SESSION

WEB SERVICES AND APPLICATIONS

Chair(s)

Extending iTV Applications through Web Services

A. Rodrigo C. M. Santos[1], B. Marcio Ferreira Moreno[2], C. Felippe Nagato[1] and D. Luiz F. G. Soares[2]
Pontifical Catholic University of Rio de Janeiro
22453-900 Rio de Janeiro, Rio de Janeiro, Brazil
[1]*{rodrigocosta, fnagato}@telemidia.puc-rio.br*
[2]*{mfmoreno, lfgs}@inf.puc-rio.br*

Abstract - *This paper discusses the use of web services in supporting broadcasted iTV applications. More precisely, we are interested in applications to improve the effectiveness of service delivery in the public sector. The focus is on social services for which the iTV applications can be transmitted and stored previously (occasionally long before) their running. The paper explores the capabilities of web servers to extend the processing power of such applications. Instead of developing an architecture specialized in just one type of service, as the solutions existing today, a Web Proxy that can hold an extensible set of web proxies is proposed. Each web proxy in the set is connected to web servers whose peculiarities can change over time without affecting the broadcasted applications. As a use example, the paper presents an early warning service describing the implementation of the Web Proxy and iTV application prototypes.*

Keywords: Web Services, iTV, Ginga-NCL, Lua, ISDB-T.

1 Introduction

Most of digital TV (DTV) systems allows broadcasters to transmit applications to run on TV receivers (set-top boxes, TV sets, etc.), along with the main audiovisual content. These software products, usually designed to enrich (through user interactions) viewer experience when watching TV, are known as interactive TV (iTV) applications. To be able to interpret and run iTV applications, receivers should implement an intermediate software layer, called middleware, responsible for providing hardware and software independence to applications and to support them, through a well-defined application programming interface (API).

The key aspect of terrestrial TV systems is their ability to broadcast content to a substantial amount of people. Concerning scalability of content delivery, terrestrial TV is more effective than Internet-based systems. This effectiveness makes terrestrial digital TV systems the ideal environment for deploying social services, especially for those whose access to information is still difficult, in particular, in zones with scarce broadband access.

DTV receivers' hardware is usually less powerful than computers' hardware. This is particularly true for low-cost receivers targeting low-income people. Developers of social services should consider this issue when designing iTV applications. An approach to overcome this barrier is to explore the capabilities of servers (or proxies) to extend iTV applications processing (taking advantage of terrestrial TV reach and the computational power of servers).

Following this approach, the use of web services to extend the functionalities of iTV applications is promising. Web services (an implementation of SOA – Service-Oriented Architecture – principles) provide interoperability between applications. Over the last years, we have saw several business migrating their communication model to use web services. More recently, web services have also been used to integrate standalone applications with cloud data or applications. In this paper we want to explore this last aspect.

This paper focuses on discussing the use of web services in supporting broadcasted iTV applications. More precisely, we are interested in iTV applications to improve the effectiveness of service delivery in the public sector. This includes the delivery of government, health care, educational, social care, and early-warning information and services to the citizens. It should be stressed that we are not concerned with situations that demand an immediate reaction, like an immediate evacuation of some place. We are interested in preventive applications instead of reactive applications.

An early-warning application is used as an example. The application has been designed to warn residents of risk areas about possible natural disasters. Architectural issues on deploying this class of applications are stressed when presenting the relevant aspects of our implementation. The idea is to provide a generic solution and not a particular solution for a specific application.

The rest of the paper is organized as follows: Section II presents and discusses some related works. Section III considers some architectural issues and presents an overview of the developed architecture. Section IV introduces the relevant aspects of our implementation and how iTV applications can access the deployed web services. Section V describes an iTV application worked as a use example. Finally, Section VI presents our final remarks and points out some future works.

2 Related work

Since the beginning of digital TV systems, their integration with the Web seems to be natural. Most of DTV standards offers some API to enable iTV applications to access the Web. However, as Tsetsos et al. [1], we believe that the convergence between the Web and iTV applications is much more than simply allow viewers to browse the Internet.

Tsetsos et al. have proposed an architecture to integrate iTV applications and web content using semantic inferences. The metadata broadcasted together with the audiovisual content is parsed and compiled into an ontology that goes through an inference machine. The system retrieves web content that matches user's profile and is related with a current TV show. Although promising, this system requires a powerful TV receiver both to run the iTV application and to access a broadband network. Besides, it adds some components in the system that are not part of the DVB standard, which they take as basis. To actually apply the proposed solution, significant changes have to be made on receivers (client sides).

Fernandes et al. [2] have proposed an extension to Ginga middleware [3] to provide support to applications that may take into account user-location information to adapt the content presented. In the proposal, the client iTV application communicates with a server application via a gateway. The architecture allows different accesses to the gateway, among them through a SOAP web service. The client constantly posts its location to the server. The server stores user locations and performs inference rules, based on this contextual information. When a set of conditions is satisfied (e.g. the client arrived at some specific place) actions may be triggered. The gateway acts as a specialized proxy to provide just one type of service, and cannot be extended. In contrast, we can say that our proposal generalizes the Fernandes et al. approach. We have designed a generic proxy capable of communicate with several web services and to be extended with new services.

Kulesza et al. [4] propose a model-driven approach to facilitate iTV applications development. This approach divides the authoring phase into four steps: the definition of the application requirements; the generation of an abstract template; the transformation of the abstract template into a concrete template; and finally, the generation of the source code. One of the templates facilitates the authoring of applications that consume web services. The authors do not discuss how costly the access to web services can be for TV receivers (and how to implement it). An eventual approach could be using the framework proposed by Kulesza et al. to generate applications that could use the architecture we propose to access web services.

Companion devices have recently gained much attention in DTV systems. Companion screen applications (also named second screen applications) refer to TV-related applications specifically designed to be used while watching TV on the main screen. In these applications, secondary devices – such as smart phones, tablets, and laptops – are used to search, interact and engage with a rich variety of applications, websites and communities while watching the main screen. Companion screen applications can be broadcasted or downloaded by other means [5]. They can be considered a kind of distributed iTV application that runs, at least partially, on secondary devices. Secondary devices can access web services, in almost all cases providing social network services [6]. Proxy servers are sometimes used to provide specific services, for example, to support fingerprinting technology.

Audio fingerprinting [7] consists first in taking an imprint (also referred as "signature") of the audio track of video contents and store it into a database. The client iTV application takes the same type of audio imprint on short timescale and sends it to the proxy server, which then searches for it into the stored imprints collection and returns the content URI when it is found. This technique has been used to synchronize the main audiovisual content of a TV program with content coming from the Internet [6]. It requires a huge dedicate server infrastructure to compare the signatures and has some drawbacks as discussed in [7].

Companion screen applications access web services via secondary devices. The result coming from the server is presented on the companion screen. When they use a proxy server, the server uses proprietary solutions and is limited to a specific goal. In contrast our proposal does not rely on secondary devices to display content coming from web services; the proxy server in our architecture is a web service that can be extended to provide access to several web services.

There are some other services designed to be accessed through an iTV application targeting a specific goal. The iTV SMS service is an example that was deployed in some countries in Europe. In [8], Zelenkauskaite and Herring study how users interact with each other using this service, and compare it with traditional Internet-based chat applications. Another example is a traffic application that enables users using mobile TV to obtain information about traffic jams [9]. Several other works have been made extending iTV application features to access extra content on the Internet. However, they usually propose specific solutions for particular applications and cannot be extended.

As for applications targeting social services, during the earthquake of 2011 in Japan, the importance of having a good EEW (Earthquake Early Warning) system becomes evident. Three transmission methods can be applied to broadcast an EEW as soon as it is issued in the ISDB-T [10] system: using an asynchronous packetized elementary stream (PES) for superimposing a caption; using data carousels sent in advance, whose data is displayed when triggered by an event message also broadcasted; or transmitting it on a pre-defined auxiliary channel. In this last case, the transmission of an EEW can activate receivers and make people aware of it, even if they are not watching the TV program.

Peru has also adopted the ISDB-T standard but with the Ginga middleware: an ISDB-T profile adopted in almost all Latin-American countries. The Japanese EEW solution is being adapted to supply the needs of Peru. In both Peru and Japan cases, the EEW system does not access the Web, even when it is transmitted in a carousel as an iTV application. EEW is an emergency system that focuses on immediate reaction. It only searches for massive reception of disaster information with minimum delay. A goal different from the one treated in this paper. In fact, both solutions could be used together in an alert system that comprises both preventive and reactive applications.

To the best of our knowledge, architectural aspects for accessing web services through iTV applications have just begun to receive its deserved attention. In this paper, besides proposing our architecture, we discuss some other approaches and point out the advantages and disadvantages of each one.

3 Architecture Overview

Since DTV receivers are usually resource-constrained systems, heavy processing routines can impose some delays in presentations. This is especially true in applications that access external web resources.

When iTV applications access web resources/services directly, besides the aforementioned processing problem, there is another drawback that must be stressed. If an iTV application is broadcasted for late running and if the web service interface changes or the service is replaced (for one better or because the former is not available anymore) before running time, the iTV application will no longer work. In other words, for each change in services (either in its interface or in its functionality) the broadcaster has to rewrite and broadcast the updated version of the application. Considering resident applications, (i.e. built-in applications) update them can be harder. Since iTV applications designed to social services usually are transmitted and stored previously (occasionally long before) their running [11], this is a focal point when providing an architecture to support them.

With a proxy server mediating the communication, part of the heavy processing can be done outside the DTV receiver. Of course, the access to the proxy should be easy otherwise this alternative is worthless. With a proxy, it is also possible to change services and maintain iTV applications working. Of course the proxy must guarantee that both its interface and its response format do not change when web services changes.

If the proxy is implemented as a web service, there are two common techniques to access it: using SOAP or REST. The SOAP (Simple Object Access Protocol) [12] protocol is based on XML and can be encapsulated by several transport protocols. SOAP has a drawback: it is necessary to go through some additional protocol layers and to parse complex XML documents to be able to make even a simple call (thus, SOAP web services are considered as having heavyweight calls). On the other hand, there are REST (Representational State Transfer) [13] web services, whose access is performed using HTTP methods and an URI to identify the resource (service) of interest. REST web services invocations have less processing overhead than when using SOAP.

However, using proxy has some drawbacks. First, it can increase the communication latency. Considering just message passing, it is faster consume web services directly than call a proxy as intermediate. This is not a negligible aspect when the available network bandwidth is narrow. Second, to deploy iTV applications accessing the proxy, not only the iTV application must be designed but also the proxy. Finally, the infrastructure in which the proxy is deployed has to be effective, since TV broadcasts can lead to a scenario of millions of clients concurrently accessing the proxy.

The architecture proposed in this paper uses proxy approach to access web services given its advantages, especially to iTV applications targeting social services. Figure 1 shows the overall workflow and message passing between the architectural components. The workflow begins (1) with a broadcaster sending to TV receivers an iTV application along with its audiovisual content. The application embeds a call (2) to the Web Proxy, which is a key component in our architecture. The Web Proxy is the interface between interactive applications and the web services they consume. When the Web Proxy receives a request, it calls (3) the appropriate web services to meet the request and waits for responses. Once the responses arrive (4), the Web Proxy processes them to an appropriate format, easier to handle by iTV applications, and forwards the result back (5). At this point, the application may present the result.

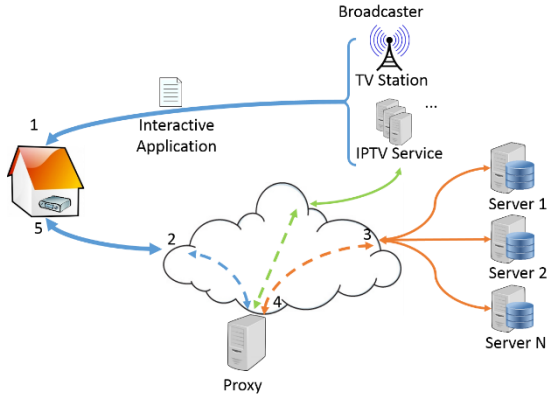


Figure 1. Overall workflow.

Note that until now we are considering that all TV viewers have a network channel available to communicate with the Web Proxy. In an IPTV scenario this assumption is reasonable. However, in a terrestrial or cable TV system this assumption is not always true. An alternative is broadcasting iTV applications that can adapt their presentation to display default content received from the broadcast channel. The broadcaster could communicate with the Proxy Server in order to obtain this default content to be broadcasted and presented to those viewers that cannot access the proxy.

As a simple example, consider a broadcaster sending an advert of a bookstore. If the TV set has a broadband channel, the application could access the Web Proxy and present the returned address of the branch closer to his/her home. Otherwise, the application could present the broadcasted headquarter address. The example presented in Section V explores this feature.

4 Proxy Architecture

Instead of developing an architecture specialized in just one type of service, we have designed a Web Proxy that can hold an extensible set of web proxies, each one providing different services. To avoid misunderstandings from now on, when we use the term "proxy" in this paper, we are referring to software components that access web services; when we use the term "Proxy Server" or just "Server", we are referring to the architecture developed that hosts a set of proxies.

The Proxy Server has a module called Proxy Manager. This module receives requests from clients and forwards them to appropriate proxies. Figure 2 presents schematically the internal architecture of the Server.

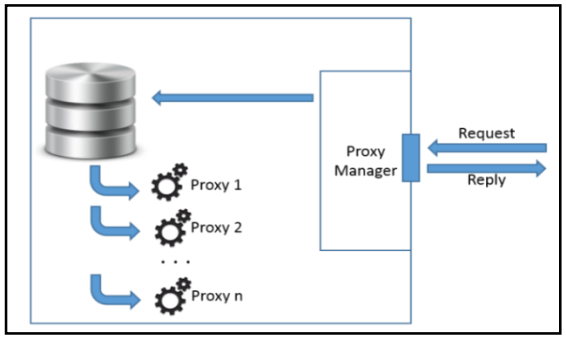


Figure 2. Proxy Server internal architecture

As aforementioned, an important issue is deploying lightweight communication API from the Proxy Server to iTV applications. This API should be complaint with the DTV middleware's API. In our architecture, clients communicate with the Server through a RESTful API. The choice of REST meet our requirements of both interoperability and lightweight calls. Listing 1 presents the base RESTful API skeleton.

```
http://{server:port}/ITVServices/{ProxyAlias}/{params}
```

Listing 1. Base URL skeleton.

Each hosted proxy is bound to an alias that associates it to some type of request. For example, if the Server receives a request as the one illustrated in Listing 2, the Proxy Manager forwards this request to the proxy bound to the alias "*AlertService*" (later this section details how the forwarding process is carried out).

```
http://{server:port}/ITVServices/AlertService/p1
```

Listing 2. Example of Proxy call.

The Server has an interface that allows for registering and removing proxies, and for changing (enable or disable) their status. The Proxy Manager relies on an XML document that describes registered proxies and defines which of them are enabled. Listing 3 illustrates an example of a XML document. Each proxy has four attributes: *id*, *alias*, *status* and *entryPoint.* When a proxy is registered, the Server assigns a random *id* to it. How a given proxy is known externally is given by the *alias* attribute. The proxy status can be set by changing the homonym attribute (only proxies whose attribute *status* is "*enabled"* are loaded). As each proxy is, in fact, a software package, the *entryPoint* attribute identifies the class within each package that implements the interface "*ITVProxy"* (the proxy entry point).

```
<proxyList>
  <proxy id="proxy1"   alias="Service1" status="enabled"
                          entryPoint="ClassService1" />
  <proxy id="proxy2"   alias="Service2" status="enabled"
                          entryPoint="ClassService2" />
  . . .
</proxyList>
```

Listing 3. XML document used by Proxy Manager.

Taking again Listing 2 as example, when the Proxy Manager receives this request, it parses the XML document looking for a proxy whose *alias* attribute matches the *AlertService* value. Once a proxy is found, the Server loads the proxy's entry point and calls the method *"runService"* (specified in the interface *"ITVProxy"*), passing *p1* as input. When the method returns a value, the Proxy Manager forwards it to client.

4.1 Accessing the Proxy Server through an iTV application

In order to illustrate how an iTV application accesses the Server to consume services provided by a hosted proxy, let us take NCL applications as an example, without losing the generality of our proposal. Taking NCL application as example will help us to explain the use example described in Section V.

NCL is the declarative language supported by the Ginga middleware. NCL and Ginga compounds the H.761 ITU-T Recommendation for IPTV services [14] and is part of the ISDB-T (International Standard for Digital Broadcasting – Terrestrial) profile adopted in most Latin-American countries.

NCL is a glue language that allows authors to specify spatiotemporal relationships among media objects without specifying the media objects' content. For example, if an author wants to specify an application in which an icon image is presented at some point, s/he has to define a <media> element as the one illustrated in Listing 4.

```
<media id="iconObject" src="icon.png"
                type="image/png" descriptor="iconDesc"/>
```

Listing 4. Example of media element in NCL.

An NCL <media> element has an id that identifies it uniquely in a document; a type attribute that indicates the type of its media content (image, text, script, etc.); a src that refers to a media content URI (Uniform Resource Identifier); and a descriptor attribute that refers to a <descriptor> element that defines the initial presentation properties of the content. In NCL, authors use casual relationships (represented by <link> elements) to express the entire presentation logic.

Ginga provides at least two approaches for accessing remote objects: the declarative approach, using the src attribute of <media> elements; and the imperative approach, using the NCL scripting Lua language [3]. In the first approach, the src attribute must specify the protocol used to get the object (Ginga accepts different protocols, including HTTP) and the remote object location. At a precise moment, Ginga retrieves the object and presents it[1].

The src attribute can contain an URI that accesses a RESTful web service, as shown in Listing 5. If the service answers with an object that the middleware can present (e.g., an HTML page or an image), it is perfect. However, if the answer should still be processed (e.g., the response is an XML document that must be processed in order to be present), this

[1] Ginga implements a prefetching algorithm [20] to avoid, or least minimize, presentation delays, keeping temporal inter-media synchronization as specified by application authors.

approach cannot be employed and we have to use the imperative approach.

```
<media id="iconObject" type="image/png" descriptor="ansDesc"
    src="http://{server}/ITVServices/{ProxyAlias}/{params}"/>
```

Listing 5. Media element referencing a web service.

The second approach, despite being more complex, is more flexible and powerful. It involves using Lua scripts to access the Proxy Server. Lua scripts are event oriented: when an event occurs (an input event, an event generated by other NCL elements, etc.), the Lua event handler is notified and receives a table describing the event. Lua scripts can also post events to communicate with other NCL elements and with Internet entities. To access the Proxy Server, a Lua script can, for example, post a TCP event and set a callback to handle the response.

Listing 6 shows a skeleton of both a TCP event posting and a table containing the answer, which should be treated by some Lua-event handler. The first excerpt in the listing is used to post a TCP event (e.g., an HTTP request). When the response for this event arrives, Ginga calls the appropriate event handler, and passes the evt table illustrated in the second excerpt. If the post calls a web service, the value field of the returned table contains the web service response that can be processed by the Lua script before present it.

```
1.  event.post { class      = 'tcp',
                 type       = 'data',
                 connection = <identifier>,
                 value      = <string> }
-------------------------------------------------
2.  evt = { class      = 'tcp',
            type       = 'data',
            value      = <string>,
            connection = <identifier>,
            error      = <err_msg>    }
```

Listing 6. TCP events in Lua scripts.

If the Lua script consumes a RESTful web service, the posted event must contain the whole HTTP construction to access the resource (e.g., GET /Service/params HTTP/1.1).

Let us now take a broad view of our solution considering other languages than NCL and its player Ginga. Some digital TV standards adopt HTML-like languages to specify iTV applications. DVB [15], ARIB BML [10], LIME [16] and HbbTV [17] are examples. All these standards support ECMAScript as HTML scripting language. Although it is possible to define an HTML <object> element to access a RESTful URL, a usual approach is delegating this task to scripts (performing web services calls imperatively).

As another example, SMIL [18], a W3C synchronization language standard, can also use the declarative approach of our proposal. As in NCL, SMIL documents do not specify their media contents. The language provides a set of XML elements (<text>, <audio>, <video>, <image>) to specify objects. Each of these elements has a src attribute that refers to the media content URI. Listing 7 illustrates three SMIL elements accessing three different services hosted in the Proxy Server. If the "VideoService", the "TextService" and the "AudioService" responses are, respectively, a video, a text and an audio, a SMIL player is able to present the content.

```
<par>
  <video src="http://{server:port}/ITVServices/
                                        VideoService/v1"/>
  <text   src="http://{server:port}/ITVServices/
                                        TextService/t1"/>
  <audio src="http://{server:port}/ITVServices/
                                        AudioService/a1"/>
</par>
```

Listing 7. SMIL application accessing the Proxy Server.

5 Early-Warning Application

Early warning services motivate this example. Let us assume the following scenario. A government agency (named NADP – National Agency for Disaster Prevention) responsible for prevention and mitigation of natural disasters has a monitoring system that is able to predict them based on geographic data. The success of such systems depends on massively warning nearby population. Due to the fact that much people have a TV set at home, and that the warning can be sent well in advance, iTV applications can be used to warning the population and interactively teach how to proceed, depending on their location (e.g., in case a torrential rain is detected, residents close to a hill should look for a shelter, while those that are in a safer distance should stay at home). We have implemented an NCL iTV application and a warning Proxy prototype to address this rain storm scenario (very common in mountain regions of Rio de Janeiro).

The iTV application calls the Proxy (AlertService Proxy) to obtain a custom map and a message according to its receiver location. The Proxy accesses the NADP web server and retrieves the coordinates of risk areas. With this data, it is able to access a map web service, get the map surrounding the risk areas, and highlight the locations with high probability of destruction and the locations that may be affected in a lower extent.

If the TV receiver does not have access to the Internet, the broadcasted iTV application presents a default map with the risk areas highlighted (this map is acquired accessing the Proxy before broadcasting the application).

In our implementation, we have emulated a NADP server to provide information about risk areas in the state of Rio de Janeiro. The AlertService Proxy prototype uses the web service of Google Maps to obtain a map representation of Rio de Janeiro city and to mark it, highlighting the risk areas. Finally, we have also emulated a TV broadcast station that transmits some audiovisual content along with the iTV application.

Listing 8 illustrates how broadcasters access the AlertService Proxy to obtain a default map. The Proxy is called passing the value "default_Rio" as parameter, indicating the request for the default map of Rio de Janeiro with risk areas highlighted.

```
http://{server:port}/ITVServices/AlertService/default_Rio
```

Listing 8. Accessing the Proxy to obtain the default map.

Figure 3 shows a snapshot of the application running in an environment without access to the Internet, in which the default map sent by the broadcaster is presented. Note also that the message being displayed is generic, since no adaptation was performed.

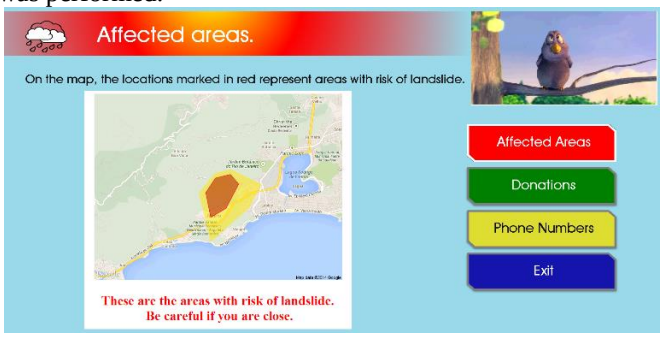


Figure 3. Default broadcasted map.

If the DTV receiver has a broadband channel, the iTV application accesses the Proxy passing its location as parameter: using ZIP code; or geographic coordinates (latitude and longitude). The first code excerpt of Listing 9 shows how to call the proxy passing the ZIP code as parameter, while the second one shows how to call it passing latitude and longitude, respectively.

```
1:  http://{server:port}/ITVServices/AlertService/{ZIP}
2:  http://{server:port}/ITVServices/AlertService/
                                    {Latitude}/{Longitude}
```

Listing 9. Accessing the Proxy to obtain a personalized map.

Because the Proxy answer is an HTML page, iTV applications can address it by means of the src attribute of an NCL <media> element, passing the receiver location. However, the same iTV application is broadcasted to receivers at different areas. Therefore, it cannot previously carry the receiver location. The NCL player keeps the receiver location in a global variable accessible to applications, named "user.location". So, in the iTV application, an NCLua object (an object with Lua code) gets this value and dynamically create a media object whose src attribute is a call for the AlertService Proxy, passing the receiver location as parameter. To dynamically create the media object, the Lua script uses another feature of NCL that allows for creating/editing applications on-the-fly (NCL Editing Commands [19]).

Note that clients are not aware of which services the Proxy actually accesses, and even if it accesses any service. Even if the Proxy eventually changes the Google Maps service to another one, like Bing Maps, which has a different API, the Proxy takes care of this, keeping the application unchanged.

Figure 4 shows a snapshot displayed to viewers within the risk areas, while Figure 5 shows the same application, but now presenting a snapshot seen by viewers in a safe area. Note in the maps that there is a sign indicating the geographic position of the viewer's receiver. Note also that not only the maps have been adapted, but also the messages below them are personalized. An extended version of this application also presents a button that, when selected, shows the escape routes if necessary.

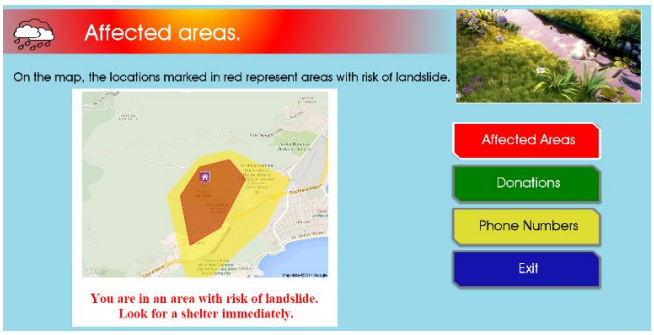


Figure 4. Snapshot on TVs within the risk areas.

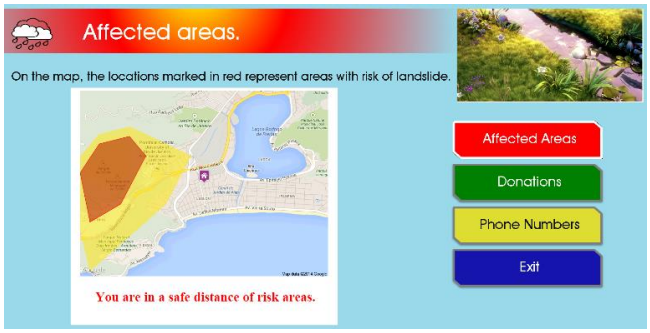


Figure 5. Snapshot on TVs in a safe area.

6 Conclusion and Future Work

The architecture proposed in this paper allows for iTV applications to consume web services using the proxy approach. The main advantage is to divide the processing with a more powerful server, and also to allow applications to remain working, even in the presence of changes in web services. These features match with requirements of social services, in which there can be TV receivers with low processing capabilities and applications deployed long before their running time. We believe that these advantages justify the use of a proxy even with the drawbacks previously discussed.

Perhaps the main difficulty in implementing our architecture is the provisioning of a powerful infrastructure capable of handling a large number of simultaneous requests. As future work, we intend to measure how our proposal scales. For the type of applications we are targeting (preventive applications), we presume that a small delay of the proxy answer is acceptable. When this is not the case, we want to explore the replacement of the Proxy Server by a network of Proxy Servers, carefully distributed, to provide access to the web services. In this case, adaptive iTV applications should address the most convenient Proxy. For some languages and players this is not difficult.

The Proxy Server should also be extended to maintain some statistics on access to their proxies. This can be useful to understand how effective the services are being provided and to redirect requests to another Proxy Server (in the network of Proxy Servers), if some loss in quality occurs.

A narrow bandwidth last mile access to the Internet can be another problem. There are social services that require considerable amount of data exchange. For instance, the application discussed in the paper transmits a map image along

with an HTML document. If a high quality map is returned, the network bandwidth can be insufficient. The same would probably happen in answers containing video or audio. In future work we plan to study some technics to mitigate the problem. First, the Proxy Server can be aware of the each client network and send to them content adapted to their context. Second, similar to the case in which there is no Internet access, the iTV application can choose to receive the default content coming from broadcasters. Third, extending the second method, iTV applications can choose to receive answers to their requests through accessing pushed broadcasted data. Aware of this scenario, some public broadcasters in Brazil already reserve part of their channel bandwidth to push data received from web servers to specific groups of clients.

However, few applications were developed, to allow us to have good answers to the previous shortcomings. The development of other applications especially targeting social services is also within our future work.

7 Acknowledgment

The authors would like to thanks FAPERJ (Foundation of Support to the Research of the State of Rio de Janeiro) and CNPq (National Council for Scientific and Technological Development) for supporting this research.

8 References

[1] V. Tsetsos, A. Papadimitriou, C. Anagnostopoulos and S. Hadjiefthymiades, "Integrating Interactive TV Services and the Web through Semantics," *International Journal on Semantic Web and Information Systems (IJSWIS)*, vol. 6, nº 1, pp. 1-18, 2010.

[2] V. R. M. Fernandes, B. J. G., C. S. Soares Neto, A. C. Paiva, H. F. Figueiredo and C. S. Baptista, "Integration Model for Location-Based Services in iDTV Applications," em *The Fifth International Conference on Advanced Geographic Information Systems, Applications, and Services* , Nice, France, 2013.

[3] ABNT NBR 15606-2:2011, "Digital Terrestrial Television - Data Coding and Transmission Specification for Digital Broadcasting - Part 2: Ginga-NCL for Fixed and Mobile Receivers - XML Application Language for Application Coding," 2nd ed. São Paulo, Brazil, 2011.

[4] R. Kulesza, S. R. L. Meira, T. P. Ferreira, E. S. M. Alexandre, G. L. S. Filho, M. C. Marques Neto and C. A. S. Santos, "A Model-driven Approach for Integration of Interactive Applications and Web Services: A Case Study in Interactive Digital TV Platform," em *Proceedings of the 2012 IEEE International Conference on Multimedia and Expo Workshops*, Melbourne, Aurstralia, 2012.

[5] L. F. G. Soares and C. E. C. F. Batista, "FSDMA Instantiation to Support NCL 3.1 Applcations in Ginga Reference Implementation," Rio de Janeiro, Brazil, 2013.

[6] Mediatvcom, "Decrypting the Second Screen Market," 2012.

[7] Audible Magic Corporation, "Digital Fingerprinting & Video Content Recognition, Enabling News Forms of Interactive Advertising," 2011.

[8] A. Zelenkauskaite and S. Herring, "Television-Mediated Conversation: Coherence in Italian iTV SMS Chat," em *Proceedings of the 41st Annual Hawaii International Conference on System Sciences*, Waikoloa, HI, 2008.

