

# Service Orientation for a Dynamic Enterprise

Arsalan Minhas, Friedrich H. Vogt

Telematics Institute, Hamburg University of Technology, Germany  
[arsalan.minhas@tuhh.de](mailto:arsalan.minhas@tuhh.de) , [f.vogt@tuhh.de](mailto:f.vogt@tuhh.de)

## ABSTRACT

Business data is one of the most critical components of the IT portfolio of any enterprise. However enterprises are facing a major challenge of providing a holistic view of it when the data spans across multiple heterogeneous systems with different data models embedded in them. To achieve maximum benefits from the SOA, it is crucial to create a common data management approach that handles all aspects of integrating information exposed through different autonomous services in an enterprise. We present an approach by which it becomes possible to provide the holistic view of business data by actively driving simplicity and SOA project acceleration. It allows us to integrate the supply chain applications of an enterprise and hence building a competitively differentiated supply chain for the future.

**Keywords:** SOA, Information Management, Data Integration, Common Data Model, Database as Service, Data Services Layer, Shared Data Services.

## 1. INTRODUCTION

Investing heavily in supply chain applications and systems in the last decade, companies are still threatened by the challenges caused by the deployment of range of stand alone applications that work in isolation, incompatibilities between the platforms, proprietary technology constraints and error prone, expensive and time consuming custom integration. This has locked the enterprises into applications offering insufficient functionality and not offering enough flexibility to access intra- or inter-organization data. People are plugged into the rigid, brittle and inflexible processes. To streamline the businesses there is a real business need to have applications communicate with other applications existing both within and outside an organization. Often the enterprises need to have many processes serving different customers because of varying set of requirements of the customers. However current off the shelf components and applications don't provide enough room to manage this diverge portfolio of requirements. According to a recent survey about 75% of the companies are of the view that supply chain software limits the amount of services they can offer customers. Therefore companies are forced to employ workarounds around existing applications to satisfy the customer because only 18% of companies manage their supply chains with little or no modifications from the standard package and only 26% say their software allows them to quickly adapt to customer requirements [1]. This dictation of business processes flow by the workflow of the applications has resulted in a battle of trading off flexibility for lower Total Cost of Ownership (TCO) [1] which is not at all desirable.

We believe that Service Oriented Architecture (SOA) [2] provides a solution to this dilemma. We at Telematics Institute of Hamburg University of Technology are working on defining the roadmaps for SOA by modelling the real world business problems. In this paper we analysed the business processes of Germanischer Lloyd (GL) AG, a global player and the market leader in the shipping industry. However our proposed solution is not just limited to the shipping industry but constitutes a generic approach of Service Oriented Modelling for any dynamic and extended enterprise.

The rest of the paper is organized as follows. Section 2 describes the challenge of information management in an enterprise by highlighting the specifics of GL, motivation for SOA is created in section 3 and then section 4 delineates how SOA can emerge as a life saving scuba-diver for the information management challenge. Finally we will draw conclusions bases upon our findings and observations in section 5.

## 2. MANAGING ENTERPRISE INFORMATION CHALLENGE

Enterprises with multiple lines of business spread over multiple geographies have critical data stored in multiple, scattered databases. In many cases, the scatter of core data is proportional to the size of the enterprise IT portfolio. Such companies may have heterogeneous data environments with varied schemas and they may contain redundant data elements. This data may be static reference data, common business data, or common external data. This can lead to serious inefficiencies and consequently higher costs because of the overhead in accessing data in multiple databases using different mechanisms. These issues lead to an incomplete view of core business data that can cause inconsistent user experience and the introduction of risk. Apart from this, interoperability between business lines is often difficult and error-prone. Diverse, distributed sources of information pose many challenges for creating and maintaining applications by integrating and aggregating data from these sources across an enterprise.

