Queensland University of Technology

Project Documentation

BPMVE XPDL Class Library

Prepared by: Renfred Law (N5637473)
Supervised by: Dr. Ross Brown

25 May 2010
Version: 0.7

Table of Contents

Introduction

Objective

Eset eiusmod tempor incidunt et labore et dolore magna aliquam. Ut enim ad minim veniam, quis nostrud exerc. Irure dolor in reprehend incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse molestaie cillum. Tia non ob ea soluad incommod quae egen ium improb fugiend.

Goals

Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat. Et harumd dereud facilis est er expedit distinct. Nam liber te conscient to factor tum poen legum odioque civiuda et tam.

Solution

Neque pecun modut est neque nonor et imper ned libidig met, consectetur adipiscing elit, sed ut labore et dolore magna aliquam is nostrud exercitation ullam mmodo consequet.

Aenean iaculis laoreet arcu

Eset eiusmod tempor incidunt et labore et dolore magna aliquam. Ut enim ad minim veniam, quis nostrud exerc. Irure dolor in reprehend incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Curabitur vulputate viverra pede

Eset eiusmod tempor incidunt et labore et dolore magna aliquam. Ut enim ad minim veniam, quis nostrud exerc. Irure dolor in reprehend incididunt ut labore et magna aliqua.

Class Diagrams

Design Specification

Technical Specification

Unit Testing

This section describes the approaches I took to write test cases, aspects of the class library that have been tested, and the results of the testing.

Test cases were written using NUnit 2.4.8 Framework. The framework must be installed on the computer in order to run the test cases. Please refer to User Documentation section for more information.

Full Data Structure Testing

Test cases that have "`Full`" post-fixed to the method name are part of full data structure testing. For example:

```
[Test()]
public void ActivityFull() { }
```

This test method tests the input and output equality of the class library. It generates a file from pre-written static properties and creates an object by deserializing the file, after that the newly created object will be serialized to a file and deserialized to another object, and then the test method will compare the equality of the two objects.

Design of this test method is to ensure every classes can be serialized and deserialized to file correctly and maintain all the information as an object.

This test method assumes the classes do not check for XPDL specification conformance, because every properties of every classes need to be fully tested for serialization and deserialization, thus the files generated by this test method will fail when `Validate()` is called.

Test Results

Name of Test Case	Result	Name of Test Case	Result
Activity	Pass	ExternalPackage	Pass
BlockActivity	Pass	LayoutInfo	Pass
Join	Pass	Package	Pass
Loop	Pass	PackageHeader	Pass
LoopMulti	Pass	VendorExtension	Pass
LoopStandard	Pass	ActivitySet	Pass
Route	Pass	Application	Pass

Name of Test Case	Result	Name of Test Case	Result
Split	Pass	ProcessHeader	Pass
TransitionRestriction	Pass	WorkflowProcess	Pass
ApplicationType_BusinessRule	Pass	ArtifactInput	Pass
ApplicationType_EJB	Pass	Assignment	Pass
ApplicationType_Form	Pass	Author	Pass
ApplicationType_POJO	Pass	Category	Pass
ApplicationType_Script	Pass	Codepage	Pass
ApplicationType_WebService	Pass	Color	Pass
ApplicationType_XSLT	Pass	Condition	Pass
ApplicationType	Pass	Coordinates	Pass
Artifact	Pass	Cost	Pass
DataObject	Pass	CountryKey	Pass
Group	Pass	DataField	Pass
Association	Pass	DataMapping	Pass
MessageFlow	Pass	Date	Pass
Transition	Pass	Description	Pass
EndEvent	Pass	Documentation	Pass
Event	Pass	Duration	Pass
IntermediateEvent	Pass	ExprType	Pass
ResultError	Pass	ExtAttr	Pass
ResultMultiple	Pass	ExtRef	Pass
StartEvent	Pass	FormalParam	Pass
TriggerConditional	Pass	Input	Pass
TriggerIntMultiple	Pass	InputSet	Pass
TriggerMultiple	Pass	Limit	Pass
TriggerResultCancel	Pass	MessageType	Pass
TriggerResultCompensation	Pass	Output	Pass
TriggerResultLink	Pass	OutputSet	Pass

Name of Test Case	Result	Name of Test Case	Result
TriggerResultMessage	Pass	Page	Pass
TriggerResultSignal	Pass	Participant	Pass
TriggerRule	Pass	ParticipantType	Pass
TriggerTimer	Pass	Partner	Pass
CostStructure	Pass	PartnerLink	Pass
Deadline	Pass	PartnerLinkType	Pass
Icon	Pass	PartnerRole	Pass
IORules	Pass	Performer	Pass
ResourceCost	Pass	Priority	Pass
SimulationInformation	Pass	PropertyInput	Pass
Transaction	Pass	RedefinableHeader	Pass
ConnectorGraphicsInfo	Pass	Responsible	Pass
NodeGraphicsInfo	Pass	Role	Pass
Implementation	Pass	Script	Pass
NoImplementation	Pass	Service	Pass
Reference	Pass	TimeEstimation	Pass
SubFlow	Pass	Time	Pass
TaskApplication	Pass	TransitionRef	Pass
Task	Pass	Unit	Pass
TaskManual	Pass	ValidFrom	Pass
TaskReceive	Pass	ValidTo	Pass
TaskReference	Pass	Vendor	Pass
TaskScript	Pass	Version	Pass
TaskSend	Pass	WebServiceFaultCatch	Pass
TaskService	Pass	WebServiceOperation	Pass
TaskUser	Pass	Lane	Pass
ConformanceClass	Pass	Pool	Pass

