Critical success factors for a SOA implementation

A case study in the financial sector

W. Vegter
woutervegter@gmail.com

ABSTRACT
This paper tests critical success factors for a SOA implementation. To this end the design principles of a SOA are discussed and the benefits of a SOA are disussed and critical success factors for a SOA implementation are proposed. Consequently these critical success factors are tested by means of a case study at a Dutch insurance company. In the discussion that follows it is found that engaging in SOA governance and focusing on reducing complexity are critical for the success of adopting a SOA. Focusing on reusability is not found to be critical for the success of adopting a SOA.

Keywords
SOA, CSF, Implementation, Financial sector

1. INTRODUCTION AND RESEARCH
The integration paradigm SOA (Service Oriented Architecture) has established a firm base in the IT of organizations. Gartner predicted with 70% reliability that SOA will be used in more than 80% of new, mission-critical applications in business processes in 2010 [1]. Despite the increasing use, there is still insufficient understanding for the implementation processes used in organizations. Service-oriented development projects are characterized by complexity [2] and misconceptions [3, 4], mainly caused by the fact that SOA is a relatively young phenomenon. This creates a strong need for more knowledge about SOA and standardized SOA implementation methodologies [5]. This knowledge defines which specific issues are important at what time, as several authors already tried to create [2, 6, 7, 8]. But most of these methodological theories are in their early stage, resulting in the little amount of case studies that support these methodologies.

Consequently, there is need for more empirical evidence in order to test and refine these methodologies [9, 10].

1.1 Research question
This research addresses the need for more empirical evidence by conducting a case study of a SOA implementation in an organization. The theory of Rockart [11] about Critical Success Factors (CSFs) will be used to gain insight in what makes a SOA implementation successful. CSFs are defined as the areas in which good performance is necessary to ensure attainment of organizational goals.

The aim of this research will be the search for CSFs and the validation of them. The validation will be based on empirical evidence from a case study.

Because of the extensive literature on general IT-projects in organizations, this research will focus solely on SOA specific CSFs that influence the outcome of a SOA implementation, see figure 1. In this sense CSFs are seen as latent variables that are inferred by analyzing organization-specific variables.

Figure 1: Focus of the research is on SOA specific CSFs

The main research question this paper tries to answer is: “What are the Critical Success Factors for a successful outcome of a SOA implementation project?”

1.2 Research method
Based on the literature, section 2 explains the principles of a SOA and section 3 elaborates on the benefits that SOA brings. This will form a base for discussion about CSFs in a SOA implementation. In section 4, the CSFs are introduced and consequently operationalized so that they can be used in this research. In section 5 it is hypothesized that the CSFs of section 4 are most important for the success of a SOA adoption within an organization. Then this hypothesis is tested by means of a case study. In section 6 this paper ends with conclusions and recommendations for further research.

2. SOA
Erl [6] and Gartner [12] use the metaphor of an organization to describe the paradigm of a SOA. On the lowest level, an organization is structured as a set of services. There can be a cleaning service, customer service, research & development services, delivery service, and legal service etcetera. Each of these services can be individually requested by other employees in the organization that needs them. These services are logically composed: It would for example be inefficient for the legal service to regulate the use of the cleaning service. That is the job of the maintenance service. The way in which services communicate should be specified (e.g. by phone or by email) and all services should decide for themselves on the execution of their work. Together these services make up a single organization with a certain mission and goals.

In the same way the IT department can be seen as a service provider to other business units. The IT department provides different services to different business units. The services offered can be used to support processes or simple tasks. A service that supports a process is composed of services that provide the execution of a single task. These services in turn are usually services that communicate with backend systems to exchange the necessary information. The composition of all these services together is called service orchestration. The
paradigm of orchestrating services in a specific manner to support business activities is called a Service Oriented Architecture.

2.1 Services
The building blocks of this SOA paradigm are services. A service is defined as an IT realization of some self-contained business functionality [13]. It either provides information or facilitates a change to business data from one valid and consistent state to another [9]. The right combination of services provides support for business activities. Ideally, a service has the following characteristics [6, 12, 13]:

**Loosely coupled.** This reduces the interdependencies in a system between services. It allows software on each side of the conversation to change without impacting the other [9] through increasing scalability, flexibility and fault tolerance. It contributes to service composability, autonomy, statelessness and reusability [6]. Loose coupling is achieved by implementing things such as asynchronous communication style between services, weak type system and a distributed control of process logic [13].

**Defined by a contract.** A service contract holds a specification of the semantic behavior of a service. Through pre conditions it states what conditions the requester of the service has to fulfill before requesting the service and through post conditions it states properties of the system after service has completed running [6, 13]. Defining a contract helps a service to be abstract, loosely coupled, composable and discoverable [6].

