MEDINFO 98
B. Cesnik et al. (Eds)
Amsterdam: IOS Press

The Federated Healthcare Record to Support Shared Diabetes Care

Onno Weier[a], Martin Kalshoven[a], Hans van der Kolk[b], Freerk Leguit[a], Maurits Ros[b], Pieter Toussaint[a]

[a]*HISCOM bv, Schipholweg 97, 2316 XA Leiden, The Netherlands*
[b]*cademic Medical Centre, Meibergdreef 9, 1105 AZ Amsterdam, The Netherlands*

Abstract

In this paper the conception of the federated healthcare record server to support shared diabetes care is described. Business process modelling is applied to describe the shared care for diabetes patients. Typical dialogues between the different users (patient, internist, GPs, and diabetic nurses) are analysed and described in terms of use cases. Next to this modelling three incremental steps are defined to realise the record server based upon results of standardisation. It proves to be successful to design and build this record server on modern technologies like CORBA and JAVA.

Keywords

Business Process Reengineering; Electronic Healthcare Record; Object Orientation; CORBA; JAVA

Introduction

The functional structure of the healthcare has resulted in an optimisation of the departments in the hospitals, medical institutions, and GP practices. This functional optimization hampers progress in efficient and cost-effective care provided between the different institutions (shared care). This is illustrated by the electronic and paper records [1] currently mostly held at isolated sources of information. This contributes to solutions to well-known problems like long waiting times, duplication of examinations, blank spaces in the schedules of diagnosis and treatment, superfluous activities, and fragmentation.

In the domain of diabetes mellitus treatment an internist, a general practitioner, and a nurse specialised in the treatment of diabetes patients are involved. Moreover healthcare providers at the outpatient clinic and various departments within the hospital are involved. The medical information about a patient (part of the healthcare record) is exchanged between the healthcare providers at the different stages of the treatment. A process oriented structuring of this healthcare record is necessary to make progress towards efficient shared care and cost-containment in the diabetes domain.

The conception of a Federated Health Care Record server to support shared diabetes mellitus care is described. This activity is conducted within the context of the European R&D project Synapses [2], in close co-operation between HISCOM, the Academic Medical Centre (AMC) in Amsterdam, and two General Practitioner-groups in Amsterdam.

The Synapses Project, partly funded under the EU Health Telematics Framework IV Program, sets out to solve the problems of sharing data between autonomous information systems, by providing generic and open means to combine healthcare records consistently, simply, comprehensibly and securely, whether the data passes within a single healthcare institution or between institutions[3]. The approach taken in Synapses is to develop specifications of a server acting as a mediator between information systems, used by the healthcare professionals to register the medical record (the so called feeder systems) and client applications, used for viewing the medical records.

We rely on results of other EU IV Framework projects to address the problems of health data protection and health information systems security.

Feasibility Study

The business process of providing shared care is modelled to conform the Business Process Redesign approach [4]. In this approach the institution boundaries separating the different functions of the shared diabetes care are exceeded. The activities and their coherence are analysed and redesigned. The processes to be improved are described from start to finish to detect their correlation and logistical dependency.

The use case model is described by Jacobson [5] as "a sequence of transactions in a system whose task is to yield a result of measurable value to an individual actor of the system". This technique is applied to the process of providing shared care for diabetes patients. This process is decomposed in five subprocesses:

1. Anamnesis
2. Examination
3. Diagnose
4. Treatment Planning
5. Treatment Conduction

The main actors in shared diabetes care are identified. These actors are classes of users or even systems. These actors are outside the process being modelled.

Actors providing shared diabetes care are:

1. The patient
2. The General Practitioner
3. The diabetes mellitus nurse
4. The internist
5. The system administrator
6. The HIS system
7. The GP systems

All actors play a vital role with one or more of the five subprocesses mentioned. For example, a patient consults the internist after being referred by the GP. Anamnesis and examination are already performed by the GP. The internist diagnoses the patient and plans the treatment. Then the patient is referred to the diabetes mellitus nurse who is responsible for the additional details in the planning and the conduction of the treatment. Such an instance of a sequence of specific steps in the process is called a scenario or process thread.

