Towards A Minimalist Multimodal Dialogue Framework Using Recursive MVC Pattern

Li Li
Avaya Labs Research
233 Mt. Airy Road, Basking Ridge
NJ 08807, USA
lli5@avaya.com

Wu Chou
Avaya Labs Research
233 Mt. Airy Road, Basking Ridge
NJ 08807, USA
wuchou@avaya.com

ABSTRACT
This paper presents a formal framework for multimodal dialogue systems by applying a set of complexity reduction patterns. The minimalist approach described in this paper combines recursive application of Model-View-Controller (MVC) design patterns with layering and interpretation. It leads to a modular, concise, flexible and dynamic framework building upon a few core constructs. This framework could expedite the development of complex multimodal dialogue systems with sound software development practices and techniques. A XML based prototype multimodal dialogue system that embodies this framework is developed and studied. Experimental results indicate that the proposed framework is effective and well suited for multimodal interaction in complex business transactions.

Categories and Subject Descriptors
H.5.2 [**INFORMATION INTERFACES AND PRESENTATION**]: User Interfaces – *Theory and methods, Prototyping, Evaluation/methodology.*

General Terms
Design, Languages, Theory.

Keywords
Multimodal, Dialogue, MVC, XML.

1. INTRODUCTION
There is a growing trend to incorporate rich multimodal interactions in various user interfaces, including web browser, mobile device, kiosk, virtual reality, etc [1]. Studies [13] have shown that multimodal interaction is the most effective interface between human and machine. However, the effectiveness comes with a cost because multimodal dialogue systems are much more complex than unimodal ones. The complexity is rooted in the variability and uncertainty introduced by different modalities at both temporal and semantic levels. A user may use speech and gesture in one dialogue turn, and GUI and pen in the next turn. Individual user's ability and environmental conditions may also change. The compound variations increase exponentially with the number of modalities. Despite the progresses made in some basic research topics, most current architectures focus on a particular set of modalities/directionalities targeted at some particular application environment. The general frameworks that we inspected still have the following limitations:

1. Lack a formal specification that defines the behaviors and interactions of the components; This hinders the adoption of other useful modern software development techniques, such as model-driven design, simulation and model checking, in MMI system development;
2. Lack an articulation of the design patterns that underline the result architecture; This prevents MMI architectural researches from taking advantage of widely used design patterns in software industry;
3. Lack a built-in mechanism to accommodate dynamic variability in modalities;

Therefore, there is an acute need for an extensible MMI framework that are based on sound design patterns. Furthermore, we believe that a general framework will facilitate sharing and reusing components and technologies in multimodal research communities. Moreover, a formal framework provides a reference to compare and evaluate MMI system designs, as well as to improve the MMI architectures in a consistent and structured way.

This paper is organized as follows. In Section 2, we describe the framework following an analysis and description of the design patterns. Section 3 describes a prototype XML based MMI dialogue system and our experimental studies. The related work is discussed in Section 4. Findings of this paper are summarized in Section 5.

2. RECURSIVE MVC FRAMEWORK
Our design of the MMI framework follows a minimalist approach by using the following complexity reduction patterns: *layering*, *recursion*, *MVC decomposition* and *interpretation*. In a nutshell, we divide the multimodal functions into layers and recursively apply the MVC decomposition to integrate these functions with interpretation. This leads to an extensible multimodal dialogue framework built upon only a few core constructs. The "minimalist" hence indicates our effort to discover and develop a minimal set of building blocks for maximal complexity reduction for MMI architectures.

2.1 Layering

The layering technique has been successfully applied to the design of many complex communication systems, including the Internet. Human language processing is believed to consist of layers: discourse, pragmatics, semantics, syntax, morphology, phonetics and phonology [4]. For multimodal dialogue systems**,** the general consensus is that there are four layers, listed from high to low: *Business process*, *dialogue*, *coordination*, and *physical presentation* [8][20], although the precise partitioning of the layers may vary by applications. The complexity reduction is achieved by layer isolation: a layer can only interact with its adjacent layers. This means that the business process is decoupled from the coordination layer, and the dialogue concern is separated from the physical presentation.

