Linux kernel: in 2009, there were 240 companies contributing overall.[3] The fact that some of these companies have been working with OS communities for many years seems to indicate that they value community relations highly and that they have found ways to communicate the value of openness to internal users and designers.

From an entrepreneurial perspective, understanding the symbolic expression of an artifact could translate into designing an environment for collaboration and overcoming the little understood hurdle of motivating distributed individuals to share or contribute to a new business. While it is clear that many entrepreneurial firms have emerged on the fringes of the OS software phenomenon as discussed above (we also see software vendors being formed around the Maemo platform), a large research gap exists concerning the community's engagement in the formation of these business. The emergence of entrepreneurs may indeed create an imbalance in the incentive structure that safeguards OS development. Some voluntary users and designers who do not expect to benefit economically, at the same levels as entrepreneurs, may defect from the development effort, which could lead to an undersupply of the external IT artifact (see discussion by Young and Rohm, 1999, on establishing Red Hat and the sensitivity to voluntary Linux developers). The overarching question regarding the auspice is: how can the promise of using an external IT artifact be communicated and promoted in order to enable efficiency and foster creativity?

Table 1 summarizes a host of design tasks that are relevant, but neither obvious nor easily tackled by existing approaches to design science in IS. A focus on the relationship between artifact and users can broaden the scope of the design science approach considerably and enrich research by incorporating theory from other fields, such as structuration theory, motivation theory (flow, cognitive dissonance, self-determination), private-collective innovation theory, or behavioral economics (reciprocity, fairness, altruism). If the technology is developed outside the influence of any one organization, and if both technical characteristics and internal needs continue to evolve, the great

3 For recent statistics see lwn.net or for a comprehensive update for 2009 see Kroah-Hartman et al. (2009).

promise of a design approach lies in the carefully structured, rigorous, and repeated questioning, understanding, and designing of the relationship between the artifact and the user.

Implications for research designs

In the following, we discuss the implications of our framework for design science research and the researcher to indicate potential avenues for future work. The result of design science research is the production of an artifact such as a construct, model, method, or instantiation. Thus, an important strength of the design science approach in IS is its close link to action and practice (March and Smith, 1995; Hevner et al., 2004; Hevner, 2007; Sein et al., 2010). Similarly, in the framework we propose, the result is the artifact that emerges from a collaboration where the design approach includes not only the artifact itself, but also the relationship between designers and users (spread out across several communities or firms). This broader focus implies extensions to the design science research process in IS.

Recall that design science consists of cycles of activities to foster academic and practical knowledge; the design cycle (build and evaluate), the relevance cycle (specify requirements, field tests), and the rigor cycle (grounding, additions to knowledge base). The framework developed in the last section raises the question of the location of these cycles and the role of the researcher within them. If the problem environment for one artifact consists of multiple and changing people, organizations, and technologies, the focus of design science and the position of the researcher become ambiguous. We cannot fix a general position but need to consider a space with multiple dimensions and multiple perspectives from which an artifact is external. Design science, then, can be understood along a vector in a three-dimensional space where its direction captures the focus of research (the user as researcher, the community, and the firm) and its scope captures the extent to which research cycles include the relationship between external artifact and user. We use the term “vector” because individual researchers can choose to focus to a greater or lesser degree on one or more players and their relationships in the overall research context.

First, let us consider direction. The orientation of design science research can be towards the researcher as user. An important motivation for OS software is for users to solve their own problems, through the building and evaluation of an artifact (von Hippel, 2001). Here, the artifact can be built from scratch or reused from the wide offerings in an OS environment. A prerequisite, however, is for researchers to build and evaluate an artifact with future potential reuse in mind. In many engineering fields, researchers build IT artifacts to solve critical scientific problems, and release them with the publications of their research results. In geography, for example, researchers have built numerous OS software components for geospatial applications. The purpose of these systems ranges from helping researchers to do more accurate mapping and tracking of geophysical changes, to better analysis through geometric algorithms (Steiniger and Bocher, 2009). So, too, in the field of economics, where authors have argued for the need for the discipline to build and evaluate new OS IT artifacts such as software packages for more effective statistic analysis (Yalta and Yalta, 2010). When researchers share these artifacts in the way the examples show, it may help advance a discipline more efficiently and effectively.

