PUBLIC HEALTH DATA STANDARDS CONSORTIUM

PHDSC

Promoting Standards Through Partnerships

USING MODEL-DRIVEN HEALTH TOOLS (MDHT)

FOR PUBLIC HEALTH CASE REPORTS

User Manual

Version 1

2012

Baltimore, Maryland

The ***Public Health Data Standards Consortium*** (PHDSC, The Consortium) is a national non-profit membership-based organization of federal, state and local health agencies, professional associations, academia, public and private sector organizations, international members, and individuals.

The Consortium is committed to bringing a common voice from the public health community to the national efforts of standardization of health information technology and population health data in order to improve individual and community health.

To fulfill its mission the Consortium:
Identifies priorities for new national standards for population health data;

Promotes integrating health-related data systems to meet the needs of public and private organizations, agencies, and individuals;

Participates in national and international efforts to standardize health-related information ;

Represents public health interests in standards development organizations, data content committees & standards harmonization entities; and

Educates the public health community about health information technology standards and the health information technology community about public health.

111 South Calvert Street
Baltimore, MD 21202
Phone: (410) 385-5201
Fax: (866) 637-6526
www.phdsc.org

Table of Contents

Table of Figures

Preparing for MDHT

Model Driven Health Tools (MDHT) is an open source application to promote interoperability in the development of a shared electronic repository of CDA templates and documents. It promotes shared artifacts between related healthcare standards, and supports the work to develop customized CDA-based document specifications for specific needs. Its other advantage is providing the framework that seamlessly integrates design, publication, and instance generation. The work involved in creating CDA template usually requires a cross project team consisted of epidemiologists, laboratorians, clinicians, standards developers, and information systems specialists. Specifically, it is advisable that the model development team comprises of the following skill areas:

- Programming interface background and experience with the Eclipse IDE
- Public health subject matter expertise
- Vocabulary/terminology experience to look up appropriate codes
- CDA knowledge/experience

To learn how to use MDHT, the Open Health Tools MDHT project site provides many resources including Guides and Tutorials, supplemental downloads, and MDHT Releases and Downloads. The MDHT documentation includes:

This Using MDHT to create PHCR guide does not replace any of the current MDHT guides and tutorials, and supplements them with information about the process for using MDHT to edit and create public health case reports (PHCR).

Installing the Eclipse software

Open Health Tools provides an MDHT Getting Started Guide that provides detailed steps for the installation of the most current release of the MDHT All-In-One windows software, and describes how to setup a new CDA-based model project. Please refer to that for the latest in-depth instructions on how to setup your environment for MDHT. The following provides a summary guideline on the process of installing and setting up MDHT for PHCR.

Windows

Download the all-in-one package to install from the MDHT project website at: https://www.projects.openhealthtools.org/sf/projects/mdht/)

You will need to unzip the file in a local directory. Make sure that you do not have spaces in any of the folder names for the location where the MDHT files are unpackaged. Eclipse may not work well with folder names that have spaces. An example of a good name is: ***c:/eclipse***

After unzipping the folder, Eclipse is ready to be used.

Mac

You will need to download the software and go through step-by-step instructions for complete install. Check the Open Health Tools website to see if there are any updated mac install instructions. At the time of printing, the most up-to-date instructions were located at: https://www.projects.openhealthtools.org/sf/discussion/do/listPosts/projects.mdht/discussion.forum1136.topc11540;jsessionid=2DB91C372FBF9456B07168CE12318946?pageSize=-1#post_post20216)

Once the software is extracted from the zip file into a folder, run the Eclipse software from the unzip folder. When it opens, you will see a welcome screen. Close out of the welcome screen such as the following, with 3 main zones:

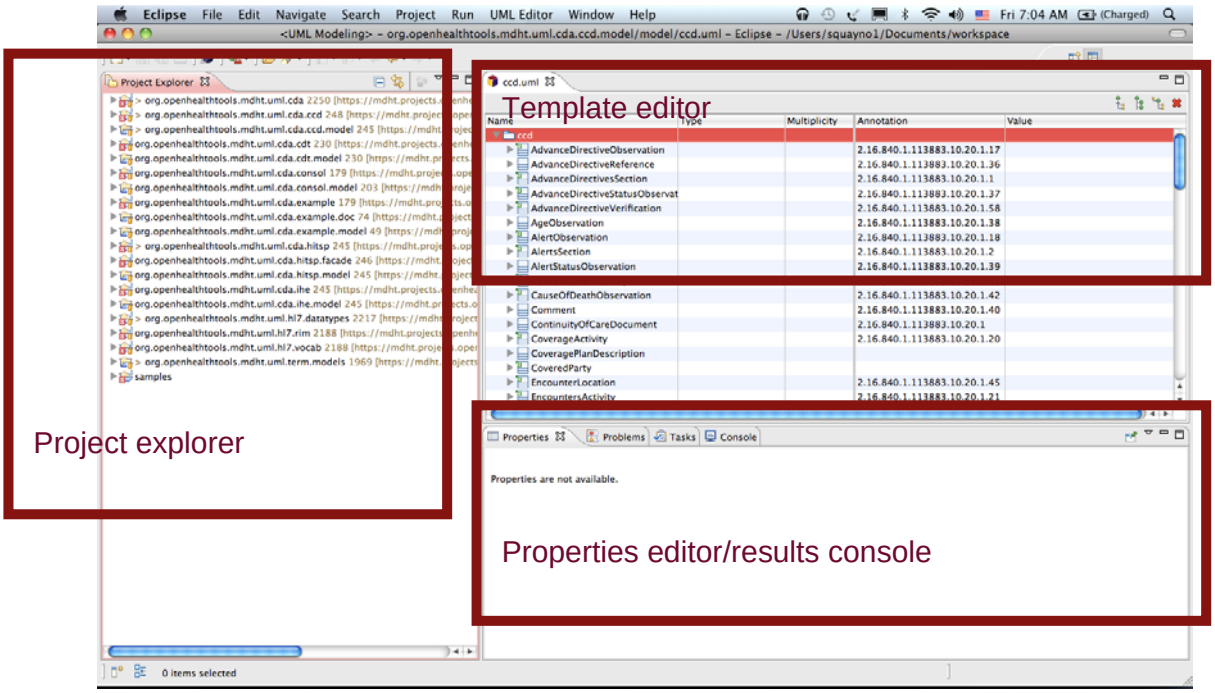

Figure 1 MDHT Screen Zones

The project explorer will list all project files that have been created in the Eclipse workspace.

The Template editor is used to add templates, associations, attributes, constraints etc. It also shows the summary values for the templates and attributes that are added to the project model

The properties editor and results console is the location where the constraints can be directly edited

Creating an Implementation Guide Model

To create a project, go to the project explorer and either a) select the  icon at the top or b) Select File->New->Project from the application menu:

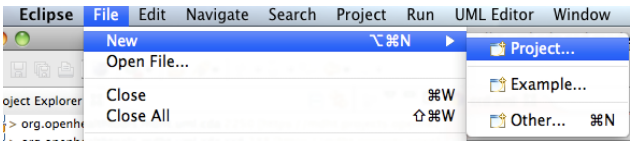

Figure 2 Create New Project

Next, you will need to select a wizard. You should see a folder ***Model Driven Health Tools*** with a wizard named, ***New CDA Model***. Select this wizard and click next.

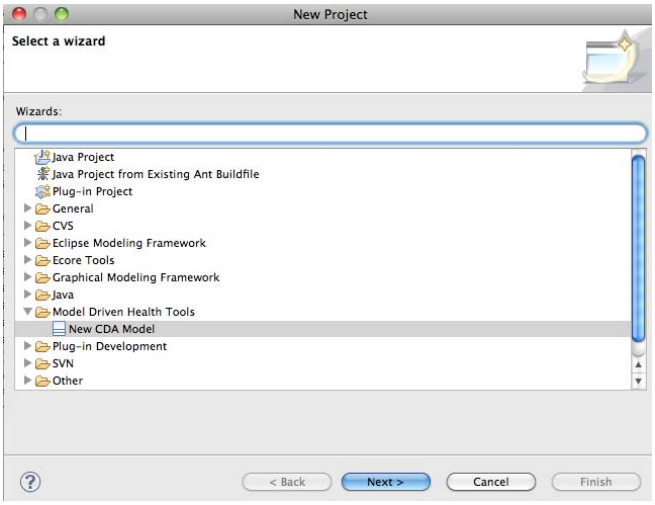

Figure 3 New CDA Model Wizard

On the next screen, enter the information for the implementation guide you will be working on. Note that you should not introduce any spaces in your model name. Use a name that is short and to the point. For example:

- Implementation Guide Namespace: ***PHCR***
- Document: ***PublicHealthCaseReport***
- CDA Document Conformance Reference: ***ccd::ContinuityOfCareDocument***
- (Consult with Sondra Renly (srrenly@us.ibm.com) for assigned names for the Public Health CDA project models.)

