

# The IMIX Demonstrator: An Information Search Assistant for the Medical Domain

Dennis Hofs, Boris van Schooten and Rieks op den Akker

**Abstract** In the course of the IMIX project a system was developed to demonstrate how the research performed in the various subprojects could contribute to the development of practical multimodal question answering dialog systems. This chapter describes the IMIX Demonstrator, an information search assistant for the medical domain. The functionalities and the architecture of the system are described, as well as its role in the IMIX project.

## 1 Introduction

The IMIX research programme covered four research areas:

- (i) Automatic speech recognition (ASR) within the context of multimodal interaction;
- (ii) Dialogue management and reasoning;
- (iii) Information presentation in multimodal systems in natural language;
- (iv) Information extraction.

An important aim of the IMIX Programme was to bring research from these different fields together. The building of a common demonstrator was seen as a means to this end. This chapter describes the common demonstrator. The challenge was to find an appropriate architecture which would allow the development of loosely coupled modules in such a way, that changes to one module during

---

Dennis Hofs

Roessingh Research and Development, Enschede, The Netherlands, e-mail: [d.hofs@rrd.nl](mailto:d.hofs@rrd.nl)

Boris van Schooten

University of Twente, Enschede, The Netherlands, e-mail: [schooten@ewi.utwente.nl](mailto:schooten@ewi.utwente.nl)

Rieks op den Akker

University of Twente, Enschede, The Netherlands, e-mail: [infrieks@ewi.utwente.nl](mailto:infrieks@ewi.utwente.nl)

the course of the project would not influence the set up of the whole system. Section 3 presents the architecture and describes the different modules contributed by the various subprojects. The next section presents the functionality of the IMIX Demonstrator and the interaction with the user. The final section 4 discusses the role the Demonstrator played in the IMIX project. The technical coordination of the Demonstrator was done by the first author of this chapter. He was responsible for the integration of the various modules. The kernel of the Demonstrator is the dialogue manager module. It is dependent on every module, so architecture and functional specification was managed by the Vidiam subproject (see the chapter by Van Schooten and Op den Akker, *Vidiam: Corpus-based Development of a Dialogue Manager for Multimodal Question Answering*, this volume). The Demonstrator was developed in three phases in the course of three years. This book describes the final version of the Demonstrator<sup>1</sup>.

## 2 A Medical Information Search Assistant

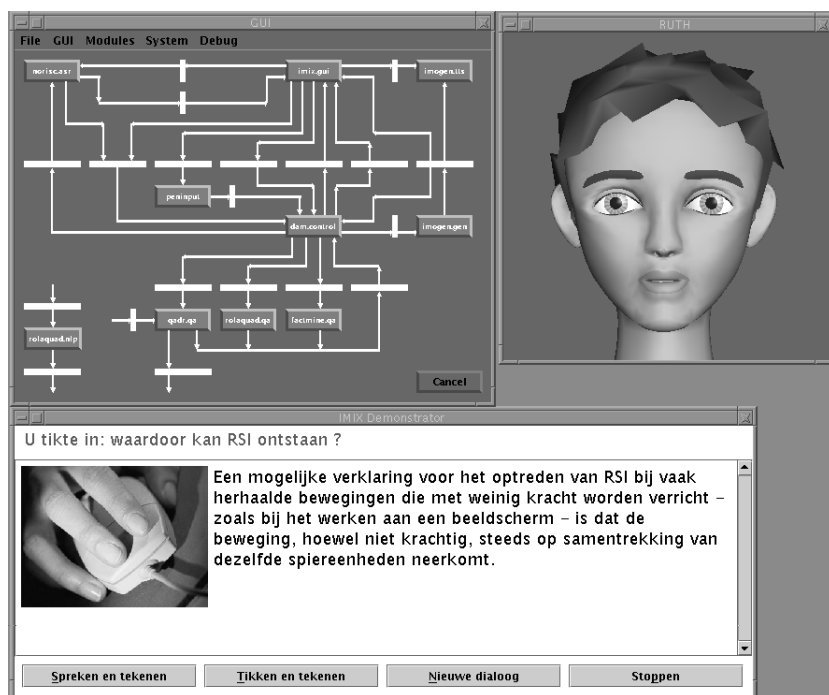
IMIX is an information assistant, a system that helps users find the information they need in the medical domain (op den Akker et al, 2005). The IMIX Programme extends iterative QA in the direction of multimodal QA. IMIX allows users to use speech input as well as text and gestures. The answer is presented in the form of text, speech and pictures. Users are allowed to refer to the pictures in the answer presentation when asking follow-up questions. They can use pointing gestures, encircle parts of pictures or words, or underline them. IMIX is able to offer users a list of options, from which they can make a selection, including using pointing.

