Complexity is dead, long live complexity! How software can help service providers manage security and compliance

Stefan Thalmann a,*, Daniel Bachlechner b, Lukas Demetz a, Markus Manhart a

a University of Innsbruck, Department of Information Systems, Production and Logistics Management, Universitätsstraße 15, 6020 Innsbruck, Austria
b Fraunhofer Institute for Systems and Innovation Research ISI, Competence Center Emerging Technologies, Breslauer Straße 48, 76139 Karlsruhe, Germany

A B S T R A C T

Service providers expected to see a simplification regarding security and compliance management as standards and best practice were applied to complex information technology (IT) outsourcing arrangements. However, security and compliance management became even more complex and is presenting greater challenges to service providers than ever before. In this article, we focus on the work practices of service providers dealing with complex and transitory security requirements and distributed IT infrastructures. Based on the results of semi-structured interviews followed by a think-aloud study, we first describe specific requirements to be met by software supporting security and compliance management in complex IT outsourcing arrangements, and discuss the extent to which existing software already meets them. We show that existing software, which is primarily designed for in-house settings, fails to meet requirements of complex IT outsourcing arrangements such as (1) the use of standardized and formal descriptions of security requirements and configurations, (2) the definition of an interface allowing to exchange messages and to delegate tasks, (3) the provision of mechanisms for designing and implementing a configuration for specific security requirements across organizational boundaries, (4) the provision of mechanisms for verifying and approving the enforcement of these security requirements, and (5) the provision of mechanisms for searching and browsing security requirements, configurations and links between them. We then propose a software architecture that claims to be capable of meeting those requirements and outline how this claim was evaluated by means of another think-aloud study in which potential end users were asked to perform a series of tasks using a prototypical implementation of the architecture. The results of the evaluation confirm that the software meets the described requirements and suggests that it facilitates the management of security and compliance in complex IT outsourcing arrangements.

© 2014 Elsevier Ltd. All rights reserved.
1. Introduction

In recent years, service providers expected to see a reduction in the complexity of their work as a result of the application of voluntary standards and best practice in complex information technology (IT) outsourcing arrangements. Over time, service providers started to outsource parts of their IT themselves which led to distributed IT infrastructures and, in combination with complex and transitory security requirements, to a series of new challenges (Bachlechner et al., 2014a). Business complexity, public scrutiny and the advent of more stringent regulations imply that compliance issues arising from regulatory and contractual requirements become critical for organizations collaborate across organizational boundaries. Communications, outsourcing parts of their IT need to ensure that their service providers are compliant with their requirements. Service providers additionally have to make sure that their service providers also meet their clients’ security requirements. Ensuring (Tracey, 2007) and verifying (Bachlechner et al., 2014b) the fulfillment of requirements are growing challenges for service providers and auditors, respectively.

Currently, generalized audit software (GAS), governance, risk management and compliance (GRC) software, IT service management (ITSM) software and vulnerability assessment (VA) software are used to support security and compliance management in complex IT outsourcing arrangements. These types of software, however, were designed for in-house settings. With respect to IT infrastructures which are distributed across multiple organizations, a lack of software support was detected (Sinclair et al., 2011; Wang et al., 2009). Service providers, however, increasingly rely on software support in the dynamic settings they operate in, particularly if they, like their clients, outsource parts of their IT. Software support is required to, for instance, ensure meaningful audits, manage heterogeneity among services provided by different service providers, and coordinate involved parties (Thalmann et al., 2012).

This article contributes to closing the described gap in software support by identifying software requirements based on an investigation of work practices of service providers acting in complex IT outsourcing arrangements and proposing a software architecture meeting those requirements. First, we outline our procedure and introduce the applied research methods. Then, we describe our approach to requirements engineering in which we provide a description of the key processes performed by service providers when managing security and compliance. Based on our understanding of the work practices and after investigating the current state of software support, we propose a software architecture that claims to be capable of meeting the identified requirements. Finally, we evaluate a prototypical implementation of the architecture.

2. Background

Outsourcing involves transferring management responsibility for the provision of a service from one legal entity to another (Lancellotti et al., 2003). The outsourcing entity typically expects benefits such as reduced costs (Barthélemy and Geyer, 2004), the ability to focus on core capabilities (Linder, 2004) and increased flexibility (Slaughter and Ang, 1996). Outsourcing of IT has been popular for decades, but new forms of outsourcing arrangements such as selective outsourcing (Cullen et al., 2005) and multi-sourcing (Oshri et al., 2009) as well as new procurements models such as cloud computing have started to attract attention recently (Martens et al., 2011). The promised benefits, however, are outweighed by the new challenges which arise as outsourcing arrangements increase in complexity. Among the challenges that are particularly critical are those related to the management of security (Jansen, 2011) and the management of uncertainty arising from concerns over the ability of service providers to sustain outsourcing relationships (Lancellotti et al., 2003). Service providers currently attempt to strike a balance between the individuality and flexibility of the services they provide and their ability to manage increasingly distributed IT infrastructures in compliance with regulatory and contractual requirements. The development of end-to-end security solutions is recommended to deal with this complexity (Papazoglou et al., 2008).

IT outsourcing generally implies that two or more organizations collaborate across organizational boundaries. Complex IT outsourcing arrangements (Vukovic et al., 2012) typically involve multiple service providers and customers (Gallivan and Oh, 1999) as illustrated in Fig. 1. Customers outsource business services to service providers and just like any other organization, service providers can outsource parts...
of their IT to other service providers. Most service providers have multiple clients and thus need to comply with a myriad of individual requirements (Gerber and von Solms, 2001). Compliance is not restricted to security requirements but plays a role in all fields where adherence with requirements is essential. Service providers have to comply not only with the requirements imposed by their clients but also with requirements stemming from laws, regulations and best practice (Daniel et al., 2008; Mulo et al., 2013). Auditors are commissioned to audit service providers and to verify if the client’s requirements are addressed suitably (Jülich et al., 2011).

