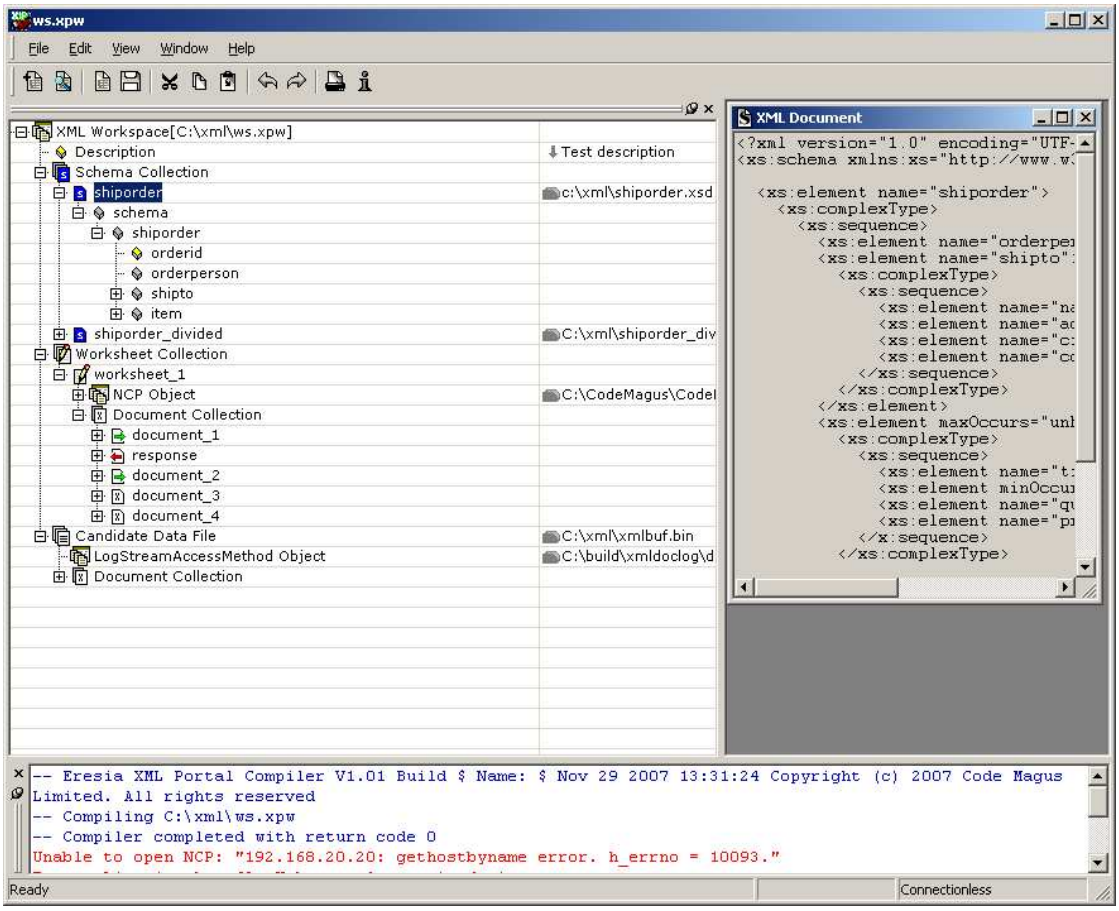
Eresia XML Interface Portal Version 1.1

CML00034-11

Code Magus Limited (England reg. no. 4024745)
Number 6, 69 Woodstock Road
Oxford, OX2 6EY, United Kingdom
www.codemagus.com

February 1, 2010



Contents

1 Introduction **2**
- 1.1 Overview . . . 2
- 1.2 XML Documents and the XIP . . . 2
- 1.3 summary . . . 3

2 Components of the XIP **3**
- 2.1 Overview . . . 3
- 2.2 The Workspace . . . 3
- 2.3 The Environment Variables collection . . . 4
- 2.4 Schema . . . 5
  - 2.4.1 Viewing Schema Definitions . . . 5
  - 2.4.2 Using Schemas to create new XML documents . . . 5
- 2.5 Worksheets . . . 5
  - 2.5.1 Network Control Program . . . 6
  - 2.5.2 Sending XML over a network . . . 6
- 2.6 XML Documents . . . 6
  - 2.6.1 Viewing XML documents . . . 6
  - 2.6.2 Editing and changing XML documents . . . 6
  - 2.6.3 Sending XML documents over a network . . . 7
- 2.7 Candidate Data Files . . . 7

2.7.1 Data file 8
2.7.2 LogStreamAccessMethod DLL 8

3 The Graphical User Interface 8
3.1 The Menu Bar 9
3.1.1 File 9
3.1.2 View 9
3.1.3 Help 10

4 GUI Components 10
4.1 XIP Workspace 10
4.2 Description 11
4.3 Environment Variables collection 11
4.4 Environment Variable 12
4.5 Schema Collection 12
4.6 Schema 13
4.7 Worksheet Collection 14
4.8 Worksheet 15
4.9 XML Document 16
4.10 Candidate Data File 17
4.11 LogStreamAccessMethod DLL 18
4.11.1 Tree Node 18

I Appendix 19

A Examples 19
A.1 Viewing a Schema 19
A.2 Using schema to create new XML documents 21
A.3 Creating an XML document from scratch 22
A.4 Candidate Data File usage 24
A.5 Setting an NCP and sending XML 25

1 Introduction

1.1 Overview

The XML Interface Portal (XIP) is a graphical interface which allows the user to perform tasks on XML documents and interact with various other components, such as XML Schemas, Candidate Data Files, Network Control Programs and XML documents.

XML documents can be viewed, edited, created and deleted and sent over a network to a server, and the response is viewed in the same fashion as all of the other XML documents in the Graphical User Interface.

On exit the XIP saves all documents and settings for a workspace in a user named file; this file is then available to be re-opened in order to continue working on these same documents.

The GUI of XIP is divided into two sections. The first section contains a hierarchical view of the components contained in XIP, and the second section is used to display information to the user.