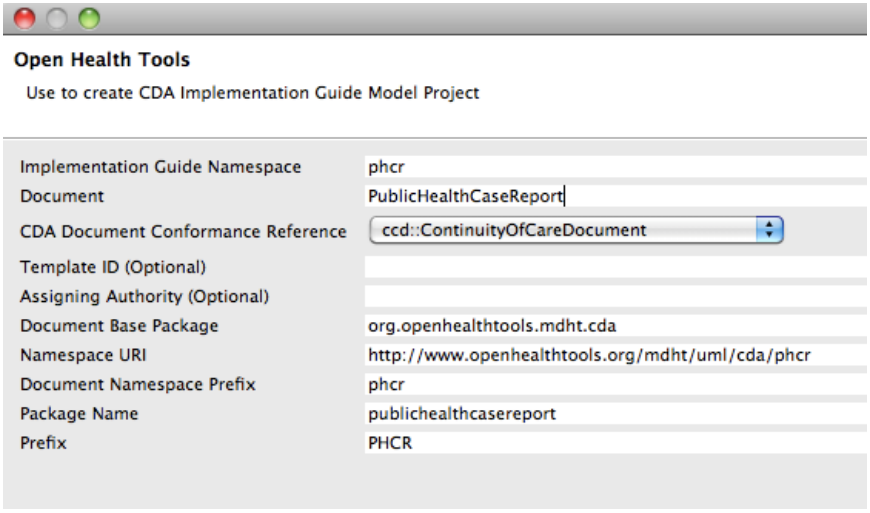

Figure 4 New Project Definition Screen

There will be a wait for a few minutes while the project is being created.

Once the application completes the task, you will have three folders created in your project explorer named:

- ***org.openhealthtools.mdht.uml.cda.<projectname.model>***, e.g. *phcr.model* - This is the UML model folder for the project you just created. This is where the model will be built - with the vocabulary details, constraints, etc.
- ***org.openhealthtools.mdht.uml.cda.<projectname.doc>***, e.g. *phcr.doc* - This uses the content from the .model project to automatically create a pdf implementation guide (uses a *dita* to transform the model into the implementation guide)
- ***org.openhealthtools.mdht.uml.cda.<projectname>***, e.g. *phcr* - This is the project that is used to generate the code to produce the project artifacts. The code is used to create the document, validate the document and process the document. Java code is generated and compiled in the first project file above.

Remove CDA builder - To temporarily turn off the building of the artifacts until the project is complete, "Right-click" on the ***<project name>.doc*** folder, and from the menu, select add/remove CDA builder. Without executing this step, every time the model is saved, the process will run the time-consuming ***dita*** transform. This can be turned off temporarily while work on the model continues, until the model is more fully developed, and then, the transform can be turned back on.

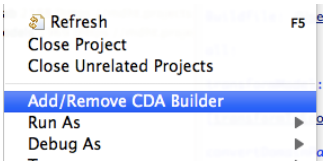

Figure 5 Add/Remove CDA Builder

You will be using the Implementation Guide for CDA Release 2: Public Health Case Reporting, Release 1 document to build these templates.

Building the Model

You will be building a sample model using the following sample constraints shown in Table 1 below from the implementation guide for public health case reporting. It provides specific sections that are required, or optional with specified constraints. The document will have a main document body, additional sections, with their own observations, etc.:

Table 1 - Sample statement excerpts from Implementation Guide[1]

Public Health Case Report CDA R2 template root [ClinicalDocument: templateId 2.16.840.1.113883.10.20.15]
SHALL contain [1..1] code/@code="55751-2" Public Health Case Report (CodeSystem: 2.16.840.1.113883.6.1 LOINC) STATIC (CONF:546)
SHALL contain [1..1] component/structuredBody (CONF:605), which
SHOULD contain [0..1] component (CONF:914) (specialized branch), which if present i. SHALL contain [1..1] PHCR Social history section (templateId:2.16.840.1.113883.10.20.15.2.22) (CONF:915) ... ...
PHCR Social History Section [section: templateId 2.16.840.1.113883.10.20.15.2.22]
Conforms to CCD Social history section Template (templateId: 2.16.840.1.113883.10.20.1.15). SHALL contain [1..1] code/@code="29762-2" Social history (CodeSystem: 2.16.840.1.113883.6.1 LOINC) STATIC (CONF:1894) SHALL contain [1..1] title="Social History" (CONF:1895) SHALL contain [1..1] text (CONF:1896) SHOULD contain [0..*] entry (CONF:1897) (specialized branch), which if present a. SHALL contain [1..1] @typeCode="DRIV" Is derived from (CodeSystem: 2.16.840.1.113883.5.1002 HL7ActRelationshipType) STATIC (CONF:1898) b. SHALL contain [1..1] Geotemporal history observation (templateId:2.16.840.1.113883.10.20.15.3.3) (CONF:1899) ... ...
Geotemporal history observation [observation: templateId 2.16.840.1.113883.10.20.15.3.3]
1. SHALL contain [1..1] @classCode="OBS" Observation (CodeSystem: 2.16.840.1.113883.5.6 HL7ActClass) STATIC (CONF:420) 2. SHALL contain [1..1] @moodCode="EVN" Event (CodeSystem: 2.16.840.1.113883.5.1001 HL7ActMood) STATIC (CONF:421) 3. SHALL contain [1..1] code/@code="55210-9" Geotemporal History (CodeSystem: 2.16.840.1.113883.6.1 LOINC) STATIC (CONF:422) ... ...

You will first create the document root template. From within the ***<project name>.model*** file, select the ***<project name>.uml*** file and double click on it to open the model

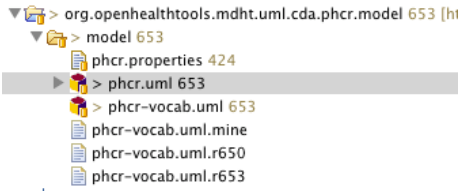

Figure 6 Create Document Root Template Part 1

[1] Note that a nested SHALL statement beneath a SHOULD becomes a SHOULD because the parent constraint is optional, meaning that if the parent component exists, then there shall be a social history section.

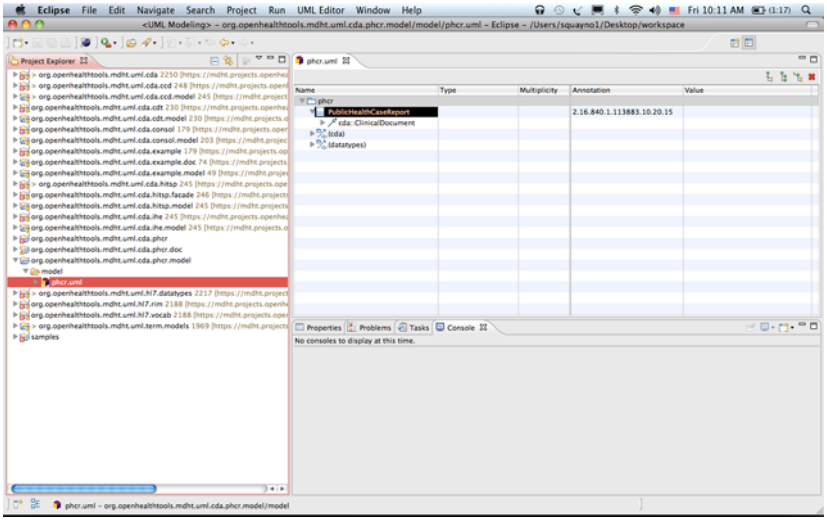

Figure 7 Create Document Root Template Part 2

Depending on which row you're on in the template explorer, the properties will change.