**Abstract.** This means that the only information a service gives to the outside world is its contract. Being abstract helps to separate the implementation logic from the metadata. The internal logic can be changed without problems while at the same time leaving the contract untouched. Abstraction increases reusability [6].

**Reusable.** Once a service is running, it can be accessed by multiple requesters. Reusability avoids redundancy, which is a common goal of software development [13]. This also shares development costs, since multiple parties can benefit from increase in functionality [12]. Service reusability enables a service to be composed in multiple other composed services [6].

**Composable.** Services are designed in such a way so that they can be called by other services. This enables them to be composed into to composite services [6, 13]. The composition of services from a business perspective leads to business process modeling [13].

**Stateless.** A stateless service is a service that maintains no state over multiple service calls [13]. Stateful services increase dependencies [9], cannot be replaced when maintaining a state and load balancing becomes much more difficult. Therefore a service should be stateless. However, requester applications require sometimes persistent state between state invocations. The state should not be stored on the service level, but at the underlying backend system or front end system instead [13]. Statelessness increases usability and composability of a service [6].

**Discoverable.** To call a service, its location should be known. This is why a service should be discoverable. Services are usually made discoverable via an enterprise service bus. Discoverability provides a medium for promoting reusability of a service [6]. Services can also be discovered by the word of mouth [13].

2.2 Enterprise service bus
The Enterprise Service Bus (ESB) functions as the technical backbone of a SOA landscape. Companies that implement large application that are critical to their success usually make use of an enterprise service bus [4]. For a service to be able to call another service [7], a service needs the metadata (description) of another service. This metadata can be made accessible through the ESB that provides a registry of service that are available for invocation by a requester service. Thus the ESB provides means for connecting services with each other.

The main goal of an ESB is to provide interoperability. It allows the connection between services of heterogeneous systems because the only thing that needs to be defined is the way in which services communicate. A fundamental part of the job is data transformation [13].

It does not matter on what platforms or on what hardware different services run, as long as the communication between services and the ESB is according to an enterprise wide standard. This may be by a platform specific API when using for example specific JAVA interfaces or protocol driven as SOAP is used in web services [13]. If necessary, the ESB may support even different communications standards and map specific communications to other formats [7].

The mediating role of the ESB may consist of providing other functionality as well [13]:

**Intelligent routing.** A request and subsequently a reply should be delivered to the right service. This routing can be based on things like priorities of different services, load balancing issues and context sensitive routing settings.

**Dealing with security.** This includes issues such as authentication and authorization of identities like foreign service requesters and also integrity and confidentiality of data.

**Dealing with reliability.** Not all protocols ensure the delivery of exactly 1 instance of a messages send out by a service. The ESB should - depending on the protocol used - modify this behavior or may even send requests to other services when there is no response in a specified time.

**Service management.** The ESB contains a repository that manages services and their artifacts. These are interfaces, contracts, Service Level Agreements, dependencies etcetera to help with identifying, designing and developing new services. From a technical perspective these details in the registries are used to use a service at runtime.

**Monitoring and logging.** is used for monitoring services and logging issues related to services performance or number of requests issued and handled. Because the ESB processes all services requests, monitoring and logging can also be used to get an overview of all business activities that makes use of the ESB. This is called Business activity modeling.

2.3 Service classification
Services can have very different purposes and consequently very different roles. Therefore it is helpful to make a distinction between different types of services. Josuttis [13] distinguishes 3 different types of services: basic, composed and process services. These services operate in their own layer in order to serve the end user in the enterprise layer (see Figure 2).
Basic services, also called Atomic services [14], are services that wrap a backend or problem domain so that other services can access data to be used in the SOA infrastructure. Basic services can be further divided into basic data services and basic logic services. Basic data services either read and return or write pieces of data who based on business needs belong together. These services have only access to one backend system or problem domain and ideally have the ACID properties so that they form reliable small building blocks to be used by other services. Examples of these services are: return customer’s address and create a new invoice. Basic logic services represent business rules to be used by other services. Examples are: define product catalog or define restricted range of dates. Functionality of this kind may already be offered by existing backend systems or legacy systems. In that case the service is only a wrapping on the existing functionality so that it can be used in the SOA infrastructure.

Composed services are services that exist out of basic services or other composed services. Composed services can modify data on different backend systems because they are composed of different basic services. They update data that is stored on different backends in order to keep all data over the whole organization consistent. Composed service may also consist out of basic services that all point to the same backend. This is useful to create service that can be requested with different attributes. The layer in which composed services are is called the orchestration layer because the composition of services is called orchestration. The orchestration layer is the point where the link is made between application logic (basic services) and business logic (process services).