Most traditional layered systems are built from a bottom-up fashion such that the higher layer depends on the lower ones (downward dependency). However, the dependency direction is reversed in our framework in that lower level depends on the higher level (upward dependency). This direction reversal reflects the fact that multimodal dialogue is essentially a top-down decision process, and the variability increases towards the lower layers. The upward dependency thus hides the complexity of heterogeneous multimodalities and devices from the dialogue and coordination layers, making these layers more portable to different applications.

2.2 MVC Decomposition

MVC [5] is a classic design pattern widely used in GUI and Web applications. Generally speaking, *Model* defines and maintains the data, *View* renders the interactions based on the data, and *Controller* coordinates actions and events that affect the Model and View..

Similar to [10], the concept of view in our framework is not confined to the graphical user interface. Instead, it means a *composite view* that coordinates activities between a set of *primitive views* according to spatial, temporal, and semantic constraints. More formally, a composite view at layer L is defined by MVC as follows: $v_c^L = (M, V_p^L, C(V_p^L))$, where the controller C represents the coordination between the primitive views V_p, and the model M records the collected data. The notion of primitive view is relative: it means the view can be rendered by the next layer. For example, a VXML script and a HTML page are primitive views rendered by browsers. A composite view that renders them in parallel at the same time can be represented as:

$$v_c^L = (M, \{v_1, v_2\}, \{(par, 1, 2), (2.begin = 1.begin)\})$$

The rendering process is achieved by an abstract function *coordinate(C,a,V)* which coordinates activities *a* for each view in *V* according to *C*. An instance of this abstraction function is a SMIL engine [14].

2.3 Recursion

Multimodal dialogue systems exhibit recursive MVC patterns in all layers as illustrated in [10]. In our framework, a composite view can be decomposed into MVC components, which creates this abstract recursive relation: *MVC = M(MVC)C*. This recursion eventually terminates when the primitive views cannot be decomposed further. Recursive MVC thus imposes a uniform hierarchical structure to represent arbitrarily complex multimodal dialogue systems using the Composite pattern [5].

2.4 Interpretation

A typical example of interpretation [5] is a parser that interprets sentences in a language based on the grammar for that language. For consistency, we will call the parser "interpreter" and the component that generates the sentences "generator". The complexity is significantly reduced because one interpreter can carry out different functions according to the sentences. Moreover, the functions performed by the interpreter are dynamically determined by the sentences, which themselves can be dynamically generated. The notion of "sentence" can expand to an entire dialogue markup, such as VXML [16], to give "second-order" expressiveness and flexibility to the framework. The combination of generator and interpreter not only supports upward dependency but also can resolve the dependency on the fly during dialogue process. For example, if we want to compose a dialogue to collect the credit card number, but do not want to lock in the modalities at design time, we can use generators to derive the modalities based on the contextual information.

2.5 Framework Specification

The conceptual framework based on the design principles is specified recursively as follows (Figure 1).

```
Layer $L$ program consists of interpreter $I^L$ and generator $G^L$.
Layer $L$ representation = $(M^L, V^L, C^L)$.
Let $S^L = \{s_i^L \mid 0 \le i \le n\}$ denote the set of states of $I^L$.
Let $t^L$ denote the timeout at layer $L$.
$display(I^L, v_p^{L-1}, t^L)$ is defined as follows:
    $(M^L, V^L, C^L) = G^L(v^{L-1})$
    $(M, v_c, s) = (M^L, v_0^L, s_0^L)$
    while $s \ne s_n \wedge time < t^L$
        $R^{L+1} = display(I^{L+1}, v_c, t^{L+1})$
        $(M, v_c, s) = update(C^L, M^L, R^{L+1}, s)$
    $R^L = result(M)$
    $return(R^L)$

$display(I^{L+1}, v_c^L, t^{L+1})$ is defined as follows:
    $(M, V_p^L, C(V_p^L)) = v_c^L$
    $coordinate(C, R = display(I^{L+1}, v_p^L, t^{L+1}), V_p^L)$
    $M = merge(R)$
    $return(M)$
```

