'How was your day?' An architecture for multimodal ECA systems

Raúl Santos de la Cámara
Telefónica I+D
C/ Emilio Vargas 6
28043 Madrid, Spain
e.rsai@tid.es

Markku Turunen
Univ. of Tampere
Kanslerinrinne 1
FI-33014, Finland
mturunen@cs.uta.fi

Jaakko Hakulinen
Univ. of Tampere
Kanslerinrinne 1
FI-33014, Finland
jh@cs.uta.fi

Debora Field
Computer Science
Univ. of Sheffield
S1 4DP, UK
d.field@shef.ac.uk

Abstract

Multimodal conversational dialogue systems consisting of numerous software components create challenges for the underlying software architecture and development practices. Typically, such systems are built on separate, often pre-existing components developed by different organizations and integrated in a highly iterative way. The traditional dialogue system pipeline is not flexible enough to address the needs of highly interactive systems, which include parallel processing of multimodal input and output. We present an architectural solution for a multimodal conversational social dialogue system.

1 Introduction

Multimodal conversational dialogue applications with embodied conversational agents (ECas) are complex software systems consisting of multiple software components. They require much of architectural solutions and development approaches compared to traditional spoken dialogue systems. These systems are mostly assembled from separate, often pre-existing components developed by different organizations. For such systems, the simple pipeline architecture is not a viable choice. When multimodal systems are built, software architecture should be flexible enough to enable the system to support natural interaction with features such as continuous and timely multimodal feedback and interruptions by both participants. Such features require parallel processing components and flexible communication between the components. Furthermore, the architecture should provide an open sandbox, where the components can be efficiently combined and experimented with during the iterative development process.

The HWYD ('How was your day?') Companion system is a multimodal virtual companion capable of affective social dialogue and for which we have developed a custom novel architecture. The application features an ECA which exhibits facial expressions and bodily movements and gestures. The system is rendered on a HD screen with the avatar being presented as roughly life-size. The user converses with the ECA using a wireless microphone. A demonstration video of the virtual companion in action is available online[1].

The application is capable of long social conversations about events that take place during a user's working day. The system monitors the user's emotional state on acoustic and linguistic levels, generates affective spoken responses, and attempts to positively influence the user's emotional state. The system allows for user initiative, it asks questions, makes comments and suggestions, gives warnings, and offers advice.

2 Communications framework

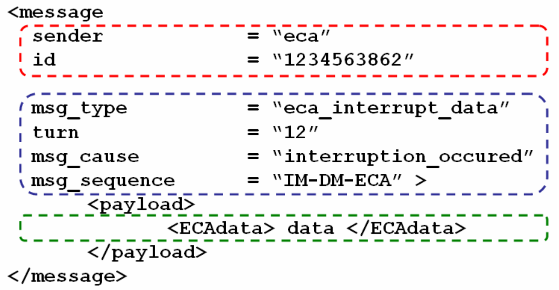
The HWYD Companion system architecture employs Inamode, a loosely coupled multi-hub framework which facilitates a loose, non-hierarchical connection between any number of components. Every component in the system is connected to a repeating hub which broadcasts all messages sent to it to all connected components. The hub and the components connected to it form a single domain. Facilitators are used to forward messages between different domains according to filtering rules. During development, we have experimented with a number of Facilitators to create efficient and simple domains to overcome problems associated with single-hub systems. For example, multiple hubs allow the

[1] http://www.youtube.com/watch?v=BmDMNguQUmM

Proceedings of SIGDIAL 2010: the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue, pages 47–50,
The University of Tokyo, September 24-25, 2010. ©2010 Association for Computational Linguistics

reduction of broadcast messages, which is for example used in the audio processing pipeline, where a dedicated hub allows very rapid message broadcast (nearly 100 messages per second are exchanged) without compromising the stability of the system by flooding the common pipeline.

For communication between components, a lightweight communication protocol is used to support components implemented in various programming languages. A common XML message "envelope" specifies the basic format of message headers as seen in Figure 1.



```
<message
  sender                    = "eca"
  id                        = "1234563862"

  msg_type                  = "eca_interrupt_data"
  turn                      = "12"
  msg_cause                 = "interruption_occured"
  msg_sequence              = "IM-DM-ECA" >
        <payload>
                <ECAdata> data </ECAdata>
        </payload>
</message>
```

Figure 1: System message XML format

.

