Alexander Eck
Stefan Keidel
Falk Uebernickel
Thomas Schneider
Walter Brenner

Not all information systems are created equal: exploring IT resources for agile systems delivery

This article explores IT resources that can plausibly contribute to increased agility in the creation of new or adaptation of existing information systems (IS), while keeping operational risks under control. Embedded in the empirical context of a large financial company and on the example of an IT architecture design we demonstrate how such IT resources can be derived systematically. To this end we first apply a portfolio technique to characterize the problem space. From this analysis we derive four target dimensions – namely speed, costs, risk, and scale – to which all subsequent design decisions contribute. We then present a possible IT architecture for systems delivery resulting from these design decisions. Finally we discuss three selected opportunities to increase organizational agility following an implementation of the previously described IT architecture.

Introduction

In light of turbulent market environments that require fast reaction to changing realities, business departments increasingly complain about slow, costly, and inflexible delivery of information systems (IS). IT departments, for their part, traditionally seek to change the systems landscape cautiously, because they regard operations of stable and secure systems as their most important task [Pavlou/El Sawy 2010].

Financial services make a good case to illustrate the differing priorities. The industry undergoes a transformation driven by digitization and regulatory changes, which forces companies to adapt quickly. But IS at the core of business processes such as transaction systems have to work correctly and protect sensitive data, which calls for a diligent approach. Due to this area of conflict the business side seems to be concerned to increase organizational agility, meaning sensing environmental change and reacting readily [Goldman et al. 1995], while the IT department seemingly puts a larger emphasis on managing operational risk, i.e. managing risk of loss resulting from inadequate or deficient systems [BCBS 2011].

In this paper we explore IT resources that are systematically designed with the goal to reconcile these different views for an important area of business and IT collaboration, namely IS delivery. Specifically, we devise an IT architecture for systems delivery with the goal to increase organizational agility while keeping operational risk under control. The paper is structured as follows. First we provide an introduction to main concepts that our research is based upon. Next we describe the empirical setting, followed by an overview of the research methodology. Subsequently we present the step-wise development of an IT architecture design. We discuss selected opportunities for increasing agile behavior afforded by this IT architecture, before concluding with summarizing our main contributions.

Background

We first define the two concepts that motivated this research, i.e. organization-al agility and operational risk. Then we provide a brief overview of agile IS delivery capabilities as a popular example of IT resources designed to increase overall organizational agility, and put forth the need of a congruent IT architecture for agile IS delivery as complementary resource.

Organizational agility

Following Goldman et al. [1995], we define organizational agility as the firm-wide ability to sense environmental changes and respond fast and readily. While sensing is associated with information gath-ering and analysis in order to anticipate environmental changes [Overby et al. 2006], responding refers to taking competitive action, either as a means to adapt to changed market conditions or for seizing a newly identified business opportunity [Sambamurthy et al. 2007].

The IT department plays a decisive role in the overall organizational agility of a company, for example through the way it manages its resources [Sambamurthy et al. 2003]. For the purpose of this paper, IT resources consist of IT assets (anything tangible or intangible that can be put to beneficial use) and IT capabilities (patterns of organizational behavior that transform input assets into output assets) [Wade/Hulland 2004].

Delivery of newly created or adapted IS has been associated with the responding aspect of organizational agility [Overby et al. 2006]. But not all systems need to be introduced rapidly or adapted to match unexpected environmental changes. There are systems that just have to work properly, are well understood and change rarely, such as accounting or transaction systems. Indeed, Weill/Vitale [2002] and Ross [2003] argue that through standardizing and continuously refining such systems, the IT organization can devote a larger share of its limited resources on those areas that provide their business departments an edge to the competition.

Operational risk

The risk of loss resulting from inadequate or deficient processes or systems is called operational risk in the financial services industry [BCBS 2011]. This definition includes IS as source of risk that needs to be managed [Ross 2011]. Financial services regulation requires that operational risk exposure is cushioned by an adequate capital buffer [Power 2005]. Through lowering its operational risk, an organization improves its competitive position, as more capital can be put to productive use [Flores et al. 2006].

Risk exposure is a multiplication of the financial consequence of a loss event with the probability of the loss event happening [Boehm 1991]. A prudent strategy to manage risk exposure is to decrease the probability of failure in particular for those IS that can potentially cause high losses, that is for business-critical IS.

Agile IS delivery capabilities

Agile IS delivery capabilities originated as practitioners' responses to the shortcomings of traditional plan-based approaches which are not designed to accommodate environmental changes in the IS delivery process [Abrahamsson et al. 2009]. A plethora of practices, methods and social aspects have been proposed [Abbas et al. 2010], with cornerstones being iterative and incremental systems delivery [Larman/Basili 2003] and increased communication throughout the systems delivery process [Hummel et al. 2013]. To bring order in the proliferating field, Conboy [2009] developed a characterization of agility in IS delivery. He proposes that agile IS delivery capabilities can be identified through their qualities to (1) create change, react to change, or learn from change; and (2) do so with high quality, cost-efficiently and fast.

Due to these assumed qualities, companies across industries have been acquiring agile IS delivery capabilities, albeit at varying degrees [West/Grant 2010]. This is a strong indicator that these capabilities are indeed beneficial to increase organizational

agility, an assertion which is backed by a growing body of empirical research [Dybå/Dingsøyr 2008; Laanti et al. 2011].

While agile capabilities have been discussed extensively [Dingsøyr et al. 2012], the debate around complementary assets is in its infancy [Tiwana/Konsynski 2010]. This is why we set out to develop an IT architecture for agile IS delivery with this research, i.e. an interlinked arrangement of IT assets that equip organizations with the means to introduce or change information systems in an agile way.

