Adaptation of Web Contents and Services to Terminals Capabilities: the @Terminals Approach

Elisabetta Di Nitto^{1,2}, Giordano Sassaroli¹, Maurilio Zuccalà¹
¹CEFRIEL, Milan, Italy ²Politecnico di Milano, Milan, Italy
dinitto@elet.polimi.it, {sassaroli, zuccala}@cefriel.it

Abstract

Special purpose terminals able to be interconnected to the Internet and offering to users some inherently limited service fruition capabilities are nowadays emerging. This is fostering the need to support dynamic adaptation of contents and services in order to overcome terminals limitations. In this paper we present the @Terminals Adaptation System, an extensible infrastructure for universal access to Web contents and services. It mediates the interaction between user terminals and service providers through dynamic adaptation, integration, and selection of contents and services.

1 Introduction

Nowadays most people exploit the Internet to access various kinds of information and services, ranging from informative services (e.g., the opening hours and address of a department store), to search engines (e.g., yellow pages), to services supporting interaction with public administration (e.g., the tax payment service) and e-commerce.

The diffusion of special purpose terminals able to be interconnected to the Internet and offering to users some service fruition capabilities inherently limited by small screen sizes, missing multimedia capabilities, etc., is fostering the need to support adaptation of contents and services in order to overcome terminals limitations.

Through adaptation, service developers and multimedia content authors will have a way to provide a coherent and extensive access to the information they offer, i.e., to provide users with *universal access* to contents and services. Otherwise, the Web could undergo a further vertical segmentation, where specific services will be accessed only by specific terminals through specific networks, and similar services will be replicated and specialized several times in order to reach every user who is using a particular kind of terminal.

The solution that is currently being adopted to support multi-terminal access to some content requires that the

content itself is made available by service developers in multiple versions, each one being specifically designed for a certain kind of terminal. Such a solution is clearly inadequate for two main reasons. First, it requires the various versions of the content to be kept in synch during service maintenance, second, it requires that the terminals exploited to interact with the service are known in advance, when the various versions of the content are designed. Thus, this approach does not work properly, e.g., as soon as a new terminal comes into place.

A number of approaches are being proposed in the literature to deal with adaptation (see Section 5). Some of them work at design time, by supporting service developers in generating the various versions of contents [31]. While this choice presents several advantages in terms of performance of the resulting system, still it does not address dynamic variations of terminal capabilities, thus requiring new kinds of adaptations [28][29]. Dynamic changes in terminals characteristics can be determined by updates of hardware and software components, modifications in the environmental conditions in which they are used (e.g., in the car it could be required not to use visual outputs that could distract the driver), and in the preferences expressed by users (e.g., John does not like to visualize advertisement banners).

To address such issues, the approach being pursued by some research groups is based on dynamic adaptation of the content to the specific terminals being used. Our approach follows this philosophy and presents the distinctive aspects described below.

1. Dynamic adaptation depends on the characteristics of terminals, users, and networks. Such characteristics are collectively called *delivery context*.
2. Dynamic adaptation is performed by combining the results produced by various basic adapters (we call them *filters*). The selection of the filters to be used in each specific case and the order in which they are executed is decided at runtime.
3. In addition to basic content adaptation, the infrastructure we have designed allows services to annotate contents with proper *metadata* that allow the adaptation process to preserve the semantics of each content thus improving the quality of transformations. The infrastructure also provides an inspection

mechanism that enables services to know the current delivery context and perform direct adaptation without relying on the adaptation support provided by the infrastructure. This is particularly useful when, e.g., adaptation consists in selecting a proper GUI to be downloaded on the user terminal.

In this paper we present our approach and its current implementation, the @Terminals Adaptation System (@TAS). The remainder of the paper is organized as follows. Section 2 briefly introduces some usage scenarios. Section 3 describes the conceptual model of our infrastructure, systematizing terms and entities that belong to this domain. Section 4 presents an architectural vision of the @TAS, explaining its original aspects. Section 5 gives an overview of other relevant works proposed as a solution to some universal access issues. Section 6 discusses the results of our research activity.

2 Universal Access: Usage Scenarios

In order to better understand the issues we want to deal with, in the following we briefly present some possible usage scenarios and highlight the requirements that emerge from their analysis. More detailed descriptions of these scenarios can be found in [13].

