Team Members | Platforms
--- | ---
Albayram, Yusuf | GNUmed
Baihan, Mohammed | CottageMed
Devoe, Gordon | OpenEHR
Etim, David | OpenEMR
Martin, Robert | SMART Platform
Moledina, Saahil | OSCAR McMaster
Papavasileiou, Ioannis | OpenMRS
Rivera Sanchez, Yaira | FreeMED
Sanzi, Eugene | WorldVistA

**Project Description:**

This team development project focuses on large scale issues related to health information technology (HIT) systems such as electronic medical records (EMRs), and personal health records (PHRs). The EMR platforms in this paper are discussed in terms of the capabilities of the individual systems and need for them to interact and collaborate with each other. There is a need to send information among these systems and allow information to be passed between patients and providers, providers exchanging information to other providers, and providers to larger organizations such as clinics, hospitals, and laboratories.

The ability to exchange information at different levels is critical. There could be individual records exchanged as well as programmatic exchange of entire record sets or subsets through APIs, or exchange of summary or statistical data. We are investigating nine platforms and the way that information can be effectively exchanged.
GNUmed

GNUmed is a free electronic medical record (EMR) for Linux, Microsoft Windows and Mac OS X and it aims to provide medical software that respects the privacy of patients and that is based on open standards. GNUmed assists and improves longitudinal care (specifically in ambulatory settings, i.e. multi-professional practices and clinics). It is developed by a handful of medical doctors and programmers from all over the world. The name GNUmed indicates firstly the basis of the GNU Project; the term "med" provides the reference to the medical industry. It can be useful to anyone documenting the health of patients including doctors, physical therapists, occupational therapists, acupuncturists, nurses, psychologists, and others. GNUmed is based on a robust SQL client-server concept and has built in mechanisms to monitor database integrity at any time. GNUmed is written mostly in Python and data is stored in a PostgreSQL database. GNUmed was launched in 2001 and it is being actively developed.

Database Platform and Accessibility:
GNUmed uses PostgreSQL as a database back-end server. PostgreSQL is known to be a rock-solid open source database server and thus it is the right tool for this kind of application. Multiple clients can work with the same database at the same time. GNUmed is accessible through a graphical user interface (GUI) based on WxPython libraries. By making use of GNUmed's interface third party software can interact with GNUmed to make use of its features. Furthermore, a Web-based GNUmed client application is currently being developed.

APIs:
GNUmed is very well documented in all respects and provides a Java-like API Doc (Please see API documentation at [http://publicdb.gnumed.de/~ncq/gnumed/api/](http://publicdb.gnumed.de/~ncq/gnumed/api/)). GNUmed supports a variety of features, many implemented as plugins which extend the core functionality. These can range a medical paper record archiving system to vaccination status handling. A list of features is provided in GNUmed's documentation system can be found at [http://wiki.gnumed.de/bin/view/Gnumed/WhatCanIActuallyDOWithGNUmedToday](http://wiki.gnumed.de/bin/view/Gnumed/WhatCanIActuallyDOWithGNUmedToday). For example, gm_ctl_client is a small helper program acting as a client for the XMLRPC API of the GNUmed client. This API can be used by third party software to remote control a GNUmed client. Some of the main features of GNUmed as listed below:

- Administration
  - E.g., activating patients in GNUmed, creating GNUmed patient from appointment managing stuff members, inbox handling.
- Medical record
  - E.g., getting an overview of patient’s care events such as allergies, lifestyle, family history etc.
- Patient handling
- E.g., adding patients, managing patients (i.e., search for patients, edit patient demographics, concurrently access a patient record), billing (i.e., managing items to be billed to patients, creating/printing bills and invoices)

- **Client functionalities**
  - E.g., Customization (e.g., configuration), interfaces (e.g., displaying a list of online medical knowledge resources, link to an online lab tests database, by using a DICOM viewer to display CD-ROMs containing X-Rays, MRs, CTs etc.), remote control GNUmed from a legacy application.

- **Database functionalities**
  - E.g., automated backup, "HIPAA compliance" oriented auditing and logging, access from Linux, Windows, and MacOSX.

**Patient vs. Provider Access:**
As GNUmed is an electronic medical record (EMR), only medical providers can view and modify patient's records. The system's current access rights model is role-based access control which is used to provide access rights to medical providers. Moreover, GNUmed does not currently support temporary transfer of access rights. Since it is targeted at medical practices with only a few physicians, it assumes an inherent agreement between the patient and all the doctors of any single practice. When another doctor is on duty the simple fact that the patient seeks medical attention implies a patient-doctor agreement.

**Availability of Patient Portal, PHR, ePrescribing:**

- **Patient Portal and PHRs:**
  - GNUmed has neither patient portal nor PHRs, since it is an EMR and it is targeted for medical providers.

- **Support for CDS:**
  - Currently there is no Clinical Decision Support System (CDS) in GNUmed. However, the GNUmed group is developing/improving their system in order to be able to support for CDS. To do so, they are working in conjunction with EGADSS which is a hibernating open source project and informs how GNUmed and other EMRs could in future usefully retool their "export"-like functions in order to aid physicians provide high quality care. Therefore, CDS will be implemented in future releases.

- **ePrescribing:**
  - Currently GNUmed does not have this feature yet. Implementation of this feature is in GNUmed group's road map list and it will be implemented in future releases.

**Installation Instructions (Microsoft Windows):**
The detailed explanation can be found at:
https://drive.google.com/file/d/0B5CK85pH9BlcN203dTZDnnpnUUE/edit?usp=sharing
CottageMed

Introduction

CottageMed is electronic medical record (EMR) software which is built on the FileMaker platform. FileMaker is a database platform similar in many ways to Microsoft Access, but with ease-of-use and security features that make it particularly well suited to the needs of a medical practice. FileMaker runs on Windows and Mac OS. Licenses are inexpensive, approximately $200 per machine. Multi-license packages are available as well as volume and academic licenses. CottageMed chose a hybrid path employing a proprietary database engine for easy installation and wider physician acceptance. In the succeeding years a dozen physicians and Filemaker community programmers have participated directly in the development of this early EMR project. In 2002 CottageMed became the first publicly distributed free EMR to be both open source and cross-platform for PC, Mac & Linux in the world. It remains cross-platform with support for all internet browsers conforming to open standards including Chrome, Safari, Internet Explorer, Opera, and more.

