CODE MAGUS

# Eresia Network Injection Portal (NIP) Version 2.1

## CML00038-21

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

IBM Business Partner

August 23, 2010

# Contents

# 1 Introduction

## 1.1 Overview

The Network Injection Portal (NIP) is a Graphical User Interface (GUI) together with a number of elements to which Eresia Thistle scripts can interact.

Within the GUI, a Workspace allows the user to manipulate network messages, view logs containing these messages and record scripts for the Eresia Test environment.

Customers can configure or have configured meta-data objects using the Code Magus object types[1] schema which uses, for example, COBOL copybook layouts. When used in the GUI, these meta-data objects can be used to view, edit, search and filter the log records in terms of the elements of the meta-data. Additionally, the GUI supports querying the contents of the logs in terms of arbitrary expressions over the log records expressed in terms of the meta-data elements.

From the GUI, messages can be viewed, edited, created, deleted and sent over a network to a server, from which the response may also be viewed.

On exit from the GUI, the NIP saves all information in the Workspace to a user named file, which can be re-opened later in order to continue working. The Workspace could also be saved in a repository and/or shared, in which case is serves as a container of canned queries and configurations.

The GUI also has the ability to record Eresia scripts. If the NIP GUI is launched from within Eresia as a result of invoking the record function, then nominated records can be recorded to Eresia. These scripts compose and interpret the indicated messages in terms of the supplied meta-data configurations.

In order to interact with the network, a network abstraction layer is used. Components that implement the interface to this layer are termed Network Control Programs, or NCPs. There are a number of standard NCPs available, and an interface description and samples are included allowing the customer to implement their own.

When used for testing, NIP scripts are executed within Eresia.

The NIP GUI is divided into two sections. The first section contains a hierarchical view of the elements contained in the NIP 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 element in the hierarchy and the second column contains the value of the element.

Section 3 on page 10 onwards contains a reference of all of the elements of the NIP. Information is provided on how all of the individual elements of the NIP work.

## 1.2 Network Messages and the NIP

The purpose of the NIP is to manipulate and make use of network messages. These messages are identified and mapped by Object Type Definitions defined in Object Type Definition Files and the corresponding COBOL copybooks. These Object Type Definitions define which fields are used to determine the type of a message. The entire message is mapped using the fields and layouts defined in the copybooks and can be viewed as a hierarchy of messages and fields. New messages can be created from one of the defined message types. The Object Type Definitions can be viewed in the NIP.

An external file (or log) containing messages and identified through a Candidate Log File may be viewed in the hierarchy of the NIP Workspace. These messages are automatically typed using the Object Types Definition File named for the Workspace. This message log is loaded into the NIP via an Open Spec library.

In the NIP, messages may also be sent to, and the response received from, a server using a defined Network Control Program (NCP); See section 2.7.1 on page 9 for further details.

Messages may be excluded or included based on various predicates for viewing in the Workspace by defining a set of filters.

## 1.3 Message Types in the GUI

### 1.3.1 Apparent types

All messages that appear in the GUI are processed over an object types definition file, and the resulting object type is displayed in the right hand column next to the record. This processing is done automatically, and it is always done for any buffer in the GUI.

This type is the apparent type of the record.

### 1.3.2 Set types

A message can also be set to a specific type, and when this is done, the message will always behave as though it is an instance of the set type, even if the message has an apparent type which is different from the set type.

Any usage of the message will reflect that the message is of the set type. As mentioned, the set type supersedes the apparent type of a record.

See sub-section 4.12 on page 22 for details on how to assign the set type of a record.

## 1.4 summary

In summary, the NIP can be used to:

- View Network Messages in a hierarchical or tree form.
- Edit and change the content of a message.
- Create a new message from an existing message type.
- Use an Open Spec to read a Candidate Log File containing a log of messages.
- Use a Network Control Program to send and receive messages over a network.

# 2 Elements of the NIP

## 2.1 Overview

Various elements make up the hierarchical structure within the NIP Workspace.

This section explains the purpose of these elements, and how they are related. Each element is further explained starting at the technical reference; Section 3 on page 10.

The main elements are:

- The Workspace. Explained in section 2.2 on page 6.
- The Environment Variables collection. Explained in section 2.3 on page 6.
- The Message Types collection. Explained in section 2.4 on page 7.
- The Query collection. Explained in section 2.5 on page 7.
- The Candidate log file(s). Explained in section 2.6 on page 8.
- The Worksheet collection. Explained in section 2.7 on page 8.

## 2.2 The Workspace

The Workspace is the root of the hierarchy displayed in the NIP GUI. All of the collections are contained within the Workspace. Each Workspace is described in a single text based file with a specific grammar to describe the contents of the Workspace. A Workspace can be saved and later re-opened.

As well as the collections, the Workspace contains the following elements:

- Description

  This is a free form text string used to describe the Workspace. It is often used to identify the purpose of the Workspace.

- The Object Types Definition File

  This element identifies the Object Types Definition File which specifies the Object Type Definitions, restrictions and typing information of messages in the Workspace.

  New messages can only be created from Object Types specified in this file.

## 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 The Message Types collection

The message types collection element contains a list of all of the messages contained in the Object Types Definition File. These message types are displayed in a tree view which makes it easy to determine which fields a particular message contains. The message types in this section can be used to create a new message within a Worksheet.

## 2.5 The Query Collection

The query collection contains a list of all of the queries stored in the Workspace. Each query is an element itself, and has its own behaviour depending on the type of query.

A query element has a name, which is simply a name the user assigns to a query.

A query also has sub-elements which depend on the type of query it is. There are three types of queries. They are:

1. Message Type Query. Used to include or exclude the given message types.
2. Like ASCII/EBCDIC Query. Used to include or exclude messages that match a given regular expression.
3. Object Types Query. Used to include or exclude messages that match the specified Object Types query.

When a query is created certain operations (depending on the query type) can be performed over the data in the Candidate Log File.

The following sub sections will describe in more detail the different types of queries. For examples of using these queries refer to

### 2.5.1 Message Type Query

The Message Type query includes or excludes messages from a Candidate Log File. First the operation, include or exclude, is selected and secondly the message types from the Object Type Definitions are selected.

### 2.5.2 Like ASCII/EBCDIC Query

This query will filter all messages that match a regular expression. The regular expression can be either ASCII or EBCDIC based. The messages that match the regular expression can be either excluded or included from the view.

### 2.5.3 Object Types Query

This query allows for the use of a real Query using the Object Types query language. First the base message type to perform the query on is selected using a right-click on the Message Type element, and then the query can be entered in the WHERE clause value.

Substitutions in place of field names can be made by specifying wild cards, which are preceded by a dollar sign. When a substitution is used more nodes are created below the query element. A field name can then be specified by right clicking on the substitution name and selecting it from the list that appears.

## 2.6 The Candidate Log File(s)

The Candidate Log File element is used to identify a physical file containing a set of messages of the types specified in the Object Types Definition File element and the name of an Open Spec object that will interpret and load the messages into the hierarchical view. The Candidate Log File element contains a message collection element under which the loaded messages will be displayed.

The Open Spec object identifies a program DLL and any required parameters. Its purpose is the interpretation and loading of the messages from the Candidate Log File.

## 2.7 The 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.

Each Worksheet contains a message collection which is used to store a list of messages of interest and as an area where the values of fields can be altered and saved.

The Worksheet can also be recorded to a script if it contains an NCP and later re-run under the Eresia Test Environment.

Other functions of the Worksheet are:

- Storing a list of messages.
- Creating new messages using the message types collection.
- Creating copies of messages using the Candidate Log File.
- Sending messages over a network using an NCP.
- Recording a script from the Eresia Test Environment.

A new Worksheet can be added to the Worksheet collection at any time. Saving the Workspace saves all the information recorded in a Worksheet.

### 2.7.1 The Network Control Program

A Network Control Program element refers to a program DLL. Once the program is identified and loaded, parameter options are presented and can be modified. Examples of parameters are the network address and port of the server.

The NCP also has a set of properties that are used after the connection has been established, but before the message has been sent and received from the server or target.

The NCP element can be specified at either a Worksheet collection level or within an individual Worksheet. The Worksheet collection NCP acts as a default NCP definition.

### 2.7.2 Viewing Message entries

A message entry is displayed as a tree structure, showing each field within the message. The message can be expanded to the leaf level to view all data within the buffer.

### 2.7.3 Editing and changing Message buffers

Messages can be edited by double-clicking the value of a field and entering the new value.

Once a Worksheet has been created messages can be created from the message types collection or copied from a Candidate Log File or from another message in any Worksheet.

Examples of the usage of messages is explained in detail in section 3 on page 10.

### 2.7.4 Sending Messages over a network

Once the NCP has been defined, the user can send and receive the response of individual messages within the Worksheet.

The response can be viewed in the hierarchy and. as any other message, be used to create new messages.

# 3 The Graphical User Interface

The Graphical User Interface (GUI) is a visual representation of the contents of a Workspace. Each of the elements that are visible in the GUI will be described in their own section (See section 4 on page 13).

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 elements available in the project, which are the Description, Message Types Collection, Query Collection, Worksheet Collection and Candidate Log Files.

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

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

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

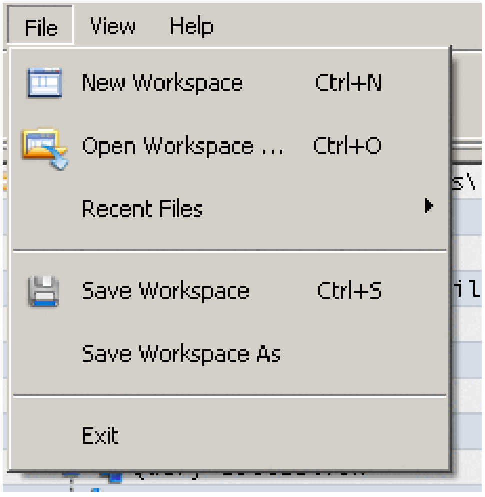
For example, if there is no description defined for the current Workspace, the user can add a description to the Workspace by right-clicking on the Workspace element and selecting ’Add’ and then ’Description’. A new entry will be added under the Workspace element for the description.

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

## 3.1 The Menu Bar

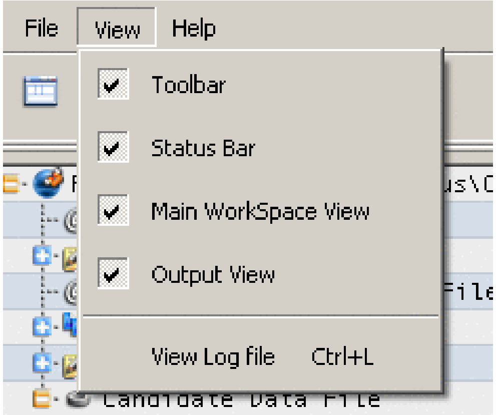
This section explains the layout of the menu bar. Individual menu items are also explained.

