Service Oriented Architecture and Business Innovation

Bendik Bygstad               Tor-Morten Grønli
Norwegian School of IT         Norwegian School of IT
bendik.bygstad@nith.no       tmg@nith.no

Abstract

While the approach of Service Oriented Architecture (SOA) has been generally accepted, there are concerns that it is mainly used as a technical IT solution, not as a vehicle for business transformation. An important question, then, is how can SOA support business innovation in practice?

We investigated this issue in an in-depth case study at an airline company. Analyzing the relationship between innovation and SOA in the company over time, we offer the following conclusion. We do find that SOA gives strong support to business innovation, under two conditions. First, the implementation of SOA has to be comprehensive, i.e. it should include the core processes of the business. Second, the top management (and partners) need to understand the principles of SOA.

1. Introduction

According to Gartner [4] Service Oriented Architecture (SOA) is now beyond the initial hype, and has entered a more sober phase of growth and exploitation. This description seems to be reasonable, but as SOA matures, the overall picture is ambiguous. On the one hand, SOA is well established as the state-of-the-art IT architecture, both by academics and in the practitioner communities [13]. Also, there hardly seems to be any viable alternatives at the moment.

On the other hand, although many organizations have committed themselves (at least intentionally) to the SOA approach, there are not many convincing success stories. Compared to the initial expectations of frictionless integration and business transformation, the practical experiences are much more modest, and even somewhat disappointing. Recent research suggests that in practice SOA has been reduced to a technical IT architecture instead of a vehicle for business innovation [5].

It is important to bear in mind that SOA initially was framed as a business innovation approach, promising a long-awaited capacity for providing businesses with a real-time flexibility in a turbulent and competitive environment [18]. The key to this promise is that IT should not only align with business to support existing processes, but also contribute to a business transformation. In practice this implies that the SOA architecture enables new business services at a speed that matches the needs of the fast-changing corporation.

From past experiences both IS researchers and IT practitioners have learned to be wary regarding such grand promises. In particular, we have learned that the relationship between IT capabilities and business performance is not always straightforward. Although it has been shown that IT capabilities tend to increase business performance [1], it is also well documented that exploiting these capabilities depends on a number of factors, and that bandwagon behavior often leads to mindless innovation [16]. As many companies have experienced, SOA is not a cheap technology that is easy to use.

We believe, however, that the basic idea of SOA supporting business innovation is attractive and exciting, and deserves to be investigated empirically. In this research we conducted a relatively detailed investigation of a successful SOA case. We have chosen to focus on a particular implementation of SOA called the Enterprise Service Bus (ESB) architecture, and its practical use in a company. The ESB is an IT architecture that aims at being able to support two seemingly contradictory features: it integrates a network of business partners at a transactional level, enabling real-time systems to communicate seamlessly. At the same time the components are loosely coupled; it is possible to add or subtract business partners at
short notice, without affecting the daily running of operations [13].

The ESB is an attractive idea, which has received much attention over the past few years. However, the ESB concept is primarily a technical architecture, and many issues remain much less known. In particular, the more direct relationship between IT infrastructure and innovative capability is not described in the ESB literature.

How should a company organize this in practice? The purpose of this research is to undertake an exploratory, but in-depth investigation of this issue, where we go beyond the promises of technology, and assess the interplay of technology and business innovation in practical setting. Our research question, then, is how can a SOA architecture such as the ESB support business innovation?

We proceed by reviewing the concepts and practices of business innovation, SOA and ESB. Then we present our method, and our case company. We describe the technical solution in some detail, before analyzing how this solution enabled the company to introduce a number of innovative services in very short time, and to exploit them successfully in a turbulent competitive environment. Finally we discuss the conditions for this approach to be viable.

2. Review

In this section we offer a brief review of business innovation, service oriented architecture and the enterprise service bus.

2.1. Business Innovation and SOA

The innovation of new services has transformed several industries over the past decade, for example travel services, banking, gaming and the music industry [19]. This development has been heavily dependent on IT, in particular Internet technology, to such a degree that we should conceptualize these innovations not as “business/IT alignment”, but as the results of a mutual reinforcement process: technology opens a space of possibilities, which spurs the creation of new services. The new services increase the space of possibilities, which again spurs the creation of new services [2].

The vision of technological flexibility in the development process was described by the agile software development community in the early 2000s [8]. The agile project promises this through short product iterations, continuous communication between user and developer and a reliance on talent and skill, rather than formal processes. The aim is to deliver business value, not software components [15].