The first section is divided into two columns. The first column contains the name of the component in the hierarchy, and the second column contains the value of the corresponding component in the left column.

Section 3 on page 8 and onwards contains a reference of all of the components of the visual part of the XML portal (the XIP). Information is provided on how all of the individual components of XIP work.

Appendix A on page 19 provides real examples of using XIP.

1.2 XML Documents and the XIP

The entire purpose of the XIP is to manipulate and make use of real XML documents. The XIP simplifies this task by automating the task of converting the hierarchical view (also known as a tree view) of XML documents into real XML documents.

This means that the user simply needs to create the XML document in a tree form, and the XIP will convert the tree into an XML document. The reverse also applies. When an XML document is received from a server, the XIP converts the actual XML document into a tree representation of that document, and this tree can then be viewed and manipulated in the XIP.

All XML documents are validated, and any errors relating to document structure, schema non-conformance and any other errors, are reported to the user. If errors in the XML documents are found, these errors are displayed in XIP.

See section 2.6 on page 6 for further details.

1.3 summary

In summary, XIP can be used to:

- View XML documents in a tree form
- Edit and change the structure and/or content in the document
- Create a new XML document starting from an empty document
- Create an XML document from an existing XML schema definition
- Use an NCP (Network Control Program) to send these documents over a network

2 Components of the XIP

2.1 Overview

The XIP uses a number of different components which the user interacts with. These components all have very specific roles in the GUI. Components are represented in a tree structure, which allows the user to easily determine the relationships of each of these components to each other.

For example, the ”Schema Collection” component, which contains a collection of schema as the name implies, will contain each of its schema documents under the schema collection element, and the ”Worksheet Collection” component will contain a collection of all of the worksheets in the workspace.

This chapter will explain what purpose these components serve, and how these components are related. The technical usage of the components is given in section 3 on page 8.

2.2 The Workspace

The workspace can be described as a set of properties, documents and settings which are grouped together in one project.

Each workspace is described in a single file, and this file has its own grammar to describe the contents of the workspace. All changes, additions etc, are contained in this workspace file.

This allows the user to make changes to their workspace, save it and be able to re-open it at a later stage to continue working with it.

The workspace contains the following components:

- Description.

  The description is simple text written by the user to describe the workspace. This is a note to the owner or users of the workspace, and serves no other purpose.

- Schema collection.

  The schema collection component contains a list of all of the schemas defined by the user. These schemas are displayed in the same manner as normal XML documents. The XIP processes a schema definition, and creates a tree representation of the schema. The nodes in this schema can be used to create new documents in a workspace. See section 2.4 on page 5.

- Worksheet collection.

  The Worksheet collection contains a list of all of the worksheets used in the workspace. Each of these worksheets can have their own NCP defined, and each worksheet contains a list of documents used in that worksheet. See section 2.5 on page 5.

  Each Worksheet contains a document collection which in turn contains a list of documents used in that worksheet. Although documents appear under a Worksheet they are described in their own section; see section 2.6 on page 6.

- Candidate data file(s).

  The Candidate data files are actual files containing documents, and although the documents are XML they may be embedded in a physical data layout that has its own data format.

  Each Candidate data file component also has a LogStreamAccessMethod DLL which is the DLL used to read the data file in the Candidate data file component. See section 2.7 on page 7.

2.3 The Environment Variables collection

The environment variables collection contains a list of environment variables that are set when a workspace is loaded. They are also set when you add a new variable or change the name or the value of an existing variable.

2.4 Schema

A Schema component has a name, which is simply a name the user assigns to a schema. This name does not have any other purpose in the XIP.

The value of this node on the right pane contains the actual file path of the schema definition document.

When a schema is added, or loaded when the workspace is loaded, the schema is parsed, and a tree is built to represent that schema.

In the latter case, they can drag the node they want to copy either to a new document, or to an existing tree node. If the latter case occurs, a copy of the tree at the source of the drag will be added as a sub-tree to the target of the drag. This makes it possible to make many copies of identical tree structures in an XML document.

2.4.1 Viewing Schema Definitions

As mentioned earlier, each Schema is processed by the XIP, and translated into a graphical tree structure view, so the user can view the structure of the schema itself as if it were an XML document. Refer to section A.1 on page 19 for an example of viewing a schema.

2.4.2 Using Schemas to create new XML documents

The schema tree can be used to create a new XML document which conforms to the given schema. This is done by dragging a schema node to a document in a worksheet. The result is that an XML document is created and the tree view of that document can be seen in the Worksheet's document collection.

2.5 Worksheets

The XIP uses a concept of worksheets to allow users to change the content of XML documents. Only when a document is contained in a worksheet can its structure and content be changed.

A worksheet has the following functions:

- Listing, editing and changing the structure of XML documents
- Creating new XML documents
- Creating XML documents from an existing schema definition
- Creating XML documents by copying nodes from other documents

- Sending XML documents over a network using an NCP.

Each workspace has a worksheet collection under which the worksheets are stored.

The user can add a worksheet to the worksheet collection, and when the workspace is saved, the list of documents inside the worksheet, as well as the settings for the optional NCP are also saved to the workspace file.

2.5.1 Network Control Program

A Network Control Program (NCP) refers to a DLL and is used to send and receive data to and from a network destination.

The function of an NCP DLL is to translate an XML document in a worksheet into a buffer and send it to the network address described by the NCP.

2.5.2 Sending XML over a network

As the NCP is defined at the worksheet level, all documents in the worksheet use the same NCP definition. If a response is generated by the server, the NCP will translate it into a new document within the worksheet as a response XML document.

2.6 XML Documents

2.6.1 Viewing XML documents

An XML document inside the XIP is displayed as a tree structure, showing each element inside an XML document. If an element has child elements, there will be a ”+” next to the corresponding component in the XIP, and each of the children will be listed under that component.

XML elements are displayed in the XIP with a grey icon.

The XIP will also show what attributes each element inside the XML document has. These are displayed with a yellow icon to distinguish them from element nodes as discussed above.

