Improvements to the HL7Query Module

Improvements to the HL7Query module as part of the GSOC 2013 program

Primary mentor	Unknown User (surangak)
Backup mentor	Rafal Korytkowski
Assigned to	Unknown User (ruwanego)

Abstract

This document contains a detailed overview of improvements to be added to the OpenMRS HL7Query module during the GSOC 2013 program.

Project Champions

Suranga Nath kasthurirathne

Objectives

The core objective of this project is to add new features which will ensure the transition of the OpenMRS HL7Query module from merely responding to user requests to an underlying service which listens to certain events, and generates pre-defined HL7 messages in response. The HL7 message generation feature already works as desired, we merely want to change the manner in which it is being used to generate responses.

The initial version of the HL7Query module behaved in a synchronous 'ask and receive' fashion. Users defined their own HL7 message structures, and then made specific requests to the module. The module created appropriate HL7 messages based on these requests, and returned them to the user.

However, based on later discussions, it seems that some users are not interested in a query feature as we developed.

Therefore, i'm proposing the following improvements to this module,

- Add features for the module to listen to specific events (creation / update / deletion)
- Add features for users to list specific implementations / sites to transmit a newly created hl7 message
- Allow users to define a new event to listen for, create an appopriate message once the event occurs, and send it to the specified receiver on the list
- Allow users to define whether to generate a message and store it, as opposed to outputting it as default

Use Case

The Key use case I have in mind for GSOC is tied to improving the HL7Query module so that it can be linked together with the OpenMRS event module and the NDC module to function jointly as a surveillance tool.

The OpenMRS event module listens to particular events, and reports them. The HL7Query module identifies these events, and translates them into HL7 messages, which are then sent to the NCD module. The NCD module can interpret the received message, use it to identify a specific medical condition, and act accordingly.

Design

I [Unknown User (surangak)] will list down the following steps / guidelines as a means of documenting my thoughts on how to build this feature.

Currently, the OpenMRS event module allows users to register for certain events (create / update etc.). My plan is to re-use this feature.

(Step 01) Create a webpage where users can select one or many events to 'follow'

OpenMRS users may have created a number of events to listen for using the OpenMRS event module. The HL7Query module will contain a web page which will list each of these event types. Users can select one or many of these events, which they want to process using HL7 messaging.

(Step 02) If a user selected to 'follow' an event using the HL7Query module, we will offer him a number of options on how to process the results.

For example, they can decide to,
1. Create the message in pipe delimited format, or hl7 delimited format.
2. Post the message to a specified user / server, or persist it in the database.

The above indicated that we should create a model object to represent a SenderProfile object, which will contain these data. Each of the events which we elect to follow must be assigned one (and only one) Sender Profile. The Sender profile will define what we intend to do with the hl7 message.

Based on the above, I feel that we need to add two different web pages into the module.

1) A web page to create / manage sender profiles
2) A web page to follow / unfollow OpenMRS events, and assign sender profiles to each.

However, this doesn't mean that we are doing away with the existing features. The query facility will still remain, and be fully supported.

Implementation process

- Rename/re-factor the module to indicate its new features

Instead of continuing to call the module HL7Query, we need to change it to something which indicates its new features. How about 'HL7Messaging module' ?

- Decide upon the best way to listen for particular events

The first thought to come to mind is the OpenMRS Events module, but we're open to alternative suggestions