This capability is obviously dependent on more than technology; it also requires a certain amount of business agility. Agility is not primarily connected to strategy and aims, but rather to resources and capabilities. To be able to act with agility, it certainly takes financial, human and technical resources. Just as importantly, it also takes a number of capabilities, such as the ability to understand changes early, to assess the opportunities and to implement new solutions quickly [9].

This mode of operating is particularly demanding, because it implies that the changes must happen on the fly. Productivity and quality must be maintained while new services are added. There is no time for year-long development projects with gradual deployment of solutions [9]. Thus, one important aspect of agility is to include resources from outside partners, at relatively short notice, fully integrated into the IT infrastructure. Obviously, these requirements put an unprecedented strain on the IT organization. One of the most promising responses to these challenges is service oriented IT architectures.

2.2. Service Oriented Architecture (SOA)

Since the term SOA was coined in 1996, it has become the state-of-the-art of software architecture thinking, and all large software vendors today offer various frameworks and implementations of SOA [13]. SOA is a framework for designing flexible and loosely-integrated services, in distributed environments. The main goal of SOA is to address the following challenge: How should a corporate IT architecture support organizational innovation and agility in an unstable global environment? Included in the challenge are such issues as, how can the architecture support both flexibility (the ability to include both new and old systems into a distributed but seamless whole), efficiency (the ability to run the operation 24/7 at acceptable costs) and reuse (sharing components over the whole organization and even outside it)?

A key challenge when implementing SOA is that to achieve full effect, traditionally siloed business processes needs to be transformed into more loosely coupled services [3]. Demirkan et al. argue that SOA may increase business agility, but that this requires that it is included as a part of the enterprise agenda, not being treated as a technical issue. In the same vein Tsai et al. (2007) emphasized that SOA is a dynamic architecture, offering the opportunity to have processes connect and reconnect in an agile manner at runtime [20].

These are rather demanding requirements. To be frank, the IT department in most companies has never
been even close to being capable of delivering such services. Fortunately, available technology may be part of the solution. One popular solution to implement SOA is the Enterprise Service Bus.

2.3. Enterprise Service Bus

In some of the literature ESB is described as a product, a middleware software. Following Rosen et al. [13] we will view ESB as an architectural pattern with existing products to be implementations of this pattern. Well known implementations are for example Mule [17] and Open ESB [13].

An ESB allows for seamless integration and communication between different systems inside and outside an organization. This is enabled by the bus middleware, which allows for loose coupling at the business level. The ESB architecture is illustrated below, very simplified, in figure 1.

![ESB Architecture Diagram](image)

The main features of ESB are [12]:

**Connectivity:** The ESB connects any application with another, without coupling the message sender and the message receiver.

**Routing:** The ESB must identify the end destination of an incoming message. It must also be able to assess the integrity of the message, and (if needed) adding the necessary information.

**Transformation:** The bus should be able to integrate seamlessly applications with different protocols, such as HTTP, JMS, SMTP and TPC. The bus must also be able to transform a message from one format to another.

**Security:** Standard security mechanisms must be available, such as authentication, authorization and encryption. This is important both to prevent malicious use of the bus, and to ensure that outgoing messages comply with the requirements of external partners.

3. Method

We conducted an in-depth case study [10], building on a critical realist approach [14], and focused on the relationship between IT architecture and innovation. The general approach was process-oriented [7]; taking a longitudinal view we were looking for explanations of events. Data collection was conducted during a period of eighteen months of 2008 and 2009.

The case company, Norwegian Corp, was chosen for two reasons. First, it was a young and successful company, with a reputation for innovation. Second, the company was expanding its initial successful IT infrastructure of booking services into new ICT-based services, thus constituting a fruitful case to study the relationship between innovation and IT architecture.

Ten managers and specialists were interviewed, each circa 2 hours, some of them twice. See table 1 below. In addition a large volume of technical documentation (business plans, project plans, contracts, technical architecture documents) was collected.

