The FAIR DBA Toolset

Abstract: The University of Massachusetts Medical School (UMMS) has undertaken a new FAIR (Familial, Associational, & Incidental Relationships) Initiative to correlate patients with other patients through certain defined relationships. The “FAIR-correlated” patients will be made available through MICARD (Massachusetts Integrated Clinical Academic Research Database), the UMass implementation of the i2b2 (http://www.i2b2.org) informatics platform for clinical research. As part of this initiative, a set of tools crafted to facilitate updating i2b2 databases has been developed to ease the tedious work of generating and updating the FAIR data in the i2b2 database. Along with the rest of the FAIR Initiative, it will be made available to all other academic institutions that are part of the i2b2 community.

Background

MICARD (Massachusetts Integrated Clinical Academic Research Database, http://micard.umassmed.edu/) is the UMass implementation of the i2b2 (http://www.i2b2.org) informatics platform for clinical research.

MICARD (i2b2) has been continuously proving its importance to the researchers as its user base grows. UMMS has deployed the web client to its researchers as convenient graphical user interface (GUI) to access and utilize MICARD (i2b2).

The i2b2 webclient contains panels for a user to issue queries through its handy drag-and-drop GUI. A user can specify that a Patient Set be generated as part of the results of a particular. These Patient Sets can be used in subsequent analyses and follow up studies.

Introduction

Based on the input and feedback from our MICARD user base, the FAIR (Familial, Associational, & Incidental Relationships) information regarding certain patients may be of appreciable importance in the identification of genetic trends or isolation of environmental effects on the incidences of certain ailments.

The challenges for addressing these needs for, and therefore the steps to compiling such FAIR data are:

1. Defining the appropriate or relevant relationships.
2. Identifying and combing through relevant data across many health and medical databases.
3. Developing a methodology to consistently and reliably filter down this information into a manageable set of data.
4. Devising a simple but flexible schema to encapsulate any such data.
5. Finding the appropriate location in the i2b2 (MICARD) database to store the new data.
6. Generating database administrator’s tools to transform, insert, or update the raw FAIR data into the i2b2 (MICARD) database.
7. Engineering a set of new i2b2 webclient plugins to allow user access and analyses of these newly available FAIR data.

This document covers in details challenge 6 (bolded) above.

**FAIR XML**

The definition of FAIR (Familial, Associational, & Incidental Relationships) data may vary among institutes, as dictated by the different focus and emphasis of each. In addition, even within the same institute, the definition may change as researches evolve.

To address this concern for flexibilities in future growths, it is therefore logical to capture and store any such FAIR information in self-defined (by each institute) simple XML (eXtensible Mark-up Language) forms.

The following are the only requirements:

- It must have the “FAIR_v1.0” root element (i.e. it must begin with “<FAIR_v1.0>” and end with “</FAIR_v1.0>” (e.g. an empty FAIR XML is “<FAIR_v1.0></FAIR_v1.0>” , no quotes).
- It may contain as many simple (no qualifying tags or sub-elements), flat (no nesting or hierarchical arrangements) elements as needed (e.g. “<elem1>some</elem1><elem2>thing</elem2>”;
  “<father>10000000011</father><mother>10000000012</mother>”, no quotes)
- Multiple entries of the same element type may be expressed together but separated with a semi-colon (”,") in the same element (e.g. “<brother> 10000000101; 10000000103; 10000000105; 10000000107 </brother>”, without any quotes), or in separate elements of that same type (e.g. “<brother>10000000101</brother><brother>10000000103</brother> <brother>10000000105</brother> <brother>10000000107</brother>”, no quotes), or both (e.g. “<brother>10000000101; 10000000103; 10000000107; 10000000109</brother> <brother>10000000105</brother>”, no quotes).
- The content entry for each element must be a Patient_ID (e.g. “10000000107” in “<brother>10000000107</brother>”, no quotes).

The following are examples of the FAIR XML (all no quotes):

```
"<FAIR_v1.0><father>10000000011</father><mother>10000000016</mother><brother>10000000013; 10000000015; 10000000017</brother><sister>10000000026; 10000000054</sister></FAIR_v1.0>"
```

```
"<FAIR_v1.0><father>10000000014</father><mother>10000000016</mother></FAIR_v1.0>"
```
FAIR Tabulation XLS / CSV

As can be readily seen, manually coding each FAIR XML for every patient can be quite tedious and typo-prone. Since it is reasonable to assume that the fruit of all the data mining / filtering / funneling operations is tabulated in some consistent format, based on which the FAIR XML is to be coded, if they were to be so generated manually; then it would be best if the FAIR XML can be automatically generated from the raw tabulation.

