A Service-oriented Architecture for Business Intelligence

Liya Wu[1], Gilad Barash[1], Claudio Bartolini[2]
[1]HP Software
[2]HP Laboratories
{name.surname@hp.com}

Abstract

Business intelligence is a business management term used to describe applications and technologies which are used to gather, provide access to and analyze data and information about the organization, to help make better business decisions. In other words, the purpose of business intelligence is to provide actionable insight.

Business intelligence technologies include traditional data warehousing technologies such as reporting, ad-hoc querying, online analytical processing (OLAP). More advanced business intelligence tools – such as HP Openview DecisionCenter - also include data-mining, predictive analysis using rule-based simulations, web services and advanced visualization capabilities.

In this paper we describe a service-oriented architecture for business intelligence that makes possible a seamless integration of technologies into a coherent business intelligence environment, thus enabling simplified data delivery and low-latency analytics. We compare our service-oriented approach with traditional BI architectures, illustrate the advantages of the service oriented paradigm and share our experience and the lessons learned in architecting and implementing the framework.

1. Introduction

Business intelligence (BI, [1, 2]) is a business management term used to describe applications and technologies which are used to gather, provide access to and analyze data and information about an enterprise, in order to help them make better informed business decisions.

Business intelligence technologies include traditional data warehousing technologies such as reporting, ad-hoc querying, online analytical processing (OLAP). More advanced business intelligence tools – such as HP Openview DecisionCenter - also include data-mining, predictive analysis using rule-based simulations, web services and advanced visualization capabilities.

Traditionally, BI systems have been architected with focus on the back-end, which is usually powered by technologies for data warehousing. Lately, architectures for business intelligence have evolved towards distributed multi-tier enterprise analytic applications.

In this paper we describe a service-oriented architecture for business intelligence that makes possible a seamless integration of technologies into a coherent business intelligence environment, thus enabling simplified data delivery and low-latency analytics. We compare our service-oriented approach with traditional business oriented architectures, illustrate the advantages of the service oriented paradigm and share our experience and the lessons learned in architecting and implementing the framework.

The remainder of this paper is structured as follows. In section 2 we discuss the architecting principles for a solution for business intelligence. In section 3 we introduce our case study of a legacy application for assessing the performance of the IT support management process. In section 4 we re-architect the legacy system according to service-oriented principles, and describe an implementation based on it in section 5. In section 6 we validate our approach by comparing ease of design, impact and cost for a few common architecture use cases, and we discuss and conclude in section 7.

2. Architecting principles for business intelligence

The principal objectives of business intelligence can be summed up as follows:

- To provide a "single version of the truth" across an entire organization.
- To provide a simplified system implementation, deployment and administration
- To deliver strategic, tactical and operational knowledge and actionable insight.

Because of the focus on information in business intelligence applications, the privileged point of view of the supporting architecture has to be the *information view* [3]. From this point of view, the most popular paradigms [4, 5, 6] are:

- The *hub-and-spoke* architecture with centralized data warehouse and dependant data marts
- The *data-mart bus* architecture with linked conformed dimensional data marts
- Independent non-integrated data marts

Figure 1 represents a typical layered view of architecture for business intelligence.

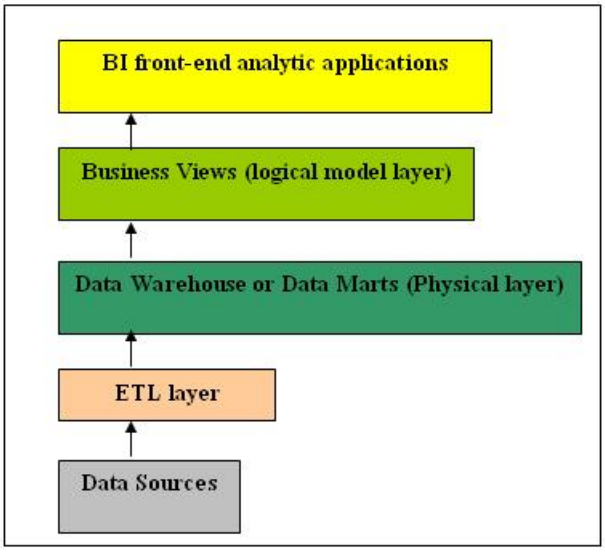

Figure 1: conceptual architecture for business intelligence

In today's heterogeneous environments where many disparate systems and domains hold different parts of the necessary data, the most difficult challenges in achieving the above mentioned objectives are effective information delivery and technology integration.

