Content Adaptation For Context-Aware Service

Jui-Fa Chen
Department of Computer Science and Information Engineering, Tamkang University, New Taipei City, Taiwan
Email: alpha@mail.tku.edu.tw

Ying-Hong Wang, Jingo Chenghorng Liao, Chen-An Wang
Department of Computer Science and Information Engineering, Tamkang University, New Taipei City, Taiwan
Email: inhon@mail.tku.edu.tw, jingo@cs.tku.edu.tw, chenan@mail.tku.edu.tw

***Abstract*—The current and developing trend for consumers is to access web contents and applications anytime, anywhere, and on any devices. Most of Internet services and most of web contents have been designed for desktop computers, and often contain rich medias, such as images, audios, and videos. However, some devices are different from network connectivity, processing power, storage capacity, display size, and formative handling capability. In many cases, the content designed for computers is not suitable for new (and often mobile) devices. Therefore, content adaptation is needed in order to optimize the service for different devices and access methods. This research discusses the context issues for web content adaptation. The CC/PP and UAProf are two related standards that define the format to describe the capabilities of devices for accessing content. A context-aware environment should allow adaptive access to context information. In this paper, first we proposed an inference mechanism for context-aware service. Through this inference mechanism, users using different devices can get appropriate contents based on inference results. Second, we can demonstrate the correlation between classes and individuals, and provide better scalability by means of building ontologies. Lastly, SWRL depends on ontology based rule languages. Rules written based on SWRL can directly use an established object relationship from ontology.**

***Index Terms*—Context-Aware, Adaptation, SWRL, Inference, UAprof, Web Services**

I. INTRODUCTION

Adaptation means a process of selection, generation, or modification of contents (texts, images, audios and videos) to suit the user's computing environment and usage context. In general, web pages incorporate a variety of media types and when using a PC connected to a website, it is possible to see the original web page (without adaptation) with headings, photos, texts and videos. When accessing the same service with a mobile device, the image is compressed. The text is summarized to one paragraph and the video is delivered as text or an image, depending on bandwidth and the capabilities of the device.

Adaptation can take place on a client, a server or an intermediate proxy [1][2]. In a client-based adaptation approach, the client needs to receive the same media encoding in different versions although only one will be used. In the fact, the other problem is all computational overheads are shifted to the client in the system.

In server-based adaptation, the functionalities of the web server are enhanced on content adaptation. Transmission time is reduced because the contents have already been adapted. Traditionally, some variants of the same content are stored on the server, and it selects to match the client identification. A common way for providing content to different devices is to store the content in Extensible Markup Language (XML) and uses Extensible Style Language Transformation (XSLT) to convert the content to the appropriate makeup language.

In the proxy-based adaptation approach, a proxy server analyzes and transcodes the contents before sending the results to the client. Content servers and proxies need to know what kind of device from the request in order to send the right contents

Context is the information that characterizes the interactions between humans, applications and the surrounding environments [3]. Many researchers have tried to define context, such as Schilit et al. who assigned context to three categories [4]: computing context, user context, and physical context. Muldoon et al. defined context as an aggregation of the user's location, previous activities, and preferences [5]. J. Sun adopted the same definition and even added physiological information to user context [6]. Huang et al. defined the context of an entity as a collection of semantic situation information that characterized the entity's internal features or operation and external relations under a specific situation.

Context-aware computing, as discussed by Dey and Abowd, refered to the ability of computing devices to detect and sense, and interpreted and responded to aspects of a user's local environment and the computing devices themselves [7]. Context-aware content adaptation dynamically adapts behavior to the user's current situation without user intervention.

In this paper, we address a content adaptation approach for a proposed context-aware architecture. The next section offers a survey of related researches about this approach. Section 3 describes the context-aware system architecture. Content adaptation is presented in section 4. And the final section is conclusions and future research that we presented.

II. RELATED WORK

doi:10.4304/jsw.7.1.176-185

With the development of network technology, the web content can offer more and more abundant. However, there are certain limitations among mobile devices and we are unable to fill all information in these devices. Context-aware technology can perceive the status of mobile devices and can adapt the contents for better presentation, especially for the limitations on network connectivity, processing power, storage capacity, display size, and formative handling capability.

A. The Semantic Web

Most meaning on the web today is inferred by users who read web pages. The semantic web is supposed to make data on the web readable and understandable, both to users and to machine. [18]. Figure 1 illustrated semantic web stack.

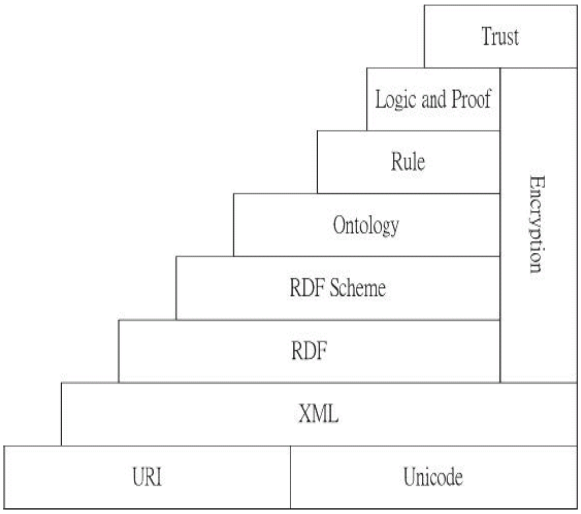

Figure 1. Semantic Web Stack

XML: XML is not a language. It is actually a set of syntax rules for creating semantically rich markup languages in a particular domain. The first key principle of XML is markup and is separated from content. An XML element is an XML container, consisting of a start tag, content (other elements, text, or both), and an end tag (except for empty elements), which use a single tag denoting both the start and end of the element. Elements maybe have zero or more attributes, each with its associated value. The second key principle of XML is that a document is classified as a member of a type by dividing its parts, or elements, into a hierarchical structure known as a tree [19][20].