Scenario 1. Monday morning, 10:00 a.m. Mr. Smith is about to leave from office, take his car and join an important work meeting starting at 11:30 a.m. Through the Web browser running on his desktop PC, he visits the Web site of a location mapping and routing service provider, and downloads a map indicating the quickest path to the destination city. Before turning off his PC, he synchronizes the map and the related information to the PDA he is going to bring with him. In the car, he exploits the map and the GPS device incorporated into his PDA as a guide to the final destination. Along the way to the meeting venue, Mr. Smith encounters an unexpected detour due to a car accident. So he takes his PDA and links it to his mobile phone, this way accessing the Web site of the maps and routes service provider, and requiring an alternative route. The new map and the related information are downloaded to his PDA so that he can continue his trip on the new route.

Scenario 2. Mrs. Wilbur is watching her favorite cooking programme on TV. She gets interested in the recipe being explained, so she decides to download the related information from the programme Web site. Via a remote control, she selects the URL on the Web browser provided by her set-top box. The right Web page is then displayed on the TV screen. She checks the ingredients list and realizes that some of them are missing. Therefore she clicks on the “send to mobile phone” icon on the toolbar placed at the top of the TV screen. The set-top box

controlling the TV downloads the current Web page, converts the HTML code into a text document, and sends it as a SMS message to Mrs. Wilbur’s mobile phone. Then she goes to the supermarket, reads the ingredients list on her mobile phone display, and purchases them. After coming back home, she connects to the recipe Web site by using her digital radio, and prepares the meal by listening to the radio transmitting the recipe description.

Scenario 3. John is keen on chess. He plays against his friends or his computer whenever he can. At home he connects to a free chess server using his tablet PC and starts playing using a 3D user interface. He can also communicate with his current opponent using microphone and earphones. At 6 p.m. he decides to go out and meet his friends. He suspends the game he is playing on the PC and turns on his multimedia mobile phone. Then John resumes the game from the mobile phone. The chess server recognizes that the terminal has changed, so it proposes to John the possibility to keep on playing using a lightweight 2D interface, integrated by a voice recognition system to submit moves just pronouncing them.

The scenarios we have presented highlight the need for an underlying adaptation infrastructure able to recognize the capabilities of users’ terminals, and, based on them, to apply a broad range of content transformations, from image resizing (the PDA would not properly visualize a map image that is easily manageable by a PC) to text-to-speech translation (so the recipe can be listened through the radio), to formatting removal and transformation, and so on. Transformations can also consist in the selection of the GUI that is more suitable to the characteristics of terminals.

3 Conceptual Model

In this section we present a conceptual model derived from the analysis of the above scenarios. This model introduces the basic concepts our adaptation infrastructure is based on.

A *service* is a combination of data and operations made available to *users* through the Internet. Services available on the Web are delivered by *service providers*. Such services usually generate multimedia *contents*, e.g., text, images, audio or video streams. End users access those services through *terminals*.

A terminal is a device having certain hardware and software characteristics. It is usually hooked up to a *network*. Terminals can greatly differ in their characteristics, since they usually fulfill different requirements, either technology or usage driven.

In general, terminals can be described in terms of the following characteristics:

Hardware attributes. They describe the physical

structure and the inner constraints of a device, e.g., the maximum screen resolution, the maximum colour depth, the amount of mass storage, the available memory, the CPU speed, or the kind of input and output peripherals.

Software attributes. They describe the software capabilities of the terminal as well as the programs installed on it, i.e., the operating system, the availability of specific applications (e.g., Web browser) and interpreters (e.g., Java Virtual Machine, Macromedia Flash, Quicktime, or Acrobat plug-ins), supported resources and document formats (e.g., HTML version, audio and video codecs).

Network attributes. They include parameters such as bandwidth, response time, faults management policies, or security support.

User preferences. They describe the user expectations about the way the content should be presented and, sometimes, about the content itself, e.g. preferences about the presence of banners, tables, or frames, as well as the subjects the user is most interested in (e.g., sports, finance, music).

All these characteristics represent the user's *delivery context*, i.e., the set of parameters conditioning the way the user perceives information through his or her terminal.