[9] L. F. Nascimento and F. S. Silva, "iTV-Traffic: A context-sensitive application based on mobile interactive digital TV for traffic condition informations access (in Portuguese)," em *Anais do Encontro Regional de Computação e Sistemas de Informação*, 2013.

[10] ARIB – Association of Radio Industries and Businesses, "Application Execution Engine Platform for Digital Broadcasting," ARIB STD-B23 Version 1.2, 2004.

[11] EBC, "Brazil 4:D: Socio-Economic Impact on Interactive Public Digital TV (In Portuguese)," Brasília, Brazil, 2013.

[12] W3C, "Simple Object Access Protocol (SOAP) 1.2," 2007.

[13] R. T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures," 2000.

[14] ITU-T Recommendation H.761, "Nested Context Language and Ginga-NCL for IPTV Services," Geneva, 2009.

[15] European Standard, "Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for digital terrestrial television," 2009.

[16] ITU-T Recommendation H.762, "Lightweight interactive multimedia environment (LIME) for IPTV services," Geneva, 2011.

[17] ETSI, "Hybrid Broadcast Broadband TV," ETSI TS 102 796 V1.2.1, November, 2012.

[18] W3C Recommendation, "Synchronized Multimedia Integration Language (SMIL 3.0)," 2008. Available: http://www.w3.org/TR/smil/.

[19] R. M. R. Costa, M. F. Moreno, R. F. Rodrigues e L. F. G. Soares, "Live Editing of Hypermedia Documents," *In Proceedings of the 2006 ACM symposium on Document engineering*, Amsterdam, The Netherlands., 2006.

[20] R. G. Rodrigues and L. F. G. Soares, "A Framework for Prefetching Mechanisms in Hypermedia Presentations," em *In Proceedings of Fourth International Symposium on Multimedia Software Engineering.*, 2002.

TrackShip – A Unified Shipment Tracking Application

Hari Krishna Bodicharla, En Cheng
Department of Computer Science, The University of Akron, Akron, OH, 44325
hb37@zips.uakron.edu, echeng@ uakron.edu

Abstract

Online shopping has made people's lives much easier. Customers can save a lot of time spent for shopping but when it comes to tracking orders we don't have any easy way to track and get the status of the orders. The customer always needs to visit the particular website through which he made the shipping and check the status. Also if the customer has several orders placed from different shipping services such as UPS, FedEx, and USPS, he needs to login different websites to track his orders which would be a huge burden. Trackship, a unified shipment tracking application, is mainly designed to help customers track their orders for different shipping services including UPS, FedEx, DHL, and USPS. Users can access Trackship on their smart phones, tablets, and computers. Using Trackship, the users can also calculate shipping services fees and compare shipping cost among different delivery companies. Last but not least, customers can also search for nearby location of drop boxes or post offices of package delivery companies using the "Nearby Location" service provided by Trackship.

Demo URL: http://cs.uakron.edu/~hb37/trackship/

1. Introduction

"We are drowning in data, but starving for knowledge"
- John Naisibitt

The invention of the Internet and the emergence of the World Wide Web revolutionized our daily lives. Today, we can order products just the way we want and receive them within days, even if the products require assembling parts from dozens of companies scattered around the world. Data integration has occurred in a variety of areas, including online shopping, job hunting, and health care services. Recently, there has been growing interest in integrating data from different services and providing a unified and full-fledged portal for Web users.

As Internet services are accessible on all kinds of devices including computers, tablets, and smart phones, online shopping has become more popular. Nowadays customers can easily browse different websites and buy products through secure payment services. Products are shipped to customers' doorstep by efficient shipping services such as UPS, USPS, DHL, and FedEx. Online shopping saves customers a lot of time but when it comes to tracking orders there is no easy way to track and get the status of the orders. Customers usually have to visit the particular website through which he made the shipping and check the status.

Furthermore, when a customer needs to send a package from one place to another, there are several factors that affect the decision about choosing shipping services. These factors include the availability of drop box locations around the customer's locality, shipping costs, and the availability of tracking delivery status. The main motivation of Trackship is to provide a unified interface to track packages from different shipment services, to locate nearby shipment services, and provide the details of shipment services based on package factors. In summary, Trackship aims at helping users effectively and efficiently make full use of different shipment services through one unified interface.

2. Trackship

Trackship is initially designed to integrate different shipment services including UPS, FedEx, DHL, and USPS. After realizing the potential of this project, we extend the Trackship application to provide shipping costs comparison and searching nearby services location. Users can access Trackship on their smart phones, tablets, and computers. The main interface of Trackship is shown in Figure 1. Trackship supports both traditional web browsers and mobile web browsers. For mobile web browsers, it automatically resizes its interface for the browser window using responsive actions of web application. The first service is shipping tracker through which customers can track their packages delivered by UPS [1], FedEx, DHL, and USPS [2]. The second service

is service analyzer through which customers can compare shipping costs among different shipment services. The third service is location finder through which customers can search for nearby location of drop boxes or post offices of package delivery companies.

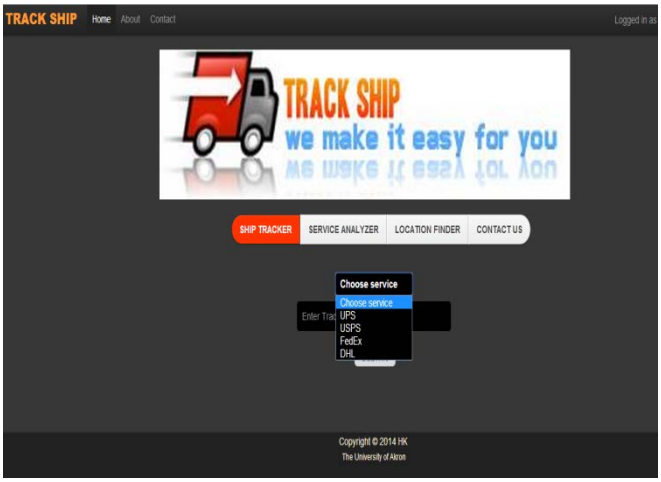


Figure 1. The Main Interface of Trackship

TrackShip gathers data in different forms from different shipment services. The architecture for Trackship is described in Figure 2. More specifically, TrackShip gathers XML responses from UPS, and HTML response from FedEx and DHL. As Trackship uses XML technologies to store and retrieve the location points for shipment services, it can pull the recent online search locations efficiently.

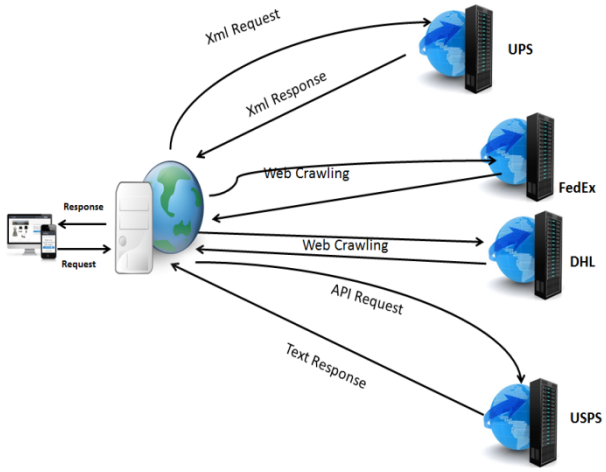


Figure 2. The Architecture of Trackship

3. Services

The main goal of TrackShip is to help users track their orders placed through different package delivery companies at one place. According to the main interface shown in Figure 1, Trackship provides three types of services, including Tracking Service, Shipping Service Analyzer, and Nearby Location Finder. All three services can be easily accessed at Trackship website. We illustrate the details of each service in the following sections.

3.1. Tracking Service

The tracking service at Trackship website provides an easy-to-use service for customers to track multiple orders from different shipping companies. Without Trackship, if a user has placed several orders through different shipping services, he has to login different shipping service websites and maintain multiple accounts to track his orders. With Trackship, this can be done much more efficiently. He can track his orders placed through shipping services including FedEx, UPS, USPS or DHL at one single place. By using a unified web interface, the user has no need to login multiple websites to track orders. Figure 3 shows an example of delivery status for UPS service. When the user selects the shipping service and enters the tracking number; he can see the status of his order. The tracking service at Trackship website supports four package delivery companies UPS, FedEx, USPS, and DHL.

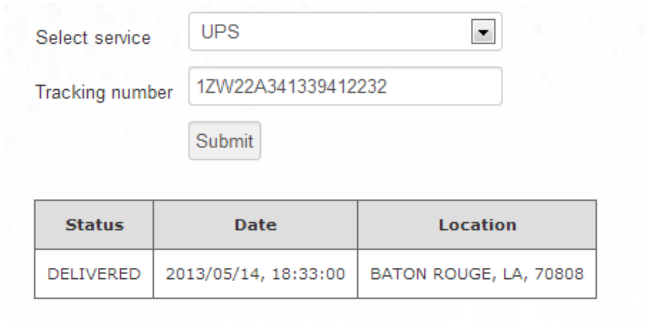


Figure 3. An example of Tracking Service on Trackship

3.2. Shipment Cost Service

Trackship allows users to compare shipping services costs among shipping services including DHL, USPS, and UPS [3]. Based on the shipping cost comparison provided by Trackship, users can choose an affordable service. Figure 4 provides a snapshot of shipping cost service on Trackship. The snapshot shows that one can use the Shipping Cost service to compare different costs for multiple shipping services. When given the zip codes of starting location and destination, as well as dimensions including length, width, height and weight of the package, Trackship displays the cost for different shipping services.

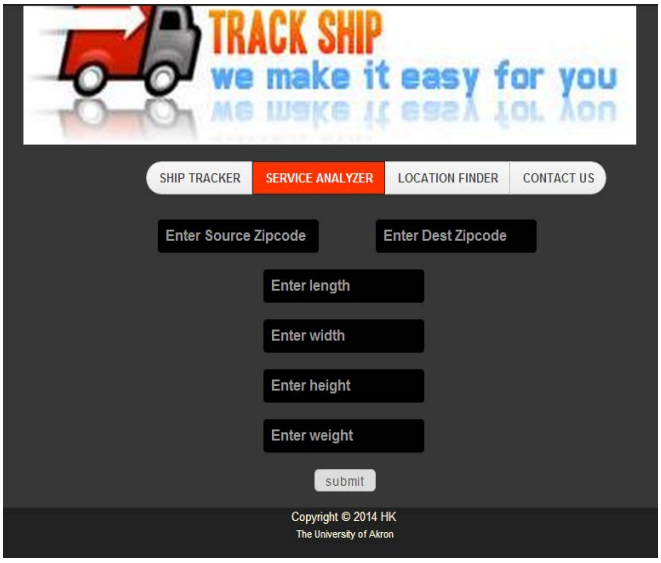


Figure 4. A snapshot of shipment cost comparison page

Figure 5 shows the comparison results between two companies UPS and USPS when given a package of weight 65 lbs and source and destination as 44304 and 44310 respectively.

UPS

Available Services	Rate
Ground	17.4624 USD
3 Day Select	36.6047 USD
2nd Day Air	50.1831 USD
Next Day Air Saver	71.8614 USD
Next Day Air Early AM	121.9779 USD
Next Day Air	85.3479 USD

USPS

Available Services	Rate
Priority Mail-	27.40 USD
Priority Mail- Large Flat Rate Box	16.85 USD
Priority Mail- Medium Flat Rate Box	12.35 USD
Standard Post-	18.78 USD

Figure 5. An example of shipment comparison results

3.3. Service Location Provider

Customers can search for nearby location of drop boxes or post offices for package delivery companies (UPS, USPS and FedEx) with the "Nearby Location" service provided by Trackship. We used Google maps API [4] to embed maps into Trackship website. Google maps is a Ajax web mapping service [5] application provided by Google. Google Maps for mobile is the world's most popular app for smartphones, with over 54% of global smartphone owners using it at least once during the month of August 2013 [6]. Figure 6 shows an example of service locations rendering on Trackship.

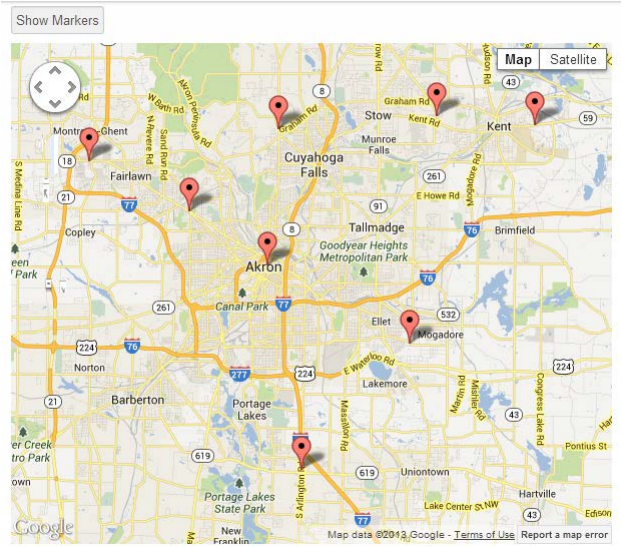


Figure 6. Service locations rendering on Trackship

3.4. Challenges

Integrating heterogeneous data from different delivery companies is a major challenge. Some services provide data in XML format, and some in HTML format. For example, UPS is communicated through API services where we send XML data as requests and get only XML data as responses whereas DHL provides HTML data which can be parsed to display the results. In addition, data-level heterogeneity including data types adds new difficulty level in developing Trackship.

The second challenge was getting API's services. As the free shipping services are being used for commercial purposes these days, there has been a strict background work going on to know whether the APIs are used for commercial purposes or for personal. Also some Shipping services like DHL are not offering any API services. We overcame this challenge by reading the HTML sources. Trackship web application is completed using the API services of UPS, USPS and reading HTML sources for FedEx, DHL.

As smart phone applications are getting popular, a large number of customers use smart phones to access web services. To target mobile users, we used Twitter Bootstrap [7] to make Trackship automatically compatible for all kinds of tablets and smart phones.

4. Implementations

Trackship is a web-based application where a customer interacts with a web interface on his smart phone, or tablet, or computer, and gets the desired results in an efficient manner.

Trackship is developed using basic HTML5 and CSS scripting languages for header information, meta data, tables, and footer. As this application is streaming data from the API web services and page sources of various shipment services, CURL PHP [8] is used as back-end technology to safely get the raw data from the shipment services. The code for the CURL PHP consists of connecting to the server for which API access is available or the page source at which results are displayed and then sending the requests to the shipping services in the form of xml request or get URL method. Figure 6 shows an example of CURL PHP request code.

```
/RatingServiceSelectionRequest>";
        $ch = curl_init("https://wwwcie.ups.com/ups.app/xml/Rate");
        curl_setopt($ch, CURLOPT_HEADER, 1);
        curl_setopt($ch,CURLOPT_POST,1);
        curl_setopt($ch,CURLOPT_TIMEOUT, 60);
        curl_setopt($ch,CURLOPT_RETURNTRANSFER,1);
        curl_setopt ($ch, CURLOPT_SSL_VERIFYPEER, 0);
        curl_setopt ($ch, CURLOPT_SSL_VERIFYHOST, 0);
        curl_setopt($ch,CURLOPT_POSTFIELDS,$data);
        $result=curl_exec ($ch);
```

Figure 7. CURL PHP requests code of the Trackship

In order to gather real-time live data from shipment services and get accurate and reliable data for customers when inputs are given, we used XML technologies to interact with shipment services and temporarily store the data. Then, it will be parsed using string parsing utilities and displayed for customers. The typical XML response of the shipment service is shown in Figure 8.

```
- <RateV4Response>
  - <Package ID="0">
      <ZipOrigination>44304</ZipOrigination>
      <ZipDestination>44310</ZipDestination>
      <Pounds>34</Pounds>
      <Ounces>0</Ounces>
      <Size>LARGE</Size>
      <Width>5</Width>
      <Length>4</Length>
      <Height>3</Height>
      <Machinable>TRUE</Machinable>
      <Zone>1</Zone>
```

Figure 8. XML response code of the Trackship

To simplify application development, different functionalities are divided into independent modules for achieving high cohesion and less coupling among functionalities for future development purposes. We used JavaScript language and JQuery to add more interactive features such as carousel with timer for images to slide, and simulating Google maps for user to easily enter zip code information. Customers can easily enter information like his current zip code and desired radius around which he needs to know the locations of shipment services. Trackship can render the shipping service locations on maps for users to interact.

We used Twitter Bootstrap technology [7] to make our application compatible and responsive to all kinds of devices including computers, smart phones, and tablets. With this feature, the user can easily interact with Trackship anywhere, anytime. Figure 9 shows a snapshot of the interface for nearby location service on smart phones.

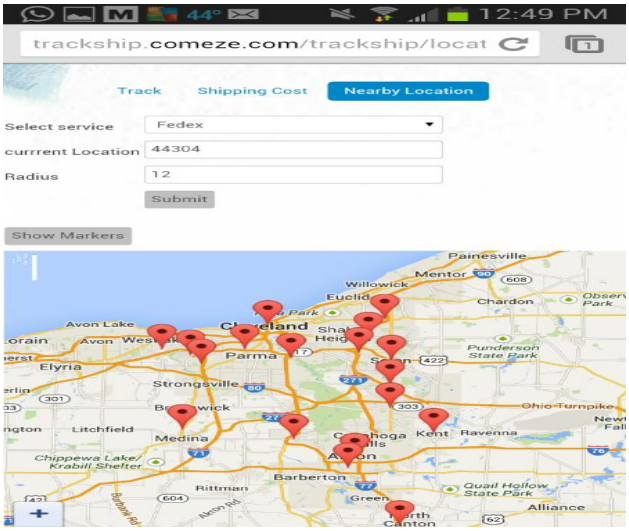


Figure 9. Trackship on Mobile device showing service Locations

Trackship application is tested for different browsers including Google chrome, Firefox, and Internet Explorer. And it is proven fast and reliable. To reduce the loading delay, we have reduced the number of JSP scripts to load before a page is reloaded.

5. User study

We conducted a user study to evaluate the usability and functionality of Trackship with 21 participants. We received 98% of positive responses regarding the unified

interface for tracking. And 65% of the participants think that the comparison of shipping costs is a useful feature. Some participants recommend that, Trackship can send an alert message when a package is delivered for future work. Another advanced feature is to provide approximate time to delivery and provide shipping labels, which can make shipping much easier.

6. Conclusion

In this paper, we presented Trackship, a unified shipment tracking application, which is designed to help customers track their orders for different shipping services including UPS, FedEx, DHL, and USPS. Users can use Trackship on their smart phones, tablets, and computers. With Trackship, users can also calculate shipping services fees and compare shipping cost among different delivery companies. Furthermore, customers can also search for nearby location of drop boxes or post offices of package delivery companies using the "Nearby Location" service provided by Trackship. In future work, we plan to add message function which notifies users when packages are delivered.

7. Acknowledgement

We thank Shruthi Munagala and Nagarjun Reddy Gaddam, Graduate Students in the Department of Computer Science, for useful discussions and assistance with our user study, as well as reviewing the manuscript.

8. References

[1]https://www.ups.com/content/us/en/resources/techsupport/integrating_apis.html

[2] https://www.usps.com/business/web-tools-apis

[3]http://www.ups.com/content/us/en/shipping/cost/zones/index.html

[4] Google Maps API, https://developers.google.com/maps/

[5] http://msdn.microsoft.com/en-us/library/gg427610.aspx

[6] "Google+ Smartphone App Popularity". Business Insider. Retrieved 2013-09-06.

[7] http://twitter.github.io/bootstrap/

[8] http://www.php.net/manual/en/ref.curl.php

Improve XML Web Services' Performance Using SOAP Compression

Iehab Al Rassan and Haifa Alyahya
King Saud University, College of Computer and Information Sciences
Department of Computer Science

***Abstract*—Applications are part of our daily life that many people use it more than anything else. Nowadays, applications are connected via the internet and the users want to use fast and efficient services offered through the web services. Web services are the core of modern application architectures that will be used for many years. Regardless of what platform or language that developers are using, the critical skill understands how web services work. "Web services are the inseparable part of any web application, as a result enhancing performance of web services will have a great effect on the overall performance of the system" [1]. Compressing and reducing the size of SOAP messages traveling over the network, improves the web services performance, which consumer invoke.**

This paper proposes and discusses a methodology that compresses the SOAP messages using compression techniques that have special features to improve the performance of SOAP messages. These compression techniques are Tagged Sub-optimal Code (TSC), Huffman Encoding Algorithm, Byte Pair Encoding (BPE) and J-bit encoding (JBE) Algorithm.

***Index Terms*—Web Services, SOAP, Performance, XML, Network, TSC, BPE, Huffman, JBE.**

I. INTRODUCTION

A **SOAP** message is an XML-based protocol for exchanging information between computers. It enables client applications to easily connect to remote services and invoke remote methods. The main challenge of **SOAP** performance is when exchanging messages over a network is the size of the **SOAP** messages which is large and effecting the transmission time. Creating Web Services without considering **SOAP** performance could reduce overall system performance. There are many ways to improve the performance of sending **SOAP** messages. In our proposed project we will apply **SOAP** message compression and evaluate the performance of Web Services using it to see which technique is better.

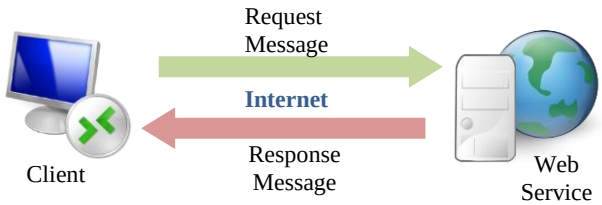


Fig. 1. Message Formats

A. Performance of web services

Performance in web services is an important issue, especially if a large amount of data has to be exchanged, so to solve this problem you need to find a way to lower the size of the data which is passed between the clients and the servers. Compressing text in Web service is most important challenge to improve its performance because message sizes in Web services are larger than in traditional web technologies. Compression means that the size of message reduced up to 80% and the data needs less time to be transferred over a network, which will affect a higher performance for client-server applications that communicate with text, like **XML** Web services.

B. Compression techniques

"Data compression refers to reducing the amount of space needed to store data or reducing the amount of time needed to transmit data. The size of data is reduced by removing the excessive information. The goal of data compression is to represent a source in digital form with as few bits as possible while meeting the minimum requirement of reconstruction of the original."[2]. There are two general types of compression algorithms:

1) Lossless compression

Lossless compression compresses the data in such a way that when data is decompressed it is exactly the same as it was before compression i.e. there is no loss of data. Lossless compression basically rewrites the data of the original file in a more efficient way.

2) Lossy compression

Lossy compression is the one that does not promise that the data received is exactly the same as data send i.e. the data may be lost. So, Lossy file compression results in lost data and quality from the original version. They are typically achieving much better compression ratios than the lossless algorithms.

II. RELATED WORK

In (**2008**) AlRassan, I., Assiri, A.[3], implementing a new compression method called "tagged suboptimal code (**TSC**)" to compress **XML** content replace with its corresponding code

word representation, then transmitted the resulted comprise file (or message).

Minaei, B., Saadat, P. in (**2009**). [1], try to improve performance of any system by enhancing performance of web services. The main idea of their studies is to avoid the redundant serialization stage of **SOAP** responses for request which have the same call parameters. They provide solution to take advantage of similar Soap requests on a web server for a particular web service by running a Middleware on top of web server called Serialization Enhancement Middleware (**SEM**).

In (**2010**) Al-Shammary, D., Khalil, I. [4], try to enhancing the performance of Web services by compressing **SOAP** messages. They are proposing two innovative techniques capable of reducing small as well as very large messages. Their research shows that high Compression Ratio is up to 7.8 and around 13.5 for large and very large messages respectively.

Al-Shammary, D., Ibrahim, K. (**2012**).[5], proposed two new redundancy-aware **SOAP** Web message aggregation models - Two-bit and One-bit **XML** status tree reduce the required bandwidth, latency, and improve the overall performance of Web services by enable the Web servers to aggregate **SOAP** responses and send them back as one compact aggregated message. Their experiments achieving compression ratios as high as 25 for aggregated **SOAP** messages in significant performance for both aggregation techniques.

In (**2012**) a new algorithm for data compression is proposed by Suarjaya, I. [2] called j-bit encoding (**JBE**). It is classified to lossless compression, because it will manipulates each bit of data inside file to minimize the size without losing any data after decoding which. To measure the performance of this algorithm you need to comparing combination of different data compression algorithms. In this experiment the algorithm gives better compression ratio when inserted between move to front transform (**MTF**) and arithmetic coding (**ARI**).

III. PROPOSED SOLUTION

We will aim to improve the Web Services performance **SOAP** Compression using tagged sub-optimal code (**TSC**). We will discuss the benefits of using it to enhance the SOAP performance and compare it with the other techniques that used in compression such that Huffman encoding and Byte Pair Encoding (**BPE**). We will focus on **SOAP** performance and propose our approach to **SOAP** message compression of the Web Services.

In our search for the aim of project, a new algorithm of data compression, called j-bit encoding (**JBE**). "J-bit encoding (**JBE**) works by manipulate bits of data to reduce the size and optimize input for other algorithm. The main idea of this algorithm is to split the input data into two data where the first data will contain original nonzero byte and the second data will contain bit value explaining position of nonzero and zero bytes. Both data then can be compress separately with other data compression algorithm to achieve maximum compression ratio. "[2] JBE will be used to compare it with other algorithms such as Huffman encoding, Byte Pair Encoding (**BPE**) and tagged sub-optimal code (**TSC**) to improve the performance of **SOAP** message.

IV. COMPRESSION ALGORITHMS

1) Byte Pair Encoding (BPE):

Byte pair encoding is one of the data compression techniques that replaced most common pair of consecutive bytes of data with a byte that does not occur within that data. Rebuild the original data by changing pair using table of the replacements. The **BPE** encoding algorithm is a multi-pass and requires that all the data must be stored in memory. The algorithm was first described publicly by Philip Gage in a February **1994** article "**A New Algorithm for Data Compression**" in the C Users Journal.

2) Huffman encoding:

Huffman coding is an entropy encoding algorithm used for lossless data compression. It is one of the simple compressing encoding schemes and can be implemented easily and efficiently. The term refers to the use of a variable-length code table for encoding a source symbol (such as a character in a file) where the variable-length code table has been derived in a particular way based on the estimated probability of occurrence for each possible value of the source symbol. It was developed by David A. Huffman, and published in the **1952** paper "**A Method for the Construction of Minimum-Redundancy Codes**".

3) Tagged Sub-optimal code (TSC):

TSC is a suboptimal coding technique that supports minimal prefix property. It generates code-words from a tree traverse that can be represented by a quad tree (degree 4). The code-words are generated in every level and its sizes depend upon the number of represented codes (2 to 14 bits long for 254 codes). Its boundaries are instantaneously detected that why it is "tagged". The algorithm was first described publicly by Bellaachia and Al Rassan in **2004**.

4) J-bit encoding (JBE):

J-bit encoding (**JBE**) is a new lossless compression algorithm that used to minimize the size without losing any data after decoding by manipulates each bit of data inside file. This algorithm will optimize the compression ratio rather than other data compression algorithms. To measure the performance of this algorithm you need to comparing combination of different data compression algorithms. The algorithm was first described publicly by I Made Agus Dwi Suarjaya on 5 Sep 2012 article "**A New Algorithm for Data Compression Optimization**" in the (IJACSA) International Journal of Advanced Computer Science and Applications.

V. METHODOLOGY

In order to test the performance of compression algorithms, the tagged sub-optimal code (**TSC**) Algorithm, Huffman Encoding Algorithm, Byte Pair Encoding (**BPE**) and J-bit encoding (**JBE**) Algorithm will be implemented and compared with a set of text files. Performances will be evaluated and tested by comparing the speed of invoking the webservices over a network for all these compression techniques.. The main goal of the experiment is to see which algorithm is effective for **SOAP** compression to improve the performance of **XML** web services. **JBE** compression techniques will be implemented and tested also to compare it with other algorithms to find out which algorithm achieves better performance and compression ratio..

• The development environment for the implementation will be **C#** on the .**NET** framework.

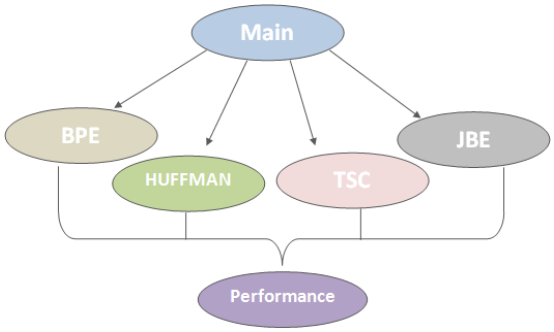


Fig. 2. Application's Class Hierarchy

A. *Proposed Implementation*

1) The web service consumer will request by send a **SOAP** message.
2) The **SOAP** message will compress before sending it by compression algorithm
3) The compressed **SOAP** message will be sent via the Internet.
4) The **SOAP** message will be de-compressed before arriving.
5) The web services provider will be receiving the request **SOAP** message.
6) The web services provider will be responding by sending the respond **SOAP** message.
7) The **SOAP** message will be compressed before sending it by compression algorithm.
8) The compressed **SOAP** message will be sent via the Internet.
9) The **SOAP** message will be de-compressed before arriving.
10) The web services consumer will receiving the response of the request.

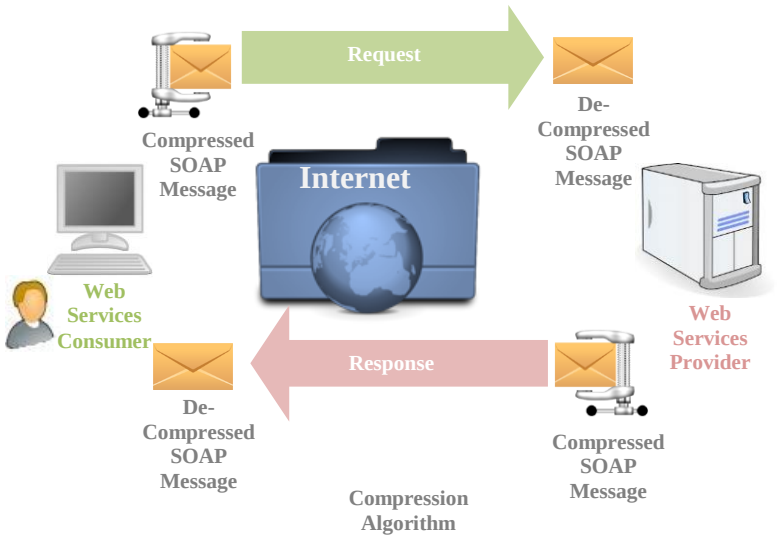


Fig. 3. Proposed Implementation

B. *Measuring Compression Performances*

The performance depends on the type and the structure of the input source. The compression behavior depends on the redundancy of symbols in the source file, so it is difficult to measure performance of a compression algorithm in general. However, the overall time needed to compress, decompress , and invoke the web services over a network will be measured and compared among all compression algorithms in order to find out which one is the best that can improve the web services performance.

