Genny XML schema Generation Toolset

User Guide

James C. McDonald
http://www.mcguru.net/

Genny XML schema Generation Toolset: User Guide

by James C. McDonald

1st Draft Edition
Published August 2004

Notice:

Revision History

Revision 0.1.0	07/27/04	Revised by: JCM
1st Draft based on Genny v0-1-9		

Table of Contents

List of Tables

List of Examples

Chapter 1. Overview

Genny is a python application that generates DTDs, xsd schemas, SQL DDL, and other items from a Logical Data Dictionary (LDD). A user creates the LDD for a specific XML transaction or message as a Microsoft Excel spreadsheet with a specific set of columns.

Genny was developed as a tool to assist in the work that is being done in the *MISMO* (http://www.mismo.org/) (Mortgage Industry Maintenance Standards) Servicing Work Group.

Genny Overall Task Flow

1. Create a new Genny project directory.
2. Create Microsoft Excel LDD Spreadsheet using Genny template spreadsheet.
3. Save Microsoft Excel LDD Spreadsheet as tab delimited text file.
4. Adjust configuration parameters and template text files to customize generated outputs.
5. Execute genny to generate outputs.
6. Modify Spreadsheet and repeat steps 2-6 until satisfied.

Genny is not an LDD Repository nor does it contain any repository functionality. Each project Genny helps to automate the work of creating and maintaining schemas and documentation related to a single type of XML instance document.

Chapter 2. Installation

2.1. Software Prerequisites

- *Python* (http://www.python.org/) version 2.3 or higher.
- Any operating system in which Python version 2.3 or higher is available (e.g. Win32, Linux, Mac OS X, Solaris, *BSD, etc.).
- Zip Utilities: **unzip** (or its equivalent) to unzip the distribution archive.

Genny has been tested on Win2K Professional, Mac OS X 10.3 Panther, RedHat Linux v8.0, FreeBSD v5.2.1, and Sun Solaris 8.

2.2. Where to Download the Software Prerequisites

- Win32: A Windows installer may be downloaded from *http://www.python.org* (http://www.python.org/). There is also an installer for python win32 com extensions.
- Mac OS X *Panther* (http://www.apple.com/macosx/developertools/) : includes the necessary Python interpreter.
- FreeBSD: please install the *"lang/python"* (http://www.freebsd.org/cgi/url.cgi?ports/lang/python/pkg-descr) port.
- Linux: most recent distributions include a Python 2.3 environment.
- Solaris: pre-built Solaris packages may be downloaded from *http://www.sunfreeware.com* (http://www.sunfreeware.com/).
- If you can't find a pre-built python distribution for your platfrom, source code for Python version 2.3 or higher is available at *http://www.python.org/* (http://www.python.org/).

2.3. Installation of Genny distribution

Genny is distributed as a Zip archive.

To install Genny, simply **unzip** the distribution zip archive in whatever directory you would like Genny to be installed into.

Example 2-1. Unzipping the Genny distribution archive

```
$ unzip genny-dist-filename.zip
```

where genny-dist-filename.zip is the name of the distribution file.

If your're installing an updated version of Genny copy your projects from the old Genny installation directory to the new one.

Example 2-2. Copying your Genny projects from previous installation dir

```
$ cp -R /path-to-old-Genny/projects/* /path-to-new-Genny/projects/
```

Chapter 3. Genny's Directory Structure

3.1. Overall Directory Structure

In the Genny application directory you will find the following directories:

- `./docs/` : contains the Genny documentation and LDD template spreadsheet
- `./docs/dtd/` : contains the Genny LDD DTD
- `./docs/srcdoc/` : contains the Genny program source documentation
- `./imports/` :contains the Genny program python modules
- `./projects/` : contains the Genny project directories

3.2. Project Directories

All paths are relative to the installation directory.

Each project has a name and that name will be a subdirectory under the `./projects/` directory. To keep your sanity, don't use spaces in the project name.

Example 3-1. Project Name subdirectory

If the name of the project is TEST-PROJECT, then that project's directory is `./projects/TEST-PROJECT`.

When you install a new version of Genny you'll need to copy your projects from the old Genny installation directory to the new one.

The rule of thumb is that inputs are placed in the `./projects/project-name/input/` directory and outputs are generated in the `./projects/projectname/output/` directory.

All the files in the `./projects/projectname/output/` directory may be recreated based on the contents of the `./projects/project-name/input/` directory.

