Web-enabling an Integrated Health Informatics System

Angel Petrovski and John Grundy
Department of Computer Science, University of Auckland
Private Bag 92019, Auckland, New Zealand
john-g@cs.auckland.ac.nz

Abstract

Potential users of Health Information Systems include General Practioners, Pharmacists, Hospital Staff, Community Nurses and patients. Ideally an infrastructure for such a system will support the integration of data and processing as well as a diverse range of user interface technologies and devices. We describe a prototype system that uses a range of current Java-based object-oriented technologies to achieve this, including CORBA, XML, Jini, Enterprise Java Beans, and Java Server Pages (for both HTML and WML). We present an architecture for this system, key parts of its object-oriented design, examples of some of its user interfaces, report on our experiences building and evaluating this system. We identify strengths and weaknesses with these technologies we hope will be useful for others considering adding a range of web- and mobile-interfaces to enterprise systems.

1 Introduction

Many enterprise systems increasingly require the provision of a range of diverse human interfaces, including web-based, PDA and mobile devices [1, 2, 3]. For example, in Health Informatics there is typically a range of stakeholders who require access to various computing facilities. General Practitioners need to record patient information, diagnoses and treatments. Hospital staff need to record similar information, along with observation results (from both computerised devices and staff). Pharmacists require prescription information and community nurses treatment information. Other health providers, such as diagnostic laboratories, and funding agencies need to exchange information with hospitals and GPs at various times. Patients should be able to access their own medical records in a secure manner.

Ideally, we can envisage a system that allows all health professionals to share their information in a timely manner, integrated via a central "brokering" health system. This has the key advantages of permitting different professionals to access an integrated, consistent history of patient information and to be aware of a patient's complete medical history. However, many of these users need quite different user interfaces e.g. patients via the WWW, GPs and Pharmacists using desktop computers, hospital staff via desktops, PDAs and pagers, and community nurses via wireless PDAs. They also operate over a variety of networks e.g. dedicated, high-performance LAN, a potentially insecure wide area network, or mobile, low bandwidth wireless networks.

Many technologies and architectures have been developed for connecting such highly distributed system components. Examples include OMG CORBA and Microsoft's DCOM (for distributed object systems); HTML and XML (for web-based systems and information exchange); Java RMI and Jini (for distributed Java components and Java-enabled devices); and WML and WAP (for mobile devices) [1, 4]. We have developed an infrastructure using a variety of these technologies in order to realise a prototype of the integrated health informatics system outlined above. We describe this architecture, focusing on the addition of web- and moible-device interfaces. We outline important parts of its object-oriented design, some example user interfaces, our implementation experiences with Java technologies to realise these designs, and include results of some empirical evaluations (usability and performance) conducted using this prototype.

2 Requirements

In most existing dealings with health professionals a patient's medical details are stored in several places, often include only information pertaining to the particular health providers storing the information, is often inconsistent, and is difficult to share among providers and insurers [5]. Paper-based prescriptions and treatment records are notorious for introducing errors due to poor handwriting or mis-interpretation [6]. Patients seldom have direct access to their own medical histories [7].

Figure 1 shows an alternative model, where each stakeholder accesses medical information via interfaces to an integrated "health information brokering" system. GPs and hospital staff maintain patient details, diagnoses and treatments. Pharmacists receive prescription information and record provided drugs. Hospital observation staff and community nurses have access to patient treatment information and can record observation and treatment results via mobile devices. Observational devices, like ECGs, could even be interfaced to the system, providing direct data capture. Information can be exchanged with other health providers and insurers, and patients have web-based access to their medical histories. Some key non-functional requirements include:

- Appropriate user interfaces e.g. desktop (GPs, hospital admissions staff); PDA and mobile pagers/phones (nurses, observation & on-call staff); Web access for patients.
- Security and data integrity is crucial, including securing data transmission across local, wide and wireless networks and appropriate authentication and access rights.
- Performance e.g. large number of admissions, number of devices

- Robustness e.g. if a network or service goes down, data needs to be cached off-line.
- Integration e.g. the use of standard interfaces and exchange formats
- Extensibility e.g. the system supports the seamless addition of new interfaces etc.