The following three scenarios describing all major aspects of the process of shared care of diabetic patients are distinguished:

- **Request for advice:** The GP consults the internist or diabetic nurse by asking a question related to the diagnosing, treatment planning or conduction of a treatment. Currently this communication is handled by telephone.
- **Referral:** The GP refers the patient to the internist for additional treatment. Currently this contact is handled by referral letters. The internist reports back to the GP through report letters.
- **Review:** Reviewing of medical data is desired by all actors, but especially by the GP, and the diabetic nurse. Currently these contacts are managed by written reports. This scenario is preceded in time by the referral scenario.

Starting from these scenarios a selection of users was interviewed resulting in the descriptions of the use cases to be supported. These use-cases make up the business processes which are defined earlier in this section.

Let's take the referral scenario to illustrate the use cases for the internist. The analysis of this scenario resulted in the following four use cases:

1. Show anamnesis results from GP. The internist is interested in the anamnesis results to prevent superfluous querying of the patient.
2. Show current medication from GP. The internist must be aware of medication prescribed by the GP for the patient to prevent undesirable interaction with other medication.
3. Show examination results from GP. The internist must be informed on the latest examination results collected by the GP to minimise aggravating examinations for the patient.
4. Show (preliminary) diagnosis from GP. In case of referral the GP is not sure on the diagnosis, but has some suspicion of the illness the patient is suffering from. This might be valuable information for the internist.

In Figure 1 the use case diagram for the internist is represented using the UML notation [6].

The names of the use cases are placed in ellipses. The large rectangle enclosing the use cases represents the system boundary.

The internist, represented by the "stick man figure", communicates with the use cases by evoking them. The GP system (also represented by a stick figure) communicates with the use cases by passing up-to-the-minute patient information to them.

Analysis

During the feasibility study the business process- and use case modelling technique is used. The functionality for all use cases is specified in more detail resulting in an object model. The architecture of this model is based on an extension on the

