Service Oriented Architecture - Developing Applications for Automotive and Aerospace Industry

Dr.-/ng. Yasmina Bock, avanion GmbH, Berlin, Germany

.1Abstract

Distributed development processes in cross-company organizations require flexible and scalable applications providing robustness, platform independency and reusability. Service oriented architectures are suitable for the realisation of EAI projects. This paper describes the whole development process of an enterprise-wide application tracing the weights of assemblies and vehicles during the product life cycle from design to SOP and aftersales.

The business process was analysed and documented by use cases and activity diagrams. The task is defined to trace actual, target, calculated weights from CAD and costs and to define measures. Data modeling started for product structures, partly imported from legacy systems and represented by powertrain and body variants. Monitored and traced weight data enables the users company-wide to take cost-driven actions already in the early design phase. In order to provide a flexible and scalable system, the application was realized complying with the principles of SOA as a client-server application applying WebSphere, Oracle and standards as J2EE and XML.

.2Introduction

Service oriented architecture can be defined as a software architecture that defines the use of loosely coupled, independent services to support the requirements of the software users. The SOA environment replaces the monolithic software systems by enabling resources in a network not depending on the underlying platform implementation. SOA also aims at the support of complex business processes. SOA is often based on web service standards like SOAP, WSDL or also CORBA or also Enterprise Java Beans, but SOA should be regarded as technology independent.

In March 2006, the OMG SOA Special Interest Group adopted this definition for SOA [1]:

Service Oriented Architecture is an architectural style for a community of providers and consumers of services to achieve mutual value, that

- Allows participants in the communities to work together with minimal co--dependence or technology dependence
- Specifies the contracts to which organizations, people and technologies must adhere in order to participate in the community}
- Provides for business value and business processes to be realized by the community
- Allows a variety of technologies to used to facilitate interactions within the community.

Service oriented architectures aims at the combination and coupling of services of different abstraction levels to achieve high value functionality. The services can be defined as reusable artifacts, providing added value [2]. A service comprises the technical content representing the business process, which provides the software user an efficient support doing his job. The specification of the services can be mapped on the adequate technology, mostly applying UML. This description can be mapped on technologies as web services or CORBA. The technical content can be analysed by the application of tools like ARIS applying use cases, activity diagrams and sequences of business processes as well as the modelling of dependencies of data. Furthermore this technical content can be enhanced with roles and rights concepts as well as transaction mechanisms which are mapped on technical frameworks and protocols.

Therefore SOA is suitable for Enterprise Integration Application projects, mapping a platform independent model on a target platform like a web application server (Java EE platforms) and tools for data management. These aim at the encapsulation of persistent data to guarantee a high flexibility and to avoid redundancies. The high flexibility, modularity and the orientation towards the business processes represented by the service is achieved with a higher implementation effort of the business logic of the services in the beginning.

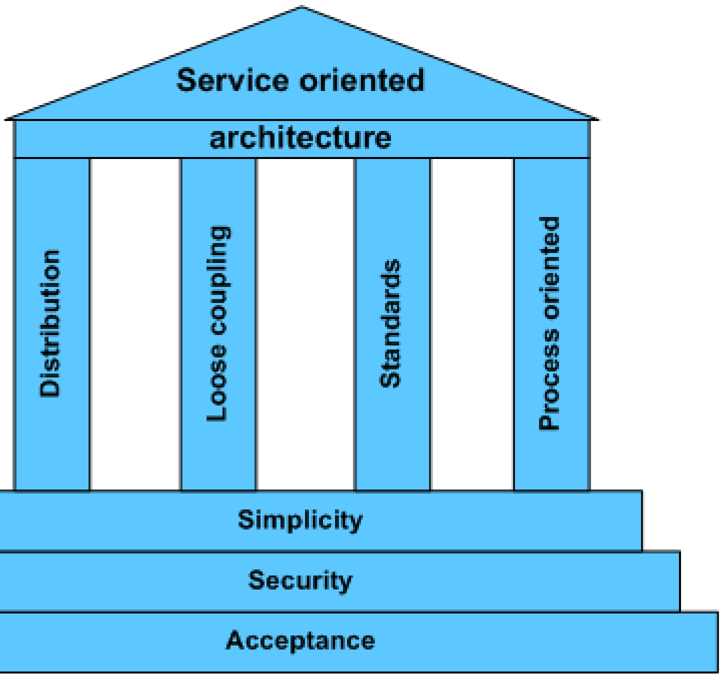

Fig. ..1.1 Principles of Service oriented architectures [3]