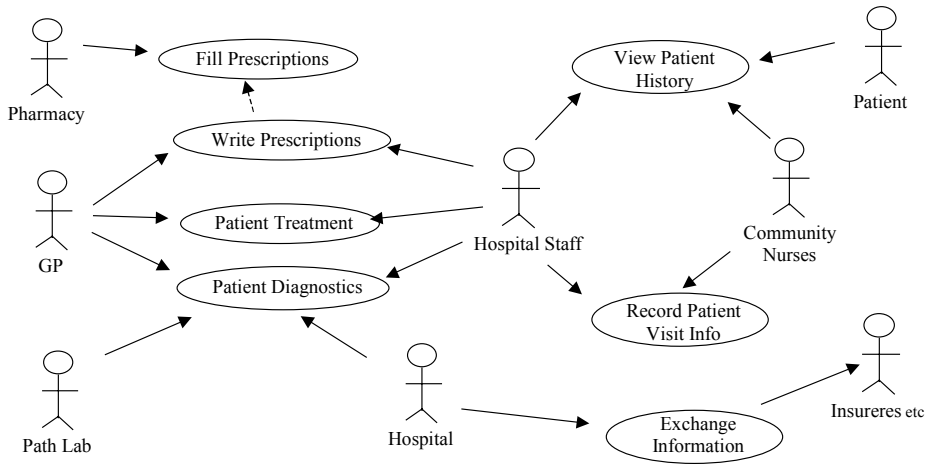

Figure 1. Key use cases in our integrated health informatics system.

3 Architecture

From detailed user requirements and an object-oriented analysis specification we developed an architecture for this integrated health informatics system that could satisfy the diverse non-functional and functional constraints. Some key criteria include:

- Need for centralised information management for large amounts of data. This lead to a multi-tier architecture with centralised business process and database servers.
- Need to support a wide range of diverse interface devices. This led to using several technologies for server-side data access and presentation, including CORBA and XML (GPs, pharmacy and external systems), Java Server Pages (browsers and WML wireless devices), and Java Jini (PDAs with applets, diagnostic devices like ECGs).
- Need for robust, high performance facilities, via multiple servers providing client interfaces, server-side presentation, and server-side business and data processing.

Our conceptual architecture is illustrated in Figure 2. The central part of this system is information management and processing, provided by a set of application servers and database servers, running on multiple hosts. We use Enterprise Java Beans (EJBs) to provide the business logic and information processing support, deployed on multiple EJB servers. EJBs provide a scalable set of abstractions for building enterprise system "middle-tier" components [8, 2]. A variety of clients connect to the EJB components to access and update data. Some are desktop staff applications running on PCs and laptops within a hospital. These do not directly access the databases but instead go via the

application server tier to provide a more scalable, robust and secure environment. Web-based clients, such as some staff applications, patient browsers and mobile WML-enabled devices access the application servers via a set of Java Server Pages and servlets. JSPs provide HTML for web-browser clients accessing the system via a LAN or WAN, or WML (Wireless Markup Language) for mobile clients. Firewalls secure access for HTML and WML clients outside the hospital LAN.