Figure 1: The formal specification of the recursive MVC framework

In this framework a MMI dialogue system consists of layers of programs (interpreter and generator). The function of each layer can be represented by MVC. The interpreter realizes the turn-taking and real-time aspects of dialogue systems by timed state transitions. The layer *L* generator derives its MVC from the layer *L-1* view and coordinates the composite views on layer *L* by delegating the primitive views to layer *L+1*. The interpreter collects results and updates its states, model and views guided by the controller. Eventually, the display will be grounded to the physical presentation through their APIs, where the interactions with the user occur. The overall execution flows of the framework are illustrated by the following diagram (Figure 2).

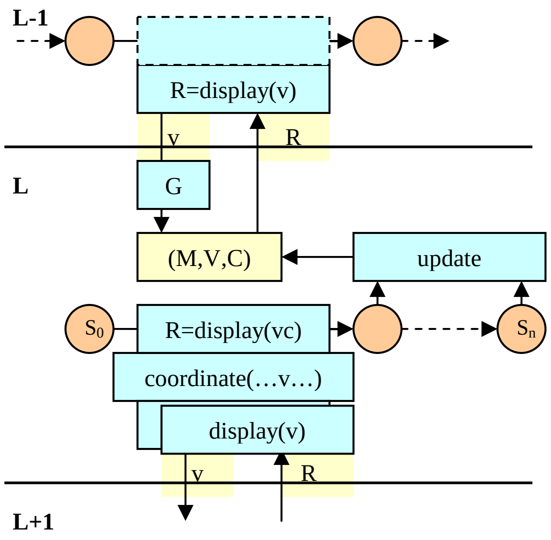

Figure 2: Execution flows of the framework

The components of the system can be distributed over the network or collocated in one process. The framework permits the interactions between layers to be asynchronous and distributed, since only views and results are exchanged between layers. The conceptual framework can also be implemented by different physical architectures, such as hub and spoke architecture [18][23].

3. A XML MMI PROTOTYPE SYSTEM

We implemented a XML based MMI prototype system based on the general framework described in the previous Section. The prototype is based on several W3C standards with some extensions. The business process is hosted on a web server. The dialogue is represented by SCXML [12]. The composite view is defined by SMIL [14] and EMMA [6] and the presentation layer includes Google Map API and MS SAPI recognizer (Figure 2 and Figure 3). The entire dialogue runs within a web browser where the interpreters (SCXML engine and SMIL engine) and generators are implemented by Java Applet (Java), ActiveX control (C++) and JavaScript. The composite views are tokens used by the generators to generate the MVC XML with table lookup. The interfaces between layers are based on [10].

The multimodal dialogue system supports simultaneous and coordinated speech, mouse, textual and map input/output for a call center agent to dispatch technicians to customer sites in response to incoming service calls or alerts. The agent is presented with web browser in which a map shows customer sites with service requests. In a typical scenario, the agent issues multimodal commands in the dialogue turns illustrated as follows:

Agent: find technicians within ten miles of this [click on marker] site.
System: [shows found technicians on map] what next?
Agent: send these technicians [click on markers] to this site [click on marker].
System: [shows technicians moving toward the site] what next?