Particular choices have been made regarding the nature and difficulty level of the questions IMIX is supposed to handle. An attempt has been made to provide coverage that is sufficient to answer the information needs of real, non-expert users. The questions this project covers have been called “encyclopedic questions”. These are general medical questions that do not require expert medical knowledge (this excludes diagnostic questions). Nevertheless, correctness of answers is less well-defined and harder to measure than with factoid questions. The system typically answers with a piece of text about 1-3 sentences long. The length and level of detail of a correct answer depends on the information needs of the user, which cannot be precisely known beforehand. Enabling the user to give feedback on their information need is one of the purposes of the dialogue system. Although IMIX focuses on a closed medical domain, many of the techniques used are also applicable to open domain QA. This is especially true of the dialogue management module. It uses minimum domain knowledge, which makes it easy to generalise to other domains.

The user can enter a question to the system in Dutch, either by typing or speaking. Both input modes can be combined with pen input (pointing or drawing). In a follow-up question, the user can use the mouse or a pen to encircle parts of a

---

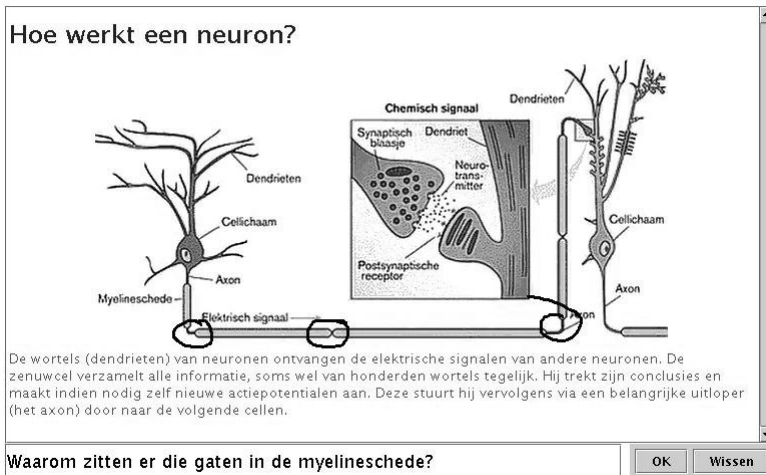
<sup>1</sup> A video presentation of the IMIX Demonstrator can be found on [www.youtube.com/watch?v=swA8\\_Y56aak](http://www.youtube.com/watch?v=swA8_Y56aak)



**Fig. 1** A snapshot showing parts of the user interface of the IMIX Demonstrator. The left-upper part shows the control GUI that comes with the MULTIPLATFORM. Boxes are system modules; they light up when activated. The right corner shows RUTH, the talking head. The buttons allow the user to select the input modality, speech or text, and to signal the start of a new dialogue.

picture when he wants to ask a question about it (see [Figure 2](#)). Follow-up questions may contain anaphoric references (“How do I recover from it?”) that need to be rewritten to form a self-contained question (“How do I recover from RSI?”) that can be handled by a QA engine. To a limited extent IMIX can handle utterances that are out of domain, in particular those that express how the user assesses the answer IMIX gave to a previous question. For the different types of utterances that IMIX can handle, consult the chapter by Van Schooten and Op den Akker, *Vidiam: Corpus-based Development of a Dialogue Manager for Multimodal Question Answering*, this volume.

A production experiment was carried out in the IMOGEN project to determine which modalities people choose to answer different types of questions. In this experiment participants had to create (multimodal) presentations of answers to general medical questions. The collected answer presentations were coded to types of manipulations (typographic, spatial, graphical), presence of visual media (i.e., photos, graphics, and animations), functions and the position of these visual media. The results of a first analysis indicated that participants presented the information in a multimodal way (van Hooijdonk et al, 2007).



**Fig. 2** An example of a multimodal follow-up question. At the top is the original question (translated: “How does a neuron work?”) and in the middle is the answer. In the box at the bottom is the user’s follow-up question (translated: “Why are there these holes in the myelin sheath?”). The circles in black are the mouse strokes made by the user.