Database Platform and Accessibility:

CottageMed uses FileMaker as database platform. FileMaker is a cross-platform database application. It integrates a database engine with a GUI-based interface, allowing users to modify the database by dragging new elements into layouts, screens, or forms. FileMaker has similarity to Microsoft Access Database. FileMaker server ran on Linux, but Linux support was abandoned with FileMaker 7, and the server currently runs only on Windows or OS X servers. It is available in desktop, server, iOS and web-delivery configurations. FileMaker, since version 9, includes the ability to connect to a number of SQL databases without resorting to use SQL, including MySQL, SQL Server, and Oracle. This requires installation of the SQL database ODBC driver to connect to a SQL database. SQL databases can be used as data sources in FileMaker’s relationship graph, thus allowing the developer to create new layouts based on the SQL database; create, edit, and delete SQL records via FileMaker layouts and functions; and reference SQL fields in FileMaker calculations and script steps. Versions from FileMaker Pro 5.5 onwards also have an ODBC interface. FileMaker 12 introduced a new function, ExecuteSQL, which allows the user to perform an SQL query against the FileMaker database to retrieve data, but not for modification or deletion, or schema changes. FileMaker allows a free import and export patient data in different format. Some of these formats are:

- USR
- fp7
- fp5
- fp3
- fm
- fmp12
- tab
- txt
- csv
- mer
- xls
- xlsx
APIs:
FileMaker provides two main API: API for PHP, and Accessing FileMaker using RESTfm.

FileMaker API for PHP:

The FileMaker API for PHP implements a PHP class—the FileMaker class—that provides an object-oriented interface to FileMaker databases. The FileMaker API for PHP enables both data and logic stored in FileMaker Pro databases to be accessed and published on the web, or exported to other applications. The FileMaker API for PHP allows PHP code to perform the same kind of functions which are available in FileMaker Pro databases:

- create, delete, edit, and duplicate records
- perform find requests
- perform field and record validation
- use layouts
- run FileMaker scripts
- display portals and related records
- use value lists

To use the FileMaker class in your PHP solution, add the following statement to your PHP code:

```
require_once ('FileMaker.php');
```

The FileMaker class defines class objects that you can use to retrieve data from FileMaker Pro databases.

<table>
<thead>
<tr>
<th>Class Object</th>
<th>Use the object to</th>
</tr>
</thead>
</table>
| FileMaker database    | Define the database properties  
                        | Connect to a FileMaker Pro database  
                        | Get information about the FileMaker API for PHP |
| Command               | Create commands that add records, delete records, duplicate records, edit records, perform find requests, and perform scripts |
| Layout                | Work with database layouts                                                      |
| Record                | Work with record data                                                           |
| Field                 | Work with field data                                                            |
| Related set           | Work with portal records                                                        |
| Result                | Process the records returned from a Find request                                 |
| Error                 | Check whether an error has occurred  
                        | Process any errors                                                              |
Accessing FileMaker using RESTfm

RESTfm turns FileMaker Server into a RESTful Web Service, so FileMaker Server databases can be accessed via HTTP using a common REST architecture with easy to understand API calls. RESTfm enables us with Create, Read, Update and Delete operations on FileMaker Server hosted data via standard HTTP GET, POST, PUT and DELETE methods.

Patient vs. Provider Access:

Provider access
Different types of providers can access CottageMed system including: physicians and nurses. Physicians have full access to CottageMed system, using FileMaker. That is, they are able to transfer between all tabs in CottageMed including: access patient information, letters, reports, measures, notes, studies, custom, filling, and account. Nurses are able to access specific parts of patient information. These parts includes: find tool, patient information, account, and notes tabs.

Patient access
Patients have no direct access to CottageMed system. However, they can add their information by registering into the system using registration forms. These forms are: patient registration, personal information, consent to care, consent for release of information, patient medical history, and review of system forms. Information entered by patient will be transfer to the system by nurses.

Availability of Patient Portal, PHR, ePrescribing:
The CottageMed system provides portal for providers only and patients have no such service. Using the portal providers, i.e. physicians and nurses, are able to prescribe drugs and medicines for patients. They can prescribe a drug by navigating to notes tab then press Rx Pad button then typing the drug name and other information.

Support for CDS:
The CottageMed system does not provide clinical decision support service.

Installation:

To install CottageMed we need to setup to pieces of software: FileMaker and CottageMed. FileMaker Pro and FileMaker Server are available directly from the vendor at www.filemaker.com. CottageMed is available for download at http://www.cottagemed.org. FileMaker server ran on Linux, but Linux support was abandoned with FileMaker 7, and the server currently runs only on Windows or OS X servers. It is available in desktop, server. We need to install both FileMaker server and FileMaker Pro.

Steps of installing FileMaker server on Linux server (only old versions of FileMaker and CottageMed):
1. Insert the FileMaker Server CD into your CD or DVD drive.
2. Switch to the root user by entering the following command at the shell prompt:
   $ su -l root
   For information about accessing the shell prompt, see your operating system documentation.
3. Enter the root password.
   The prompt changes to #, indicating that you are now logged in as the root user.
4. Mount the installation CD by entering:
   # mount /mnt/cdrom
5. Change to the CD directory by entering:
   # cd /mnt/cdrom
6. Enter the following command to install the FileMaker Server files on your hard disk:
   # rpm -Uvh fmserver-5.5-1.i386.rpm
7. Enter the registration command followed by your installation code and the location of the
   FileMaker Server binary on your server computer. For example:
   # fms_registration 1-1111-1111-1111-1111 /usr/bin/fmserverd

Steps of installing FileMaker server on Windows server:
1. Exit all other applications and turn off virus protection utilities.
2. Insert the FileMaker Server CD into your CD or DVD drive.
3. Double-click the Read Me icon to read important information about this release of FileMaker
   Server. Then close the Read Me file.
5. Click Next.
6. Read the license agreement. If you agree to these terms, select I accept the terms in the
   License Agreement, and then click Next.
7. Personalize this copy of FileMaker Server by typing your name, organization name, and
   indicate who can access FileMaker Server from this computer (only you or anyone who uses the
   computer). Then click Next.
8. Choose Complete in the Setup Type screen to install FileMaker Server and the example files,
   and then click Next. Note If you don’t have enough disk space on the server computer, you
   see the Out of Disk Space dialog box. Quit the installation, remove files from the installation
   volume to free some space, and try again.
9. Select an option for FileMaker Server start-up, and then click Next.
10. Type your installation code, and then click Next. Tip Your installation code is a seventeen
    digit number located on a multi-part sticker on the installation code sheet with the FileMaker
    Server CD.
11. If the Microsoft Management Console (MMC) software is not installed on your server
    computer, the Setup Wizard will prompt you to install it now. The installation of MMC can take
    several minutes and you can’t cancel this installation once it has begun. If you want to install
    MMC and continue the installation of FileMaker Server, click Next.
12. Click Install to begin FileMaker Server file installation. The Setup Wizard installs FileMaker
    Server 5.5 and the example files.
13. Click Finish to close the Setup Wizard. Remember to turn virus protection utilities back on.
Steps of installing FileMaker server on MAC server:
1. Quit all other applications, especially virus protection software, and disable file sharing.
2. Insert the FileMaker Server CD into your CD or DVD drive.
   You see the CD window on your screen. If you see only the disc icon, double-click the icon to open its window.
3. Double-click the Installer icon named Start Here Mac OS Classic.
4. Read the license agreement. If you agree to these terms, click Accept.
   You see the FileMaker Server Installer dialog box.
5. To install the application onto a different disk or into a specific folder, click the Install Location pop-up menu and choose Select Folder. In the dialog box that appears, choose a destination disk and folder, and then click Select.
6. Click Install.
7. Type your name, company name (optional), and your installation code.
   Tip Your installation code is a seventeen digit number located on a multi-part sticker on the installation code sheet with the FileMaker Server CD.
8. Click OK.
   If the hard disk you chose has enough space, the Installer begins installing files.
9. Click Quit to leave the Installer when the installation is finished.