GL has multiple lines of businesses that are represented by different applications running across their network. They use *Technical Reporting and Ordering Network (TRON) New Building* for surveying which is an IT tool that considerably speeds up and simplifies the entire surveying process by making increased use of the intranet. It is very important for a chartered ship to spend as little time as possible in port if it is to operate profitably. In the cut-throat competitive climate that exists in the industry it is often a case of "every minute counts". Speed is one of TRON's greatest strengths. TRON replaces complicated, time-consuming, paper based correspondence with a largely electronically based surveying process linked to the GL intranet, a process that even includes electronic transfer of the relevant data to speed up invoicing. TRON uses a safe internet connection for two-way transmission of survey data between surveyor on the ship and GL head office. Every surveyor is provided with the data necessary for survey preparation from the online central database. Surveyor is able to send the completed survey documentation back to the head office. Preparing and handling the survey statement has become faster due to this process reengineering. TRON offers significant improvements over old surveying process because different technical reports which are in use up to now (e.g. F100, F400, etc.) have been brought together in a single Survey Statement that focuses on the demands of the kind of vessel to be surveyed [7]. Now, the surveyor on the spot has to work his way through a catalogue that is tailored to the specific scope of the survey in question and takes into account all the relevant survey parameters. It is from this that the technical report is automatically drawn up. This significant simplification of the survey process brings important benefits to all concerned: for the customer the survey process is more transparent and thus easier to follow; for the surveyor it means faster processing and a reduction in double entries. Data is stored in a huge Oracle 9i based RDBMS repository.

*TRON Materials and Components* is currently being developed to manage the information for discrete components and different types of materials that are used in a ship. This information is later on used to generate certificates. Earlier the administrative process of issuing and handling of certificates was very laborious as the surveyor used to compile the blotter certificates by hand and inspection office using MS Word. After signing the certificates by various parties, they were printed and sent to the customer and a copy to the head office. At head office the certificates were registered, scanned and filed electronically. Since the certificates and the basis of the materials and components data were not filed as structured data, therefore the recorded data for the certification of materials and components was not available for the processes of *TRON New Building* and other systems. This motivated GL to plan for *TRON Materials and Components* which will be a structured data model storing information for raw materials, discrete components and their assemblies in a hierarchical manner. It will make the certification process efficient and the data will also be available to *TRON New Building* and other running processes. *TRON New Building* should be able to fetch this certification data of materials and components via certificate number which will act as a link between the two systems. However its realization is an issue as both the systems will have their own repositories, independent data models and possibly different technology infrastructures. Moreover, during the analysis we also observed that there is a lot of common data in them which is causing extra storage and synchronization issues.

Finally *Technical Information System (TIS)* is a Product Data Management (PDM) system that stores the engineering data related to a ship. It models ship and its parts in terms of project, system, component, part, part instance, material, property sets, functional relationships etc. This generic modelling approach allows it to model any kind of ship and any part irrespective of its hierarchy. However mostly they use this system to store engineering data of very granular components and materials in a ship like screws, bolts etc. Moreover the unique aspect of this system is its ability to provide multiple views of a same object. For instance, property set of a pump or its functional relationships with other objects in a ship vary significantly if its location is changed from engine to toilet. Dynamic links are used to serve different queries from different angles on the same part. It uses SQL Server as an underlying repository modelling an Object-Oriented data base structure. Despite its innovation,

this system is unable to make use of existing data residing in *TRON New Building* and *TRON Materials & Components* and hence relies on its own data set.

For GL, dynamic business needs and the push from competition have created drivers for cross-business integration and the desire to provide users with an integrated experience. This in turn has led to increased demands for the availability of core data from other lines of business and the need for the data to be in synchronization. Using custom synchronization routines by programming against the available APIs for each of the disparate data sources is not recommendable as they are costly, unstable and difficult to maintain. Apart from this, these types of solutions are not scalable given the expanding nature of business, the nature of different business units, and the technology platform that may be in use in multiple lines of businesses. In addition to the Web applications, note that there is a team of dedicated back-office people who directly update databases through stored procedures or SQL. These update channels present an additional challenge to the integration of data and the creation of consistent views of core corporate data. To summarize the key challenges of this problem domain:

- Critical business data is located in multiple repositories
- Data has multiple channels of update
- Data needs to be synchronized between repositories
- Needs and usage patterns of data are diverse across systems
- Different lines of business have different technologies, hence it is difficult to create a common data dissemination strategy
- No common data model exists for the common data existing in multiple systems
- In the long term, it is more beneficial to move away from independent data models from the cost and agility perspective

Based on this problem statement, let's examine how SOA can tackle these problems.

