1998-01-01

A CORBA-base Integration of Distributed Electronic Healthcare Records using the Synapses Approach

Jane Grimson
Trinity College Dublin

William Grimson
Dublin Institute of Technology, william.grimson@dit.ie

Damon Berry
Dublin Institute of Technology

Gaye Stephens
Trinity College Dublin

Eoghan Felton
Trinity College Dublin

See next page for additional authors

Authors
Jane Grimson, William Grimson, Damon Berry, Gaye Stephens, Eoghan Felton, Dipak Kalra, Pieter Toussaint, and Onno Weir

A CORBA-Based Integration of Distributed Electronic Healthcare Records Using the Synapses Approach

Jane Grimson, *Member, IEEE*, William Grimson, Damon Berry, Gaye Stephens, Eoghan Felton, Dipak Kalra, Pieter Toussaint, and Onno W. Weier

***Abstract*—The ability to exchange in a meaningful, secure, and simple fashion relevant healthcare data about patients is seen as vital in the context of efficient and cost-effective shared or team-based care. The electronic healthcare record (EHCR) lies at the heart of this information exchange, and it follows that there is an urgent need to address the ability to share EHCR's or parts of records between carers and across distributed health information systems. This paper presents the Synapses approach to sharing based on a standardized shared record, the Federated Healthcare Record, which is implemented in an open and flexible manner using the Common Object Request Broker Architecture (CORBA). The architecture of the Federated Healthcare Record is based on the architecture proposed by the Technical Committee 251 of the European Committee for Standardization.**

***Index Terms*— Common Object Request Broker Architecture (CORBA), distributed information systems, electronic healthcare records (EHCR's), health informatics, open systems.**

I. Introduction

THE DEVELOPMENT of the electronic healthcare record (EHCR) is taking place against a background of increasing computerization throughout the healthcare sector and a growing movement toward team-based or shared delivery of healthcare. Until relatively recently, the main use of computers in the health sector was to support administrative and financial functions, whereas today there is agreement that the emphasis should be on supporting clinical functions with administrative support as a byproduct [1], [2]. Good clinical computing is still more the exception than the rule [3]. The increase in computerization means that health information about patients—especially in the larger teaching hospitals—is often available in electronic form, albeit frequently in nonintegrated systems.

Shared delivery of healthcare depends crucially on the ability to share information between health professionals and, in particular, the ability to support shared access to the healthcare record. In such an environment, the physical limitations of the paper-based record that restrict access to a single user in a single location at one time become a major impediment. Whereas in the past, information technology (IT) could not support all of the complex demands of the EHCR, this is no longer the case and EHCR systems are beginning to emerge on the marketplace. How these systems will develop and, more particularly, how they will be integrated with existing health information systems, is still an open issue and standards are essential [4]. However, it is virtually certain that if they are to support shared care successfully across primary, secondary, and tertiary sectors, they must be able to integrate health data that is distributed across heterogeneous computing systems. There is no possibility of a single solution emerging that will meet the needs of all those involved in the delivery of healthcare. The EHCR of an individual patient will always reside in a multivendor, multisite environment with the individual parts, e.g., a general practitioner (primary care physician) record, a single hospital record, being complete in their own context.

IT today offers a variety of solutions to "sharing," including generic approaches, such as federated database management systems [5], [6], gateways [7], data warehousing [8], and more recently, the World Wide Web (WWW). Many of these technologies have been adapted for the healthcare sector, including for example data warehousing, such as OACIS [9], integration engines, such as Cloverleaf [10] and DataGate [11], EDI through HL7 [12], and EDIFACT [13], and several examples using WWW [14]–[17]. All of these systems only go part of the way toward solving the problems of full syntactic and semantic interoperability. They do not provide truly open systems in which it is possible for users to select best-of-breed applications, plug them into some "middleware," and expect them to be able to exchange data in a meaningful way. In order to achieve such a flexible and open plug-and-play environment, the large number of possible middleware architectures and interfaces must be constrained through internationally accepted standards. In the European context, it is the working groups of the Comitié Européen de Normalization Technical Committee 251, CEN/TC251, which are responsible for developing the relevant standards for the health sector. In the specific context of sharing EHCR's, the critical standard is the EHCR architecture [18]. And as regards system architectures, an

Manuscript received May 6, 1998; revised August 31, 1998. This work was supported by the European Commission.

J. Grimson, G. Stephens, and E. Felton are with the Department of Computer Science, Trinity College, Dublin 2, Ireland (e-mail: jane.grimson@tcd.ie).

W. Grimson and D. Berry are with the Faculty of Engineering, Dublin Institute of Technology, Dublin, Ireland.

D. Kalra is with the Centre for Health Informatics and Multiprofessional Education, University College, N19 5NF London, U.K.

P. Toussaint and O. W. Weier are with HISCOM BV, 2300 AX Leiden, The Netherlands.

Publisher Item Identifier S 1089-7771(98)08582-3.

important contribution to an ongoing debate is the Healthcare Information System Architecture (HISA), which defines a middleware-based approach [19]. Taken together, these two standards provide a basis for sharing electronic records between heterogeneous healthcare information systems [20].

It is against this background that the Synapses project was conceived. The aim of Synapses is to develop an open and generic means for sharing healthcare records and related medical data in a simple, consistent, and secure way. The term federated healthcare record (FHCR) is used in the same way as in a federated database system. A federated database system is a collection of independent, autonomous database systems, each with their own set of global users, which cooperate together to form an alliance or federation that enables global users to access data across the participating systems in a transparent manner [5], [6]. Formally, an FHCR is defined as an integrated, communicable, combinable, and comprehensible healthcare record that is based on a standard object model. It therefore ensures that the integrity and context of the information being exchanged is guaranteed.

This paper presents a solution to the problem of sharing EHCR's that has been developed as part of the Synapses project—a three-year project funded under the European Union's (EU's) Fourth Framework Health Telematics Programme [21]. The consortium consists of 26 partners from 14 different countries representing the health software industry sector, research institutes and universities, and end users through the participation of several hospitals. Section II outlines the Synapses solution to sharing EHCR's from the information and computational viewpoints, while Section III discusses the implementation of the Synapses server using CORBA. Section IV describes results from pilot applications in two diverse clinical domains—namely, an intensive care unit (ICU)—and to support shared care between a group of general practitioners and a specialist diabetic clinic in a hospital. Finally, some general conclusions are made in Section V.

II. Description of Synapses

Synapses sets out to solve problems of sharing data between autonomous information systems by providing generic and open means to combine healthcare records or dossiers consistently, simply, comprehensibly, and securely, whether the data pass within a single healthcare institution or between institutions. The Synapses computing environment consists of client applications accessing in a controlled manner distributed components of healthcare records through a server, where the server is connected to "feeder systems," in which such records and patient information are stored [22]. As the components of the healthcare record are in general distributed across the feeder systems, a client requests patient information in a federating process leading to an FHCR: the Synapses server is said to be an FHCR server. It is noted that once the federation process has taken place the record exists as an EHCR and has the same properties as a record obtained from a single feeder system.[1] One of the aims of Synapses is to present client applications with a uniformly consistent view of records. Thus, bearing in mind the heterogeneity with respect to how patient information and records are currently stored, it follows that the main characteristics of the Synapses computing environment are as follows:

- federation;
- integration;
- adaptation.

Adaptation implies that the server has an appropriate adapter for each connected feeder system unless the feeder system is already Synapses compliant. The adaptation required is one in which diverse record architectures are mapped or converted to a common architecture. Federation is facilitated by the adoption of a common record architecture, and Synapses has based its design around the work of CEN TC/251/PT1-011 [18].

A. Information Viewpoint: The Record Architecture

Standardization of the architecture of EHCR's is essential if the records are to be used to support shared care involving clinicians from different disciplines and to enable the transfer of records across national and cultural boundaries either for reasons of increased patient mobility or for accessing expert healthcare advice (teleconsultancy). A number of groups in Europe have been engaged in research into the architectures of EHCR's, in particular, Working Group 1 of the CEN TC/251, who has developed a prestandard [18]. This prestandard defines the basic architectural components of an EHCR and their logical interrelationships. The architecture is defined to enable clinicians to make their own decisions about what to record and in what format. It supports a common understanding of the necessary variety of the content and format of records. The prestandard was built on extensive experience developed in a number of projects funded under the Advanced Informatics in Medicine Programme of the EU, in particular, the Good European Health Record (GEHR) project [23]. GEHR developed a comprehensive multimedia data architecture for using and sharing EHCR's, with a strong focus on meeting the clinical, technical, educational, and ethicolegal requirements.

The approach in Synapses is built around a canonical model, referred to as the Synapses object model or SynOM, which provides structure to the Synapses FHCR, and in which the individual components of a patient's healthcare record are combined. Synapses is therefore exploiting many of the ideas from federated database technology. More details of how federated database technology is exploited within Synapses can be found in [24]. Federated database management systems (FDBMS) seek to provide a loose coupling between heterogeneous information systems, allowing global users uniform and transparent access to the data within those systems. Wrappers are placed around the data in the local information systems to hide the underlying heterogeneity. In spite of considerable research effort, the goal of providing a generic solution to heterogeneous database interoperability has proved elusive, forcing organizations to constrain the problem by adopting a more pragmatic approach. In the case of Synapses.

