# Where to Start with SOA

## Criteria for Selecting SOA Projects

Thorsten Hau, Nico Ebert, Axel Hochstein and Walter Brenner
Institute of Information Management
Iniversity of St. Gallen, Switzerland
Email: thorsten.hau@unisg.ch

***Abstract*—The concept of service oriented architectures has received considerable attention lately. Despite the hype our experience from five major German chemicals companies shows that firms do not blindly adopt the concept but want to see its value for their organizations. This paper identifies criteria for projects that should serve especially well as first proof of concept SOA implementations. Therefore the paper compares project goals to SOA benefits and requirements and deduces decision supporting criteria. Complex projects with many involved stakeholders and high risk of changing requirements, to name a few of the criteria, are more likely to profit from SOA than small simple projects with few involved people. The criteria are then applied in five cases to evaluate projects the authors analyzed. The application shows that the criteria give a good indication for or against SOA adoption in a project.**

## I. Introduction

Service Oriented Architecture (SOA) is more than a buzzword used by marketing departments to sell IT products. Further down in chapter II-B we will show that the relevant aspects of SOA are those that enable a firm to modularize its IT application landscape. While this modularization makes managing an IT landscape less hazardous, the process of disentangling today's tightly coupled applications is a tedious one.

Assuming that SOA will enter the average organization through small and rather isolated projects, there is an obvious conflict of interest between project management that needs to achieve short term goals and the extra investment needed to adhere to SOA design principles. In this work we will derive a set of criteria that helps to find a starting point for the adoption of SOA ideas within a company. In order to achieve this we look for projects within a company for which SOA adoption yields immediate benefits. Our goal in this is to ease the tension between long term global benefits of SOA and short term local needs of project management.

A basic assumption made in this work is that there are two possible scenarios for the adoption of SOA ideas. In a top down scenario, top management would realize the potential of SOA and conduct a high level project to establish SOA paradigms within the organization. This scenario is not unrealistic and there are several examples like Deutsche Post (the German Post, a company that boasts its SOA competence) where SOA was introduced this way [1]. However, the bottom up approach appears to be the more relevant scenario because often SOA might not be considered to be of strategic importance. The examples of top level SOA projects mostly represent instances where the inability to modify the IT landscape had become a pressing problem which in turn made SOA the remedy to an acute problem top management was facing. Supposing that in most firms there are no such acute pain points and SOA in turn is not a top priority, one must consider a gradual scenario in which SOA enters an organization through many small projects with little resources to spare. In these projects SOA would typically be a second order goal. For example introducing a new logistics tool could have one requirement that reads: make functionality available via a specified interface. After several such projects one would have two parallel worlds: an old one with close coupling, dependencies and uncontrollable complexity and a new (SOA-) world that can easily be controlled and adapted. The new world would gradually expand while the old part would become less important. As promoted by [2] we consider this soft introduction of SOA the most feasible approach and this work develops a set of criteria which can be used to evaluate whether a project is suitable for a proof of concept SOA implementation.

## II. Theoretical Background

### A. Project Management

The "iron" goals of project management are time, scope and money. These criteria form the magic triangle within which the project management has to find an optimal compromise [3], [4]. These criteria have been modified, extended and rearranged. For example, [5] emphasized that the customers point of view must be integrated into a project success metric. They also argue, that success can not be easily defined and that it depends on the perspective of the judging person, whether a project can be considered successful or not. Interestingly the authors also name "preparing for the future" as a relevant aspect. However, despite all these efforts to provide different perspectives on project management, some sort of general agreement among practitioners seems to exist that the three goals named above are still the most important ones. One variation which this article will use is the split-up of scope into quantity (which we will call scope) and quality. With this differentiation we get the following four dimensions to judge a project's success:

**Time:** Can the project be finished on time?

1530-1605/08 $25.00 © 2008 IEEE

**Scope:** Can the project achieve the goals that were set? This is a quantitative measure that doesn't take into account the quality of a solution.
**Money:** Does the project stay within budget?
**Quality:** Is the delivered output of acceptable or the defined quality?

Project management will commonly be evaluated along these four criteria. Since the evaluation of a project's success usually happens upon its conclusion, we will assume the average project manager to optimize his short term success. It follows that everything that makes a project more expensive or longer will be judged as having a negative impact on his chances for success and thus be avoided. Tools, on the other hand, that reduce the spendings for a certain task or improve the quality that can be achieved, will be used if their potential can be made plausible.

