Enhancing Conversational Flexibility in Multimodal Interactions with Embodied Lifelike Agents

Kyoshi Mori **Adam Jatowt** **Mitsuru Ishizuka**

Department of Information and Communication Engineering

Graduate School of Information Science and Technology

University of Tokyo

7-3-1, Hongo, Bunkyo-ku, Tokyo, Japan 113-8656.

kmori@miv.t.u-tokyo.ac.jp

ABSTRACT

Research carried out in authoring systems for embodied agent based presentations have traditionally been confined to scripted interactive presentations. In recent years, however, there has been a gradual shift to adopting a more dynamic approach that supports a higher degree of flexibility in user-agent interactivity, where the user is allowed to engage in more natural conversations with the agent. In this paper, we will describe a conversational module based on techniques used in chat-bots, that we have implemented as an extension to our previously developed agent authoring system.

Categories and Subject Descriptors :

H.5.2 **[Information Interfaces and Presentation (e.g., HCI)]**: User Interfaces---interaction styles (e.g., commands, menus, forms, direct manipulation), natural language; H.1.2 **[Models and Principles]**: User/Machine Systems---human factors

General Terms

Human Factors, Languages

Keywords

Conversational Agents, Multimodal Interaction, Chatbots, Presentation Markup Language

INTRODUCTION

In the last couple of years, an increasing number of attempts have been made to develop agent authoring systems that are able to generate agent applications like presentations on the fly. More recent development has led to presentations with increased user interactivity.

We believe that such interactivity should involve direct interaction between the presenter (agent) and the intended audience (user) in an unrestricted dialogue setting. However, as it is impossible for the author to predict all possible user contributions, a mechanism that allows the agent to engage in dialogues not defined by the author is necessary. We will describe our implementation of such a mechanism, which combines chat-bot techniques and domain-to-domain transitions.

MULTIMODAL PRESENTATION MARKUP LANGUAGE

The Multimodal Presentation Markup Language (MPML)[1] developed at the Ishizuka Laboratory, is an XML style markup language designed to allow authors to easily script animated agent presentations. MPML is actually a collection of projects each with a different focus and emphasis. Compared to previous versions, the current version of MPML, MPML 3.0 allows the author to script greater interactivity into a presentation. The author is thus allowed to create interactive background pages containing selections, which when clicked, trigger the appropriate decision branch in the presentation. In this way, the user is able to interact in a somewhat limited degree with the agent. However, such interactive presentations lack believability due to two reasons.

One is that the user interacts directly with the web page as opposed to the agent. By giving the user a set of answers to choose from, the user is momentarily distracted from the agent in attempting to click the right choice. This is inconsistent with live presentations where the audience interacts directly with the presenter. Clearly, agent presentations should also emphasize direct interactions between the user and the agent presenter.

The other reason is that interaction is confined to a limited number of answers that the user is forced to choose from. Although it is not impossible to increase the number of answers, each choice provided requires an additional

IUI'03, January 12-15, 2003, Miami, Florida, USA
ACM 1-58113-586-6/03/0001..

presentation branch. This is due to the sequential structure of the script, which could easily become convoluted if too many choices are scripted into the presentation. In a dialogue where the conversation could easily lead to various topics, a different approach has to be taken.

We will attempt to deal with these problems in our implementation of the Conversational Module, an extension of the current MPML system that allows the creation of conversation based interactive content while maintaining the ease of authoring of such content.

CONVERSATIONAL MODULE FOR MPML3.0

The Conversational Module for MPML 3.0 was implemented using a client-server approach, the client being a modified version of the MPML3.0 system that controls the agent presentation and the server consisting of a dialog engine which accesses a dialog database. A diagram explaining the main components of the module is shown in Figure 1 below.

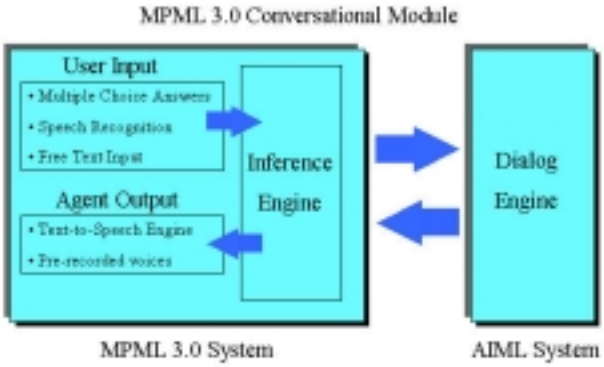

Figure 1

The Conversational Model

Our approach to handling conversations with the agent is to have a case-based mechanism that runs on top of a reasoning mechanism (Figure 2). The case-based mechanism, which does simple pattern matching has the advantage of being quick and thus, being able to return near instantaneous responses to the user. This is crucial in maintaining believability in the interaction as the user does not have to wait long for a response. This reactive component is handled by the Dialog Engine.

On the other hand, simply having a reactive component is not sufficient in maintaining believability. The agent also needs to be pro-active in taking the initiative and ensuring that it achieves its aim in delivering a good presentation. To handle this, the Reasoning Engine tracks the conversation, monitoring both user input and responses from the Dialog Engine and intervening based on changes in the conditions.