Effective information delivery

BI systems need to deliver the right information to the right consumers at right time. Since the source information can potentially come from many different and non-integrated sources, data has to be processed before it is effectively delivered to the end user. There are many data flows existing in any business intelligence system. The variety of data flow paths is described in figure 2:

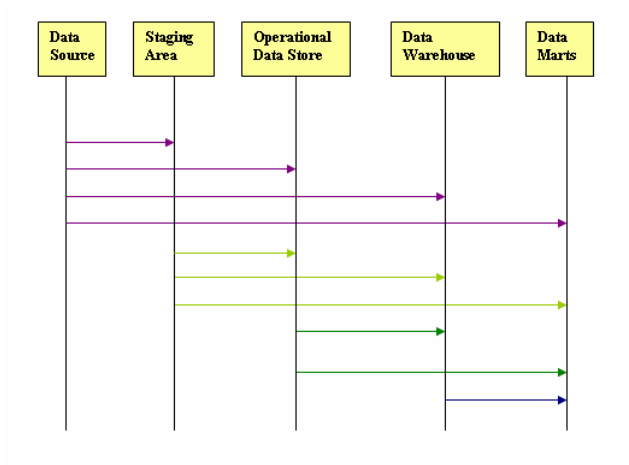

Figure 2: Data Flows for BI systems

Technology integration

In most deployed business intelligence environments, multiple "stovepipe" business intelligence systems - each with their tools, processes and data architectures – can be found across multiple business units and divisions of the enterprise. These non-integrated BI systems (whether built in-house or acquired) result in high component redundancy, inconsistent knowledge information, proprietary, non-open standard integration interfaces, and highly maintained point-to-point integration that ultimately increase the cost of development and prevent the achievement of a single version of truth across the organization.

For business intelligence to deliver on its promises of real time, zero latency information delivery and closed-loop processing, technologies and techniques have emerged or have been introduced. One such evolution is the transformation of traditional BI architectures into service oriented, component-based ones. Moving from our belief that Service Oriented Architecture (SOA, [7]) technology has great potential for delivering enhanced BI, we present an approach to architecting BI systems using a service oriented approach.

3. Case study: a business intelligence system for assessing IT service management performance

The case study that we consider is a business intelligence system for assessing the performance of an IT service management organization. In this section we describe the legacy solution and we take a fresh look at it to see how it can benefit from re-architecting it following service oriented principles.

Our legacy system provides historic views of IT performance, allocates the process bottlenecks, and delivers the information for IT performance improvement for IT organizations. It extracts service management and asset management performance data from multiple data sources. It has its own proprietary *extract, transform and load* (ETL) tool. Independent data marts stored in the *reporting data store* (RDS) are delivered and used for front-end BI reporting tool. The standard BI tool, which is integrated with the application, provides online reporting, ad-hoc querying, and online analytic processing (OLAP) capabilities. BI Portal system architecture diagram is illustrated in figure 3.

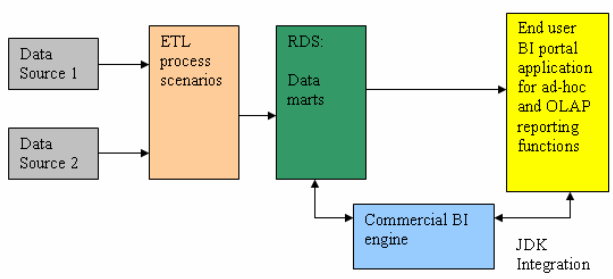

Figure 3: Legacy BI Portal System for IT service management

The shortcoming of the legacy systems that we are addressing are:

1. Independent non-integrated data marts for separate data sources
2. There is no default solution to provide actionable insight across multiple domains from service management and asset management
3. Inflexible, non-open, proprietary ETL processes which will be hard to be reused as sharable, loosely-coupled, service oriented pluggable ETL components
4. Front-end functions are implemented with tight integration with one specific BI tool. It will be hard to provide agnostic BI portal to be BI tool neutral
5. There is no simple closed-loop process workflow to provide the actionable insight back to the source systems

In the next section we will see how re-architecting the business intelligence systems using service-oriented architecting principles help us overcome these limitation, as was argued in [8, 9, 10].

