A Web Service Approach to Universal Accessibility in Collaboration Services

Sangmi Lee[1,2], Sunghoon Ko[1], Geoffrey Fox [1,3], Kangseok Kim[1,3], Sangyoon Oh[1,3]
Community Grid Computing Labs, Indiana University [1]
Department of Computer Science, Florida State University[2]
Department of Computer Science, Indiana University [3]
slee@csit.fsu.edu, {suko, gcf}@indiana.edu, {kakim, ohsangy}@cs.indiana.edu

Abstract

The enormous growth in wireless communications and miniaturized handheld devices in the last few years, have given rise to a vast range of new services, for heterogeneous user environments. In this paper, we investigate the nature of designing distributed services that accommodate pervasive devices. We introduce Universal Accessible Web service architecture, *CAROUSEL Web service*. We present our collaborative Web services model linked with an event brokering service, NaradaBrokering. We discuss how a rich synchronous and asynchronous collaboration environment can support virtual communications while being built on top of a Web service infrastructure, based on industry standard portal technologies such as XML, Apache's Jetspeed, and WSDL. The linkage of the event brokering system with a Web service based architecture is one of the critical design issues in message-based collaborative application. We also describe the approach to universal access mechanisms in our collaborative Web service model. Also included is a discussion of the prototype collaborative SVG (Scalable Vector Graphics) browser that we developed as a Web service.

Key words: Web Services, Universal Access, Collaboration, Pervasive Computing, Event service

1 Introduction

The portability of new miniaturized devices, together with their ability to connect conveniently to networks in different places, makes *mobile computing* possible. Mobile computing is the performance of computing tasks while the user is on the move, or visiting places other than its usual environment [16]. Similarly, *Universal access* refers to the capability that all users are able to access information systems independent of their access device and their physical capabilities. Recent advances that have led to increasing bandwidths, for wireless communications, enable mobile users to utilize a vast range of sophisticated services such as real-time multimedia services.

The Web service concept allows objects to be distributed across web sites where clients can access them via the Internet. Moreover the use of XML [14] and standard interfaces like WSDL (Web Services Definition Language) [2] gives the interoperability between services, including Grids [23], which compromise robust largely asynchronous shared resources. The current Web service infrastructures such as Apache project's JetSpeed [4] or WebSpheres [24] from IBM provide services and development tools for mobile devices. However, these approaches based on portal services are not efficient to support real time services such as synchronous *collaborative services*.

Collaborative service provides the capability for geographically distributed users to share resources and work together on a specific problem. At a high level, collaboration involves sharing and in our context this is the sharing of Web services, objects or resources. We can expect that Web service interfaces to "many software packages" will be available and we will take this point of view below; where MS-Word, a Web Page, a computer visualization or the audio-video (at say 30 frames per second) from some video-conferencing system will all be viewed as objects or resources with a known Web service interface.

With the emergence of various portable devices and advances in wireless network communications, current collaborative systems require universal accessibility. We address the universal access problem for PDAs (Personal Digital Assistants) and generalize it to universal collaboration – the capability of multiple users to link together over the web with disparate access modes. Mobile systems are typically slow, unreliable, and have unpredictable temporal characteristics. Further, the user interface is clearly limited. The design of distributed mobile applications needs to identify the practicalities, reliability, and possibilities of continuous interaction and integrate synchronous and asynchronous collaboration. One of the steps to enable universal accessibility is by intelligently defining user "profile" and the semantic of Web services. We will describe how collaboration and universal access can be incorporated in the proposed collaboration Web service architecture.

Current multi-functional collaboration services require a scalable framework to adapt new Web services and third-party Web applications. The research pertaining to scalability will be focused on how we define needed functionalities as Web services, with ports providing interoperability between these services. Local or remote Web services are integrated into portals as *portlets*, which are *user-facing*, interactive web application components. A Web service is started with the exchange of *messages* between the service provider and the requestor. WSDL defines the exchange of messages between the service provider and the requestor as an *operation* [2]. The messages in operations are described abstractly, and

bound to a concrete network protocol and message format. The operation is defined with an input or output port as a minimal description. Our research extends negotiation capabilities on output ports to support diverse dynamic client requirements for universal access.