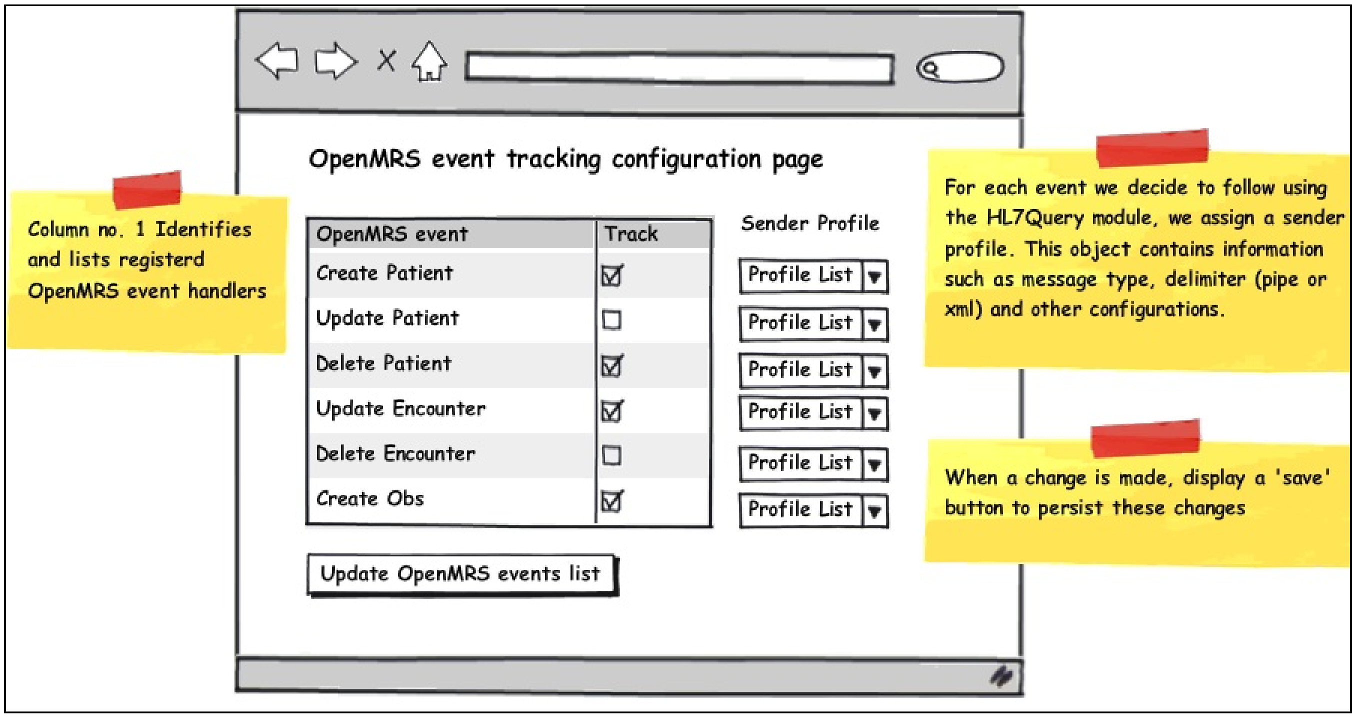
- Build new interfaces

Build the interfaces/ features which allow users to specify where to send the HL7 messages. This will include storing urls, ports, message structure preferred as well as user name / password management.

Sample Mockups

The following mock-ups provide an overview of expected page design and features,

Image A: HL7 event tracking configuration page used to identify which events to track



OpenMRS event tracking configuration page

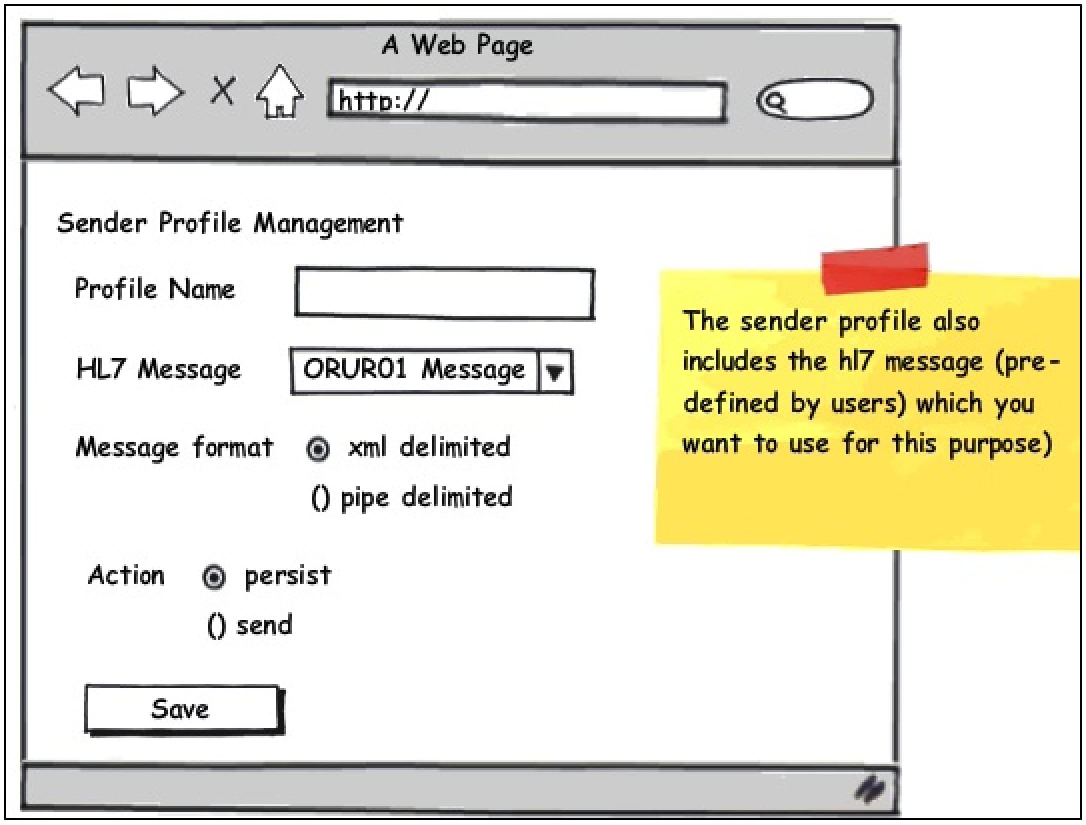
Column no. 1 Identifies and lists registerd OpenMRS event handlers

OpenMRS event	Track	Sender Profile
Create Patient	☑	Profile List ▼
Update Patient	☐	Profile List ▼
Delete Patient	☑	Profile List ▼
Update Encounter	☑	Profile List ▼
Delete Encounter	☐	Profile List ▼
Create Obs	☑	Profile List ▼

For each event we decide to follow using the HL7Query module, we assign a sender profile. This object contains information such as message type, delimiter (pipe or xml) and other configurations.

