Conf. of the American Medical Informatics Ass., Washington, 1994

An Infrastructure for Cooperation and Communication in an Advanced Clinical Information System

K. Kuhn[1], M. Reichert[2], M. Nathe[2], T. Beuter[2], P. Dadam[2]
[1]Medical University Hospital, University of Ulm, Germany
[2]Dept. Databases and Information Systems, University of Ulm, Germany

ABSTRACT

In a research project, organizational and technological requirements for an advanced clinical information system have been analysed, and a concept has been developed. From the application's perspective, medical personnel is more actively relieved from routine tasks by support of organizational tasks and by coordination of distributed activities. Program development is supported by a concept of simple and complex services with well-defined interfaces, and by the use of activity templates, i.e. pre-modeled activities describing possible sequences of services. The concept is based on an open systems approach, with a reliable and secure communication infrastructure. In addition, monitoring facilities are provided.

INTRODUCTION

In the development of hospital information systems, several trends of increasing importance are emerging. Today, physicians confronted with a massive load of data, that have to be intellectually processed and structured, tend to make mistakes, such as overlooking rare events [1].

An advanced computer-based system should therefore increase the availability of patient specific information and present it in a structured way. Moreover, it should relieve medical personnel from tracking the state of medical activities like processing of an order, and from caring about the timely collection of information about the outcomes.

In addition, medical and organizational tasks are highly interrelated, especially if scheduling is involved. In typical activities, several parties at different sites interact at various points of time. From modeling organizational aspects inherent in medical activities, and from an active support of cooperation, positive effects can be expected.

Furthermore, physicians have to select tests and interventions under aspects like invasiveness, efficiency and costs. Here, system support of medical guidelines [2] and consequently the representation of medical knowledge inherent in complex medical activities will become increasingly important.

The outlined need of systems playing a more active role has to be seen in the context of general tendencies in hardware and software development, that can also be observed in the medical environment. The number of heterogeneous and decentralized application modules is increasing, while centralized solutions, which tend to be too static, are going to play a less important role [3]. This development is combined with an increase in application-oriented programming skills and fewer personnel with system know-how in computing centres. Therefore, an appropriate software engineering environment *for developing secure and stable distributed applications* is mandatory. In addition, a reliable communication framework with *sufficient monitoring facilities* is needed.

This paper presents first results of a research project, which is aimed at the systematic analysis of organizational and technological requirements for an advanced hospital information system. The basic concepts developed and an exploratory prototype will be described.

BASIC CONCEPTS

Distributed, heterogeneous systems are integrated by means of a *software bus* [4,5,6], which offers *services* to applications, and abstracts from heterogeneity and distribution (see Fig. 1). The concept of services with well-defined interfaces has been introduced to reduce the need of system know-how for application programmers.

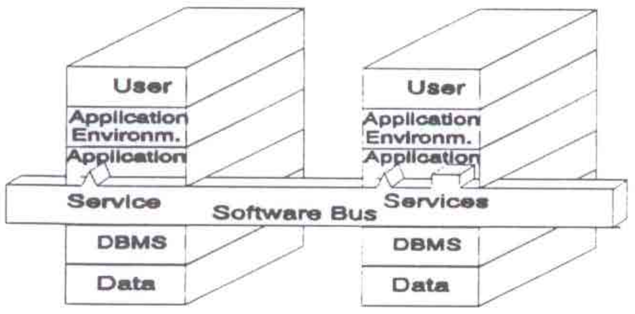

Figure 1. Simplified model illustrating the software bus [4].

Services are provided or utilized by application components called *agents* [7]. *Basic* services are executed by single agents. *Complex* services consist of several basic services executed in parallel or sequentially; they may be nested. Examples of complex services, which are typically handled by distributed agents, are the entry of an order into a local (e.g. ward) as well as into a remote (e.g. radiology) database, or the notification of a patient transmission to the local and to several remote (departmental, administrative) systems. To guarantee consistency, extended *transaction concepts* are used, if the two phase commit protocol is too restrictive [8,4]. For these cases, compensation methods necessary to undo each basic service have to be provided; these compensation methods are used for *semantic rollback* of complex services.