4. Our solution

The first step in re-architecting our legacy system is to break down the legacy components into service-oriented reusable components able to communicate through open standard messaging protocols, based on XML, WS-* and SOAP. The resulting service-oriented architecture of our IT performance management system (SOA-ITPA) is described by figure 4.

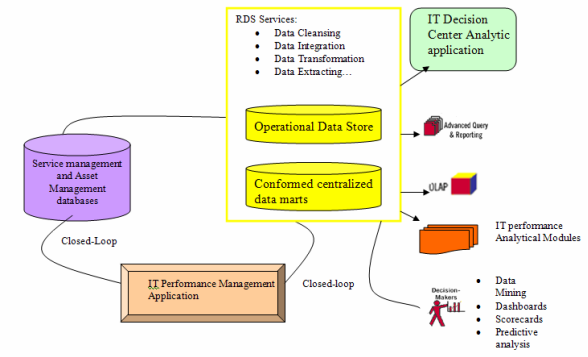

Figure 4: IT performance management system

The key benefits our SOA-ITPA architecture are:

- Integrated and consistent "single version of truth" data architecture
- Scalable and flexible ETL processes
- Reusable and extensible services, providing acceptable return on investment
- Actionable insight BI solutions to send BI analytical results to users and help them to understand the information so the appropriate actions can be taken in BI real time environment

The architecture is described to the next level of detail in figure 5. Its principal component services are explained in detail in the following sections.

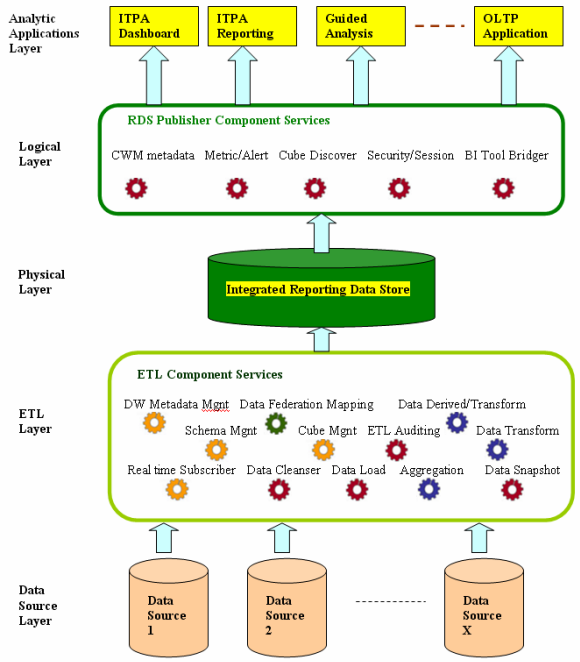

Figure 5: Service Oriented Architecture for IT Performance Analytic (SOA-ITPA)

4.1. Integrated reporting data store

The essential part of SOA-ITPA is the integrated centralized integrated reporting data store (RDS). RDS consists of historic, current and predictive data. A number of component services are built to populate data from sources into RDS. To achieve collaboration and closed-loop processing, a real-time subscriber service needs to be built to allow real-time synchronization. Other derived services, (snapshot service, transmit service and extract service) are used to ensure consistency of information providing a deeper insight in the organization performance. The persistent RDS cannot generally store all the (cleansed) data to be shared across the various components, including the ones supporting auditing and tracking activities. Trade-offs need to be made, depending on how much real-time data is required by the applications, and how complicate are the ETL transformations. However, the principle is to have full, rich, standardized, well-tested and certified ETL services to use for different projects and products.

4.2. SOA enabled BI component services

The tightly-coupled legacy ETL system – with dedicated ETL procedures for each component - is not reusable and maintainable. The SOA version that we propose breaks down the ETL process into generic smaller servicing modules. The design is parameter-driven, and XML metadata are driven around to provide specific functions within ETL service layers. The benefits of this design will be felt more in full when the system grows and becomes more complex, and to meet additional integrated applications' data requirements, as we argue in the validation section of this paper. The reusability of these services also provides more flexible and scalable ETL process.

The implementation of BI analytical modules for dashboard and reporting, performance management guided analysis application or other decision making applications is made easier and quicker by the use of the publisher-subscriber communication paradigm. The RDS contents are published and made accessible to other products and components through the standard web services interfaces and protocols, to make RDS fully functioning in any SOA based organization environments.

