A Framework for E-Government Based-on Web Service

Hua-Liang Hu*

Contents

Abstract

The modern government's role: As an enterprise (To be efficient and effective.); as a service and information provider (government processes that are unique to government). The modern government's role demand interoperation. The Web services architecture is based upon the interactions among three components: Service provider, service broker and service requestor. Service broker is sometimes referred to as service registry. The interactions involve publish, find and bind operations. The paper introduces the concept of web services as a way to realize interoperation between distributed applications. The paper addresses interoperation between e-government and citizen. The motivation of this work was to determine the potentials of the Web services technology for an interoperability infrastructure for e-government. In many countries, the traditional governments have many problems that stem from both insufficient and improper use of ICT. "Insufficient use" refers to the traditional means such as manual archiving systems. "Improper use", on the other hand, refers to the lack of an interoperability infrastructure within and among the government agencies. In this work.

* Hua-Liang Hu, hwjandy@163.net

I. Introduction

1.1. Government

Interactions between governmental organizations and entity as following:

- **Government-to-government (G2G) or administration-to-administration (A2A)**: internal processes of the administration.
- **Government-to-business (G2B) or administration-to-business (A2B)**:

Interactions between governmental organizations and commercial or business. Organizations, e.g. taxes.

- **Government-to-citizen (G2C) or administration-to-citizen (A2C)**:

Interactions between governmental organizations and citizens,

E-government has been on the agenda of many developed and developing countries recently. An interoperability infrastructure is at the heart of e-government It is this infrastructure which would make the interaction between government and citizens (G2C), government and business enterprises (G2B), and inter-agency relationships (G2G) more friendly, convenient, transparent, and inexpensive.

The motivation of this paper was to determine the potentials of the Web services technology for an interoperability infrastructure for e-government. In many countries, the traditional governments have many problems that stem from both insufficient and improper use of ICT. "Insufficient use" refers to the traditional means such as manual archiving systems. "Improper use", on the other hand, refers to the lack of an interoperability infrastructure within and among the government agencies.

Identified problems were classified under five different though inter-related categories. These are the lack of auto-control mechanisms, high economical losses, and high cost of services, poor service quality, and low efficiency. These problems are briefed below.

Auto-Control: The lack of auto-control mechanisms is the direct result of improper use of ICT. One of the most striking side effects of this problem is high economical losses.

Actually, an interoperability infrastructure which would facilitate the updates is not in place. Therefore, updating is either neglected or addressed by auxiliary archiving mechanisms. It is very difficult to trace such modifications in manual archiving systems considering especially very high pace of the large municipalities such as Istanbul.

Efficiency: Due the traditional ways of doing things, the pace of activities is rather slow. By traditional ways we mean manual archives, manual procedures, visual analyses, traditional ways of interoperability, and finally requesting data

from the citizens, which is already under responsibility of some government agency or municipality department. There are many examples to this issue. A characteristic example is the preparation of zoning plan forms, which will be mentioned below. Low efficiency may have a negative impact on the economy since it may postpone transforming resources to the economy. Consider how the time to get a building permit will affect the construction business and related sectors for instance. On the other hand, low efficiency will yield poor quality services.

Quality of Services: Quality of services not only causes dissatisfaction of citizens with their government or municipality, but also results in economical losses. Several problems may be associated with the quality of government and municipality services. The main problem stems from the lack of how a service should be. In the traditional way, citizens are perceived as, in a sense, a "worker" of the services. Because, traditionally and even backed by the regulations, almost all government agencies and municipalities ask citizens to collect some of the data needed for their applications. And this is the data that government agencies and municipalities could have obtained from each other If there had been an interoperability infrastructure such as NSDI in place. This view of services is far behind the implications of "Information Age".

Another aspect of the quality of services is related to the Public participation in governance. Let alone the participation, the way of informing citizens about government and municipality decisions is rather clumsy at the moment. A characteristic example of this is proclaiming the zoning plan modifications to the citizens from a bulletin board where citizens may hardly see their parcels and related owner information.

