EDI Support System over RESTful Web API

A. Fujii[*], M. Nakayama[**], K. Tanaka[***], and K. Nagamura[***]
[*] Hosei Univ., Applied Informatics, Tokyo, Japan
[**] Next Object Inc., [***] Multi-paradigm Systems Inc., Tokyo, Japan
e-mail fujii@hosei.ac.jp

Abstract—Recently Web API (Application Programming Interface) is widely utilized for the service interfaces in enterprise applications. In this article, the possibility of applying Web API to electronic commerce, especially to EDI (Electronic Data Interchange), is discussed. We propose architecture called "Open EDI" as a platform for such activities. And the notation of "Cloud_Object" is introduced that is the structured operational unit for EDI. We aim the system works as a general web applications without specific operational environment for the data-exchanges.

Keywords: Web API, RESTful, B2B, e-commerce, EDI

I. Introduction

The value of Cloud computing is in the way of delivering different IT resources to users/developer through service platform which is commonly provided by open web technology. When it comes to design methodology for linking data resources, common design pattern or rules are necessary in each business domain. In other words, the methodology for utilizing Web API (Application Program /Programming Interface) efficiently through SOAP (Simple Object Access Protocol) or RESTful (Representational State Transfer) composition is necessary. [4]

Most electronic commerce applications center on the product catalog and transaction modules. Traditional paper catalogs are usually costly and difficult to update. The Web-based digital catalogs provide more efficient information updating with lower costs. However general Web-based catalog is still based on so called "static" representation which is based on mere transformation from photo and data table to pdf-based files.

On the contrary, recent technological progresses make it possible that catalog information can be delivered through so called "dynamic" representation that is provided through Web API (Application Program /Programming Interface) features. This dynamic methodology for catalog management is quite suitable for enterprise business use in cloud computing environment where the computational resources such as data and operational functionalities are virtually provided to users through standard web-based gadgets.[10][14][15]

In this paper, RESTful catalog management system is introduced and the development of EDI (Electronic Data Interchange) feature over the system is discussed. The operational linkage between resources or functionalities that may be provided different enterprises are composed "mashup" technology. For the design architecture, we introduced a concept of "Cloud_Object" in order to define resource management where the resources are represented based on RESTful features. Then we discuss a model of business platform with such systems.

II. Technological Trends and Related Research

2.1 Technological Trends

Currently, major online commercial websites including Google and Amazon are using Web APIs to provide their users with information resources, such as Google Maps. In this way, software development methods that incorporate Web APIs are increasingly being used. The series of developments in this area was triggered by Yahoo's 2007 release of Pipes, an information mashup tool for its users. Following this, a trend toward constructing business applications using this newly feasible function for linking data began to gather momentum. In particular, this technique occasionally came to be known as "enterprise mashup" when it was adopted by enterprises in their information systems and business applications.

In the future, with new developments in data linking, the software development and how software-based services will be provided will see radical changes. It is also possible that the users themselves will create a range of new business applications. This is because, even if the users are not experts in software development, by combining the existing software components provided as Web APIs they will still be able to easily and quickly develop functions that are useful to them, and then make them publicly available. This trend can be called "end-user development."

978-1-4244-7395-3/10/$26.00 ©2010 IEEE

2.2 Related Research

A research example is the European FAST Project[2], which has been funded from the ICT Policy of the European Commission to research enterprise information system methodologies. This project began in March 2008 and will continue until the end of March 2011, with a total budget of 5.5 million euros (about 720 million yen). It is the successor to the "SecSE (Service Centric Software Engineering) Research Project[3]," which was operated between 2004 and 2008 with a budget of 15.2 million euros (about 1.98 billion yen). The goal of this project was to construct a platform that would allow multiple enterprises to coordinate their services. It aimed to develop service-centered applications and also effective methods and tools for use.

