OpenMRS Sync 2.0 module development, implementations and maintenance

Submitted by Jakub Sławiński (SolDevelo) on January 18, 2018 - 1:19am
Last revised by Web Producer on June 21, 2018 - 3:09pm

Proposal Status: Awarded--Fully Funded

Executive summary

OpenMRS is the leading open-source EMR made to support healthcare in low and middle-income countries.
The original sync project was the brainchild of Christian Allen in 2007 while he was working for Partners in Health. PIH had multiple sites that were not continuously connected by internet across the Rwandan countryside. Transporting paper forms 4-5 hours on a daily or even weekly basis was not working. His vision of a bidirectional asynchronous database synchronization technique lived and developed in a branch or two for several years. In mid 2009 it was clear that sync was not going to be a part of core code going forward. OpenMRS community made changes to core that were incorporated into the OpenMRS 1.5 release to enable sync to be a module. Next, the code was ported from the branch into the sync module to allow for maximum flexibility and ease of use. The most recent Sync module (1.1) has been released in 2014 and can be used only with OpenMRS v1.x.
Sync module has some serious drawbacks:

- It is not working with recent OpenMRS v2.x
- It is no longer maintained
- It is highly coupled with the database
- It does not have a reliable merging functionality
- It does not have any secure error handling and transaction retry mechanisms
- It is hard to extend and does not support syncing with systems other than OpenMRS

We have information that many existing OpenMRS implementations are desperate for a usable and reliable sync solution. The lack of such module is blocking some organizations from upgrading to OpenMRS v2.x. We realised that there is a huge demand for creating a new Sync 2.0 module that will not have the mentioned drawbacks and will work with the newest OpenMRS.
Bahmni contributed to the initial idea of Sync 2.0 module by sharing their findings, experience with implementations and suggesting some of the solutions. Moreover, they expect to leverage the output of that proposed work as the basis of an equivalent Bahmni feature, as described in their Digital Square proposal.

Why posting a new proposal instead of joining an existing OpenMRS one?
Sync 2.0 module is a well-defined and strictly limited investment, both from the time and budget perspective. We strongly believe that smaller projects with clearly defined scope and high impact are the most effective ways of spending money into global goods.

Consortium team

Consortium includes the following organizations:

- SolDevelo - dynamic IT company focused on delivering high-quality software and innovative solutions. SolDevelo started contributing to OpenMRS in 2013 and is consistently supporting this fantastic community since then. We are performing such activities within our Social Impact Program for many years now.

  - Responsible for: technical project management, software development, trainings, helping implementers, contact with community and partners, coordinating volunteers effort, public relations
  - Point of contact: Jakub Sławiński - SolDevelo CTO

The above consortium is open for any collaborators that could strengthen the impact of this project. We would definitely welcome any interested OpenMRS implementers that could benefit from using Sync 2.0 in their current/future installations. Moreover, we can also coordinate our work with the OpenMRS community to make sure that we will achieve the common goals effectively together.

Project description

This proposal is about building a new Sync 2.0 module that will be a replacement for openmrs legacy Sync module. The new module will be based on Fast Healthcare Interoperability Resources (FHIR), a flexible next generation standards framework created by HL7, and atom feeds as a communication channel between different nodes. This new solution will be secure, reliable and provide OpenMRS a completely new level of interoperability. After implementing FHIR it will be possible to synchronize data between OpenMRS and other systems, that implement the FHIR specification.

The new module will have the following characteristic:

- It will be based on FHIR and atom feed modules
- It will work with the current versions of OpenMRS (v2.x)
- It will be a usable and reliable sync solution ready to be used with different systems than OpenMRS, for example a Master Patient Index (MPI) could be used as the central repository of patients for OpenMRS installations using Sync
- It will automatically try to deal with errors
- It will handle potential loss of internet connection for extended periods of time, common at rural sites
- It will keep an audit log with the possibility to retry failed transactions through a web interface for the administrator
- It will be built with in a way that will allow extending it in the future

We will work closely with interested implementing organizations and will build the new synchronization solution that will meet their requirements.