Services that constitute out of combined services are called process services. From a business perspective, business processes in a SOA environment are in each of their activities (partly) supported by services. Processes are controlled by process engines based on process business process languages such as BPEL that sequentially invoke the required process services. This combining of services with workflows is called workflow integration [5]. Because processes exist over time, process services should in some way maintain a state. This is done in the frontend that provides interaction with the process service or in the backed on which a process service operates.

These processes controlled by process engines are in turn accessed by end users through the frontend, in the form of some Graphical User Interface. This is called desktop integration [5] and occurs in the enterprise layer.

2.4 SOA implementation process

A SOA implementation is a process in which the organizational and information technical situation will change over time. Different phases in the implementation require different points of attention. Therefore various authors have set up methodologies that form a guideline for SOA implementations.

The methodologies of Gartner [4], Josuttis [13], Erl [6], Papazoglou [10] and Lee, S. P., Chan, L. P. & Lee, E.W [2] were analyzed to get a general overview of the phases of an implementation process.

Combining these 5 models results in an incremental, cyclic approach in which is started with the planning phase. After the planning phase the methodology proceeds with cyclic behavior to reflect the succession of additional SOA projects in order to gradually adopt SOA throughout the whole organization.
the services are designed that provide support for the execution of the separate tasks within the process [6].

When all the services are specified, the services are composed in process models by defining them in a business process language such as BPEL. Also the business roles are defined in this step to be used by a business process language.

2.4.4 Implementation phase
In the implementation phase the services are constructed according to the specifications created in the design phase [6, 10, 2].

2.4.5 Testing phase
Because of the generic nature of services and its likeliness of reuse, services should undergo extensive testing [6]. Testing is characterized by ascertaining that services are according to standards defined during analysis, design and implementation phase. According to Papazoglou [2], services can be tested in 5 ways: dynamic testing, functional testing, performance testing, interface testing and assembly testing.

2.4.6 Deployment phase
In the service phase, the services and its processes are rolled out to all participants, including other enterprises and applications as well. Papazoglou [16] describes three activities that take place in this phase: (1) Publishing the service interface in a service registry, (2) deploying the web services and integrating them with processes so that they are able to run and (3) publishing service implementation details about where the service can be invoked.

3. BENEFITS OF SOA
Newcomer & Lomow [17] and Erl [6] elaborate in a detailed way about the benefits that SOA brings. Newcomer & Lomow make a distinction between technical benefits and business benefits. Except for increased customer satisfaction all business benefits are covered by the classification of Erl. The technical benefits of Newcomer & Lomow can be related to the “reduced IT burden” benefit from Erl.

Erl makes a distinction between strategic benefits and strategic goals.

3.1 Strategic goals
According to Erl [5], organizations will achieve three strategic benefits through the attainment of four strategic goals of a SOA:

3.1.1 Increased intrinsic interoperability.
Software programs that are interoperable are able to share data with each other. Programs that are integrated through SOA are interoperable.

Dacl et.al. [18] define 3 metrics for interoperability:
1. Conceptual compatibility: The extent to which the information exchanged between components has the same syntax and the same semantics
2. Technological compatibility: The extent to which the IT platform technology and the software languages are compatible.
3. Organizational compatibility: Are authorities / responsibilities clearly defined at the two sides and are the organization structures compatible?

3.1.2 Increased federation.
This refers to an IT environment “where resources and applications will be united while they maintain their individual self autonomy and self governance” [5]. Federation will lead to an enterprise-wide situation were all logic becomes harmonized.
The amount of federation can be measured by the number of SOA domains that have sole autonomy over their own services with the help of tools like routing service, service registry Enterprise Service Bus [19].

3.1.3 Increased Vendor diversification options
By designing all endpoints in a SOA with standardized service contracts, organizations find themselves in the opportunity to choose how the abstraction behind a service contract is realized. Organizations constantly have the choice between different providers, thus preventing a vendor lock-in.

This benefit is operationalized as the extent to which there is free choice of vendors when deciding on the acquisition of new software.

3.1.4 Increased business and technology alignment
Alignment is often referred to as to what extend Information Technology is applied in an appropriate and timely way in harmony with business strategy, goals and needs [20].

3.2 Strategic benefits
Attainment of the four strategic goals will result in the following three concrete benefits:

3.2.1 Increased return on investment.
Traditional silo based applications are extended and modified over time to meet organizational demands. This increases the complexity of these applications which will result in increased total costs of ownership. Services can be reused multiple times and do not increase as much in complexity when extended or modified as is the case with silo-based applications. Thus services can be used for more purposes, yielding a bigger return on investment.