<table>
<thead>
<tr>
<th>Date</th>
<th>Informant</th>
<th>Key topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>March 2008</td>
<td>Business developer</td>
<td>Business models for Norwegian Corp</td>
</tr>
<tr>
<td>April 2008</td>
<td>CIO</td>
<td>IT architecture and innovation</td>
</tr>
<tr>
<td>April 2008</td>
<td>Marketing consultant</td>
<td>Revenue management</td>
</tr>
<tr>
<td>May 2008</td>
<td>Bank manager</td>
<td>Establishing Bank Norwegian</td>
</tr>
<tr>
<td>May 2008</td>
<td>Sales manager</td>
<td>Innovation processes, Internet sales</td>
</tr>
<tr>
<td>July 2008</td>
<td>CIO</td>
<td>Culture and management</td>
</tr>
<tr>
<td>Aug 2008</td>
<td>IT project manager</td>
<td>Project culture in Norwegian</td>
</tr>
<tr>
<td>Sept 2008</td>
<td>CIO</td>
<td>Call Norwegian project</td>
</tr>
<tr>
<td>Oct 2008</td>
<td>IT dept. manager</td>
<td>IT architecture and Call Norwegian project</td>
</tr>
<tr>
<td>Oct 2008</td>
<td>Project manager</td>
<td>Call Norwegian project and portal architecture</td>
</tr>
<tr>
<td>Jan 2009</td>
<td>Systems director</td>
<td>IT architecture</td>
</tr>
<tr>
<td>May 2009</td>
<td>Systems director</td>
<td>IT architecture and Call Norwegian</td>
</tr>
<tr>
<td>Dec 2009</td>
<td>CIO</td>
<td>Call Norwegian, business development</td>
</tr>
</tbody>
</table>

Table 1. Informants at Norwegian.
Data analysis was conducted in the following three steps [11]. First, a time line was established, and important events were identified. We used the technique of forward-chaining to understand the intentions of key stakeholders and the subsequent projects and results. Then backward-chaining analysis was conducted, in order to find explanations, in particular for unexpected outcomes.

Second, a comprehensive analysis of architecture development and business innovation was done, focusing particularly on the interplay between these dimensions. As shown in the Discussion section we tracked and analyzed how a particular service innovation was directly linked to the IT architecture. For example, we analyzed in detail some self-reinforcing mechanisms of the innovation process, such as the “NAPI” interface. This was a service interface that Norwegian offered to partners, for example travel agencies. When agencies started to use it they proposed that, in addition to booking information, it should also be possible to exchange economic transactions the same way. This triggered the development of an extension, which enabled the partners to do economic clearing through a service from SAP.

Third, to ensure internal validity the preliminary findings were discussed with informants. For example, we conducted a full session with a technical specialist in order to confirm the correctness of our initial analysis of the role of the ESB. Finally, paper drafts were sent to key informants for comments.

3.1. The case

Norwegian Corp is a privately owned company registered on the Oslo stock exchange. Led by the charismatic former jet fighter pilot Bjørn Kjos, the Norwegian airline company grew from virtually zero in 2002 to become the largest low-cost carrier in Scandinavia in 2009. Today Norwegian operates a total of 176 routes to 86 destinations in Europe (see figure 2), and carried 9.1 million passengers in 2008. The company has 1400 employees and revenues in 2008 were 6.2 bn. NOK ($ 900 M.) Profits, however, have been moderate.

The company has achieved this growth through an aggressive pricing and an agile business culture, with the following attributes.

First, there is a strong entrepreneur culture. The respondents of the case study were all asked to characterize the culture, and were quite unanimous in their views. There is still a strong entrepreneur culture, with innovation in small teams, flat organization, empowered employees and a strong determination to succeed.

Second, there are no “IT projects”, only business projects. New ideas come along as business proposals. They are evaluated on the sole criterion of financial benefits. If approved the involved units will co-operate to accomplish the necessary work, small changes in an informal group; larger initiatives in formal projects. This way, the people who design a solution will also operate it.

Third, it is a relatively flat organization. Employees are empowered to a large degree, with clear business accountabilities. A middle manager commented: “As long as I reach the company objectives I am free to choose my actions. This includes the right to propose and implement changes in the computer systems. Of course, this does not mean that Norwegian is a flat organization, but it gives you the feeling that it is. The reason is that a good idea is never rejected for hierarchical reasons.”

4. The ESB Architecture

As a new entrant into the very competitive airline market, Norwegian started in 2002 with a very basic IT solution. As the company expanded quickly, the need for an IT architecture was acknowledged, and a CIO and two IT architects were hired from one of the competitors.

They had rather clear ideas on what to do (which they had not been allowed to develop at their former employer), and started in 2003 to experiment with a bus architecture. The aim was to develop a quite simple service bus, enabling later business development through reuse of components. The ESB was developed in 2004, and expanded gradually over the following years.