Steps of installing FileMaker Pro on any platform:
For each platform just run the executable file to install FileMaker Pro on that platform.
OpenEMR

Introduction
OpenEMR is a medical practice management software that supports electronic medical records (EMR). It features fully integrated electronic health records, practice management, electronic billing, as well as appointment scheduling. OpenEMR is a free and open-source software subject to the terms of the GNU General Public License (GPL). There are ongoing efforts by OpenEMR for internationalization and localization in multiple languages, and there is free support available in different forums around the world. This platform is one of the most popular free electronic medical records today with almost 4,000 downloads a month.

Database Platform and Accessibility
The server side is written in PHP and can be employed with any operating system with support for PHP. OpenEMR can be installed using XAMPP database platform which is easy to install for Apache distribution containing MySQL, PHP, and Perl. When installing OpenEMR through XAMPP, phpmyAdmin allows access to users and administrators into the database and updating tables of information for OpenEMR. The database accessibility facilitates efficient office management through automated patient record journaling and billing integration.

APIs
The APIs for OpenEMR were completed with phpDocumentor. The success of completing them were driven through documentation of the OpenEMR codebase, which had phases to allow for new code, existing code to be converted, and organization of codebase. Here are examples of the API documentation:

Interfaces
- AmccFilterIF
- CqmFilterIF
- CqmPopulationCriteriaFactory
- RsFilterIF
- RsReportIF

Classes
- Allergy
- BillingContact
- Communication
- DenominatorAllPatients
- Diagnosis

Patient vs. Provider Access
The administrator privileges allows access to most functions that include: creating and updating appointments, sending messages to other providers and patients, searching for patients, summary of visits, downloading reports, administrative duties for facilities, users, access to
address books, codes, layouts, alerts, and electronic billing access. So when the provider is in the role of administrator using OpenEMR, they have access to everything in the system.

The patient has access to view, print, and download reports and as well as view their lab results and list of problems noted by the clinicians and physicians. The medication list, medication allergy list, and appointments are also viewable to the patient. However, the patient is not able to update any information on OpenEMR without communicating with a physician or another person in the role of administrator to allow for any addition or deletions in these reports or lists. So the patient is very limited to what they can do on their side, but are able to view their medical records electronically.

**Availability of Patient Portal, PHR, ePrescribing**
There is availability of the patient portal and access to personal health records. Physicians and other members of clinics and hospitals are able to electronically prescribe supplements and drugs for patients, but the patients are not able to do that for themselves.

**Support for CDS**
The support enables a limited set of selected users to activate one or more electronic clinical decision support interventions based on the following items: problem list, medication list, medication allergy list, demographics, laboratory tests, and vital signs.

**Instructions for Installing OpenEMR (Ubuntu)**

**Desktop**

1. Download:  

2. Double click openemr_4.1.2-2_all.deb file

3. Click “Install” and enter Ubuntu password

4. During installation, follow prompts and supply user input as needed. If prompted, create a mysql root password if you’re installing the mysql server. Also, enter your mysql root password during the installation.

5. Then login to OpenEMR. For desktop, browse to [http://localhost/openemr](http://localhost/openemr). For server, go to [http://server_IP_address/openemr](http://server_IP_address/openemr). The username is “admin”, and the password is “pass”. The server IP address can be found using the “ifconfig” command.

**Command Line**

**Commands (commands are in red)**

```
wget downloads.sourceforge.net/openemr/openemr_4.1.2-2_all.deb
```
sudo apt-get update
sudo dpkg -i openemr_4.1.2-2_all.deb

If there was an error(s), then issue the following command (type 'Y' after): **sudo apt-get install -f**. If prompted, create a mysql root password (applicable if installing mysql server) and enter your mysql root password during openemr installation.

Then login to OpenEMR:

Desktop - Browse to http://localhost/openemr. The username is 'admin', and the password is 'pass'.

Server - From another computer browse to server at http://server_IP_address/openemr. The username is 'admin' and the password is 'pass'. The server IP address can be found using 'ifconfig' command.
SMART Platform

Database Platform and Accessibility

SMART is broken up into three components; applications, containers, and interfaces. Applications can be developed around the patient or client and provide visual and helpful information within a web browser window. Containers deal with handling the health data and providing context for the app to run on. Interfaces deal with the given APIs, in our case SMART Connect and SMART REST API (which deal with separate areas of the architecture).

The main portion of SMART app runs the container in an IFRAME. IFRAME allows multiple web pages to open on the same screen in a browser window. Authentication is dealt with by using oAuth. This container uses an internal logic to deal with the system's permissions. And finally medical data is handled in a approach which SMART platforms calls an "80/20" approach. This means they try to pick data about the patient from 20% of the fields which represents 80% of the patient's data. An example they give of this is using RxNorm for medications, SNOMED for problems, and LOINC for labs. Additionally SMART uses RDF for data payloads and is extensible as shown in the cloud-hosted sandbox. Data is returned in the form of RDF graphs containing the following:

- Object requested (from GET command)
- Properties of core data elements, meaning the key factors of a given object (like an allergy name)
- Properties of other top-level SMART objects which are possibly linked
- Core data elements which belong to these other top-level SMART objects

SMART Platform provides a cloud-hosted sandbox for SMART EMR. This system currently has about 15 applications and 50 sample patients. Their SMART API illustrates GET command returns in RDF/XML, N-Triples, JSON-LD, and an interactive mode (similar to a command prompt). They also have preset applications like BP Centiles and Growth Charts which allow a user to interact with the given patient's data in a very visual and helpful manner.

One interesting plugin is given in the following link: https://community.i2b2.org/wiki/display/SMArt/The+SMART+Views+Plugin. In this we see the merging of multiple apps to create a framework in SMART. In this way we see the depiction of a patient's information in a more concise view for whoever is viewing the content (researcher, provider or patient).

APIs

SMART uses Connect and REST APIs. SMART Connect deals with the front end components like executing GET calls for patient data. SMART REST deals with the container's back end. The given APIs provide a broad range of different call functions.

SMART has an extensive API which can be found here: http://docs.smartplatforms.org/framework/api/. The various API calls are broken down into three different categories: Container calls, Record calls, and User calls. Container calls consist of
more generic call to a container. An example of this call could be sending a GET command to a database container to either get a user by a given ID number or by name. The User calls allows an admin to adjust a user’s preferences by either deleting, adding, or fetching their given SMART preferences. And finally the Record calls consist of all other API commands. These could be getting a user’s allergies, accessing their lab results, or getting all clinical notes on a given patient (to name a few examples).

