Journal of Universal Computer Science, vol. 4, no. 11 (1998), 856-864
submitted: 27/10/98, accepted: 15/11/98, appeared: 28/11/98 © Springer Pub. Co.

Telematic Platform for Patient Oriented Services

Bernhard Zwantschko
(IICM, Graz University of Technology, Austria,
bzwan@iicm.edu)

Dieter Freismuth
(IICM, Graz University of Technology, Austria,
dfreis@iicm.edu)

Klaus Schmaranz
(IICM, Graz University of Technology, Austria,
kschmar@iicm.edu)

Abstract: This paper describes in brief a basic framework for implementing a telematic-platform for patient oriented services. We first show the current situation in the medical field and work out the requirements for an integrating platform. Afterwards we introduce Active Node Technology (ANT), the technology the platform will be based on.

1 Introduction

The always increasing life expectancy and the increasing number of long term and care patients has led to a dramatic explosion of costs. The goal of implementing a telematic-platform for patient oriented services is to increase the quality of health care while reducing the costs through an efficient employment of information and communication technology. A consortium in Germany headed by the Deutsche Luft- und Raumfahrt (DLR) is developing a system which will provide national access for all.

The basic element of the system will be an integrative, network based workflow system with a realtime database – the telematic-platform. The platform should provide a simple and easy to extend standard, bringing together all involved parties as there are patients and health care staff and other users, service provider or device manufacturer. This will set national and international standards and thus reduce the uncertainty for developers and service providers. Based on this platform, health services will be provided. Some examples of planned services are: monitoring of heart circulation patients at home (e. g. before and after an operation), communication services for care patients, integration of doctor’s practices and emergency services, administration services, expert databases and more.

Patient data including medical as well as administrative information is stored at the moment at many different places on different systems. The doctor often lacks the results of previous examinations and thus repeats the examination or even more dangerous, does not make the right diagnosis. Using the platform, patient data will still be stored distributed physically. But it will be virtually available as a single, structured patient record. Data in such a complex system can be of various type. There can be simple textual information, multimedia documents as X-ray images or even realtime data like ECG or EEG records. For that reason a further requirement for the platform is to be open for current and future data standards.

On the other hand patient data may be very sensitive. Only authorized people may be allowed to get access. Sometimes even the patient himself may not be authorized to read his data (think of a critical illness of a psychological unstable person). Beyond the problematic of this in general, there is a strong request for security and confidentiality on the platform.

In the following section we will work out the requirements for a telematic-platform and show the improvements it can make over the current situation. After that we introduce Active Node Technology, a data and communication framework which serves as the basic technology for the platform.

2 Current Situation and Requirements for a Platform

2.1 Integration

At the moment in the medical sector there is a large number of incompatible platforms. This is due to the fact that existing systems are only designed for one local zone. (E.G. only for one specific Hospital) Further the aspect of data exchange has been treated to be unimportant or even has not been taken into account. Up to the last few years there were no adequate transport mediums available, but with the rapid growth of networks we now have the possibility to think about the integration of different systems.

Although sometimes a complete reengineering of such local systems will be necessary to get them fit into a national or international platform, the main goal of integration must be the preservation of investment. Therefore an integrative platform must be considered as a higher layer serving as an interpreter. Gateways allow to extend the framework and to 'plug in' existing or newly developed systems.

Existing systems are implemented not only for local use, but they are based on different platforms technically. Thus an integrative layer should be platform independent. Java as a platform independent and worldwide accepted and widespread language will serve this requirements best. In addition Java allows both, to get the interface of the platform simple and easy to extend.

A simple interface is important to reduce development costs for system integration and to get accepted by the industry. Java is widespread, it has a simple language syntax and comes with a large library. Together with a simple interface definition this solves the conditions for an easy platform. Second, an easy extension of the platform is indispensable. We can not know yet which requirements future medical services will request. With Java we can use dynamic loading of classes, allowing simple integration of new modules.

2.2 Distributed System - Global View

It will not be possible to store and manage the huge amount of data the platform will have to handle centrally. Nevertheless, the user of the platform will want to get a global view on relevant data. In addition, not only data will be handled by the platform, also distributed services will be offered. And the user should get access nationwide or even worldwide.

To fulfill these requirements we use the concept of distributed server and clients but offer a global address space for all data and services. Think of this virtual address space as a large file-system, organized hierarchically. Navigating down the tree, the user will visit different spaces and servers transparently. Although the user connects to different machines and uses different services he has the imagination of serving one single space.

The users view of the medical framework depends on his current role, (doctor, stewardess in an emergency role, ...) on his current location and on his personal settings. Although the address space is global for each user it is not unique or static. Quite the reverse, each user gets an adapted view, fitting his requirements. This is a very important feature of the framework. Heaving only one static address space would very soon led to a lost of comprehensibility.