### *B. SOA*

Service oriented architecture (SOA) is a very ill defined buzzword in the IT community. This is largely due to the fact that every player in the market for IT uses its own definition of SOA. SAP for example sells its ERP as SOA, IBM sells databases and middleware as SOA and BEA sells a source code repository as SOA. To clarify our understanding of SOA we first explain its two building blocks:
**Architecture** is a term that can not be properly defined without putting a lot of contextual meaning into the definition. In the realm of large distributed software systems like the application landscapes of large companies the term architecture can have two meanings. The first means the model according to which different elements relate to each other [6]. The second meaning is the concrete instantiation of such a model and thus is synonymous to 'all applications, hardware and their relations to each other'. In the following we will use both meanings since the context will clarify which one is intended.
**Services** are the envisioned building blocks of an SOA (model and instantiation). A service encapsulates a certain functionality like providing customer data and makes it available through a well defined interface. This point of view switches the focus away from applications towards functions that need to be performed. Consequently a business process as a series of tasks to be performed can be composed of different services performing these tasks [7].

On the level of principles, SOA can be considered as the transfer of object oriented programming (OO) paradigms, i.e. objects hide information and encapsulate functions, to the world of enterprise systems. Translating the principles of OO-programming to SOA speech one gets a definition of SOA like the one provided by [2]: Service Oriented Architecture (SOA) combines component orientation with loose coupling and external process control into a method to design system landscapes. The objects in OO programming are the components of a system landscape, loose coupling is achieved through encapsulation, information hiding and external process control. Since external process control is rather a means to achieve loose coupling than and end in itself, we can modify the above definition and say that the basic principles underlying SOA are:

- Component Orientation and
- Loose Coupling.

These two paradigms are supposed to lead the way to more manageable system landscapes. However, in order to put these two abstract requirements into practice one must go one step further and define a set of concrete measures to achieve these two goals. Focusing on large software landscapes one can identify the following rules necessary to construct SOA compatible architectures [1]:
**Interface Orientation** corresponds to the OO paradigm of encapsulation. How a functionality is executed doesn't matter to the user of the functionality. In order to put this principle into practice it is necessary to have well defined service specifications that describe the service's interfaces, capabilities and availability.
**Standardization** between the components of an SOA is necessary to be able to integrate different elements into a new functionality. Standardized ways of interaction make sure that one software component can exchange data with another over the same interface. In analogy to electricity this means that all applications use the same plug and socket. Today it is common that one firm uses US-style plugs and European sockets within one domain.
**Autonomy and Modularity** are necessary to achieve component orientation. Autonomy in this context means that a Service is an entity that can execute a certain task without relying on other services. There should be no interdependencies between services and furthermore the communication between the services should be loosely coupled (i.e. message based and stateless) [7].
**Business Orientation** is the translation of object orientation to the world of enterprise systems. Object orientation tried to draw the link between the real world and the programming abstraction by using objects as representations of real things. In this line of thinking, a service should represent a meaningful abstraction of a business related real object. A service providing customer data might be such a meaningful entity.

Having discussed the different aspects of SOA we will now turn to the question why all this effort should be made. Therefore we follow [8] and name the four main benefits of adopting SOA:
**More agility** will make a firm able to adapt faster to new requirements. If a process changes, for example because of new regulatory requirements, one can simply use the available services in a different order without causing major changes in the software. Agility also finds its manifestation in smaller projects.
**Less complexity** makes IT infrastructure easier to handle because each component can be considered individually. Changes in the implementation of a component don't affect the users of that component since they only know the interface. Furthermore changes should not have impact anywhere else in the software landscape since there should be no uncontrollable dependencies.

**Increased reusability** is possible when software components can be used in several contexts and identical tasks don't need to be implemented several times. The service get_customer_data for example can be used by several consumers.

**Better interoperability** is a likely benefit of SOA because components of a software landscape can more easily interact and exchange information since only interfaces and middleware need to be considered instead of whole implementations.

One aspect all these benefits have in common is that they are likely to only arise in the long term and can not be realized instantaneously because the architecture of an enterprise system landscape needs to be service oriented in order to be less complex and more agile than they are today. Becoming service oriented, however, is a process that takes time because all systems must adhere to the above mentioned SOA rules.

## III. Deriving Criteria for Good SOA Projects

### A. The Adoption Scenario

As said already the authors believe that the top down SOA introduction approach where top management commits to SOA and a major effort is made, is far from reality in most companies. The realistic scenario in our opinion is an iterative development in small steps. This means that isolated projects are conducted which follow a subset of the SOA principles. After a few projects have implemented services it might be recognized that SOA makes project management more efficient and that one should consider coordinated action to broadly enforce the adoption of SOA methods. In such a setting project management is likely to evaluate the concepts available and use those that provide a superior cost benefit ratio. If SOA is to be chosen it must be better than other approaches or yield benefits that no other approach can provide.

