Multi-modal e-Business

Rafah Hosn, Jan Kleindienst, Stéphane H. Maes, TV Raman, Jan Sedivy

Presenting Author: Stéphane H. Maes, smaes@us.ibm.com

IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, USA

Extended Abstract

1. Terminology

We first introduce some terminology. The term **channel** is used in this paper to denote a particular renderer, device, or a particular modality. **Multi-channel** applications are applications designed for ubiquitous access through different channels, one channel at a time. No particular attention is paid to synchronization or coordination across different channels. **Multi-modal** applications are multi-channel applications, where multiple channels are simultaneously available and synchronized. From a multi-channel point of view, multi-modality can be considered as another channel. **Conversational** applications are applications supporting multi-modal, free flow interactions (mixed initiative dialogs) within applications and across independently developed applications, and using short term and long term context (including previous input and output) to disambiguate and understand the user's intention. Typically, such application makes use of Natural Language Understanding (NLU) [9]. In the rest of this paper, we assume that multi-modal platforms will evolve to progressively encompass conversational capabilities as the underlying technologies improve.

It is also worth noting that the conventional understanding of multi-modality is changing over time. For example, it is now widely accepted that large screen graphical output, keyboard and mouse input constitute the GUI user interface while being clearly multi-modal. As platforms and tools are designed to seamless design applications that integrate these different modalities and applications become widely available, we do not distinguish them any more as multi-modal. They rather become a well understood and well accepted composite modality.

Note that there are no fundamental differences between multiple devices and multiple modalities.

2. Introduction

The computing world is evolving towards an era where billions of interconnected pervasive clients communicate with powerful information servers. The new decade will be characterized by the availability of multiple access channels that for the first time really enable ubiquitous access to information and e-business applications. The ability to access and transform information via a multiplicity of channels appliances and modalities, each designed to suit the user's specific usage environment, necessarily means that these interactions will exploit all available input and output modalities to maximize the band-width of man-machine communication.

There will be an increasingly strong demand for devices and browsers that present the same set of functionalities and behaviors when accessing and manipulating the information, independently of the access device. We postulate that the resulting uniform interface must be inherently multi-modal and dialog driven.

For this new (Mobile) Internet, new content and applications are developed with the intent of delivering it through many different channels with different characteristics. Therefore the content and applications must be adapted to each channel. Since new devices and content emerge continuously, this adaptation must be made to work for new devices not originally envisioned. In addition, it is important to be able to adapt existing content that may not have been created with this multi-channel or multi-modal deployment model in mind.

The increasing availability of information, along with the rise in the computational power available to each user to manipulate this information, brings with it a concomitant need to increase the band-width of man-machine communication; users will come to demand multi-modal interaction in order to maximize their interaction with information appliances in hands-free, eyes-free environments.

Realizing these dreams will require fundamental changes in the user interface and the supporting underlying platforms and browsers. Lacking this, users will be unable to truly access, act on, and transform information and applications any time, anywhere, independently of the access device, the environment or their current activities.

Recently multiple standard activities have been initiated to address related issues [3,4,5,6].

It is also important to note that multi-channel and multi-modal applications are by definition offering support for Web Accessibility [4].

3. Multi-channel and Multi-modal e-Business

During the course of 2000, the frontier of interactive e-business has started to shift. The desktop is no longer the only Internet access device: it is now possible to conduct interactive e-business from a range of access devices, including Internet phones, PDAs, telephone, TV, and so on. We have entered the era of **multi-channel** e-business.

Today, most multi-channel applications are actually multiple mono-channel applications that offer different services and functions and are typically running on different infrastructure, written and maintained by different IT organizations. It is only very recently that we observe an evolution towards truly multi-channel applications with a same application running on a same infrastructure, developed and maintained by a same IT organization and offering the same services, functions and behaviors is accessible through numerous channels. It becomes possible to offer web and wireless (e.g. WAP or iMode) access to similar services. With the introduction of VoiceXML [7], voice applications (call centers and CRMs) developed on IVR are now evolving towards the web programming model while numbers of web side start to offer telephony access.

We believe that the next frontier of interactive e-business will be **multi-modal**. In the multi-channel era, each access device typically supports just one mode of interaction, such as display/mouse, stylus, voice, and so on, or one mode at a time (for example, a smart phone). Multi-modal e-business is characterized by multiple interaction modes per device, for example voice + WAP, PC + tablet, PDA + phone, and so on, and the flexibility to let the interaction mode depend on the task and the situation - stationary or mobile, hands-on or hands-free, visual or eyes-free, and audible or quiet.

Although multi-channel applications enable access to information through any device, they are associated to significant pain points for the user:

- It is hard to enter and access data using small devices: keypads and screens are tiny.
    - Entering an e-mail address on a phone keypad becomes rapidly a very arduous task.
- Voice access is more prone to errors and voice output is inherently sequential
    - Repeated speech recognition error can prevent completion of a desired transaction and it usually result into user's frustration and even abandon of the service.
- One interaction mode does not suit all circumstances: each mode has its pros and cons
- One optimal interaction mode at a moment can no more be optimal at another moment or for another user.
- All-in-one devices are no panacea, and many different devices will co-exist.