4.3. Analytical application solutions

In the SOA-ITPA architecture, the data marts are engineered to meet different applications requirements. They can be in different formats: XML, micro-cubes or Excel files. This also enforces that any data mart implemented as part of the IT performance management system will source its data from the federated, conformed and centralized RDS, tailored to provide the specific IT performance management solutions with data quality standards.

The main purpose of a conformed data mart can be defined as follows:

- To store pre-aggregated, multi-dimensional information
- To control end user access to information
- To provide fast access to information for specific analytical needs or user group
- To represent the end users view and data interface of the data warehouse
- To create the multidimensional/relational view of the data

5. Implementation

The SOA-ITPA system was implemented in multiple phases with agile development principle. The focus of the first phase was the centralized reporting data store (RDS) schema management, to support metadata management of the data warehouse, creation and customization of the RDS schema. Building on the first phase, the second phase concentrates on extract, transform and load (ETL) services, implementing a number of services to do with data cleansing, mapping, loading and corresponding auditing services. Following on that, the third phase sees the implementation of RDS publish services, where data from the RDS can be made available to be consumed as analytical information by external systems. The focus of the fourth phase is on the measurements of IT key performance indicators from the information made available in the previous phase and the presentation of such measurements through scorecards that allow a guided analysis of IT performance. Finally in a fifth phase a fully fledged business intelligent solution was implemented that orchestrated the ETL services and processes to provide data marts for service management, asset management and other aspect of IT service management performance.

An example of implementation of a common service (cube management) is given in detail in the next subsection