SOA as frequently communicated can hardly be called a light weight approach since for example governance structure and IT infrastructure are needed to exploit all potential benefits of the concept. In the following paragraph we will analyze costs and benefits of SOA, compare them to project goals and constraints and identify those aspects of SOA that yield immediate benefits. The intention behind this approach is to establish SOA as a tool useful for daily business. Once this is achieved further steps towards the SOA vision might become feasible and an investment in infrastructure becomes an option.

### B. SOA Benefits vs. Project Goals

As explained In part II-B the benefits of adopting SOA are more agility, complexity reduction, increased reusability and better interoperability. These benefits of adopting SOA practices are furthermore of mostly strategic and long term nature. In contrast, projects commonly have a duration of months to years and need to achieve very strictly defined goals within a set time frame. With this work we try to find a compromise to ease the tension between theses conflicting targets and benefits.

It might not be immediately clear why the SOA benefits are in opposition to project goals. Wouldn't agility be beneficial to achieve a timely project conclusion? The problem is that SOA benefits can only be achieved through adhering to certain design principles which possibly make a project longer and more expensive without yielding immediate benefits. In other words we have an incentive problem. Project management today must do certain things that project managers in the future will benefit from. In the following we will derive a set of criteria that help to identify projects and SOA practices that, in combination, promise superior performance. This paper is supposed to provide a set of guidelines to identify promising first step SOA projects and the SOA principles that should be applied. These projects provide immediate tangible benefits and partly implement SOA ideas.

In order to resolve the conflict of interest between project management's goals and long term SOA benefits one should first analyze which SOA benefits can be realized at least partly in the short run. Furthermore one should analyze which cost factors of SOA can be avoided in a first attempt.

**Interface orientation**, which is absolutely necessary for an SOA, is a consequence of implementation guidelines and will only cause minimal increases of programming effort. On the other hand even minor changes might punish too tightly interwoven parts of functionality. Therefore the benefits of interface orientation cost little, reduce the risk from changing requirements and simplify project management since different teams can work on the basis of defined interfaces which greatly reduces coordination effort. It follows that this design paradigm does not increase the cost or the duration of a project but has the potential to greatly reduce project duration and project risk in case of unexpected changes. The reduction of coordination effort further reduces project duration.

**Standardization** can probably be expected to be 'built in' in a project that implements a functionality from scratch. If, however, the extension of an existing function is the goal, project management should carefully analyze cost and benefits of a standardized way of communication between different modules. On the one hand standardization simplifies the task because only one standard needs to be understood but on the other hand it is possible that a lot of functions need to be wrapped in order to appear to be standardized elements. Besides wrapping elements that don't follow the set standard one could also let some sort of middleware perform the task of standard translation so that all modules can interact without having to implement different message types. Since both options cause costs because either wrappers need to be programmed or licenses need to be bought, the question of standardization has to be decided in the specific context. This is in opposition to the global perspective since standardized interfaces will certainly reduce integration effort in the future.

**Autonomy and Modularity** are necessary for an SOA because services as fundamental building blocks are supposed to provide certain functions to any requester. However, the reason for todays application landscapes' interdependencies is that often a quick and dirty close coupling solution is much faster

and cheaper than a properly executed separation of functions. Whether or not these paradigms provide benefits to a project therefore depends on the size and complexity of the project. Small projects are likely to be quicker with close coupling but large projects might profit from modularizing the tasks and thus making management easier.

**Businesss Orientation:** Correct granularity is an aspect which is of importance for the SOA idea because it is an important aspect of "business orientation". A service should be so large that it provides a meaningful business functionality. This requirement provides benefits on the small project scale because one service that provides a whole chunk of functionality is easier to handle than a multitude of small functions that need to be known on both sides of an interface. However, similar to the above said, bigger projects are likely to profit from this aspect while it adds overhead to small ones.

From what was said above one can identify projects for which SOA principles yield bigger benefits than for others. The bigger and more complex the project, the bigger the benefits from cutting it up into small chunks of functionality. The more time pressure on the project, the more important it is to start at different points and meet in the middle. It follows that the main drivers for potential SOA benefits in projects are complexity, size and time pressure because here SOA can help to cope with them. Comparing these to the project goals (time, scope, money) it appears to be feasible to see SOA as a means of achieving the project goals because it helps to control some of the main factors responsible for projects not meeting their targets.

In addition to the above mentioned inherent characteristics of promising SOA projects which we deduced from SOA paradigms, there are several characteristics of the projects that would profit from SOA ideas, which can be deduced from possible SOA benefits.

