Open Source Telemedicine

Sana Android Client and OpenMRS

Winter 2012 IAP

Images of phone in this presentation © Google. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/fairuse.
Course Agenda

- General introduction to the platform
- How to get started developing
- Platform software components
- Topics related to the software
- Guest speakers.
- Case studies and related topics
- Hands on projects
- Plans for the future
# Daily Schedule

<table>
<thead>
<tr>
<th>Day</th>
<th>Agenda</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan 26th</td>
<td>Introduction to the platform Development Environment</td>
</tr>
<tr>
<td></td>
<td>Guest Speaker: Dr. Trishan Panch, Clinical perspectives on software tools</td>
</tr>
<tr>
<td>Jan 27th</td>
<td>Sana Android Client Project Discussion</td>
</tr>
<tr>
<td></td>
<td>Guest Speaker: Members of hearing loss study from UFRN, Rio Grande Norte, Brazil</td>
</tr>
<tr>
<td>Jan 30th</td>
<td>Data Layer: Client and Permanent Data Stores, OpenMRS</td>
</tr>
<tr>
<td></td>
<td>Guest Speaker: Chris Moses, Cardiovascular disease study, Philippines</td>
</tr>
<tr>
<td>Jan 31st</td>
<td>Dispatch layer</td>
</tr>
<tr>
<td></td>
<td>Guest Speaker: Rich Fletcher, Medical Device Integration</td>
</tr>
<tr>
<td></td>
<td>Hacking Day</td>
</tr>
<tr>
<td>Feb 1st</td>
<td>Systems and Scalability: Platform based approach</td>
</tr>
<tr>
<td></td>
<td>Hacking day</td>
</tr>
<tr>
<td>Feb 2nd</td>
<td>Hacking day</td>
</tr>
<tr>
<td>Feb 3rd</td>
<td>Wrap up</td>
</tr>
<tr>
<td></td>
<td>Participant project presentations</td>
</tr>
</tbody>
</table>
Open Source Telemedicine
Introduction
Agenda

• Fundamental problem
• Goals
• Challenges
• Why Sana?
• Solutions
Fundamental Problem

Given a shortage of trained clinical specialists coupled with limitations on connectivity and a myriad of local challenges, how do we effectively project medical decision support and expertise into remote areas to capture and store data for future use as well as provide follow up instructions for care?
General Goals

- Measurable outcomes.
- High availability.
- Cost Efficiency.
- Interoperability.
- Customizable.
Challenges

1. Physical barriers
   Physical access may be difficult.
2. Technical barriers
   Connectivity can be poor or intermittent.
3. Cultural and language barriers
   Users will expect compliance with local and cultural norms.
4. Technical skill barriers
   User experience can limit adoption
5. Financial Barriers
   Cost of adoption may be prohibitive
Assumptions

- Technology can provide a path to addressing a number of the challenges.
- Technology alone will not solve the problems.
- No one group has all of the necessary skill and expertise.
- Collaborative innovation is necessary to address the wide range of local issues that face healthcare access and delivery.
- Education of health providers as well as those managing healthcare systems is critical to long term success.
Our Approach
Addressing Challenges

● **Physical barriers**
Fundamentally, mobile networks provide increased connectivity into areas that were previously more difficult to access.

● **Technical barriers**
Given poor or intermittent connectivity, mobile networks alone are not a perfect solution. We employ a combination of techniques which react to local conditions and use a retry-on-fail approach to maintain high availability.
Integrates with other 3<sup>rd</sup> party Android apps.
Addressing Challenges

● Cultural and language barriers
Components that can be localized. Education and information resources can be modified to adapt to local norms.

● Technical skill barriers
Provide end users with an intuitive user interface. Use a reasonably simple document format for any that may need to be edited by end users.

● Financial Barriers
All components are freely available. Components are compatible with low-grade hardware to the greatest extent possible.
Sana Tech Solution

Provide a turn-key, open source, platform consisting of a mobile client, network transport layer, and permanent data store for collecting and moving data from remote areas to centralized clinical specialists and back.
Sana Solution

Workflow
1. Data collection instructions are provided to a Clinical Health Worker, CHW, on a mobile device.
2. CHW collects data following those instructions.
3. Collected data is sent to a centralized medical records system.
4. Data is reviewed by a clinical specialist.
5. Notification of diagnosis and follow up instructions are returned to CHW.
Open Source Telemedicine Platform Overview
Sana Platform

Client

Dispatcher

Data Store
Sana Platform

- Mobile Client System (Android)
- Mobile Dispatch Server
- Medical Records (Java, OpenMRS)

Python
Django
Sana Mobile Client

Dispatcher
Network Layer
Procedure
Engine
Android APIs

Client
UI Layer

Data Store
SQLite
Mobile Dispatch Server

- Mobile Dispatch Server
- Dispatcher
  - Web Server
  - SMS Client
  - Email Client
- Client
  - Http Request
- Data Store
  - Caching
  - MySQL
OpenMRS Modules

Dispatcher
Sana Module
Tomcat
Webapp

Client
Http Request

Data Store
MySQL
Design Patterns
Sana Platform
Operating System

Ubuntu OS + Sana Packages

Sana provides prebuilt packages for Ubuntu – a Linux based OS
Sana Platform

Why Open Source?