This system will be an advanced research prototype of the "sharing input/output port" [1] collaboration model. To satisfy the requirement of sophisticated collaboration, the collaborative system should consider sharing resources in various ways. The resources here include many kinds of objects such as databases, data streams or visualizations. Our approach originates from the refinement of this data processing. Each object is processed in a well-defined data flow called the "data-pipeline", which is designed to facilitate flexible data management. Each stage of the pipeline is a Web service with data flowing from one stage to another. We will be investigating how the modular data-pipeline can be defined, and how data-pipelines transmit information to each other for collaborative features in a pervasive environment.

The rest of this paper is organized as follows: Section 2 reviews other related works in the area of pervasive computing and collaboration systems. In section 3, we describe our collaboration model based on the object flow in Web service architecture. We will discuss the design of major components in the CAROUSEL Web service, and integrate these with messaging service in section 4. Section 5 presents the prototype of collaborative SVG [15] browser focusing on the aspect of universal accessibility. Section 6 will provide the conclusion and our future directions for the work discussed in this paper.

2 Related Works

Adapting new devices and emerging technologies innovatively to collaborative services are described in [5-7]. Rendezvous [8], GroupKit [9] and several groupware toolkits utilize the model-view so that services can generate different views for different users. However, since the situation is greatly simplified by using a centralized architecture, they were not able to support the diversity in devices accessing the service. One of the early efforts to adapt PDAs to work with conventional desktop computers is Carnegie Mellon University's Pittsburg Pebbles PDA Project [6]. Pebbles is designed to communicate with PDAs through the communication server, PebblesPC, and every message is conforms to the Pebbles protocol defined in their header files.

Transcoding is one of the most popular ways to tune content from the service provider. Transcoding is the transformation that converts the multimedia object from one form to another, frequently trading object fidelity for size [25], and is used to convert image or video formats (reduction resolution or compress data). However, it is also used to fit document and graphics files to the unique constraints of mobile devices and other Web-enabled products. A number of distributed services use transcoding technology to generate documents for their heterogeneous clients [25-29]. The Apache Cocoon [32] project allows automatic generation of HTML, PDF, and WML files through the processing of statically or dynamically generated XML files using XSL [33] and XSLT [11]. The idea of transcoding is also adapted to Web service architectures. Duke University's Quality Aware Transcoding project [30] investigates differentiated Web services, which enables the Web services and Web servers to manage their available bandwidth with its quality aware transcoding. IBM introduces WebSphere Everyplace Access [12], which lets developers and administrators build wireless applications conveniently. Transcoding technology is developed as a component of WebSphere, which is IBM's Web service infrastructure [24]. WebSphere Everyplace Access is designed to perform its transcoding process in their individual portlet. This approach is different from existing distributed systems supporting the transcoding process for heterogeneous devices, because it enables the transcoding process to be separated from the proxy architecture. We agree with WebSphere's approach which incorporates efficient network utilization strategies to avoid download large amount of contents into proxy server. It also enables content providers to provide high-quality transcoding technologies on the server side. We extend this idea by utilizing a powerful messaging service, which supports heterogeneous network communication protocols, NaradaBrokering [10]. This messaging service enables communications between transcoding portlets and other portlets without routing through the portal aggregator after the collaboration session is invoked once.

One of the most popular approaches to accommodate heterogeneous network communication technologies is to develop a *proxy-based* system for the new client network environment. Here the client network environment includes new communication hardwares, new network protocols, new security methods and so forth. A proxy is defined as a service representative that resides at the client's site [13]. It provides an interface to the service and manages the communication protocol with that service, thereby making distribution transparent to the client. Due to this transparency, proxy-based approaches have been adapted by many distributed systems [17]. To support seamless access, *context awareness* is required in the design of the messaging system supporting heterogeneous network environments. W3C initiated a working group called CC/PP (Composite Capabilities/Preferences Profiles) [19] to create a structured and universal format that allows a client device to tell an origin server or proxy about its profile. iMobile [17] from AT&T Wireless and the ICEBERG project [31] from UC-Berkeley are proxy-based platforms that act as a message gateway, while allow mobile devices using various protocols, on different access networks, to relay messages to each other.

NaradaBrokering is utilized as a message server in CAROUSEL Web service also focuses on the seamless access from the diverse user network environments. For mobile users, such as PDAs and smart phones, HHMP (HendHeld Message Protocol) [20] is integrated as a communication protocol within the NaradaBrokering transport framework. This would allow PDA's and other devices to interact directly with the messaging system; and with each other across firewalls, proxies and NAT boundaries.

3 Dataflow in Web services and Collaborative Services

Here we discuss the concept of dataflow and the collaborative service model utilized in the CAROUSEL Web service.