3.2.2 Increased organizational agility
This is expressed as the efficiency as to which an organization can respond to change. When SOA is applied throughout the whole organization, services becomes highly standardized and reusable. As more processes are supported by services, new processes can be made up of services that already exist to support other processes. This reduces the time to support new processes, increasing organization agility.

Agility is defined as the time it takes to change or create new processes.

3.2.3 Reduced IT burden
Applying SOA throughout the whole organization will enable uniform control over the IT, reducing waste, redundancy, size and operational costs. This will reduce the IT burden. This reduction can be related to the technical benefits of Newcomer & Lomow [17] who define it in terms of efficient development, more reuse, simplified maintenance, incremental adoption and graceful evolution. Incremental adoption refers to the adoption process of SOA: SOA does not have to be adopted rigorously; it can be adopted process by process. Graceful evolution refers to the evolution of a service. Because the implementation of a service is hidden from the outside world, the inside of a service can gradually improve without negatively affecting the service it delivers to its requestors.

Reduced IT burden is defined as [17]: efficient development, more reused, simplified maintenance, graceful evolution of a service.

4. CRITICAL SUCCESS FACTORS
Implementing SOA is an IT project. Therefore the general IT project management factors like involve all stakeholders, make a good initial design to have strong foundations, adopt incremental approach, have top management support and engage in staff training also hold for SOA.

Various authors have found different SOA-specific factors that are important for a SOA implementation to be successful. While some of these factors show some overlap, other factors did not. Lawler et.al. [22] tried to make exhaustive lists of important factors. Much of these SOA-specific factors can be traced back to three central themes: reusability of the services, complexity of the SOA and governing of the SOA adoption process [19, 21, 22, 23, 24, 25]. Reusability is chosen because it is often regarded as essential to service (and thus a SOA) design [11, 26, 27, 28]. Complexity is chosen because it is frequently cited as a reason for SOA projects to fail [29, 30]. Governance is chosen because it is often regarded essential for the implementation of a SOA [31]. Other themes can be related to factors that are critical for the adoption of a SOA as well. Examples are: train employers to let them understand SOA, create linkages between IT and the business, and security issues [21]. But these also apply to other IT projects and are therefore not included in this research.

Therefore the following CSFs are defined: Focus on reusability, focus on reducing complexity and engage in SOA governance.

According to the theory of Rockart [8], CSFs are defined as the areas in which good performance is necessary to ensure attainment of organizational goals.

When the CSFs are related to a SOA implementation, the following is assumed: when good performance is attained in the areas defined by the CSFs, a SOA implementation will be successful.

4.1 Focus on reusability
The reusability characteristic of a service primarily contributes to the benefits of increased agility and increased return on investment [11]. A reusable service can be composed in other services to quickly create new processes, increasing agility. Reusing a service saves development costs, contributing to return on investment. But for reuse to payoff, the service must be designed so that it can be reused by a large variety of different systems. Many companies fail to make a sufficient part of their services reusable resulting in SOA implementations that do not meet their expected benefits [25].

According to Gartner [19], reuse of a service is difficult because of the following challenges: (1) When a service is going to be reused by another process, it rarely can be reused without changing the interface. (2) The underlying data source of a service are most of the time not general-enough so that it can be reused by other processes. (3) Requirements regarding security, integrity and performance characteristics vary too much amongst different requesters. (4) Different services are usually developed and owned by different departments. Using a service from a different department requires trust in the quality of the service [26] and the ability to change the service to meet requirements of reusing. The owning department or development team may form an obstacle in easily changing the services.

To cope with these challenges, [27] states that a service should be made flexible through the following three characteristics: genericity, configurability and extensibility. Genericity makes a service usable across different business contexts by minimizing the hard-wiring of context specific functionality [28]. Being
configurable makes a service adaptable to different business contexts, and extensibility makes a service able to extend a service with new functionality.

Dan et.al. suggest 4 practices to make a service flexible and thus reusable. (1) There should be the governing of an enterprise wide glossary that encourages everyone to use consistent business terms. Specific persons or teams should be held responsible for adding, changing or removing terms and there should be a defined methodology for managing the terms and using the terms when creating services. (2) There should be a governing body (usually called SOA Competency Center) that specifies and supports the roles in the solution lifecycle and the different processes. It is also responsible for guiding the organization through the SOA adoption and helps development teams to use the same design principles. (3) Service quality should be made reliable by defining a SLA about how often a service can be used, what the allowed down time is, the response time and other quality related metrics. There should also be a visible quality process for the development and testing of a service in order to be able to trust the reusability of a service [26]; this may even go as far as using service certification. (4) A service that already is in use is critical for process execution. Also, because a service is rarely immediately suited for reuse and of the ownership problem, there should be a well defined methodology for improving a service and gathering user requirements.