From the implementation guide, you will be implementing the document constraints shown in the Table 1 above. These are the ***SHALL*** and ***SHOULD*** statements in the implementation guide for the document body. Note that if a SHALL statement is nested under a SHOULD statement, it becomes a SHOULD, (optional). For each statement, you will either be adding a section, or entering attributes and constraints to build the public health case report.

From ***publichealthcasereport*** document in the template editor, right click to get the menu and then select <CDA Tools> and then <Open Template Editor>

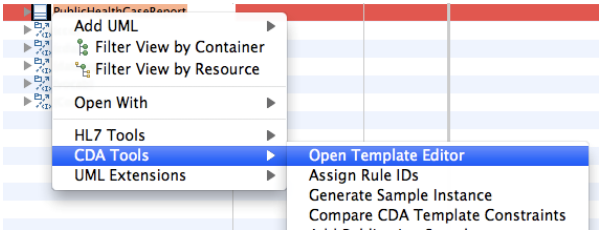

Figure 8 Open Template Editor

Because PHCR is a generalization from the CCD (Continuity of Care) document, there may be existing populated knowledge/requirements within the tool as a result. Select the code checkbox to change the code as specified in the implementation guide, and click ok

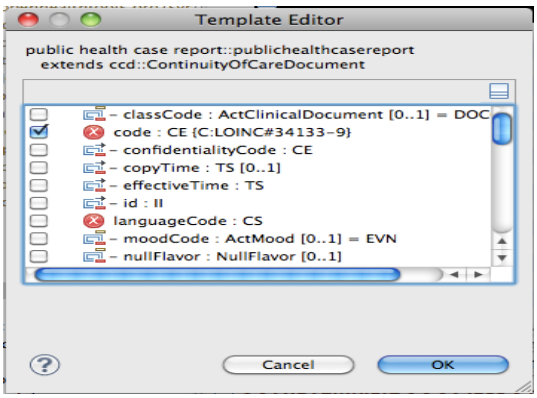

Figure 9 Template Editor

The default value is what the CCD would specify. To change this, make sure you are on the code row in the ***template editor***. Then, in the ***properties explorer*** below, under the ***Properties*** tab, select the **General** tab and change the ***Multiplicity*** to [1..1]

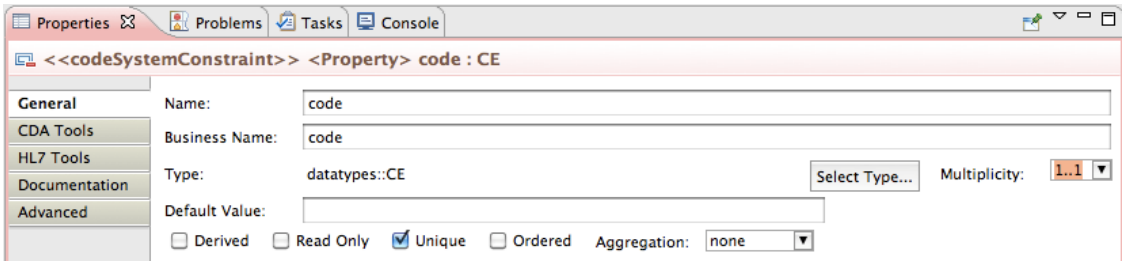

Figure 10 Properties Section

On the next tab - CDA Tools tab - is where you put vocabulary constraints. That is, the associated LOINC code and its display name from Table 1

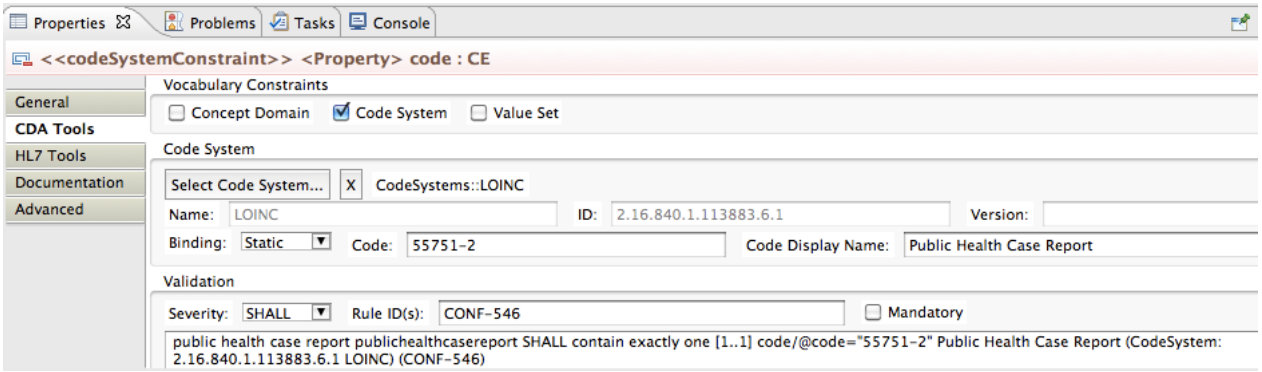

Figure 11 Edit Properties on CDA Tools Tab

There is a lot of content in CCD that is static - by using that as the basis for the model, we can hard code many of the rules/constraints for the user of the template, and then only populate or change with data that are dynamic

To save, hit Ctrl S or go to menu, select File and then Save. You will be asked about additional objects that have to be saved with the model. The application does a bit of work in the background while saving.

Creating Section Templates

Now you can create the Public Health Social History section. As Table 1 shows, the implementation guide states that there may be an optional PHCR Social history section, and if there is one, there are further constraints specified for the content of the section. You have created a document template and will now create an associated social history section template.

To create this section, go up to the root node, and from the right-click menu, create the template through the <Add CDA> and then select <Template>

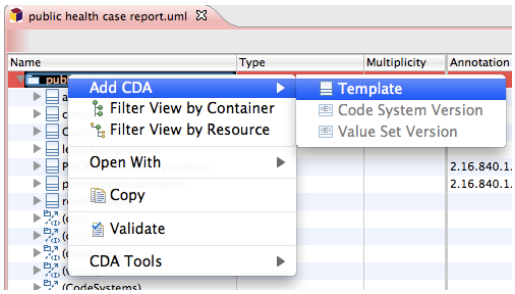

Figure 12 Add CDA Template or Section

You will get a New template screen

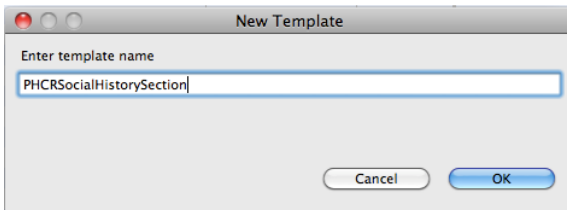

Figure 13 New Template Screen

After selecting OK, you will then get the Model Element Selection screen to allow you to select the CCD template that your new template will be inheriting. Since the social history section is a section that will be constrained beyond CCD Social History Section, the tool already has the CCD Social History Section built in, so we can reference that. Scroll through and select the matching SocialHistorySection and select OK

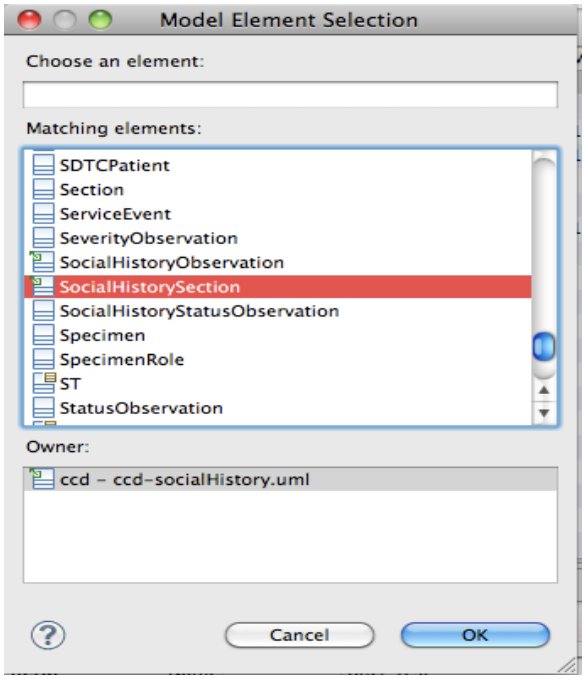

Figure 14 Model Element Selection Screen

There are additional constraints that we need to specify (e.g. in Table 1 above, there is a code and title constraint.) To implement the constraints, on the Template Editor screen that displays next, select the check boxes for code, and title to make sure they are required, then select OK.