Economical losses: These are the result of the problems in the other categories. As already explained, one of these problems are the lack of auto-control as in the case of real estate taxes. And the rather one is the rather slow pace of activities as in the case of building permits.

Cost of services: There are many causes of high costs in traditional governments. First of all, due to the lack of interoperability, development and production costs are high. For instance, data transfers between municipalities and other parties and between different offices within the same municipality are still performed by traditional methods. As very well documented in the literature, this is a costly and time consuming operation. As an example, the rate of using Internet for data transfer between municipalities and government agencies or private sector is very low at the moment. There are even cases where one has to actually travel to another city and get the data.

The modern government's demand as

following:

Modern joined-up government demands joined-up ICT systems. Interoperable systems working in a seamless and coherent way across the public sector hold the key to providing better services, tailored to the needs of the citizen and business and at a lower cost.

Clearly defined policies and specifications for interoperability and information management are also key to staying connected to the outside world and aligned to the global information revolution. The e-GIF provides these. It is a fundamental framework policy for the e-Government strategy.

Government information resources are not only of value in themselves. They are valuable economic assets, the fuel of the knowledge economy. By making sure the information we hold can be readily located and passed between the public and private sectors, taking account of privacy and security obligations, we can help to make the most of this asset, thereby driving and stimulating our economy.

1.2. Web Services Architecture

Web services, were first uttered by Microsoft chairman Bill Gates at the Microsoft Professional Developers Conference in Orlando, July 12, 2000. Web services have emerged as the next generation of Web-based technology for exchanging information. Web services are modular, self-describing, self-contained applications that are accessible over the Internet. Based on open standards, Web services enable constructing Web-based applications using any platform, object model, and programming language (Barefoot, 2002). A service is a collection of operations accessible through an application-programming interface that allows users to invoke a service, which could be a response to a simple request to create a map or a complicated set of image-processing operations running on several computers (Hecht, 2002). Once a Web service is deployed, other applications and Web services can discover and invoke that service.

The Web services architecture is based upon the interactions among three components: Service provider, service broker and service requestor. Service broker is sometimes referred to as service registry. The interactions involve publish, find and bind operations. In a typical scenario, a service provider hosts a network-accessible software module (an implementation of a Web service). The service provider defines a service description for the web service and publishes it to a service broker. The service requestor uses a find operation to retrieve the service description locally or from the service broker and uses the service description to bind with the service provider and invoke or interact with the Web service

implementation). The components are explained as following:

Service provider: From a business perspective, this is the owner of the service. From an architectural perspective, this is the platform that hosts access to the service. Service provider is responsible for developing and deploying the Web Services. The provider also defines the services and publishes them via the service broker.

Service requestor: The service requestor is the one who request a service. The requestor locates the Web service using the service broker, invokes the required services, and executes it from the service provider.

Service broker: The service broker is responsible for service registration and discovery of the Web services. The broker lists various service types, descriptions, and locations of the services that help the service requestors find and subscribe to the required services.

1.3. Web Services Technologies {1,2,3}

Web Services can be developed using any programming language and can be deployed on any platform. Web Services can communicate because they all speak the same language: the Extensible Markup Language (XML). Web Services use XML to describe their interfaces and to encode their messages. XML-based Web Services communicate over standard Web protocols using XML interfaces and XML messages, which any application can interpret.

The three core XML-based technologies for building and enabling Web services are:

Simple Object Access Protocol *(SOAP)* defines a standard communications protocol for Web Services.

Web Services Description Language *(WSDL)* defines a standard mechanism to describe a Web Service.

Universal Description, Discovery and Integration *(UDDI)* provides a standard mechanism to register and discover Web Services.

When a service provider wants to make the service available to service requestors, he describes the service using WSDL and registers the service in a UDDI registry. The UDDI registry will then maintain pointers to the WSDL description and to the service. When a service requestor wants to use a service, he queries the UDDI registry to find a service that satisfies his needs and obtains the WSDL description of the service, as well as the access point of the service. The service requestor uses the WSDL description to construct a SOAP message with which to communicate with the service.