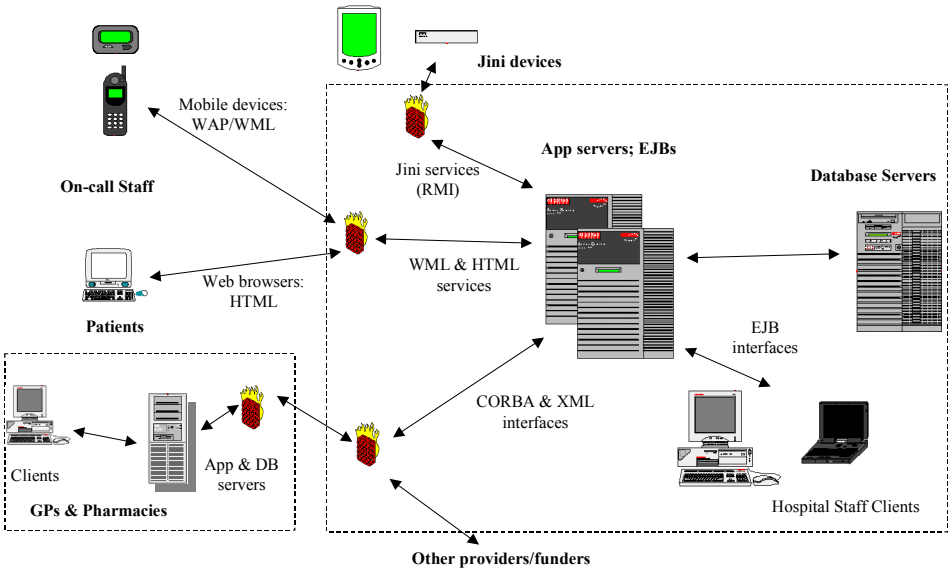

Figure 2. An overview of the architecture of our system.

GPs and Pharmacists run their own local health informatics systems. We provide a set of CORBA interfaces through which these external provider systems can incrementally access and update patient information. CORBA object wrappers or interceptors can be used to provide a set of security measures, such as data encryption. Inter-person messaging, patient information annotation and notifications are supported between hospital staff, GPs, pharmacists and community health nurses. XML is used for bulk data transfer, such as with a Path Lab or Health Insurer or funding agency.

Some mobile PDAs might run applets rather than WML browsers, as simpler devices like pagers and mobile phones are likely to do. This is to support off-line work where the applet can cache both server-sourced data e.g. patient information, and user-entered data, such as treatment details, which can later be sent to the server . We interface these to the system using Java Jini, which supports device inter-connections dropping and being re-established. We also allow diagnostic devices to be interfaced via Jini for data capture.

4 System Design

Our application server components use the Enterprise Java Beans (EJB) architecture and design guidelines. EJBs are software components encapsulating enterprise business logic and basic data management facilities [8]. "Session Beans" encapsulate the logic processing of maintaining patient and staff information, adding and updating treatments, adding and updating diagnostics, and providing queries over patient histories. "Entity Beans" provide transparent data management facilities using a set of database servers. These Beans (components) run in a collection of EJB containers which provide security, transaction and thread management support. We deploy several EJB servers on different application server hosts with replicated Beans to provide a scalable, robust set of business logic and data management components. Figure 3 illustrates these components.

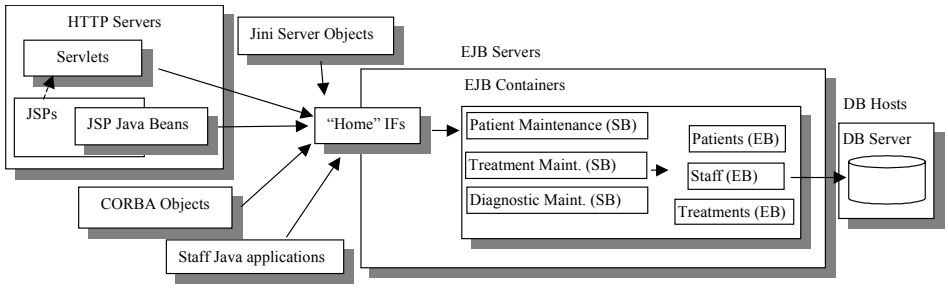

Figure 3. EJB-based Application Server components.