In addition, Gartner [19] advises to reward the reuse of software designed by others and fostering an environment in which reuse is considered a characteristics of excellent software engineering preferable to custom programming. In line with this, IBM advises prioritizing the development of reusable service over the development of individual services [21].

The focus on reusability is measured by the extent to which a company follows the following sub-factors:

1. Governing of an enterprise wide glossary.
2. The governing of SOA by a SOA Center of excellence.
3. Service quality is being defined in a SLA.
4. A well defined methodology for improving a service and gathering user requirements (e.g. as defined in section 2.4).
5. Rewarding of reusing software designed by others.
6. Prioritizing the development of reusable services over the development of individual services.

4.2 Focus on reducing complexity

Implementing SOA in a wrong way will lead to unnecessary complexity, with which lots of company struggle [29, 30]. Traditional applications are made up of fitting pieces of functionality assembled by its manufacturer who makes sure it runs well (stable, scalable). With SOA, all kinds of mash-ups and composite applications are reassembled where services from all over the organization are being combined to make new sorts of applications [33]. But for the mash-up or composite application, there is no manufacturer but only the IT department. The responsibility of a well-running system shifts from the external application provider to the internal IT department.

To cope with this increased responsibility, Woods [33] states that all the services must be monitored in terms of load, response time, capacity of the service and peak hours of the services. Also the accompanying processes must be monitored.

And when services are becoming popular, load balancing or virtualization of services should occur.

Gartner [19] advises to use a middleware based intermediary consisting of tools like application servers, business process management tools and ESB when the number of services deployed increases to more than 20-30. When SOA is deployed throughout a big organization consisting of semiautonomous business units and subsidiaries, separate ESBS can be established that are linked to each other to arrive at federated SOA which may prevent political, organizational and technical hurdles.

Also for the sake of reducing complexity, IBM [21] advises to keep application-specific business logic and processing out of the ESB. Next to this, services should be extensively tested on performance, scalability and stability.

To cope with the responsibility shift, companies should make as much use of specification standards and tools where appropriate so companies can focus their attention on unique issues that are specific to their own situation.

The focus on reducing complexity is measured by the extent to which a company follows the following sub-factors:

1. Because more than 100 SOA “standards” exist, Forrester [32] advises to focus on the usage of the following standards: SOAP 1.1, WSDL 1.1, WS-I Basic Profile 1.0 or 1.1, UDDI 3.0.2, WS-Security 1.0 or 1.1, WS-BPEL 2.0, BPMN, WSRP 1.0, XML Schema 1.0, XSLT 1.0, XPath 1.0, XQuery 1.0, XML Signature and XML Encryption
2. Monitoring of services in terms of load, response time, capacity of the service and peak hours of the service.
3. The use of an ESB as defined in section 2.2.
4. Use of federated SOA approach with multiple SOA domains in big organizations.
5. Keeping application specific business logic out of the ESB.

4.3 SOA governance

Because SOA is adopted incrementally throughout the organization, SOA has to grow, step-by-step [13]. This process has to be overseen.

According to the IT Governance institute, IT governance exists to direct IT endeavors to ensure that IT performance meets business objectives in the following ways [34]: IT alignment with the business resulting in realizing promised benefits, exploiting opportunities and maximizing benefits, responsible use of IT resources and management of IT-Related risks. In this context SOA governance can be seen as an evolution of IT governance introducing a tighter business involvement in supporting IT service components [34].

SOA governance focuses on that everyone is working together and that separate efforts are not working against each other; SOA governance is more about managing organizational dependencies. Governance is not about making decisions but about guiding who is making decisions and making sure they are coherent with each other across the organization [34].

The governing of SOA can be divided in non-technical and technical tasks [13]. The extent to which an organization engages in SOA governance is measured by to what extent they execute tasks described in section 4.3.1 and section 4.3.2.
4.3.1 Non-technical tasks
Maintaining a reference architecture. A reference architecture is an architectural blueprint that provides a high-level abstraction of a SOA. It specifies what technologies are being used and separate layers, architectural building blocks and the relation between them. Furthermore it includes what services interaction patterns are used and specifications of the message standards that are used [35].

Managing roles and responsibilities. Defining who is owner and responsible for a service (including agreements like SLA) is important in order to know where to go with service issues like improvements or performance problems [31]. The same goes for defining who is responsible for making high level solution designs that identify new services and defining where the responsibility of the ESB ends [13].

Choices on architecture and technologies used lead to policies and requirements. These should be defined using standards and suitable formats.

Processes and lifecycles should be defined and listed for services in order to have common terms about the state of a service. IBM [31] recommends the following stages: planned, testing, active, deprecated and sunsetted.

4.3.2 Technical tasks
Maintaining a documentation for the sake of transparency. All nontechnical issues of governance (like processes, responsibilities, policies) should be documented.