Empirical setting

We chose the financial services industry as empirical setting for this research, because it is a challenging competitive environment in which companies must increase their organizational agility while simultaneously putting a strong focus on managing operational risk. These diverging requirements can be illustrated with three central challenges in which the IT department plays a crucial role: (1) keeping pace with technological progress, (2) implementing changing regulations, and (3) ensuring information security.

The financial services industry is built around IS and has a strong history of embracing technology innovations [Heise 2011]. Adapting the company strategy to gradual or sudden technology shifts that enable innovations, e.g. mobile banking, remains a central task for industry players [Möwes et al. 2011].

Another recurring theme are regulatory changes, which imply that IS need to be changed [Abdullah et al. 2010]. Recent examples with wide-ranging consequences are the FATCA[1] regulations and Basel III[2] standards. As complying with regulations at any time is a pure cost factor and not a differentiating feature in the market, there is substantial pressure on the IT organization to work error-free and cost-efficiently [Anderson et al. 2006].

Finally, information security is of utmost importance in the financial services industry, both for reputational and regulatory reasons [Webster 2006]. Compared with other industries the standards are strict, because storing and processing sensitive data such as customer records are central to many business processes [McKelvey 2001]. It is also due to information security considerations that progressive paradigms such as cloud computing are just slowly finding their way in financial services [Wenge et al. 2014].

Research leading to this paper was conducted at the Swiss IT department of UBS. As of mid-2014 UBS had around 21,400 employees in Switzerland and 60,000 employees across all regions it operates in. Group revenues were CHF 27.7bn for 2013 and client assets under management were about CHF 2,400bn.[3] Swiss operations serve Private Clients, Wealth Management Switzerland, Institutional Clients, Investment Bank Switzerland and Asset Management Switzerland. Also many of the central corporate functions are located in Switzerland. At the time of investigation, the IT department carried out a program to foster agile IS delivery capabilities through-out all organizational units under top-management supervision.

Selecting UBS for anchoring research in a real-world setting made sense for three reasons: First, the company operates in the financial services industry, sharing common industry challenges in technology shifts, continuous regulatory change, and information security. Second, UBS is a leading bank in Switzerland and serves a broad range of clients, making its business rather complex and organizational agility an important success factor [Arteta/Giachetti 2004]. And third, at the time of investigation, the organization repositioned specifically its IT department towards delivering more value to its business counterparts and contributing

(1) Foreign Account Tax Compliance Act, see http://www.irs.gov/businesses/corporations/article/0,,id=236667,00.html

(2) See http://www.bis.org/bcbc/basel3.htm

(3) These information are taken from the 2013 annual report and the company website, http://www.ubs.com

to higher organizational agility.

Research methodology

In order to produce actionable research that is of current relevance to practitioners [Benbasat/Zmud 1999], we applied design science research [Hevner et al. 2004]. The main design goal was developing an IT architecture for systems delivery that helps to increase organizational agility within the organizational context of UBS. Research was carried out for eight months in the period of January 2013 throughout August 2013. The research methodology was informed by the phases proposed by Peffers et al. [2007], namely: (1) define the problem; (2) define design goals; and (3) build the artifact and evaluate it.

To define the problem, we conducted six semi-structured interviews with senior expert personnel, mainly to distill shortcomings of the existing ensemble of IT assets utilized to create or change IS. On average, each interview lasted for about one hour. Two of the researchers extracted the shortcomings that were identified in the interviews independently, and through discussion distilled five possible problem areas. These were reduced to a more manageable problem space of two areas – responsiveness and business criticality – after a joint reflective round with four of the interviewed experts.

The design goals were defined in a series of workshops with a small group of experts, as were fundamental design decisions. The results of each workshop were refined in concept documents by the researchers, which were then reviewed by a larger group of experts from within UBS.

Subsequently, the artifact was constructed and evaluated incrementally in collaboration with experts from UBS. Apart from this collaborative work, analysis of company-internal documentation allowed identifying links to existing resources which the newly constructed artifact could leverage. The researchers were granted access to relevant material such as project documentation, organizational charts, descriptions of the IS landscape, documentation of standard processes, and general company policies. Overall a group of thirteen experts were involved in design and construction activities, selected to cover a wide variance of competencies within the IT organization.

The IT architecture presented in this paper has yet to be implemented for practical use, which constitutes a refinement in the cyclical design science research setting. Accordingly, only pre-implementation evaluation activities [Sonnenberg/Brocke 2012] were carried out so far.

Design of an IT architecture for agile IS delivery

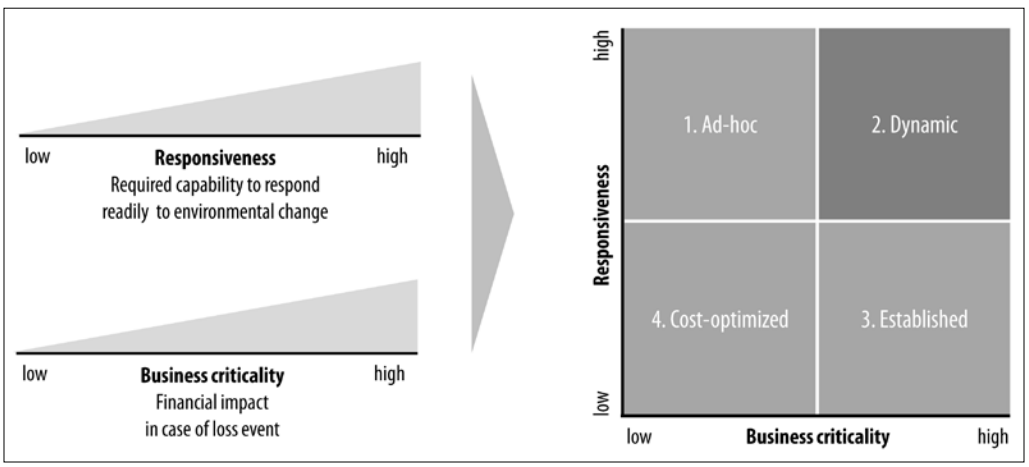
In the following we present the step-wise design of an IT architecture for agile IS delivery, which we call Application Elevator. Progress towards the design is illustrated along three artifacts: (1) classification matrix to characterize the problem space; (2) target dimensions and design decisions; and finally (3) IT architecture.