Requirements imposed by a customer are usually documented in contracts between the customer and its suppliers. Service level agreements (SLAs), which define, among other things, the rules for the provision of services, are commonly specified within such contractual arrangements (Goo, 2010). SLAs play an important role with respect to security in complex outsourcing arrangements (Karakasheh, 2012). Services and SLAs are usually highly standardized and thus barely meet the specific security requirements of the majority of customers (Liang et al., 2006; Sun et al., 2008). To differentiate themselves from their competitors, service providers increasingly adapt the services they provide as well as the contracts to their individual clients’ security requirements (Cao et al., 2006; Mietzner et al., 2009). Almost two decades ago, Kühnhauser (1995) already recognized the ever-increasing multitude of individual security requirements as a major security challenge. As the number of requirements increases (Gerber and von Solms, 2008), keeping track of them becomes a burden. In heterogeneous IT infrastructures, implementing requirements requires the performance of complex configuration management tasks. This, in turn, increases the likelihood of misconfigurations (Berger et al., 2009). Security misconfigurations are a source for security vulnerabilities (Berger et al., 2009; Mundada et al., 2011) that may not only result in data loss or data leakage (Juels and Oprea, 2013) but also in loss of revenue and reputation as well as additional costs (Ahmad et al., 2014). In 2011, for instance, Amazon lost data from its customers due to a mis-configured network device (Amazon, 2011). Similarly, customer data stored in Microsoft’s Business Productive Online Suite could be downloaded due to a configuration issue (PCWorld, 2010). Research shows that additional costs resulting from security breaches can be significant (Campbell et al., 2003).

Breaux et al. (2009) developed a management framework that aims to help keeping track of requirements throughout their life cycle. A recent study indicates that contract complexity is increasing year by year (KPMG, 2012). According to KPMG, contract complexity has increased not only because of the expanding scope of outsourcing and the increasingly global nature of service delivery but also as a result of attempts to achieve goals that are both more complex and more susceptible to concerns related to security and compliance. SLAs in contractual agreements increase in both number and complexity (Wood and Anderson, 2013); they govern organizational structures as well as constrain business processes (Lotz et al., 2008) and they ensure that organizations processing confidential data treat it appropriately (Pauley, 2010). Additionally, it is also not uncommon for service providers to see client requirements conflicting with each other (Casalino et al., 2012).

In complex IT outsourcing arrangements, the growing number of contractual and regulatory requirements makes verifying compliance more difficult (Accorsi et al., 2011); it is no longer a question of being compliant with one, but with a complex web of requirements (Gerber and von Solms, 2008). To support decision makers in finding secure service providers, Ouedraogo and Mouratidis (2013) propose the Complete-Auditable-Reportable (C.A.RE) approach. This well-defined approach helps to select a service provider by assessing its ability to manage the risks it is exposed to. Current guideline-based and mainly manual approaches to security and compliance management quickly reach their limits when faced with complex and transitory requirements in combination with a distributed IT infrastructure. Mulo et al. (2013) propose an approach for implementing an event-based compliance monitoring infrastructure applying a model-based approach. Software support nourishes the hope that current limits can be overcome in the foreseeable future (Sinclair et al., 2011). Montanari et al. (2013) took a first step in this direction by proposing a tool for monitoring and validating compliance in distributed IT infrastructures named Odessa. ESPONERBAC, a model for enforcing access control policies in outsourced parts of the IT (Asghar et al., 2013), is another promising step toward software support for security and compliance management in complex IT outsourcing arrangements. Nevertheless, there is a lack of integrated software for security and compliance management in complex IT outsourcing arrangements.

The goal of this article is to identify requirements for software supporting security and compliance management in complex IT outsourcing arrangements. Prior research focused on specific problems faced by organizations involved in complex IT outsourcing arrangements such as the detection or avoidance of security breaches (Jansen, 2011; Miede et al., 2009). Other approaches such as the ones described by Abdulrahman and Sarfraz (2012) and by Takabi et al. (2010) focus on the negotiation of security requirements among service providers and potential customers (Almorsy et al., 2011). In addition, cloud software, virtualization-based data center management software (Sotomayor et al., 2009) and cloud monitoring approaches (De Chaves et al., 2011) have been discussed in the literature but their focus rather lies on service integration than on security and compliance aspects. Summing up, scattered research on specific aspects can be found in the literature so far rather than an overarching investigation or even an integrated solution approach. To the best of our knowledge, the specific requirements of service providers regarding software supporting the management of security and compliance in complex IT outsourcing arrangements have not yet been investigated.

3. Procedure and research methods

The research problem addressed in this article was formulated in a research and development project in close collaboration with three partners from industry, a medium-sized
service provider in Germany, a large service provider in Spain and the French branch office of an auditing organization which is among the Big Four accounting firms. Overall, we conducted design science research as characterized by Hevner et al. (2004). The four-step procedure we performed to achieve the goals set is illustrated in Fig. 2. Whereas the first two steps constitute the requirements elicitation, the latter two constitute the requirements validation. The multi-method approach was chosen based on the premise that separate and dissimilar data sets drawn on the same phenomena would provide a richer picture (Sawyer, 2001) than would a single-method approach. A sequential design (Hanson et al., 2005; Mingers, 2001) was used. To elicit requirements, semi-structured interviews which informed a subsequent think-aloud study were conducted; to evaluate the requirements, desk research on current software support and another think-aloud study were conducted. Prototypes were used to validate the identified requirements with real end users. The combination of several research methods not only provided a rich context but also allowed taking a considerable number of opinions into account (Kaplan and Duchon, 1988). According to Benbasat and Zmud (1999), empirical research should induce implementable results or stimulate critical thinking among information systems practitioners in design science research. The empirical research we conducted was intended to inform the design of a software architecture.