One of the known drawbacks of the legacy Sync module is error handling. Sync 2.0 will be transparent to administrators and will be providing visibility into its operations, as well as show what was synced. Sync 2.0 will be more reliable and allow administrators a way to troubleshoot more easily into how it operates.

It will be possible to configure Sync to store information about the failures it faced, as well as provide the ability to retry a failed transaction. It will be also possible to store details about all messages in order to provide a more comprehensive audit of Sync operations.

Sync will provide a UI for each Child that gives an overview of the transactions and failures that happen along the way, combined with the ability to retry operations that have failed. Each failure will have enough information attached to it, so that an admin can act on it. and resolve the potential issue.

This project is not only about writing a new piece of software. We will also have to perform the following tasks:

1. Write a comprehensive documentation along with the step-by-step tutorials to use the new module.
2. Advertise the new module and search for first implementers that would like to use it.
3. Train the first users and make sure that the adoptions are as smooth as possible.
4. Work with the implementers, troubleshoot any potential issues, fix found bugs and implement new improvement requests.
5. Support and maintain module for the specified period. We will have to answer any questions, help with issues, advice with configuration and update documentation.

We strongly believe that providing a new Sync solution for OpenMRS will greatly strengthen it and will benefit the whole community. The first reactions to our initiative were very positive and we already found several interested implementers that are willing to test new module as soon as possible.

Current status of the OMRS Sync 2.0 development

I. Sync solution is built from the FHIR, Atomfeed and Sync 2.0 modules and here is the status of each one of them:

A. FHIR

1. FHIR Client and server endpoints
2. Currently conversion from/to FHIR is supported for the following entities in OpenMRS:

a. Allergy(not yet implemented in Sync 2.0)

b. Appointment(not yet implemented in Sync 2.0)

c. Condition(not yet implemented in Sync 2.0)

d. Location(not yet implemented in Sync 2.0)

e. Observation(not yet implemented in Sync 2.0)

f. Patient

g. Person

h. Related Person(not yet implemented in Sync 2.0)

i. Client error handling

B. Atomfeed

1. Event based feed propagation (AtomFeed entry represents an event in the system)
2. Supported Atomfeed configuration
3. Default Atomfeed configuration added
4. Reusable Atomfeed client (Child can use it's own FHIR server/REST representations)

C. Sync 2.0

1. Supported Sync 2.0 configuration in JSON
2. Sync resources using FHIR or REST API (implemented entities: Patient, Location, Privilege, AuditLog). Send to Parent through FHIR and REST (CRUD operations)
3. Possibility to configure which objects should be synchronized

D. Error handling

1. AuditLog - logging synchronized patients
2. Success/Failure push and pull operations
3. Parents can see child audit logs
4. The ability to trigger a retry for every failed operation.

II. What can be done:

A. FHIR/Sync module extensions to support additional entities

1. Support generation of FHIR representations for objects not included in the current implementation. Listed here: https://wiki.openmrs.org/display/projects/FHIR+Resource+List+for+Sync
2. Design FHIR mappings for OpenMRS forms and implement their conversion
3. Transfer non-FHIR translatable entities through REST API

B. More control over the synchronization process
   1. The parent should have possibility to decide which children could push data to it
   2. Allow manual synchronization of selected records. Add custom widgets to OpenMRS screens that will allow this - for example from the patient page
   3. Support for handling different catchment areas - allow to configure what is synced where through configuration

C. Conflict resolution
   1. Proposals of conflict resolution: minimum solution is to show on both parent and child that there is a conflict, in the future give possibility for implementer to configure the solution
   2. Add extension points which allow implementers to add custom behavior with minimal effort
   3. Integrate conflict resolution with the current Sync 2.0 audit log interface.