The enterprise system architecture was created as an enterprise service bus architecture, rather similar to the ideas and structures used by the openESB initiative [6]. An overview is offered in figure 2.
Figure 2 gives an overview of the enterprise architecture and positions the ESB in the middle as the core system. As with most other architectures, different components communicate through a set of layers. One advantage of using the ESB structure is a loose coupling between enterprise components. The different layers are more easily maintained separately. We will further explore each of the layers in the ESB architecture more thoroughly.

4.1. User Interface Layer

The enterprise system contains several modules developed for communication with external partners and customers. These modules include Norwegian’s web page for reservations, mobile portal, agent portal, and the publicly available Norwegian Application Programming Interface (NAPI) (See UI layer in Figure 2). All these modules connect to the internal system by connecting to Web Service endpoints on the ESB. Data are exchanged as XML through standard communication protocols.

All modules have separate points of contact, each offering a public file describing the available business methods. The majority of the interface modules are designed and programmed in Microsoft .Net technology. Figure 2 shows the outline of the system highlighting the separation of concerns through modularization and interfaces. The possibility to communicate across technologies, such as Microsoft .Net and Oracle’s Java, is another strength made possible by the ESB and common formats for data exchange.

4.2. Business Infrastructure

The core module of the ESB is illustrated with the grey box in the centre of figure 2, in the domain layer. This core application controls business process management and flow of control between services. This is a Java based system which runs in a clustered environment on a load balanced, multi-core, machines (JBoss servers). The system has a front façade layer which handles all requests from external partners, based on individual service contracts through the interface endpoints.

A Norwegian Application Interface (NAPI) is offered to partners needing individually tailored interfaces.

During sales campaigns the number of searches may approach 1 million per day. To increase performance a cache is implemented on top of traditional server load balancing functionality up front. This ensures that all calls are handled here before being redirected to the internal modules.

The core application is driving all processes in the business, in addition to handling requests from external sources. The requests may as well originate from other internal applications in the enterprise. Examples of internal business processes the core system component handles are: availability searches, ticket purchases, service purchases (extra luggage, customer requests etc) and module synchronization.

A problem often seen in components evolving and growing over time is the temptation to keep loads of business process data in the core. Instead, this is solved by utilizing components further down in the hierarchy more suited for such work. Deploying
stateless components increases the ability to handle a large number of simultaneous requests.

Aiming for as loose coupling as possible between the core and other modules a service layer is included. Web services in this layer process communication and output from other modules. Within each web service technical modules enable event handling and data storages, as well as retrieving data and performing message calls.

One feature of standard enterprise service bus systems is a security module. In this solution security and authorization is quite complex since the number of modules and external systems is high. The problem is addressed by using standard open source security modules, which are adapted and wrapped by custom tailored code to ensure sustainability of the application suite.

4.3. Service Layer

Facade modules and wrappers, illustrated in the service layer in figure 2, encapsulate technical services and create abstraction by introducing a private programming interface. This approach enables programming interfaces between application modules, thereby enabling an easy plug-in of new modules. A wrapper module may have additional tasks such as performing multiple updates, performing caching services and executing message services. These modules and wrappers, though small components, are nonetheless key parts of this architecture to be able to sustain a viable ESB.

4.4. Foundation Layer

The underlying persistence foundation in the architecture (illustrated in the bottom of Figure 2) includes Data warehouses, Amadeus booking system (Amadeus Reservation and Amadeus Inventory), the Norwegian legacy back office system (BONO) and SAP. These services are separate applications, but due to façade modules and wrappers they can also be exploited as mere data storages. This highlights the strength of ESB, enabling the architecture to use other large systems as sources of information rather than having these systems setting constraints on ESB.

5. Analysis

From 2003 Norwegian introduced a number of new ICT-based services in order to establish a strong position in the home market, and to grow in Europe. In this section we describe and analyze how Norwegian exploited the flexibility of their ESB architecture in order to innovate new services. The services were developed in close cooperation between the top management group and the IT senior executives. For each innovation we discuss the role of the ESB architecture, presented in the previous section.

2004: Internet portal

After establishing the ESB architecture in 2003, the solution was set into production in 2004. The main challenge at the time was how to make customer book on the Internet, and not at travel agents (which services were quite expensive for a low-cost airline).

The first new services on the ESB were SMS (Short Message Service) booking and, most importantly, an Internet portal. The portal used the full features of the ESB from the start, enabling easy online bookings. In addition, the layered structure of the architecture allowed for parallel teams in development, speeding up delivery.

