Implementational Issues of Service-Oriented Cloud-Based Enterprise Management Dashboard

Yuanyuan Li, George Q. Huang, Meng Cheng, Ray Y. Zhong, Hao Luo, L.Y. Pang, Shulin Lan
Department of Industrial and Manufacturing Systems Engineering, The University of Hong Kong, Hong Kong, PR China

Abstract
This paper focuses on the implementations of enterprise dashboard with service-orientated cloud-based considerations. It aims to report practical approaches in the development of real-time enterprise dashboard which supports the dispersed team decisions and finally improves team-awareness. These approaches feature in using some cutting-edge technologies such as real-time data collection solution, data integration from heterogeneous sources, SOA-based system structure, cloud-based deployment manner, as well as user interfaces development. With these approaches, companies are able to effectively reduce the IT system start-up costs, lighten the re-coding workloads, maximize system integrity and scalability as well as standardize the style of user interfaces.

Keywords: Service-Orientated Architecture, Cloud Computing, Enterprise Dashboard

1. INTRODUCTION

It is believed that better process understanding promises final product quality [1]. Through the years, enterprises have innovated kinds of tools for business process management. Dashboard is a mean to support real-time monitoring, analysis, and management of business process [2]. However software engineers often encounter problems during dashboard implementation process.

In the process of dashboard development, IT developers may face the following issues:

- Integration of heterogeneous data sources;
- Wide range of end-users in different locations;
• Limited project investments and development cycle;
• User-friendly User Interface (UI) based on end user’s reference;
• End users are able to real-time interact with dashboard;
• Real-time interaction between end users and dashboard.

Some cutting technologies have been addressed to solve parts of the challenges. Cloud computing is one of them. Advanced in computing resources on demand, cost-effective cloud is available in a pay-as-you-go manner for the public. Enterprises are able to loan the storage resources in the cloud instead of capital investment [3]. Before cloud computing emerging, service oriented architecture (SOA) was already used with web service technology. In SOA architecture, services are loose coupled and reusable. It is flexible to construct services dynamically [4]. Application components are proposed to be assembled into a network of services [5]. In addition, RFID-enabled real-time reconfigurable manufacturing system gives a way to retrieve data bottom-up from the work-cell [7]. It is the fundament facility for real-time dashboard application. Besides the back-end supporting technology, XML-based language is utilized for UI development. It will facilitate to rapid prototyping of UIs [6].

However, each technology only can solve part of the challenges. There are limited studies on dashboard implementation with the combinations of all the technologies mentioned above. In order to address all the above issues, this paper focuses on introducing a solution with cutting-edge technologies, including cloud computing, SOA-based data integration, web services, flexible UI components, and RFID technology, to provide, support and create the possibilities of dashboard development. This paper targets to develop a flexible access dashboard with real-time monitoring functionality through limited investments and development life cycle.

2. DASHBOARD IMPLEMENTATION ARCHETECTURE

As illustrated in Fig 1, the solutions equipped with RFID technology are proposed for the IT developers. There are summarized below and will be further discussed in the following section:

• Manage the service components with SOA;
• Develop UI with XML-driven components;
• Centralize data source with information source service.

In the following section, each part of the structure will be reported from the aspect of IT development.
3. **Service-Oriented Architecture (SOA)**

   A. **Information source integration**

   1) **Information source categories**

   As shown in the right-hand side of Fig 1, three categories are included in the information source stated below:

   - **Standard web service category**: It includes different optimization algorithms of dashboard;
   - **Third-party legacy enterprise information /application systems (EISs) category**: It covers a set of existing decision-making systems such as ERP, SCM, CRM, WMS, PDM systems;
   - **Various native databases category**: It is consisted by heterogeneous databases that are used for keeping various data, supporting the decision-making activities within different EISs.

   During the implementation of dashboard, there are three main challenges. Firstly, dashboard applications should interact with these heterogeneous data sources. The applications and data from EISs may mismatch the dashboard’s requirements such as data format, interface parameter etc. Secondly, the EISs and native database might be developed several years ago. Thus, current IT developers may not exactly get the know-how when contemplating dashboard implementations. In order to handle the above two challenges, the approach, integrating data sources by exposing some functions and data from EISs as web services, is used to design the information source service. Thirdly, if the data sources would be integrated, the relations of the sources should be considered. Each source should be independent and interoperable. If some sources have been modified, the others should remain unchanged. As the independence of each source, the dashboard application would invoke these sources in different sequence.