A well-formed XML document must satisfy the following requirement [20]:

- It must be a tree with a single root.
- Every start tag must have a matching end tag.
- All attributes must be quoted.
- Empty tags must have a trailing slash.
- All entities must be declared.
- Child elements must be properly nested.

The validity of an XML document can be determined by comparing it to a specified Document Type Definition (DTD) or a specified XML Schema document. XML schema is a definition language. That enables you to constrain conforming XML documents to a specific vocabulary and a specific hierarchical structure. There are two types of documents, a schema document and multiple instance document, that conform to the schema. A schema can play two roles [20]:

- Template for a form generator to generate instances of a document type.
- Validator to ensure the accuracy of documents.

RDF: The Resource Description Framework (RDF) is an XML-based language to describe resources. While the definition of "resource" can be quite broad, let's begin with the common understanding of a resource as an electronic file available via the web. Such a resource is accessed via a Uniform Resource Locator (URL). In RDF model, a statement is often called a "triple" because it has three parts. Those three parts are described in terms of the grammatical parts of a sentence: subject, predicate, and object. In common, the predicate is called the property of the triple. And the object is called the value of the triple. Figure 2 displays the elements of the tri-part model.

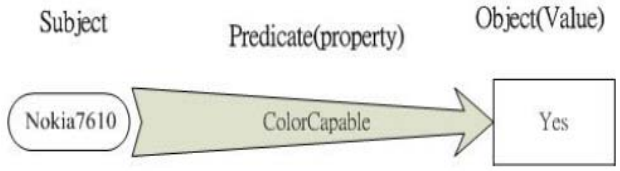

Figure 2. The RDF Triple

To play its role of describing data and meta data, RDF needs to include the following capabilities [21]:

- Able to describe most kinds of data that will be available.
- Able to describe the structural design of data sets.
- Able to describe relationships between bits of data.

RDF defines three special resource types to group resources or literal [19] [21] [20]:

- Bag: The element is used to describe a list of values that does not has to be in a special order. Duplicates are allowed in the collection.
- Sequence: The element is used to describe an ordered list of values. Duplicates are allowed in the collection. A reason to use a sequence would be to preserve the alphabetical order of elements.
- Alternate: The element is used to describe a choice of multiple values or resources. This is referred to as a choice in XML. Some examples would be a choice if image format (JPG, GIF, PNG, GMP).

There are multiple reasons that RDF has been weak [21]:

- RDF does not yet play well with XML documents.
- Parts of RDF are complex.
- Early RDF examples are weak.

There are five primary reasons that RDF's adoption will grow [21]:

- Improved tutorials
- Improved tool support
- Improved XML schema integration

- Ontologies
- Non-contextual modeling

RDF schema is used to define vocabularies. It used RDF to define a standard set of predicates that enable simple semantic relationships to be captured [20]. The data model for RDF schema allows you to create classes of data. A class is defined as a group of things with common characteristics and a template for an object composed of characteristics and behaviors. Because a class is the template, you can create many instances. An object is one instance of a class. Section 2.6 displays the Protégé Editor developed by Stanford University. It allows you to easily describe classes and class hierarchies. It allows also you to generate both the RDF schema and RDF document if you create instances of the schema [19].

Ontology: Ontology is a popular research issue in various communities such as knowledge engineering, natural language processing, intelligent information integration and knowledge management. Singh and Huhns define that an ontology is a kind of a knowledge representation describing a conceptualization of some domain [20]. Ontology specifies a vocabulary including the key terms, their semantic interconnections, and some rules of inference. It provides a shared and common understanding of a domain that can be communicated between people and heterogeneous and widely spread application systems. Ontology has been developed in AI to facilitate knowledge sharing and reuse. Ontology provides an explicit conceptualization that describes the semantics of the data. [22] [23].

Currently computers are changing from single isolated device to entry points into a worldwide network of information exchange. Therefore, support in the exchange of data, information and knowledge is becoming the key issues in communication technology. It can communicate between people and application systems. Providing shared and common domain structure is becoming essential, and it uses to describe the structure and semantic of information exchange. Now, Internet technology and WWW are the main technology infrastructure for on-line information exchange. It is not surprising to see that a number of initiatives are arising in this area to provide notations for data structures and semantics. For example:

- The resource description framework (RDF).
- The Extendible Markup Language (XML).
- XML schemes provide a standard for describing the structure and semantics of data.
- The transformation language of XSL (XSL-T).
- Various querying language for XML (XQL, XML-QL).

Depending on their generality level, different types of ontologies may be identified that fulfill different roles. We can distinguish the following ontology types:

- Domain ontology: capture the knowledge valid for a particular type of domain.
- Metadata ontology: like Dublin Core [24] provide a vocabulary for describing the content of on-line information sources.
- Generic or common sense ontology: aim at capturing general knowledge about the world.
- Representational ontology: do not commit themselves to any particular domain.

Ontological engineering is concerned with the principled design, modification, application, and evaluation of ontologies. Ontologies can be adopted in situations where the capability for representing semantics is important to overcome XML's disadvantages in terms of maturity. One well-known ontology language, OWL (Web Ontology Language), is designed to process information instead of just presenting information to humans. OWL facilitates greater machine interpretability of web content by providing additional vocabulary along with formal semantics.

OWL has three increasingly-expressive sublanguages: OWL Lite, OWL DL, and OWL Full. OWL is used when the information is contained in documents and needs to be processed by applications. It also opposes to situations in which the content only needs to be presented to humans. OWL can be used to explicitly represent the meanings of terms in vocabularies and the relationships between those terms. This representation of terms and their interrelationships is called "Ontology". OWL has more facilities for expressing meaning and semantics than XML, RDF, and RDF-S do; Thus, OWL goes beyond these languages in its ability to represent machine interpretable content on web. OWL is a revision of the DAML+OIL web ontology language and incorporates lessons learned from the design and application of DAML+OIL.