Design science research can be oriented toward a community that engages in collective building and evaluation of reusable artifacts. Researching OS software communities is nothing new, of course, but following the principal cycles of design science (Hevner, 2007) researchers are more than passive observers—they must join the community, build and evaluate through participating in discussion forums, testing software, reporting bugs, fixing bugs, documenting, writing code, coordinating projects, and so on. The focus of design science research would be on the extent to which reusable artifacts that fit with the emerging needs and problems of the community can be built. To our knowledge, this type of research design is currently almost non-existent (an exception is Bodker et al., 2007), but we think it will be instrumental for the development of our academic and practical understanding of OS communities. For example, community oriented design science will allow us to examine in detail a number of context-sensitive issues linked to the building and evaluation of an artifact. For example: if, why, and how do small changes to an artifact initiate supportive or negative reactions from the community? When and how does the community perform search? What is the form or content of discussions closed to people outside the small circle of OS developers? What changes in an artifact relate to or emerge from the ongoing discussion on project developer lists?

Finally, design science research can be oriented toward the firm as the user organization. Here researchers place themselves within the organization, jointly searching for and evaluating communities and external IT artifacts for reuse. Design science has contributed significantly to information systems research in this domain; their contributions have strategic implications for how to improve working environments in organizations and leverage information systems for business purposes. Consequently, it is here where our framework most obviously builds on prior work and extends the design science approach to include the relationship to external IT artifacts. We are not aware of scholars using the three design science research cycles in conducting research that takes into account the impact of the dynamic properties of external IT artifacts under development, of their functional affordance, and of their symbolic expression. However, Hevner (2007) sees parallels between action research (e.g. Susman and Evered, 1978) and design science research, and suggests that both could benefit from each other. Very recently, Sein and colleagues (2011) developed a methodology that combines action research and design science research by building on their own action research and on prior work in design science that explicitly takes into account the organizational context of the design process (Gregor and Jones, 2007; Orlikowski and Iacono, 2001; Orlikowski and Scott, 2008). There are several studies in IS that report having used an action research methodology concerning the implementation of OS software in public organizations, developing economies, and the educational system. For example, Braa and Hedberg (2002) report on an action research program to develop a health information system based on OS software for African countries, including a number of challenges such as acceptance of the new systems by local users. However, many of these studies fall prey to a common criticism that action research tends to favor strong relevance over academic rigor. An exception here is Fitzgerald and Kenny (2004) who provide a rigorous account of OS software implementation in a large Irish hospital. Their study identifies challenges such as the shift of mindset needed to work with OS software, as well as the resistance among staff who felt their expertise was threatened as the hospital abandoned popular commercial software. Yet, while the study pays significant attention to the change resulting from the OS software implementation, reporting case-based findings of high relevance to the IS discipline, it is not a direct application of a design science research approach. In particular, it does not place the researcher in the focal position of external IT artifact design and reuse. This will need to be the orientation of future design science research.

Second, the scope of the vector represents the building and evaluation of the artifact itself, dealing with the extent to which researchers take into account relationships with external IT artifacts. It is here that our framework suggests specific questions for design science research, specifying methods to decide on the fundamental trade-offs in reuse as well as community relations. While the application of our framework is not limited to firms or communities, the questions are most fruitfully investigated in a context where more than one individual (the researcher) is affected by external IT artifacts and organizational IS challenges can be approached with the relationship explicitly considered.

As an iterative process, the cycles imply that not only the external artifact underlies a build and evaluation process but also the nature of the relationship with the community and the design efforts that can be controlled by the internal designers. The search for a reuse target is a process that compares expected search costs with estimated costs of designing in-house: only if the efforts to design an artifact by internal designers are considered to be higher than the potential identification of a reusable artifact will search be undertaken (Haefliger et al., 2008). All three dimensions of our framework should be considered when making decisions about reuse and community relations. First, regarding *agenda,* should search focus on currently available artifacts or include, or even focus on, technological roadmaps and development agendas set by communities? Any candidate artifact needs to be considered in terms of its relationship with internal designers and users. This means that the estimated adoption effort depends on current and likely future needs. The design of a search process is already known to be fraught with "uncontrollable forces in the environment" (Hevner et al., 2004: 88), without taking into consideration the relationship between an externally available artifact and internal designers and users at a given time. The resulting complexity may well overtax pre-defined search procedures and call for new search heuristics.

An externally developed system comes with a history. Its architecture has evolved based on the needs of the community developing it and its features may make it a promising candidate for reuse. The community's agenda to develop the system further may be in line with internal needs and first tests might suggest adoption. *Designing for adoption* requires the design science researcher to answer a number of specific questions such as: how does the system technology interact with the internal technical environment? Do internal designers sufficiently understand the system to make use of it? Is compatibility ensured across key processes? What is the release history of the community and can we expect the community to continue to maintain and release updated versions? Is the community responsive, offering documentation or a help forum?