1. Reduces financial barriers.
2. Freedom to modify and adapt to our needs.
3. Open standards.
Open Source Telemedicine Workflow
Sana Platform

Workflow Implementation

**Supports** text and binary data

**Customizable** decision trees

**Integrated** with digital medical records system

**Optimized** data transfer for poor coverage areas

**CHW** collects data and uploads over GPRS, 3G, or Wifi

**Clinical Specialist** reviews encounter through OpenMRS and sends notification back to the mobile client
Sana Platform

Android Client

1. CHW Instructions are form based – XML

2. GUI layer interprets instructions into more user friendly format – buttons, etc.

3. Integrates with full data capture capabilities of device.

4. Integrates device capabilities for patient education and CHW instruction.

5. Local data storage.

6. Agile and secure network layer.
Sana Platform

Sana XML Forms

1. Step wise data collection using markup language.
2. Collection of specific data types.
3. Conditional branching logic.
4. Required constraint.
5. Plugins
Example:

```xml
<Procedure title="Example" author="Sana">
  <Page>
    <Element type="RADIO" id="1" concept="TEST SELECT ONE"
      question="Demonstrate selecting a single value"
      answer="1" choices="1,2,3,4" required="true"/>
  </Page>
  <Page>
    <ShowIf>
      <Criteria type="EQUALS" id="1" value="4"/>
    </ShowIf>
    <Element type="PLUGIN" id="2" concept="SUBJECT VIDEO"
      question="Video media file for patient encounter"
      action="android.media.action.VIDEO_CAPTURE"
      mimeType="video/*" answer="" />
  </Page>
</Procedure>
```
Sana Platform

Mobile Dispatch Server, MDS

1. Connection Based REST API-JSON formatted HTTP requests.

2. Supports packetization

3. Communicates with EMR

4. Sends SMS and email notifications

5. Typically configured to run on same server as EMR
Sana Platform

Encounter upload example

POST mds encounter url
upload url
Request params:
Params:
  - Encounter identifier
  - Encounter identifier
  - Client id
  - Client id
  - Authorization
  - Authorization
  - Collected Data
  - Collected Data

Recieves “OK” from MDS

POST OpenMRS
Request

Responds “OK”
to MDS
Sana Platform

Dispatch Server
Packetization Implementation

1. Client sends binary data as smaller file POST requests.
2. Client algorithm will reduce packet size based on network conditions.
3. Mobile client will retry on fail.
OpenMRS

- Free, Open Source EMR
- Active deployments globally
- Deployed as Tomcat web application
- Extensive plug-in system
Sana OpenMRS Modules

• **Sana Queue Module**
  - Supports downloads from MDS
  - Stores uploaded encounter data directly into the OpenMRS tables
  - Stores a queue table in the OpenMRS database, linked to OpenMRS
  - Encounter table
  - Provides views of items waiting to be reviewed as a queue

• **Media Viewer Module**
  - Flash/JS based visualizer for rich media content

• **REST Module**
  - RESTful access to the OpenMRS data model
Sana Platform Scalability

- Target baseline is to be compatible with OpenMRS performance.
- Platform capable of handling thousands of patient encounters/reviews daily including significant binary content.
- Additional content/plugins can be added at run time.
Open Source Telemedicine Roadmap
Going forward

Generic architecture applicable to any areas that may suffer from differential access to skilled personnel and provides analysis.

1. Generalized analytics API
2. Common language.
3. User land tools.
4. Ready made systems.
Going Forward

Issues and areas of research. Examples

1. Ease of Use
   Examples:
   - WYSIWYG Form builders
   - Automatic client content updates

2. Administration
   Examples:
   - Content Package Management
     Central repositories for content and diagnostic tools
   - Network diagnostic tools
   - Remote client management

3. Network Architecture
   Examples:
   - Increased support for varied network topologies
   - Load balancing

4. UI independent analytics libraries
   - jSana, pySana
Going forward

How to get involved

1. Join the mailing lists.
2. Attend the development workshops and meetings.
3. Attend the general meetings.
4. Let us know about yourself.
5. Check out the code.
6. Have fun and remember that work you do for Sana does have a direct impact.
Additional Information

Sana Wiki

Additional details are available on the Sana Wiki:

http://sana.mit.edu/wiki

Includes installation and configuration for developers and end users.