SOA **reduces complexity** by cutting things up into separate domains. Within each domain Services might not be extremely advantageous but they probably are for inter domain communication. It follows that it might be beneficial to consider projects or problems at the boundary of a domain. For example, trying to use SOA or simply services within SAP R/3 is a rather challenging task. Even though this might possibly be beneficial it is more feasible to start with a project that for example implements a functionality that gets data from R/3 to another application. At the domain boundary where two systems need to be connected it is much easier to find SOA benefits. Complexity reduction is especially important in very complex environments. Therefore it is important to find projects that work on a sufficiently complicated problem. Writing a plain html website is not very complex. Creating a website that provides information from different sources and presents it all on one screen is considerably more interesting and it might be considerably easier to provide the content with the use of services.

The most mentioned SOA benefit is **agility**. By using a service oriented approach to design ones IT landscape, a firm is supposed to be able to adapt to changes faster. Since this only plays a role in environments where changes actually do happen, one should pick an environment that is not static. Human resource operations for example are, at least at the core, very static. Managing personnel data has not changed fundamentally during the last years. Logistics operation on the other hand are constantly changing. As new possibilities for communication are developed the need for new channels of interaction arises. Furthermore, logistics have seen major changes in data availability which created the need for integrating more data sources into a more tightly woven net of data sinks. Along with the availability of data came the need to use it and implement more sophisticated instruments to control the flow of goods. These changes are still going on as innovations like RFID chips become more widely used. To sum up, it is most promising to look for SOA benefits in changing environments.

**Interoperability** is also mentioned in the context of SOA. Consequently one should look for situation where connectivity between different systems is a topic. For example, where two SAP systems need to be connected, a short script in proprietary format will almost always do the job. This way of interconnecting components will most definitely become expensive during the next release change, but since project managers are, as stated above, rather short sighted, this argument won't help. However, if two systems from different vendors need to be connected to each other chances are that middleware is required. Here it is feasible to argue for adherence to some standards from the SOA portfolio to reduce programming effort.The most important argument in favor of SOA is that one only needs people who understand the respective system. One doesn't need somebody who understands both sides. Furthermore the two sides don't need a lot of interaction because once the interface is agreed upon, they can do their job without caring for what the other side is doing. This is an important point because coordination of efforts costs a lot of time and specialists who know two systems are more expensive than those who only know one.

**Reusability** is the last SOA benefit we want to discuss here. SOA is supposed to provide services that can be used in more than one context. For services like "get_customer_master_data" this is immediately clear, but reuse is a feature that will mostly be considered a long term benefit. In the short run, however, reuse can also be interesting. For a project that has to implement different versions of similar tasks service orientation might provide substantial benefits. Consider for example a multi channel scenario. The same data has to be presented to users on different devices, or data has to be sent from different devices to on sink. Instead of programming all the different pipes to and from each device, the application could provide a single service that bundles the necessary functionality. The different devices can then access this service. This solution should be much more efficient than the "do it n times" approach.

Tables I and II summarize the above criteria for "good" SOA projects. Table I focuses on the SOA principles and lists

TABLE I
APPLYING SOA PRINCIPLES IS ESPECIALLY BENEFICIAL IF THE FOLLOWING CRITERIA ARE MET.

| SOA Concept | Helps With |
|---|---|
| Interface Orientation | Size<br>Complexity<br>Risk of Changing Requirements<br>Multiple Teams<br>Time Constraints |
| Standardization | Building from Scratch |
| Autonomy/Modularity | Size<br>Complexity<br>Risk of Changing Requirements<br>Multiple Teams<br>Time Constraints |
| Business Orientation | Size |

TABLE II
POTENTIAL SOA BENEFITS ARE MOST LIKELY TO BE REALIZED IF THESE CRITERIA ARE MET.

| SOA Benefit | Helps With |
|---|---|
| Complexity Reduction | Complexity<br>Risk of Changing Requirements<br>Multiple Teams |
| Agility | Changing Environment |
| Interoperability | Domain Boundary<br>Different Systems involved<br>Scarce Expert Knowledge |
| Reusability | Shared Services<br>Multi Channel Scenarios |

project characteristics for which they are beneficial. Table II on the other hand focuses on the benefits of SOA and lists criteria where these benefits are most likely to have a profound impact. The redundancy of the criteria is removed in the tables in the following chapter that show the application of our ideas.

### *C. Explanation and Discussion of the Criteria*

The criteria in the tables as such are not very telling. Therefore we now give a more detailed explanation of their intended meaning:

**Size** relates to the size of a project. Typical metrics would be number of people involved, duration or budget. A big project requires more coordination which makes well defined interfaces as single point of contact between different implementations beneficial.

**Complexity** relates to the degree of difficulty of a project. Metrics might be the number of function points, a subjective estimation of complexity on an arbitrary scale or the newness of the problem to be solved.