The Nature of Object Flow

With regards to an object in Web services, the object is typically in some pipeline, as seen in figure 1(a), from the original object to the eventual displayed user interface. Each stage of the pipeline is a Web service with data flowing from one stage to another. Rather than a simple pipeline, one can have a complex dynamic graph linking services together. We consider the output stage of this pipeline as a "document" – each with its own document object model; preferably different instances of the W3C DOM (Document Object Model). The final user interface could be a pure audio rendering for a visually challenged user, or a bitmap transmitted to a primitive client not able to perform full browser functions.

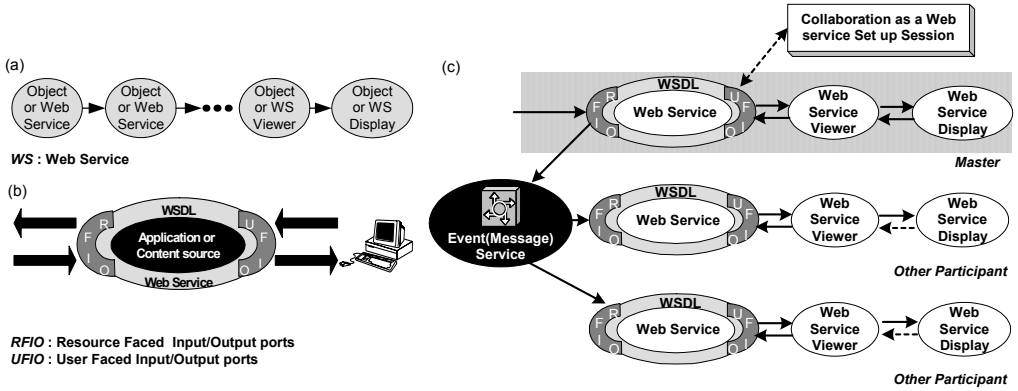


Figure 1 (a) Web service pipeline flow from originating objects to display
(b) Web services can have resource facing and user facing ports
(c) Shared Web Services using Input Ports (message)

Collaborative replicated Web services

In order to describe a general approach to collaboration, we need to assume that every Web service has one or more ports in each of the three classes shown in figure 1(b). The first class is *resource-facing input ports*, which supply the information needed to define the state of the Web service. *User-facing input port*(s), which allow control information to be passed by the user, may augment these resource-facing input ports. The final class is *user-facing output ports* that supply information needed to construct the user interface. Asynchronous collaboration can share the data (e.g. URL for a Web page or body of an email message) needed to define a Web service (display Web page or browse e-mail in examples).

Collaboration in the CAROUSEL Web service is approached based on the shared input port model, which we have also proposed as an object-sharing model in a Web service environment [1]. Clients on heterogeneous devices each get a different presentation view of the replica of each Web service. Moreover, each user will get non-collaborative services, such as individual scrolling or zooming image, customized based on their user profiles. The object sharing in our model of figure 1(c) is achieved by intercepting the pipeline before the "master" Web service and directing copies of the messages on each input port of the master Web service to the replicated copies. To support devices with limited capabilities, such as PDAs or smart phones, the most elaborate processing is implemented in the content server, and only ready-to-use data is delivered to users. However, each user can have their object instance with non-collaborative input/output ports, and every presentation view is customized with the user profile. Therefore we expect that the CAROUSEL Web service will inherit advantages of the shared input port model. Compared to the shared-output-port model, the shared-input-port model offers greater customizability as each client has the freedom to accept or reject data defining the shared Web service.

4 The Architecture of CAROUSEL Web Services

The CAROUSEL Web service contains four major components: content servers, aggregator, client application and event service. To provide rich synchronous/asynchronous collaboration features we build message based collaboration architecture. The set of cooperating services in our framework are deployed using the message service for communication. Each component is linked together with their input/output ports designed to support Web services semantics.

For demonstrative purposes, we have developed a *collaborative SVG browser*. SVG is a 2D vector graphics standard format from W3C and has a structured XML syntax [15]. We have adapted SVG as a format for shared export within our collaboration research. The user-interface will display an SVG image rendered by the content server, provide a display

customization environment and process basic collaborative functionalities.

