

An Architecture for Integrating Virtual Sales Assistant Technology into E-Business Environments

Thorsten Gurzki

Fraunhofer Institut fuer Arbeitswirtschaft und Organisation (IAO)

Nobelstr. 12, 70569 Stuttgart, Germany

Phone: +49 711 / 9 70 23 49; Fax: +49 711 / 9 70 23 00

Email: thorsten.gurzki@iao.fhg.de

Abstract. Major challenges of virtual sales assistant research lie in enhancing the quality of the customer consulting and in guaranteeing efficient maintenance of the product data. Current concepts lack of tight and intelligent integration options into existing business systems (like shop systems), and they lack of possibilities to reuse existing product data. The missing of integration concepts decreases the quality of the dialogue and make most scientific approaches invaluable for business use. This paper presents a virtual sales assistant system architecture that meets the challenges of tight backend system integration.

1. Introduction

Virtual sales assistant systems with natural-language text dialogues are a broad field of research. Most systems focus on the efficiency of the assistant system generating the dialogue and have limited capabilities for the integration into existing business systems. The systems are either standalone solutions which cannot be integrated or require the existing business environment to be adapted to the assistant system. There are different reasons for the lack of integration capabilities. Many systems rely on complex knowledge models which require special software tools and manual work for changing both model and data. Another reason is the approach of the systems as a proof of scientific concept which is not directly designed for business use. The requirements of companies using these technologies in operative environments are clearly defined: product data must not be maintained separately in a virtual sales assistant system, and the systems must be seamlessly integrated into existing business processes. In nearly all cases the assistant is one piece of software in a business scenario with hundreds or even thousands of other software components. Many of these components are primary components like Enterprise Resource Planning (ERP) systems or shop systems, ensuring the core operability of the company. Secondary components provide additional functionality which is not necessary for the company's core operability. Secondary components must not disturb or endanger the operability of the primary core systems, in which important data and information is stored and maintained. A virtual sales assistant must therefore be integrated into the core systems. This paper uses the scenario of the IST project ADVICE (Virtual Sales Assistant for the Complete Customer Service Process) as described at [1]. The overall objective of the project is to create a natural-language based system for customers accessing the commerce system with a problem-centric approach in contrast to a product-oriented

approach of existing solutions. The system consists of highly specialized components using different programming languages starting from C++ up to JAVA and PROLOG. These components build agents, form an agent society and exchange speech acts as inter-agent communication. This high-level approach is necessary for achieving a high-quality text-based customer dialogue and an output by an animated virtual sales assistant able to produce gestures. The given challenge was to build an architecture which can be tightly integrated into existing business environments.

The paper will first describe an agent-oriented view on ADVICE, then discuss the architecture of the system and the integration of the agent approach into business environments, and finally present an integration into a shop system.

2. State of the Art

Existing virtual sales assistant systems based on agent-oriented technology are research systems and in most cases not open for the integration into third-party systems. They focus on the improvement of the dialogue and the algorithms. Commercially available systems are commonly not agent-oriented (e.g. WebSell [2]) and in many cases use database integration into third-party systems (e.g. the ALife-WebGuide [3]). Currently, there is a gap between agent-oriented research approaches and commercially available systems which can be integrated easily into business environments.

3. The Agent's Point of View

The ADVICE system contains a multi-agent system for the processing of the natural-language text-based customer dialogue (fig. 1). The customer types the natural-language text in the textbox of the animated 3D assistant. The input is sent to the Interface Agent for analysis. The Interface Agent contains a natural-language analyser module which generates speech acts [4] from the user utterance. The speech acts are represented using XML, based on the corresponding document type definition. The speech acts are transmitted to the Dialogue Processing Component (DPC) consisting of the Interaction Agent and the Intelligent Agent. The DPC generates the answer to the question which is represented on a semantic level in speech acts.

