Customer Information Quality Specifications Version 3.0 – Package Overview

Public Review Draft 02

15 June 2007

Specification URIs:

This Version:
http://docs.oasis-open.org/ciq/v3.0/prd02/supp/ciq-package-overview-v3-prd2.html
http://docs.oasis-open.org/ciq/v3.0/prd02/supp/ciq-package-overview-v3-prd2.doc
http://docs.oasis-open.org/ciq/v3.0/prd02/supp/ciq-package-overview-v3-prd2.pdf

Previous Version:
N/A

Latest Version:
http://docs.oasis-open.org/ciq/v3.0/supp/ciq-package-overview-v3.html
http://docs.oasis-open.org/ciq/v3.0/supp/ciq-package-overview-v3.doc
http://docs.oasis-open.org/ciq/v3.0/supp/ciq-package-overview-v3.pdf

Technical Committee:
OASIS Customer Information Quality TC

Chair(s):
Ram Kumar (kumar.sydney@gmail.com)

Editor(s):
Ram Kumar (kumar.sydney@gmail.com)

Related work:
This version of the CIQ specifications replaces or supercedes:

- OASIS CIQ extensible Name Language (xNL) V2.0 Committee Specification
- OASIS CIQ extensible Address Language (xAL) V2.0 Committee Specification
- OASIS CIQ extensible Name and Address Language (xNAL) V2.0 Committee Specification

- OASIS CIQ extensible Customer Information Language (xCIL) V2.0 Committee Specification

Abstract:

This document provides an overview of the CIQ Specification V3.0 package that is available for download from the OASIS CIQ TC web site (http://www.oasis-open.org/committees/ciq).

Status:

This document was last revised or approved by the OASIS CIQ TC on the above date. The level of approval is also listed above. Check the current location noted above for possible later revisions of this document. This document is updated periodically on no particular schedule.

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the "Send A Comment" button on the Technical Committee's web page at www.oasis-open.org/committees/ciq.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (www.oasis-open.org/committees/ciq/ipr.php.

The non-normative errata page for this specification is located at www.oasis-open.org/committees/ciq.

Notices

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The names "OASIS", “CIQ”, “xNL”, “xAL”, xNAL”, “xPIL”, “xPRL”, “xCIL”, “xCRL” , “Genericode”, and “UBL” are trademarks of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see http://www.oasis-open.org/who/trademark.php for above guidance.

Table of Contents

1 About the Package

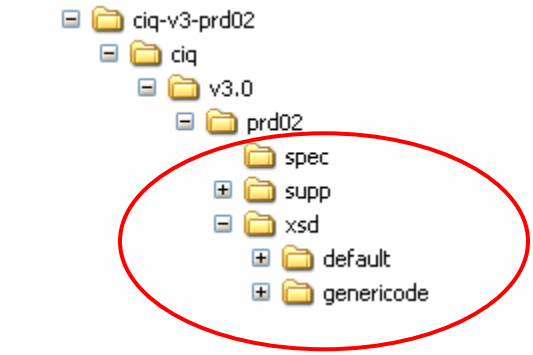
The purpose of this document is to assist users who have downloaded the CIQ Specifications Version 3.0 package from the OASIS CIQ TC web site (http://www.oasis-open.org/committees/ciq) to understand the contents of the package.

1.1 Name of the Package

The name of the package is “OASIS CIQ V3.0 PRD02.zip”.