[1] Paper records are themselves generally a federation of components, each component being delivered to the "folder" by healthcare professionals.

this takes the form of an application-specific canonical data model, the SynOM. This restriction is essential to not only preserve the meaning of the information being exchanged, but also its context and structure, safeguarding the legal, ethical, and clinical integrity of the record. An active object-oriented data dictionary, the *SynOD*, stores the definitions of the record components/data objects that can be requested through the server by the client applications, together with all of the information needed to decompose the queries to and consolidate the responses from the feeder systems.

1) Synapses Object Model: The challenge being addressed by the Synapses FHCR architecture is to provide a formal representation of the generic characteristics applicable to any potential healthcare record entry arising from a feeder system, now or in the future.

The very extensive investigations of user and enterprise requirements that have taken place over several years have sought to capture the diversity and specialization across primary, secondary, and tertiary care, between professions and across countries. These requirements have been distilled and analyzed by expert groups across Europe to identify the basic information that must be accommodated within an EHCR architecture to do the following:

- capture faithfully the original meaning intended by the author of a record entry or set of entries;
- provide a framework appropriate to the needs of professionals and enterprises to analyze and interpret EHCR's on an individual or population basis;
- incorporate the necessary medicolegal constructs to support the safe and relevant communication of EHCR entries between professionals working on different sites.

At present, however, a considerable wealth of healthcare information is held and will in the foreseeable future be held in diverse record architectures ("legacy systems"), including some very simple stand-alone applications. Synapses servers must, therefore, be capable of accommodating requests for clinical information from this wide range of data architectures. Future work within the CEN/TC251 may result in modifications to the current standard, and it is therefore important for the FHCR, as modeled by the SynOM, to be as generic and flexible as possible to cope with future changes.

2) Representing Contextual Information: The work of GEHR and ENV 12 265 has drawn attention to the essential nature of contextual information captured alongside the individual clinical entries at the time of recording. This contextual information will include the following:

- record authorship, ownership, and duty of care responsibilities;
- subject of care;
- dates and times of healthcare actions and of their recording;
- version control;
- access rights;
- aspects of certainty and accuracy;
- emphasis and presentation;
- links to other record entries, medical knowledge, and protocols.

This kind of information will exist at different levels of detail within different feeder system architectures, but where it exists, this information must accompany the specific clinical data values to present their meaning faithfully and preserve medicolegal integrity.

a) SynOM: The proposed SynOM defines and extends the set of constructs defined in ENV 12265 to optimize the faithful mapping to and from a wide range of clinical databases and comprehensive EHCR architectures. Its classes and attributes provide a flexible and comprehensive model for clinical data derived from a diversity of feeder systems and from which more sophisticated healthcare record models and messages can be constructed to suit the needs of individual client domains. The class diagram of the SynOM is shown in Fig. 1. A description of the principal SynOM classes follows.

b) RecordComponent: RecordComponent is the abstract base class for RecordItemComplex and RecordItem (see definitions below). It defines the common attributes applicable to all of the major classes of the SynOM for the following:

- record authorship, ownership, and duty of care responsibilities;
- subject of care;
- dates and times of healthcare actions and of their recording;
- version control;
- access rights;
- emphasis and presentation.

The complete set of attributes and their data types is still being refined, and different versions for this are presently being evaluated.

The SynOM distinguishes between the aggregation necessary to convey compound clinical concepts and aggregation within a record that provides a flexible way of grouping observations that relate to the healthcare activities performed. An example of the former would be *blood pressure*, which is a compound concept composed of *systolic* and *diastolic* values. Examples of the latter would be the grouping together of observations under the general heading of *Physical Examination*. The RecordItemComplex and RecordItem constructs, respectively, represent these two broad categories of aggregation.

c) RecordItemComplex (RIC): This class corresponds to the ENV 12 265 construct of the same name. In the SynOM, RecordItemComplex is the common abstract superclass for the high-level grouping of observations that relate to the healthcare activities performed. Two broad categories of RIC are defined in the standard and reflected in the SynOM through two abstract subclasses of RecordItemComplex, as follows.

- OriginalRIC: This set of classes represents the original organizational structure (grouping) of sets of record entries, as defined by the author(s) of those entries; it provides the medicolegal representation of the underlying information.
- ViewRIC: This set of classes provide the means by which alternative groupings and subsets of the original information may be organized and preserved as permanent views

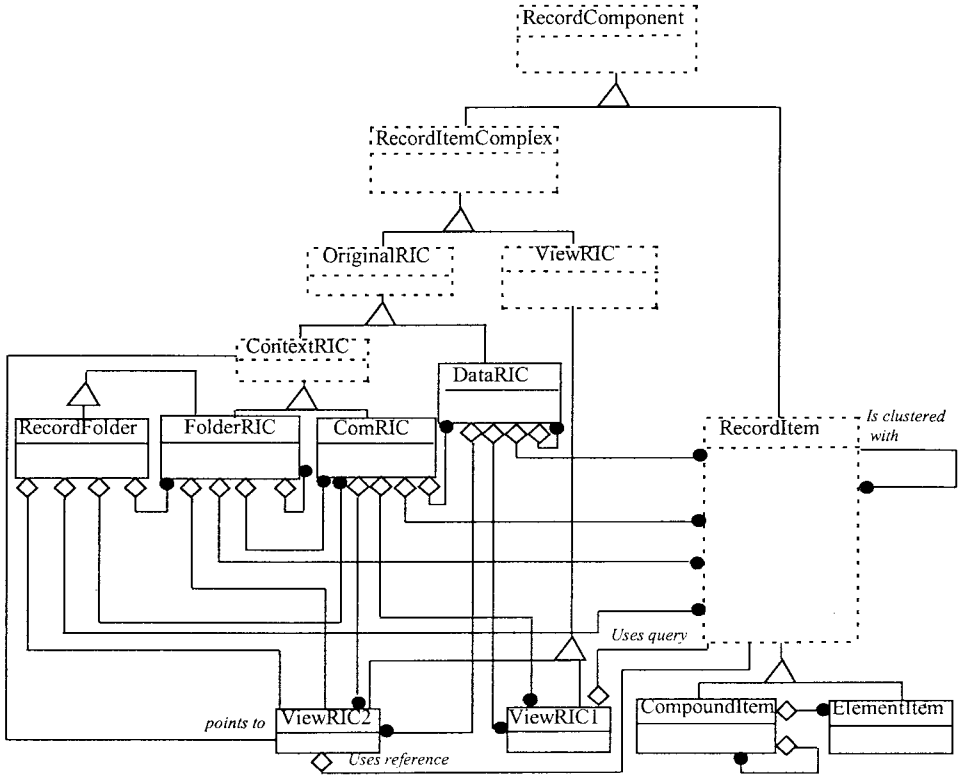

Fig. 1. Class diagram of the Synapses EHCR record components.

in a patient's record, unlike those generic views provided in an *ad hoc* way by a client system.

d) OriginalRIC: Three concrete classes of OriginalRIC are defined in the SynOM to provide for the nested aggregation of original groupings for record entries.

i) FolderRIC: FolderRIC's define the highest levels of organization within healthcare records. They will often be used to group large sets of record entries within departments or sites, over periods of time, or to demarcate a prolonged illness and its treatment. Examples of FolderRIC's include an episode of care, an inpatient stay, or one stage of a disease process.

ii) RecordFolder: The RecordFolder class is a special subclass of FolderRIC. It defines the root folder within a single patient's healthcare record, i.e., a Synapses FHCR must consist of exactly one RecordFolder object.

iii) ComRIC: A medicolegal set of record entries required by the author to be kept together (to preserve meaning) when information is communicated using Synapses. The original context of exchanged record entries is preserved by ensuring that all persistent EHCR stores comprise only whole ComRIC's. This explicitly includes caches and cache mechanisms. The ComRIC also defines the medicolegal cohort for the inclusion of new entries within an EHCR: any new EHCR entry (even if stored on a local feeder) must be a whole ComRIC.

Examples of ComRIC's include the following:

- data entered at one date and time by one author (similar to a GEHR transaction);
- information gathered through the use of a protocol or template;
- serialized set of readings taken over time but contributing to one examination;
- definition of structures corresponding to electronic documents.

iv) DataRIC: This class is intended for grouping observations under headings *within* a ComRIC. It therefore provides for the fine granularity grouping and labeling of record entries with names that relate the clinical concepts to the healthcare activities and processes surrounding the patient. Examples of DataRIC names include presenting history, symptoms, investigations, treatment, drug prescription, needs, or plan.

v) ViewRIC: Two concrete classes of ViewRIC are defined in the SynOM to provide for two differing mechanisms by which views may be generated.

a) ViewRIC1: ViewRIC1 corresponds to the view record item complex, subtype 1 in ENV 12 265. The ViewRIC1 provides a means for grouping entries within ComRIC's, at a similar hierarchical level in a record to the DataRIC. However, the data within a ViewRIC1 is derived through the use of a predefined query procedure, i.e., a ViewRIC1 comprises a query that generates a set of entries dynamically at the time of a client request. The mechanism by which search criteria can be defined in a generic, durable, and portable manner within the ViewRIC1 class is presently being developed. At present, as in ENV 12265, the query procedures may only return RecordItems.

b) ViewRIC2: ViewRIC2 corresponds to the View Record Item Complex, subtype 2 in ENV12265. The ViewRIC2 provides a static view of original information through a set of references to the original entries or to groups

of entries (i.e., RecordItems, DataRIC's, and/or ComRIC's). It therefore provides a mechanism by which information within one ComRIC may logically appear inside another ComRIC since the originals of these cannot be nested.

vi) RecordItem: RecordItem corresponds to the "record item" concept of the ENV 12 265. This class defines the structure of the individual clinical entries within a record. It is defined in ENV 12 265 as "the smallest unit of information that remains meaningful as an entry in a healthcare record." An important aspect of its definition is the binding of a name (acting as a label) to each content value, providing the individual quantities, dates, or clinical terms with a primary context. The detailed model of the RecordItem class is still being refined, and different versions for this are presently being evaluated.