Visual Studio Generated XML Testing

Test cases that have “**VSXML**” post-fixed to the method name are part of Visual Studio generated XML testing. For example:

```
[Test()]
public void ActivityVSXML() { }
```

This test method tests the validity of the class library for serializing and deserializing Visual Studio generated XML files. Test files (in XML format) were pre-generated and put into the test directory (under TestArtifacts/). This type of test cases reads in the file and create an object corresponding to the file, after that the test case will generate another file using the newly created object and create another object using the output file. If this process can be run smoothly then the test case will compare the equality of both objects.

Design of this test method is to ensure XPDL (or compatible XML) files can be serialized and deserialized correctly and maintain all the information as an object.

Test files were generated using Visual Studio 2010 Ultimate Edition with the official XPDL 2.1 Schema file (bpmnxpdl_31.xsd). At the stage it is not known if the file conforms to XPDL 2.1 Specification but I assumed that it is, thus the test files can also be used to test XPDL 2.1 specification conformance at the later stage when all **Validate()** methods are fully implemented.

Test Results

Name of Test Case	Result	Name of Test Case	Result
Activity	Pass	LayoutInfo	Pass
BlockActivity	Pass	Package	Pass
Join	Pass	PackageHeader	Pass
Loop	Pass	VendorExtension	Pass
LoopMulti	Pass	ActivitySet	Pass
LoopStandard	Pass	Application	Pass
Route	Pass	ProcessHeader	Pass
Split	Pass	WorkflowProcess	Pass
TransitionRestriction	Pass	ArtifactInput	Pass
Artifact	Pass	AssignmentAuthor	Pass
DataObject	Pass	Category	Pass
Group	Pass	Codepage	Pass

Name of Test Case	Result	Name of Test Case	Result
Association	Pass	Condition	Pass
MessageFlow	Pass	Coordinates	Pass
Transition	Pass	Cost	Pass
EndEvent	Pass	CountryKey	Pass
Event	Pass	DataField	Pass
IntermediateEvent	Pass	DataMapping	Pass
ResultError	Pass	Date	Pass
ResultMultiple	Pass	Description	Pass
StartEvent	Pass	Documentation	Pass
TriggerConditional	Pass	Duration	Pass
TriggerIntermediateMultiple	Pass	ExprType	Pass
TriggerMultiple	Pass	ExtAttr	Pass
TriggerResultCancel	Pass	ExtRef	Pass
TriggerResultCompensation	Pass	FormalParam	Pass
TriggerResultLink	Pass	Input	Pass
TriggerResultMessage	Pass	InputSet	Pass
TriggerResultSignal	Pass	Limit	Pass
TriggerRule	Pass	MessageType	Pass
TriggerTimer	Pass	Output	Pass
CostStructure	Pass	OutputSet	Pass
Deadline	Pass	Page	Pass
Icon	Pass	Participant	Pass
IORules	Pass	ParticipantType	Pass
ResourceCost	Pass	PartnerLink	Pass
SimulationInformation	Pass	PartnerLinkType	Pass
Transaction	Pass	Performer	Pass
ConnectorGraphicsInfo	Pass	Priority	Pass
NodeGraphicsInfo	Pass	PropertyInput	Pass

Name of Test Case	Result	Name of Test Case	Result
Implementation	Pass	RedefinableHeader	Pass
NoImplementation	Pass	Responsible	Pass
Reference	Pass	Script	Pass
SubFlow	Pass	TimeEstimation	Pass
TaskApplication	Pass	TransitionRef	Pass
Task	Pass	Unit	Pass
TaskManual	Pass	ValidFrom	Pass
TaskReceive	Pass	ValidTo	Pass
TaskReference	Pass	Vendor	Pass
TaskScript	Pass	Version	Pass
TaskSend	Pass	WebServiceFaultCatch	Pass
TaskService	Pass	WebServiceOperation	Pass
TaskUser	Pass	Lane	Pass
ConformanceClass	Pass	Pool	Pass
ExternalPackage	Pass		

TIBCO Files Testing

Test cases that have “`TIBCO`” post-fixed to the method name are part of TIBCO files testing. For example:

```
[Test()]
public void SimpleTIBCO() { }
```

This test method tests the validity of the class library for serializing and deserializing TIBCO generated files. Test files (in XPDL format) were pre-generated and put into the test directory (under TestArtifacts/TIBCO). This type of test cases reads in TIBCO generated XPDL file and create an object that holds the information. The newly created object will be serialized to an output file and another object will be created using the output file. The method will then test the equality of both objects.