In fact, no immediate relief is in sight for making multi-channel e-business easier:

- Devices are getting smaller, not larger
- Devices and applications are becoming more complex requiring more complex or efficient user interfaces
- Adding color, animation, streaming, etc. does not simplify the e-business issues mentioned above

These pain points directly affect the success of mobile e-business. While very popular gizmos, WAP and iMode phones are not yet widely used for e-business transactions. Their usage is still mostly limited to PIM, notification and very simple transactions and messaging. Similarly, voice access is still limited to domain specific applications.

Considering these factors leads to the conclusion that an improved user interface will accelerate the growth of mobile e-business.

On the other hand, the value proposition for multi-modal e-business solutions is to let users

- enter and access data easily using small mobile devices:talking is easier than typing and reading is faster than listening
- choose the interaction mode that suits the task and circumstances for input (key, touch, stylus, voice...) and output (display, tactile, audio...) possibility to easily correct an error etc..
- use several devices in combination take advantage of the unique capabilities of each device

Multi-modal interfaces make it possible to choose at any time the optimal interaction mode for each interaction between a user and an application. The choice of interaction mode can be made by the developer of the application, or it can be left up to the user.

By improving the user interaction, multi-modal interfaces can accelerate the growth of mobile e-business. Therefore, we expect that the mobile internet will lead to newly well accepted and understood composite user interfaces.

It is important to emphasize the importance of supporting seamless transitions in the user interaction amongst the different modalities available to the user -- that it be on one or across multiple devices. The user interfaces must always support dynamic and often unpredictable dynamic switches across modalities. Indeed, based on the user's activities and environment, the preferred modality may suddenly change. For example, a speech-driven (or speech and GUI) banking transaction will probably become GUI only if other people enter the room. Transactions that the user could not complete in his office are to be completed in voice only or voice only / GUI constrained mode in the car.

4. Presenting Unified Information Views – The Model View Controller Principle

The prerequisites outlined in the previous section necessarily postulate a Model, View, Controller (MVC) view of the world, where a single information source, the Model, is viewed via different views and manipulated via different controllers. The significant departure from the traditional MVC paradigm adopted by graphical user interface environments as implemented in SmallTalk [8] is that multiple controllers now interact with the application.

Indeed, in order to enable seamless switches between channels at any time, the dialog must be at any time in the same state in all the interacting views, that it be different devices or different modalities. The resulting MVC is explained in Figure 1.

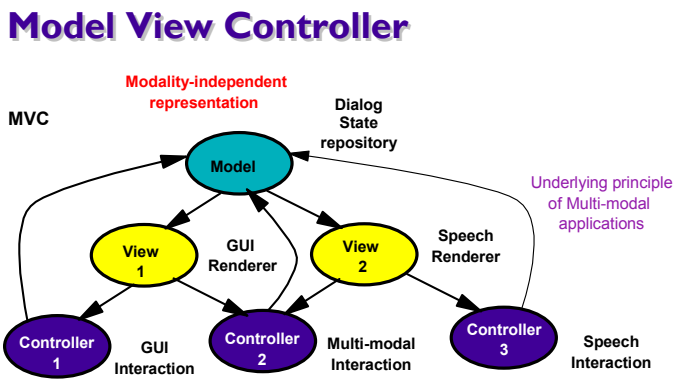

Figure 1 – Model View Controller (MVC): a single model is mapped to multiple views: the different synchronized channels. Multiple controllers act on and transform the underlying model by interacting via one or more views. A single model is transformed to multiple synchronous views. These transformations can be inverted to map specific portions of a given view to the underlying model. Let us denote T_i the mapping of the model to the view i and T_i^{-1} the inverted mapping. Composing T_i with T_j^{-1} for appropriate values of i,j enables us to synchronize amongst the various views.

A further consequence of the decision to embody multi-modal systems as collections of controllers all of which manipulate the same underlying model is to provide synchronized views, that the system be local (e.g. fat client) or distributed. This synchronization of views is a direct consequence of generating all views from a single unified representation that is continuously updated; the single modality-independent (channel-independent) representation provides the underpinnings for coordinating the various views.

To see this, consider each view as a transformation of the underlying modality-independent representation and consider that the modality-independent representation is described in XML (declarative case). It can be viewed as an abstract tree structure that is mapped to channel-specific presentational tree structures. These transformations provide a natural mapping amongst the various views ---since any portion of any given view can be mapped back to the generating portion of the underlying modality-independent representation, and this portion consequently mapped back to the corresponding view in a different modality by applying the appropriate transformation rules.

As a conclusion, multi-modal user interfaces should always follow the MVC paradigm. The existence of a modality independent representation of the applications enable implementation of the MVC, where the state of the application (tier 1 – dialog or presentation part) in that representation can be considered as the model of the MVC architecture. More specifically, there must always be a model of the interaction, independent of the rendering channel or modality, that is the repository of the current dialog state, the dialog flow as currently known by the application and the whole conversation history and context when context management is needed. Any user interactions within a modality must act on the conversation model before being reflected on the different views.