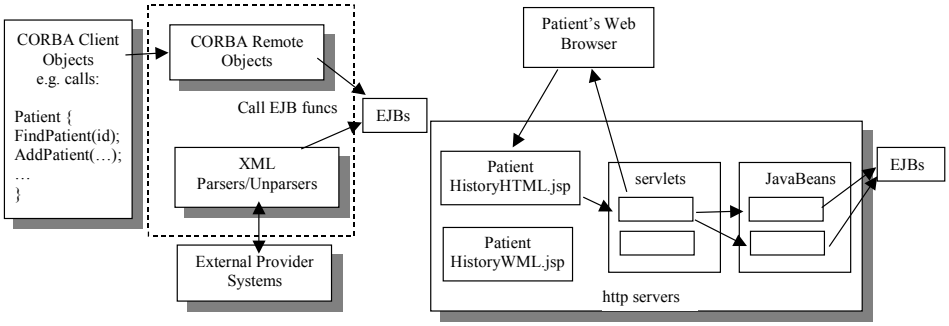

Figure 4. (a) GP, Pharmacy and other provider access; (b) HTML/WML client access.

GP, Pharmacist and other provider systems access the centralised health information brokering system via CORBA or XML interfaces. CORBA provides a distributed object model which includes a variety of useful services, such as security, event subscription and object lifecycle management. We use CORBA for systems which need incremental data exchange. For example, GPs obtaining or sending information, pharmacists obtaining prescription details or supplying updated details etc when patients visit the GP

or Pharmacist. We use XML for batch data exchange, for example when sending an insurance claim details to a health insurer's system. We used the HL7 XML encoding for data exchange [9]. Figure 4 (a) shows our CORBA and XML interface components.

Patients (and some hospital staff) may access information using a web-based interface. We use a set of JSP pages to provide these interfaces. The JSPs are run on application server hosts running servlet-enabled http servers. The JSPs themselves provide presentation logic, and make use of JavaBeans to provide data access/update functionality. These JavaBeans run in the http server processes and access the EJB components running on other application server hosts. Wireless devices that use the WML markup language (e.g. mobile phones, pagers and some PDAs) access other JSPs which provide WML-encoded data. Figure 4 (b) illustrates these components.

We chose to use Jini technology to support the interfacing of diagnostic capture devices and applet-enabled PDAs to our intergrated health information system. Jini provides an architecture for connecting a wide range of server devices to computer systems and advertising server characteristics for clients to locate and use. We provided some Jini servers that provided patient detail and diagnostic lookup and update facilities, and implemented prototype Jini clients for PDA and diagnostic devices. The diagnostic devices and PDAs discover different services depending on locality and needs e.g. wireless PDA wanting data look-up only can use different services to a PDA in cradle with data update requirements. Figure 5 illustrates the basic design of our Jini-based interfaces. A Jini "proxy" provides a TCP/IP server socket for the PDA, as current PDA Java Virtual Machines don't typically support the Jini API directly.

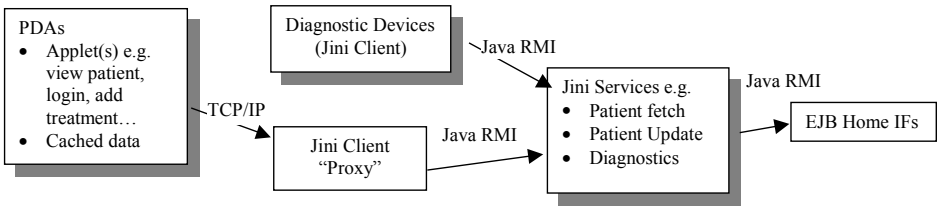

Figure 5. PDA and Device components.

5 Implementation and Evaluation

We implemented a prototype of this integrated health information system using a variety of Java technologies and tools. We used the Java 2 Enterprise Edition (J2EE) software development kit to build our Enterprise Java Beans to encapsulate a range of business logic. We deployed these on a collection of EJB servers running on Windows NT machines. We used the Interbase 5.5 database server to provide Bean-managed persistency for the EJBs. We implemented some basic data management Java applications for hospital staff that access the EJBs to add patients, treatment details etc.