**Risk of Changing Requirements:** This is a constant threat to project management and needs to be subjectively estimated. When requirements change the functionalities of a system must be adjusted which is easier when interfaces are in place.

**Multiple teams** make coordination more difficult which is why interfaces are necessary to reduce coordination effort.

**Time constraints** make it necessary to start a project at different points. With interfaces agreed upon the implementation can work form different sides and meet in the middle rather than work sequentially.

**Building from scratch** requires definition of interaction protocols. One might as well agree on one rather than let different standards evolve. Different protocols mostly appear in systems that have evolved over time.

**Domain Boundary:** Here one often needs to provide interoperability between different systems and one might as well do it in a proper way. Within one homogeneous domain interfaces add too much overhead but they provide valuable interoperability between domains.

**Different Systems** require interfaces and therefore doing it properly doesn't cost a lot.

**Scarce expert knowledge** is a problem when one needs people who understand two different systems. Experts for one system alone are easier to find and if interoperability is guaranteed via interfaces the experts spend less time integrating their solutions and more time on actually implementing their parts.

**Shared Services:** Reuse is especially likely if one service is determined to be available to different users.

**Multi Channel Scenarios** are a special case of shared services. The different channels use the same data and just provide it differently to the user. One service providing the data in a generic format can then be reused in every channel and only needs some adaption to the needs of the specific presentation device.

The Criteria as stated above are rather soft and one might want them to be more strictly specified in order to allow a consistent application of the criteria in several different projects. Complexity for example could be measured in function points [9]. This method allows a project manager to estimate the time needed to implement a project depending on its complexity which is measured by different factors like programming effort or number of people involved. In addition to defining metrics for measuring the criteria one would also need to define when exactly (from which number of function points for example) a project starts being complex or when a project has a high risk of changing requirements and when not. However, how important is a consistent application of the criteria above? We are not trying to establish a method to compare different projects in a consistent way. Our goal is to provide a tool that supports the decision for or against the adoption of SOA ideas in the specific context of a project. In the real world such a decision will always depend on subjective judgment rather than on highly sophisticated measuring frameworks. Therefore we consider the derivation of the criteria as the most important aspect of the given problem and the specification of strict metrics a second order goal which might be addressed in further research.

## IV. Application

Following [10], part of design science is the evaluation of ones research effort regarding utility, quality and efficacy. In an attempt to live up to those standards we will now demonstrate the application of our criteria to five projects from the chemicals industry. The projects were analyzed as part of an attempt to evaluate the benefits of SOA in the chemicals industry. The selection process was aimed at finding projects which would benefit from SOA. Therefore one might assume that the project selection is somewhat biased towards SOA. This is true with respect to the fact that the projects would all benefit from SOA. However, their suitability as first time SOA projects was not a selection criterion. However, the chosen selection should be considered as an illustration of the above criteria and their intended use and not as an attempt to empirically validate this work. To show the application of the criteria we will firstly give short descriptions of the projects we analyzed and then apply the criteria. The first column of the tables indicates whether the criterion gives and indication in favor or against SOA. The bottom row simply adds up the that column. This adding up should not be over emphasized and we will discuss the question of numerical evaluation of the criteria further down.

### A. The Case of Achem

At Achem we evaluated a project that analyzed the possible extension of an existing web portal. The portal is used to provide manufacturers with information and let them capture information that they need to give to Achem. In order to provide this functionality the web portal is integrated with the backend SAP Systems. Data is taken from special tables and displayed in the front-end. If data needs to be stored it is written to certain tables using SAP proprietary functions. For the users of the web portal there are several ways to retrieve and store data. The standard way is manually reading and capturing information. For larger volumes it is also possible to transmit files in predefined formats. Both approaches work in either direction. With the goal to further automate the process of retrieving and capturing data a connection to the ERP systems of the partners is being considered. However, this task is not simple because on the side of the partners the systems are very heterogeneous. Everything from spreadsheets to sophisticated ERP systems can be expected. Furthermore the possible partners change frequently.

### B. The Case of Bchem

At Bchem a complete redesign of the logistics systems was attempted in a project because the existing solution was terminal based and could not be adapted to serve the changing needs. The core of the system is a modified off the shelf logistics tool in combination with a large data base that is used to consolidate all relevant information. This information is drawn from ERP systems of the several business units. Upon consolidation the data is used for transport organization which involves several tasks like scheduling, giving a transport order to a carrier or managing in plant traffic. These tasks involve several rounds of bidirectional cross company communication.