### 3.1.1 File



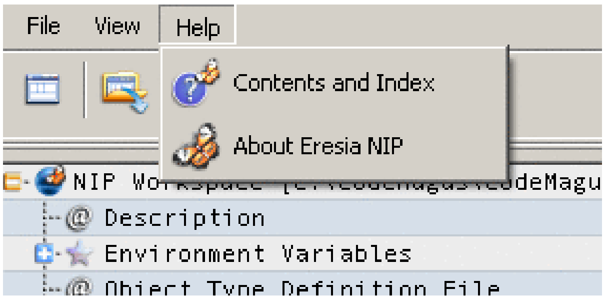
| Actions | |
|---|---|
| New Workspace | Create a new, empty Workspace |
| Open Workspace | Open an existing Workspace from file |
| Recent Files | Display and load a recently opened Workspace |
| Save Workspace | Save the current Workspace to file |
| Save Workspace As | Save the current Workspace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

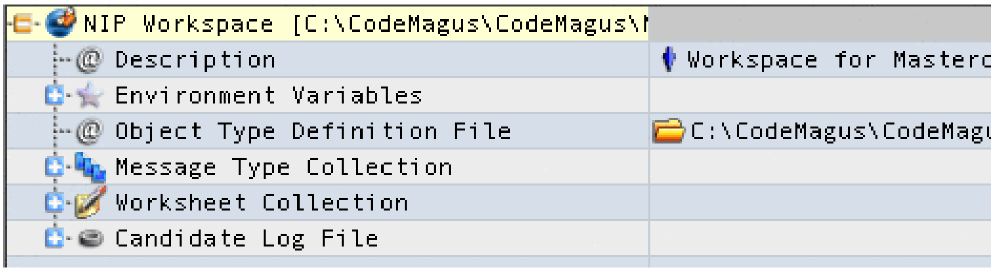


| Actions | |
|---|---|
| Contents and Index | Display the help file for the NIP |
| About Eresia NIP | Show the Help/About dialogue |

# 4 GUI Elements

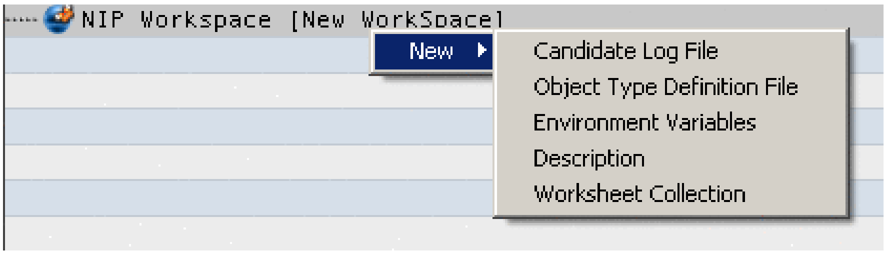
Each element of the GUI is described in this section, along with its actions and some examples of usage. The tables listed below contain descriptions on the actions (right-click options) for each element.

## 4.1 NIP Workspace



| | |
|---|---|
| NIP Workspace [C:\CodeMagus\CodeMagus\ | |
| Description | Workspace for Masterc |
| Environment Variables | |
| Object Type Definition File | C:\CodeMagus\CodeMagu |
| Message Type Collection | |
| Worksheet Collection | |
| Candidate Log File | |

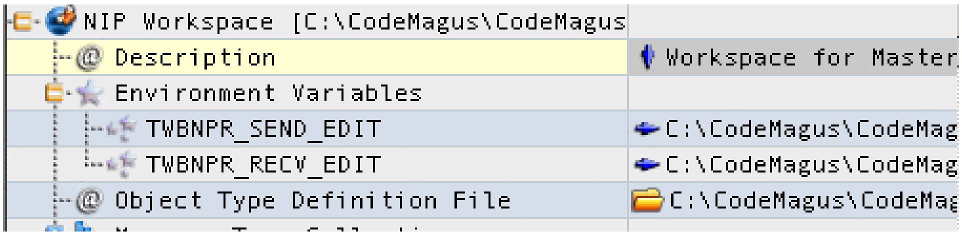
The NIP Workspace element contains all of the other elements pertaining to a project. It, and all the items below it, can be saved to a file; normally with a suffix of ".npw".



| Actions | |
|---|---|
| New/Candidate Log File | Add a log file containing messages to the Workspace |
| New/Object Type Definition File | Add an Object Type Definition File |
| New/Description | Add a description of the Workspace |
| New/Query Collection | Add collection of queries to the Workspace |
| New/Worksheet Collection | Add collection of Worksheets to the Workspace |
| View Object Type Definition File | Display the OTDF in a window |

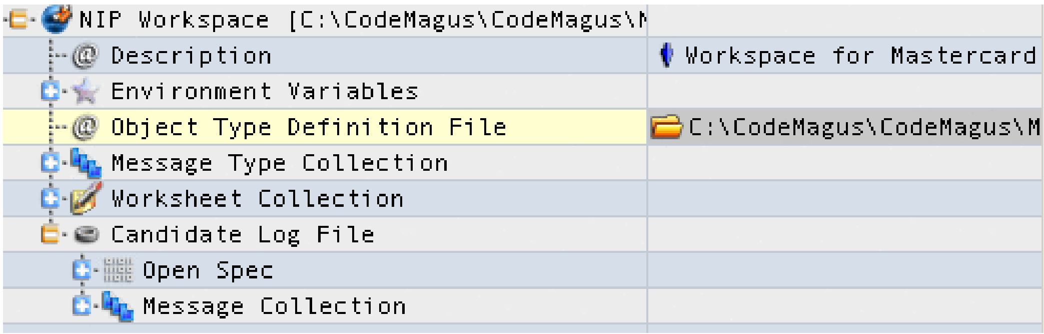
## 4.2 Description

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



| | |
|---|---|
| NIP Workspace [C:\CodeMagus\CodeMagus | |
| Description | Workspace for Master |
| Environment Variables | |
| TWBNPR_SEND_EDIT | C:\CodeMagus\CodeMag |
| TWBNPR_RECV_EDIT | C:\CodeMagus\CodeMag |
| Object Type Definition File | C:\CodeMagus\CodeMag |

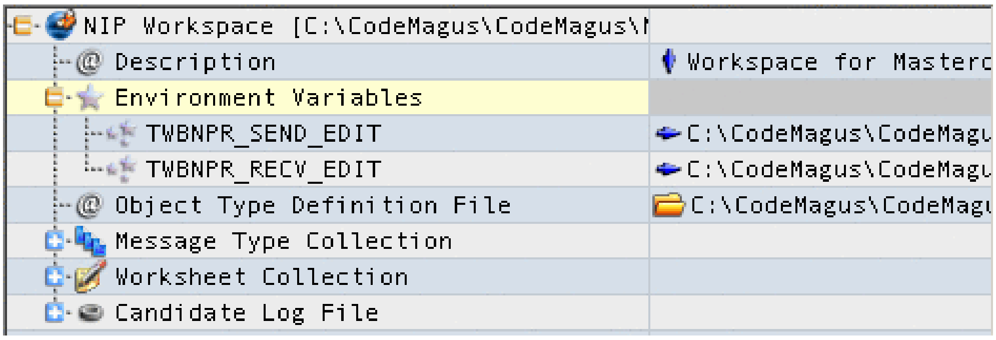
## 4.3 Object Types Definition File



| | |
|---|---|
| NIP Workspace [C:\CodeMagus\CodeMagus\M | |
| Description | Workspace for Mastercard |
| Environment Variables | |
| Object Type Definition File | C:\CodeMagus\CodeMagus\M |
| Message Type Collection | |
| Worksheet Collection | |
| Candidate Log File | |
| Open Spec | |
| Message Collection | |

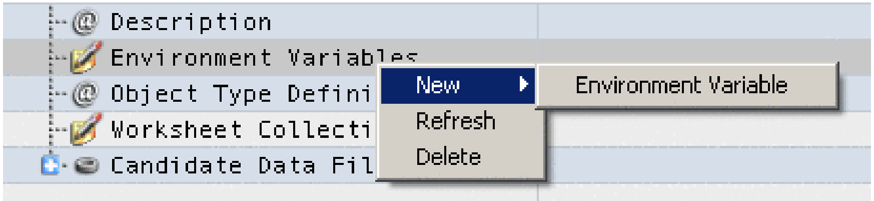
The Object Types Definition File specifies the file which acts as the source of the definition of the message types which will be used in the Workspace. When an object types file is defined, an entry will appear under the Workspace which is called Message Types Collection. See section 4.6 on page 15.

## 4.4 Environment Variables collection



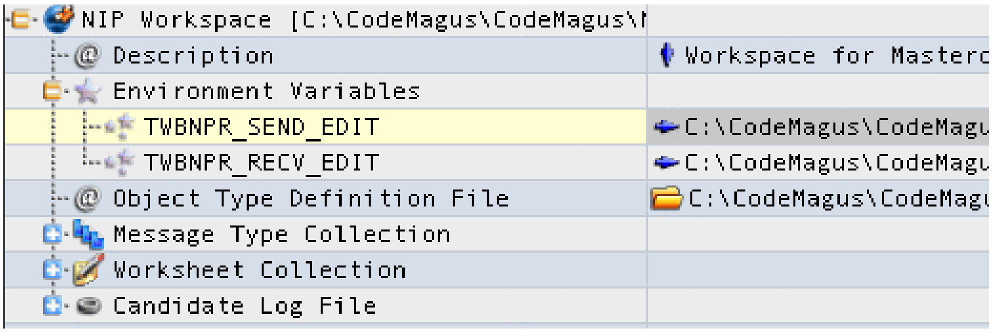
| | |
|---|---|
| NIP Workspace [C:\CodeMagus\CodeMagus\M | |
| Description | Workspace for Masterc |
| Environment Variables | |
| TWBNPR_SEND_EDIT | C:\CodeMagus\CodeMagu |
| TWBNPR_RECV_EDIT | C:\CodeMagus\CodeMagu |
| Object Type Definition File | C:\CodeMagus\CodeMagu |
| Message Type Collection | |
| Worksheet Collection | |
| Candidate Log File | |

The Environment Variables Collection contains a list of all defined environment variables. Each of these has its own entry. See section 4.5 on page 15.

