Interoperability and Integration Considerations for a Process-Oriented Clinical Decision Support System

H. Shah M.B.B.S, M.Surg.,
G. Krishnan, P. Williams, A. Vogler,
R.D. Allard M.D.
Henry Ford Health System
Detroit, USA
proteme@gmail.com

P.M. Nadkarni M.D.
Center for Medical Informatics
Yale University Medical School
New Haven, USA

***Abstract*—Electronic medical record systems (EMRs) can be made more attractive to the clinicians if Clinical Decision Support Systems (CDSS) are integrated with them. However, CDSS have to be developed with integration in mind, such that they may be integrated not just with the local EMR but EMRs developed by others as well. Web Services Technology ameliorates the challenge of integration if the CDSS is well-designed but several other issues still need to be considered. The integration has to allow two-way data exchange between the CDSS and the EMR, which requires the EMR also to expose a set of interfaces. Further, the CDSS itself needs integration with services on which it depends for its functionality. In the semantic data capture initiative (SDCI) project, we integrated, Proteus (http://proteme.org), an open source, process-oriented clinical decision support system with Henry Ford Health System's EMR, CarePlus. The effort involved addressing some of these challenges and some that are unique to a system like Proteus.**

Keywords-Web Services, Integration, Clinical Decision Support, Electronic Medical Record, Clinical Process, Clinical Workflow

I. INTRODUCTION

Healthcare professionals have increasingly started adopting computer based systems to record patient data, a trend that is encouraged more by carrots and sticks policies of governments and organizations than by inherent benefits from these systems [1]. However, sustaining the increase in use will require the systems to offer significant advantages to the clinical staff, directly from their use [2]. Clinical Decision Support Systems (CDSS) have demonstrated the potential of making electronic systems valuable to the clinicians while improving the quality of healthcare [3][4]. Despite their potential the use of CDSS is not as widespread [3][4].

The acceptance of CDSS by clinicians can be significantly improved if they are well integrated with the EMRs. The CDSS-EMR integration allows previous data from the patient to be accessed by the CDSS for its own inferencing needs and to save any data that the CDSS may collect or generate to be saved back into the EMR. This helps in avoiding redundant data entry and errors due to missing data or due to incorrect data being entered. The integration between the two systems also has several other advantages such as single log on and consistent user interfaces.

There are many examples of successful efforts of integration of CDSS and EMRs but most such efforts target one time integration of a custom made CDSS with a specific EMR. The CDSS in such efforts are developed for use with that EMR alone and are not suitable for integrating with other EMRs. Since these CDSS cannot be reused, much duplication of efforts occurs each time a new one is created. Therefore, it is an obvious imperative that a CDSS be developed in a manner that allows integration, with minimal efforts, with other clinical information systems (CIS). If the CDSS is designed with integration in mind, Web services technology (WST) offers a relatively straight forward way for doing so. Web services technology also offers a mechanism for the CDSS to interact with other resources that it requires for making its inferences.

Henry Ford Health System (HFHS) is an integrated healthcare organization and the largest healthcare provider of the state of Michigan, USA. In the Semantic Data Capture Imitative (SDCI) project at HFHS, we integrated an executable clinical process based decision support system, Proteus, with the organization's EMR, CarePlus. We adopted WST as the main mechanism for this integration and we plan to continue to improve the Proteus system by developing additional web services to leverage other resources. Recently we successfully concluded pilot testing for the system, in which we deployed two clinical processes, one to help to diagnose upper respiratory infection and the other, to assist in the assessment of known patients of hypertension during their follow up visits. These processes were designed to assist in making clinical decisions, and to ensure that data collected for them was consistent, complete, standards-based and error-free.

In this paper, we briefly discuss the integration challenges for an decision support system and describe how we addressed some of those challenges to integrate Proteus with our EMR in the SDCI project. Since Proteus is a process-oriented decision support system, several other issues were needed to be addressed.

