Architectural Case:
The Swedish Tax Agency

(Skatteverket)

2004-10-08

Martin Henkel

SERVIAM-ARC-07

Version 1.0

Table of Contents

1 Introduction

This report gives a brief overview of how the Swedish Tax Agency (Skatteverket) uses, and intends to use, Web Services. The description is based on presentations held by Anders Eriksson, Håkan Westergren, Magnus Kling and Jonas Öholm at the agency 7 October 2004.

The tax agency is responsible for operating the Swedish tax system. This task includes collecting and controlling tax payments from industry as well as individuals in Sweden. The work means that the agency needs to support systems that provide information to individuals and businesses, as well as maintaining several large databases with tax information. Currently the agency's IT department is responsible for maintaining three types of externally accessible systems:

- Tax information systems (income tax, VAT and special taxes)
- Customer information systems (information about tax regulations, tax declarations and tax information about organizations). These information systems are available through the agency's web site, and through system APIs.
- The national registry of Swedish citizens ("Folkbokföringsregistret").

In total, the agency is responsible for several large databases, and provides access to these to individuals, other governmental agencies and commercial organizations.

The agency currently has a few web services running, but envisions that the future use of service oriented architecture and web services will be much greater.

1.1 The Need for Service Oriented Architecture at the Agency

Recently the agency has gone through changes that emphasize the need for a structured approach to electronic services and the use of Web service technologies. The new architecture must solve upcoming and existing needs for integration, flexibility and process-orientation:

Need for integration: External customers are increasingly demanding to get real-time access to the agencies systems. For example, individuals might require access to their tax-accounts, while other agencies want to synchronize their registers with the internal systems of the agency. The government's vision of "24-hour" on-line governmental agencies ("24 timmarsmyndigheten") also pushes the demand for cooperation between agencies.

Need of flexible architectures: This need stems from two sources: increased responsibilities and changed organizational structure. Through the years the agency responsibilities have increased, for example the responsibility to handle the registry of citizens and the handling of heritage taxes (arvsskatt) where transferred from other agencies to the tax agency. A change in how the agency is structured also requires a flexible architecture. The new way of working, with several specialized departments geographically distributed, requires that information can be readily shared and that work tasks can be shared and transferred between departments. These changes, and future, means that the agency needs to be able to quickly integrate new systems into their IT architecture. Also, the architecture must be flexible enough to cope with future re-organizations.

Demand for process orientation: The organization has an increasing interest for process orientation. In the future, this will require that the systems can support a process oriented way of working. This will require that the architecture enable the coordination/combination of existing and new services.

While the above stated needs stem from the desire to evolve the agency's work, the current IT architecture also poses some problems that need to be dealt with:

- Current systems are integrated in a point-to-point fashion. This means that there are a lot of custom-made system adapters.
- Copies of central registries exist in several systems. This design is due to that the system owners do not like dependencies to other systems. The design leads to problems when distributing registry updates.
- Some applications do not have a separate layer of business logic. For example, some old systems build with Oracle forms are very hard to integrate with other systems, since the business logic is tightly integrated with the user interface.

Both the problem with the current systems and the future needs have inspired the agency to develop a new, service oriented architecture. This architecture is briefly described in the next section.

1.2 Solution Overview

To cope with the problem and needs previously outlined, the agency intends to deploy an architecture based on a Message broker (see figure 1, below). While this architecture is based on the concept of an Enterprise Service Bus (ESB, Chappel 2003) it got some specific characteristics:

- The dominating message delivery style will be synchronous
- Workflow/process execution is provided by a separate service.

The message bus will probably be implemented with technologies that the agency already has, for example BEA Tuxedo and/or BEA Weblogic Server (WLS).