3.3. Input Directories

Under each projects directory are the following directories

- `./projects/project-name/input/` : contains all of the input files that Genny uses (Tab delimited Text LDD, XML LDD, boilerplate text, etc.)
- `./projects/project-name/input/docbook-components` : contains individual text files each with the same name as the CONTAINERs in the defined in the project's LDD. These files should contain valid docbook formatted XML. These text files get merged with the LDD information to create the IGuide chapter in docbook format.

3.4. Output Directories

- `./projects/project-name/output` : contains the primary output generated by Genny, this includes DTDs, HTML LDD, schemas, SQL DDL, etc.
- `./projects/project-name/output/ele/` : contains the MISMO "ele" style DTD fragments that are generated. One "ele" file is created for each container defined in the LDD.
- `./projects/project-name/output/logs/` : as Genny does its work a set of log files are created in this directory.
- `./projects/project-name/output/templates` : contains a currently incomplete set of programming related templates
- `./projects/project-name/output/templates/python/` : some initial work to auto generate a python class that knows how to produce and consume XML based on the LDD.
- `./projects/project-name/output/templates/cobol/` : some incomplete work on auto generating cobol FD lineups from the LDD.

Chapter 4. Tasks

4.1. How To Create a New Project

To create a new project, execute a command like the following.

```
$ python genny.py PROJECT-NAME --new
```

where PROJECT-NAME is the name of the project.

A new project directory of `./projects/PROJECT-NAME/` and the template input files will be created.

One may have an unlimited number of projects, making it easy to work on several transactions concurrently, with each project having its own set of directories.

4.2. How To Build a Genny LDD Spreadsheet

4.2.1. Using the Default Template

An empty Template can be found in the `./docs/` directory called `GennyLDDMap-template.xlt`. Open the file and "Save As" the template to an Excel spreadsheet in the `./project-name/input/` directory for your project. The filename that you choose will be the primary name that is use when creating the output files so choose something descriptive. For example something like `TransactionName-YYYY-MM-DD-RC2.xls` .

4.2.2. Populating The Spreadsheet

Enter the information about your Containers and Data Items into the spreadsheet to define the transaction. The columns in the template are defined in the included Appendix.

Please keep in mind the following guidelines:

- All rows must be sorted by Container Name (all data items for a given container must be consecutive in the spreadsheet).
- All the rows must have a Container Name and a Data Item Name.
- The order of the spreadsheet will be the order of the output.
- The Parent and Child Container lists are what provides the structure.
- The valid data types are (enumerated,Alphanumeric,DateTime,Date,Money,Numeric,xsd:Boolean,Fixed,ID,IDREF,IDREFS).

4.2.3. Correcting Errors

If the data entered into the spreadsheet has errors, then your output will not be correct, and it may not be able to be processed by Genny.

DO NOT include Cariage Returns or Tabs Cells in your spreadsheet. This will prevent the tab-delimited spreadsheet from being processed.

If the structure of your document is not coming out as you intended, please check the Parent and Child Container lists and verify that they look Ok.

If the data type of a Data Item doesn't look correct, please check the data type entered for that data item.

Column AK - should always contain "EOL" for each line. This helps work around inconsistencies in how Excel creates the tab-delmited file. The Column A "update" button will do this automatically for you, but if you're having a problem you might want to verify that every row has "EOL" in column AK.

4.2.4. Saving the Excel file as a Tab Delimited Text File

It is recommended that you use the "Update" button in Column A of the template to renumber the spreadsheet prior to saving it as a tab-delimited text file.

In Excel save the completed worksheet as a Tab-demlited text file with FILENAME.txt in the `./projects/PROJECT-NAME/input/` directory. You will get warnings from Excel about how you may lose formatting, and that some features are not supported in this format. Those warnings may be ignored.

4.3. How To Generate Outputs from the LDD Spreadsheet

Once you have your tab-delimited LDD text file saved, the next step is to generate your output. To generate the output, execute a command like the following.

```
$ python genny.py PROJECT-NAME FILENAME
```

where PROJECT-NAME is the name of the project, and FILENAME is the name of the tab delimited LDD input file without the .txt file extension.

If all goes well you will see a bunch of messages scroll by showing all of the processing that is being done to produce the output documents.

4.4. How To Customze the Output

4.4.1. How To Include a Set of Containers that are not in the LDD Spreadsheet

If the file `ADD-TO-GENNYLDD.xml` is present in the `./project-name/input/` directory, then the contents of that file will be included in the FILNAME.xml Genny LDD that is used to generate the rest of the outputs. The `ADD-TO-GENNYLDD.xml` must be a valid XML fragment in Genny LDD format of a set of Containers.