The category of the compression algorithm will affect the result. The space efficiency and time efficiency would be high if we are using a lossy compression algorithm to compress a particular source file, but this would not be accepted for webservices We should use different measurements to evaluate the performances of the compression algorithms.

Compression Ratio is the ratio between the size of the compressed file and the size of the source file.

$$compression\ \ Ratio = \frac{size\ \ after\ \ compression}{size\ \ before\ \ compression} \quad (1)$$

Compression Factor is the inverse of the compression ratio. That is the ratio between the size of the source file and the size of the compressed file.

$$compression\ \ Ratio = \frac{size\ \ before\ \ compression}{size\ \ after\ \ compression} \quad (2)$$

Saving Percentage calculates the shrinkage of the source file as a percentage. (3)

$$saving\ \ percentage = \frac{size\ \ before\ \ compression - size\ \ after\ \ compression}{size\ \ before\ \ compression}\% \quad (3)$$

VI. CONCLUSION

Transferring large message size is one of the web services issues, so the best way to improve performance by using data compression. The compression algorithms comes as a solution to reduce the message size and improve the performance of web services by a number of researchers, using different concepts and strategies. In our project, we will implement some algorithm of compression such as the tagged sub-optimal code (**TSC**) Algorithm, Huffman Encoding Algorithm, Byte Pair Encoding (**BPE**) and J-bit encoding (**JBE**) Algorithm, so we can figure out which one is the best to use. Some factors are considered for comparison in order to identify the best solution. The main challenge is to achieve high compression ratio and at the same time speeding up the overall time needed for invocation process over the network.

In the future, we will develop a programming interface with a set of text files to measuring the compression performances of each Algorithm. After the implementation is done an experiment will be conducted to evaluate which algorithm is good for improving the **SOAP** message compression.

REFERENCES

[1] Minaei, B., Saadat, P. (2009). SOAP Serialization Performance Enhancement DESIGN AND IMPLEMENTATION OF A MIDDLEWARE. International Journal of Computer Science and Information Security, IJCSIS, Vol. 6, No. 1, pp. 105-110, October 2009, USA.

[2] Suarjaya, I. (2012). A New Algorithm for Data Compression Optimization. Information Technology Department, Udayana University, Bali, Indonesia. (IJACSA) International Journal of Advanced Computer Science and Applications.

[3] AlRassan, I., Assiri, A.(2009). Optimizing Web Services for Mobile Devices using Tagged Sub Optimal Code (TSC). MoMM '09 Proceedings of the 7th International Conference on Advances in Mobile Computing and Multimedia.

[4] Al-Shammary, D., Khalil, I. (2010). SOAP Web Services Compression Using Variable and Fixed Length Coding. Network Computing and Applications (NCA), 2010 9th IEEE International Symposium on Date of Conference: 15-17 July 2010.

[5] Al-Shammary, D., Ibrahim, K. (2012). Redundancy-aware SOAP messages compression and aggregation for enhanced performance, School of Computer Science & IT, RMIT University, Melbourne, Australia.

[6] Swarnkar, V., Satao, K.(2013). An Implementation of Efficient Text Data Compression. International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-3, Issue-2, December 2013.

[7] Altarawneh, M., Altarawneh, H. (2011). Data Compression Techniques on Text Files: A Comparison Study. International Journal of Computer Applications.

[8] Girish Tere, G., Jadhav, B. (2011). Improving Performance of XML Web Services. Technology Systems and Management, Communications in Computer and Information Science.

[9] Shanmugasundaram, S., Lourdusamy, R. (2011). A Comparative Study of Text Compression Algorithms. International Journal of Wisdom Based Computing, Vol. 1 (3).

[10] Rosu, M. (2012). Adaptive parsing and compression of SOAP messages. IEEE International Conference on Web Services (ICWS 2007).

Hybrid Software Framework For Web/Mail Interface

Toshiyuki MAEDA[1], **Yae FUKUSHIGE**[1], **Takeshi MATSUDA**[2], **and Masumi YAJIMA**[3]
[1]Faculty of Management Information, Hannan University, Japan
[2]Faculty of of Comprehensive Informatics, Shizuoka Institute of Science and Technology, Japan
[3]Faculty of Economics, Meikai University, Japan

Abstract— *We address hybrid software framework for Web/Mail interface. For that purpose, it is important to be easy not only to reconfigure but also maintain and improve MCS, and thus we propose hybrid software framework with a point of view to GoM and multi MVC framework and Bulk mail interface, and present an improved system using mobile devices.*

Keywords: Software framework, Granularity of message, Mobile communication, Multi interface, Bulk mail

1. Introduction

Recently mobile communication systems (MCSs) are getting more and more important for our everyday life as we cannot live without mobile phones, not only for business but also private time.

There are various education systems supported by computers. A web-based digital model railroad platform has been developed as a client-server system for education of computer science [7]. Also, a web-based system has been developed for control engineering education [8]. To put communication skills into engineering curriculum, a web-based system to integrate workplace has been developed. The purpose of the web-based systems is to establish efficient education, and to communicate sufficiently among faculties and students. In various education areas, many problems are solved using web-based systems [3]. One of the most critical problems is, however, that web-based systems cannot be used in classrooms where computers are not settled for all students, and is essential for many cases. We have thus developed an e-mail-based system using mobile phones, which almost all students have in Japan. There are only few e-mail-based systems for similar purpose, such as [4].

So far, there are several researches (for instance [6], [5]), especially for learning support management system. We furthermore have an attempt to apply for health care education and we have to consider spread of various devices such as tablet, ultrabook, and so on. For that purpose, it is important to be easy not only to reconfigure but also maintain and improve MCS for supporting hybrid communication interfaces such as Web and/or e-mails (Web/mail), with a point of view to "granularity of message" (GoM, explained below). We therefore introduce software development environment, based on MVC framework such as [2], [1] and so on, where M stands for Model (information structure), V for View (input and output as user interface), and C for Controller (data control and processing). In our study, one M and one C are prepared and several Vs are developed for each interface. In this paper, we address our proposal of Web/Mail software framework for mobile education.

2. Granuality of Message

For designing Web/mail interfaces for realizing above objectives, we introduce concept of GoM. For instance, when a user has a small mobile device, he can read and write a few characters at a glance even though the device has a high-resolution display. At the same time, however, if another user uses a quite big tablet device, then he can treat more characters rather than by a small device. We regard the variation of preferance for GoM (see Figure 1). For that reason, mobile systems have to offer several interfaces depending on the preference. To realize requirements, we propose the Web/mail software framework, and introduce multi MVC framework and Bulk mail interface describing details as followed.

3. Framework for Web/Mail applications

For above objectives, we have to serve the most preferable interface for students who use mobile devices. In this paper, mail application denotes an application where sending mails from users to a server is treated as inputs and replies from the server to corresponding users are as output, and repeating this interaction (round-trip) makes a primitive of communication for an application system. In general, MCS has three layers structure as for users;

1) System-administrator (SA),
2) Intermediate-administrator (IA): teacher, doctor, nurse, etc.,
3) End-user (EU): students, patients, etc.

For EUs, not WWW but e-mails are seemed to be essential for MCS. That is because;

1) Carrier- or terminal-independent; as some WWW browsers on mobile phones are not compatible (HTML nor Java applet) with others, especially in Japan, and so we have to provide several variation of application for those phones, though e-mails are compatible with each other.

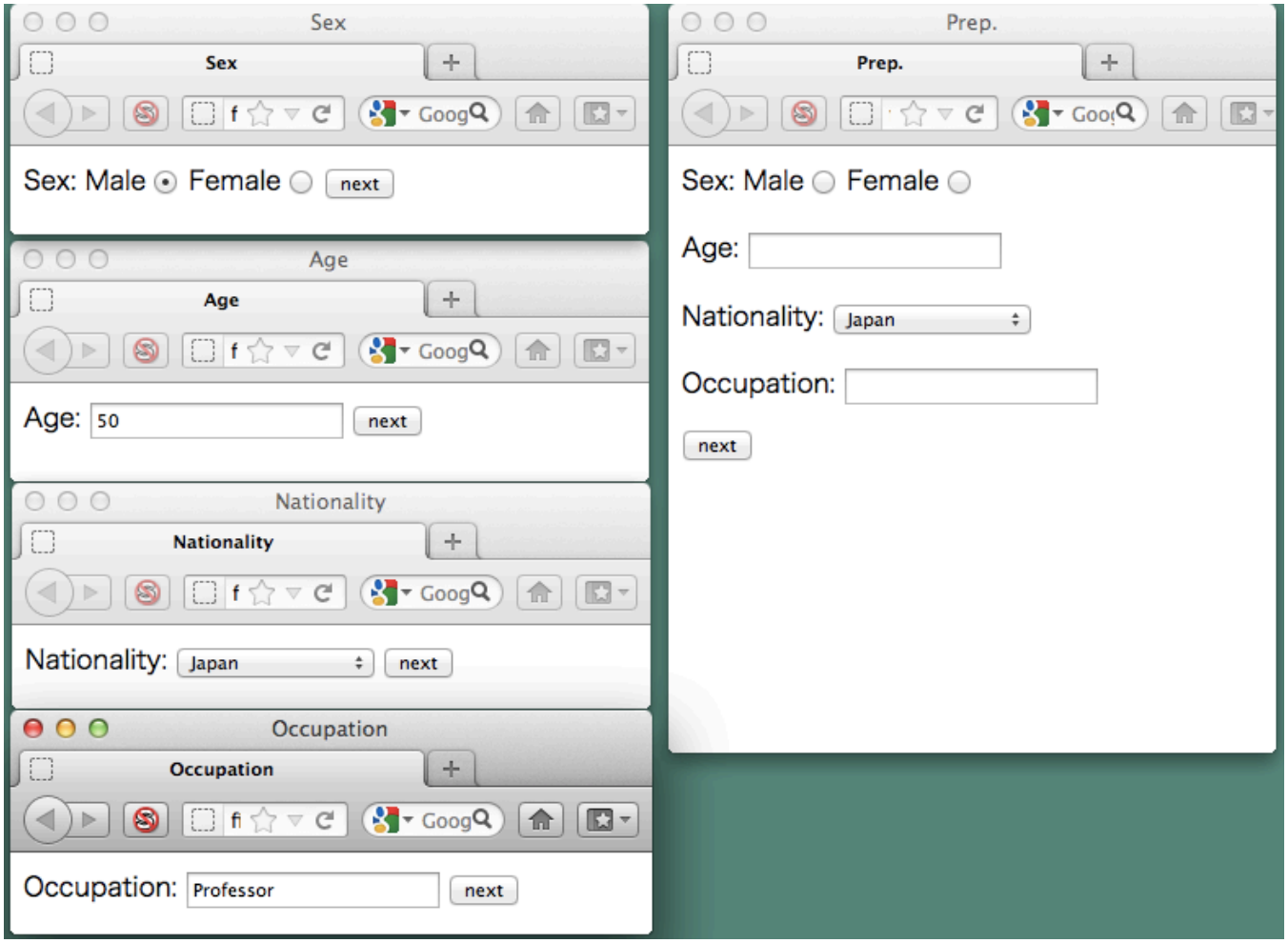


Fig. 1: Examples of GoM.

2) Lightweight process; as WWW applications are heavy for servers in general.
3) Quick response; though some exemption may happen.

It seems, however, to be convenient for not only SAs but also IAs, who know not so much about internal structure of the MCS.

3.1 Multi MVC Framework

We refer CakePHP[cak] as MVC style development methodology, such as

- M(odel):Abstract data structure and procedure.
- V(iew):Output presentation, as well as some of input function included in this layer as for bulk mail interface (mentioned below).
- C(ontrol):Control layer, for interface of M and V in itself, and partially dealing with input.

As mentioned above, we design a framework, which has one M, one C, and several Vs, and they are corresponding to interfaces such as Web/mails, and depending on GoM (see figure 2).

3.2 Bulk Mail Interface

Figure 3 shows procedures of Bulk mail interface corresponding to Web interface. On a Web application, HTML FORM is used for input to an application. We can send multiple input data simultaneously on web application, though it is not so easy for mail application. To realize FORM input function, we propose bulk mail interface. Bulk mail means repeated "round-trip" mails where sequence is controlled by the application system. On this sequence, e-mail address and quoted word (message from the server) is treated as a "cookie" data.

4. Implementation

Our prototype system is implemented on Linux (Cent OS 5.3) with Apache, php, perl and MySQL, so called LAMPP. Moreover part of our system is built referring to CakePHP

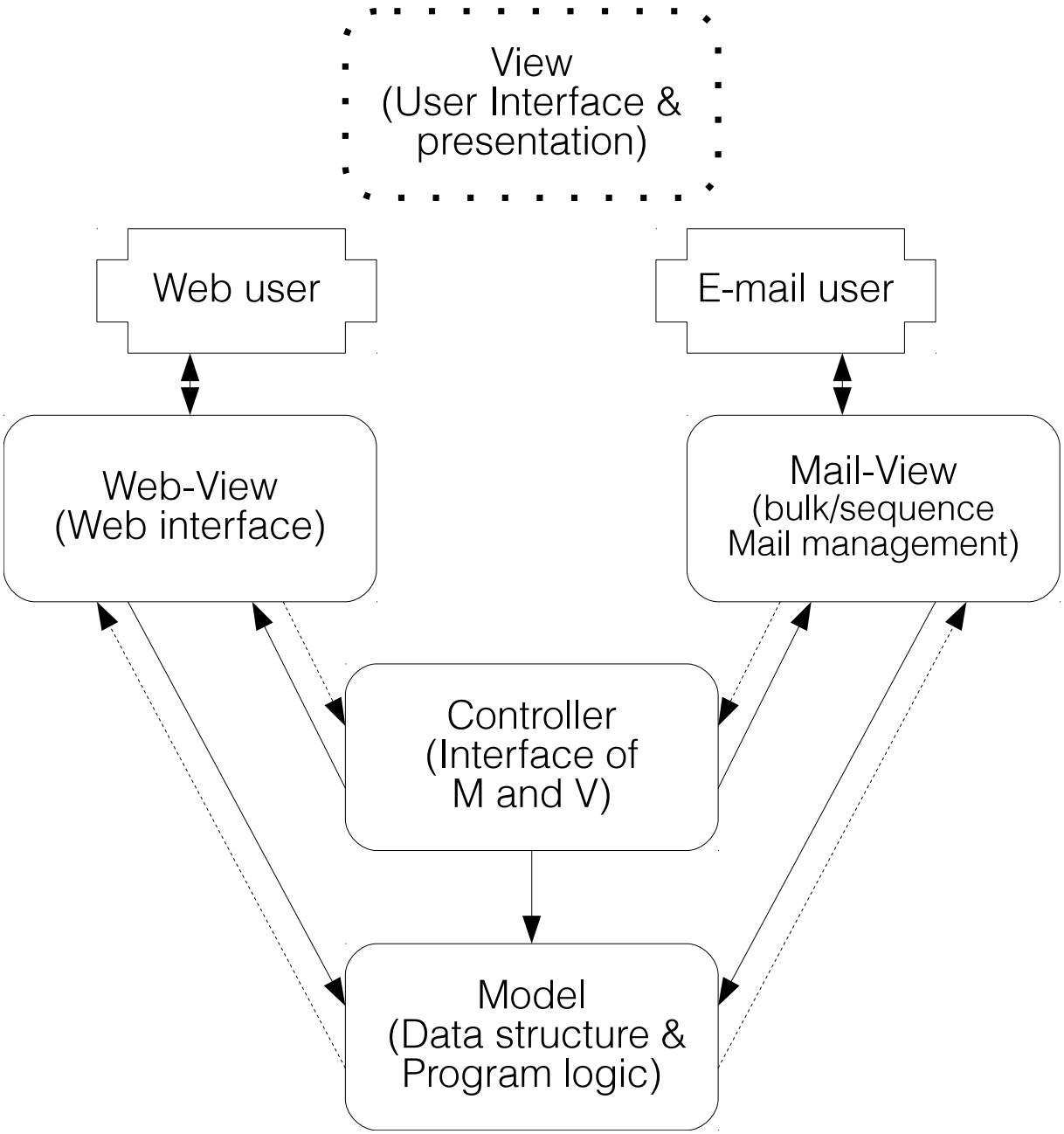


Fig. 2: Multi MVC Framework.

formats. We present some sample codes of our system as follows.

4.1 Description of data structure

List 1: Data structure.

```
CREATE TABLE qas (
id INT UNSIGNED AUTO_INCREMENT,
PRIMARY_KEY,
address VARCHAR(40), # send a question
question VARCHAR(200), # send an Answer
answer VARCHAR(40), # Done
created DATETIME DEFAULT NULL,
modified DATETIME DEFAULT NULL
) DEFAULT CHARSET=utf8;
```

List 1 is an example of description of data structure, or database table definition. In this example, areas between "#" and end-of-line are comments, which are used by the tool explained below.

4.2 Description of data control

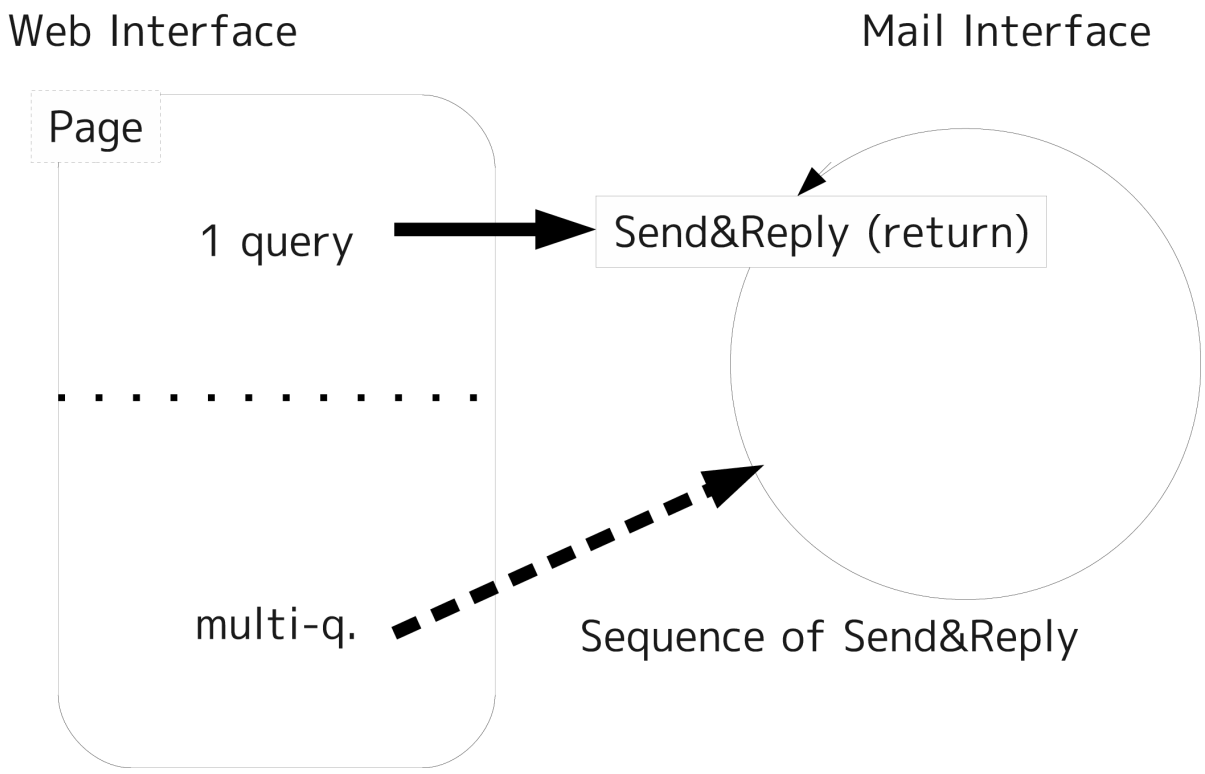


Fig. 3: Bulk mail procedure.

List 2: Example of data control.

```
class QasController extends AppController {
  var $helpers = array('Html','Form');
  var $scaffold;
  function regqa() {
    // /corresponding to "add"
    if (!empty($this->data)) {
      if ( $this->Qa->save($this->data)) {
        $this->redirect('/qas');
    }}
  }
}
```

List 2 is an example of description of data control. This description follows CakePHP manner.

4.3 Description of I/O control for Web application

List 3: Example of I/O control.

```
<h1>Sample</h1>
<table>
<?php
echo $html->tableHeaders(array('Number','Question','Answer
    '));
foreach($datas as $data):
 echo $html->tableCells(array(array($data['Qa']['id'],
      $data['Qa']['question'],$data['Qa']['answer'])));
endforeach;
?>
</table>
<h2> Search questions </h2>
<?php
echo $form->create('Qa',array('action' => 'qanda'));
echo $form->input('qid',array('label' => 'question ID'));
echo $form->submit('Submit');
echo $form->end();
?>
```

List 3 is an example of description of I/O control. This description follows CakePHP manner as well.

4.4 Mail interface configurations

List 4: I/O control description for mail interface

```
[Address]
aimqas-add@example.jp, where
  aim::= prefix,
  Qas::= model name, and
  add::= control name

[/etc/postfix/virtual]
/^aim.*@m.aimos.jp$/i
aimos@m.aimos.jp

[/etc/aliases]
aimos: |/usr/local/bin/php
/var/htdocs/aimos/m2wb/perp.php
```

List 4 shows mail address syntax and configuration files (/etc/postfix/virtual and /etc/aliases, each of contents are originally one line respectively) for mail interface. This system uses virtual domain hosting function on Postfix.

As mentioned above, we have developed the multi interface system as revisal and improvement on CakePHP framework. Figure 4 shows a sample image of View for question lists. In this case, users can check and edit some

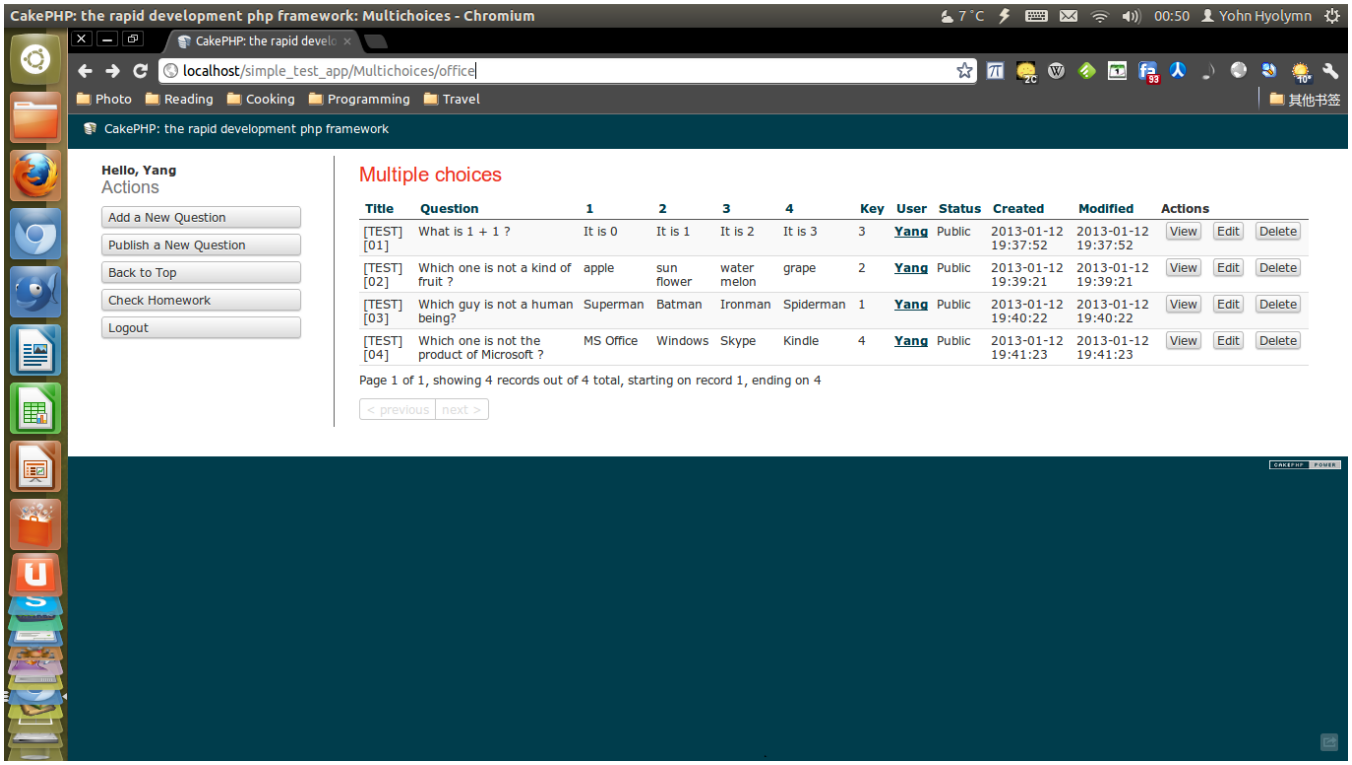


Fig. 4: View of question list (WWW Interface).

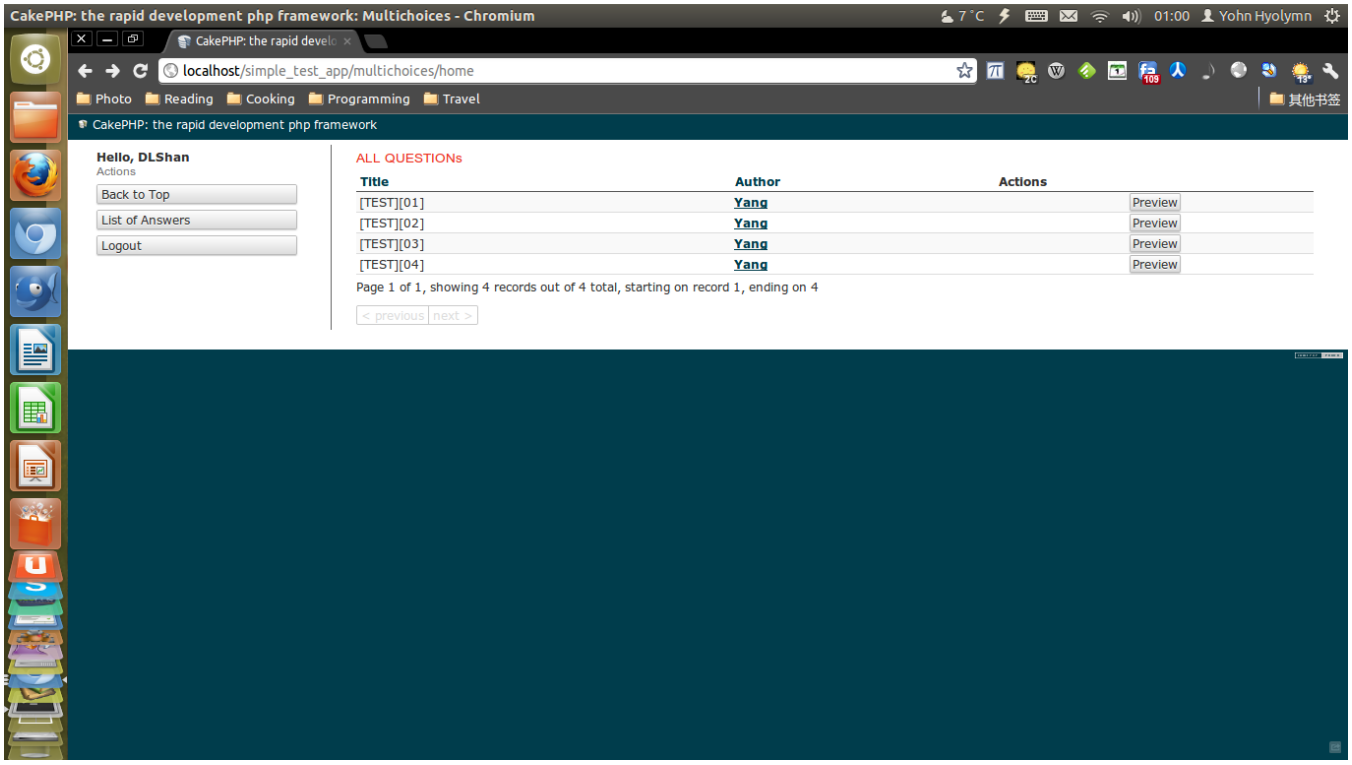


Fig. 5: View of selecting questions (WWW Interface).

of questions. Figure 5 shows a view of selecting questions as student users.

List 5: A sample mail body (Mail interface).

```
There is a new question from A Simple Test Application.
=== This is the QUESTION ===
Question : Choose one ?
Choice 1 : 1
Choice 2 : 2
Choice 3 : 3
Choice 4 : 4

Please reply this mail only with the message below, and
     REPLACE the [Username] with your USERNAME, and the [
     Answer] with your Answer.
And don't use "<" or ">".
For example:
MultipleChoice.01.01.Tom.Title.1
==== Just send this message below back ====
MultipleChoice.2.[TEST][Try 00011].[Username].MAIL.[Answer
     ]
```

List 5 shows a sample mail body for answering a question. This mail is sent to all registered students at a time, and the answer should be replied by mail as well. To realize mail functions into the system, we introduce some modules in addition to the original CakePHP framework.

5. Conclusion

We address hybrid software framework for Web/Mail interface. For that purpose, it is important to be easy not only to reconfigure but also maintain and improve MCS, and thus we propose mobile communication software framework with a point of view to GoM and multi MVC framework and Bulk mail interface. As the future plan, we install the rigid system and then have field tests to refine our system for evaluating our framework.

Acknowledgement

Part of this research was supported by the Ministry of Education, Science, Sports and Culture, Japan; Grant-in-Aid for Scientific Research (C), 24520173 (2012-2014), and 25380630 (2013-2015). This research was also partially conducted under a sponsorship of Grant of Institute of Industrial and Economic Research, Hannan University. The authors greatly appreciate the supports.

References

[1] CakePHP: the rapid development php framework. http://cakephp.org/. (2009.7).

[2] Ruby on Rails: Web development that doesn't hurt. http://rubyonrails.org/. (2009.7).

[3] N. Hanakawa, K. Goto, T. Maeda, and Y. Akazawa. Discovery Learning for Software Engineering –A Web based Total Education System: HInT–. In *Proceedings of The International Conference on Computers in Education (ICCE 2004)*, pages 1929–1939, Melbourne (Australia), 2004.

[4] Dag Johansen, Robbert van Renesse, and Fred B. Schneider. Supporting Broad Internet Access to TACOMA. In *Proceedings of the 7th SIGOPS European Workshop*, pages 55–58, Connemara, Ireland, 1996.

[5] T. Maeda, T. Okamoto, Y. Fukushige, and T. Asada. Mobile Communication System for Health Education. In *Proceedings of World Conference on Educational Multimedia, Hypermedia & Telecommunications (ED-MEDIA 2009)*, pages 1156–1161, Honolulu (HI, USA), 2009.