An ontology includes the following [19]:

- Classes (general things) in the many domains of interest.
- Instances (particular things).
- Relationships among those things.
- Properties(and property values) of those things.
- Functions of the processes involving those things.
- Constraints on and rules involving those things.

B. Methods of Client Capability Recognition

The hardware resource of mobile devices is very limit. We can not expand the desired information arbitrarily. So, we have to select the most useful information rather than all the information. In order to deliver suitable content for different devices and user profiles, a website needs to be able to differentiate between the capabilities of different devices. The following possible methods are for providing device recognition.

Web clients identify themselves when they send requests to web servers. The current HTTP/1.1 standard uses four Accept header fields to describe the capabilities and preferences of the client: Accept, Accept-Charset, Accept-Encoding, and Accept-Language. The Accept field describes the MIME types which are accepted by a browser. The other Accept header fields describe preferences for the character set, encoding, and language. Information in HTTP headers is limited and hard to extend. It was designed for browser descriptions and lacked the meaning for context and device capability.

In addition to Accept headers, clients send a User-Agent header to identify themselves. It is very simple but enough to provide some client-specific information, such as browser, operating system and sometimes hardware information. However, a User-Agent header only works for nearly static device properties. With hundreds of different browsers in use today, it gets tricky to support everyone by relying on the User-Agent field.

The User-Agent header contains information about the browser and the operating system via the request and sometimes hardware information (see Table 1 for examples).

Table I shows examples of the request header fields produced by different browsers:

Table I.
Examples Of User-Agent Headers

IE7 in Windows XP	Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; ozilla/4.0(compatible; MSIE 6.0; Windows NT 5.1; SV1);.NET CLR 1.1.4322;.NET CLR 2.0.50727;.NET CLR 3.0.04506.30;.NET CLR 3.0.04506.648; InfoPath.2)
Firefox 3.0.1 in windows XP	Mozilla/5.0 (Windows; U; Windows NT 5.1; zh-TW; rv:1.9.0.1) Gecko/2008070208 Firefox/3.0.1
Nokia 6230 built-in browser	Nokia 6230/2.0 (03.06) Profile/MIDP-2.0 Configuration/CLDC-1.1
Nokia 6600 built-in browser	Nokia 6600/1.0 (3.42.1) SymbianOS/7.0s Series60/2.0 Profile/MIDP-2.0 Configuration/CLDC-1.0

A resource might have many diverse presentation types, such as different languages, media types or versions. And we usually select the best method that provides an index for desired data that we want. But some browsers can only provide the specific header information by itself when requesting. Then the server will acquire these information directly from the browser. It's important to note that none of these browsers correctly obeys the HTTP/1.1 content negotiation standard. Because a browser obeys the standard, the server will be hard to acquire the information of browser from these headers. We mention again there are two kinds of content negotiations in HTTP: server-driven negotiation and agent-driven negotiation, these two kinds of negotiations are orthogonal and thus may be used separately or combinatively. So when the client sends a request, the server will acquire these information.

Scripting languages such as JavaScript, VBScript, Jscript, and WMLScript can provide device–specific information. ActiveX components can also report the browser device and connection characteristics.

C. The Standard of Context Extraction

Several standards have been defined and also address the interoperability problems. A few of these standards are described here.

Composite Capability/Preferences Profiles (CC/PP): As some variety of devices grows on Internet, we need to tailor the contents specifically for the different devices. The User-Agent header information discussed earlier is insufficient. The Composite Capability/Preference Profiles (CC/PP) [8] recommendation from the W3C describes a method for using the Resource Description Framework (RDF) of the W3C to provide a way for user agents and browsers to specify metadata about device capabilities and user preferences. This information can be provided by the user to servers and content providers. The servers can use this information describing user preferences to customize the services or contents. CC/PP is an extensible framework that can be used for communicating the delivery context (screen size, audio capabilities, bandwidth, etc) from a device or a web server, resulting in the delivery of web content usable on a given device. CC/PP allows different devices to specify their capabilities in the some way.

There are several implementations of the CC/PP standard. Below is a list of a few of the current implementations:

- Nokia Activ Server
- Cysive Cymbio Interaction Server (Cysive)
- Aligo M-1 Mobile Application Server (Aligo)
- DELI (HP laboratories Open Source CC/PP server API) [9]
- Wokup! Server and Studio (Wokup)
- J2EE CC/PP Processing Reference Implementation (Sun)
- Device Independence Access for the World Wide Web (W3C Greece Office)

DELI is an open-source library developed from HP Labs. It allows Java Servlet to resolve HTTP requests containing user context information from CC/PP or UAProf capable devices and query the resolved profile. It also provides to support for legacy devices so that the proprietary user context descriptions currently used by applications. It can be replaced by standardized CC/PP descriptions. In this paper, DELI is used to acquire user context information.

User Agent Profile (UAProf): The User Agent Profile (UAProf) specification developed by the Open Mobile Alliance (OMA, former WAP Forum) is a concrete CC/PP vocabulary dedicated to mobile phone description, and defines an efficient transmission of the CC/PP descriptions over wireless networks. Mobile phones complying with the UAProf specification provide CC/PP descriptions of their capabilities to servers. Content servers, gateways, and proxies can use this information and optimize the content for the device and the user. UAProf consists of description blocks for the following key components:

- Hardware Platform: the type of device, model number, display size, input and output methods, color capability, etc.
- Software Platform: operating system software, mine type, character sets, transfer encoding, video and audio encoders supported by the device, etc.
- BrowserUA: browser info, HTML/XHTML, Java, JavaScript, frames and tables capability.
- Network Characteristics: GSM/GPRS capability, security support, Bluetooth support.

- WAP Characteristics: WAP/WML support, deck size.
- Push Characteristics: push content types, push message size.