2.6.2 Editing and changing XML documents

The XIP allows the user to edit XML documents in terms of the data that is contained in elements within the XML documents. This is done simply by double-clicking the value of an element and entering the new value. The XIP will then output the new value of the element when the XML document is sent over a network, or output to any other

destination. This process of changing the actual XML document is transparent to the user.

The user can not only change the data within the XML document, but also change the actual structure of the XML document. For example, the user can add and remove child elements, add and remove attributes of an element, and can copy entire trees of a document to anywhere within an existing, or empty document by simply dragging these elements, or by using the menus of the GUI to perform these tasks.

Once a worksheet has been created, the user can add documents to the worksheet, each with its own structure and content. This content can be created from an empty document, or built from an existing schema definition.

When a document is created from a schema definition, a document is generated from and conforms to the schema definition. Using an existing schema definition makes it fast and easy to create correctly conforming documents.

When a document is created from scratch, the GUI will generate an XML document from the created tree in the GUI.

Both the editing and changing of documents are explained in detail in section 3 on page 8.

2.6.3 Sending XML documents over a network

The XIP provides the user with a mechanism of sending XML documents over a network (for example, to an XML server). If a response is generated by the destination network server, then this response is processed and displayed by the XIP. The user can then examine this response and analyse the returned data.

The returned document cannot be edited or changed because the response has been sent to XIP from a server, and not generated by the user or by the XIP. This returned document is therefore real data returned by the server, so it does not make any sense to change this data.

The user can, however, use the returned document to create a new document, and in this fashion, can edit and change the returned XML document, which can then be sent to a server if required.

2.7 Candidate Data Files

Candidate Data Files hold XML documents within a physical data structure which is accessed via the LogStreamAccessMethod object.

2.7.1 Data file

As mentioned earlier, the candidate data file itself is a disk file which contains a list or buffer of records. This data file has its own data format, and therefore needs a program to decode this information. This is the role of the Log Stream Access Method DLL (see section 2.7.2 on page 8).

The Candidate Data File value refers to the actual file that holds the physical data.

2.7.2 LogStreamAccessMethod DLL

The Log Stream Access Method Dynamic Link Library (DLL) is a DLL executable which is used to read and interpret a candidate data file and extract the XML documents from the physical structure.

3 The Graphical User Interface

The Graphical User Interface (GUI) is a visual representation of the contents of a workspace. Each of the components that are visible in the GUI will be described in their own section (See section 4 on page 10).

The GUI is split into two sections. These sections are divided by a line down the screen. The section on the left contains the different components available in the project, which are the Description, Schema Collection, Worksheet Collection and Candidate Data Files.

These items are all displayed in a tree structure. In this way it is easy to tell how the components are related to one another.

The section on the right hand side of the line displays the value of the corresponding component on the left. This value can be double-clicked and edited if the corresponding component has a value that can be changed. An example of this is, the ”description” component of a workspace, which contains a simple textual description of the project.

There are actions that can be performed on many of these components. These actions can be performed by right-clicking on the component. A drop-down menu will appear, and will display a list of all of the available actions that can be performed on that particular component.

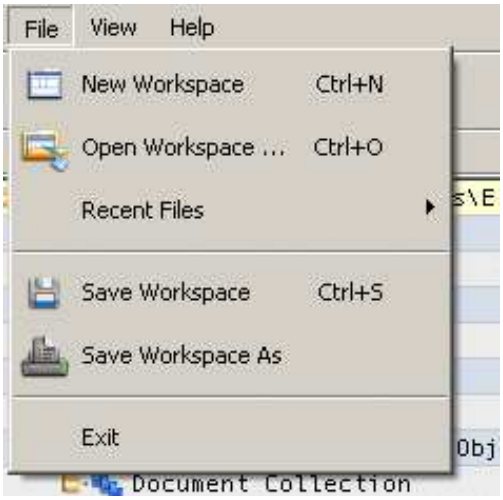
For example, if there is no description defined for the current workspace, the user can add a description to the workspace by right-clicking on the Workspace component and selecting ”Add” and then ”Description”. A new entry will be added under the Workspace component for the description.

These actions will be discussed in detail in their own section in the document.

3.1 The Menu Bar

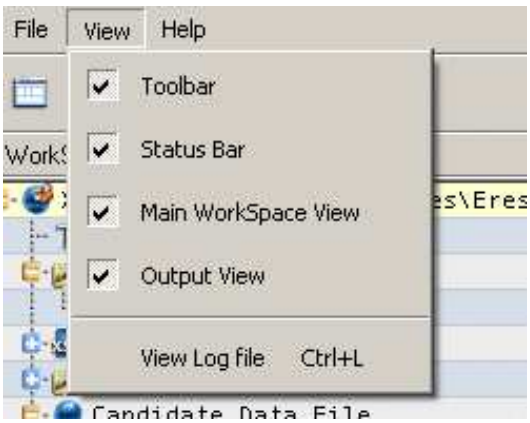
This section explains the layout of the menu bar. Individual menu items are also explained.

3.1.1 File



Actions	
New Workspace	Create a new, empty Workspace
Open Workspace	Open an existing Workspace from file
Recent Files	Display and load a recently opened Workspace
Save Workspace	Save the current Workspace to file
Save Workspace As	Save the current Workspace to a new file
Exit	Close the GUI

3.1.2 View



Actions	
Toolbar	Toggle displaying of the toolbar
Status Bar	Toggle displaying of the status bar
Main Workspace View	Toggle displaying of the Workspace tree
Output View	Toggle displaying of the output window
View Log file	Display the log file of the current session