1.2 Extracting the Package

Extracting the downloaded package creates the following directory structure:



Directory Name	Contents
spec	Contains the document describing the Name, Address, Name and Address, and Party specifications
supp	Contains supporting documents namely, introduction to CIQ TC, technical overview, release notes, this document, and technical FAQ
xsd	Contains the directory for CIQ entity XML schemas. Classified into two parts, 1. Default CIQ entity XML schemas using Option 1 of Code List, 2. CIQ entity XML schemas using Option 2 of Genericode based Code List
xsd/default	Contains default CIQ entity XML schema using Option 1 of Code List and XML schema documentation (HTML)and sample XML document instances for entities
xsd/genericode	Contains CIQ entity XML schema using Option 2 of Genericode based Code List and XML schema documentation and sample XML document instances for entities. Also contains all code lists represented using genericode, utilities to run the two pass validation, and batch/shell files to prepare

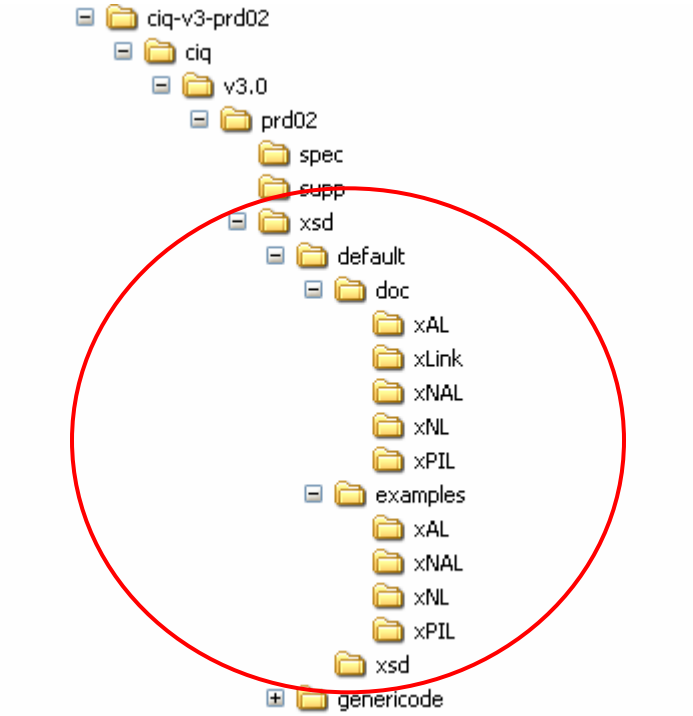
Directory Name	Contents
	two pass validation

1.3 CIQ Specification Entity XML Schemas using Default/Standard Code List Approach

CIQ Specification entity XML schemas are available in two types:

- One set uses default code list approach (Option 1 – all code lists are represented as XML schemas (*xNL-types.xsd, xAL-types.xsd*, *xNAL-types.xsd*, and *xPIL-types.xsd*) and "included" in entity XML schemas (*xNL.xsd*, *xAL.xsd*, and *xPIL.xsd*).
- The other set uses genericode based code list approach (Option 2 – all code lists are represented in genericode)

This section outlines the structure of Option 1. Users who are not interested in genericode based code list approach, should concentrate on this directory structure.