When a change is made, display a 'save' button to persist these changes

Image B: The creation of sender profiles used to manage message processing data

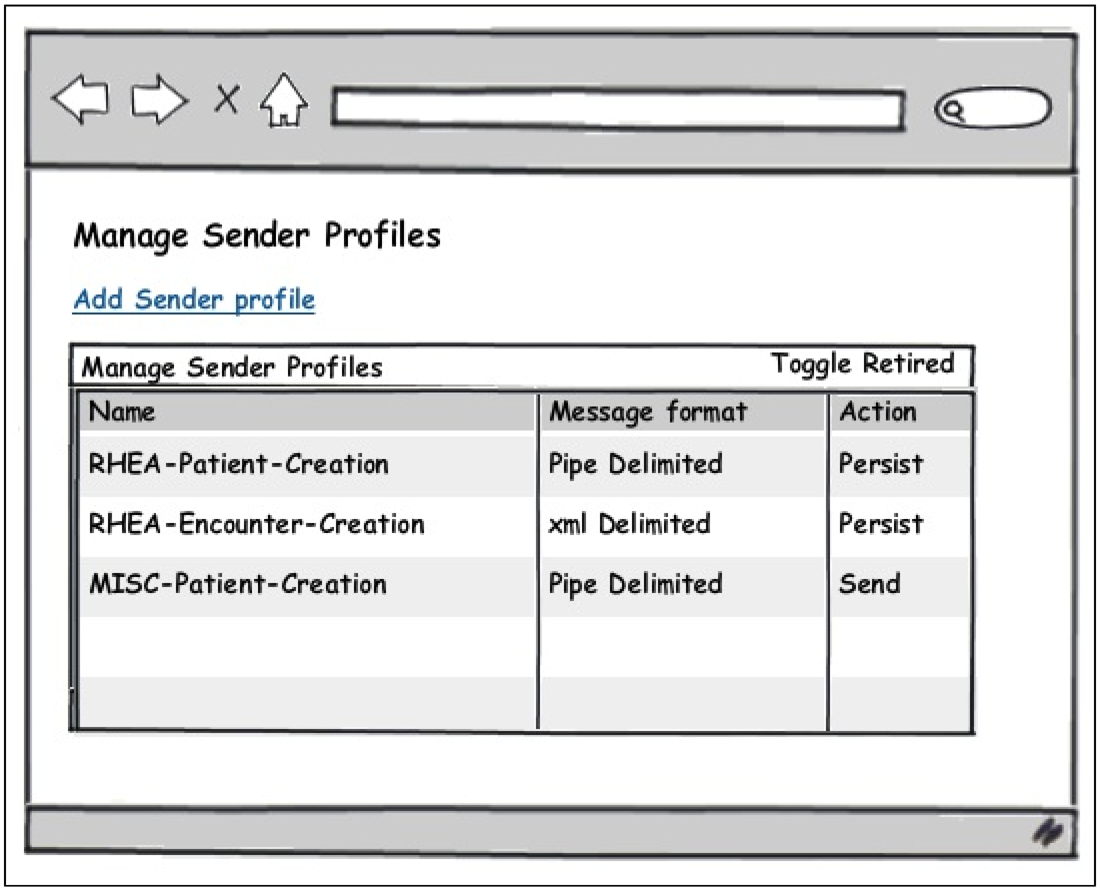


Sender Profile Management

Profile Name	
HL7 Message	ORUR01 Message
Message format	[✓ xml delimited] () pipe delimited
Action	[✓ persist] () send

The sender profile also includes the hl7 message (pre-defined by users) which you want to use for this purpose)

Image C: The management of Sender Profiles



Manage Sender Profiles

Add Sender profile

Manage Sender Profiles — Toggle Retired

Name	Message format	Action
RHEA-Patient-Creation	Pipe Delimited	Persist
RHEA-Encounter-Creation	xml Delimited	Persist
MISC-Patient-Creation	Pipe Delimited	Send

Extra Credit

Once these features have been implemented and tested, we may also work on adding out of the box support for ADTA08 messages.

- ADTA08 messages deal with Patient related information. Adding support for this message type would not be too difficult as we already have the completed ORUR01 messages to use as examples. Furthermore, some of the message components created for ORUR01 can also be re-used for ADTA08 messages.

***HL7 noobs can stop worrying, I can assure you that writing an ADT message will be no harder than writing xml or Json. You have the easy job. We (the experts) will have the far more difficult job of agreeing on the structure of the message we want you to write ***

Required skills

- Java / Spring / Hibernate
- Groovyscript (moderate knowledge is adequate, not compulsory)
- Soft kills (mandatory)

Resources

- The OpenMRS HL7Query module documentation can be found here
- For source code and omod, see here and here
- The OpenMRS Event module can be found here
- For source code and omod, see here and here
- For more information on ADT message structures, google.com is your friend :-)