Content Adaptation for Context-aware Service in Ubiquitous Computing Environment

Ying-Hong Wang
Department of Computer Science and Information Engineering
Tamkang University, Tamsui, Taipei 251, Taiwan
inhon@mail.tku.edu.tw

Chen-An Wang
Department of Computer Science and Information Engineering
Tamkang University, Tamsui, Taipei 251, Taiwan
chenan@mail.tku.edu.tw

Abstract:

The current trend is to access Web content and applications anytime, anywhere and on any device. Most Internet services and World Wide Web content has been designed with desktop computers, and often contain rich media, such as images, audio, and video. But The devices differ in network connectivity, processing power, storage, display size, and format handling capabilities. In many cases, this content is not suitable for the new (often mobile) devices. Therefore, content adaptation is needed in order to optimize the service for different devices and access methods.

This research discusses the way context is used for Web content adaptation. The CC/PP and UAProf are two related standards that define the format to describe the capabilities of the devices for accessing content. Context-aware environment must allows adaptive access to context information.

Keywords: context-aware, adaptation, SWRL

1. Introduction

Adaptation means a process of selection, generation or modification of content (text, images, audio and video) to suit to the user's computing environment and usage context. In general, normal web page with different media types. By using a PC connected to a Web site it's possible to see the original web page (without adaptation) with headings, photos, text, and video. 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 image depending on bandwidth and the capabilities of the device.

Adaptation can take place on the client, on the server or in an intermediate proxy. In client-based adaptation approach, a client needs to receive the same media encoding in different versions while only one will be used. Another problem is the fact that all computational overhead is shifted to the client.

In server-based adaptation, functionalities of web server are enhanced with content adaptation. Transmission times are reduced by delivering already adapted content. Traditionally, multiple variants of the same content are stored on the server and selected to match the client identification. One common way of providing content to different devices is to store the content as Extensible Markup Language (XML) and use Extensible Style Language Transformation (XSLT) to convert the content to appropriate makeup language.

In proxy-based adaptation approach, a proxy server analyzes and transcodes the content before sending the result to the client. Contents server and proxies need to know what kind of device is making the request in order to sending the right content.

Context is the information that characterizes the interactions between humans, applications and the surrounding environment [1]. Many researchers have tried defining context such as Schilit et al. who decompose context into three categories [2]: computing context, user context and physical context. Muldoon et al. define user context as an aggregation of his location, previous activities, and preferences [4]. Sun adopts the same definition and even adds physiological information to user context [5]. Huang et al. define context of an entity is a collection of semantic situation information that characterizes the entity's internal features or operation and external relations under a specific situation.

Context-aware Computing as mentioned by

Dey and Abowd refers to the ability of computing devices to detect and sense, interpret and respond to, aspects of a user's local environment and the computing devices themselves[6]. Context-aware content adaptation dynamically adapts their behavior to the user's current situation without user intervention.

In this paper, we address a content adaptation approach for the proposed context-aware architecture. The next section offers a survey of related researches about context-aware. Section 3 describes the context-aware system architecture. In section 4,content adaptation is presented. Finally, conclusion and future work is addressed.

2. Related Work

2.1. Methods of Client Capabilities Recognition

In order to deliver suitable content for different devices and user profile, The Web Site needs to be able to differentiate between the different device capabilities. The following are two possible methods for providing device recognition.

2.1.1. HTTP Request header field and User Agent Detection

When Web clients send requests to Web servers, they identify themselves. Current Http/1.1 use 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 which MIME types are accepted by a browser. The other Accept header fields describe preference for character set, encoding and language. Information in HTTP headers is limited and hard to extend. It was designed for browser descriptions and it lacks means for context and device capability.

In Addition to Accept header, clients send a User-Agent header to identify themselves. It is very simple but powerful enough to provide some client-specific information. User-agent strings have been used to perform content adaptation since the early days of the Web. User-agent header contains information about the browser and operating system and sometimes hardware information. But User-Agent header works only for nearly static device properties. With hundreds of different browser existing today, it get tricky to support every one by relying on the User-Agent field.

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

Table1 are examples of the Request header fields produced by different browsers:

Table 1. Examples of User-Agent headers.

IE7 in Windows XP	Mozilla/4.0(compatible; MSIE 7.0; Windows NT 5.1; Mozilla/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)
Firefox3.0.1 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	Nokia6230/2.0 (03.06) Profile/MIDP-2.0 Configuration/CLDC-1.1
Nokia 6600 built-in browser	Nokia6600/1.0 (3.42.1) SymbianOS/7.0s Series60/2.0 Profile/MIDP- 2.0 Configuration/CLDC-1.0

It's important to note that neither of these browsers correctly obeys the HTTP/1.1 content negotiation standard.

2.1.2. Client Script or ActiveX application

Scripting languages such as JavaScript, VBScript, Jscript, and WMLScript can be used to provide device –spcific information. AcitveX components can also be written to report the browser device and connection characteristics.

2.2 The Standard of Context Extraction

Several standards have been defined which address the interoperability problem. A few of these standards are described here.