For the requirements elicitation, we gathered data by means of a series of semi-structured interviews followed by a think-aloud study. The purpose of the interviews that were conducted in step 1 was to gain a deep understanding of how requirements are managed by service providers acting in complex IT outsourcing arrangements and to identify the key tasks in this regard. We conducted 16 interviews with professionals having at least five years of experience in the domain. First, we conducted two face-to-face interviews with senior representatives from the German service provider and the auditing organization which took 2 h each. Then, we conducted 14 telephone interviews of about 1 h each with further experienced professionals from service providers, clients of service providers and auditing organizations. The interviewees were asked to describe the processes in which they are involved, and to comment on how well they consider them performed, how they are supported by software and what they perceive as the main challenges. The initial face-to-face interviews differed from the telephone interviews in that they played a special role in the sampling process for the subsequent telephone interviews (Wengraf, 2001).

Based on the findings of the series of interviews, and assisted by the senior representatives who participated in the initial face-to-face interviews, we identified three tasks: (1) elicit and formalize requirements, (2) implement requirements and (3) analyze and optimize requirements. These tasks, which were confirmed by the senior representatives, are performed by service providers in day-to-day business. Furthermore, they map to the key elements of the compliance management life cycle as it was described by Ramezani et al. (2011).

In step 2, we asked professionals to perform these three tasks in a think-aloud study. Think-aloud studies are known as an effective method used by system designers to investigate requirements (Wright and Monk, 1991) and work practices (Hughes and Parkes, 2003). They require participants to verbalize their thoughts while performing defined tasks and usually comprise three phases (Hertzum and Jacobsen, 2001): (1) a preparation phase during which participants are familiarized with the assessment and the work environment, (2) test sessions, where participants are recorded whilst performing the required tasks and thinking out loud as they are doing so, and (3) a debriefing of participants. Six representatives of the German service provider participated in the think-aloud study. First, we provided an overview of the study procedure and made sure that the participants were familiar with it. Among the participants were a security manager, a systems architect, an internal auditor, a customer relationship manager, a supplier relationship manager and a system administrator. Then, we asked the participants to verbalize their thoughts while performing the three tasks using real business data and the software they also use in day-to-day business. After performing the tasks, we clarified issues that were raised by the participants. Finally, we debriefed the participants. All sessions were audio and video recorded.

To check the extent to which the identified requirements for software support are fulfilled by software currently available, we conducted desk research on current software support for security management in step 3. We first investigated the types of software available to support security and compliance management, taking into account recent market studies (e.g., McClean et al., 2011; Oehrlich et al., 2010) as well as academic literature (e.g., Ahmi and Kent, 2013; Singleton, 2006). We identified four types of software typically used by service providers: GAS as well as GRC, ITSM and VA software. We then investigated the features of the software types to check whether they support the five requirements. We identified 18 GAS solutions as well as 11 solutions for GRC, 4 for ITSM and 7 for VA software. Subsequently, we investigated the degree to

![Fig. 2 – Procedure.](image-url)
which the software solutions met the five requirements. As proposed by Knopf (2006), we (1) described the content and conclusions of the individual contributions and (2) summarized the collective results. Guided by our empirical results, we then propose a software architecture based on nine key components. The architecture claims to address the described software requirements and represents our main design artifact. The software architecture is implemented prototypically to serve as vehicle for the evaluation of our design artifact.

Finally, in step 4, we evaluated the proposed software architecture in another think-aloud study with potential end users at the premises of the Spanish service provider. The evaluation was based on six prototypical implementations developed on the basis of the requirements identified in step 2. For each prototype we conducted a separate think aloud session. Each session comprised three parts: First, a short demonstration and joint walk-through guided by the prototype developers was performed. In 15–30 min the key features of the prototype to be evaluated were presented. Second, the potential end users performed predefined tasks with the prototype verbalizing their thoughts. This took between 1 and 2 h for each prototype. The tasks were specified in collaboration with the prototype developers and were exemplary cases for the key tasks identified in step 1. All walk troughs and think-aloud sessions were audio and video recorded. Third, we performed debriefing group interviews in which the prototypes were discussed. One interview was conducted for each prototype. The interviews took between 20 and 50 min and were audio recorded. The potential end users were asked about usability (perceived ease of use, operability, understandability, learnability, attractiveness), as well as the organizational impact and behavioral change precipitated by using a software product based on the prototype within the end user’s organization.

All parts described can be assigned to typical steps for a think-aloud study as described by Seel et al. (2008). In total, six potential end users (a security manager, a system administrator, a presales manager, an internal auditor, as well as two IT auditors) and six prototype developers as well as two interviewers participated in the evaluation. We deemed it as important to also incorporate the view of two IT auditors as they have also been considered as users of the prototypes in the course of IT audits. Further, auditors are frequently engaged as consultants to establish a control design. Hence, we considered their feedback as important. Given the number of the end users, a think-aloud study seemed to be the best method for the evaluation (Clemmensen et al., 2009).

We analyzed the recordings of the semi-structured interviews and the think-aloud studies following the guidelines specified by Silverman (2005). More specifically, we applied qualitative summarization to extract the facets relevant to our research goals. Both the interviews and the think-aloud studies were transcribed and coded. The purpose was to make valid inferences from the content gained through the qualitative inquiry (Weber, 1990). The analysis of the video recordings of the think-aloud studies allowed us to consider the participant’s actions which complemented the spoken explanations of their actions. We applied an open coding technique in which we assigned codes representing certain concepts to the notes. Finally, we reflected on the results in a series of conference calls with representatives from the involved organizations.

4. Requirements engineering

To identify requirements for software supporting security and compliance management in complex IT outsourcing arrangements, we analyze current work practices and discuss potentials for improvement.