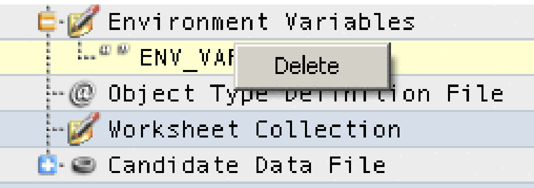


| Actions | |
|---|---|
| New/Environment Variable | Add a new entry |
| Refresh | Refresh the list of variables |
| Delete | Remove this element |

## 4.5 Environment Variable

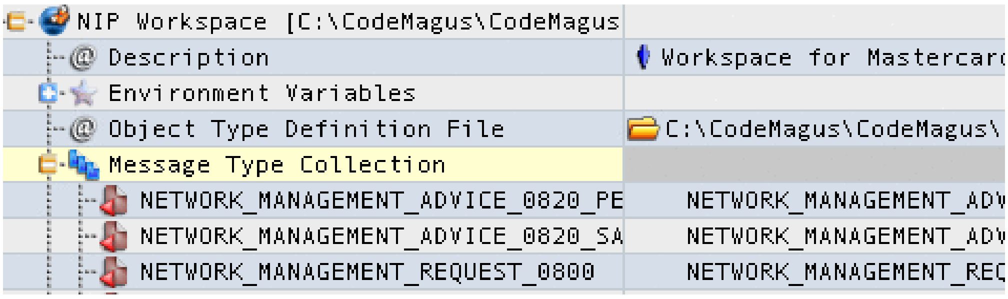


The Environment Variable node has a name and a value. The name is the actual name of the environment variable to set, and the value is the actual value.



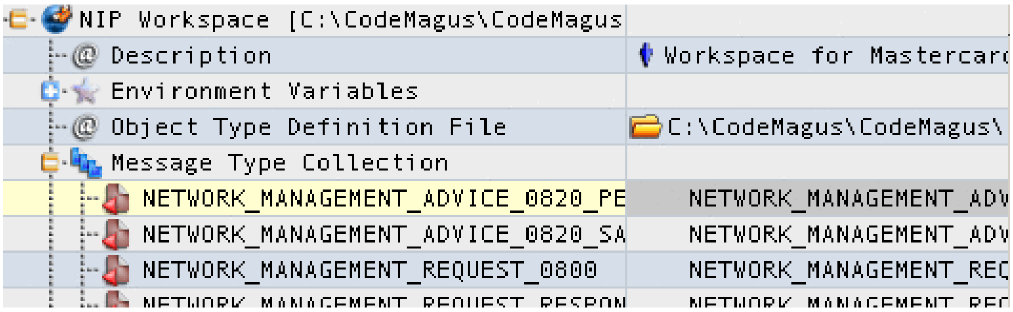
| Actions | |
|---|---|
| Delete | Remove this element |

## 4.6 Message Types Collection



The Message Types Collection contains a list of all defined message types within the specified Object Types Definition File. Each of these message type entries can be used to get more information about individual types defined in the object types definition file. See section 4.7 on page 16

## 4.7 Message Type



NIP Workspace [C:\CodeMagus\CodeMagus

Description — Workspace for Mastercar

Environment Variables

Object Type Definition File — C:\CodeMagus\CodeMagus\

Message Type Collection

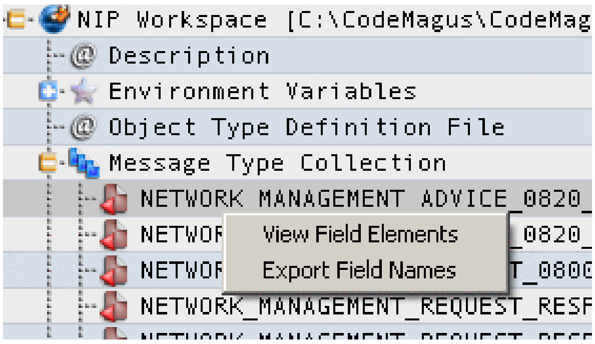
NETWORK_MANAGEMENT_ADVICE_0820_PE — NETWORK_MANAGEMENT_ADV

NETWORK_MANAGEMENT_ADVICE_0820_SA — NETWORK_MANAGEMENT_ADV

NETWORK_MANAGEMENT_REQUEST_0800 — NETWORK_MANAGEMENT_REQ

NETWORK_MANAGEMENT_REQUEST_RESPON — NETWORK_MANAGEMENT_REQ

A Message Type element has the type's name, and can be used to view a full message of that type. These message types can be used to create a new, empty message if a type is dragged to a Worksheet, or if a user performs a Copy/Paste of a message type into a Worksheet.



NIP Workspace [C:\CodeMagus\CodeMag

Description

Environment Variables

Object Type Definition File

Message Type Collection

NETWORK_MANAGEMENT_ADVICE_0820_

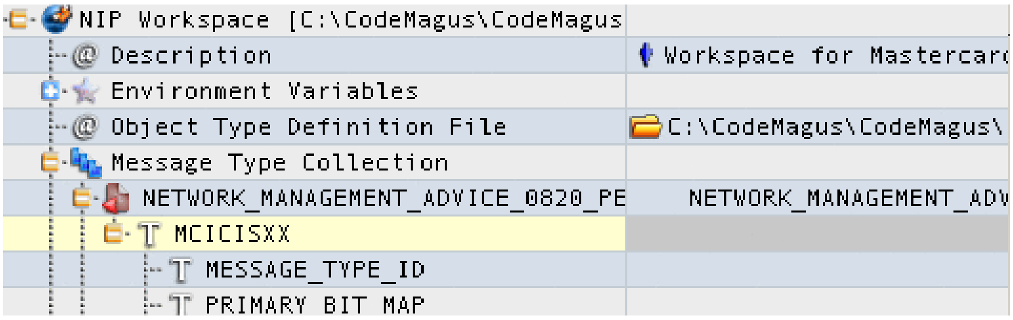
NETWOR ... 0820_

NETWOR ... T_0800

NETWORK_MANAGEMENT_REQUEST_RESP

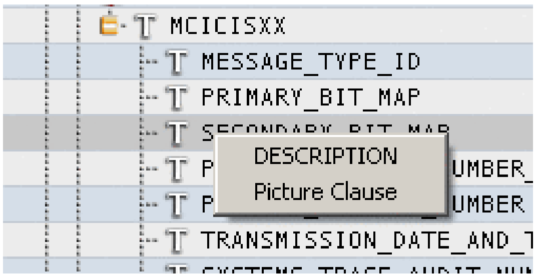
| Actions | |
|---|---|
| View Field Elements | Populate the fields of the message in the tree |
| Export Field Names | Display a list of field names of the message |

## 4.8 Message Type Field



| | |
|---|---|
| NIP Workspace [C:\CodeMagus\CodeMagus | |
| Description | Workspace for Mastercar |
| Environment Variables | |
| Object Type Definition File | C:\CodeMagus\CodeMagus\ |
| Message Type Collection | |
| NETWORK_MANAGEMENT_ADVICE_0820_PE | NETWORK_MANAGEMENT_ADV |
| MCICISXX | |
| MESSAGE_TYPE_ID | |
| PRIMARY BIT MAP | |

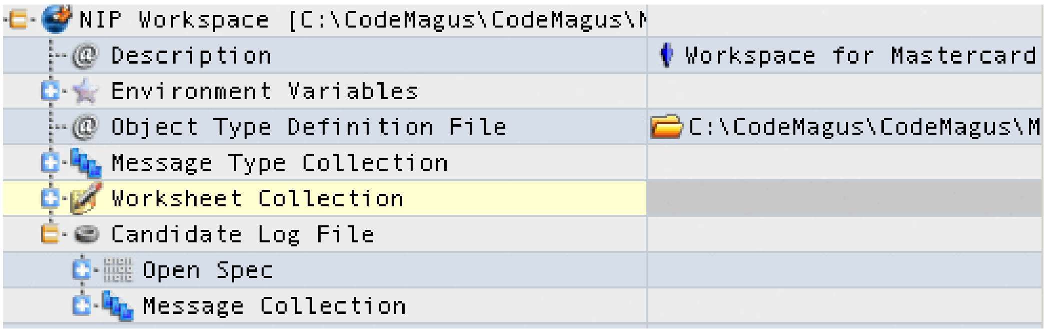
Each field within a message type displays the name of the field as it is defined in the object type and copy book. This field shows the name of the field. The value of this entry will always be empty since this element has no buffer. It serves only as a means for the user to view which fields and sub-fields are contained within a given type. See section 4.13 on page 23 for an example of an editable field.



MCICISXX
MESSAGE_TYPE_ID
PRIMARY_BIT_MAP
SECONDARY_BIT_MAP
P UMBER_
P UMBER
TRANSMISSION_DATE_AND_T

| Actions | |
|---|---|
| DESCRIPTION | Display the description of the field as per the copybook |
| Picture Clause | Display the picture clause of the field as per the copybook |

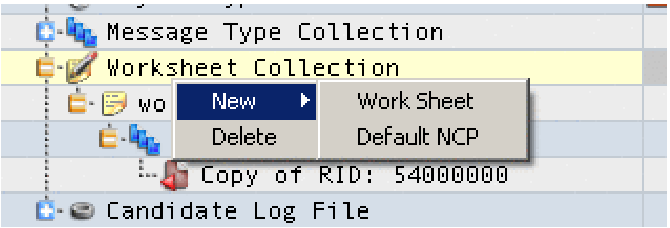
## 4.9 Worksheet Collection



| | |
|---|---|
| NIP Workspace [C:\CodeMagus\CodeMagus\M | |
| Description | Workspace for Mastercard |
| Environment Variables | |
| Object Type Definition File | C:\CodeMagus\CodeMagus\M |
| Message Type Collection | |
| Worksheet Collection | |
| Candidate Log File | |
| Open Spec | |
| Message Collection | |

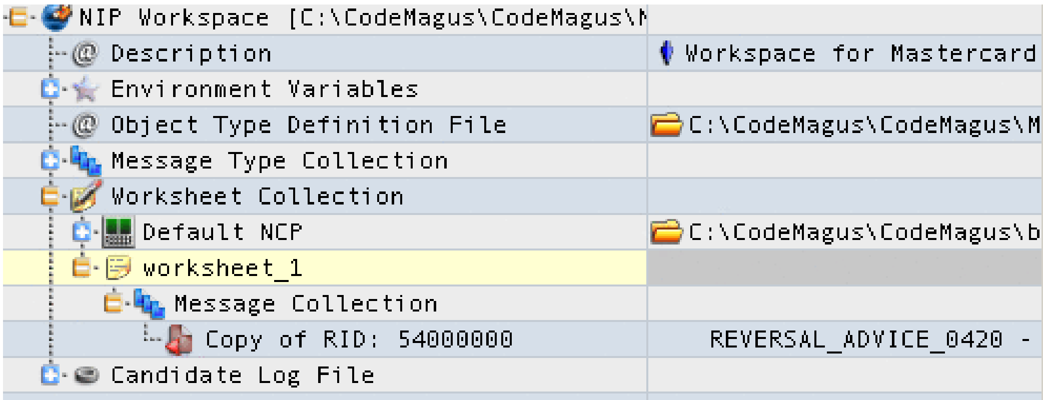
A Worksheet Collection contains a list of Worksheet elements (see section 4.10 on page 18).

The Worksheet Collection can be used to add new Worksheets to the Workspace, and also to set a default NCP. This default NCP is used when a NCP is not set for an individual Worksheet.