### **3. MOTIVATION FOR SOA**

SOA is a set of best practices for the organization and use of IT. Its foundation rest on the flexible business processes management, standard and cheaper integration infrastructure. It allows applications to support business workflows and not the case where business flow is being constrained by the technology as SOA is designed to support, not dictate, business processes. Instead of plugging people into the technology, technology is plugged into people day to day lives. It requires business users and IT staff to work hand in hand to evolve an architecture that can accomplish business goals and satisfy the needs of customers with a high satisfaction rate. IT resources are available on demand to business users as services which business users can compose on the fly to create applications. Lines of business should work with metadata without direct IT involvement and IT provides and implements the services satisfying that meta-data.

Instead of building fatty and monolithic applications that do not talk with others and are not flexible enough to be integrated with others, SOA recommends to build business services offering a well defined functionality. Business users can then select the desired services and assemble them in an appropriate order to create a process fulfilling the needs of a customer. The work flow can be tailored by changing the order of services, adding or removing services to comply with the needs of a different customer.

SOA allows application and information to be fetched on demand from the provider's library of business services and combined at will to create a composite business process. It removes the need of building stand alone, gigantic and tightly coupled applications and allows composing loosely coupled applications based on autonomous services by extracting slices of functionalities from different vendor's libraries. This allows supply chain data and software to be accessed directly, real time, from any where and whenever required.

Legacy assets play a key part in Service Oriented composite applications as existing mainframe and legacy applications can be exposed as services in SOA world and combined with new composite business logic. This avoids stitching a new solution by abandoning investments in existing systems but promoting weaved together processes, applications and data by reusing existing implementations in a timely and efficient manner.

SOA also allows reaching partners and suppliers businesses by sharing or using others' provided services and thus achieving a true B2B collaboration. In order to create a competitively differentiated supply chain, 97% of the companies rated integration of internal supply chain applications, 95% organizations desired the ability to reconfigure workflows to meet market demands, 88% wanted the real time integration of data between partners

and 90% liked to share the application functionality with business partners as most important technology capabilities [1]. Implementing SOA can give life to all these desirables to develop a competitively differentiated supply chain.

#### 4. THE SOA WAY

In order to provide a cost effective, reusable, flexible and future proof solution addressing all the key challenges, we based our modelling approach on the best practises laid down by Service Orientation. We can divide our methodology in three steps as depicted in fig.1.

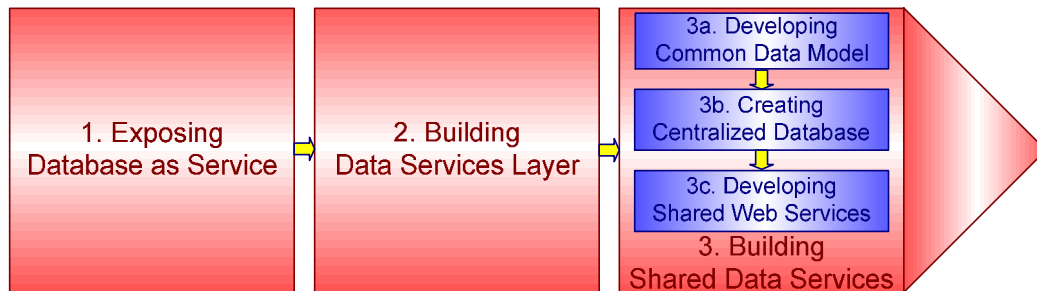


Fig. 1. SOA for Information Management

##### 4.1 Step 1 - Exposing Database as Service

Web Services provide access to remote content and application functionality using industry standard mechanisms and without any dependency on the service implementation, data format, service provider's platform or the location. Web Services are also a preferred way to implement SOA as we already discussed in [3].

The proliferation of structured and unstructured data and data logic in databases, the increasing momentum of XML as a data exchange format, and the de facto acceptance of HTTP as a ubiquitous transport mechanism in the context of heterogeneous environments has aroused interest in database Web Services which make sense when applications have significant amounts of business logic that run in the databases. It enables to share data and metadata across corporate intranets and access the database operations via SOAP [4] requests.