[22], a research paper by a SAP research center can be listed as one of the FAST Project's research results. It analyzed of a number of data-link patterns between enterprises using Web APIs, and used the results of this analysis to create models. Moreover, there are even ongoing discussions toward linking data via Web APIs within the Private Cloud, an environment currently being researched by, for example IBM [5].

When combining input and output from a Web API, it is necessary to describe the API and input and output as a series flow in order for high-level data linkage, such as EDI, to be achieved. For example, the W3C standard XProc [1]has been defined as a potentially interoperable, industry-standard method for processing XML documents, and considerable energy is being spent on promoting its use. Also, Yahoo has been increasing the potential uses for its platforms, including offering its Pipes tool to the end user, and in doing so it could be said to be promoting user-created business applications that utilize a data-linking function.[19] [23]

III. The research target

We consider solutions for a certain specific business domain, namely mechanical parts distribution in Japan and Asian countries. At first, we discuss the practical situations of mechanical parts distribution in Japan, then look back the trial of EDI so far.

3.1 Current conditions in mechanical parts distribution

The general characteristics of production distribution in Japan are as follows:

1) A multistage distribution process

2) A large number of products handled with a lack of uniformity within them, such as for product names

In particular, there are many medium-to-small trading companies involved in the distribution processes for mechanical parts. These trading companies are positioned between the design and manufacturing sites and parts manufacturers. In addition to maintaining distribution and inventory facilities in locations near to these sites, they also provide a wide variety of customer services based on their product knowledge.

In such a situation, paper based catalog plays very important role in business. Companies publish their original books almost every year. Refinement and printing usually cast over a few hundred thousands euros (a few 10Mllion JPY).

3.2 General Brief history of EDI

Looking back on the introduction of Electronic Data Interchange (EDI) by distribution-related businesses, we can see there has been not only a wide range of both large and small scale initiatives, but also considerable differences to the extent to which they are being utilized. Within these initiatives, some offer highly advanced and novel technology. For example, efforts to formulate and popularize the ebXML standard have been attracting attention, and it is the standard being promoted by the United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT) and the Organization for the Advancement of Structured Information Standards (OASIS). The Next Generation Electronic Promotion Council of Japan (ECOM) played a leading role in the introduction and popularization of ebXML in Japan, and in March 2007 it released its Report on the Promotion of Next Generation EDI through ebXML. As a potential high-level scenario for the future of ebXML and the popularization of this specification, the report stated that "it will become increasingly technologically integrated with other next-generation standards, such as for web services, and it will enable them to be managed more cohesively and interactively." [6][7][11]

A major reason why different enterprises have problems sharing information is the significant cost of constructing systems to achieve this task. Costs can be roughly divided into two categories: those based on technological factors, such as infrastructure for EDI operation and costs for the introduction of platforms and application programs; and those based primarily on business factors, including the integration of product codes between businesses and mapping.

3.3 Research Taraget

When we look at recent developments in information communication technology, we can see that the methods to link data that utilize Web APIs are increasingly being used. By using Web APIs, linking data, resources and collaboration among enterprises is possibly achieved at a comparatively low cost. In the followings, RESTful catalog management system is introduced and the development of EDI feature over the system is discussed.[9]

IV. CONTENTS MANAGEMENT SYSTEM

We have developed a contents management system, named *Cataraku*.[12] [13] This software system adopts a common RESTful architecture for the resource management. Our proposed EDI feature is implemented over this software system. The explanations of the system configuration and scenario for data/resource linkage are given below. For building up the linkage between resources, so called mashup technology is used. We are going to explain the method shortly.

4.1 The elements comprising the system References

First, we defined a model for the basic information processing environment within which we would build an EDI system that uses a Web API. We assumed an information processing environment from the general Web for the system and that it would be comprised of the following three elements.

1) Resources, such as product information accumulated in the database

2) Functional units to utilize the Web API

3) Mashup functions to realize EDI

Each of the elements was considered to carry out the following types of processing. This kind of information processing environment is a prerequisite system component element for the so-called Private Cloud information processing environment. [22] [23]

1) Resources: databases and their management. The product database reference information is interfaced with the high-ranking middleware level through the API. For example, if various enterprises are using different coding systems to manage the same product, the function to publicly disseminate these "codes" in order to unify them is positioned in this level.