II. PROTEUS

Proteus (http://proteme.org) is an open source system that allows clinical processes to be authored using reusable modules called Knowledge Components or KCs. The KCs

978-0-7695-4461-8/11 $26.00 © 2011 IEEE
DOI 10.1109/SERVICES.2011.14

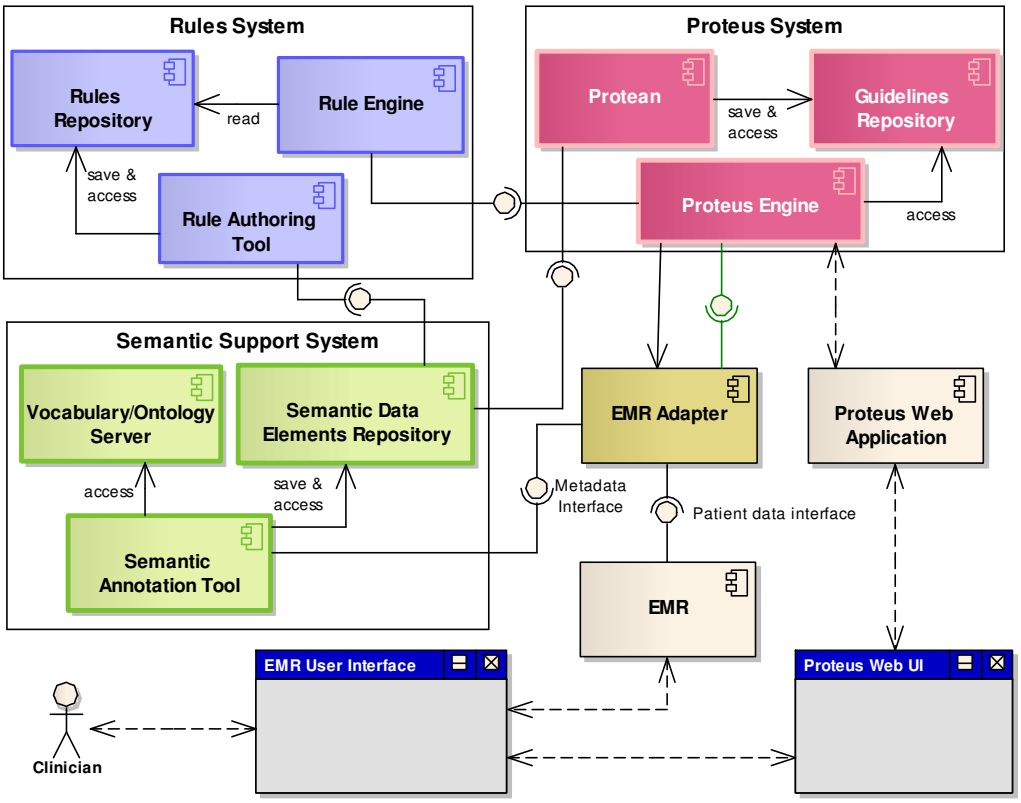

Figure 1. Different components of our system and their integration through APIs to allow exploiting Proteus system for clinical decision support

and processes can then be executed by users to get intelligent guidance[5][6]. The KCs represent activities in the process and have a graphical representation similar to Activity diagramming of UML. The modularity and graphical notation system allows creating workflows by organizing activities in many different ways e.g., by sequencing and nesting of activities. Each KC abstracts the values of other KCs nested within it to a single value, to reduce complexity. The inferencing needed to create the abstractions and to decide which next activity to trigger and thereby determine the subsequent path to follow within the workflow is delegated to other software components called Inference Tools. The inference tools are specified in Proteus only as interfaces, allowing any technology appropriate for the inferencing task for the KC to be deployed.

By using Proteus, recommended medical guidelines can be included into clinical processes, effectively creating an executable version of the guidelines. Since Proteus offers ease of editing for the clinical processes, rapid testing-modification iteration cycles allow creating executable processes that fully meet the expectations of the clinicians. In the SDCI project, we were able to demonstrate feasibility of the approach and some of the benefits that it offers the clinicians.

III. Integration Challenges for Proteus

The challenge of integrating the CDSS with EMRs is not a new one. One reason why Arden Syntax, a language proposed to represent clinical decision support logic [7] in the early 90s, could not achieve wider adoption was its lack of a feasible mechanism to access data from diverse EMRs in a transparent and portable fashion [8]. The data access mechanism for Arden Syntax was devoid of an abstraction layer for the data, making portability of logic modules very difficult. The syntax was designed for authoring of its logic modules by the doctors by shielding them from the complexity of computer systems. However, porting of one module to another system required tackling site-specific details to ensure the data accessed from the EMR matched the data used by the rules in the module – a task certainly beyond most doctors. Further, if the definition of an element changed because of system redesign, every Arden module that utilized that element had to be modified because the old access method was rendered invalid.