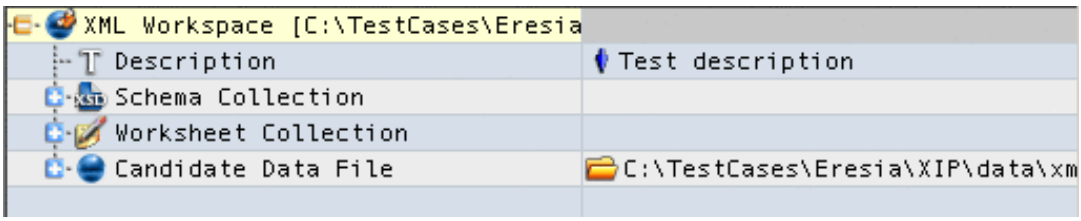
3.1.3 Help

Actions	
Contents and Index	Display the help file for the XIP
About Eresia XIP	Show the Help/About dialog

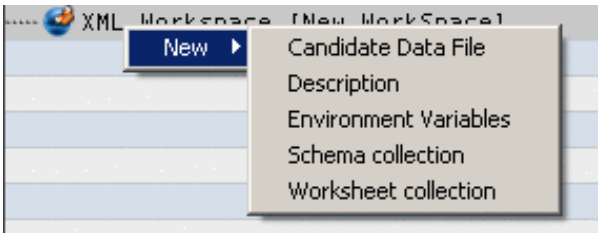
4 GUI Components

Each component of the GUI is described in this section, along with its actions and some examples of usage. The tables listed below contain descriptions on the actions (right-click options) for each component.

4.1 XIP Workspace

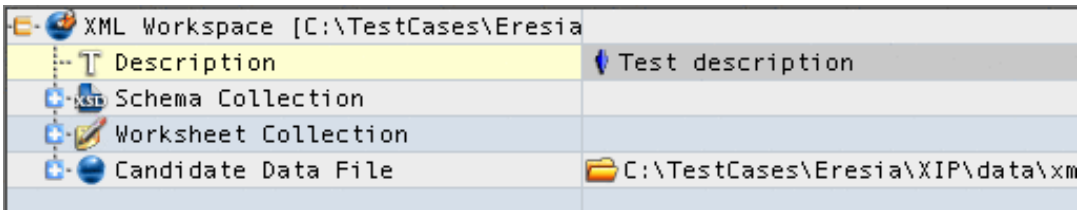


The XML Workspace component contains all of the other components pertaining to a project. It, and all the items below it, can be saved to a file; normally with a suffix of ".xpw".



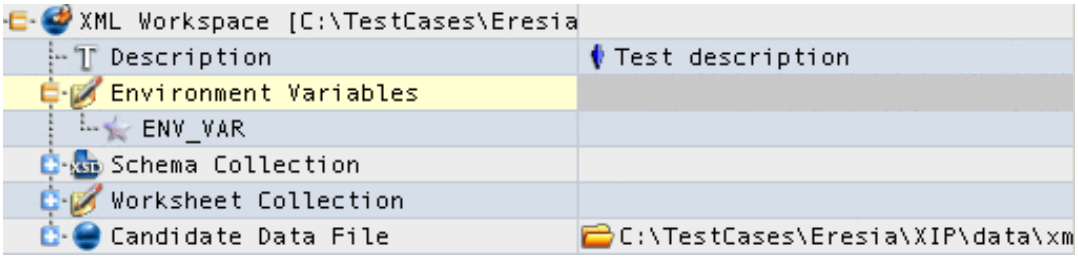
Actions	
New/Description	Add a description of the workspace
New/Schema Collection	Add collection of schema documents to the workspace
New/Worksheet Collection	Add collection of worksheets to the workspace
New/Candidate Data File	Add data file containing XML documents to the workspace

4.2 Description

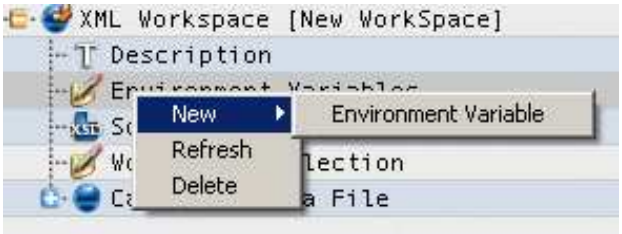


The description simply provides a textual description of the current Workspace.

4.3 Environment Variables collection

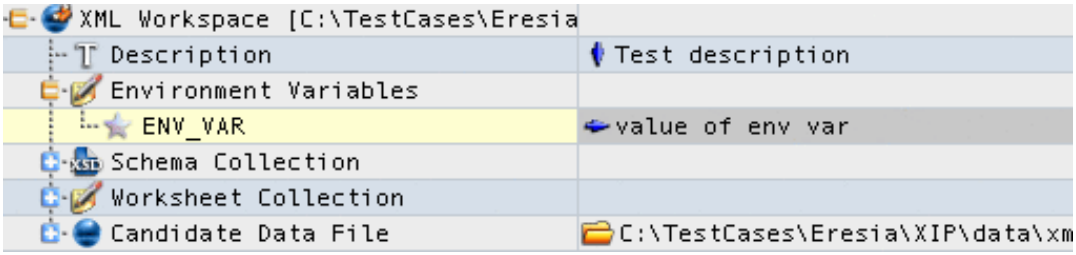


The Environment Variables Collection contains a list of all defined environment variables. Each of these has its own entry. See section 4.4 on page 12.

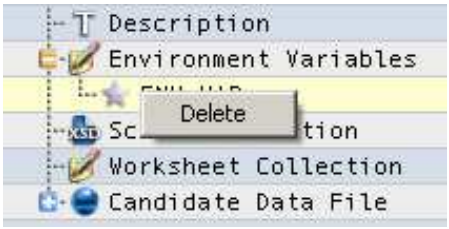


Actions	
New/Environment Variable	Add a new entry
Refresh	Refresh the list of variables
Delete	Remove this element