2) Functional units: provides the middleware function to utilize the Web API. It is equivalent to Pipes when using the Google API; for example, it is called the "Gadget" in document [23]. The EOS function and EDI are realized through combining this function with the API and then carrying out the necessary processing. The transfer of information by middleware is described by XML.

3) Mashup: provides a function to create business applications. The individual Web APIs are combined together to create applications in accordance with each of their specifications and regulations for use.

The processing on the Web to manage the applications that are created is defined as a role of this element, as in the majority of cases the users will have a new information input-output interface on the Web.

4.2 Examples of Mashups

In order to illustrate possibilities of EDI over Web API, take, for example, an idea to establish a home-delivered pizza shop on the Internet. Suppose there is a small pizza shop that is famous for making delicious pizzas. Let's say that it is a family-run shop and is therefore unable to deliver pizzas. Upon receiving an order, the shop can make delicious pizzas on time, but it cannot deliver them.

Meanwhile, another person gets the idea to open a virtual pizza shop on the Internet. Functions necessary for a virtual pizza shop are "delivery service" and "settlement service" in addition to "making pizzas at a specified time in accordance with an order." The delivery and settlement services are to be provided by delivery companies and financial companies.

By combining "requests" and "responses", it is possible to operate a new virtual pizza shop that has an added value of "capable of delivering to distant places." In other words, the new virtual pizza shop can sell delicious pizzas from existing pizza shops to people in larger areas by simply manipulating data linkage.

4.3 EDI scenarios

The phrase "EDI operation" encompasses both a wide range of processing content and differences in the complexity of the operation as we looked in previous section. For the time being, we shall avoid a discussion on complicated processes and provide an easy-to-understand example of data linkage between affiliated enterprises under the following scenario.

1. A "manufacturer" positioned in the upper level of the distribution value chain uses Web API to provide information about its own products in a form that can be accessed by others for reference.
2. A "dealer" in charge of sales has a catalogue of the brands his company offers in order that he may deal with the products of multiple manufacturers. The information used to create the catalogue is referenced to the product information provided by the manufacturers via the Web API.
3. The dealer implements product-related processing, such as inventory management, price management, and delivery and distribution. The EDI function, which is the topic of research for this article, is one of the functions available in the appendix.

A specific example in this scenario would be that the manufacturer provides the Web API product database, while the dealer provides the Web API offering sales management functionality–such as inventory management, price information, and payment clearing–and these functions then have to be linked.

The reality is that the distribution processes may be more complicated than this, and this complexity cannot be covered solely by this scenario. However, this example includes the basic points necessary in order to use Web APIs in EDI. Specifically:

1) Three enterprises or more will be taking parting in the e-commerce transaction, including the product buyers.

2) The information necessary for the transaction to take place will be made up of data provided by two or more enterprises.

3) This information can be obtained from the Resource, mentioned above.

4) The referencing of this Resource is carried out through Web APIs.

5) A platform is used to combine the Web APIs.

Below, we use the so-called Cloud Object design concept to attempt to structure the data-linked units to use Web APIs. The requirements for these structured units can be said to be included in conditions 1) to 5).

4.4 Functional Unit for EDI

In order to do a mashup by using multiple Web APIs, a software development environment needs to combine more than one Web API. For instance, Google makes its development environment, called "Google Web Toolkit," available to general users. Among other development environments are Pipes (Yahoo! Inc.), Mashup Center (IBM Corp.), and ToofTop (SAP AG). In Japan, the Intramart of NTT Data Corp. provides similar functionality[20]. These development environments are equipped with graphical operation screens, making it easy to combine available information resources via Web APIs.