Classification matrix to characterize the problem space

Ward [1988] advocated the use of two-dimensional matrices as practical tool for classifying IS and deriving distinct strategic directions for each resulting portfolio. We employ this technique to characterize the overall problem space and define the specific segment for which the Application Elevator is supposed to be designed.

The first dimension for delimitating is responsiveness, chosen from the observation that some IS remain rather stable over time, while others need to be introduced fast and adapted frequently in response to environmental change. The second dimension is business criticality, meaning financial consequence of a loss event caused by inadequate or deficient IS. We assume that some IS are uncritical, while others are utilized in business processes with substantial consequence in case of a loss

Figure 1: Classification matrix



event happening. The resulting classification matrix is shown in Figure 1.

For each quadrant we formulate a distinct high-level strategy, analogous to the recommendations of Weill/Ross [2005] and Ward/Peppard [2002]:

1. **Ad-hoc:** Typically an IS with lower complexity or a prototype falls in this quadrant. The main purposes are to satisfy an immediate business requirement or to explore an assumed business value. Such a system mandates an ad-hoc and decentralized approach in systems delivery. Business departments manage systems delivery autonomously, while the IT department may provide basic IT assets [Behrens 2009].

2. **Dynamic:** The Application Elevator targets this quadrant. The typical system has been classified as being business-critical, for example by processing sensitive data. At the same time business departments demand fast and frequent changes. Continuous business involvement is required to define requirements and set priorities, but the IT department leads systems delivery. With a standardized and scalable IT architecture in place [Tiwana/Konsynski 2010] and adequate delivery capabilities, the IT department is able to deliver fast and frequently and control operational risk at reasonable cost.

3. **Established:** A system in this quadrant reached a certain maturity and functional requirements change rarely. Usually it is highly integrated with backend systems and complex. A prominent business goal is to increase efficiency, for example through business process automation.

4. **Cost-optimized:** Such an IS either supports an inherently uncritical business process, or business criticality has decreased through sparse usage. Following a cost-benefit analysis, the IS will either be operated in a mode that incurs minimal costs or it will be discontinued.

Target dimensions and design decisions

Segmentation of the problem space in mutually exclusive portfolios resulted in specific goals for the design of the Application Elevator: first, the IT architecture is supposed to allow responsiveness in IS delivery and second, it should minimize exposure to operational risk.

To increase common understanding of these goals, we chose to break them down into four more operational target dimensions that guided subsequent design, namely speed, costs, risk, and scale:

- Speed: speed of systems delivery, from idea formulation to delivered IS
- Costs: costs of a systems delivery iteration, and subsequent costs of IS use
- Risk: operational risk exposure of delivered IS

Table 1: Main design decisions for Application Elevator architecture

Design decision	Impact on speed, costs, risk, scale
Standardized, centrally operated and configurable runtime environment with regard to computing, storage, and network	*Speed:* Very positive impact if provisioning and configuration are automated and accessible via self-service interface. *Costs:* Dynamic reconfiguration aligns cost incurrence with system utilization. *Risk:* If runtime environment is mature, probability of loss event decreases. *Scale:* Possibility to reconfigure computing, storage and network resources facilitates scaling.
Runtime environment with long-term support from vendor and IT department	*Speed:* No particular impact. *Costs:* Know-how and skills in IT department are readily available, with moderately positive impact on costs. *Risk:* Long-term support should be beneficial for risk, as runtime environment is kept up-to-date. *Scale: No particular impact.*
Continuous delivery pipeline	*Speed:* Depending on maturity of pipeline, integration and deployment may be fully automated and thus fast. *Costs:* Manual work required for integration and deployment is minimized, which decreases costs. *Risk:* The pipeline enables more frequent deployments, which decreases risk. Automation should decrease probability that errors are done in integration and deployment. *Scale:* Scaling capabilities are fostered, as changes can be introduced fast and cost-efficiently.
Skeleton IS templates; reference implementations of common functionality	*Speed:* Templates and other reference material increases delivery speed, especially for newly introduced IS. *Costs:* Leveraging templates usually saves costs, as IS delivery starts with a solid foundation. Curating templates incurs additional effort, however. *Risk:* Depending on template quality, probability of loss event may decrease. *Scale: no particular impact.*
Standard mechanisms to display, print, and manipulate sensitive data	*Speed:* Fewer interactions with security departments might be required, which should increase speed. *Costs:* Costs of IS use probably increases, as security measures typically incur additional costs. *Risk:* Standard mechanisms to display, print, and manipulate sensitive data reduce the probability that loss events happen. *Scale:* No particular impact.
Logging, monitoring, reporting, and wider diagnostics capabilities	Main impact is on *risk*: Monitoring and diagnostics support preventing loss events happening. In case of failure, these capabilities help to identify the root cause and implement counter-measures fast.

