**caTissueLoader**

**Version 0.6**

**23-Nov-2010**

**Ian Fore**

**NCI Center for Bioinformatics**

Document history

|  |  |
| --- | --- |
| **Date** | **Notes** |
| 5-Dec-2007 | Interim version distributed to 2 potential users |
| 7-Dec-2007 | Updates on included JAXB libraries.  Running instructions added.  Described keychain file  First attempt to make this document usable by informatics savvy users. |
| 13-Dec-2007 | Added validation of mapping files  Notes re third party software used  Notes re caTissue client jars  Additional examples  Initial deployment to GForge |

SVN details

$Date$

$Revision$

$Author$

caTissue text file loader 4

Example 5

Prerequisites 8

Running 9

Directory structure 10

Manually loaded data 10

Mapping document 10

Joining parent records 11

Further examples 11

Participant 11

Group and Specimens 12

Group only (not done yet) 12

Child specimens 12

Mapping files 12

Case sensitivity 12

Editing the mapping document 12

Checking the mapping document 13

Mapping document schema 13

Saved value mappingss 14

Attribute names 14

Participant & Registration 14

Specimen Collection Group 15

Specimen 15

Race 16

Mapping based on another value 16

caTissue keychain 16

Keychain schema 17

Exporting data files from applications 17

Microsoft Excel 17

Microsoft Access 17

FileMaker Pro 18

Third party software 18

JAXB 18

FlatPack 18

Known lack of robustness 18

Possible improvements 18

Get data over the web 20

Test cases 20

Requirements 21

Notes on design decisions 21

Types of objects that may be loaded 21

Granularity 21

Notes for Suite 1.0 21

Building the distribution 21

Appendix 1 - Valid values for specimen type 22

# caTissue text file loader

This loads data from text files. These may easily be saved from commonly used programs such as Microsoft Excel. Microsoft Access and FileMaker Pro. You can use tab delimited or comma separated values. Currently tab delimited file have been more extensively tested and are therefore to be preferred – at least for now.

This version permits loading of participants, specimen collection groups and specimens plus the basic classes that go along with them such as registrations, collection and received events.

This version of the tool does not provide a dedicated user interface for loading files. Details of how to load data are provided by editing simply structured XML files. It is expected for now that the tool will be used by people with a moderate level of computer expertise and who would be able to edit simple XML files. Care does need to be taken to provide a “well formed” XML file. The XML files are validated by the program before use.

The following diagram shows the overall scheme for loading data.

Data file

Mapping file

caTissue

caTissueLoader

Value Mapping file

caTissue keychain

Bad Data file

Data file – This is the source data that is to be loaded into caTissue.

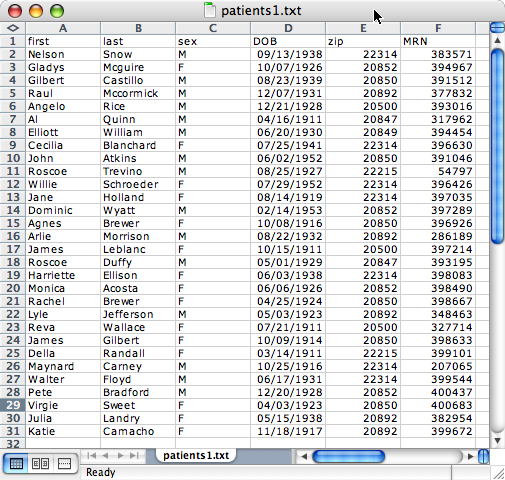
Mapping file – This describes how the data file is to be loaded. A mapping file defimes the structure of the data file and can be used to many data files which have the same structure. It defines what each column of the data file contains. It also can be used to provide data required by caTissue Core but which is not used in the source system from which data is being loaded. Mappings for individual data items can also be provided – . These may be in the mapping file or an optional external value mapping file. The mapping file also provides information such as the date format used in the data file.

caTissue keychain – this is a file used to contain information about the location and login information for a number of caTissue instances.

Bad data file – Any rows of the data file that generate an error when loading will be written to this file. The exact row is written to the file with a message as to why the row could not be loaded. After fixing the errors , the same file may be used as the sou

## Example

This example will use the following example excel file



Each row of this file contains information for a patient that we wish to load to caTissue as “Participants”. Several columns contain information that we want to import, but there is also a column that we don’t need to load.