   2) **Exposure as web services**

   As mentioned above, dashboard will interact with various EISs. Package applications (EISs) expose some of their functionality and data as web services [8] in the cloud. From SOA aspect, all the data sources will be interacted in terms of services. They are loosed coupled and reusable. Dashboard applications are able to invoke those services on business demands. And it does not influent the other services or dashboard operations, if some of service has been modified. These functionalities will be reusable to dashboards through adapters. Details about adapter developments will be reported in part 4.

   In the data source side, the EISs should provide the interfaces to adaptors for requesting processing [9]. Based on the data sources fundamentals, the main development fundamental tools and processes should be used to achieve SOA.
B. **Fundamental tools**

During the implementation process, a set of fundamental tools should be used. The key tools are UDDI, WSDL and SOAP that are detailed clarified as follows.

1) **UDDI**

The Universal Description, Discovery and Integration (UDDI) specification defines a standard way for registering, unregistering, and looking up Web Services [9]. UDDI defines two APIs in publishing services. They are publisher API and inquiry API. Publishers API is applied to register services while Inquiry API is used to discovery service registry. Besides defined APIs, UDDI has specified an XML schema for SOAP message.

2) **WSDL**

An XML-based Web Service Description Language (WSDL) describes the interface and various dashboard services and concrete implementation elements. In other words, this self-described language describes the functionality, the location and the invocation methods of each web service. A web service, such as an analysis algorithm should be registered in UDDI with WSDL so that it will be discovered efficiently on receipt of requests.

The main elements of interface description are 1) definition element; 2) types elements; 3) message and part element; 4) operation element; 5) input, output, and fault elements; 6) portType element.

3) **SOAP**

Simple Object Access Protocol (SOAP) is an XML-based message format which defines a set of rules for data types. It is used to invoke the peer web services by exchanging XML-based message. SOAP is a platform-independent protocol and defines a uniform XML format for data exchange. Defined format includes envelop head and body. The actual data of dashboard is in the envelop body.

C. **SOA process**

This section reports a SOA process to implement the dashboard so as to cut down the development cycle and ease the operations within different system deploy stages. Four phases are utilized as follows.

1) **Service deploy**

The web services mentioned above can be deployed and run in the cloud which is called infrastructure as a service (IaaS) to reduce the investments. End users from different regions can access these web services by Internet. Fundamental computing resources such as processing, storage, networks are defined as standardized services over the network. There are three main process needs to be followed:
• Plug the web services components into the IaaS;
• Configure environment parameters and install the archive files (e.g. class or jar files) into corresponding folder;
• Test the service to see if it has been deployed successfully.

2) Service publish

After having been deployed in the cloud, the web services could be published in an independent platform called UDDI platform. Before the service publication, a UDDI platform should be developed. It will manage the services so that they can be found and invoked. Besides the service publish phase, it also plays as an important role in the service search and service bind phases.

Once the platforms are developed, web services can be published and deployed. Details information such as service location, capability, interfacial description will be provided in the platform. Four data types are required during service publish process including businessEntity, businessService, bindingTemplate and tModel. Reference [10] has provided more details of them.

3) Service search

Service searching could be executed programmatically or by specific explorer. The first method will be triggered automatically by requestor in terms of SOAP message. The second measure should be completed manually with explorer.

Both search measures require searching criteria from categoryBag. The searching criteria are stated below [10]:
• Uniform resource identifiers (URI) which reflect the service providers;
• Remote methods to be invoked and input parameters;
• The input and output parameters of the remote method with the corresponding data types.

4) Service bind

After the delivery of services, the bind phase is in process. There are two main steps should be followed:
• Send SOAP request message to web services of dashboard: including input parameters of platform element and requirements elements;
• On receipt of SOAP request, the web services could be ready for the further orders. Feedbacks could be sent to requestor in the form of SOAP when web services accomplish the task.
4. STANDARD AND COMPONENT – BASED DEVELOPMENT OF USER INTERFACE

Third party components of dashboard will improve the UI design skills within a short time as well as uniform the UI styles of the whole web page layouts in one dashboard. Therefore, this approach is adopted with the component-based design and development principle to realize the dashboard.