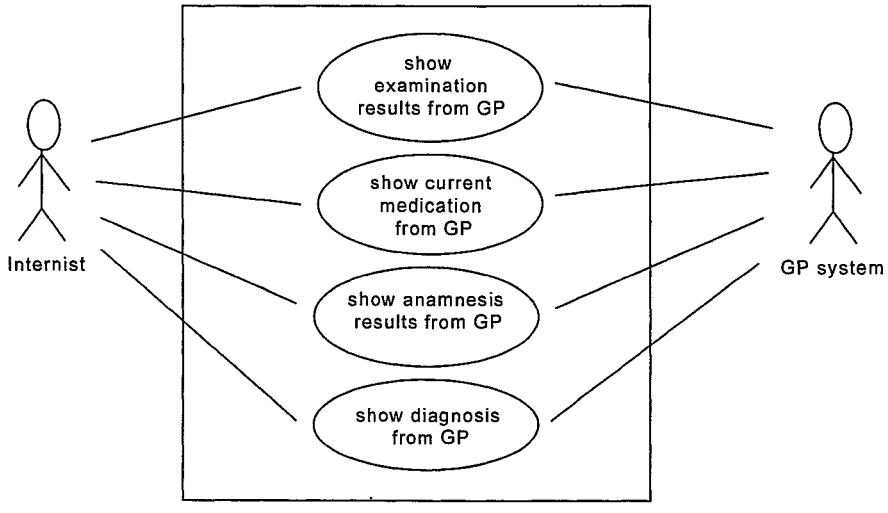

Figure 1 - The use case diagram for the internist

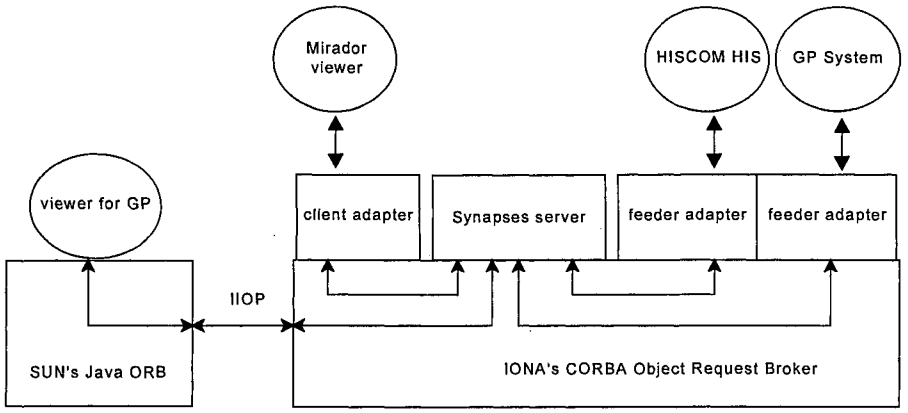

Figure 2 - The distributed architecture of Synapses

GEHR architecture and the ENV 12265 [7] pre-standard on the Architectures of Electronic Health Care Record. This generic architecture is the foundation of the Common Object Model which defines a standardised description of all information found in the healthcare record. The clinical objects supporting the use cases make up the Common Object Dictionary of the system. The structure of the clinical object model in the Synapses server is of vital importance because over time new (versions of) clinical objects will be introduced. The introduction of new clinical objects will be desirable to support new clinical domains and new medical views.

Design and implementation

The creation of the Synapses solution itself is a process incorporating iterative prototyping. Three incremental deliveries are defined supporting the use-cases identified for all actors.

The **first** delivery encompasses a number of components:

- **The Synapses server.** This server, which is the core of the realisation of the Synapses solution, federates the patient record.
- **The viewer for the internist and diabetic nurse.** This viewer is based upon the HISCOM medical workstation MIRADOR [8] to offer a view on the federated health care record of the patient.
- **The viewer for the GP.** This viewer is used by the GP to view the federated health care record.
- The **HISCOM HIS feeder.** In the HISCOM HIS the diabetes related elements of the health care record are registered.

Viewers and feeders that are not according to the Synapses specifications need to be wrapped [9] by an adapter. This is the case for most applications built outside an object environment.

The scenarios requests for advice, referral and review demand a function to notify the arrival of requested information. This will be realised by embedding e-mail like functionality in the viewers. In this way is assured that requests are sent from the correct clinical context.

In the **second** delivery the **GP systems** (Euroned ARCOS and SMS-Cendata MicroHIS) feeders are added. These systems are not Synapses compliant. So specific adapters are necessary to encapsulate these feeder systems.

Users applying different clients do not necessarily use the same terminology as known in the Synapses server. Synonym servicing is realised to map locally used terms to uniquely defined objects within the Synapses server.

The Synapses server is connected to the viewers and feeders by means of the CORBA distributed object technology [10]. This technology allows components to interoperate across the network, run on different platforms, and coexist with existing applications which are not CORBA compliant. Requests for information are handled by the CORBA Object Request Broker (ORB).

In Figure 2 the distributed architecture of the Synapses solution is represented.

The presence of the feeders are announced to the Synapses server. The Synapses server also gets the knowledge in which feeder(s) the information of interest is registered. The clients ask for specific patient information via the Synapses server. The clients automatically get the latest information because registered data is only federated by the Synapses server and not stored locally. The client for the GP is build directly on SUN's Java ORB and communicates with IONA's CORBA ORB by means of the Internet Inter-ORB Protocol (IIOP) [11], which uses TCP/IP as its transport protocol.

The **third** delivery encompasses functionality necessary for usage of the Synapses solution in clinical practice. For this purpose the following functions are included:

1. **Patient identification.** Patient IDs may not be identical across different feeder systems. In absence of any standard on patient IDs, matching of these patient IDs is required. In first instance a pragmatic solution is designed. A number of patient identifying attributes are used as keys to query patients satisfying these constraints (e.g. family name, date of birth, and ZIP code). The correct patient is selected from a list of patients

found. To avoid superfluous user interaction a list of approved matched patient IDs is maintained.

