Platform for flexible integration of multimodal technologies into web application domain

[1]IZIDOR MLAKAR, [2]MATEJ ROJC
[1]Roboti c.s. d.o.o, [2]Faculty of Electrical Engineering and Computer Science, University of Maribor
[1]Tržaška cesta 23, [2]Smetanova ulica 17
SLOVENIA
[1]izidor.mlakar@revolutionary-robotics.com, [2]matej.rojc@uni-mb.si

Abstract: - Most users in either desktop or ubiquitous environments access Web applications from Web browser interfaces. Majority of standard Web applications are still based on GUIs and usually support user-machine interaction using traditional human-machine interfaces (e.g. mouse, keyboard). In order to make access to the Web content more natural and to improve user experience, advanced user interfaces, enabling additional modalities in human machine interaction, must be developed and provided. This paper presents a concept and proposes a multimodal web platform (MWP) used for flexible integration of advanced multimodal technologies into web applications. The multimodal web platform suggests a process of integration of web applications into multimodal framework, instead of traditional integration of multimodal interfaces into web application, and formation of platform independent multimodal interface. The proposed MWP platform was developed in order to provide multimodal interface for interactive kiosk setup and multimodal e-commerce application. The MWP platform is based on Apache Tomcat web server, Java Web technology as middleware environment and complex distributive infrastructure that takes care for providing multimodal services.

Key-Words: multimodality, multimodal interfaces, Apache Tomcat, Java Web, multimodal web platform

1 Introduction

In recent years physical infrastructure hosting internet connections has advanced and expanded to the point where uplink/downlink speeds no longer present an issue within network based technologies. In addition wireless networks are expanding and providing connectivity anywhere at any time. Parallel to the advances of the infrastructure, web services and applications have evolved and range from simple presentation pages to complex social networks, e-commerce, e-learning and other business applications (e.g. B2B, CMS etc.). Browser enabled devices are emerging and enabling accessing the web content virtually anywhere at any time. Since speech is the most basic and efficient way of communication, multimodal interfaces can further expand the convenience and accessibility of services as well as lower the complexity of traditional unimodal GUI based interfaces [1][2]. The disadvantage of using multimodality within web applications lies in the fact that the capabilities of multimodal web interfaces are limited to the capabilities of their underlying standard markup language [3]. Most advances in multimodal interfaces for web based services are nowadays driven by ubiquitous computing environments (pervasive computing) [4][5].

The representative environment for multimodal Web Applications is MONA [6][7]. The focus of MONA is oriented towards providing middleware for multimodal applications running on diverse mobile clients (e.g. PDA, Smart Phones, mobile phones etc.). The MONA architecture supports deployment of device independent multimodal applications providing both device adaptable GUI interface and speech enhanced input and output. To ensure flexible and uniform application concept across different device categories, a browser-based approach was chosen. The speech based data is transferred using either circuit - or packet-switched voice connection. Target languages are HTML, WML, VoiceXML and two proprietary multimodal formats combining VoiceXML and HTML or WML, respectively. Project TERESA [8] and ICARE [9] reach beyond web services. TERESA provides an authoring tool for development of multimodal interfaces in multi-device environments. It automatically produces combined XHMTL and VoiceXML multimodal web interfaces specific to a targeted multimodal platform. ICARE provides a component based approach for development of web multimodal interfaces. Microsoft and IBM provide development frameworks and products for integrating speech in web applications. SASDK (Microsoft Speech Application SDK) [10] is a development kit for adding speech interfaces to Microsoft ASP.NET Web applications. It encapsulates Speech Application Language Tags (SALT) functionality. Speech Controls extend HTML and XHTML with a small number of SALT elements and objects that add speech recognition input, audio, and text-to-speech playback, DTMF input, and telephone call control to an ASP.NET Web application. IBMs WebSphere Voice Toolkit [11]

includes all tools needed to develop, debug, and deploy voice based applications.

Most of presented frameworks are mostly directed towards ubiquitous computing and provide approaches for development of new multimodal web applications rather than enabling the empowering existing web applications with multimodal services and solutions. On the other hand the concept proposed in this paper does not specify or anyhow enforce the environment within the targeted web application developed. Similarly as in MONA, the proposed MWP platform also supports the idea of implementing web browser based solutions in either enhancement or integration form. Namely, the usage of web browser can enable device independent web applications in its true meaning, not only to support diversity of mobile devices, but also to include desktop computers, set-top boxes, and other devices featuring network or mobile communication capabilities.