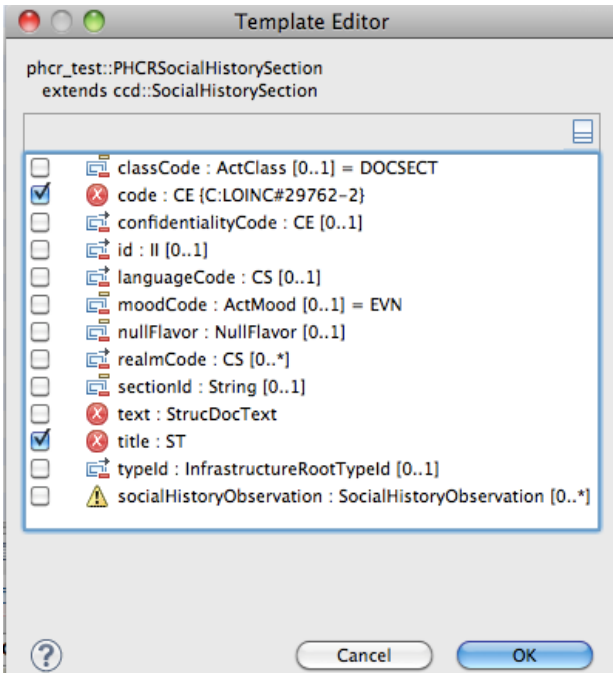

Figure 15 Template Editor Screen

Once they are added in the template editor, select each one, and enter the additional attributes - on the Properties tab in the properties console. Enter the section title for the name and business name on the properties tab. Then on CDA Tools, put in template ID, severity, etc.

So now you have a new CDA template that was just created - PHCRSocialHistorySection. You now need to be able to link this to the Public Health Case Report document, which you will do using an association.

Creating an association

Once you have created section templates for the document, you will have to link them to the document root. This will be through an ***association***, which is used to create a relationship between the root document, in this example the ***PublicHealthCaseReport***, and the new section that you just created, in this example the ***PHCRSocialHistorySection***

In MDHT, you can create a relationship between 2 objects using either an association represented by a solid-headed arrow ( ), or a generalization represented by an open-headed arrow ( ).

An association represents the relationship of one section to another, e.g. a PHCR Social History Section is associated to a Public Health Case Report

Whereas a generalization describes which template the current element is derived from, e.g. derived from the CDA or the CCD

Therefore, to link the PHCR Social History Section to the document root, you will be creating an association. To do this, select the root document, ***PublicHealthCaseReport*** in the template editor, and then on the right click menu, <Add UML> and then <Association>. Templates are listed in alphabetical order, so the document template might not be at the top of the templates list in the template editor.

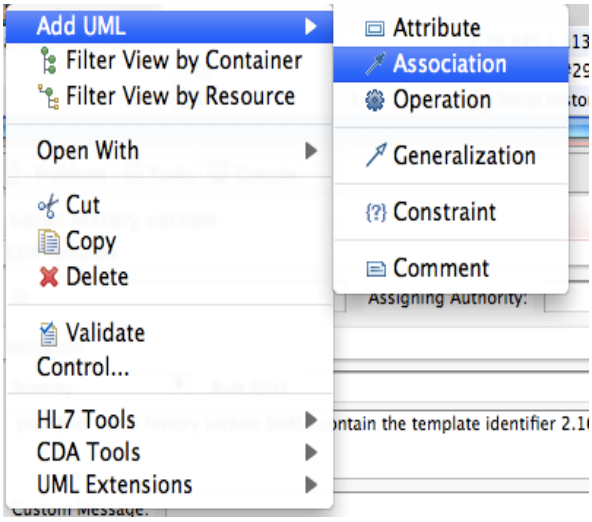

Figure 16 Add an Association

Then in the Association Selection screen, scroll through and select the new ***PHCRSocialHistorySection***. Note from Table 1 that this section is constrained as required under an optional constraint. There is a nested SHALL statement beneath a SHOULD, which therefore becomes a SHOULD because the parent constraint is optional. Meaning that if the parent component

exists, then there shall be a social history section. Therefore, the association should have a multiplicity of [0..1]

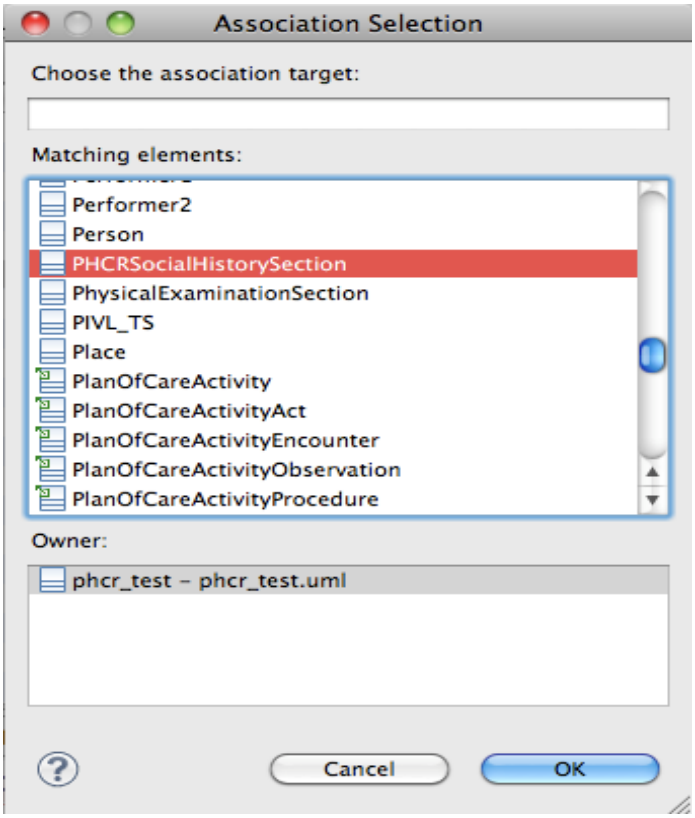

Figure 17 Association Selection

Now that you have created the PHCRSocialHistorySection, you can create the associated entries for that section as described in Table 1 above. You will create some observation entries that you will then associate to the PHCRSocialHistorySection (similarly to the steps above for creating the PHCRSocialHistorySection and association).

For example, to create the Geotemporal History Observation, repeat step 16 to create a template and enter GeotemporalHistoryObservation as the template name.

Then on the Model Element Selection screen, select ***Observation*** as the matching element, and click OK

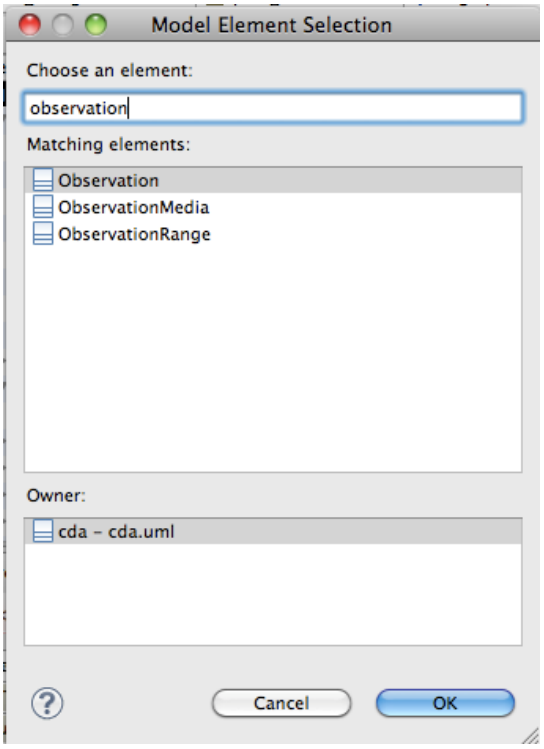

Figure 18 Model Element Selection for Template

You will now select all the required elements from the Template editor as described in Section 4.14 of the implementation guide)