Fig. 1 provides an overview of the major components and the interfaces between them, which allowed integrating the Proteus system with the EMR. These are discussed in the following sections.

A. *Accessing of EMRs from the Proteus System*

At each step, an executing Proteus process seeks patient data pertinent to that step, from the substrate EMR or from the user. The EMR data is pulled up to pre-populate appropriate fields, whereas additional data (e.g., clinical observations and test results that are not already in the EMR) are entered by the users in the fields of the data entry templates generated by the Proteus system. The user-entered data and the interpretations that the Proteus system makes on the basis of that data are stored back into the EMR. This two-way data transfer between the CDSS and EMR is the most important integration point. Interestingly, this is not really within the scope of CDSS design but that of the EMR; this integration requires the EMR system to expose its API. Ideally, such an API should be conformant to an information model that reflects the semantics of the healthcare domain. The leading healthcare information technology standard developing organization (SDO) Health Level 7 (HL7) has a Reference Information Model (RIM) which allows representing such semantics [9]. An effort within HL7 is attempting to create a set of interfaces that are true to RIM's worldview, collectively labeled as Virtual Medical Record (vMR) [10]. However, the vMR effort is still in its early stages and even adoption of HL7's RIM is very limited; not many EMRs in existence can be expected to offer RIM conformant APIs. For these reasons we decided that it was more pragmatic to achieve interoperability at data element level rather than interoperability based on navigating the relations between different classes of HL7 RIM. We developed an API for our EMR, the patient data interface, which allows retrieval and storage of values at the level of granularity of data elements. These were then made accessible as a set of SOAP based web services.

All data retrieval and storage requests were confined to occur from within a single class of the Proteus system called *EMRAdapter*. This was to follow the principle of loose-coupling, with the aim to minimize the efforts of porting the Proteus system to any other EMR, when needed in the future; the EMRAdapter is the only class that will need to be modified when the system has to interface with different EMR. The calls to the EMR web services are mediated through this class. This class accesses the database tables which contain the mappings between the data element representations of the Proteus system and means to retrieve them from the EMR. For instance, an entry in the table for a data element, 'Gender', would provide details about the web service method to be invoked and its parameters, to get the value for it from the EMR. Once a data value is retrieved, it may need to be further massaged to match the version that Proteus understands, by using other fields in the mapping table, before it is passed on to the Proteus engine. For instance, if the EMR's data element for gender has a value "Ambiguous" and if the equivalent data element used by Proteus not have any value matching the concept, it could be mapped to a value, most acceptable amongst the ones that are available (e.g., "Other"). The contents of the mapping table may be created when the Proteus processes are authored, either by using the authoring tools by accessing the EMR's metadata through a separate web service (discussed in section "Accessing Resources for Semantic Interoperability", below). Integrations with other EMR systems do not have to follow this mapping table based approach but could implement their own mechanism by inheriting from the EMRAdapter class and modifying its behavior. The EMRAdapter class may be considered as an equivalent of the vMR, discussed before, and in future may be made compliant to the standard.

It is pertinent to note that much of Proteus system is developed on Java platform while the CarePlus EMR is developed using Microsoft technologies. To facilitate interaction between the two without WST would be much more difficult. The EMR team used .NET technology to create the required web services, which were developed in multiple layers to provide loose-coupling components to serve as "glue" objects to the EMR backend. The web service was deployed on a Windows server and contained custom-built modules that served as a middle layer between the public web methods and the EMR backend processes and database repository.

```
<s:element name="getPatientLabResultsByTestCode">
  <s:complexType>
      <s:sequence>
          <s:element minOccurs="0" maxOccurs="1"
            name="PatientID" type="s:string" />
          <s:element minOccurs="0" maxOccurs="1"
            name="QueryFromDate" type="s:string" />
          <s:element minOccurs="0" maxOccurs="1"
            name="QueryThruDate" type="s:string" />
          <s:element minOccurs="0" maxOccurs="1"
            name="TestCodes" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1"
            name="LastResultsOnly" type="s:boolean"
            />
      </s:sequence>
  </s:complexType>
</s:element>
```

A method definition in the EMR web services is displayed above as an illustration.