2. **Access rights handling.** Access rights are handled differently by the feeder systems. Although research on security is not covered in the Synapses project itself abundant checking of access rights should be overcome. For this purpose the Synapses server itself can be accessed without user authorisation. Once the Synapses server found out in which feeder the data of interest resides, the user is asked for identification for that specific feeder system. If possible, the connection with the feeder is kept on line as long as the user session is not finished. We will consider in later stage of the project to introduce smartcards letting people carry their own information and identities with them. The smartcard will be challenged with a string each time the user tries to connect to the Synapses server. The challenge string changes each time to prevent illegitimate use. Moreover, this takes away the necessity for the user to identify each time a feeder is queried.
3. **Trusted architecture.** The JAVA/WWW security model claims to trust the environment where the JAVA applets [12] are executed. Principally, this implies that a JAVA applet, loaded in the client for the GP, can invoke CORBA objects without harming the environment of the GP. However, the current JAVA/WWW security model is limited because i) The loaded JAVA/WWW applet is still able to invoke another program harming the GP's environment, and ii) a firewall restricts the communication between the GP's environment and the intranet available in the HIS environment. The major JAVA ORB vendors provide firewall tunneling to overcome the applets security restrictions.

Conclusions

Starting with business process modelling offers a good handle to describe the complexity of the shared care. Advantageous are the resulting use cases which are presented to the users to verify the correctness of the model of their method of working.

As soon as the use cases for all actors are known it is easier to specify the functionality supported by the objects in the object model. Existing results in standardisation of the electronic health care record are applied.

The process of iterative prototyping resulting in three incremental deliveries is a fruitful method. This spreads out the implementation of the Synapses solution over a long period of time. In this way all users are involved in the stages from analysis up to the evaluation.

In the AMC the first version of the Synapses server is implemented and coupled to the HISCOM HIS feeder. Currently the viewers for the GP and the users in the hospital are made Synapses compliant. Until now the CORBA Distributed computing technology and JAVA technology have proven their approriateness. Based upon these technologies the engineering architecture of federated healthcare record server to support shared diabetes care is realised.

References

[1] Wyatt, J. C., Clinical data systems, part 1: data and medical records, in: The Lancet, Vol 344, December 3, 1994, pp 1543-1547.

[2] Viewing the Electronic Healthcare Record, Osseyran A., Toussaint P., Kalshoven M., Bruin Slot H., Hooymans M., Hofdijk J., in: Medical Informatics Conference '96 proceedings, 1996, pp 165-174.

[3] Grimson, J. et al., SYNAPSES - Federated Health Care Record Server, in: Medical Informatics Europe'96, IOS Press, 1996, pp 695-699.

[4] Davenport, Th. H. & J.E. Short; The New Industrial Engineering: Information Techno-logy and Business Proces Redesign; in: Sloan Management Review nr. 4, 1990.

[5] Jacobson, Ivar Jacobson, et al, The Object Advantage: Business Process Reengineering with Object Technology, Addison Wesley, 1994.

[6] Unified Modelling Language Notation Guide, Version 1.0, Rational Software Corporation, 13 January 1997.

[7] ENV 12265, pre-standard on Electronic Health Care Record Architectures, published by CEN/ TC 251/WG 1.

[8] Hooymans M, Osseyran A, Bakker A.R., A GUI integrated in an existing large HIS, a Chain of Many Links, in: Medical Informatics Europe '96, 1996, pp 271-275.

[9] Mowbray T.J., Malveau R.C., CORBA Design Patterns, Part II Application Design Patterns, Chapter 5 Improving Object Implementations, Object Wrapper, John Wiley & Sons, Inc, 1996, pp 121-125.

[10] Orfali R., Harkey D., Edwards J., The essential Distributed Objects survival guide, John Wiley & Sons, Inc, 1996.

[11] Vogel A, Duddy K, JAVA Programming with CORBA, Chapter 2 CORBA Overview, John Wiley & Sons, Inc, 1997, pp 11-29.

[12] Anuff, E, The JAVA sourcebook, John Wiley & Sons, Inc, 1996.