While SMART API is not extremely large, they do note that this work is in no way their final model. Therefore, we should expect for this API to continue to grow with the expansion of SMART Platform. Additionally, the current API has many milestones currently under construction which allow users to track current development.

Current API client libraries consist of the following programming languages:
- JavaScript
- Python
- Java
- .Net
- iOS

Patient vs. Provider Access

SMART platforms is developed more for the provide, in which they can handle information in meaningful ways. SMART allows users to view data is visually appealing ways through the integration of frameworks and APIs. Based on some of the current apps a provider can send patient information to HealthVault very easily. There are also many other monitoring and apps to directly assist the provider, like BMJ Content Discovery to assist in diagnosis of a patient. While these are only two examples, SMART has the capability to export its information to other sources based on the open concept of its applications. Furthermore, this open concept allows for a patient like framework to be constructed, with authentication and other components being monitored by the different containers in place. For the current pre-installed apps in the provided sandbox, SMART is focused on provider access.

Availability of Patient Portal, PHR, ePrescribing

Currently there is an outsource to HealthVault for SMART, but based on its open app concept this is not limited to this patient portal. SMART can run the following (without any adaptation from the developer):
- SMART i2b2, which is a clinical discovery system
- SMART Indigo, a health record system
- Mirth Results, clinical data repository system
- Think!Med Clinical, openEHR-based clinical information system
- WordVistA, Veterans open source EMR

In this way, we can see just an example of the ease of availability and integration of SMART into many systems in current use today.

Support for CDS
Current support for CDS is limited to applications like BMJ. This app outsources you to BMJ's website to provide a second opinion based on the given criteria. While this does not match an integrated software program within SMART it does show its ability to have CDS applied in the near future.

**Additional Details:**

SMART is an application driven platform. This platform gives users the framework, based around medical needs, and lets them build everything up from there. While this type of application may not be friendly to small clinicals wanting to customize a system around them, it does have a large amount of potential based on its flexibility and adaptability.

**Installation Instructions:**

Application Development environment and startup (very easy):

- Account creation following simple directions on the following link (to run off their sandbox):
  - [http://docs.smartplatforms.org/guide/setup.html](http://docs.smartplatforms.org/guide/setup.html)
  - Works extremely easy with only a account creation needed

- Setting up your app and using SMART API is given by the following link:
  - [http://docs.smartplatforms.org/guide/quickstart.html](http://docs.smartplatforms.org/guide/quickstart.html)
  - Tested quickstart. Very easy installation with little background knowledge needed. Just need to follow the guide provided. Can verify by having webserver response to an html file and outputting a given patient's name under “My App” in the given SMART sandbox.

- Linux server installation instructions (not tested):
  - [http://docs.smartplatforms.org/framework/reference-implementation/install-ubuntu.html](http://docs.smartplatforms.org/framework/reference-implementation/install-ubuntu.html)
OSCAR McMaster

OSCAR McMaster is a web-based electronic medical record (EMR) system initially developed for academic primary care clinics. It has grown into a comprehensive EMR and billing system used by many doctor’s offices and private medical clinics in Canada and other parts of the world. The name is derived from where it was created and an acronym; OSCAR stands for Open Source Clinical Application and Resource and McMaster refers to McMaster University, where it was developed. It enables the delivery of evidence resources at the point of care.

Database Platform and Accessibility:
MySQL or Postgres can be used as database to make Drugref2 (http://code.google.com/p/drugref2/downloads/list) which is the database they use. Hibernate is used as an interface between java and MySQL.

Access of the database is done by DAO /Model Object idiom. JPA with annotations (the model object configuration and the dao transaction boundaries) is used to do the Object Relational Mapping(ORM), Spring is used to manage the transactions. The objects should be loosely coupled in terms of relationship mappings, they should not be embedded. We need to create one Dao and model class per database table. Normal Standards of JPA/Hibernate/OpenJPA are followed. There are 2 classes you should subclass for your model and Dao objects, AbstractModel.java and AbstractDao.java. Relationships between model objects should not be embedded. i.e. if User and Role are two top level entities, there should not be a User.getRoles() method. Instead you should have a RoleDao.findRolesByUserId() method. Note that the Dao class should always return items of its own type. i.e. you should not have a UserDao.findRolesByUser(). If it's a one-to-many relationship you should still separate the 2 entities. i.e. User to PetDogs, PetDog should have a OwnerId. So to retrieve a PetDogs owner you would do UserDao.find (petDog.getOwnerId ()), whereas to find a user's PetDogs you would do PetDogDao.findByUserId(user.getId()). There are memory, CPU, and coupling implications of using embedded vs. Non embedded objects. The simplest and most cpu and memory friendly way is to use loosely coupled model objects. Especially when transactions are in use, a lot of people get confused when different objects which are embedded are either saved when not expected or saved in a different order causing foreign key errors etc. The price of doing so is that the code is a little more verbose and relationships need to be manually maintained. The conceptual model is simpler and less error prone though.

Hence JPA Spring annotated configuration is used and no xml file required for Dao objects (there is some xml to configure the original spring + JPA framework but there's no need for xml entries for each class). Keep model objects loosely coupled and not embedded. JDBC connections are used for Entity managers. Finally there is Report by Template which allows users to create custom SQL queries.

APIs:
It used Hibernate as an API to access the database but now it changed that to JPA with annotation. It used JSP for the front end purposes and it has java libraries for building and interacting the components. OSCAR is open source software and its API's are available to public for development purpose. Security should be the most important feature of all. An EMR is
only as secure as its least secure component, so there should be focus on abstracting away as much data access as possible from the application so that it stores and retrieves data in a sandbox-style environment through a main data access layer API that has been audited by a third-party and found to be adequate for storing medical information. Hence a good API is very important for OSCAR and it has one. OSCAR API has many components like oscarEncounter is the patient's electronic medical record. It contains all of the patient's medical information, including patient history, progress notes, forms and documents. oscarRx is the OSCAR prescription program, which allows you to create electronic prescriptions, save personalized prescription information, and stay informed about possible drug interactions and allergy conflicts for each patient. Prescription printing on label printers is supported, as are "wet" signatures placed on electronic prescriptions using external signature pads. oscarBilling is an integrated billing module that supports automated billing submission in the provinces of ON and BC. oscarDocument, a system for attaching any type of file to a patient's chart.CAISI, (Client Access to Integrated Services and Information) an award-winning case management, bed management and program facility management system used by several agencies in Toronto, Hamilton and Ottawa. The system is specifically designed to share relevant patient information among providers, while maintaining the highest levels of patient privacy. OSCAR Integration (CAISI): able to seamlessly integrate electronic patient information between separate EMR installations including seamless patient controlled integration of progress notes, practice membership, medication information, prevention information and ability to make electronic referrals between clinics (e.g. primary care OSCAR office and specialist OSCAR office).