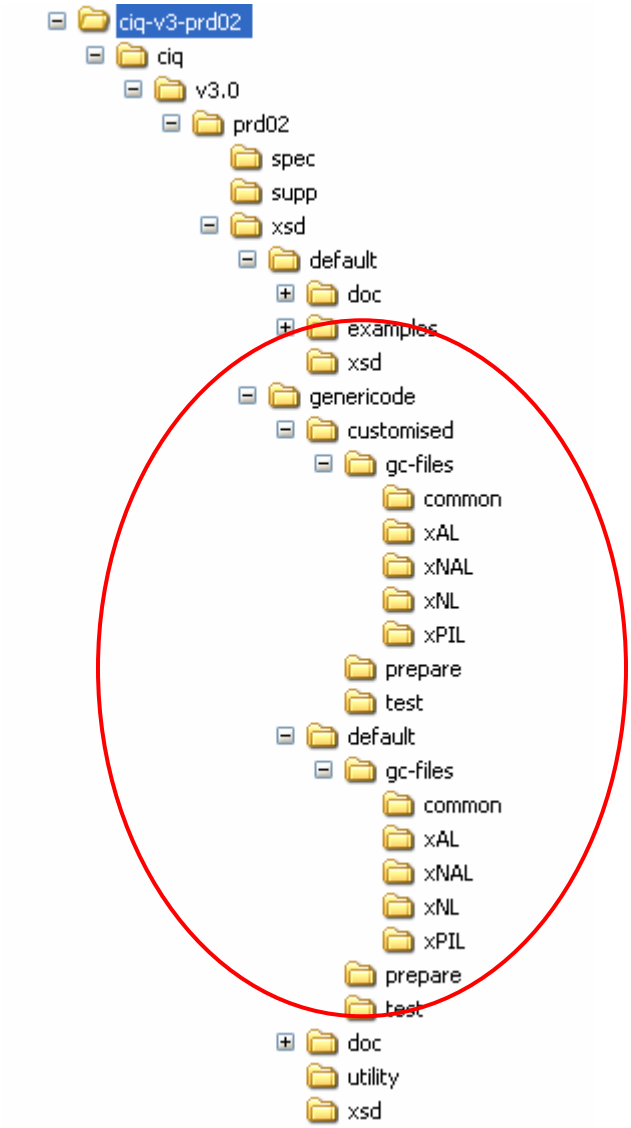


Directory Name	Contents
xsd/default/xsd	Contains the default entity XML schemas for Name, Address and Party. • **xNL.xsd** – xNL schema for Name entity. Users must not modify this file. • **xAL.xsd** – xAL schema for Address entity

Directory Name	Contents
	Users must not modify this file. • **xPIL.xsd** – xPIL schema for Party entity. Users must not modify this file. • **CommonTypes.xsd** - Schema reused by all the above entity schemas. Users must not modify this file. • **xNL-types.xsd** – Defines all code lists and values for xNL.xsd. Users can modify this file. • **xAL-types.xsd** – Defines all code lists and values for xAL.xsd. Users can modify this file. • **xNAL-types.xsd** – Defines all code lists and values for xNAL.xsd. Users can modify this file. • **xPIL-types.xsd** – Defines all code lists and values for xPIL.xsd. Users can modify this file.
xsd/default/doc	Provides HTML documentation for all default XML schemas (xNL, xAL, xNAL, xPIL, and xLink) in individual sub directories
xsd/default/examples	Contains XML sample files for Name, Address, Name and Address, and Party Schemas using Option 1 for code lists

1.4 CIQ Specification Entity XML Schemas using Genericode based Code List Approach

This section outlines the structure of Option 2. Users who are interested in genericode based code list approach, should concentrate on this directory structure.



Directory Name	Contents
xsd/genericode/default	This directory contains all default genericode files along with files for preparing genericodes and test files. Users should not modify files under this structure as everything is prepared for the user as part of this package. Users should only apply constraints on these default genericodes and this is done in a separate directory (xsd/genericode/customised)

Directory Name	Contents
xsd/genericode/default/gc-files	This sub-directory contains all the default genericode files to support CIQ Specification entity schemas namely, xNL, xAL, xNAL, and xPIL
xsd/genericode/default/common	Contains the common genericode code list files (2) used by Name, Address and Party XML schemas.
xsd/genericode/default/xAL	Contains the common genericode code list files (30) used by Address XML schema (xAL.xsd).
xsd/genericode/default/xNL	Contains the common genericode code list files (10) used by Name XML schema (xNL.xsd).
xsd/genericode/default/xNAL	Contains the common genericode code list file (1) used by Name and Address XML schema (xNAL.xsd).
xsd/genericode/default/xPIL	Contains the common genericode code list files (56) used by Party XML schema (xPIL.xsd).
xsd/genericode/default/prepare	This directory provides all files required to prepare the default genericode files. Users should not modify any of the files in this directory as it has already been prepared for this as part of this package. However, if the default genericode files are changed, then the files to prepare for validation should be used. • **prepare-default-cl.bat** – editable windows batch file to prepare the genericode representation of the default code lists for two pass validation. This is the “main” program that executes other batch programs. This file does not require modification by the user as it has been already updated. Users should define appropriate relative paths in this file if they change the default directory structures • **prepare-default-cl.sh** - editable unix/linux shell file to prepare the genericode representation of the default code lists for two pass validation. This is the “main” program that executes other shell programs. This file does not require modification by the user as it has been already updated. Users should define appropriate relative paths in this file if they change the default directory structures • **default-cl-constraints.cva** - lists all of the genericode expressions of agreed-upon default value list value enumerations, and lists all of the default contexts in which the value enumerations are used. All constraints for CIQ are already defined and requires no modifications to this file by the user. Users should define appropriate relative paths in this