This can be accomplished with the set of database administrator tools (DBA Toolset), discussed in the next section, provided that the following simple stipulations are observed:

- The tabulation must be in either an Excel spreadsheet (XLS), or a comma-separated-values (CSV) file.
- For XLS, the following straight-forward requirements must be followed –
  - All tabulation data must be contained in the first or top sheet only.
  - Row 1 must contain the “patient” and each FAIR element (no quotes).
  - Rows 2 – N must contain the corresponding entries, with multiple entries for an element separated by semi-colon (“;”).
- For CSV, the following requirements must be followed –
  - Row 1 must contain the “patient” and each FAIR element, each separated by a comma (“,”) (no quotes).
  - Rows 2 – N must contain the corresponding entries, with multiple entries for an element separated by semi-colon (“;”), with entries for each pair of different adjacent elements separated by a comma (“,”) (no quotes).

The following are examples of such required tabulation files:

<table>
<thead>
<tr>
<th>patient,</th>
<th>father, mother, brother, sister, son, daughter, husband, wife</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000000033, 1000000023, 1000000015, 1000000070, 1000000071, 1000000108, 1000000109 , , 1000000111, 1000000083, 1000000081, , , , 1000000111, 1000000112, 1000000113 , , , 1000000083 , , , ,1000000085; 1000000097; 1000000089; 1000000094 , , ,</td>
<td></td>
</tr>
</tbody>
</table>

Figure 1: Example of a FAIR tabulation CSV file.
Figure 2: Example of a FAIR tabulation XLS file.

It should be obvious that either tabulation form is fundamental, logical, and necessary, and requires negligible additional effort, if any. Furthermore, the simplicity and straightforwardness of either form should also compare favorably against the actual XML.

**FAIR DB Administrator’s Toolset**

Part of the FAIR Initiative is to make it as easy as possible for a DB administrator (DBA) to insert or update the FAIR data for each patient into the i2b2 DB, after all the tedious upfront works has been done in compiling them.

The FAIR DBA Toolset consists of the following utilities:
- PL/SQL scripts containing useful procedures to be stored and used in the i2b2 DB
  - `FAIRupdate.sql` – contains the `FAIRupdate()` procedure.
  - `FAIRshow.sql` – contains the `FAIRshow()` procedure.
- VBS scripts for generating PL/SQL scripts to call the aforementioned stored procedures to update and show the FAIR XML for specified patients in the i2b2 DB
  - `FAIRxls2sqls.vbs` – takes an XLS input file that contains the patient FAIR data.
  - `FAIRcsv2sqls.vbs` - takes a CSV input file that contains the patient FAIR data.

![Diagram of the FAIR DBA Toolset](image)

Figure 1: The FAIR DBA Toolset.

With these tools, a DBA simply need to do the following when it is necessary to input / update patient FAIR data into the i2b2 DB:

1. Run `FAIRupdate.sql` and `FAIRshow.sql` scripts on the i2b2 DB (e.g. via the Oracle APEX GUI) to load / store the two procedures (note that this only needs to be done once).
2. Compile the patient FAIR data into either an Excel spreadsheet (XLS) file or a comma-separated-values (CSV) file.

3. Run either FAIRxls2sqls.vbs or FAIRcsv2sqls.vbs against the XLS or CSV FAIR data file, respectively, to generate the resulting “FAIRxml-updates.sql” and “FAIRxml-shows.sql” (note: name of each generated file to be specified by the user as part of the invocation of either VBS utility) script files.

4. Run the “FAIRxml-updates.sql” script on the i2b2 DB (e.g. via the Oracle APEX GUI) to input / update the patient FAIR data accordingly.

5. Optionally run the “FAIRxml-shows.sql” script on the i2b2 DB (e.g. via the Oracle APEX GUI) to display the latest patient FAIR data accordingly.

**FAIRxls2sqls.vbs**

This Windows 7 utility will take as its input the FAIR tabulation XLS file, and generate two PL/SQL files. When run on the i2b2 DB (e.g. via the Oracle APEX), the first PL/SQL script will update the FAIR XMLs of the specified patients in the DB; while the second PL/SQL script will simply display the latest FAIR XMLs of the specified patients.

This utility can be invoked in Windows 7, either from a command line, or by direct double-clicking its icon in Windows Explorer.