.3Building up the services

.3.1General requirements

When new applications are developed, today decisions are more influenced by the aspect of integration. Building a new software applying modern tools and methods is not enough, as there are lacks in the provision of a continuous data flow, data access for a highly linked user group with differentiated views on the information. Therefore the decisions for software architectures are influenced and driven by concerns like re-usability, serviceability and the degree of integration.

The analysis of the business process for weight tracing and monitoring showed, that it is not sufficient only to calculate different weights like actual weight, target weight or the overall weight of a vehicle like a passenger car, truck or airplane, much more information and data has to be managed. Beside of the weights, relevant information about the product like the product structure, various configurations of the product and the related information about the responsible contact persons in numerous development departments had to be taken into concern. Furthermore, some of this information is already available in third party systems and legacy systems like PDM systems and has to be imported. Another requirement for the availability of the data was the provision of defined interfaces. To enable this transparency, it is agreed to refer to standardized tools and methods.

.3.2Analysis of the business process

The business process was analysed applying ARIS. As a result, all relevant data, data flows, third party and legacy systems as well as the dependencies have been retrieved. Furthermore all relevant organisational information like contact persons, departments involved in the product development process have been detected and documented which will be later on reflected in a rights and rolls concept. The analysis of all stakeholders is extremely important to get an accepted overview and definition of all responsibilities. The product information comprises all weight data related to the product structure and single parts of the product. The product structure can be related to the organisational data, which is not limited to the departments of a brand, it can also comprise the whole company as well competitors.

A detailed description of the business process and the process steps depending on internal and external events is provided by use cases, activity diagrams and sequences as well as the data, structured in a data model. These descriptions serve also as functional requirements of the application and will be combined and enhanced with technical information like transaction behaviour, rights and rolls concepts, and constraints. Furthermore a detailed description of the graphical user interface was built up to achieve an optimized support of the working process and efficient support of the users.

.3.3Mapping the platform independent model

On the realization level the platform independent model can be transformed to the target platform (like Java EE or .NET). Figure 1 shows the general requirements on SOA: the architecture is based on acceptance, security and simplicity, its features are distribution, the loose coupling of services, the application of standards and its process orientation realized by the choreography and orchestration of several small services instead of one complex implementation. The triangle of relationships, typical for the components of SOAs, leads to questions which have to be answered for the selection of a platform [4]:

- Style of communication: instead of RPC, a notification driven, asynchronous communication can be chosen.
- Which protocols are available and supported?
- Which concepts for autorisation and authentification are supported?
- Is the security guaranteed?
- How common is the platform, is the life-cycle of the services supported?

Beside of OpenSource application server projects like JBoss or GlassFish, several integration platforms are already available like SAP's NetWeaver, Bea's WebLogic or as chosen in this case, the implementation was based on IBM WebSphere.

The lowest layer of a SOA component are the data services, the data base applied is Oracle 10i. The persistence and the O/R mapping of the data access objects is assured applying Hibernate as persistence framework. Referring to the architecture of a J2EE server, the business logic and all beans like session bean and message-driven-bean are part of the EJB container.

The client can access the EJB container either via the home interface and the JNDI-service or the remote interface. The GUIs of the client have been designed together with the end-user to support their work and process to a high extend.