Design of this test method is to ensure the class library is able to serialize and deserialize TIBCO generated XPDL files correctly and maintain all the information of the file as an object.

Test files were generated using TIBCO Business Studio 3.2 for Windows. Please note that the output files generated by the class library do not, and will not, include default value for attributes and elements. For example:

```
<xpdl2:NodeGraphicsInfo IsVisible="true"/>
```

and

```
<xpdl2:NodeGraphicsInfo />
```

These two elements will be treated as exactly the same by the class library because `IsVisible` is defaulted to be `true` as specified in XPDL 2.1 Specification.

Mono for Mac/Linux developers be careful, the class library will sometimes generate empty elements in the output file, but rest assured that those elements will not have any negative impact of importing to TIBCO. For more information please look at *Platform Variations* section.

Test Results

Name of Test Case	Pass	Fail

User Documentation

This section provides technical information for future developers, as well as user guide for users who wish to use the library.

System Requirements

Windows Platform

- Microsoft Visual Studio 2008 or 2010 IDE (Professional or Express edition), or
  http://www.microsoft.com/express/
  http://www.microsoft.com/visualstudio/en-us
- MonoDevelop 2.2.2 IDE (with GTK#)
  http://monodevelop.com/Download
- Microsoft .NET Framework 3.5
  http://www.microsoft.com/net/
- NUnit 2.4.8 Unit Testing Framework
  http://www.nunit.org/?p=download
- TIBCO Business Studio 3.2 for Windows (Optional)
  http://developer.tibco.com/business_studio/default.jsp
- Doxygen Documentation Generator (Optional)
  http://www.doxygen.org/
- GraphViz 2.26.3 Documentation Graph Generator (Optional)
  http://www.graphviz.org/Download_windows.php

Mac/Linux Platform

- MonoDevelop 2.2.2 IDE
  http://monodevelop.com/Download
- Mono Framework 2.4.3
  http://www.go-mono.com/mono-downloads/download.html
- NUnit 2.4.8 Unit Testing Framework
  http://www.nunit.org/?p=download
- TIBCO Business Studio 3.2 for Linux (Optional, no Mac support)
  http://developer.tibco.com/business_studio/default.jsp
- Doxygen Documentation Generator (Optional)
  http://www.doxygen.org/
- GraphViz 2.26.3 Documentation Graph Generator (Optional)
  Mac: http://www.graphviz.org/Download_macos.php
  Ubuntu: http://www.graphviz.org/Download_linux_ubuntu.php

Limitations

XPDL 2.1 Data Types Support

Future Directions

BPMN 1.2 and 2.0 Support

XPDL 2.1 Specification Conformance

MySQL Persistent Database Support

Platform Variations

Generations of Empty XML Elements

Mono developers please note that the class library will sometimes generate empty elements to the output file, but rest assured that those elements will not have any negative impact of importing to TIBCO. For example the following two elements are exactly the same with or without the empty `NestedLanes` because it is empty and also an optional element according to the specification.

Mono Framework	.NET Framework

`<xpdl2:Lane ID=””>`
`    <xpdl2:NestedLanes />`
`</xpdl2:Lane>`
	`<xpdl2:Lane ID=””>`
	`</xpdl2:Lane>`

Example : Frameworks Variation

This is a design and implementation flaw of Mono framework, it works correctly on Windows with .NET framework 3.5. All class properties that implement the {name}Specified pattern will be serialized to file without empty elements. For more information about {name}Specified design pattern please refer to *Design Specification* section.

Licensing

This project is hosted on Google Code under Lesser General Public License (LPGL). Please refer to http://www.gnu.org/licenses/lgpl.html for additional information. Any developer who wishes to use and/or distribute this project under any circumstances must agree to the license.

This license is a set of additional permissions added to version 3 of the GNU General Public License. For more information about how to release your own software under this license, please see our page of instructions.

GNU LESSER GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.

0. Additional Definitions.

As used herein, “this License” refers to version 3 of the GNU Lesser General Public License, and the “GNU GPL” refers to version 3 of the GNU General Public License.

“The Library” refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An “Application” is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.

A “Combined Work” is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the “Linked Version”.

The “Minimal Corresponding Source” for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or

b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the Combined Work with a copy of the GNU GPL and this license document.

c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.

d) Do one of the following:

0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.

1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.

e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.

b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

Appendix

Source Code Repository

Google Code Repository

http://code.google.com/p/bpmve-xpdl-library/

Application Programming Interface (API) Repository

000webhost Free Web Hosting

http://bpmve.comxa.com/

The repository is hosted at a free server, thus it would not be made available permanently. Please follow the instruction at User Documentation to generate the API using Doxygen.

Project Documentation Repository

Apple iWork.com (Online Viewer)

http://public.iwork.com/document/?a=p289467908&d=BPMVE_XPDL_Document.pages

Online version of this document can be found at the above link. You can choose to download in the format of PDF, Microsoft Word 2007, or iWork Pages ’09.

Contact Information

I can be contacted by the following email addresses:

Permanent: renfred@me.com

University: chin.law@connect.qut.edu.au

If you wish to join the project at Google Code, please send me an email.