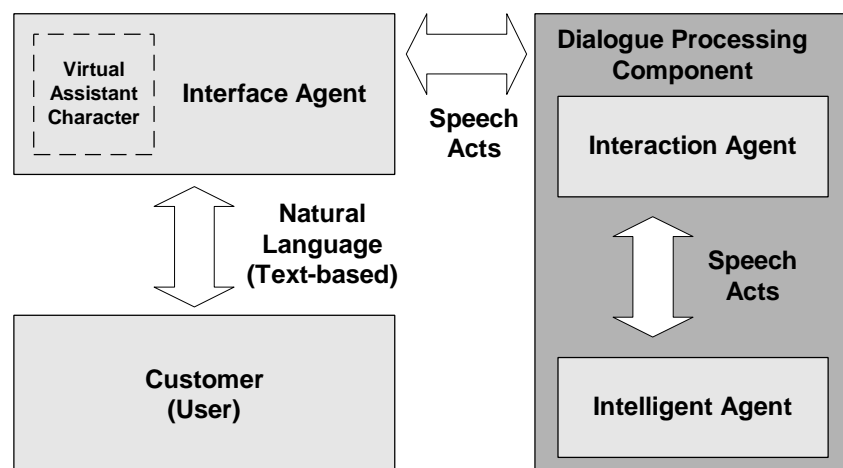


fig. 1: agent architecture of the ADVICE system

The answer is also generated in a semantic representation form. These speech acts are transmitted back to the Interface Agent where natural language is generated from the speech acts. The natural language must be enriched with additional information to enable

the virtual assistant character to perform a product presentation like presenting products using the web pages of a store and making suitable gestures. The information about which additional information is required is directly derived from the semantics stored in the speech acts according to this answer. Thus the information which is presented to the customer consists of verbal and non-verbal information. ADVICE uses a highly specialized agent approach compared to general multi-agent system approaches which prevents the usage of development tools like the ROPE agent-oriented software engineering environment [5]. The Dialogue Processing Component (DPC) of ADVICE is described in detail in [6] and [7].

4. Integrating Architecture

One objective of the ADVICE project is to design an innovative architecture (fig. 2) for a virtual sales assistant system which allows a tight integration into other systems and which takes into account business user requirements. Meeting these requirements is important for the efficient transfer of scientific results to business applications.

4.1 Client Communication

ADVICE can either be accessed by the native JAVA 3D assistant client (fig. 4) or the Wireless Application Protocol (WAP). Both access the system using HTTP. The native client uses a XML-based representation for data exchange with the server. The use of HTTP allows communication with nearly all customers even if a firewall is used with standard security options. On the sales side, the commerce infrastructure can be protected by a firewall. This allows an integration of ADVICE without changing security policies at the company.

4.2 Agent Integration

The agents and their subparts are implemented in different programming languages. The first prototypes of ADVICE used TCP/IP connections for exchanging speech acts in an XML representation. For a commercially available system, the interfaces have to rely on more abstract and stable interfaces. Thus the recent design uses the JAVA Remote Method Invocation [8] for transmitting XML speech acts. The design adheres to transmitting XML due to the easy enhancement possibility without changing the system's interfaces. The question here was whether to use RMI or the standard CORBA [9]. More and more business systems are built using the JAVA technology, and the advanced CIAO PROLOG system [10] used in the project is able to be encapsulated in JAVA. Considering these aspects, RMI is used for interaction. Another important aspect is the centralized control of the distributed components. Business environments need a single command centre for applications which provides the ability to start and shut down components, to change parameters and access the logs.

4.3 Interface Agent and Presentation Manager

The Interface Agent acts as the interface between external data sources and the agent system. The Interface Agent contains a Presentation Manager which generates the presentation of products or product application examples for the client. This presentation contains the utterance of the assistant (e.g., the information that a specific product is suitable for the customer), gestures and a web page with the desired information (e.g., a detailed description of the product). In order to offer the best suitable presentation, the

Presentation Manager analyses the speech act provided by the Dialogue Processing Component to get information on the semantics of the speech. This process is similar to the generation of the natural-language utterance of the assistant. Detailed descriptions of products are not maintained within the virtual sales assistant system. The description is derived either from a shop system or – in case of an application example – a content management system. A major decision of the ADVICE project was to rely on the functionality and the data available in a shop system and not to reinvent the functionality of a shop solution. Shop systems usually provide a broad range of functions for setting up commerce sites. The interesting functionalities for the ADVICE project are: product catalogue system, templates for product presentation, basket function and user data management. Catalogue information creation and maintenance is an expensive task. Companies therefore like to reduce the costs by having only one single point for catalogue storage.