Content servers

Each collaborative feature of the CAROUSEL Web service is designed as individual remote content servers in figure 2. The major requirements of content servers are,

- Content generation/processing for collaborative/non-collaborative features
- Ports facing to resources
- Customizing content processing for pervasive users

The requests from the user to the content server can be classified as collaborative and non-collaborative requests. The bitmap based user output can easily be managed on a given user interface, while the output rendered from filtered document should process every user command in rendering units for its quality of service. For instance, the SVG content server generates and delivers a new SVG image when a user requests zoomed image only for his or her interest but not for sharing. Otherwise, every user application must keep its own copy of original documents and the processing unit too. Every collaborative and non-collaborative event is transmitted via the NaradaBrokering messaging service.

The content server is designed as a portlet comprising Web services, and provides input and output ports. Each portlet in the CAROUSEL Web service defines the distributed objects in a XML-based IDL (Interface Definition Language) called WSDL [2]. The overall structure of every message, including input and output, are defined in WSDL. The output/input ports support dynamic communication channels to NaradaBrokering and general HTTP communication. In order to customize the portlet presentation for each user on pervasive devices, the content server allows any number of transformations. The stylesheets specially designed for universal devices map the original document to the customized one. Every customization is performed based on the user profile from the client.

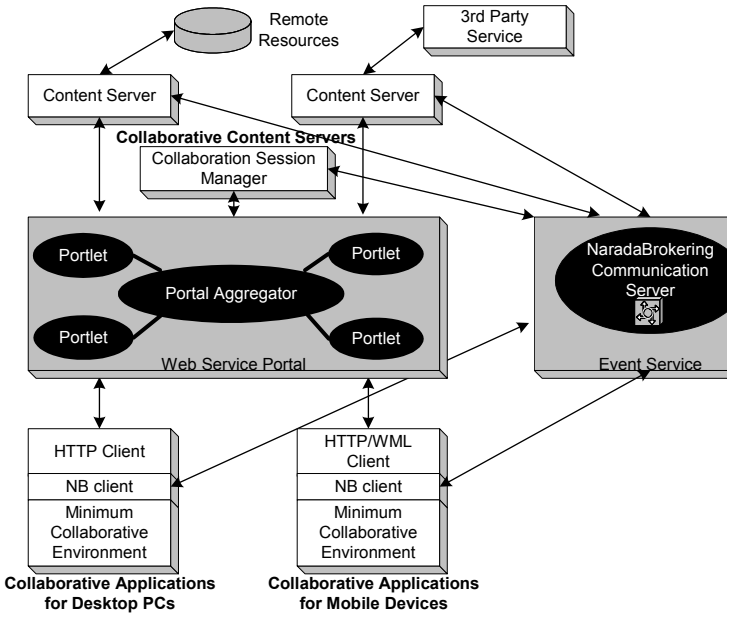


Figure 2 Architecture of CAROUSEL Web service

Event service

The CAROUSEL Web service is a message-based collaborative system. To provide messaging between the heterogeneous user network environments and Web services, NaradaBrokering [10] from the Community Grid Labs is adapted as a general event brokering system. NaradaBrokering supports centralized, distributed and peer-to-peer (P2P) messaging models with a dynamic collection of brokers supporting a generalized publish-subscribe mechanism. NaradaBrokering can operate either in a client-server mode like JMS (Java Message Service) [3] or in a completely distributed JXTA-like peer-to-peer mode [21]. By combining these two disparate models, NaradaBrokering can allow optimized performance-functionality trade-offs for different scenarios. At the transport level NaradaBrokering provides support for TCP, UDP, Multicast, SSL and RTP. For remote resources behind a firewall, NaradaBrokering provides the capability to communicate across firewall/proxy boundaries.

For mobile users, such as PDAs and smart phone, HHMP [20] will be integrated as a communication protocol within the NaradaBrokering transport framework. This would allow PDA's and other devices to interact directly with the messaging system; and with each other across firewalls, proxies and NAT boundaries. We expect that the collaborative system developed based on NaradaBrokering's messaging infrastructure will provide the collaboration service to the users in heterogeneous network environments with greater reliability within a scalable network framework.

Client application

The client application is designed to entail minimal data processing. The customized output data is delivered and displayed through the client application. The major features of the client application are; user specification, display portlet presentation, and processing user input events.

In the first step, pertaining to the user specification, every user can setup their working environment for their specific machines as well as preferences in the portal presentation view. The operating systems, display types, communication method and preferred resolution are the basic factors selected in this phase.