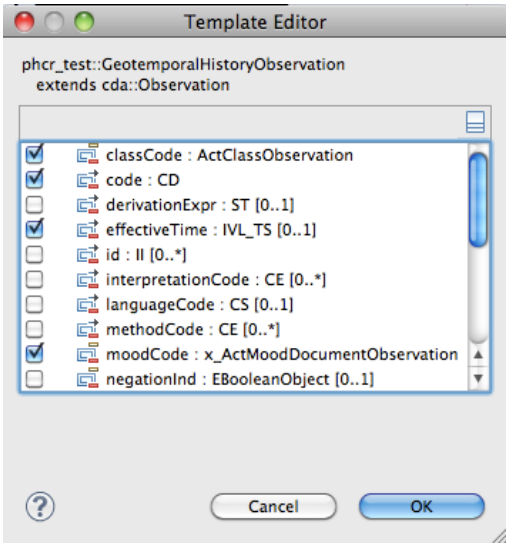

Figure 19 Template Editor

Once they are added in the template editor, you'll see them listed as shown in the figure below

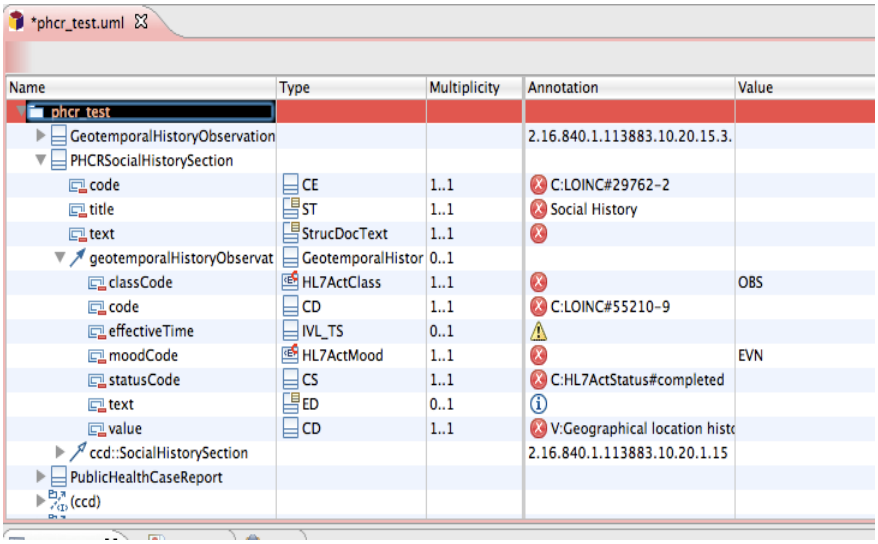

Figure 20 Model After Adding Required Elements

Select each one, and enter the additional attributes on the General and CDA Tools tabs in the properties console that display below the template editor.

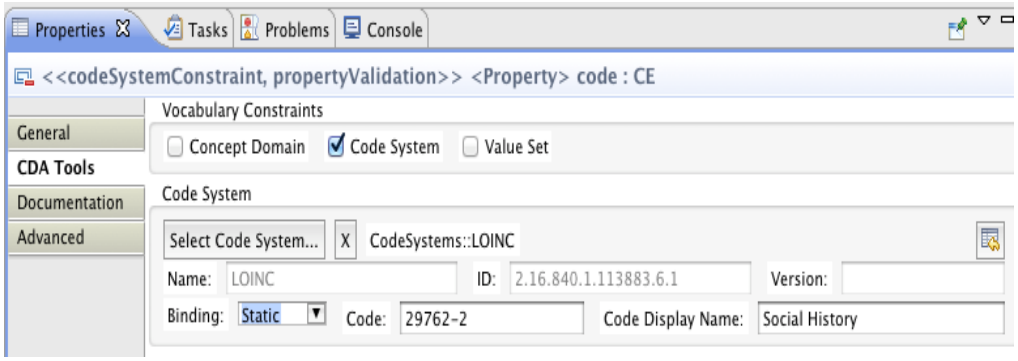

Figure 21 Entering Additional Attributes in Properties Section

To specify the multiplicity of the section or template you just added, you have to do that on the association that allows the template to be integrated into the case report. This is done in the template browser as shown in the picture below.

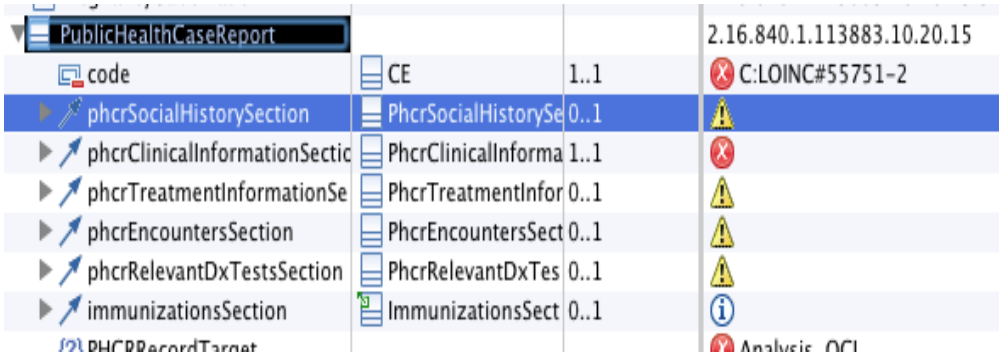

Figure 22 Adding Multiplicity to Associations

For example, to enter the multiplicity for the geotemporalHistoryObservation template, expand the public health case report to show the associated templates. Select the geotempoeralHistoryObservation association, which will have an association arrow in front of it, and

enter the multiplicity in the multiplicity column of the template browser. Also select the CDA Tools tab within the properties section, and enter a Type Code of "DRIV" for the association

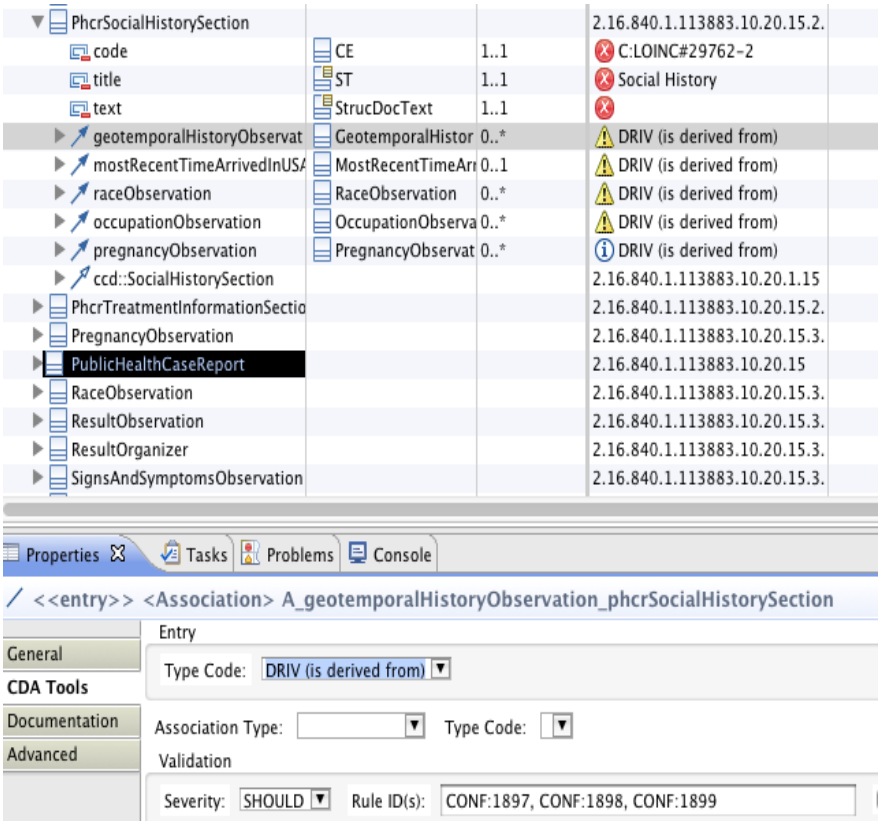

Figure 23 Editing Properties for Association

Once you've built all the section templates and have completed the model, the next step is to populate that first project. This is done by building all the MDHT objects for your model.