UAProf files are quite comprehensive and they tend to grow large. That is why only the URL of the device profile is transmitted from the mobile terminal to the server. The content server fetches the profile from the device profile repository and may store it in its own database for further use. The WAP gateway or HTTP proxy must support UAProf header forwarding.

D. Semantic Web Rule Language (SWRL)

Semantic Web Rule language (SWRL) is based on a combination of the OWL DL and OWL Lite sublanguages of the OWL. Web Ontology Language with the Unary/Binary Datalog RuleML sublanguages of the Rule Markup Language [10][11]. SWRL includes a high-level abstract syntax for Horn-like rules in both the OWL DL and OWL Lite sublanguages of OWL. A model-theoretic semantics is given to provide the formal meaning for OWL ontologies, including rules written in this abstract syntax. The rules are of the form of an implication between an antecedent (body) and consequent (head) [10].

SWRL's structure is composed of four parts: Imp, Atom, Variable and Building, as explained below:

- Imp: the head and body of the rules, consisting of Atoms.
- Atom: descriptions of the head and body.
- Variable: the variables used in recording rules.
- Building: records the logic comparison relationship that can be used by SWRL.

E. Jess

Jess is a rule engine and scripting environment written entirely in Java language [10][12][13][14]. Jess is small, light, and one of the fastest rule engines available. Jess is fast rule engine and its powerful scripting language gives the access to all of Java APIs. Jess can solve the difficult many-to-many matching problem. In Jess, a rule consists of a left-hand side(LHS) and a right-hand side(RHS).

Jess is a reimplementation of a subset of Clips in Java. Jess implements some additional functionality, not provided by Clips. Like Clips, reasoning in Jess is based on a list of known facts and a set of rules that try to match on these facts in its fact base.

Facts are basically a list of known values. The rule's preconditions are patterns with wild cards and conditional expressions, which match such facts. Rules can assert new facts when they fire, which in turn could result in other rules firing. Jess uses the Rete algorithm to ensure efficient pattern matching and rule activation. Integrating Jess with other Java programs is easy because it has a well-defined API for controlling the reasoning engine from Java. For example, Java programs can send expressions to the Jess engine for evaluation. In addition, we can define new Jess functions in Java. The JessTab implementation takes advantage of the Jess API to map information in the Protégé knowledge base to Jess facts and to extend Jess with additional functions for communication with Protégé.

F. Protégé

Protégé Ontology Editor is a free, open source ontology editor and knowledge-base framework [12]. The Protégé platform supports two main ways of modeling ontologies via the Protégé-Frames and Protégé-OWL editors. Protégé ontologies can be exported into a variety of formats including RDF, OWL and XML Schema. Protégé is based on Java and is extensible. Protégé also provides a plug-and-play environment that makes it a flexible base for rapid prototyping and application development.

G. Agent

Agent technique is one of the important technologies developed to support Internet applications. Especially, Internet and WWW technologies break the limitation of space for enterprise marketing, and the agent techniques solve the problems of temporality. Even though the users are off-line, the agents can still work in the network and play the pre-defined scenarios. Agent is software that assists the behavior of users in the network.

In the book titled, Programming and Deploying Java Mobile Agents with Aglets [25], the authors, Danny B. Lange and Mitsura Oshima, define agent into two viewpoints:

- End-User Perspective: An agent is a program that assists people and acts on their behalf. Agents function by allowing people to delegate work to them.
- System Perspective: An agent is a software object that:
  - Is situated with an execution environment.
  - Processes the following mandatory:
    - Reactive: senses changes in the environment and acts according to those changes.
    - Autonomous: has control over its own actions.
    - Goal-Driven: is proactive.
    - Temporally continuous: is continuously executing.
  - And many posses any of the following orthogonal properties:
    - Communicative: able to communicate with other agents.
    - Mobile: can travel from one host to another.

In the book titled, Mobile-Agent Systems [26], the authors, Jacques Ferber, defines agent as follows:

An agent is a physical or virtual entity:

- Which is capable of acting in an environment,
- Which can communicate with each other
- Which is driven by a set of tendencies,
- Which possesses resources of its own,
- Which is capable of perceiving its environment(but to a limited extent),
- Which has only a partial representation of this environment(and perhaps none at all),

- Which possesses skills and can offer services,
- Which may be able to reproduce itself,
- Whose behavior tends to towards satisfying its objectives, allowing for the resources and skills available to it and depending on its perception, its representations and the communications it receives.

In the book, "Programming and Deploying Java Mobile Agents with Aglets" [25], the basic properties of agents are "Reactive, Autonomous, Object-Oriented, Communicative, Mobile, Learning, and Believable".

A mobile agent is an agent, which has the capability of mobility in the network. Danny B. Lange lists several advantages of mobile agent technologies are applied on network: [25]

- They can reduce the network load.
- They can overcome network latency.
- They can encapsulate protocols.
- They execute asynchronously and autonomously.
- They adapt the changes dynamically.
- They are heterogeneous and fault-tolerant.

Agents are pieces of software that work autonomously and proactively. Conceptually they evolved out of the concepts of object-oriented programming and component-based software development. A personal agent on semantic web will receive some tasks and preferences from the person, seek information from web sources, communicate with other agents, compare information about user requirements and preferences, select certain choices, and give answers to the user.

H. Examples of Content Adaptation System

Cocoon is an open-source Java server framework that allows dynamic multi-channel Web publishing of XML content using XSLT transformations [15]. By relying on XML to describe content, and XSLT as a means of transforming that content into multiple formats, Cocoon provides a platform for building applications with separation between content, logic, and presentation. Cocoon can also serve static files as well as dynamically generated responses with wide variety of output formats including XML, XHTML, PNG, JPEG, SVG, and PDF. Support multiple clients, layouts and languages without code duplication. Cocoon includes CC/PP and UAProf support through another open-source product called DELI, which is a library that provides an API to allow Java servlets to resolve HTTP requests containing user context information from CC/PP or UAProf enabled devices and query the resolved profile.