Second, regarding *adaptability,* searching for the right contribution to a community, given the functional affordance of the external artifact and the pre-defined and emerging needs of internal designers, may result in experimentation and improvisation with new approaches such as design for incompleteness (Garud et al., 2008). *Designing the contribution* means evaluating the gap between existing (and potential) functionality and internal needs in terms of adapting the system. This may entail contributing to the community in order to benefit from the community's future improvements (Dahlander and Wallin, 2006). Design evaluation, here, means gauging the potential community relations that allow quality improvements to flow both ways. Is there mutual understanding in terms of quality, usability, and security standards? If not, can new requirements be introduced into the community by contributing and relating to the community on productive and friendly terms?

Third, the extended design cycles may benefit particularly from the *auspice* that external artifacts carry. Wide-spread use and popularity of an artifact designed by a global community may signal quality and the visibility of the artifact may reduce the search cost (e.g. components in the GNU Linux distribution Debian)—but such artifacts may also have an "image" that could deter internal users from embracing them. Consider the ease of identifying software distributed under the Apache umbrella of projects hosted by the Apache Software Foundation. The public fame of Apache and the outstanding reputation of the Apache web server enhances the visibility of affiliated programs, such as Lenya, an OS content management system. Opting for a highly visible artifact may increase internal acceptance and a perception of openness in the firm. However, it may also have a reverse effect and aggravate a "not invented here" response by internal designers, or raise flags about potential legal issues when using free and OS software rather than a commercial alternative. *Designing for goodwill* has to do with internal acceptance and leverage of the community's work despite uncertain community relations. How is the community perceived by users and internal designers? Is close interaction with the community desirable and promising? Does the community's work and practice (including visible signs, communication style, quality perception, etc.) inspire the creativity of internal designers and users?

Establishing research cycles in design science that take into account the dimensions outlined in our framework shows the complexity of an iterative and often intractable process. The AST approach suggested in our framework emphasizes the mutual influence between the reuse of artifacts and community relation decisions and the ability of internal designers to build and evaluate effective IT artifacts in collaboration with external communities. Thus, the extensions to design science research contained in this framework directly inform reuse and community relation decisions, possibly to a finer level of detail and practicability than the innovation literature has so far achieved.

Conclusion

Design science has evolved into a powerful set of theories and methods of design and action in IS. A conventional approach to design science in IS benefits from a stable context along the dimensions of people, organization, and technology. We argue that the phenomenon of OS software introduces major changes in this context, calling for a new and important strategic research agenda for design science. A focus on the relationship between external artifacts and organizational users and designers can considerably broaden the scope of the design science approach and enrich research by incorporating theory from other fields, such as adaptive structuration theory (AST). Based on AST, we developed a new research framework covering agenda, adaptability, and auspice at three levels of granularity—middle management, top management, and entrepreneur. The framework sorts design tasks and challenges pertaining to the different dimensions and shows how technological artifacts impact internal designers and users. Future work needs to develop methods and processes that ultimately help designers make key decisions regarding reuse and community relations. We outline a design science research vector along which researchers can organize their research activities.

A focus on the user-artifact relationship also opens up new research topics in other areas, such as motivation theory (flow, cognitive dissonance, self-determination), private-collective innovation theory, or behavioral economics (reciprocity, fairness, altruism). If the external IT artifact is developed outside the influence of any one organization, and if both technical characteristics and internal needs continue to evolve, the great promise of a design science approach lies in the carefully structured, rigorous, and repeated questioning, understanding, and construction of the relationship between artifact and user. Working on the research areas defined by the framework may even bring forth the contours of an "open design science."

References

Armstrong, C., V. Sambamurthy. 1999. Information technology assimilation in firms: The influence of senior leadership and IT infrastructures. *Information Systems Research* **10**(4) 304-327.

Banker, R., R. Kauffman. 2004. The evolution of research on information systems: A fiftieth-year survey of the literature in Management Science. *Management Science* **50**(3) 281-298.

Bodker, M., L. Nielsen, R. Orngreen. 2007. Enabling user Centered design processes in OS communities. *Lecture Notes in Computer Science* 4559 10-18.

Bonaccorsi, A., S. Giannangeli, C. Rossi. 2006. Entry strategies under competing standards: Hybrid business models in the OS software industry. *Management Science* **52**(7) 1085-1098.