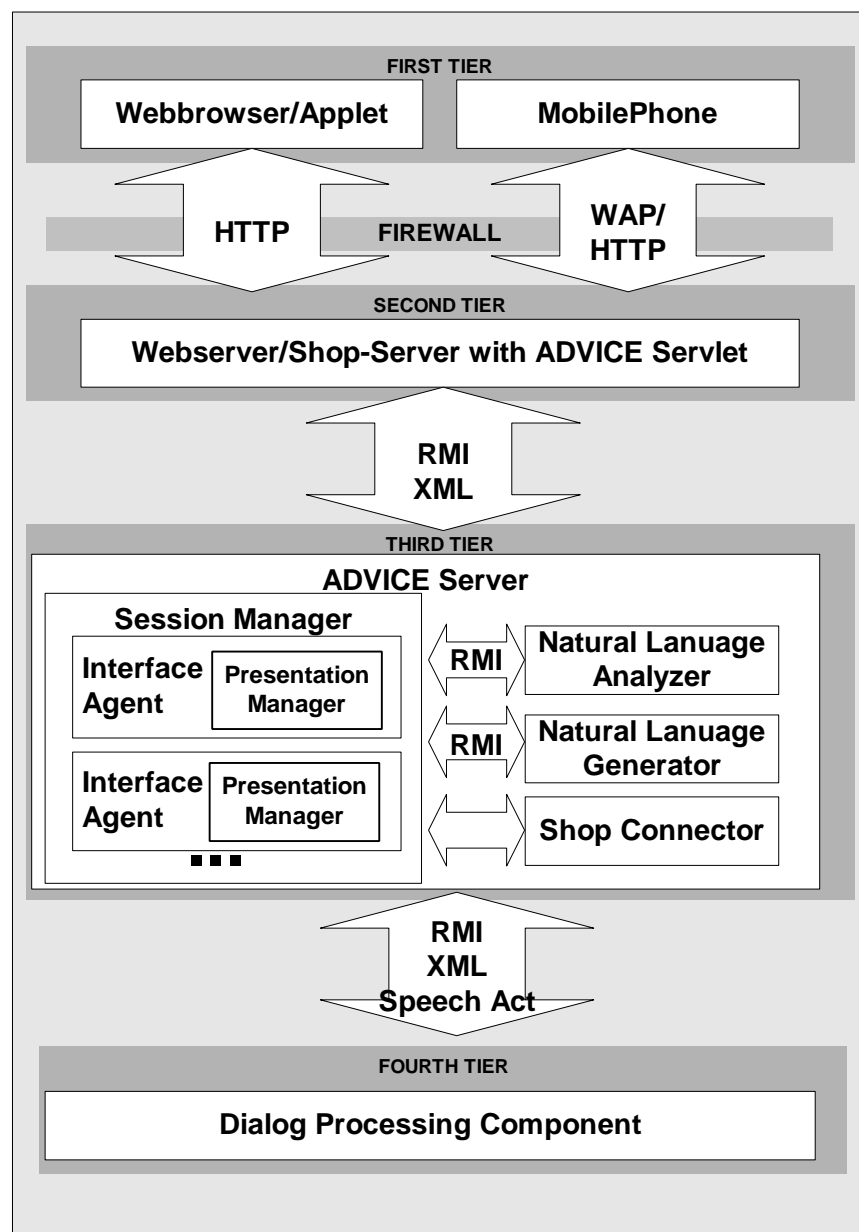


fig. 2: ADVICE architecture

In a commerce environment, this information is most likely available at the shop system. The information of this system is also available to the customer. The Presentation Manager

contacts the shop system and gets back a unified resource locator, which points to the appropriate web page. The shop-system specific algorithms which retrieve the information are encapsulated in the Presentation Manager. Customer information is also stored in the shop system. The Presentation Manager accesses the user data to get information on the name for addressing the user and the personal profile (like occupation, position in the company etc.) to offer the appropriate products according to this user profile. Already available information about the user will be reused with this data exchange. To be open for a later ADVICE product customisation, each Interface Agent is registered with the corresponding session identifier (which is the session ID of the shop system) in the RMI registry. Applications knowing the identifier can access the Interface Agent and read or write data to this session. If a shop system is able to execute code while generating a web page from a shop page template, it might inform the corresponding Interface Agent of the products which are currently shown in the template. This mechanism allows the user to interact either with the shop system, the assistant or with both. The ADVICE assistant always remains informed of the user action even when it is temporarily switched off by the user. The Interface Agent presents the text output and the gestures using a 3D animated sales assistant. The appearance of the assistant can be adapted to different business scenarios, for example, the use in service portals (for service portal requirements refer to [11]), and to different types of users with different cultural background as described in [12], [13] with different appearance and functionality [15].