DELI is an open-source library originally developed at HP Labs that allows Java servlets to resolve HTTP requests containing delivery context information from CC/PP or UAProf capable devices and query the resolved profile [9]. It also provides support for legacy devices so that the proprietary delivery context descriptions currently used by applications can be replaced by standardized CC/PP descriptions. DELI allows Cocoon to determine the user context of a client device, for example a UAProf enabled device.

Oracle is better known for its database products. Oracle Application Server Wireless is the mobile component of the Oracle Application Server [16]. It includes Multi-Channel Server which abstracts the underlying networks, protocols, devices and gateways to one protocol and one language, namely HTTP and XML. The server enables applications to be accessed through multiple delivery methods and devices such as 2-way pagers, devices with SMS, MMS, voice, or WAP capabilities, and handheld computers. The server automatically translates applications written in Oracle Wireless XML, XHTML Mobile Profile, or XHTML+XForms for any device and network. For example, an XHTML+XForms application passed through the Multi-Channel Server will be translated to VoiceXML if a phone is accessing the application through a voice interface or to WML if a WAP phone issues the request. The tools to support the most popular devices are maintained and regularly updated by Oracle. As new devices come to the market, support is added to the Multi-Channel Server. The framework also supports session management, service linking, and location awareness.

IBM WebSphere Transcoding Publisher is server-based software that dynamically translates Web content and applications into multiple markup languages and optimizes it for delivery to mobile devices- based on user preferences and device capabilities [17]. Content adaptation for many devices and languages eliminates the need to maintain multiple versions of a web site. The product can convert images to links to retrieve images, convert tables to lists, and remove comments and features not supported by the device.

III. CONTEXT-AWARE SYSTEM ARCHITECTURE

The context-aware system architecture is presented in this section. Figure 3 shows the proposed context-aware system architecture.

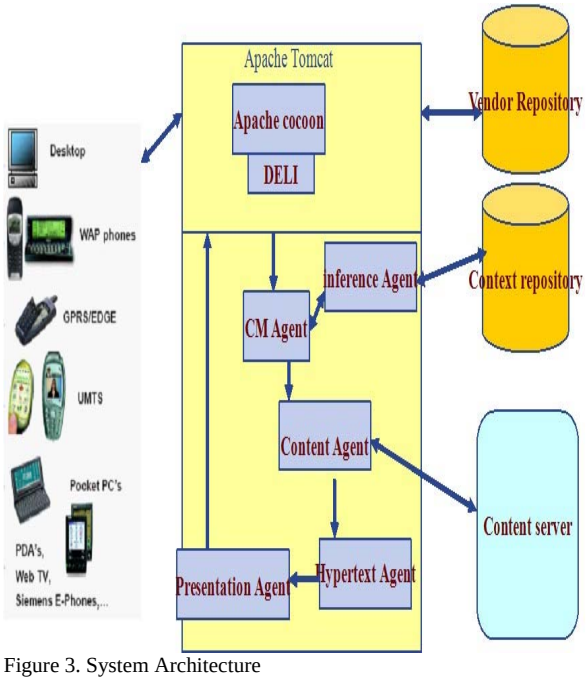

Figure 3. System Architecture

In this system, there are five main agents, as follows:

- Context-Management Agent (CMA): CMA is the system administrator. When a user send request to the server, the server forwards a request to CMA. The CMA can also cancel, modify, or renegotiate context because of environmental changes. The CMA also stores relevant information in a knowledge base repository for future inference and knowledge sharing.
- Inference Agent (IA): IA manages the inference process. It uses context captured from CMA and users as facts in the context inference process. It uses these facts to build a system knowledge base repository for reasoning new context information. The IA also reasons user preferences from user history.
- Content Agent (CA): CA selects the most appropriate content according to user context and user preferences.
- Hypertext Agent (HA): HA organizes the hypertext structure of the web interface. When the bandwidth is limited, it decomposes a large volume of content into linked pages.
- Presentation Agent (PA): PA builds an adequate layout for the web pages according to the layout capabilities of the device.

In this paper, CMA and IA are the main roles for context inference. When user sends a request to server, the server forwards the request to CMA. CMA negotiates the context and requests header. IA requests the context from context repository for reasoning new context information. The context-aware system adapts the behavior to user context without user intervention. Adaptation requires for sense changed in user context, inference about the change of user context, and they react appropriately.

IV. Content Adaptation

Adaptation means a process of selection, generation, or modification of content to suit the user's context. Each web page depending on its contents has a different layout that requires adaptation to match device capabilities. Before content adaptation, the system needs to first acquire user context information. Then, the system selects the most appropriate content according to the user context and user preferences.

A. Acquiring Context Information

Wherever adaptation takes place, it must be based on information about the user context. This can include the device's capabilities about the network characteristics, user preferences and other parameters, such as users' preferred language or their location.

In section 2, several strategies for acquiring user context were mentioned. The most popular method is to analyze the HTTP User-Agent parameter that comes with the HTTP request header, and map this parameter to a device or browser repository on the server side. However, the User-Agent header works only for nearly static device properties. Using the UAProf base on the CC/PP, the framework establishes a more effective mechanism for gathering dynamically changing context information on the server. However, UAProf only provides a common vocabulary for WAP devices, but most of the vocabulary can be adopted for other non-WAP devices like normal Web browsers on PCs, notebooks, or PDAs.

In this way, the mechanism illustrated in Figure 4 distinguishes between UAProf enabled devices, the devices provide the user agent and support for client side code like JavaScript, JScript and Java (combinations are possible). The client side code directly gathers the device properties of the client.