4.1. Key tasks and underlying processes

We identified the underlying processes for each of the key tasks based on the study participants’ statements on both the processes’ relevance and their suitability for software support. The following tasks were not only confirmed by the senior representatives of the German service provider and the auditing organization but also represent key elements of the compliance management life cycle as it was described by Ramezani et al. (2011).

4.1.1. Elicit and formalize requirements

In complex outsourcing arrangements, this task is mainly about negotiating contractual details. Negotiating contractual details heavily depends on human actors and the potential for software support thus was considered low. An interviewee said, “That’s mainly about contracting and understanding the contractual details.” However, service providers need to check if their clients’ requirements can be enforced. Therefore, it needs to be verified if new requirements are in conflict with existing requirements. Currently, this is done manually and perceived as laborious and error prone. This verification requires an analysis of the entire IT infrastructure and its configuration. The study participants reported a considerable need for software support in this context. One interviewee stated, “Checking if we can fulfill the customer requirements is a really ambiguous job with a substantial risk of errors.” Consequently, we attested high potential for improvement through software with respect to the detection of requirement conflicts.

4.1.2. Implement requirements

This task comprises both the design and the implementation of a configuration which enforces a set of requirements. It was mentioned in the study that the design of a suitable configuration is considerably more challenging than its implementation which is already supported quite well by software. As possible side effects of changes to the configuration need to be taken into account, enforcing new requirements is highly complex and would thus clearly benefit from software support. An interviewee reported, “Currently, we mainly rely on experience and gut feeling, however a more systematic approach would be desirable.”

4.1.3. Analyze and optimize requirements

This task comprises preparing and executing IT audits as well as exploiting their results. The study indicates that determining whether one or several requirements are not enforced is particularly laborious and error prone. Detecting
misconfigured IT infrastructure elements is a particularly time-consuming aspect in this context. An interviewee said, “Checks are typically performed by sampling and we engage junior staff for performing this boring job.” At the same time, there seems to be potential for improvement through software support. Manually performed checks could be automated not only to increase the reliability of the detection of misconfigurations but also to increase its efficiency.

4.1.4. Detect requirement conflicts
In essence, detect requirement conflicts, enforce requirements, and detect misconfigurations are the three processes we focus in the following. Our partners from industry stressed not only the processes’ high business relevance but also their suitability for software support. The processes are interim results and provide a point of reference for identifying software requirements. The processes are closely linked with each other and cover a considerable portion of the superordinate security and compliance management process.

Requirement conflicts have to be detected whenever the set of requirements to be enforced by an organization changes. An organization’s set of requirements has to be conflict-free before it can be enforced by configuring the relevant IT infrastructure. Finally, whenever the IT infrastructure of an organization changes, it has to be checked for misconfigurations. In complex IT outsourcing arrangements, an organization’s IT infrastructure is typically spread across multiple sites. The interviews as well as the think-aloud study provided deep insight into how this works in practice. In the following, we describe the three processes in detail:

The aim of this process is to detect conflicts in a set of requirements. The process is typically performed whenever a requirement is added or changed. In the interviews, it was stated that in case of a new customer requirement, the customer relationship manager usually becomes aware of the new requirement during contract negotiations. Because conflicting requirements cannot be enforced, it is necessary to detect requirement conflicts as early as possible.

In the think-aloud study, the customer relationship manager received a new requirement to be met by a specific business service from a customer. Together with the system architect, the customer relationship manager first studied the business service’s composition to identify the sites involved in the provision of the service. This was necessary as the business service was found to be composed of both services provided in-house and services provided by service providers. With respect to the services provided in-house, it was the security manager’s responsibility to check whether there were any conflicts. Further, it turned out that the detection of requirement conflicts was not a straight forward process. The security manager, supported by the system architect, first had to identify the relevant services provided in-house as well as the underlying IT infrastructure elements. The security manager noted, “It is about to determine which components use this configuration.” Then, the security manager investigated whether the configurations of the IT infrastructure elements can be changed in a way that not only the new but also the existing requirements are met. The security manager could find such a configuration in the think-aloud study after some time. Apart from that, the security manager checked which of the other sites involved also had to enforce the new requirement. It was indicated in the interviews that this critical sub-process is typically performed in a similar manner also at other service providers. Then, the supplier relationship manager forwarded the relevant aspects of the new requirement to the other sites involved and requested confirmations that there were no conflicts. The security manager documented the results of the in-house conflict detection and merged it with the results of the conflict detections performed at the other sites. Finally, the security manager forwarded the overall conflict status to the customer relationship manager.

The security manager explained that the detection of requirement conflicts is a cascading process in which, depending on the business service’s complexity, quite a lot of sites can be involved on various tiers. The process ends as soon as all involved service providers have reported their requirement conflict status to the customer relationship manager who triggered the process. Only if all confirm that there are no requirement conflicts in their areas of responsibility, the changed set of requirements can be considered conflict free.

4.1.5. Enforce requirements
The aim of this process is to find a configuration which is fully compliant with a set of requirements and to configure the relevant parts of the IT infrastructure accordingly. The identification of possible configurations already happens within the requirement conflict detection process. It was stressed in the interviews, however, that the enforce requirements process also takes other criteria such as performance and cost into account. The interviewee noted, “It is not just about effectiveness – we also want to be efficient.” The detect requirement conflicts process focuses on compliance only. Apart from that, the actual configuration of the IT infrastructure is also something that is unique to the enforce requirements process. The enforce requirements process is also a cascading process in complex IT outsourcing arrangements. The sub-processes where the involved sites as well as possible IT infrastructure element configurations are identified and where tasks are forwarded to other service providers do not differ much from the ones performed within the detect requirement conflict process. In the think-aloud study the supplier relationship manager certainly requested confirmations of enforcement. A major difference, however, was that within the enforce requirements process, one of several compliant configurations had to be selected and implemented. Besides the security manager, the internal auditor was involved in the selection of the configuration to be implemented. The actual implementation was performed mainly manually by the IT administrator. Finally, the security manager forwarded the overall enforcement status comprising the in-house results as well as the results of the requirement enforcements performed at the other sites to the customer relationship manager.