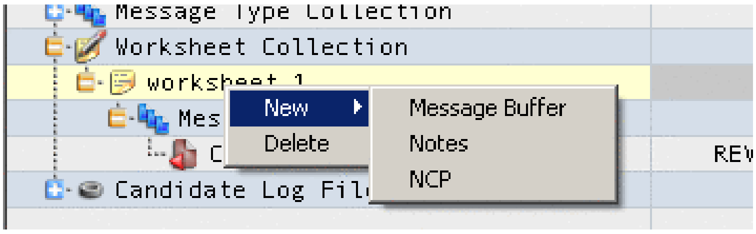
| Actions | |
|---|---|
| New/Work Sheet | Add a new Worksheet to this Worksheet Collection |
| New/Default NCP | Add a default NCP that applies to all Worksheets |
| Delete | Delete the Worksheet Collection |

## 4.10 Worksheet


NIP Workspace [C:\CodeMagus\CodeMagus\N
Description Workspace for Mastercard
Environment Variables
Object Type Definition File C:\CodeMagus\CodeMagus\M
Message Type Collection
Worksheet Collection
Default NCP C:\CodeMagus\CodeMagus\b
worksheet_1
Message Collection
Copy of RID: 54000000 REVERSAL_ADVICE_0420 -
Candidate Log File

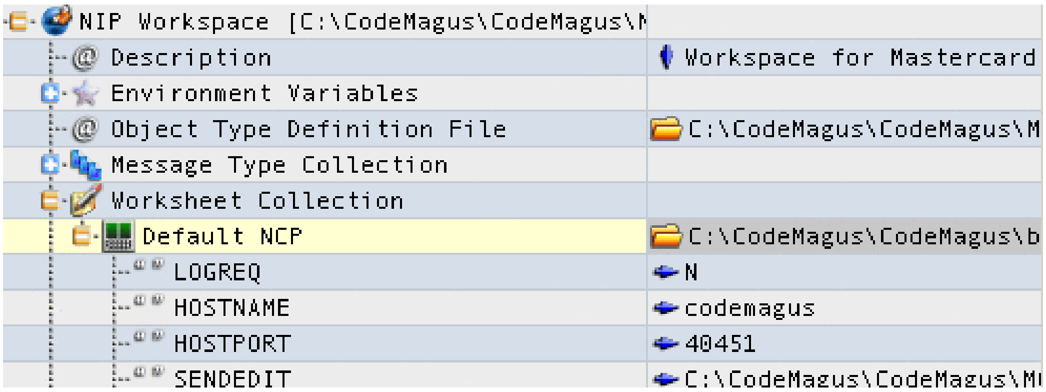
Worksheets are used to store a collection of records which are copied from a Candidate Log File. These records are physically stored in the workspace file when the workspace is saved.

Worksheets can be recorded if an NCP has been defined for the worksheet, or if a Default NCP has been defined in the Worksheet Collection.



| Actions | |
|---|---|
| New/Message Buffer | Add an empty record buffer to the Worksheet |
| New/Notes | Add a text note to the Worksheet |
| New/NCP | Set the NCP for this Worksheet |
| Delete | Remove this Worksheet from the Worksheet Collection |
| Record | Record this Worksheet in the Automated Test Environment |

## 4.11 Network Control Program

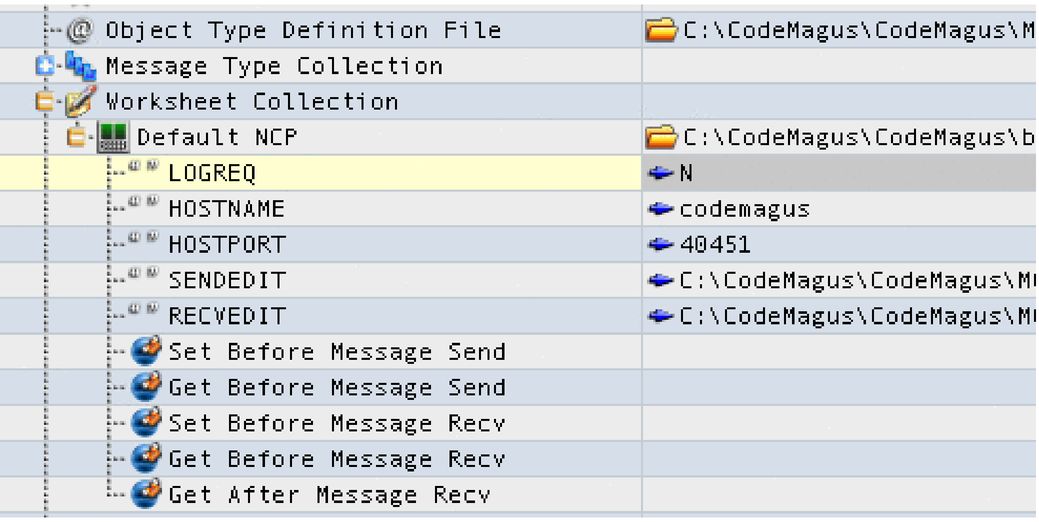


| | |
|---|---|
| NIP Workspace [C:\CodeMagus\CodeMagus\N | |
| Description | Workspace for Mastercard |
| Environment Variables | |
| Object Type Definition File | C:\CodeMagus\CodeMagus\M |
| Message Type Collection | |
| Worksheet Collection | |
| Default NCP | C:\CodeMagus\CodeMagus\b |
| LOGREQ | N |
| HOSTNAME | codemagus |
| HOSTPORT | 40451 |
| SENDEDIT | C:\CodeMagus\CodeMagus\M |

The Network Control Program (NCP) allows the user to specify the parameters and the attributes used in the NCP to connect to a remote point. Parameters are used to specify information about the connection. Attributes are used to specify information of interest before and after the sending and receiving operations of the NCP.

Both the parameters and the attributes are specified in the DLL, and are automatically populated in the GUI.

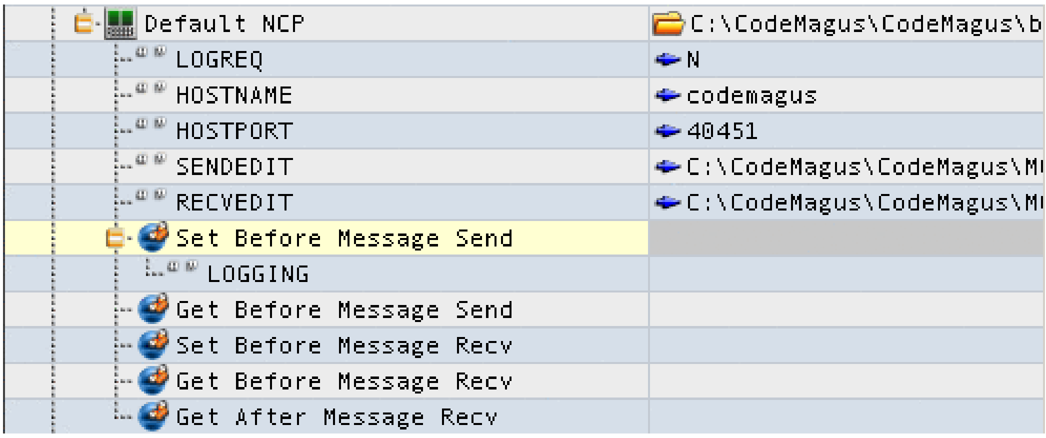
### 4.11.1 NCP Parameter



| | |
|---|---|
| Object Type Definition File | C:\CodeMagus\CodeMagus\M |
| Message Type Collection | |
| Worksheet Collection | |
| Default NCP | C:\CodeMagus\CodeMagus\b |
| LOGREQ | N |
| HOSTNAME | codemagus |
| HOSTPORT | 40451 |
| SENDEDIT | C:\CodeMagus\CodeMagus\M |
| RECVEDIT | C:\CodeMagus\CodeMagus\M |
| Set Before Message Send | |
| Get Before Message Send | |
| Set Before Message Recv | |
| Get Before Message Recv | |
| Get After Message Recv | |

The required and optional parameters are simply populated in a list. These elements cannot be added or removed as this is done automatically by retrieving information from the NCP DLL itself.

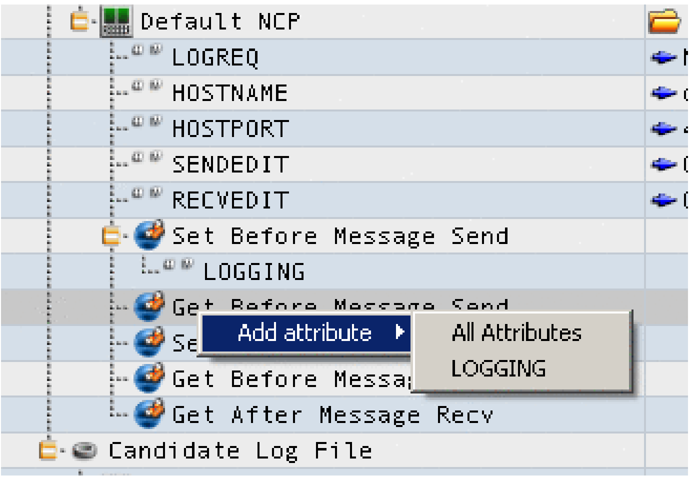
### 4.11.2 NCP Attribute Group



| | |
|---|---|
| Default NCP | C:\CodeMagus\CodeMagus\b |
| LOGREQ | N |
| HOSTNAME | codemagus |
| HOSTPORT | 40451 |
| SENDEDIT | C:\CodeMagus\CodeMagus\M |
| RECVEDIT | C:\CodeMagus\CodeMagus\M |
| Set Before Message Send | |
| LOGGING | |
| Get Before Message Send | |
| Set Before Message Recv | |
| Get Before Message Recv | |
| Get After Message Recv | |

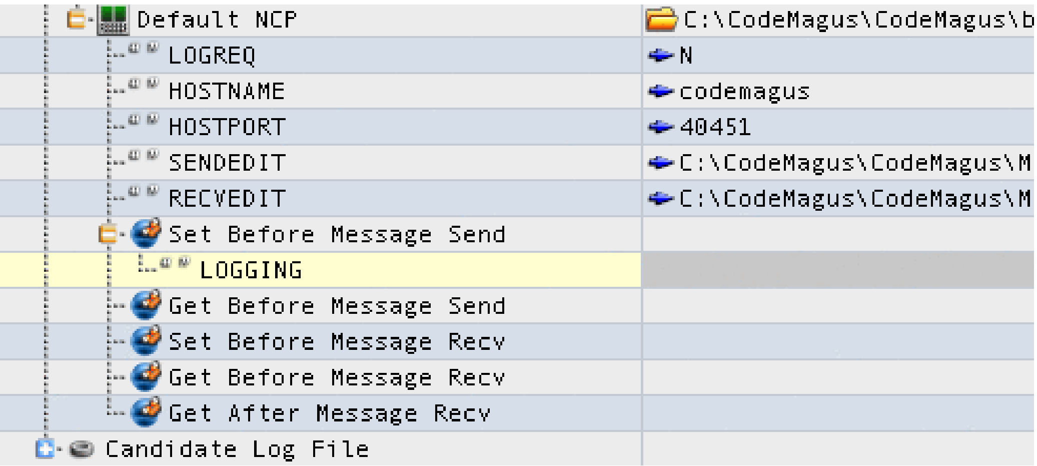
The Attributes area displays a list of all of the attributes used during the usage of the NCP. These parameters allow the user to specify which attributes are set and retrieved before and after the send and receive operations. These set and get operations are performed after the NCP network circuit is opened.