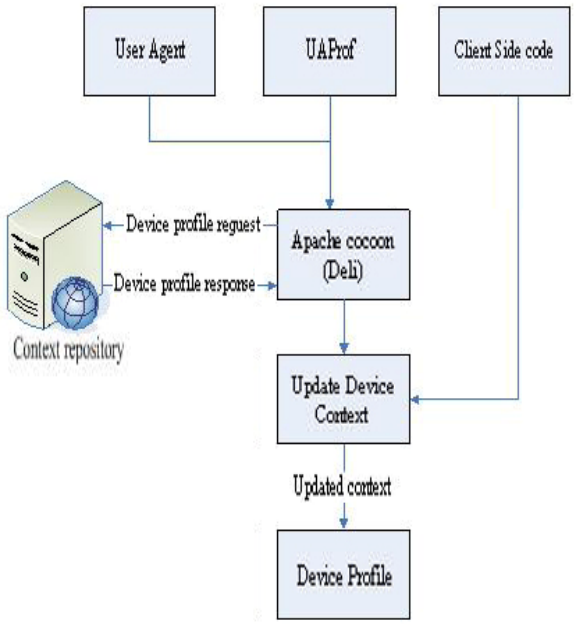

Figure 4. The Mechanism Of Acquiring User Context

Figure 5 illustrates flow chart of user context data. The process of the user context on the server depends on the obtained request.

- If the request only includes the User-Agent parameter, this parameter is mapped to the according device profile in a device repository
- If a UAProf enabled device sends a User-Agent profile within the request, the information is handled by DELI on the server side, which provides an API for Java servlets to determine client capabilities using CC/PP and UAProf.
- When the devices don't support UAProf, this system collects the context of the devices via client side code. The gathered context information on the client is sent within the HTTP request header. The server processes that information and merges it with an existing profile or generated by a DELI device.
- When the client sends the request to the server, the request includes user context information. The HTTP agent forwards the request to the Ontology agent. Because companies responsible for authoring profiles are often different to those creating CC/PP processors, there may be

disagreements when interoperability problems occur. The Ontology agent validates CC/PP profiles.

- The CMA then acquires user context data and negotiates with the context repository. After the CMA acquires context information, it modifies the profile and forwards it to IA. The IA uses the information to build a system knowledge base repository for reasoning new context information. The IA also reasons user preferences from the user history.

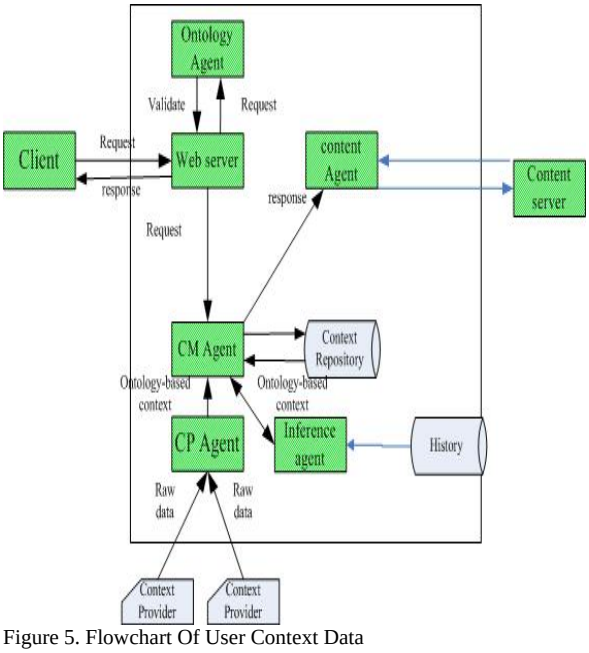

Figure 5. Flowchart Of User Context Data

B. Building SWRL Rules

SWRL rules can be directly tied together with Ontology using Ontology's description of relationships and words. Due to this advantage of SWRL [11], the correlations between classes don't need extra rule descriptions, and so this research used SWRL to develop rules.

The following two correlations need to be described when developing the rules:

- Correlations within the same domain.
- Correlations between different domains.

For example, when determining whether a device supports full color images or not, we can describe as follows:

SWRL rule: device(?d) ∧ ColorCapable(?d,?boolean) ∧ hasImageCapable(?d,?boolean) ∧ SWRLb:equal(?boolean, "Yes") → hasColorfulImg(?d, "Yes")

C. Executing Jess Inference and Updating the OWL Knowledge Base

Using rules in conjunction with ontologies is a major challenge for the semantic web. We first achieved an implementation of SWRL using the Protégé OWL plugin. We used the Protégé OWL plugin for editing OWL ontologies and SWRL rules. Racer is for reasoning with OWL ontologies, and the Jess inference engine is for reasoning with SWRL rules. In this paper, we build ontologies and rules by Protégé [12]. Protégé is a free Java-based, open source ontology editor and knowledge-base framework. Protégé is extensible and provides a plug-and-play environment that makes it a flexible base for rapid prototyping and application development. Figure 6 illustrates how to build an individual and his class. To bridge between Protégé SWRL and Jess, we used the Protégé plugin JessTab, allowing us to integrate Protégé and Jess.

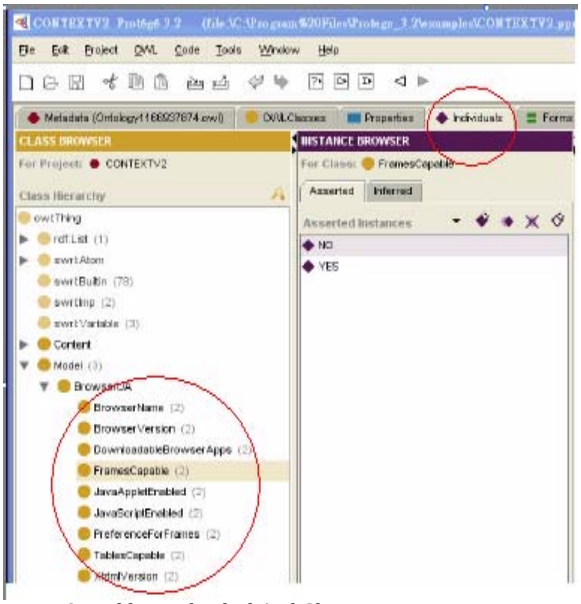

Figure 6. Building Individual And Classes

Figure 7 illustrates the inference architecture for IA. The main tasks here can be broken down into the following four steps:

1. Representing OWL Concepts as the JESS Fact Base.
2. Representing SWRL Rules as Jess Rules.
3. Combining the Fact Base and Rules Base to execute JESS Inference.
4. Executing Jess Rules and updating the OWL Knowledge Base according to inference results.