[6] T. Maeda, M. Tomo, and T. Asada. Integrated lecture-support system using e-mail. In *Proceedings of National University Symposium on Information Edication Methods (in Japanese)*, pages 26–27, 7 2004.

[7] P. Sanchez, B. Alvarez, A. Iborra, J.M. Fernandez-Merono, and J.A. Pastor. Web-based activities around a digital model railroad platform. *Journal of IEEE Transaction on Education*, 46(2):302–306, 2003.

[8] C. Schmid and A. Ali. A web-based system for control engineering education. In *Proceedings of the 2002 American Control Conference*, pages 3463–3467, Chicago, IL (USA), 2002.

SESSION

SEMANTIC WEB AND RELATED ISSUES

Chair(s)

Using Ontological Semantic Agent to Data Extraction in the Context of Big Data: a tool applied to Higher Education Institutions

C. S. Coneglian[1], **E. Fusco**[1], **and L. C. Botega**[1]
[1]Department of Computer Science, UNIVEM –University Center Euripides of Marilia, Marilia, São Paulo, Brazil

Abstract - *The large increase in the creation and dissemination of data on the Internet can offer information of high value-added to organizations. This information can be provided by heterogeneous databases that may not be considered relevant by most systems, e.g., social media data, blogs and more. If organizations would use such sources, they could build a new management vision known as Competitive Intelligence. In the context of architectures of Information Retrieval, this research aims on implementing a semantic extraction agent for the Web environment, allowing information finding, storage, processing and retrieval, such as those from the Big Data context produced by several informational sources on the Internet, serving as a basis for the implementation of information environments for decision support. Using this method, it will be possible to verify that the agent and ontology proposal addresses this part and can play the role of a semantic level of the architecture.*

Keywords: Big Data; semantic web; semantic agent; ontology.

1 Introduction

The massive diffusion of generated data is testing the ability of the most advanced techniques of information storage technological, treatment, processing and analysis. The areas of treatment and information retrieval are being challenged by the volume, variety and velocity of semi-structured and unstructured complex data, offering opportunities for adding value to business-based information providing organizations a deeper and precise knowledge of their business.

Opportunities to add value to the businnes-based information arise due to both the internal and external environment. Hence, there is a need for a new approach to structure Information Technology (IT) companies to transform data into knowledge, which cause broader impact.

To aggregate and use information that are scattered in the internal and external environments of organizations, there is the concept of Competitive Intelligence, which according Fleisher [1] is a process by which organizations gather actionable information about competitors and the competitive environment and, ideally, apply it to their decision-making and planning processes in order to improve their performance.

A proactive informational process leads to a better decision, whether strategic or operational, in order to discover the forces that govern the business, reduce risk and drive the decision maker to act in advance, besides protecting the generated knowledge.

In the current scenario of the information generated in organizational environments, especially in those who have the Internet as a platform, there is data that, due to its characteristics, is classified as Big Data.

In the literature, Big Data is defined as the representation of the progress of human cognitive processes, which generally includes data sets with sizes beyond the capacity of current technology, methods and theories to capture, manage and process the data within a specified time [2]. Gartner [3] defines Big Data as the high volume, high speed and/or high variety of information that require new ways of processing to allow better decision making, new knowledge discovery and process optimization.

In the process of information search for Competitive Intelligence and Big Data robots, data mining techniques on the Internet are used.

According to Deters and Adaime [5] robots are systems that collect data from the Web and assemble a database that is processed to increase the speed of information retrieval.

According to Silva [6], the extraction of relevant information can rank a page according to a domain context and also draw information structures them and storing them in databases. To add meaning to the content fetched, the robots are associated with Web search semantic concepts, which let the search through a process of meaning and value, extracting the most relevant information.

The ontology in the philosophical context is defined by Silva [6] as part of the science of being and their relationships; in this sense, the use of ontologies is essential in the development of semantic search robots, being applied in Computer Science and Information Science to enable a smarter and closer search to the functioning of the cognitive process of the user so that data extraction becomes much more relevant.

Thus, an agent presents itself as a solution to retrieve information on the web by semantic means. Currently, the

content is organized in a jointly manner, in which syntactic structures do not have semantic data aggregation. In this sense, the role of the agent is to extract the information from the content and use syntactical ontology to achieve semantic relations and apply them to retrieval information.

This research aims to implement a semantic agent for searching on the Web and allowing the retrieval, storage and processing of information, i.e., Big Data from various informational sources on the Internet. Such semantic agent will be the main mechanism for building a computational architecture that transforms disaggregated information on an analytical environment of strategic, relevant, accurate and usable knowledge to allow managers the access to opportunities and threats in the field of higher education institutions, based on concepts of competitive intelligence. The semantics of the agent will be built using ontological structures.

To achieve this goal, the Semantic Agent will be built using the domain of higher education institution, addressing the problem related to scientific research.

In section II is shown on the related work. In section III it is explained how it works the architecture and the differential of this architecture for the others that already exist. Section IV explains what was the basis for the construction of ontology. Already in section V explains the entire process of building and operating a semantic agent within the architecture. And finally in section VI is spoken of the conclusions and the next jobs.

2 Related Work

Information retrieval architectures with use of agents have been proposed by some other studies, where conducted ways to retrieve information for the subsequent use of this information in any scenario.

In this sense, Beppler [16] proposes a type of architecture of information retrieval. This recovery occurs only by analyzing documents, and removing and storing information, without observing the existing context, e.g., using syntactic analyses. This proposal is interesting because it can retrieve information in a successful way but is limited because it only a syntactic search, which narrows the results obtained by the architecture.

Already Wiesner [17] proposes a semantic solution for this issue, using for making ontologies the recovery of information. This proposal already assembles an architecture that uses a semantic solution using ontologies to assemble a base of knowledge integration, using an agent to make associations and integration of such knowledge. So this research can do a lot of good associations required for each knowledge, but the semantics are limited because only the Association of information and cannot determine what each represents information, and if it in fact will be of real value for that particular domain.

3 Information Retrieval in Big Data

The traditional information systems are unable to cope efficiently with all new data sources and multiple contexts of information that have mainly the Internet as a platform.

Problems are encountered in retrieving, standardising, storing, processing and usage of information generated by various heterogeneous sources that are the basis for enabling systems for decision support organizations.

In this context, it is questioning whether the computing environments of information actually present completely all relevant information to decision makers in organizations.

The solution proposed in this paper is to create an architecture for information retrieval in the context of Big Data as seen in Figure 1.

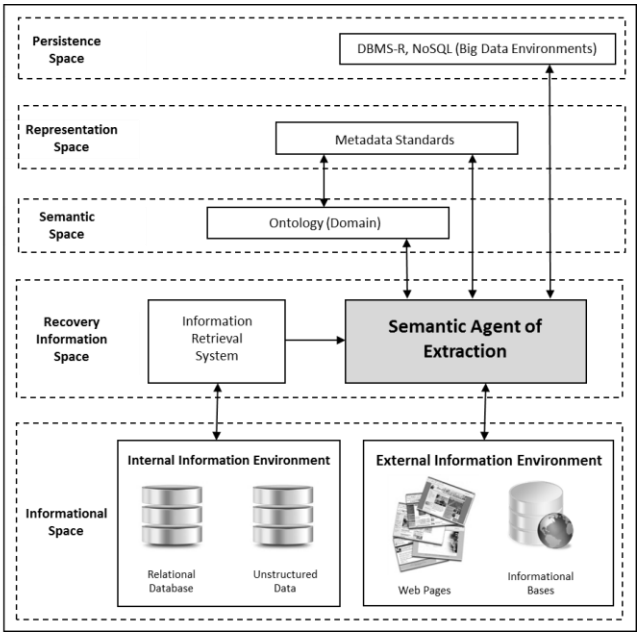


Figure 1: Architecture Context of Semantic Agent of Extraction

This architecture was proposed so that the recovery of the information can be made using the semantic space. The architecture has the agent as the structure of information retrieval and integrates with all elements and layers of the architecture.

This architecture will be used at an institution of higher education, because in this area there are many information not being used several times in competitive intelligence.

This architecture distinguishes itself from others by doing all the recovery information using the same domain, therefore, only will be extracted information relating to the problem in accordance with the defined ontology.

4 Ontology

To set a network of Semantic Web, ontologies have been used frequently [13].

According to Clark [10], an ontology is organized into concept hierarchies, because it cannot reflect optimally specific formalism, then it is possible to consider an ontology as the embodiment of the knowledge level.

To Mizoguch [11], there are several types of ontologies, and can be outlined below:

- Upper Ontology: this type of ontology serves to explain what exists in the world. And most are used to represent large knowledge bases;
- Domain Ontology: is a more specific area of knowledge;
- Task Ontology: This ontology serves to solve a specific problem of a domain;
- Heavy-Weight Ontology: these ontologies are much more defined, have well-defined rules, be very careful when conceptualizing the world;
- Light-Weight Ontology: this type need not be precise as large in the conceptualization.

Noy [8] explains the seven steps that are required to build an ontology: 1. Determine the domain and scope of the ontology; 2. Consider reusing existing ontologies; 3. Enumerate important terms in the ontology; 4. Define the classes and the class hierarchy; 5. Define the properties of classes—slots; 6. Define the facets of the slots and; 7. Create instances.

The focus of the use of ontology in this case is for being the semantics of the agent. The agent will acquire the information from web pages, and then the data will be submitted to the implemented ontology.

5 Semantic Agent of Extraction

The creation of a software agent that aggregates semantically information available on the web in a given domain can bring grants to a computational platform for creating an information environment for decision support giving a through broader view of the internal and external scenarios of information relevance in organizational management.

In this context, we understand the extreme importance of using agents to extract data through scrapper semantic search with the use of technologies like NoSQL persistence in information processing with characteristics of Big Data, essential in the recovery, storage, processing and use of various types of information generated in these environments of large volume data sets on Competitive Intelligence.

In the context of the architecture presented in Figure 1, this research are dealing the problem of automatic and semantic information extraction of web environments that have as informational sources: web pages, web services and database with the development of the agent semantic of data extraction.

This agent should communicate with internal and external information spaces of Big Data basing their search on ontological rules on a metadata standard to perform the semantic extraction of the domain proposed and supported by other systems in a broader context of Information Retrieval.

From this semantic search, the scrapper actuates as a tooling strategy in the search and find the information that really add value to the decision-making process. Inside a huge and massive data structure scattered throughout the web, it is essential that the search engines do not support only syntactic structures of decision in information retrieval, but also in investigations of the use of semantic extraction agents.

Our research uses the domain of higher education institutions as a case study to apply the proposed computing platform in the architecture described in Figure 1. For the development of the prototype of the ontology, we used the issues of scientific research within educational institutions, such as notices, grants, funding agencies, search directories, events, journals, among others.

5.1 Ontological Conceptual Notation

To create the ontology, first it was necessary to check within the domain of scientific research in higher education institutions, which are the classes that are involved in this issue.

It was checked what were these classes, and analyzed the hierarchy between them, based on the experience of the authors ' research and other researchers, and thus was sealed the hierarchy between the classes seen in Figure 2, using mind maps to represent the ontology.

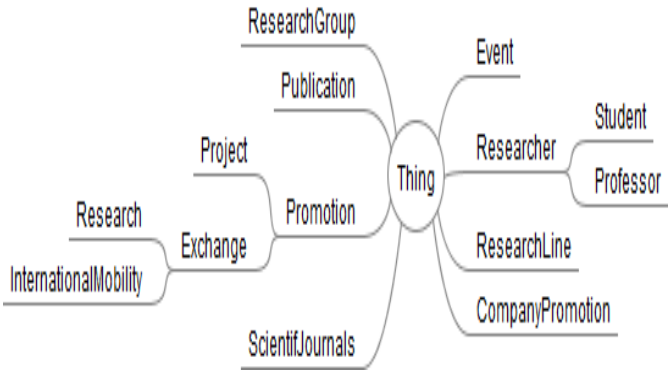


Figure 2: Class Hierarchy of Ontology

It was carried through the mental maps of the ontology schema and was drafted this conceptual notation of ontology, using the Protégé software [14], as shown in Figure 3. It was built the relationship between classes. In this figure the dotted arrows are properties of objects in each class, i.e., when a dotted arrow goes from one class to another, means that the class from which emerged the arrow contains an object of destination class of the arrow.

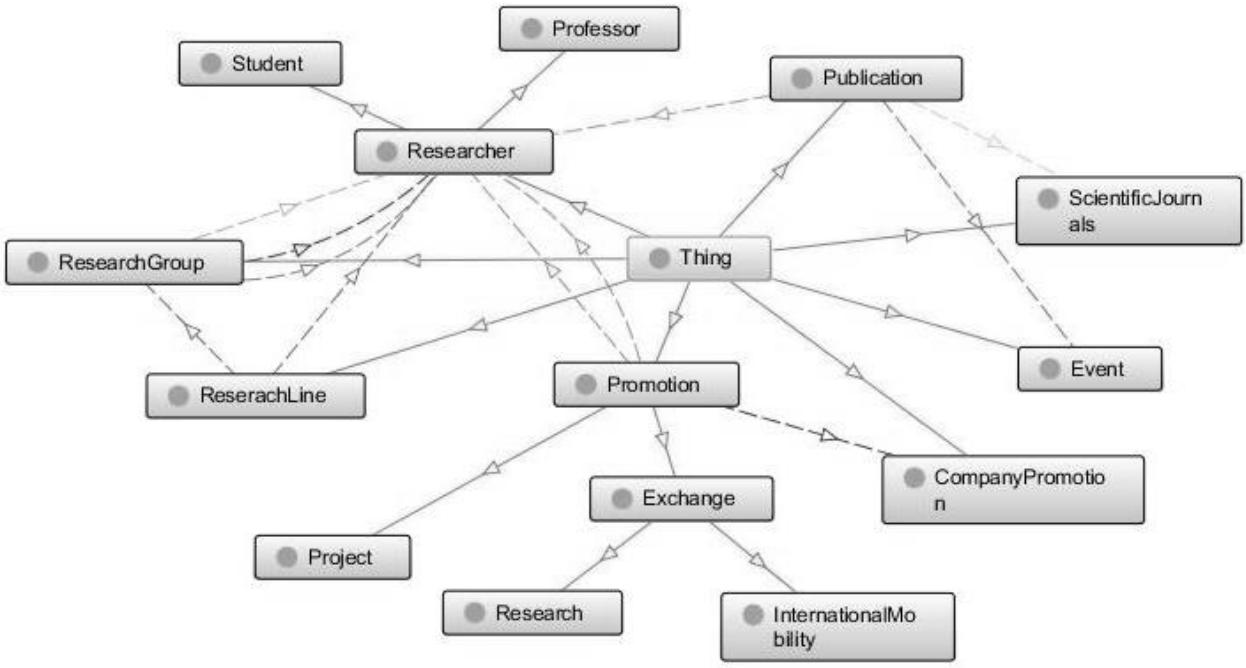


Figure 3: Relationship of classes of ontology of scientific research

After created the class hierarchy of the ontology has been defined all the properties of each class. The properties are shown in Table 2.

Table 2: Slots of ontological classes

Class	Slots	Type
Publication	publication_year	int
	publication_class	string
	publication_title	string
	publication_key_words	string
	publication_type	string
	publication_city	string
ScientificJournals	journal_name	string
	journal_edition	int
	journal_editor	string
	journal_page	String
Event	event_name	string
	event_edition	string
	event_organizer	string
	event_date	date
	event_type	string
Book	book_name	string
	book_chapter	string
	book_date	date
ResearchGroup	group_name	string
	group_area	string
ReseachLine	line_name	string
Researcher	researcher_name	string
	researcher_title	string
	researcher_curriculum	string
	researcher_internal	boolean
Professor	professor_title	string
Student	student_registry	string
Promotion	promotion_value	float
Exchange	exchange_duration	int
	exchange_start	date
Research	research_title	string
	research_area	string
InternationalMobility	mobility_university	string
	mobility_country	string
Project	project_area	string
CompanyPromotion	company_name	string
	company_type	string

The agent will act on this proposed ontology that this scenario is called Task Ontology, according Mizoguch [11].

It is an ontology that solves a specific problem within a domain, that is, solves the problem of scientific research within the domain of an institution of higher education.

We implement the ontology in Protégé software, creating the class diagram and its properties, being implemented in a file OWL (Web Ontology Language) (WC3, 2002H).) There after the Owl2Java [18] tool transformed

OWL in classes Java (Java, 2004); thus making the implemented ontology.

5.2 Semantic Agent Working Method

The agent for performing searches captures information through pre-defined web pages and uses the ontology to classify and make a semantic search.

The figure 4 shows the operating scheme of the semantic extraction agent, which shows the entire process carried out by the system since the retrieval of the information to storage in the database.

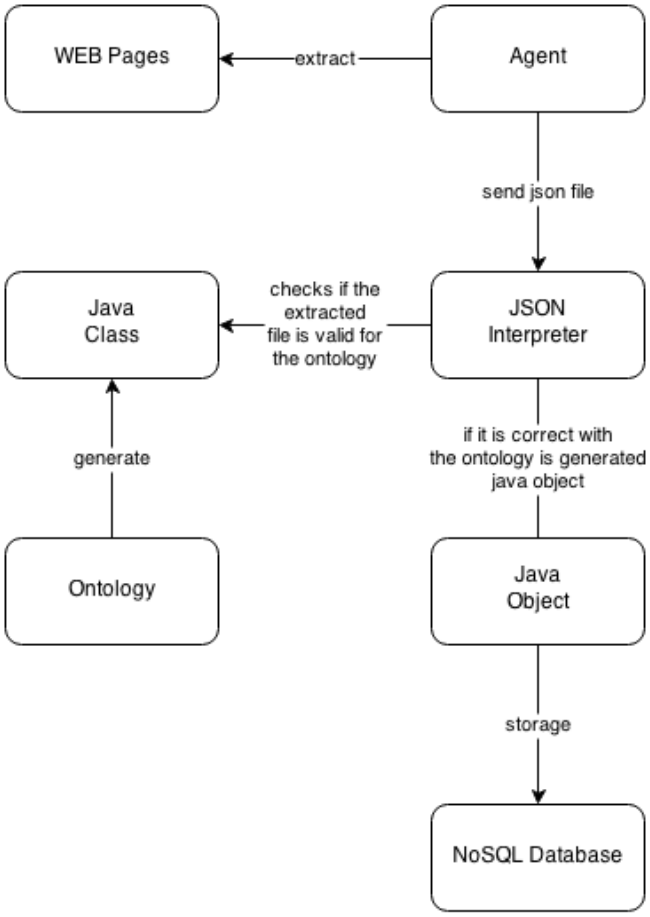


Figure 4 – Schematic of operation of agent

The following describes the sequence of tasks performed by the agent

5.2.1 Information Extraction

The agent will extract pages previously defined data pages. The list of pages that are extracted information were based on domain ontology represents. All of this information is extracted from the HTML format and transformed into a JSON file.

All the extracted data will be placed in a JSON file, and for this there must be a standard metadata to identify and classify each item of information to assist in this process, we use the Dublin Core metadata standard.

The Dublin Core assists as well, can catalog the various information contained in a web page, and thus define the characteristics of each page and this way is possible to use the ontology in a semantic search over these metadata.

5.2.2 JSON Interpreter

JSON interpreter has the function of receiving the JSON file generated by the agent, and interpret to verify that file fits within the ontology.

This process, the interpreter, verifies that the information that is contained within that file, is compatible with the ontology, i.e. will check the semantics of the document, if it indeed fits the ontology

If the JSON interpreter to recognize that given how useful, he will create a Java object based on the Java class created by ontology, so that it can be stored in the database.

5.2.3 Use of Ontology for Information Retrieval

Having the information of web pages with their metadata defined, the use of ontology becomes more viable, because from there the information can be classified by ontology implemented by Owl2Java software, and thus obtaining classified information have each according to their class ontology, and may thereafter be used for competitive intelligence.

5.2.4 Storage in the Database

Because the information is not structured and don't know exactly what will be given and what is the structure of each type of information, by the use of a non-relational database MongoDB. Since the MongoDB is a document-oriented database, and do not need to have a well-defined schema for guards information.

And from the Java object that comes from the JSON parser is done the storage in the database.

6 Acknowledgements

The work presented in the paper was supported by the FAPESP (Fundação de Amparo a Pesquisa do Estado de São Paulo), process 2013/16369-3.

7 Conclusions

From this research it was observed that the use of ontology to trailer search agent is an effective way to get retrieve value information, and manage to get an efficient competitive intelligence.

The ontology can be effective in this process, because it becomes a way to organize the information semantically, and thus, only meaningful information will be used.

In addition, that was achieved through the implementation of the ontology and use it as a reference search for the agent.

Having access to information from your business domain is a fundamental requirement for management and decision making in organizations.

An Information Retrieval system has the ability to provide relevant information for accessing Web sites and services, it is necessary the existence of software agents that add semantic information from various informational sources for a specific domain.

Although the term semantic web be used for a few years, there is still a limitation of its use, because much of the web is still organized in a syntactic way, this semantic agent is a solution to this problem, because the semantics are treated off the web, through the ontology and achieving achieve their results.

8 References

[1] Craig S. Fleisher and David L. Blenkhorn, "Managing Frontiers in Competitive Intelligence". Greenwood Publishing Group, 2001.

[2] D. Graham-Rowe, et. al. "Big data: science in the petabyte era". Nature, 455, 1-50, 2008.

[3] Laney Douglas. "The Importance of 'Big Data': A Definition." Gartner (June 2012), 2012.

[4] Mauricio Diana and Marco A. Gerosa, "NOSQL na Web 2.0: Um Estudo Comparativo de Bancos Não-Relacionais para Armazenamento de Dados na Web 2.0" ("NoSQL Web 2.0: A Comparative Study of Non-Relational Data Storage Benches for Web 2.0"). São Paulo, 2010.

[5] Janice I. Deters and Silsomar F. Adaime, "Um estudo comparativo dos sistemas de busca na web" ("A comparative study of search systems on the web"), Anais do V Encontro de Estudantes de Informática do Tocantins. Palmas, TO, 189-200, 2003.

[6] Tércio. M. S. Silva, "Extração De Informação Para Busca Semântica Na Web Baseada Em Ontologias" ("Information Extraction for Semantic Search In Web Based On Ontology"). Florianópolis, 2003.

[7] Marco A. A. Mesquita, "Web Semântica E Recuperação Da Informação Na Internet : O Que Esperar Do Futuro?" ("Semantic web and information retrieval on the Internet: What to Expect From the Future?"), 2010.

[8] Natalya F. Noy and Deborah L. McGuiness, "Ontology Development 101: A Guide to Creating Your First Ontology". <http://www.ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness-abstract.html> [retrieved: 05/27/2014], 2001.

[9] Thomas R. Gruber, "Towards Principles for a Design of Ontologies Used for Knowledge Sharing", International Journal of Human and Computer Studies, 907-928. 1995

[10] D. Clark, "Mad cows, meta-thesaurus and meaning, IEEE Intelligent Systems". 1999.

[11] Riichiro Mizoguchi, "Tutorial on Ontological Engineering". NEW GENERATION COMPUTING-TOKYO- 21.4, 363-364, 2003.

[12] J. Davies, D. Fensel and F. Van Harmelen, "Towards The Semantic Web: Ontology-Driven Knowledge Management", John Wiley & Sons Ltd, 2003.

[13] Ricardo A. Falbo, et al., "ODE: Ontology-based software Development Environment". IX Congreso Argentino de Ciencias de la Computación, p. 1124-1135, La Plata, Argentina, 2003.

[14] Protégé. Stanford University. <http://protege.stanford.edu/> [retrieved: 03/10/2014].

[15] John E. Prescott. "The Evolution of Competitive Intelligence", International Review of Strategic Management v. 6, 71-90, 1995.

[16] Fabiano D. Bepler, et al. "Uma Arquitetura Para Recuperação De Informação Aplicada Ao Processo De Cooperação Universidade-Empresa" ("An Architecture for Retrieval of Applied Information In Case Of University-Industry Cooperation"), São Paulo, 2005.

[17] Kevin Wiesner, et al. "Recovery Mechanisms for Semantic Web Services" DAIS 2008, LNCS 5053, pp. 100–105, 2008.

[18] Owl2Java.<http://www.incunabulum.de/projects/it/owl2java> [retrieved: 03/14/2014].

Open *km*-Learning: Using Semantic Web Technologies to Facilitate E-Learning

Grace Zhao[1], **Xiaowen Zhang**[2], **Abdullah Uz Tansel**[3]
[1]Computer Science, Graduate Center, CUNY, New York, NY, USA
[2]Computer Science, College of Staten Island, Staten Island, NY, USA
[3]Computer Information Systems, Baruch College, New York, NY, USA

Abstract— *We propose an e-learning framework that takes domain knowledge-specific Open Educational Resources (OERs) from the World Wide Web and stores the metadata of the top qualified entries into an ontology-driven structural data store. When responding to users' learning queries, the framework constructs a cluster of related subjects around the query topic, including the topic itself, to form a knowledge map, along with the related Learning Objects*[1] *(LOs). We call this e-learning framework Open km-Learning (OkL), where km stands for "knowledge map."*

Keywords: E-Learning Platform, Semantic Web, Learning Object, Knowledge Map, Ontology

1. INTRODUCTION

Open Learning Objects (LOs) are available on the Internet in many forms, such as Massive Open Online Courses (MOOCs), individual LOs, discussion rooms on given subjects, or simply definitions of particular terms. This paper proposes a novel e-learning platform, Open *km*-Learning (O*k*L), to reinforce learning by providing all forms of quality open LOs relevant to a specific topic of interest, as well as a list of *related* topics. We explain the term 'related' in detail in Section 4. Semantic Web (SW) technologies, especially ontologies, make it possible to conceptualize a learning path – a graph of related topics, in other words, a knowledge map.

We will use Information Security[2] in this paper as an example of domain knowledge to illustrate the framework.

The remainder of this paper is organized as follows: Section 2 gives a brief overview of some related works. Section 3 identifies and describes the e-learning components of the O*k*L framework. Section 4 discusses the Semantic Web aspects of the platform. In Section 5, we propose our O*k*L framework and present some of its key features. The last section concludes our paper, addresses challenges, and offers directions for future study.

2. RELATED WORKS

Ever since the inception of the Semantic Web, many researchers have studied the benefits, possibilities, and methods for combining e-learning with SW technologies.

Keenoy et al. strove to use Semantic Web technologies to syndicate and personalize educational resources [5] [6]. The Self E-Learning Networks (SeLeNe), proposed in their work package, carried personalized learning through distributed and autonomous RDF[3]/RDFS[4] creation and management.

Rashid, Khan and Ahmed introduced an ontology-based e-learning management system model [10] addressing course syllabus, teaching methods, learning activities and learning styles. In this model, Administration, Instructor, and Learner are inter-related through Learning Resource and Ontology-based Contextual Knowledge. However, the paper does not provide any Semantic Web implementation details.

Bansel and Chawla [1] proposed an approach to move from content-focused learning services to semantic-aware and personalized learning services. The paper focuses on knowledge representation techniques in semantic e-learning. It also briefly addresses Semantic Web-based intelligent information retrieval method.

In sum, most of the studies within the Semantic Web e-learning literature tend to use SW technologies to tackle the full spectrum of an e-learning system, from pedagogy, assessment, and user profiling to administrative activities. The painstaking efforts demonstrated in the work of Keenoy et al. show a glimpse of the complexities involved in building a semantic e-learning system. In addition, it is difficult to articulate a feasible and salient framework out of the platforms delineated in most of the studies, because they are either too abstract or too specific.

[1]A "Learning Object" was defined by Beck [3] as âĂIJa collection of content items, practice items, and assessment items that are combined based on a single learning objective.âĂİ In this paper, we define an LO as any self-contained, re-usable, and web-representable unit of learning. An LO could be an hour-long video lecture, a three-webpage tutorial, a podcast, or a full-length research paper in PDF format. The metadata and supporting materials (practice, assessment, copyright, etc.) bound with a learning resource are also part of the LO.

[2]Information security, sometimes shortened to InfoSec, is the practice of defending information from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction.

[3]The Resource Description Framework (RDF) is a family of World Wide Web Consortium specifications originally designed as a metadata data model.

[4]RDF Schema (RDFS) is a set of classes with certain properties using the RDF extensible knowledge representation language, providing basic elements for the description of ontologies, otherwise called RDF vocabularies, intended to structure RDF resources.

This paper attempts an empirical approach to define a semantic e-learning platform – O*k*L, to achieve a presentable and feasible route to semantic e-learning.

3. THE E-LEARNING COMPONENTS OF O*k*L

An e-learning system refers to the automation of the processes of learning and training through the use of information technology [11]. Khan [7] identified eight dimensions of an e-learning framework: pedagogical, technological, interface design, evaluation, management, resource support, ethical, and institutional.

Since the O*k*L platform does not manufacture LOs, but rather re-assembles and disseminates LOs that are available on the Web, we will focus our efforts only on technological, management, and interface design perspectives. Technological and interface design involves the technical infrastructure and representation of functional aspects of the system, respectively. The management of e-learning refers to the storage, distribution, and maintenance of learning data and information structure, the focal point of the O*k*L platform.

4. THE SEMANTIC WEB COMPONENTS OF O*k*L

In the O*k*L platform, we adopt the essential elements of SW technologies: ontology, vocabularies, Resource Description Framework (RDF) and Triple Store[5].

Ontology is notoriously difficult to design and create, yet it is the 'yoke' of a SW platform. It usually requires domain experts to devote a tremendous amount of time and effort to building the initial version, and the maintenance of the ontology is no easier. In an effort to ameliorate the complexities and errors of building an ontology graph, we limit the nodes of an ontology to class (*rdfs:Class*, *owl:equivalentClass*), subclass (*rdfs:subClassOf*), and instance (*rdf:type*) only. As a result, the output graph is a hierarchical acyclic structure. We created an ontology of Information Security Encryption Algorithm (Fig. 1) to demonstrate the hierarchical structure. The root node is the super class, the child nodes are the subclasses, and the leaves are instances.

However, as we increase the scope of a knowledge domain (e.g., domain of Information Security vs. Information Security Encryption Algorithm), more concepts will be introduced. It is challenging, if not impossible, to include each additional concept into the same hierarchical tree. But a more logical and empirically sound approach is first to identify and create sub-trees within each sub-domain of knowledge while preserving the hierarchical knowledge structure. Then, we can selectively link the sub-trees, based on relationship patterns within the specific rule set.

[5]A Triple Store is a purpose-built database for the storage and retrieval of Resource Description Framework metadata.

