# Management of Service-oriented Architectures (SoA)

FINANCIAL INSTITUTIONS NEED FLEXIBLE IT ARCHITECTURES TO REACT APPROPRIATELY TO MARKET DEMANDS. THUS, WE PRESENT AN ARCHITECTURAL APPROACH BASED ON THE CONCEPT OF A SERVICE-ORIENTED ARCHITECTURE (SOA) THAT ENABLES THE MANAGEMENT OF FLEXIBLE BUSINESS PROCESSES BY INTEGRATING LEGACY SYSTEMS.

RAINER BERBNER
RALF STEINMETZ
OLIVER HECKMANN

## Introduction

IT architectures within banks and financial institutions are characterised by a large amount of different legacy systems, middleware platforms, programming languages, operating systems and communication mechanisms. This heterogeneity has led to a high complexity that is barely manageable. Traditional EAI (Enterprise Application Integration) solutions have often failed to overcome this heterogeneity. Thus, the IT architectures within financial institutions suffer from a lack of agility and from inefficiency. This means that financial institutions often are not able to match their business requirements onto the underlying IT architecture to react appropriately to market demands.

A Service-oriented Architecture (SoA) aims at overcoming heterogeneity of legacy systems as well as establishing flexible and agile business processes (e.g. [4][5]). However, the management of business processes based on a SoA is often neglected. Thus, we present a comprehensive approach that enables the management of flexible business processes based on a SoA.

## Service-oriented Architecture (SoA)

A Service-oriented Architecture (SoA) is based on services as fundamental elements for developing applications/solutions. Services are self-describing, platform-agnostic computational elements that support rapid, low-cost composition of distributed applications [5]. A SoA is characterised by the loose-coupling of the services involved. This means that services can be replaced by other services at runtime. Services communicate with each other by sending and receiving messages. Furthermore, a SoA supports location transparency. Services should have their definitions and location information stored in a repository and could be accessible by a variety of clients that could locate and invoke the services irrespective of their location.

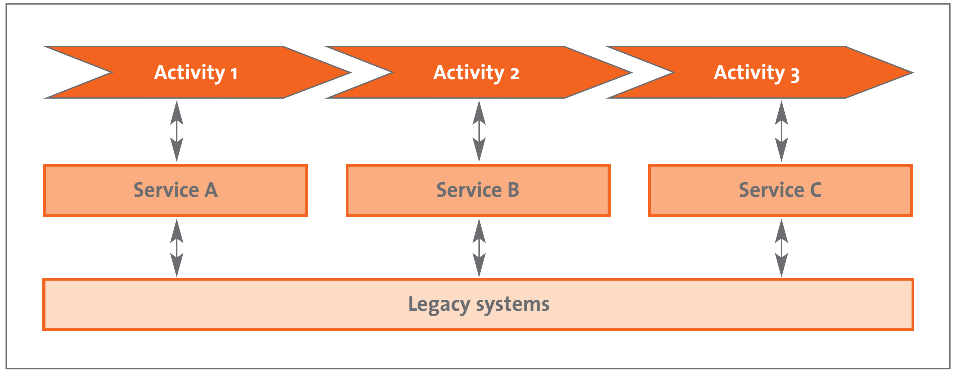

*Figure 1: Integration of legacy systems based on a Service-oriented Architecture (SoA) [2]*

## Cost Savings and Agility

Due to the simplification of the IT landscape, the streamlining of the code base, and technology independence, a SoA can help to reduce the long-term costs of IT [4]. Additionally, future changes can be made more easily and maintenance efforts can be targeted to business functionality. Furthermore, services can be reused in different processes and applications. A SoA is very well suited to support changing business processes, because capabilities (in form of services) can be composed in the most efficient way to achieve a high level of agility. A SoA is not dependent on a certain technology. As a consequence, a SoA decouples the long lifecycle of the IT architecture from the shorter lifecycle of underlying technologies.

## Decomposing Business Processes by means of a SoA

To achieve optimal support by a SoA, first business processes have to be conceptually decomposed into activities (so-called process patterns). The decomposition is continued until the process patterns have the optimal granularity which is determined by the business context, the functional handling, as well as the optimal support by the information systems. Figure 1 shows the mapping of conceptually modelled activities (process patterns) onto services. A credit process can be decomposed into the process patterns loan request, credit assessment, servicing and workout. The process pattern credit assessment is decomposed again into the process patterns internal rating, external rating, and decision [2]. The functionality of these process patterns is provided by services.

This architectural approach supports the reusability of process patterns in different business processes of a financial institution. Abstract process patterns and their implementations can be stored in repositories. As a consequence, business processes can be composed by combining process patterns out of a construction kit. This approach becomes even more effective if these process patterns can be customised by parameters. This allows additional flexibility and adaptability of business processes. This architectural approach therefore brings together standardisation and individualisation, because the combination of standardised components allows the creation of individualised business processes. Since services are loosely-coupled, this architectural approach allows the integration of services offered by external service providers (Figure 2). As an example, the credit rating of a customer can be provided by an external service provider.