4.1.6. Detect misconfigurations
The aim of this process is to detect misconfigurations of the IT infrastructure meaning that one or several requirements are not enforced. In practice, the detection of misconfigurations is usually performed in accordance with organizational guidelines. Apart from that, it was indicated in the interviews, that
security managers trigger this process whenever they deem it necessary, for instance when the IT infrastructure was changed substantially. Detecting misconfigurations, as the other two processes, is a cascading process.

In the think-aloud study, the participants were asked to check the compliance of a business service with respect to a set of requirements. The sub-process where the involved sites are identified does not differ from the one performed within the detect requirement conflict or the enforce requirements process. The internal auditor then requested audit reports from the involved service providers. After requesting the reports, the internal auditors commented, “That is basically what we do, when we do not really understand what is going on.” The internal auditor explained that this was necessary because there were no valid audit reports already available to analyze. The internal auditor then identified the relevant services provided in-house as well as the underlying IT infrastructure elements. Then, the internal auditor identified possible configurations of IT infrastructure elements (to-be configurations) and compared them with the actual configuration (as-is configuration). Subsequently, the supplier relationship manager forwarded the relevant aspects to the other sites and requested confirmations that there were no misconfigurations. The internal auditor documented the results of the in-house misconfiguration detection and merged it with the results of the misconfiguration detections performed at the other sites. Only if all confirm that there are no misconfigurations in their areas of responsibility, the IT infrastructure can be considered configured properly.

4.2. Potentials for improvement

The analysis of the data collected by means of the think-aloud study focused on potentials for improvement in general and the potential for software support in particular. The following four areas which would benefit in particular from improvements emerged during the data analysis.

4.2.1. Automation of tasks

Many of the activities are currently performed manually and in a time-intensive way, which is not only costly but also error-prone. In the think-aloud study, for instance, to-be configurations were manually compared with as-is configurations to detect misconfigurations using two computer screens.

4.2.2. Standardized reporting

Major information objects such as requirements and configurations are in different formats and spread across different repositories. Employees are often forced to create overviews and reports, which are mostly compiled in a manual procedure, by themselves. This is again error prone as it easily causes inconsistencies. In the think-aloud study, for instance, the security manager created an overview of the IT infrastructure manually in preparation for a customer meeting.

4.2.3. Standardized data exchange

The communication among the sites involved in the provision of a business service is laborious due to the absence of standardized descriptions of requirements and configurations. This applies to the communication with both customers and suppliers. In the study, such miscommunications were observed when the ones involved had to coordinate certain things several times.

4.2.4. Formalized and documented procedures

The execution of processes highly depends on the expertise and experience of the employees involved. This is due to the lack of documented and formalized rules and procedures. While this may not be particularly problematic in in-house settings, it definitely is in the context of complex IT outsourcing arrangements where employees can no longer grasp decision problems as a whole easily. The limited traceability of decisions makes error handling and the verification of decisions particularly difficult.

4.3. Software requirements

Based on the potentials for improvement outlined above, we identified five software requirements (RQ) for security and compliance management in complex IT outsourcing arrangements.

4.3.1. RQ1 – standardize and formalize

The requirement refers to the specification of standardized and formal descriptions of requirements and configurations. RQ1 is particularly relevant for complex IT outsourcing arrangements. Security requirements need to be described in a standardized and formal way. ‘Standardized’ means that a widely agreed representation is used, and ‘formalized’ means that semantic reasoning is possible. This is necessary to facilitate detecting conflicts, enforcing requirements as well as detecting misconfigurations across multiple sites. Such descriptions are also useful for organizations to automate related processes internally but particularly to provide service providers with information about security requirements that they need to fulfill. Standardized and formal descriptions of configurations allow further automation of internal processes. This is indispensable if software support were to be used for detecting misconfigurations. Across multiple sites, such descriptions become even more important when service providers are expected to make their IT infrastructures transparent to their clients. In such cases, providing information about the enforcement of security requirements in terms of the particular configuration of the IT infrastructure is not necessary but desirable. If provided, however, configurations can be optimized across organizational boundaries.

4.3.2. RQ2 – communicate and delegate

RQ 2 refers to interfaces between customers and suppliers allowing to exchange messages (standardized, formal descriptions of security requirements and configurations) and to delegate tasks to other sites. Whereas RQ1 is, to some extent, also useful for automating internal processes, RQ2 is relevant only in outsourcing arrangements. Communicating standardized and formal descriptions of security requirements and configurations is only possible if the software in place at the involved sites is able to read and process such messages. For exchanging messages, appropriate interfaces are needed.
Task delegation allows to coordinate tasks between sites. Both exchanging messages and delegating tasks allow a seamless integration of all service providers involved in the provision of a business service.

4.3.3. RQ3 – design and implement
This requirement refers to a mechanism for designing and implementing a configuration for a specific security requirement. RQ3 is particularly relevant within the context of defining suitable configurations for security requirements. Possible configurations of the IT infrastructure enforcing a specific security requirement are identified, then one configuration is selected and implemented. A mechanism for designing and implementing a configuration for a specific security requirement is valuable for organizations with a complex, internal IT infrastructure but becomes indispensable if specific security requirements need to be enforced in a distributed IT infrastructure. In this case, however, standardized, formal descriptions of requirements and configurations (RQ1) and interfaces allowing their exchange (RQ2) are both prerequisites.