Building all the MDHT objects for your model

Now that you have created your UML model, the building process can begin. Building auto-generates all of the java artifacts to be used to create, validate, and digest a case report, and creates a human readable version of the CDA from the generated java artifacts.

To start the build process, first save your model. Then, go to the main project folder and expand the folder. Locate the transform.xml file in that folder (it'll have an image  of an ant next to it)

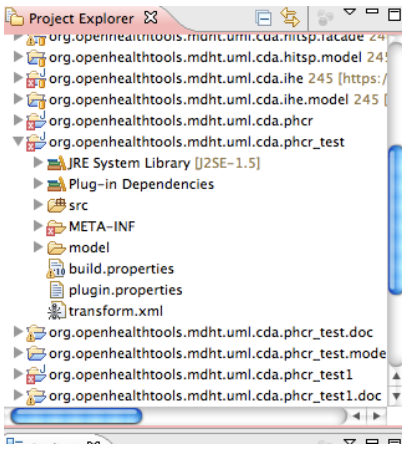

Figure 24 Running transform.xml

Right click on the file, and from the associated menu, select '***Run As'*** and then ***'Ant Build***' - (NOTE: Select ***1 Ant Build,*** not the 2nd one)

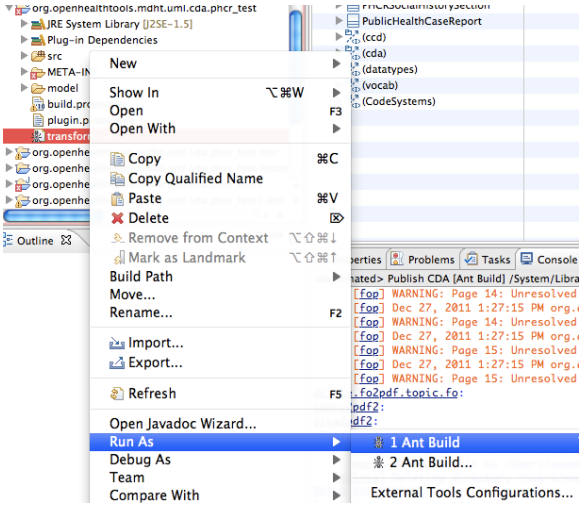

Figure 25 Run As Ant Build Screen

You may get an Edit Configuration screen next. Click on the JRE tab and select the radial button '***Run in the same JRE as the workspace'***, then click 'Apply' and 'Run'.

After the process finishes (building and saving...), return to the model folder and select the ***<project name>***.genmodel file from the model folder

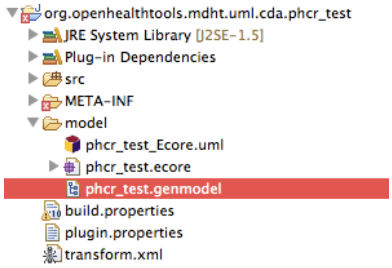

Figure 26 Select Genmodel

Right-click on this file and select '***Reload***' from the associated menu.

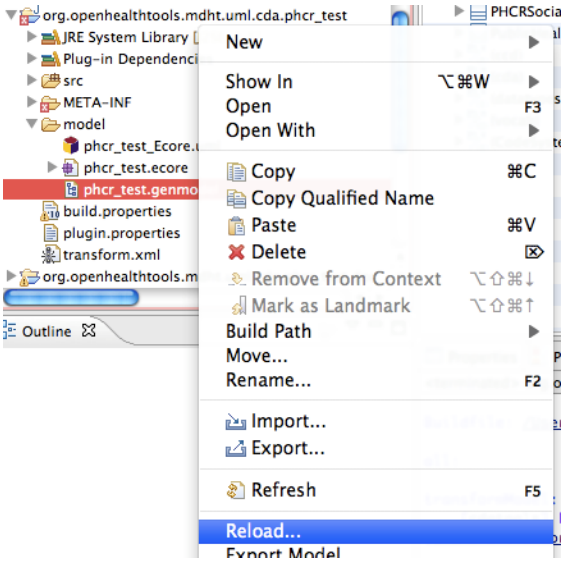

Figure 27 Reload Genmodel

In the Reload screen that will appear, you will accept all defaults and click on 'Next'

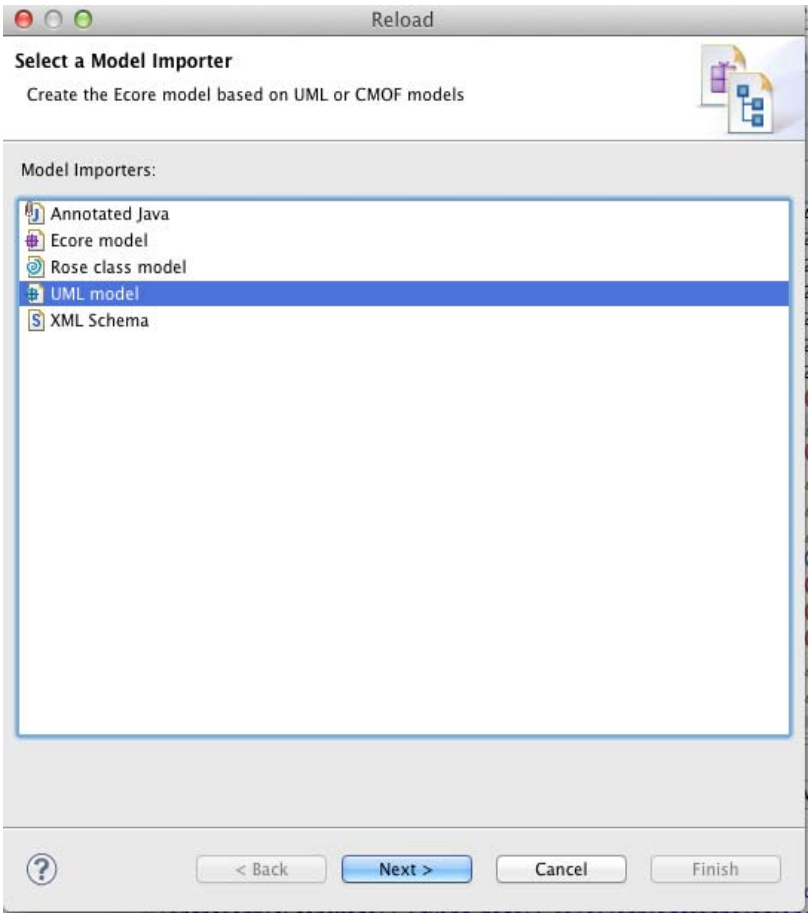

Figure 28 Select a Model Importer Screen

You will then get a 'UML Import' screen. You shouldn't need to do anything special here. Accept the defaults and click "Next".