B. *Accessing CDSS from EMRs*

1) *Need for APIs for CDSS*

Most CDSS are created independently and exist as add-on adjuncts to CIS such as EMRs, i.e. they are not at the core of the tools that support clinical activity but adventitious features, invoked only by some event in the clinical process (e.g., prescription of a drug to detect any contraindications or interactions with other drugs) or by the clinician explicitly requesting additional intelligence for a decision about to be made. In this relationship between the CIS and the CDSS, the former retains the control of the process and invokes the services of the latter, only as needed. Therefore the CDSS have to expose their methods to accept such requests from the client CIS. The methods accept data from the patients and their clinical contexts as parameters and return, as their outputs, recommendations for the context.

2) *Need for a Standard for APIs – HSSP CDSS services*

The notion of a standard API for compliance by CDSS from different organizations is an attractive one. Such an API

can allow the CDSS to be used with different EMRs, allow different systems to swap CDSS service from different vendors, and may lead to sharing of knowledge and rule bases, across the systems used at different provider organizations. However, until now all efforts to create standard CDSS service specifications have not been pursued with much diligence and have yielded little. Currently, a project to define such specifications is in progress within the Health Services Specification Project (HSSP), a collaboration of two organizations, HL7 and Object Management Group (OMG) [11]. This effort envisages the decision support service to provide access to the functionality of logic units, known as Knowledge Modules (KMs), each of which is responsible for providing an inference based on the data provided to it. Each KM is equivalent to a logical function, e.g., If last HbA1c >= 7% then the recommended retesting frequency is every 3 months, else the frequency is every 6 months. The KM specifies the requirements for inference making, e.g., the value of "HbA1c" data element be available and that the patient be a known case of diabetes, in the previous example.

3) Why Proteus cannot be a HSSP Clinical Decision Support Service?

The Proteus approach can effectively leverage the HSSP services when they become available, by designating KMs provided from such services as inference tools for its KCs. Though Proteus functionality could be offered as a HSSP-CDS service, it is not particularly suited to play this role. This is because the focus of Proteus is clinical processes and activities rather than inferencing; it uses inference making systems rather than serve as one. Inference making in Proteus is done to interpret the data from activities or to decide whether other activities have to be initiated, and may be delegated to other components (see section, 'Accessing Logic Components of Proteus' below), such as those proposed by HSSP. Further, once a Proteus process is launched, the control of flow of the process, either intelligently or simply by following a pre-defined sequence, is done from within the process, i.e., the workflow control remains with Proteus. Thus, the Proteus system can be a client of the proposed HSSP CDS services but it cannot be a service provider.

4) Accessing Proteus from the EMR in SDCI project

In the SDCI project, the only way to launch the Proteus system is from within the context of a user-clinician + patient combine that is already open within the EMR system. This leads to displaying of the Proteus web application's home page in a web browser component within the application. Once the Proteus application is launched, the workflow is assumed to be under the guidance of Proteus system. The Proteus system is invoked only once by the EMR, at its launch. The URL used for launching the web app also provides the encrypted authentication information about the clinician and the patient. As discussed above, the decision support provide by a Proteus system is continuous, step-wise guidance, in contrast with the other types of CDSS, which are only triggered intermittently, by discrete events the CIS is programmed to recognize or by explicit user requests for its assistance. For these reasons, in the SDCI project we did not develop web services to expose Proteus functionality for invocation by external applications. We do believe that in the future such calls would be necessary e.g., for accessing the state of a particular process or details of past steps, to expose process semantics. The application can then use that data to do other interesting things, e.g., estimate the costs of managing the condition based upon the likely course the patient may follow going forward. We plan to develop and publish the required APIs for this purpose.

C. Meeting Logic Needs of Proteus

When an executing Proteus process needs to create abstractions or decide next actions it seeks the intelligence required for doing so from the Inference Tools. The inference tool in Proteus model is, by design, specified only as an abstract entity. This allows any software component to be treated as an inferencing mechanism for a Knowledge Component by wrapping it in an adaptor to make it conformant with the Proteus inference tool interface specification. Thus the computing technology most appropriate for the kind of intelligence required by the KC may be exploited. Examples of technologies that may be used to provide this intelligence include simple algorithms as binary components, rule engines, artificial neural networks and image processing systems and even a human expert may be designated to serve as a Proteus inference tool.

Though the technology-neutral inferencing approach of Proteus makes it powerful and flexible, it also poses some challenges. The computing resources demand for the software system serving as inference tool could vary from trivial to very heavy. Indeed, in our pilot project, performance profiling revealed that our computing resources were most taxed by the inferencing needs. Depending on the degree of nesting of KCs, many inference tools would be working simultaneously or in rapid succession, increasing the needs for computing. To be able to distribute such computing load over different processors would significantly enhance performance. Further, the actual software component (or human expert) specified as an inference tool could be at any physical location, possibly even remote ones. These issues make for an ideal setting to deploy web services which can help tackle both these challenges. We are now in the process of developing web services for Proteus tools to reap these advantages.

