Cartella Clinica Elettronica su Piattaforma Java InfoBus

Paolo Anedda, Gavin Brelstaff, Sascha Moehrs, Massimiliano Tuveri, Gianluigi Zanetti
CRS4, CP94, Uta (Ca), 09012 Sardegna Italia
{ciccio,gjb,sascha,mtuveri,zag}@crs4.it

Abstract

Descriviamo un sistema di cartella clinica elettronica che utilizza tecnologie internet di seconda generazione, tra le quali XML, XSL, CSS, JavaBeans e InfoBus, per distribuire in maniera sicura cartelle interattive di dati clinici nel web browser dei medici. Il sistema mette a disposizione strumenti grafici per la visualizzazione della cartella di un paziente, per la visione di immagini e per la redazione di referti clinici. Di seguito viene presentato un insieme di applet Java e client-side Javascript. La comunicazione tra le applet avviene attraverso l'InfoBus. Il sistema e` compatibile con gli attuali browser maggiormente utilizzati. Questo prototipo sperimentale costituisce la base per un sistema che verra` sviluppato per un ospedale di Cagliari.

1. Introduzione

Come parte della modernizzazione dei loro servizi di sanita‘ pubblica, molte nazioni europee stanno pianificando l'introduzione di sistemi elettronici di cartella clinica (EPR). Nel Regno Unito per esempio, un recente report [1] così descrive gli obbiettivi del servizio sanitario nazionale:

2.7 The arguments for a move towards an electronic record are compelling. Such records are more likely to be legible, accurate, safe, secure, and available when required, and they can be more readily and rapidly retrieved and communicated. They better integrate the latest information about a patient's care, for example from different "departmental" clinical systems in a hospital. In addition, they can be more readily analyzed for audit, research and quality assurance purposes.

Tra i sei benefici chiave elencati vi sono: l'integrazione di cure tra i medici di base (MDB) e gli ospedali, e l'aumentata efficienza dovuta alla riduzione dei tempi per la raccolta dei dati esistenti.

La qualita` della comunicazione clinica tra gli ospedali e i MDB e` stata per lungo tempo un elemento controverso ad entrambe le estremita` del processo. Questi problemi si presentano come una minaccia per la qualita` e la sicurezza della cura del paziente. I MDB rivendicano coerentemente la qualita` e tempestivita` dei risultati degli esami e le informazioni concernenti i pazienti. Da parte loro, gli ospedali sono ugualmente critici sull'inadeguatezza e incompletezza delle informazioni di supporto ai referti.

E` chiaramente insufficiente costruire una Intranet nazionale per il servizio sanitario, come la britannica NHSnet, perche` gli addetti alle cure possano comunicare in modo efficace [2] e sicuro [3]; recentemente un corrispondente del British Medical Journal [4] ha raccomandato che i MDB non si connettessero alla NHSnet sino alla sua ristrutturazione.

Il principale obbiettivo di questo progetto e` dimostrare come i medici possano collaborare efficientemente e in modo sicuro sulle EPR dei propri pazienti anche se seduti di fronte a scrivanie geograficamente distanti.

La tolleranza alle disconnessioni dalla rete e` un requisito fondamentale per un sistema che debba essere usato dai MDB. In Italia, per esempio, i MDB si collegano alla rete semplicemente chiamando un convenzionale ISP; in questo modo essi si concentrano sulla minimizzazione dei costi di connessione telefonica.

Il sistema proposto di seguito, e` stato progettato pragmaticamente affinche` i MDB si possano sconnettere dalla rete dopo aver scaricato la EPR sul proprio desktop.

Una cartella clinica elettronica puo` essere mappata naturalmente in quello che oggi viene definito un oggetto multimediale [5], dal momento che combina un insieme di testo, grafici e immagini in modo piu` intuitivo e in una forma maggiormente comprensibile.