2.2.1 Composite Capability/Preferences profiles (CC/PP)

As the numbers of variety of devices connected to the Internet grows, there is corresponding in the need to deliver content tailored for the different devices. The User-Agent header information discussed earlier is not enough. Composite Capability/Preference

Profiles (CC/PP)[7] 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 provider. The servers can use this information describing he user preferences o customize the service or content provided. C/PP is an extensible framework that can be used for communicating the delivery context(screen size, audio capabilities, bandwidth, etc) from a device o a web server, resulting in the delivery of web content s usable on a given device. CC/PP allows different devices to specify their capabilities in a uniform way.

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

- Commercial implementations
  - Nokia CC/PP SDK
  - Intel CC/PP SDK
  - Aligo M-1 Mobile processing Server
- Non-Commercial implementations
  - DELI(HP laboratories Open Source CC/PP server API
  - DICE(University of Wales, berystweyth)
  - Panda and Skunk(Keio University)

2.2.2 User Agent Profile (UAProf)

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 it 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. User Agent Profiles consists of description blocks for the following key components:

- Hardware Platform: for example, 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 trend 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 later use. The WAP gateway or HTTP proxy must support UAProf header forwarding.

2.3. Semantic Rule language (SWRL)

Semantic Rule language (SWRL) based on a combination of the OWL DL and OWL Lite sublanguages of the OWL. Web OntoIoy Language with the Unary/Binary Datalog RuleML sublanguages of the Rule Markup Language. 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 arc of the form of an implication between an antecedent (body) and consequent (head)[8]

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

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

3.Context-Aware System Architecture

The context-aware system architecture is presented in this section. Figure 1 shows the proposed context-aware system architecture.

Figure 1. System architecture

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

- Context- management agent (CMA): The context management agent (CMA) is the system administrator. The CMA negotiates with context provider agents to acquire the required context. 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 inference and knowledge sharing.
- Inference Agent: The inference agent manages the inference process. It uses context captured from context providers 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 inference agent also reasons user preference from history of user.
- Content agent: The content agent selects the most appropriate content according to user context and user preferences
- Hypertext agent: Hypertext agent organizes the hypertext structure of the web interface. When the bandwidth is limited, it decomposes large content in linked pages.
- Presentation agent: The presentation agent builds an adequate layout for the web pages according to the layout capabilities of the device

3. Content adaptation

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

4.1 Acquiring context information

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

In section 2, several strategies had be mentioned for acquiring user context. 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, user-agent header works only for nearly static device properties. Using the UAProf basing on the CC/PP framework establishes a more effective mechanism for gathering dynamically changing context information on the server. However, UAProf only provided 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 illustrates in Figure 2 distinguishes between UAProf enabled devices, devices providing the user agent and devices giving support for client side code like JavaScript, Jscript, and Java(combinations are possible).The client side code directly gather device properties on the client.

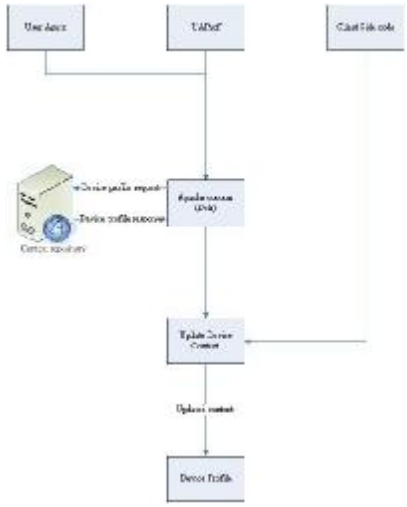

Figure 2. the mechanism of acquiring the user context

The processing 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, that information is handled by DELI [3] 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 collect 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 or by DELI generated device profile.

When the client sends the request to the server, the request includes user context information. Http agent forwards the request to ontology agent. Because companies responsible for authoring profiles are often different to those creating CC/PP processors, it is possible there

may be disgreements when interoperability problems occur. Ontology agent validates CC/PP profiles.

Then CM agent acquires user context and negotiates with context repository. After CM agent acquires context information, it modifies the profile and forward to inference agent. Inference agent uses the information to build a system knowledge base repository for reasoning new context information. The inference agent also reasons user preference from history of user

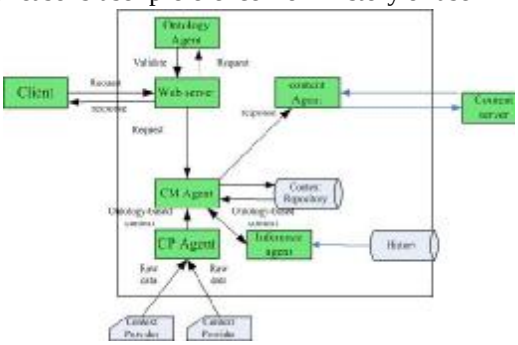

Figure 3. The flow of user context

4.2 Building SWRL Rules

SWRL rule can be tight together with Ontology and directly use Ontology's description of relationship and words. Because of this advantage of SWRL[9], so the correlations between Classes don't need extra rule descriptions. so this research uses SWRL to develop rules