Organizational knowledge inherent in clinical tasks is being modeled under the *concept of medical processes or activities*, which are executed by several distributed agents over time. To *actively support and coordinate* the execution of typical activities like processing of an order (from order entry to transmission of the results, possibly including scheduling), or patient transmission, activity templates supporting the coordinated execution of services are used. Activities can be tracked by users, and, more important, by watchdog agents without need of user interaction. It is possible to react on problems and to issue reminders (e.g. in the case of unexpected delays or loss of a sample).

Descriptions of data, services, and agents are kept in a repository [9]. To support application programmers, medical data, especially the medical record, are managed under an object system which abstracts from distribution and from technical details like physical storage or replication. Services are built upon predefined (medical) object classes and upon methods for the access and management of objects and for graphical presentation.

The concept assumes the existence of a medical entities dictionary. As there are considerable research efforts in this area [e.g.10], this project intends to build upon their results.

Cooperation Infrastructure

As outlined, the system plays a more active role for users and for developers than a pure message handling approach. In addition and as a consequence, it has to maintain a consistent distributed state in the situation of concurrent, potentially interfering activities and in the case of system problems like node failures, network disruptions, or faulty service implementations. The components proposed for the necessary coordination of distributed services and their underlying communication infrastructure are shown in Fig. 2, which will be explained in the next sections.

Service Processing. Users interact with agents. Agents issue service requests, which are sent to the local task manager. The *task manager* stores the request. This persistent request handling is needed for requests that cannot be executed immediately (e.g. a service has to wait for remote

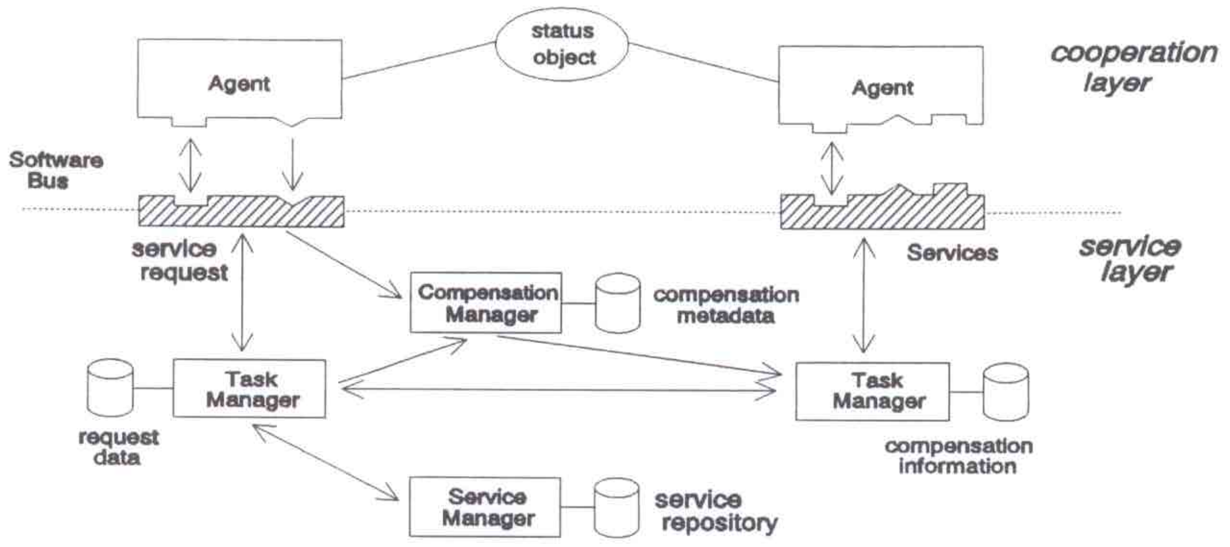

Figure 2: Cooperation Infrastructure (partial view)

user interaction) and for the case of system problems (e.g. a call has to be repeated if a remote system was not available). The task manager calls a *service manager* acting as a global request broker [11], which selects a task manager running on a node offering the requested service. The request is forwarded to the selected task manager, which activates the corresponding agent. This agent executes the service and results are passed back to the task manager on the client node. The client task manager removes the corresponding request entry from its logfile and forwards the results to the requesting agent. The information necessary to compensate a basic service is stored by the task manager at the host side.