Database Web Services should be preferred instead of middle tier Web Services when data centric logic stored in databases or stored procedures needs to be extracted instead of the business logic running in the middle tier. This separation of concerns is more cost effective, cleaner, efficient, allows reuse and localizes the effect of schema change.

In order to integrate multiple databases with different technology and platform infrastructures, existing databases should be able to act both as service provider and service consumer as shown in fig 2.

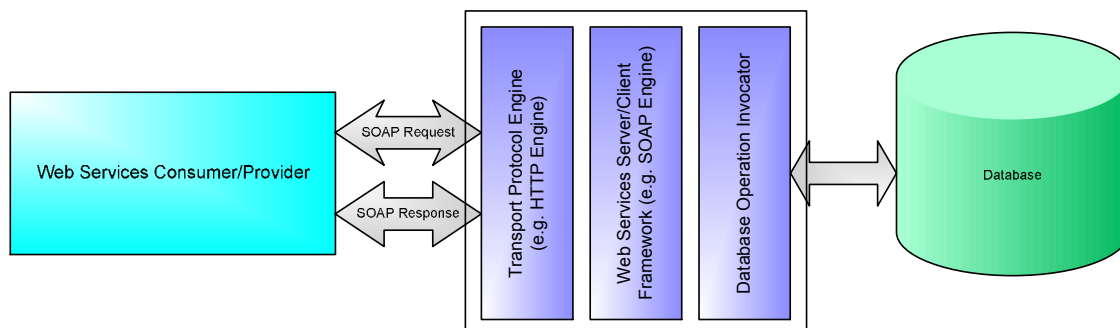


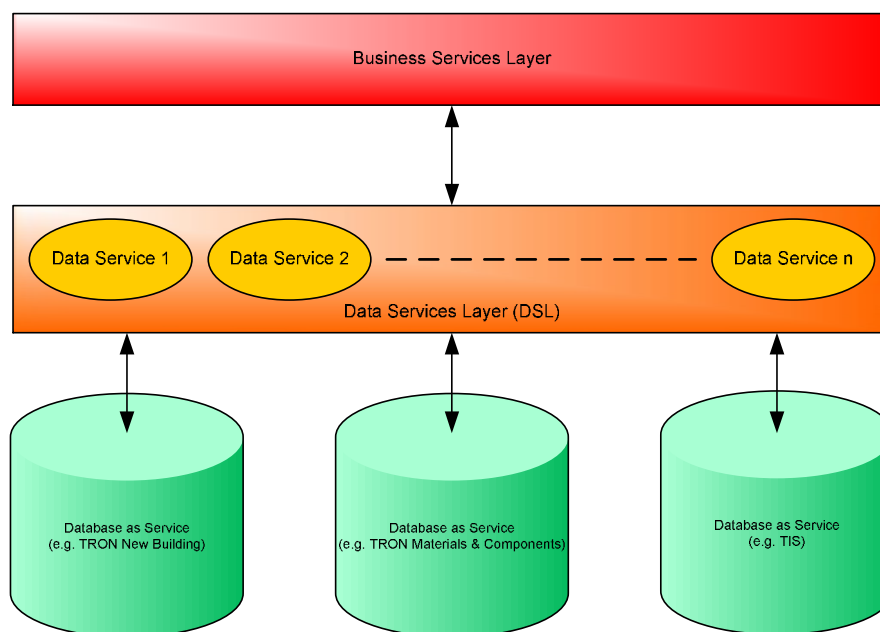
Fig.2. Database as Service

Database being a service provider allows clients to access the database via Web Services mechanisms (SOAP, WSDL [5], UDDI [6]) while database as service consumer allows database to consume an external Web Service by building the SOAP message based on the Web Service endpoint; sending the SOAP message over HTTP to the Web Service endpoint; receiving the SOAP response; extracting resultant data from the SOAP message body,

and dealing with faults and exceptions. In our case, it will allow *TRON New Building* to fetch information of certification data from *TRON Materials and Components* using the certification number as a link irrespective of the heterogeneous landscape. Moreover wrapping the databases as Web Services will reuse the investments in existing systems and avoid the costs, time and unpredictable return on investment (ROI) associated with the new solution. It is also flexible enough to allow plugging in more services in the future that can push/pull information to/from the existing services.