Create a mapping file as follows

The first line of the file is standard for XML files and needn’t concern us too much for now – other than that it should be there.

<?xml version="1.0" encoding="UTF-8"?>

We will look next at the first real row of the file and last row.

<columnMap headerRow="true" fileType="xlText">

… rest of the file here

</columnMap>

The attributes tell the loader that the file contains a header row containing column names and that the loader should ignore that row. If there are no column names the headerRow attribute should be set to false and the loader will import the first row. fileType designates that this is an excel tab delimited text file.

The file ends by closing the columnMap like this

</columnMap>

In between those two tags the details of the mapping are provided.

First the format to be used for dates in the data file is given.

<dateFormat>mm/dd/yyyy</dateFormat>

This format means that dates will be in the form 03/23/1924. Other formats can be used as described in more detail later in this guide.

Next comes an element called rowClass which tells the loader it is to create a Participant for each row imported.

<rowClass>Participant</rowClass>

Next follows a sequence of column elements which each define an attribute to be loaded to caTissue and where to find it. The first column element is as follows.

<column attributeName="firstName" columnNo="1"/>

This indicates that the firstName attribute is to be found in the first column of the data file. Note that as this tag does not contain other elements it can be closed with a / before the final >.

The next line for lastName is also straightforward, indicating simply that the values are in the second column of the datafile.

<column attributeName="lastName" columnNo="2"/>

It is not actually necessary to specify column details in mapping file in the order they appear in the data file in this way, but it does make it easier to keep track of things!

The next column specifies that gender is contained in the third column of the data file. However, caTissue has valid values for gender of “Female Gender” and “Male Gender”. However looking at the third column of the data file we can see it uses M and F. To perform translation of the source values to valid caTissue values we will provide a mapping of the values by using a valueMapping tag. Each valueMap tag within the valueMapping contains the value which should be translated (fromValue) and the valid caTissue value (toValue).

<column attributeName="gender" columnNo="3">

<valueMapping>

<valueMap fromValue="F" toValue="Female Gender"/>

<valueMap fromValue="M" toValue="Male Gender"/>

</valueMapping>

</column>

Next we have two columns which again are straightforward

<column attributeName="birthDate" columnNo="4"/>

<column attributeName="medicalRecordNumber" columnNo="5"/>

The second of these is the medical record number for the participant. In fact caTissue allows us to associate multiple medical record numbers with a participant. Different numbers may be used for the same individual at different sites. In this loader a simplification has been made for now that there will be a single medical record number.

caTissue also requires a number of attributes which are not present in the example data file. To supply values for these we can replace the columnNo attribute of the column element with a fixedValue element. This value will be used for all participants loaded from our data file to caTissue. The next row of the mapping file provides a fixed value for the Site.

<column attributeName="siteName" fixedValue="Haven outpatients clinic"/>

The loader also creates “Collection Protocol Registrations” for each participant. In our example all the participants will be registered to the same protocol – so we can use a fixed value in the mapping file rather than reading values from the data file. Each registration should have an identifier for that participant on that protocol. The designers of the study we are importing did not see it as necessary to have such a separate identifier and relied on the medical record number to identify patients. In this case, then, we can reuse the medical record number in column 5 of the data file and map it to the Protocol Participant Identifier.

<column attributeName="protocolTitle" fixedValue="Demo of collection protocol"/>

<column attributeName="PPI" columnNo="5"/>

Finally we have two other attributes which caTissue requires – the Activity Status of both the Participant and Registration.

<column attributeName="participantActivityStatus" fixedValue="Active"/>

<column attributeName="regActivityStatus" fixedValue="Active"/>

To see everything in one place the whole mapping file looks like this. This file is the examples folder as simplePatientTextMap.xml.