The user can specify which attributes are to be set and retrieved by right-clicking on the appropriate node and selecting it from the pop-up list. The user can also select all attributes. To delete an entry, the user can simply right-click on the attribute and select delete.



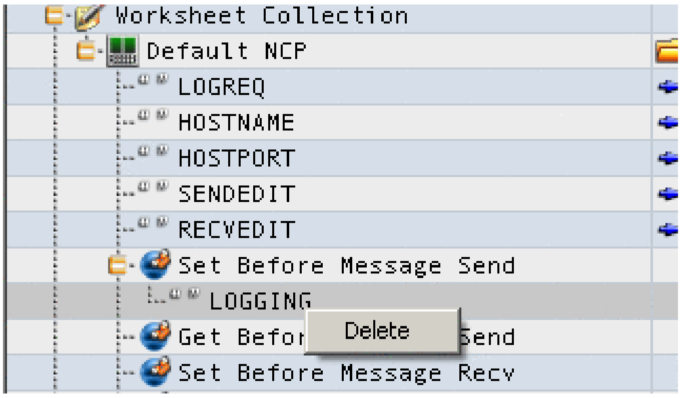
| Actions | |
|---|---|
| Add attribute/All Attributes | Add all attributes to the node |
| Any other entry | Add only this attribute to the node |

### 4.11.3 NCP Attribute



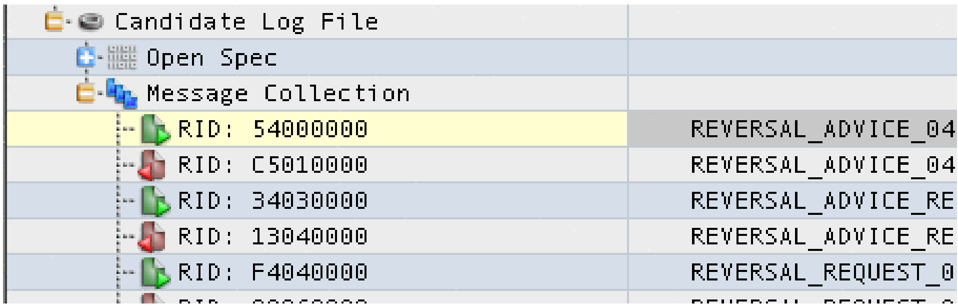
```
Default NCP      C:\CodeMagus\CodeMagus\b
LOGREQ           N
HOSTNAME         codemagus
HOSTPORT         40451
SENDEDIT         C:\CodeMagus\CodeMagus\M
RECVEDIT         C:\CodeMagus\CodeMagus\M
```

This node is simply one instance of an attribute in one of the NCP Attribute Groups. The menu here simply allows the user to delete this entry from its parent node.

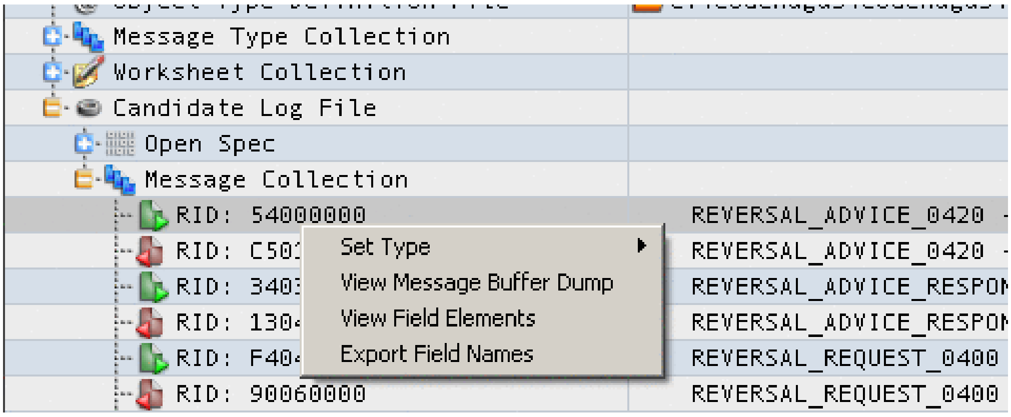


| Actions | |
|---|---|
| Delete | Remove this message from its parent node |

## 4.12 Message


Candidate Log File
Open Spec
Message Collection
RID: 54000000 REVERSAL_ADVICE_04
RID: C5010000 REVERSAL_ADVICE_04
RID: 34030000 REVERSAL_ADVICE_RE
RID: 13040000 REVERSAL_ADVICE_RE
RID: F4040000 REVERSAL_REQUEST_0

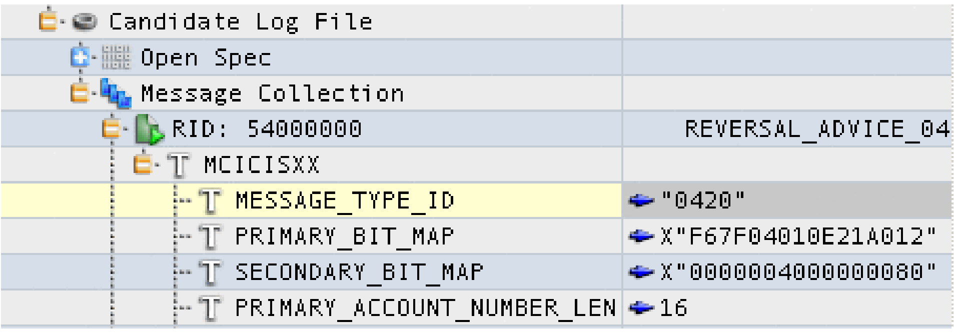
The Message element is a representation of a buffer of data which conforms to a type defined in the object types. This element has different actions available depending on its position in the tree.


Message Type Collection
Worksheet Collection
Candidate Log File
Open Spec
Message Collection
RID: 54000000 REVERSAL_ADVICE_0420
RID: C501 REVERSAL_ADVICE_0420
RID: 340 REVERSAL_ADVICE_RESPON
RID: 130 REVERSAL_ADVICE_RESPON
RID: F40 REVERSAL_REQUEST_0400
RID: 90060000 REVERSAL_REQUEST_0400
Set Type
View Message Buffer Dump
View Field Elements
Export Field Names

| Actions | |
|---|---|
| Set Type | Assign the Set Type of a record to a specific message type |
| Send Using NCP | Send this message over the network |
| View Message Buffer Dump | View the actual buffer of the message |
| View Field Elements | Populate the fields of the message in the tree |
| Export Field Names | Display a list of field names of the message |
| Delete | Remove this message from its parent Message Collection |

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

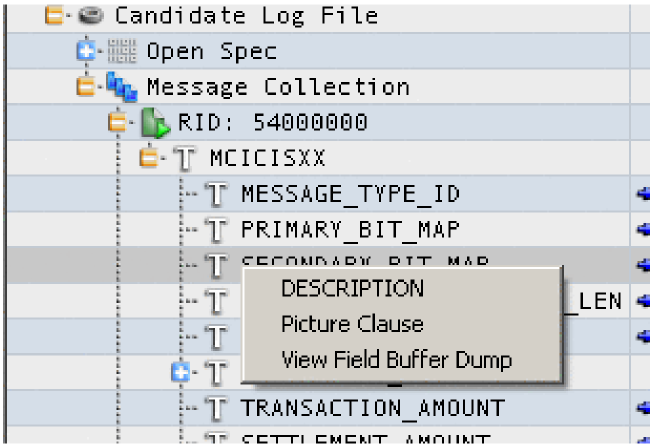
## 4.13 Message Field



```
Candidate Log File
  Open Spec
  Message Collection
    RID: 54000000                       REVERSAL_ADVICE_04
      MCICISXX
        MESSAGE_TYPE_ID                 "0420"
        PRIMARY_BIT_MAP                 X"F67F04010E21A012"
        SECONDARY_BIT_MAP               X"0000004000000080"
        PRIMARY_ACCOUNT_NUMBER_LEN      16
```

Each field within a message displays the name of the field as it is defined in the object type and copy book. This field name can not be changed, but the value of the data within this field can be changed if and only if this field is contained within a message which is in a Worksheet. This allows the user to change the data and the actual buffer of the message.

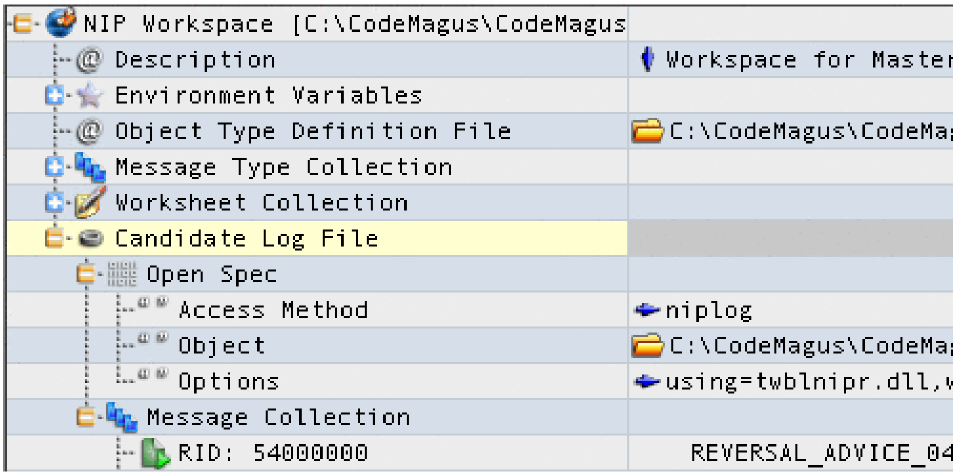
A field may also contain sub-fields. These sub fields can be displayed by expanding the node.



```
Candidate Log File
  Open Spec
  Message Collection
    RID: 54000000
      MCICISXX
        MESSAGE_TYPE_ID
        PRIMARY_BIT_MAP
        SECONDARY_BIT_MAP
                                  LEN
        TRANSACTION_AMOUNT
        SETTLEMENT_AMOUNT
```

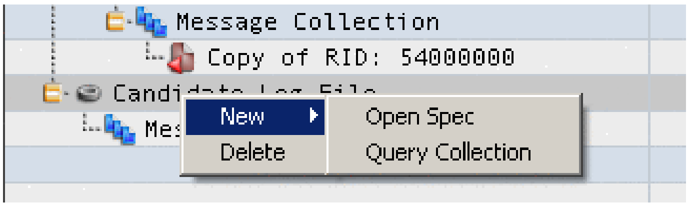
| Actions | |
|---|---|
| DESCRIPTION | Display the description of the field as per the copybook |
| Picture Clause | View the picture clause of this field |
| View Field Buffer Dump | View the buffer dump of this field |

## 4.14 Candidate Log File



```
NIP Workspace [C:\CodeMagus\CodeMagus
  Description                       Workspace for Master
  Environment Variables
  Object Type Definition File       C:\CodeMagus\CodeMa
  Message Type Collection
  Worksheet Collection
  Candidate Log File
    Open Spec
      Access Method                 niplog
      Object                        C:\CodeMagus\CodeMa
      Options                       using=twblnipr.dll,
    Message Collection
      RID: 54000000                 REVERSAL_ADVICE_04
```

A Candidate Log File (CLF) is a data file containing a number of messages, in a specific format. This file is read using an Open Spec object.