#### 4.2 Step 2 – Building Data Services Layer

However just exposing databases as services is not a solution because the application/developer needs to be aware of multiple heterogeneous APIs of all the data sources as there is no uniform interface. Moreover, when each application contains an independent and unique data access layer, each will require its own specific tuning and corresponding maintenance. Expanding business needs are also hindered by the repetition of same work as many applications do the same thing but only slightly different. So there is a need to have a layer that radically simplifies access and integration of distributed data by providing a level of abstraction which makes disparate data sources transparent to end users and provides an easy way to compose additional services. We call it as a Data Services Layer (DSL) shown in fig. 3.



**Fig.3.** Data Services Layer (DSL)

The DSL acts as the abstract layer that talks with underlying resources and takes away the data location, type, and administration from the application, leaving behind a virtual data source. To an application developer, a virtual data source means a focus on the data problem at hand, forgoing the need to write the plumbing to access heterogeneous data. One of the motivations behind SOA is a loosely coupled system and data layering provides this at the data realm. DSL provides centralized control on data which makes management of data easy and simplifies the data access as a uniform interface now replaces multiple heterogeneous APIs that would otherwise be needed. Moreover DSL becomes an interface that is easily reused by other applications or developers.

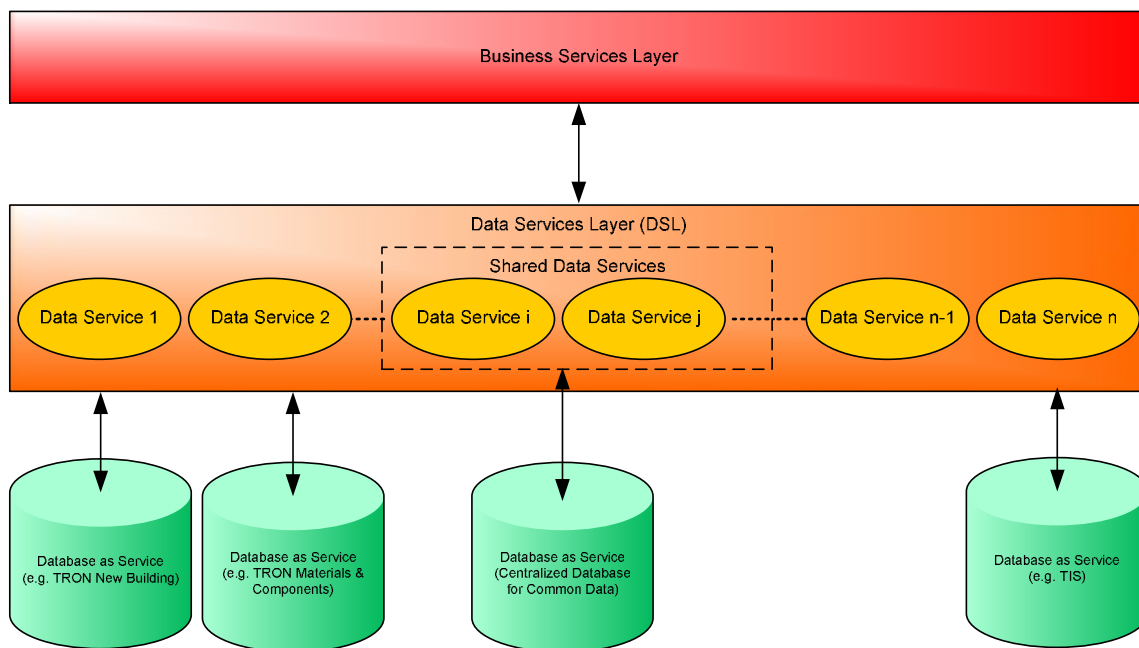
DSL is logically segmented into different segments known as Data Services that can be accessed through a standard API. Data Service exposes public functions that consumers may be interested in and relieve users from underlying details. Data service is a self-contained module as it contains: meta-data about the data source, connection information, interfacing functions for reading and updating data sources, relation information among data services, security information and meta-data about the data service itself. This data service contains enough information to look up, execute, and transform information in useful formats for the client applications.