**Web Service Technology**
Web Service technology can be seen as a realization of the SoA concept. Web Services as an emerging technology based on open XML standards have the potential to overcome integration problems.

A Web Service is defined as a software system identified by a URI, whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web Service in a manner prescribed by its definition, using XML based messages conveyed by Internet protocols. WSDL (Web Service Definition Language), SOAP (Simple Object Access Protocol), and UDDI (Universal Description, Discovery and Integration) form the Web Service core standards. WSDL is used as interface definition language, SOAP as communication protocol, and UDDI as a repository to publish and search for particular Web Services. Legacy systems expose their functionality as Web Services. Web Services can be composed to establish business processes without any effects on the underlying legacy systems. As a result enterprises can react much faster to dynamically changing markets and sophisticated customers. Furthermore, cross-organisational business processes can be established by the integration of external Web Services (e.g. external Web Services for credit rating).

**Management of a Service-oriented Architecture (SoA)**
We present an enhanced Service-oriented Architecture (SoA) realized by Web Services (Figure 3) with comprehensive management support that aims at reliable business processes.

A crucial management issue to establish reliable business processes is the Quality of Service (QoS) of the Web Services involved. Especially if external Web Services are considered. Thus, the decision, which particular Web Service among a group of Web Services with similar functionality (e.g. external credit rating) is invoked, depends on its QoS properties. The main QoS criteria (besides the price of the Web Service) are:

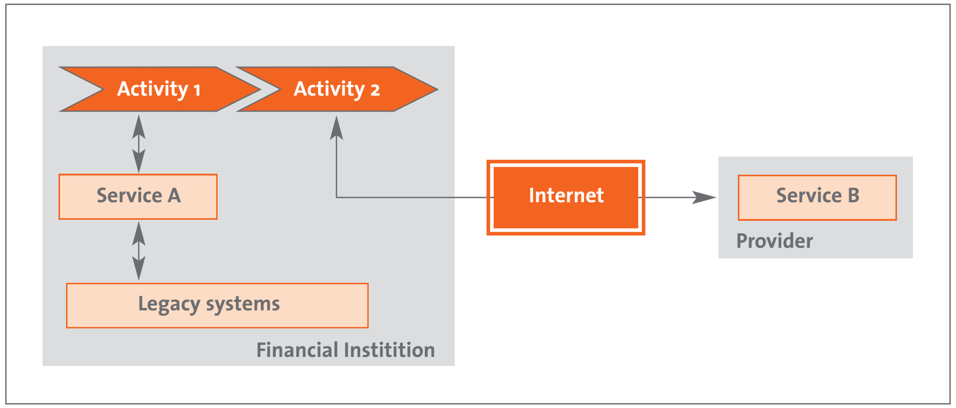

*Figure 2: Integration of external Services [2]*

- Availability: Availability of a Web Service means the probability that a certain Web Service is available when invoked by a client. A Web Service is considered as available if it is able to respond to a request within a defined time interval.
- Performance: Performance as a generic term can be measured by the throughput and the response time. Throughput means the number of requests that can be processed during a defined time slot. The response time is the sum of transmission time and processing time and can be measured as the time required processing a request.
- Error rate: The error rate specifies the number of processing errors within a particular time interval.
- Security: Security is a comprehensive criterion. Therefore, it is assessed by the sub criteria authenticity, authorisation (of the participating partners), and data encryption.
- Reputation: The reputation of a Web Service and its providers aims at the past experiences with a certain Web Service and its provider. This criterion also considers external references of the provider.

These QoS criteria have to be guaranteed by the particular service provider within a Service Level Agreement (SLA). The SLAs are handled by the SLA Management component. The SLA Management component analyses the SLAs and stores relevant information about guarantees in a database. The evaluation and assessment of the QoS criteria proposed above is performed by the Rating component. The Rating component creates a ranking based on the SLAs. For this, the Rating component evaluates the criteria defined within the SLAs. The non-measurable values like