4.3.4. RQ4 – verify and approve
RQ 4 refers to a mechanism for verifying and approving the enforcement of security requirements by means of the IT infrastructure configuration. This requirement is particularly relevant for detecting misconfigurations. Possible-to-be configurations of the IT infrastructure enforcing a specific security requirement are identified and compared with the as-is configuration. If the as-is configuration can be verified as corresponding to one of the to-be configurations, the enforcement of the security requirement by means of the IT infrastructure configuration is approved. This mechanism is particularly useful in case of a distributed IT infrastructure. However, formal descriptions of requirements and configurations (RQ1) and interfaces allowing their exchange (RQ2) are prerequisites in this case.

4.3.5. RQ5 – search and browse
This requirement refers to a mechanism for searching and browsing requirements and configurations as well as the links between them. RQ5 is relevant because the design and implement mechanism (RQ3) as well as the verify and approve mechanism (RQ4) still require human interactions. Therefore, individuals need to be able to search and browse information available on requirements and configurations within and across organizational boundaries.

5. Proposal for a software architecture

Before proposing and evaluating a software architecture, we investigate whether and how software solutions currently available on the market support organizations with respect to the elicited requirements. The reviewed literature covers market studies as well as academic literature.

5.1. Current software support

GAS is used by service providers for internal audit purposes (Flowerday et al., 2006). GRC software provides an integrated, organization-wide approach for ensuring that organizations act in compliance with their requirements through the alignment of strategy, processes, technology and people (Racz et al., 2011). ITSM software leverages a business view of the IT and enables organizations to quickly resolve or escalate issues and problems, improve root cause isolation, and provide higher levels of business user satisfaction (Brooks and Greene, 2012). VA software analyses the security of systems and services and detects known vulnerabilities (Guo et al., 2005). Table 1 shows whether and how software currently used addresses the requirements we identified.

With respect to RQ1, software currently available offers integration within organizations, but lacks a standardized description of security requirements and configurations. Different XML-based languages to standardize the communication of security requirements as well as configurations have been developed. Their purpose is to, for example, standardize the representation and processing of authorization policies (XACML), policy related information (SAML), or representing the configurations for filtering, channel protection, and access control policies (PCIM). These standards cover very specific partial aspects of standardization. Our architecture, however, does not limit its scope to specific security aspects like access control or authorization. However, we could not find any overarching standard. This absence of a widely agreed representation and formalized descriptions leads to cumbersome manual work as no automation is possible. Another issue with current software support is the lack of interfaces to exchange messages across organizational boundaries.

### Table 1 – Requirements addressed by current software.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>GRC</th>
<th>ITSM</th>
<th>GAS</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standardize and formalize (RQ1)</td>
<td>Not supported</td>
<td>Not supported</td>
<td>Not supported</td>
<td>Not supported</td>
</tr>
<tr>
<td>Communicate and delegate (RQ2)</td>
<td></td>
<td>Web portals for customers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Design and implement (RQ3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Verify and approve (RQ4)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Search and browse (RQ5)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
boundaries (RQ2). Some GRC solutions offer the possibility to integrate third-party technology for data exchange. Lacking standardized and formalized descriptions of security requirements and configurations increases the need for such interfaces. Some ITSM software solutions support the communication of service requests from clients. However, this functionality fulfills only parts of RQ2 and is not useful to delegate tasks to other sites. With respect to RQ3, we found that GRC as well as ITSM software is appropriate to define, catalog and implement security requirements. However, this is only possible within organizational boundaries. With respect to RQ4, GRC software allows to verify and approve controls, as well as to determine the compliance of as-is configurations with security requirements. Some ITSM software solutions also support the verification and approval of configurations. Further, ITSM software provides an overview of the IT infrastructure and allows the definition, documentation and cataloging of processes, risks, controls and control objectives. VA software can be used to verify network configurations. However, GAS is the main type of software for supporting the verification and approval of the enforcement of security requirements. It supports the verification of configurations against security requirements. With respect to RQ5, GRC and VA software provides central repositories that can be queried for risks. Especially GRC software also offers central repositories with information on requirements, controls, processes, infrastructure elements, laws, regulations and best practices. ITSM software provides support for querying and visualizing information on services and IT infrastructure elements within organizational boundaries. GAS also provides central repositories to gain insight into the compliance with security requirements.

In a nutshell, currently available software supports the management and implementation of security requirements quite well within organizational boundaries, but is stretched to its limits in complex IT outsourcing arrangements. It focuses on decision support or process automation with respect to RQ3 as well as RQ4 for individual organizations. The software, however, generally lacks features with respect to RQ1 as well as RQ2. Most of the discussed types of software meet RQ5.

5.2. Software architecture

The results of our first think-aloud study indicate a lack of adequate software support for service providers acting in complex IT outsourcing arrangements. In our joint research project with partners from academia and industry, we aim to develop design artefacts which are not only relevant in the light of the existing body of scientific knowledge but also useful for our industry partners (Osterle and Otto, 2010). Our empirical research informed the subsequent design work. In this sense, the described software requirements provide the basis for the proposal of a software architecture (Baskerville and Pries-Heje, 2010). Fig. 3 shows the proposed software architecture consisting of nine key components whose development was guided by our empirical results. The architecture claims to meet the described software requirements.

Two components are interfaces, one for exchanging messages with customers and the other one for exchanging messages with suppliers. Messages to other sites are exchanged in a standardized and formalized way. Suitable approaches in this respect are offered by the specification language proposed by Damianou et al. (2001), the predicate logic approach described by Van der Aalst et al. (2011) and the model-based approach put forth by Breu et al. (2008) and by Mulo et al. (2013). A standardized and formalized representation makes it feasible to semi-automatically compute messages which is particularly important for the task management component. The component is capable of processing incoming messages and delegating tasks to other components or sites via interfaces. The work of Sillaber et al. (2013) offers a viable avenue for documenting and exchanging security requirements. The task management component queries the central repository to identify the sites relevant for a specific business service. The central repository lies at the heart of the proposed architecture and stores standardized descriptions of all security requirements and, in case of in-house settings, the configurations of the relevant parts of the IT infrastructure. The repository goes beyond existing configuration management databases as it also stores the links between security requirements and configurations. Arsac et al. (2013) propose
an architecture for developing holistic policy chains that link security requirements with the corresponding configurations. In case multiple sites are involved, the audit reports of the service providers as well as the contracts including the SLAs are stored. Additionally, the compositions of all business services and the responsible sites are stored. The repository can be accessed via the search and browse component.