4.4 Environment Variable

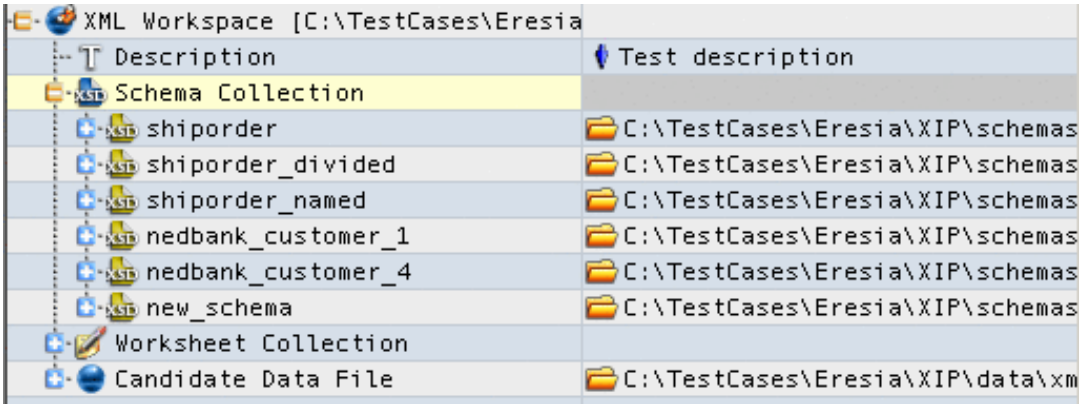


The Environment Variable node has a name and a value. The name is the actual name of the environment variable to set, and the value is the actual value.



Actions	
Delete	Remove this element

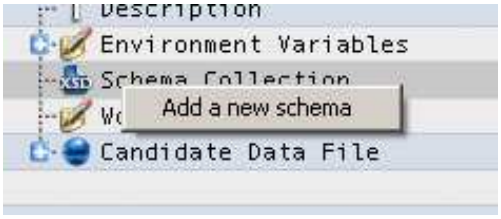
4.5 Schema Collection



A schema Collection contains a list of Schema components. Each of these Schema components has its tree structure which represents the structure of the actual Schema document this component refers to.

A Schema component has a name (in the left pane), which is a name the user can give to each schema. This name does not have any other purpose in the XIP.

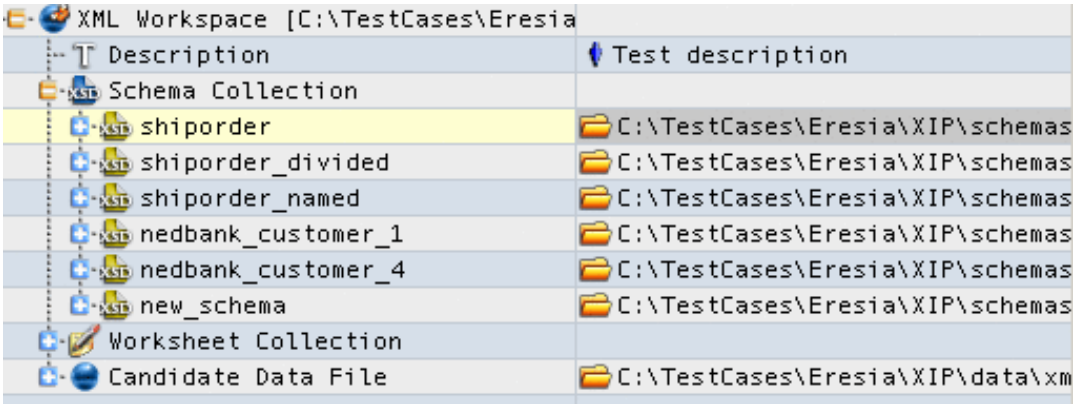
The value of this node contains the actual file path of the schema definition document.



Actions	
Add a new Schema	Add a new Schema document into the schema collection

Once a schema has been added using the above action, it will appear in the Schema collection, and will have its own tree and its own actions (See section 4.6 on page 13).

4.6 Schema

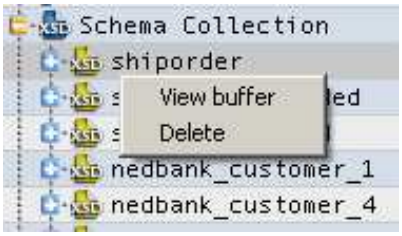


A Schema component has a name (in the left pane), which is simply a name the user can give to each schema. This name does not have any other purpose in the XIP.

The value of this node on the right pane contains the actual file path of the schema definition document.

When a schema is added, or loaded when the workspace is loaded, the schema is parsed, and a tree is built to represent that schema.

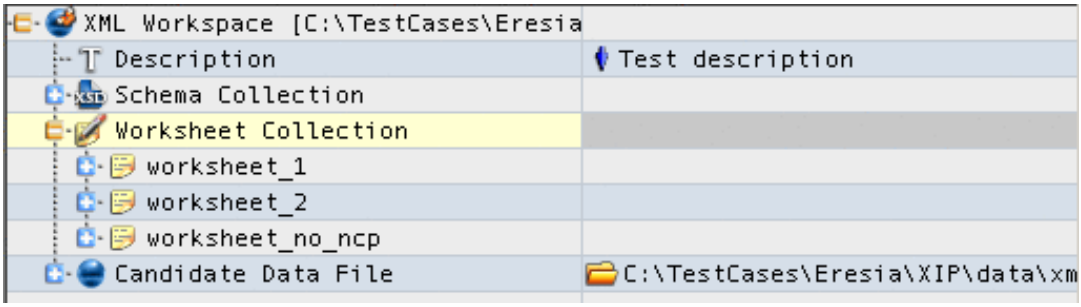
If a user wishes to create a document from this schema definition, the user can drag the node they want to copy either to a new document, or to an existing tree node. If the latter case occurs, a copy of the tree at the source of the drag will be added as a sub-tree to the target of the drag. This makes it possible to make many copies of identical tree structures in an XML document.



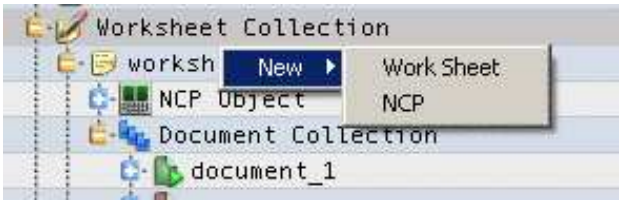
Actions	
View XML buffer	View the actual XML document of this schema

Selecting ”View XML buffer” will display the actual XML schema document.