Managing the service and their service contracts. Service providers must make their services available in order for them to be used. Also service consumers must locate the services they need [31]. Maintaining an up to date listing of the services and their contracts is usually done in combination with tools like repositories and registries [13].

Monitoring. The goal of monitoring is to ensure that services live up to their SLA, rules and contracts. Monitoring also helps to notice when a service stops working.

Change and configuration management. Changing a service to reuse it might cause trouble to another service that is already using it. Version management is used to maintain different version of a service that is used by different other services. Some services are bound to stop working someday (sunsetted), while other services still depend on it. The solution to this is migrating a service or letting the requesting service get the needed functionality elsewhere [31].

5. CASE STUDY
A case study was held to test whether the CSFs described in section 4 could be found in practice. Case selection is important for providing useful information for the research proposed. For a company to provide insightful information during a SOA case study, Hau et al. [36] suggested it should have the following characteristics: complexity, risk of changing requirements, multiple teams, changing environment, domain boundary, different system involved, scarce expert knowledge, shared services, multi channel scenarios. A Dutch insurance company — which had these characteristics — was used for the case study.

The case study was organized as follows: the CSFs and the benefits were operationalized and consequently measured by means of a questionnaire. Various interview sessions with employees of the company followed to allow a detailed analysis of the CSFs and possible other CSFs.

5.1 Questionnaire method
10 employees (lead developers, functional architects and technical architects) were asked to fill in a questionnaire. The questionnaire consisted out of two parts.

The first part asked the employees about how important they considered the critical success factors mentioned in section 4 for attaining SOA success. They could give the critical success factors a rank ranging from 1 to 4. 1 stands for not important while 4 stands for very important. An employee giving a CSF a rating of 4 would mean that the employee considers the specific factor as a CSF. 3 or lower would indicate some importance, but important enough to regard a factor as a CSF. Next they were asked to rank on the same scale on what they thought was needed to attain good performance in the areas defined by the CSFs. This was done for all CSFs. For example: for focus on reusability, the employees were asked about how they thought the sub-factors mentioned at the end of section 4.1 would contribute to attaining focus on reusability. These questions are outlined in Appendix A. The sub-factor “The governing of SOA by a SOA Center of excellence” was not included in the questionnaire because it strongly resembles the third CSF: SOA governance.

The second part of the questionnaire asked the respondents what type of benefits were gained from attaining good performance with the CSFs. For each CSF, respondents had to indicate how much it contributed to each of the benefits of SOA as described in section 3. They could rank the specific benefits with a range of 1 to 4. 1 stands for no contribution from a specific CSF to a specific SOA benefit while 4 stands for a high contribution from a specific CSF to a specific SOA benefit. These questions were asked for all the three CSFs. The questions that were about the first CSF are outlined in Appendix B.

5.2 Questionnaire results
Out of the 10 employees that were approached for filling out the questionnaire, 9 responded. The results are displayed in the first and second column of table 1. The numbers in the second column display the average importance indicated by the respondents for the sub-factors and CSFs in the first column.

Focus on reusability was given a 3.1 on average, meaning that it is not regarded critical for the success of a SOA. Remarks from respondents include that technical services should be reusable if possible, but business services should not necessarily be reusable and should only serve the needs of the domain. Things that complicated reusable services are: (1) Predicting the chance that services might be reused in future is difficult and (2) investing in reuse requires a kind of altruism for future users which is sparse because there are tight budgets for the design of services.

Focus on reducing complexity was given a 3.6 on average, meaning that focusing on reducing complexity is regarded as critical for the success of a SOA. Respondents stated that complexity increases when services of other domains are embedded in a service. Having the right granularity of a service is very important in this aspect. Complexity within a domain is no problem as long as services are not complex for requestors from another domain. Another comment was that reducing complexity might look trivial, but this responsibility lies in the hand of a few architects.

SOA governance scored on average a 3.8 thus it can also be regarded as critical for the success of a SOA. A frequent comment was that there should be clear roadmap on where to go with SOA, as different projects tend to focus on the short term contributing little to the overall SOA.
For attaining good performance with the CSFs, most sub-factors mentioned at the end of the sections 4.1 until 4.3 were considered important at a range of 2.8-3.3 on a scale on 1-4. Regarding good performance with focus on reusability, an exception includes the sub-factor to have a “rewarding of reusing software designed by others”. This sub-factor was on average rated with 2.0. Next to this exception, the following were found to be of very high importance (score ≥ 3.7) to attain success in the area of a CSF:

- Maintaining a business vocabulary for focusing on reusability.
- Using software standards, having a federated SOA approach and keeping application specific business logic out of the ESB for focusing on reducing complexity.
- Maintaining a reference architecture, managing the service and their service contracts and engaging in change and configuration management to attain good performance in the CSF of SOA governance.

