EmoEmma: Emotional Speech Input for Interactive Storytelling

Fred Charles, David Pizzi, Marc Cavazza

University of Teesside, School of Computing
Borough Road, Middlesbrough
TS13BA, United Kingdom
{f.charles, d.pizzi, m.o.cavazza}@tees.ac.uk

Thurid Vogt, Elisabeth André

Multimedia Concepts and Applications
Faculty of Applied Informatics, University of Augsburg
Eichleitnerstr. 30, 86159 Augsburg, Germany
{thurid.vogt, elisabeth.andre}
@informatik.uni-augsburg.de

ABSTRACT
Whilst techniques for narrative generation and agent behaviour have made significant progress in recent years, natural language processing remains a bottleneck hampering the scalability of Interactive Storytelling systems. This demonstrator introduces a novel interaction technique based solely on emotional speech recognition. It allows the user to use speech to interact with virtual actors without any constraints on style or expressivity, by mapping the recognised emotional categories to narrative situations and virtual characters feelings.

Categories and Subject Descriptors
H.5.1 [**Information Interfaces and Presentation**]: Multimedia Information Systems.

General Terms
Algorithms, Human Factors.

Keywords
Interactive Storytelling, Emotional Speech Recognition, Embodied Conversational Agents.

1. INTRODUCTION AND OBJECTIVES
Interactive Storytelling systems [3, 6] can be characterised by their level of user involvement, as well as the user position with respect to the unfolding story. Early systems [1] have introduced a “god mode” approach, in which the user influences virtual actors at some key stages of the narrative action, while retaining a spectator position. Many descriptions of Interactive Storytelling seek inspiration from the “Holodeck” paradigm, in which the user is allowed to play the role of one member of the cast. In both cases, speech is the ideal mode of interaction, where in “god mode” it does not interfere with the user viewing the action, and in immersive mode, it takes part in the natural communication between characters in the narrative. However, interacting from inside the virtual stage has an important consequence: whatever the user says becomes an element of the play/film, and as such should comply with the general style of the narrative genre. This becomes a formidable challenge for speech and natural language processing, well beyond the state-of-the-art.

Emotional speech recognition offers a practical solution to this problem, by allowing unconstrained and natural speech interaction, which can be part of the narrative whilst being analysed in terms of a small number of emotional categories.

Our demonstrator is a complete Interactive Storytelling system featuring virtual agents in a 3D world (Figure 1). The underlying narrative is based on a classical XIXth century psychological novel: *Madame Bovary*, by Flaubert. The interactive narrative is driven by the emotional state of characters and their relationships: this has been achieved by modelling the action corresponding to two chapters from the novel using an emotional planner [4]. The user can impersonate one of the feature characters (Rodolphe Boulanger) to address Emma Bovary or respond to her complaints and love declarations. The real-time analysis of the affective parameters of his utterance determines his influence on his relationship with Emma, which in turn determines scene evolution.

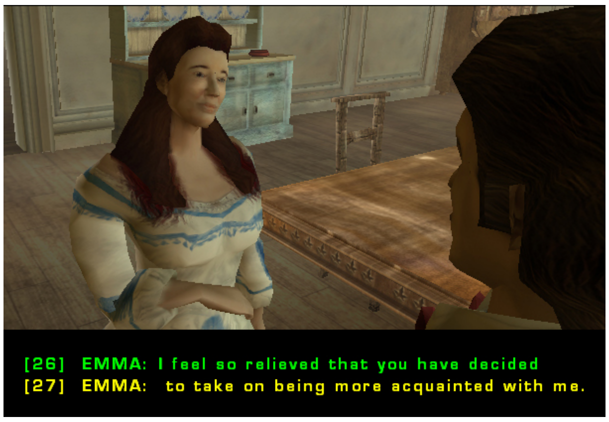
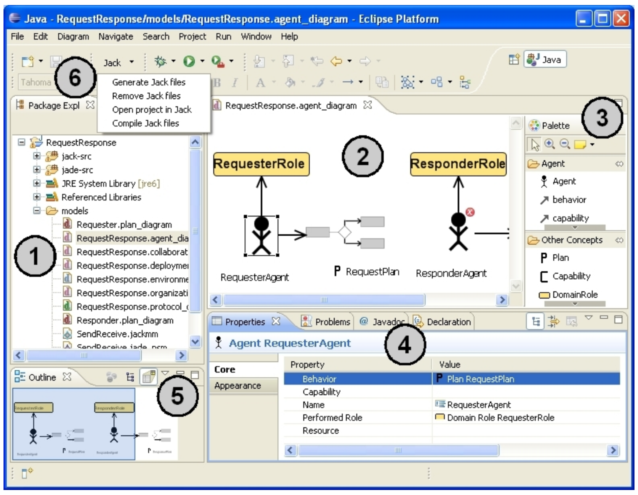

[26] EMMA: I feel so relieved that you have decided
[27] EMMA: to take on being more acquainted with me.