D. Support for OpenMRS 1.X
   1. Implementation request from Mozambique - support OpenMRS version 1 entities in older deployments with older versions
   2. Support legacy UI - make Sync work with implementations that do not use the Reference UI app

E. Support for manual synchronization by exposing an import/export mechanism through the UI

F. Deploy Sync 2.0 with partner implementations for beta testing and react to their feedback - fix bugs, extend features, make sure Sync 2.0 is ready for a production environment

G. Extensibility support to work with central repositories that are not OpenMRS - a Master Patient Index, an SHR server and so on
   1. Extend the configuration to give fine grained control that allows integration with different systems for given entities.
   2. Create extension points that allow implementers control over the sync process through code - example is adding an extension to send CDA documents using XDS.b, instead of FHIR/REST based communication.
   3. Make it possible to replace the Atomfeed change tracking mechanism with different ones - example is making use of FHIR bundles

H. Create documentation & a user guide for implementers, describing how to adop & extend Sync 2.0

Use Cases, User Stories and Activities

Usage models, coming in from real requests and usage of Sync 1.0:

1. Two sync networks, one with 25 children and the other with 20. These networks sync our modules, configurations, and html forms as well as patient data. Currently, patient data is synced up to the central server and then that same data is synced down to each server. The servers are connected to each other either through the internet or a cellular modem-based APN connecting the rural servers.
2. Standalone sites - sync is a means of aggregating the patient data from multiple sites into a single instance.
3. Child server only connected to internet once a week with large amount of items waiting to be synced from the child server to the parent server.
4. Frontline workers gathering medical data remotely in rural areas, synchronizing all data to central sites once they are online.

User stories:

- As a health worker I want all the data I enter into OpenMRS synchronized to the central server in the background
- As an OpenMRS administrator I want to control how often data is synchronized between OpenMRS servers and which data is synchronized

- As a health worker I want to manually synchronize a patient record right away, so that it is available in the central server
- As an OpenMRS implementer I want Sync 2.0 to synchronize my data with an SHR/MPI with minimal development effort required on our side
- As an OpenMRS implementer I want to extend Sync 2.0 to synchronize data according to the requirements of my implementation
- As an administrator I want visibility into the data synchronization between sites through a means of an audit log
- As an administrator I wish to resolve synchronization conflicts easily through the user interface, so that I do not have to manually access the database
- As a health worker I want my data to be synchronized even if my site has lost internet connection, so that all of my data is in sync even in case of connectivity issues
- As an administrator I wish to use Sync 2.0 to synchronize OpenMRS manually transferring the data on a pendrive, so that I can synchronize sites with very bad internet access
- As an OpenMRS implementer I wish Sync 2.0 to work with OpenMRS 1.X, so that I don't have to update my implementation to the latest version
- As an OpenMRS implementer I wish the Sync 2.0 UI to work with the legacy OpenMRS, so that I don't have to adapt the Reference Application

Digital Health Technologies

OpenMRS Sync 2.0 is heavily based on FHIR module and it also uses REST and AtomFeeds as a communication channel for notifications on what needs to be synchronized.
FHIR (Fast Healthcare Interoperability Resources) is a standard for exchanging healthcare information electronically. It is the primary method of data transfer between the Child and Parent nodes.
REST(Representational state transfer) is an alternative method of data transfer between the Child and the Parent. Application is designed to optionally change transfer data method between REST and FHIR

Atom feeds are used to publish events representing resources to be synced.

Similarly to other OpenMRS modules, Sync 2.0 has been developed using Java.

Community Feedback

Sync 2.0 is an important feature for OpenMRS implementations as For the past several years, there was a continual call for a workable synchronization solution. First version of Sync had some drawbacks which were a reason to create a new product.