3) SynOD: The SynOD contains a standardized set of definitions of healthcare objects, which can be mapped to the local data representations of the "synapsed" feeder systems. These object definitions are expressed using only classes derived from those found in the SynOM. Experience has shown that a SynOD in a typical Synapses implementation could contain tens to hundreds or even thousands of such object definitions. In contrast to the SynOM, the SynOD is domain specific and contains the definitions for the healthcare objects that will directly populate the record. In addition to the record object definitions, the SynOD can be used to store supporting sets of standard clinical object definitions, which can be used to build part or all of the record. It could, for instance, contain a realization of a domain-specific object model produced by a standards organization. From this perspective, the SynOD could be considered as a clinical class dictionary. It is important to note, however, that although a class is a design-time concept, users can interact with the SynOD and add new clinical object definitions while the Synapses server is running.

At run-time, the SynOD acts as template for the record. If a client issues a request for a record, the server first consults the SynOD to determine the nature of the record that is to be built and then constructs the record, again using only objects derived from the SynOM base classes.

The healthcare objects defined in the SynOD are capable of being transferred between components of Synapses systems or between Synapses servers using an exchange format that is based on the SynOM. However, for two Synapses systems or servers to communicate meaningfully, they must at least have shared those parts of their respective SynOD's, which describe the record components to be exchanged. The ultimate aim is to have large standards-based portions of the SynOD agreed between all Synapses sites. This would enable servers to communicate in an open manner, while still allowing parts of each SynOD to be customized to the local environment at each Synapses site. It is hoped to be able to enhance the SynOD with objects defined by the object management group domain task group, CORBAmed [25].

In the meantime, the advantage of the SynOD is that it facilitates the exchange of objects between servers that share standard sets of object definitions, while allowing IT personnel the freedom to define their own site-specific healthcare objects.

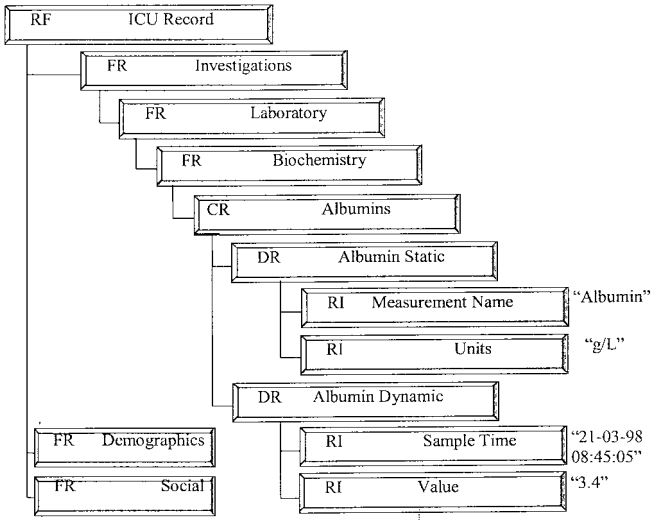

Fig. 2. ICU record, flexibly built using the objects defined in the SynOD.

An example of using the SynOD to build part of an ICU record is shown in Fig. 2. The part of the record illustrated in the figure concerns investigations (FolderRIC) that can consist of a particular type of investigation, namely, laboratory (again a FolderRIC), which in turn has a specific set of laboratory investigations—biochemistry (another FolderRIC). The biochemistry FolderRIC contains two Data RIC's, which hold the static and the dynamic data of an albumin test result. Note, other information content included in the above record, namely, the demographic and social information, are in the two additional FolderRIC's.

B. Computational Viewpoint: Synapses Interfaces

The interfaces in the Synapses computing environment are shown in Fig. 3 in the form of a data flow diagram (DFD), where the feeder systems are assumed to be non-Synapses compliant.[2] The main interaction path corresponding to a client request for a record and the response to that request involves P2, P6, P7, and either P11 and P12 or P10 and P8, where the processing in P11 and P10 takes account of the non-Synapses compliance of current feeder systems. The long-term objective when standards have been widely adopted is to connect P6 directly to P12 and dispense with the adapter-like processing required in P10 and P11.

The details of the individual processes of the Synapses environment shown in Fig. 3 are described in Table I.

The intent in Synapses, which is entirely consistent with the promotion of standards, is that the processing shown in P7, P11, and P10 is part of a migration strategy and will become redundant when EHCR standards have been widely adopted.

C. Record Retrieval Interface

It is beyond the scope of this paper to describe each Synapses interface in detail, and this section concentrates on

[2] It is assumed that there will be a migration period during which Synapses will not be fully accepted and in which noncompliant feeder systems are likely to be the norm. These feeder systems may be either full EHCR systems or non-EHCR systems but contain patient data.

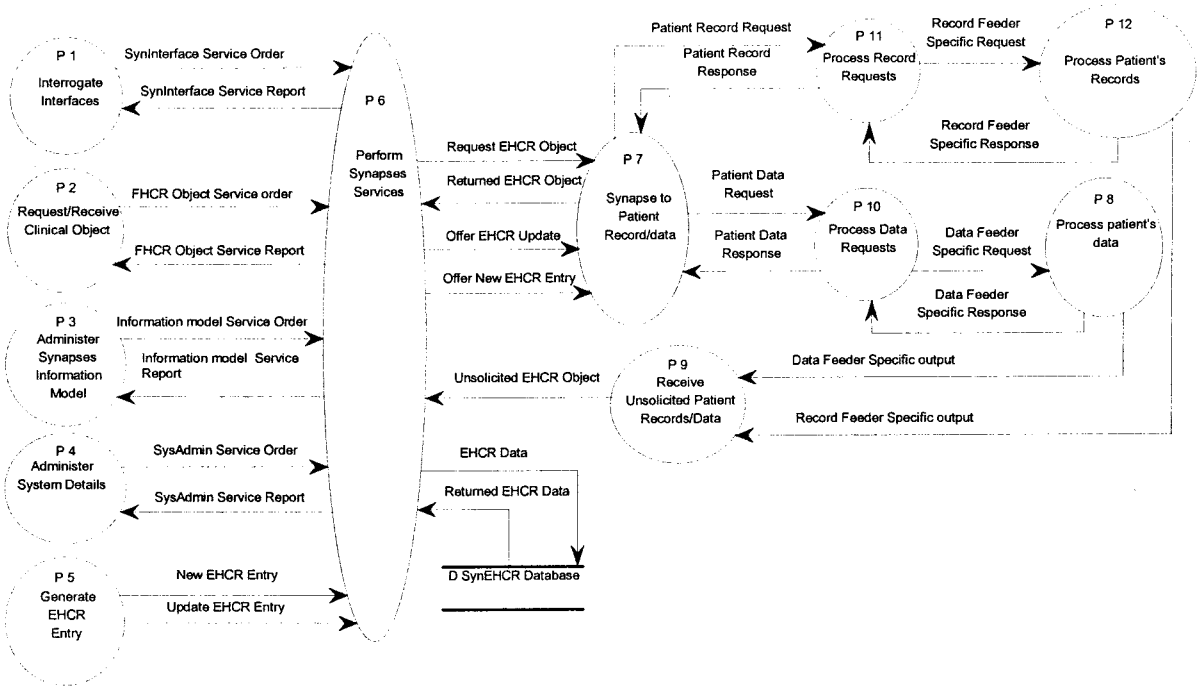

Fig. 3. Top-level data flow diagram showing a Synapses server operating with non-Synapses compliant record and data feeder systems.

the Synapses interface that corresponds to the **SynInterface-ServiceOrder/SynInterfaceServiceReport** traversal from the DFD in Fig. 3. In the Synapses specification, this request report traversal is implemented as the **record retrieval interface**, and a use-case [26] description of the interface is shown in Fig. 4.

Each of the six operations supported by the record retrieval interface, as shown in Fig. 3, are described below. For all of the operations except interrogate interfaces (P1 in Fig. 3), the Synapses client must provide its identification so that the server can ascertain the client's access rights.

1) Find Record: The find record interface enables the Synapses client to retrieve details concerning record identification. The attributes that constitute record identification can vary according to healthcare institutions and within healthcare institutions.