Hosptial staff can also add notes against patient records or treatments, and exchange messages with GPs, Pharmacists and Patients via a desktop application interface.

We implemented a set of CORBA IDLs and remote objects and servers to provide access to integrated data for GP and Pharmacy enterprise systems. A set of GP applications hold their own data locally, but use the CORBA interfaces to lookup and update centralised data. Our CORBA clients use Delphi, to validate the ability of CORBA to support cross-language access. GPs system functionality includes maintaining patient records, maintaining treatment and diagnostic records, and messaging Pharmacists, Patients and hospital staff. GPs also receive message notifications when their patient prescriptions are filled or new treatment/diagnostic information has been added. Figure 6 (a) shows an example of one of the GP user interfaces. A "health insurer" was prototyped using a Java application to which XML-encoded patient treatment information is sent. The insurer application parses patient and treatment information encoded using the HL7 health EDI messaging standard, represented as an XML document.

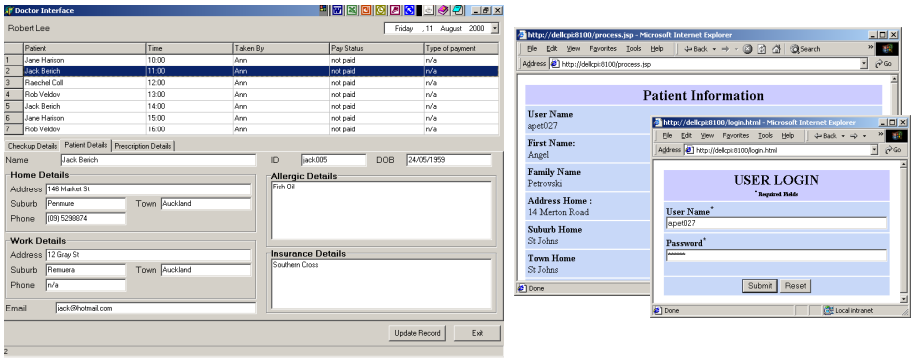

Figure 6. Examples of (a) GP desktop and (b) patient Web user interfaces.

Patient access to their information is provided by a set of JSPs. Patients can view their basic details, treatment and diagnostic histories and parts of their GP and hospital staff notes. They can also send and receive messages to GPs and hospital staff, and receive treatment and diagnostic update notifications. We used the Tomcat JSP reference server implementation to provide a simple JSP server for our prototype. Figure 6 (b) shows an example of patient details displayed by one of the JSPs.

Some wireless devices use our WML-producing JSPs to obtain or update information. This is suitable for light- weight messaging, notification and basic patient information browsing, as might be used to notify on-call staff and provide them with limited information content. Figure 7 (a) shows examples of some of our WML interfaces. We provided observational staff and community nurses a more flexible, robust mobile PDA-based interface to find patients, view patient treatment needs, record observations (simple diagnostics), and upload/download data. We used the Palm III with its mini-Java Virtual Machine to implement PDA applets, with two examples shown in Figure 7 (b).

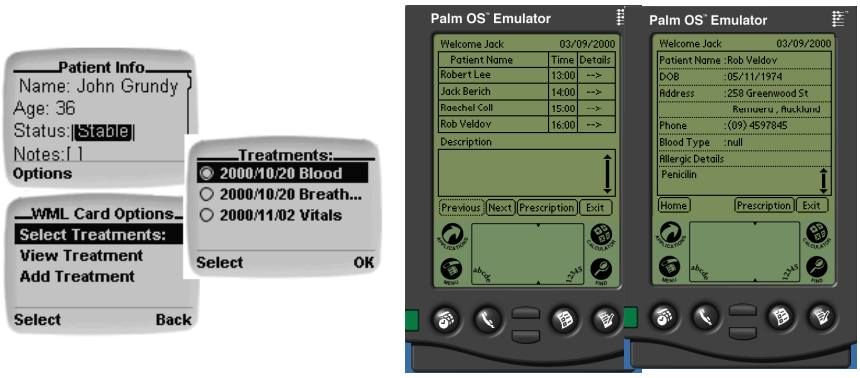

Figure 7. Examples of (a) staff WML and (b) staff PDA applet user interfaces.