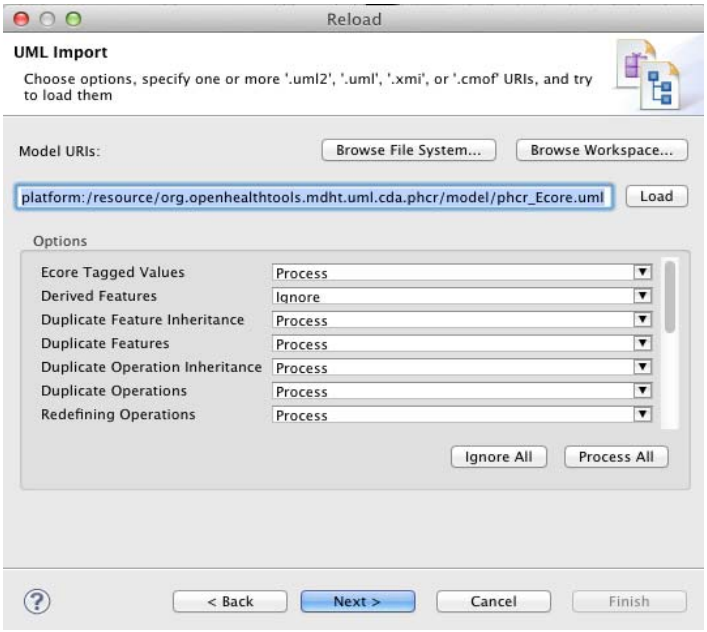

Figure 29 UML Import Screen During Genmodel Reload

That will take you to a 'Package Selection' screen. If everything is setup correctly, you should only see one package in the list of packages. Then select 'Finish'

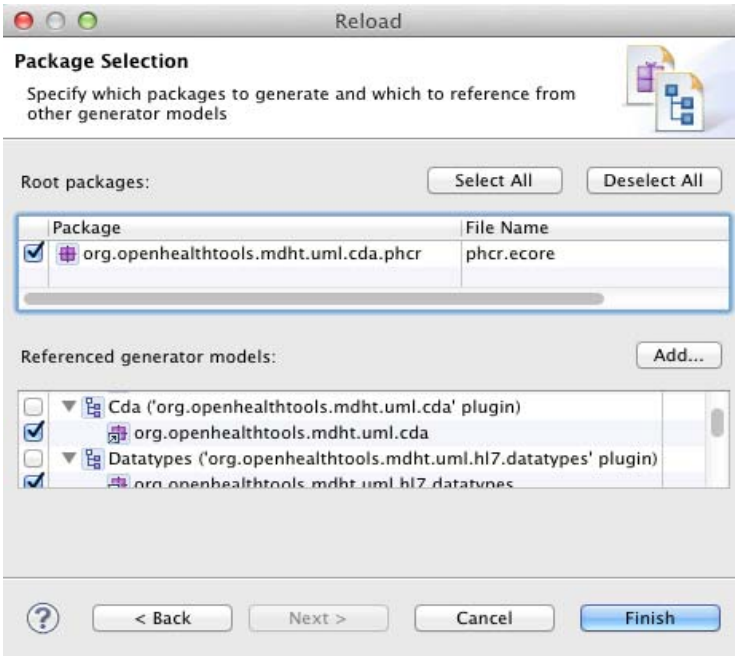

Figure 30 Package Selection Screen During Genmodel Reload

Once the process completes, you will have the <project name>***_Ecore*** folder open in the Template Editor. Right click on this file and select '***Generate Model Code*** '

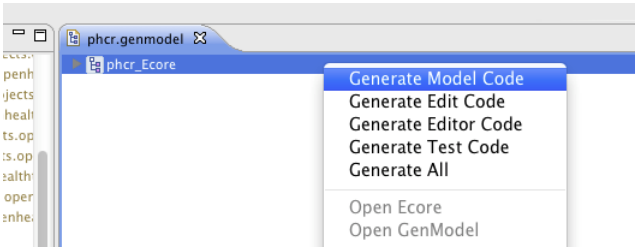

Figure 31 Run Ecore

After the process completes successfully, you will get a '***BUILD SUCCESSFUL***' message. If not, that means there is a problem that needs to be resolved with the model, so you'll have to go through troubleshooting steps. Once the model has been built successfully, this means that the model has now been used to auto-generate the associated files. You can now browse the src folder that is created in your project folder to see the java code that was generated for the project.

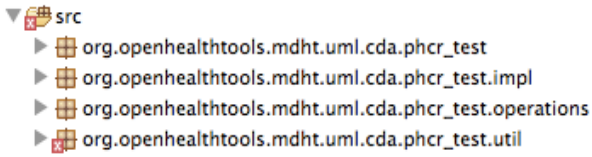

Figure 32 Folders Created from After Generating Model Code

Creating value sets and adding them to your model

The easiest way to start off adding value sets is to copy one of the vocab UML files from one of the other project model folders that are in the project explorer.

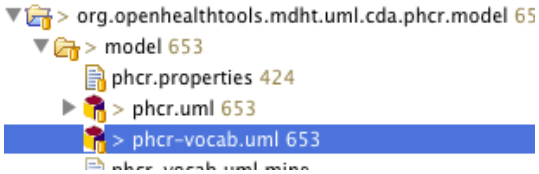

Figure 33 Example of PHCR-vocab UML File

Copy this file and paste it in your model folder for the case report you are working on. Right-click and rename the file using the naming convention <project root name>-vocab.uml. You will then be modifying it to include the value sets that you require for your model.

Once you have the file copied, double click on it to open it for editing

It will open in the template browser.

Delete all the value sets that are listed by right clicking and selecting delete for each one until you have no value sets listed in your <project>-vocab.uml file.

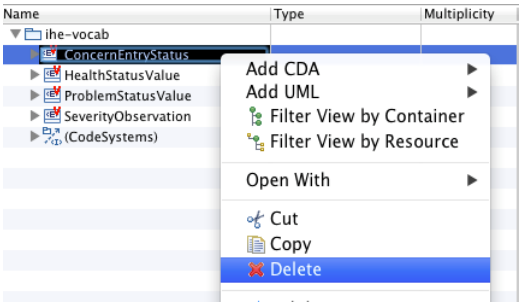

Figure 34 Deleting Existing Value Sets from <proj>-vocab.uml

Then select the <old project name>-vocab listing that is the only entry left, and rename it by editing the Properties tab on the bottom half of the screen

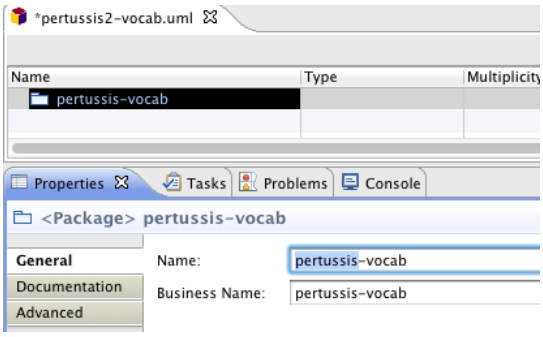

Figure 35 Renaming Root Name in Copied <project>-vocab.uml File

Now you're ready to add in your value sets. Currently, you can only import value sets from the CDC PHIN-VADS system. In order to import a value set, you must first download a zip file of the value set from the https://phinvads.cdc.gov/ website. Note that at this time, 50 values is the size limit for value sets that can be imported into MDHT.

Once you've downloaded the zip file for the value set, you're ready to import the value set into MDHT.

Right click on <project>-vocab file in the project browser and select OHT Model Tools, Import from PHIN VADS

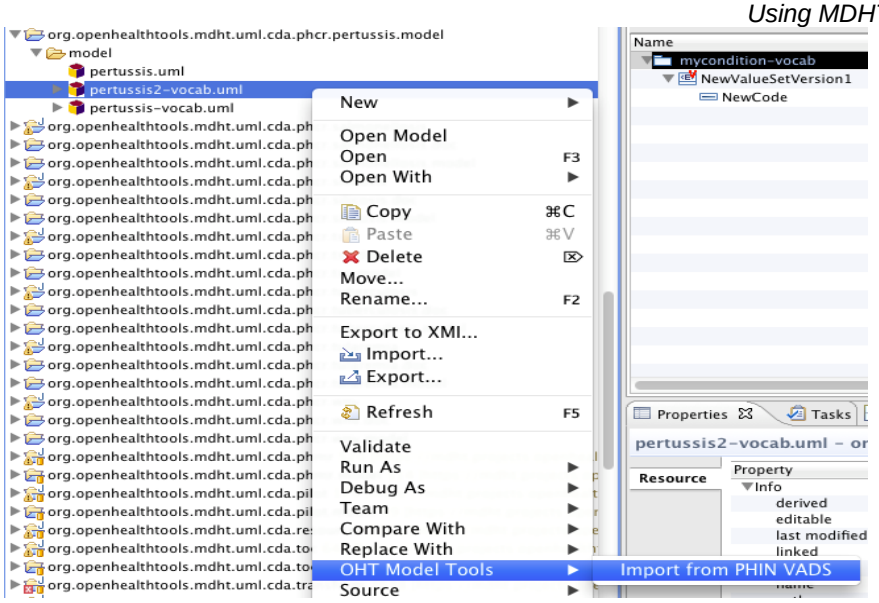

Figure 36 Importing Value Sets into MDHT

You will get a file selection screen. Browse to the location for your downloaded value set zip file and select the file. The process will run and the value set file will be imported. At the end of the process, you should see a 'BUIILD SUCCESSFUL' message. You will have to troubleshoot the process if you do not see this message.