The design guidelines for building an ontology of a knowledge domain can be generalized as follows:

We identify and construct all sub-trees in the knowledge domain. Each sub-tree represents an ontology, containing one root node and at least one subclass or instance. We then examine the relationship among the root nodes of all sub-trees. If there exists a relationship between the root nodes of two subtrees, we use a generic properties entry, *:isRelatedTo*, to connect the two root nodes. (We will not consider any relationship existing in non-root nodes.) After completing all operations, we should be left with either a tree or a forest.

Based on the ontology structure described above, we are ready to introduce the concept of a *knowledge map*.

We will map the query topic to one of the nodes in the ontology graph. We call the node the *anchor node*. Next, we seek the following nodes in relation to the anchor node in the ontology graph:

- *Parent node*: immediate superclass of the anchor node, usually indicating a more general concept.
- *Child node*: immediate subclass/instance of the anchor node, usually indicating a more specific concept.
- *Sibling node*: subclass/instance derived from the same immediate superclass of the anchor node; or a class with '*:isRelatedTo*' property associated with the anchor node.

We define the above operation more rigorously below:

Definition 1: Let G be a rooted graph whose root is rN. Let n be a node (anchor node) of G. There may be more than one path from n to rN. Let $f_p : G \backslash rN \rightarrow G$ be the mapping defined by:

$$f_p(n) := \{x| \text{ node adjacent to } n \text{ on the path to } rN\}. \quad (1)$$

Then $f_p(n)$ is the **parent node set** of n.
The **sibling node set** of n is defined as:

$$\Phi_s(n) := \{x|y \in f_p(n), y \in f_p(x)\}. \quad (2)$$

Subsequently, the **child node set** of n will be:

$$\Phi_c(n) := \{x|n \in f_p(x)\}. \quad (3)$$

Let $\Phi(n)$ be the set of all nodes in the knowledge map of n; then:

$$\Phi(n) := \{f_p(n) \cup \Phi_s(n) \cup \Phi_c(n) \cup \{n\}\}. \quad (4)$$

Another case for the sibling node should be also included in the knowledge map – if two root nodes of two sub-trees are connected with the properties entry '*:isRelatedTo*,' the two root nodes are siblings of each other.

Definition 2: Let $G = (V, E)$ be an undirected graph, where V is the vertex set, and E is the edge set, of G. Let G_1, G_2 denote two rooted graphs whose roots are rN_1 and rN_2, respectively, and $rN_1, rN_2 \in V, G_1, G_2 \subset G$.
If $(rN_1, rN_2) \in E$, then rN_1 is the sibling node of rN_2. Since G is an undirected graph, it implies $(rN_2, rN_1) \in E$, thus rN_2 is the sibling node of rN_1.

However, it should be noted that an anchor node will not necessarily have all three types of node sets, though it will have at least one of the three. Therefore, a minimum knowledge map should contain two nodes: the anchor node and one of the three node types. In addition, an anchor node may appear in more than one sub-trees in the ontology graph. (In this case, we will then construct more than one corresponding knowledge map.)

All nodes in the ontology graph will be defined in the controlled vocabularies. The metadata of each LO will be saved in RDF format in order to be stored into the Triple Store.

5. THE O*k*L FRAMEWORK

The framework is a logical and functional conceptual model that represents the information flow and linkages between various modules and main processes. Any subsequent system architecture should be derivable from this logical framework.

Fig.3 below illustrates the landscape of the O*k*L platform. We briefly explain each step delineated in the diagram, so as to present the entire workflow of the e-learning conceptual system.

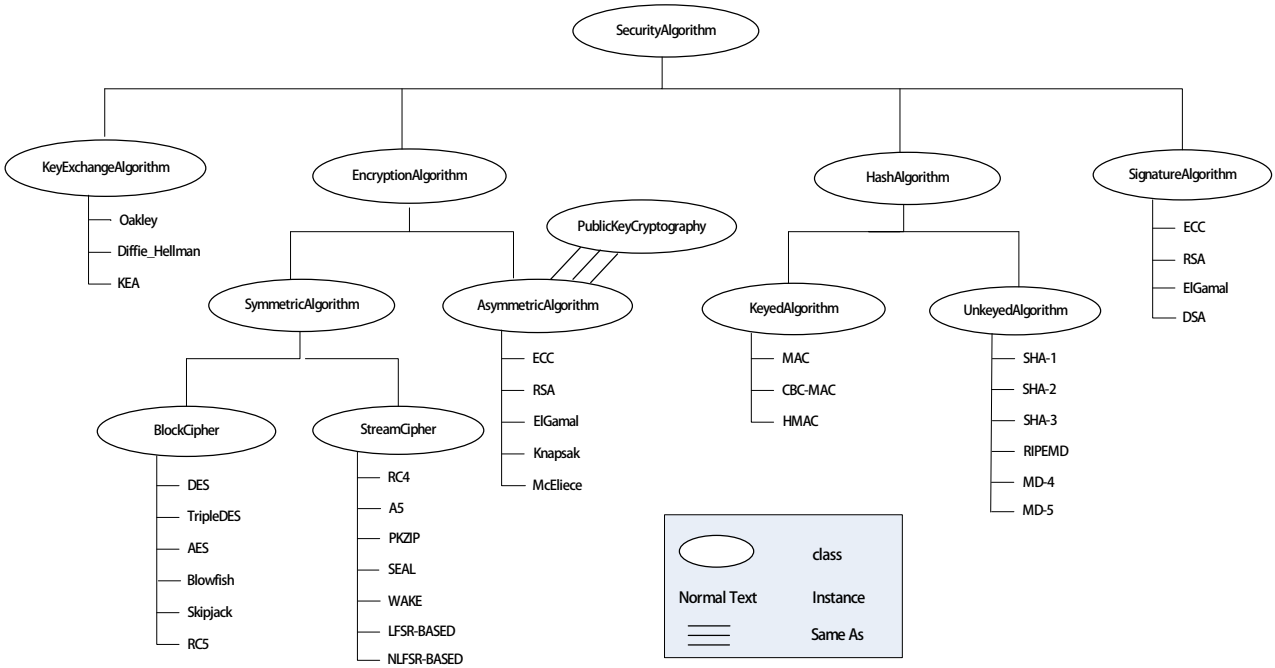


Fig. 1: Ontology of Information Security Encryption Algorithm.

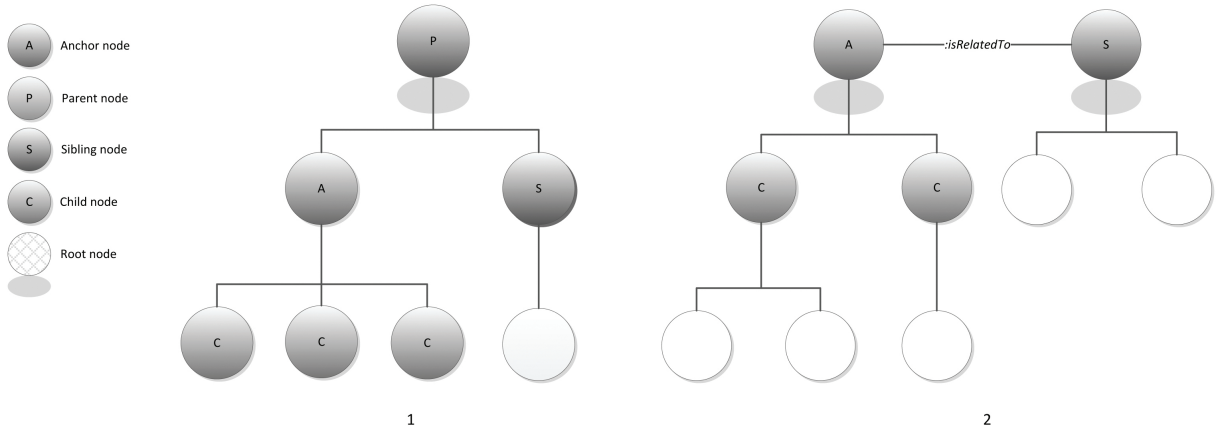


Fig. 2: Illustrations of the *anchor node*'s *related* nodes.

5.1 Building the Knowledge Base

Step 1: Crawling for LOs from the Web (1)

In this step, a crawler is to collect OERs through top cross-domain knowledge search engines, such as Google, YouTube, etc, and authoritative domain knowledge-specific sites. We collect only ω number of top search results (ω is a small integer parameter) for further processing.

Prior condition: The query subject should match one of the nodes in the ontology graph.

Step 2: Parsing and processing metadata (2)

After obtaining the LO URLs, we retrieve the related pages and extract the metadata of the LOs, including generic web metadata, custom metadata, and possible SW metadata.

Step 3: Filtering and re-ranking LOs (3)

In this step, the Intelligence Engine kicks in. Based on the metadata of each LO, the engine decides whether the LO is relevant to the query subject and to what degree the LO meets other criteria (if any), such as a rating, length, and complexity.

The related algorithms of filtering and re-ranking LOs are outside the scope of this paper.

Step 4: Creating RDF entries (4)

In this step, we create an RDF file for each LO, based on the predefined LO RDF template.

Step 5: Storing LO RDF in the Triple Store (5)

We use the Triple Store API to store the LO RDF into the graph database.

Steps 1 – 5 will be a recurring process running in the background on a repeated schedule, as a means to update LO records and to ensure the validity of LO URLs.

5.2 Semantic Web Maintenance

Step 1: Maintaining the accuracy and currency of ontologies and vocabularies (6)

This step can be managed through machine learning techniques and user contributions.

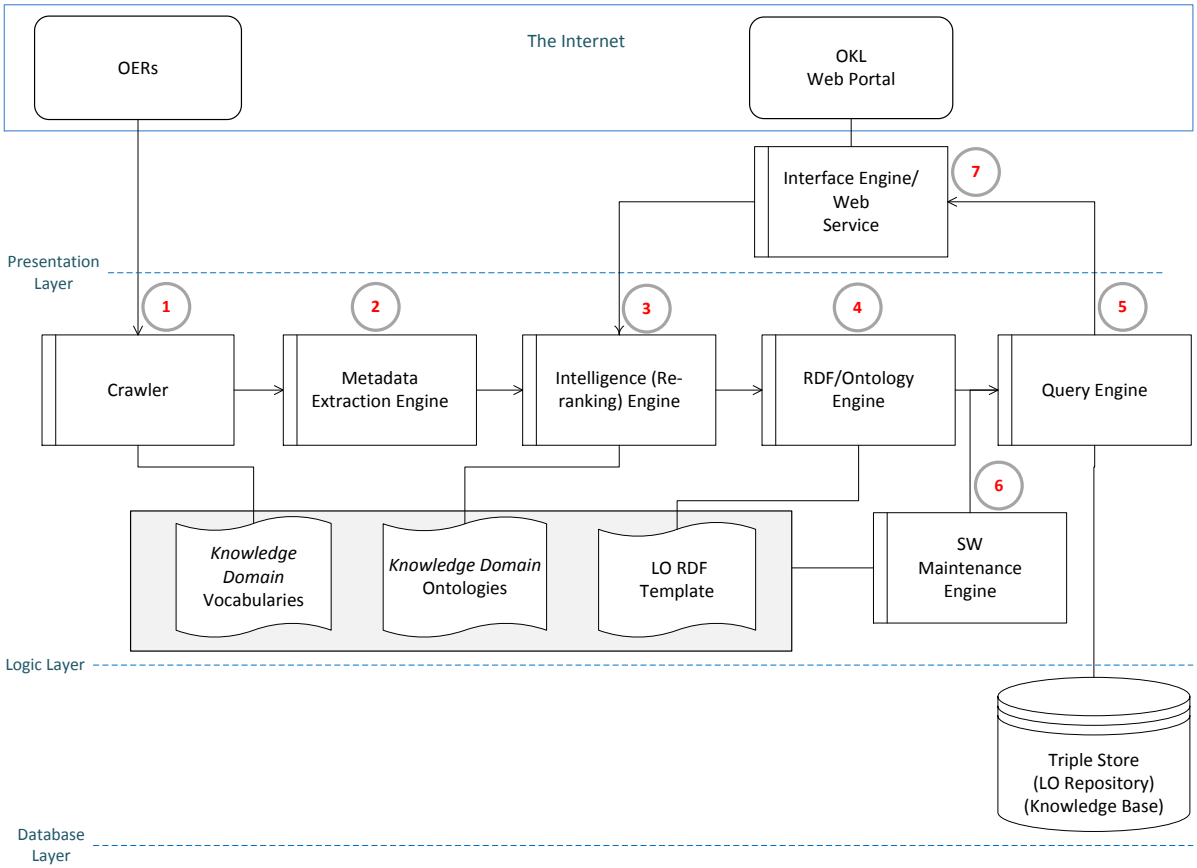


Fig. 3: O*k*L platform conceptual design block diagram.

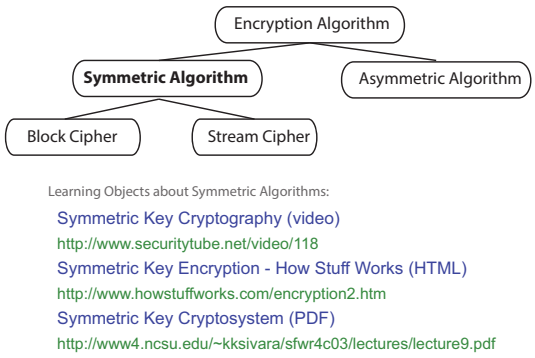


Fig. 4: A sample screen of the knowledge map and the recommended LO links for the *anchor node* 'Symmetric Algorithm.'

5.3 Querying the Knowledge Base

Step 1: Ontology node mapping (7) (3)

After the user or the crawler issues a query, we map the query string to an ontology node.

Step 2: Parsing ontology to obtain the 'knowledge map' node set (4)

In this step, we will build the mini knowledge map around the query string.

Step 3: LO retrieval (5)

In this step, we query the Triple Store using SPARQL and get the LO result set.

Step 4: Generating Interface (7)

We construct the knowledge map and display related LO links.

For example, based on our Information Security Encryption Algorithm Ontology (Fig. 1), if a user searches for 'Symmetric Algorithm,' the application will display a list of LOs concerning 'Symmetric Algorithm.' It will also provide the learner with a diagram of concepts *related* to 'Symmetric Algorithm': the more generic concept related to 'Symmetric Algorithm' is 'EncryptionAlgorithm,' while the more specific concepts are 'BlockCipher' and 'StreamCipher.' The correlated or parallel concepts are those sibling class/instance nodes of 'SymmetricAlgorithm' in the ontology, namely, 'AsymmetricAlgorithm.' The interface is illustrated in Fig. 4.

6. CONCLUSION

The key elements of the O*k*L platform are the domain knowledge ontologies and the knowledge maps deduced from the ontology graphs. We aim to construct these elements in an efficient and effective way and to provide the web learners with an enhanced learning experience.

There are many challenges ahead, such as designing the system at a more granular level in terms of defining the re-ranking and recommendation algorithms, ontology adjustment/updating algorithms, etc.

We hope the O*k*L framework can help to streamline the practice of applying Semantic Web Technologies into e-learning platforms.

References

[1] Bansal, R., and Chawla, C.: Inculcating Intelligence into E-Learning. International Journal of Computer Applications & Information Technology, November 2012 I(III), 4-9. ISSN: 2278–7720 (2012)

[2] Barros, H., Silva, A., Costa, E., Bittencourt, I. I., Holanda, O., and Sales, L.: Steps, Techniques, and Technologies for the Development of Intelligent Applications Based on Semantic Web Services: A case study in e-learning systems. Engineering Applications of Artificial Intelligence, 24(8), 1355–1367 (2011)

[3] Beck, Robert J.: What Are Learning Objects? Learning Objects, Center for International Education, University of Wisconsin-Milwaukee (2009)

[4] Herzog, A., Shahmehri, N., and Duma, C.: An Ontology of Information Security. International Journal of Information Security and Privacy (IJISP), 1(4), 1-23 (2007)

[5] Keenoy, K., Poulovassilis, A., Christophides, V., Rigaux, P., Papamarkos, G., Magkanaraki, and A., Wood, P.: Personalisation Services for Self E-Learning Networks. In Web Engineering. Springer Berlin Heidelberg, 2004, 215-219 (2004)

[6] Keenoy, K., Levene, and M., Peterson, D.: Personalisation and Trails in Self E-Learning Networks. SeLeNe | Self E-Learning Networks, IST-2001-39045. WP4 Deliverable 4.2 (2004)

[7] Khan, B.: Learning Features in an Open, Flexible and Distributed Environment. AACE Journal, 13(2), 2005, 137-153 (2005)

[8] Kim, A., Luo, J., and Kang, M.: Security Ontology for Annotating Resources. Springer Berlin Heidelberg, 2005, 1483-1499 (2005)

[9] Pidcock, W.: What Are the Differences Between A Vocabulary, A Taxonomy, A Thesaurus, An Ontology, and A Meta-Model? 2003. Retrieved September 9, 2013, from Unicam University: http://www1.cs.unicam.it/insegnamenti/reti_2008/Readings/vocabolarioTassonomiaTesauroOntologia.pdf

[10] Rashid, S., Khan, R., and Ahmed, F.: A Proposed Model of E-Learning Management System Using Semantic Web Technology, 2013. Retrieved September 9, 2013, from Semantic Web Journal: http://semantic-web-journal.org/sites/default/files/swj234.pdf

[11] Robson, R.: Explaining E-Learning to A Stranger. E-learning Magazine, March 2002, 49 (2002)

[12] Takahashi, Y., Abiko, T., Negishi, E., Itabashi, G., Kato, Y., Takahashi, K., and Shiratori, N.: An Ontology-Based E-Learning System for Network Security. Advanced Information Networking and Applications, 2005. AINA 2005. 19th International Conference on, vol. 1 Vol. 1, March 2005. 197-202 (2005)

The Use of Semantic-based Suggestions for Web Service Composition

Diogo Phelipe Busanello da Silva and Frank Siqueira
Department of Informatics and Statistics, Federal University of Santa Catarina
Florianópolis, Santa Catarina, Brazil

Abstract - *The creation of complex SOA applications requires multiple services to be combined. Building service compositions, though, requires analyzing if inputs and outputs of composed services are compatible, and this is a costly task to be done manually. The semantic description of Web Services using ontology concepts to describe the provided operations, inputs and outputs, is able to facilitate the service composition process. This paper describes a mechanism that helps the user to build a service composition, providing service recommendations based on the semantic description of Web services. The semantic service suggestion mechanism is integrated with a composition tool, called CVFlow, in order to provide a complete support for composition developers.*

Keywords: Semantic Web Services; Service Composition; SAWSDL.

1 Introduction

Web Services is a well-known technology for building and reusing services in a SOA-based environment. Although this technology provides means to describe the abstract functionality of services and to specify how they can be invoked, this description has merely syntactic meaning for service consumers running on remote machines. Therefore, in order to consume the provided service, it is still necessary the intervention of a developer, who assimilates the meaning of service operations and the corresponding inputs and outputs and builds the code to invoke the service properly.

Semantic Web Services are a merge of the Web Services technology with the concept of Semantic Web defined by Tim Berners-Lee [1]. With the association of these technologies, Web Services become meaningful for other machines and the automation of Web Service compositions is made possible.

This paper proposes a semi-automated approach to compose Web Services based on a semantic service suggestion mechanism, called S^3M, which is integrated with an existing workflow tool. SAWSDL (Semantic Annotation for WSDL and XML Schema) was chosen as underlying semantic technology due to its recent adoption as a W3C standard [2]. Recommendations are provided based on the semantic match between inputs required by services stored in a repository and a list of outputs produced by services in a partially built composition.

The remainder of this paper is organized as follows. Section 2 presents the existing technology to describe and compose Semantic Web Services. Section 3 introduces the semantic service suggestion mechanism proposed in this paper and presents its architecture and selection algorithms. Section 4 shows how the proposed service suggestion mechanism is integrated with a composition tool, called CVFlow. Section 5 discusses related proposals found in the literature and compares them with the service suggestion mechanism described in this paper. Finally, Section 6 presents the concluding remarks and proposes some future work in this research field.

2 Technological background

2.1 SAWSDL

The lack of semantic support on the Web Service Description Language (WSDL) is an obstacle for automating service discovery and composition. Ambiguities may arise while a composition is being built, given that type compatibility between input and response messages is neither sufficient nor necessary to guarantee that they are semantically equivalent. A response message produced by a service may have the same type of an input message required by another one, and despite this, data semantics may be completely different. On the other hand, response and input messages may have different identifiers and/or types but still be semantically compatible.

SAWSDL [2] is one of the existing solutions for the lack of semantic information on service descriptions. It is based on a previous proposal entitled WSDL-S [3], and was recently published as a W3C standard.

SAWSDL by itself does not provide any specific semantics, but allows the service developer to add annotations to elements described in purely syntactic WSDL descriptions that associate them with semantic concepts defined by ontologies [4]. This is the main difference between SAWSL and other approaches, such as OWL-S (Web Ontology Language for Web Services) [5] and WSMO (Web Service

Modeling Ontology) [6], i.e., it does not create another language; instead, it is just an extension of the WSDL specification [7].

The attributes defined by SAWSDL to enable semantic annotation on WSDL elements are summarized by Table 1 and described in the following paragraphs [4].

Table 1. SAWSDL Syntax Summary

Name	*Description*
modelReference	Reference to concepts in a semantic model
liftingSchema Mapping	List of pointers to data-lifting transformations
loweringSchem a Mapping	List of pointers to data-lowering transformations
attrExtensions	Attaches attribute extensions where only element extension is allowed

The *modelReference* attribute allows multiple annotations to be associated with a given WSDL component through URIs (Uniform Resource Identifiers) that identify concepts in a semantic model [2].

The *liftingSchemaMapping* and *loweringSchema-Mapping* annotations address post-service discovery issues. The former converts XML data obtained from a service into a semantic model, while the latter maps a semantic model into XML data [4].

The former attributes are valid for all versions of WSDL. The *attrExtensions* attribute is used in WSDL 1.1 files, in which the extensibility attribute is not allowed, but element extensibility is.

2.2 Web service composition

Web Service composition is a process that combines and binds services, either atomic or composed ones, in order to create a new service. Different techniques to compose Web Services can be found in the literature. This process is usually classified into two categories: dynamicity and automation [8].

Compositions may be classified as either static or dynamic. Static composition takes place during the design phase, while the components are chosen, linked together, compiled and deployed. This technique works fine as long the components do not change. On the other hand, a dynamic composition occurs at request time and is more adaptable to changes in the employed web services [9].

The second classification category is based on the degree of automation with which compositions are built. A manual composition relies on the developer to combine services. An automated composition is the process that selects, combines, integrates, and executes services automatically [10]. Even an automatic approach relies on the user to specify parameters such as the existing inputs, desired operations and required outputs. There is also a semi-automatic approach that provides an intermediate degree of automation, with which the composition is built using a GUI and, as the user creates the composition, the tool suggests services that can be composed based on their characteristics.

The use of semantic annotations in Web Service descriptions allows the automation of the composition process. There are two aspects that directly affect the automatic service composition: the service description technology and the matching between services.

The most adopted standard for Web Service description (i.e., WSDL) is only capable of describing service interfaces syntactically. In order to avoid this limitation, the research community created ways to add support for semantic technology to describe services (e.g., OWL-S, WSMO, SAWSDL). This is done by associating vocabulary terms defined in a domain ontology with operations provided by a service and the corresponding input and output messages. The more detailed the domain ontology is, the more accurate the inference process of operations, inputs, and outputs becomes.

Semantically described services may be subject to semantic matching, in order to verify the compatibility of two operations by comparing the outputs produced by a given operation with inputs required by the other operation.

Fig. 1 shows an example of match combining three operations. In the portrayed example, output *m_out* matches with input *o_in1*, and output *n_out* matches with input *o_in2*, allowing operations *M* and *N* to be composed with operation *O*.

Paollucci et al. [13] defined a set of criteria to measure the similarity between the output of one operation (*Op1*) and the input of another one (*Op2*). A match is considered exact when the input required by *Op2* and the output produced by *Op1* are associated with the same ontological concept. When the output of *Op1* is an ontological subclass of the input of *Op2* it is considered a plugin match. The subsumption match in the opposite case, when the output of *Op1* is an ontological superclass of the input of *Op2*. When there is no correlation between the (input, output) pair, the similarity is considered a fail. Table 2 associates these similarity levels with values that are employed in this work to calculate the semantic mismatch between operations that are candidates to be composed.

Table 2. Similarity between operations

Similarity	*Value*
Exact	0
Plugin	1
Subsume	2
Fail	3

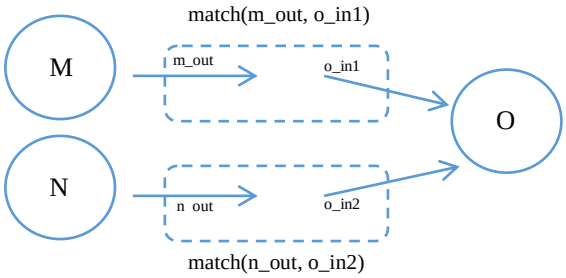


Fig. 1. Semantic match between operations

3 The semantic service suggestion mechanism

The service composition approach proposed in this paper is based on the use of semantic technology aiming to create a service suggestion mechanism, which selects services from a repository with inputs that semantically match outputs produced by services that are part of a composition that is being designed.

3.1 Architecture

The architecture of the Semantic Service Suggestion Mechanism, from now on called S^{3}M, is illustrated by Fig. 2. The figure shows two user profiles: the provider, which is responsible for registering services in the repository; and the composer, which receives suggestions from the S^{3}M mechanism and creates the service composition. Besides, eight software components are shown in the figure. These components will be described in the following paragraphs.

The service registration interface is provided by the *Semantic Web Service Registry*, which takes semantically annotated WSDL files and extracts their operations, inputs, outputs and the corresponding semantic annotations, and stores them in the *SAWSDL Repository*. This is done to speed up the service discovery and suggestion procedure, since querying the database is much faster than parsing WSDL files every time a new suggestion is requested. This approach is also adopted in [12]. Fig. 3 shows the entity-relationship model of the database built within the SAWSDL Repository. This component also executes a procedure that checks if registered services are running and updates the availability metric stored in the Service table.

The *Service Composer* provides the user interface for the composition designer. This component is responsible for arranging the composition in a linked graph of connected operations. In the beginning of the design process, the composition has only a set of input parameters provided by the user. These inputs are consumed by operations that are added to the composition, which provide outputs that can be consumed by other operations, and so on.

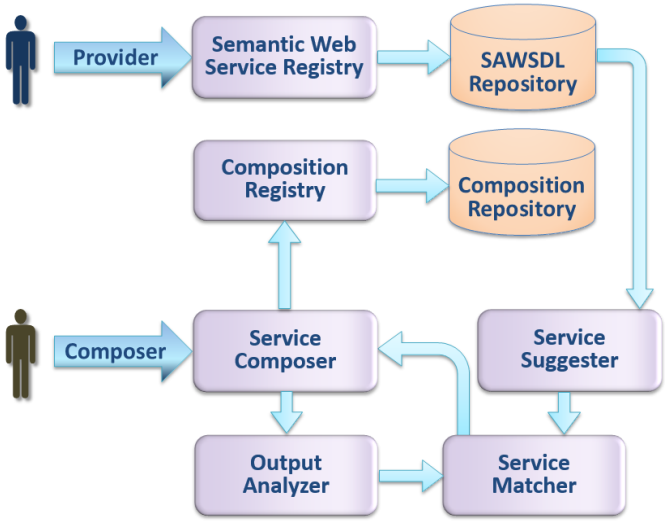


Fig. 2. Architecture of the Semantic Service Suggestion Mechanism (S^{3}M)

The *Output Analyzer* checks the list of outputs available in the composition that is being built and passes them to the *Service Matcher* module, which identifies operations with inputs that are semantically compatible with them.

The *Service Suggester* takes all operations with inputs that are semantically compatible with the outputs produced so far by the composition and passes them to the Service Matcher module, which orders operations executing the algorithm that will be described in the remainder of this chapter. The ordered list is provided to the Service Composer and then presented to the user, which can select a new service from the list and add it to the composition.

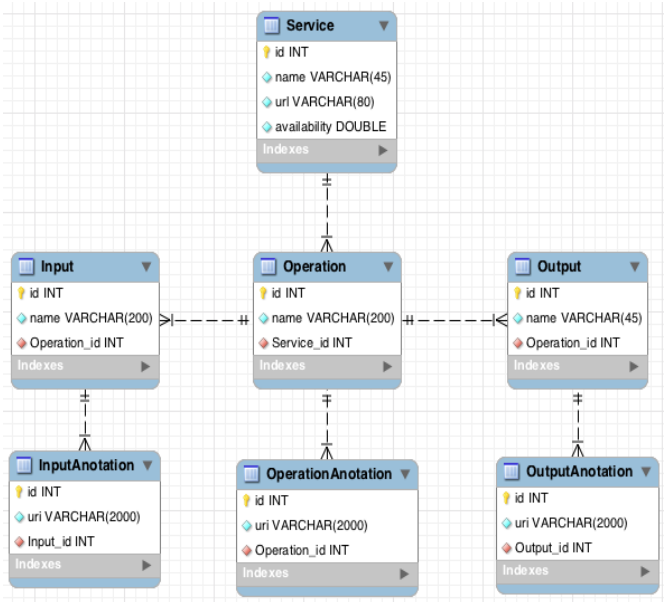


Fig. 3. Entity-relationship model of the SAWSDL Repository

```
01 function semanticMatch(outputList, inputList)
02    def matchList = ∅, mismatchValue = none
03    for each input in inputList do
04       mismatchValue = Match.FAIL
05       for each output in outputList do
06          mismatchValue = min(mismatchValue, mismatch(output, input))
07       end for
08       if mismatchValue in [Match.EXACT, Match.PLUGIN] then
10          matchList.add ( input )
11       end if
12    end for
13    return matchList
14 end
```

Fig. 4. Pseudo code of the Semantic Match Algorithm.

Finally, the *Composition Registry* is responsible for persisting designed compositions in the *Composition Repository*. This last component provides the information needed for executing the composition. Execution is done by a third-party tool that is out of the scope of this paper.

3.2 Availability function

A process executed in the background by the SAWSDL Repository updates the metric that is stored in the availability column of the Service table. This metric takes into account the availability history of each service stored in the repository. In the first two attempts to monitor a service, available services receive value 1, while services that are offline receive value 0. A weighted mean is calculated in subsequent monitoring intervals, giving increasing relevance to recent attempts than to past ones. The following description shows how the weighted mean is calculated by the availability function.

$$m_n = 1 \text{ if available, } 0 \text{ otherwise}$$
$$av_1 = m_1$$
$$av_2 = m_2$$
$$\dots$$
$$av_n = \omega * m_n + (1- \omega) * av_{n-1} \quad (1)$$

where:

m_n: result of the monitoring procedure on time n;

av_n: availability metric on time n;

ω: weight given to the last monitoring operation.

The weight factor ω is a value in the (0, 1) range that defines the proportion in which the metric considers the most recent monitoring procedure and the past ones. For example, with ω=0.3, the result of the most recent monitoring procedure is given 30% of the weight, while all previous values account for 70% of the availability metric.