The information assistant is personified in the form of a Dutch-speaking version of the Rutgers University Talking Head (RUTH<sup>2</sup>). In the Demonstrator shown in the movie on YouTube the Festival TTS system is used. Figure 1 shows a screen capture of the user interface of IMIX. Presentation of the information assistant by means of a talking head or an embodied conversational agent was not the focus of one of the IMIX research projects. It is well known that adding a character on the interface, it does not matter how primitive the figure is, has an impact on the way the user expresses their information requirement. If there is a face, users are more inclined to formulate full sentences (full questions) instead of only keywords, as they often do when the system lacks a human-like interface.

For the motivation behind the choice of the medical domain see Boves and den Os (2005), which also presents a short overview of an earlier version of the IMIX Demonstrator.

### 3 Architecture of the Final Version

This section describes IMIX’s architecture. The interaction with the user that it supports is best explained by the graphical user interface. Figure 3 gives a schematic impression of the various screens of the user interface of IMIX. The GUI starts with

<sup>2</sup> [www.cs.rutgers.edu/~village/ruth/](http://www.cs.rutgers.edu/~village/ruth/)

a welcome screen (1) with a Start button that takes the user to screen 2. At the start of a dialogue that screen shows some initial information and instructions. The user can choose to speak or type a question, or to stop IMIX.

The first time speech input is chosen, the ASR module needs to be initialised (screen 3). Then, the screen shows that the user can speak. If no speech is detected before a time-out, the GUI will go back to screen 2. Otherwise, the recognised speech is shown in screen 6, while the DAM is processing it. For speech input, the user can enter a question in screen 5, which will also end in screen 6.

When the Dialogue and Action Manager (DAM) and other modules have finished processing the user input and an answer has been generated, the GUI goes back to screen 2, but now showing the answer rather than the start-of-dialogue text.

Now, when the user chooses to speak or type a new question, the last answer remains visible and the GUI allows the user to draw (encircle, underline) in the answer. It is also possible to only draw, without any textual input.

In screen 2, the user can start a new dialogue at any time. IMIX includes the function to let the user fill in a survey after completing a dialogue (screen 7). The dialogue act options shown in screen 2 are a function to facilitate development of the DAM, and are not visible in the final Demonstrator.

IMIX is implemented using MULTIPLATFORM, an integration platform based on a general framework for building integrated natural-language and multimodal dialogue systems. MULTIPLATFORM was developed by DFKI in the VerbMobil and the SmartKom (Wahlster, 2006) projects and was used in COMIC, a European project on multimodal dialogue systems (Catizone et al, 2003) to develop integrated system prototypes (Herzog et al, 2004). The framework supports the modularisation of the dialogue system into distinct and independent software modules to allow maximum decoupling of modules, a facility that is urgently needed in a project like the IMIX project, with largely autonomous partner projects that deliver their modules for the Demonstrator. Communication between distributed software modules can be implemented in various languages and on distributed machines. This is realised by a publish/subscribe approach. The sender publishes a notification on a named message queue, so that the message can be forwarded to a list of subscribers. This kind of distributed event notification makes the communication framework very flexible as it focuses on the data to be exchanged and it decouples data producers and data consumers. A named message queue is called a data pool. Every data pool can be linked to an individual data type specification to define admissible message contents. The platform includes a control GUI that shows the activated modules and their connections (see the left-upper part of [Figure 1](#).)

[Figure 4](#) shows a detailed architecture of the last version of IMIX. Rounded rectangles denote data pools of MULTIPLATFORM, the grey boxes are modules. Shared data stores, ontologies, encyclopedia and image databases are shown in the bottom right of the picture. The next section introduces the functional modules of the architecture.