6. SOA-ITPA service sample implementation: CubeManagementWS

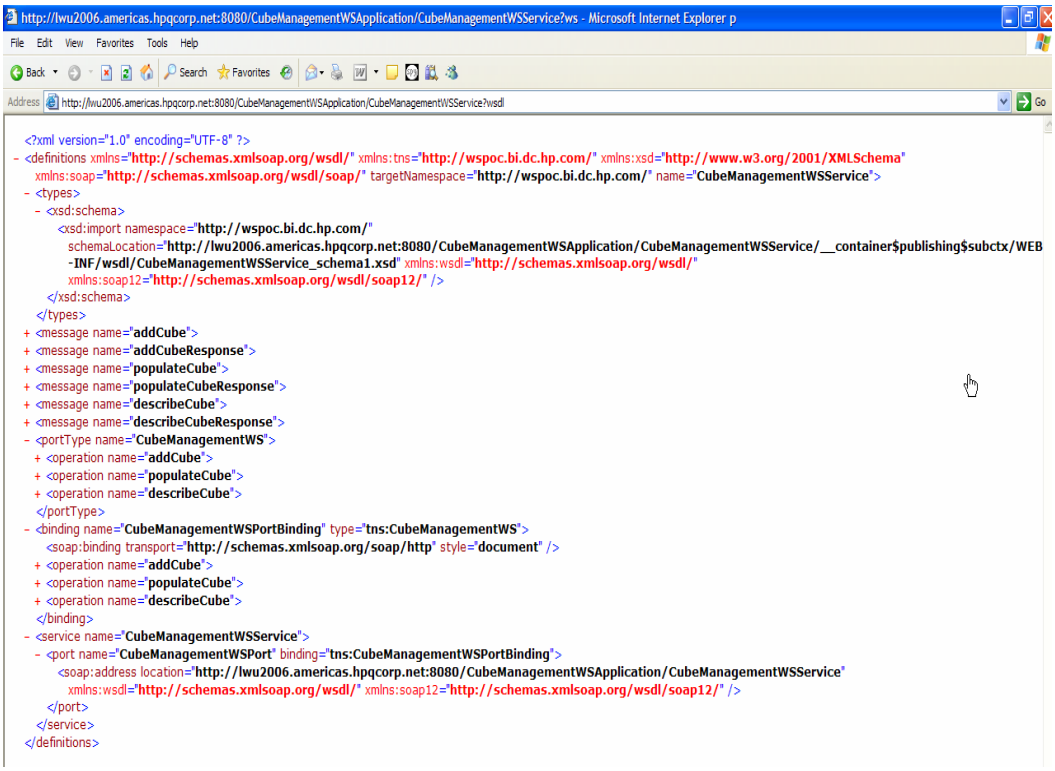

Figure 6: WSDL description of CubeManagementWS web service

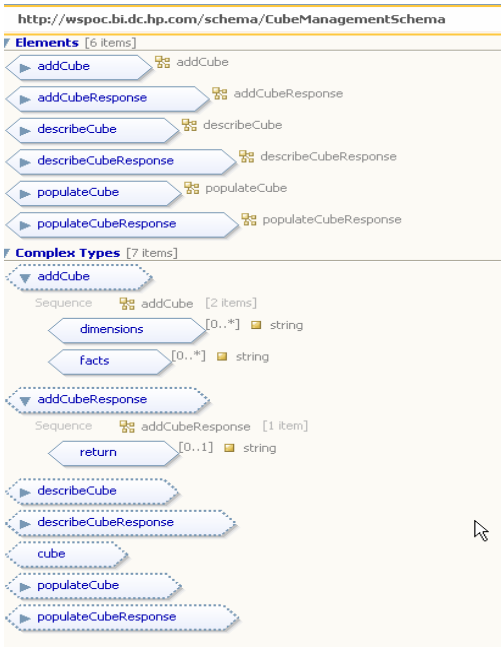

Figure 7: WSDL schema of CubeManagementWS web service

`CubeManagementWS` (WSDL description in Figure 6) is a web service sample which has all datamart (in the following "cube") related operations for other applications or products to use for creating, population and describing the cube data.

A *cube* is a specialized version data available in the data warehouse. Cubes contain snapshots of operational data that helps business people to strategize based on analyses of past trends and experiences. The creation of a cube happens on a specific, predefined need for a certain grouping and configuration of select data. The *dimensions* of a cube represent the various facets of a given *fact* (which is the unit of information in the data warehouse).

The messages used for `CubeManagementWS` are based on the `CubeManagementSchema` file (Figure 7), which contains the data types for dimensions, facts and cubes. The `Dimension` data type includes information like dimension name, dimensions hierarchy level information. And fact data type contains name, transformation formula. Cube is composed by related dimensions and facts.

Example of SOAP request and response for its addCube method invocation are given below in Figure 8.

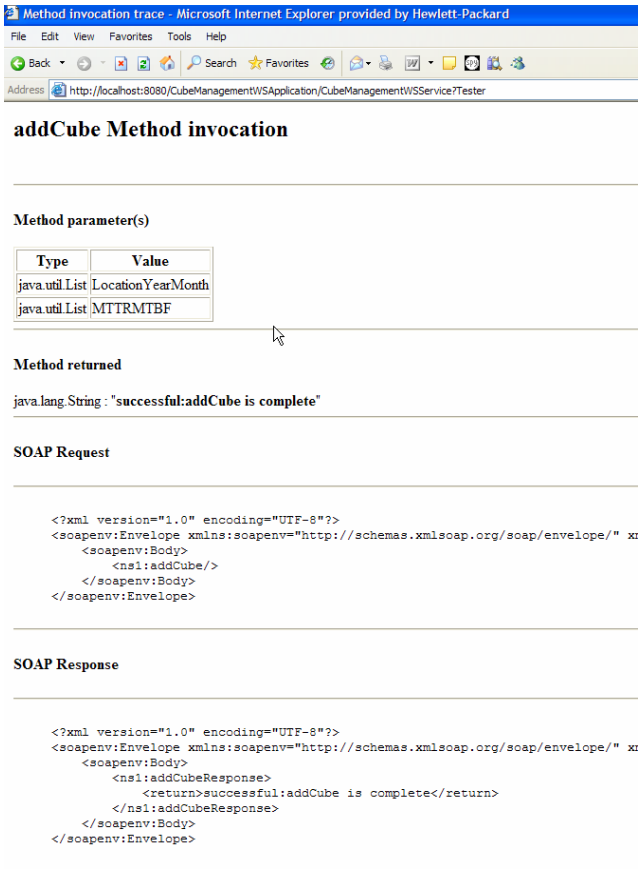

Figure 8: CubeManagementWS::addCube method invocation: SOAP request and response

7. Validation: A use case-based comparison of the service oriented vs. legacy approaches

We validated our proposed SOA architecture through three standard architectural use cases. For each of them, the comparison is based on implementation, impact and cost across all business intelligence system layers.