A. Selection of third party component

To select the suitable 3rd-party component, there are several concerns:

- Reusability: components should be easily reused. In dashboard design, some main components could be employed several times;

- Flexibility: document-view representations of the same data content may be varied according to different reference from different language countries. For example, in a multination corporation, Chinese people would like to view the dashboard in Chinese while the Americans prefer English. These requirements should be flexible according to different references;

- Functionality for dashboard design: input and output of UI tools should be designed to meet the requirements in practical work process. Reference [11] has addressed thirteen common mistakes in dashboard design;

- Cost factor: the costs usually include 1) software cost 2) support cost. Selection should balance the cost and its benefits.

B. UI Layout design

The UI layout includes two parts, including Menu Zone and Workspace designs. Before designing each party, IT developers should have a prototype framework such as the size and position of each part.

1) Display principle

Dashboard should be designed within a single screen. Exceeding boundaries may reduce the level of arbitrary and

![Figure 2 UI component](image-url)
cause disturbance [11]. The dashboard should be understandable at a glance. Selection of appropriate representation tools is important to dashboard UI design.

2) **Elements in Menu Zone**

Elements in Menu Zone could be developed in master page with Visual Studio. Its target is to indicate the views and the location of the existing web page to end users clearly. Some end users may have many web pages displayed especially when they drill down some dashboard.

In general, there are three main elements in Menu Zone as shown in fig 2:

- **Menu bar**: It is on the top of the dashboard, helping end users to link the applications;
- **Navigation bar**: As there are several web pages in a dashboard, navigation bar prevents the end users to be lost in piles of web sites;

3) **Elements in Workspace**

Dashboards can be divided into three categories based on the type of business activities that they support [10]. The three categories are 1) Dashboard for strategic purpose; 2) Dashboard for analytical purpose; 3) Dashboard for operational purpose.

Dashboards in different categories need different UI present elements. Based on the business and work process that they support, each dashboard has its own specific requirements on the representation media.

The UI elements are expected to present the KPIs and other information visually at one glance with appropriate color. As shown in Fig 2, Workspace can be divided into two sub-parts. One sub-part is Workspace header and the other is Workspace body. The former contains auxiliary tool, while the latter embraces main visualized information. Fig 2 displays main components in each sub-part.

- **Workspace description**: It describes the basic information the summary of the dashboard. It is usually developed by form and label component.
- **Area tools**: End users are able to interact with the area tools. They are usually in terms of button, text box as well as check box components.
- **Work process Navigation Bar**: It is kind of area navigation. It provides a distinct view of the whole work process. Besides, End Users, especially operators, can jump to different phase of work process through the tool.
- **Key information**: the main information of the dashboard will be virtualized in this zone. It often comprises the grid, treegrid, tree, kinds of charts [11] as well as calendar components.
C. XML-Driven Document-View application

There are challenges in component selection, layout design as well as UI components usages. But for IT developers, they may pay more attentions on the UI design challenges.

1) UI design challenges

When considering the UI flexibility, IT developers usually face the two challenges below [12]:

- Strong coupling between UI display media and business logic layer: if business logic has changed, the UI layer should be modified accordingly. It will cost time and energy to modify every relative point;

- Lack of customized views for different users: Different users may require different document-view on same data content.

2) Implementation considerations

The sources of document-view are saved in XML files. The formats of XML files are defined by the third party component. IT developer only needs to modify the XML file when business logic or customer reference changes.

When the dashboard is invoked, the web page will load customized layout. Each customized representation preference could be saved in an XML file. Dashboard will find the corresponding XML which should be loaded according to the login portal information.

5. INFORMATION SOURCE SERVICE

All the components mentioned above should be connected appropriately. The following adapters will play as bridges among them.

A. Information Source Adaption Service

As illustrated in the middle of Fig 1, the information source adaptor targets to load or update data from heterogeneous resource. As EISs are exposed as standard web services, the adaptor converts these data sources into standard XML segments like ISA 95 and B2MML and sends to UI component adaptor. B2MML schemas describe the information about man, machine, material, and production process.

It is proposed to Develop Service Adapters Using Mismatch Patterns. Reference [13] has provided more details on the development.

B. Component Adaption Service

Third party components usually define some XML formats different from standard XML segments. The data models are strictly defined by the UI component providers. In the middle of Fig 1, the second level of adaption service is to convert standard XML file into the format that the
third party components defined. Because each third party components provider defines its specific data model, the adaptor has to be changed once the UI component suit changed.