<?xml version="1.0" encoding="UTF-8"?>

<columnMap headerRow="false" fileType="xlText">

<dateFormat>mm/dd/yyyy</dateFormat>

<rowClass>Participant</rowClass>

<column attributeName="firstName" columnNo="1"/>

<column attributeName="lastName" columnNo="2"/>

<column attributeName="gender" columnNo="3">

<valueMapping>

<valueMap fromValue="F" toValue="Female Gender"/>

<valueMap fromValue="M" toValue="Male Gender"/>

</valueMapping>

</column>

<column attributeName="birthDate" columnNo="4"/>

<column attributeName="siteName" fixedValue="Haven outpatients clinic"/>

<column attributeName="medicalRecordNumber" columnNo="5"/>

<column attributeName="protocolTitle" fixedValue="Demo of collection protocol"/>

<column attributeName="PPI" columnNo="5"/>

<column attributeName="participantActivityStatus" fixedValue="Active"/>

<column attributeName="regActivityStatus" fixedValue="Active"/>

</columnMap>

## Prerequisites

Java SDK 1.5

Ant 1.7

caTissue API client libraries

The caTissue client libraries are included for the currently released and fully tested version. You can check your caTissue hoe page for the version you have. If it says version 1.2 (RC2) you are OK with the files distributed with this program.

If you are using a later version of caTissue Core (e.g. 1.2.0.1 or 1.2.1) you must use the client files for that release. When you downloaded the caTissue application the unzipped package contained a caTissueCore\_caCORE\_Client/lib folder. Copy (duplicate) the contents of the lib folder to the lib folder of the caTissueCoreLoader application. You will need to replace the files already there.

## Running

Overview

Edit build.xml to specify the name of the data file , the mapping file and other parameters as follows.

Type

ant loadData

at the command line.

Details

The following lines show

<!-- edit these properties for a particular load of data -->

<property name="catissue.instance" value="MacBook" />

<property name="keychain.file" value="./conf/caTissueKeychain.xml" />

<property name="data.file" value="./textdata/10patients2.txt" />

<property name="map.file" value="./textdata/PatientTextMap.xml" />

<property name="baddata.file" value="./textdata/badData/bad10patients2.txt" />

Parameters are defined as follows. All pathnames for files should use the forward slash character “/” irrespective of the operating system under which the program is running.

|  |  |
| --- | --- |
| catissue.instance | The name of the instance of caTissue to which data should be loaded. This must match an entry in the keychain. |
| keychain.file | A file containing location and login information for instances of caTissue that you use. |
| data.file | Pathname of file containing data to load. This may be a relative pathname such as ./mydatafiles/newPatients.txt if you have a subdirectory called mydatafiles. |
| map.file | Pathname of xml file containing mapping of the datafile to caTissue data attributes |
| baddata.file | Pathname of text file into which bad records will be written. If this is not supplied a file will be saved in the same directory as the data file with bad prepended to the name. for example bad records for the file Tissuespecimens.txt would be saved to badTissuespecimens.txt |

## Directory structure

Conf

The default place to put the keychain file

Log4j configuration files would also go here

Mappings

Value type mappings for some standard terminologies. Currently a work in progress.

examples

Example data files and mappings described in this document.

## Manually loaded data

Some data required for data to be loaded must be first entered manually into caTissue Core via the user interface. For example:

Collection Protocol

Site

User

## Mapping document

This is an XML file which specifies

1. The column numbers in the datafile containing each attribute
2. Values in the datafile for a given column and a mapping to a value that is valid in caTissue.
3. Fixed values that should be used for columns not contained in the file, or which we want to load the same value for each entry.

It is possible to use the same column in the data file to load more than one attribute of caTissue. For example, specimens require both a collection date and a received date but only one date may be available in the file

The date format is specified in the mapping file e.g. as “mm/dd/yyyy” format for dates like 11/03/1994.[[1]](#footnote--1)

If an external mapping is used its path and name should be given relative to the mappings subfolder.

## Joining parent records

Almost all records added to caTissue Core need to be linked to one or more existing objects in the system. For example, a specimen must be linked to a specimen collection group.

The following list identifies for each object type being added what the existing object must be and how to link to it.

|  |  |  |
| --- | --- | --- |
| **Object being loaded** | **Object it links to** | **Link by** |
| Participant | Loading a participant also registers them to the specified collection protocol. Use the name of the collection protocol in the file. | **protocolTitle** |
| Collection group | Each collection group and specimen row record must specify the participant and protocol to which it should be added. Provide the protocol title and the protocol participant identifier. | **protocolTitle**  **PPI** |
| Specimen | Each collection group and specimen row record must specify the parent to which it should be added. Provide the SCG name. | **SCGName** |
| Collection Protocol Event | Use the label of the collection protocol event | **calendarPoint** |
| User | Use the email address of the User | **userEmail** |
| Site | Use the site name | **siteName** |

## Further examples

### Participant

The patient2Cols.txt file and patient2ColsMap.xml illustrate how Participants might be imported where only a gender and a medical record number are available.