This feature is useful when one is concerned with building a new transaction that "wraps" and existing structure and no changes are planned to the existing structure. The containers of the existing structure may be included in the `ADD-TO-GENNYLDD.xml` file and do not need to be in the spreadsheet.

4.5. How To Cleanup a Project's output

To cleanup all output in a project, execute a command like the following.

```
$ python genny.py PROJECT-NAME --cleanup
```

where PROJECT-NAME is the name of the project.

All of the created output files will be deleted. This is especially useful between major changes to the LDD and/or changes to the FILENAME that is used as the input LDD.

4.6. Command Reference

Genny has a command line interface. The following command line options are available:

python genny.py { *PROJECT-NAME* } {--new | --cleanup | *FILENAME* }

where PROJECT-NAME is the name of the project, and FILENAME is the name of the tab delimited LDD input file without the .txt file extension.

Example 4-1. Creating a New Project

```
$ python genny.py PROJECT-NAME --new
```

where PROJECT-NAME is the name of the project to be created.

Example 4-2. Removing all output from a project's output directories

```
$ python genny.py PROJECT-NAME --cleanup
```

where PROJECT-NAME is the name of the project.

Example 4-3. Generating all output documents for a project

```
$ python genny.py PROJECT-NAME FILENAME
```

where PROJECT-NAME is the name of the project, and FILENAME is the name of the tab delimited LDD input file without the .txt file extension.

Chapter 5. Genny's Outputs

5.1. DTDs

5.1.1. Commented and Uncommented Standard Names

Genny will produce a DTD in the MISMO style where data items are expressed as XML attributes, and Containers are expressed as XML elements. The spreadsheet container and data item names are converted into the appropriate MISMO attribute and tag names automatically by Genny. The spreadsheet Common Valid Values are also converted according to MISMO v2.x design guidelines to the corresponding enumerated values.

Both commented and uncommented versions of the DTDs are produced.

5.1.2. Commented and Uncommented Short Names

As per the MISMO v2.x design guidelines, a "Short Name" version of the DTDs will be produced. The contents of Column AJ is used for the short name. If a short name is missing for a data item an error is flagged in the DTD.

XSLT transformations from Short Name to Long Name and Long Name to Short Name are also produced.

5.1.3. ELEs

A separate DTD fragment file for for each container is created in the `./outputs/ele/` directory.

5.2. W3C Schemas (WXS)

W3C Schemas that are equivalent to the corresponding DTDs are produced. Definitions are included as annotations in the schemas.

5.3. Docbook Implementation Guide Chapter 3

In order to assist in the preparation of an implementation guide for the transaction, a docbook xml format file is produced that contains a chapter that documents the XML structure that may be included in a docbook format implementation guide.

5.4. HTML LDD

A HTML format LDD is producted to document the transaction

5.5. SQL DDL

SQL DDL is produced that attempts to model the XML transaction. The SQL DDL may be "imported" into an ER modeling tool to produce an ER Model of the transaction.

Appendix A. Genny's LDD Spreadsheet Column Format

The columns in the LDD spreadsheet correspond directly to the LDD XML format.

Table A-1. Genny LDD Spreadsheet Column Descriptions

Column	Name	Definition
A	Line Number	Not Used by Genny
B	Container Name	Name to use for the defined container.
C	Container Description	Text defintion for the defined container.
D	Data Item Name	Name to use for the defined container.
E	Data Item Definition	Text defintion for the defined data item.
F	Data Item Common Valid Values	Used for data items whose type is enumerated. Enter the Enumerated Value.
G	Data Item Common Value Definition	Text definition for the Data Item Common Valid Value
H	Data Item Data Type	Data Type of the defined data item. The valid data types are (enumerated, Alphanumeric, DateTime, Date, Money , Numeric, xsd:Boolean, Fixed, ID, IDREF, IDREFS).
I	IDREF Reference / Fixed Value / Default Value	
J	Container Sequence	Not Used by Genny
K	Parent Containers	This column is filled in for the first data item in a new container, and is a comma delimited list of the names (in uppercase) of the containers that are the parents of the container that you are defining e.g. the parent of the PAYEE container is