The results of the relative importance of the CSFs to the SOA goals are displayed in table 2. Looking at the results, it can be said that all CSFs roughly equally contribute to the different SOA goals. Exceptions include SOA governance that highly contributes to intrinsic interoperability. Focusing on reusability does contribute relatively little to federation and a reduced IT burden. The SOA focus on reducing complexity does contribute little to federation.

5.3 Interviews method

The questionnaire checked the CSFs from a closed perspective: respondents were only allowed to think in terms of the presented CSFs. Three open interviews were held with architects and lead developers to check if the CSFs were also found important when they were not explicitly referred to. At the beginning of the interview the respondents were asked to mention points they thought should not be forgotten during a SOA project. It was said that the points should be related to SOA and a bit more specific than points that are important for general IT projects. No further directions were given in the interview. In total, 5 times concluding statements were given to an employee that referred to points that contribute to the CSFs. The interviews lasted between 20 and 30 minutes. During the interviews, notes were written down and afterwards converted to transcripts. Within the transcripts, topics were identified that related to CSFs, sub-factors of the CSFs or other points of attention for a SOA project.

5.4 Interviews results

The results of the interviews are displayed the first and third column of table 1. The number in the third column refers to how many times the sub-factor or CSF in the first column was referred to by the employees during the three interviews.

One employee indicated the relative importance of focusing on reusability: “The extent of the reusability of services should be high. But when dealing with domains that serve a specific purpose, services should not necessarily be designed to be reusable as it unlikely that they are reused a lot. Designing for reuse requires much extra effort.”

Next to these points, it was mentioned one time that a significant amount of legacy applications were still running, costing much money to the organization. This prevented SOA from increasing the return on investment because SOA applications and legacy applications were running at the same time, requiring double maintenance and monitoring. To this end, as an employer indicated, more money should be made available to wrap up legacy application and rely on only a SOA. This wrapping up involves risks, stability and infrastructural issues. Furthermore, employees spoke of a required level of familiarity with the technology and SOA-concept before a SOA can be a success. The other thing mentioned was a warning that an organization should only start when it has strong foundations and a vision on where to go with SOA (including a vision on the migration of functionality from legacy applications to the SOA). This can be related to SOA governance.

Table 1. Questionnaire and interview results

<table>
<thead>
<tr>
<th>Focus on reusability</th>
<th>Questionnaire</th>
<th>Open interview</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use of glossary</td>
<td>3.7</td>
<td>1</td>
</tr>
<tr>
<td>Use SLA</td>
<td>3.2</td>
<td></td>
</tr>
<tr>
<td>Use of defined methodology</td>
<td>2.8</td>
<td>1</td>
</tr>
<tr>
<td>Rewarding of reuse</td>
<td>2.0</td>
<td></td>
</tr>
<tr>
<td>Prioritizing reuse</td>
<td>3.1</td>
<td></td>
</tr>
</tbody>
</table>

5.3 Focus on reducing complexity | Questionnaire | Open interview |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Use of standards</td>
<td>3.8</td>
<td>2</td>
</tr>
<tr>
<td>Monitoring services</td>
<td>2.9</td>
<td>1</td>
</tr>
<tr>
<td>Use of ESB</td>
<td>3.1</td>
<td>1</td>
</tr>
<tr>
<td>Federated approach</td>
<td>3.7</td>
<td></td>
</tr>
<tr>
<td>No application logic in ESB</td>
<td>3.9</td>
<td>1</td>
</tr>
</tbody>
</table>

SOA governance | Questionnaire | Open interview |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Maintaining a reference architecture</td>
<td>3.8</td>
<td>2</td>
</tr>
<tr>
<td>Managing roles &amp; responsibilities</td>
<td>3.3</td>
<td>1</td>
</tr>
<tr>
<td>Listing of processes and lifecycles</td>
<td>3.0</td>
<td></td>
</tr>
<tr>
<td>Maintaining documentation</td>
<td>3.2</td>
<td>2</td>
</tr>
<tr>
<td>Managing services &amp; contracts</td>
<td>3.7</td>
<td>2</td>
</tr>
<tr>
<td>Monitoring</td>
<td>2.9</td>
<td>1</td>
</tr>
<tr>
<td>Change &amp; configuration management</td>
<td>3.7</td>
<td></td>
</tr>
</tbody>
</table>

6. CONCLUSION

The most striking conclusion from this case study is that reusability is not as critical to SOA as it is stated in some literature [6, 11, 28]. Both the questionnaire and the interviews indicated reusability as being important instead of being very important. Therefore focusing on reusability is not regarded as
being critical for the success of adopting a SOA. Considering the fact that reusability is promoted as a fundamental advantage of SOA [6, 17], realized SOA benefits may be lower than promised in the literature. Also the fact that reusability is being considered a very important part of the SOA paradigm [6, 11, 28] may need some rethinking. However, this does not mean that reusability is irrelevant. It still important to take account of reusability issues when designing services, especially with regard to technical services (see basic data services, figure 3).