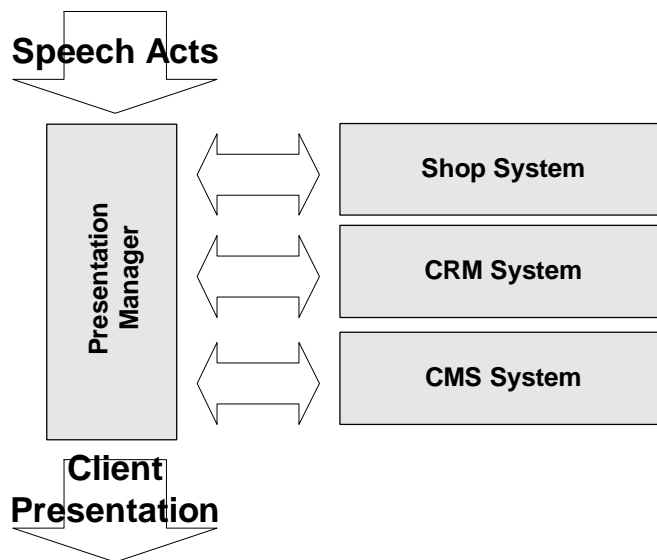


fig 3: Function of the Presentation Manager

5. Example: Intershop Enfinity and SAP R/3 Integration

For test implementation and technical demonstration, an Intershop Enfinity System ([15], [16]) is used. Enfinity is using pipelines for generating content for the shop web templates. Pipelines consist of pipeline elements, called pipelets [17]. Pipelets perform tasks like accessing the catalogue system or third-party systems. The pipelines can be arranged using a visual editor. Pipelines therefore can easily be changed. Since Enfinity makes intensive use of JAVA technology, the pipelets have to be implemented using JAVA. ADVICE integrates itself by providing ADVICE pipelets which inform the Interface Agent of the products that are currently shown to the user by the shop system. The ADVICE pipelets read the internal data structures of the shop system, extract information and send it to the Interface Agent using RMI. Since the RMI object reference is the identifier of this session

in the shop, the pipelet has all the information needed to access the object. To enable ADVICE to present information on a specific product, a template and the corresponding pipeline have been created. The pipeline contains a pipelet which accesses the Interface Agent according to this session. It receives information on what product ADVICE wants to show to the user and fills the shop web template with the required information from the shop's catalogue. For availability checking an enterprise resource planning system is involved. In the test environment a SAP R/3 system [18] is used. Though the architecture would allow a direct coupling, the SAP R/3 is not directly coupled with the ADVICE system. Instead the R/3 system is coupled with the shop system, and ADVICE may request information from the shop system. This approach reduces the amount of different interfaces to the R/3 system and enables fast changes in the interface, for example, an introduction of a data cache between internet systems and the SAP system. The update agent (not shown in fig. 2) updates the knowledge model of the system with the latest product data. The agent itself is a modified Interface Agent, which is running continuously. The agent is only able to update the product data, or, in terms of knowledge-based systems, to change or create product facts. These product facts must comply with the knowledge model. If a new product category is introduced, the model must be manually extended and/or changed since the sale strategies and extended consulting information for this product must be added. The update agent is triggered by a scheduled pipeline in the Intershop Enfinity system. The pipeline sends the updated product data to the update agent, which sends the data to the knowledge model maintainer.

6. Conclusions

The ADVICE architecture allows the tight integration of agent-oriented virtual sales assistant technology into existing business environments. The use of agent technology enhances the quality of the dialogues and therefore the satisfaction of the customer. The architecture contains components (like the Presentation Manager) which build a bridge on a semantic level from agent-orientation towards a business view. The presented architecture is open for the integration of third-party systems including secure integration into mission-critical systems like ERP or shop systems. The tight integration reduces the costs for operating the system since the required data for the system (e.g., product data, user data) are stored in the existing business systems. The system has been tested in a prototype environment. In this environment the architecture has proven itself as a stable and good performing design with an easy set-up. The next step of the project will be the set-up of pilot applications on the sites of the user members of the project consortium. The required time for setting up the pilots is expected to be short due to open interfaces.