The FullPatients.txt example with its associated mapping file (FullPatientTextMap.xml ) demonstrates loading data containing all participant attributes.

### Group and Specimens

See SpecimenTextMap.xml in the textdata folder for a mapping that could be used for the SpecimenText4.txt file and others. This example includes a mapping from colloquial names for container types such “Red top” to the valid container types.[[2]](#footnote-0)

### Group only (not done yet)

### Child specimens

See the childSpecimenTextMap.xml in the textdata folder for a sample mapping and the childSpecimenText.txt data file.

## Mapping files

### Case sensitivity

So that it is not necessary to map all case variants of a value the ignoreCase attribute allows you to specify whether values that match the fromValue in every respect but upper/lowercase should be allowed.

For example a particularly disorganized, but maybe not unrealistic, data file might have all the following values for gender.

F

female

male

Female

m

M

The following mapping would map both f and F to the valid caTissue value for Female gender as well as male and Male to the value without having to specify all these case variants individually.

<column attributeName="Sex" columnNo="3">

<valueMap fromValue="F" ignoreCase="true" toValue="Female Gender"/>

<valueMap fromValue="M" ignoreCase="true" toValue="Male Gender"/>

<valueMap fromValue="Female" ignoreCase="true" toValue="Female Gender"/>

<valueMap fromValue="Male" ignoreCase="true" toValue="Male Gender"/>

</column>

### Editing the mapping document

Mapping documents may be edited in any text editor. There are benefits to using an XML editor such as XMLSpy on Windows or Oxygen on Mac OS X. These, and tools like them, use the schema to help edit the document. They will autocomplete the right tags in the right places and can be used to verify that the file matches the schema.

### Date format

The <dateFormat> element in the mapping file should be specified as a string recognized by the java class java.text.SimpleDateFormat. See the documentation the class for details. Currently a single date format is used for all dates and time attributes in the file.

[***http://java.sun.com/j2se/1.5.0/docs/api/java/text/SimpleDateFormat.html***](http://java.sun.com/j2se/1.5.0/docs/api/java/text/SimpleDateFormat.html)

### Checking the mapping document

At the command line type

ant validateMapping

Note that the result of a bad document can be very verbose. Do not be alarmed, the fix is usually a simple one.

For example, this is generated by a single misspelling of a single attribute name in the mapping file!

validateMapping:

/Users/ian/dev/caTissueCoreLoader/textdata/SpecimenTextMap.xml:19:57: cvc-enumeration-valid: Value 'speccCollTime' is not facet-valid with respect to enumeration '[availableQuantity, birthDate, calendarPoint, clinicalDiagnosis, clinicalStatus, collectionProcedure, container, deathDate, ethnicity, firstName, gender, groupActivityStatus, groupCollTime, groupRecTime, label, lastName, lineage, medicalRecordNumber, middleName, parentSpecimenLabel, participantActivityStatus, pathologicalStatus, PPI, protocolTitle, quantity, race, race1, race2, race3, race4, receivedQuality, regActivityStatus, SCGName, sexGenotype, siteName, socialSecurityNumber, specCollTime, specimenActivityStatus, specimenAvailable, specimenType, specRecTime, tissueSide, tissueSite, userEmail, vitalStatus]'. It must be a value from the enumeration.

/Users/ian/dev/caTissueCoreLoader/textdata/SpecimenTextMap.xml:19:57: cvc-attribute.3: The value 'speccCollTime' of attribute 'attributeName' on element 'column' is not valid with respect to its type, 'attributeName'.

/Users/ian/dev/caTissueCoreLoader/textdata/SpecimenTextMap.xml is not a valid XML document

This particular error is verbose because the error lists all valid attribute names. A large number of errors in the document might result in pages of output. Take a deep breath, this can be fixed! Look at each error carefully, there is useful information in this torrent!

Alternatively, if you use a good XML editor with support for XML Schema (e.g. XMLSpy or Oxygen) to create your mapping document checking your documents will be less traumatic. In fact really good XML editors, including those above, check against the schema as you type – which helps avoid creating invalid documents in the first place.