### C. The Case of Cchem

At Cchem we analyzed a project that had been started to integrate a manufacturing system with the ERP. In the chemicals industry in general there is little interaction between ERP systems and the productive systems and there are few standard solutions to improve this situation. The specific problem at Cchem was that production quantities were manually entered into the ERP system after production. This is a constant source of errors, is expensive because a trained worker has to enter the information and most importantly causes delays because other processes can only be started once the production data has been entered. In order to improve this Cchem had decided to use a program that took data from the production system and triggered the necessary ERP actions. After rollout in one plant, the solution will be adapted for use at up to twelve different sites.

### D. The Case of Dchem

The question Dchem wanted to answer was whether or not SOA could help IT to perform better in the early phase of post merger integration. During a merger time is a crucial factor for success. Therefore Dchem's objective was to prepare the IT in order to be able to react very fast in case a merger happened. In the post merger phase one can differentiate between three types of data that must be integrated into the buying firms systems. The highest priority is given to core financial data on a very high level. This information must be available within weeks after the merger. Secondly Dchem needs more fine grained information on the success of each business units which is why data on a per customer per article base must be available to management within two months. Then in a third wave all other systems of lower priority are integrated with Dchem' systems.

### E. The Case of Echem

Echem uses a rather old self developed application to track changes in its SAP System. The application is used to document the whole workflow from first change request to the final transportation of the changes from the test system to the productive system. The application has evolved for several years and is perceived to be rather hard to modify. Changing business rules for example need to be hard coded which is expensive and tedious since business rules are subject to frequent changes. The present project has to construct an application that fulfills the need for flexibility and ease of use.

### F. Summary and Discussion of the Examples

From the short descriptions of the projects alone it is probably rather hard to tell, whether or not the evaluation in the tables really gives an indication of the right decision. Therefore we now discuss whether or not our personal judgment corresponds with the results in the tables.

**Achem** received six points in the evaluation and in fact we believe that SOA with web services is the only feasible way to

TABLE III
EVALUATING THE ACHEM PROJECT

| | | |
|---|---|---|
| + | Building from scratch | The connectors to the backend systems are already in place but the connectors to the partners are completely new |
| + | Size | Development teams of Achem and its partners are involved |
| - | Time | Not relevant |
| + | Domain boundary | Cross firm Interaction |
| + | Complexity | Heterogeneity of partner systems |
| - | Risk of changing requirements | Stable data requirements |
| + | Multiple teams | Implementations at both ends are necessary |
| - | Changing environment | The present system has been running for several years without changes |
| + | Different systems involved | Heterogeneous partner systems |
| + | Scarce expert knowledge | Experts for either system are in house but don't know the other systems respectively |
| + | Shared services | Once implemented a service could be used by all partners |
| + | Multi channel scenarios | Visual data retrieval stays necessary but automation is desired |
| **+6** | | |

TABLE IV
EVALUATING THE BCHEM PROJECT

| | | |
|---|---|---|
| - | Building from scratch | Backend and partner systems stay but the logistics system is built from scratch |
| + | Size | A multi million Euro project with several departments involved |
| +/- | Time | Not critical |
| + | Domain boundary | Cross firm interaction |
| + | Complexity | Heterogeneity of partner and backend systems |
| + | Risk of changing requirements | Logistics are a dynamic field at the moment |
| + | Multiple teams | Implementations at both ends are necessary |
| + | Changing environment | Logistics are a dynamic field at the moment |
| + | Different systems involved | Heterogeneous partner and backend systems |
| + | Scarce expert knowledge | Experts for each system are in house but don't always know the other systems respectively |
| + | Shared services | Once implemented a service could be used by all partners |
| + | Multi channel scenarios | Mobile devices are envisioned to be integrated |
| **+9** | | |

TABLE V
EVALUATING THE CCHEM PROJECT

| | | |
|---|---|---|
| - | Building from scratch | All systems exist already |
| - | Size | A small project with few people involved |
| - | Time | Of little relevance |
| + | Domain boundary | Cross systems interaction |
| - | Complexity | Few involved systems |
| - | Risk of changing requirements | No changes likely during implementation |
| + | Multiple teams | Implementations at both ends are necessary |
| - | Changing environment | All involved systems stay in place for decades |
| + | Different systems involved | Communication between different systems |
| + | Scarce expert knowledge | Experts for either system are in house but don't know the other system very well |
| + | Shared services | Once implemented a service could be used several times |
| - | Multi channel scenarios | Not relevant |
| **-2** | | |

automate the portal. The reason for this is that a 1:1 connection between Achem and the different partners is too expensive and therefore the portal was put in place as a platform for manual interaction.

**Bchem** currently conducts a very big project that would probably benefit considerably from SOA. Interestingly the project uses many aspects of SOA without calling it SOA. The communication among the different elements of the transportation management solution for example all communicate message based via one middleware.