Mandatory elements in the envelope (top block) are necessary so other modules can identify the purpose of the message and its contents upon a shallow inspection. These include the *sender* component and a unique *message id*. Additional envelope fields elements include: *message type*, *turn id*, *dialogue segment identifier*, *recipient identifier*, and a list of message identifiers corresponding to the previous messages in the current processing sequence.

For system-wide and persistent knowledge management, a central XML-database allows the system to have inter-session and intra-session 'memory' of past events and dialogues. This database (KB) includes information such the user and dialogue models, processing status of modules, and other system-wide information.

3 Data flow in the architecture

To maximize the naturalness of the ECA's interaction, the system implements parallel processing paths. It also makes use of a special module, the **Interruption Manager (IM)**, to control components in situations where regular processing procedure must be deviated from. In addition, there are 'long' and 'short' processing sequences from user input to system output. Both 'loops' operate simultaneously. The Main Dialogue ('long') Loop, which is the normal processing path, is indicated by the bold arrows in Fig. 2, and includes all system components except the IM. The dotted arrows signal the deviations to this main path that are introduced by the interruption management and feedback loops. The system has an activity detector in the input subsystem that is active permanently and analyses user input in real-time. If there is a detection of user input at the same time as the ECA is talking, this module triggers a signal that is captured by the IM. The IM, which tracks the activity of the rest of the modules in the system, has a set of heuristics that are examined each time this triggering signal is detected. If any heuristic matches, the system decides there has been a proper user interruption and decides upon a series of actions to recover from the interruption.

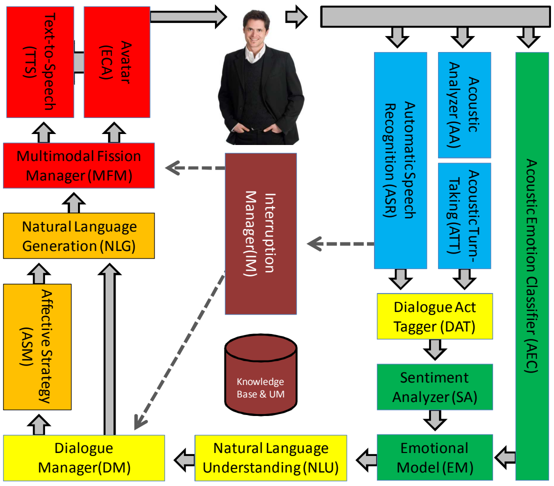

Text-to-Speech (TTS)
Avatar (ECA)
Multimodal Fission Manager (MFM)
Natural Language Generation (NLG)
Affective Strategy (ASM)
Dialogue Manager (DM)
Interruption Manager (IM)
Knowledge Base & UM
Natural Language Understanding (NLU)
Automatic Speech Recognition (ASR)
Acoustic Analyzer (AA)
Acoustic Turn-Taking (ATT)
Dialogue Act Tagger (DAT)
Sentiment Analyzer (SA)
Emotional Model (EM)
Acoustic Emotion Classifier (AEC)

Figure 2:HWYD Companion main modules

4 Module Processing Procedure

The first stage in the processing is the acoustic processing. User speech is processed by the Acoustic Analyzer, the Automatic Speech Recognizer, and the Acoustic Emotion Classifier simultaneously for maximum responsiveness.

The **Acoustic Analyzer (AA)** extracts low-level features (pitch, intensity and the probability that the input was from voiced speech) from the acoustic signal at frequent time intervals (typically 10 milliseconds). Features are passed to the Acoustic Turn-Taking Detector in larger buffers (a few hundred milliseconds) together with time-stamps. AA is implemented in TCL using Snack toolkit (http://www.speech.kth.se/snack/).