In order to be properly presented to end users, Web

contents and services must be adapted according to the current delivery context. By *content adaptation* we mean the manipulation of the structure of a Web resource, like a Web page or an image, the selection of its content, or the modification of the nature or the form of the content itself and its related resources. These selection and modification activities often deal with deciding whether specific content elements should be kept or discarded and, if they need to be adapted, what kind of adaptation should be performed, e.g., to which extent to resize an image, without losing the message it transmits. Content adaptation is carried out by software elements called *information filters* that are able to perform basic actions on the content or on its subparts. The overall adaptation of a Web resource is the result of basic filtering actions performed on each single content element, according to a proper filter plan.

Contents can be enriched by *annotations* [18], i.e., textual additions to the original content that are provided by the content author. Annotations are designed to support content adaptation, providing additional information about the structure of the content elements and the way they should be manipulated and managed in order to properly adapt them to suit terminal characteristics and user preferences.

By *service adaptation* we mean the process of

Figure 1. The @Terminals conceptual model

modifying services – and their related contents – performed by service providers in order to match user preferences and terminal characteristics. In fact, information about the delivery context can be exploited by a service provider in order, e.g., to carry out preliminary content adaptation or to propose to the user a set of alternatives that are the most appropriate ones to meet his or her expectations in those specific usage conditions.

A *session* is a series of interactions maintained between the two communication end points during the span of a service fruition. A session characterizes the context that is shared by both the communication end points, i.e., the terminal and the accessed service or its content.

The way the user perceives an adapted service is therefore the result of the adaptation performed on all the content elements that the service is composed of, as well as the adaptation of the service itself, within a session.

Figure 1 summarizes the conceptual model our infrastructure refers to.

4 The @Terminals Adaptation System

The @Terminals Adaptation System – @TAS hereafter – is an extensible infrastructure for universal access to Web contents. The infrastructure mediates the interaction between user terminals and service providers and performs dynamic adaptation, integration, and selection of contents and services. Figure 2 depicts the major functional components of the @TAS infrastructure.

Figure 2. The @TAS architecture

The *Terminal Manager* component carries out the physical communication with terminals, recognizes terminals and captures the corresponding delivery context. The *Session Manager* component manages the interaction between users and services, keeping trace of the navigation sessions. This component is responsible for managing the change of terminal during a navigation session, allowing users to suspend the session and resume it on a different terminal. The *Service Adapter* component performs the adaptation activity, invoking the available

Filters for adapting the provided content to the current delivery context. Finally, the *Service Integrator* component implements a public interface that service providers can use for accessing the current delivery context, the current session of navigation, and data about the user.

In the reminder of this section we analyze the architecture of the @TAS in detail, explaining how its components handle the many aspects of the content and service adaptation process, and how its specific features make it an original and extendable solution to the universal access issues.

4.1 @TAS as a Web Intermediary

We designed and deployed the @TAS infrastructure as a Web intermediary [21], i.e., an entity acting as a mediator between clients and Web servers, that is physically and logically separated from both terminals and services.

Acting as an HTTP proxy [22], the infrastructure can monitor and manage the information streams flowing between the two end points of the communication channel, and performs two main activities:

- Request monitoring, pre-processing, and filtering to capture the delivery context, manage the interaction with terminals, and identify the requested services.
- Response adaptation and generation, to carry out the real adaptation activity on the contents fetched from service providers.

From the provider's point of view, the infrastructure is seen as an extended-capability client that is able to accept a wide set of contents (that it then adapts to the target terminal) and to provide a rich set of information on the context in which services are exploited. Basing on this information, service providers can perform preliminary content selection and adaptation.

The intermediary approach leads to a low degree of coupling between terminals and service providers, reducing mutual dependencies and constraints on both the actors.

We used WBI [1][11] to develop and test the first prototypes of @TAS. WBI is a Java-based programmable Web intermediary platform developed by IBM. @TAS components, including filters, have been developed as WBI plug-ins that are dynamically activated by the WBI infrastructure when specified conditions on incoming requests and generated responses occur. This approach has led to the development of a highly modular, scalable and flexible solution. The main advantage of this solution is the possibility to easily extend the @TAS platform by dynamically adding new filters.