In analogy to database transactions, services can be combined into blocks which represent complex services or activities, and which can be nested. A *compensation manager* receives the appropriate nesting information whenever basic services are executed; it also handles compensation requests.

Activity Modeling and Control. The *development* of distributed applications is difficult and error prone. Therefore, to the application programmer, a graphical tool [5] will be offered for *modeling* activities. Here, pre-modeled activities (activity templates) which are kept in the repository, can be used and modified. These activity templates describe

- the states an activity may go through
- the kinds of services that can be called in the context of the given type of activity (where services are classified as system-driven or as user-driven)
- possible state transitions (caused by service executions)
- data flow
- parameters holding auxiliary information like time constraints
- the call interface of the activity as a whole.

The *application programmer* is composing concrete activities using activity templates. The two main tasks are to assign (where still necessary) appropriate service implementations to the corresponding entries in the templates, and to connect the in/out interfaces of the activity to the application environment.

At *run-time*, an instantiated activity template is passed to the local agent which informs all other agents involved in the activity. Each activity in execution is represented by one global status object (logical view, see [12] for possible implementation details). This status object is modified by the participating agents according to the state transitions resulting from service executions. System-driven services are initiated automatically by an agent when a matching state is found in the status object. User-driven services are initiated by user interaction. Time constraints for state transitions are handled by watchdog agents querying the status objects. They enable the system to warn the user, if, for example, a scheduled activity is about to fail.

Further aspects

In this paper, the *basic* infrastructure has been presented in some detail. In addition, and to introduce "intelligent" system behavior, there a specialized agents like planning agents, scheduling agents, and knowledge agents [13]. These agents cooperate in order to support the planning, supervision and execution of complex medical activities according to medical guidelines. The planning agent takes into account dynamic inter-activity dependencies; as an example, an actual result of a laboratory examination may be needed before the next examination starts.

PROTOTYPE IMPLEMENTATION

Although some of the described manager components and the global data resemble a centralized approach, they have to be implemented in a distributed way. To better understand the interaction of modules, an exploratory prototype has been implemented based on UNIX, OSF/DCE, C++ and the relational DBMS INGRES. For presentation services, X11 and InterViews [14] are used. Example activities have been realized as distributed DCE applications. Task managers communicate by means of the DCE Remote Procedure Call mechanism. Authorization and authentification are based on the DCE security service. Service managers utilize the DCE cell directory service. Compensation data and task data have been made persistent by using INGRES. Object methods are implemented on top of the SQL level. Medical Logic Modules [13] have been realized by means of the classification-based knowledge representation system LOOM [15].

DISCUSSION

The basic idea of the described project was not to implement a functional HIS, but to analyse application problems and to relate them to recent informatics research. Under this objective, an exploratory prototype has been realized, and further prototypes will follow.

Related Work

In the HELIOS project [6], a software bus has been used in a comparable way. There are differences, however, in the scope and functionality of the services, as well as in mechanisms for error handling and guaranteeing consistency.

The Kernel $K/2_R$ [5] approach shows an increased functionality compared to HELIOS. S-transactions [8] and workflow management have been integrated.

For the medical environment, several approaches for integrating heterogeneous systems have been presented [e.g.3,16,17] which have resulted in functional systems.

An approach close to the one outlined has been described by Gangopadhyay and Wu [18]. The importance of modeling and automating medical processes is stressed by these authors in the same way as in our concept. Their approach is object-oriented as well, and they use a finite state machine similar to our description of state transitions.

Finally, there are parallels to the RICHE project [19], where knowledge servers and the idea of 'act management' have been introduced.

Conclusion

In actual systems of the medical environment, cooperation mechanisms of the proposed scope are not yet playing an important role. In our opinion, however, these concepts are definitely helpful. A positive effect on the coordination and support of (distributed) clinical processes as well as on application programming can be expected. The high level modeling approach is flexible, and it should help to reduce programming errors. Besides supporting program development at a higher level of semantics, the concept also provides a stable and reliable communication platform. In general, the acceptance of clinical systems will depend on their reliability and on the flexibility of both applications and application design.