Fig.1 shows a part of API functional unit of the *Cataraku* system. Users mash up resources through the interface. EOS (electronic ordering system) features are also available so that catalog system is able to utilized as a platform of e-commerce and the Web API can be reusable for other sites like books information provided by Amazon.com, flexible construction of e-commerce environment is achieved.

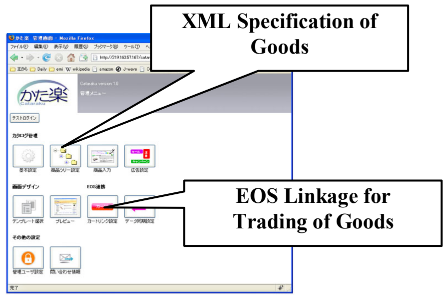

Figure1 Functional Unit of *Cataraku* System

4.5 Definitions of "Cloud Object"

In this article, "Open EDI" refers to the open EDI structure created from the Web API we advocate. The objects referred to are named Cloud Objects in order that we may hypothesize an operation environment such as the Private Cloud.

Table 1 outlines the definition of the Cloud Object, which includes Resource and Operation; each has the functions stated in the table.

The Definition of Cloud Object		
Resource The data used by this Cloud Object. Expressed in a URL.	*Internal Resource*	Exists locally (in the same domain) within the Resource.
	External Resource	Exists remotely within the Resource.
Operation The API provided by this Cloud Object. Business functions.	*Resource Direct Access Operation*	Within the Operation, provides a simple CRUD operation with regards to the Resource. Has the reference system (R) and the update system (C/U/D). In the update system under the production environment, it is necessary to prohibit calling within conventional access controls.
	Resource Simple Query Operation	Within the Operation, provides simply enquiry operations with regards to a single resource.
	Resource Complex Query Operation	Within the Operation, various other enquiries pertaining to the Resource are carried out like SQL SELECT. In Cloud Object = Function, Resources can be accessed within the range declared "usable". This Operation is unconditionally default setting when Cloud Object=Function is "configured". It is a Reference system Operation.
	Resource Lifecycle Management Operation	Resource Lifecycle = installation to progress transaction processing. For instance, "Cancel an order" and "Deliver,". It is necessary to describe the logic specific to this Operation. An Update system Operation.

Table 1 Definition of Open EDI Resource

First, a Cloud Object does not express a physical item in the traditional meaning of the word "object." Instead, in addition to encompassing actions (logic) that relate to operations, it also refers to the model created from the data necessary for that action.

For example, it's the model starting with the action of "ordering."

Next, the reference system-related "Resource Complex Query" Operation and the update system-related "Resource Lifecycle Management" Operation have the following characteristics; for the API in the Complex Query, the Resource directly provides, while for the Lifecycle Management the Function provides the installed logic.

4.6 Web API Specification

Web API for *Cloud_Object* API is based on REST design paradigm. Requests to API is structured and represented by XML documents and communication procedure uses URL and HTTP method. Management is based on "Digest access authentication". Primary, user should resister ID/PWD in Control page. In the header sentences of HTTP requests for API access, there should be sentences of Digest Access Authentication with the ID/PWD. In the case of failure, return variable is a sentence of failure announcement.

About the XML model procedure, there defined three element classes in XML representation for *Cataraku* API.

-itemgourp: Representing a group of items

-item: Goods or item of products so forth

-reference: links to the external resource of information

In case of {class="itemgroup"}, children element is able to nest descendent elements. Parent elements are indexed with pid attribute. While in case of {class="item"}, with the id as an reference parameter, user can invoke iteminfo(doGet()) API in order to refer item information(iteminfo). A part of example XML is shown in the figure 2 below.

```
<?xml version="1.0" encoding="UTF-8"?>
<cataraku xmlns="http://www.onezero.co.jp/cataraku/ns/" version="1" level="1" >

  <element class="itemgroup" id="1" pid="0" label="商品">
    <children>
      <element class="itemgroup" id="2" pid="1" label="小ねじ">
        <thumbnail>/cataraku/test/110_img/g_0.jpg</thumbnail>
        <description>一般的に使用されるねじです。</description>
        <children>
          <element class="item" id="4" pid="2" label="プラスチック材ポリカボ十字穴付なべ小ねじ">

            <thumbnail>/cataraku/test/110_img/neji_0002_a.jpg</thumbnail>
            <description>neji_0002</description>
            <orderNumber>PPN </orderNumber>
          </element>
          <element class="item" id="3" pid="2" label="十字穴付なべ小ねじ">
            <thumbnail>/cataraku/test/110_img/neji_0001_a.jpg</thumbnail>
            <description>neji_0001</description>
```