SOAP is a lightweight protocol for exchange of information in a decentralized, distributed environment. It is an XML based protocol at the core of which is an envelope that defines a framework for describing what is in a message and how to process it

and a transport binding framework for exchanging messages using an underlying protocol. Adjuncts to the envelope and binding framework include a set of encoding rules for expressing instances of application-defined data types and a convention for representing remote procedure calls and responses. Given the Web' s intrinsically distributed and heterogeneous nature, communication mechanisms must be platform-independent, international, secure, and as lightweight as possible.

XML is now firmly established as the lingua franca for information and data encoding for platform independence and internationalization. Building a communication protocol using XML is thus a natural answer for WSs. SOAP was initially created by Microsoft and later developed in collaboration with Developmentor, IBM, Lotus, and UserLand. SOAP is an XML-based protocol for messaging and remote procedure calls (RPCs) and works on existing transports, such as HTTP, SMTP, and MQSeries. The XML part of the SOAP request consists of three main portions:

- The Envelope defines the various namespaces that are used by the rest of the SOAP message.
- The Header is an optional element for carrying auxiliary information for authentication, transactions, and payments. Any element in a SOAP processing chain can add or delete items from the Header; elements can also choose to ignore items if they are unknown. If a Header is present, it must be the first child of the Envelope.
- The Body is the main payload of the message. When SOAP is used to perform an RPC call, the Body contains a single element that contains the method name, arguments, and e-service target address. If a Header is present, the Body must be its immediate sibling; otherwise it must be the first child of the Envelope.

WSs Description Language is an XML-based language used to define WSs and describe how to access them. WSDL is used to specify its location, and describe the operations (i.e. methods) it exposes, similar to the way how a type library is used to describe a component [SHOHO-2002]. For an application to use a WEB-service, the programmatic interface of the e-service must be precisely described. In this sense, WSDL plays a role analogous to Figure 4 SOAP fundamentals 6 Interface Definition Languages (IDLs) used in several distributed environments (e.g. CORBA).

The WSs Description Language (WDSL) [WSDL-2002] is a new specification to describe networked XML-based services [OGBUJ-2000]. It provides a simple way for service providers to describe the basic format of requests to their systems regardless of the underlying protocol (such

as Simple Object Access Protocol or XML) or encoding (such as Multipurpose Internet Messaging Extensions). WSDL is a key part of the effort of the Universal Description, Discovery and Integration (UDDI) initiative to provide directories and descriptions of such on- line services for electronic business.

A WSDL document defines services as collections of network endpoints, or ports. In WSDL, the abstract definition of endpoints and messages is separated from their concrete network deployment or data format bindings. This allows the reuse of abstract definitions: messages, which are abstract descriptions of the data being exchanged, and port types that are abstract collections of operations. The concrete protocol and data format specifications for a particular port type constitute a reusable binding. A port is defined by associating a network address with a reusable binding, and a collection of ports defines a service. Hence, a WSDL document uses the following seven elements in the definition of network services:

- Type: A container for data type definitions using some type system (such as XSD).
- Message: An abstract, typed definition of the data being communicated.
- Operation: An abstract description of an action supported by the service.
- Port Type: An abstract set of operations supported by one or more endpoints.
- Binding: A concrete protocol and data format specification for a particular port type.
- Port: A single endpoint defined as a combination of a binding and a network address.
- Service: A collection of related endpoints.

The Universal Description, Discovery, and Integration specifications offer users a unified and systematic way to find service providers through a centralized registry of services that is roughly equivalent to an automated online "phone directory" of WSs. The browser-accessible UDDI Business Registry (UBR) became available shortly after the specification went public. Several individual companies and industry groups are also star ting to use "private" UDDI directories to integrate and streamline access to their internal services. UDDI provides two basic specifications that define a service registry' s structure and operation:

- a definition of the information to provide about each service, and how to encode it;
- A query and update API for the registry that describes how this information can be accessed and updated. Registry access is accomplished using a standard SOAP API for both querying and updating.

UDDI encodes three types of information

about WSs: (1) "white pages" information includes name and contact details; (2) "yellow pages" information provides categorization based on business and service types; and (3) "green pages" information includes technical data about the services.