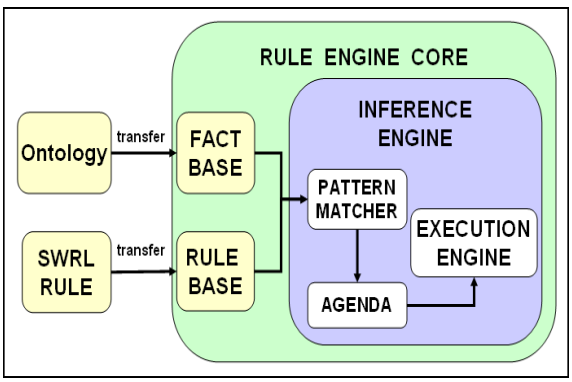

Figure 7. Inference Architecture

The Inference Engine acquires the data from Rule Base and Fact Base, through the Pattern Matcher to do comparison. If there is compliance with the rules and conditions, the Agenda will arrange the priority of the rules and conditions. Eventually, we will trigger the events of these rules by the Execution module.

- Rule Base: It stores the relevant inference rules, to present the knowledge and experience.
- Fact Base: it stores the relevant facts of reasoning process.

The Jess template provides a mechanism for representing the OWL class hierarchy. A Jess hierarchy can be used to model an OWL class hierarchy using a Jess slot.

- Define Jess template to represent the owl:Thing class:
  - *(deftemplate OWLThing (slot name))*
- A class "device" so that a direct subclass of owl: Thing could then be represented as follows in Jess:
  - *(deftemplate device extends OWLThing)*
- The OWL individual can be asserted as a member of the class Model :
  - *(assert(device(name Nokia7610)))*
- The property of the individual Nokia7610:
  - *(assert (colorCapable Nokia7610 Yes))*

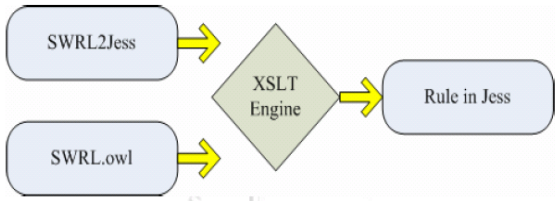

Figure 8 Presenting SWRL Rules As A Jess Rule

Figure 8 illustrates how to present a SWRL Rule as a Jess Rule. The following SWRL atom indicates that the Boolean variable must be Yes or No:

[Yes,No](?boolean)

For example, the following SWRL rule determines if a device supports full color images. It reasons new context information from existing context information.

device(?d) ∧ ColorCapable(?d,?boolean) ∧ hasImageCapable(?d,?boolean) ∧ SWRLb:equal(?boolean, "Yes") → hasColorfulImg(?d, "Yes")

The above rule can be represented in Jess as follows:

(defrule aRule (device(name ?d)) (ColorCapable ?d "Yes") (hasImageCapable ?d "Yes") => (assert (hasColorfulImg ?d "Yes"))

D. Providing Appropriate Content to Users Based on Inference Results

When a CA acquires the reasoned context from CMA, it requests the content based on user context. It selects the most appropriate content according to user context and user preferences. The CA decides which content will be adapted. The video adaptation may be removal, substitute, video framerate (resolution) reduction, or format conversion.

The adapted content is forwarded to the HA. The HA defines the hypertext structure according to user context by introducing links, and decomposes large pages to take into account the display limitations of the device. Then, the PA builds an adequate layout for the web pages.

V. CONCLUSION

In this paper, first we proposed an inference mechanism for context-aware service. Through this inference mechanism, users using different devices can get appropriate content based on inference results. Second, we can demonstrate the correlation between classes and individuals and provide better scalability by means of building ontologies. Lastly, SWRL depends on ontology based rule language. Rules written based on SWRL can directly use an established object relationship from ontology.

How to build the content recommendation according to user context is the future work. And a better content decomposing is necessary by Hypertext Agent to ensure the content sizes won't exceed device capabilities.

REFERENCES

[1] Y. W. Kao, T. H. Kao, C. Y. Tsai, S. M. Yuan, "A personal web page tailoring toolkit for mobile devices", Computer Standards & Interfaces, Vol. 31, Issue 2, pp. 437-453.

[2] P. Salomoni, S. Mirri, S. Ferretti, M. Roccetti, "A multimedia broker to support accessible and mobile learning through learning objects adaptation", ACM Transactions on Internet Technology (TOIT), vol.8, Issue 2, pp. 9:1-9:23.

[3] P. Brézillon, "Focusing on context in human-centered computing", IEEE Intelligent Systems, vol. 18, pp. 62-66, 2003.

[4] B. Schilit, N. Adams, R.Want, "Context-Aware Computing Applications", Proc. of IEEE Workshop on Mobile Computing Systems and Applications, Santa Cruz California, USA, 1994, pp. 85-90.

[5] C. Muldoon, G. O'Hare, D. Phelan, R. Strahan, R. Collier, "ACCESS: An Agent Architecture for Ubiquitous Service Delivery", Proc. of The Seventh International Workshop on Cooperative Information Agents. (CIA'2003), Helsinki, Finland, 2003, pp. 1-15.

[6] J. Sun, "Information Requirement Elicitation in Mobile Commerce", Communications of ACM, 46, 12, pp 45-47, December 2003.