The article is structured in the following way. Section 2 presents the architecture of multimodal web platform MWP into details. Section 3 describes wrapping process for integrating multimodal web platform and web applications together. In section 4, BQ and iPortal web applications supported by multimodal interfaces are presented. We conclude with research and development activities for the near future.

2 Multimodal Web Platform - MWP

The concept of universal multimodal web platform should flexibly solve the following problems. Let's have some web application on one hand and a system able to provide multimodal services on the other. How to make the web application enabled with multimodal interfaces in flexible way? Further, let's have 2 contextually different web applications providing different services to different users. What if those users use devices with significantly different characteristics (e.g. desktops, mobile phone, PDAs etc.)? The obvious solution would be either to redevelop those web applications and adapt them to the desired specifications or to design a wrapper that could create context dependent multimodal interfaces with no significant changes made to the existing web application. In the proposed concept we presume that the web application is W3C compliant, CSS enabled and that web browser can support java and java script. The presumptions should be fulfilled in order to enable the CSS and Jscript based automatic adaptation of the multimodal web interface to the device it is being rendered on.

Figure 1 presents the proposed concept on which the MWP platform is based on. The basic idea of the concept is to divide any multimodal based web application into four relatively independent objects. The provider of the multimodal services (first object) has to be implemented as distributed standalone system, providing various non-web based multimodal services (e.g. speech recognition, speech synthesis, gesture recognition, etc.) and has to be accessible remotely. The second object represents web application that is actually typical desktop web application (e.g. e-learning, e-business, information kiosk, etc.). Since web application itself is developed for desktop domain, we can assume that before mentioned assumptions regarding CSS and W3C compliance are fulfilled. The third object represents multimodal shell that wraps targeted web application with multimodal web user interface. Web browser interface represents the fourth object and can be described as a set of browser enabled devices with different characteristics.

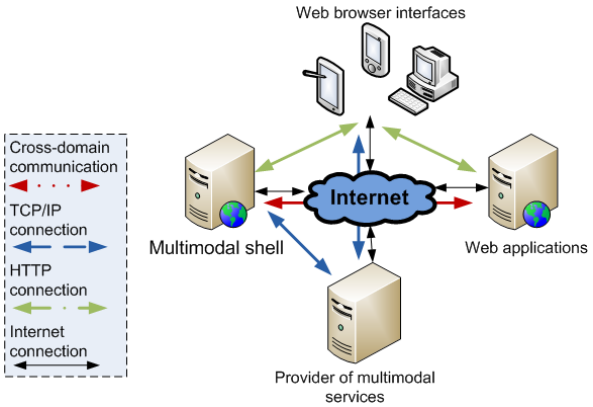

Figure 1: MWP platform concept.

All these objects are interconnected via Internet, either by network (LAN/WAN) or mobile based connections (e.g. GPRS, 3GP). The communication between multimodal shell and web applications (cross-domain communication) serves for java script cross-domain communication using fragment identifiers [12][13][14]. The TCP/IP based communication is used for audio/video data exchange (e.g. speech for recognition, synthesized speech etc.). The HTTP connection is used in web application related data exchange.

2.1 System architecture

In figure 2 MWP platform architecture is shown into more details. The idea has leaded us towards 2 layered multimodal wrapper implemented as device independent web application with solitary or joint database and with no visual interface elements. First layer, "Web platform interface", is part of multimodal wrapper middleware. This layer implements three types of communication sessions. The JDBC session is created to access relational database MySql. The supported database connection classes perform connections by using MySql-connector-java driver. HTTP sessions are used for

interaction between *multimodal user-interface API* and *wrapper and for data exchange between web browser interface and targeted web application*. Each HTTP session, in the context of multimodal shell, carries data such as: targeted web page current location/location changed, results of speech recognition, TTS content/content identification, HTML code of web targeted web application etc. The TCP/IP sessions are created for data transactions between multimodal shell and provider of multimodal services and are formed as follows: one dedicated session serves as management connection and each registered user-interface is granted one dedicated client session.

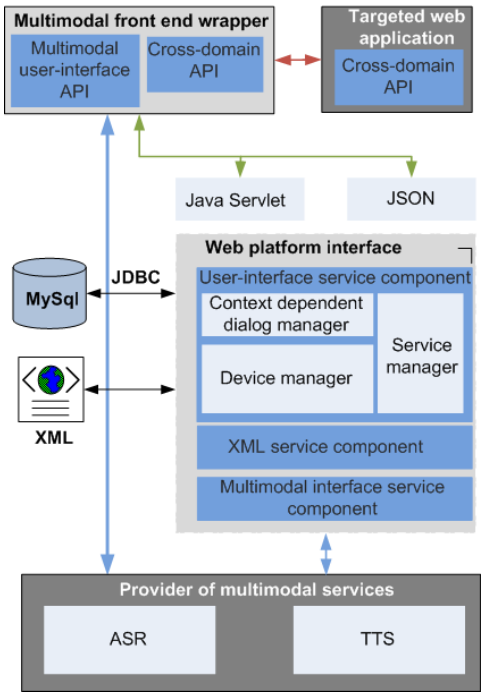

Figure 2: Architecture of the MWP platform.