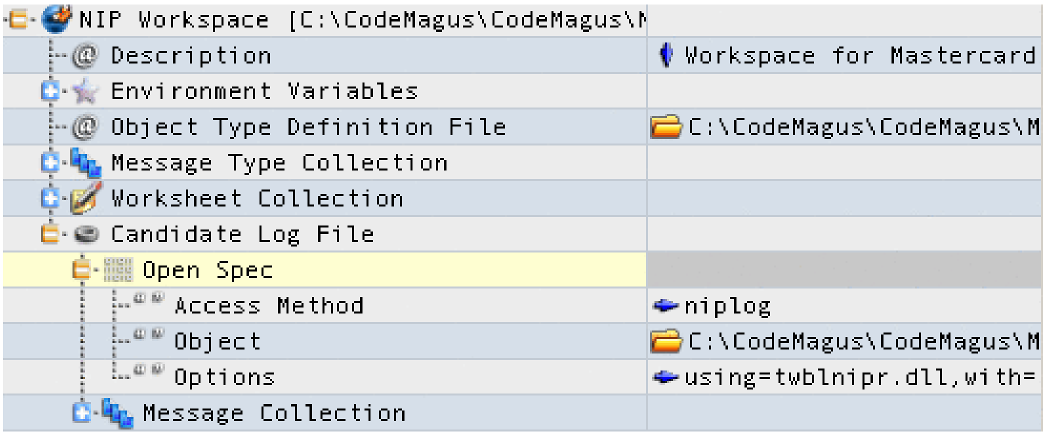
| Actions | |
|---|---|
| New/Open Spec | Add an Open Spec object |
| New/Query Collection | Add a set of queries |

The Open Spec under the Candidate Log File specifies the parameters used to read the data file. If the Open Spec successfully opens and reads the data file, then the messages that are contained inside that data file will be added to the Candidate Data File element under a 'Message Collection' element. See section 4.15 on page 25

Each of these can be viewed, displayed and used to generate new messages in a Worksheet, but can not be edited or changed (See section 4.12 on page 22).

The Candidate Log File can also contain a Query Collection (see section 4.16 on page 26), which contains a list of queries that can be executed on this Candidate Log File.

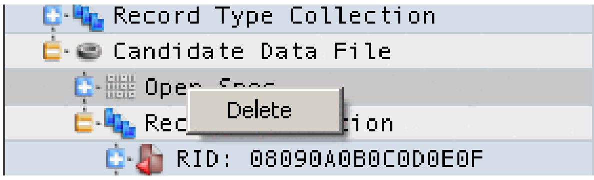
## 4.15 Open Spec



| | |
|---|---|
| NIP Workspace [C:\CodeMagus\CodeMagus\N | |
| Description | Workspace for Mastercard |
| Environment Variables | |
| Object Type Definition File | C:\CodeMagus\CodeMagus\M |
| Message Type Collection | |
| Worksheet Collection | |
| Candidate Log File | |
| Open Spec | |
| Access Method | niplog |
| Object | C:\CodeMagus\CodeMagus\M |
| Options | using=twblnipr.dll,with= |
| Message Collection | |

This element simply allows the user to set up the parameters to access the Candidate Log File (see section 4.14 on page 24).

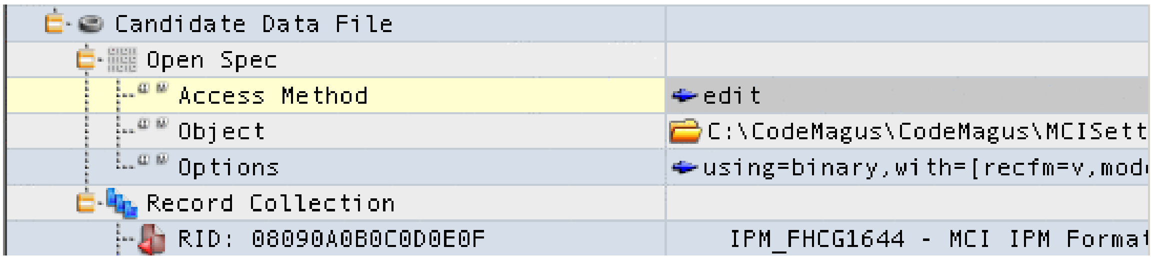
The Open Spec has three properties. The Access Method property defines which access method to use to read the data file. The Object property specifies the full path of the data file. And the Options property is used to define the parameters to pass to the Open Spec library. Further details are supplied in the sub-sections that follow.



Record Type Collection
Candidate Data File
Open Spec
Record Collection
RID: 08090A0B0C0D0E0F

| Actions | |
|---|---|
| Delete | Remove this Open Spec from its parent |

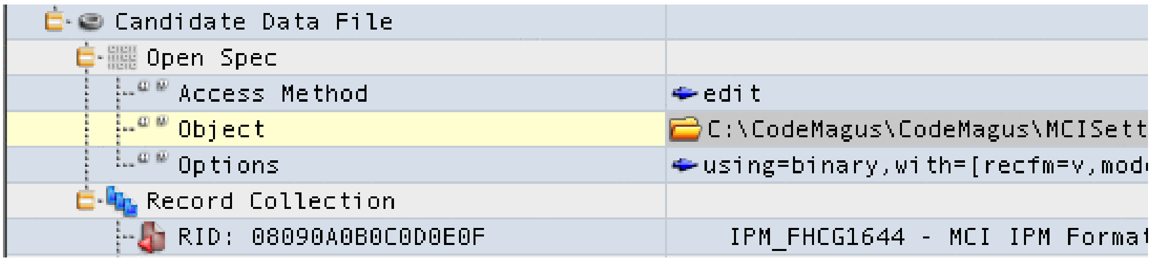
### 4.15.1 Access Method



| | |
|---|---|
| Candidate Data File | |
| Open Spec | |
| Access Method | edit |
| Object | C:\CodeMagus\CodeMagus\MCISett |
| Options | using=binary,with=[recfm=v,mod |
| Record Collection | |
| RID: 08090A0B0C0D0E0F | IPM_FHCG1644 - MCI IPM Format |

This element is used to specify which access method to use to read from or write to a data file (log). The name supplied here specifies the name of the AMD file to use to look up the required information. Some options for this item are FILE, BINARY and NIPLOG.

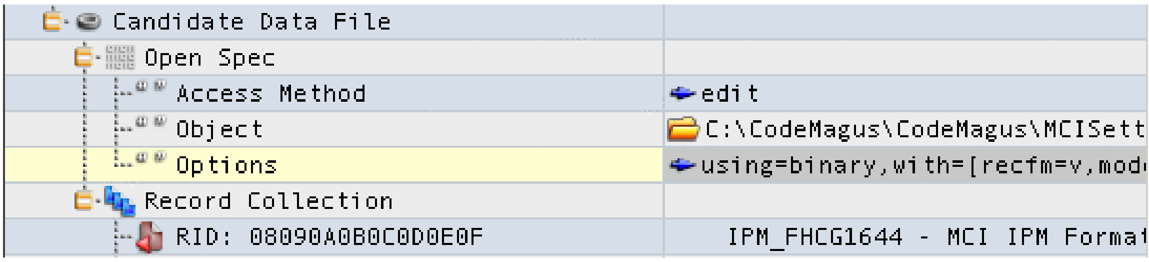
### 4.15.2 Object



| | |
|---|---|
| Candidate Data File | |
| Open Spec | |
| Access Method | edit |
| Object | C:\CodeMagus\CodeMagus\MCISett |
| Options | using=binary,with=[recfm=v,mod |
| Record Collection | |
| RID: 08090A0B0C0D0E0F | IPM_FHCG1644 - MCI IPM Format |

This element is used to specify the full path of the Object which the Open Spec will access using the specified Access Method.

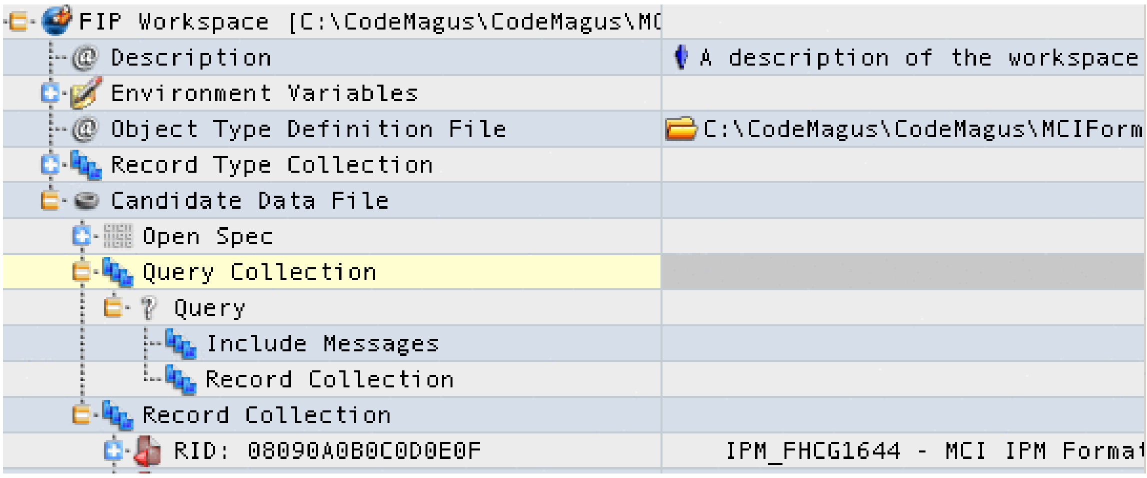
### 4.15.3 Options



| | |
|---|---|
| Candidate Data File | |
| Open Spec | |
| Access Method | edit |
| Object | C:\CodeMagus\CodeMagus\MCISett |
| Options | using=binary,with=[recfm=v,mod |
| Record Collection | |
| RID: 08090A0B0C0D0E0F | IPM_FHCG1644 - MCI IPM Format |

This element is used to specify the full set of options passed to the Open Spec when it is being opened for read or write access.