The **Acoustic Turn-Taking detector (ATT)** is a Java module, which estimates when the user has finished a turn by comparing intensity pause lengths and pitch information of user speech to configurable empirical thresholds. ATT also decides whether the user has interrupted the system ('barge-in'), while ignoring shorter backchanneling phrases (Crook et al. (2010)). Interruption messages are passed to the Interruption Manager. ATT receives a message from the ECA module when the system starts or stops speaking.

Dragon Naturally Speaking **Automatic Speech Recognition (ASR)** system is used to provide real-time large vocabulary speech recognition. Per-user acoustic adaptation is used to improve recognition rates. ASR provides N-best lists, confidence scores, and phrase hypotheses.

The **Acoustic Emotion Classifier (AEC)** component (EmoVoice (Vogt et al. (2008)) categorizes segments of user speech into five valence+arousal categories, also applying a confidence score. The Interruption Manager monitors the messages of the AEC to include emotion-related information into feedback loop messages sent to the ECA subsystem. This allows rapid reactions to the user mood.

The Sentiment Analyzer (SA) labels ASR output strings with sentiment information at word and sentence levels using valence categories *positive*, *neutral* and *negative*. The SA uses the AFFECTiS Sentiment Server, which is a general purpose .NET SOAP XML service for analysis and scoring of author sentiment.

The **Emotional Model (EM)**, written in Lisp, fuses information from the AEC and SA. It stores a globally accessible emotional representation of the user for other system modules to make use of. Affective fusion is rule-based, prefers the SA's valence information, and outputs the same five valence+arousal categories as used in the AEC. The EM can also serve as a basis for temporal integration (mood representation) as part of the affective content of the User Model. It also combines the potentially different segmentations by the ASR and AEC.

The **User Model (UM)** stores facts about the user as objects and associated attributes. The information contained in the User Model is used by other system modules, in particular by Dialogue Manager and Affective Strategy Module.

The **Dialogue Act Tagger and Segmenter (DAT)**, written in C under Linux, uses the ATT results to compile all ASR results corresponding to each user turn. DAT then segments the combined results into semantic units and labels each with a dialogue act (DA) tag (from a subset of SWBD-DAMSL (Jurafsky et al. (2001)). A Stochastic Machine Learning model combining Hidden Markov Model (HMM) and N-grams is used in a manner analogous to Martínez-Hinarejos et al. (2006). The N-grams yield the probability of a possible DA tag given the previous ones. The Viterbi algorithm is used to find the most likely sequence of DA tags.

The **Natural Language Understanding (NLU)** component, implemented in Prolog, produces a logical form representing the semantic meaning of a user turn. The NLU consists of a part-of-speech tagger, a Noun Phrase and Verb Group chunker, a named-entity classification component (rule-based), and a set of pattern-matching rules which recognize major grammatical relationships (subject, direct object, etc.) The resulting shallow-parsed text is further processed using pattern-matching rules. These recognize configurations of entity and relation relevant to the templates needed by the Dialogue Manager, the EM, and the Affective Strategy Module.

The **Dialogue Manager (DM)**, written in Java and Prolog, combines the SA and NLU results, decides on the system's next utterance and identifies salient objects for the Affective Strategy Module. The DM maintains an information state containing information about concepts under discussion, as well as the system's agenda of current conversational goals.

One of the main features of the HWYD Companion is its ability to positively influence the user's mood through its **Affective Strategy Module (ASM)**. This module appraises the user's situation, considering the events reported in the user turn and its (bi-modal) affective elements. From this appraisal, the ASM generates a long multi-utterance turn. Each utterance implements communicative acts constitutive of the strategy. ASM generates influence operators which are passed to the Natural Language Generation module. ASM output is triggered when the system has learned enough about a particular event to warrant affective influence. As input, ASM takes information extraction templates describing events, together with the emotional data attached. ASM is a Hierarchical Task Network (HTN) Planner implemented in Lisp.

The **Natural Language Generator (NLG)**, written in Lisp, produces linguistic surface forms from influence operators produced by the ASM. These operators correspond to communicative actions taking the form of performatives. NLG uses specific rhetorical structures and constructs associated with humour, and uses emotional TTS expressions through specific lexical choice.