[7] A. K. Dey, G. D. Abowd, "Towards a Better Understanding of Context and Context-awareness.", Proc. of the CHI'2000 Workshop on Context-Awareness, The Hague, Netherlands, April 2000, pp. 304-307.

[8] G. klyne, F. Renolds, C. Woodrow, H. Ohto, J. Hjelm, M.H. Butler, L. Tran, "Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies", W3C Working Draft (January 2004), http://www.w3.org/TR/2004/REC-CCPP-struct-vocab-20040115/

[9] M. Butler, "DELI: A Delivery context Library for CC/PP and UAProf", HP External Technical Report HPL-2001-260, 2002.

[10] M.O. Conner, H. Knublauch, T. Samson, M.Musen, "Writing Rules for the Semantic Web Using SWRL and Jess", 8th International Protégé Conference, Protégé with Rule Workshop, Madrid, Spain, SMI-2005-1079, 2005.
[11] V. Ricquebourg, D. Durand, D.Menga, B.Marhic, L. Delahoche, C. Loge, A.Jolly-Desodt, "Context Inferring in the Smart Home : An SWRL Approach", Advanced Information Networking and Applications Workshops, 2007, AINAW 2007. 21st International Conference. Volume 2, 21-23 May 2007, pp. 290-295.
[12] C. Golbreich, A. Imai, "Combining SWRL Rules and OWL Ontologies with Protégé OWL Plugin, Jess and RACER", 7th International Protégé Conference, Bethesda, MD, 2004, from http://protege.stanford.edu/conference/2004/abstracts/Golbreich.pdf.
[13] Jess Rule Engine: http://herzberg.ca.sandia.gov/jess/
[14] E. Friedman-Hill (2002). Jess in Action -Java Rule-based (In Action series) Systems. Manning Publications Co., from http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/1930110898
[15] N. Dimitris, A. Kanellopoulos, "Adaptive multimedia systems based on intelligent context management", International Journal of Adaptive and Innovative Systems, vol. 1, No. 1, pp. 30-43.
[16] Oracle Application Server, Oracle Corporation, from http://www.oracle.com/appserver/index.html
[17] IBM. WebSphere Transcoding Publisher, from http://www.ibm.com/software/webservers/transcoding/
[18] J. Hendler, T. Berners-Lee, E. Miller, "Integrating Applications on the Semantic Web", 2002 Journal of the Institute of Electrical Engineers of Japan, vol. 10, no. 122, pp. 676-690.
[19] M. C. Daconta, L. J. Obrst, K. T. Smith, "The Semantic Web: A Guild to the Future of XML, Web Services, and Knowledge Management", Wiley 2003, ISBN: 978-0-471-43257-9.
[20] M. P. Singh, M. N. Huhns, "Service-Oriented Computing: Semantics, Processes, Agents", Wiley 2005, ISBN-10: 0-470-09148-7.
[21] T. B. Passin, "Explorer's Guide to the Semantic Web", Manning 2004, ISBN:1-932394-20-6.
[22] D. L. McGuinness, F. V. Harmelen, "OWL Web Ontology Language Overview", W3C Recommendation, form http://www.w3.org/TR/owl-features.
[23] Y. Labrou, T. Finin, "As An Ontology-Using Yahoo! Categories to Describe Documents", 8th International Conference on Information and Knowledge Management", New York, USA, ACM Press, 1999, pp.180-187.
[24] S. Weibel, J. Gridby, E. Miller, "OCLC/NCSA Metadata Workshop Report", Dublin, EUA, Retrieved 1995, From http://www.oclc.org:5046/oclc/research/conferences/metadata/dublin_core_report.html.
[25] D. B. Lange, Mitsuru Oshima, "Programming and Deploying Java Mobile Agents with Aglet", Addison-Wesley 1998, ISBN:0-201-32582-9.
[26] J. Ferber, "Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence", Addison-Wesley 1999, ISBN:0-201-36048-9.

Jui-Fa Chen received his PhD, MS, and BS degrees in the department of Computer Science and Information Engineering from Tamkang University (TKU), Tamshui, Taipei, Taiwan, in 1998, 1992, and 1990, respectively.

He is an assistant professor in the Department of Computer Science and Information Engineering of Tamkang University (TKU). His research interests include intelligent avatar, wireless sensor network, and cyber physical system.

Ying-Hong Wang received the B.S. degree in Information Engineering from Chung-Yuan University, Taiwan, in 1986 and the M.S. and Ph.D. degrees in Information Engineering from the Tamkang University, in 1992 and 1996, respectively.

From 1988 to 1990, he worked in the Product Development Division of Institute of Information Industry (III). From 1992 to 1996, he was a lecturer in the department of computer science and information engineering of Tamkang University. From August 1st, 1996 to January 31, 2008, he was an associate professor in the department of computer science and information engineering of Tamkang University. From August First, 2004 to July 31, 2008, he served as the Chair of Department of Computer Science and Information Engineering, Tamkang University. Beginning February 1st, 2008, he promoted to be a professor of the department of computer science and information engineering, Tamkang University.

He has over 100 technological papers published on International journals and International conference proceedings, he also join many International activities been program committee, workshop chair, session chair and so on. He had been invited as Visiting Researcher in The University of Aizu, Japan, from January to March 2002. He is also an Adjunct Professor of Beijing Jiaotong University, China, starting from April 2007 and an Adjunct Professor of Shanghai University, China, starting from September 2010.

His current research interests are Software Engineering, e-Learning, Wireless Communication, and Mobile Agent.

Jingo Chenghorng Liao is now a doctoral student of Department of Computer Science and Information Engineering, Tamkang University since 2005. He received the M.B. degree in Department of Computer Science and Information Engineering from Tamkang University in 2004, and B.S. degree in Information Management of Chinese Culture University in 2003, respectively. He is a Senior IT Engineer at E-Lifemall Corporation in Taiwan since 2000. And he is an adjunct lecturer at Tamkang University since 2006.