Additionally, clients and server in this framework get the same interface and can therefore interact with equal rights. A gateway or another software module written for a client can be used on a server as it is. On the platform, the borders between client and server become blurred and sometimes they will even change roles. Thus the platform is more a network of partners than of clients and server. This makes development and integration of new services an easy task.

2.3 Collaboration

Most current systems focus on the storage, processing and retrieval of data. A telematic-platform for patient oriented services must include collaboration. Collaboration not only as a support for communication like email or chat. Collaboration in this environment means possibilities to perform collaborative work on data. An example could be the "online specialist knowledge". A practical doctor who asks a X-ray specialist for his opinion on a distinct X-ray image. Or the request of one doctor on another to check his diagnosis, knowing that he is a specialist for that kind of illness.

A platform with collaboration features will better connect people, using knowledge more efficient and overcome the problems of geographically distant operating medical staff. Collaboration will strongly be used between care patients at home and the hospitals and e. g. between a flight attendant treating an emergency patient and a doctor on the earth. Such collaboration data might be ECG data to find out the danger of an emergency or to control the patient while he sleeps. Thus the support of collaboration is one very important requirement for the platform.

2.4 Multimedia

Even if you gain control of the problematic of data exchange between different systems there is a further problem: Current systems do not use the potential of multimedia data. Although some systems can handle specific data like ECG streams they are not able to link them together. And a complete patient record consists of all possible data types. To make a diagnosis, the doctor needs all the information of a patient, current and previous.

Thus an adequate platform must support all types of multimedia and real-time data. Not only known data formats must be include, the platform must be open to integrate data types and services not yet invented. A powerful concept is to use a self contained object oriented approach. Data, and means to handle that data (e.g. viewing or editing) are bundled and shipped together. Thus inventing new data types is very easy. As a technical solution for this approach we choose the usage of the programming language Java in combination with the Java-Beans Activation Framework of Sun Microsystems. We assume a rapid spreading of this technology and have therefore included the concept in the platform.

2.5 Structured Data

Patient data will become very extensive and many different people are interested in this data. To serve the needs of the user, the information will have to be structured effectively. Structuring has a number of aspects. A defined structure for patient records means to provide different views on the same data. So a doctor may view all the data of one examination or all data according to a special discipline like X-ray or eye diseases. A defined structure is necessary to make different patients data comparable either for expert systems or for statistical research.

But structuring cannot stop at the document level. The document itself will or must be structured. Concepts like HTML are not adequate since structural information and content are mixed in one document. The upcoming XML standard may be one solution to think about. At the moment one information management system fulfills most of the requirements: Hyperwave [Maurer 96]. It is object oriented and allows to create well structured information and to include multimedia.

2.6 Security

One of the most critical aspects for the acceptance of a telematic-platform for patient oriented services is security. It is no service of the platform it is a necessary condition. Security in this area covers two different aspects: restrained and controllable access to data and confidentiality.
It is obvious that patient data is very sensitive. Additionally access and management of data is restricted by law. Only authorized user should gain access to different types of data. But it is not that easy to determine who and when should get which access rights. Doctors, politicians, patient representational and different organizations will discuss the problematic possibly forever. In addition access rights are very often case sensitive, think of an emergency situation as an example.

A platform must include a security concept, including the use of smart cards, encryption, secure data transfer, dynamic user management and access logging. (With access logging each request for sensitive data is logged and the affected person gets periodically reports of all people which had access to the data. Thus access can be checked and if abused be punished by justice.)

The second aspect of security is confidentiality. If the patient record exists in written paper, the doctor can view the signature on the document and thus trust it. If stored electronically, the author of each document must be unequivocal acertainable. Therefore, each document will have to be signed to guarantee that the document has not been changed by another person. At this place it should be mentioned that the term document is more complex than it seems. To get its correct meaning a single document often is based on a number of other documents. If one of the applied documents is removed or changed, the note or diagnosis may get meaningless or even wrong. This means that a platform will have to include versioning and signing of documents within their context. The problem of confidentiality is not treated in more detail in this document.

2.7 Roles

At last we want to describe another requirement for a telematic-platform. We mentioned above, that there are a large number of different people who will use the platform. And each of them will usually have more than one role in the system. A doctor will also be a patient, a flight attendant as normal user may sometimes be in the role of an emergency staff, a medical student may be interested in statistical data as well as having to treat some patients and will have night duty with responsibility for more patients than over the day for example.

This implies the requirement for an easy to manage, highly dynamically user management. It also has to provide the possibility to gain extra access rights without complicated administration. (e.g. in case of an emergency situation)

In addition the system must provide different views of the whole system depending on the current role the user is in. This includes data as well as services. An administrator of a health insurance company will need data like patient Id or cost of a stay in hospital but should never gain access on medical data while it may be vise a versa in the doctors view. Other possible views are anonymous statistical data for research or controlling a medicine. Another useful service may be an expert database which possibly automatically checks the patient data for conspicuous patterns.

