**OpenEMR Architecture**

**T. Simbini**

From GPLMedicine

16 February 2006

**Contents**

1 OpenEMR Architecture   
1.1 Background   
1.2 Introduction   
1.3 Distributed Computing Environment   
1.4 Centralized Computing Environment   
1.5 Data transmission   
1.6 Data exchange between disparity systems.   
1.7 Working Environment   
1.8 Challenges of using openEMR.   
1.9 Acknowledgements   
1.10 References

**Background**

My task in the project is to give a technological background to the OpenEMR system. Specifically to discuss transmission protocols and data storage, and discussion of OpenEMR's computing architecture including implementation issues present in distributed vs. centralized computing.

**Introduction**

Web based electronic medical record systems allow for multiple users to enter data simultaneously from different geographical locations. This is an ideal solution in a hospital setting where different health care providers enter patient data from different locations. OpenEMR software suite is a web-based application that allows users access patient data from any computer on the Internet or the Intranet. Web-based electronic medical record systems reduce the total cost of ownership through increased sharing of resources, low end user equipment costs and, depending on the computer architecture, low maintenance costs.

OpenEMR is set up as a three tiered web page design with communication passing from the client Web Browser to the receiving Web Server in an unencrypted request format (see Fig. 1). This transmission is passed using Transmission Control Protocol/Internet Protocol (TCP/IP) Suite. If the administrator sets the communication for the Web Server to use Secure Socket Layer (SSL) sessions then encryption will be used. The Web Server (Apache mostly) receives the browser request and uses the PHP parser to evaluate the request and act on the incoming packets. The Web Server then passes queries to the database engine as necessary.

The database engine executes the queries on the database server (MySQL). After executing the query the query result is returned to the Web Server through the PHP engine, which forms an http page with the appropriate formatting. The Web Server returns this http page to the requesting browser. The Web Server can be any PHP compliant web server and openEMR uses Apache mostly. The database server theoretically can be any SQL database. MySQL is the standard backend database that openEMR uses. OpenEMR has not yet been tested on other databases such MS SQL Server or Oracle.

Fig. 1 Schematic flow chart of data transmission in openEMR. The three-tier web design means the components of openEMR can run from the same computer or from three different computers. Depending on the setup environment openEMR can run as distributed (client server) or centralized computer system.

**Distributed Computing Environment**

A distributed computing system consists of a collection of autonomous computers connected through a network and distribution middleware, enabling computers to coordinate their activities and to share the resources of the system, so that users perceive the system as a single, integrated computing facilityi. All components (database servers, billing servers etc) of openEMR can be distributed transparently across a network of heterogenous computers and still be perceived as a single computing facility by the users. OpenEMR allows for MySQL database to run from one computer, FreeB billing server from another computer and SQL-Ledger accounting system can be running on a different computer. This setup increases flexibility of using openEMR in a major hospital setting where updates are done on one machine and replicated to all other computers on the same network. In addition there is low data transmission costs with increased information processing capabilities. However, compared with centralized computing, end-user equipment costs and network management costs are increased in a distributed computing environment hence it is ideal in major hospital settings (see Fig.2)

Fig 2. Distributed Computing System

**Centralized Computing Environment**

Physicians’ office practices, clinics and small private hospital would not necessarily require the excessive processing power provided in distributed computing systems. An office practice might require just cash receipting stationsii across an inexpensive network. OpenEMR is flexible enough to allow for centralized computing in small practice office to reduce the implementation costs. In a centralized computing environment all critical data and programs are stored on one main computer, the central computer. OpenEMR can be implemented using thin clients or fat clients and one central server2. The centralized computing environment has low end-user equipment costs, and lower network management costs (see Fig. 3).However, central computing has performance implications closely linked to the hardware and software that is in use.

Fig. 2 Centralized Computing

**Data transmission**

Patient data in openEMR can be transmitted on the Internet using wide area network technology (WAN) or on the Intranet using local area network (LAN) technology. OpenEMR relies on transmission control protocol/internet protocol (TCP/IP) for data exchange across the network. TCP/IP is a set of communication protocols on which the Internet and the Intranet networks run. TCP/IP has different protocols that operate at different levels of the communication process known as layers. The application layer includes the Hyper-Text Transfer Protocol (HTTP), which supports the World Wide Web application and openEMR browser window. Transmission Control Protocol (TCP) ensures accurate and reliable data transmission across the network. The normal way to secure OpenEMR is to enable SSL in the web server. Since OpenEMR is browser-based this is a straightforward thing to do. In the case where access occurs over the Internet, SSL certificate can be installed in the web browser. This way the client authenticates itself to the server, and outside intrusions become very unlikely.

**Data exchange between disparity systems.**

OpenEMR can export some of its data to other systems. Demographics (names, addresses, phone numbers, insurance information) can be transferred directly but would require a filter for the receiving system to know which fields mean what. The easiest way to transmit the information would be an output in a standard format such as HL7. The X12 format is already being used in this way for transmission of financial information for electronic billing between practices and insurance intermediaries. Other data are kept in the database only as a series of yes/no check boxes. This information would be hard to interpret without the associated web page information. This would be typical of the clinical encounter forms. Doctors generally are not typists and prefer point and click systems. These are kept in the database as Boolean information (yes-no, true-false).

**Working Environment**

OpenEMR has been tested in a 1-5 practitioner environment and works very well. It has not been tested in a large multi-user or multi-clinic environment.

**Challenges of using openEMR**

Installation can be problematic if the instructions are not carefully followed. The system works on all known distributions of Linux, Mac Os X, Windows 2003 server and Windows XP Professional. The installation on Windows 2000, Windows XP Home Edition, FreeBSD and OpenBSD are less friendly and usually take quite a bit of troubleshooting. PHP‑MySQL based web sites are not very secure since it is open source and the code is known. As a result openEMR would not be suitable for direct access over the Internet.

**Acknowledgements**

Dr. Samuel Bowen. Staff at Frye Regional Medical Center and Catawba Valley Medical Center in Hickory, N.C. provided most of the background information on the architecture of openEMR for this project. Dr. Rod Roake. Offered his opinion on the network security of openEMR.

**References**

W. Emmerich. Distributed System Principles [online]. Cited online 13 February 2006. Available from website: <http://www.cs.ucl.ac.uk/staff/W.Emmerich/lectures/ds98-99/dsee3.pdf>

J. Stafford. Cutting out Windows and fat clients [online]. Cited online 13 February 2006. Available from website: <http://searchopensource.techtarget.com/originalContent/0,289142,sid39_gci1066589,00.html>

**128**-**bit encryption** is a data/file **encryption** technique that uses a **128**-**bit** key to **encrypt** and decrypt data or files. It is one of the most secure **encryption** methods used in most modern **encryption** algorithms and technologies. **128**-**bit encryption** is considered to be logically unbreakable