We carried out two evaluations of our prototype. A usability study was conducted with a range of users, looking at usability of the GP desktop, patient web and Palm PDA interfaces. A dozen staff and students from the University of Auckland participated in the study. A set of individual tasks were assigned to groups of participants using each interface in turn for similar tasks. We found "users" generally liked all of the interfaces and found them relatively easy to use. Most supported the basic tasks users were asked to perform. Some problems occurred with the PDA interface, which required a considerable number of screens to display information. Performance tests used database and EJB servers running on separate Windows NT hosts, EJB, CORBA and HTTP clients simulating up to 100 multiple, concurrent requests and a PDA client simulating up to 15 concurrent requests. Performance was found to be adequate in terms of response time for the EJB clients, degraded significantly for the CORBA clients and became relatively slow (several seconds) for PDA clients when maximum levels of simulated clients were reached.

6 Discussion

The trend towards complex, multi-tier data processing and management systems has occurred in many domains [10, 11, 2, 12]. Typical solutions utilize database, business logic, presentation and client tiers. These design and implementation approaches typically provide scalable solutions for large numbers of clients and complex server-side processing [1, 2]. We chose EJBs as a server-side organization technology as it offered useful abstractions, performance, scalability, and compatible presentation-tier technologies in servlets and JSPs and connectivity facilities in CORBA and Jini [2].

The need to integrate a wide variety of enterprise systems has become important in many application domains [1, 11]. One common approach in the health sector is Electronic Data Interchange (EDI), using various message formats [9, 13]. While many EDI standards exist, many are now moving to XML-based solutions, which offer a more

easily parsed and processed representation [11]. Other integration approaches include database integration, remote objects and Enterprise Application Integration solutions [1, 2]. We used CORBA remote objects to provide incremental data exchange because of its wide acceptance for language and platform independence. We used an XML-encoded health EDI standard, HL7, due to its wide usage in the health sector.

Web-based interfaces typically use Java applets, Active X controls or HTML browsers [13]. Applets and Active X controls provide a wider range of interaction mechanisms, more dynamic content and more flexible middleware connectivity than HTML-based interfaces. However, for our Patient interfaces and simple staff data maintenance screens, HTML was sufficient. We used Java JSP scripts as these were easy to build and very flexible, and more scalable than CGI-based solutions [15].

Mobile devices are becoming more widely used for information access [4]. We chose to use Java applets, accessing the enterprise system via Jini services, to provide a more flexible and powerful interface than WML-based interfaces. Applets also allow caching of data on our PDAs, very useful to improve performance and make them less vulnerable during intermittent network disconnection. Jini also provides a useful access technology for other periodically disconnected devices. We used WML-based interfaces for mobile devices (phones, pagers) which require light weight data access facilities.

The various technologies we used to realize our prototype integrated health informatics system generally worked well. The EJB-based application server facilities performed reasonably well and provided reasonably good component abstractions. It is unclear whether large scale EJB-based servers will provide adequate enterprise systems performance, and we found problems with the current J2EE SDK tools for EJBs. We found CORBA and XML provide good enterprise system integration platforms for our needs. Unfortunately current XML parsing and unparsing tools and APIs are quite rudimentary. We found Java Server Pages (JSPs) to provide good support for building both HTML and WML interfaces. JSPs integrate well with Enterprise Java Beans and we found they appear to give relatively good performance. We had numerous problems in realizing our PDA-hosted applets. The PDA "K-virtual machine" (a mini-Java Virtual Machine) has numerous bugs and inconsistencies with the Java Virtual Machines that desktop and web browsers use. Quite different communication and user interface APIs need to be used, both of which are unstable. The K-virtual machine doesn't support Jini APIs, so we had to connect our PDA applets to a "Jini client proxy", via a socket connection, which then connected to Jini services.