4.7 Worksheet Collection

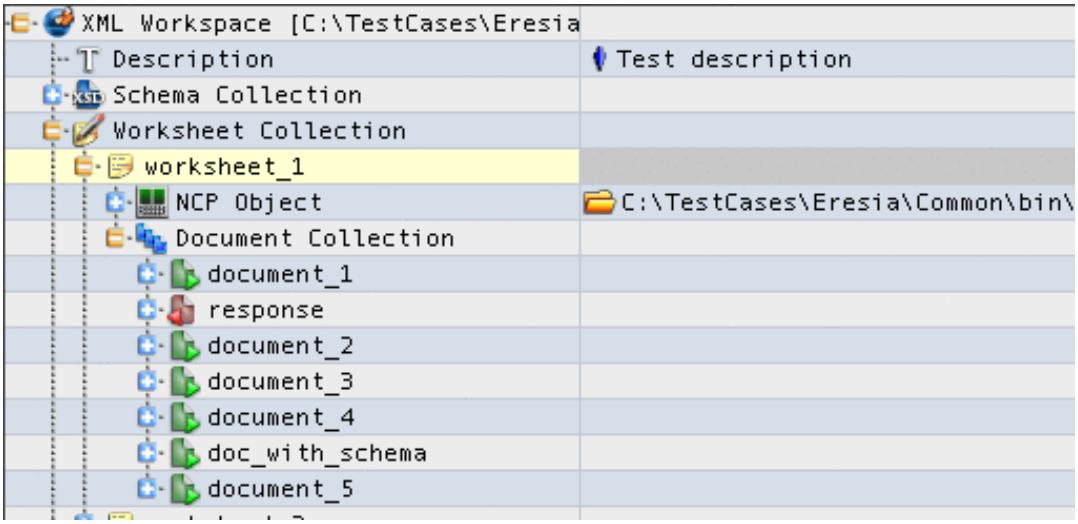


A Worksheet Collection contains a list of Worksheet components (see section 4.8 on page 15).



Actions	
New/Work Sheet	Add a new Worksheet to this worksheet collection

4.8 Worksheet



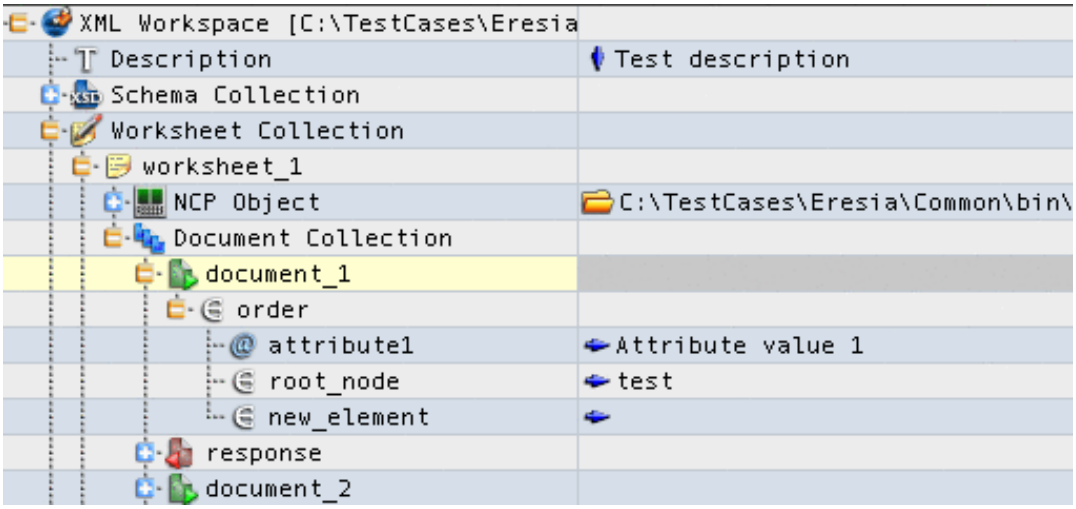
The worksheet contains an NCP and a list of XML documents.

Actions	
New/Document	Add a new XML document to the document collection
New/NCP	Add an NCP component to the Worksheet
New/Notes	Add a text note to the Worksheet Collection
Delete	Remove this worksheet from the Worksheet Collection
Set output file	Specify the name of the file to export documents to
Set input file	Specify the name of the file to import documents from

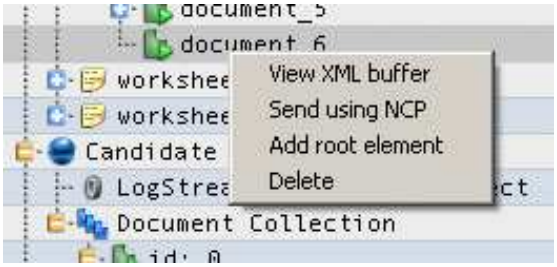
The New/Document option will add a new, empty document is added to the document collection. This document can then be used to build an XML document.

New/NCP adds an NCP component, which allows documents to be sent over a network (See section 4.9 on page 16).

4.9 XML Document



The Document component is a representation of an actual XML document. This component has different actions available depending on its status in the tree.



