

Agent-Based Information Management in an Emergency Room

Martin FRANCK, Matthias GÖBEL, Wolfgang FRIESDORF

Berlin Technical University, Department for Ergonomics and Human Factors Engineering, Berlin, Germany

Abstract: Clinical care is characterized by the distribution of data among personnel, locations, and various types of media. Clinical information systems (CIS) should provide help in collecting, preparing and presenting data. The dispersal of information emerges as a problem in clinical information systems especially when decisions have to be made with emergency patients.

The primary goal of the research project AGIL (agent based information logistics)* is to develop a multi-agent system that supports the clinical information flow. The secondary goal is to explore and generalize an agile development methodology for such multi-agent systems. A tool for the modeling of processes and the development of the software is developed.

Keywords: Emergency Room, Clinical Information Systems, Process Modeling, Agile Development, Multi-Agent System.

1. Introduction: Patient Care in an Emergency Room

Patient care in an Emergency Room (E.R.) setting is characterized by cooperation of several clinical disciplines and the processing of different data (Knublauch, H., Rose, T., and Sedlmayr, M., 2000). The E.R. physician is in charge of coordinating these different disciplines, tests and treatments and creating the plan while acting as the advocate of the patient.

The adequate information at the right place and time is essential in an efficient patient flow. All personnel involved in the care of the patient need to be able to access their specific information in a mobile manner and need to be able to interchange the information for a continuation of care as the patient is moving to different places. This is especially true for a patient that needs emergent treatment.

Although nowadays clinical data systems and paperless records are more and more common, they are in general passive and not very flexible. It is not only very time consuming to actively look up data, but important information might also be ignored. In heterogenic clinical information systems that do exist, often each department uses their own software that is adapted to their respective needs. A switching to another system is often not possible and so it is necessary to integrate the data from various sources. A problem might be the access to these different databases for security reasons.

Support facilities for seamless information flow and proactive delivery services are instrumental to improve the efficiency and effectiveness (Lanzola, G., Falasconi, S. and Stefanelli, M., 1995). The research project AGIL (agent based information logistics in the emergency room) and its second phase project AGIL², is aimed at assisting the personnel of an emergency room by optimizing clinical information flows.

* supported by German Research Foundation (DFG) Project number FR 1364/1-1

2. Software-Agents as a Support for the Information Flow

Software-agents are a modular software that are able to work independently and to interact with each other. They are able to move around within a distributed environment on different platforms, to collect data, and proactively present the information to the personnel taking care of the patient. Each agent provides access to a subset of these resources and can serve as an information source to other agents (Lanzola, G., Falasconi, S. and Stefanelli, M., 1995). The coordination mechanisms found in multi-agent systems offer high flexibility, e.g. when unexpected emergency patients need to be treated. Compared to conventional (client-server) architectures, agents represent a more natural mapping of the clinical distribution. Multi-agent systems are inherently distributed and decoupled, so that the overall system remains functional even if single agents fail.

3. Process Analysis and Agent Design

If the application scenarios of agents are unknown in the beginning of the project, domain experts and developers must at first perform a process analysis to detect process bottlenecks in order to detect areas that benefit most from support (Friesdorf, W., Groß-Alltag, F., Konichezky, S., Arndt, K., 1994). The existing processes were modeled with support of a Java-based tool called AGIL-Shell, that was developed in the first phase of the project. This tool can be used for the design and implementation of multi-agent systems. Our approach consists of three steps:

1. Domain experts build a model of their existing working processes
2. These existing processes are analyzed to detect application scenarios of agents
3. The processes are incrementally supported by introducing agents into them

Based on the existing process, in collaboration between engineers and clinical experts an “agentified” process is designed, in which agents fulfill previously human routine tasks. The human actors in the original process were equipped with interface agents, and as now humans were represented by their individual interface agents these human actors were removed from the design model in order to reduce complexity. Since the workplace of anesthesiologists and other staff is not bound to a certain location, mobile information services on small but functional devices such as PDAs are required. (Kirn, S., 2001, Orphanoudakis, S. C., 1998).