### Mapping document schema

There is a W3C XML Schema for the mapping document in the xml folder. This is what is used to validate the mapping files as described.

While the schema currently does validate attribute names it does not check all the necessary logic required by the loader. For example, mandatory attributes are not checked for. So please take care when creating mapping documents and use the

### Saved value mappingss

Not completed – yet.

This feature will allow commonly used terminologies to be saved as mappings so users of this tool can share mappings for those terminologies.

These must be stored in the mappings folder. Their names may contain spaces. The files may be hierarchically organized in different folders – for example according to subject such as ethnicity, diagnosis etc.

## Attribute names

The following tables list names must be used in the mapping file to describe the attribute in caTissue into which any particular value should be loaded.

The following tables only have definitions for some of the attributes. Many of them are fairly evident. To supplement this use the caTissue Object model. An html extract of this model showing definitions of most attributes and associations is included in the docs folder as “caTissue Core Model Extract.html”.

To determine valid values for any attribute browse the caTissue web interface and review the appropriate drop down menu.

### Participant & Registration

|  |  |  |
| --- | --- | --- |
| ***Attribute name*** | ***Required Y/N*** |  |
| **birthDate** | N |  |
| **gender** | N |  |
| **firstName** | N |  |
| **lastName** | N |  |
| **participantActivityStatus** | Y |  |
| **middleName** | N |  |
| **vitalStatus** | N |  |
| **deathDate** | N |  |
| **sexGenotype** | N |  |
| **socialSecurityNumber** | N |  |
| **ethnicity** | N |  |
| **race** | N |  |
| **race1**  **race2**  **race3**  **race4** | N | To include multiple races enter them in the data file in separate columns. Label them in the mapping file as race1, race2, race3 or race4. If you need more than 4 – modify the source code – it’s an easy change. |
| **medicalRecordNumber** | N |  |
| **siteName** | N – but required if medicalRecordNumber present |  |
| **protocolTitle** | Y |  |
| **PPI** | Y |  |
| **regActivityStatus** | Y |  |

### Specimen Collection Group

|  |  |  |
| --- | --- | --- |
| siteName | Y | The site must already be in caTissue |
| groupActivityStatus | Y |  |
| clinicalDiagnosis | Y |  |
| SCGName | Y |  |
| clinicalStatus | Y |  |
| protocolTitle | Y |  |
| calendarPoint | Y |  |
| userEmail | Y |  |
| collectionProcedure | Y |  |
| container | Y |  |
| groupCollTime | Y |  |
| receivedQuality | Y |  |
| groupRecTime | Y |  |

### Specimen

|  |  |  |
| --- | --- | --- |
| Attribute | Required Y/N |  |
| specimenType | Y | See Appendix 1 for valid values |
| specimenAvailable | Y |  |
| label | Y |  |
| pathologicalStatus | Y |  |
| tissueSide | Y |  |
| tissueSite | Y |  |
| lineage | Y |  |
| quantity | Y |  |
| availableQuantity | Y |  |
| SCGName | Y | Name or label for the Specimen Collection Group in which this specimen was collected |
| parentSpecimenLabel | N | Label of the specimen from which this specimen was derived, if any. |
| specimenActivityStatus | Y |  |
| userEmail | Y | Identifies the user who collected and received the specimen – the user must must already be defined in caTissue. |
| collectionProcedure | Y | Method of specimen collection from participant (e.g. needle biopsy, central venous line, bone marrow aspiration) |
| container | Y | Container type in which specimen is collected |
| specCollTime | Y |  |
| receivedQuality | Y | Grossly evaluated quality of the received specimen |
| specRecTime | Y |  |

## Race

To allow multiple races to be associated with a participant these should be in separate columns on the same row. These may be named “race1”, “race2”, “race3”, “race4”.

### Mapping based on another value

## caTissue keychain

This is to prevent storing usernames and password in code files. The file should be edited with the url, username and password of caTissue instances which you wish to connect.

The following is an example of a keychain file included in the conf directory.