- Scale: capability to scale IS with ease when needed

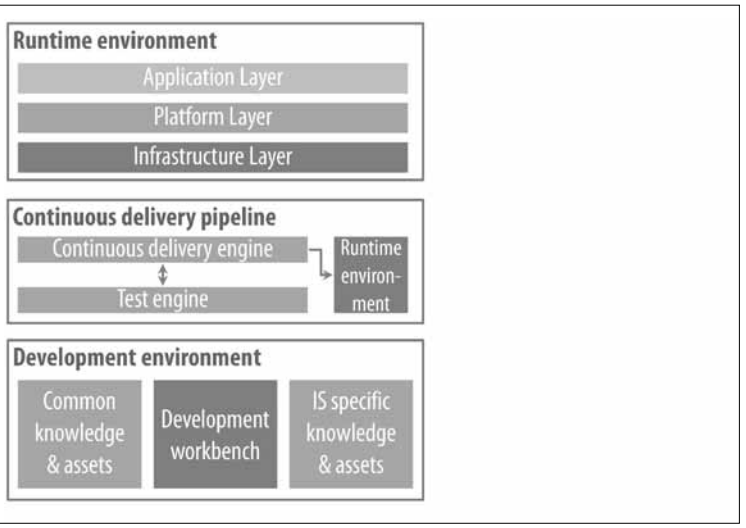
The first two dimensions, speed and costs, are characteristics commonly associated with the response aspect of organizational agility [Overby et al. 2006], and hence a direct translation of the first matrix dimension. We chose risk as proxy for the second matrix dimension. If the Application Elevator is designed with risk minimization in mind, it will be fit for purpose for business-critical IS. Scale, added as fourth dimension, captures a dynamic aspect which is not directly addressed by the classification matrix: each system follows a life cycle, which archetypically starts in the first matrix quadrant (ad-hoc) and traverses every quadrant in turn, ending in the cost-optimized quadrant [c.f. Ward 1988].

These target dimensions were helpful to assess the impact of fundamental design decisions. An overview of these decisions and their expected favorable impact on the targets is summarized in Table 1.

Application Elevator – an IT architecture for agile IS delivery

The developed IT architecture, called Application Elevator, is composed of three parts, see Figure 2:

Figure 2:
IT architecture overview



- Runtime environment on which the resulting system is operated. The runtime environment is layered and consists of virtualized infrastructure, a common platform, and IS specific functionality.
- Continuous delivery pipeline which allows automatically integrating, testing, and deploying a system.
- Development environment that provides a development workbench, common knowledge and assets, as well as IS specific knowledge and assets.

Runtime environment

The runtime environment (Figure 3) is separated into three layers and is reminiscent of cloud computing architectures [Mell/Grance 2011].

The infrastructure layer provides computing, storage, and network assets. Through virtualization [Armbrust et al. 2010] the exposed assets are independent of the underlying physical hardware. Virtualization also allows responding to variation in demand through flexibly adjusting the hardware configuration.

The platform layer provides basic functionality, consisting of an operating system, an application server, and a database server; also connectivity and messaging components are exposed for integration with a larger ecosystem. This layer defines the main parameters for IS implementation, for example the programming language to use or which connectivity options are available. A stand-alone system for instance might be deployed on a platform layer with deactivated connectivity and messaging components, and a configuration for testing purposes might connect to mock-up backend components.

The application layer bundles the IS business logic. Additional libraries provide common functionality,

Figure 3:
Runtime environment

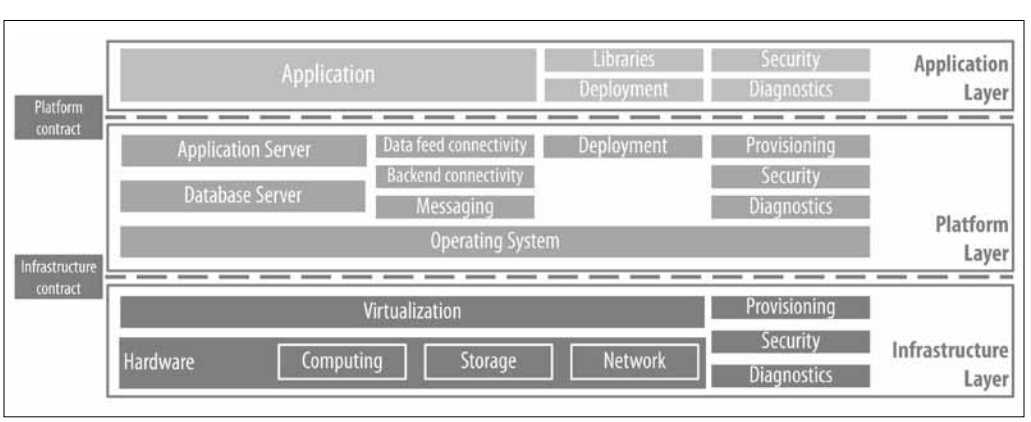
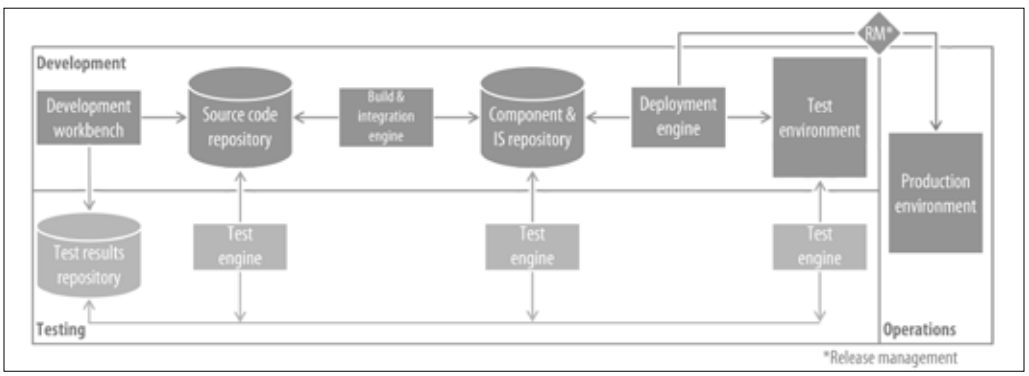


Figure 4: Continuous delivery pipeline



for example user interface or data handling functionality. The application layer is the most flexible of all three layers, and the one that ultimately provides the perceived value to the system user.