4.2 Delivery Context Awareness

The main peculiarity of the @TAS is to exploit delivery contexts to drive adaptation decisions and content selection.

Delivery context cannot be statically defined: during a single navigation session the user can change his or her preferences, the terminal can update its capabilities because of a new installed plug-in or a variation in the environmental conditions, or the network bandwidth may suddenly decrease because of increased traffic.

The @TAS manages delivery context information through proper *profiles* that describe the user, the terminal, and the network. The component in charge of managing the interaction with terminals in order to obtain their profiles is the Terminal Manager.

To define profiles we exploit well-known standards proposed by the W3C [33] and the WAP Forum (now Open Mobile Alliance [20]) for describing and exchanging terminal capabilities: the CC/PP framework [7][8] and the UAProf vocabulary [30]. CC/PP, which is based on RDF [19][24], provides a structure to represent generic profile attributes that can be easily serialized in the form of XML documents. UAProf adds the vocabulary, defining which attributes compose the profile, their type, and their meaning. Through these attributes, terminals can be fully described in all the aspects we described in Section 3 Conceptual Model, providing information about software environment, hardware equipment and network capabilities. Both standards provide extension facilities [2], allowing the specification of new attributes and new structure elements, e.g., to seize the characteristics of a new-born terminal.

During user navigation, the terminal interacts with the Terminal Manager communicating either its entire profile, or a reference to an external source where the profile can be fetched, or also a subset of its attributes in the case the complete profile has been already communicated and some changes have occurred. This mechanism is realized by adding proper attributes to HTTP requests generated by the terminal, as suggested by the CC/PP Exchange Protocol [10][9].

The Terminal Manager implements a fully compliant CC/PP processor for parsing, exchanging, and accessing profiles, thus making them available to the other components of the @TAS platform.

This way, adaptation components and service providers deal only with abstractions of terminals, users, and networks: whatever terminal is currently used, the place the user is accessing the service, or the network used to access it, the related data can be accessed through profiles.

4.3 Dynamic Content Adaptation

Content adaptation performed by the @TAS is dynamic: decisions about the way the content should be adapted to suit the delivery context are taken only when the content is requested, because only in that moment the delivery context from which the request has been originated is known.

The infrastructure can perform generic adaptations on standard Web contents (e.g., image resizing to fit limited screen sizes, elimination of specific kind of not supported formatting). In addition, it can perform more sophisticated adaptations in the case the structure of contents and the role of contents elements are described by content authors through proper annotations. E.g., unsupported tables could be ignored if they have a non-important role, or their elements could be displayed in a proper list in the opposite case.

In general, annotation-based adaptation can help in maintaining the original purpose of the content. Annotations, also called metadata, are RDF descriptions associated to the content by internal references. We classify them as follows:

- *Structural annotations.* They provide information about physical characteristics of the document (e.g., document format, target display size) or a single element (e.g., image format, size, and color depth).
- *Logical annotations.* They provide information about the role of an element within the document (e.g., an image can play the role of main content or just of border or bullet) and its importance.
- *Alternatives.* They are a set of references of other resources that could effectively substitute a content element (e.g., an image instead of a video, a text description instead of an image). Alternatives have associated additional metadata that express their level of *semantic fidelity* to the substituted resource. E.g., a movie trailer could have associated two alternatives elements: the corresponding poster or a textual description of the plot, where the semantic fidelity associated to the plot could be higher than the one associated to the poster.
- *Adaptation constraints.* The content author may require some elements of a document not to be manipulated by the platform, in order to assure a minimum level of fidelity to the original content.

Content adaptation is the result of the composition and execution of filters, i.e., lightweight components that are able to operate on specific structural or informational content elements.

Filters are treated by the @TAS infrastructure as “black-box” operators implementing a well known interface fully described by their profile. *Filter profiles* are

XML descriptions that specify the type of transformation the filter is able to perform, the input it expects, the output it produces, and pre and post conditions to filter activation. These profiles allow the infrastructure to automatically recognize filters once installed, and understand which adaptation activities they are able to perform. In other terms, by means of filter profiles, the adaptation capabilities of the @TAS are fully extensible, so to support existing delivery contexts as well as future ones.