<?xml version="1.0" encoding="UTF-8"?>

<caTissueKeyChain xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:noNamespaceSchemaLocation="../xml/caTissueKeychain.xsd">

<caTissueInstance

name="My Production Instance"

host="http://catissue.myplace.edu:8080"

username="auser@myplace.edu"

password="Login123"

version="1.2"

/>

<caTissueInstance

name="My Production Instance"

host="http://192.168.0.20:8080"

username="auser@myplace.edu"

password="Login123"

version="1.2"

/>

</caTissueKeyChain>

Note that the host name is just that – the URL of the host. This is not the same thing as the URL of the caTissue application. The loader program will build the URL from the hostname.

The URL at which caTissue will be accessed is

*hostname*/catissuecore/http/remoteService

That is the URL that is automatically created when you deploy caTissue on JBoss. For reference regular users of the application would be accessing the application at

*hostname*/catissuecore

### Keychain schema

There is a W3C XML Schema for keychains in the xml folder

## Exporting data files from applications

The following notes describe things you should look out for when exporting data from various applications in order to create text files to be imported by this loader.

### Microsoft Excel

Use *Save as…* from the *File* menu – and select the appropriate file format

Tab delimited text

Text – usually a synonym for the above

Comma separated values

Format dates and timestamps to match the format that was specified in the mapping file.

See the Excel on line help on how to do this by defining a custom number format. Note that when you open and save a text file in Excel it may lose the date formatting. In which case, open the file reapply the formatting and save again as text.

For troubleshooting, rely on a text editor to show the real contents of a text file, but make sure it doesn’t replace tab characters with spaces.

### Microsoft Access

Contributions here please! Not difficult though.

### FileMaker Pro

Contributions here please! ! Not difficult though.

## Third party software

The following third party software is used by caTissue Core Loader

### JAXB

JAXB from Sun Microsystems Inc. allows the simple manipulation of XML files by creating appropriate bindings. JAXB 1.0.3 is included here from the Java Web Services Development Pack 1.4. The following jar files are included in the lib folder. See the licenses folder for the JAXB license agreement.

JAXB JAR files (jaxb-api.jar, jaxb-impl.jar, jaxb-libs.jar, jaxb-xjc.jar)

JAXP 1.1 or higher JAR files (jaxp-api.jar, sax.jar, dom.jar, xalan.jar)

JWSDP Shared JAR files (jax-qname.jar, namespace.jar, xsdlib.jar, relaxngDataype.jar)

### FlatPack

FlatPack is a flexible set of tools for parsing flat files of various types. caTissueCoreLoader uses a single class from FlatPack to read a line from a text file.

Full details of FlatPack are available at http://flatpack.sourceforge.net/.

## Known lack of robustness

The following issues with the program mean that data must be loaded with care.

Currently the program relies on there being mappings for all the required attributes but does nothing to ensure this is the case.

Precaution: Check required attributes listed in the tables in this document are present for the types of objects being loaded.

If a row will result in the creation of more that one object in the database and insertion of the second or subsequent objects fails the first loaded objects for that row will remain in the database. For example, if a Participant is added along with a medical record number from each row – as long as the Participant details are valid the Participant object will be created even if the medical record number were invalid.

Precaution: Check to ensure you will not be loading duplicate medical record numbers. The same applies for protocol participant identifiers.

No elegant failure on bad lookups e.g. site, protocol title

## Possible improvements

These should be added to GForge

Specify date format – done – one format for the whole input format

Could do a format on an attribute by attribute basis.

Might at least need a format for timestamps versus dates

Parse a text file and identify unmapped values. Save them to a mapping document.

Map and load from a JDBC database.

Adding specimen locations

Concatentations of columns

e.g. of a date and time in separate columns

Keep standard mappings for individual CDEs separate from full file mapping documents.

Make these user creatable

Provide mappings for commonly used data standards

In UI make these available from picklists.

Report all errors that would prevent a row being loaded – not just the first that causes a failure.

Add validation of XML files prior to using them. Bale out if a problem.

Provide choice of fields which may be used to map to parent rows. For example, map to a participant via medical record number rather that via PPI.

Allow child specimens to be imported. – done – 1 hour including generating test data and unit testing.

Allow for institutions that might have reconfigured the deployment that results in non-standard URLs for the API.

Add a standard mapping for colloquial names

#### UI

Display data in a grid

Assume each row represents a row in the source data.