The code for these “agentified” processes can be generated with the help of the AGIL-Shell. The process metamodel was designed to be comprehensible and easy to use by end users of the agent application, to be extensible for specific types of agents, and to allow automatic and semi-automatic transformation into executable code. Figure 1 illustrates AGIL-Shell, that provides various graphical views of the processes and agents.

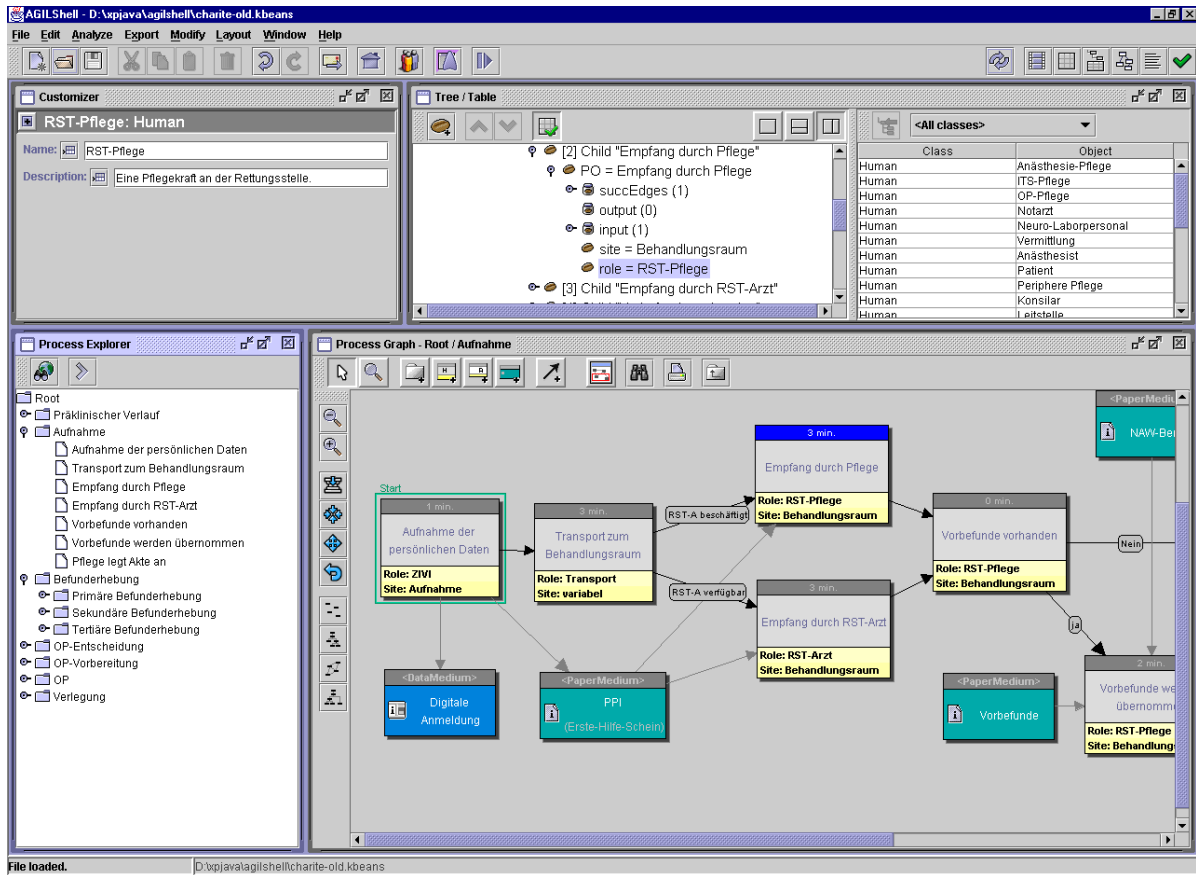


Figure 1: A screenshot of the process and agent modeling tool AGIL-Shell.

Process Graph. The main view of AGIL-Shell is a graph in which domain experts and engineers can define processes by arranging activities and media as nodes, and by representing the media flow and the sequence of activities by means of arrows. For each activity, its name and the performing role is visualized. For each agent message, the command and content ontology are shown.

