Reporting Sprint

Topic: Reporting Sprint

Product Owner: Mike Seaton

Developer Lead: Rafal Korytkowski

Known Developers Assigned (amount of effort in hrs dedicated to this work if possible):

Date: 1/9 to 1/23

Participants

- Mike Seaton
- Rafal Korytkowski
- Andrea Patterson
- Daniel Kayiwa
- Wyclif Luyima

How To Participate
Add your name to the list on this wiki page (with any comments about your availability). If you want to join after the sprint has started just join the IRC channel mentioned above and say hello.

The general process:

1. New to OpenMRS sprints? Want help getting started? Join the IRC channel and say "???": I'd like to participate in the sprint!"
2. Checkout code:
    a. Fork https://github.com/openmrs/openmrs-module-reporting
    b. Clone your fork: git clone https://github.com/YOUR_USERNAME/openmrs-module-reporting.git
    c. Add upstream: git remote add upstream https://github.com/openmrs/openmrs-module-reporting.git
    d. Fetch all branches: git fetch --all
    e. Checkout the sprint branch: git checkout **sprint-201301**
3. Compile code:
    a. Run: mvn clean install
4. Setup OpenMRS. Get the standalone: http://sourceforge.net/projects/openmrs/files/releases/OpenMRS_1.9.2/openmrs-standalone-1.9.2.zip/download
    a. Unpack and start openmrs-standalone.jar. Choose the demo database.
    b. Optionally load **demo reports** using this sql. It can be loaded with mysql client. DB connection details can be found in openmrs-standalone-runtime.properties in the standalone directory.
    c. Go to http://localhost:8081/openmrs-standalone/admin/modules/module.list and upload a compiled reporting omod, which you can find in reporting/omod/target.
5. Pick a ticket from the available tickets in the top-left of the sprint dashboard page (listed below). Make sure it does not depend on a ticket that is incomplete. If you have any questions about the ticket, ask on the group chat
6. Create a local branch for your ticket (see also our HOWTO for git, use '**sprint-201301' instead of 'master'**)
    a. git checkout -b REPORT-123 sprint-201301
    b. git push origin REPORT-123
7. Do the ticket.
    a. Stage changes: git add -i
    b. Commit: git commit -m "REPORT-123: ticket title"
    c. Push changes: git push -f
8. Issue a pull request for the '**sprint-201301'** branch (see github help) or if you have push rights: whatever works for you! If you don't like pull requests, don't send them. Commit and push directly to the upstream repo. If you do like pull requests, fork the repo and send pull requests, but merge them right after. My favorite way is to work on the repo, but create local branches (without pushing them to the repo). Merge branches locally to the sprint branch and push to the repo.
9. If you push directly to the repo check if tests for the sprint branch in reporting are passing: https://ci.openmrs.org/browse/BUNDLED-REPORT0 (Favorite the branch to be notified about failures: Login into Bamboo, Choose Actions -> Favorite Branch)
10. Join the daily scrum to share your updates

Overview

There are 3 primary goals we would like to accomplish during this sprint:

- Enable all reporting metadata to be shareable across OpenMRS instances using the metadatasharing module: REPORT-464
- Enable reporting to expose it's relevent data definitions as Calculations via the Calculation module: REPORT-461
- Improve the reporting module documentation on the OpenMRS wiki and prepare an OpenMRS University screencast: REPORT-463

There is also a significant backlog of tickets, of all levels of complexity, suitable for new and experience developers to contribute.

Goals

Must Have:

- The ability to share all reporting metadata via metadata sharing
- The ability to expose patient data definitions and person data definitions as calculations

Should Have:

- Create a reporting module screencast for OpenMRS university
- Improve reporting module documentation organization and content

Could Have:

- Add a ScriptedCohortDefinition
- Any "Ready for Work" feature that is Ready for Work and is "Low" complexity
- Any "Ready for Work" ticket in the reporting module that is of type "Bug"

Won't Have:

- Everything else

Kickoff Meeting

Kickoff meeting Date: 2012/01/09 after the scrum meeting

(Meeting setup with known developers after the final design meeting, but prior to the start date):

Kickoff Meeting Checklist

1. Do we know where we are going? (Yes/No)
    a. Do we know the problem we are solving (*yes/no*).
    b. Do we have a complete backlog of items to complete this work? (yes/no)
    c. Do we know our scope and priorities? (yes/no)
    d. Have we defined success? (yes/no)
2. Do we know how to get there?
    a. Do we have any unknowns to be decided during the sprint? *If Yes what are they?*
    b. Do we know who is doing what on our project? (yes/no)
    c. Do we have a test to complete prior to completion beyond our normal submit process? (Yes/*No).*
    d. Do we have a high level architecture that is understood by the whole team? (This page fully completed will accomplish this) (Yes/No)
    e. Do we know the biggest constraint that is likely to inhibit our success? *(Yes and no) If no what is it?*
    f. Is this a part of a larger story or epic? (Yes/No) If Yes please link
3. Are we set up to succeed?
    a. Do we have the right people? *(yes*)
4. Have we cleared the decks of all other distractions?

Backlog

These are a combination of parent tickets with child tickets underneath to break up the work or tickets needed for completion of the work. These tickets should be made in as much detail as possible going into the design calls through use of the mailgroups and small group design meetings. Larger design calls will be used to answer questions these groups cannot or for the community to resolve.

All tickets should have in their description a DOD (Definition of Done) or what it should have accomplished

Tickets involved: https://tickets.openmrs.org/secure/RapidBoard.jspa?rapidView=15

Design

There should be multiple design meetings. The first starts off with a small group working with the leader to determine the high level scope of the project and then to break down into smaller pieces to eventually place as tickets. Once those meetings havehappened use of the community design meeting time should be used to refine the tickets and if possible assign them to who will be doing the work.

Risks (these should be resolved prior to or during the design meetings):

1.

Enter anything that is still questionable or worrisome that has not been answered: (open issues, potential concerns) Once a decision is made make sure that a summarized answer is included.

Example: Q. How will we handle issues with conflicting userID's? A. All ID's will have an added identifier that is unique.

Effort Accounted For: (At the end of the design call all tickets should have an estimated effort and that total should be balanced against the known available effort of the developers assigned) (Yes/No)

If not in balance in favor of success why and the action plan for remaining items

via IRC on the #openmrs channel on freenode.

Use this channel for **ALL** debugging and random questions having to do with the sprint. Please avoid direct messaging to personal contacts. If you have a question, someone else most likely does too, and our geographically distributed community benefitsfrom public group discussion.

During Project Notes

Add notes, comments, concerns, that occurred during the sprint that should be noted to help in the retrospective.

Post Project (Retrospective)

Did we complete tickets to a 100% DOD? (Yes/No) if no, have those tickets been assessed and placed for future work if needed?

What did we do well?

What could we do better?

What should we not do again?

Resources

- Kickoff meeting recording: During first part of Design call 1/9
- Rapid Board: https://tickets.openmrs.org/secure/RapidBoard.jspa?rapidView=15
- Repository Preview: https://github.com/openmrs/openmrs-module-reporting/tree/sprint-201301
- CI: https://ci.openmrs.org/browse/BUNDLED-REPORT0