The layers provide components for provisioning (infrastructure layer and platform layer), deployment (platform layer and application layer), security (all layers) and diagnostics (all layers). Decoupling such cross-cutting functionalities keeps layer separation intact; the layers remain independent and thus interchangeable. Instead of conceiving and maintaining an integrated security component, for instance, each layer implements and maintains its own part of the security chain [Yildiz et al. 2009].

For such a loosely coupled architecture to work, service contracts on each interface must be defined. The contracts define how components interact with each other and what they can expect from each other, for example in terms of service availability or interface standards [Zou et al. 2010].

Continuous delivery pipeline

The continuous delivery pipeline establishes a highly automated link between development envir-onment and runtime environment [Humble/ Farley 2010]. The pipeline architecture is depicted in Figure 4. Its key components are:

- A common source code repository in which all source code is retained and versioned.
- A build and integration engine that compiles source code and stores resulting working software in a common component and IS repository.
- A deployment engine which deploys the application to a runtime environment, either for testing or production purposes. Deployment to production environment may be gated by separate release management governance.
- Complementary test engines and a common test result repository automate test execution and test documentation [Stolberg 2009].

Maturity of a continuous delivery pipeline can be assessed by how many of the presented components are in place and by grade of automation [Forrester Consulting 2013].

While the deployment steps for test and production environment are identical in principle, deployment onto the latter may be governed by an additional release management process. In this case release management acts as gatekeeper to operations. The continuous delivery pipeline supports release management by automatically triggering sign-off processes on request.

This unified integration and deployment chain makes progress transparent, not least because all increments and their associated test results are stored in repositories. Also all activities are recorded which enhances traceability.

Development environment

The development environment provides all necessary tools and assets required for IS delivery. It also stores and retains knowledge and assets that are developed during a delivery cycle. An illustrative overview with its three logical parts is depicted in Figure 5.

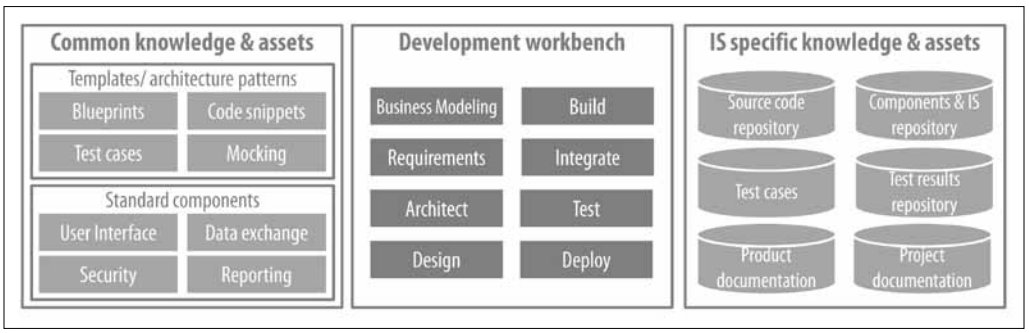


Figure 5: Development environment

Common knowledge and assets subsumes standard functionality such as security or reporting which can be utilized across several systems, and patterns for common requirements which accelerate individual systems delivery.

The development workbench consists of all tooling required at the different stages of systems delivery [cf. Kruchten 2004]. If feasible the tools are integrated so that artifacts created in one tool can be used in another.

Under IS specific knowledge and assets all artifacts associated with a particular IS are retained, which is a measure for knowledge retention. These assets prove beneficial for example when unexpected issues arise during system utilization. It is worth noticing that a large part of these assets are man-aged through the continuous delivery pipeline mentioned above.

Discussion – opportunities to increase organizational agility

We introduced the Application Elevator as an IT architecture for agile IS delivery, with the goal to increase responsiveness and still allow business-critical systems to be operated in a safe environment. The ultimate ambition, however, is to increase over-all organizational agility. In the following we discuss three opportunities that such IT architecture creates and when exploited may lead to higher organizational agility. More specifically, we regard (1) opportunities to capture life cycle dynamics; (2) opportunities to foster intra-IT collaboration; and (3) opportunities to broaden the scope of business-operated systems.

Opportunities to capture life cycle dynamics

The presented classification matrix distinguishes four quadrants, with each calling for different capabilities in IS management. For example, the ad-hoc quadrant requires low-cost and short-term problem solving capabilities, whereas long-term planning and incremental refinement capabilities are more adequate in the established quadrant. As it matures, a particular system will most certainly move from one quadrant to another, reflecting changes in its inherent business criticality or required responsiveness to change. At times there is a need to switch from one mode (e.g. ad-hoc) to another (e.g. dynamic).

IT architecture can help making this switch less painful, if it is modular enough to be shared across all different modes of operation [Weill/Ross 2005]. Modularity can be achieved through loose coupling and standardized interfaces, among other things. It leads to increased configurability and contributes to organizational agility, because intended change can be brought about more effortlessly [Tiwana/Konsynski 2010].

Considering that an individual system usually moves around the matrix during its life cycle, an additional benefit is uncovered: when transitioning from one quadrant to another, the IS specific knowledge and assets (cf. Figure 5) can be carried over. In practical terms this means for example that the technical IS implementation does not have to be re-written in another programming language, or that data can stay in the same database management system.