After the initial setup, the collaboration features customized with the user profiles specified in client's setup are provided. Collaborative events such as major presenter's zooming in on a SVG image or changing new URLs are wrapped with the collaborative application protocol and the event message is delivered to content servers via event services as an XML message. The non-collaborative events are delivered to content servers with the information which identifies the type of the event.

Aggregator

Several collaborative features designed as content servers and supporting services, are implemented as portlets and can be aggregated within a portal. The user sees the portal presentation view, which is the aggregation of the presentations of each portlet. The portlet presentation views can be in separate windows and can either be distinct or partially layered on top of each other. This portlet presentation view includes all kinds of machine-dependent outputs such as bitmap display or audio streams.

5 Universal Access in CAROUSEL Web service

Universal access in the CAROUSEL Web service is approached with the user output/input defined intelligently by an interaction between the user "profile" (specifying user and client capabilities and preferences) and the semantics of general Web services [4]. The service itself specifies the most important parts of its user-facing view and the output should be modified for clients with limited capabilities. This implies that the data processing in service is deficient in the sense that there must be a clear flow not only from the "basic Web services" to the user, but also back again. This can be quite complicated and it is not clear how this is achieved in general as the pipeline from Web services to user can include transformations, which are not reversible. For this reversibility problem, the content servers of CAROUSEL Web service are designed such that it keeps the original document in itself, and provides an interface to the original document to generate new output for each event. The ambiguity of reverse functionalities still exists, but we can expect that every reverse function will get correct output with this design.

In WSDL, the inputs and outputs of operations are termed as *ports*. The CAROUSEL Web service is designed with one or more ports in each Web service component to provide a general approach as collaborative application. Here we will discuss how we linked each Web service component via these input and output ports and approach the universal access in this object flow of Web services.

Each Web service is designed with three major user-facing ports as an output port of the modular pipeline of figure 3. First, there is the main user-facing specification output ports that in general do not deliver the information defining the display, but rather a menu that defines many possible views. A selector in figure 3 combines a user profile from the client (specified on a special profile port) with this menu to produce the "specification of actual user output" that is used by a *portal*, which aggregates many user interface components (from different Web services) into a single view. The result of the transformer may just be a handle, which points to a user facing customized output port. This output port allows users to select user interface components - operating systems, display types, and resolution preferences.

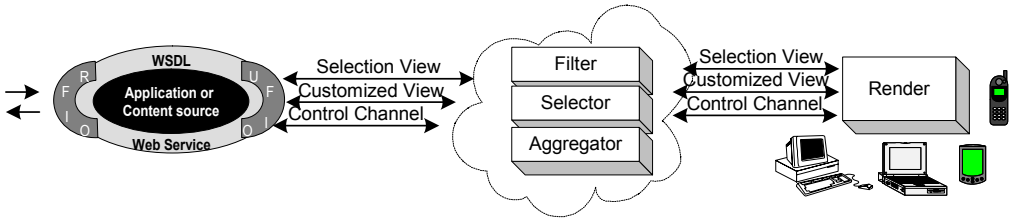


Figure 3. Data flow in CAROUSEL Web service to support Universal Access

Second, the customized user-facing output port that delivers the selected view from selector of the Web service to the client. This in general need not pass through the portal, as this only needs the specification of the interface and not the data defining the interface. For collaborative SVG, specification of the output port could involve a choice of display type or resolutions in figure 4 (a). Rendering an image from an SVG document transformed with CSS stylesheets [22] is a customized user–facing output presentation in figure 4 (b). The conversion between stylesheets could in fact involve a general filter capability of the event service as another Web filter service. It seems appropriate to consider interactions with user profiles and filters as outside the original Web service, since they can be defined as interacting with the service using a general logic valid for many originating Web services. Figure 5 is a snapshot of using our system in collaborative task.