Fig. 4: User Interface

Acknowledgements

The ADVICE project is funded by the European Commission under IST-1999-11305. The consortium members of the ADVICE project are: Technical University of Madrid (UPM) ISYS, Spain; FESTOOL SRL Italy; FESTOOL LLC, USA; Pixelpark (Schweiz) AG, Switzerland; Universität Stuttgart IAT, Germany and Fraunhofer IAO, Germany.

References

- [1] ADVICE project Website; <http://www.advice.iao.fhg.de>
- [2] Cunningham, Pádraig; Bergmann, Ralph; Schmitt, Sascha; Traphöner, Ralph; Breen, Sean; Smyth, Barry; WEBSELL: Intelligent Sales Assistants for the World Wide Web; in KI-Künstliche Intelligenz 01/01, Gesellschaft für Informatik, arenDTaP, Bremen, Germany, 2001
- [3] Artificial Life Website; <http://www.artificial-life.com>, 2001
- [4] Smith, Barry; Towards a History of Speech Act Theory ; in: Speech Acts, Meanings and Intentions. Critical Approaches to the Philosophy of John R. Searle, Berlin/New York: de Gruyter 1990, p. 29-61
- [5] Gurzki, Thorsten; Becht, Michael; Klarmann, Jürgen; Muscholl, Matthias; ROPE: Role oriented Programming Environment for Multiagent Systems, Programming Environment for Multiagent Systems, in: Proceedings of the Fourth IFCIS Conference on Cooperative Information Systems (CoopIS'99)
- [6] Garcia-Serrano, Ana Hernandez-Diego, Josefa; Martinez, Paloma; Intelligent Assistance to E-commerce; in: Proceedings of the E-business and E-work Conference 2001 Venice, Italy, 2001
- [7] Garcia-Serrano, Ana Hernandez-Diego, Josefa; Martinez, Paloma; Una Propuesta para una Interaccion Usuario-Sistema Avanza en Aplicaciones de Comercio Electronico; in: Proceedings of the Informatica Conference 2000, 2000
- [8] Sun Microsystems, Java Tutorial Trail: RMI; <http://java.sun.com/docs/books/tutorial/rmi/>
- [9] Brose, Gerald; Vogel, Andreas, Duddy, Keith; Java Programming with CORBA: Advanced Techniques for Building Distributed Applications; OMG Press
- [10] M. Hermenegildo, F. Bueno, D. Cabeza, M. Carro, M. García de la Banda, P. López-García, G. Puebla. The Ciao Logic Programming Environment. International Conference on Computational Logic, CL2000, July 2000.
- [11] Gurzki, Thorsten; “Beyond Transaction: Serviceportale“, in: Proceedings of the Fraunhofer IAO Forum “Elektronische Marktplaetze und Serviceportale”, Fraunhofer IAO, Stuttgart, Germany, 2001
- [12] Prabhu, G., Harel, D., “GUI Design Preference Validation for Japan and China – A Case for KANSEI Engineering ?”, Human-Computer Interaction: Ergonomics and User Interfaces, Proceedings of the 8th International Conference on Human-Computer Interaction 1999, Hans-Jörg Bullinger, Jürgen Ziegler (editors), 1999, p. 521-525
- [13] Nissler, Joerg; Machate, J.; Hitzges, Arno; “How to get the Right Outfit for My Agent? Classification- and Design Methodology for a Virtual Shopping Assistant in a 3D World“, in: Information, Communication and Cooperation Interfaces, ID Volume 2, August 1999, p. 162-167
- [14] Nissler, J.; Thoma, V.; Appearance, Features and Characteristics of Software Agents from the user’s perspective; Software-Ergonomie 99; Conference Proceedings SAP, Walldorf, Germany, 1999
- [15] Intershop Communications Inc. Website; <http://www.intershop.com>
- [16] Intershop Communications; Inside Intershop Enfinity, Intershop Communications GmbH, Jena, Germany, 2000
- [17] Intershop Communications; Developer Guide: Programming with Enfinity Cartidge API, Intershop Communications GmbH, Jena, Germany, 2000
- [18] SAP AG; www.sap.com