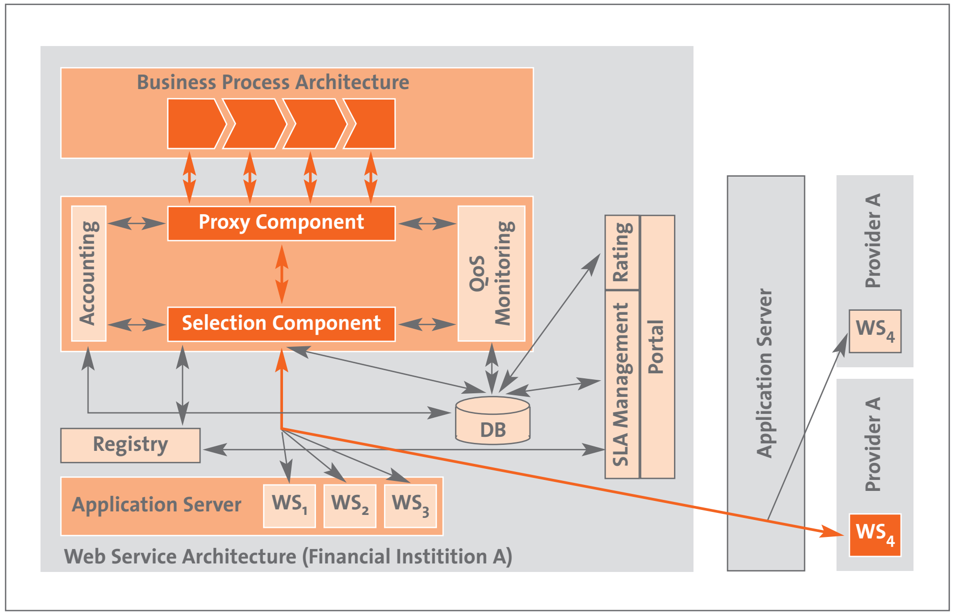

*Figure 3: Extended Service-oriented Architecture (SoA) based on Web Services technology [3]*

reputation and security have to be assessed by IT experts. Furthermore, IT experts can define rules to exclude Web Services that do not satisfy certain minimal QoS requirements. An example for such a rule can be: "Do not admit Web Services with a response time longer than 10 ms". Web Services passing this stage are integrated by the Rating component in the internal registry. This registry contains links to all Web Services that comply with the defined constraints. These Web Services can be made available by internal business units or external service providers. Internal Web Services are favoured to encapsulate the functionality of legacy systems. The Proxy component and the Selection component dynamically invoke Web Services considering QoS attributes of the registered ones. The Proxy component receives the Web Service invocations of the process and forwards them to the Selection component. The Selection component chooses the Web Service with the highest score calculated by the Rating component on basis of the SLAs. Then the Selection component executes the particular Web Service. When a Web Service is invoked by the Selection component, the accounting mechanism is started as well. The Accounting component tracks start and end time of a Web Service invocation as well as potentially occurring errors. Accounting is the process of tracing information systems activities to a responsible source and is usually conducted by the service provider as a foundation for charging and billing. However, the approach we propose enables accounting at the client side as well. This can be helpful to assign costs to internal business units according the cause of the costs. Additionally, the service requestor can make use of accounting information to check the provider's invoice. The QoS Monitoring component controls the compliance of the Web Service execution with the SLA. Thus, it evaluates the measurements conducted by the Accounting component. The measurements have to be compared to the guaranteed metrics within the SLA. If there are any violations of the SLA, the provider of the particular Web Service as well as the service requestor are notified. Furthermore the QoS Monitoring component can initiate the substitution of a bad-performing Web Service, whose availability broke down, by another Web Service with the same functionality. For this, a message is sent to the Selection component to terminate the bad-performing Web Service and to start another one.

For further details of our work we refer to [1][2][3][6].

## References

**Berbner, Rainer; Mauthe, Andreas; Steinmetz, Ralf:**
Unterstützung dynamischer E-Finance-Geschäftsprozesse. In: Horster, P. (Hrsg.): Elektronische Geschäftsprozesse 2004. Klagenfurt. Syssec Verlag 2004, S. 44-54. [1]

**Berbner, Rainer; Heckmann, Oliver; Steinmetz, Ralf:**
An Architecture for a QoS driven composition of Web Service based Workflows. In: Proceedings of the 8th International Conference on Electronic Commerce Research (ICECR-8). Tunis 2005. [2]

**Berbner, Rainer; Heckmann, Oliver; Mauthe, Andreas; Steinmetz, Ralf:**
Eine Dienstgüte unterstützende Web-Service-Architektur für flexible Geschäftsprozesse. Erscheint in: WIRTSCHAFTSINFORMATIK 47 (2005) 4. [3]

**Krafzig, Dirk; Banke, Karl; Slama, Dirk:**
Enterprise SOA. Best Practise. Prentice Hall, Upper Saddle River 2005. [4]

**Papazoglou, Mike P.:**
Service-Oriented Computing: Concepts, Characteristics and Directions. In: Proceedings of the 4th International Conference on Web Information Systems Engineering (WISE 2003). Rom 2003, S. 3-12. [5]

**Steinmetz, Ralf; Berbner, Rainer; Martinovic, Ivan:**
Web Services zur Unterstützung flexibler Geschäftsprozesse in der Finanzwirtschaft. In: Sokolovsky, Z.; Löschenkohl, S. (Hrsg.): Industrialisierung der Finanzwirtschaft. Gabler Wiesbaden 2004. [6]