Directory Name	Contents
	file if they change the default directory structures • **DefaultCodeList.sch** – Defines the default code list namespace constraints. This file requires no modification as all required constraints have been added. Users should define appropriate relative paths in this file if they change the default directory structures **Other files not listed above**– All the other files in this directory are automatically generated when prepare-default-cl.bat/sh file is executed and must NOT modified by user and so, do not touch them
xsd/genericode/default/test	This directory provides files to test the default genericode lists by performing two pass validations. Users can modify the .xml files to test different cases. Sample test files have been provided for users to test them. • **test-default-xnl.bat** – editable windows batch file to test xNL sample document instance (using default genericode based code lists) using two pass validation. Users should define appropriate relative paths in this file if they change the default directory structures • **test-default-xnl.sh** – editable Unix/Linux shell file to test xNL sample document instance (using default genericode based code lists) using two pass validation. Users should define appropriate relative paths in this file if they change the default directory structures • **test-default-xal.bat** – editable windows batch file to test xAL sample document instance (using default genericode based code lists) using two pass validation. Users should define appropriate relative paths in this file if they change the default directory structures • **test-default-xal.sh** – editable Unix/Linux shell file to test xAL sample document instance (using default genericode based code lists) using two pass validation. Users should define appropriate relative paths in this file if they change the default directory structures • **test-default-xnal.bat** – editable windows batch file to test xNAL sample document instance (using default genericode based code lists) using two pass validation. Users should define appropriate relative paths in this file if they change the default directory structures • **test-default-xnal.sh** – editable Unix/Linux shell file to test xNAL sample document

Directory Name	Contents
	instance (using default genericode based code lists) using two pass validation. Users should define appropriate relative paths in this file if they change the default directory structures • **test-default-xpil.bat** – editable windows batch file to test xPIL sample document instance (using default genericode based code lists) using two pass validation. Users should define appropriate relative paths in this file if they change the default directory structures • **test-default-xpil.sh** – editable Unix/Linux shell file to test xPIL sample document instance (using default genericode based code lists) using two pass validation. Users should define appropriate relative paths in this file if they change the default directory structures • **xAL-default.xml –** User modifiable sample test file for xAL default genericode based code lists • **xPIL-default.xml** - User modifiable sample test file for xPIL default genericode based code lists • **xNL-default.xml** – User modifiable sample test file for xNL default genericode based code lists • **xNAL**-default.xml - User modifiable sample test file for xNAL default genericode based code lists
xsd/genericode/xsd	Contains the genericode list based entity XML schemas for Name, Address and Party. Note: No *xNL-types.xsd*, *xAL-types.xsd*, *xNAL-types.xsd*, and *xPIL-types.xsd* exist in this directory as genericode approach for code list is used. • **xNL.xsd** – xNL schema for Name entity and is customised (extra attributes for genericode based code list metadata information) from the default xNL.xsd. Users must not modify this file. • **xAL.xsd** – xAL schema for Address entity and is customised (extra attributes for genericode based code list metadata information) from the default xAL.xsd. Users must not modify this file. • **xNAL.xsd** – xNAL schema for Name and Address entity and is customised (extra attributes for genericode based code list metadata information) from the default xNAL.xsd. Users must not modify this file. • **xPIL.xsd** – xPIL schema for Party entity and is customised (extra attributes for genericode based code list metadata information) from the