Figure 2 Example Web API Specifications of *Cataraku* System

4.7 Discussions over "Cloud_Object"

In this section, we consider the novelty of the proposed Open EDI through a comparison of related systems. First, as part of software engineering's so-called "distributed object orientated technology," the global industry organization Object Management Group (OMG) defined the specifications for the Common Object Request Broker Architecture (CORBA) standard. It encapsulates the concept of combining the various software parts distributed on networks. Other standards include the Java Remote Method Invocation (RMI), which is implemented in the Java program. The aim of the systems using these standards is to provide a solution for the problem of distributed execution at runtime. Therefore, the purpose of the data linked across several enterprises was not assumed in advance.

Next, platforms such as "salesforce.com," which have recently enjoyed a dramatic increase in users, have pushed the concept of SaaS (Software as a Service) to the front of its service, and aimed to increase user convenience by providing services offered by third parties on a common platform. The form that it is assumed this kind of service will take is basically created on a person-by-person basis and does not intend for shared, linked data to be actively utilized.

Conversely, the concept behind services such as "Yahoo Pipes" is that the output of one service is used as the input of another. Through this, it is considered that the services of different enterprises can be linked, but that no particular model will be created for the supplier of the service.

With regards to the above, the novel aspect of the Open EDI advocated in this article is the modeling of each enterprise involved in EDI and of each of their API service design departments through the Cloud Object concept. Through the mutual provision of API "mashup" services, these Cloud Objects as a whole become a system design that constitutes a single EDI application. In "Appendix", Java/XML interface of Cloud Object is described in detail. It is difficult to explain implementation in detail in this limited amount of pages, readers may obtain the flavor of our Cataraku System.

V. VISION OF API PLATFORMS

Among information systems designed to share services, attention has focused on the so-called design concepts of SOA (Service-Oriented Architecture) and SaaS (Software as a Service). [7][8] In addition to creating a wide variety of information processing functions within enterprises based on these concepts, attempts have begun to construct platforms that are designed to link the services of multiple enterprises. Specific examples of this include the European Union's (EU) SecSE R&D project. [2][3] Also, within Japan there has been progress made in building an environment that utilizes Web service technology, such as the establishment of the Manufacturing XML Promotion Conference and the Electronic Commerce Promotion Council of Japan (ECOM). [7] Below is an outline of a mechanical parts distribution environment once the functions that the platforms need to provide to link the services have been established.

5.1 The functions that the platforms need to provide

In this section, the general environmental conditions necessary in the environment where the technology will be used are first examined. When such a platform is created, the functions that are required include the provision, use, and mediation of services. If we consider these functions in terms of units of utilization for individual services, then generally speaking, the following three elements will be involved in each of the services and the creation of an SOA environment.

Service Consumer : the party who uses service (SC)

Service Provider: the party who provides service (SP)

Service Broker: the party who mediates between the above-mentioned parties (SB)

Specifically, as a B2B environment is supposed, then among customers who are from within trading company groups rather than the final consumer, then we can suppose they will be enterprises that are using information services provided by other enterprises, such as a service to provide inventory listings. Further, in order to create this kind of environment, the following two parties and the roles they play are crucial.