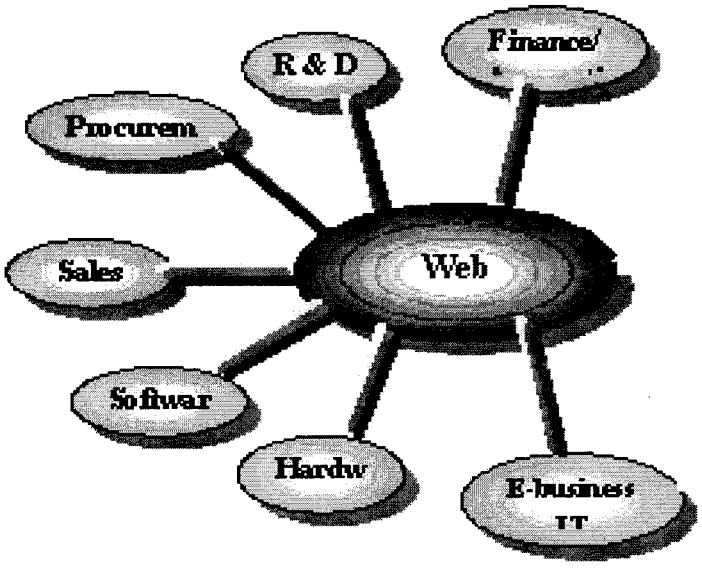
The UDDI registry is organized around two fundamental entities that describe businesses and the services they provide. The business Entity element provides standard white-pages information, including identifiers (tax IDs, social security numbers, and so on), contact information, and a simple description. Each business entity might include one or more business Service elements, which represent the services it provides. Both business and service entities can specify a category Bag to categorize the business or service.

Ⅱ. Design

2.1. E-Government with Web services

The realization of a citizen portal as a one-stop-government solution is a typical scenario which demonstrates the advantages of using the Web services concept. Web services allow for easy integration of different applications and information sources into one unified portal. A so-called portlet is a content container: basically the user's view of the customized content which shall be integrated into the portal. A portal taking advantage of the Web services concept integrates portlets from the different content providers into a single portal. Figure 7 gives an example how a portal finds and integrates remote portlets.

2.2 Design



Ⅲ. Conclusions

Traditional governments have many problems that stem from both insufficient and improper use of ICT. These problems have been identified and classified in this paper. These are high economical losses, high cost, lack of auto control, poor quality of services, and low efficiency. The main goal of E-government is to solve these problems. E-government requires an interoperability infrastructure which would make interactions between government and citizens (G2C), government and business enterprises (G2B), and government and government (G2G) more friendly, convenient, transparent, and inexpensive. As a result we have found that Web services enable quick and high quality services. And they yield cost savings in software development and service provision. Web services do have problems like security. We have not tackled these problems in this study. We have only examined the usability of a new technology, Web services, for the e-municipality infrastructure. Since e-government and e-municipality share similar structures and problems, the results of this work is valid within the e-government context as well. And to our knowledge, this work is one of the very first of its kind.

This paper described the basic concepts of Web services and the main related protocols SOAP, WSDL, and UDDI. The implementation of e-government applications which are based on a Web service framework has advantages in interoperability and represents the actual state of the art in the development of distributed systems. While non Web service based (proprietary) e-government applications have been developed over the last few years, most of them are still in an experimental or prototype stage.

New e-government applications should be based on common standards to allow for easier integration with other applications. The issue becomes even more relevant under the aspects of European integration. The number of existing e-government applications is still small; therefore upgrading them to a Web service based framework is not a problematic issue at the moment. The experiences gained with the existing applications should go directly into the next generation, which should be based on Web service standards.

The challenge for the forthcoming e-government applications is to learn from the e-commerce experiences of the last five years and to avoid the apparent shortcomings. Founding the applications on a Web services framework promises to be a feasible solution. By adopting the Web services technology integration on a European scale can be supported.

Reference

1. http://www.w3.org/XML
2. http://www.uddi.org
3. http://www.w3.org/TR/2002/WD-ws-desc-usecases-20020604/
4. http://www.w3.org/TR/2001/WD-soap12-part1-20011217/