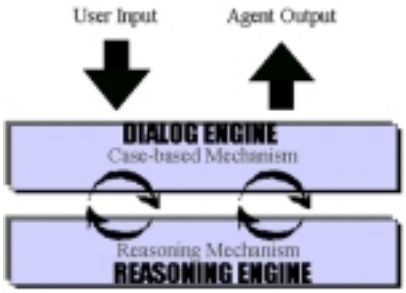

Figure 2

The Dialog Engine

The Dialog Engine is based on the Artificial Linguistic Intelligent Computer Entity (ALICE)[2]. The ALICE chat engine implements the Artificial Intelligence Markup Language (AIML)[3], which allows dialogs between the user and agent to be easily scripted. Based on XML, we believe AIML to be the perfect dialog extension to our MPML presentation system as authors already familiar with scripting in MPML can easily learn to script in AIML.

We are currently using the web version of the ALICE chat engine, where a PHP script accesses AIML defined scripts stored in a mySQL server. The interface with the Javascript based MPML presentation script is implemented using Javascript Remote Scripting.

The Reasoning Engine

Domain Model

The Reasoning Engine is implemented based on the Domain Model. According to Dahlback and Jonnson [4], the Domain Model represents the structure of the world and usually comprises a subset of general world knowledge. It holds knowledge of the world that is talked about. In relation to our research, the Domain Model is the knowledge related to the topic of the presentation, or a collection of subset topics of the main presentation content, depending on the nature of the content. This is advantageous to the author as it reduces the need to predict every single input the user will make, and instead, allows the author to concentrate on scripting conversation within a specified domain.

Application to the Conversational Module

In our system, we have implemented a two-domain system, where the system determines whether the user is conversing within the domain of the presentation topic or whether he or she is speaking out of topic. We will call the two domains In-Domain and Out-of-domain. The In-Domain dialog set is defined by the author, while the Out-of-Domain dialog set is taken from a pre-defined dialog set. The author is of course free to modify the Out-of-Domain set. The transition set is based on the table shown in Figure 3.

Based on the type of input provided by the user, the agent's response state moves from the In-Domain to Out-of-Domain field. To allow for a smooth transition, two intermediate states; Reluctant and Concede, are used. The Reluctant state is based on the fact that if a user says something out of the specified domain, the agent indicates that it is unable to understand the user and is therefore, 'reluctant' to pursue the direction proposed by the user. If the user persists in speaking out of domain, the agent then shifts to the Concede state where it eventually allows the user to take control of the direction of the conversation.

Given input condition : In-Domain **Result in Agent State:**In-Domain **Agent Response:** Standard author defined response
Given input condition : Out-of-Domain **Result in Agent State:**Reluctant **Agent Response:** Lack Understanding handling
Given input condition : Out-of-Domain persists **Given agent state:**Reluctant **Result in Agent State:**Concede **Agent Response:** User initiative handling
Given input condition : Out-of-Domain persists **Given agent state:**Concede **Result in Agent State:**Out-Domain **Agent Response:** Out-of-Domain System defined response

Figure 3

Currently, the factors influencing transitions between states are the user's input and the agent's previous state. Thus, when the user persists in speaking out of domain, the agent state will progress towards the Out-Of-Domain field. However, the agent can only proceed from the In-Domain to Out-of-Domain state by passing through the Reluctant and Concede states.

Although the agent allows the user to control the conversation in the Out-of-Domain state, it will continue to remind the user of the topic of the current presentation, conveying its intention of desiring to move back to its original In-Domain state. Thus, the agent will always direct the conversation towards In-Domain in an attempt to keep the user focused on the presentation. An example of conversation between the user and agent presenter is shown in Figure 4. The conversation between the user and the agent took place after the agent gave a presentation on the topic of the Biography of Albert Einstein.

Agent : I would be glad to answer any questions you have.
User : Tell me more about Einstein's marriage
Agent : Einstein was married twice, first to his classmate and then to his cousin.
User : His cousin?
Agent : His cousin's name was Elsa. Would you like to see pictures related to his marriage?
User : Yes, please.
<Agent presents on Einstein's marriage>
Agent : What else would you like to know?
User : This is boring, let's talk about movies instead.
Agent : I'm not sure how that is of any relation to Einstein. Why don't you ask me about his family instead?
User : No, why can't we talk about movies?
Agent : Fine then, let's talk about it
User : So, do you know Tom Cruise?
Agent : Yes…<Conversation proceeds>

Figure 4

CONCLUSION

In this paper, we have shown how an interactive presentation authoring system allowing natural conversations between the user and the agent presenter can be implemented using our proposed Conversational Module, which incorporates the ALICE chat-bot sub-module. With our module, it is now also possible to script a dialog interaction within a presentation without having to predict all possible interactions that could occur. We plan to extend our work by focusing on increasing the functionality of the module, such as using more efficient state transition techniques.

REFERENCES

1. Sylvain Descamps, Helmut Prendinger and Mitsuru Ishizuka. A Multimodal Presentation Markup Language for Enhanced Affective Presentation, Advances in Education Technologies: Multimedia, WWW and Distant Education In *Proceedings of the International Conference on Intelligent Multimedia and Distant Learning* (ICIMADE-01), Fargo, North Dakota, USA, pp. 9-16, 2001. Available at http://www.miv.t.u-tokyo.ac.jp/MPML/en/
2. Artificial Linguistic Internet Computer Entity (ALICE) Resources. Available at http://alice.sunlitsurf.com/alice/about.html
3. Artificial Intelligence Markup Language (AIML) Resources. Available at http://alice.sunlitsurf.com/alice/aiml.html
4. N. Dahlbäck and A. Jönsson. Integrating domain specific focusing in dialogue models. In *Proceedings of Eurospeech'97*, volume 4, pages 2215-2218, Rhodes, Greece, 1997.