Data Migration

General tips

Every migration is different as it depends to some degree on the source data and the particular configuration of OpenMRS you are migrating data into. Here are some tips based on the way we did it in Tanzania but there may be other ways.

- Prerequisites
  - Before migrating data you should first thoroughly understand both the data model of OpenMRS and that of the old system
  - You should also have a good understanding of SQL “Insert Into” queries (also called append queries)
  - You should also be familiar with how to do configuration such as forms configuration and other configuration in OpenMRS

- Configuring OpenMRS
  - In OpenMRS configure one or more patient identifier types corresponding to the patient identifiers in your old system
  - In OpenMRS configure one or more locations where encounters take place. If your old system assumed all encounters took place in one clinic, then set up that clinic as a location in OpenMRS
  - In OpenMRS configure some patient attributes for the static, unchanging data you collect about patients in your old system which is not included in OpenMRS already in the person, names, addresses or identifiers tables
  - In OpenMRS configure a suitable address format (ask Ben for help on this)
  - In OpenMRS ensure you have concepts which correspond to regularly-collected observation data collected in your old system. For coded concepts ensure the answers correspond to the possible answers in your old system. Of course they don’t have to have the exact same names as the fields in your old system but should correspond and be mappable.
  - In OpenMRS configure encounter forms containing the same data points as the forms in your old system
  - If you have some sort of way in your old system for monitoring status changes like lost to follow-up, transferred out etc you may like to configure them in OpenMRS as workflow states

- Migration general
  - You will need to somehow have the tables from your old system and OpenMRS viewable/editable from the same place - there are many ways of doing this. You could create a new tool/interface/database somewhere and set up links to both the OpenMRS back end and the back end of your old system, or you could just import all the tables from your old system into the OpenMRS database, migrate the data from the old tables into the new tables and then delete the old tables but try not to get mixed up as to which tables are from which system.
  - You may want to create a user called “Import” or similar to whom you attribute all the imported data in the “creator” fields in many tables of OpenMRS. You can create this user in the OpenMRS user interface but you will have to look up their person_id / user_id in the back end.
  - Append data to the person table for all your patients. You may wish to create a temporary field in the person table of OpenMRS to hold your old identifier for the patient which you will need as a linking field during the other append processes before the identifier goes into its proper place in the person_identifier table. Then append data to the patients table for all your patients. Then append data to the person_address, person_name and patient_identifier tables for all your patients.
  - If your old system has a staff or care provider list, you should append these, first into the person table, into the personal_name table, then into the users table, then into the user_role table (give all care providers a role of “provider”). Again, a temporary field in the person table could be created and used for storing your staff ID from your old system as a linking field while you do other queries.
  - Append data for each of your patient attributes into the person_attribute table. If there are many you could do a mapping table otherwise you could just do one query per attribute type.
  - Append a list of encounters or visits into the encounter table. If you have different types of encounters or forms you should link these encounters to the appropriate form_id. You may wish to append initial encounters first and then return encounters giving them the appropriate encounter_type_id.
  - If you have statuses in your old system as above, append all your patients to patient_program and append the patients with their appropriate states to patient_state. You may want to do a mapping table to assist you.