**Patient vs. Provider Access:**
The permission model for Oscar McMaster was through a project CAISI (Client Access to Integrated Services and Information). The permissions model in OSCAR is extremely powerful and can be used to create just about as many roles and permission sets as possible. Providers belong to programs (as staff) where they have a specific role. Each program takes place at a facility. Each role has a description (for example, "doctor", "nurse", "social worker", and so on) and a set of attached global permissions. The permissions are written in a format that makes them very easy to understand: "read nurse notes" may be a permission that a doctor role may have, but the nurse role may not have the "read doctor notes" permission. The name of the role that the current provider has is checked against its list of permissions for a match with the action that they are trying to perform. For example, a provider attempting to read a doctor's notes would cause "read doctor notes" to be checked for each and every note written by a doctor. One of the problems is that People using Oscar in other languages also have to write the permissions in English. Another implementation is a role based access. Providers are assigned one or many roles (for example, "doctor", "nurse", "admin", and so on). They can be assigned as many roles as necessary—the roles' permissions stack on top of each other. OSCAR's Integrator component is a separate web application that independent OSCAR instances use to exchange patient, program and provider information over a secure link.

**Providers Access:**
If you are a provider then a provider view will be seen instead of a normal one for the users which is the appointment screen. But access also varies with the role of the provider. e.g. a nurse would not have the same features as an admin. Both can go to the administrative page but they will not have the same options. In Oscar only one or two individuals are allowed to
access the administration area and only those can see all the records of all the people and can create new providers and users and can assign roles to the providers. They can access the provider list, admin special search, provider ids, schedule, they can change appointment access control, schedule options, add groups etc. It is their responsibility to educate new users to the importance of protecting their login and password in OSCAR. This is important to protect the integrity of the system and the patient’s confidential records. They can block and unblock accounts too. They also have to manage the billing process. They can change the Base URL. They are responsible for taking backups of OSCAR files and generate reports. Here are some more functions Update Drugref: This function updates Drugref 2/3, Add New Queue: add a new Queue for categorizing and directing uploaded documents, Clinic Admin: Clinic Admin and Address

This link gives you opportunity to alter the Clinic or Agency address when pulled elsewhere in Oscar. Manage Satellite: Sites Manage Satellite Sites lets you set up providers grouped by separate physical locations. Referral Doctors: Admin Lists of referral doctors, Demographic Export: Export patient files in the CMS standard for other Oscar, and compatible, programs

Demographic Import: Import demographic allows you to import demographic and clinical data in a standard format from another Oscar or other compatible EMR, Merge Demographics: Demographic merge allows you to merge demographics where patient data has been entered into several separate charts by mistake, Patient Provider Update: Update Provider

This allows you to update a patient’s provider, en-mass when there is turnover of clinic staff and many more. Hence this kind of power cannot be given to too many people.

**Patients Access:**
The patients are responsible in the OSCAR only for appointment screen. They have access only to the appointment screen and the functionalities complementary to it such as billing.

**Availability of Patient Portal, PHR, ePrescribing**

**Patient Portal and PHR:**
OSCAR also interfaces with an interoperable PHR, a Patient Controlled Health Record called MyOSCAR which is the patient’s portal to the OSCAR. This was jointly developed with Harvard and MIT.MYOSCAR (http://myoscar.org/myoscar). MyOSCAR is a secure, private on-line health record. User has total control as to who can put information into the user’s record and with whom the user can share this information. He can also: communicate securely with your healthcare team request copies of the records from their doctor which may be lab results, prescription profile, or scanned documents manage prescription renewals make appointment requests or cancel appointments access reliable health information about topics of interest to you with MyOSCAR you can do all this from the privacy of your home. All you need is access to a computer, a MyOSCAR account, and the Internet.

**ePriscription:**
OscarRX 9.12 available which allows you add prescription. You can prescribe a drug by searching in the database or prescribe a custom drug or add special instructions or discontinue it.

**Support for CDS:**
It has rudimentary CDS. It does Clinical Decision support using an evidence based clinical decision support system. The evidence based decision making uses the experience of the patients as a research literature to guide the decision making. This system has been proved to be very effective.

**Installation of OSCAR McMaster 12.1**

Steps:
1. Install prerequisites.
   a) Ubuntu
   b) Java
   c) Tomcat
   d) MySQL

2. Download the OSCAR deb package from Sourceforge
   

   Now simply run the package (try double clicking or the following command) and follow the instructions (Note this should uninstall any previous deb installed OSCAR, but will preserve its database)

   ```
   sudo dpkg -i oscar_emr12.1.1-49general356.deb
   ```

   The deb will provide feedback as to what it is doing however if you want more verbose details open another terminal window and invoke

   ```
   tail -f /usr/share/OscarMcmaster/Oscar12.1.1install.log
   ```

3. New installs get the following install screens
   Start by providing the MySQL password you assigned when you installed MySQL.

   A screen will present and ask for your localization preference for billing. If you are in BC pick BC and if you are in neither province (or not in Canada) then pick other. The default is Ontario.

   Optionally supply your particular phone prefix as a convenience for when patients are added to the demographics database. This prefix can always be easily overwritten by front office staff as needed but saves keystrokes for them and reduces the opportunity for data entry errors.

   In urban areas we suggest that you just add the area code, in rural areas it might be longer

   The fax support that this option configures is global and includes signature and some other support. However to take full advantage of these features

   1. You must use Firefox or Chromium for the newest Rich Text Letter to work (with signature and fax support)
2. You must have OSCAR configured as per Security Hardening including and specifically https on port 8443 for printing or faxing (the deb takes care of this for most cases)

3. Also Lan or Internet Faxing needs to be configured on your server for facsimile transmissions

4. While you can mouse sign the results are poorer than if you sign on another device (track pad etc.)

This window will give you the option to populate the database with a demo patient with which to develop familiarity with OSCAR. As mentioned you can remove the demo data later with the

/usr/share/OscarMcmaster/undemo.sql

The Eyeform version is the most developed of the specialist interfaces possible with OSCAR. The default is no.

Most people will want access to the intake form.

The new contacts interface is optional.

These settings, and more, can be changed manually through editing

/usr/share/tomcat6/Oscar12_1.properties

Oscar/Tomcat need to restart whenever the properties file is modified to have the settings take effect.
OpenMRS

The mission of OpenMRS is to improve healthcare delivery in resource-constrained environments by coordinating a global community that creates a robust, scalable, user-driven, open source medical record system platform. Initially, OpenMRS was built to support HIV care in a couple of settings (in Kenya and Rwanda). However it was clear that other diseases had also to be tracked. Therefore, it was designed to be a generic electronic medical record (EMR) system that can support the care of patients, gathering observations, encounters, notes, and other data from the healthcare system and rendering those in summaries, reports, and data views that would improve the effectiveness of the people using the system.

OpenMRS is programmed in Java and the core application works through a web-browser. Hibernate is used as an interface layer to the database. Tomcat is used as the web application server. The back end database is currently in MySQL. The system creates XML schemas for form design. Form design and form data entry is currently done in Microsoft Infopath, HTML, or XForms. When form data entered is submitted, it is converted into a HL7 message before going into the database.