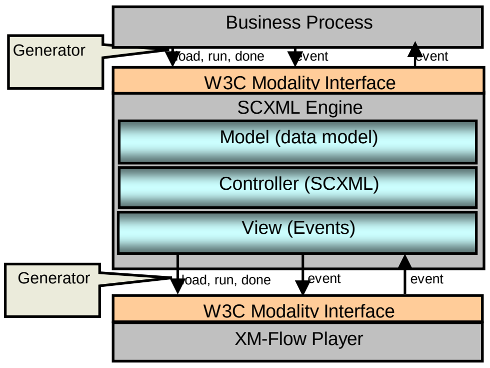

Figure 3: The business, dialogue and composite view layers

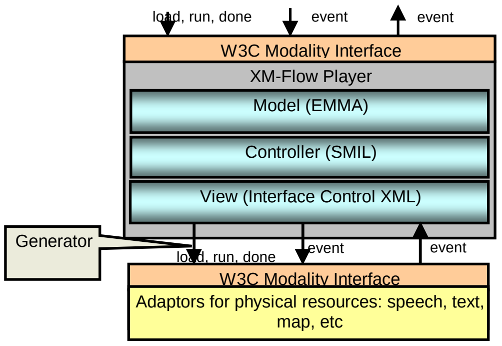

Figure 4: The primitive view and presentation layers

Table 1. Time spending on loading and analyzing XML files (millisecond), N=20

	Load & Analyze State Chart XML File		Load & Analyze XM-Flow XML File
	1st time	Thereafter	
Mean	1956.90	396.40	38.24
Std	142.97	20.14	7.85

Table 1: initialization time of interpreters

Table 2. Time spending on each stage (component) for a state-state transition (millisecond), N=20

	Total	State machine engine t1	State engine to XM-Flow player t2	XM-Flow player t3	XM-Flow player to state engine t4
Mean	172.91	55.58	29.52	47.5	40.31
Std	77.9	13.4	28.33	11.65	41.52

Table 2: execution time of interpreters and generators

To investigate the performance overhead caused by interpretations, we measured the speed of the XML interpreters and generators on a PC with 1.6 GHz CPU and 512 MB RAM. The results of 20 trials are summarized in Table 1 and Table 2.

The studies indicated that the performance of the interpretation is acceptable and the small average standard deviation indicates that the system's response time was quite consistent and predictable.

4. RELATED WORK

There are many research projects on multimodal interaction architecture: [2], [3], [7], [8], [9], [11], [15], [18], [19], [21], [22], [23]. But these architectures are mostly focus on a particular set of modalities in a fixed environment rather than a general framework. [20] and [1] propose layered general architectures for multimodal interactions based on a comprehensive list of multimodal dialogue components. However, the coordination, dependencies and interactions between the layers are not clearly defined. Nor do they use recursive MVC patterns to provide a formal specification. Recent VXML v3.0 [17] moves to a modular architecture design based on separation of Data, Flow and Presentation, which has a direct relation to MVC design pattern. But its current design only provides unimodal (voice) input and multimodal output.

Our design patterns follows the design goals of MMAI [10] and our architecture thus shares some key features, such as reclusiveness, with MMAI. However, our focus and contribution are towards a formal specification of the MMI architecture derived from the basic design patterns, instead of the interface and messages between the components. In particular, we emphasize the dynamic aspect of the MMI architecture based on interpretation, which is not articulated in the MMAI proposal.

5. SUMMARY

In this paper, we presented a minimalist framework for multimodal dialogue systems by applying a set of sound complexity reduction patterns in MMI system architecture design, namely, the layering, recursion, MVC, and interpretation. These techniques are combined to create a formal framework that is modular, concise, flexible and dynamic. We described a XML based multimodal dialogue prototype system based on the proposed framework. Experimental studies indicate that the proposed approach is effective.

We believe such structured approach will help us gain more insights into multimodal interactions and advance multimodal dialogue systems in an extensible and consistent way. Further studies are on-going to investigate issues in dynamic composition and synchronization for distributed multimodal dialogue systems.

ACKNOWLEDGMENTS

Authors would like to thank Fei Cao, Quagnzhi Li and Feng Liu for their contributions to this work.

REFERENCES

[1] Stock, O., Zancanaro, M. (eds.): Multimodal Intelligent Information Presentation. Series Text, Speech and Language Technology, Vol 27. Springer, 2005.