Un fattore chiave che ostacola lo sviluppo di sistemi di EPR multimediali e` la complessita` dei sistemi software. Le tecnologie legate ad internet hanno iniziato a ridurre parte di questa complessita` consentendo [1] la visualizzazione su diverse piattaforme: HTML, CSS; e [3] la gestione della sicurezza delle reti: crittografia, firma digitale e controllo degli accessi sicuri. Tuttavia la complessita` cresce quando il software necessita di scambiare dati

La gestione ergonomica degli errori di connessione e` un compito complesso che, se e` possibile, e` meglio evitare. La nostra intenzione e` di evitarla in fase di

progettazione. Le transazioni di documenti rappresentanti la cartella di un paziente sono asincrone. Dopo essere stato inviato come testo formattato in XML [6] il documento viene caricato in un Document Object Model (DOM) [7], dove rimane disponibile al MDB. La complessita‘ del software e‘ stata governata realizzando dei piccoli componenti indipendenti che svolgono il proprio compito utilizzando i dati che risiedono nel client-side DOM. In questo modo una transazione sulla rete avviene solo per il consulto di esami di immagini mediche o per inviare il lavoro effettuato. Tutte le altre transazioni avvengono all'interno del desktop tra i componenti e il DOM.

Abbiamo implementato e integrato i seguenti componenti:

Un EPR Browser della cartella di un paziente che semplicemente trasforma se stessa in una form per l'inserimento dei dati. Complementare a questo browser e` l'XSL Processor che genera HTML dall'XML seguendo un mecanismo standard di XSL/T template matching.

Un Image Viewer che consente al medico di interagire con sequenze di immagini di esami clinici. Ad esso si integra un xy-Graph bean che abbiamo configurato per visualizzare i livelli di grigio dei profili delle immagini.

Un Clinical Reporter bean che consente ai medici di preparare dei referti clinici basati sulle immagini che stanno osservando.

Diversi bean che forniscono una struttura uniforme ai precedenti e gestiscono l'InfoBus: l' IO Manager e il DOM Manager.

Tutti questi elementi verranno descritti in dettaglio nelle sezioni sucessive.

2. Infrastruttura

Il progetto WMED [8] ha dimostrato come le tecnologie internet di prima generazione possano fornire un'infrastruttura base per la distribuzione sicura di documenti clinici nei web browser. Secure Socket Layer (SSL) e` un protocollo standard [9] attraverso il quale, molti degli odierni browser sono in grado di abilitare la crittografia dei dati scambiati tra un web browser e un web server sicuro. In pratica, la crittografia, e cosi` la sicurezza dei dati confidenziali dei pazienti, e` assicurata gratuitamente a coloro che installano e configurano un server web attinente alle specifiche SSL, come Apache-SSL [10]. WMED ha inoltre rivelato come possa essere scomodo sincronizzare il contenuto di un database con quello delle applicazioni dinamiche che risiedono sul client. Sebbene esista una pletora di metodi per la mediazione tra i web server e gli esistenti database relazionali (RDBS): CGI, JDBC, Apache-mod-perl, ASP, LiveWire, JSP, nessuno e` particolarmente facile da programmare o manutenere. L'avvento di XML ha introdotto un formato dati standard, gerarchicamente strutturato e orientato a internet, che si puo` integrare piu` naturalmente con un RDBS come Oracle 8i. L'inclusione del DOM negli emergenti browser significa che effettivamente un sottoinsieme strutturato del contenuto di un RDBS puo` essere trasportato facilmente sul desktop del client in modo sicuro. Il nostro approccio anticipa i vantaggi di un simile approccio fornendo al browser del medico tre componenti nella forma di applet Java, in grado di processare i dati del paziente all'interno di DOM. Questi componenti sono i bean IOManager, DOMManager e XSLProcessor. L'IOManager e` il responsabile della lettura dei dati XML da un URL. Qualora l'URL non fosse lo stesso d'origine dell'applet, generalmente si richiede qualche configurazione extra del browser per consentire il superamento delle limitazioni imposte dalla sandbox Java (Abbiamo constatato che la soluzione piu` flessibile e` scaricare le applet da un server SSL in un browser Netscape). Il DOMManager converte i dati XML in DOM, mentre l'XSLProcessor consente di trasformarne il contenuto in pagine HTML. La pagina risultante dalla formattazione e` utilizzata in seguito per lanciare le altre applet.