The command line invocation syntax is (assuming it locates in the current command prompt folder)

```
 wscript FAIRxls2sqls.vbs input-fullyQualifiedPathName output-FQPNprefix
```

e.g.

```
 wscript FAIRxls2sqls.vbs h:\example\path\infile.xls   h:\example\path\outfilePrefix
 wscript FAIRxls2sqls.vbs h:\winscripts\test.xlsx       h:\winscripts\testPrefix
 wscript FAIRxls2sqls.vbs h:\winscripts\tmp\FAIRxls2sqls-in-sample.xls   h:\winscripts\tmp\FAIRdemo
```

where

the input XLS file must observe the rules for a FAIR Tabulation XLS file; and

the resulting output files will be named as

```
 output-FQPNprefix-updates.sql, and
 output-FQPNprefix-shows.sql, respectively;
```

as indicated in a dialog, as in the following:
Note that if any of the resulting files already exist, then a dialog will be displayed to warn about it first, as in the following:

![Figure 5: The FAIRxls2sqls warning dialog about replacing the "-updates.sql" file.]
Figure 6: The FAIRxls2sqls warning dialog about replacing the "-shows.sql" file.

In response to the icon invocation (double-clicked), this utility will display a set of dialogs to solicit the necessary input, as following:

Figure 7: The FAIRxls2sqls input dialog soliciting the input file name.

Figure 8: The FAIRxls2sqls input dialog soliciting the output file prefix.
Any errors in the full path in either case (also in the command line invocation) will result in error dialog. The resulting output files (“-updates.sql” and “-shows.sql”) will be discussed in a later section.

**FAIRcsv2sqls.vbs**

This Windows 7 utility will take as its input the FAIR tabulation CSV file, and generate two PL/SQL files. When run on the i2b2 DB (e.g. via the Oracle APEX), the first PL/SQL script will update the FAIR XMLs of the specified patients in the DB; while the second PL/SQL script will simply display the latest FAIR XMLs of the specified patients.

This utility can be invoked in Windows 7, either from a command line, or by direct double-clicking its icon in Windows Explorer.

The command line invocation syntax is (assuming it locates in the current command prompt folder)

```plaintext
wscript FAIRcsv2sqls.vbs input-fullyQualifiedPathName output-FQPNprefix
```

e.g.

```plaintext
wscript FAIRcsv2sqls.vbs h:\example\path\infile.csv h:\example\path\outfilePrefix
wscript FAIRcsv2sqls.vbs h:\winscripts\test.csv h:\winscripts\testPrefix
wscript FAIRcsv2sqls.vbs h:\winscripts\tmp\FAIRcsv2sqls-in-sample.csv h:\winscripts\tmp\FAIRdemo
```

where

the input XLS file must observe the rules for a FAIR Tabulation CSV file; and

the resulting output files will be named as

```
output-FQPNprefix-updates.sql, and
output-FQPNprefix-shows.sql, respectively;
```

as indicated in a dialog, as in the following:
Figure 9: The FAIRcsv2sqls results dialog.

Note that if any of the resulting files already exists, then a dialog will be displayed to warn about it first, as in the following:

Figure 10: The FAIRcsv2sqls warning dialog about replacing the "-updates.sql" file.
Figure 11: The FAIRcsv2sqls warning dialog about replacing the "-shows.sql" file.

In response to the icon invocation (double-clicked), this utility will display a set of dialogs to solicit the necessary input, as following:

Figure 12: The FAIRcsv2sqls input dialog soliciting the input file name.

Figure 13: The FAIRcsv2sqls input dialog soliciting the output file prefix.
Any errors in the full path in either case (also in the command line invocation) will result in error dialog.

The resulting output files ("-updates.sql" and "-shows.sql") will be discussed in a later section.

**The “-updates.sql” and “-shows.sql” PL/SQL Scripts**

These scripts are generated by either the FAIRxls2sqls.vbs as result of processing the FAIR Tabulation XLS file, or the FAIRcsv2sqls.vbs as result of processing the FAIR Tabulation CSV file.

Figure 14: The generation of the "-updates.sql" and "-shows.sql" scripts.
The "-updates.sql" PL/SQL Script