User-interface service component represents the heart of the MWP platform. This component is implemented as multi-threaded Java bean class and supports serialization. The serialization aspect is used for direct access to java methods from user interface (web page front-end). The main purpose of this component is to serve different context dependent requests (e.g. user wants content to be read or user controls the targeted web application). *Context dependent dialog manager* and *service manager* are the main modules of the component. *Device manager* is used to manipulate and store web application relevant data about rendering devices (e.g. screen size, audio capture/play options, supported codec, browser/mini browser type and version, special control shortcuts, OS information, etc.). When *user interface* registers to the web application and its java applet provides proper device and service parameters, the *service manager and device managers* are populated with the appropriate data. Both are presented as java classes, containing descriptions and identifications of services and device properties. The service description is provided by *targeted web application* and by *provider of multimodal services* and is stored within service manager. The device description is provided by java API running on web application front-end and by device description entries (either in form of XML or database entries). It is stored within Device manager. The generated service list is also labeled with *service available* and *service allowed parameters* and contained in appropriate context dependent dialog manager. The *service available* parameter is deduced by device parameters (e.g. can device capture/play audio, can the device be controlled using tactile controls, etc.). The *service allowed parameter* is gathered from the service description entry (XML or database entry). When both *device manager* and *service manager* are prepared, the *context dependent manager* is initiated by fusing information, provided by both managers, and by adding the context information of the targeted web site (e.g. sitemap, actions, etc.). The *context dependent dialog manager* is implemented as light java based multi-threaded decision mechanism. Each connected user interface is granted its own *context dependent dialog manager* that handles requests and generates consequent device and targeted web application dependent procedures and responses.

XML service component is used as XML parser/generator. It is based on SAXParser and Java API for XML processing. The *XML service component* is implemented as global class and can be used by any multimodal wrapper process or targeted web application. The final component is called *multimodal service interface component*. As already mentioned, it is used for establishing TCP/IP connections between *multimodal wrapper* and *provider of multimodal services*. It is developed as java based TCP/IP socket listener, hosting management and interface (client) sessions. When interface device connects and registers to the system, a new communication session is generated and registered with the appropriate *context dependent dialog manager thread*. The threaded client session is reserved for the device, until either device unregisters properly or the user interface is closed (e.g. browser is closed or the browsing domain is changed). Upon closing all interface relevant data and both context dialog manager thread and threaded client session are stopped and properly destroyed. Additionally, *multimodal interface service component* provides necessary interfaces and methods for accessing non multimodal services. The behavior of these services is described within java beans that are part of (or can be included into) multimodal service interface software package. The architectural concept presumes

that services are made as global java classes, each service contained within its own java bean.

The second layer of multimodal wrapper, *multimodal front end wrapper*, is used as front end bridge between *provider of multimodal services* and *targeted web application*. During initialization process it is used as a process for gathering device information, application and service parameters. While the interface is running, the *multimodal front end wrapper* is used for processing (capturing, playing etc.) multimodal input and output. The *multimodal front-end wrapper* is basically a web page without visual elements. It contains client side procedures for multimodal communication and the *targeted web applicatio*n encapsulated in an IFRAME object.

The *user-interface multimodal API* implements its own TCP/IP connection to the provider of multimodal services. The session is used for both management and data transfer processes. The management processes are used for interface identification and registration, multimodal service registration and for gathering device parameters. Data transfer process, currently handles audio transfer (audio for speech recognition and generated synthesized speech to be played out within client interface), but can already be used for transferring also other multimodal related data (e.g. ECA – embodied conversational agents,, video feed for visioning, etc.). The *cross-domain API module* is used for java script based communication between documents of different domain. The cross-domain communication line presents a virtual interface that is used to pass variables between two documents of different origin (domain).