Queste applet implementano una specifica interfaccia JavaBean che ne consente la comunicazione attraverso l'InfoBus [11], [12]. Abbiamo sviluppato un bean dedicato, il BusMonitor, che consente di monitorare graficamente lo stato dell'InfoBus, mostrando i bean che popolano il bus e il flusso delle informazioni. Il BusMonitor, come mostrato nella figura 1, indica la presenza di tutti e tre i bean necessari come infrastruttura; la loro instanziazione avviene semplicemente usando il tag Applet. In realta`, abbiamo scoperto che per evitare la possibilita` della presenza di diverse istanze dell'InfoBus, e` necessario precaricare l'InfoBus Java Archive nel classpath della VM del browser. In caso contrario il BusMonitor non potrebbe mostrare la presenza di tutti i bean e questi non potrebbero comunicare tra di loro. Una volta garantita la presenza di tutti e tre i bean e dopo aver caricato l'XML relativo alla cartella del paziente nel DOM, l'utente puo` selezionare uno stylesheet XSL. La figura 1 illustra la selezione dell'English Style. L'XSLProcessor richiede e riceve i dati dal DOM ai quali applica successivamente uno XSL stylesheet's template per generare l'HTML per l'EPR Browser.

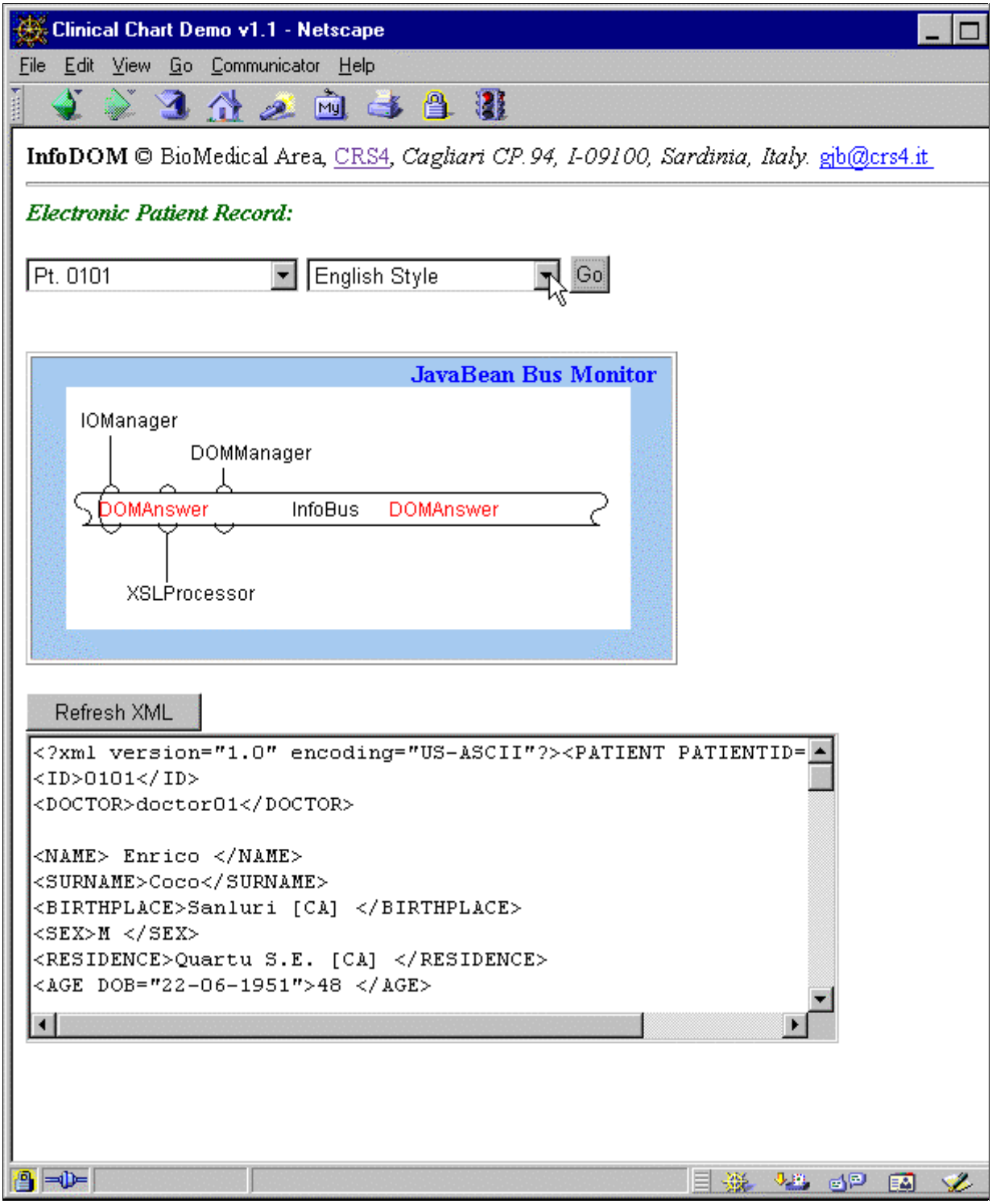