**Cchem** is a very good example why SOA is an idea that is worth consideration. The whole project is only necessary because the production system is shipped with the wrong SAP connectors. Proper interfaces would have saved a lot of money in this context. However, the project itself is not very well suited for a proof of concept because it would require a lot of preconditions like the said interfaces in order to profit from SOA.

**Dchem** is a debatable case because on the one hand SOA offers notable benefits like the possibility for reuse of different components of an integration solution, on the other hand "quick and dirty" should be much faster. Therefore Dchem

TABLE VI
EVALUATING THE DCHEM PROJECT

| | | |
|---|---|---|
| + | Building from scratch | All source systems exist already but the interconnection part is new |
| - | Size | Rather small project with few involved members |
| + | Time | The crucial factor |
| + | Domain boundary | Cross company interaction |
| - | Complexity | Few systems with well defined requirements |
| - | Risk of changing requirements | No changes likely during implementation |
| + | Multiple teams | Implementations at both ends are necessary |
| - | Changing environment | Stable during implementation |
| + | Different systems involved | Communication between different systems |
| + | Scarce expert knowledge | There can be no experts who know the systems at Dchem and at a possible take over candidate |
| + | Shared services | Services could be reused in many mergers |
| - | Multi channel scenarios | Not relevant |
| **+2** | | |

TABLE VII
EVALUATING THE ECHEM PROJECT

| | | |
|---|---|---|
| +/- | Building from scratch | System is completely new but the target systems stay in place. |
| - | Size | Rather small project with few involved members |
| - | Time | Not important |
| + | Domain boundary | Interaction between the new application and SAP |
| - | Complexity | Few systems with well defined requirements |
| - | Risk of changing requirements | Rather small |
| - | Multiple teams | One team does the implementation |
| - | Changing environment | Target system will stay |
| + | Different systems involved | Communication between different systems |
| + | Scarce expert knowledge | Experts for application development and for SAP transports are required |
| + | Shared services | Once implemented the program can possibly used to track changes in other systems as well if the architecture is flexible enough |
| - | Multi channel scenarios | Not relevant |
| **-3** | | |

would surely not be a prime example for a proof of concepts SOA project because it would not benefit a lot in the short run.

**Echem** is a case that is similar to Dchem. The project itself would probably not benefit from SOA in the short run. Therefore this project would not be very well suited for a first SOA implementation.

Even though all the projects would benefit from SOA in one way or the other, only two of them promise to be successful as a proof of concepts for SOA. Achem would profit becuase SOA takes away complexity and Bchem would profit because SOA helps to cope with the complexity. Cchem shows very clearly how ignoring the SOA paradigms can become expensive in the long run while Dchem and Echem would benefit little from SOA in the short run. Therefore we conclude that the criteria listed in the tables serve well as indicators for suitable first SOA projects. They do not allow a precise measurement but they give an indication which is probably all that can be achieved without an overly sophisticated approach.

## V. CONCLUSIONS AND FURTHER RESEARCH

SOA in the authors' opinion is an interesting approach to designing and managing large system landscapes. When ignoring the marketing hype, well established ideas like modularity and loose coupling build the foundation of the concept which is then enriched with business oriented concepts that make SOA more tangible and facilitate transfer to real systems. Since we consider SOA more than a hype the question how this idea can be brought into the companies is an important one because even good ideas need good marketing to gain acceptance. One facet of this is to use it where it is most likely to provide immediate benefits. This is exactly what this work is trying to support by providing criteria for the selection process.

Possible extensions of the presented framework for the selection of SOA projects would be a weighing of the presented criteria. Complexity and project size for example are stronger drivers towards SOA than the number of teams involved in the development process. Therefore one could weigh the different factors and get a more differentiated picture. However, this step doesn't really provide any further insight form the authors' point of view. If a project is evaluated using the given criteria one will intuitively weigh them and the presented framework will serve as a checklist. On the other hand, if projects were evaluated solely based on the number in the last row, weighing the factors would be worth further consideration.

Extending the idea of weighing the factors one could use more formal tools like cost benefit analysis [11] or its extension by [12]. Practitioners apparently consider the AHP method [13] especially promising. Less so because it forces a ranking through pairwise comparisons but because it appears to be more sophisticated. This work didn't apply any of those

approaches because the focus was on criteria for SOA and not on the application of such a method.

In the presented way our criteria are little more than a checklist.An rather important next step would be the definition of proper metrics to measure the different criteria in a more transparent and reproducible manner. This would also enhance the applicability of the framework because comparisons across projects evaluated by different people would become feasible.