In case new security requirements need to be implemented in-house, a task is delegated to the design component. This component proposes a new control design based on the current configuration of the IT infrastructure. Therefore, the component checks new and existing security requirements for conflicts. Due to the standardized description, an optimization of this design task becomes feasible. Basile et al. (2010) present two approaches to provide decision support for this design problem by using optimization models. The design component delegates the implementation of the selected configuration to the implementation component. This component configures the IT infrastructure automatically according to a specified to-be configuration. In this respect, Basile et al. (2010) present an approach aimed to assist IT administrators in configuring network devices according to security requirements.

The task management component delegates the confirmation that security requirements are properly enforced. The approve component determines the control design and all corresponding configurations. Building on formalized descriptions and predicate logic, Van der Aalst et al. (2011) propose a conformance checker performing this approval continuously. In this respect, Accorsi et al. (2011) present another approach to automatically analyze workflows to support the certification. The verify component then compares the as-is configuration with the to-be configuration for every requirement. Van der Aalst et al. (2010) describe an approach which supports the verification by means of a combination of process mining and petri nets.

To deploy the proposed software architecture, we consider the following issues critical: First, service providers need to assure that all data is fed correctly into the repository which includes the development of a functional system model describing the business and the IT environment of the respective service provider. This model comprises business and infrastructure layers because software developed on the basis of the architecture described above needs to gather information on each of the layers. Service providers should be prepared to handle the problem of fragmented configuration management databases holding only parts of the infrastructure information. Adopting the DMTF standard CMDBf1 could be useful to solve related issues. Further, creating the functional system model requires to create links between the different layers. Second, service providers need to identify all enforcement mechanisms to ensure that they cover all relevant configurations of the infrastructure elements. Third, service providers that want to implement the architecture need to develop an authorization concept defining the availability of the different functions to internal and external stakeholders. Finally, service providers need to ensure that the efforts described above are outweighed by the benefits the architecture brings. Simulations or scenario analyses could be useful approaches in this regard.

5.3. Prototype evaluation

We evaluated the software architecture in a think-aloud study with potential end users at the premises of the Spanish service provider. To increase the validity of our results, we selected a service provider for the evaluation which was different from the one used for the requirement elicitation. The evaluation was based on six prototypes implementing alone or in combination at least one of the architecture components. The prototypes were evaluated successively. The idea behind this evaluation was to provide potential end users with prototypical implementations of the architecture components, allow them to gain hands-on experience, and collect feedback on their impressions.

The prototype realizing the search and browse component links business-layer information to technical information. It uses a central repository and allows to identify the compliance status of all implemented security requirements and to query each requirement's relevance for business processes and IT infrastructure elements (Sillaber et al., 2013). The pre-sales manager stated that “it prevents you from manually searching for distributed documentations”, whilst one of the IT auditors highlighted that the “centralization of all the information is a major benefit [as] collecting the audit evidence is usually difficult”. This centralization is achieved by the help of a central repository which stores information relevant to all architecture components.

The verify and approve component is implemented by three prototypes. One prototype supports creating checklists which are required to compile automated checks for detecting misconfigurations. The security manager noted that the prototype helps “to shorten the time to get results, saving time and resources to achieve it”, especially if templates for checklists are provided. The second prototype allows checking whether usage logs indicate violations of specified rules. The prototype shows the compliance status with respect to the rules. The internal auditor stated that “usually you do a sample and hence some cases are not covered” as well as that “without the tool you can lose some violations”. The third prototype allows to check parts of IT infrastructure for vulnerabilities. Using information from a public Common Vulnerabilities Exposures (CVE) dictionary, the prototype creates, if possible, an attack graph for a given configuration. In addition, the prototype provides remediation actions such as changes to the configuration and shows the changed attack graph assuming a remediation activity was applied. The security manager stated that the prototype “certainly will boost vulnerability issues, time for assessment and time for remediation”, as well as “if properly implemented, it will allow much more confidence in outputs”. The three prototypes realizing the verify and approve component help to understand the IT infrastructure and the relationships between its elements. The individual prototypes were discussed in detail by Bettan et al. (2012) and Verbeek et al. (2010).

The design and implement component was realized by one prototype. To design controls for a certain security requirement, the prototype uses an optimization algorithm taking into account factors such as security level, performance and

---

1 http://dmtf.org/standards/cmdbf.
costs. Existing information on infrastructure elements, controls and requirements can be accessed through the central repository. The prototype is currently able to work with authentication and authorization requirements. The prototype was discussed in detail by Basile et al. (2012). Regarding the prototype’s usage, the security manager stated that once the tool is understood, end-users “can spare time with this tool”. Furthermore, the security manager reported, “if I want to achieve the same thing, I have to use different tools today, and that’s what I like [about this prototype]” and “the enforcement of security that you can provide is really helpful […] it really enables to increase the security posture”. The system administrator stated that checked configurations are “all linked-up to the requirements”. This would improve the work of the end users as currently there is “no link to the IT, to the landscape, to the requirements, to make that picture as it is shown by the prototype”.

The customer interface component and supplier interface component are implemented by one prototype in form of a portal, collecting all the key information provided by the different components and displaying this info by means of dashboards. Depending on the role of an end user, dedicated views providing selected targeted information is displayed. The presales manager stated that this would help customers in negotiations with a “provider […] because it is easier to check what would be the problems, what would be the things in which they have to spend money to fix” and that “it’s a useful tool to quick check about SLAs, compliance, work in progress”. Further, a security manager stated, the prototype can “definitely be helpful to set such overall information in such quick and centralized way”. For a detailed description of this prototype please see Gallego-Nicasio Crespo et al. (2013).