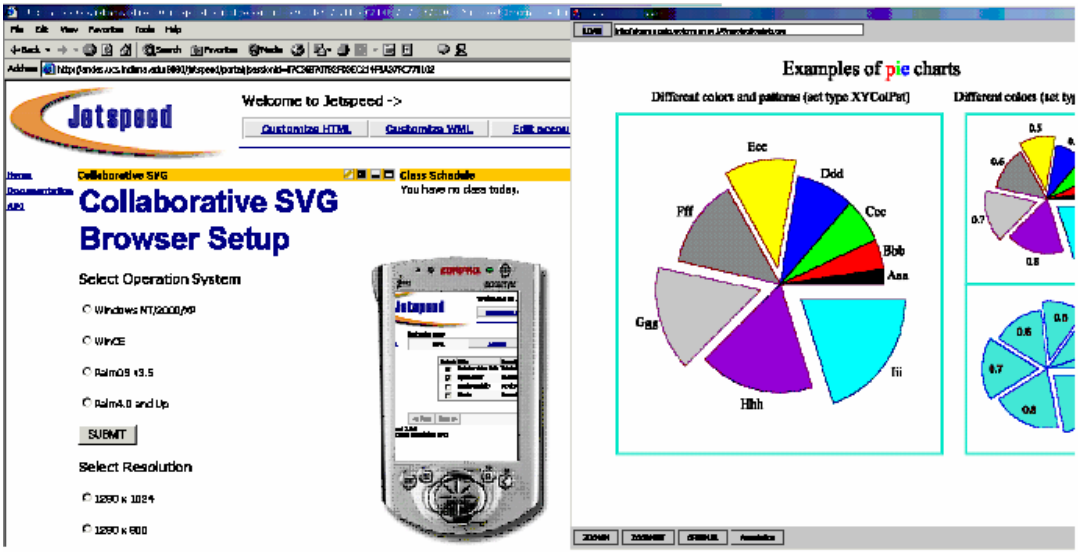
Finally, User-facing input/output port, which is the control channel, shown in figure 3.

Note that in figure 3, we have lumped a portal (such as Jetspeed [4] from Apache) as part of the "event service" as it provides a general service (aggregating user interface components) for all applications (Web services). This packaging may not be the most convenient, but architecturally, portals share features with workflow, filters and collaboration. These are services that operate on message streams produced by Web services. Considering universal access in this fashion could make it easier to provide better customizable interfaces and help those for whom the current display is unsuitable.

6 Conclusion and Future work

We introduced architecture of universal accessible Web service for collaborative applications. In summary, collaboration between users in the CAROUSEL Web service is based on the input port sharing model for more flexible collaboration. Each Web service in this architecture provides one or more ports facing to resources and users to implement an integrated collaboration application. CAROUSEL Web service is linked to a message based event service, and every message is defined with XML. We discussed object-flow pipeline in CAROUSEL Web service, and how we deploy it to facilitate the universal access.

For future work, we will integrate HHMP as a communication protocol within the NaradaBrokering transport framework. We expect that it will provide more seamless network communication to mobile users collaborating with users in traditional computers. We are also investigating more reliable communication methods for mobile devices accessing the CAROUSEL Web service. Integration of 3rd party services into the Web service architecture is another issue that meets classic investigation. It includes wrapping and integrating remote service from other venders, such as the XML Presence Protocol based instant messenger (XXMP) [18] from Jabber Open server. The research on the mobile communication protocols continues in parallel. With the remarkable improvements in mobile services, there are more possibilities to adapt new communication technologies for PDAs such as JXTA and JMS style event brokers. We expect that the investigation of these communication methods will provide better performance and quality of service.



(a) **(b)**

Figure 4. Snapshots of Collaborative SVG Browser
(a) User specification from client's setup
(b) User Interface with customized output display

Figure 5. Snapshot of Heterogeneous Collaboration

References

[1] Geoffrey Fox, Dennis Gannon, Sung-Hoon Ko, Sangmi Lee, Shrideep Pallickara, Marlon Pierce, Xiaohong Qiu, Xi Rao, Ahmet Uyar, Minjun Wang, Wenjun Wu Book chapter on Peer-to-Peer Grids http://grids.ucs.indiana.edu/ptliupages/publications/p2pgridbook.pdf
[2] Web Services Description Language (WSDL) version 1.1 http://www.w3.org/TR/wsdl
[3] Sun Micro Systems, Java Message Service http://java.sun.com/products/jms
[4] Jetspeed Portal from Apache http://jakarta.apache.org/jetspeed/site/index.html
[5] I, Marsic, "An Architecture for Heterogeneous Groupware Applications," Proc. of the 23rd International Conference on Software Engineering, pp.475-484, IEEE Computer Society Press, 2001.
[6] Brad A. Myers. "Using Hand-Held Devices and PCs Together," Communications of the ACM. Volume 44, Issue 11. November, 2001. pp. 34 - 41.
[7] Harakan Software, PalmVNC, http://www.btinternet.com/~harakan/PalmVNC/
[8] R.D. Hill, T. Brinck, S.L. Rohall, J.F. Patterson, and W. Wilner, "The Rendezvous architecture and language for constructing multiuser applications", ACM Transactions on Computer-Human Interaction, 1(2):81-125, June 1994.
[9] M. Roseman, and S. Greenberg, "Building real-time groupware with GrouKit, a groupware toolkit.", ACM Transactions on Computer-Human Interaction, 3(1):66-106, March 1996.
[10] G.C.Fox, and Pallickara, S., "The Narada Event Brokering system: Overview and Extensions", In Proc. of the 2002 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'02).http://grids.ucs.indiana.edu/users/shrideep/narada/
[11] W3C, "XSL Transformations (XSLT) Version 1.0", W3C Recommentation, Nov, 1999. http://www.w3.org/TR/xslt
[12] IBM Raleigh Lab., "Transcoding Technology in WebSphere Everyplace Access: Using Transcoding Technology to Expand your Pervasive Portal", IBM WebSphere Developer Technical Journal, Sept, 2002. http://www7b.boulder.ibm.com/wsdd/techjournal/0209_thrasher/thrasher.html
[13] Adaptability in CORBA: The Mobile Proxy Approach, Benjamin Aziz and Christian Jensen, International