Directory Name	Contents
	default xPIL.xsd. Users must not modify this file. • **CommonTypes.xsd** - Schema reused by all the above entity schemas and is customised (extra attributes for genericode based code list metadata information) from the default CommonTypes.xsd. Users must not modify this file. • **xlink-2003-12-21.xsd** – Same schema as the default version and must not be modified by user.
xsd/genericode/utility	This directory provides a set of utility files to prepare genericode files such as XML parsers, creation of schematron patterns and XSLTs. • **prepare-cva.bat** – editable windows batch file to prepare context/value associations, and users are allowed to modify it to include relative paths if the default directory structure is changed and to turn documentation generation feature on or off (default is off). Do not run this file on its own. • **prepare-cva.sh** – editable Unix/Linux shell file to prepare context/value associations,and users are allowed to modify it to include relative paths if the default directory structure is changed and to turn documentation generation feature on or off (default is off). Do not run this file on its own. • **prepare-gc.bat** – editable windows batch file to prepare genericodes, and users are allowed to modify it to include relative paths if the default directory structure is changed and to turn documentation generation feature on or off (default is off). Do not run this file. • **prepare-gc.sh** – editable Unix/Linux shell file to prepare genericodes, and users are allowed to modify it to include relative paths if the default directory structure is changed and to turn documentation generation feature on or off (default is off). Do not run this file. • **twopass.bat** – editable windows batch file to perform two-pass structure/lexical validation and value validation, and users are allowed to modify it to include relative paths if the default directory structure is changed. Do not run this file. • **twopass.sh** – editable Unix/Linux shell file to perform two-pass structure/lexical validation and value validation, and users are allowed to

Directory Name	Contents
	modify it to include relative paths if the default directory structure is changed. Do not run this file. • **w3cschema.bat** – editable windows batch file that calls the appropriate java files to perform XML parsing. Users are allowed to modify this file to define the relative paths if the default directory structure is changed. Do not run this file. • **w3cschema.sh** – editable Unix/Linux shell file that calls the appropriate java files to perform XML parsing. Users are allowed to modify this file to define the relative paths if the default directory structure is changed. Do not run this file. • **xslt.bat** - editable windows batch file that calls the appropriate java file to create XSLT. Users are allowed to modify this file to define the relative paths if the default directory structure is changed. Do not run this file. • **xslt.sh** - editable Unix/Linux shell file that calls the appropriate java file to create XSLT. Users are allowed to modify this file to define the relative paths if the default directory structure is changed. Do not run this file. **Other files not listed above** – do not touch them
xsd/genericode/customised	This directory contains all customised genericode files (for demonstration purposes to show how default genericode files supplied in this package can be customised) along with files for preparing customised genericodes and test files. Users can modify the files under this structure to apply constraints on default genericode files in order to meet their specific requirements. (xsd/genericode/customised)
xsd/genericode/customised/gc-files	This sub-directory contains all the customised genericode files (for demonstration purposes) from default genericode files
xsd/genericode/customised/gc-files/common	Provides the directory to store genericode code list file that is customised for use by CommonTypes schema (CommonTypes.xsd). This directory is currently empty.
xsd/genericode/customised/gc-files/xAL	Provides the directory to store genericode code list files that is customised for use by Address XML schema (xAL.xsd). This directory has some sample test genericode files to demonstrate customisation. Users can modify the sample genericode files or add more genericode files