Role of ESB architecture: The portal was enabled and supported by the ESB. Due to highly adaptable data formats, both the SMS and Internet portals could use the same façade endpoints. Data could be restructured after retrieval, facilitating for quick, tailored responses.

2005: The low-price calendar

A major obstacle for low-price passengers at the time was how to find the cheap tickets, which used to be hidden inside a complex pricing structure. Capitalizing on their new ESB architecture Norwegian solved this problem in 2005 when the low-price calendar was introduced, which showed the cheapest flights to any chosen destination. The low-price calendar was an outstanding success, increasing the number of bookings substantially. It was internationally patented, although later copied by many other airlines.

Role of ESB architecture: The low-price calendar was mainly a result from innovation in the UI layer, made possible by the services in the façade layer. To avoid response time, the availability cache (building on existing processes) enabled complex searches with fast responses.

2005: The NAPI interface to partners

To expand their distribution, agencies and search engines were now given full access to Norwegian services, through the Norwegian Application Interface (NAPI).
Role of ESB architecture: SAP was connected to the ESB, providing clearing with partners.

2006: Electronic dialogue with 90% of customers
This dialogue included email and web marketing, on-line sales, booking and check-in. This was enabled by the easy use of the Internet portal and the performance of ESB.

Role of ESB architecture: These innovations in the service layer emerged as the result of CRM and ERP systems now being used as sources of information rather than isolated systems. Inventory was now moved to Amadeus, enabling a more seamless access to pricing data. High performance was enabled by load balancing in front (performance cache), and by load balancing in back (technical façade layer). The design of the domain layer allowed for quick response time, with up to one million searches each day.

2007: Bank Norwegian
In 2007 the company decided to enter the banking market with Bank Norwegian. From the outset, CEO Bjørn Kjos stated that:

“We today have one of the most visited web pages in Norway, with 2-3 million visitors each month. We aim at coupling this traffic towards bank services.”

The Director of Business Development commented:

“We had established a very flexible IT architecture, and we realized at the time that it would be possible to innovate new services on this. First we were just brainstorming rather freely; how could a combination of brand and technology generate new business?”

The establishment of the bank was done during 6 months, serving 50.000 customers in 2008.

Role of ESB architecture: Bank Norwegian was linked to the ESB, as services in the foundation layer allowed for new systems to be used as pluggable parts of the system architecture. Each booking generates an XML message to bank Norwegian, and customer collects frequent flyer points in Bank Norwegian.

2008: Call Norwegian Portal
The aim of the mobile portal was to allow for easy airline booking, and to offer mobile broadband on the airport and (in 2009) during flight. Said the Portal Director:

“We focused on how to make money on new services, analyzing which services we should provide ourselves, which we should buy and how they should be integrated. At the same time we are very concerned about our architecture; it is as an important ambition to maintain it as ‘clean’ as possible. We don’t really go for cutting-edge solutions. Rather, we combine known and stable components in new ways.”

Role of ESB architecture: Booking and ticketing are using the services on the ESB. The web server routes the calls to mobile server, and presents a simple html GUI. The Call Norwegian Portal was made possible by innovation over an existing service foundation in the service layer as well as technical wrappers in the façade layers.

2008: Fly Nordic merger
In 2008 Norwegian also bought the Swedish airline FlyNordic. It was fully integrated into the Norwegian network, including scheduling and booking.

Role of ESB architecture: With the ESB, conversion of data and integration was done in 3 months.

2009: Bar code ticket on the mobile phone
The mobile solution was extended in 2009, when the possibility of having a bar code ticket on the mobile phone was introduced.

Role of ESB architecture: The mobile portal for sales uses the ESB in much the same way as the Internet portal, needing only new GUIs. The overall design and continuous innovation and adaption of services elsewhere in the ESB over the recent years led to the implementation being a small effort.

Summing up this section, we find that the ESB supports innovation in several aspects:

- Simplifies the introduction of new UIs and technologies, such as mobiles
- Integrates different technologies and applications into new services
Enables rapid integration with partner systems
Supports high performance and low response time

Thus, we argue overall that the ESB architecture has enabled Norwegian to innovate quickly in their core airline business, by adding more services. ESB also enabled the company to expand their services horizontally, by adding new line of business, drawing on the shared resources. In the words of the IT architect of Norwegian:

“The ESB makes services available that enable and support creativity at front-end. Service innovators need only relate to the defined services in the ESB façades”.

6. Discussion

In this section we first discuss the possible lessons learned from our case; then we briefly assess limitations.