Highlight good data in one colour (red) , bad data in another (red)

Mark whole rows as good or bad

Indicate whole data set as good or bad

Provide an issues list

List should not repeat instances of the same issue. i.e. if every gender value is bad because the input file simply uses “M” and “F” instead of “Male Gender” , “Female Gender” that issue should be reported once rather than multiple times.

“Next” button to allow quick navigation to next bad row/cell.

Add some classification to key chain data

e.g. to indicate production, test, demo instances

Allow columns to be specified by letters like in Excel.

Allow columns to be specified by name

#### This document

Needs re-review to check its style and content is appropriate for its audience.

Update with details for excelCSV.

### Get data over the web

web/grid services/ remote APIs

Data mappings

Following the model

From caDSR or other EVS services/APIs

Keychains

Not the username and passwords – but a web service to provide URLs and other classification information about caTissue instances.

## Test cases

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Data file | Check |
| 1 | Extra columns in file |  |  |
| 2 | Attributes in code not in map |  |  |
| 3 | Short lines i.e. fewer columns |  |  |
| 4 | Excel CSV |  |  |
| 5 | Double quotes in data |  |  |
|  | Single quotes in data |  |  |
| 6 | Commas in data |  |  |
| 7 | Case insensitivity on valid values (i.e. unmapped columns) |  |  |
| 8 | Valid values in an otherwise mapped column | Fullpatients.txt |  |
| 9 | All participant columns | Fullpatients.txt |  |
| 10 | Loading of timestamps |  | Need to check that hh, mm and ss can be specified in the date format and correctly loaded. |

## Requirements

The following are required for the program to run and are included in the lib folder

JAXB.

## Notes on design decisions

### Types of objects that may be loaded

This version was written to load the types of data loaded in bulk as opposed to those which are added less frequently. For example, while specimens and participants can be imported, collection protocols cannot. Collection protocols should be added manually via the usual use interface.

The program is structured in such a way as to make adding other object types easy. Look at methods like addSpecimen and addParticipant to work out how to add other methods like addCollectionProtocol if you want to.

### Granularity

The program makes some simplifications of the caTissue object model in order to maintain a simple view of the data from a user point of view. E.g. how it handles multiple races, and the simplification that says there is only one medical record number for a patient.

### Notes for Suite 1.0

Consent import will need to be added.

## Building the distribution

To prevent extra downloads the following are included in the lib folder. These files are used for JAXB 1.0.3 which is used to provide simple programmatic access to the XML files used by the loader (the keychain and the mapping files). See JAXB for more information.

JAXB JAR files (jaxb-api.jar, jaxb-impl.jar, jaxb-libs.jar, jaxb-xjc.jar),

JAXP 1.1 or higher JAR files (jaxp-api.jar, sax.jar, dom.jar, xalan.jar),

JWSDP Shared JAR files (jax-qname.jar, namespace.jar, xsdlib.jar, relaxngDataype.jar)

## Appendix 1 - Valid values for specimen type

|  |
| --- |
| RNA, poly-A enriched |
| RNA, nuclear |
| protein |
| cDNA |
| RNA |
| DNA |
| RNA, cytoplasmic |
| Total Nucleic Acid |
| Whole Genome Amplified DNA |
| Frozen Cell Pellet |
| Fixed Cell Block |
| Frozen Cell Block |
| Cryopreserved Cells |
| Slide |
| Whole Bone Marrow |
| Saliva |
| Plasma |
| Body Cavity Fluid |
| Milk |
| Pericardial Fluid |
| Lavage |
| Whole Blood |
| Vitreous Fluid |
| Gastric Fluid |
| Bone Marrow Plasma |
| Urine |
| Serum |
| Amniotic Fluid |
| Cerebrospinal Fluid |
| Bile |
| Synovial Fluid |
| Sweat |
| Feces |
| Sputum |
| Fixed Tissue |
| Fixed Tissue Block |
| Frozen Tissue Block |
| Fixed Tissue Slide |
| Frozen Tissue Slide |
| Fresh Tissue |
| Microdissected |
| Frozen Tissue |
| Molecular Not Specified |
| Cell Not Specified |
| Fluid Not Specified |
| Tissue Not Specified |

1. The string is that required by a Java DateFormat class. See [↑](#footnote-ref--1)
2. Note that these are not actual mappings – just examples. I have no idea that

   "Red Top” actually corresponds to "EDTA Vacutainer". [↑](#footnote-ref-0)