Database Platform and Accessibility:
It uses MySQL database. Typical access to the database server through the terminal is possible.

APIs:

OpenMRS is intentionally built with multiple layers in mind. One of the layers is a java API that can be used in other projects just as easily as it can be inside of OpenMRS. The API allows developers to interact with the complex OpenMRS Data Model (explained later) with common Java objects. This provides greater data integrity as well as a simple to use approach. Except the JAVA API that is provided, one can use Web Services to access the same functionalities and get access to that with any other programming language.

APIs provided:
- JAVA APIs
- REST web services:
  - if an OpenMRS instance is running the Webservices.REST module, other programs (and languages) can connect to retrieve and post certain information to an OpenMRS database.
- SOAP web services (initial phase of development):
  - JaxWS Web Services: allows JaxWs webservice to be written for OpenMRS
  - JaxWS API Web Services : adds actual methods that mimic the OpenMRS API

Data Model:

At the heart of any enterprise electronic medical record system lies a robust, explicit representation of how care information is stored. The structure of this data model dictates the scalability and flexibility of a system. The OpenMRS collaborative therefore invests continuous effort into shaping the OpenMRS data model using knowledge and experience gleaned from practical experiences from the Regenstrief Institute, Partners in Health, and all of our developmental partners around the world. The core of this data model addresses the who, what,
when, where, and how of medical encounters. This model is divided into ten basic domains. Domains:

- **Concept**: Concepts are defined and used to support strongly coded data throughout the system.
- **Encounter**: Contains the meta-data regarding health care providers’ interventions with a patient.
- **Form**: Essentially, the user interface description for the various components.
- **Observation**: This is where the actual health care information is stored. There are many observations per Encounter.
- **Order**: Things/actions that have been requested to occur (Labs, prescriptions, etc.).
- **Patient**: Basic information about patients in this system.
- **User**: Basic information about the people that use this system.
- **Person**: Basic information about person in the system.
- **Business**: Non-medical data used to administrate openmrs.
- **Groups/Workflow**: Workflows and Cohort data.

**Patient vs. Provider Access**

Patient access is provided in OpenMRS. It can be done if the administrator attaches a User account to the Patient record in the system. Providers can view, add and modify patient records. OpenMRS uses privileges and roles to control access to data within the system. Privileges define what can or cannot be done in the system (e.g., *Edit People* or *Add User*) while Roles are used to group privileges into more manageable grouping. To make the system easier to manage, roles can contain other roles as well as privileges. Roles inherit all privileges that exist within the child roles.

Role-Based Access Control (RBAC), which OpenMRS implements, provides a reasonably robust mechanism for restricting access to information; however, OpenMRS does not yet have a mechanism for restricting access to specific data (e.g., a clinician is allowed to access the record of patient X, but not patient Y; or, a clinician is permitted to access a patient’s data except for specific lab results).

**Availability of Patient Portal, PHR, ePrescribing:**

- **Patient Portal and PHR**: Yes, but still under construction. The PHR module (i.e. personalhr) allows you to create a patient controlled health records application. It gives the patient the full control of his/her own health records and other personal information, and enables him/her to share part or all of those information to any one in his/her social network such as a family member, a doctor, or any other caregiver he/she trusts.
- **ePrescribing**: Yes, it also supports export to the open source IDart pharmacy system through the PharmacyDataExport module. Also, any pharmacy system could use the exported data as long as corresponding Mirth channel is created.

**Support for CDS:**

No support for CDS currently. However, this is a very early work-in-progress with the intention of eliciting a full design spec for the decision support infrastructure of OpenMRS.

**Installation Instructions:**

Requirements:
• Firefox browser
• Java runtime environment (version 7 preferred)
• Tomcat webserver
• MySQL database server

Full instructions for multiple operating systems and requirements can be found here: https://wiki.openmrs.org/display/docs/Installing+OpenMRS

STEPS (Windows):
Firefox:
• Download the latest stable release of Firefox and run installation program
• Accept the license agreement
• Select standard mode for installation or Custom, make sure the application is installed on the default directory. C:\Program Files (x86)\Mozilla Firefox

JRE (ver 7):
• Minimum version required is Java 6 although it is recommended to install Java 7
• To download the application use the this link http://www.oracle.com/technetwork/java/javase/downloads/index.html
• Accept the license agreement and make sure you download the correct file for your windows version, whether its x32 or x64 bit.
• It is recommended to download the executable file (.exe) for a simpler run
• Execute the downloaded file, accept the license agreement and follow the instructions in the wizard, installing in default installation directories.

Tomcat:
• Java must be installed before installing Apache Tomcat.
• There are issues with versions of Tomcat later than 6.0.29 that have yet to be resolved. Installation through a package manager is not recommended as this will likely install a later version
• With OpenMRS 1.8 it is necessary to increase the Tomcat Permgen memory after installing Tomcat but before deploying OpenMRS. More information: https://wiki.openmrs.org/display/docs/Troubleshooting+Memory+Errors
• Download Tomcat 6.0.29. You can use the exe version, which installs Tomcat as a service or the zip archive.
• Execute the file and install running the default settings o Accept the license agreement
• Accept default destination folder
• Accept HTTP/1.1 Connector Port 8080
• Set Administrator login (username/password)
• Accept the Java directory detected
• Select Install Tomcat# After installation is complete you will need to change users roles by following this directory on your windows explorer
  • C:\Program Files\Apache Software Foundation\Tomcat 6.0\conf
  • Locate the file "tomcat-users.xml" and try to open it.
    ■ Most likely your operating system will fail to detect the application that opens the file so make a right-click on the file then select down the menu Open With > Notepad
You will notice that a text editor will show up then locate this character set
<tomcat-users> The character set is located on line 18 of the file.

- Open the Tomcat users file (e.g. C:\Program Files\Apache Software Foundation\Tomcat 6.0\conf\tomcat-users.xml) in a text editor.
- Create a new user called admin with the roles admin, manager and manager-gui. This file should be protected so you will need to open it as Administrator (right-click on your text editor and select "Run as administrator")
- <user name="admin" password="XXXXXXX" roles="tomcat,admin,manager,manager-gui"/>

MySQL:
- Download the latest MySQL installer using this link
  https://dev.mysql.com/downloads/installer/
- Run the install program (.msi)
- Accept the license agreement
- When given the option to update installer please do so
- Under Feature Selection select Full Installation Setup and select the right Architecture for your computer (32-bit / 64-bit)
- Click next and you will be shown a list of applications that you need in order to meet the requirements for installing all services. Make sure you satisfy all the requirements, if not, please install the missing applications on your machine.
- On the next configuration options select “Developer Machine”
- Leave all other settings to default
- Enter a username and password. Note: These will be the credentials for the user with root privileges. Do Not Forget the Password
- Click next and finish the installation.