Adding a New Value Set

If you're adding a brand new value set that does not need to be imported, or adding a reference to a value set that is too large to be imported into MDHT, then right click on the <project>-vocab and select Add CDA, and then Value Set Version.

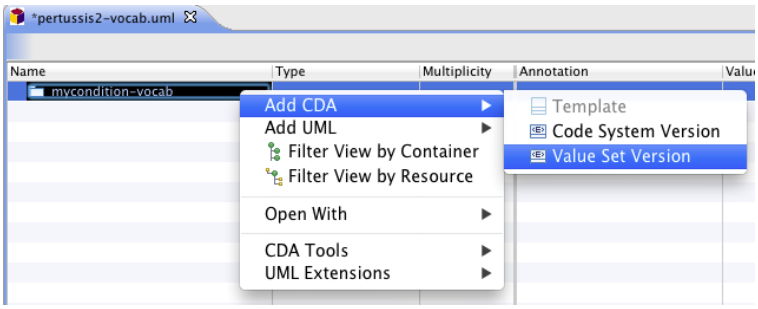

Figure 37 Adding a New Value Sset

Edit the value set information in the Properties tab by adding the value set name under General tab, and the Code system, OID, Source, URL, and additional information under the CDA Tools tab.

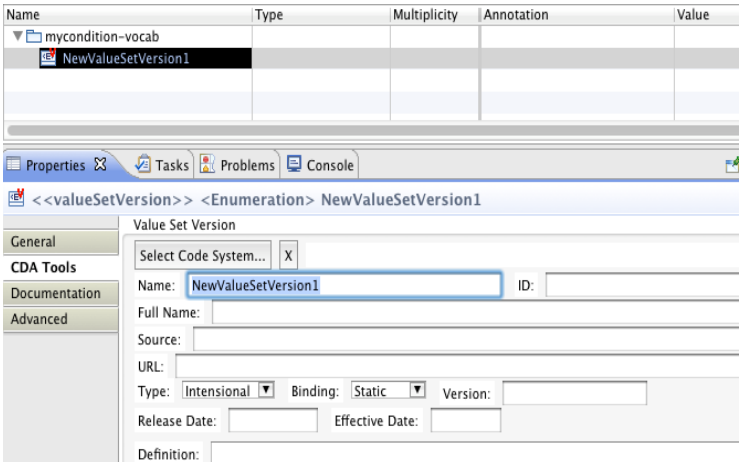

Figure 38 Adding Definition to New Value Set

Now you're ready to add values to the value set. Select the value set and right click and from the menu, select Add CDA and then Value Set Code

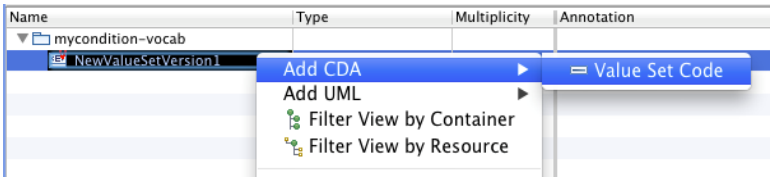

Figure 39 Adding a New Code to a Value Set

You will get a NewCode entry added to the page. In the Properties section beneath the template editor, enter the code in the Name field of the General tab, and add in the Code System, Concept Code, Concept Name, and Usage Code on the CDA Tools tab. Select the '***Add New Code***' button to continue to enter the value set codes until you are done.

Publishing the implementation guide for your model

At the point that you have a valid MDHT model, after all errors have been correct, you can generate the implementation guide automatically for your model. This steps provides you with a pdf that will look like a typical implementation guide with all the constraints specified for your model.

Access your <project name>.doc folder, which contains all the dita files for generating the documentation for your project

Right click on the dita-transform.xml file and select Run as Ant

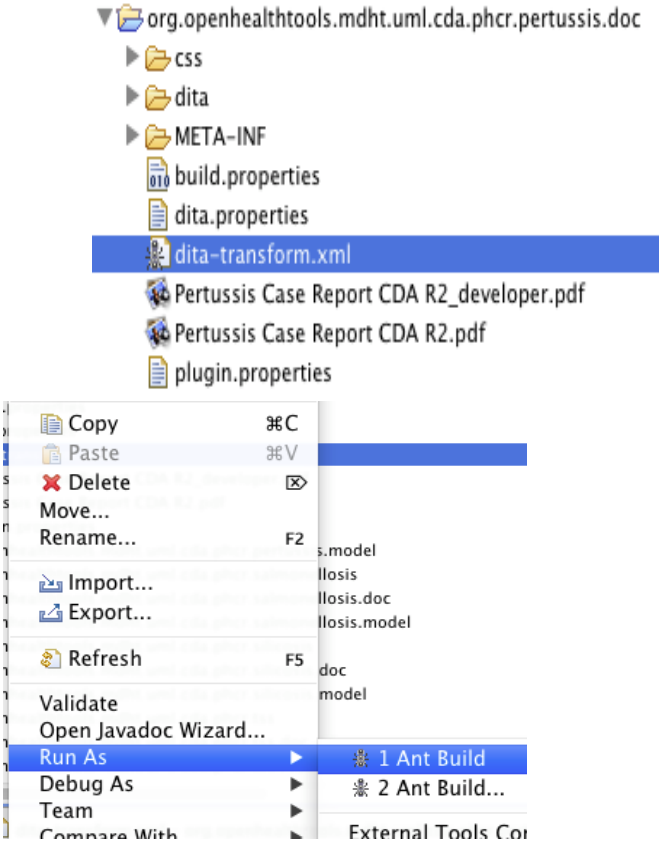

Figure 40 Running the Dita Transform

You may get a dialog box that asks you to choose an Ant configuration to run. Just accept the default selection and hit OK

The process will run and if it completes successfully, you will get a “BUILD SUCCESSFUL” message.

Now go to the dita folder and right-click on spec-book.ditamap and select **MDHT** from the menu, and then select the **Publish** option

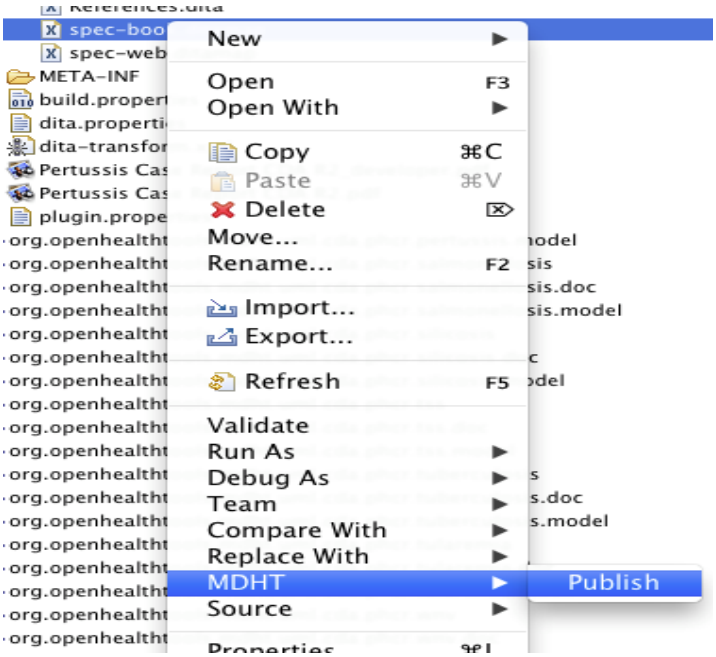

Figure 41 Running Publish Step for Implementation Guide

Once this process completes successfully, you will get a 'BUILD SUCCESSFUL' message, and the implementation guide will be generated and placed in the <project name>.doc folder (you should find it listed under the dita transform (e.g. Pertussis Case Report CDA R2 developer.pdf)