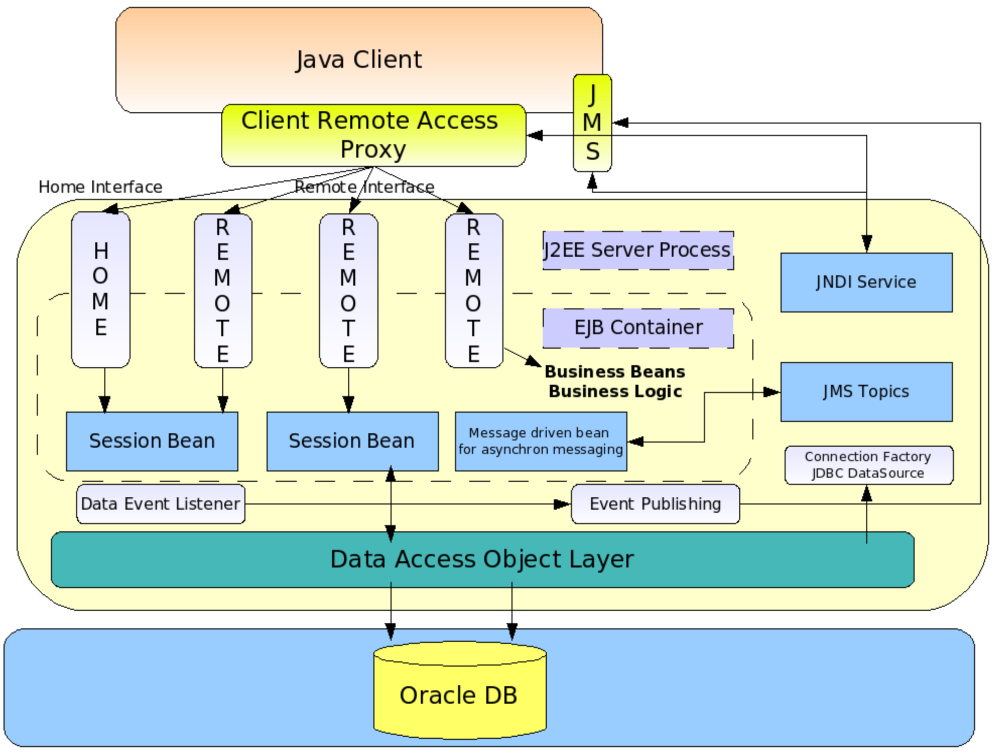

Fig. ..1.1 Architecture of the weight monitoring system in compliance with SOA principles

.3.3Implementation of the weight tracing service

The weight tracing application allows the differents users to create, edit and delete information concerning vehicle weights depending on their rights and rolls. Weights can be either retrieved from legacy systems or they can be edited by users according to their responsibilities and affiliation to an organizational unit. The origin of the weight information can be either the real weight of existing parts, a target weight which shall be achieved or a calculated weight,
which is provided by the CAD system.

The weight information can be calculated and compared for different vehicle variants as well as different powertrain variants. Vehicle variants can be coupe, sedan or station wagon for the automotive branch / application or different layouts of aircrafts like passenger, passenger and cargo combined or cargo for the aerospace application respectively. Like this, at an early stage of the design the influence of additional technical equipment on the overall weight and the centre of gravity can be examined. Furthermore, the controlling of the weight information allows to take actions at an early stage. These actions can be proposals to change positions or material of components or others to reduce the weight of parts and assemblies. The changes of material or even the position of parts and assemblies can implicate a change of costs for example due to a more complex and time consuming assembly. These costs have also to be taken into concern as early as possible.

The structured information of all brands within a company, the type series and all belonging vehicle variants allows also to take advantage of design solutions already made and to reduce the number of variants and to increase the number of parts already used in other vehicle projects. Experiences to solve problems within different type series can be transferred and parts can be exchanged between type series even if they belong to different brands. The structure is shown in figure 3.3.