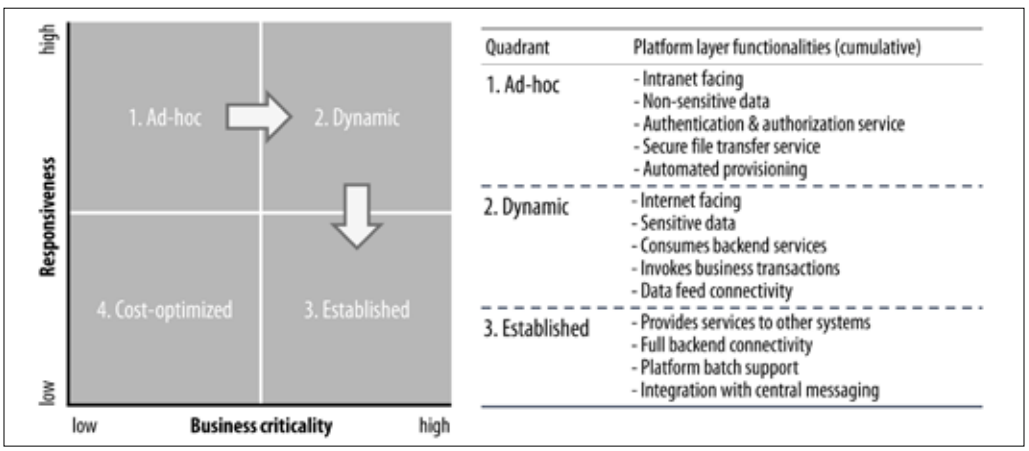


Figure 6: Quadrant transition through platform layer configuration (illustrative)

The Application Elevator runtime environment has a three-layered structure, which helps at transition through exposing or revoking functionality on the middle platform layer. A transition from ad-hoc to the dynamic quadrant then translates to a reconfiguration of the platform layer and a subsequent adaptation of the business logic on the application layer. Figure 6 illustrates the idea with fictional platform layer configurations for the first three quadrants.

Extending beyond IT architecture, the systems delivery process might also share common components across all quadrants. For example, the delivery milestones and associated management artifacts [OGC 2009] could be the same, irrespective of the quadrant. Agile systems delivery becomes one of several delivery modes, among which the IT department is able to choose depending of the situation at hand [Boehm/Turner 2004]. For this approach to work the IT department must be sufficiently skilled for adapting to context, and additional investments might be necessary for supporting more than one systems delivery mode in parallel.

Opportunities to foster intra-IT collaboration

Continuous delivery originated from software engineering disciplines [e.g., Feitelson et al. 2013], while discussion within the IS community has been of rather abstract nature [e.g., Gericke et al. 2010]. Nevertheless there is a dimension beyond purely technical aspects, which is worth being discussed. In particular, there is the potential to increase organizational agility when adequate resources are put in place that allow a highly automated pipeline for systems integration, testing, and deployment.

First, through automation the effort and time put into integration, testing, and deployment is lowered and outcome variance is reduced. Consequently the rate of deployment can be increased, which enables delivering increments and iterating often. Second, frequent deployments increase communication among the stakeholders [Cockburn/Highsmith 2001], which is associated with higher customer satisfaction [Hummel et al. 2013]. And third, if paired with automated recording of all process steps, traceability is increased, which facilitates root cause analysis in case of error [Humble/Farley 2010].

The technology challenges to introduce a continuous delivery pipeline are considerable [Kim/Ryoo 2012]. But an arguably larger challenge is institutionalizing close collaboration between systems delivery and systems operation within the IT department, in addition to business and IT collaboration (Figure 7).

This intra-departmental collaboration is known as DevOps in practitioner journals [Debois 2011] and requires that the different functional teams within the IT department align their work and share responsibility for both system delivery and system operations. Only with such alignment and collaboration a continuous delivery pipeline can be suc-

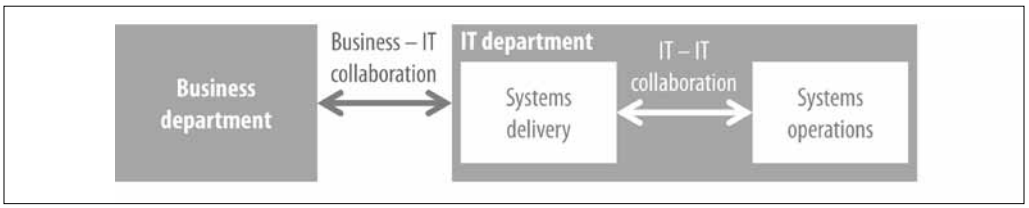


Figure 7: Business – IT collaboration and IT – IT collaboration

cessful and lead to increased organizational agility, by closing possible agility gaps [van Oosterhout et al. 2006] not only between business and IT, but also within the IT department.

Opportunities to broaden the scope of business-operated systems

IS operated outside the control of the IT organization are referred to as shadow IS [György et al. 2012]. The most prominent examples are workplace systems like Microsoft Excel[4] that are routinely employed by business departments for data capturing, modeling, and reporting [Raden 2005].

Shadow IS add complexity to the overall systems landscape which may increase overall costs for the organization. In addition shadow IS are hard to govern [Shaw 1997], because business departments implement governance mechanisms (if any) autonomously, with little or no central oversight and limited transparency. And most importantly in the financial services industry, shadow IS are sources of operational risk, further adding to their overall life cycle cost.

Shadow IS are a wide-spread phenomenon, and for good reasons. They are an effective means to respond to imminent environmental changes through bottom-up initiatives [Behrens 2009], thus increasing organizational agility. For this purpose shadow IS should be promoted [György et al. 2012]. One way to capture the value of shadow IS and mitigate some of their shortcomings is through making IT assets such as the Application Elevator accessible to the business department.

In such a model, the IT department exposes a specific configuration for business to use autonomously, e.g. the ad-hoc configuration illustrated in Figure 6. Granting access to IT assets originally designed for use by IT professionals demands appropriate capabilities in the business department, which few are willing to develop. But the business side gains access to tools that are more powerful than what is usually available to them. For example when considerable computing power or multi-user functionality is required, such complex assets might be attractive for business departments to utilize. The continuous delivery pipeline can be leveraged to monitor and control activity of the business department, which helps with governance. Even more profoundly, handover to the IT department, when necessary, is rather straightforward, because the same assets are being utilized on both sides.