2.2 Provider of multimodal services

Distributed system, named DATA, serves as multimodal service provider within MWP platform. DATA system is implemented by using proprietary DATA framework. It is fully event-based and distributed framework, which is based completely on finite-state machine formalism. Therefore, it enables development of finite-state machine based distributed engines that can be used for flexible and efficient implementation of several multimodal service providers. DATA framework supports implementation of efficient and flexible data flow throughout the whole distributed system (consisting of control/configuration data, audio/video data, etc.) and behavior specification of DATA system's modules based on XML descriptions. Finite-state machine based DATA system's architecture enables implementing efficient event-based processes, capable to process and perform several tasks simultaneously. It's architecture uses also novel behavior specification mechanism by using JavaCC and UniMod frameworks. In the context of MWP platform, multimodal service provider DATA system currently hosts services such as: TTS and ASR.

3 Wrapping process

The initialization of MWP platform includes implementation of TCP/IP listener. Since the multimodal shell is java based, we decided to implement typical java based socket listener (socket server). Socket server is used to implement TCP/IP sessions, one management session and one communication session between MWP platform and DATA system for each connected device. Firstly, the management session is initiated. It implements simple hand shake protocol, used for exchanging basic system parameters (e.g. available multimodal services, hosted multimodal web applications, allowed devices, etc.). The data exchange is implemented in the form of XML packets over TCP/IP connection. Upon initiation of management session, multimodal shell is ready to serve as MWP platform. When user starts the targeted web application, he/she actually initiates the multimodal shell web interface. Within the context of multimodal shell, posting an URL invokes a web servlet that launches the initialization and registration process of user interface. The end result of this process is initialized *context dependent dialog manager* thread, registered for use with device web browser interface that launched the process. Upon dialog manager initiation, the servlet initiates java client named *user interface API*. *User interface API* is actually light java application that in its essence acts as TCP/IP client. The application is based on the DATA framework and takes the role of a client. So-called DATA client is an event-based finite-state machine, able to process random events and respond to them in a clear and flexible manner at very low processing power consumption. Within the context of MWP platform, DATA client application serves in registration of services within DATA system (what services are desired to be used by the user) and in device specification process (providing device parameters such as: screen resolution, browser type, etc.). It is also used to capture/present multimodal data (e.g. capture speech for recognition or play synthesized speech). When *user interface API* registers to the DATA system, DATA system notifies the multimodal shell of such event occurrence over the management session. Based on the data received, the multimodal shell will invoke new TCP/IP communication session between DATA system and itself, and register it to the appropriate *context dependent dialog manager* thread. The interface dialog manager of MWP is used to handle events that relate to multimodal domain (e.g. read text requests, speech control, etc.). The dialog manager itself is a multi-threaded java class containing service description, context description,

device description, and dedicated communication socket session to the DATA system. The dedicated session ensures non interrupted parallel communication of all device interfaces connected to the MWP platform.

4 Results and discussion

To evaluate the concept of MWP, multimodal web platform, two different multimodal web applications were implemented. Both multimodal web applications use DATA system as multimodal service provider. The difference between both web applications is mainly in the usage context and the procedure used to develop/deploy multimodal interfaces. The first multimodal web application *BQ-portal*, is used as information kiosk. The concept of MWP platform is used as the basis of multi-modal infrastructure, on which the multimodal web application is build (integration and development process). The second multimodal web application, named as *iPortal*, is used as multimodal e-store application. The concept of MWP platform was used as explained in the article. In the following subsections both web applications are presented into more details.

Figure 3: GUI of the multimodal web application BQ-portal.

4.1 BQ-portal

BQ portal is multimodal web-application, used as information kiosk. It was developed as a graphical user interface (figure 3), from which users can access and use web and non-web services supported by the system (e.g. RFID management, registration for practical sessions, control of intelligent environment, speech recognition service, speech synthesis etc.). MWP platform serves as infrastructure on which services and multimodal user interfaces are developed. Similarly, as multimodal shell of the MWP platform, the BQ-portal web application is developed under Java environment. BQ-portal provides several services: RSS feed reader, articles reading, flash based video player, IP camera control, web browsing, e-mail sending, etc. These services can be used either in tactile mode (by using touch screen and virtual keyboard), or by speech. The text based content displayed, can also be synthesized by using TTS service, supported by DATA system, and then played out by DATA client. Additionally, *Google translator API* for java is integrated. Therefore, selected text can be translated and then also synthesized.

Figure 4: GUI of multimodal web application iPortal.

4.2 iPortal

iPortal is a commercial multimodal e-business solution, and is used as a multimodal e-shop, supporting speech technologies (ASR and TTS). The targeted e-business web application (http://www.laptop-shop.si/) is written in PHP and is working as standalone web application. The context and user services for this targeted web application are well defined. User can perform actions like registration, purchase, and browse items. The page also provides standard commercial modules such as new products, featured products, etc. In addition back-end environment was developed, in order to support management and commercial activates (e.g. XML based advertising). The graphical user interface is presented in figure 4. In order to enhance targeted web application with multimodal services, the application was wrapped with MWP platform. Similarly as BQ-portal web application, iPortal also integrates speech recognition and synthesis, supported by DATA system.