This is a simple script to insert / update the FAIR XML data in the i2b2 DB (within the PATIENT_DIMENSION.PATIENT_BLOB field). It contains only calls to the FAIRupdate( ) procedure (refer to a later section), one for each patient.

```
-- this file was generated by FAIRxls2sqls.vbs (a U. of Massachusetts Medical School DBA tool), from the
-- 'C:\SWC\i2b2\webClient_plugin\FamilyFiles\FAIRxls2sql\FAIRxls2sqls-in-sample.xls' file
-- for updating FAIR xml data in PATIENT_DIMENSION.PATIENT_BLOB of the i2b2 DB

BEGIN
  FAIRUpdate(1000000009,
    '<FAIR_v1.0<father>1000000012</father><mother>1000000016</mother><brother>1000000011</brother><sister>1000000071</sister><son>1000000010</son><daughter>1000000019</daughter><FAIR_v1.0>',false);
  FAIRUpdate(1000000011,'<FAIR_v1.0<father>1000000033</father><mother>1000000031</mother><brother>1000000011</brother><daughter>1000000019</daughter><FAIR_v1.0>',false);
  FAIRUpdate(1000000033,<FAIR_v1.0<sister>1000000095</sister> 1000000095; 1000000094</sister><FAIR_v1.0>',false);
  FAIRUpdate(1000000031,'<FAIR_v1.0<father>1000000014</father><mother>1000000016</mother><brother>1000000012</brother><FAIR_v1.0>',false);
  FAIRUpdate(1000000014,'<FAIR_v1.0<father>1000000014</father><mother>1000000016</mother><brother>1000000012</brother><FAIR_v1.0>',false);
  FAIRUpdate(1000000012,'<FAIR_v1.0<father>1000000016</father><mother>1000000011</mother><brother>1000000014</brother><FAIR_v1.0>',false);
  FAIRUpdate(1000000014,'<FAIR_v1.0<father>1000000016</father><mother>1000000011</mother><brother>1000000014</brother><FAIR_v1.0>',false);
  FAIRUpdate(1000000012,'<FAIR_v1.0<father>1000000016</father><mother>1000000011</mother><brother>1000000014</brother><FAIR_v1.0>',false);
  FAIRUpdate(1000000014,'<FAIR_v1.0<father>1000000016</father><mother>1000000011</mother><brother>1000000014</brother><FAIR_v1.0>',false);
  FAIRUpdate(1000000012,'<FAIR_v1.0<father>1000000016</father><mother>1000000011</mother><brother>1000000014</brother><FAIR_v1.0>',false);
  FAIRUpdate(1000000014,'<FAIR_v1.0<father>1000000016</father><mother>1000000011</mother><brother>1000000014</brother><FAIR_v1.0>',false);
  FAIRUpdate(1000000012,'<FAIR_v1.0<father>1000000016</father><mother>1000000011</mother><brother>1000000014</brother><FAIR_v1.0>',false);
  FAIRUpdate(1000000014,'<FAIR_v1.0<husband>1000000014</husband><FAIR_v1.0>',false);
  FAIRUpdate(1000000012,'<FAIR_v1.0<husband>1000000014</husband><FAIR_v1.0>',false);
  FAIRUpdate(1000000014,'<FAIR_v1.0<wife>1000000014</wife><FAIR_v1.0>',false);
  FAIRUpdate(1000000012,'<FAIR_v1.0<wife>1000000014</wife><FAIR_v1.0>',false);
  FAIRUpdate(1000000014,'<FAIR_v1.0<husband>1000000014</husband><FAIR_v1.0>',false);
  FAIRUpdate(1000000012,'<FAIR_v1.0<husband>1000000014</husband><FAIR_v1.0>',false);
  FAIRUpdate(1000000014,'<FAIR_v1.0<husband>1000000014</husband><FAIR_v1.0>',false);
  FAIRUpdate(1000000012,'<FAIR_v1.0<husband>1000000014</husband><FAIR_v1.0>',false);
  FAIRUpdate(1000000014,'<FAIR_v1.0<husband>1000000014</husband><FAIR_v1.0>',false);
  FAIRUpdate(1000000012,'<FAIR_v1.0<husband>1000000014</husband><FAIR_v1.0>',false);
  FAIRUpdate(1000000014,'<FAIR_v1.0<husband>1000000014</husband><FAIR_v1.0>',false);
  FAIRUpdate(1000000012,'<FAIR_v1.0<husband>1000000014</husband><FAIR_v1.0>',false);
END;
```

Figure 15: A typical "-updates.sql" script generated by either FAIRxls2sqls.vbs or FAIRcsv2sqls.vbs.

Note that each of the FAIR XMLs has been generated automatically as a result of either .vbs parsing the FAIR Tabulation file. In addition, a DBA needs not even worry about the invocation syntax of the procedure call (even though it will be explained in a later section).
The “-shows.sql” PL/SQL Script

This is a simple script to fetch and render, in readily readable form, the FAIR XML data in the i2b2 DB (within the PATIENT_DIMENSION.PATIENT_BLOB field). It contains only calls to the FAIRshow( ) procedure (refer to a later section), one for each patient.