The task management component as a back-end component was not in the focus of the evaluation by means of a think-aloud study.

6. Summary and outlook

In this article, we described the particular work practices of service providers dealing with highly complex and transitory security requirements, and distributed IT infrastructures. Our results indicate that current work practices are cumbersome and that the software currently used does not come close to satisfying the service providers’ requirements. To support the time-consuming and error prone work currently performed, appropriate software is needed, in particular, as the complexity can be expected to increase even further (Bachlechner et al., 2014a). This increase is a consequence of the trend that services are increasingly built by combining other services (Kulkarni and Dwivedi, 2008), additionally boosted by the steady growth in the number of services available (Kraemer et al., 2009).

Fig. 4 provides an overview of how the components of the proposed software architecture emerged from the potentials for improvement. The key processes served as foundation for the think-aloud study and they also provide the contextual framework for our results. First, the potentials for improvements and the requirements for software support were identified. Then, the components of the software architecture meeting the requirements and supporting service providers in executing the key processes were specified after investigating how the requirements are met by software currently available.

Specific types of software used by service providers such as GRC, ITSM and VA software or GAS work fine for individual organizations. However, they barely meet the specific requirements of service providers dealing with manifold security requirements which have to be enforced in complex IT outsourcing arrangements. We argue that several requirements have to be met which include (1) standardized and formal descriptions of security requirements and configurations, (2) interfaces allowing the exchange of messages and delegation of tasks between different sites, (3) mechanisms designing and implementing a configuration for a specific security requirement, (4) mechanisms verifying and approving the compliance of configurations with security requirements, and (5) mechanisms allowing to search and browse security requirements and configurations as well as the links between them. Based on these requirements, we developed a software architecture in close cooperation with partners from industry. Our industry partners concluded that

---

**Fig. 4 – Trace of data analysis.**
the proposed architecture “will facilitate [their] work, especially by automating a certain number of everyday tasks.” They further stated that the software “brings added value compared to the GRC tools that already exist.”

One might consider the rather small sample for the think-aloud studies a limitation but this is not uncommon for qualitative in-depth investigations. For several reasons, we are confident that our results can be generalized to a certain extent. First, we checked the relevance of the investigated processes in a series of interviews. Second, guidelines and standards in the domain also consider the processes we investigated central since they embrace all of the key processes relevant to security and compliance management. Third, we also discussed the results with auditors who provided valuable insights into the work practices of service providers. Fourth, we validated the architecture with a service provider not involved in the requirements elicitation. Finally, the investigation of existing tools also showed that our results are in line with the current state of practice.

In future research, the prototypical implementations of the architecture components will be integrated and evaluated using a larger sample. Network effects generated by using the architecture components will be integrated and evaluated. First, we checked the relevance of the investigated processes in a series of interviews. Second, guidelines and standards in the domain also consider the processes we investigated central since they embrace all of the key processes relevant to security and compliance management. Third, we also discussed the results with auditors who provided valuable insights into the work practices of service providers. Fourth, we validated the architecture with a service provider not involved in the requirements elicitation. Finally, the investigation of existing tools also showed that our results are in line with the current state of practice.

In future research, the prototypical implementations of the architecture components will be integrated and evaluated using a larger sample. Network effects generated by using the software across organizational boundaries will play a significant role with respect to the architecture's success. The interface components lay the ground for the exploitation of network effects. To show the economic benefit of our architecture, we plan to develop an economic model considering not only organization-internal factors such as the costs for executing security and compliance processes but also factors such as network effects occurring from its usage across organizational boundaries.

Appendix A. Supplementary data

Supplementary data related to this article can be found at http://dx.doi.org/10.1016/j.cose.2014.05.012.

References


Amazon. Summary of the Amazon EC2 and Amazon RDS service disruption in the US East Region; 2011.


Brooks JM, Greene J. Magic quadrant for IT service support management tools. Gartner; 2012.


Brooks JM, Greene J. Magic quadrant for IT service support management tools. Gartner; 2012.


Singleton T. Generalized audit software: effective and efficient tool for today’s IT audits. ISACA J 2006;7(2). 

Stefan Thalmann holds a PhD in Information Systems from the University of Innsbruck, Austria. He worked as researcher at the Martin-Luther-University of Halle-Wittenberg, Germany, and as visiting researcher at the London Metropolitan University, UK and at the University of Jyväskylä, Finland, and is currently employed as assistant professor at the University of Innsbruck. He has published articles in journals and conference proceedings on knowledge management and knowledge protection, information security and compliance management as well as on technology-enhanced learning.

Daniel Bachlechner holds a PhD in Information Systems from the University of Innsbruck, Austria. He was employed as researcher and teaching fellow at the University of Innsbruck until recently and is now working as senior researcher and project manager at the Fraunhofer Institute for Systems and Innovation Research ISI in Karlsruhe, Germany. He has published articles in journals and conference proceedings on information security management, knowledge management and Semantic Web technologies.

Lukas Demetz holds an MSc in Information Systems from the University of Innsbruck, Austria. He is currently working as a research assistant at the University of Innsbruck, Austria, where he also works on his PhD focusing on economic aspects of information security and compliance management. He has published articles in conference proceedings on online social networks and information security and compliance management.

Markus Manhart holds an MSc in Information Systems from the University of Innsbruck, Austria. Since November 2010 he works as a scientific project assistant at the Information Systems Unit of the University of Innsbruck, School of Management and is currently involved in the EU FP7 research projects “PoSecCo” and “Learning Layers”. He also works on his PhD focusing on knowledge protection in business networks.