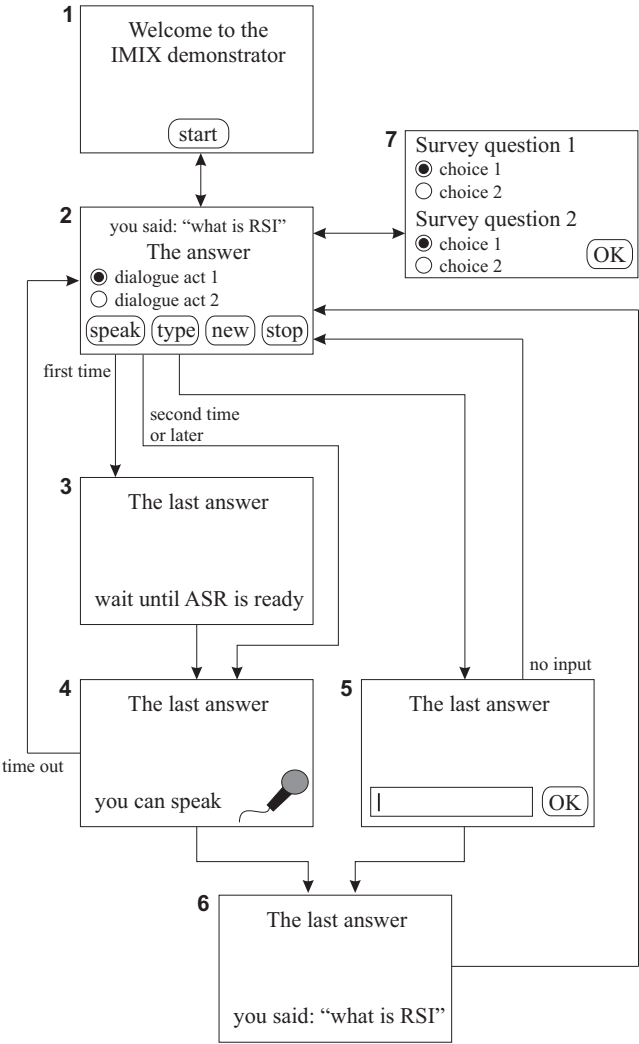
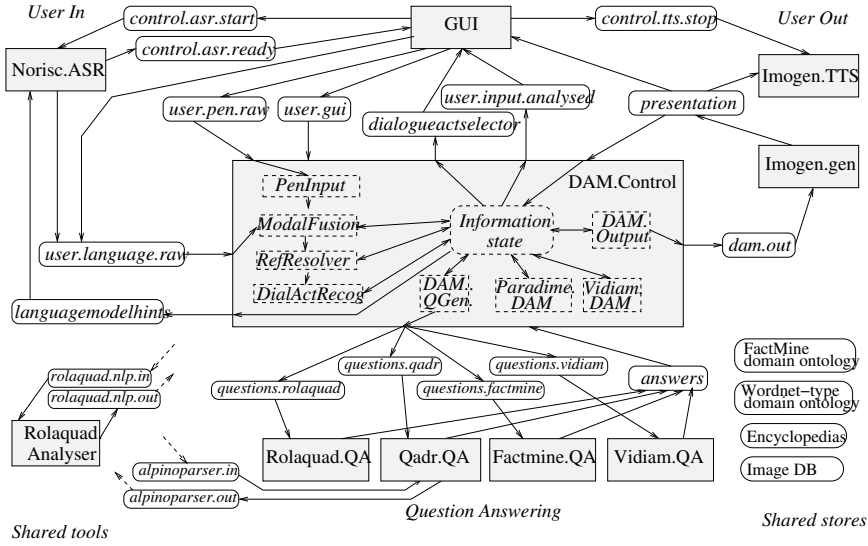


Fig. 3 Schematic overview of the various screens of the user interface.

3.1 The Modules

The main module is the Dialogue and Action Manager, DAM.Control. It has connections with all other modules, the GUI, ASR, the four QA modules, TTS and GEN.



**Fig. 4** A detailed architecture of the third and last version of the IMIX Demonstrator.

**DAM.Control.** DAM.Control consist of several submodules that process the multimodal pen/text/speech input data in the order shown in the architecture: PenInput, ModalFusion, RefResolver and DialActRecog. They communicate via the Information State, which holds the information about the dialogue acts. Feature structure unification is used for completing elliptic follow-up questions and for solving anaphoric references by the RefResolver.

PenInput matches pen actions with annotated visual areas. It can handle visual areas and aggregates of areas, arrows, and underlining. The module was tested with the multimodal follow-up question corpus and annotated images with good result (88% of identifiable visual elements found). The RefResolver resolves visual referents using the input from PenInput, combined with properties of the visual elements on the screen (in particular, colour, shape, name, and function).

DialActRecog tries to classify the type of speech act that the user performs (see the Vidiam chapter for the various types of acts that are distinguished.)

After the analytical steps the DAM module decides what system action will be executed next. Paradime DAM and Vidiam DAM are interchangeable for this function. The chosen system action is either a question to be sent to the QA engines or another dialogue act to be sent to the user directly. The output is sent to DAMOutput.

**Norisc.ASR.** ASR is provided by the Norisc.ASR module, based on HTK (Young et al, 2006). It is only used when speech input is switched on. The ASR acoustic models and language models are trained for the specific RSI subdomain to get maximum performance.