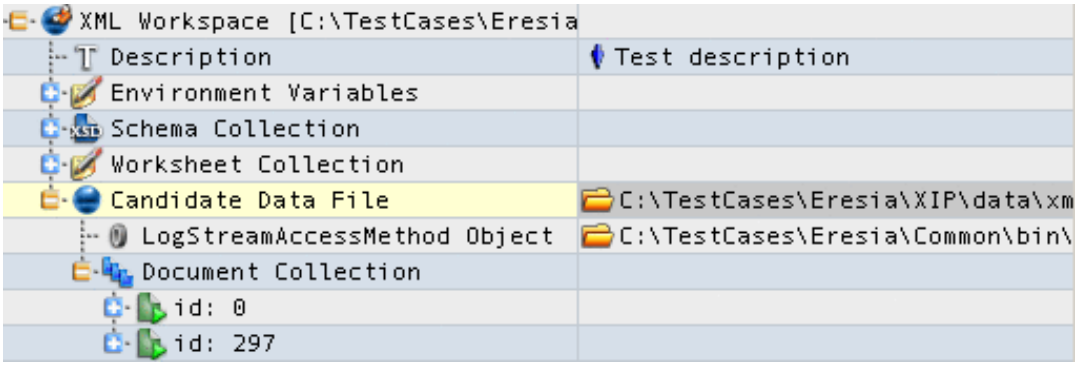
Actions	
Send using NCP	Send this document using the worksheet's NCP
View XML buffer	View the actual XML document for this component
Add a root element	Add a root XML element to this document
Delete	Remove this document from the XIP GUI

Note that only some of these actions will be available to the user depending on where this document is in the tree and on other conditions.

The "Send using NCP" option will only be available if this document is in a worksheet, and if the worksheet has a valid NCP defined. When this option is selected, the XIP will send this document using the NCP, and add the response to the GUI if a document is received.

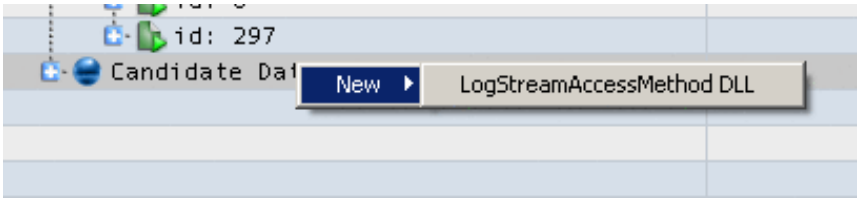
When "Add a root element" is shown, it allows the user to add a child component which will be the name of the root node in the actual XML document.

4.10 Candidate Data File



A candidate data file is a data file containing a number of XML documents, in a specific format.

The value of the component on the right hand side is the full path to the actual XML data file mentioned above.

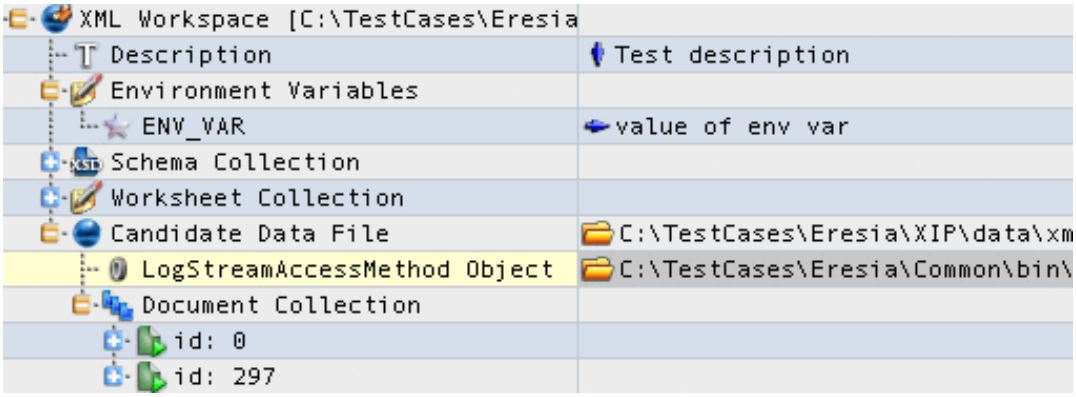


Actions	
New/LogStreamAccessMethod DLL	Add a Log Stream Access Method DLL

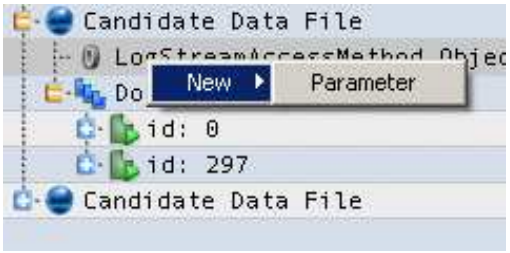
The Log Stream Access Method DLL specifies the path of the DLL used to read the data file. If both the path of the data file, and the DLL are valid, then the documents that are contained inside that data file will be added to the Candidate Data File component under a ”Document Collection” component.

Each of these can be viewed, displayed and used to generate new documents in a worksheet, but can not be edited or changed (See section 4.9 on page 16).

4.11 LogStreamAccessMethod DLL



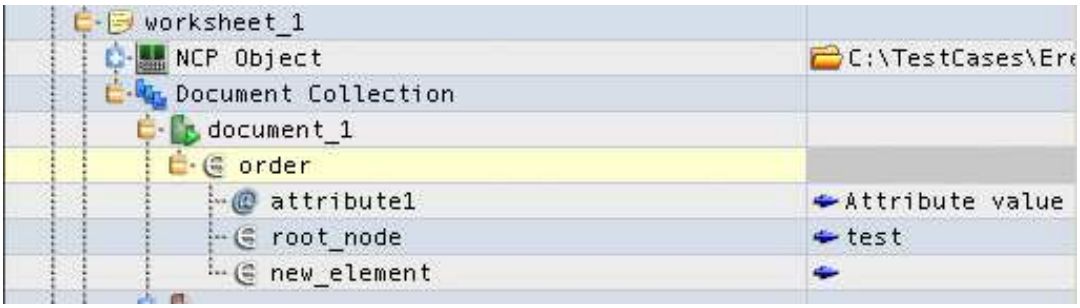
This component simply allows the user to set up the actual DLL used to read the candidate data file (see section 4.10 on page 17).



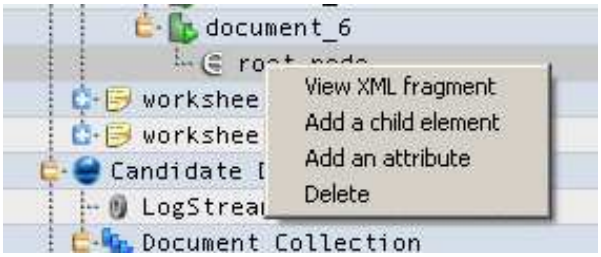
Actions	
New/Parameter	Add a parameter for this LogStreamAccessMethod DLL

Each of the parameters created can then be edited. Each NCP has its own specific parameters.