Figura 1. L'accesso alla cartella clinica del paziente avviene attraverso la richiesta del file XML associato al codice del paziente. In questo caso e` stata richiesta la cartella del paziente 0101. Il JavaBean Bus Monitor indica la presenza sull'infobus di tutte e tre le Applet necessarie all'elaborazione dei dati della cartella. Una volta che il contenuto della cartella e` disponibile, l'XSLProcessor applica uno stylesheet XSL per la formattazione dei dati. Il monitor mostra la risposta del DOMManager alla richiesta di dati.

3. EPR Browser

L'EPR Browser e` semplicemente una pop-up window creata, come sivede sullo sfondo della figura 2, dalla stringa del contenuto della pagina HTML fornita dall'XSLProcessor , usando client-side Javascript (CCJS). I colori e lo stile di presentazione di questa pagina sono specificati usando uno stylesheet CSS [13]. L'aspetto della pagina puo` essere commutato in Edit mode per consentire la modifica dei campi di testo forniti (es. Residence). CSJS e` utilizzato per filtrare i dati in ingresso ed evitare l'inserimento di valori errati o fuori scala. Abbiamo riscontrato una varieta` di bug nei browser che impediscono la corretta visualizzazione di questa pop-up. Tutti i cambiamenti avvenuti nella cartella sono salvati nel DOM contenuto nella pagina di partenza originale. In entrambi i modi la pagina e` arricchita di link che in seguito all'attivazione consentono l'apertura di nuove pop-up window contenenti l'ImageViewer o il Reporter, parametrizzati appropriatamente per il particolare paziente e esame in questione.