6. Case study

This case study describes a container loading dashboard development including UI design and back-end development. Before the dashboard in used, the enterprise users execute the loading process manually.

Container loading dashboard was to be developed targeting to improve work efficiency. The implementation issues and solutions are stated in the following sections.

A. Component-based Front-End development

Challenges

- How to virtualize the data report of cargoes, containers as well as their relations clearly at one glance;
- Staff from different countries would like to read different language in the same data content;
- Ease the UI layout change if business logic changes;
- Improve the real-time interaction between end users and the dashboard.

Besides the representation media, the customized document-view should be realized. The Chinese operator may like to view the dashboard in Chinese while another operator based in U.S. refers English. XML-driven Document-View method is applied to fulfill this requirement. The customized Document-Views are saved in different XML files. When operator logins the dashboard, the corresponding XML-based Document-View file will be loaded. When container business logic has changed, IT developers only need to modify the corresponding XML files.

As shown in Fig 3, the relations between each element are illustrated through display media. And operators are able to feedback to dashboard. Data exchange between UI and the web services communication is in XML format. Hidden filed is used to send feedback information to the back-end in forms of XML.

B. Back-End development

Challenges:

- Data integration: Data information is stored in different EISs system. In addition, container loading optimization algorithms are newly-developed, not from any of the EISs;
• Besides the basic container loading rules, in practice, there are another five optional rules. Based on the clients’ reference, container loading optimization should obey some of the rules.

It is proposed to solve these problems based on SOA technology. All the data source or algorithm can be treated as services which can be deployed in the cloud. The service consumers will find and bind the services as they need.

1) data source Integration

To integrate the heterogeneous data source, EISs are proposed to be exposed as web services. The optimization algorithms are also developed as web services. To reduce the hardware cost, these web services are deployed in a public cloud. And then IT developer will deploy them in UDDI platform. The service consumers will find these services and bind them.

2) Main Web services

The main web services are stated below:

• Smart Gateway service: container loading dashboard will monitor the status of the cargoes, product to be loaded, containers as well as the task of forklift. Each of these is dynamic. Smart Gateway services are used to update the status of the RFID enabled smart object [7];

• Planning service algorithms: As mention above, there are several loading rules. The optional rules are developed as separate web services;

• Scheduling service algorithm: The warehouse supervisor assign the tasks to forklift drivers through this service;

• Web service exposed by EISs: EISs have been developed interfaces for data exchange.

3) Service Selection Panel

Shipping planner could select optional container loading algorithms according consignee’s requirements. The panel is designed for shipping planner to selected optional rules. These algorithms map different web services. In other words, from the aspect of SOA, the invoked services are possibly different case by case.

7. SUMMARY

This paper describes a set of solutions to overcome the implementation issues on dashboard realization. It focuses on both front-end and back-end facets. For assisting the IT developers, a dashboard with high cost-effectiveness, functionality and flexibility could be developed through the proposed solutions.

The solutions have demonstrated several key contributions. Firstly, cloud-based application allows the computing storage to cost less. IaaS benefits enterprises from saving costs of IT
hardware and easily extending data storage on demand. Secondly, heterogeneous data sources expose their functions as web services which ease data exchange between client side and various databases. Thirdly, based on SOA, the services are loose coupled and reusable. Dashboard is able to group the services dynamically according to the business operations. Fourthly, XML-driven UI component supports customized document views. Finally, a uniform styled dashboard with various representation media could visualize the data clearly.

However, only container loading dashboard is implemented according these solutions. In the next step, the current work will be extended to other kind of dashboard implementations.

ACKNOWLEDGMENT

The authors are grateful to Schools and Research Institutions Project (2010A090200054) for the partial sponsorships for this research, 2010 Guangdong Department of Science and Technology Funding (2010B050100023), Guangdong University of Technology and International Collaborative Project of Guangdong High Education Institution (gjhz1005) and Hong Kong SAR RGC.

References


Component Based Explorers: Scheduling Explorer

Information Source Service

Component XML

Component Service

B2MML/ISA95 XML

EIS Adaptation Service

Bind

Find

Register

Deploy

Web Service Description Discovery and Integration

Information and application Service Providers

Information Source

Proprietary EISs

ERP

WMS

MES

Native Databases

MS SQL

Access

My SQL

Standard Web Services

Smart Gateway service

Planning service

Scheduling Service

Figure 1 Innovative uses of cutting-edge technologies in Dashboard