4.11.1 Tree Node



A tree node is any node that is contained under a document component. Like the document component, the actions that can be performed depend on many factors like whether it is in a worksheet, whether it is in a Candidate Data File, etc.



Actions	
Add a child element	Add a child element (sub element) to this element
Add an attribute	Add an attribute to this element
Delete	Remove this element from its parent element

As discussed in section 2.6.1 on page 6, the action of adding a child element or an attribute will create XML structures in the corresponding document.

All of the other functions of the XML document itself are performed by selecting actions on the Document component.

Part I

Appendix

A Examples

A.1 Viewing a Schema

This example illustrates the usage of the workspace, schema collection and schema components.

Open a text editor and enter this text:

1. Find a Schema definition file (e.g. shiporder.xsd).
2. Open the XIP.
3. Select ”File/New Workspace” from the menu
4. Right-click on XML Workspace
5. Select ”New/Schema Collection”
6. Expand the XML Workspace node
7. Right-click on the Schema Collection node

8. Select ”Add new schema”
9. Expand the Schema Collection node
10. Double-click on new schema and change the name to shiporder
11. Double-click on the right hand pane next to shiporder
12. Browse to the folder where you saved the above file and select it
13. Expand all the nodes
14. Right-click on new schema node
15. Select ”View buffer”

You will notice some of the expanded nodes are grey (these are elements in an XSD document), and some of the nodes are yellow (these are attributes of a particular node in an XSD document).

You can now use this structure to create a new document in the XIP (see section A.2 on page 21 for an example).

After you selected ”View buffer” a window pops up showing the actual schema document.

A.2 Using schema to create new XML documents

This example will show the user how to create a new XML document from an existing Schema definition.

1. Perform all steps in example A.1 on page 19
2. Right-click on XML Workspace
3. Select ”New/Worksheet collection”
4. Right-click on Worksheet collection
5. Select ”New/Worksheet”
6. Expand the Worksheet collection node
7. Right-click on worksheet 1
8. Select ”New/Document”
9. Expand the Document collection node
10. Go to the node in Schema collection called shiporder
11. Go to shiporder’s first child called schema
12. Drag the node under schema called shiporder to the new document
13. There is now a new XML document in the document collection which conforms to the shiporder schema
14. Right-click on the new document
15. Select ”View buffer”

You will now have a new document which conforms to the shiporder schema, and a window containing the actual XML document.

A.3 Creating an XML document from scratch

This example will show the user how to create a new XML document from scratch.

1. Open the XIP.
2. Select ”File/New Workspace” from the menu
3. Right-click on XML Workspace
4. Select ”New/Worksheet Collection”
5. Right-click on Worksheet Collection node
6. Select ”New/Work Sheet”
7. Expand the Worksheet Collection node
8. Right-click on Worksheet 1
9. Select ”New/Document”
10. Expand the Worksheet 1 node
11. Right-click on document 1
12. Select ”Add root node”
13. Expand
14. Right-click on the document node
15. Select ”Add child element”
16. Expand
17. Double-click on new element and type ”email”
18. Right-click on the email node
19. Select ”Add an attribute”
20. Right-click on the email node
21. Select ”Add child element”
22. Repeat the last two steps twice
23. Rename the first child of email to ”emailtype”
24. Double click next to emailtype on the right pane and type ”email message”
25. Rename the second child of email to ”emailFrom”
26. Rename the third child of email to ”emailTo”
27. Rename the fourth child of email to ”emailMessage”

28. Double-click next to ”emailFrom” in the right pane and enter ”somebody”
29. Double-click next to ”emailTo” in the right pane and enter ”anybody”
30. Double-click next to ”emailMessage” in the right pane and enter ”testing the XIP”
31. Right-click on the document 1 node
32. Select ”View XML buffer”

You will see the new, complete XML document in the right hand pane.

You can now change the values of the tree and add and delete nodes and when you select ”View XML buffer”, you will see the updated document.

You can edit the actual XML in the right hand pane. If the changes are made and you select to keep these changes, the tree on the left will be updated to match the document in the document editor.

A.4 Candidate Data File usage

This example shows the user how to load a new Candidate Data File.

1. Perform all steps in example A.3 on page 22
2. Right-click on the XML Workspace
3. Select ”New/Candidate Data File”
4. Right click on the value of CandidateDataFile
5. Select an existing XML data file
6. Right-click on Candidate Data File
7. Select ”New/LogStreamAccessMethod DLL”
8. Right-click on LogStreamAccessMethod Object
9. Select ”Add/Parameter”

At this stage all of the documents inside the specified data file will be read out of the file using the DLL specified by the LogStreamAccessMethod Object, and using the parameters defined by selecting Add/Parameter.

The document collection under the candidate data file will be populated with all of the documents inside the data file, if any exist, and the reading of the data file was successful.

These new documents can be used to create new documents, in exactly the same fashion as a new schema (see example A.2 on page 21).

A.5 Setting an NCP and sending XML

This example shows the user how to use an NCP in a worksheet to send a message to a network server.

1. Perform all steps in example A.3 on page 22
2. Right-click on worksheet 1
3. Select ”New/NCP”
4. Double-click the value of the new NCP node and select an NCP DLL.
5. Right-click on document 1 in worksheet 1
6. Select ”Send using NCP”

This process simply adds an NCP to a worksheet, and uses that NCP to send the document to the network defined by the parameters of the NCP (by expanding the NCP node and setting the parameters).

This is only possible if the NCP is a valid and present DLL.

If the message is sent correctly, and the response received correctly, the icon of the sent document will change to an arrow pointing right (indicating that it is an outbound document), and a new entry will be inserted into the document collection directly underneath the sent message. The new entry will have an icon with an arrow pointing left (indicating an inbound document).