2) Retrieve Folder RIC: This interface enables the client to retrieve details concerning FolderRIC's that appear in a specified record. The details returned by the server are specified in FolderRIC object format that has yet to be determined. The details concerning FolderRIC's are stored in the SynOD.

3) Retrieve Search Criteria: Using this interface, the Synapses client retrieves a template of the criteria used to search for patient records. With this template, the client can provide a combination of attributes that can identify either a set of records or a single record. In order to retrieve the details of record identification, the client uses the find record interface.

4) Retrieve ComRIC: This interface supports retrieval of any ComRIC that is held in a patient's record. When a ComRIC is requested, everything below it, i.e., everything it contains, is returned to the client.

5) Find ContextRIC: The Find ContextRIC interface allows the Synapses client to retrieve the RicId's of the ContextRIC's contained within a particular record. ContextRIC's are ComRic's and FolderRIC's.

6) Retrieve Class Definition: The Synapses client uses the retrieve class definition interface to obtain definitions from the SynOD concerning classes against which it can issue queries against. The classes it can issue queries against are FolderRIC's and ComRIC's.

III. CORBA Engineering and Technology: An Implementation Perspective

A number of different engineering platforms are being used to validate the Synapses server and the implementations at two sites, Dublin (Ireland) and Amsterdam (The Netherlands), are based on the Common Object Request Broker Architecture (CORBA) technology. The next section gives a brief overview of CORBA.

A. Overview of CORBA

CORBA is a middleware standard that is based on the concept of the Object Request Broker (ORB), as shown in Fig. 5. The CORBA standard [27] describes ORB's as mediators between clients and application (server) objects, which arrange for those objects to access each other across networks at run time. CORBA permits local proxies of the server object to be transparently created in the client's address space. The client can operate on the proxy object to change the state of the object on the server. The means of achieving this goal is platform independent and hidden from the developer.

TABLE I
PROCESS DESCRIPTIONS FOR TOP-LEVEL DATA FLOW DIAGRAM

Process No	Description
P1	This process is used to enquire about the interfaces which are implemented and available from the Synapses server. The process issues a *SynInterface Service Order* to P6, which returns a *SynInterface Service Report.*
P2	This process is concerned with requesting and receiving clinical objects which have been defined by the SynOD. The process issues a *Request FHCR Object* to P6 and receives a *Returned FHCR object* from P6.
P3	It is necessary that the SynOD be administered i.e. that definitions of record components and pointers to methods which are used to retrieve them from feeder systems can be inserted, updated, deleted and selected by any process with permission to do so. This process interacts with P6 using *Dictionary Service Orders* and *Dictionary Service Reports.*
P4	Administration of a Synapses server also includes tasks such as access control management, audit trail specification and browsing, naming services and maintaining export schema. This process issues a *SysAdmin Order* to P6 to request a certain administration function and it receives a *SysAdminReport* in return.
P5	This process is used to update patient records or create new entries in records through the use of *New EHCR Entry* and *Update EHCR Entry.*
P6	This process represents the functionality required to: – inform other processes of the interfaces it supports. – federate patient records. – manage the SynOD. – issue requests and receive replies concerning information stored in feeder systems. – accept and process, in a limited way, updates to patient records. – administer the above functionality in a secure and accountable fashion.
P7	This process accepts requests from P6 to retrieve certain Synapses defined objects from feeder systems. Synapsing to the feeder system requires that the object requested is mapped to the specific notation for the relevant feeder system, and when the data is returned from the feeder system that it is "placed" into a Synapses defined object , *Returned EHCR Object*, and returned to P6. P7 is a process which has generic and specific functionality. The generic functionality can accept requests for certain Synapses defined objects and the specific functionality relates to the manner in which each type of feeder system is connected to a Synapses server.
P8	This process represents a feeder system which is a non-record system and contains patient data which can be inserted into records and interacts with Synapses through *Data Feeder Specific Request* and *Data Feeder Specific Response* - the wrapping being performed either side of the interaction.
P9	This process is designed to provide the Synapses server with asynchronous notification of events e.g. alarms or alerts
P10	This process is required when patient information is being retrieved from non-record systems through *Patient Data Request* with the returned *Patient Data Response.* This process basically builds synapses like record objects in Synapses' own terms.
P11	This process is used where record objects are to be retrieved from record systems from as yet non-Synapses compliant feeder systems through *Patient Record Request* and *Patient Record Response.*
P12	This process represents a feeder system which is a record system, though non-Synapses compliant, and contains patient records which are retrieved through *Record Feeder Specific Request* and yielding the return *Record Feeder Specific Response.*

The architecture defines the following:

- architecture for communication between objects, which may exist in a distributed and/or heterogeneous computing environment;
- interface definition language (IDL) for defining the interactions between those disparate objects.

The first version of CORBA, Version 1.1, was introduced by the Object Management Group (OMG), Framingham, MA, in 1991.[3] CORBA 2.0, which was adopted in 1994, defines true interoperability by specifying how ORB's from different vendors can interoperate, thus allowing CORBA-based systems to span heterogeneous computer systems [28].

The architecture supports the following:

- location transparency, which enables applications to invoke interfaces without prior knowledge of the location of their interface implementations;
- access transparency, which enables applications to interwork across heterogeneous computer architecture and programming languages.

B. Implementation of the Server

The implementation of the Synapses server prototype intended for use at St. James's Hospital (SJH), Dublin, comprises several important components that are described below. Later in Section III-C, a description of the operations in the server is presented.

1) In-Memory Persistent SynOD and SynOD Database: The development of the SynOM was influenced by the following implementation concerns.

[3]More information can be found at OMG's website: http://www.omg.org/.

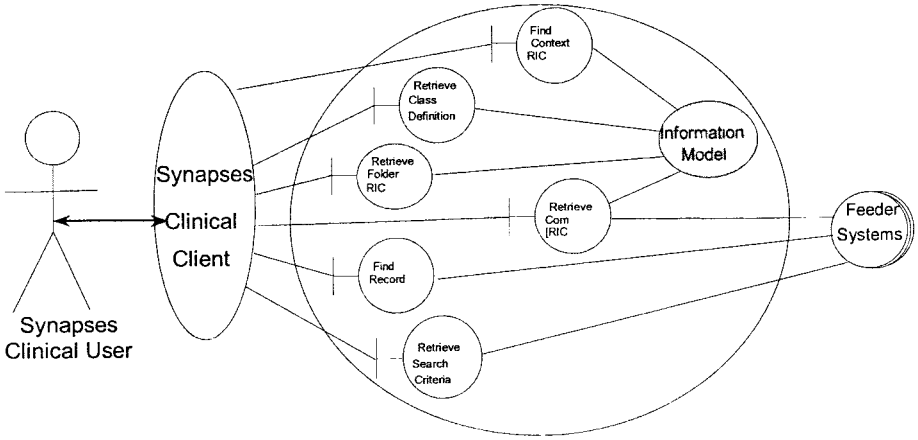

Fig. 4. Use-case for the record retrieval interface.

- SynOM structure should be stable while permitting the definition of additional Synapses ECHR objects at run time. This measure minimizes the recoding necessary to add new objects to the SynOD and keeps the exchange format consistent between server implementations.
- SynOD should be built such that the Synapses EHCR could be (re)structured without, if possible, adding any code to the system.

As a result of the above criteria, the SynOD is built so that each entry in the SynOD is an instance of a SynOM class. To improve efficiency, the SynOD is implemented as a set of persistent clinical object definitions that are constructed in memory at server start up. During construction of the SynOD, open database connectivity (ODBC) calls are used to communicate with a database in which the information that comprises the SynOD is stored persistently.

2) Record Structure Builder—A Tool for Creating SynOD's: The SynOD for the server prototype is managed using a graphical editing tool known as the record structure builder (RSB). This application stores the metadata required to construct both SynOM and SynOD in a database. The tool and the underlying database are designed to allow both the SynOM and SynOD to be remodeled with certain restrictions, without connecting to the server. The "live" SynOD on the server is updated to "point to" the new SynOD database file.

C. End-User Application—Server Interface Using Orbix

In order to allow the server to return medical information (in record format), the prototype uses a minimum subset of server operations from the Synapses record retrieval interface. This section of the paper describes some of the key aspects of the implementation. The server makes use of the SynOD (in recovering patient data or structuring information for the electronic record) when replying to these calls. The implemented operations that are described here are: FindRecord, RetrieveFolderRIC, and RetrieveComRIC. Fig. 6 is a sequence diagram that shows a typical set of calls to the record retrieval and user archive interfaces. In fact, these are the calls that would result from a user logon to a Synapses client, followed by an automatic traversal by the client to the "Albumins" ComRIC in the SynOD example shown in Fig. 6.

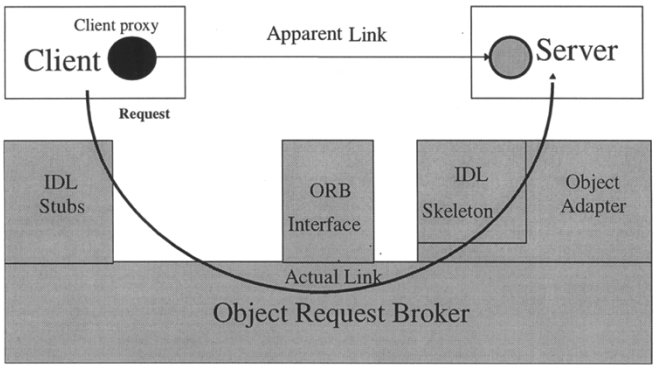

Fig. 5. Elements of the common object request broker architecture.