Braa, J., C. Hedberg. 2002. The struggle for district-based health information systems in South Africa. *The Information Society* **18**(2) 113.

Brynjolfsson, E., L.M. Hitt. 1998. Beyond the productivity paradox. *Commun. ACM* **41**(8) 49-55.

Butler, T., J. Feller, A. Pope, B. Emerson, C. Murphy. 2008. Designing a core IT artefact for Knowledge Management Systems using participatory action research in a government and a non-government organisation. *The Journal of Strategic Information Systems* **17**(4) 249-267.

Ciborra, C. 1998. Crisis and foundations: an inquiry into the nature and limits of models and methods in the information systems discipline. *Journal of Strategic Information Systems* **7**(1) 5-16.

Cross, N. 2007. Forty years of design research. *Design Studies* **28**(1) 1-4.

Cybulski, J., Neal, R., Kram, A., and Allen, J. 1998. Reuse of early life-cycle artifacts: work products and methods, and tools. *Annals of Software Engineering* **5**, 227-251.

Dahlander, L. 2007. Penguin in a new suit: a tale of how de novo entrants emerged to harness free and OS software communities. *Industrial and Corporate Change* **16**(5) 913-943.

Dahlander, L., M.W. Wallin. 2006. A man on the inside: Unlocking communities as complementary assets. *Research Policy* **35**(8) 1243-1259.

DeSanctis, G., M.S. Poole. 1994. Capturing the complexity in advanced technology use—adaptive structuration theory. *Organization Science* **5**(2) 121-147.

Elbanna, A. 2010. Rethinking IS project boundaries in practice: A multiple-projects perspective. *Journal of Strategic Information Systems* **19**(1) 39-51.

Fitzgerald, B. 2006. The transformation of OS software. *MIS Quarterly* **30**(3) 587-598.

Fitzgerald, B., T. Kenny. 2004. Developing an information systems infrastructure with OS software. *IEEE Software* **21**(1) 50.

Frakes, W., S. Isoda. 1994. Success Factors Of Systematic Reuse. *IEEE Software* **11**(5), 15-19.

Garud, R., S. Jain, P. Tuertscher. 2008. Incomplete by design and designing for incompleteness. *Organization Studies* **29**(3) 351-371.

Gregor, S. 2006. The nature of theory in information systems. *MIS Quarterly* **30**(3) 611-642.

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

Haefliger, S., G. von Krogh, S. Spaeth. 2008. Code Reuse in OS Software. *Management Science* **54**(1) 180-193.

Henkel, J. 2006. Selective revealing in open innovation processes: The case of embedded Linux. *Research Policy* **35**(7) 953-969.

Hevner, A.R. 2007. The Three Cycle View of Design Science Research. *Scandinavian journal of information systems* **19**(2) 87.

Hevner, A.R., S.T. March, J. Park, S. Ram. 2004. Design science in Information Systems research. *MIS Quarterly* **28**(1) 75-105.

Hickman, L. 1998. *Reading Dewey: interpretations for a postmodern generation*. Indiana University Press.

Iivari, J. 2007. A paradigmatic analysis of information systems as a design science. *Scandinavian journal of information systems* **19**(2) 39.

Kerner, S.M. 2010. You Can't Control Linux. *CIO Update*. April 14, 2010. URL: http://www.cioupdate.com/features/article.php/3876581/You-Cant-Control-Linux.htm

Kim, Y., E. A. Stohr. 1998. Software Reuse: Survey And Research Directions. *Journal of Management Information Systems* **14**(4), 113-147.

Kroah-Hartman, G., Corbet, J., McPherson, A. 2009. Linux Kernel Development: How Fast it is Going, Who is Doing It, What They are Doing, and Who is Sponsoring It: An August 2009 Update URL: http://www.linuxfoundation.org/publications/whowriteslinux.pdf

Lee, G., R. Cole. 2003. From a firm-based to a community-based model of knowledge creation: The case of the Linux kernel development. *Organization Science* **14**(6) 633-649.

Lynex, A., P. Layzell. 1998. Organisational Considerations For Software Reuse. *Annals of Software Engineering* **5**, 105-124.

Maillart, T., D. Sornette, S. Spaeth, G. von Krogh. 2008. Empirical Tests of Zipf's Law Mechanism in Open Source Linux Distribution. *Physical Review Letters* **101**(21).