Other approaches may result in multi-modal dialog inconsistencies: forms completed in multiple modalities may be rejected as incomplete; page navigation in a modality may not reach the same dialog state in the other, etc.... These problem can be solved by appropriate authoring of the application. However, fixing these inconsistencies, without implementing MVC, requires overhead with numerous roundtrips to the server, multiple connections between the views or monolithic synchronization authoring; in the end, these solutions, once fixed, are weak version of MVC. Alternatively, non-MVC architectures can provide command and control level of capabilities, where a channel can be driven from another channel. It implies that the interaction flow is determined by the controlled channel rather than the channel actually used by the user. Such mismatches typically reduce usability and negatively impact the user experience.

5. Authoring issues

The problem of authoring for multiple synchronized modalities is closely related to the issues of device-independent authoring and authoring applications to be rendered in different channels (modalities) [1,2]. More details can be found in [3,4].

Let us define gestures as the units of synchronized blocks. The set of gestures depend on the authoring approach supported by the browser.

For single authoring, gestures are the elementary units defined by the language syntax and for which transformation rules are available for each channel / view. Pages are authored in the modality-independent representation and transformed dynamically into the different channel specific languages. Naming conventions or node identification are associated to each of the resulting elements in each channel. Since any portion of any given view can be mapped back (through the node identification) to the generating portion of the

underlying modality-independent representation, and this portion consequently mapped back to the corresponding view in a different modality by applying the appropriate transformation rules, the approach automatically satisfies the MVC principle.

For multiple authoring, gestures are the blocks in each modality that are synchronized with one another. Different approaches can be considered:

- Explicit synchronization tags: co-visit URL tags indicate that when reaching this item a new page must be loaded by the other view. Typically the tags delimitate the gestures on the pages. These approaches extend each channel specific presentation language to add the co-visit tags. If the synchronization tags result into an update of the model, followed by and update of all corresponding associated views, the MVC principle is satisfied. Submit result into polling all the views before submitting to the backend.
- Naming conventions: pages in each modality/channel are authored in unmodified channel-specific languages and gestures are defined by re-using appropriate name conventions for the corresponding elements in each channel. Submit result into polling all the views before submitting to the backend. This is the authoring method used for the loosely coupled browser described in figure 10.
- merged pages: the application is authored by combining snippets from each synchronized modality, re-using unmodified channel specific languages. Gestures are clearly delimitated as combined snippets. If the merged file is parsed in the model, a multi-modal implementation can be obtained by:
    - Shipping well-formatted channel specific snippets pages, one at a time to each view and having the model repository act as a server in between gestures. Submit result into polling all the views before submitting to the backend.
    - Adding automatically synchronization tags or naming convention and proceed as above.

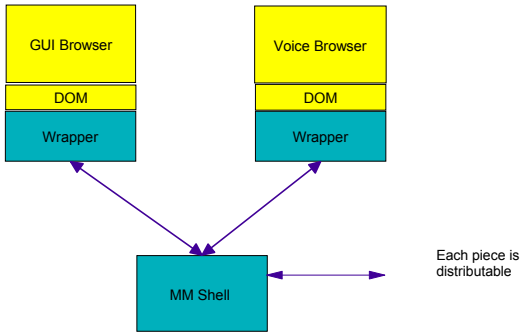

Figure 2 – DOM-based MVC Multi-modal browser architecture.

In conclusion, the proposed MVC architecture can support single and multiple authoring approaches.

Figure 2 illustrates how a multi-modal browser can be implemented following the MVC principle and using existing channel specific DOM compliant browsers.

6. Conclusions

In this paper, we have introduced the importance of multi-modal e-business. We have proposed a standard-based multi-modal browser architecture based on the MVC principle and linked it to the different method that can be used for multi-modal applications.

7. References

[1] S. H. Maes and T. V. Raman, IBM position paper, “Position paper for the W3C/WAP Workshop on the Multi-modal Web”, W3C/WAP joint Workshop on the Multi-modal Web, Hong Kong, September 2000.

[2] R. B. Case, S. H. Maes and T. V. Raman, “Position paper for the W3C/WAP Workshop on the Web Device Independent Authoring”, W3C/WAP joint Workshop on Web Device Independent Authoring, Bristol, October 2000.

[3] W3C/WAP Workshop: the Multimodal Web, Hong Kong, HK, September 2000, http://www.w3.org/2000/09/Papers/Agenda.html

[4] W3C/WAP Workshop: Web Device Independent Authoring, Bristol, UK, October 2000, http://www.w3.org/2000/10/DIAWorkshop/

[5] ETSI STQ Aurora DSR Working Group, Applications and protocols sub-group

[6] W3C Voice Browser working group, Multi-modal Sub-group, http://www.w3.org/Voice/

[7] VoiceXML Forum, http://www.voicexml.org

[8] S. T. Pope and G. Krasner., A cookbook for using the model-view-controller user interface paradigm in smalltalk-80, Journal of O-O Programming, 1(3):26--49, 1987.

[9] K. A. Papineni, S. Roukos, R. T. Ward,Free-flow dialog management using forms, Proc. Eurospeech, 1999.