Process Explorer. Complex process models were represented and visualized using a hierarchical order of processes and sub-processes in a logical tree view.

Customizer. The properties of the currently selected activity, process, message, agent or ontology class can be edited in a so called customizer.

Community Viewer. This graph (not shown in this paper) displays the communication pathways between the performing roles (human technical, actors etc.) in the system. The agents are shown as nodes which also display the role type (human, interface agent, etc.). This view allows to detect information bottlenecks. Nodes with many incoming and outgoing arrows represent roles that operate on a large accumulation of data. These focal points of information might benefit from agent-based information services.

Besides these views, other types of graphical visualization of (clinical) processes can be derived from the metamodel. For example, the sequence of activities allows to extract a life-cycle view in which only those activities that are performed by a given agent or human are shown. Such views allow to cross-check the design with the human process participants, who can – from their local point of view – assess whether their daily routine is sufficiently covered by the overall process.

4. Agile Development of Software

The complexity of communication scenarios between agents make multi-agent systems difficult to build. Most of the existing Agent-Oriented Software Engineering (AOSE) methodologies face this complexity by guiding the developers through a rather waterfall-based process with a series of intermediate modeling artifacts. While these methodologies lead to executable prototypes relatively late and are expensive when requirements change, in our project a rather evolutionary approach with explicit support for change and rapid feedback is explored. In particular, eXtreme Programming (XP), a modern agile methodology from object-oriented software technology, for the design and implementation of multi-agent systems is applied. Agile Modeling (AM) and eXtreme Programming are explicitly very user-centered and thus can achieve a higher acceptance rate and a more quicker development. It is an approach that is adaptive and able to address the specific needs of software development in the face of vague and changing requirements. This approach relies on process modeling to capture and clarify requirements, and to enable communication with domain experts. The only modeling artifacts that are being maintained in this approach are a process model with which domain experts and developers design and communicate agent application scenarios, and the executable agent source code.

5. Discussion and Conclusion

The modularity of agents allows one to customize a clinical information infrastructure to a hospital's individual requirements and prerequisites. Existing agents and context-aware, proactive clinical user interfaces can be reused. Furthermore, the design of typical agent application scenarios can be reused, since recurring patterns can be identified among different hospitals. The approach to have the user actively involved in the development process improves the quality of the software. But as user and developers must work together throughout the project, it still has to be shown that the costly procedure is balanced by a higher user satisfaction and reduced maintenance costs. Besides the question arises to which degree the involved user is a representative for all the future users. As this user is involved in the development process, the level of acceptance might not be shared by users with a different background.

Another promising extension of this approach is to build algorithms that detect inconsistencies in the process models automatically, particularly to find scenarios in which an agent lacks access to data that are produced by another agent in previous process phases. In the medical domain security still remains an important issue, so that agents which change their behavior were not implemented in contrast to the common definitions of agents.

6. References

1. Friesdorf, W., Groß-Alltag, F., Konichezky, S., Arndt, K., 1994. Clinical information process units (CIPUs) – a system ergonomic approach to medical information systems. *Technology and Health Care*, 265-272.
2. Kirn, S., 2001. Agententechnologie-Kooperierende Softwareagenten im betrieblichen Einsatz. *Wirtschaftsinformatik*, 2.
3. Knublauch, H., Rose, T., and Sedlmayr, M., 2000. Towards a Multi-Agent System for Pro-active Information Management in Anesthesia. *Proc. of the Agents-2000 Workshop on Autonomous Agents in Health Care*, Barcelona, Spain.
4. Lanzola, G., Falasconi, S. and Stefanelli, M., 1995. Cooperative Software Agents for Patient Management. *Proc. of the 5th Conference on Artificial Intelligence in Medicine in Europe (AIME)*, Pavia, Italy.
5. Orphanoudakis, S. C., 1998. Healthcare Sector: The Wireless Vision. *Proc. of the Information Society Technologies Conference & Exhibition (IST'98)*, Vienna, Austria.