Conclusion

With this article we introduced the Application Elevator, an IT architecture for IS delivery with the goal to increase responsiveness while keeping operational risk under control. Two main contributions can be distilled. First, we applied design science research to develop a systematic process for ex-ploring and designing IT resources for a specific goal, in our case an IT architecture design. We showed useful artifacts that assisted in the three steps of problem identification, goal definition and artifact design. And second, we presented how an IT architecture for agile systems delivery might look like. The presented design consists of (1) a three-layered runtime environment reminiscent of cloud computing architectures; (2) a continuous delivery pipeline for automating integration, testing, and deployment; and (3) a development environment to provide tools for systems delivery and to capture assets created in the process. In this context, we engaged in a discussion on how the developed artifact may create opportunities for increasing organizational agility.

[4] http://microsoft.com/excel

This research has its limitations, which lead to worthwhile fields of further investigation. First and foremost, the utility of the Application Elevator towards contributing to organizational agility has not been evaluated empirically. A natural avenue would consist of implementing the Application Elevator in an empirical setting and evaluating its contribution to organizational agility. With this research we provided three promising areas (IS life cycle, collaboration, and shadow IS) in which utility might be demonstrated. And secondly, the presented IT architecture is idiosyncratic to one particular organization, and therefore not generalizable outside its empirical context. We made transparent how the artifact was designed, creating the opportunity to replicate the process. Designing similar artifacts in different empirical settings and synthetizing the results would pave the way towards a reference IT architecture for agile IS delivery, besides providing evidence whether the presented process is robust.

Bibliography

Abbas, N./Gravell, A.M./Wills, G.B. (2010). Using Factor Analysis to Generate Clusters of Agile Practices. (A Guide for Agile Process Improvement). AGILE 2010 Proceedings:11-20.

Abdullah, N.S./Sadiq, S./Indulska, M. (2010). Emerging Challenges in Information Systems Research for Regulatory Compliance Management. In: Pernici, B. (ed.), Advanced Information Systems Engineering. Berlin/Heidelberg:251-265.

Abrahamsson, P./Conboy, K./Wang, X. (2009). ‚Lots done, more to do': the current state of agile systems development research. European Journal of Information Systems 18(4):281-284.

Anderson, R./Dunkinson, A./Ebert, M./Faubion, S./Gabriel, K./Law, C./Lindig, M./Nardine, D./Ryan, P./Salazar, M./Soles, R./Torchia, T./Zarrella, E.A. (2006). The Compliance Journey: Leveraging Information Technology to Reduce Costs and Improve Responsiveness.

Armbrust, M./Fox, A./Griffith, R./Joseph, A.D./Katz, R./Konwinski, A./Lee, G./Patterson, D./Rabkin, A./Stoica, I./Zaharia, M. (2010). A view of cloud computing. Communications of the ACM 53(4):50-58.

Arteta, B.M./Giachetti, R.E. (2004). A measure of agility as the complexity of the enterprise system. Robotics and Computer-Integrated Manufacturing 20(6):495-503.

BCBS (2011). Principles for the sound management of operational risk, Bank for International Settlements. Basel.

Behrens, S. (2009). Shadow systems: the good, the bad and the ugly. Communications of the ACM 52(2):124-129.

Benbasat, I./Zmud, R.W. (1999). Empirical research in information systems: the practice of relevance. MIS Quarterly 23(1):3-16.

Boehm, B.W. (1991). Software risk management: principles and practices. Software, IEEE 8(1):32-41.

Boehm, B.W./Turner, R. (2004). Balancing agility and discipline. A guide for the perplexed. Boston.

Cockburn, A./Highsmith, J. (2001). Agile software development: the people factor. Computer 34(11):131-133.

Conboy, K. (2009). Agility from First Principles: Reconstructing the Concept of Agility in Information Systems Development. Information Systems Research 20(3):329-354.

Debois, P. (2011). Opening Statement. In: Debois, P. (ed.), Devops: A Software Revolution in the Making? Arlington, MA, USA:3-5.

Dingsøyr, T./Nerur, S./Balijepally, V./Moe, N.B. (2012). A decade of agile methodologies: towards explaining agile software development. Journal of Systems and Software 85(6):1213-1221.

Dybå, T./Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. Information and Software Technology 50(9-10:833-859.

Feitelson, D.G./Frachtenberg, E./Beck, K.L. (2013). Development and deployment at Facebook. IEEE Internet Computing 17(4):8-17.

Flores, F./Bónson-Ponte, E./Escobar-Rodríguez, T.

(2006). Operational risk information system: a challenge for the banking sector. Journal of Financial Regulation and Compliance 14(4):383-401.

Forrester Consulting (2013). Continuous delivery: a maturity assessment model. Building Competitive Advantage With Software Through A Continuous Delivery Process. Cambridge, USA.

Gericke, A./Kleese, M./Winter, R./Wortmann, F. (2010). Success factors of application integration: an exploratory analysis. Communications of the AIS 27:677-694.

Goldman, S.L./Nagel, R.N./Preiss, K. (1995). Agile competitors and virtual organizations. Strategies for enriching the customer. New York.

György, A./Cleven, A./Uebernickel, F./and Brenner, W. (2012). Exploring The Shadows: IT Governance Approaches To User-Driven Innovation. ECIS 2012 Proceedings:Paper 222.

Heise, L. (2011). Retail banking and the dynamics of information technology in business organizations. In: Batiz-Lazo, B./Maixé Altés, Joan Carles/Thomes, P. (eds.), Technological innovation in retail finance. International historical perspectives. New York: Routledge:275-285.

Hevner, A.R./March, S.T./Park, J./Ram, S. (2004). Design science in information systems research. MIS Quarterly 28(1):75-105.

Humble, J./Farley, D. (2010). Continuous Delivery. Reliable software releases through build, test, and deployment automation. Upper Saddle River, NJ.