The following 2 correlations need to be described when developing the rules:

- the correlations between the same domain
- the correlations between different domain

For example, determines if a device supports full color image:

SWRL rule:

device(?d)ΛColorCapable(?d,?boolean)
ΛhasImageCapable(?d,?boolean)
ΛSWRLb:equal(?boolean,"Yes")
→hasColorfulImg(?d, "Yes")

4.3Executing Jess inference and Updating OWL Knowledge Base

Using rule 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 use the protégé OWL plugin for editing OWL ontologies and SWRL rules, Racer for reasoning with OWL ontologies, the Jess inference engine for reasoning with SWRL rules. Figure 4 illustrates how to build individual and his class. To bridge between protégé SWRL and Jess ,we use the protégé plugin JessTab, allowing to integrate protégé and jess.

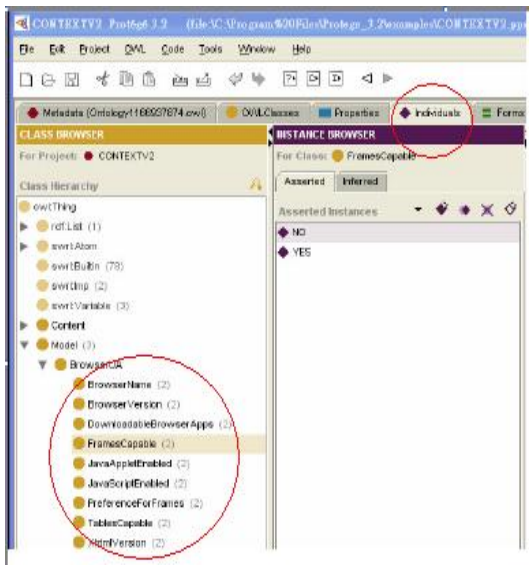

Figure 4. building individual and classes

Figure 5 illustrates the inference architecture. The step here can be broken down to the following four steps:

1. Representing OWL Concepts as the JESS Fact Base
2. Representing SWRL Rules as Jess Rules
3. Combine the Fact Base and Rule Base to execute JESS Inference
4. Executing Jess Rules and Update OWL Knowledge Base According inference results

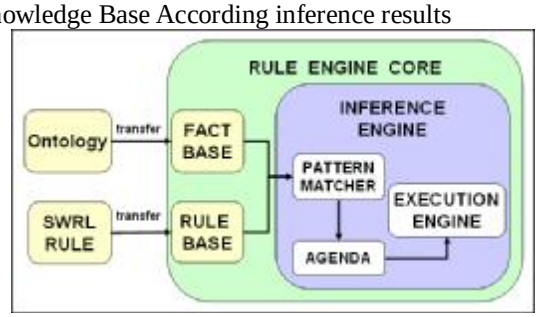

Figure 5. inference architecture

The Jess template provides a mechanism for representing 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 the owl:Thing class:
  - (deftemplate OWLThing (slot name))
- a class "device" 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))

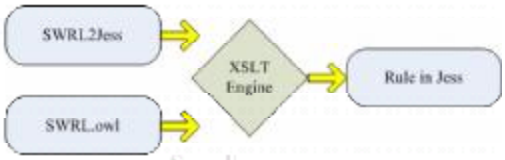

Figure 6 Presenting SWRL Rule as Jess Rule

Figure 6 illustrates how to present SWRL Rule as Jess Rule.the following SWRL atom indicates that variable boolean must be Yes or No:

- [Yes,No](?boolean)

For example, the following SWRL rule determines if a device supports full color image

- device(?d)Λ ColorCapable(?d,?boolean) ΛhasImageCapable(?d,?boolean) ΛSWRLb:equal(?boolean,"Yes") →hasColorfulImg(?d, "Yes")

The above rule can be representing in jess as following:

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

4.4 Provide appropriate content to users based on the Inference results

When content agent 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 content agent decides which content will be adapted. The video adaptation may is removal, substitute, video framerate (resolution) reduction, format conversion.

The adapted content is forwarded to Hypertext agent. Hypertext agent defined the hypertext structure according to use context by introducing links and decomposes large pages to take into account the display limitation of the device. Then presentation agent builds an adequate layout for the web pages.

5.Conclusion

First, this paper proposes a 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 individual and provides better scalability by means of building ontologies. Last, SWRL is based on ontology based rule language, Rules written based on SWRL can directly use established object relationship from ontology.

References

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

[2] B. Schilit, N. Adams, and R.Want, "Context-aware Computing Applications", in Proc. Of IEEE Workshop on Mobile Computing Systems and Applications, Santa Cruz California,USA,1994

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

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

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

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

[7] G. klyne, F. Renolds, C. Woodrow, H. Ohto, J. Hjelm, M.H. Butler and 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/

[8]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

[9]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 '07. 21st International Conference. Volume 2, 21-23 May 2007 Page(s)290 – 295,Digital Object Identifier 10.1109AINAW.2007.130