4. Image Viewer

L'Image Viewer e` un'applet dedicata che legge sequenze di immagini da uno specifico URL della rete e le mostra in una sequenza animata controllabile dall'utente. L'animazione puo` essere fermata in qualsiasi momento per analizzare in dettaglio una particolare immagine. Per esempio si puo` utilizzare una finestra che ingrandisce 2,3 o 4 volte i dettagli, spostandola con il cursore del mouse. Si possono eseguire delle annotazioni sulle immagini di interesse, utilizzando elementi grafici come e` illustrato nella figura 2. Questi elementi grafici possono essere letti dal DOM o registrati in esso andando a costituire una parte della cartella. Ci siamo preoccupati di lasciare i dati delle immagini come una risorsa esclusivamente leggibile. Questa scelta e` sensata in quanto la sincronizzazione dei cambiamenti di grandi sequenze di immagini tra i client della rete potrebbe portare a degli inevitabili ritardi e comporterebbe una complicata integrazione con sistemi server di legacy PACS. Infatti, abbiamo adottato lo standard Scalable Vector Graphics (SVG) [14] per gli elementi grafici in modo da consentire di poter salvare facilmente le annotazioni e trasportarle come elementi XML. Il nostro prototipo attualmente e` configurato per la lettura di immagini GIF da un web directory. In pratica, queste immagini potrebbero essere precaricate a differenti risoluzioni in un server PACS per consentire l'accesso alla cartella di un particolare paziente. Abbiamo svolto dei test preliminari [15], [16] utilizzando una PACS conforme a DICOM 3 [17] in conformita` al lavoro svolto in altre parti del mondo [18], [19].
Un altro bean che mostra in un grafico i dati relativi ai livelli di grigio puo` essere attivato cliccando su un'immagine: l'xy-Graph bean. Ancora una volta CSJS e` utilizzato per generare un nuovo bean ad ogni click del mouse.

5. Reporter Bean

Cliccando sul link "[View Report]" nell'EPR Browser si esegue il Report bean appropriatamente configurato usando la stessa tecnica CSJS descritta precedentemente. La figura 3 mostra la configurazione per la refertazione delle arterie coronarie. Questa applicazione [20] fu mostrata per la prima volta nel sistema WMED dove non era integrata nell'EPR. Ora i dati sono scritti e letti dal DOM e trasferiti come XML. La struttura dell'immagine e` letta come SVG il contenuto dei suoi pop-up menu e specificato in XML. Per esempio l'immagine mostra che l'arteria IVA-II e` abilitata a lanciare tre tipi differenti di referti: Arteriosclerosis, Bypass o Myocardial Bridge. Selezionando un elemento del menu, si esegue un reportlet window – Arteriosclerosis, com'e` illustrato nella figura 3. Ogni reportlet puo` essere considerata come una form HTML graficamente ampliata da elementi dello standard Java AWT, come drop down selection menu. Le estensioni includono image-menus e slide bar. I dati della form sono acquisiti da un appropriata sezione del DOM dell'EPR e salvati in essa. Inoltre viene fornito un primitivo sistema di supporto alle decisioni. Per esempio il pie-chart nella figura e` stato inserito per indicare la probabilita` relativa di classificare la lesione come tipo A, B o C sulla base delle scelte effettuate.
Tutte le impostazioni grafiche come image-menu o immagini SVG sono organizzate in maniera gerarchica seguendo uno stile web-based cosi` che possano essere riutilizzate da altre applicazioni basate su URL.