Deploy OpenMRS:
- With OpenMRS 1.8 it is necessary to increase the Tomcat Permgen memory before deploying OpenMRS. More information:
  https://wiki.openmrs.org/display/docs/Troubleshooting+Memory+Errors
- Ensure that Tomcat is started by checking to see if icon in the tray is green
- Download the latest stable release of OpenMRS http://openmrs.org/download/
- Navigate to http://localhost:8080/manager/html and enter your Tomcat administrator credentials (username and password chosen when installing Tomcat)
- In the Tomcat Web Application Manager, enter the location of the downloaded “openmrs.war” file to deploy
  - The deployment could take some time while the file is copied to the folder C:\Program Files\Apache Software Foundation\Tomcat 5.5\webapps and decompressed
  - Note that the OpenMRS.war file is most easily downloaded with Mozilla Firefox. Internet Explorer tries to open the file as a Zip file.
- At the end of this process, the web page will refresh and /openmrs should be displayed under Applications. Apache Tomcat should also start the application (Running = True; and in Commands, Stop is underlined)

Start using OpenMRS:
After you have finished configuring OpenMRS, RELOAD the application in Tomcat Manager.

Open http://localhost:8080/openmrs. You will see a login page. If you're using the OpenMRS standalone package, the page is at http://localhost:8080/openmrs-standalone.

- You will need to log in initially using the username and password you specified in Step 6 - Configuring OpenMRS, sub step 4. If you did not specify a username and password, try the default username admin and password test (both are in lowercase).
- Alternatively, while Tomcat is running you can start OpenMRS by entering http://localhost:8080/openmrs/login.htm (assuming 8080 is your port number for Tomcat; insert the appropriate port number if it is not 8080). For OpenMRS standalone package, you can start OpenMRS by entering http://localhost:8080/openmrs-standalone/login.htm.

Troubleshooting: https://wiki.openmrs.org/display/docs/Troubleshooting+Installation
Configuration: https://wiki.openmrs.org/display/docs/Step+6++Configure+OpenMRS
**FreeMED**

FreeMED is an opensource Electronic Medical Record (EMR) system based on Linux, Apache, MySQL and PHP. It could also be seen as a GPL-licensed EMR and Practice Management (PM) system for medical providers that runs in any web browser and can be utilized in many languages. It provides an XML-RPC backend, which allows complex data structures to be transmitted, processed and returned, and it provides multiple export and import formats. It also provides reporting and other features that will be discussed later on. The project started in 1999 by Jeffrey Buchbinder in Willimantic, Connecticut and is still being used and updated throughout the country.

**Database Platform and Accessibility:**

The database platform is MySQL and the databases are accessible through the terminal. The web-based application can import databases through the website and can export the ones that already exist.

**APIs:**

As said before, FreeMED is a web-based application. It utilizes modules to store and represent medical data. Such modules are virtually connected through a relational database, which has the advantage of being able to add and take off modules without the need to reprogram its interface every time you want to modify a feature.

FreeMED has implemented several APIs, which can be seen at: [http://developer.freemedsoftware.org/manual/c36.html](http://developer.freemedsoftware.org/manual/c36.html). The APIs are under FreeMED::namespace so that it is easier to differentiate them from other APIs. Some of the APIs are the following:

- boolean freemed::check_access_for_facility(int patient_number, [int user_id]): Mainly used to determine if a home office has access to a medical record.

- boolean freemed::user_flag(int flag): Determines if flag is set for the current user. For example, if the USER_ADMIN flag is set, then the user has administrator privileges and if the USER_DATABASE flag is not set then this means that the user does not have access to modify the contents of the database.

Some of the main features that FreeMED contains are the following:

- **Billing system:** FreeMED utilizes the REMITT billing system ([http://remitt.org/](http://remitt.org/)). Such system is primarily utilized for preparing and submitting medical billing data. It can be used on any EMR system which implements its own API. In the case of FreeMED, it has the billing system as a self-contained J2EE application and it supports features such as remote configuration management, SFTP push support and eligibility verification.

- **Push Notifications:** This means that the user interface updates itself whenever the data updates itself, therefore, you will always view updated data without the need to refresh the page.
• Drug sample tracking: Track drug sample lots in the event of a recall or other tracking purposes.

• Access Control: Grant the right access to staff and providers.

• Scheduler: Drag and drop module. It has the appointment templating feature and all of the modern scheduler trimmings.

• Reports: Utilizes a standard reporting engine to facilitate for providers the task of doing reports.

• Webcam patient image support: Take a headshot of a patient for identification purposes with any web browser and a webcam.

• Data normalization: Since the system relies on normalized data fields against industry standard data sets, the data of patients is portable.

Patient vs. Provider Access:
The system has been downloaded over 80,000 times and is utilized by a range of clinical settings around the world that go from large government hospitals to small private practices. The application is an EMR, therefore, it can only be used by medical providers (patients can’t use this web-based application). The system utilizes role-based access control in order to provide the right access to medical providers. Such permissions can be established in the website if the user has enough privileges to do so.

Availability of Patient Portal, PHR, ePrescribing:
• Patient Portal and PHRs:
The system doesn’t have these features since it does not support patient access (the application is focused on providers since it is an EMR).

• Support for CDS:
The creators of the system were in the talks with the people of EGADSS, which is an open source tool designed to work along EMRs to help physicians provide high quality care, in order to integrate their Clinical Decision Support System (CDS) with FreeMED. Nevertheless, this hasn’t been established in the system yet, therefore, there is no CDS on FreeMED.

• ePrescribing:
The FreeMED people are planning to implement this feature using some of the standards that the National Council for Prescription Drug Programs (NCPDP) provide. But, the script of the standards they want to utilize to do this are outdated and they are waiting to obtain updated scripts from the NCPDP.

Installation Instructions (for Mac):
1) Download VMWare Fusion at:
http://e5.onthehub.com/WebStore/OfferingDetails.aspx?o=c66286e6-3c87-e211-87e9-f04da23e67f4&ws=ab065a5b-58ca-de11-baeb-0030487d8896&vsro=8
Note that you have to have an engineering account at UCONN in order to access the download link. You can download VMWare through an external source but you have to pay for it.

2) Download FreeMED live cd at:
http://sourceforge.net/projects/freemed/files/FreeMED%20Live%20CD/

3) Once you have VMWare Fusion, open it and do the following steps: File -> New -> Continue without disc -> Choose a disc or disc image… -> Select the FreeMED cd you downloaded -> Continue -> Continue again -> Finish.

4) Now FreeMED should start installing itself in VMWare. After this, we can see that the cd installed Kubuntu so that it can run FreeMED.

5) Open a browser in Kubuntu and type ‘http://localhost/freemed/’.

6) Now, you should be able to login to the demo site utilizing the username “demo” and the password “demo” or the username: “admin” and the password “admin”.

7) Also, you can access the database through the terminal by entering the following command: mysql -u root. In here, you can view all the databases that the system contains along with their respective tables and also, modify their contents.

Note: We tried to install the EMR in three different ways than the one we discussed in this document, but they were unsuccessful. We provide the links below of the installation guides in case someone wants to try to make them work in a future.