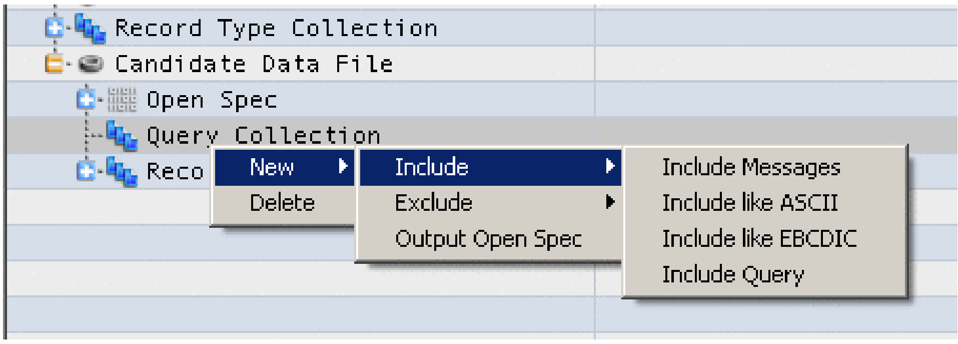
## 4.16 Query Collection



| | |
|---|---|
| FIP Workspace [C:\CodeMagus\CodeMagus\MC | |
| Description | A description of the workspace |
| Environment Variables | |
| Object Type Definition File | C:\CodeMagus\CodeMagus\MCIForm |
| Record Type Collection | |
| Candidate Data File | |
| Open Spec | |
| Query Collection | |
| Query | |
| Include Messages | |
| Record Collection | |
| Record Collection | |
| RID: 08090A0B0C0D0E0F | IPM_FHCG1644 - MCI IPM Format |

This element simply allows the user to add queries that can be performed on the Candidate Log File. See sections subsection 4.17.1 on page 28, subsection 4.17.2 on page 28 and subsection 4.17.3 on page 29 for more information on these query types.

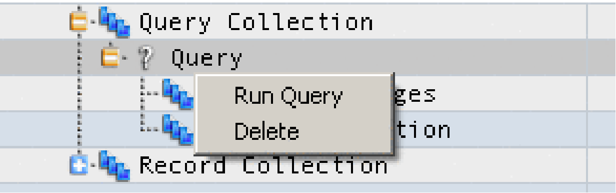
Note that when a Query Collection node is a child of the Workspace node, then no menu items will be accessible. The use of this collection in the workspace is simply to store queries. Queries can be copied here by copy/paste from other query collections.



| Actions | |
|---|---|
| New/Include/Messages | Add a message types include filter |
| New/Include/like ASCII | Add an ASCII include filter |
| New/Include/like EBCDIC | Add an EBCDIC include filter |
| New/Include/Query | Add an include Query |
| New/exclude/Messages | Add a message types exclude filter |
| New/exclude/like ASCII | Add an ASCII exclude filter |
| New/exclude/like EBCDIC | Add an EBCDIC exclude filter |
| New/exclude/Query | Add an exclude Query |
| Delete | Remove this query collection from its parent |

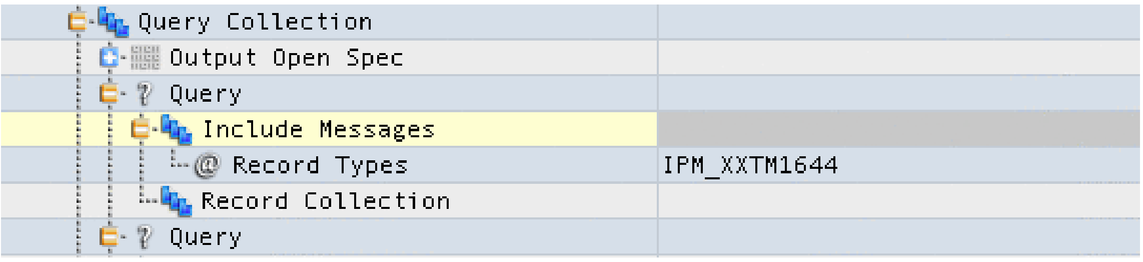
## 4.17 Queries

All Queries have the following menu options at their root node:



| Actions | |
|---|---|
| Run Query | Execute the query |
| Delete | Remove this query from the query collection |

### 4.17.1 Include/Exclude Messages Query

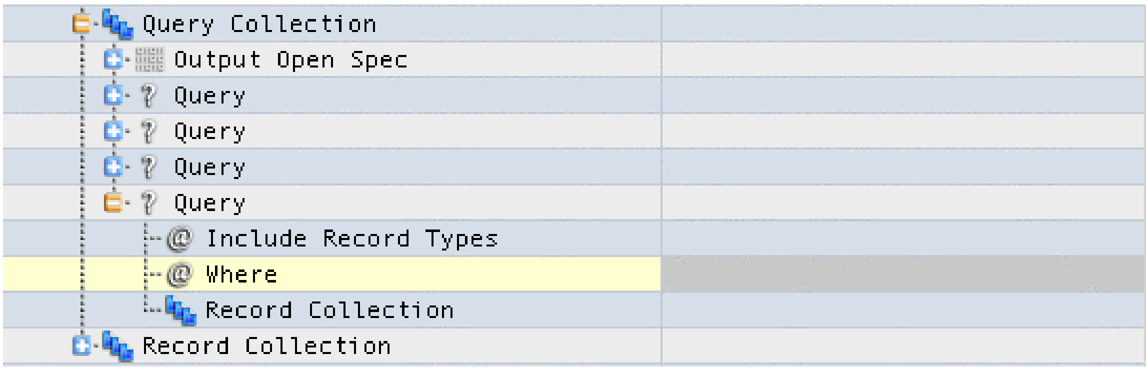


```
Query Collection
  Output Open Spec
  Query
    Include Messages
      Record Types          IPM_XXTM1644
    Record Collection
  Query
```

This Query allows the user to filter the messages in a Candidate Log File by the message type. The user can either include or exclude a list of message types. The user can add to this list by right-clicking on the node ’Message Types’ and selecting the message type from the list.

| Actions | |
|---|---|
| Run Query | Perform the query and populate the message list |
| Delete | Remove this query from the query collection |

### 4.17.2 Include/Exclude ASCII/EBCDIC Query

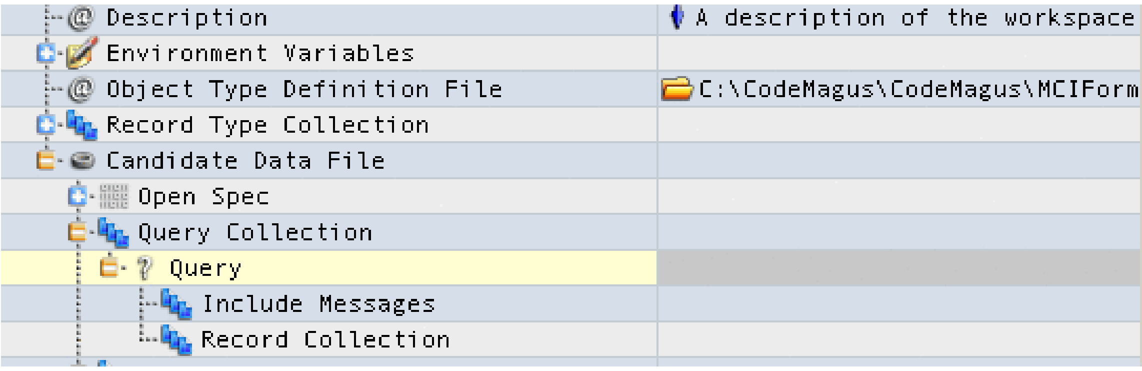


```
Query Collection
  Output Open Spec
  Query
  Query
  Query
  Query
    Include Record Types
    Where
    Record Collection
Record Collection
```

This Query allows the user to filter the messages by performing a regular expression on the buffer of the message. The user specifies this filter by setting the value of the search in the value of the ’Include ASCII’ or ’Include EBCDIC’ field.

| Actions | |
|---|---|
| Run Query | Perform the query and populate the message list |
| Delete | Remove this query from the query collection |

### 4.17.3 Include/Exclude Query



| | |
|---|---|
| Description | A description of the workspace |
| Environment Variables | |
| Object Type Definition File | C:\CodeMagus\CodeMagus\MCIForm |
| Record Type Collection | |
| Candidate Data File | |
| Open Spec | |
| Query Collection | |
| Query | |
| Include Messages | |
| Record Collection | |

This Query allows the user to filter the messages by performing a query on the Candidate Log File by using the `expeval` library. The user selects the message type by right-clicking on the Include Message Type node and selecting the type from the list.

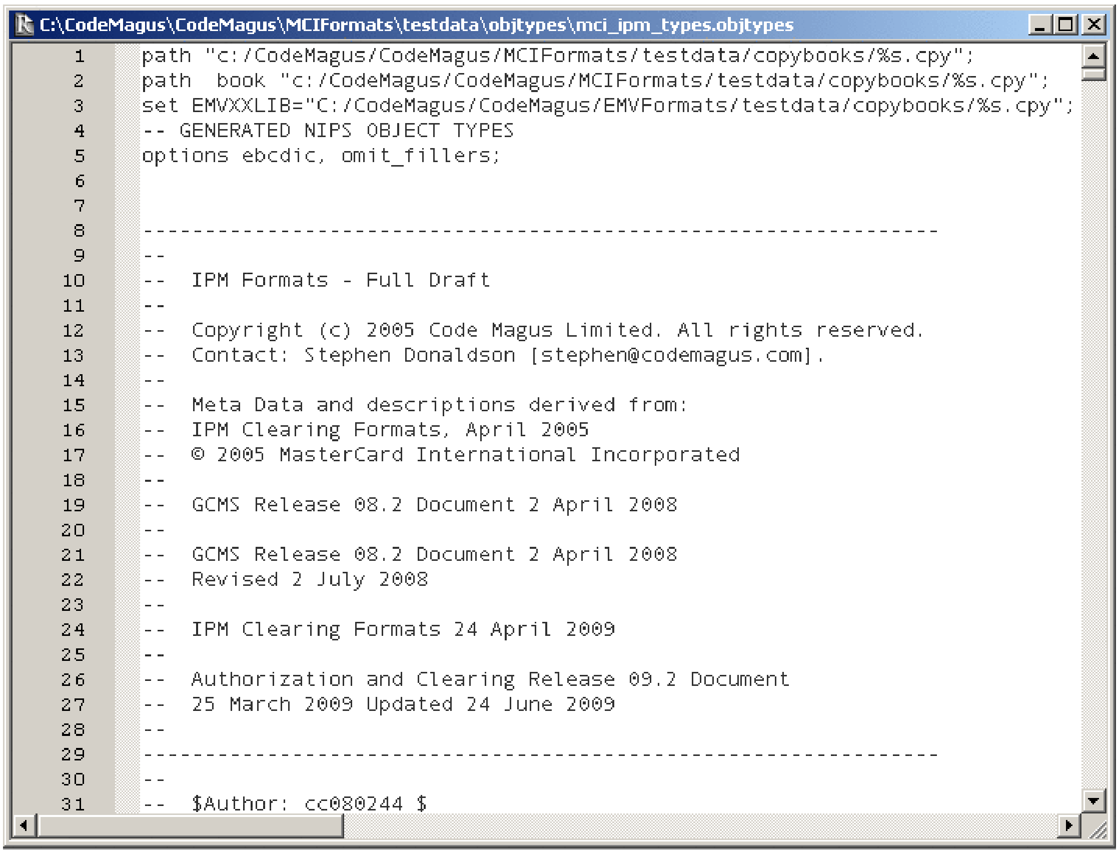
If the user specifies any substitutions in the search string, another node will appear under Substitutions. The user can then right click on each substitution and select the field name from the list.

| Actions | |
|---|---|
| Run Query | Perform the query and populate the message list |
| Delete | Remove this query from the query collection |

# 5 GUI Output Windows

## 5.1 Object Type Definition File

This window displays the Object Type Definition File.