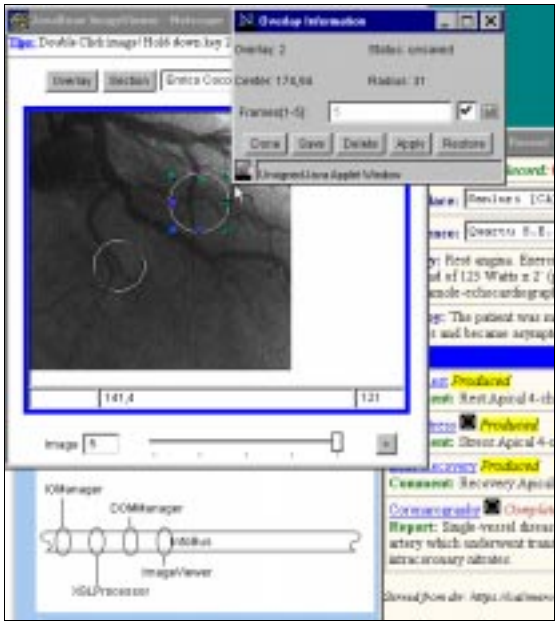

Figura 2. Come è possibile vedere sullo sfondo dell'immagine, parti del file XML sono visualizzate come HTML in una finestra del browser. Questa operazione viene effettuata dall'XSLprocessor applicando un template XSL/T specificato dallo stylesheet XSL. Cliccando su di un link alle immagini di un esame, viene lanciato il JavaBean Image Viewer che e` capace di mostrare sequenze animate; e` possibile inoltre creare e modificare delle annotazioni grafiche che possono poi essere salvate in formato SVG.

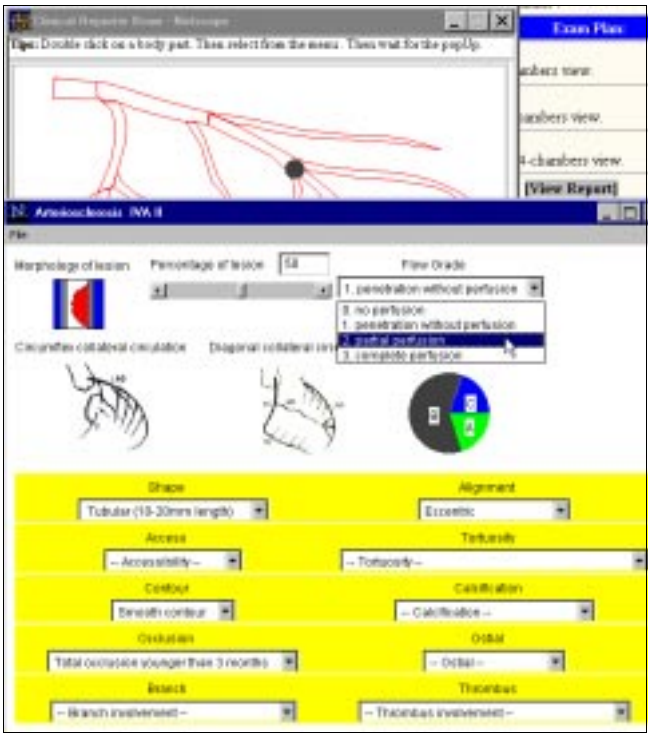

Figura 3. Il *Reportlet* per l'arteriosclerosi consente al medico di specificare graficamente gli appropriati segni clinici appopriati al paziente. L'intefaccia e` costituita da slider, pull-down text e pop-up menu. Il pie-chart e` un sistema per il supporto alla decisione che suggerisce il tipo di lesione basandosi sulle informazioni derivanti dai dati.

6. Valutazione e discussione

Il sistema sperimentale sopra illustrato e‘ stato adottato in un ospedale in Sardegna a Cagliari. Come tale esso costituira‘ un sistema di prova [21] al quale i medici parteciperanno sia come utenti, fornendo dei riscontri, che come collaboratori alla progettazione, fornendo informazioni riguardanti la propria specialita‘. Inizialmente ci si focalizzera‘ sulla comunicazione tra radiologi e neurologi. E‘ durante questa fase che la flessibilita‘ inerente il sistema sara‘ d’aiuto consentendo una rapida prototipizzazione e integrazione di risorse di rete addizionali, senza un significativo aumento della complessita` del sistema. Questo dovrebbe essere un particolare vantaggio rispetto a simili sistemi sviluppati in una maniera cosi` flessibile o a sistemi legacy che aderiscono agli standard EDI o HL7.

7. Ringraziamenti

Questo progetto e` parzialmente supportato dall’equipaggiamento ottenuto attraverso il programma filantropico della Hewlett Packard (http://webcenter.hp.com/grants/). Un supporto addizionale e` stato fornito dalla Regione Sardegna. Ringraziamo Andrea Giachetti, Alessandro Pomata e Kateryna Radchencko per la loro assistenza

8. Riferimenti

[1] NHS Executive, Information for Health, An Information Strategy for the Modern NHS 1998–2005 Published by the NHS Executive, UK Crown Copyright, web document http://www.imt4nhs.exec.nhs.uk/strategy/full/contents.htm 23 September 1998.