Service Developer: the party that develops the API for the service (SD)

Service Integrator: the party that creates the new service through a mashup (SI)

Up until the present time, there has been considerable IT investment in creating databases for product catalogues and facilitating EDI within enterprises, or between an enterprise's head office and its branch offices. Even among medium-to-small enterprises, there are companies that have been actively investing in IT and that have already created a range of databases and information-provision services. It is assumed that this kind of service would be shared through an API common interface, and to achieve this, the existence of the enterprise that plays the role of the service developer is absolutely vital. At the initial stage of introducing SOA, it is considered that the service-supplier enterprise will undertake this SD role.

Next, the service integrator creates a new business based on the service being provided as an API. At the initial stage, this may also be carried out by the same party as the SB. But if we consider the need for business impartiality, the SB should as much as possible remain independent, while the SI, which has clear business objectives, coordinates its activities with the SP and SC. It is considered that this model for the provision of the new service has significant potential for the future.

In order for this kind of environment to be cultivated in a specific business area, in the case of Japan it is necessary to consider the major role played by medium-to-small enterprises. It is important that the cost of entry is relatively low to enable the SP and SC to participate in platform operations. From the perspective of achieving this goal, APIs represent a promising component technology.

5.2 Data Commons in each business domain

Design concept described in the previous section is ultimately nothing more than a description of a theoretical framework and is nothing more than an ideal. Based on this perspective, in this section, method of how to actually realize an API platform will be discussed. The enterprises that will be considered will be industrial machine designers and several distribution companies that deal with materials (hereinafter referred to as Material X) that are frequently used in the manufacturing field.

It is thought that the information infrastructure model, which uses API technology to share information processing functions for business purposes, can be created by differentiating its functions in roughly three levels.

First, the bottom level is the basic information infrastructure to enable the Internet environment to provide an "API platform (hereinafter referred to as PF)". The middleware, central level indicates the API layer provided by each of the enterprises; when required, "search" and other functions are provided in this level. This PF is the SOA environment that the "service mediator (SB)" described in the previous section provides, and can also be described as the "shared services bus." In the uppermost level, which provides the "business applications", services for business are created that presuppose information sharing between enterprises. This layer provides the applications that will be explained in detail in the next section. It is necessary to prescribe a Service Level Agreement (SLA) to provide for application interoperability and to have rules for interface descriptions to use the API functions. Currently, the variety of standards relating to Web services formulated by W3C might be effective for such purposes. However, even within these technological systems, just as with UDDI it is considered that a considerable amount of time will be required for the functions to automatically realize even service discovery. At this point, the area dealing with "service discovery" accommodates by listing in details the introduction of specific enterprises that accommodate the business position of the functions offered by APIs, and the qualifications of the available enterprises. In this way, it is considered that automatic service discovery will be replaced by a method by which each person responsible for service use within their enterprise investigates and then selects from a choice of services.

VI. Concluding Remarks

In this article, we have described EDI over Web APIs, explained mashups, a technique to combine them, and talked about software design pattern, *Cloud_Object* over *Cataraku* System.

At present, the modes of use of information systems that are collectively referred to as Web 2.0 or cloud computing has been expanding, and information system design concepts, such as SOA and SaaS, have become widely used in the field of software development. Under such circumstances, it is important to be able to link data and services flexibly via the Internet. Mashup is a technology to realize this via Web API. The term "mashup" may be nothing but a passing keyword in the field of IT. This is because a framework to effectively link different organizations' data via the Web has matured. Since this is expected to become a new trend in the development and operation of corporate information systems, trends in software technology merit attention. If a mashup-like technique comes to be utilized in the business field, its advantage of being able to provide new functions in a short period of time at low cost will prove to be effective in increasing business efficiency, especially for small and medium-sized enterprises.[17][18]

In order to create added value by linking data, it is necessary to accumulate valuable data and make them available in such forms as Web API. Therefore, as a measure to promote the creation of new services by mashup, it is first necessary to encourage data accumulation in individual business areas. It is true that databases have been constructed for various purposes. However, in order to link them with new services, it is necessary to promote a wide use of such databases by releasing them in a recyclable way.[16]

Moreover, in order for a data linkage to function effectively in a specific business area, "linkage rules" that can be commonly used in the business area must be in place.[21] In the case of using mashup technology, the rule is XML schema definition. For instance, in order to implement EDI (electronic data interchange) for the distribution of products of a certain field, it is necessary to develop XML schema for shared use by companies involved. For expanded use of XML schema, it is effective to provide supports to business bodies' data-linkage and service-linkage attempts, in addition to implementing deregulation measures on open information. In doing so, for the sake of technical innovation in this field, it is important for the process of the development and utilization of schemas to be open.