Column	Name	Definition
		PAYEEDATA. The Root container MUST have a Parent of "Root".
L	Child Containers	 is filled in for the first data item in a new container, and is a comma delimited list of child containers (in uppercase) of the container that you are defining where each item in the list is of the form CONTAINER_NAME:x:y where x:y represents the cardinality of each child container. x may be 0 or 1 y may be 1 or u So 0:1 is zero or one (? in the DTD) 1:1 is exactly one 0:u is zero or unbounded (* in the DTD) 1:u is one or unbounded (+ in the DTD) e.g. for the SERVICING_TRANSFER container list is CONTACT_DETAIL:0:u,PAYEEDATA:0:1,INVESTORDATA:0:1,POOLDATA:0:1,LOANDATA:0:1,VERIFICATIONDATA:0:1 This column is critical to getting the desired output from Genny.
M	Container Default Min Occurs	The minimum number of occurrances of the container, valid values are 0, 1, or unbounded.
N	Container Default Max Occurs	The maximum number of occurrances of the container, valid

Column	Name	Definition
		values are 1, or unbounded.
O	Container Primary Key Item Name	Not Used by Genny
P	Container Implementation Note	Not Used by Genny
Q	Container Example	Not Used by Genny
R	Container Used By Process Area	Not Used by Genny
S	Container Doc Graphic Filename	Not Used by Genny
T	Data Item Sequence	Not Used by Genny
U	Data Item MinOccurs	Not Used by Genny
V	Data Item MaxOccurs	Not Used by Genny
W	Data Item Implementation Note	Not Used by Genny
X	Data Item Example	Not Used by Genny
Y	Data Item Used By Process	Not Used by Genny
Z	Data Item Data Mapping Note	Not Used by Genny
AA	Data Item Source Document	Not Used by Genny
AB	Data Item minLength	Not Used by Genny
AC	Data Item maxLength	Not Used by Genny
AD	Data Item regex pattern	Not Used by Genny
AE	Data Item retain whitespace indicator	Not Used by Genny
AF	Data Item numeric precision	Not Used by Genny
AG	Data Item numeric scale	Not Used by Genny
AH	Data Item min Value inclusive	Not Used by Genny
AI	Data Item max Value inclusive	Not Used by Genny
AJ	Data Item Short Name	Not Used by Genny
AK	EOL	Must be "EOL". Automatically entered by Template macro that renumbers the lines.
AL	Type and Class Validation	Not Used by Genny. Template macro that displays whether the data item name uses a valid MISMO class word.