**QA modules.** There are four QA modules. They all share the same interface with the DAM: question in and retrieved results out. Rolaquad.QA is based on machine learning and semantic tagging of words using an ontology of medical terms. QADR is a QA system that exploits full parsing of the document and questions (Bouma et al, 2005). It uses the Alpino dependency parser. FactMine.QA is based on an unsupervised learning method. Vidiam.QA uses a simple TF.IDF based search method.

**Imogen Output.** The QA engines return a ranked list of text snippets, usually consisting of one sentence, that should contain an answer to the question. The DAM passes this list to the Imogen module, which generates the answer text that is presented to the user. It decides which of the potential answers to use and how to extend this ‘core’ answer so that the output is a comprehensive and verifiable answer to the question. TTS runs with the talking head RUTH in one module.

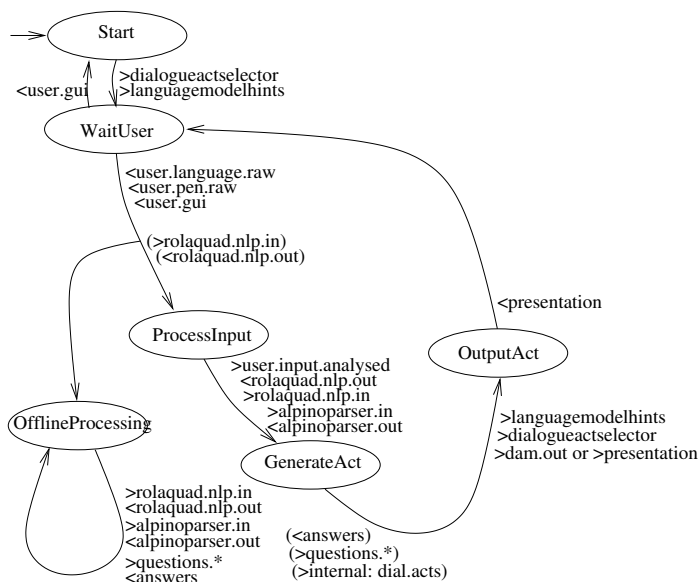
### ***3.2 The DAM State Machine***

The DAM state diagram (see [Figure 5](#)) describes the sequencing of messages as input and output by the DAM. This largely determines the control flow of the entire application, as all modules except the GUI simply reply to the DAM’s messages. In WaitUser, the DAM waits for the GUI to pass the user utterance. Once the GUI messages are received, the system goes to ProcessInput. Here, the DAM calls the appropriate taggers and analysers to analyse the input as necessary, after which the GenerateAct state is reached. Then it calls the QA engines if necessary, and generates the output, after which it reaches OutputAct. As soon as Imogen.gen has finished generating the final presentation, the cycle repeats. There is a special state called OfflineProcessing, which is active when an “offline processing” flag is set at startup. This enables the DAM components to be tested using offline data.

### ***3.3 A Modular Version of the Demonstrator***

The version of IMIX implemented on MULTIPLATFORM runs on a Linux machine. This version did not allow remote access. To make the IMIX system accessible for multiple users a modular multi-client server configuration was developed. The Dialogue Manager module and the GUI now run on the client machine that is connected with the MUP server via TCP. Various TTS or ASR systems can be connected to the client application.





**Fig. 5** The DAM state diagram. The ellipses represent the states, the arrows the transitions. Alongside the arrows are the pools that are read from or written to. The `<` sign indicates reading, `>` indicates writing. The order in which the operations are shown is the order in which they take place. Read operations block the DAM until the message is received.

## 4 Conclusion

The IMIX project developed a working system that satisfies its functional requirements for demonstrative purposes. The Demonstrator also makes clear where the missing bits are, that stand in the way of a practical multimodal information search assistant. Therefore, good and fast retrieval systems for multimodal information are the least that is needed, a topic that was not properly covered by the IMIX Programme. This requires automatic recognition of visual elements and automatic annotation of parts of illustrations, as well as research into how text and visual elements are connected in media. The annotations are also needed for the user interacting with the system through pointing. More research is needed to see how people refer to visual elements and how this is related to their speech, and the dialogue context. More research is needed in generation of speech, gestures and expressions (audible and visible) of the mental (cognitive, affective) state of the assistant to make the assistant a more believable character, and give it a more sophisticated face. And finally, ASR and dialogue researchers should work more closely together to come up with improved models of dialogue and context dependent speech recognisers. Domain knowledge is a prerequisite for a practical system, that will always be focussed on a specific domain. It will be clear that

for a dialogue system with “natural” turn-taking not only effectiveness of these technologies is important; they must also be fast.