6.1 Lessons learned

We have described Norwegian’s solution in some detail in order to assess the role of SOA in business innovation at a practical level. Related to our initial discussion of the lack of SOA success cases, and to the general impression that SOA has been reduced to a technical issue [5], we believe that the Norwegian case illustrates that SOA indeed can support business innovation.

Further, we argue that there is a mutually reinforcing process between business innovation and SOA; SOA enables an agile organization to innovate new services, while the agile organization increases the options of the SOA by adding more services and components [2, 18].

This dynamics transcends the purely technical role of SOA, by placing SOA at the center of business innovation process. While innovation at Norwegian certainly is business (not technology) driven, it is the interplay of business and SOA that allows Norwegian to innovate new services, and to implement them quickly. Managing this process has been a learning partnership between business specialists and IT people, developing a flexible and very operative solution step by step.

While we do think that there are important lessons to learn from the Norwegian experience, we would like to emphasize that there are conditions which need to be satisfied in order to succeed with the approach described in this paper.

Through our analysis of Norwegian, we have identified two important conditions for the approach to work. First, the implementation of ESB has to be comprehensive, i.e. it should include the core processes of the business. As shown in section 4.2 SOA was introduced in the central business processes of the organization, and business critical operations were implemented in the bus architecture.

This is crucial for achieving the desired flexibility, because the effect of a minor or partial ESB implementation is bound to be very limited [3]. This would lead to two parallel architectures, with a new set of integration and workflow challenges.

It is also a very demanding condition, because it means that a company with many legacy applications may find it excruciatingly difficult to redesign them in order to work on the ESB. In our case, Norwegian was in a lucky position, building their IT architecture more or less from scratch.

Keeping the architecture “clean”, i.e. avoiding direct communication between services (omitting the ESB) is essential for the sustainability of the solution. The pressures of project deadlines is obviously a threat to this aim, and the IT architect of Norwegian emphasized that “my first and top priority is protecting the integrity the ESB structure, no matter how important a project deadline is”.

Second, the top management (and partners) needs to understand the principles of SOA. It is often said that the IT people should understand the needs of the business. We definitely subscribe to this view, but would also add that in order to succeed the business people should also understand the principles of the SOA. Why? The reason is that IT-based business innovation is inherently dependent on understanding and exploiting the resources of the company [1]. We believe that one interesting finding of the Norwegian case is that the quick innovation and deployment of new services was the result of very close cooperation between the business and IT side of the company. Often in projects, business people of Norwegian would comment, “OK, let’s solve that problem in the bus”.

The essence of this condition is that managers need to understand the reinforcement mechanisms of business innovation and the SOA. A flexible IT architecture provides resources on which the business people may innovate. As long as the architecture is kept clean (i.e. adhering to the principles of the SOA), the new services on the SOA not only increase the income of the company, but also increase the spectrum of possibilities for new innovations.
6.2 Limitations to findings

We acknowledge that there are limitations to our findings. These limitations are strongly associated to our research design, which is a process case study. As characterized by Langley, such studies are strong on accuracy and depth, but weaker on generalization [7].

Our findings then, have relevance as an example of how an organization may capitalize on the ESB architecture, and the conditions for the approach to work. The area of usefulness is likely to be similar modern business organizations with a highly skilled IT department.

7. Conclusion

This research was conducted in response to current concerns with the diffusion and benefits of Service Oriented Architecture (SOA), where some researchers have found that SOA is mostly perceived as a technical issue, failing to give a clear business contribution. Our interest was to investigate the potential role of SOA in business innovation, in a practical setting.

Our research question was, how can a service oriented architecture support business innovation? We researched this question through a longitudinal case study in an international airline.

We found that SOA gives strong support to business innovation, through a flexible and modularized architecture that integrates seamlessly at transaction level. In terms of service innovation we found that SOA supports the design and implementation of new services by combing internal and external IT resources. In terms of business agility we found that the SOA enables the deployment of such services within short time spans, in windows of opportunity.

We have identified two conditions for the approach to be viable. First, the implementation of ESB has to be comprehensive, i.e. it should include the core processes of the business. Second, the top management and business partners needs to understand the principles of ESB.

Our findings were based on a single case study. Further research should broaden the scope of investigation to include more organizations in other lines of business. In particular, we think that the area of e-Government, which is in need of both service innovations and loose couplings between various actors, is a relevant field.

Acknowledgements

We thank the Norwegian Corp, in particular Helge Bergh, for engaging in this case study. We also thank George Ghinea and Wanda Presthus for comments.

References