Appendix B. Genny's LDD XML Format

```
<!-- ================================================================= -->
<!-- GennyLDD.dtd                                                      -->
<!-- Version 2.02                                                      -->
<!-- ================================================================= -->
<!--
                  v1.xx changes
        06/15/2001 changes to the process area element
        06/18/2001 switch to an element base implemetation
        06/20/2001 modified PROCESS_AREA to TRANSACTION_SCHEMA
                   and back to an all attribute implementation
        06/26/2001 added the PrimaryKeyItem
        07/09/2001 added datatypes for ID, IDREF, and Fixed
        07/15/2001 added UsedByProcess,DataMappingNote,SourceDocument
        07/18/2001 added datatype for IDREFS
                   added DocGraphicFilename to container as attribute
        07/23/2001 added IDREF references
        10/16/2001 added many datatypes and attributes to align the LDD
                   with w3c schema generation
        01/24/2002 added EnumeratedWithDefault to the list of valid data types

                   v2.xx changes
        04/19/2002 added elements to designate multiple parent containers,
                   and multiple child containers with cardinality specified
                   for each child container
        10/08/2002 v2.01 added DateTime data type
        07/25/2003 v2.02 added XMLShortName to DATA_ITEM
  -->
<!-- ================================================================= -->
<!-- GENNY_LDD                                                         -->
<!-- ================================================================= -->
<!ELEMENT GENNY_LDD (TRANSACTION_SCHEMA)>
<!ATTLIST GENNY_LDD VersionId CDATA #FIXED '2.02'>

<!-- ================================================================= -->
<!-- TRANSACTION_SCHEMA                                                -->
<!-- ================================================================= -->
<!ELEMENT TRANSACTION_SCHEMA ( CONTAINER* )>
<!ATTLIST TRANSACTION_SCHEMA Id              ID #REQUIRED>
<!ATTLIST TRANSACTION_SCHEMA ProcessArea     CDATA #IMPLIED>

<!-- ================================================================= -->
<!-- CONTAINER                                                         -->
<!-- ================================================================= -->
<!ELEMENT CONTAINER (
                      PARENT_CONTAINER*,
                      CHILD_CONTAINER*,
```

```
                    DATA_ITEM*
                  ) >
<!ATTLIST CONTAINER Id                    CDATA    #REQUIRED>
<!ATTLIST CONTAINER XMLName               NMTOKEN  #REQUIRED>
<!ATTLIST CONTAINER Definition            CDATA    #IMPLIED>
<!ATTLIST CONTAINER XMLSequence           CDATA    #IMPLIED>
<!ATTLIST CONTAINER PrimaryKeyItem        CDATA    #IMPLIED>
<!ATTLIST CONTAINER ImplementationNote    CDATA    #IMPLIED>
<!ATTLIST CONTAINER Example               CDATA    #IMPLIED>
<!ATTLIST CONTAINER UsedByProcessArea     CDATA    #IMPLIED>
<!ATTLIST CONTAINER DocGraphicFilename    CDATA    #IMPLIED>
<!ATTLIST CONTAINER DefaultMinOccurs      CDATA    #IMPLIED>
<!ATTLIST CONTAINER DefaultMaxOccurs      CDATA    #IMPLIED>
<!ATTLIST CONTAINER ParentContainerList   CDATA    #IMPLIED>
<!ATTLIST CONTAINER ChildrenContainerList CDATA    #IMPLIED>

<!-- ================================================================= -->
<!-- PARENT_CONTAINER                                                  -->
<!-- ================================================================= -->
<!ELEMENT PARENT_CONTAINER EMPTY >
<!ATTLIST PARENT_CONTAINER Id                CDATA    #REQUIRED>

<!-- ================================================================= -->
<!-- CHILD_CONTAINER                                                   -->
<!-- ================================================================= -->
<!ELEMENT CHILD_CONTAINER EMPTY >
<!ATTLIST CHILD_CONTAINER Id           CDATA    #REQUIRED>
<!ATTLIST CHILD_CONTAINER XMLMinOccurs CDATA    #IMPLIED>
<!ATTLIST CHILD_CONTAINER XMLMaxOccurs CDATA    #IMPLIED>

<!-- ================================================================= -->
<!-- DATA_ITEM                                                         -->
<!-- ================================================================= -->
<!ELEMENT DATA_ITEM ( DATA_ITEM_ENUM_VALUE*,
                      DATA_ITEM_ENUM_DTD_STRING?
                    ) >

<!ATTLIST DATA_ITEM Id                 CDATA    #REQUIRED>
<!ATTLIST DATA_ITEM XMLName            NMTOKEN  #REQUIRED>
<!ATTLIST DATA_ITEM XMLShortName       NMTOKEN  #REQUIRED>
<!ATTLIST DATA_ITEM XMLDataType (Enumerated|
                                 EnumeratedWithDefault|
                                 Alphanumeric|
                                 Numeric|
                                 Integer|
                                 Date|
                                 DateTime|
                                 Percent|
                                 Money|
                                 Boolean|
                                 Fixed|
                                 ID|
                                 IDREF|
```

```
                                IDREFS|
                                timeDuration|
                                recurringDuration|
                                binary|
                                uriReference|
                                NMTOKEN|
                                positiveInteger|
                                negativeInteger|
                                time|
                                timeperiod|
                                month|
                                year|
                                recurringDate
                                ) #REQUIRED>
<!ATTLIST DATA_ITEM Definition         CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM XMLSequence        CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM XMLMinOccurs       CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM XMLMaxOccurs       CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM ImplementationNote CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM Example            CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM UsedByProcessArea  CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM DataMappingNote    CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM SourceDocument     CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM IDREFReference     CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM FixedValue         CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM minLength          CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM maxLength          CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM regexPattern       CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM RetainWhitespaceIndicator  (true|false)   #IMPLIED>
<!ATTLIST DATA_ITEM numericPrecision   CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM numericScale       CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM minInclusive       CDATA    #IMPLIED>
<!ATTLIST DATA_ITEM maxInclusive       CDATA    #IMPLIED>

<!-- ============================================================= -->
<!-- DATA_ITEM_ENUM_VALUE                                          -->
<!-- ============================================================= -->
<!ELEMENT DATA_ITEM_ENUM_VALUE EMPTY >
<!ATTLIST DATA_ITEM_ENUM_VALUE Value       CDATA #REQUIRED>
<!ATTLIST DATA_ITEM_ENUM_VALUE Definition  CDATA #REQUIRED>
<!ATTLIST DATA_ITEM_ENUM_VALUE XMLSequence CDATA #IMPLIED>

<!-- ============================================================= -->
<!-- DATA_ITEM_ENUM_DTD_STRING                                     -->
<!-- ============================================================= -->
<!ELEMENT DATA_ITEM_ENUM_DTD_STRING EMPTY >
<!ATTLIST DATA_ITEM_ENUM_DTD_STRING Value  CDATA #REQUIRED>
<!ATTLIST DATA_ITEM_ENUM_DTD_STRING Count  CDATA #IMPLIED>
```