3 Active Node Technology (ANT)

The following chapter introduces ANT, the basic technology of the platform. ANT will fulfill some of the requirements worked out above, others will be reached by adding components to the ANT network, like Hyperwave server technology, security and encryption modules, logging services and more. ANT is implemented in pure Java. ANT is therefore per definition platform independent. Loading of classes is done at runtime, thus dynamic extension is inherited from Java too.

Active Node Technology or short ANT is a technology to provide a virtual global space of interconnected active objects. The virtual address space describes the base idea of this technology. But only when considering the other constituting functions of ANT it reaches its full potential. The functions are: implicit structuring of objects, property support, asynchronous as well as synchronous communication, a specified service interface, included security mechanisms and not at least strict distinction between location/protocol and data (mime type).

3.1 Virtual Address Space

The virtual address space is much like a file system, especially like that one of UNIX. The addressing starts with one and only one single root object. All other objects are children of the root object or their children. So all objects in ANT constitute a single, hierarchical but virtual structure. The double dimension of the word virtual will be worked out later.

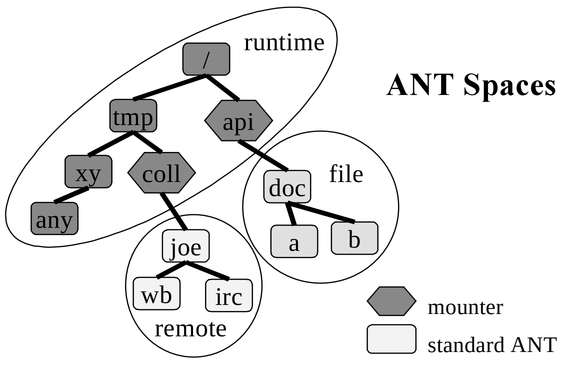

Figure 1

Since each object has a single position it can be assigned an unambiguous address. As an example, the address of the object 'joe' in the image above is `/tmp/coll/joe`. Each object can therefore be identified and found very easy. Looking at the tree structure of the object hierarchy it is obvious, why we choose an explorer, the ANTExplorer, as the basic tool for operating with ANTs. (Here and later ANT is used as a term for an object in the hierarchy. The term ANT is no accident. It should make clear, that we operate with interconnected and communicating, independent objects.)

The top of each virtual address space is formed by the so called runtime space. The ANT of this space only exists in memory. Subspaces of that may be objects at any location. There can be file ANTs, http ANTs, remote ANTs and more. Subspaces are generated by a mounter. The procedure to mount a subspace is similar to mounting in UNIX. There you first generate a directory (`mkdir`) and then you mount (`mount`) any device on it. Thus you can access the content of e. g. a CD-ROM as if it is a subdirectory of the mounting point. The picture above shows two spaces mounted, where `api` and `coll` are the mounters. The mounter is always part of the parent space.

Mounted spaces like the content of a Hyperwave server may not exist physically on the mounting Java Virtual Machine. Only the mounter really exists, the spanned subspace is provided by the mounter as a virtual subtree. But this is transparent to the user, all objects seem to be part of one large address space.

Spaces mounted in ANT are not only passive data like data on a file system or on a WWW server. You can also mount the ANT tree of another machine and thus get access to all objects mounted there. So if each server in a network mounts a number of other servers you will find the content of all of them by virtually navigating down only one single address space.

Beyond the mounting of data access protocols like ftp or file and mounting of other ANT trees, you can use the mounting facility to mount services. Thus a logging service or a telephone book service will be mounted the same way as structured data. The folders and subfolders will show a structured view of offered services. An electronic store thus can provide a view of all products structured by department and additionally by label. Selecting a product you will get all information you need about it and you will be able to buy it e.g. by usage of drag and drop, or any other simple interaction. If not needed, the mounter may even have no children just providing some services without the usage of structuring.

3.2 Structure modification

One fundamental function group of ANT is the structure modification operation. If you for instance want to move the file `a` into the remote machines folder `irc` you simply process the following instructions (simplified):

```
Ant a = new Ant ("/api/doc/a");
a.move("/tmp/coll/joe/irc");
```

Further, ANT has native support for creation, removal, copying, moving, linking and mounting. So the most important operations for e.g. publishing documents are built in functionality of ANT. Using the native ANT tool, the ANTExplorer, this operations are simply done with drag and drop or menu calls. Downloading of documents from any location using any protocol works in the same way.

Moving or copying of objects is very simple. Since objects in ANT are alive, they can move them self thus implementing agents. If they want to leave a secretary at the old place they simply create a link. Provided the access rights, objects can travel around gathering information and processing data.