Figure 1. 3D virtual stage and characters: *Emma*, and *Rodolphe* impersonated by the user.

Cite as: EmoEmma: Emotional Speech Input for Interactive Storytelling, Fred Charles, David Pizzi, Marc Cavazza, Thurid Vogt, Elisabeth André, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009),* Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. 1381 – 1382



Property	Value
Behavior	Plan RequestPlan
Capability	
Name	RequesterAgent
Performed Role	Domain Role RequesterRole
Resource	

Figure 1: The DSML4MAS development environment.

Refinement. To support developers in specifying behaviors that conform to a certain protocol, we provide refinement functionalities that transform PIM4AGENTS protocols into PIM4AGENTS behaviors. The transformation saves a lot of redundant work.

Extensibility. DDE is seamlessly integrated into the Eclipse workbench. This implies that other researches can easily develop own extensions for DDE (e.g. transformations, views, model validation, etc.) and plug them into the Eclipse workbench. Furthermore, DDE directly benefits from new developments around the very active Eclipse modeling project[4] and other Eclipse tools.

Open source. At the time of writing this paper, we are finalizing a version that integrates all described features. We will launch an open source project until AAMAS'09. The source code will be published under LGPL. DDE will be available at http://dsml4mas.sourceforge.net.

3. THE DEVELOPMENT ENVIRONMENT

DSML4MAS covers the most import aspects of MASs such as agents, roles, collaborations of agents, protocols, behaviors, deployment aspects, and their environment. For each of these views, DDE offers a diagram that allows modeling MASs with the according language constructs.

Figure 1 depicts the graphical interface of DDE. It consists of several parts: (1) depicts the project explorer, the *models* folder that contains all diagrams and models, and the source folders for the generated Jack and Jade code. (2) shows the modeling area and (3) the palette that contains the language constructs and relations that are available in the current diagram. The properties view which shows the details of the currently selected diagram element is depicted in (4). The diagram outline that is useful for large diagrams is shown at (5). Finally, (6) depicts the Jack menu which offers special actions for the generated Jack code (see Section 4). The icon close to ResponderAgent visualizes a validation error. The error message can be looked up in the problems view of Eclipse.

[4]http://www.eclipse.org/modeling/

4. CODE GENERATION SUPPORT

To close the gap between design and implementation is one of the main objectives of DDE. The code generation within DDE is achieved through model transformations in accordance to the principles of Model-Driven Architecture. Two agent-based execution platforms are supported by DDE, however, other platforms can easily be supported in the same manner.

Jade. The code generation for Jade consists of (i) a transformation of the PIM4AGENTS model to a platform specific model (PSM) for Jade and (ii) a code generation step from the Jade PSM to Jade Java code. The generated Java code can be edited, executed, and debugged like any other Java code in Eclipse. We provide a mechanism for protecting manual changes at the generated source code.

Jack. The code generation for Jack consists of (i) a transformation of the PIM4AGENTS model to a Jack PSM and (ii) a code generation step from Jack PSM to Jack gCode. The generated gCode can be opened with the Jack development environment (e.g. to refine it). We are also able to generate Jack files from the Jack gCode within DDE. Jack files are extended Java files. Finally, the Jack files are compiled by the Jack compiler into Java files (inside Eclipse).

5. CONCLUSION

This paper discusses a novel development tool for designing MAS called DDE. The modeling language is based on a platform independent metamodel for MAS called PIM4AGENTS that defines the vocabulary of the language. Beside several features, code generation facilities are offered by DDE to seamlessly transfer the generated design into executable code. The demo will demonstrate (i) how to specify MASs with DDE, (ii) model validation, (iii) model transformation and code generation for Jack and Jade, and (iv) the execution of the generated code.

6. ACKNOWLEDGEMENTS

We want to thank Cristián Madrigal-Mora, Torsten Gründel, and Stefan Nesbigal who contributed to DDE.

7. REFERENCES

[1] C. Hahn. A domain specific modeling language for multiagent systems. In *Proceedings of 7th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2008)*, pages 233–240, 2008.

[2] C. Hahn and K. Fischer. The static semantics of the domain specific modeling language for multiagent systems. In *Proceedings of the 9th International Workshop on Agent-Oriented Software Engineering (AOSE 2008). Workshop at AAMAS'08*, 2008.

[3] C. Hahn, S. Nesbigall, S. Warwas, I. Zinnikus, K. Fischer, and M. Klusch. Integration of multiagent systems and semantic web services on a platform independent level. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Intelligent Agent Technologies (IAT 2008)*, 2008.

[4] M. Luck, P. McBurney, and J. Gonzalez-Palacios. Agent-based computing and programming of agent systems. In *Proceedings of Programming Multi-Agent Systems, Third International Workshop, ProMAS 2005*, volume 3862 of *Lecture Notes in Computer Science*, pages 23–37, Berlin et al., 2006. Springer.