Equation (1) is periodically calculated to obtain the availability of each service registered in the SAWSDL Repository. The user is free to adjust the monitoring period and the weight factor.

3.3 Semantic match algorithm

The algorithm adopts the classification proposed in [13] to verify the semantic compatibility between an input, output pair. This is represented by Equation 2:

$$SM(A,B)=\{x \mid x \in (exact, plugin, subsume, fail)\} \quad (2)$$

where $SM(A,B)$ assumes the mismatch value taken from Table 2 for the pair A and B.

Fig. 4 shows the pseudo code that identifies matches between a list of inputs and a list of outputs, associating a match value with each acceptable pair. The match value assumes the minimum value between the match of the input and each output, representing the value of the closest match.

The command in line 07 verifies if the comparison between semantic concepts associated with a given input, output pair results in an exact or plugin match. The algorithm keeps looking for better matches for inputs that do not have an exact match. After all input, output pairs are semantically analyzed, a list with all matching inputs found is returned by the algorithm (line 13).

3.4 Suggestion algorithm

The semantic service suggestion algorithm is the core function executed by the Service Matcher. Its pseudo code is presented in Fig. 5.

The algorithm receives as parameters an operation that is the focus of the matching procedure, a list of outputs available in the composition and a list of candidate operations provided by the Service Suggester to be evaluated as possible matches. It then begins by stepping through each operation (line 03), and then verifies the availability of the service that provides

```
01 function semanticSuggestion(focusOperation, outputList, operationList)
02    def suggestList = ∅
03    for each operation in operationList do
04       if getAvailability(operation.service) >= 0.5 then
05          smFocus = semanticMatch(focusOperation.outputs, operation.inputs)
06          if smFocus.matchList ≠ ∅ then
07             smOutputs = semanticMatch(outputList, operation.inputs – smFocus.matchList)
08             suitability = calcCSo(operation, smFocus.matchList ∪ smOutputs.matchList)
09             suggestList.add(operation,  suitability)
10          end if
11       end if
12    end for
13    return suggestList.sortBy(suitability)
14 end
```

Fig. 5. Pseudo code of the Semantic Service Suggestion Algorithm

the operation (line 04). If the availability is lower than 0.5, it is discarded for being considered unreliable.

If the service meets the availability requirement, the algorithm verifies the match between inputs required by the candidate operation and the outputs produced by the focus operation (line 05). The candidate operation will continue being considered only if at least one acceptable match is identified. In this case, it identifies the outputs available in the composition that match the inputs required by the candidate operation (line 07).

Finally, the algorithm calculates the composition suitability of this candidate operation (line 08). The lower the value, the more compatible the operation is. Equation 3 shows how the composition suitability of a candidate operation *O* is calculated:

$$CS_O = SM_O * \alpha + SR_O * \beta + (1 - av_O) * \gamma \quad (3)$$

where:

CS_O: Composition Suitability of operation *O*;

SM_O: Semantic Mismatch between the available inputs and operation *O*;

SR_O: Semantic Mismatch between the non-matching inputs of operation *O* and the available inputs;

av_O: availability of service that provides operation *O*;

α: weight associated with the matching inputs;

β: weight associated with non-matching inputs;

γ: weight associated with service availability.

After all candidate operations on the list are verified, the algorithm returns a list with suggested operations, ordered by their composition suitability (line 13).

4 The CVFlow tool

A composition tool has been integrated with the semantic service suggestion mechanism aiming to evaluate its appropriateness. This tool, called CVFlow, provides resources for defining workflows through the construction of a graph, in which a service corresponds to a node and the edges represent outputs produced by a service being used as inputs by another one. The creation of service compositions with this tool can be classified as dynamic and semi-automatic composition.

CVFlow is a workflow tool that was originally developed by the Lapix research group at the Federal University of Santa Catarina (UFSC). Although its first version focuses on image processing, it was designed to allow multiple uses. The Lapix group is currently developing extensions for hardware and logical control, for building 3D images and for video processing [11].

The nodes of a flow modeled with CVFlow consist of two types: Algorithms and Resources. Algorithms are programming functions defined by the developer that can be either local or remote. Resources are data items that serve as input parameter for Algorithms, e.g., images, numbers, strings, data structures and scripts.

Fig. 6 shows the graphical user interface of CVFlow. The central area of the interface is the composition area. The list of available Algorithms and Resources is displayed on the left side. The user can drag and drop any node to build the composition. The composition illustrated by Fig. 6 has three Resources and four Algorithms.

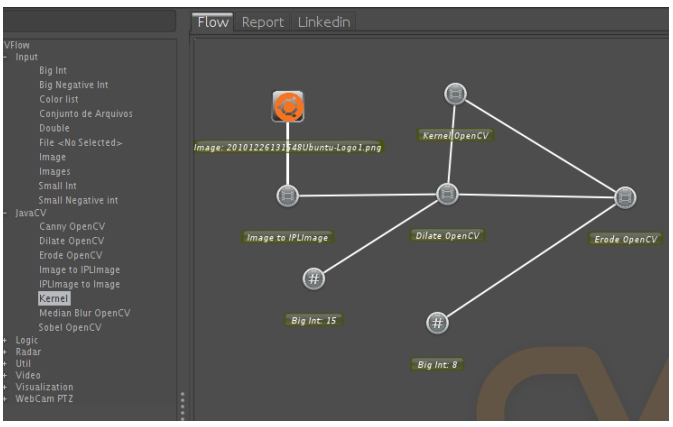


Fig. 6. User Interface of CVFlow

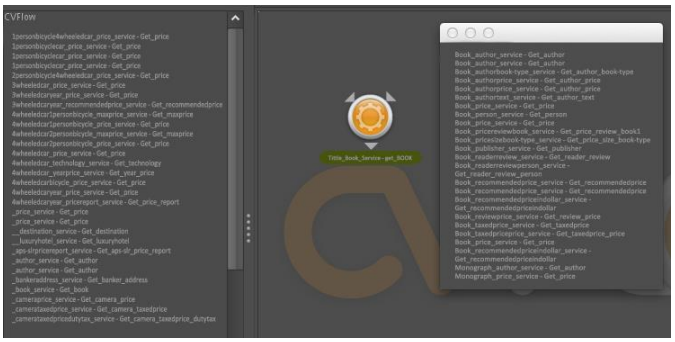


Fig. 7. CVFlow with Suggestion Mechanism

CVFlow has an internal architecture composed by three layers: the visualization, the workflow, and the core layer. The visualization layer is the GUI, shown by Fig. 6. The workflow layer represents every aspect of the composition, such as the inputs, outputs, nodes (i.e., Resources and Algorithms) and the connections between nodes. The Core layer is responsible for executing the Algorithms that are part of the composition.

In the original version of CVFlow, it is up to the user to verify if outputs from one service match with inputs from other ones. Execution errors occur when incompatible nodes are connected by the user.

The S^{3}M mechanism has been integrated with CVFlow in the aim of helping the user to easily select semantically compatible services (i.e., Algorithms in CVFlow terminology) and build a composed service. When the user selects a service in the composition, the mechanism orders the list of available services based on their suitability, and shows the ordered list on the right side of the CVFlow interface, as shown by Fig. 7. The user can then identify a suitable service and drag it to the composition area, connecting it to the selected service.

Based on the services suggested by the S^{3}M mechanism, users can more easily compose services. Besides, they avoid connecting incompatible services, which result in execution errors, and avoid using services with low availability.

5 Related work

The literature has several examples of software tools aimed at helping designers to build service compositions, including some that feature service suggestion mechanisms. These suggestions, though, are mainly based on syntactic information contained in service descriptions, which is often misleading. Just a few proposals found in the literature adopt semantic technology to identify compatible services.

Hobold and Siqueira [12] approach the composition of service using SAWSDL to discover and compose services automatically. To optimize the discovery process, during the publication of a service they extract from the WSDL the semantic annotations associated with operations, inputs and outputs, and store them in a relational database. This approach is also adopted in this paper. The main difference between these approaches is the way service compositions are created.

Another related research work was developed by Prazeres *et al.* [15]. These authors proposed an approach for automatic composition of semantic web services exploiting a graph-based model that takes into account the semantic similarity of services computed using subsumption reasoning on their inputs and outputs. This is an automatic approach for service composition that uses OWL-S for describing semantic web services. The way similarity between inputs and outputs is computed is similar to the strategy adopted by the S^{3}M mechanism.

Automatic approaches such as the ones proposed in [12] and [15] lead to combinatory explosion, resulting in poor performance. Besides, the designer is excluded from the composition process, becoming unable to contribute with the knowledge he often has on the problem domain.

Pi *et al.* [14] propose a tool for semi-automatic composition of semantic web services based on a GUI called Flow Editor. This tool was developed using Flex and is capable of converting WSDL files into OWL-S files. The tool is capable of exporting and executing the compositions. The main difference between the Flow Editor and our work is that the first uses OWL-S for expressing Semantic Web Services, while S^{3}M is based on SAWSDL annotations. Besides being a W3C standard, SAWSDL leverages from WSDL instead of requiring a parallel semantic description such as OWL-S does. Another important difference between the S^{3}M mechanism and the Flow Editor is that the first takes into account the availability history of services in its suggestion algorithm.

6 Conclusions and future work

This paper described a mechanism that provides suggestions for designers during the service composition process. The S^{3}M mechanism takes into account the semantic annotations contained in the WSDL files and verifies the semantic match between operations. The mechanism also evaluates service availability, in order to avoid the use of low quality services that may compromise the execution of a composition.

The proposed mechanism has been integrated with an existing workflow tool, called CVFlow, which allows the composition to be graphically built using the provided suggestions. Preliminary productivity tests have shown very promising results.

As future work, we plan to use the composition repository to include behavioral patterns of users in the suggestion mechanism. This can be done by modifying the suggestion algorithm to take into account the frequency in

which users select a given operation. Integration with a service orchestration mechanism described in [12] is also envisaged as future work. Furthermore, usability tests are being planned, aimed at evaluating how helpful service suggestion is for composition designers and how much time of the composition process is saved using the suggestion mechanism.

7 References

[1] T. Beneers-Lee et al. "The Semantic Web", Scientific American, vol., no., pp 28-37, May 2001.

[2] J. Farrel; H. Lausen. "Semantic Annotation for WSDL and XML Schema", 2007. Available at http://www.w3.org/TR/sawsdl

[3] R. Akkiraju et al. "Web Service Semantics – WSDL-S", 2006. Available at http://www.w3.org/ Submission/WSDL-S/

[4] J. Kopecky et al., "SAWSDL: Semantic Annotations for WSDL and XML Schema". IEEE Internet Computing, v. 11, n. 6, p. 60-67, 2007.

[5] A. Ankolekar. "OWL-S: Semantic Markup for Web Services", 2003. Available at http://www.daml.org/ services/owl-s/1.0/

[6] D. Roman et al., "Web Service Modeling Ontology". In Applied Ontology, Vol. 1(1), pp. 77–106, 2005.

[7] M. Schumacher; H. Schuldt; H. Helin, "CASCOM: Inteligent Service Coordination in the Semantic Web". Berlin: Springer, 2008.

[8] Kapitsaki, G.; Kateros, D.A.; Foukarakis, I.E.; Prezerakos, G.N.; Kaklamani, D.I.; Venieris, I.S., "Service Composition: State of the art and future challenges", Mobile and Wireless Communications Summit, 2007. 16th IST, 1-5 July 2007.

[9] S. Dustdar; W. Schreiner, "A Survey on Web Services Compositions". International Journal on Web and Grid Services, vol 1, pp1-30, 2005.

[10] D. Martin, M. Burstein, D. McDermott, S. McIlraith, M. Paolucci, K. Sycara, D. L. McGuinness, E. Sirin and N. Srinivasan. "Bringing Semantics to Web Services with OWL-S". World Wide Web 2007, pp. 243-277, 2007.

[11] M. H. Webber. "CVFlow", 2012. Available at http://www.lapix.ufsc.br/cvflow

[12] G. C. Hobold.; F. Siqueira, "A Platform for Discovery and Execution of Semantic Web Services Compositions", Proceedings of the International Conference on Semantic Web and Web Services (SWWS 2012), July 2012.

[13] M. Paolucci, T. Kawamura, T. Payne and K. Sycara, "Semantic matching of web services capabilities" in: Proceedings of 1st International Semantic Web Conference (ISWC), 2002.

[14] Bingfeng Pi; Gang Zou; Chaoliang Zhong; Jun Zhang; Hao Yu; Matsuo, A., "Flow Editor: Semantic Web Service Composition Tool," 2012 IEEE Ninth International Conference on Services Computing (SCC), 24-29 June 2012.

[15] C. V. S. Prazeres, C.A.C. Teixeira and M.G.C. Pimentel. Semantic Web Services discovery and composition: paths along workflows. In ECOWS'09: Proceedings of the 7th IEEE European Conference on Web Services, pp. 58-65, Eindhoven, Netherlands, 2009.

Concept Map Based Semantic Representation and Knowledge Visualization

Junfang Zeng
Institute of Automation, Chinese Academy of Sciences, Beijing, China

Abstract - *This paper presents a concept map (CM) based semantic representation and knowledge visualization model. The model is featured as: concept map based semantic representation, CM-web information mapping model, knowledge visualization and maintenance. The model is supposed be applied for constructing domain knowledge system visually, knowledge retrieval from web information, knowledge-driven information organization and navigation, and knowledge lifecycle management.*

Keywords: Concept Map, Semantic Representation, Knowledge Visualization, Mapping

1 Introduction

Nowadays, the fields of web information retrieval, knowledge management and digital content service are facing more challenges than ever. How to integrate resources in the future Internet environment, semantic representation, knowledge visualization and other fundamental advanced technologies find extraordinary opportunities for novel and future oriented applications.

Researchers have contributed concept maps, semantic networking [5, 6], knowledge networks [7, 9] etc. for representing semantic information and visualize knowledge construction. Firstly presented by Joseph D. Novak [1], concept maps are tools for organizing and representing knowledge visually [8]. They include concepts, usually enclosed in circles or boxes of some type, and relationships between concepts or propositions, indicated by a connecting line between two concepts. Words on the line specify the relationship between the two concepts. Concept Map takes its advantages comparing with other knowledge representations: 1) Provides a clear knowledge network in an intuitive and visualized way close to human thinking [2, 3]; 2) Easy to construct due to flexible definition of relationship between concepts to form a true network; 3) Applicable for computer processing and can be maintained.

With the increasing development of web technologies, huge amount of information can be accessed by people on Internet. Semantic information with high value are buried in natural language among web pages, such as insights, experiences, attitudes, values, expectations, perspectives, opinions, predictions etc. These semantic information or we call knowledge need to be represented, structured, visualized, stored, analysed and utilized. These are challenges not only for research but also for application. Knowledge can be explicit and implicit. Explicit knowledge may be understood easily, while large amount of implicit knowledge is difficult to discover, represent and process. We need a model to represent them, and make them visualized, easier to be understood, gathered and propagated.

We present a concept map based semantic representation and knowledge visualization model as a framework to realize the above purpose. This paper is organized into four sections: concept map based semantic representation, CM-web information mapping model, knowledge visualization and maintenance, and conclusions are given at last.

2 Concept map based semantic representation

2.1 Concept map

In natural language representation, we use sentences to express semantic information, the simplest one is: subject-predication-object, e.g. "*Dog is kind of animal*", where subject="dog", predication="is kind of", object= "animal". Concept map provides a clear semantic representation way close to natural language.

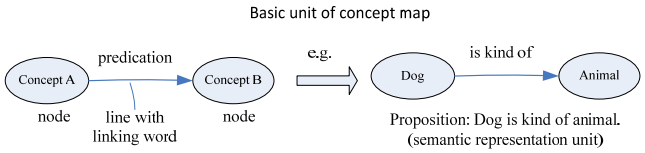


Fig. 1 A basic unit of concept map.

Concept map includes Concept (node), Relation (link) between nodes, two nodes and a line labelled with linking word form a proposition (Fig.1). Concept map focuses on concept, relation, link (hierarchy and crosslink), and instance (resource associated with concept). Key factors are shown in Fig. 2.

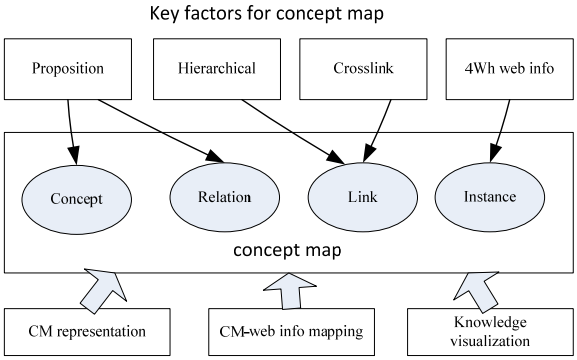


Fig. 2 Key factors for concept map.

(1) Concept

We use thriple to represent concept:

Concept=<ConceptName, Property, PropertyValue>

ConceptName: name of concept, defined by a noun or pronoun, e.g. concept "Student".

Property: main properties of concept, e.g. {name, sex, age, birthdate, grade}

PropertyValue: value of each properties, e.g. name.value=Linda, sex.value=female, age.value=17, birthdate.value =19970401, grade.value=G10.

(2) Relation

In CM, relation between two concepts is defined flexibly, any relationship is allowed so CM is a network with complex relationships. We define four typical relations distinguished by their different formats for representing information, shown in Fig.3.

A. Whole-part: concepts are organized by placing the whole in the center of the map, connecting parts surround the whole of the map. The relation is defined as "A is part of B" or "B is comprised of A".

E.g. Computer=(hardware, software), hardware=(CPU, memory, input device, output device), software=(system software, application software).

B. Parent-child: the parent is placed on the above, connecting children below the parent. The relation is defined as "A is father of B" or "B is child of A".

E.g. Food=(staple, meat, vegetable, fruit), staple={rice, dumpling, noodle, sandwich, pizza, ...}, meat={pork, beef, mutton, chicken, fish, duck, ...}, fruit={apple, pear, cherry, strawberry, …}, apple={red fuji, ralls apple, green apple, …}.

C. Ordinal: organizes concepts in a linear format, presenting information in a descending order of happening time or importance. The relation is defined as "A is followed by B" or "B begin with A".

E.g. Step=(step1, step2, step3, …)

D. Input-output: organizes concepts in a format with input and output. The relation is defined as "A has input B", "A has output B".

E.g. System=(input, parameter, process, output)

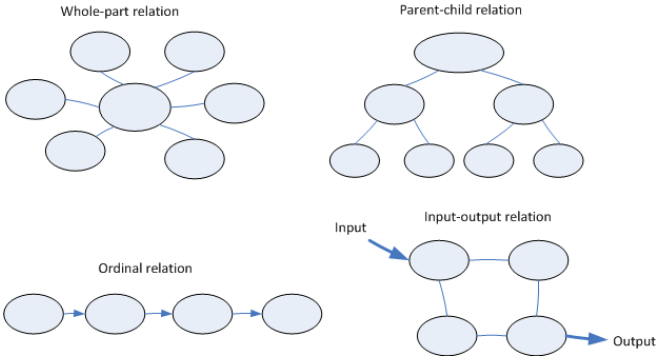


Fig. 3 Typical relations of concept map.

Besides the above four typical relations, predications are flexibly used to define the relations between concepts. Concept-predication-concept forms a proposition, which is close to natural language expression.

Predication={is, is part of, is comprised of, is made of, is kind of, is related to, is similar to, is defined as, is followed by, connect, has, include, involve, describe, specify, support, show, become, form, necessary for, may be, need to see, help to answer, begin with, add to, point out, combine with, transfer to, belong to, in, to, of, with, between, and, or, …}

2.2 CM Based semantic representation

Concept nodes are organized in a format according to the relations between nodes, resulting in a directed graph. We use concept map (CM) to represent domain knowledge, which is generally constructed by domain experts at the early stage.

To construct concept map based knowledge representation for computer processing, we need build corresponding data structure of database (logic structure and physical structure) and processing algorithms.

CM based semantic representation is formalized as:

Conceptmap=(Cnode, Clink)

Concept map is a set of concept nodes and a set of concept links (relations).

Cnode={cnodei, i=1…n}

Cnode represents a set of concept nodes, where *cnodei* represents a concept of domain knowledge.

cnode=(id, name, property, value)

id: id of node; name: name of node; property: concept property, value: property value

Clink={<cnodei, cnodej>, i,j=1…n, i<j}

Clink represents a set of links between nodes.

Concept maps are stored in relational database. Tables of Concept, Predication, Relation, Property, Instance etc. would be designed.

Concept map can be realized by adjacency list.

- General node

typedef struct concept-map{} concept-map;

- Head node

typedef struct head{} information;

3 CM-Web information mapping model

3.1 Hypertext represented web information

Web information on Internet generally adopts hypertext, content (topic oriented semantic information) lies in web pages (info nodes), and nodes are linked in super chain mode (hyperlink), to form a complex network structure.

Hypertext=(Hnode, Hlink)

Hnode={hnodei, i=1…n}

Hnode is a set of hypertext info nodes of domain information; *hnodei* represents an info node.

hnodei=(id, name, content-type, content)

id: id of node (URI); name: name of node; content-type: content format; content: content of web information.

content-type={text, graph, image, sound, video, complex-media}

content=<Who, What, When, Where, How>, formalized as "4Wh" information model in this paper, which are key factors for topic oriented event or activity.

"4Wh" information model

Info=<Who, What, When, Where, How>

Who: agent (person/object) of event

What: event related information

When: time of event happens

Where: place of event happens

How: how is event going on

Hlink={<nodei, nodej>, i,j=1…n}

Hlink represents a set of links between two info nodes.

3.2 Knowledge-information ~ CM-hypertext mapping

To organize web pages according to a topic oriented knowledge system (concept system), we construct concept map, link concept to related web resources (web pages). Semantic connection is established by the mapping between CM and hypertext (Fig.4), using a mapping table.

Mapping={<cnodei, hnodej>, i,j=1…n}

cnodei: a concept node of CM; *hnodej*: an info node of hypertext

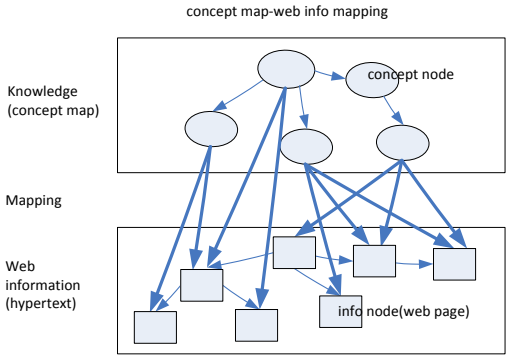


Fig. 4 Concept map-hypertext mapping.

Conversely, to retrieve knowledge from web information, we construct sub graph of concept map (CM segment) from hypertext, then define the mapping relationship between CM and hypertext. In this way, semantic information of web page can be structurally represented as concept map, and stored in relational database, and visualization is realized as well.

Also with the CM-web information mapping, we can realize knowledge-driven information organization and navigation. Concept map in hypertext format can be used for navigation, click the concepts on the concept map to browse linked resources. And through knowledge-driven maintenance, topic oriented web information could be changed dynamically with the development of domain knowledge.

4 Knowledge visualization and Knowledge maintenance

Knowledge visualization [4] is defined to represent knowledge in an intuitive graphic way. We use concept maps to realize knowledge visualization and maintenance.

4.1 Knowledge visualization

In the proposed CM based model, domain knowledge is visualized as concept maps. Concept maps can be generated by supporting software (ConceptDraw, Inspiration, Visio etc.), and stored in database automatically. Concept maps are drawn in software environment, concepts are listed, the user clicks a chosen concept, this concept is added to the map; a line is added by drag-drop to connect two concepts; predications are listed, choose a predication to mark the line; then proposition "concept-predication-concept" is generated which represents semantic information. Domain experts predefine the whole concepts and predications. Concepts can be used only once, while predications can be used as needed.

To construct concept map of domain knowledge, main steps include:

Step1. List concepts: determine top concept, key concepts and concept hierarchy.
Step2. Put the most important, abstract concept on the top; Put more concrete concepts below.
Step3. Connect concepts by linked line (relation). Write a predication to indicate the line. The line can be with one-direction, bi-direction, non-direction, form sub-graph of CM.
Step4. Find link between different sub-graphs of concept map, connect them and label with predication. Combine many inter-connected sub-graphs together, a whole map of domain knowledge is constructed.
Step5. Write instances or link web resources beside concept nodes to illustrate the concepts.
Step6. Continuously modify and improve the concept map to maintain the domain knowledge.

A sample of concept map oriented to a certain topic is given in Fig. 5.

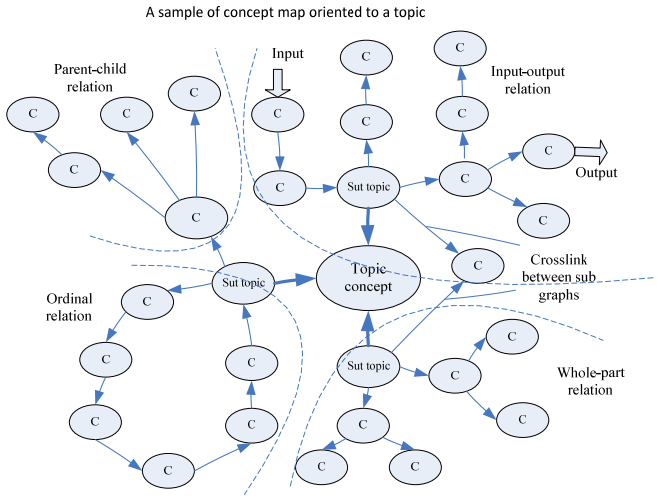


Fig. 5 A sample of concept map.

4.2 Knowledge maintenance

Concept map can be used for knowledge representation and knowledge visualization. However, concept maps are constructed by domain experts at the early stage to represent static knowledge system. To realize dynamic process of knowledge system lifecycle management, concept map maintenance is needed. Algorithms to maintain graphs in "data structure" are adopted to maintain concepts in "concept map", operations include: search concept, modify concept, modify relation, add new concept, delete concept, merge sub-graph, decompose graph.

(1) Search Concept

We adopt top-down search method.

-Search concept in CM

Function search-concept-map(conceptx); //search a concept in CM

-Search concept in CM-hypertext mapping

For an arbitrary node (conceptx) in CM, we define S as a set of all corresponding info nodes of conceptx.

Function search-mapping(sid); //search mapping table, to construct set S automatically

-Search concept in web page(hypertext)

Function search-hypertext (); //Search concept in hypertext, full-text information retrieval algorithm is adopted.

(2) Modify Concept

In knowledge-driven web information management, when a concept in domain knowledge CM changes, the concept included in its corresponding info nodes (web pages) should also be modified. To modify conceptx in all its info nodes, we first get a node1 in set S, modiy all conceptx of node1, then get the next node in S, modify conceptx, …until all nodes in S is processed.

Function modify(conceptx);

(3) Add Concept

Add a new node to CM, but where to put, should associate to existing concepts, analyzing the relationship between concepts. Judge the layer of the new concept, and put it to the corresponding position, connecting it with the related existing concepts.

Function add(conceptx, Clinkx); //add a concept in CM

Clinkx: a set of links connected conceptx to existing concepts

(4) Delete Concept

Delete an existing node from CM, and remove its links with related concepts. Links should be repaired and confirmed to avoid link breakage due to deleting a node.

Function delete-concept(conceptx); //delete a concept in CM

(5) Merge sub-graph

Two sub graphs are merged into one, which is an iterative process for concept map construction, from node to segment, from segment to sub graph, and merge sub graphs into the whole concept map.

Function merge-subgraph(subgraph1, subgraph2); //merge two sub graphs into one

(6) Segment graph

Segment one graph into two sub graphs, which can be used to manage knowledge segment and its linking web information.

Function segment-graph(graph); //segment a graph into two sub graphs

5 Conclusions

In order to make it easier and more efficient to represent and visualize knowledge, this paper puts forward a concept map based model for semantic representation and knowledge management. The model first constructs concept map of domain knowledge by domain experts; then maintains the concept map dynamically by using functions (APIs); further more, utilizes domain knowledge for web information organization and navigation. The model can be further applied in knowledge system construction, web information intelligent retrieval, teaching-learning system, requirement analysis (understand, interact and determine), digital library sorting system etc.

Future work includes: (1) Integration of multiple semantic representation approaches (eg. concept map, semantic web) to make semantic representation more accurate and easier for knowledge reorganization, exchanging, mining and utilization; (2) Knowledge visualization may transfer from 2D to 3D merging with Virtual Reality technology; (3) Knowledge visualization is used not only for static knowledge system, but also for dynamic procedure of knowledge lifecycle management, from its generation to development to losing effectiveness, an iteration and cooperation mechanism should be established.

Acknowledgement: This work is supported by National Hi-Tech Program of P.R. China (Y2W1011IC2).

6 References

[1] Novak, J.D., The Theory Underlying Concept Maps and How to Construct Them. Cornell University,1982.

[2] Eden, C., Cognitive mapping. European Journal of Operational Research, 36,1-13. 1988.

[3] Eden, C., On the nature of cognitive maps. Journal of Management Studies,29,261-265. 1992.

[4] Eppler, M.J. & Burkard, R.A., Knowledge Visualization: Towards a New Discipline and its Fields of Application. ICAWorking Paper #2/2004, University of Lugano, Lugano. 2004.

[5] Fisher, K. M. , Semantic Networking: The new kid on the block. Journal of Research in Science Teaching, 27(10), 1001-1018. 1990.

[6] Fisher, K. M., SemNet Software as an assessment tool. Assessing Science Understanding ,197-221. 2000.

[7] Knowledge Representation and Reasoning with Graphs., Knowledge Representation and Reasoning with Graphs, http://www.lirmm.fr/~cogito/. 2004.

[8] Newman, B. , Agents, Artifacts, and Transformations: The Foundations of Knowledge Flows. The Knowledge Management Handbook,1,301-316. 2003.

[9] Paivio, A. , Mental Representations. New York: Oxford University Press. 1986.

Semantic Hub Agent for Data Commercialization

Dongsheng Wang
Institute of Automation, Chinese Academy of Sciences
Beijing, China

Abstract – *There are few papers about data commercialization framework and we first try to propose a Semantic Hub Agent (SHA) centered data commercialization framework to inspire real-time data interaction among general agents. SHA is supposed to be built as a professional third-party agent that is delegated by them to transfer, maintain, and sell their dataflow. Data format is multiple and thus it is a good opportunity for ontology, a formalized format, to play this role in a globalized way. SHA works on data transferring, data grading, and releasing with powerful reliable server. The prototype experimental system is implemented based on the framework, demonstrating the feasibility of data grading and reorganizing in a clean and effective way. This framework is very meaningful not only for Semantic Web field but also for the "big data" communities who are always facing the problem that their frequently mined data is outdated soon.*

Keywords: Semantic Web, Data Commercialization, Ontology, Semantic Hub Agent, SHA

1. Introduction

The concept of Semantic web [1] is raised and developed for decades with a variety of separate semantic technologies generated from the developing progress, such as the ontologies (OWL and RDF) that are adopted by many applications. Another significant progress is that many research groups and non-profit organizations released large-scale knowledge bases such as YAGO, DBPedia [7], WordNet, and other SKOS [12] standard based knowledge systems.