1) Implementation of FindRecord: The FindRecord operation is the simplest of the operations described here. It returns a set of patient ID's that represent patients whose medical information is available to the server. The server makes use of ODBC calls to get a list of patient ID's for ICU patients who fall into this category.

2) Implementation of RetrieveFolderRIC: RetrieveFolderRIC is used to return the contents of a particular FolderRIC (see Fig. 7). The SynOD provides the template for the composition of a record using RecordFolders, FolderRIC's, and ComRIC's. This template is the same for each patient. This allows the server to retrieve record structure information from the SynOD rather than from a particular patient record.

RetrieveFolderRIC recovers a set of FolderRIC's that have a particular HomeRIC in common. This HomeRIC is indicated by an input parameter. In effect, RetrieveFolderRIC returns the immediate contents of a particular FolderRIC. This allows clients to show a directory tree hierarchy of the various FolderRIC's provided by the server while issuing iterative calls to RetrieveFolderRIC to build the record.

3) Implementation of RetrieveComRIC: While RetrieveFolderRIC allows the client access to information regarding the structure of the EHCR, RetrieveComRIC allows access to the actual patient data. These data are stored on feeder systems, so the server must make requests to the feeder systems and assemble the recovered data "on-the-fly" into the Synapses return format. The RicId provided by the client indicates which

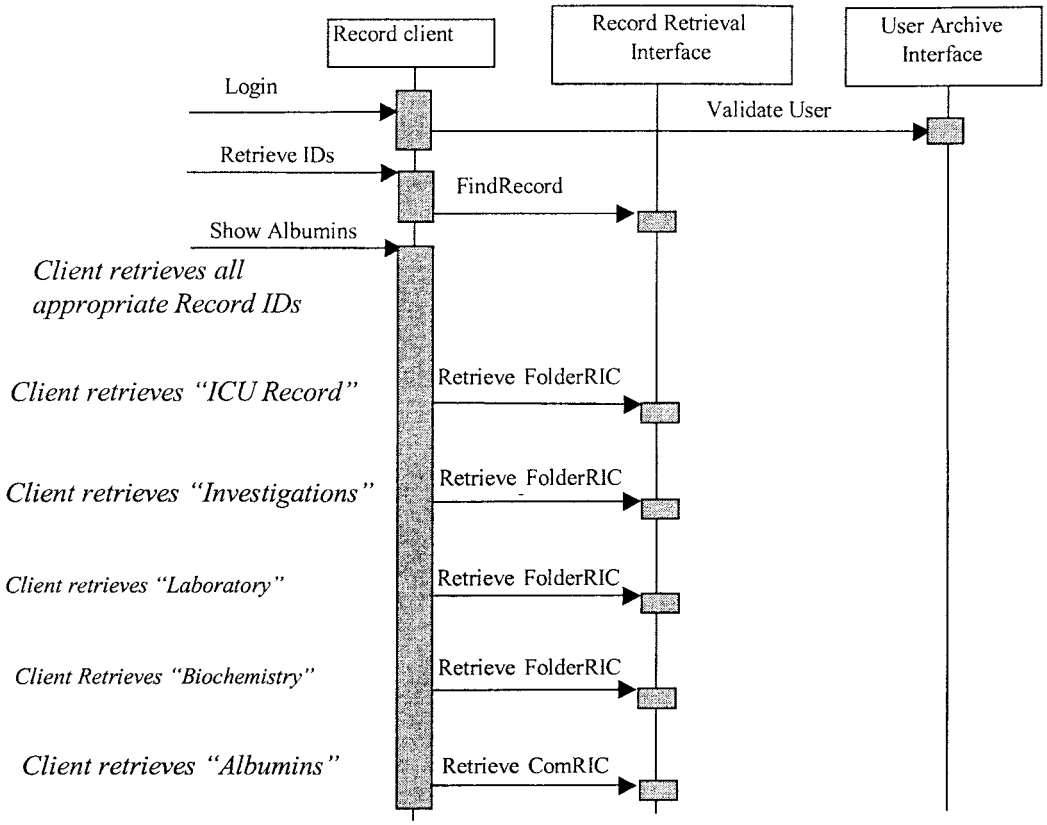

Fig. 6. High-level sequence diagram showing a typical set of calls to the IDL.

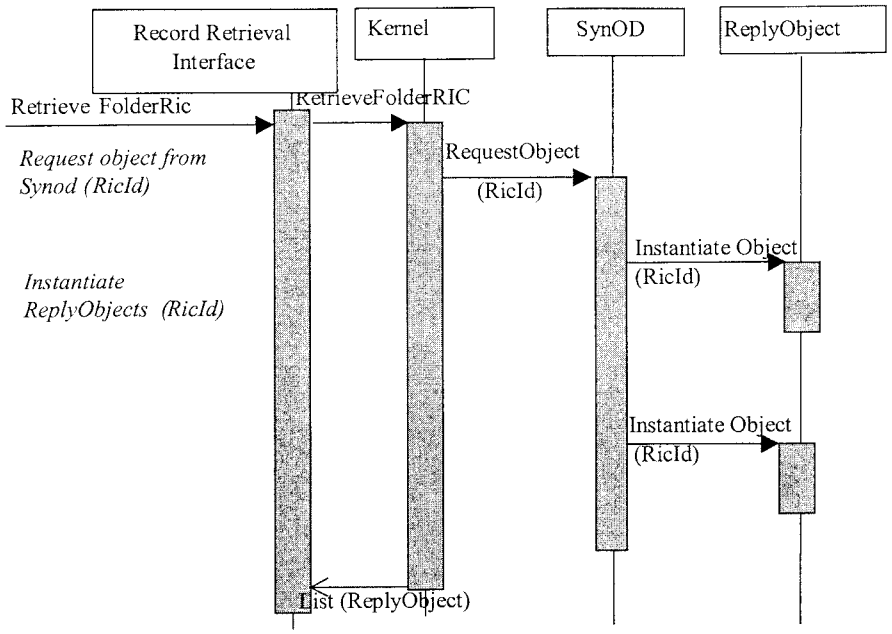

Fig. 7. Sequence diagram detailing a traversal resulting from a RetrieveFolderRIC call to the record retrieval interface.

ComRIC is requested. This RicId is used as a general index to many associated parameters. The data retrieval module in the server that accesses the data in the feeder system is configurable. Although it does not explicitly support connection to any relational database, the design is generic enough to allow for future integration of additional relational feeder systems. Fig. 8 shows how the server prototype responds to a RetrieveComRIC call to the record retrieval interface.

In this traversal, the server composes a set of reply objects that are returned as record components at the record retrieval interface.

4) Server Kernel–Record Fragment Construction within the Synapses Server: Since a central record data store is not available to the server prototype to store newly created record fragments, all requests for patient data to the server result in the construction of an "on-the-fly" Synapses container data

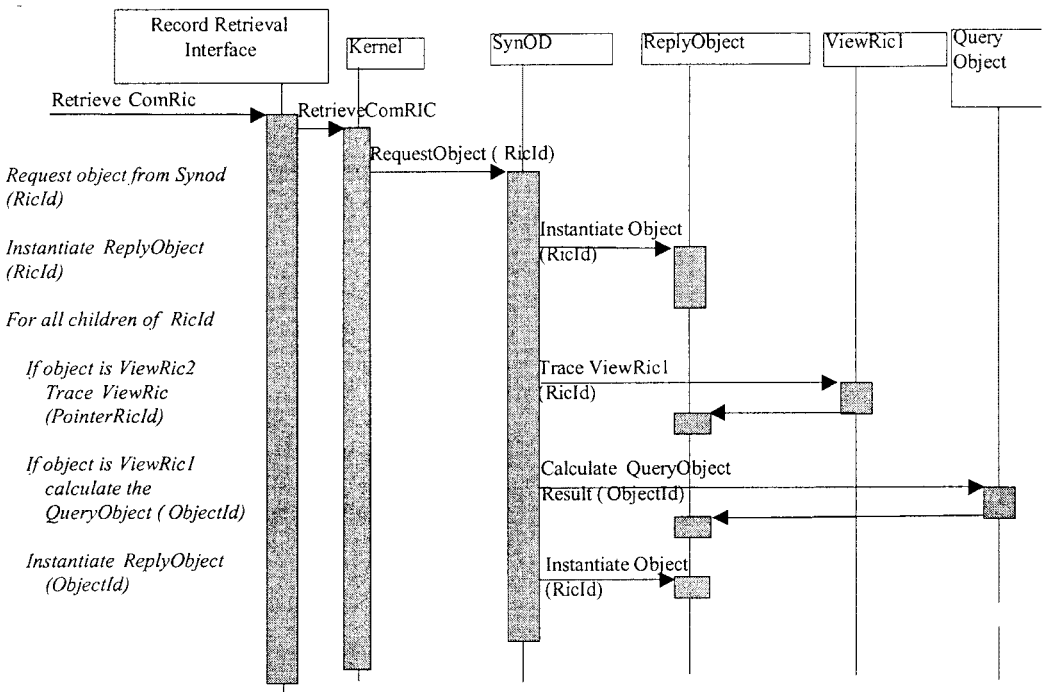

Fig. 8. Sequence diagram of a traversal resulting from a RetrieveComRIC call to the record retrieval interface.

structure to hold the data. This subsection describes how the SynOM and SynOD interoperate to form the "record fragments" that can be passed on request to the client.