Hummel, M./Rosenkranz, C./Holten, R. (2013). The Role of Communication in Agile Systems Development. Business & Information Systems Engineering 5(5):343-355.

Kim, E./Ryoo, S. (2012). Agile adoption story from NHN. COMPSAC 2012 Proceedings:476-481.

Kruchten, P. (2004). The rational unified process. An introduction, Addison-Wesley, Boston.

Laanti, M./Salo, O./Abrahamsson, P. (2011). Agile methods rapidly replacing traditional methods at Nokia: A survey of opinions on agile transformation. Information and Software Technology 53(3):276-290.

Larman, C./Basili, V.R. (2003). Iterative and incremental developments. a brief history. Computer 36(6):47-56.

McKelvey, B. (2001). Financial Institutions' Duty of Confidentiality to Keep Customer's Personal Information Secure from the Threat of Identity Theft. University of British Columbia Law Review 34(2):1077-1128.

Mell, P.M./Grance, T. (2011). The NIST Definition of Cloud Computing.

Möwes, T./Puschmann, T./Alt, R. (2011). Service-based Integration of IT-Innovations in Customer-Bank-Interaction. WI 2011 Proceedings:Paper 102.

OGC (52009). Managing successful projects with PRINCE2. London.

Overby, E./Bharadwaj, A./Sambamurthy, V. (2006). Enterprise agility and the enabling role of information technology. European Journal of Information Systems 15(2):120-131.

Pavlou, P.A./El Sawy, O.A. (2010). The "Third Hand": IT-Enabled Competitive Advantage in Turbulence Through Improvisational Capabilities. Information Systems Research 21(3):443-471.

Peffers, K./Tuunanen, T./Rothenberger, M.A./Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. Journal of Management Information Systems 24(3):45-77.

Power, M. (2005). The invention of operational risk. Review of International Political Economy 12(4):577-599.

Raden, N. (2005). Shedding Light on Shadow IT: Is Excel Running Your Business? Santa Barbara, CA.

Ross, J.W. (2003). Creating a Strategic IT Architecture Competency: Learning in Stages. MIS Quarterly Executive 2(1):31-43.

Ross, R.S. (2011). Managing information security risk. Organization, mission, and information system

view. Gaithersburg, MD.

Sambamurthy, V./Bharadwaj, A./Grover, V. (2003). Shaping Agility through Digital Options: Reconceptualizing the Role of Information Technology in Contemporary Firms. MIS Quarterly 27(2):237-263.

Sambamurthy, V./Wei, K.-K./Lim, K./Lee, D. (2007). IT-Enabled Organizational Agility and Firms' Sustainable Competitive Advantage. ICIS 2007 Proceedings:Paper 91.

Shaw, P. (1997). Intervening in the shadow systems of organizations: Consulting from a complexity perspective. Journal of Organizational Change Management 10(3):235-250.

Sonnenberg, C./Brocke, J. (2012). Evaluation Patterns for Design Science Research Artefacts. In: Helfert, M./Donnellan, B. (eds.), Practical Aspects of Design Science. Berlin/Heidelberg:71-83.

Stolberg, S. (2009). Enabling Agile Testing through Continuous Integration. AGILE 2009 Proceedings:369-374.

Tiwana, A./Konsynski, B. (2010). Complementarities Between Organizational IT Architecture and Governance Structure. Information Systems Research 21(2):288-304.

van Oosterhout, M./Waarts, E./van Hillegersberg, J. (2006). Change factors requiring agility and implications for IT. European Journal of Information Systems 15(2):132-145.

Wade, M./Hulland, J. (2004). Review: The resource-based view and information systems research: Review, extension, and suggestions for future research. MIS Quarterly 28(1):107-142.

Ward, J.M. (1988). Information Systems and Technology Application Portfolio Management and Assessment of Matrix-Based Analyses. Journal of Information Technology 3(3):205-215.

Ward, J.M./Peppard, J. (32002). Strategic planning for information systems. Chichester, West Sussex.

Webster, M. (2006). Data protection in the financial services industry. Aldershot, Burlington, VT.

Weill, P./Ross, J. (2005). A Matrixed Approach to Designing IT Governance. MIT Sloan Management Review 46(2):26.

Weill, P./Vitale, M. (2002). What IT Infrastructure Capabilities are Needed to Implement E-Business Models? MIS Quarterly Executive 1(1):17-34.

Wenge, O./Lampe, U./Müller, A./Schaarschmidt, R. (2014). Data Privacy in Cloud Computing – An Empirical Study in the Financial Industry. AMCIS 2014 Proceedings.

West, D./Grant, T. (2010). Agile Development: Mainstream Adoption Has Changed Agility.

Yildiz, M./Abawajy, J./Ercan, T./Bernoth, A. (2009). A Layered Security Approach for Cloud Computing Infrastructure. ISPAN 2009 Proceedings:763-767.

Zou, J./Wang, Y./Lin, K.-J. (2010). A Formal Service Contract Model for Accountable SaaS and Cloud Services. SSC 2010 Proceedings:73-80.

Authors

Dipl.-Ing. Alexander Eck is doctoral student at the Institute of Information Management at the University of St.Gallen and research associate at the chair of Prof. Dr. Walter Brenner.

Lic. oec. HSG Stefan Keidel is currently working for UBS AG, Group Technology Infrastructure Services, in the area of IT asset management.

Dr. Falk Uebernickel is assistant professor and project manager at the chair of Prof. Dr. Walter Brenner, Institute of Information Management, University of St.Gallen.

Dr. Thomas Schneider is currently working for UBS AG. He is Head of Global Production Services in Group Technology.

Prof. Dr. Walter Brenner holds a chair for information management and the position of management director at the Institute of Information Management at the University of St.Gallen.