[2] Keen, J. “Rethinking the NHS networking”, British Medical Journal, No 7140, Vol. 316, 25 April, 1291-1293, 1998.

[3] Anderson, R.J. “Security in clinical information systems”, London: BMA, 1996.

[4]Kelly, G, S-B,. “MDBs should not connect to NHSnet until it is restructured”, British Medical Journal, No 7140, Vol. 318, p533, 20 Feb, 1999.

[5] Sacoor, D. “Integration of images for multimodal medical applications” Report of Workshop organized by the European Commission DG XIII, Brussels, March 1997.

[6] Bray T, et al, “Extensible Markup Language (XML) 1.0”, W3C Recommendation, http://www.w3.org/TR/1998/REC—xml-19980210, 1998.

[7] Wood L et al, “Document Object Model (DOM) Level 1 Specification”, version 1.0, W3C Recommendation, http://www.w3.org/TR/1998/REC-DOC-Level-1-19981001, 1998.

[8] Brelstaff, G., Loddo, S. “WMED – An experimental electronic patient record system based on the WWW” CRS4 Internal Report: http://www.crs4.it/~gjb/BMA, 1996.

[9] Hirsch, F.J. “Introducing SSL and Certificates using SSLeay”, World Wide Web Journal, Summer 1997.

[10] McKay, F. “Fortify for Netscape”, public-domain software, http://www.fortify.net since 1998.

[11] Sun Microsystems, “The JavaBeans Component Architecture”, http://java.sun.com, 1998.

[12] Hoque, R, “Connecting JavaBean with InfoBus”, Wiley: New York, USA, 1999.

[13] Bos B. et al, “Cascading Style Sheets level 2, CSS Specification”, W3C Recommendation, http://www.w3.org/TR/1998/REC-CSS2-19980512, 1998.

[14] Ferraiolo, J. et al, “Scalable Vector Graphics (SVG) 1.0 Specification”, W3C Working Draft, http://www.w3.org/1998/08/WD-SVG-19990812, 1999.

[15] Giachetti, A. Donizelli M, & Scheinine A.L. “DICOM image handling for medical analysis and the ViVa Project”, Proceedings of EuroPACS’ 98, Barcelona.

[16] Loddo, S. Brelstaff, G. Zanetti, G. “A distributed heterogeneous image server”, Proceedings of EuroPACS, Tipografia Editrice Pisana, Pisa, C.Bartolzzi & D.Caramella (Eds), Italy, 199-1202, 1997.

[17] Moore, S.M. Hoffman, S.A. Beecher, D.E. “DICOM Shareware: A Public Implementation of the DICOM standard”, Medical Imaging, 1994.

[18] Ligier, Y. Ratib, O. Logean, M. Girard, C. “OSIRIS, a medical image manipulation system”, M.D. Computing Journal, Vol. 11, No 4, 212-218. 1994.

[19] Bayo, J.F. Gomez, R. Catusus, X. Barbero, O. Feron, M. Sentis, M. Bellon,E. “DICOM Java viewer for a using WWW Internet Technology and DICOM 3.0 standard”, Proceedings of EuroPACS, Tipografia Editrice Pisana, Pisa, C.Bartolzzi & D.Caramella (Eds), Italy, 133-136, 1997.

[20] Loi, B. Tuveri, M. Pescosolido, M. Paddeu, G. Pili, P. Zanetti, G. "MEDREP: a WWW System for compilation and integration of medical data". In INCIS'96 3rd international workshop on integrating cardiology information systems, , pg. S2.6, April 1996.

[21] IOM, “Telemedicine: a guide to assessing telecommunications in health care”, Institute of Medicine, National Academy Press, 1996.