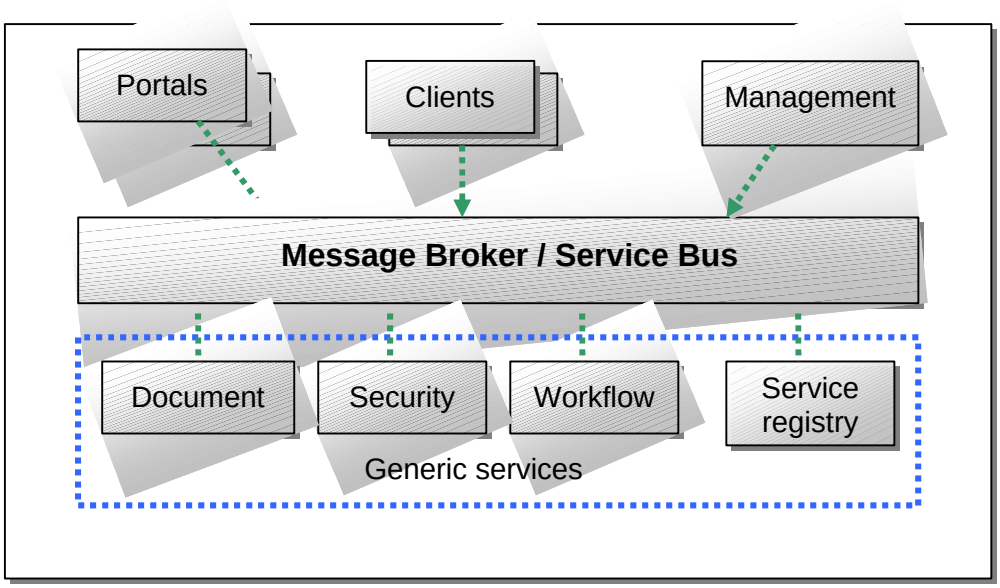

Figure 1, The planned message bus and its services.

Furthermore, the agency has identified two areas that need to be solved to enable this architecture:

Integration technology. A message-bus that integrate the systems. Adapters need to be developed for each type of systems.

Business applications. Partially new applications need to be developed, these applications will combine the services published on the message bus.

While the above described architecture isn't implemented yet, the Tax Agency has tried the web service technologies in a couple of applications. Two of the applications are described in the following sections.

1.2.1 Case 1: "Navet"

Business need: External organizations need to lookup information about citizens in the agency's registry. For example it is common that hospitals need to find parents and other relatives in case of an accident. Access to personal information of this type is limited to authorized governmental organisations. Signed agreements should regulate the content of the information that is returned on each request.

Solution: The agency constructed a SSL secured web service that provides a request-response search capability. An authorization check was set up such that each organisation supplies a client-side certificate. Authorization on the end-user level is done by the organization that requests the service. The certificate also provides the means to adjust the filtering of the response – the filtering can be applied on a per-client basis.

Special web service feature: SSL using both client and server-side certificates. Special certificates with organizational number (that identifies the client) was provided by Posten. A current problem is that Posten is about to stop issuing new certificates. Currently, negotiates with other Certificate Authorities are underway, with the goal that other CA:s also include organizational numbers in their certificates.

1.2.2 Case 2: "Kontakt- N"

Business need: New companies need to be registered at two governmental agencies, the Tax agency and the Swedish Companies Registration Office. A goal is to provide an easy registration process, and at the same time supply the systems at both agencies with information.

Solution: The registration office supplies a web site for company registration. By using web service technology, the web site was hooked up to the back-end systems of both agencies. Just as for the "Navet" solution, these web services use both client and server side certificates.

2 Service Design

To design its web services, the agency employs a model-driven design process. Figure 2, below, show the work process for developing a web service.