![Figure 16: A typical "-shows.sql" script generated by either FAIRxls2sqls.vbs or FAIRcsv2sqls.vbs.](image)

Note that again, each of the call has been generated automatically as a result of either .vbs parsing the FAIR Tabulation file. Furthermore, a DBA needs not even worry about the simple invocation syntax of the procedure call (even though it will be explained in a later section).

**FAIRupdate.sql**

This PL/SQL contains only the FAIRupdate( ) procedure, which should be run and loaded onto the i2b2 DB (e.g. via the Oracle APEX).
The FAIRupdate( ) procedure invocation syntax is

FAIRupdate(patient_num, FAIRxml, debugDetails {true / false})

where FAIRxml should be in the form of examples below

e.g.

FAIRupdate(1000000083,'<FAIR_v1.0><sister>1000000085; 1000000087</sister></FAIR_v1.0>',false);

FAIRupdate(1000000009,'<FAIR_v1.0><father>1000000023</father><mother>1000000025</mother><brother>1000000070</brother><sister>1000000071</sister><son>1000000108</son><daughter>100000109</daughter></FAIR_v1.0>',false);

This procedure covers the following conditions when inserting / updating the FAIR XML in the PATIENT_DIMENSION.PATIENT_BLOB field:

1) PATIENT_BLOB is null or empty.
2) PATIENT_BLOB doesn't yet contain any FAIRxml.
3) PATIENT_BLOB only contains FAIRxml.
4) FAIRxml is at the end of PATIENT_BLOB.
5) FAIRxml is at the beginning of PATIENT_BLOB, and is trailed by other sub-blob.
6) FAIRxml is sandwiched by other sub-blobs in PATIENT_BLOB.

Conditions 1 and 2 are the cases of inserting the FAIR XML into the PATIENT_BLOB.

![Example result of executing FAIRupdate() procedure calls.](image)

Note that the FAIRupdate() procedure calls in the figure above is atypical, in that for the FAIR purpose one would not be inserting anything other than the FAIR XML. However, this example does serve to prepare for the next example, which demonstrates the functionalities of FAIRupdate() procedure in handling conditions 6 and 4 above.
Figure 19: Example calls to FAIRupdate(), demonstrating the update of sandwiched FAIRxml, etc.

FAIRshow.sql

This PL/SQL contains only the FAIRshow() procedure, which should be run and loaded onto the i2b2 DB (e.g. via the Oracle APEX).
The FAIRshow() procedure invocation syntax is

\[ \text{FAIRshow}(\text{patient\_num}) \]

e.g.

FAIRshow(1000000083);

FAIRshow(1000000009);

This procedure also covers the following conditions when rendering the FAIR XML in the PATIENT\_DIMENSION.PATIENT\_BLOB field:

1) \text{PATIENT\_BLOB} is null or empty.
2) \text{PATIENT\_BLOB} doesn't yet contain any FAIRxml.
3) \text{PATIENT\_BLOB} only contains FAIRxml.
4) FAIRxml is at the end of \text{PATIENT\_BLOB}.
5) FAIRxml is at the beginning of \text{PATIENT\_BLOB}, and is trailed by other sub-blob.
6) FAIRxml is sandwiched by other sub-blobs in \text{PATIENT\_BLOB}.
Note that FAIRshow() renders the entire PATIENT_BLOB content, as well as just the FAIRxml itself (if present), plus the lengths of both.

The figure below shows the “blob” in PATIENT_BLOB when retrieved without this procedure:
Conclusions

MICARD (i2b2) has been and continues to be invaluable in facilitating translational research at UMMS. The UMMS Biomedical Informatics Core, funded by the CTSA, is constantly seeking ways to enhance an investigator’s ability to use the platform for cohort discovery.

The goal of the FAIR (Familial, Associational, and Incidental Relationships) Initiative undertaken here by UMMS is one such effort to further expand the capabilities of the researchers in their endeavors.

The DBA Toolset crafted as part of the FAIR Initiative consists of handy utilities that should replace the long, arduous works involved in coding, storing, checking, and updating the patient FAIR data in the i2b2 DB with push-button operations.

The total outcomes of the FAIR Initiative, like the approach and methodology developed to identify and quantify FAIR members of patients, the simple FAIR Tabulation file requirement defined, the simple flexible FAIR XML schema enunciated, the location in the i2b2 DB to store the FAIR XML proposed, the DBA Toolset crafted to generate and update the patient FAIR data in the i2b2 DB, and the FAIR i2b2 webclient plugins, should be quite useful for other institutions as well.

The end result of this FAIR Initiative should thus enhance i2b2 DBA and researcher experiences, and further expand the MICARD user base at UMMS. It should also be an appreciable contribution to the entire i2b2 community.