7 Summary

Our integrated health information system prototype includes Enterprise Java Beans to provide scalable, centralized data management. Other providers exchange information using CORBA objects and XML. Patients and staff use Web-based interfaces, realized with Java Server Pages. On-call staff are notified and may browse brief patient details via WML-based interfaces. Observation staff and community nurses access the system via intermittently connected PDAs running Java Applets, using Jini services to access

information. Performance and usability evaluations demonstrate the architecture is very promising. Future work includes the development of general-purpose groupware components and adaptable user interface techniques to ease the development of the same interface for diverse user and display device needs. We are exploring the use of XML message transformation support using XSLT transformation scripts, and ways to use peer-to-peer communications and to improve performance, robustness and efficiency.

Acknowledgements

Support for this research from the University of Auckland Research Committee and the New Economy Research Fund is gratefully acknowledged.

References

1. Aleksy M, Schader M, Tapper C.: Interoperability and interchangeability of middleware components in a three-tier CORBA-environment-state of the art. In 3rd International Enterprise Distributed Object Computing Conference. IEEE CS Press, 1999, pp.204-213.
2. Vogal, A.: CORBA and Enterprise Java Beans-based Electronic Commerce, International Workshop on Component-based Electronic Commerce, Fisher Center for Management & Information Technology, UC Berkeley, 1998.
3. Varshney, U. Vetter, R.J. and Kalakota, R. Mobile Commerce: A New Frontier, Computer 33 (10), October 2000, IEEE CS Press.
4. Amoroso, D.L. and Brancheau, J.: Moving the Organization to Convergent Technologies: e-Business and Wireless, In Proceedings of the 34th Annual Hawaii International Conference on System Sciences, Maui, Hawaii, Jan 3-6 2001, IEEE CS Press.
5. Garshnek, V. and Burkle, F.M.: Telecommunications and Informatics Contributions to the Future of Public Health: A Forecast, In Pacific Medical Technology Symposium, Honolulu, Hawaii, 17-20 August 1998, IEEE CS Press.
6. Del Fiol, G., Nohama, P., and Rocha, B.H.S.C.: Modeling a Decision Support System to Prevent Adverse Drug Events, In 13th IEEE Symposium on Computer-Based Medical Systems, June 2000, Houston, Texas, IEEE CS Press.
7. D'Alessandro, M.P. et al: The Iowa Health Book, In Advanced Digital Libraries 1996, Washington DC, May 13-15, 1996, IEEE CS Press.
8. Monson-Haefel, R.: Enterprise JavaBeans, O'Reilly, 1999.
9. Health Level 7, http://www.hl7.org/.
10. Lewandowski, S. M.: Frameworks for component-based client/server computing, ACM Computing Surveys 30 (1), March 1998, ACM Press.
11. Moshfeghi, M. and de Greef, B.: XML in a Multi-Tier Java/CORBA Architecture, In 8th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, June 1999, Palo Alto, CA, IEEE CS Press.
12. Smailagic, A. et al, MoCCA: A Mobile Communication and Computing Architecture, In 3rd International Symposium on Wearable Computers, San Francisco, 1999, IEEE CS Press.
13. Huemer, C., Defining Electronic Data Interchange Transactions with UML, In 34th Annual Hawaii International Conference on System Sciences, Maui, HI, Jan 2001, IEEE CS Press.
14. Evans, E. and Rogers, D.: Using Java Applets and CORBA for multi-user distributed applications, Internet Computing 1(3), 1997, IEEE CS Press.
15. Fields, D., Kolb, M., Web Development with Java Server Pages, Manning, May 2000.