Appendix C. Potential Improvements / TO DO List

C.1. General

- Add a command line option "--validate-LDD" to validate LDD xml to Genny DTD
- Add a command line option "--convert-LDD" just to convert the Tab Demlimeted to XML
- Add file stat checks to only convert/regen LDD xml if it is out of date
- Check for required python version in code and quit if not correct
- Modularize / change the code to OO style
- Create a project preference file and parse / allow xml config xml to mirror a python dict with a simple parser->dict
- Automate packaging of Genny and/or convert to python disutils package
- Provide a way to specify the naming convention to use in the config file, and then use that option in the code
- Write a DTD, and code to parse a configuration. Parameter file, write the code to process the configuration file in genny.py
- Build a list of containers that have root as a parent, apply the sequence no, to the list, and process
- Build a list of containers for each contained container , apply the sequence no, to the list, and process until we have no more left
- Modify Genny to be xmlnamespace aware, add a parameter(s) to the GennyLDD
- Write code to support an all ELEMENT style of XML
- Improve error checking, Change all errors to a set of exceptions, and write to a different log file with an exception handler (one and only one Root container)
- checking for/ warning for duplicate tag/container names
- checking for/ warning of invalid characters in tag/container names
- separate the filter from txt to xml file from the generation (modular input)

C.2. Documentation

- Generate HappyDoc for python source documentation modules/classes
- Schema document: LDD mapped to schema class words
- Schema document: on genny architectural approach

C.3. DTD Generation

- Word wrap comments

- Better formatting of the enumerated lists
- Better layout/formatting of DTD
- for DTD gen process code in attribute style for each container set of container rows passed ELEMENT and list of contained elements from the object list including sequence and cardinality then output the attribute for each element in the sequence, apply a sort to the rest of the rows

C.4. Output

- DTD Generation:word wrap comments
- DTD Generation:better formatting of the enumerated lists
- DTD Generation:better layout/formatting of dtd
- DTD Generation: for dtd gen process code in attribute style for each container set of container rows passed ELEMENT and list of contained elements from the object list including sequence and cardinality then output the attribute for each element in the sequence, apply a sort to the rest of the rows
- XML Sample File Generation: ability to generate an xml skeleton
- LDD Documentation: html LDD document(s) including xml authority pictures for each container
- Schema Generation: bring back Enumerated value data types ?
- Schema Generation: bring back Attribute Groups for each of the element (containers)?
- UML/XMI Generation: start on XMI generation for UML model http://www.visualobject.com http://xmlmodeling.com/portal/index.jsp
- UML/XMI Generation: document on mapping mismo to UML models
- SQL Generation: Isolate handling of "DEFAULT" as a container name, way to specify SQL keywords that we will mangle new names for in SQL DDL
- SQL Generation: Add configuration parameter (generate sql for ORACLE| MYSQL)
- SQL Generation: Modify the sql generation to use proper data types to match the data type in the LDD, we need a structure (DICT?) that map rdbms datatype to GennyLDD datatype, just declare a dict and use in the print statement
- SQL Generation: Add a capability to look for and generically map SQL keywords in SQL generation
- SAX python: class/python class for object representation
- Java JAX: generate: java JAX data binding
- Graphic Generation: Idea about generating SVG XML that would correspond to a graphic representation of the containers, etc. http://sketch.sourceforge.net/index.html

Appendix D. Acknowledgements

- Genny includes software developed by Laurence Tratt. His "ASV.py" python module is used to parse the tab-delimited input files.
- Hamish B Lawson's "TextFormatter.py" python module provides some of the formatting capability for the DTD output.
- Craig Foote wrote all of the enhancements to Genny's Excel template.

Glossary

These terms are defined from Genny's point of view.

Container

A conceptual structure to enclose a set of related Data Items.

Data Item

A discrete, atomic value of business data.

Logical Data Dictionary

A collection of data items that are meaningfull for a specific business transaction. Contained in a single LDD spreadsheet or Genny LDD XML file.

Transaction

A collection of data items that are meaningful for a specific business purpose, contained in a single LDD spreadsheet or Genny LDD XML file. A transaction is expressed in an XML instance document that validates against a generated schema.