The paper used theoretical sources to deduce criteria for projects that promise to benefit from SOA in the short run. These criteria were then evaluated by applying them to five projects the authors conducted at major German chemicals firms. The result of this work is a set of criteria that should serve as a guideline for the selection of early SOA projects. Having such criteria is important because they make it more likely that successful projects can be conducted which in turn provides momentum for the move towards SOA.

## REFERENCES


[1] C. Legner and R. Heutschi, "Soa adoption in practice - findings from early soa implementations," in *Proceedings of the 15th European Conference on Information Systems (ECIS 2007), St. Gallen*, 2007.

[2] J. Siedersleben, "SOA revisited: Komponentenorientierung bei Systemlandschaften," *Wirtschaftsinformatik*, vol. 49, pp. 110–117, 2007.

[3] H. Kerzner, *Project Management: A Systems Approach to Planning, Scheduling, and Controlling*. John Wiley and Sons, 2003.

[4] R. Atkinson, "Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria," *International Journal of Project Management*, vol. 17, no. 6, pp. 337–342, 1999.

[5] A. Shenhar, O. Levy, and D. Dvir, "Mapping the dimensions of project success," *Project Management Journal*, vol. 28, no. 2, pp. 5–13, 1997.

[6] M. Cook, "Building enterprise information architectures: reengineering information systems," 1996.

[7] D. Krafzig, K. Banke, and D. Slama, *Enterprise SOA: Service-oriented Architecture Best Practices*. Prentice Hall Ptr, 2004.

[8] R. Heutschi, "Serviceorientierte Architektur, Architekturmodell und Umsetzung in der Praxis," Master's thesis, IWI-HSG, 2006.

[9] G. Low and D. Jeffery, "Function points in the estimation and evaluation of the softwareprocess," *Software Engineering, IEEE Transactions on*, vol. 16, no. 1, pp. 64–71, 1990.

[10] A. Hevner, S. March, J. Park, and S. Ram, "Design Science in Information Systems Research," *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, 2004.

[11] C. Zangemeister, *Nutzwertanalyse in der Systemtechnik*. Wittemann, 1970.

[12] H. Grob, "Investitionsrechnung für Informations-und Kommunikationssysteme auf der Grundlage von Preis-Leistungs-Modellen," *Integration und Flexibilität: Eine Herausforderung für die Allgemeine Betriebswirtschaftslehre, Hrsg.: D. Adam ua*, pp. 335–352, 1989.

[13] T. Saaty, *What is the analytic hierarchy process?* Springer-Verlag New York, Inc. New York, NY, USA, 1988.

[14] G. Alonso, F. Casati, H. Kuno, and V. Machiraju, "Web Services: Concepts, Architectures and Applications," 2004.

[15] D. Dvir, T. Raz, and A. Shenhar, "An empirical analysis of the relationship between project planning and project success," *International Journal of Project Management*, vol. 21, no. 2, pp. 89–95, 2003.

[16] T. Erl, *Service Oriented Architecture: Concepts*. Technology, and Design (Upper Saddle River, NJ: Prentice Hall, 2005).

[17] M. Freeman and P. Beale, "Measuring project success," *Project Management Journal*, vol. 23, no. 1, pp. 8–17, 1992.

[18] R. Heutschi, C. Legner, B. Nr, B. HSG, C. BN, A. Back, W. Brenner, H. Österle, and R. Winter, "Serviceorientierte Architekturen: Vom Konzept zum Einsatz in der Praxis," *Integration, Informationslogistik und Architektur–Proceedings der DW2006, September 21-22, Friedrichshafen, Germany*, pp. 361–382.

[19] K. Nagel, *Nutzen der Informationsverarbeitung: Methoden zur Bewertung von strategischen Wettbewerbsvorteilen, Produktivitätsverbesserungen und Kosteneinsparungen*. R. Oldenbourg Verlag, 1990.

[20] E. Newcomer and G. Lomow, *Understanding SOA with Web Services (Independent Technology Guides)*. Addison-Wesley Professional, 2004.

[21] J. Schemm, R. Heutschi, T. Vogel, K. Wende, C. Legner, B. Nr, B. HSG, C. BN, A. Back, W. Brenner *et al.*, "Serviceorientierte Architekturen: Einordnung im Business Engineering," *Universit of St. Gallen*, 2006.

[22] D. Woods and T. Mattern, "Enterprise SOA: Designing IT for Business Innovation," 2006.

[23] R. Yin, *Case Study Research: design and methods*. Sage Publications Inc, 2003.

[24] R. Zarnekow, W. Brenner, and U. Pilgram, *Integriertes Informationsmanagement: Strategien und Lösungen Für das Management von IT-Dienstleistungen*. Springer, 2005.