Installation of Vmware-FreeMED image in Mac OSX:
https://docs.google.com/a/uconn.edu/file/d/0B0se5mbYKcOSYjRlN2QwNWMtM2IxMi00ZWQ0LWFhZTQtYWYwOWM1YzZhZmU1/edit?hl=en

Installation of FreeMED v.085 on Ubuntu 14.04:
https://docs.google.com/a/uconn.edu/document/d/1DLW-hqteeFl_Vtt5yvbDhdOOCsLR-8Wq107cwfOUf50/edit?hl=en

Installation of FreeMED 0.9.x series on Debian 6:
https://github.com/freemed/freemed/wiki/Installation
**WorldVistA**

WorldVistA is a derivative of the Veteran Administration’s EHR called VistA. WorldVistA adds bugfixes and important features for private sector adoption, such as support for pediatrics, while remaining compatible with the public VA release of VistA through the Freedom of Information Act commonly referred to as FOIA VistA. WorldVistA is programmed in the MUMPS language and uses the built-in MUMPS database for raw data storage. Structure is provided using a collection of record manipulation MUMPS routines collectively called VA FileMan. The client used to connect to a WorldVistA database is called CPRS (Computerized Patient Record System), which is written in Delphi. While the WorldVistA server can run under Linux or Windows (the purchase of a commercial license is necessary for use in Windows unless a VM is used), the CPRS client is only available for Windows. WINE can be used to run CPRS in a Linux environment if necessary. There is a community of WorldVistA users that call themselves Hardhats that have a Google Group and regularly discuss WorldVistA, improvements that can be made, and help new users set up and troubleshoot new installations.

Astronaut installer is a popular installer package that offers an easier setup for WorldVistA. However, this assessment was performed on the vanilla version of WorldVistA.

**Installation Instructions For Debian-Derived Linux (Tested in Ubuntu 10.04 - 13.04)**


- Run the script as sudo, read and listen to the instructions in every prompt - some instructions need to run on their own without any user input or the install will fail

- The script will need to install xinetd and

- Default values for all prompts are OK

- Ignore any errors during MUMPS compiling, errors are printed but they relate to a different platform

- When the script has finished running CPRS (the client) still will not be able to connect, in order to fix:
  - Open sudo /etc/xinetd.d/WVEHR-gui-’(port number)’
    where [port number] is the port you chose to run world vista on
  - Change the line: port = '[port number]' to port = [port number]
    (The single quotes prevent xinetd from recognizing the script - do this fix if netstat -a | grep [port number] returns nothing after installation)
- run `sudo chmod +x /home/[username]/EHR/WVEHR-gui`
  where [username] was the user created/chosen to run World Vista in the script

- Run `/etc/init.d/xinetd restart`

- `netstat -a | grep 9297` should show the port running with tcp as LISTEN

- Download [http://opensourcevista.net:8888/NancysVistAServer/WVEHR-2.0-Install-GTM/HowToFixLackOfHomeDevice.doc](http://opensourcevista.net:8888/NancysVistAServer/WVEHR-2.0-Install-GTM/HowToFixLackOfHomeDevice.doc) and follow the instructions if this error was output near the end of installation


- Start at step 22 to download and use CPRS to connect to the server

Database Platform and Accessibility

World Vista uses a language called MUMPS to manage its database. MUMPS is both a programming language and a hierarchically structured key-value database engine. The two MUMPS implementations used by WorldVista are InterSystems Caché, a proprietary Windows MUMPS implementation used by the VA in VistA, and GT.M, an open source implementation that can be used in Linux in WorldVista and OpenVista. The MUMPS coding for WorldVista exists in separate modules called packages, and each package performs some function or subroutine for the system. Examples are medical data management, patient/provider management, and billing management.

MUMPS has a hierarchical database structure rather than a relational database structure. Variables marked with a caret (^) are considered “global” and get stored in the database. VA FileMan is a MUMPS application that organizes this information into a schema. FileMan was part of the original VistA project from which WorldVista is derived and performs needed services such as adding/editing data, allowing the retrieval of records within the database, and forming a consistent structure for each record.

Interfaces for EWD, a method by which web browsers may connect to a WorldVista installation, are being developed by members of the WorldVista community. However, these changes are being made for inclusion in the Astronaut flavor of WorldVista. EWD provides an interface for Java, PHP, and ASP, and also allows direct access to the database.

APIs
WorldVistA has published APIs for all of its functions including patient update, records of visits, medications, allergies, etc. These APIs are accessed through WorldVistA’s RPC functionality. Examples of functionality can be seen in WorldVistA’s CPRS client. MUMPS as a language does not have a concept of encapsulation so any user with access to the MUMPS database has full access to any routine available within the database. The public API is more of a convention whereby the code writers provide assurance that the function listed is meant as an entry point into the database and that they will attempt to keep the specific call compatible with the interface specified. Likewise, users of the API should generally not attempt to bypass the call in the API and call the “private” function calls because although there is nothing that specifically disallows them from doing so, changes in the system may break their code.

APIs for import/export of data are also provided. The tools use ASCII data exchange in delimited and fixed-length formats. Data must be in a format specified by the user, and these formats are denoted in templates. There is a list of default templates provided including templates for: Word, Excel, Lotus 1-2-3, and Oracle. Support for the CCR/CCD is being integrated into WorldVistA.

There is an implementation that supports Harvard’s SMART platform. The support was not added by the WorldVistA team to the main WorldVistA distribution, but was created in a joint effort by members of the WorldVistA team, M/Gateway Developments Ltd, and E-cology Corp. The software is freely available at https://trac.opensourcevista.net/browser/smart and a ready to run version can be downloaded at http://www.fourthwatchsoftware.com/ as a dEWDrop virtual machine. This machine comes preconfigured with WorldVistA and the SMART platform.

**Patient vs. Provider Access**

Providers can access an instance of WorldVistA through CPRS, the client application, if they have an account. Interfaces for EWD and m2web are being developed so that access can occur through a web browser rather than the CPRS client. Providers are assigned roles and those roles have their access limited according to HIPAA standards.

MedSphere is in the process of working on a patient portal through which a patient can access their data for OpenVista, but patient access in WorldVistA is not supported. If the patient’s medical record is stored by the VA, the VA has a website which allows users to view their medical records and download them for personal use.

**Availability of Patient Portal, PHR, ePrescribing**

Patient portals are not provided in the base WorldVistA package. The VA maintains a website where members can view and edit their health records, or move them into other EHR systems. MedSphere is in the process of adding a patient portal to their base OpenVista package, but it is unclear when this will be finished.

ePrescribing functionality is not present within WorldVistA at this time. It is listed as a Clinical Enhancement on the WorldVistA wiki.
Support for CDS

According to the VA VistA monograph, clinical decision support systems are outside the scope of VistA, and therefore are missing in WorldVistA. The WorldVistA website makes references to a system called EBMeDS that can be integrated with WorldVistA if a provider makes the effort to set up the queries necessary on the database and pass the data to their clinical decision support system, but there is no native support for CDS. The Socratic Grid Project is another possible CDS for WorldVistA.