The Synapses container data structure is a collection of SynOM classes (e.g., ComRIC, PrimaryData, RecordFolder, etc.). The information on which classes are used and how they are linked to each other to form the relevant record fragment container structure is obtained from the SynOD. The data required to populate the record fragment container can be one of two types: static or dynamic. Static data are the same for each instance of the record fragment. Since these data do not vary between records, they can either be stored once in the SynOD or as part of the server code. Dynamic data vary according to time, circumstance, patient identity, etc. and, therefore, need to be recovered from the appropriate feeder systems. Some record data, such as signatures and comments, will not be stored on the feeder systems. Data of this type must then be stored on the server on a per patient basis.

When a request for a particular RIC arrives at the server, the server follows six steps to complete the request, as follows.

a) In response to a RetrieveFolderRIC or RetrieveComRIC call from the end-user application, the server issues a request to the appropriate SynOD object to recover container information for the particular fragment of the record that has been requested.
b) SynOD constructs an empty record container. This container is a recursive structure that can be filled with the relevant data. As part of the container construction process, static information is copied into the container from the SynOD. This information consists of EnclosedRIC, SucceedingRIC, LogTime, and LogSign. The RecordID and RicId are also added at this point.
c) Data recovery methods from the server–feeder interface module are invoked on the relevant feeder systems. The results of these methods are parsed and formed into a set of data that will be used to fill the container.
d) If a particular piece of data appears more than once in the result set in step c), more container objects are instantiated to hold the data.
e) Data are copied from the result set into the container.
f) The now filled container is converted into an Orbix-CORBA specific form and returned to the client through the interface.

5) Server–Feeder System Interface Using ODBC: ODBC is used to connect the Synapses server prototype to the distributed feeder system. When the server is about to issue a query, a module in the server uses meta-information stored in record items in a ViewRIC1 to form the query. When the response is returned, this same module dynamically creates and fills record items with the data.

In line with the idea of interoperable record servers and feeder systems, it is intended that in the future servers will communicate with other record servers and record-based feeder systems by exchanging Synapses record components using XML.

6) Some Comments on Using the Extendable SynOD:

- SynOD is flexible and allows users to develop a record structure that suits their enterprise. This also means that it is more likely to be able to cover different healthcare domains than a model that is explicit and difficult to extend.
- Server presents a very simple (CORBA) interface to the client that allows access to records. This contrasts with the fixed model approach that, given the complexity of a healthcare record, would tend to lead to a detailed and complicated server interface.
- Unlike a detailed and fixed model of the record that gives application developers an explicit-but-complex model on

which to base their implementation, the approach described in this paper requires that the client accommodate a changing record structure. This requires added intelligence on the client side.
- There are storage and operational overheads associated with using full record items in place of simple class attributes.
- Flexible nature of the SynOD can lead to interoperability problems in the short term. For instance, using a detailed and fixed model of the record, if every record server uses the same model, there is less chance of misinterpretation of patient record data when they are transferred between systems.

D. IDL Description of the Record Retrieval Interface

The Synapses server specification does not dictate which technologies should be used to implement a Synapses server. Technologies such as ActiveX, CORBA, XML, and Java have been used at other Synapses sites. However, the implementations carried out in both Dublin and Amsterdam have been based on the use of CORBA as the communication mechanism.

The use cases for the Record Retrieval Interface shown above can be expressed using the following IDL:

```
//Record Retrieval Interface

interface RecordRet{
//operations
void FindRecordIDs
(in UserIdTyp UserId in SearchItemsTyp,
  SearchItems out RecordIdSq FoundRecordIds);
void FindContextRic
(in UserIdTyp UserId, in RecordIdTyp RecordId,
  in RicIdTyp ContextRicId,
                    out
  ContextRicIdSq, FoundContextRicIds);
void RetrieveFolderRic,
(in UserIdTyp UserId in RecordIdTyp RecordId,
  in DispDateTyp DisplayDate, in RicIdTyp
  FolderRicId,
                          out
  ContextRicSqFolderRicContent);
void RetrieveComRic
(in UserIdTyp UserId in RecordIdTyp RecordId,
  in DispDateTyp DisplayDate,,
  in RicIdTyp ComRicId
                               out ComRIC
  ComRicContent);
};
```

In Section III-C, it was seen that the record retrieval interface provides an entry point for the client, which makes recursive invocations of the RetrieveFolderRIC and RetrieveComRIC operations to create the record in its own address space. It is apparent from the definition of the RetrieveFolderRIC and RetrieveComRIC operations that there is still something missing—this is the Synapses record exchange format. The Synapses exchange format is a specification of interfaces to the concrete RIC and record item classes of the SynOM. This exchange format takes the form of a set of IDL interfaces. A small but representative portion of this IDL—an IDL description of the ComRIC interface—is included below:

```
...
typedef sequence < RecordItem > RecordItemSq;
typedef sequence < DataRIC > DataRICSq;
...
...
//ComRICInterface.
interfaceComRIC : ContextRIC
{
        //Aggregation using sequences
        readonly attribute DataRICSq
        CRContentsDR;
        readonly attribute RecordItemSq
        CRContentsRI;
};
```

IV. Application Domain Results/Previews

A. Overview

The principal output of Synapses is a set of specifications of the Synapses object model, the SynOM and its associated object dictionary, the SynOD, and the server and its interfaces. As with all telematics application program projects, Synapses places great emphasis on the importance of validation. "Validation is the process of evaluating a system or components during and at the end of the development process to determine whether it satisfies specified requirements" [29]. In Synapses, the objectives of validation are identified as follows:

- verification of the results of Synapses through selected pilots, with broad coverage of both clinical and technical domains;
- demonstration that the solutions found within the domains of the pilot projects are also applicable in a wider context by extending the validation to new sites.

The five verification sites are SJH (intensive care), Royal Marsden Hospital NHS Trust (oncology), Academic Medical Centre, Amsterdam (shared care—diabetes), Central Hospital of Akershus, Oslo, Norway (internal medicine and general surgery), and Geneva Canton Hospital (general), Geneva, Switzerland. An additional four sites are introduced in the demonstration phase. The aim is not only to validate the specifications in a variety of geographical settings and clinical domains, but also using a variety of underlying

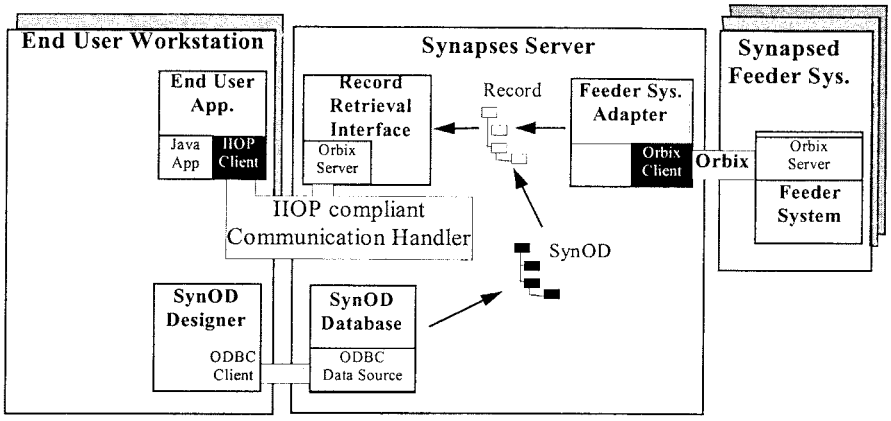

Fig. 9. Engineering/technology description of the AMC Synapses implementation.

technologies, including, in particular, distributed object technology. Two of the main sites have implemented the Synapses specification using Orbix, Iona Technologies implementation of the Object Management Group, Common Object Request Broker [30].

B. AMC Synapses Server Implementation

The AMC has implemented the Synapses server to support shared care between two groups of general practitioners and a hospital (endocrinologist and diabetes nurse specialist) for the care and management of patients suffering from Diabetes Mellitus Type II. The prototype supports general practitioner (GP) referral to the hospital, GP request for and receipt of advice form either the endocrinologist or the diabetic nurse, as well as providing access to a shared diabetic patient record [31].

The Amsterdam prototype, Fig. 9, features two client applications for viewing medical record information, as follows:

1) GP client, an internet-based client for the general practitioner;
2) Mirador—a Windows 95-based medical workstation for the internist and diabetes nurse.

Both clients are connected to the Synapses server by means of specific client adapters. These adapters translate requests and responses exchanged between the clients and the Synapses server. The prototype also includes the following feeder systems:

1) MicroHIS—a UNIX-based GP system;
2) ARCOS—a PC-based GP system;
3) HISCOM HIS—a UNIX-based hospital information system.