3.3 Properties

All ANTs support properties, key – value pairs. You can perform all common operations on this properties. So you can modify access rights on a file system by changing properties or adding a second name for an object in Hyperwave. With the ANTExplorer you can modify properties by using a submenu of the context menu or the file menu.

3.4 Communication

ANTs have built in support for synchronous as well as asynchronous communication. In addition you can invoke any method on the destination object (if allowed). The mechanism is much like RMI [Sun Microsystems 97] but usage is easier. The ANTs thus provide all the means to simply set up collaboration services from simple chat mechanisms over workflow controlled collaborative work to collaborative work on shared data.

3.5 Services

Each ANT can define its own services. This may be simple services like providing a local print stream, more special services like property editors or even extended services like server administration, workflow management, user logon and management and more. Services may have a visualization, most times they will have one. With the ANTExplorer you will access services by first selecting the ant you want the service from and then selecting the appropriate submenu in the context menu or in the tools menu.

Providing services is a very powerful and simple means of ANT. To provide services you simply write a special mounter, e.g. a conference submission mounter. Once mounted, any user can navigate to your submission folder. There he will be presented some nice documents and in addition find all your services in the context menu of its ANTExplorer. Submission of a document is simply done with drag and drop or by providing an URL.

3.6 Data

One of the most important features of ANT is the separation of data on the first side and protocols and structure on the other side.

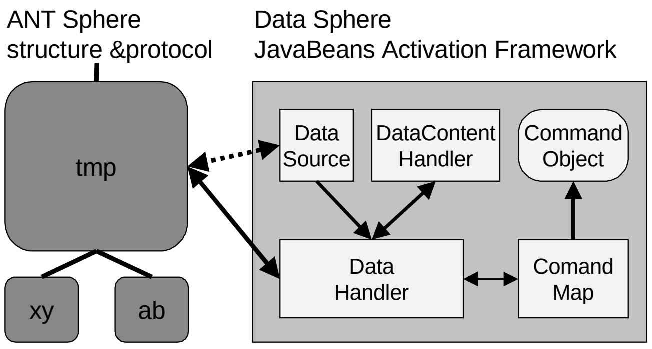

Figure 2

The ANT sphere provides all the handling of different protocols. It is completely transparent for the user where the data comes from and which protocol is spoken. The ANT sphere provides a structure, the virtual address space. The data share is completely separated. It is based on the mime type of the data and is not interested in any protocol or structure. We use the JavaBeans Activation Framework [Calder 97] of Sun Microsystems (in short JAF). Using ANTs you simply say getData() setData() to access the data while data-handlers will do the work for you. So storing a file or publishing a document on a http server seems to be the same. And uploading files to a server is simply done with drag and drop in the ANTExplorer.

The JAF provides commands for each data type. Commands can be editors, viewer and more. To get a command for a distinct ANT you simply use the context menu of ANTExplorer or the file menu respectively.

3.7 Security

As mentioned above, security and confidentiality is fundamental for a telematic-platform for patient oriented services. ANT provides simple and transparent means to deal with this requirement. The first security mechanism concerns data transfer. Connections between different ANT systems are done via secure connections (e.g. SSL), thus providing security at the transportation layer.

The second mechanism uses a ticket validation system for checking and logging each request for secure data. Each time a user tries to access protected data, he is requested to provide a suitable ticket. This will be done by a ticket factory, a built in service of ANT. The ticket itself is a multiple encrypted piece of data, containing information like signer, date of issue, valid period, issuer and more. When creating a ticket the factory pops up a dialog to inform the user of the type of request he should sign and will possibly ask the user for an appropriate combination of login and password. When using smartcards the user will only have to agree the request by clicking ok. Each issue of tickets is logged at the issuing server, so supervision and later checks are possible.

4 Examples using ANT

Although invented as a concept for a telematic-platform for patient oriented services, the ANT concept is suitable for much more fields. Examples where we currently use ANT are Hyperwave server data visualization and navigation aids and server administration and publishing. The basic tool therefore is the mentioned ANTExplorer, so e.g. downloading of even multiple documents to the filesystem, or any other system, can be done by drag and drop.

Another example is a controlling tool for distributed organizations. Based on Hyperwave for storing data and providing the user management special CHRONOS ANTs provide services like creating a project plan, online recording of working time for each projects collaborator and tools to create reports and deviation analysis.

References

[Gosling et.al 96] Gosling, J., *The Java(TM) Language Specificaton*, The Java Series, Addison-Wesley, (1996)

[Maurer 96] Maurer, H., *HyperWave - The Next Generation Web Solution*, Addison-Wesley, (1996)

[Calder 97] Calder, B., & Shannon, B., *JavaBeans Activation Framework Specification*, http://www.sun.com, (1997)

[Sun Microsystems 97] Sun Microsystems, *Remote Method Invokation Specification*, http://java.sun.com, (1997)