Focusing on reducing complexity and the governing of the SOA adoption process are confirmed to be critical to the success of a SOA adoption. The suggested sub factors that contribute to success within the two CSFs are also found relevant based on the questionnaire outcomes. The fact that during the open interviews the employees referred to most of the sub factors and the two related CSFs indicates that the sub factors and CSFs hold some empirical validity. Also most of the things mentioned in the interviews except for wrapping up legacy were included in the research model. This strengthens the assumption that the 2 CSFs and its related sub factors really matter for a SOA adoption. While not all sub factors were referred to during the interviews, the small number of interviews does not have enough statistical power to rule out or include other sub factors. But the results point in the direction of validation of the two CSFs.

6.1 Limitations
Due to its small research sample, the results provide insufficient evidence for statistically grounded conclusions and generalization of the conclusions to SOA projects in general. Furthermore, the three posed CSFs provide a unique perspective on SOA implementations, dividing points (sub factors) of attention in three different CSFs. Other papers can easily give another segmentation of the CSFs and sub factors that are important for a SOA implementation.

6.2 Future research
More empirical evidence can be collected to increase the solidness of the model presented in this paper. This applies in particular to the conclusions about reusability. Also different sub factors might need more elaboration in order to give organizations more insight in how to successfully implement a SOA.

REFERENCES
APPENDIX A
Table 3. Questions of the first part of the questionnaire that were about focus on reusability

<table>
<thead>
<tr>
<th>Question</th>
<th>Not important</th>
<th>Very important</th>
<th>No opinion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. A reusable service can be combined with other services to in order to create composed services. To gain benefits from this, a service should be designed for reuse.</td>
<td>O O O O O O</td>
<td></td>
<td>O</td>
</tr>
<tr>
<td>According to your experience, how important is it to focus on the reusability of a service during the implementation of a Service Oriented Architecture?</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Comments:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. According to your experience, which factors contribute to focusing on the reusability of a service?</td>
<td>← Not important</td>
<td>Very important →</td>
<td>No opinion</td>
</tr>
<tr>
<td>Governing of an enterprise wide glossary.</td>
<td>O O O O O O</td>
<td></td>
<td>O</td>
</tr>
<tr>
<td>Defining service quality in a SLA.</td>
<td>O O O O O O</td>
<td></td>
<td>O</td>
</tr>
<tr>
<td>A well defined methodology for improving a service and gathering user requirements.</td>
<td>O O O O O O</td>
<td></td>
<td>O</td>
</tr>
<tr>
<td>Rewarding of reusing software designed by others.</td>
<td>O O O O O O</td>
<td></td>
<td>O</td>
</tr>
<tr>
<td>Prioritizing the development of reusable services over the development of individual services.</td>
<td>O O O O O O</td>
<td></td>
<td>O</td>
</tr>
</tbody>
</table>

APPENDIX B
Table 4. Questions of the second part of the questionnaire that were about focus on reusability

<table>
<thead>
<tr>
<th>Question</th>
<th>No positive influence</th>
<th>Much positive influence</th>
<th>No opinion</th>
</tr>
</thead>
<tbody>
<tr>
<td>11. According to your experience, to what extent does focusing on reuse contribute to the following SOA benefits:</td>
<td>←No positive influence</td>
<td>Much positive influence →</td>
<td>No opinion</td>
</tr>
<tr>
<td>Increased intrinsic interoperability.</td>
<td>O O O O O O</td>
<td></td>
<td>O</td>
</tr>
<tr>
<td>Increased federation.</td>
<td>O O O O O O</td>
<td></td>
<td>O</td>
</tr>
<tr>
<td>Increased vendor diversification options.</td>
<td>O O O O O O</td>
<td></td>
<td>O</td>
</tr>
<tr>
<td>Increased Business and IT alignment.</td>
<td>O O O O O O</td>
<td></td>
<td>O</td>
</tr>
<tr>
<td>Increased return on investment.</td>
<td>O O O O O O</td>
<td></td>
<td>O</td>
</tr>
<tr>
<td>Increased organizational agility.</td>
<td>O O O O O O</td>
<td></td>
<td>O</td>
</tr>
<tr>
<td>Reduced IT burden.</td>
<td>O O O O O O</td>
<td></td>
<td>O</td>
</tr>
</tbody>
</table>