The essential value of Semantic Web is the perspective of distribution and globalization instead of local application. However, the ontology in the world is still very barren since the original purpose of semantic web is to lead to a web revolution (Web 3.0). Currently, the further development of ontology propagation appears to meet its bottlenecks. The vital reason is that we do not motivate the most general agents such as companies, organizations and governments. This is very challenging because even if some general agents are motivated to release their data as linked data, they generally do not have the ability or their servers cannot accommodate a large amount of users in an open environment. Thus, the service of data from them is always not stable nor promised and we often need to dump big datasets frequently and set them into our local running environment. The primary two problems we focus on can be summarized as the followings:

Problem1: *The quantity of semantic data (ontology) is still very barren since most general hub agents such as companies, organizations and governments are not well motivated to release their data.*

Problem2: *Even some of them agree to, they do not have the ability or cannot provide a stable, dynamic, and reliable service.*

In this paper, we solve the problems by proposing a SHA centered framework for data commercialization which can encourage general agents to provide their data in a profit-motivated way and SHA professionals to supply a reliable service. To the best of our knowledge, there are few papers about data commercialization or related frameworks. The proposed framework is designed to be established as a third-party platform which is delegated by various agents to transfer, maintain, and sell data as like fresh water flow.

A comparison between relational database model and ontology model indicates that both of them have their own characteristics and neither of them can totally substitute the other [13]. Thus coexistence and mutual transferring when necessary is a tradeoff for now. Since it is notable most databases used by general agents are relational databases such as MySQL and Oracle, dynamic data transferring can be conducted with D2R tool. Also, data grading are important to protect classified data, and this is implemented with SPARQL CONSTRUCT pattern, which can additionally reorganize the data. Data servicing is provided through endpoint server like Fuseki based on TDB or more powerful RDF server based on Hadoop, resulting in an open and stable dataflow as a profitable product.

The prototype system is implemented based on the framework, demonstrating the feasibility of data grading and reorganizing in a clean and effective way. The proposed framework not only is significant for Semantic Web, we can also expect the methodology revolution of "big data" that the analyzers no longer need to crawl data, costing much time and finance, neither do they need to worry about the data will be outdated soon and crawl them again. Moreover, the well-structured and refined data is provided right from the sources.

2 Motivation and Related Works

General hubs particularly indicate large agents such as companies, schools, institutes and organizations instead of individual persons. From some non-formal surveys and investigation, three major factors that are paid most attentions on are summarized in the followings.

- We should not (or decrease as much as possible) request the general agents for additional works or investments.
- The data of general hub agents is selectively opening in a grading way, so that the private and classified data can be protected well.
- Their opening of graded data will bring them additional benefits as by-product profits.

LOD (Linked Open Data) [6] is well-known as the best practice to expose, share, and connect information or knowledge on Semantic Web. Though it has resulted in a great number of agents joining in LOD, the non-profit initiative limits its propagation in a wider range.

In part, Government is relatively easy to be motivated to release their data in an open way. For example, the work of [3] is an effort to make government data freely available to everyone. They publish the linked data on top of the relational databases. LOGD (Linked Open Government Data) is developed to be more scalable and interoperable in [5].

There are few papers directly related with data commercialization or related frameworks. The proposed framework is a lightweight one that is easy to control and manipulate. The framework is based on some existing technologies and tools.

First of all, there are some sophisticated tools available for us to transfer relational databases into ontologies. D2R Server [2] is a significant one, which is used to map non-RDF databases to RDF databases. It generates a mapping file that specifies how relational database schemas are identified as RDF resources. However, the mapped ontologies require further pruning, and as discussed in the principles there is not a good strategy to motivate general agents to use it, which may even bring additional burden to their own relational database server. Recently, there are more and more people use SPARQL CONSTRUCT patterns to do reasoning and build new data graph such as [4]. The pattern not only can build sub-graphs with reasoning function but also can work as a method to grade, and protect classified data.

SPARQL Endpoint Server can provide SPARQL update, query, and using the SPARQL protocol over HTTP. It is very convenient for users especially programmers to get real-time structured data. Some researchers try to federate SPARQL endpoint server via various methods in [9]. Jena framework also provides us an endpoint server tool generally based on TDB called "Fuseki" [10]. It is promising to deploy a Hadoop based cloud storage to accommodate large-scale RDF data such as the work in [9]. For the conceptual ontology (OWL/RDFS), we usually visualize the data and manually edit concept with Protégé tools [11].

3 SHA Centered Framework

Definition of Data Commercialization: Data Commercialization refers to a commercial mechanism for data interaction between any agents in an open and globalized way.

Definition of Semantic Hub Agent (SHA): SHA indicates an online platform providing stable and clean semantic data service based on large-scale RDF ontology maintained by ontological professionals, which is dynamically updated from a variety of non-RDF data sources.

Since ontology professionals are experts in ontology generating and maintaining, they establish and maintain large SHA center which acts as a huge cache to dynamically image general agents' data in ontology.

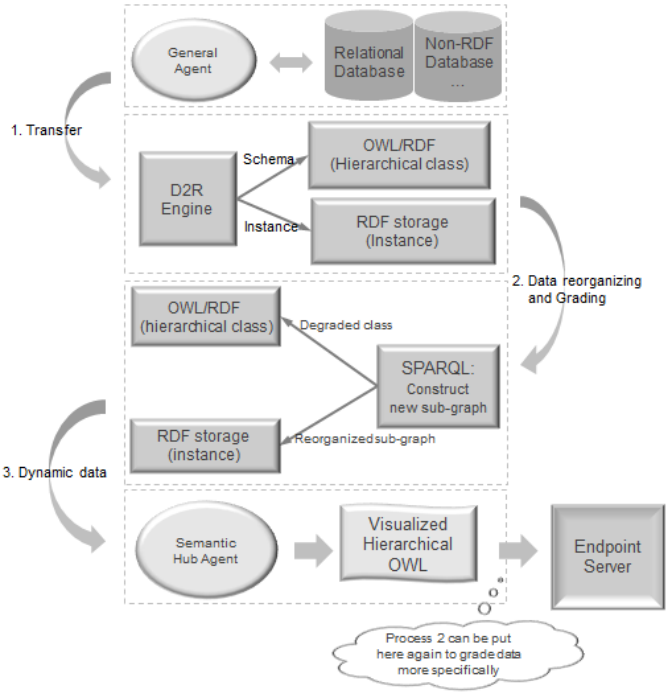


Figure 1. Single pipeline connecting from General Agent to Semantic Hub Agent

We will discuss the workflows that build connections between a general agent and a SHA as shown in Figure 1. The primary components in the framework are described in the following sub-sections.

3.1 Dynamic transferring

D2R tool is a fundamental component for a SHA. D2R tool clarifies the separation between conceptual schema and

instance data, and instance data is typed with conceptual schema. Ontological professionals generate the mapping file for D2R, and refine the mapping file into a clean conceptual ontology (RDF/OWL). It presents a complete and integrate RDF graph of database but is not really stored physically. Also, the single connecting from general agent to SHA that dynamically transfer and update data will not bring too much load for general agent. It is a virtual mapping between relational database and RDF graph when submitted with a SPARL query.

3.2 Data grading and reorganization

Data grading is vital for most of agents to protect their private data. Ontology professionals can grade the data with the advantage of SPARQL CONSTRUCT pattern. Recently, more and more researchers recognize the convenience of applying SPARQL CONSTRUCT to filter data, construct and do reasoning in a flexible and quick way. Also, it can artfully be used to reorganize data in a format we want and to grade RDF data into sub-graphs to protect the classified data. It is very convenient since SPARQL CONSTRUCT returns a customized integrate RDF graph.

The data transferred from D2R in previous step reflects the whole scene of the databases. Therefore, data grading is needed to hide some data such as "*user password*" and store the filtered data in an isolated storage. At the same time, we can reorganize the data property into a familiar one with a unified habit. For example, given that some databases use "*:stu_name*" to identify a student name, and some use "*:student_name*", we can unify them into just familiar one "*foaf:name*".

3.3 Endpoint server and interface

As discussed before, we transfer relational databases of general agent into ontologies with D2R tools. Thus every table is transferred as one Class with the schemas as their properties, and one whole database generates one ontology file with a group of classes from conceptual level. As a consequence, if we want to search instance data, we can refer to the conceptual ontology and input our query for what we want. We can visualize the conceptual ontology relatively easily as a hierarchical concept map.

Ontology professionals in SHA commercialize data as a "waterworks" to provide stable, graded and dynamic dataflow, with a conceptual ontology (refined mapping file) to show what source data the SHA has. It is important to use separate big RDF storage to store the graded and reorganized data because most of the visiting burden is located on this large storage. The stability of the endpoint server should be promised so that data consumers can consider them as like their own local databases without the need of dumping all dataset.

In this way, various source data are merged together in one single platform, and thus users who need data can access the dynamic data conveniently from one interface. The profits can be distributed to different general agents according to the usage of flow meter of data via automatic computing. The SHA can get their server maintaining revenue.

4 Depolyment and Case Study

The overall deployment of SHA framework is depicted in Figure 2. SHA acts as a manager collecting various source data from diverse databases and providing a unified service with unified data format – RDF/OWL.

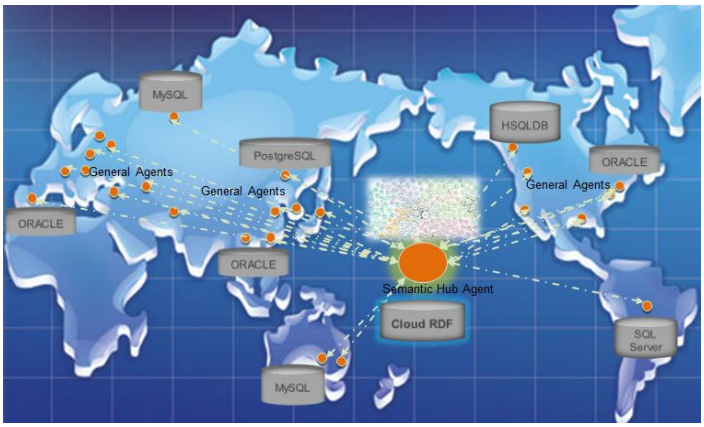


Figure 2. SHA dynamically update data from diverse sources and supply unified service

We give a simple case study. There is a relational database singly include three tables, and their relationship is demonstrated in Figure 3. It is about a house renting website where users can leave messages requesting their desired house.

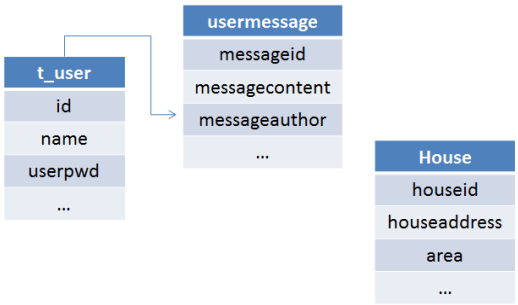


Figure 3. A simple relatioinal database with three tables

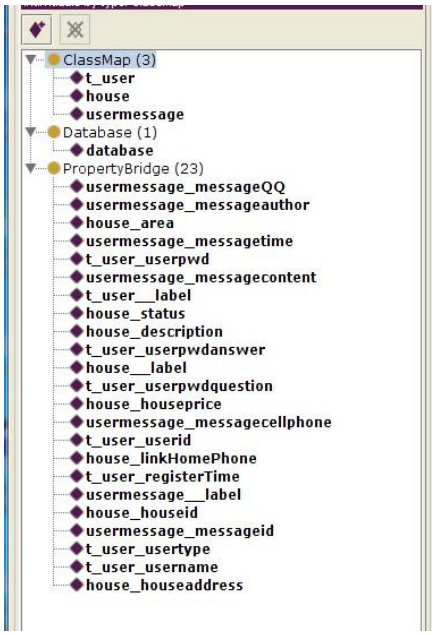


Figure 4. Mapping file shown in protégé

Then, we first utilize D2R tool to automatically generate a mapping file, which we open in a protégé. As shown in Figure 4, there are three upper layer classes including "ClassMap", "Database" and "PropertyBridge". "ClassMap" expresses a class in the OWL ontology or RDF Schema. There are three individuals of ClassMap corresponding to three tables in Figure 3, including "t_user", "house" and "usermessage". "Database" indicates the database and commonly includes single one individual. The "PropertyBridge" is another important concept identifying the properties of class in OWL or RDF Schema.

According to this whole scene ontology, we employ SPARQL CONSTRUCT pattern to construct a sub-graph from database as a degraded and reorganized one.

```
PREFIX vocab: <http://localhost:2020/resource/vocab/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

CONSTRUCT {
 ?user foaf:name ?name.
 ?message db:by ?user.
 ?messsage ?mss_pro ?mss_obj.
 ?message db:content ?content.
}
WHERE {
 ?user db:vocab/t_user_username ?name.
 ?user db:vocab/t_user_registerTime ?registerTime.
 ?user db:vocab/usermessage_messageauthor ?user.
 ?user db:vocab/usermessage_messagecontent ?content.
}
```

Figure 5. SPARQL CONSTRUCT example.

The example shown in Figure 5 demonstrates a SPARQL CONSTRUCT pattern which uses well-known "foaf:name" to replace "db:vocab/t_user_username" that is automatically generated. Also, it constructs a graph what the rules specify, excluding information such as "password" and returns a integrate sub-graph. The output is demonstrated in Figure 6.

```
SPARQL results:

<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:vocab="http://localhost:2020/resource/vocab/"
    xmlns:foaf="http://xmlns.com/foaf/0.1/"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:db="http://localhost:2020/resource/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:map="http://localhost:2020/resource/#">
  <rdf:Description rdf:about="http://localhost:2020/resource/usermessage/003">
    <db:content>It is important to have the house shined</db:content>
    <db:by>
      <rdf:Description rdf:about="http://localhost:2020/resource/t_user/003">
        <foaf:name>angSun</foaf:name>
      </rdf:Description>
    </db:by>
  </rdf:Description>
  <rdf:Description rdf:about="http://localhost:2020/resource/usermessage/001">
    <db:content>I want a house with 70mm in Haidian strict</db:content>
    <db:by>
      <rdf:Description rdf:about="http://localhost:2020/resource/t_user/002">
        <foaf:name>HuafangZhang</foaf:name>
      </rdf:Description>
    </db:by>
  </rdf:Description>
  <rdf:Description rdf:about="http://localhost:2020/resource/usermessage/002">
    <db:content>I can not find a house in Guomao with a price less than 5/d/mm</db:content>
    <db:by>
      <rdf:Description rdf:about="http://localhost:2020/resource/t_user/007">
        <foaf:name>XuZhang</foaf:name>
      </rdf:Description>
    </db:by>
  </rdf:Description>
</rdf:RDF>
```

Figure 6. Graded graph

As we can see, it includes some open-possible information from table "t_user" and "usermessage" in Figure 3. The graded and reorganized data can be updated into storage such as Fuseki or cloud RDF storage Server, and we can update it according to time property at regular intervals.

Finally, we demonstrate various refined conceptual ontologies for users, with introduction about their data sources. Then, users can write their query according to them and visit data through endpoint server or get data through a URL in program automatically.

5 Conclusions

We aim to motivate the widest range of agents to join in as a provider source of semantic data such as companies, schools, governments and organizations which are extensively distributed. We discussed some principles they paid most attention on and proposed the concept of SHA. There are few papers about data commercialization framework and we first try to propose the SHA centered data commercialization framework to motivate the general agents.

The D2R is a component to transfer relational database into RDF data, based on which the data degrading and reorganization via SPARQL CONSTRUCT pattern, and storing in stable server like Jena Fuseki to provide a stable endpoint service is designed. For SHA operators, they can obtain profits by connecting more general agents and maintaining SHA, promoting the data interaction between any two agents worldwide in an effective and fast way. The case study implemented based on the framework demonstrates the

feasibility of data grading and reorganizing in a clean and effective way.

This framework is very meaningful not only for Semantic Web field but also for the "big data" communities who are always facing the problem that their frequently mined data is outdated soon. They can use dynamic data from multiple sources in a convenient and cheap way by paying data flow fee to one platform. For Semantic Web, the fundamental ontology environment for it will be widely proliferated, and the further development will be captured more naturally such as entity linking among multiple sources and intelligent information system application.

For the future work, it is supposed to build endpoint server as a distributed cloud environment such as the usage of Hadoop [8]. Also, it is not limited to build only one SHA; instead, we should build connected SHAs according to regions, countries or domains. We encourage and expect the appearance of SHAs established by other researchers or organizations in the future. The SHA is expected to be a breakthrough opportunity for Semantic Web with data commercialization mechanism leading to a real web revolution (Web 3.0).

Acknowledgement: This work is supported by National Hi-Tech Program of P.R. China (Y2W1011IC2)

6 References

[1] Berners-Lee, Tim, James Hendler, and Ora Lassila. "The semantic web."Scientific American 284.5, 28-37, 2001.

[2] Bizer, Christian, and Richard Cyganiak. "D2R server–publishing relational databases on the web as SPARQL Endpoints." Proceedings of the 15th International World Wide Web Conference, 2006.

[3] Kalampokis, Evangelos, Efthimios Tambouris, and Konstantinos Tarabanis. "On publishing linked open government data." Proceedings of the 17th Panhellenic Conference on Informatics, 2013.

[4] Meditskos, Georgios, et al. "SP-ACT: A hybrid framework for complex activity recognition combining OWL and SPARQL rules." Pervasive Computing and Communications Workshops (PERCOM Workshops), IEEE, 2013.

[5] Dominic DiFranzo, Li Ding, John S. Erickson, Xian Li, Tim Lebo, James Michaelis, Alvaro Graves, Gregory Todd Williams, Jin Guang Zheng, Johanna Flores,Zhenning Shangguan, Gino Gervasio, Deborah L. McGuinness and Jim Hendler. TWC LOGD: A Portal for Linking Open Government Data. In:Semantic Web Challenge, International Semantic Web Conference, 2010.

[6] Linked Open Data (LOD), http://linkeddata.org/

[7] Morsey, Mohamed, et al. "Dbpedia and the live extraction of structured data from wikipedia." Program: Electronic Library and Information Systems 46.2, 157-181, 2012.

[8] Papailiou, Nikolaos, et al. "H2RDF+: An Efficient Data Management System for Big RDF Graphs.", 2014.

[9] Görlitz, Olaf, and Steffen Staab. "SPLENDID: SPARQL Endpoint Federation Exploiting VOID Descriptions." COLD 782, 2011.

[10] Fuseki, http://jena.apache.org/documentation/serving_data/

[11] Protégé, http://protege.stanford.edu/

[12] Miles, Alistair, et al. "SKOS Core: Simple Knowledge Organisation for the Web.", International Conference on Dublin Core and Metadata Applications, 2005.

[13] Martinez-Cruz C, Blanco I J, Vila M A. "Ontologies versus Relational Databases: are they so different? A comparison" , Artificial Intelligence Review, 38.4, 271-290, 2012.

Introduction to Ubit Semantic Computing

Shengyuan Wu
Independent researcher, Ubit inventor, retired professor, Shandong University, Jinan, China

Abstract - *Compute science faces two basic and hardly solved problem; the first is semantic ambiguity; the second is the semantics only understandable to human, not understandable to machine. The difficulties stem from unstructured bits. Based on a new theory, called as Ubit theory, this paper introduces a new generation of semantic computing, called as Ubit Semantic Computing (USC), which can solve the problems successfully. This paper first introduces Ubit theory briefly, then explains how to eliminate semantic ambiguities of natural language, video, audio, and image; and how to make them understandable to both machine and human; then introduces how to make semantics of program and web understandable to both machine and human; making semantic translating tools no needed any more, such as compiler, interpreter, semantic analysis, web parser, domain name resolution; then introduces an integrated interface of USC, by which, anyone can access anything, from anywhere, and in anytime; at last presents an architecture of USC.*

Keywords : Ubit; Ucode; semantic; understandable; content; intent

1. INTRODUCTION

J. Glenn Brookshear says: "Computer scientists dream that the source program is not forced translated to machine language; and machine could perform algorithm discovery process rather than just obeyed execution. [1]"

Tim Berners-Lee points out: "The Web was designed as an information space; with the goal that it should be useful not only for human-human communication, but also that machines would be able to participate and help."

"One of the major obstacles to this has been the fact that most information on the Web is designed for human consumption," [2]

Therefore, semantic web had to be first expressed in a "machine processable form" ; then transform the machine processable form into machine executable form [2]; which is still not machine understandable form.

The aim of semantic computing is user's intents can match author's intents precisely and efficiently.

The semantic computing definition made by IEEE says "Semantic computing concerns with connecting the (often vaguely formulated) intentions of humans with computational content [3]."

Why the dream of Computer scientists can't become true? Why is semantics represented in ambiguous form? Why the semantics is not represented by machine and human understandable form?

This stems from bit; stems from codes.

Basic on Ubit theory [4], this paper divides bit into Vbit and Ubit; divides code into Vcode and Ucode; and divides semantic computing into Vbit Semantic Computing (VSC) and Ubit Semantic Computing (USC); the basic idea of USC is that the semantics of contents is expressed in the format understandable to both humans and machines.

All existed semantic computing belongs to VSC.

Authors create contents according to author's intents, users retrieve contents according to user's intents; intents need containers; contents are intents inside containers; the containers are codes. Contents are the set of all codes.

The intent container of VSC is Vcode; the intent container of USC is Ucode.

Vcode is unstructured; the number of compatible and consistent Vcodes is limited. VSC can't represent unlimited intents by limited Vcode, Vcode and intent is not mapped one to one; therefore, ambiguities are inevitable.

The length of compatible and consistent Vcode is limited; therefore, no enough room contains machine understandable semantics.

In contrast, Ucode is structured; the number of compatible and consistent Ucodes is unlimited. Unlimited Ucodes can represent unlimited intents.

Ucode and intent is mapped one to one; therefore, ambiguities are eliminated.

The length of compatible and consistent Ucodes is unlimited, and Ucode is structured. Ucode can contain any machine understandable semantics; and also can associate semantics with machine and human. That is Ucode builds a bridge between machine understandable semantics and human understandable semantics.

This paper introduces Ubit theory briefly, then explains how to eliminate semantic ambiguities of natural language, video, audio, and image; and how to make them understandable to both machine and human by experiment examples; then introduces how to make semantics of program and web understandable to both machine and human; then introduces an integrated interface of USC, by the interface, anyone can access anything, from anywhere, and in anytime; at last presents an architecture of USC.

2. A BRIF INTRODUCTION TO UBIT THEORY

2.1 Vbit, Vcode

Bit is the basic unit of information capacity; a bit can have the value of either 1 or 0 only. Since digital computer was born, bit has only acted as value; this kind bit is called as Vbit. Code only consisted of Vbits is called as Vcode. All the existed codes are Vcodes.

Because Vcodes are not structured, Vcodes in a sequence usually distinguished by code length, different length of Vcodes can hardly be distinguished in one Vcode sequence; generally, only Vcodes in one code set can be mixed in a Vcode sequence.

2.2 Ubit, Ucode

Ubit is not acted as value; Ubit is used to represent data structure, making data structured and distinguishable from the most bottoms. [4]

Ubit can be used in dividing a bit sequence into bit groups.

Fig. 1 (a) contains a bit sequence, which is divided into 5 bit groups: 10, 110, 0, 0, 11110.

The bit group above is characterized as:

If a bit group is one bit long, the bit must be 0, for example the third group "0" and fourth group "0".

If a bit group is more than one bit long, the right most bit must be 0, the others must be 1; for example: 10, 110, 1110.

The left bit neighbor of a group can't be 1, but the right bit neighbor of a group can be 1 or 0,

There is one and only one bit 0 in one bit group.

A bit sequence can be distinguished into groups by one Ubit rule; there are ten Ubit rules; the most useful is Right 0 Ubit rule.

2.3 Right 0 Ubit rule

Right 0 Ubit rule: Scan a binary 1 and 0 sequence from left to right, if the first bit is 1, continue scanning until a bit 0 is met, then from the first bit 1 to the bit 0 is distinguished as a group; if the first bit is 0, then the bit 0 is distinguished as a group; continue scanning and distinguishing until the end of the sequence, if the last bit of the sequence is not 0, then the last group is an incomplete group.

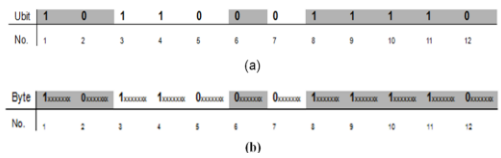


Figure 1. *Grouping by Right 0 Ubit rule (a) distinguishing bit sequence into bit groups (b) distinguishing byte sequence into byte groups*

Example 1, distinguish the bit sequence in Fig. 1 (a) into bit groups.

Scan the binary 1 and 0 sequence in Fig 1 (a) from left to right,

The 1st bit is 1, continue scanning, the 2nd bit is 0, therefore, "10" is distinguished as a group;

Continue scanning, the 3rd bit is 1, continue scanning, the 4th bit is 1, continue scanning, the 5th bit is 0, therefore, "110" is distinguished as a group;

Continue scanning, the 6th bit is 0; then, the bit 0 is distinguished as a group.

Continue scanning until the end of the bit sequence; 5 groups are divided by Right 0 Ubit rule.

Here, bit 0 or 1 is used in distinguishing bits into groups, not acted as value. This kind of bit is called as Ubit.

A complete bit group in Figure 1(a), consisting of only Ubits; it is an Ubit group. An Ubit group is called as Uframe.

In Fig. 1(b), each Ubit in (a) is copied to the leftmost bit of each byte respectively; then, the byte sequence is divided into 5 byte groups by the Right 0 Ubit rule, or by Uframe. The distinguishing procedure is the same as example 1; the difference is each Ubit in the leftmost bit of each byte; therefore, the distinguishing is by scanning each Ubit in the leftmost bit of each byte.

Each of the 5 byte groups consists of Ubit and Vbit, each byte group is called as Ucode; the leftmost bit of each byte is Ubit. The length of Ucode in the Ucode sequence is different.

Example 2， distinguish byte sequence into Ucode,

Fig. 2 (a) is a byte sequence, each byte is represented in two hex numbers, the leftmost bit of each byte is Ubit; if the first hex number of a byte is greater than 7, then the Ubit of the byte is 1, else the Ubit of the byte is 0; (here, the old display system takes Ubit as value) for example, the first underline three bytes are displayed as {F2 A0 2D}, the Ubits of three bytes are {1,1,0}, respectively.

The Ubits of the 16 bytes of the first row are 1101101101101110

Then, the 16 byte sequence is distinguished into 4 Ucode by Right 0 Ubit rule.

The underline Ucodes in Fig. 2 (a) are 3, 3, 3, 4, 2, 6 bytes and 1 byte long, respectively. The Ubits of the underline Ucodes are {1,1,0},{1,1,0},{1,1,0}, {1,1,1,0}, {1,0}, {1,1,1, 1,1, 0}, { 0 }, respectively.

It is clear that variable length Ucodes are mixed in a code sequence, compatible with one another.

An Ucode consists of sections, here, the section is byte; the section can be any long, but the shortest section is 2 bits.

Here, Ubit is assigned in the leftmost bit; it can be positioned in any bit of the section as needed.

A section sequence can be distinguished into Ucodes by Ubit rule.

Thus, bit is divided into Vbit and Ubit.

Vbit is used to represent data value; Ubit is used to represent data structure, making data structured and distinguishable from the most bottoms.

Code is divided into Vcode and Ucode.

Vcode only consists of Vbit, unstructured and undistinguishable. All existing codes belong to Vcodes.

Ucode consists of Vbit and Ubit; the Ubits in an Ucode represent the structure of the Ucode, making it structured and distinguishable; the Vbits in an Ucode represent the value of the

Ucode. Variable length Ucodes are compatible with one another; unlimited amount codes are available; the length of Ucode can be very short, or very long, it can contain any kind of semantics, and any amount of semantics; it can also associate with semantics; suitable as contents containers; Ucode acts as an bridge between human and machine understandable semantics.

Ubit theory is a fundamental theory of computer science, it studies how represent data structured from the bit level, how make semantics of contents expressed in the format understandable to both human and machine; in another hand, Ubit theory also studies how to make semantics of private contents not understandable to unauthentic persons and machines.

3 HUMAN AND MACHINE UNDERSTANDABLE SEMANTICS

3.1 Outline of human and machine understandable semantics

Fig. 2 (a) is an Ucode sequence understandable to machine, containing 1, 2, 3, 4, or 6 byte length of Ucodes.

Fig. 2 (b) is a notation group sequence, the human understandable semantics. Each notation group is an object, each object is associated with one and only one Ucode in Fig. 2 (a), The underlined notation groups are audio object, video object, image object, two-Chinese character word object, one-Chinese character word object, three-Chinese character word object, and one English letter object respectively.

3.2 Human and machine understandable semantics of audio object, video object and image object

3.2.1 Human understandable semantics of audio, video or image object associated by Ucode

In Fig. 2 (b), the symbol pair of "♪ ♪", "♗ ♗" and "▣ ▣" represents audio, video and image object respectively, the character string inside each symbol pair of "♪ ♪", "♗ ♗" and "▣ ▣" is the name of the object. These notations make the objects understandable to human. For example, the first object is an audio object: ♪致爱丽丝♪, the second is video object: ♗黑虎 1♗, the third object is video object: ♗猫和老鼠 10♗(video clip of cat 10) , and the fourth is an image object: ▣红棕马▣. As shown in Table 1; the notation of each video clip of cat and mouse is associated by its related Ucode.

3.2.2 Machine understandable what kind of object semantics embedded in Ucode

In Fig. 2 (a), each audio, video or image object is coded in a 3 byte Ucode; the first bit of each byte is Ubit, the others are Vbit. Machine can distinguish each Ucode by Ubit rules [4]. The type semantics of the audio, video and image object is contained in the first byte of the related Ucode, that is the Ucode is divided into two parts, the first byte is for object type; the Ucode is structured. For example, the first underlined Ucode {F2 A0 2D} is an audio object, the second underlined Ucode {F3 A0 6A} is a video object; the third underlined Ucode {F1 A2 7C} is an image object; the first byte of the Ucode of audio, video and image object is F1, F2, F3, respectively. Because the leftmost bit is an Ubit, the real value of the first byte is 0x71, 0x72 and 0x73; that means the real type attribution of audio, video and image is 0x71, 0x72 and 0x73.