Companies/Institutions interested in implement this new synchronization solution are:

- eSaude, Mozambique
- FGH Mozambique, Mozambique
- UgandaEMR, Uganda
- Partners in Health, Haiti
- Global Brigades

A self-assessment on the Global Good Maturity Model

Since this proposal is about creating an OpenMRS module, we can provide a link to Global Good Maturity Model prepared by OpenMRS community in their proposal:
https://open-proposals.ucsf.edu/digital-square/notice-b/proposal/14295
Global Good Maturity Self Assessment
https://docs.google.com/spreadsheets/d/1SU1Ngn7nxLRurNTxmzm_Oiv6vTM81OQITbxHawaIT7k

Workplan, Project Deliverables & Schedule

Current plan:

Below are the man day estimations for the tasks in the project.

Task	**Estimation (development)**
FHIR/Sync module extensions to support additional entities	20 days
More control over the synchronization process	15 days
Conflict resolution	20 days
Support for OpenMRS 1.X	20 days
Support for manual synchronization by exposing an import/export mechanism through the UI	5 days
Extensibility support to work with central repositories that are not OpenMRS - a Master Patient Index, an SHR server and so on	15 days
Deploy Sync 2.0 with partner implementations for beta testing and react to their feedback - fix bugs, extend features, make sure Sync 2.0 is ready for a production environment	20 days

Team composition

For this proposal, following team composition is proposed:

- 2 FTE developers
- 0.5 FTE Tester/QA
- 0.25 FTE Project Manager

This team composition is assumed in the RACI Matrix below, and the Gantt chart timeline.

RACI Matrix

The RACI Matrix includes the roles on the project team, as well as the role of the OpenMRS Community, which will be regularly consulted and updated during the development process.

Task\Role	Developers	QA	Project Manager	Ope Com
Development of features for OpenMRS modules	Responsible	Informed	Accountable	Cons
Review of code submissions	Responsible/Accountable		Informed	Cons
Test Infrastructure management	Responsible	Consulted	Accountable	Cons
Testing in the test environment	Consulted	Responsible	Accountable	Infor
Documentation and user guide	Consulted	Responsible	Accountable	Infor
Working with implementations to identify features and potential issues and desired improvements	Informed	Responsible	Accountable	Infor
Fixing potential issues coming in from implementations	Responsible	Consulted	Accountable	Infor
Translating implementation feature requests into upcoming work items	Consulted	Consulted	Responsible/Accountable	Cons
Collaborating with the community to adopt Sync 2.0 for beta testing	Informed	Responsible	Responsible/Accountable	Cons

Timeline

The timeline below assumes the already mentioned team composition. Notes:

- Work with implementations will be ongoing during the lifecycle of the project. Because of the nature of support, it is impossible to predict its exact impact on the development effort or ts timing, but it should require no more effort than estimated : 20 man days total.
- Work with implementations on beta testing should take off after additional entities are synchronized.
- Start & end dates are exemplary and can be shifted. (start assumed at 04.06.2018)

Deliverables:

- An extended and reorganized OpenMRS Sync 2.0 module
- FHIR module with changes adopted for Sync 2.0
- Atomfeed module with changes adopted for Sync 2.0
- Bi-Weekly demonstrations of the development progress for the OpenMRS community and all other stakeholders
- Comprehensive OpenMRS Sync 2.0 documentation and user guide
- Beta-testing reports based on the experience of implementers using Sync 2.0

Budget Narrative

Project budget consist only of labor costs. As described in estimates and team composition paragraphs, effort needed for completing the project will be 115 man-days of development, 30 man-days of testing/qa and 15 man-days of project management. More details can be found in the attached detailed budget.

We can briefly list the individuals along with their staff time:

- **Software Developer** - responsible for code development and documentation. We are requesting 100% salary support over 3 months
- **Software Developer** - responsible for code development and documentation. We are requesting 100% salary support over 3 months
- **Quality Assurance Engineer** - responsible for testing, documentation and support. We are requesting 50% salary support over 3 months
- **Project Manager** - responsible for project management, coordination and communication with stakeholders. We are requesting 25% salary support over 3 months.

Total Request from Digital Square $37,840.