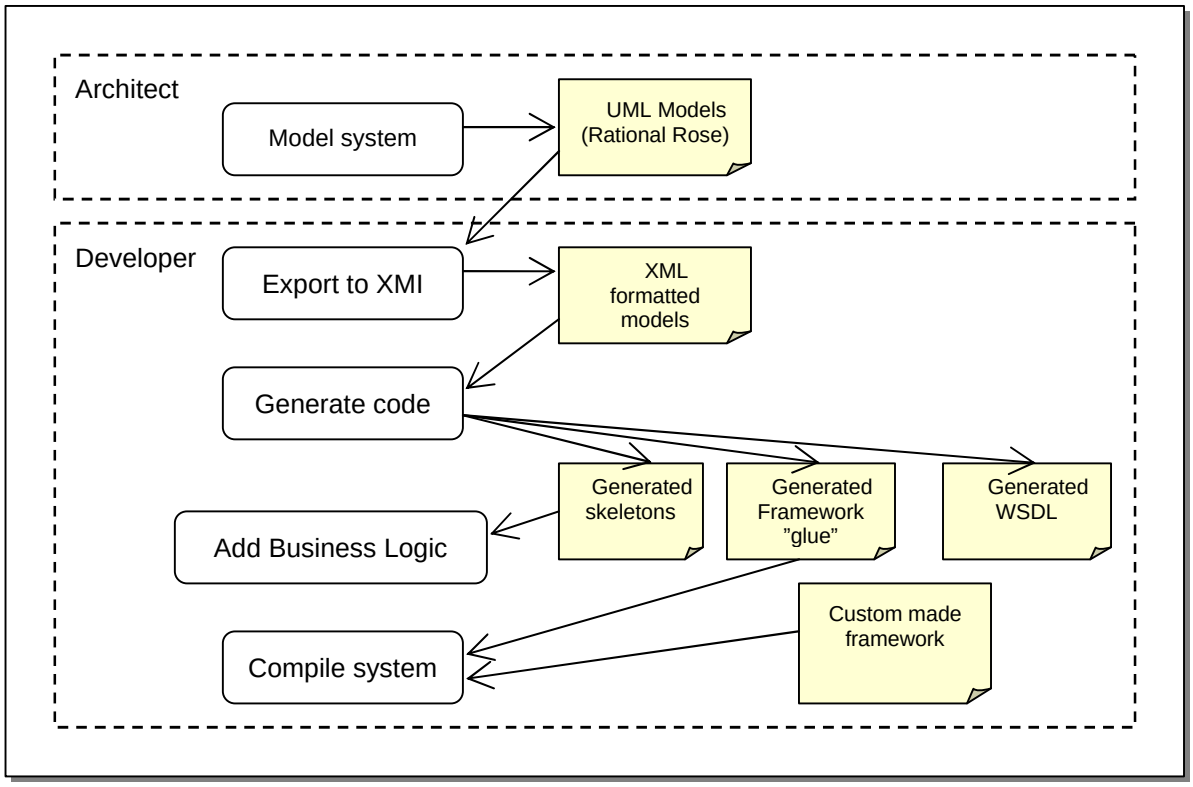

Figure 2, Model-driven process for web service creation (modified from Håkan Westergrens presentation)

The architect work in IBM Rational Rose Modeller to create UML diagrams of the intended service design. The tax agency uses their own UML stereotypes/symbols to define service interfaces and value objects. The model is then exported to XML Meta Data Interchange Format (XMI) files. These are used by the developer to generate skeletons for the web service implementation (java classes). The code generation tool also generates "glue" code that integrates the code with a custom made web service framework. The framework takes care of generic coding tasks such as security checks and logging. The code generation tool takes advantage of the Apache AXIS toolkit to generate WSDL.

A big advantage of this model-driven approach is that most of the "boiler-plate" code is generated, the developer can thus concentrate on adding the business logic. The use of the framework also imposes a uniform handling of security and logging.

3 Communication

See next chapter for a summary of the technologies used at the Tax Agency.

4 Web Service Technology

Layer	Standards	Usage
Service composition/ Process	BPEL4WS	Currently, InConcert is used to handle business workflow. However, other solutions to provide process choreography are examined.
Composable service assurance	WS-Transaction, WS-Coordination, WS-Reliable Messaging, WS- Security	Not used.
Description	WSDL, XSD, UDDI	Automtic generation of WSDL are done thought the Apache AXIS toolkit.
Messaging	SOAP, XML	SOAP, XML
Transports	HTTP, HTTPS, SMTP	HTTP, HTTPS/SSL

Table 1, Use of the web service stack

5 Success Factors

- Introduce service-orientation in the organization and in the systems in parallel.
- Continuously check that performance isn't degrading when integrating systems.
- Strive to remove batch oriented systems, as these do not cope with the demands on 24-hour service.
- Introduce general principles guiding who is responsible for user authentication on the individual and role level. If possible, let the "client" system authenticate on the individual level, the services can then authenticate on the system level.

6 References

Chappell, D., 2002, "Asynchronous Web Services and the Enterprise Service Bus," http://www.webservices.org/index.php/article/articleview/352/4/24/, Accessed 7 Jan 2003.