Directory Name	Contents
xsd/genericode/customised/gc-files/xNL	Provides the directory to store genericode code list files that is customised for use by Name XML schema (xNL.xsd). This directory has some sample test genericode files to demonstrate customisation. Users can modify the sample genericode files or add more genericode files
xsd/genericode/customised/gc-files/xPIL	Provides the directory to store genericode code list files that is customised for use by Party XML schema (xPIL.xsd). This directory has some sample test genericode files to demonstrate customisation. Users can modify the sample genericode files or add more genericode files
xsd/genericode/customised/prepare	This directory provides all files required to prepare the customised genericode files. • **customised-cl-business-rules.sch** – Defines the business rules to constraint the use of code lists using schematron language and is modifiable by user. A sample business rules for demonstration purpose is currently defined and requires modification to this file by the user to meet their specific requirements. Users should define appropriate relative paths in this file if they change the default directory structures • **customised-cl-constraints.cva** - list all of the genericode expressions of agreed-upon ciustomised value list value enumerations, and lists all of the customised contexts in which the value enumerations are used. This file is modifiable by user. Sample constraints for demonstration purposes are currently defined and requires modifications to this file by the user to meet their specific requirements. Users should define appropriate relative paths in this file if they change the default directory structures • **CustomisedCodeList.sch** – Defines the customised code list namespace constraints. This file is modifiable by user as they define constraints on default code lists. Users should define appropriate relative paths in this file if they change the default directory structures • **prepare-customised-cl.bat** – editable windows batch file to prepare the customised genericode files to constrain the default genericode code lists for two pass validation. This is the “main” program that executes other batch programs. This file requires modification by the user as the contents in this file are for

Directory Name	Contents
	demonstration purposes only. Users should define appropriate relative paths in this file if they change the default directory structures • **prepare-default-cl.sh** - editable unix/linux shell file to prepare the customised genericode files to constrain the default genericode code lists for two pass validation. This is the "main" program that executes other batch programs. This file requires modification by the user as the contents in this file are for demonstration purposes only. Users should define appropriate relative paths in this file if they change the default directory structures **Other files not listed above**– All the other files in this directory are automatically generated when prepare-ciq.bat/sh file is executed and are to be NOT modified by user and so , do not touch them
xsd/genericode/customised/test	This directory provides files to test the customised genericode lists (for demonstration purposes) by performing two pass validations. Users can modify the .xml files to test different cases. Sample test files have been provided for users to test them. • **test-customised-xnl.bat** – editable windows batch file to test xNL sample document instance (using customised genericode based code lists for demonstration purposes) using two pass validation. This file is modifiable by user. Users should define appropriate relative paths in this file if they change the default directory structures • **test-customised-xnl.sh** – editable Unix/Linux shell file to test xNL sample document instance (using default genericode based code lists for demonstration purposes) using two pass validation. This file is modifiable by user. Users should define appropriate relative paths in this file if they change the default directory structures • **test-customised-xal.bat** – editable windows batch file to test xAL sample document instance (using default genericode based code lists for demonstration purposes) using two pass validation. This file is modifiable by user. Users should define appropriate relative paths in this file if they change the default directory structures

Directory Name	Contents
	• **test-customised-xal.sh** – editable Unix/Linux shell file to test xAL sample document instance (using default genericode based code lists for demonstration purposes) using two pass validation. This file is modifiable by user. Users should define appropriate relative paths in this file if they change the default directory structures • **test-customised-xpil.bat** – editable windows batch file to test xPIL sample document instance (using default genericode based code lists) using two pass validation. This file is modifiable by user. Users should define appropriate relative paths in this file if they change the default directory structures • **test-customised-xpil.sh** – editable Unix/Linux shell file to test xPIL sample document instance (using default genericode based code lists for demonstration purposes) using two pass validation. This file is modifiable by user. • **xAL-customised.xml** – User modifiable sample test file for xAL customized (for demonstration purposes) genericode based code lists • **xNL-customised.xml** – User modifiable sample test file for xNL customized (for demonstration purposes) genericode based code lists • **xPIL-customised.xml** – User modifiable sample test file for xPIL customized (for demonstration purposes) genericode based code lists

2 Customizing your Code Lists / Enumerations

In this section, w explain how to customize and execute the customized code lists using the two Options for code lists provided, to meet your specific requirements.