7.1. Use Case 1 – adding new data source to provide additional analytic module

For the legacy system, the steps to add a new data source are:

1. Modify physical schema for new facts and dimensions
2. Modify physical schema for the changes of existing table attributes and relationships
3. Modify ETL process to map from new data source into data marts
4. Redeploy the modified ETL jobs for data source ETL process
5. Modify the business views to expose new set of cubes, hierarchy and aggregates
6. Modify or create new analytics including reports, metrics and dashboards
7. Publish the analytic content to make public available

In SOA-ITPA, this is much simplified, as the only steps to be undertaken are adding new XML metadata model for the new source and creating or modifying reports, metrics and dashboards. Other implementations are already available because of existing ETL and publish component services.

Not only are the overall development steps reduced more than 70%, but the effort of extension is much less as well. And the system impact and risk are reduced dramatically.

7.2. Use Case 2 – delivering a single web application to support multiple BI tools

The legacy system has to rely on point-to-point integration to support multiple BI tools. For example, when needing to support two BI tools as in figure 9, as many web applications need to be developed and packaged singularly to support the different JDK libraries of the BI tools. It is impossible to deliver a single web application.

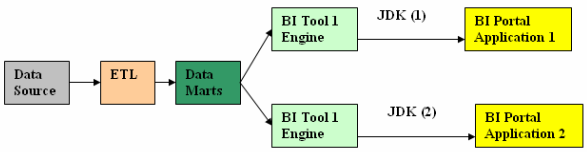

Figure 9: BI portal – Multiple BI tools

With SOA-ITPA, on other hand, by using only one "BI tool bridge" service, we can support additional BI tool API integration. No other components in the system will be impacted. The single analytic web application will be delivered with the web service component updated.

7.3. Use Case 3 – providing closed-loop process to feedback predicted information

The solution to provide closed-loop processing is to create multiple point-to-point integrations to provide the predictive information to all requested applications. The

implementation of integration can be at application level or database level. The integration development effort will be repeated for every requesting system.

For SOA-ITPA, with the availability of the metric publishing service, the loosely-coupled integration is already available for any number of requesting applications.

8. Discussions and conclusions

In this paper we described a service-oriented architecture for business intelligence that makes possible a seamless integration of technologies into a coherent business intelligence environment, thus enabling simplified data delivery and low-latency analytics. We compared our service-oriented approach with traditional business oriented architectures, illustrate the advantages of the service oriented paradigm and share our experience and the lessons learned in architecting and implementing the framework.

In particular, with respect to the legacy application that we described in section 3, our SOA-ITPA approach yielded significant gains in simplicity and cost and impact, as we validated through comparing a number of common architectural use cases.

However, the gain in flexibility comes at a cost in complexity of a service-oriented architecture such as SOA-ITPA presented here with respect to traditional BI systems. This is the trade-off consideration that we need to make. For simpler BI systems than the one we considered here, it could be possible to build viable solutions with the traditional approach. But the SOA approach appears to be the best way to reduce the total development and maintenance cost, and to minimize the risk and impact across an entire enterprise when introducing business intelligence solutions. Another advantage of the approach is that there exist SOA development guidelines and many integrated development environment (IDE) tools that can simplify the development of complex SOA applications.

9. References

[1] C. Nicholls. “BI 2.0 – how real time Business Intelligence is irrevocably changing the way that we do business - In Search Insight,” www.seewhy.com, 2006.

[2] C. White. “Business Intelligence Network – The Vision for BI and Beyond,” www.b-eye-network, June 19, 2006.

[3] J.R. Putman, “Architecting with RM-ODP”, Prentice-Hall, 2001

[4] C. Bussler, “B2B integration: Concept and architecture”

[5] T. Ariyachandra, H. J. Watson. “Key Factor in Selecting a Data Warehouse Architecture,” Business Intelligence Journal, Vol. 10, No. 2, (Spring 2005), 19-26.

[6] T. Ariyachandra, H. J. Watson. “Which Data Warehouse Architecture is Most Successful,” Business Intelligence Journal, Vol. 11, No. 1, (First Quarter 2006), 4-6.

[7] T. Erl, “Service-Oriented Architecture”, A Field Guide to Integrating XML and Web Services”, Prentice-Hall, 2004

[8] H. J. Watson. “Real Time: The Next Generation of Decision-Support Data Management,” Business Intelligence Journal, Vol. 10, No. 3, (Summer 2005), 4-6.

[9] L. Agosta. “Data Strategy Advisor: Data Warehousing Raises the Bar on SOA,” DMReview.com, May, 2006.

[10] L. Agosta. “Data Strategy Adviser: Advance from Traditional to Dynamic Data Warehousing,” DMReview.com, December, 2006.