The server combines diabetes related patient information from the three-feeder systems to form the record.

C. SJH Synapses Server Implementation

The target environment for Synapses in SJH is the ICU, where patient information is required to operate a patient management system (PMS). The central component in the architecture is the Synapses server, which obtains patient information from the following feeder systems:

1) centralized laboratory information system;
2) HIS;
3) blood gas analyzer;
4) vital signs monitors.

In this implementation, each of the feeder systems will be accessible using SQL, although it is possible also to use non-SQL databases. The connection between the server and some databases will be supported through the use of ODBC. The connection between the client and server is implemented as specified in the Synapses specification and using Iona's CORBA technology, Orbix [30] (see Fig. 10).

The end-user application at the SJH site has been developed using Microsoft Visual Basic and uses an ActiveX-CORBA bridge to invoke operations on the server prototype. The current Synapses server prototype has been constructed using Orbix version 2.1C and Microsoft Visual C++. It retrieves the data required to build the record from the feeder systems using ODBC. The next version of the server will connect to the laboratory information system using SQL.

V. CONCLUSIONS

A. General Comments

As has been indicated, the principal output of the Synapses project is a specification of the Synapses common object model, the SynOM, together with its associated dictionary, the SynOD, and of the Synapses server and its interfaces. These specifications are currently undergoing rigorous validation in a variety of geographical settings and clinical domains and using different technology solutions. Of particular relevance to this paper is the use of CORBA as the underlying communications technology at two of the five main sites: Amsterdam Medical Centre (AMC) and SJH. These two sites demonstrate two quite different environments for the delivery of shared care and impose quite different demands on the Synapses server. The ICU is a highly data-intensive environment requiring the close integration of complex data, both synchronous (e.g., from online monitors) and asynchronous (e.g., from laboratory in-

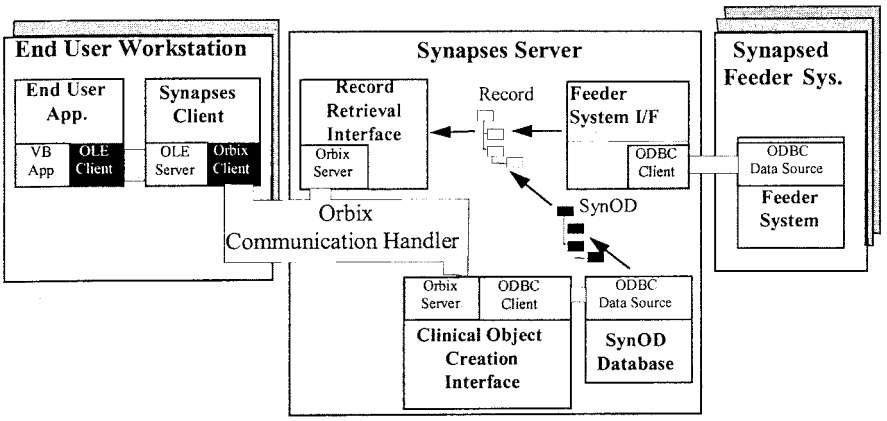

Fig. 10. Engineering/technology view of the SJH Synapses server.

formation systems). Shared care of patients suffering from a chronic disease, such as diabetes, represents a more loosely coupled environment with less time-critical interactions between carers. It is clear from the validation so far that both the Synapses specifications are sufficiently flexible to support what are effectively almost the two extremes of the shared care spectrum. Client applications can use the Synapses server to build both a shared intensive care and a diabetic record, incorporating information from a variety of legacy feeder systems. The prototypes that have been built to date are limited in scope, but the performance using CORBA appears to be satisfactory with most responses being delivered to the client in under 3 s.

Other sites in Synapses—a total of nine in all—are using a variety of other technologies, and the specification is proving robust and flexible to accommodate these approaches as well. Currently, a set of guidelines is being developed that will detail the steps required to migrate gradually to a Synapses environment. A major advantage of this approach is that it facilitates the migration to an EHCR, the FHCR, by populating the record with data from existing heterogeneous systems rather than requiring major investment in new technology.

Secure exchange of information between the feeders, the Synapses server, and client applications means dealing appropriately with the issues of authentication, encryption, and authorization. Although security is not formally to be addressed as part of the present project, it is clearly an essential feature of the Synapses approach and is being addressed in a limited way. In particular, it is planned in association with the Ishtar project [32] to identify the security requirements for Synapses in light of legal and other considerations. Individual demonstrators of Synapses, in which they are dealing with real patient data will, of course, have to ensure confidentiality and security, but it may not prove possible to implement a common strategy across all sites. A very important decision to be made when dealing with these issues is how to allocate the responsibilities. Both encryption and authentication are the responsibility of the distributed system as a whole and must therefore be handled centrally. It is envisaged that, in many Synapses installations, it will be necessary to encrypt all communication between clients, feeders, and the server. This implies that a uniform encryption policy must be selected and used by all components. Furthermore, an end user must only be required to identify themselves once (single signon). This implies that a central authentication facility must be offered. Once a user has signed on to the distributed system via the Synapses server, their identifiers are passed on to the respective feeder systems when data are requested. However, authorization is the responsibility of the different feeder systems, as it is they who "own" the data.

Other open issues include full support for server-to-server interoperation as well as the important issue of unique patient identifiers, which is a particularly challenging problem in those countries that do not have national patient identifiers. However, even when there is a unique national identifier, it is not unusual to find a variety of different patient identifiers being used by patients in different systems. The CORBAmed person identification service specification suggests basically two ways for correlating patient identifiers: attended and unattended mode. In the former case, end users are presented with a list of matching alternatives and they make the final selection themselves. In the unattended mode, the final selection is made automatically [25]. The strategy within Synapses would then be to map general identifiers onto specific feeder identifiers as close as possible to the feeder systems, preferably in the adapters to these feeder systems.

The use of agent technology is also being investigated, particularly in relation to the location of records. A prototype agent has been developed that actively searches for and gathers links to records in other record servers. The links can be added by the user to electronic records in the "local" server. It is envisaged that agents of this type would operate in a hierarchical manner, with each successive layer of agents in a traversal providing additional location details across fewer information systems.

B. Some Comments on Using CORBA

There is a trend in hospitals toward increasing automation and distribution of instruments and data. As medical technology is notoriously heterogeneous in nature, it can be

assumed that in the short to medium term there will be an increasing requirement for technologies that can cope with distributed heterogeneity. It has been well documented [28] that CORBA facilitates interoperability between heterogeneous distributed systems while hiding the complexity of the networked environment. The added benefit of CORBA is that it allows both data and services to be specified. While it could be argued that a record only consists of data, a fully automated record system requires security, user validation services, patient identification, and correlation services, among others.

A record server may not only use the pull-based approach of asking for data and getting it, but also a subscribe and receive approach, which would be used for real-time information, such as vital signs data. CORBA provides a standard way of integrating data and services of this type. This type of service is essential for a full record server implementation in a real-time data intensive location, such as an ICU. Apart from presentation functionality, there is little requirement for the client to invoke operations on the record objects themselves. We found that CORBA was quite efficient at passing record objects between server and client. Nevertheless, it was concluded that in the absence of methods on record objects, the efficiency of the exchange could be improved if records could also be passed in a second mode of transfer, as a set of serialized strings (for example, as XML document fragments) in a CORBA response.

Substantial time delays were noted between the client and server when large amounts of data were transmitted. This is under investigation to see whether this delay is due to the Orbix/OLE automation server or due to the server side of the implementation.

Finally, the level of abstraction in CORBA means that it is possible for a group of designers, developers, and domain experts to discuss domain-specific problems in a way that is extremely useful for implementers. Use of IDL aided the discussion on aspects of the implementation work. For example, it could be used by client developers and the server interface developers to discuss and agree on objects that could be transmitted between them. It could also be used by the server interface developer and the server developer to discuss mapping between server code and interface code.

In summary, the experience gained in the Synapses project in the use of CORBA technology to support the integration of distributed EHCR's has been positive. Two demonstrators have been built, one to support shared care in an ICU and the other in diabetes care. The scalability and distribution transparency benefits to be derived from the use of CORBA will be even greater in more widely distributed environments involving server-to-server communication between hospitals, regions, and nationally.

ACKNOWLEDGMENT

The authors gratefully acknowledge the contribution by members of the Synapses Consortium to the work described in this paper.