However we like to warn that that Data Services should not be developed to act as a generic data pipes like ODBC or JDBC because of difficulty in maintenance which can result when underlying data model changes and thus affecting multiple data consumers; data tempering that can occur when databases are opened to many people who may not know the data models; and unpredictable queries because of difficulties in determining the usage patterns. The solution is to pre-package queries and expose packages as services. These services will use

the data model correctly and completely predictable and interface will remain stable most of the times when the underlying data model changes for some reason. There are no issues with the developers because they won't have to learn the data model to get what they need. Looking at this concept closely, it mimics application integration where data shielded by business logic is exposed through a standard interface. Here data services are mini-applications, created by data owners to make sure that data gets used correctly. Hence data integration requires a veneer of application integration.

### 4.3 Step 3 – Building Shared Data Services

We are not yet done with the architecture as the common data spanning across multiple systems is posing threats like extra storage, inconsistent views and synchronization issues. We believe that this problem can be handled by developing shared data services as represented in fig. 4. The principle behind shared data services is the consolidation of common data along with the development of interfaces for data dissemination using open standards. We will present the steps required to achieve this goal in the following sub-sections.



**Fig. 4.** Shared Data Services

#### 4.3.1. Developing Common Data Model

The business and IT people have to work in collaboration to derive the rationalize schemas for common data spreading across various databases. However it can be very challenging from the organizational perspective and much more so than from the technical standpoint. We are currently working on defining the common data structure which is acceptable to both technology and business stakeholders. Plan is to use an XML containing metadata and appropriate namespaces to look at the common data from different views of multiple lines of business.

#### 4.3.2. Creating Centralized Database

A centralized database that maintains data common to multiple lines of businesses has to be developed. This database has to be created by analyzing schemas and formats of data required considering current and future business needs. This database becomes the system of record and the owner for the common data. However this should not affect the line of business data that continues to reside in existing databases.

#### 4.3.3. Developing Shared Web Services

Developing a set of shared Web Services that manage data access/update for all databases. These services manage the relationships between the database and related systems and become the de facto access layer for all the applications. These services are the only channel for read/update of all data for all business applications. Designing of service contracts should be based on the needs of client applications. Moreover, performance can be improved by caching frequently requested data in the middle tier which alleviates the bottleneck that can easily occur when multiple applications access common data simultaneously. More requests can be satisfied without querying the database which increases scalability and reliability.

## 5. CONCLUSIONS

SOA will, over time, revolutionize how supply chain management applications are bought, deployed, and managed. It removes the constrictions where business processes are dictated by the applications. With the introduction of SOA strategy, the processes these applications support will become much more agile due to improved integration techniques and more flexible business processes. Our presented approach backed up by the case study of Germanischer Lloyd AG shows how enterprises can address the challenge of information management by building a robust and balanced SOA, enabling information and business integration and avoiding problems such as isolated data silos, data inconsistencies, and untapped information assets.

## 6. REFERENCES

- [1] Aberdeen Group: *The Service-Oriented Architecture in the Supply Chain Benchmark Report*, September 2005.
- [2] Don Box: *A Guide to Developing and Running Connected Systems with Indigo*. Published in MSDN, January 2004, available at: <http://msdn.microsoft.com/msdnmag/issues/04/01/Indigo/default.aspx>
- [3] Arsalan Minhas, Friedrich H. Vogt: *Using Service Orientation to Drive Business Processes*. Proceedings of IEEE International Multi-topic Conference 2005 (INMIC'05), Karachi/Pakistan, December 23-25, 2005. <http://khi.nu.edu.pk/inmic2005/>
- [4] Mitra, N. (2003, June 24). *SOAP version 1.2. part0: Primer, technical report*, World Wide Web Consortium (W3C), available at: <http://www.w3.org/TR/2003/REC-soap12-part0-20030624/>
- [5] Christensen, E., F. Curbera, G. Meredith and S. Weerawarana: *Web Services Description Language (WSDL) 1.1 (2001)*, W3C Note, available at: <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>.
- [6] Karsten Januszewski: *The Importance of Metadata: Reification, Categorization, and UDDI*, MSDN Library, September 2002, available at: <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnuddi/html/impmetadata.asp>
- [7] *TRON New Building description*, retrieved on November 18, 2005 from <http://www.gl-group.com/>