D. Accessing Resources for Semantic Interoperability

Simply being able to access third party computational resources and software capabilities, using mechanisms such as WST, is not enough. To be able to work effectively with the information exchanged, all parties involved in the exchange have to 'understand' the information. For example, an algorithm invoked by an application might return the diagnosis to be 'pre-diabetes' for a patient but if the application has no notion of 'pre-diabetes', it will be of no use to it. The result is much like putting a US dollar bill in a change machine in USA only to get coins of Indian currency, of little use to you in USA. The application and the algorithm, in the example above, need to have a shared understanding of the currency of exchange. The shared

understanding could be at a macro level, the information models of the healthcare domain and the APIs that represent them or at a finer level of granularity. However, not only are systems based on shared models of semantic interoperability such as HL7 RIM not widely available yet (discussed in section, "*Accessing of EMRs from the Proteus System*", above), they will still leave much scope for semantic mismatch due to 'too many moving parts' problem when they do become available. For these reasons, we believe that the ideal unit of semantic exchange is a data element. We have therefore structured our clinical rules syntax to accept standard-based (ISO/IEC 11179) data elements which allow annotation with concepts from ontologies or controlled medical vocabularies such as SNOMED-CT or UMLS. An ISO/IEC 11179 standard-based data element can not only be understood by the applications, they make complete sense to the non-informaticist domain members (subject matter experts or end-users), they are convenient for the programmers, and they can be in perfect alignment with the domain information models created using UML's class modeling language.

As is the case with the HL7 related models and standards, the adoption of ISO/IEC 11179 is limited in the present day EMRs. For this reason, we created a tool to allow creating standard-based versions of the non-standard data elements in use by the EMRs, which allows parsing each data element into its constituent components and annotate the components with concepts in SNOMED-CT. The existing data elements are accessed from the EMR using another interface, the metadata interface. As with the patient data interface, the access to this interface is also localized within the EMRAdapter class, described earlier. The concepts used for annotation are accessed from a vocabulary/ontology server, the Apelon's open source Distributed Terminology Server. The annotated data elements are then stored in the semantic data element repository for later use by other systems. Together the semantic annotation tool, the semantic data element repository and the vocabulary server are known as the Semantic Support System.

In the course of our future development, we expect all tools developed by us will exchange information expressed in data elements like these. The authoring tools for the clinical processes and for rules, for example, will be accessing common data element collections with semantic underpinnings that are common to both. This way if a rule is authored to refer to a data element, "Gender", it will still be correctly recognized by a Proteus KC, even though the KC's version of the data element is named "Sex' and have the same set of permissible values, because they have the same data element ID assigned to them by a data element registry to which they both comply.

If the authoring tools for processes and rules that use them, as well the engines that execute them (viz. Proteus engine and the rule engine) rely on a standard collection of data elements, it is possible to have comprehensive, enterprise wide semantic interoperability. Each of these components could then be replaced by a different piece of software, if they expose same interfaces and allow using the data elements belonging to a shared semantic space. We have plans to make services of both, the data element repository and the ontology/vocabulary resources, available using web services as part of our future steps

IV. Discussion

The task of integrating a CDSS with EMRs is feasible yet a challenging one, and is rendered somewhat easier with the use of WST. The requirements for a process-oriented decision support system, like Proteus, make these challenges even more daunting and consequently a technology like web services even more desirable. To effectively leverage WST, the components that need to be integrated have to be ideally designed ground up, with integration as a core consideration; integration as an after-thought is far more difficult. The CDSS itself depends upon other components such as rule engine and common data element repository, which calls for additional integration efforts and opportunity to exploit WST.

The integration needs and challenges are highlighted by the needs for CDSS but are not unique to it. Many other tools and systems for healthcare could benefit from similar approaches. An indication for the need for integration comes from the Meaningful Use criteria [12] listed by the US government for incentives and penalties related to healthcare IT adoption. Although the regulation does not specifically mention integration as a criterion but many of the criteria, in particular those proposed for the stage 2 and 3, can only be met by rich integration between different parts of the healthcare information systems. The regulation not only covers CDSS explicitly but also lists criteria that will require features that will need integration with CDSS. Such features include those related to prescription, patient self-management, clinical summaries, medication reconciliation and CPOE.