The Service Adapter component is in charge of managing the installed filters and executing the adaptation process. When receiving a content request, the Service Adapter analyzes the current delivery context, the content itself, and, if it exists, the associated description. Elements that do not fit the delivery context are identified and a filter plan is built as a sequence of filters, properly parameterized to fulfill all the current requirements and constraints. This activity is performed by a proper *document filter* that identifies all possible adaptation alternatives for each content element. These include the application of proper filters, the substitution of the element with another one declared in the correspondent metadata, and the removal of the element. Then, from the identified alternatives, one is selected based on some heuristic rules that we do not present for the sake of space.

The set of the adaptations selected to transform all content elements identifies the filters to be used in the transformation. Such filters are organized in a sequence (the filter plan) according to the preconditions specified in the filter profiles. Filters performing adaptations on different content elements can be executed in any order. The order is important in case more than one filter should be applied on the same resource.

The most critical issue in the definition of the filter plan concerns the selection of the heuristic rules that support the identification of content adaptations when various alternatives are available. Based on an analysis of a limited number of examples, we have identified some simple rules that need to be further tested and verified by applying them to a significant number of case studies. Other rules can be identified as well and codified in alternative document filters. These can be easily installed in the Service Adapter and replace the existing ones.

As an example, Figure 3 shows the results of the adaptation performed on a simple annotated Web page. The left hand side shows the Web page as seen using Microsoft Internet Explorer browser and a 800 pixel screen width. The right hand side shows the same Web page as seen on the Nokia Mobile Internet Toolkit WAP emulator, when a 120 pixel screen width is specified. In the Nokia emulator the image was removed because the related logical annotation indicated that its importance was low. The table was eliminated and the contained elements

presented as free text, because the related structural annotation indicated that the table was used just to arrange the layout of the elements on the page.

Figure 3. Adaptation performed by the @TAS on a single Web page

4.4 Service Adaptation

The Service Integrator component exposes proper APIs through which providers – that of course must be aware of the existence of the @TAS and the way to access it – can access data about users (e.g., user identity, user preferences, and navigation history), terminals, and networks of the current active usage session, together with other statistical data calculated and stored by the infrastructure.

E.g., when receiving a request for a specified service, providers can query the @TAS and switch from a high quality and interactive user interface, designed for full featured PCs, to a lightweight interface, suitable for small displays and limited bandwidth connections.

The Service Integrator is available as a SOAP Web service to assure a full compatibility with any server-side platform of service providers, and to enable an easy-to-integrate access to any valuable information that the @TAS can offer.

4.5 Changing terminal during a service session

A *navigation session* is a set of interactions between a user and a service. A session starts when the first request for the service is generated by the user terminal, and finishes when the fruition of the service completes, i.e., when the user decides that a navigation session can be terminated. Through the Session Manager component, the infrastructure is able to monitor and record the navigation

activity of the user during the current session. More precisely, the @Terminals infrastructure:

- Records the history of the user navigation, keeping trace of visited contents and services.
- Caches resources and visited documents, enhancing the response time during navigation.
- Manages the status of the provided service, e.g., storing user cookies on the Web intermediary on behalf of the user's terminal.

These activities are usually carried out by terminals. Delegating this responsibility to the Web intermediary is the basis for supporting terminal switching during a navigation session.

When the user moves to another terminal, he or she is provided with an easy-to-use Web interface allowing him or her to freeze the current session, control the list of all the suspended sessions, resume a certain session, or simply navigate the personal history. By selecting an item from the navigation history, the corresponding session is resumed and the related contents, readapted according to the new delivery context, are offered to the user.

5 Related Work

A significant effort has been spent in the last few years on providing universal access to Web contents, and many different solutions have been proposed. Among the others that support dynamic adaptation we cite the Espial Escape Web browser [6]. It is a device-specific Web browser that is executed directly on the user terminal and carries out basic resources adaptations, such as rescaling of images or shrinking of documents. A similar approach can also be found in the Device Mosaic Web browser by OpenTV [4].

Other projects take a Web intermediary approach that is more similar to the @TAS. Spyglass Prism [25] is one of these. It can compress (JPEG) images and change the size and color depth of images. It can also perform some HTML filtering and modification, like removal of Java and JavaScript, and conversion of tables to lists.