Majchrzak, A., Wagner, C., and Yates, D. N. 2006. Corporate Wiki Users: Results of a Survey, in *Proceedings of the 2006 International Symposium on Wikis*, New York: ACM Press, pp. 99-104. URL: http://www.wikisym.org/ws2006/proceedings/p99.pdf

March, S., G. Smith. 1995. Design and natural-science research on information technology. *Decision Support Systems* **15**(4) 251-266.

March, S.T., V.C. Storey. 2008. Design science in the information systems discipline: An introduction to the special issue on design science research. *MIS Quarterly* **32**(4) 725-730.

Markus, M.L., A. Majchrzak, L. Gasser. 2002. A Design Theory for Systems That Support Emergent Knowledge Processes. *MIS Quarterly* **26**(3) 179-212.

Markus, M.L., M.S. Silver. 2008. A Foundation for the Study of IT Effects: A New Look at DeSanctis and Poole's Concepts of Structural Features and Spirit. *Journal of the Association for Information Systems* **9**(10) 609-632.

Massey, A., M. Montoya-Weiss, T. O'Driscoll. 2002. Knowledge management in pursuit of performance: Insights from Nortel Networks. *MIS Quarterly* **26**(3) 269-289.

Mata, F., W. Fuerst, J. Barney. 1995. Information technology and sustained competitive advantage: A resource-based analysis. *MIS Quarterly* 19(4) 487-505.

O'Mahony, S., F. Ferraro. 2007. The emergence of governance in an OS community. *Academy of Management Journal* **50**(5) 1079-1106.

Orlikowski, W.J. 1992. The duality of technology - rethinking the concept of technology in organizations. *Organization Science* **3**(3) 398-427.

Orlikowski, W., C. Iacono. 2001. Research commentary: Desperately seeking the "IT" in IT research - A call to theorizing the IT artifact. *Information Systems Research* **12**(2) 121-134.

Orlikowski, W., S. Scott. 2008. Sociomateriality: Challenging the Separation of Technology, Work and Organization. *Academy of Management Annals* **2** 433-474.

Ravichandran, T., M. Rothenberger. 2003. Software reuse strategies and component markets. *Communications of the ACM* **46**(8) 109-114.

Sarasvathy, S., N. Dew. 2005. New market creation through transformation. *Journal of Evolutionary Economics* **15**(5) 533-565.

Sawhney, M., E. Prandelli. 2000. Communities of creation: Managing distributed innovation in turbulent markets. *California Management Review* **42**(4) 24.

Sein, M.K., O. Henfridsson, S. Purao, M. Rossi, R. Lindgren, 2011. Action Design Research. *MIS Quarterly*. Forthcoming.

Shah, S. 2006. Motivation, governance, and the viability of hybrid forms in OS software development. *Management Science* **52**(7) 1000-1014.

Simon, H.A. 1996. *The sciences of the artificial*. MIT Press.

Spaeth, S., S. Haefliger, G. von Krogh, B. Renzl. 2008. Communal resources in OS software development. *Information Research* **13**(1).

Spaeth, S.; Stuermer, M. & von Krogh, G. 2010. Enabling Knowledge Creation through Outsiders: Towards a Push Model of Open Innovation. *International Journal of Technology Management*, forthcoming.

Steiniger, S., E. Bocher. 2009. An overview on free and OS GIS developments. *International Journal of Geographical Information Science* **23**(10) 1345-1370.

Steyaert, C. 2007. 'Entrepreneuring' as a conceptual attractor? A review of process theories in 20 years of entrepreneurship studies. *Entrepreneurship and Regional Development* **19**(6) 453-477.

Stuermer, M., S. Spaeth, G. von Krogh. 2009. Extending private-collective innovation: a case study. *R & D Management* **39**(2) 170-191.

Susman, G., R. Evered. 1978. Assessment of scientific merits of action research. *Administrative Science Quarterly* **23**(4) 582-603.

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

von Hippel, E. 2001. Learning from open-source software. *MIT Sloan Management Review* **42**(4) 82–86.

von Hippel, E., G. von Krogh. 2003. Open source software and the "private-collective" innovation model: Issues for organization science. *Organization Science* **14**(2) 209-223.

West, J. 2003. How open is open enough? Melding proprietary and open source platform strategies. Research *Policy* **32**(7) 1259.

Yalta, A.T., A.Y. Yalta. 2010. Should Economists Use OS Software for Doing Research? *Computational Economics* **35**(4) 371-394.

Young, R., W.G. Rohm. 1999. *Under the Radar: How Red Hat Changed the Software Business - and Took Microsoft by Surprise*. Coriolis Group Books.