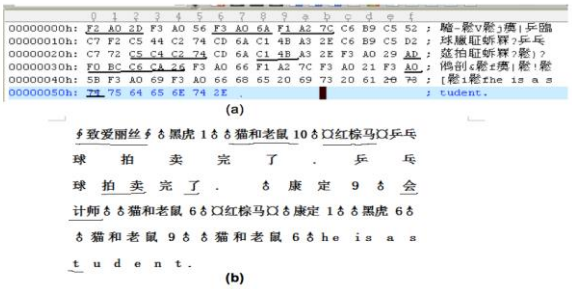


Figure 2. *Human and machine understandable semantic with mixed objects: audio object, video object, image object, one-Chinese character word object, two-Chinese character word object, three-Chinese character word object, and English letter object. (a) Ucode sequence; (b) Notation group sequence*

The maximum value of the first byte of the Ucode is 127; that means it can hold 127 types of objects for this three byte Ucode.

The ASCHII codes in Fig. 2 are no change; they can be distinguished by Right 0 Ubit rule; therefore, ASCHII code is Ucode. It is easy to make Vcode be Ucode by adjusting a bit here; for example, the Chinese GB2312 in Fig. 2; only the leftmost bit of the last byte of each word is changed from 1 to 0.

It is easy to judge what is Ucode and Vcode, if the codes in a sequence are distinguished by Ubit rule; then, they are Ucodes.

Ucode is hierarchically structured; however, Vcode is not structured. For example:

GB2312 is 2 byte code set in Chinese; JISX 0208 is 2 byte code set in Japan; and KSC 5601 is 2 byte code set in Korea. These are local code sets, unstructured, conflict one another; therefore, they can't be distinguished in a mixed code string.

Unicode is universal or global code set, the relation between global code set and local code sets should be in hierarchical form; however, it's not. Unicode is only a recoded code set; UTF16 contains 65,536 codes, including most of the characters needed for writing system in the world. But, different kinds of codes are mixed in Unicode, not structured.

The codes of GB2312, JISX 0208, KSC 5601 and Unicode can't be put into one singular string, because no way to distinguish what kind of codes.

However, GB2312, JISX 0208, KSC 5601 or Unicode can be easily represented by Ucode, for example, here double byte Ucode is used for GB2312 as default; JISX 0208 and KSC 5601 can be represented by two type of object of the Ucode in Fig. 2(a); Unicode can be divided into 4 classes, each can be represented by one type of object of the Ucode. Then all code sets are hierarchically structured, each set can be distinguished by its type attribution [5]. Local code sets, various code sets in various locations are compatible one another, this is called as space consistent.

Here, the object type attribution is embedded in Ucode, not represented in notation form, understandable to machine.

3.2.3 Machine understandable object addressing semantics associated by Ucode

In order to access an object, machine must need to know the address in storage space. There are two ways: one is embedded the physical address in Ucode, another is associating the address by Ucode.

Usually, an object address is represented by directory path as shown in table 1; this is a notation form, only understandable to human. Therefore, the notation form address must be translated to machine.

Table 1 (a) is Ucode sequence for clips of video "cat and mouse"; (b) Ucode associated with human's language for clips of video cat and mouse: the notation form; (c) Ucode associated with physical address for clips of video "cat and mouse".

TABLE I. THE ASSOCIATION BETWEEN UCODE AND SEMANTICS

Ucode	F3A0601	…	F3A069	F3A06A

(a) Ucode sequence for clips of video cat and mouse

Notation	♁猫和老鼠 1♁	…	♁猫和老鼠 9♁	♁猫和老鼠 10♁

(b)Ucode associated with human's language for clips

File l address		…		

(c) Ucode associated physical address for clips of video cat and mouse

3.3 Human and machine understandable semantics of natural language

The most difficult language might be Chinese; so we take Chinese as an example.

3.3.1 Chinese word segmentation problem

Word segmentation is the problem of dividing a string of written language into its component words. In English, the space is a good approximation of a word divider (word delimiter); however, in Chinese character string, there are no spaces as word delimiter; therefore, word segmentation becomes the most difficult problem in Chinese language processing.

In Fig. 2 (b), the two sentences are segmented different; the meaning is totally different.

The first sentence is segmented by author as: "乒乓" "球拍" "卖" "完" "了" "．"

The meaning is: Ping pong/ rackets/sell/finished/already.

The second sentence is segmented by author as: "乒乓球" "拍卖" "完" "了" "，"

The meaning is: Ping pong balls/auction/finished/already.

However, if no word delimiters, it's very difficult to know the author's real intents.

The two sentences are displayed the same in Fig. 2 (b) as the following.

"乒乓球拍卖完了．"

The ambiguity of word segmentation is caused by many to one mapping, multiple characters relate to one word; Ucode eliminates this kind of ambiguity by making each word relates to one and only one Ucode; in Fig. 2 (a), the segmentation semantics has been represented by Ucodes, which can be distinguished by Ubit right 0 rule; therefore, Ucode acts as word delimiters; understandable to machine; then machine can translate the segmentation semantics to human, for example, to display words in different colors.

3.3.2 Chinese polyphone word ambiguous problem

In Fig. 2 (b), the first character of Chinese word "会计师" is a polyphone character with two different phonemes, and the meaning is totally different.

The ambiguity of polyphone word is caused by one to many mapping; one character code relates multiple phonemes and meanings. Ucode eliminates this kind of ambiguity by making multiple Ucodes, each is only related to one phoneme and one meaning. Therefore, the right semantics is distinguished by Ucode as shown in Fig. 2.

However, it can't do so with Vcode, because no enough compaible Vcodes can be used.

4 HUMAN AND MACHINE UNDERSTANDABLE PROGRAM SEMANTICS

Obviously, programming languages are notations form, only understandable to people, not understandable to machine. Therefore, before a program can be run, it must be translated into machine language by compile or interpreter [6]. However, Ucode source program is totally different.

4.1 Ucode source program

Source program usually includes two parts: instruction part and data part. Instruction part is an Ucode sequence. Take the following program as an example:

yCoordinate = intercept + Slope * xCoordinate

There are four variables in the program; each notation form of them is associated by one Ucode, the human understandable semantics, as shown in Table 2.

TABLE II. UCODE OF VARIABLE ASSOCIATES NOTATION FORM AND ADDRESS

Ucode	$Ucode_{11}$	$Ucode_{12}$	$Ucode_{13}$	$Ucode_{14}$
Notations	yCoordinate	intercept	Slope	xCoordinate
Address	0x0000	0x0004	0x0008	0x000C

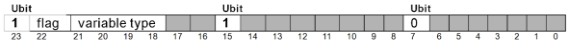
There are three operators in the program; each of them is represented by one Ucode, and the human understandable semantics of each operator is associated by the related Ucode as shown in Table 3.

TABLE III. UCODE OF OPERATOR ASSOCIATES NOTATION FORM

Ucode	$Ucode_{21}$	$Ucode_{22}$	$Ucode_{23}$
Operator	=	+	*

The Ucode format of keyword, operator, and variable is shown in Table 4, in which, 22^{nd} bit is flag, 0 for identifier, 1 for keywords or operators; for variables, the 16 bits, shaded grey, coding the memory address of each variable ; if 22^{nd} flag=1, 21^{st} acted as flag, 0 for keywords, 1 for operators.

TABLE IV. THE UCODE FORMAT OF KEYWORD, OPERATOR, AND VARIABLE



Assume the type of the four variables is float with 4 byte long. Then the relative address is show in Table 2.

The Ucodes of operators are predefined as show in Table 3, and the privilege of the operators is embedded in Ucode by coding the Ucode according to the privilege of operators; for example, here, the value of $Ucode_{23} > Ucode_{22} > Ucode_{21}$. So the operating rules have been embedded in Ucodes.

There is a program developing interface, just like the interface in section VII.

As program inputting, the variable name, such as: yCoordinate, intercept, Slope, xCoordinate, is inputted by user, variable type semantics is also inputted by selecting the type displayed. An Ucode is assigned to each variable, and the related address of each variable is filled in the Ucode according to the memory pointer, which is increased according to the type of variable.

At beginning, the interface initializes a variable table as the Table 2; and initialize the memory pointer=0.

As input "yCoordinate"; the interface first check if the variable has been inside Table 2, if not, then ask user to input the variable type; then assign an Ucode to the variable, here $Ucode_{11}$, and fill the variable type in the Ucode, and put the value of memory pointer into the Ucode as shown in Table 2; then, increase the memory pointer according to the variable type, here increase 4 because the variable type is float, 4 byte long; then put "yCoordinate" into the table 2.

The interface put $Ucode_{11}$ into the source program, and displays "yCoordinate".

Then, input "=", the interface put $Ucode_{21}$ into the program and display "=".

Similarly, input other variable and operators.

At last, the content of the source program is as:

$Ucode_{11}$ $Ucode_{21}$ $Ucode_{12}$ $Ucode_{22}$ $Ucode_{13}$ $Ucode_{23}$ $Ucode_{14}$

And displayed in notation form as:

yCoordinate = intercept + Slope * xCoordinate

This is one statement; the Ucode source file consists of a sequence of Ucode statements, an Ucode can be selected as deliminater among statement; for example, the feedback of ASCHII code.

The notation form can be any character form, not limited to English; and in the source program, user can input remarks, notes by any language.

4.2 The executiion of Ucode source file

The machine knows the data type, the addresses of the variables; the privilege of operators, the operating rules; therefore, the machine knows what instructions should use, what operation sequence, where to access the variables; i.e. how to execute the program.

There are three executable forms: human and machine understandable execution form, suitable for program developing and testing; second, machine understandable execution form; third, machine obeyed execution form.

In human and machine understandable execution form, both human and machine understandable semantics can be displayed.

The machine scanning the Ucode source program as following;

$Ucode_{11}$ $Ucode_{21}$ $Ucode_{12}$ $Ucode_{22}$ $Ucode_{13}$ $Ucode_{23}$ $Ucode_{14}$

From the type attribution of Ucode, machine knows there are three Ucodes of operator: $Ucode_{21}$ $Ucode_{22}$ $Ucode_{23}$ (relate to = , + , * ;); compare the value of $Ucode_{21}$ $Ucode_{22}$ $Ucode_{23}$; (the privilege of the operators by the value of their Ucodes), the highest is: $Ucode_{23}$ (for *); so the machine understands calculating $Ucode_{23}$ first; select an $Ucode_{buffer}$ for storing the middle value; and assign the memory address: x000F. Then, calculate operation "+", then store the value to $Ucode_{11}$. Therefore, machine can divide the cource program above into two five Ucode form as the following:

$Ucode_{buffer}$ $Ucode_{21}$$Ucode_{13}$ $Ucode_{23}$ $Ucode_{14}$;

$Ucode_{11}$ $Ucode_{21}$ $Ucode_{12}$ $Ucode_{22}$ $Ucode_{buffer}$.

A complier can generate two three address code after lexical, syntax and semantic analysis [6] as following:

*buffer=Slope * xCoordinate;*

yCoordinate = intercept + buffer.

It is clear that each of five Ucodes form relates to one three address code generated by a complier after lexical, syntax and semantic analysis [6].

Assuming variable: intercept, Slope, xCoordinate have be assigned value: 0.0, 0.5, 2.0, as show in Table 5.

Next, discuss how machine executes the source program in human and machine understandable execution form.

Machine executes the first five Ucodes as following

$Ucode_{buffer}$ $Ucode_{21}$$Ucode_{13}$ $Ucode_{23}$ $Ucode_{14}$;

Display to human as following:

*buffer=Slope * xCoordinate;*

TABLE V. UCODE OF VARIABLE EMBEDDED ADDRESS AND VALUE

Ucode	$Ucode_{11}$	$Ucode_{12}$	$Ucode_{13}$	$Ucode_{14}$	$Ucode_{buffer}$
Address	0x0000	0x0004	0x0008	0x000C	0x000F
Initial Value		0.0	0.5	2.0	
Last value	1.0	0.0	0.5	2.0	1.0

From the type attribution of Ucode, machine knows there are two Ucodes of operator: $Ucode_{21}$, $Ucode_{23}$ (relate to = , *); compare the value of $Ucode_{21}$ and $Ucode_{23}$; (the privilege of the operators relates the value of their Ucodes), the higher is: $Ucode_{23}$ ("*"); therefore, First calculate:

*Slope * xCoordinate;*

Extract the value in the address embedded in $Ucode_{13}$: address 0x08, value: 0.5; extract the value in the address embedded in $Ucode_{14}$: address 0x000C, value: 2.0;

Do operation of $Ucode_{23}$ "*": 0.5*2.0; and store the result to 0xF;

*buffer= Slope * xCoordinate = 1.0*

The second five Ucodes form is executed in the same manner.

At last, the value in the memory is shown in Table 5.

Human and machine understandable execution form is suitable for program developing and testing;

Machine understandable execution form is similar to human and machine understandable execution form, the difference is that the human understandable notations are not displayed.

The advantages of the human and machine understandable execution from and the machine understandable execution form is as following:

Machine knows the data type, the addresses of the variables; the privilege of operators clearly; therefore, parallelism is much easier. The machine understands what Ucode instructions can be calculated in parallel. The machine can efficiently schedule Ucode instructions, according to the resources available. Therefore, the execution speed is quickened; and the CPU could become the real brain of the machine.

For machine obeyed execution, first extract the machine instruction part and data part according to the machine available resource; and generate an executable file as the existed one.

Because the three execution form does not need compiler; they are machine independent, platform independent and network independent. For machine obeyed execution, the machine is just obeyed to execute an instruction sequence, knows nothing about the program, i.e. not understand the program.

The notations used in Ucode source program can be any language, any notations; there nearly no any syntax requirements. This means you can develop your program by any languages and with great freedom.

5 HUMAN AND MACHINE UNDERSTANDABLE WEB SEMANTICS

The aim of Berners-Lee's Semantic Web is to make web understandable to machine and human.

However, semantic web is only described in notation form, the goal can't be realized. Fortunately, Ucode can achieve the goal easily.

Web tags can be easily represented by Ucodes, for example, each Ucode of one type of three byte Ucodes in Fig. 2 can be used to represented one tag; one type Ucodes can represent about ten thousand tags.

Here, mainly discuss how to represent global address in both human and machine understandable form.

Global address is usually called as Uniform Resource Locator (URL). A typical URL is as the following:

http://example.org/semantic-web/Semantic Web

It's a notation form, only understandable to human; translating to machine is necessary; and the translation is called as domain name resolution.

URL can be divided into two parts, remote address and local address, as shown in Table 6.

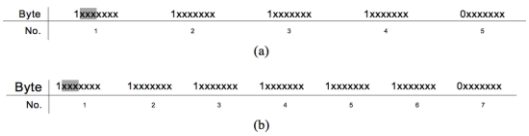
TABLE VI. REMOTE ADDRESS AND LOCAL ADDRESS

$Ucode_{remote}$	$Ucode_{local}$
http://example.org	/ semantic-web/Semantic Web

The local address understandable to machine and human has already discussed as shown in Table 1.

The remote address includes protocol and IP as shown in Table 7.

TABLE VII. DIAGRAM OF REMOTE SEMANTIC ADDRESSING



Assume the remote address is expressed in 5 byte Ucode format; the first bit of each byte is Ubit, the left 35 bits are Vbit, three Vbits for coding protocol type; the left 32 Vbits for coding IPv 4 as shown in Table 7. (a).

An $Ucode_{Remote}$ in Table 6 is associated with human understandable notation: http://example.org; and the protocol type and the IP address is embedded inside the Ucode. An $Ucode_{Local}$ is similarly as shown in Table 1 (b), which associates with human understandable notation: / semantic-web/Semantic Web, and also associates with the file address associated as shown in Table 1 (c).

Here, "http://example.org/semantic-web/Semantic Web" is described in Ucode form, understandable to machine and human.

The displaying of Ucode URL can be any language, any notations; there nearly no any syntax requirements. This means you can use any character or word to represent your name of URL.

As the $Ucode_{Remote}$ address space is not enough, the length of Ucode can be extended; 7 byte Ucode form as shown in Table 7 (b); then, the address space is 214 times space of IPv 4. Some years later, can be further extended to more bytes, but the old one can continue to use. The current using, the used before, and the shall use in future, are consistent with one another, this is called as time consistent. Therefore, the address space is unlimited and time consistent; domain name resolution no need any more.

6 SECURITY OF USC

Based on Ubit theory, the semantics of public contents can be made easily understandable to human and machine; moreover, the semantics of private contents can also be easily made not understandable to unauthentic persons and machines; therefore, the contents are much safer.

The Ucode contents are structured from the most bottoms, the foundation is secure.

Because machine understandable the languages, no translation needed, the vulnerabilities caused by translation are not existed; for example, DNS attacks are stemmed from domain name resolution; so DNS attacks no exist again.

Software becomes bigger and complicated; vulnerabilities can hardly be avoided in developing stage, updating and modification often needed. Adopting the human and machine understandable execution form, the source program executes step by step; developers can easily debug, easily update software and easily detect and repair vulnerabilities. The inside back door attack is also easily detected.

Computer virus infects a program file by inserting virus code into executed code; which consists of a sequence of instructions and data; because the machine just obeyed to execute the instructions, has no way of knowing what is data and what is program; therefore, virus injection is hardly prevented.

However, with the information inside Ucodes, the machine not only knows what is data and what is program; but also knows the data type, the addresses of the variables; the computation rules clearly. That is the address space is numbered already, variables in program space and data space mapped. It's very difficult for hacker to insert virus into Ucode program.

This is just like the following scenario: when people are waiting on a line, the line is not clearly ordered, not know who is in front and who is behind; then it is easy to cut in. However, if the line is ordered and each one is numbered, each one knows who is in front and who is behind; then it is difficult for someone to cut in line.

Because type and size attribution of each variable is embedded in related Ucode, as source program executing, the machine knows the size of each buffer clearly; overflow attack can be prevented effectively.

The existed security is only based on value operation; the encryption strength only depends on value computing complex [7-9].

USC security is based on both value and structure encryption; the encryption strength not only depends on value computing complex; and also depends on structure computing complex; therefore, much stronger.

USC security can be further strengthened by combining with in-out key, in-out password and in-out nonce. In-out key makes key distribution absolutely secure; in-out password makes password unbreakable; and in-out nonce makes replay attacks useless [10]. Ubit theory combined with in-out key, in-out password and in-out nonce can lay a solid foundation of computer security.

7 AN INTEGRATED INTERFACE OF USC

7.1 Introduction to an experimental interface of USC

An experimental interface of USC is shown in Fig. 3: (a) Interface Menu (only part of the experiment menus listed). (b) Interface, an integrated input method; which is used to create, convert, manage, and retrieve contents by one's intents.

The interface bases on precisely relation between Ucodes and objects, one to one mapped.

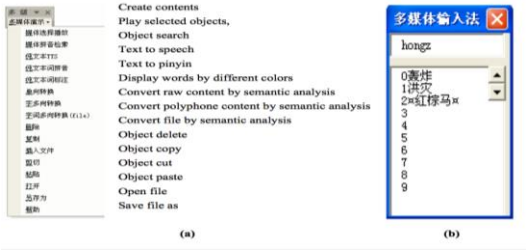


Figure 3. *An experimental Interface of USC (a) Interface Menu; (b) An integrated input method*

(a) Interface Menu

- Creating contents based on author's intents
- Converting raw contents by semantic analysis

"Convert raw contents by semantic analysis", "Convert raw polyphone contents by semantic analysis"

"Convert raw content file by semantic analysis":

The raw content refers to contents with ambiguities.

- Retrieving contents based on user's intents

"Play selected objects", "Object search", "Text to speech", "Text to pinyin", "Display word by different colors"

- Editing menu

"Object delete", "Object copy", "Object cut", "Object paste"

- File menu:

"Open file", "Save file as"

(b) An integrated Interface

The interface is an integrated input method, For example, in (b), input five letters, "hongz", then, three objects are matched in the window: two word objects, each with two characters, and an image object. By this interface, user can interact precisely and semantically with the contents, such as audio, video, image object, Chinese text objects, English text and etc. Anyone from anywhere can access the contents, no matter want kind of languages, want kind of objects.

This interface makes ambiguity no exist; and is machine and human understandable. Author creates contents based on author's intents; user retrieves contents based on user's intents.

7.2 Create semantic contents based on author's intents

7.2.1 Create video, audio and image object semantic contents based on author's intents

In Fig. 4 and Fig. 5, three video, an audio and an image objects have been created into the contents.

Each object relates to one, and only one Ucode in the contents; that is author's intents map to contents one to one.

Each object is distinguishable; the object type embedded in the Ucode; and object address relates to the Ucode. This makes the

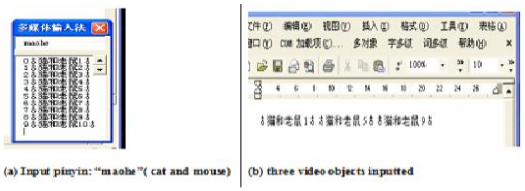


Figure 4. *Three video objects inputted*

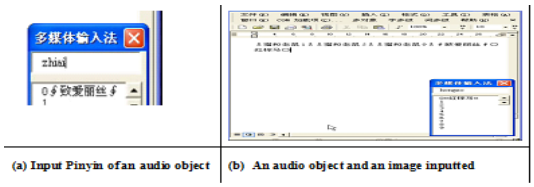


Figure 5. *An audio object and an image object inputted*

semantics of contents understandable to machine, machine can access the object directly; no semantic translation needed.

The Ucode also relates to human understandable languages. For VSC, each video, audio and image relates to a notation string, not understandable to machine.

7.2.2 Create phoneme semantics based on author's intents

In Fig. 6, Author inserts a Chinese three character word object with polyphone character: "会计师" into the contents.

The first character is a polyphone character, with two different phoneme, and different meaning; there are two codes, "BBE1" and "ADF0", and each relates to one phoneme; here "BBE1" is selected according to user's intent as shown in Fig. 2.

Author's intents also map to contents one to one.

Further, the three characters can be segmented as: "会计师" or "会计"; here one Ucode maps to "会计师"; it is impossible to be "会计"; so the contents with author's real intents.

For VSC, the word relates to three characters, one to three mapping, and not one to one mapping. The polyphone character is only expressed by one Vcode: "ADF0"; one Vcode in contents relates to two different intents, neither one to one mapping.

7.2.3 Create word segmentation semantics based on author's intents

The following example illustrates Chinese word semantics inputting

In Fig. 7, as author inputs "pinpang", two words: "乒乓", "乒乓球" appear in the input window; if author

Figure 6. *Insert a Chines three character word object with poly-phoneme character: "会计师" between two video objects: "猫和老鼠 3", and "猫和老鼠 9".*

Figure 7. *word segmentation semantics inputting based on author's intents.*

Selects "乒乓", then the object "乒乓" is saved as a 4 byte Ucode; if author selects "乒乓球", then the object "乒乓球" is saved as a 6 byte Ucode as shown in Fig. 2.

Here, Ucode is created by combining the character codes according to user's selection. Each word is an object, and relates to one and only one Ucode in the contents, author's intents map contents one to one.

However, for VSC, multiple characters in contents relates to one author's intents; this is also multiple to one mapping.

7.3 Retrieve semantic contents by user's intents matching author's intents

The main characteristics of USC Retrieving are:

Precisely Author's intents: Author's intents are expressed precisely;

Precisely User's intents: User's intents are expressed precisely.

Precisely matching: User's intents matching author's intents precisely;

Directly and efficiently retrieving: because machine understandable semantics, machine can extract the matched contents directly and efficiently.

The content retrieving is based on user's intent.

As in Fig. 7, if user wants to search object: “乒乓”; “乒乓”is an Ucode, this can only be found in the first sentence of the contents;, if user wants to search object: “乒乓球”; “乒乓球”is an Ucode, this can only be found in the second sentence of the contents; if user wants to search object: “拍卖”, this can only be found in the second sentence. User’s intents map to author’s intents.

User’ retrieving can be in different way: mixed way, it search’s all kinds of contents, all kinds of objects to match user’s intents.

Retrieving in specified area, for example, retrieving from audio objects to match user’s intents, from video object to match user’s intents, from image object to match user’s intents, from one-Chinese character word objects to match user’s intents, from two-Chinese character word objects to match user’s intents; from polyphone words to match user’s intents, from specified languages to match user’s intents, and etc.

In order to be consistent with VSC, retrieving can also be done in VSC retrieving manner.

Display word segmentation semantics to human as retrieving, in Fig. 8, the word semantics is displayed in different colors.

The word segmentation semantics can also be expressed by “Text to speech” as shown in Fig. 9.

There are polyphone word objects in Fig. 10; the word phoneme semantics is distinguished by Pinyin.

乒乓球拍卖完了. 乒乓球拍卖完了.

Figure 8. *The word segmentation semantics created by author’s intents is displayed by different colors*

Figure 9. *The word segmentation semantics distinguished by text to speech in different time gaps*

The semantics of polyphone words can be retrieved correctly by user’s intents. There are two “单” in Fig. 10 and 11;

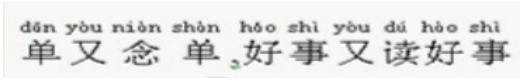


Figure 10. *displaying the word phoneme semantics by Pinyin*

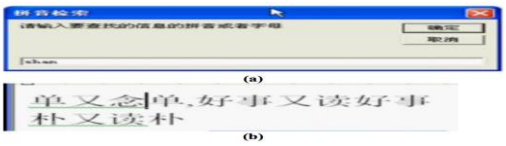


Figure 11. *retrieving by user’s intents (a) User input “shan” to search the surname “单” (b) the surname “单” has been located by a vertical line*

The first “单”, pronounced “dan”, meaning “singular”; the second “单” (the fourth character), pronounced “shan”, a surname. These are author’s intents.

User want to search the surname “单”, by input “shan”, as showed in Fig. 11 (a); the result located at the fourth character as showed in Fig. 11 (b); it is impossible to locate the first character “单” (dan), because it is a different objects. Here, User’s intent matched user’s intent.

A real story about ambiguous word semantics once happened years ago. Two persons, A and B signed a contract as following:

“还欠款 20 万元”, the first character “还” is ambiguous; there are two different pronunciations and two different meaning. It is just the different semantics incurred a lawsuit. In the court,

Person B based the first semantics: A “hasn’t paid ¥20000” to B;

Person A based the second semantics: A “has paid ¥20000” to B”.

The judger no way judged who is right.

Semantic analysis is the foundation of semantic computing [3]; it involves extraction of context-independent aspects of a sentence's meaning. However, even the judger can’t extract the author’s real intents, how an agent can extract the author’s real intents?

Because Vbit and Vcode have no way to express or store author’s intents precisely, the real intents are often blurred, some of the real intents can be extracted, but some can’t. Therefore, the ability of semantic analysis can’t be overvalued.

However, with this interface, there is only one object of two word objects in the contract; therefore, the meaning is precisely, no ambiguity in the contract, with the real intents; therefore, this kind of lawsuit would never happen again.

7.4 Retrieve any kinds of objects

Example 1, Retrieve one video object: “maohelaoshu10” as shown in Fig. 12.

Example 2, Retrieve image object as shown in Fig. 13.

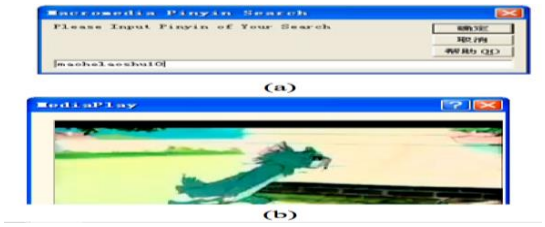


Figure 12. *(a) Search video: “cat and mouse 10”, (b) play the searched video*

Figure 13. *Image object searched, and displayed.*

7.5 Retrieve the objects selected

User can select the objects by user's intent, then click "Play selected objects", then all selected objects plays one by one, for video object, play the video; for audio object, play the audio; for image, display the image; for text, read the text by text to speech.

8 THE ARCHITECTURE OF USC

The architecture of USC is shown in Fig. 14.

The contents consist of every kinds of objects which are represented in different kinds of Ucodes. There exist contents which are represented in Vcodes, which can be represented by Ucodes in processing.

From precisely view, the contents can be divided into two classes: one is in Ucodes with precisely author's intents; another is in Vcodes with author's blurred intents, called as raw contents.

From security view, contents can be divided into two classes: public contents and private contents.

The interface of USC consists of various Ucode programs, by which author can create contents based on author's intents; user can retrieve contents based on user's intents; manager can manage contents. Anyone could be author, user or manager.

Because the Ucode programs are machine independent, platform independent and network independent, and Ucodes are space consistent and time consistent, contents can be anything, can be processed in anywhere, created, modified, retrieved or managed by anyone in anytime.

Managements of contents can be:

- Convert raw contents by semantic analysis
- Management of Interface tools
- Security management

…

9 CONCLUSION

USC can make computer scientists' dream become true; however, this paper is very elemental; this is just the beginning of USC.

It is the right time to shift VSC to USC; with more scientists taking part in, a bright future is in front of us.

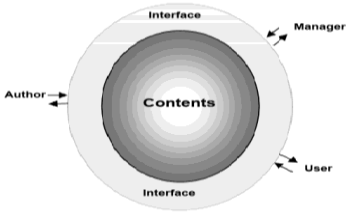


Figure 14. *The architecture of USC*

Acknowledgments Thank Bin Wu, my wife, for her great contributions and fully supporting.

REFERENCES

[1] J. Glenn Brookshear, Computer Science: An Overview (11th Edition), Pearson Higher Education, 2012

[2] Tim Berners-Lee, http://www.techrepublic.com/article/an-introduction-to-tim-berners-lees-semantic-web/

[3] IEEE Press Editorial Board, Lajos Hanzo, Phillip C.-Y., Sheu Heather Yu, C. V. Ramamoorthy Arvind K. Joshi, Lotfi A. Zadeh, SEMANTIC COMPUTING, IEEE Press, the Institute of Electrical and Electronics Engineers, Inc. 2010

[4] Shengyuan Wu, Methods and apparatuses of digital data processing, PCTIB2013060369, 11, 2013

[5] Shengyuan Wu, Introduction to Multilevel Mark Coding Theory, Proceedings of The 2007 International Conference on Foundations of Computer Science (FCS'07), June 2007

[6] Aho, Alfred V. Lam, Monica S, Compilers Principles, Techniques and Tools, New Delhi Pearson, 2011, P 1,1 ,1, 99, 522, 382-385

[7] William Stallings, Cryptography and Network Security principles and Practice, Fifth Edition, Pearson Education, Inc., 2011

[8] Gilles van Assche, Quantum Cryptography and Secret-Key Distillation, Cambridge University press, 2006

[9] Yin Hao, Han Yang, The principles and technology of quantum communication, Electronic Industry Press, 2013

[10] Shengyuan Wu, One-time Pad Cipher Based on Out-Key Distribution, Proceedings of The 2014 International Conference on Wireless Networks of Computer Science (ICW'14), 07, 2014