- Migration of observations
  - Rather than having a separate query for each concept it is easier to make a mapping table between the tables and fields in your old system and the concept IDs in OpenMRS and you can use this mapping table to generate an append query for each concept. In your mapping table on the OpenMRS side you could have the fields
    - concept_id
    - concept_name_name (for your ease of reference only)
    - concept_datatype_name
    - concept_answer_id (contains data only in the case of coded concepts)
    - concept_name_1_name (the name of the answer concept - for your ease of reference only)
  - You should also have some fields on the old system's side of the mapping table such as LocalTableName, LocalFieldName, LocalFieldValue which you should fill in manually so that your OpenMRS concepts map to your old system tables and fields.
  - Here is some SQL to get the OpenMRS side of the mapping table. It only chooses concepts which are fields in any of your configured OpenMRS forms: SELECT concept.concept_id, concept_name_name, concept_datatype_name, concept_answer_answer_concept, concept_name_1_name FROM (((field INNER JOIN concept ON field.concept_id = concept.concept_id) INNER JOIN concept_name_1.name FROM ((((field INNER JOIN concept ON field.concept_id = concept.concept_id) INNER JOIN concept_name_1.concept_id) ON concept.concept_id = concept_name_1.concept_id)) INNER JOIN concept_answer ON concept.concept_id = concept_answer.answer_concept) ON concept.concept_id = concept_answer.answer_concept) ON concept.concept_id = concept_answer.answer_concept) LEFT JOIN concept_answer_answer_concept ON concept_answer_answer_concept.concept_answer_id = concept_answer_id) LEFT JOIN concept_answer_answer_concept_id ON concept_answer_answer_concept_id = concept_answer_answer_concept_id)

- Points to note
  - when appending data you should set all voided fields to “0”
  - when appending data it would be recommended to set the date_created to the date you are doing the append
  - when appending data it would be recommended to set the creator to the person_id or user_id of the “Import” user you created in OpenMRS
  - boolean values in OpenMRS are actually stored in value_numeric (1=true, 0=false) in the obs table, not in value_boolean

Rwanda
Here's the code that PIH used to migrate data from our legacy Rwanda HIV-EMR to OpenMRS. Sorry for the sparse documentation. Hopefully I or someone else will be able to comment more later.

Get migration.zip

To start off you want to look at the comments in org.pih.migration.MigrationTemplate class. If you're just migrating registry information, and encounters+obs, that should be all you need.

(Actually you will almost certainly need to change my code to deal with your particular patient_identifiers.)

We also migrated drug orders, program enrollments, and relationships. You'd have to browse through VersionOneExport.java to see how.

Tanzania

In Tanzania we have a tool for migrating data into OpenMRS from the NACP CTC2 database. The tool is in MS Access which links to tables both in MySQL OpenMRS back end (via ODBC) and the MS Access CTC2 database data file. It has a mapping table which maps between OpenMRS concepts and tables and fields in the CTC database. It uses a series of append queries and visual basic procedures to append the data.

Get ExportToOpenMRS.zip

How to use this tool:

- Use the Tanzanian version of OpenMRS with the Tanzanian concept dictionary and without any patient data.
- Create a user in OpenMRS with username "Import" (using the OpenMRS web interface).
- Create two temporary fields in the person table in OpenMRS called PatientID_Temp and StaffID_Temp (using Navicat or similar tool or the MySQL command line), NOTE: PatientID_Temp should be a varchar data type and StaffID_Temp int data type.
- Link the export tool to the tables in the CTC2 database data file (double click the form frmLinkDataFile, browse to the data file (usually called CTC2dataV4.mdb) and click "Link Data File")
- Link the export tool to the MySQL OpenMRS back end via ODBC (Delete all the old table links (the green icons in the tables list), choose file, get external data, link tables, choose link tables, in files of type choose ODBC databases, machine datasource, new, system datasource, MySQL ODBC 3.51 driver (if this is not in the list you need to install the MyODBC connector available from MySQL), next, finish, put in the connection information for MySQL, select all, OK)
- Run the numbered queries qry01 to qry20 one after another in order (this exports patients, identifiers, addresses, staff, patient attributes, workflow states). To run the queries simply double click them.
- Run the AppendObservations procedure (double click the module modImportToOpenMRS, ctrl+g to see the immediate window and in the immediate window type "Call AppendObservations()" and press enter.) Allow a few minutes for it to run.
- Run queries qry21 to qry24 (these delete the adult observations for children or the child observations for adults for CD4 and WHO stage).
- You may now delete the two temporary fields in the person table in OpenMRS called PatientID_Temp and StaffID_Temp if you wish

The MyODBC connector is available at

Unknown macro: 'html'