WebSphere Transcoding Publisher (WTP) by IBM [32] takes into account the delivery context through the *User-Agent* attribute. Such an attribute is part of any HTTP request and identifies the browser that has originated it. The value of this parameter is statically defined and does not provide any information about the real software and hardware capabilities of the terminal (on the contrary, CC/PP profiles can provide such information). Adaptation decisions are based on a mapping, defined by the service provider, between the *User-Agent* attribute and a set of available filters. Filtering activity on documents can be driven by annotations of the original HTML documents [3]. Annotations specify

elements to be discarded, and policies to split pages for small size displays. Differently from the @TAS, such annotations do not provide any other structural information, such as color depth, nor more sophisticated logical annotations and constraints on content elements.

The iMASH platform [12][27] is quite similar to the @TAS since it captures the delivery context through device management and registration: terminal and network capabilities are described through proper profiles. The architecture provides also session management facilities to support terminals change. External metadata provide information about content structure and resource characteristics (e.g., image size), but do not provide any further information about the role of content elements, their importance, or their semantics.

Another example of external annotations used to dynamically perform proxy-based content adaptation can be found in [17]. The proposed process is called "semantic transcoding" because through annotations it takes into account also the semantic content of Web documents, e.g., performing text and video summarization, or language translation. Besides user preferences this approach does not consider any other parameter describing the delivery context, e.g., terminal or network characteristics.

6 Conclusions

The @TAS solution proposes dynamic adaptation of the accessed content based on delivery context awareness, through user, terminal, and network profiling.

Thanks to filter profiling and dynamic filter planning, adaptation capabilities of the platform are fully extensible, allowing the development of new filters to support new terminals, new resource types, or new services.

One of the key features of the infrastructure is the use of metadata (annotations) to provide information about the structure and, to a certain extent, the semantics of contents. Most of existing contents do not provide metadata yet. Nonetheless, the infrastructure can properly manage existing contents because it is able to work in "best effort" mode, taking minimal decisions on how to adapt contents about which no information is available.

Session management facilities allow users to exploit a navigation session through different terminals. A change in the used terminal during the same session affects the way services and contents are offered to the user, but it does not affect their semantics.

We developed a Java-based prototype implementing the architecture presented in this paper. The prototype exploits the IBM WBI framework for the development of Web intermediary mechanisms and the Jena library [16] for RDF parsing, manipulation, and serialization. This prototype has demonstrated the practical feasibility of the

proposed approach, principally for what concerns the delivery context awareness, the dynamic adaptation of annotated Web contents, and the session management.

Future work can move towards a further extension of the platform, for supporting other protocols than the HTTP, in order to allow the adaptation of streaming content. This extension should be accompanied also by the integration of mechanisms to manage quality of service.

Another area of interest can be the adaptation and composition of Web services, for the dynamical provision of high value services.

Finally, higher quality adaptation and selection of Web contents could be achieved leveraging on emerging technologies and standards for content description, such as the semantic Web and ontologies.

7 Acknowledgements

We are indebted to Alessandro Giupponi, Francesco Cannistrà, and Sergio Polito for developing the first prototypes of the @TAS platform. We also thank the other members of the @Terminals consortium who joined us in several valuable discussions on the @TAS requirements and architecture.

This research has been partially supported by the Italian MIUR (Ministero dell'Istruzione, dell'Università e della Ricerca) within the European ITEA programme.