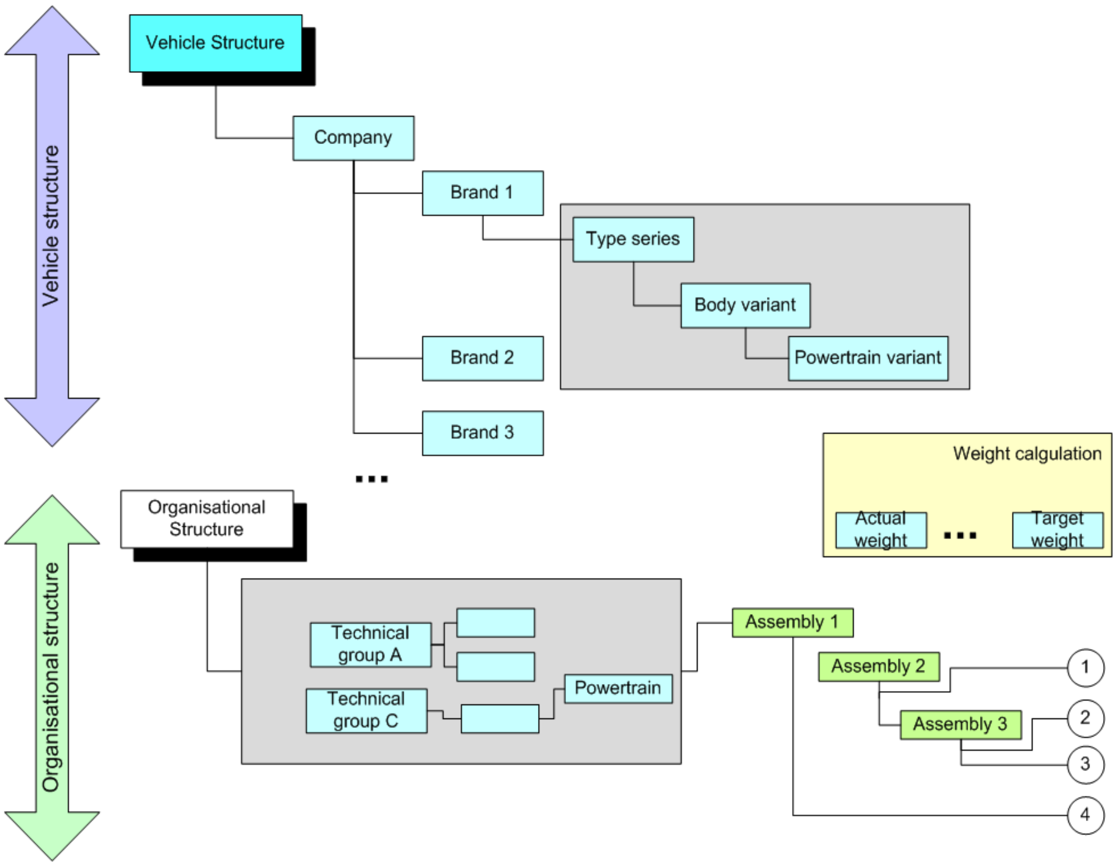

Fig. ..1.1 Structure of vehicle and organisational information

.4Conclusion

The paper shows the benefits of the application which has been realized for a service oriented architecture. The careful and profound examination and analysis of the business process leads to an agreed procedure for the work process, responsibilities, the implementation and acceptance of the software. The business process was transformed from an isolated, process into a shared service which is easy to maintain. Future integration of other services or legacy systems can be achieved at lower costs, avoiding the complexity of point-to-point integrations. All relevant data is now available for all users and contact persons of the development departments during the whole product development process. Actions in the development process can be taken at an early stage which will avoid tremendous costs for changes in the late design process.

References

[1] Rob High, Ed Cobb, Sanjay Patil, Greg Pavlik: The SOA Programming Model, SCA Collaboration Team, Session TS-3608, JavaOne Conference 2006
http://developers.sun.com/learning/javaoneonline/2006/coreenterprise/TS-3608.pdf

[2] Adam Bien: SOA Blueprints: Was bei einer serviceoritentierten Architektur alles zu beachten ist, Enterprise Architektur Magazin, S. 52-60, in JavaMagazin 11/2005

[3] Wolfgang Dostal, Mario Jeckle: Semantik, Odel einer Service-orientierten Architektur, S. 53-56, Java Spektrum 1/2004

[4] Bärbel Burkhard, Guido Laures: SOA - wertstiftendes Architektur-Paradigma, Web-document
http://www.softwarekompetenz.de/servlet/is/21805/burkhard_OS_06_03.pdf?command=downloadContent&filename=burkhard_OS_06_03.pdf