2.1 Using Option 1 for Code Lists (Default)

Modify enumeration lists in xNL-types.xsd, xAL-types.xsd, xPIL-types.xsd, CommonTypes.xsd as required to add/delete code list values. This is a pretty straight forward approach that requires no further work.

2.2 Using Option 2 for Code Lists (Genericode approach)

This approach requires quite a bit of effort to set it up.

2.2.1 Pre-requisites

Following skills are required to use this approach:

- Good knowledge and understanding of the OASIS Code List Representation scheme
- Good knowledge and understanding of the OASIS UBL Code List Value Validation Methodology
- Experience in creating/updating windows batch files/Unix/Linux shell files
- Knowledge of writing schematron patterns
- The default XML parsers used in this package are Java parsers and hence, the user environment should have Java runtime environment to run the programs.

2.2.2 XML Parsers

The XML and XSLT parser provided with this package under the “utility” directory (clvv/utility) are only sample parsers. Users can use any parsers (not necessarily written in Java) of their choice that can do this job. There is no restriction.

2.2.3 Known Bug

There is a known bug in the xjparse java program (in “utility” directory). For this program to run correctly, the “prepare-gc.bat/prepare-gc.sh” under “utility” directory needs to provide the absolute path of where this java program is located for the program to run. The “prepare-gc.bat” file lists the code below.

2.2.4 Steps to “Prepare and Test” the two pass validation when modifying the supplied default package to customise default genericode

Following are the steps to prepare and test the files for validation using the code list value validation methodology if changes are done to the supplied default package:

1. Create the .gc files to restrict or add to the code lists in the default .gc files
2. Update the “prepare-customised-cl.bat/prepare-customised-cl.sh” file to include the customized .gc files for validation
3. Update the “customised-cl-constraints.cva” file to define appropriate constraint rules reflecting step 1

4. Update the “customises-cl-business-rules.sh”file to define any specific business rules (using Schematron language) to constrain the use of code lists
5. If the default directory structure provided by the CIQ Specification package is changed, ensure that the relative paths in the following files are updated accordingly:
   - .gc files (in genericode/default/gc-files and genericode/customised/gc-files directories)
   - default-cl-constraints.cva (in genericode/default/prepare directory)
   - customised-cl-constraints.cva (in genericode/customised/prepare directory)
   - DefaultCodeList.sh (in genericode/default/prepare directory)
   - CustomisedCodeList.sh (in genericode/customised/prepare directory)
   - customised-cl-business-rules.sh (in genericode/customised/prepare directory)
   - prepare-gc.bat/prepare-gc.sh (in genericode/utility directory)
   - prepare-cva.bat/prepare-cva.sh (in genericode/utility directory)
   - xslt.bat/xslt.sh (in genericode/utility directory)
   - prepare-sh.bat/prepare-sh.sh (in genericode/utility directory)
   - twopass.bat/twopas.sh (in genericode/utility directory)
6. Change the absolute path coded in “prepare-gc.bat/prepare-gc.sh” in genericode/utility directory to the correct absolute path where this package is installed
7. Run “prepare-default-cl.bat/prepare-default-cl.sh”. The output should produce no errors.
8. Run “prepare-customised-cl.bat/prepare-customised-cl.sh” if the code lists are customized. The output should produce no errors.
9. Now test two pass validations by using the “test-default-xal.bat/test-default-xal.sh”, “test-default-xnl.bat/test-default-xnl.sh”, and “test-default-xpil.bat/test-default-xpil.sh”.
10. Play with the sample xml files used in the default testing to check whether two pass validation produces no errors.
11. To test the customised file, run “test-customised-xnl.bat/test-customised-xnl.sh”, “test-customised-xal.bat/test-customised-xal.sh”, and “test-customised-xpil.bat/test-customised-xpil.sh” files.
12. Use the sample xml files or create sample xml files to test the validation of values

2.2.4.1 Steps to test two pass validation when the default package is not modified

Following are the steps to test the supplied default files for validation using the code list value validation methodology if no changes are done to the supplied default package:

1. Change the absolute path coded in “prepare-gc.bat/prepare-gc.sh” to the correct absolute path where this package is installed
2. Now test two pass validations by using the “test-default-xal.bat/test-default-xal.sh”, “test-default-xnl.bat/test-default-xnl.sh”, and “test-default-xpil.bat/test-default-xpil.sh”.
3. Play with the sample xml files used in the default testing to check whether two pass validation produces no errors
4. To test the customised file (if default code lists were customized), first execute “prepare-customised-cl.bat/prepare-customised-cl.sh”. Then, run test-customised-xnl.bat/test-customised-xnl.sh, test-customised-xal.bat/test-customised-xal.sh, and test-customised-xpil.bat/test-customised-xpil.sh files.
5. Use the sample xml files or create sample xml files to test the validation of values

A. Acknowledgements

The following individuals have participated in the creation of this specification and are gratefully acknowledged:

Participants:

John Glaubitz	Vertex, Inc	Member, CIQ TC
Max Voskob	Individual	Former Member, CIQ TC
Hidajet Hasimbegovic	Individual	Member, CIQ TC
Robert James	Individual	Member, CIQ TC
Joe Lubenow	Individual	Member, CIQ TC
Mark Meadows	Microsoft Corporation	Former Member, CIQ TC
John Putman	Individual	Former Member, CIQ TC
Michael Roytman	Vertex, Inc	Member, CIQ TC
Colin Wallis	New Zealand Government	Member, CIQ TC
David Webber	Individual	Member, CIQ TC
Graham Lobsey	Individual	Member, CIQ TC
George Farkas	XBI Software, Inc	Member, CIQ TC

OASIS CIQ Technical Committee (TC) also wishes to acknowledge contributions from former members of the TC since its inception in 2000. Also, the TC would like to express its sincere thanks to the public in general (this includes other standard groups, organizations and end users) for their feedback and comments that helped the TC to improve the CIQ specifications.

Special thanks to Mr.Hugh Wallis, Director of Standards Development of extensible Business Reporting Language (xBRL) International Standards Group (http://www.xbrl.org) for working closely with the CIQ TC in jointly implementing W3C xLink specification that is now used by both xBRL and CIQ Specifications to enable interoperability between the two specifications.

Special thanks to Mr.Carl Reed, Chief Technology Officer of Open Geospatial Consortium (OGC – http://www.opengeospatial.org) for his guidance and assistance to the TC in referencing the work of OGC on GeoRSS and Geo-Coordinates for addresses/locations as part of CIQ Address Specifications.

Special thanks to Mr.Ken Holman, Chair of OASIS Code List TC (http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=codelist) for his assistance to the TC in releasing the OASIS Code List version of CIQ V3.0 XML Schemas.

Last but not least, the TC thanks all users of the CIQ TC specifications in real world and for their continuous feedback and support.

B. Intellectual Property Rights, Patents, Licenses and Royalties

CIQ TC Specifications (includes documents, schemas and examples[1 and 2]) are free of any Intellectual Property Rights, Patents, Licenses or Royalties. Public is free to download and implement the specifications free of charge.

[1]xAL-Australia.XML

Address examples come from AS/NZ 4819:2003 standard of Standards Australia and are subject to copyright

[2]xAL-International.xml

Address examples come from a variety of sources including Universal Postal Union (UPU) website and the UPU address examples are subject to copyright.

xLink-2003-12-31.xsd

This schema was provided by the xBRL group in December 2006.

C. Revision History

Revision	Date	Editor	Changes Made
V3.0 PRD 01	13 April 2006	Ram Kumar and Max Voskob	Prepared 60 days public review draft from Committee Draft 01
V3.0 PRD 02	15 June 2007	Ram Kumar	Prepared second round of 60 days public review draft from Committee Draft 02 by including all public review comments from PRD 01. Also included is implementation of OASIS Code list specification