The recent thrust by the government to promote health information exchanges, at the national and regional level, and the related development of the NHIN project [13] is yet another indication of increasing significance assigned to the interoperability of systems. A group of US federal agencies is currently in the process of developing an open source infrastructure to fulfill the NHIN goals [14]. When the infrastructure becomes available, the systems developed with integration in mind will be well positioned to benefit from it.

Acknowledgment

Funding for the SDCI project came from U.S. Army Medical Research and Materiel Command (Award Number: W81XWH-08-2-0073 / W23RYX8028N603) and the project was administered by the Telemedicine and Advanced Technology Research Center (TATRC) of U.S. Department of Defense. We are also grateful to the Henry Ford Health System's CarePlus EMR development team for its cooperation in meeting the engineering challenges to make this integration effort a smooth operation. We are grateful to Teresa Hantz for playing a leading role in testing, disseminating information about the project, and in creating and managing the post-pilot survey. We truly appreciate the HFHS employees who volunteered to test and review our applications. Their feedback and suggestions allowed us to

develop tools that were both, useful and error free. Finally, we remain grateful to the clinical and administrative staff of HFHS Canton clinic, who took time from their demanding work to learn the applications and for their keenness to use them extensively during the pilot phase of the project.

REFERENCES

[1] W.A. Bowes, "Assessing readiness for meeting meaningful use: identifying electronic health record functionality and measuring levels of adoption" Annu Symp Proc. 2010 Nov 13;2010:66-70.

[2] J. Fortin and W. Zywiak, "Beyond meaningful use: getting meaningful value from IT" Healthc Financ Manage., 2010 Feb, 64(2):54-9

[3] J.A. Lyman, W.F. Cohn, M. Bloomrosen and D.E. Detmer, "Clinical decision support: progress and opportunities" J Am Med Inform Assoc. 2010 Sep-Oct;17(5):487-92

[4] R.A. Jenders, J.A. Osheroff, D.F. Sittig, E. A. Pifer and J.M. Teich, "Recommendations for clinical decision support deployment: synthesis of a roundtable of medical directors of information systems" AMIA Annu Symp Proc. 2007 Oct 11:359-63

[5] H. Shah, "Proteus – a model for clinical protocols created from knowledge components" Los Alamitos, CA: IEEE Computer Society. July 20;01, 59-64

[6] H. Shah and G.R. Thoma, "A software tool for editing and executing knowledge component based clinical guidelines" Proc AMIA Symp. 2002

[7] G. Hripcsak, J.J. Cimino, S.B. Johnson and P.D. Clayton "The Columbia-Presbyterian Medical Center decision-support system as a model for implementing the Arden Syntax" Proc Annu Symp Comput Appl Med Care. 1991:248-52

[8] R.A. Jenders, R. Corman and B. Dasgupta "Making the standard more standard: a data and query model for knowledge representation in the Arden syntax" AMIA Annu Symp Proc. 2003:323-30

[9] G. Schadow, D.C. Russler, C.N. Mead and C.J. McDonald, "Integrating medical information and knowledge in the HL7 RIM." Proc AMIA Symp. 2000:764-8

[10] "Virtual Medical Record (vMR) - HL7Wiki." [Online]. Available: http://wiki.hl7.org/index.php?title=Virtual_Medical_Record_(vMR). [Accessed: 17-Mar-2011].

[11] "The HL7-OMG Healthcare Services Specification Project: Motivation, Methodology, and Deliverables for Enabling a Semantically Interoperable Service-oriented Architecture for Healthcare."[Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3002132/?tool=pubmed. [Accessed: 17-Mar-2011].

[12] Office of the National Coordinator for Health Information Technology, Department of Health and Human Services, "Health information technology: initial set of standards, implementation specifications, and certification criteria for electronic health record technology. Interim final rule" Fed Regist. 2010 Jan 13;75(8):2013-47

[13] "HealthIT.hhs.gov: Nationwide Health Information Network." [Online]. Available: http://healthit.hhs.gov/portal/server.pt?open=512&mode=2&cached=true&objID=1142. [Accessed: 17-Mar-2011].

[14] "The Direct Project." [Online]. Available: http://directproject.org/. [Accessed: 17-Mar-2011].