REFERENCES

[1] A. C. Curtis, "Multi-facility integration: An approach to synchronized electronic medical records in a decentralized healthcare system," in *Proc. MEDINFO,* 1992, pp. 138–143.
[2] R. van de Welde, A. van der Werff, A. Kilsdonk, W. Damen, and M. Hubert, "Toward a European framework reference model and architecture for the development of open hospital information systems," in *Procs. MEDINFO,* 1992, pp. 188–193.
[3] V. W. Slack and H. L. Bleich, "Barriers to clinical computing in American hospitals," in *Procs. MEDINFO,* 1992, pp. 178–181.
[4] J. Dudeck, B. Blobel, W. Lordieck, and T. Burkle, *New Technologies in Hospital Information Systems, Studies in Health Technology and Informatics,* vol. 45. Amsterdam, The Netherlands: IOS, 1997.
[5] D. A. Bell and J. B. Grimson, *Distributed Database Systems.* New York: Addison-Wesley, 1992.
[6] A. P. Sheth and J. A. Larson, "Federated database systems for managing distributed, heterogeneous, and autonomous database systems," *ACM Comput. Surv.,* vol. 22, no. 3, pp. 183–236, 1990.
[7] M. Brodie and M. Stonebraker, *Migrating Legacy Information Systems: Gateways, Interfaces and the Incremental Approach.* New York: Morgan Kaufmann, 1995.
[8] W. H. Inmon, *Building the Data Warehouse.* New York: Wiley, 1996.
[9] http://www.oacis.com/.
[10] http://www.healthcare.com/.
[11] http://www.stc.com/.
[12] http://dumsscc.mc.duke.edu/ftp/standards.html.
[13] EDIFACT Syntax for interactive applications, UN/EDIFACT.
[14] J. J. Cimino, S. A. Socratous, and P. D. Clayton, "Internet as clinical information system: Application development using the World Wide Web," *J. Amer. Med. Inform. Assoc.,* vol. 2, pp. 273–284, 1995.
[15] I. S. Kohane, M. S. Greenspun, J. Fackler, C. Cimino, and P. Szolovits, "Building national electronic medical record systems via the World Wide Web," *J. Amer. Med. Inform. Assoc.,* vol. 3, pp. 191–207, 1996.
[16] H. J. Lowe, E. C. Lomax, and S. E. Polonsky, "The World-Wide Web: A review of an emerging Internet-based technology for the distribution of biomedical information," *J. Amer. Med. Inform. Assoc.,* vol. 3, pp. 1–14, 1996.
[17] C. E. Chronaki, G. Katehakis, X. C. Zabulis, M. Tsiknakis, and S. C. Orphanoudakis, "WebOnCOLL: Medical collaboration in regional healthcare networks," *IEEE Trans. Inform. Technol. Biomed.,* vol. 1, pp. 257–269, Dec. 1997.
[18] *Electronic Healthcare Record Architecture,* P. Hurlen, Ed., CEN/TC 251/WG1 N95-38, 1995.
[19] F. Ferrara, "Healthcare information systems architecture, new technologies in hospital information systems," in *Studies in Health Technology and Informatics,* vol. 45. Amsterdam, The Netherlands: IOS, pp. 1–9.
[20] W. Grimson and P. A. Sottile, "Synapses in the context of healthcare information systems," in *Studies in Health Technology and Informatics,* vol. 45. Amsterdam, The Netherlands: IOS, pp. 30–39.
[21] J. Grimson *et al.,* "Synapses—Federated healthcare record server," in *Procs. MIE'96,* J. Brender, J. P. Christensen, J. R. Scherrer, and P. McNair, Eds. Amsterdam, The Netherlands: IOS, pp. 695–699.
[22] W. Grimson, D. Berry, J. Grimson, G. Stephens, E. Felton, P. Given, and R. O'Moore, "Federated healthcare record server—The Synapses paradigm," *Int. J. Med. Inform.,* vol. 52, pp. 3–27, 1998.
[23] D. Ingram, *The Good European Health Record: In Health in the New Communication Age,* M. F. Laires, M. F. Ladeira, and J. P. Christensen, Eds. Amsterdam, The Netherlands: IOS, 1995, pp. 66–74.
[24] J. Grimson, E. Felton, G. Stephens, W. Grimson, and D. Berry, "Interoperability issues in sharing electronic healthcare records," in *Proc. 3rd IEEE Int. Conf. Eng. Complex Comput. Syst.,* pp. 180–185.
[25] http://www.org.omg.org/corbamed/home.htm.
[26] I. Jacobson *et al., Object-Oriented Software: A User Case Driven Approach.* New York: Addison-Wesley, 1992.
[27] Object Management Group, *The Common Object Request Broken Architecture and Specification,* Framingham, MA, 1991; http://www.omg.org/store/pub.htm.
[28] R. Orfali, D. Harkey, and J. Edwards, *The Essential Distributed Objects Survival Guide.* New York: Wiley, 1996.
[29] "IEEE Standard for software verification and validation plans," *IEEE Standard,* 1012-1986.
[30] http://www.iona.com/.
[31] O. W. Weier, M. Kalshoven, H. van der Kolk, F. Leguit, M. Ros, and P. Toussaint, "The federated healthcare record to support shared diabetes care," in *Proc. MEDINFO'98.* Amsterdam, The Netherlands: IOS, pp. 103–106.
[32] http://www.ehto.be/.

Jane Grimson (M'89) received the B.A.I. degree in computer engineering from Trinity College, Dublin, Ireland, the M.S. degree in computer science from the University of Toronto, Toronto, Ont., Canada, and the Ph.D. degree in computer science from the University of Edinburgh, Edinburgh, Scotland.

She has been with the Department of Computer Science, Trinity College, since 1980, where she is now an Associate Professor and Dean of Engineering and Systems Sciences. She is Project Manager of the Synapses project. She has written over 60 papers and co-authored a textbook on distributed database systems. Her main research interests include distributed database systems and health informatics.

Dr. Grimson is a Chartered Engineer, Fellow and Vice-President of the Institution of Engineers of Ireland, Fellow of the Irish Academy of Engineering, Trinity College, Fellow of the Irish and British Computer Society of the Royal Academy of Medicine, member of the Irish Council for Science, Technology, and Innovation, and member of the ACM.

William Grimson received the B.A.I. degree in electronic engineering from Trinity College, Dublin, Ireland, and the M.Sc. degree from the University of Toronto, Toronto, Ont., Canada.

He was with Ferranti Ltd., Edinburgh, Scotland, where he worked with the Laser Systems Group for three years. He is currently Assistant Head of the School of Electrical Engineering, Dublin Institute of Technology. His research interests include medical informatics, in which he heads a small research group whose activities include the development of applications-supporting clinical laboratories, development of electronic patient record systems, and standardization in CEN TC 251 and CORBAmed. Recent and current European Union-funded projects in which the group has participated include OpenLabs, Synapses, SynEx, and Tudor.

Mr. Grimson is a Chartered Engineer and a member of the Institution of Engineers of Ireland.

Damon Berry received the B.Sc. degree in control systems and electrical engineering from the Dublin Institute of Technology, Dublin, Ireland, and the M.Sc. degree in computer science from Dublin City University.

He is currently involved in various European Union-funded projects at the Dublin Institute of Technology concerned with the design and development of medical software for Irish hospitals. He specializes in healthcare record architectures, medical instrument interfacing, and test request protocol systems.

Gaye Stephens was born in Dublin, Ireland, in 1967. She received the degree in computer science from the Dublin Institute of Technology, the degree in computer science from the British Computer Society, Swindon, Wilts, U.K., and the M.Sc. degree from Trinity College, Dublin. She is currently pursuing the Ph.D. degree at Trinity College.

She has been a Research Assistant with the Knowledge and Data Engineering Group (KDEG), Department of Computer Science, Trinity College, since 1994. Her research interests include end-users' views on and client aspects of distributed systems.

Eoghan Felton was born in Lusaka, Zambia, in 1973. He received the B.S. degree in computer science in 1995 and the M.S. degree in 1996, both from Trinity College, Dublin, Ireland.

He is a Research Assistant with the Knowledge and Data Engineering Group (KDEG), Department of Computer Science, Trinity College.

Dipak Kalra was a General Practitioner Principal for eight years until April 1995 and has been involved in the research and development of GP computer systems since 1989. He is currently a full-time Clinical Senior Lecturer with the Centre for Health Informatics and Multiprofessional Education (CHIME), University College, London, U.K. He has an active role in European research in the field of electronic healthcare records, including the Good European Health Record (GEHR) project, funded under the European Union's Advanced Informatics in Medicine Program, Synapses, and Synex, funded under the Health Telematics Program. His research interests include the organization of healthcare and the development of methods for making more effective use of healthcare information, encouraging the wider use of audit methods and information technologies.

Dr. Kalra chaired the City and East London Medical Audit Advisory Group for six years until 1996.

Pieter Toussaint received the B.S. degree in general linguistics in 1990 and the M.S. degree in computer science, both from the University of Leiden, Leiden, The Netherlands. He is currently pursuing the Ph.D. degree at the University of Leiden.

He has been with the Research Department, HISCOM, Leiden, since 1993, where he is a Senior Researcher/Consultant. He has published several papers on the subject of integration of information systems.

Mr. Toussaint is a member of the IEEE Computer Society and the ACM.

Onno W. Weier received the M.Sc. degree in electrical engineering in 1988.

He was with Nucletron Research, Veenendaal, The Netherlands, from 1989 to 1994, where he was responsible for the development of an image processing system to be applied in cancer treatment. He joined HISCOM, Leiden, The Netherlands, in 1994 to develop a PACS for nuclear medicine. At the end of 1996, he became Project Manager for the Dutch contribution to the Synapses project, which concerns the development of a federated medical record server. In 1998, he was appointed to Research Manager and is responsible for the initiation and management of all of the company's (inter)national R&D projects.