Symposium on Distributed Objects and Applications, pp295-304, 2000
[14] W3C, Extensible Markup Language (XML), http://www.w3.org/XML/
[15] W3C, Scalable Vector Graphics (SVG), http://www.w3.org/Graphics/SVG/Overview.htm8
[16] G. Coulouris, J. Dollimore, and T. Kindberg, Distributed Systems Concepts and Design, Addison Wesley, Third Edition, 2001
[17] C. Rao, D, Chang, Y, Chen and M, Chen, "iMobile: A Proxy-Based Platform for Mobile Services", Wireless Mobile Internet, 2001, pp.3-10
[18] P. Saint-Andre, Jabber Server v1.2 Technical White Paper, Jabber.com, Inc., http://www.jabber.com/pdf/Jabber_Server_White_Paper.pdf
[19] W3C, CC/PP Working Group, http://www.w3.org/Mobile/CCPP/
[20] Geoffrey Fox, Sung-Hoon Ko, Kangseok Kim, Sangyoon Oh, Sangmi Lee, "Integration of Hand-Held Devices into Collaborative Environments "; proceedings of the 2002 International Conference on Internet Computing (IC-02) , June 24-27 Las Vegas. http://grids.ucs.indiana.edu/ptliupages/publications/PDA_IC2002.pdf
[21] Sun Micro Systems JXTA Peer to Peer technology. http://www.jxta.org
[22] Cascading Style Sheets, http://www.w3.org/Style/CSS
[23] The Grid Forum http://www.gridforum.org
[24] IBM, WebSpheres, http://www-3.ibm.com/software/info1/websphere/index.jsp
[25] S. Chandra, A. Gehani, C. Ellis and A. Vahdat, Transcoding Characteristics of Web Images, Multimedia Computing and Networking, Jan, 2001, San Jose, CA.
[26] Intel Inc., Intel quickweb, http://www-us-east.intel.com/quickweb/
[27] Oracle Inc., "Oracle portal-to-go; any service to any device", white paper, Oct,1999, http://www.oracle.com/mobile/portaltogo/
[28] IBM, "Websphere transcoding publisher.", 2001, http://www.ibm.com/software/webservers/transcoding.
[29]J.R. Smith, R. Mohan, and C.Li, "Scalable multimedia delivery for pervasive computing." In Proc. of ACM Multimedia, Oct. 1999, pp.131-140, ACM Multimedia, Orlando, Florida
[30] S. Chandra, C. Ellis and A. Vahdat, "Differentiated Multimedia Web Services using Quality Aware Transcoding", InfoCom, 9th Annual Joint Conference Of The {IEEE} Computer And Communications Societies, 2000
[31] H. Wang, B. Raman, C.Chuah, R. Biswas, R. Gummadi, B. Hohlt, X. Hong, E. Kiciman, Z. Mao, J. Shih, L. Subramanian, B. Zhao, A. Joseph, and R. Katz, ICEBERG: An Internet-core Network Architecture for Integrated Communications, IEEE Personal Communications, Special Issue on IP-based Mobile Telecommunication Networks, 2000
[32] The Apache XML project, COCOON, http://xml.apache.org/cocoon/
[33] W3C, The Extensible Stylesheet Language (XSL), http://www.w3.org/Style/XSL/