[2] Stanciulescu, Q. Limbourg, J. Vanderdonckt, B. Michotte, F. Montero, 2005. A Transformational Approach for Multimodal Web User Interfaces based on UsiXML, *Proceedings of ICMI'05*, pp. 259-266, October 2005.

[3] Christian Elting, Gregor Mohler, 2002. Modeling Output in the EMBASSI Multimodal Dialogue System, ICMI 2002:111-116, *Fourth IEEE International Conference on Multimodal Interfaces (ICMI'02)*, 2002.

[4] Daniel Jurafsky, James H. Martin, 2000. *Speech and Language Processing*. Prentice Hall, 2000.

[5] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, 1995. *Design Patterns*. Addison Wesley, 1995.

[6] EMMA: Extensible MultiModal Annotation markup language, W3C Candidate Recommendation, 11 December 2007.

[7] Jullien Bouchet, Laurence Nigay, 2004. ICARE: a component-based approach for the design and development of multimodal interfaces. *ICMI 2004*: 1325-1328, 2004.

[8] Li Li, Quanzhi Li, Wu Chou, Feng Liu, 2007. R-Flow: An Extensible XML Based Multimodal Dialogue System Architecture, *Proceedings of 9th IEEE International Workshop on Multimedia Signal Processing* (MMSP 2007), page 86-89, 2007.

[9] M. Honkala, M. Pohja, 2006. Multimodal Interaction with XForms, *Proceedings of ICWE'06*, pp. 201-208, July 2006.

[10] MMAI: Multimodal Architecture and Interfaces, W3C Working Draft, 14 April 2008.

[11] SALT: SALTForum, http://www.saltforum.org/

[12] SCXML: State Chart XML (SCXML): State Machine Notation for Control Abstraction 1.0, W3C Working Draft, 16 May 2008.

[13] Sharon Oviatt, 1999. Ten myths of multimodal interaction, in *Communications of ACM*, 42(11):74-81, November 1999.

[14] SMIL: Synchronized Multimedia Integration Language (SMIL 3.0), W3C Candidate Recommendation, 15 January 2008.

[15] Stephane H. Maes, Chummun Ferial, 2001. Multi-Modal Browser Architecture, *presentation (T2-010705) on 3GPP T2 meeting*, September 2001.

[16] VXML v2.1: Voice Extensible Markup Language (VoiceXML) 2.1, W3C Recommendation, 19 June 2007

[17] VXML v3.0: Sneak Preview: VoiceXML 3.0, www.w3.org/Voice/2006/voicexml3.pdf

[18] Johnston, M. et al. MATCH: An Architecture for Multimodal Dialogue Systems, *Proceedings of the 40th Annual Meeting of ACL*, Philadelphia, July 2002, pp. 376-383, 2002.

[19] Herzog, G. et al. MULTIPLATFORM Testbed: An Integration Platform for Multimodal Dialog Systems. *Proceedings of the HLT-NAACL 2003 workshop on Software engineering and architecture of language technology systems - Volume 8*, Pages: 75 – 82, 2003.

[20] Bunt, Harry, et al. Fusion and Coordination for Multimodal Interactive Information Presentation. In [1], 2005

[21] Kaiser, E. et al. Mutual Disambiguation of 3D Multimodal Interaction in Augmented and Virtual Reality, *ICMI-PUI '03*, November 5-7, 2003.

[22] Dumas B. et al. Strengths and weaknesses of software architectures for the rapid creation of tangible and multimodal interfaces. *TEI 2008*, February 18-20, 2008.

[23] Cohen, P.R. et al. QuickSet: Multimodal Interaction for Distributed Applications. Proceedings of ACM Multimedia 1997, pages 31-40, 1997.

[24] Larson, James. et al. (eds). W3C Multimodal Interaction Framework, W3C NOTE, 06 May 2003.