8 References

- [1] Alphaworks, <http://alphaworks.ibm.com>
- [2] D. Brickley, R.V. Guha. "RDF Vocabulary Description Language 1.0: RDF Schema", W3C Working Draft, 30 April 2002, <http://www.w3.org/TR/rdf-schema>
- [3] D. Raggett, A. Le Hors, I. Jacobs, "HTML 4.01 Specification", W3C Recommendation, 24 December 1999, <http://www.w3.org/TR/html401>
- [4] Device Mosaic Web Browser, <http://www.opentv.com/dm>
- [5] Espial DeviceServer, <http://www.espiat.com>
- [6] Espial Escape Web Browser, <http://www.espiat.com>
- [7] F. Reynolds, J. Hjelm, S. Dawkins, S. Singhal. "Composite Capability/Preference Profiles (CC/PP): A user side framework for content negotiation", W3C Note, 27 July 1999, <http://www.w3.org/TR/NOTE-CCPP>
- [8] G. Klyne, F. Reynolds, C. Woodrow, H. Ohto. "Composite Capabilities/Preference Profiles (CC/PP): Structure and Vocabularies", W3C Working Draft, 15 March 2001, <http://www.w3.org/TR/CCPP-struct-vocab>
- [9] H. Nielsen, P. Leach, S. Lawrence. "An HTTP Extension Framework", RFC 2774.
- [10] H. Ohto, J. Hjelm. "CC/PP exchange protocol based on HTTP Extension Framework", W3C Note, 24 June 1999, <http://www.w3.org/TR/NOTE-CCPPexchange>
- [11] IBM WBI, <http://www.almaden.ibm.com/cs/wbi>
- [12] iMASH Project, <http://pcl.cs.ucla.edu/projects/imash>
- [13] ITEA @Terminals, "WP2 – Needs and Requirements", Public Deliverable, 5 October 2001, <http://www.esi.es/@Terminals>
- [14] J.R. Smith, R. Mohan, C-S. Li. "Transcoding Internet Content For Heterogeneous Client Devices", Proc. IEEE Int. Conf. on Circuits and Syst. (ISCAS), May 1998.
- [15] James Clark, "XSL Transformations (XSLT)", W3C Recommendation, November 1999, <http://www.w3.org/TR/xslt>
- [16] Jena Semantic Web Toolkit, <http://www.hpl.hp.com/semweb/jena-top.html>
- [17] K. Nagao, "Semantic Transcoding: Making the World Wide Web More Understandable and Usable with External Annotations", 1999.
- [18] M. Hori, R. Mohan, H. Maruyama, S. Singhal. "Annotation of Web Content for Transcoding", W3C Note, July 1999, <http://www.w3.org/TR/annot>
- [19] O. Lassila, R.R. Swick. "Resource Description Framework (RDF) Model and Syntax Specification", W3C Recommendation, 22 February 1999, <http://www.w3.org/TR/1999/REC-rdf-syntax-19990222>
- [20] Open Mobile Alliance, <http://www.openmobilealliance.org>
- [21] R. Barrett, P. Maglio. "Intermediaries: new places for producing and manipulating Web content", 1998, <http://www.almaden.ibm.com/cs/wbi/papers/www7/intermediaries.html>
- [22] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee. "Hypertext Transfer Protocol -- HTTP/1.1.", RFC 2616.
- [23] R. Mohan, J. R. Smith, C-S. Li. "Adapting Multimedia Internet Content for Universal Access", IEEE Transactions on Multimedia, Vol. 1, No. 1, March 1999.
- [24] Resource Description Framework (RDF), <http://www.w3.org/RDF>
- [25] Spyglass Prism - OpenTV Educational Services, http://www.opentv.com/support/ed_services/spyglass_prism.html
- [26] T. Kamada. "Compact HTML for Small Information Appliances (cHTML)", W3C note, 9 February 1998, <http://www.w3.org/TR/1998/NOTE-compactHTML-19980209>
- [27] T. Phan, G. Zorpas, R. Bagrodia. "An Extensible and Scalable Content Adaptation Pipeline Architecture to Support Heterogeneous Clients", 22nd International Conference on Distributed Computing Systems (ICDCS 2002), July 2002.
- [28] W. Chisholm, G. Vanderheiden, I. Jakobs. "Web Content Accessibility Guidelines 1.0", W3C Recommendation, May 1999, <http://www.w3.org/TR/WAI-WEBCONTENT>
- [29] W3C Working Group on Device Independence Activity, <http://www.w3.org/2001/di>
- [30] WAP Forum. "WAG UAProf - User Agent Profiling Specification", October 2001, <http://www1.wapforum.org/tech/documents/WAP-248-UAProf-20011020-a.pdf>
- [31] WebML, <http://www.webml.org>
- [32] WebSphere Transcoding Publisher, <http://www-3.ibm.com/software/webservers/transcoding>
- [33] World Wide Web Consortium, <http://www.w3.org>