5 Multimodal ECA Control

Multimodal control of the ECA, which consists of a tightly-synchronized naturalistic avatar and affective **Text-To-Speech (TTS)** generation, is highly challenging from an architectural viewpoint, since the coordinating component needs to be properly synchronized with the rest of the system, including both the main dialogue loop and the feedback and interruption loops.

The system Avatar is in charge of generating a three-dimensional, human-like character to serve as the system's 'face'. The avatar is connected to the TTS, and the speech is synchronized with the lip movements. The prototype is currently using the Haptek^{TM} 3D avatar engine running inside a web browser. The Haptek engine provides a talking head and torso along with a low level API to control its interaction with any SAPI-compliant TTS subsystem, and also allows some manipulation of the character animation. An intermediate layer consisting of a Java applet and Javascript code embeds the rendered avatar in a web page and provides connectivity with the Multimodal Fission Manager. We intend to replace the current avatar with a photorealistic avatar under development within the project consortium.

Loquendo^{TM} TTS SAPI synthesizer is used to vocalize system turns. The TTS engine works in close connection with the ECA software using the SAPI interface. TTS includes custom paralinguistic events for producing expressive speech. TTS is based on the concatenative technique with variable length acoustic units.

The Multimodal Fission Manager (MFM) controls the Avatar and the TTS engine, enabling the system to construct complex communicative acts that chain together series of utterances and gestures. It offers FML-standard-based syntax to make the avatar perform a series of body and facial gestures.

The system features a template-based input mode in which a module can call ECA to perform actions without having to build a full FML-based XML message. This is intended to be used in the feedback loops, for example, to convey the impression that the ECA is paying attention.

6 Conclusions

We have presented an advanced multimodal dialogue system that challenges the usual pipeline-based implementation. To do so, it leverages on an architecture that provides the means for a flexible component interconnection, that can accomodate the needs of a system using more than one processing path for its data. We have shown how this has enabled us to implement complex behavior such as interrupt and short loop handling. We are currently expanding coverage and will carry out an evaluation with real users this September.

Acknowledgements

This work was funded by Companions, a European Commission Sixth Framework Programme Information Society Technologies Integrated Project (IST-34434).

References

Vogt, T., André, E. and Bee, N. 2008. EmoVoice – A framework for online recognition of emotions from voice. In: *Proc. Workshop on Perception and Interactive Technologies for Speech-Based Systems*, Springer, Kloster Irsee, Germany.

Cavazza, M., Smith, C., Charlton, D., Crook, N., Boye, J., Pulman, S., Moilanen, K., Pizzi, D., Santos de la Camara, R., Turunen, M. 2010 *Persuasive Dialogue based on a Narrative Theory: an ECA Implementation*, Proc. 5th Int. Conf. on Persuasive Technology (to appear).

Hernández, A., López, B., Pardo, D., Santos, R., Hernández, L., Relaño Gil, J. and Rodríguez, M.C. 2008 Modular definition of multimodal ECA communication acts to improve dialogue robustness and depth of intention. In: Heylen, D., Kopp, S., Marsella, S., Pelachaud, C., and Vilhjálmsson, H. (Eds.), *AAMAS 2008 Workshop on Functional Markup Language*.

Crook, N., Smith, C., Cavazza, M., Pulman, S., Moore, R., and Boye, J. 2010 Handling User Interruptions in an Embodied Conversational Agent. In *Proc. AAMAS 2010*.

Wagner J., André, E., and Jung, F. 2009 Smart sensor integration: A framework for multimodal emotion recognition in real-time. In *Affective Computing and Intelligent Interaction 2009.*

Cavazza, M., Pizzi, D., Charles, F., Vogt, T. André, E. 2009 Emotional input for character‐based interactive storytelling *AAMAS (1) 2009*: 313-320.

Jurafsky, D. Shriberg, E., Biasca, D. 2001 *Switchboard swbd‐damsl shallow‐ discourse‐function annotation coders manual.* Tech. Rep. 97 ‐01, University of Colorado Institute of Cognitive Science

Martínez‐Hinarejos, C.D., Granell, R., Benedí, J.M. 2006. *Segmented and unsegmented dialogue‐act annotation with statistical dialogue models*. Proc. COLING/ACL Sydney, Australia, pp. 563‐570.