The basic idea of wrapping procedure was to encapsulate content of web page with multimodal wrap and leave its original content and functionalities relatively intact. The concept of MWP platform enables the encapsulation in form of *IFRAME* and *cross domain* communication using proxy frames with fragment identifiers enables interaction between multimodal shell and e-business solution in real-time, directly on user front-end.

Within the context of ASR service, the possible recognition patterns and their meaning are contained in ASR service description. Speech recognition result can either result in direct link (1:1 mapping) or a formation of statement. In cases of statements, links are described as base links with additional GET parameters (attribute tags). Two types of content selected text (text within clipboard e.g. copied text) and tts text content can be given to the TTS engine, hosted within DATA system. When input content is of type tts text (text written contained in known DOM container), a TTS description entry has to contain the name of the location, link of the location and its DOM container. When TTS service is issued, a java script method will either grab the selected text form clipboard or read and forward the content of the DOM container. The content is then forwarded to the TTS service, hosted within DATA system.

5 Conclusion and future research

This paper presents a concept of providing multimodal services within several web applications. The proposed framework enables either integration or wrapping processes to be invoked when multimodality is to be merged with web applications. In both cases the presented concept suggests a network based approach where multimodal technology is provided as a service and rendering device processing is stripped to the minimal possible burden. The concept of MWP platform also takes into account differences between device capabilities and suggests a context manager to be formed that will be able to handle different device and web application contexts. To evaluate the proposed concept in real life, two web applications to be used in different contexts, were developed. We used enhancement and integration processes, both resulting in multimodal web application interface. Speech based human–computer interfaces (HCI) present only one part of multimodal interfaces. Our future plans include research into technologies such as visioning and biometrics. In addition, further research will include research in fields of context, context management and ability of automatic interface adaptation.

Acknowledgements

Operation part financed by the European Union, European Social Fund.

References:

[1] Shuchih Ernest Chang, Michael S. H. Heng, *An Empirical Study on Voice-Enabled Web Applications*, IEEE Pervasive Computing, vol. 5, no. 3, pp. 76-81, (July-Sept. 2006)

[2] Project P1104 *MUST – multimodal and multilingual services for small mobile terminals*, EURESCOM, (May 2002)

[3] Adrian Stanciulescu and Jean Vanderdonckt, *Design Options For Multimodal Web Applications*, Springer Netherlands, Computer-Aided Design Of User Interfaces V, (2007)

[4] Kisub Song, Kyong-Ho Lee, *Generating multimodal user interfaces for Web services*, Elsevier, Interacting with Computers archive, Volume 20, Issue 4-5, (September 2008)

[5] Shuchih Ernest Chang, Boris Minkin, *The implementation of a secure and pervasive multimodal Web system architecture*, Information and Software Technology Volume 48, Issue 6, (June 2006)

[6] Raimund Schatz, Rainer Simon, Hermann Anegg, Florian Wegscheider, Georg Niklfeld, *Developing Mobile Multimodal Applications*, HCI 2005 PROCEEDINGS VOLUME 2, (September 2005).

[7] Georg Niklfeld, Hermann Anegg, and others, *Device independent mobile multimodal user interfaces with the MONA Multimodal Presentation Server*, Proceedings of Eurescom Summit 2005, (April 2005)

[8] Berti, S., and Paternò, F., Migratory MultiModal Interfaces in MultiDevice Environments, in Proc. of 7th Int. Conf. on Multimodal Interfaces ICMI'2005 (October 2005), ACM Press, New York, 2005

[9] Bouchet, J., Nigay, L., Ganille, T., 2004. *ICARE software components for rapidly developing multimodal interfaces*, Proceedings of 6th International Conference on Multimodal Interfaces, pp. 251–258.

[10] http://msdn.microsoft.com/en-us/library/ms986944.aspx

[11] http://www-01.ibm.com/software/pervasive/tech/demos/tts.shtml

[12] Collin Jackson, Helen J. Wang, *Subspace: secure cross-domain communication for web mashups*, Proceedings of the 16th international conference on World Wide Web, ACM (2007)

[13] Frederik De Keukelaere, Sumeer Bhola, Michael Steiner and others, *SMash: secure component model for cross-domain mashups on unmodified browsers*, Proceedings of the 17th international conference on World Wide Web, ACM (2008) [14] Michael Mahemoff, *Cross-Domain Communication with IFrames*, http://softwareas.com/cross-domain-communication-with-iframes