This chapter concludes with a few words about the various roles the Demonstrator played in the project.

Demo systems definitely have a function in the relationship between the research community and the broader public. For example, in engaging interested high school students in research in a particular area of research and technology. People generally like to see something demonstrable. According to the self evaluation report by the Programme Committee “the IMIX demonstrator was never meant to be an early version of a prototype of an operational application. On the contrary, the goal of the common demonstrator was to provide hard and fact evidence of the capability of the research teams funded through IMIX to collaborate on a level that would allow the integration of most of the results in a comprehensive system. The possibility of using the common demonstrator as a means for attracting attention from the general public and thereby increasing the interest in applications of language and speech technology for commercial and social purposes was only an ancillary goal.” Clearly, the Demonstrator had an *internal* function: to have partners from different disciplines collaborate, not just present their work in a joint workshop, but have them deliver their software in a way that “allows integration in a comprehensive system.” Certainly in the first half of the project period the regular Demonstrator meetings had a positive impact on the communication between the research groups. Towards the end of the project there were clearly other priorities, PhD students had to finish their thesis and postdocs had to apply for other jobs. The second internal role of the Demonstrator was that of a research subject itself. It is clear that for some of the subprojects in IMIX the Demonstrator was more important than for others. For the Vidiam dialogue manager project a working Demonstrator was the only way to evaluate the dialogue manager as a complete system. The architecture of the Demonstrator had a strong impact on the DAM.Control module. The QA engines and the DAM.Control modules are now loosely coupled: the DAM sends a question to the QA engine and gets the results from it. This reflects the “modular” structure of the project as a whole, with its highly autonomous subprojects. Ideally the DAM-QA interface would be much broader and they would even share the dialogue information state, to properly handle follow-up questions. The same is true of the connection between the ASR and the dialogue modules. Ideally, the speech recognition is informed by the dialogue state which induces prior word expectations.

The Demonstrator software was hardly used for research, for example to collect dialogues. This is also the case with the individual modules. Most collaboration between subprojects was not related to the Demonstrator, but to the use of supplementary resources, such as NLP tools (for example the use of the Alpino parser) and shared data, such as the collections of annotated questions and annotated pictures. All in all, the conclusion is that the Demonstrator itself did not work as a research tool. So, was the Demonstrator a good idea? It served as a collaboration tool, but there may be more efficient methods.

## References

- op den Akker HJA, Bunt HC, Keizer S, van Schooten BW (2005) From question answering to spoken dialogue - towards an information search assistant for interactive multimodal information extraction. In: Proc. 9th European Conference on Speech Communication and Technology (Interspeech 2005), Lisbon, Portugal, European Speech Communication Association (ESCA) / CEP Consultants, pp 2793–2796
- Bouma G, Mur J, van Noord G, van der Plas L, Tiedemann J (2005) Question answering for dutch using dependency relations. In: Working notes for the CLEF 2005 workshop, Springer
- Boves L, den Os E (2005) Interactivity and multimodality in the imix demonstrator. IEEE International Conference on Multimedia and Expo 0:1578–1581
- Catizone R, Setzer A, Wilks Y (2003) Multimodal dialogue management in the COMIC project. In: Proceedings of the EACL 2003 Workshop on Dialogue Systems: Interaction, Adaptation, and Styles of Management, Budapest, Hungary
- Herzog G, Ndiaye A, Merten S, Kirchmann H, Becker T, Poller P (2004) Large-scale software integration for spoken language and multimodal dialog systems. *Natural Language Engineering* 10 (3/4):283–305
- van Hooijdonk C, Krahmer E, Maes F, Theune M, Bosma W (2007) Towards automatic generation of multimodal answers to medical questions: a cognitive engineering approach. In: Proceedings of the Tenth International Symposium on Social Communication
- Wahlster W (ed) (2006) *SmartKom: foundations of multimodal dialogue systems*. Springer
- Young S, Evermann G, Gales M, Hain T, Kershaw D, Liu XA, Moore G, Odell J, Ollason D, Povey D, Valtchev V, Woodland P (2006) *The HTK Book*. Cambridge University Engineering Department

Interactive Multi-modal Question-Answering

van den Bosch, A.; Bouma, G. (Eds.)

2011, XII, 280 p., Hardcover

ISBN: 978-3-642-17524-4