References

[1] W3C: http://WWW.W3c.org/

[2] Website for FAST project: http://fast.morfeo-project.eu/

[3] SecSE: Service Centric System Engineering, http://WWW.secse-project.eu/

[4] Yohei Yamamoto, Leonard Richardson and Sam Ruby, "RESTful Web Services", O'REILLY, ISBN978-4-87311-353-1, (Japanese)

[5] IBM Japan, "Enterprise Mashup"; http://www-06.ibm.com/jp/provision/no59/pdf/59_article4.pdfG.

[6] Manufacturing XML Promotion Forum of Japan: http://WWW.mstc.or.jp/mfgx/index.html

[7] Electronic Commerce Promotion Council of Japan: http://WWW.ecom.or.jp/

[8] Toshiharu Aoki, ed., "Web Service Computing," Institute of Electronics, Information and Communication Engineers (2005)

[9] Masao Matsumoto, ed., "Management Information Technology in the Web Service Era," Institute of Electronics, Information and Communication Engineers (2009)

[10] "API & Mashup Method," SE Editorial Department of Shoeisha (June 2007)

[11] Akihiro Fujii, "Expanding Use of Web API-Vast Potential of Mashup-"; Science and Technology Trends (January 2010), National Institute of Science and Technology Policy, Ministry of Education, Culture, Sports, Science and Technology

[12] Akihiro Fujii, "API Platform for Machine Parts Distribution EAI", Study report EPI44-3 (June 2009), Information Processing Society of Japan

[13] Akihiro Fujii, Maki Nakayama, Koichi Tanaka, Kunihiro Nagamura, "EDI Support System over Web API", Study report EPI48-9 (June 2010), Information Processing Society of Japan

[14] Meriji de Jonge et al, "eServices for Hospital Equipment," Service-Oriented Computing-ICSOC2007, Springer, LNCS 4749 (2007)

[15] http://WWW.programmableWeb.com/

[16] Ministry of Economy, Trade and Industry, "SLA Guidelines for SaaS" (January 2008)

[17] Hiroki Fukui, Akira Iwata, Kimitake Wakayama, Shunyo Suzuki, "Proposal for Integrated Public-Private Electronic Application System Utilizing Mashup Technology"; Study report CSEC (2007). 2007-DPS-130, 2007-CSEC-36, Information Processing Society of Japan

[18] Koushiro Yuhki, Jiro Mimura, Yoshikazu Ueda, "Development of Mashup Construction Support System," Study report SE (2008), 2008-SE-159, Information Processing Society of Japan

[19] Masao Mori, Tetsuya Nakatoh, Sachio Hirokawa; "Mashup Resources and Mashup Glue"; Information Processing Society of Japan, WebDB Forum 2008, 2A-1

[20] Takayuki Kato, Yuki Sakuma, Ryosuke Sekido, Yudai Iwasaki, "Web API Practice Reference Book," Mainichi Communications (May 2007)

[21] "Mashups Interoperability and eInnovation"; Research materials of Harvard University Law School; http://cyber.laW.harvard.edu/interop

[22] Till Janner et. al. "Patterns for Enterprise Mashups in B2B Collaborations to foster LightWeight Composition and End User Development", IEEE International Conf. on Web Service (2009)

[23] Alessandro Bozzon et. Al "A Conceptual Modeling Approach to Business Service Mashup Development", IEEE International Conf. on Web Service (2009)