Acknowledgement: The authors are grateful to Drs. G.Adler, K.Beckh, B.Böhm, H.Heimpel, M.Reinshagen, and A.Seibold for valuable discussions on application questions and to C.Heinlein for technical discussions. This work was supported by a research grant of the State of Baden-Württemberg, Germany.

References

1. McDonald CJ, Tierney WM. Computer-Stored Medical Records. JAMA 1988, 259:3433-3440.
2. Audet A-M, Greenfield S, Field M. Medical Practice Guidelines; Current Activities and Future Directions. Ann Intern Med 1990, 113: 703-714.
3. Clayton PD, Sideli RV, Sengupta S. Open Architecture and Integrated Information at Columbia-Presbyterian Medical Center. MD Computing 1992, 9: 297-303.
4. Kuhn K, Reichert M, Nathe M, Beuter T, Heinlein C, Dadam P. A Conceptual Approach to an Open Hospital Information System. In: Barahona P, Veloso M, Bryant J (eds) Proc 12th Intl Congr Europ Federation Med Informatics (MIE 94), Lisbon, 1994: 374-378.
5. Adomeit R, Deiters W, Holtkamp B, Schülke F, Weber F. K/2_R: A Kernel for the ESF Software Factory Support Environment. In: Systems Integration '92, Proc 2nd Intl Conf Sys Integration. Ng PA, Ramamoorthy CV, Seifert LC, Yeh RT (eds). Los Alamitos: IEEE Comp Soc Press, 1992: 325-336.
6. Jean FC, Jaulent MC, Coignard J, Degoulet P. Distribution and Communication in Software Engineering Environments. Application to the HELIOS Software Bus. In: Proc 15th SCAMC 1991, Clayton P (ed). New York: McGraw-Hill, 1992: 506-510.
7. Lee KC, Mansfield WH, Sheth AP. A Framework for Controlling Cooperative Agents. IEEE Computer 1993, 26(7): 8-16.
8. Elmargarmid AK (ed). Database Transaction Models for Advanced Applications. San Mateo CA: Morgan Kaufmann, 1992.
9. Hsu C, Bouziane M, Rattner L, Yee L. Information Resource Management in Heterogeneous, Distributed Environments. IEEE Trans SE 1991, 17: 604-25.
10. Cimino JJ, Clayton PD, Hripcsak G, Johnson SB. Knowledge-based Approaches to the Maintenance of a Large Controlled Medical Terminology. JAMIA 1994, 1: 35-50.
11. Nicol JR, Wilkes T, Manola FA. Object Orientation in Heterogeneous Distributed Computing Systems. IEEE Computer 1993, 26(6): 57-67.
12. Schill A, Malhotra A. Language and Distributed System Support for Complex Organizational Services. In: Conference on Organizational Computing Systems, 1991, SIGOIS Bulletin 12, 1991, no 2,3.
13. Heinlein C, Kuhn K, Dadam P. Representation of Medical Guidelines on Top of a Classification-based System. To appear in: Proc 3rd Intl Conf Information and Knowledge Management (CIKM), Gaithersburg, MD, 1994.
14. Linton M, Vlissides J, Calder P. Composing User Interfaces with InterViews. IEEE Computer 1989, 22(2): 8-22.
15. MacGregor R. The Evolving Technology of Classification-Based Knowledge Representation Systems. In: Sowa JF (ed) Principles of Semantic Networks. San Mateo: Morgan-Kaufmann, 1991, 385-400.
16. Van Mulligen EM, Timmers T, van Bemmel JH. A New Architecture for Integration of Heterogeneous Components. Meth Inf Med 1993, 32: 292-301.
17. Gierl L, Greiller R, Landersdorfer D, Müller H, Überla K. A User-oriented Protocol for Integrating Heterogeneous Communication Systems of Medical Facilities Using Ports. Meth Inf Med 1989, 28: 97-103.
18. Gangopadhyay D, Wu PYF. An Object-Oriented Approach to Medical Process Automation. In: Proc 17th SCAMC 1993. Safran C (ed). New York: McGraw-Hill, 1993: 507-511.
19. The RICHE Consortium: RICHE Final Report. Louveciennes, France, 1992.