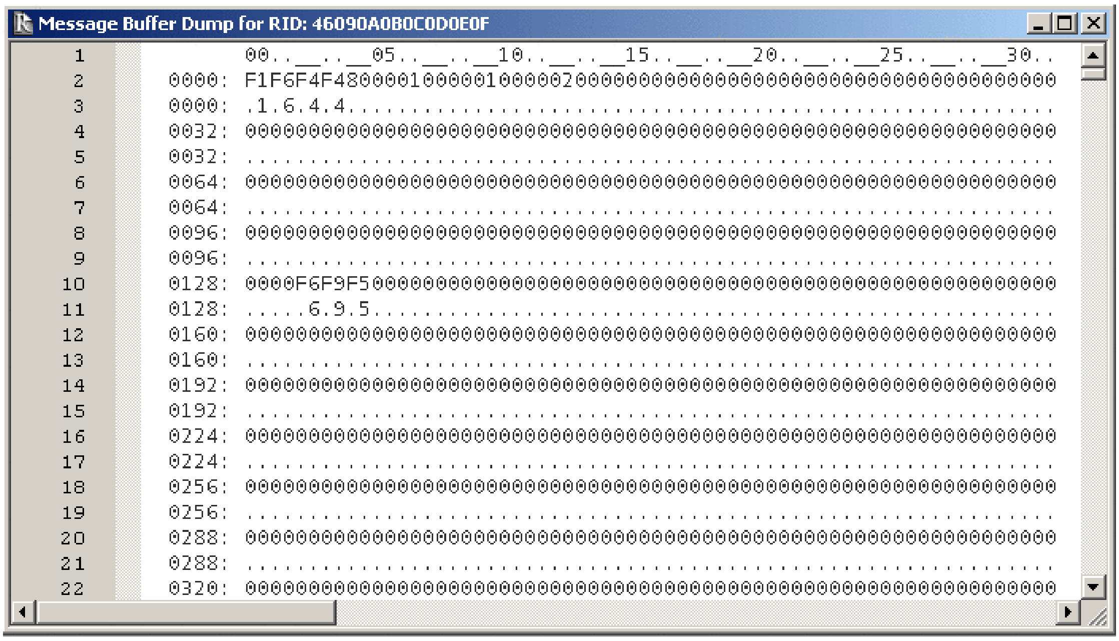

C:\CodeMagus\CodeMagus\MCIFormats\testdata\objtypes\mci_ipm_types.objtypes

```
path "c:/CodeMagus/CodeMagus/MCIFormats/testdata/copybooks/%s.cpy";
path  book "c:/CodeMagus/CodeMagus/MCIFormats/testdata/copybooks/%s.cpy";
set EMVXXLIB="C:/CodeMagus/CodeMagus/EMVFormats/testdata/copybooks/%s.cpy";
-- GENERATED NIPS OBJECT TYPES
options ebcdic, omit_fillers;


----------------------------------------------------------------
--
--  IPM Formats - Full Draft
--
--  Copyright (c) 2005 Code Magus Limited. All rights reserved.
--  Contact: Stephen Donaldson [stephen@codemagus.com].
--
--  Meta Data and descriptions derived from:
--  IPM Clearing Formats, April 2005
--  © 2005 MasterCard International Incorporated
--
--  GCMS Release 08.2 Document 2 April 2008
--
--  GCMS Release 08.2 Document 2 April 2008
--  Revised 2 July 2008
--
--  IPM Clearing Formats 24 April 2009
--
--  Authorization and Clearing Release 09.2 Document
--  25 March 2009 Updated 24 June 2009
--
----------------------------------------------------------------
--
--  $Author: cc080244 $
```

## 5.2 Message Buffer Dump

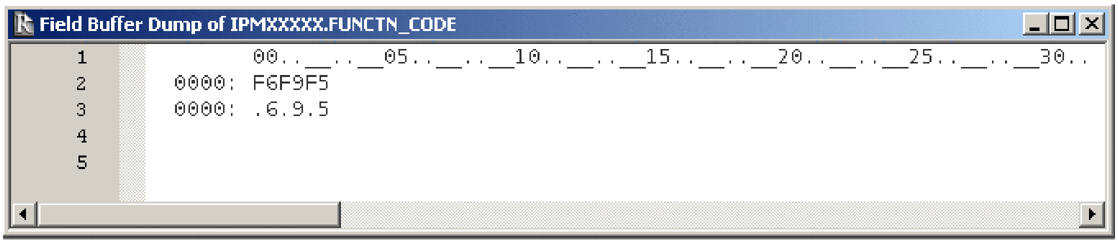
This window displays a the buffer of a given message.



```
       00..__..__05..__..__10..__..__15..__..__20..__..__25..__..__30..
0000:  F1F6F4F480000100000100000200000000000000000000000000000000000000
0000:  .1.6.4.4........................................................
0032:  0000000000000000000000000000000000000000000000000000000000000000
0032:  ................................................................
0064:  0000000000000000000000000000000000000000000000000000000000000000
0064:  ................................................................
0096:  0000000000000000000000000000000000000000000000000000000000000000
0096:  ................................................................
0128:  0000F6F9F5000000000000000000000000000000000000000000000000000000
0128:  .....6.9.5......................................................
0160:  0000000000000000000000000000000000000000000000000000000000000000
0160:  ................................................................
0192:  0000000000000000000000000000000000000000000000000000000000000000
0192:  ................................................................
0224:  0000000000000000000000000000000000000000000000000000000000000000
0224:  ................................................................
0256:  0000000000000000000000000000000000000000000000000000000000000000
0256:  ................................................................
0288:  0000000000000000000000000000000000000000000000000000000000000000
0288:  ................................................................
0320:  0000000000000000000000000000000000000000000000000000000000000000
```

## 5.3 Field Buffer Dump

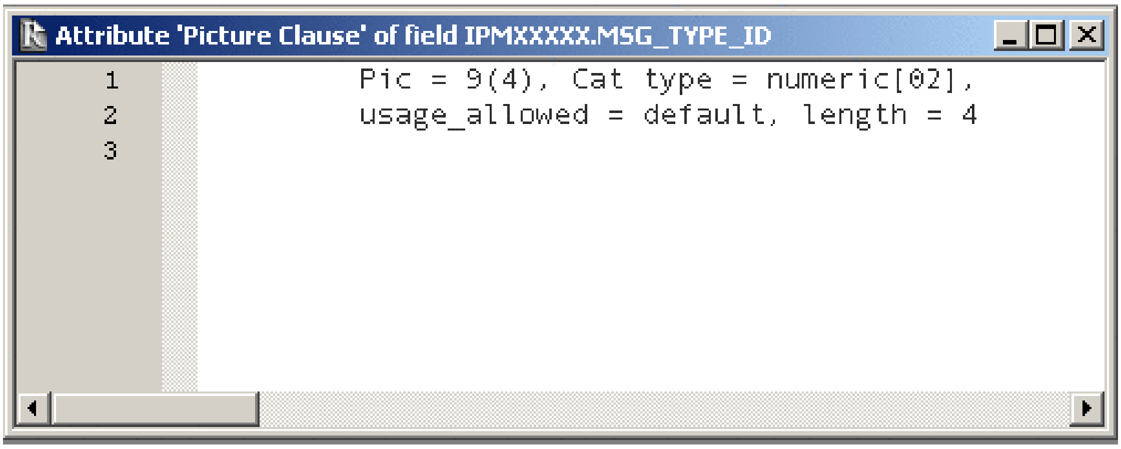
This window displays the buffer of the a given field of a message.



```
       00..__..__05..__..__10..__..__15..__..__20..__..__25..__..__30..
0000:  F6F9F5
0000:  .6.9.5
```

## 5.4 Field Picture Clause

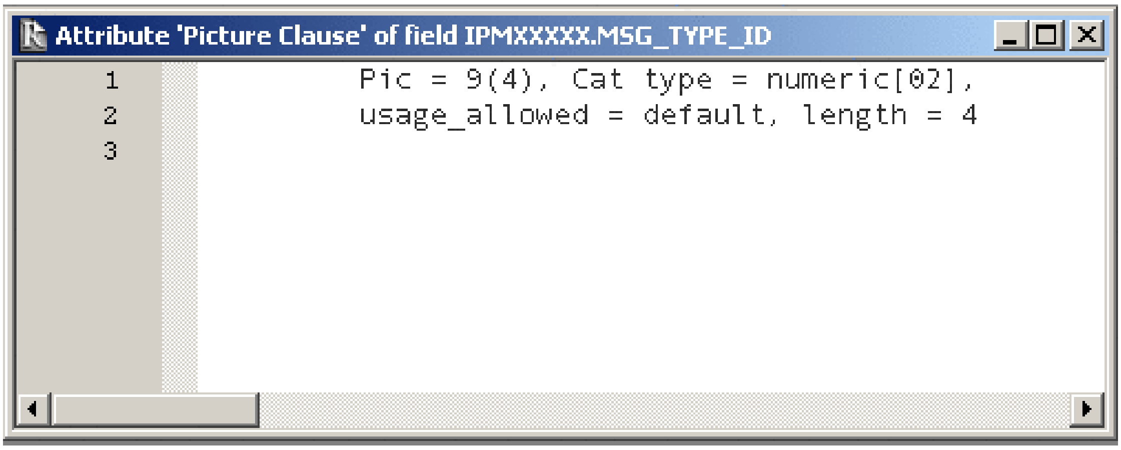
This window displays the picture clause of the given field.



```
Pic = 9(4), Cat type = numeric[02],
usage_allowed = default, length = 4
```

## 5.5 Field DESCRIPTION clause

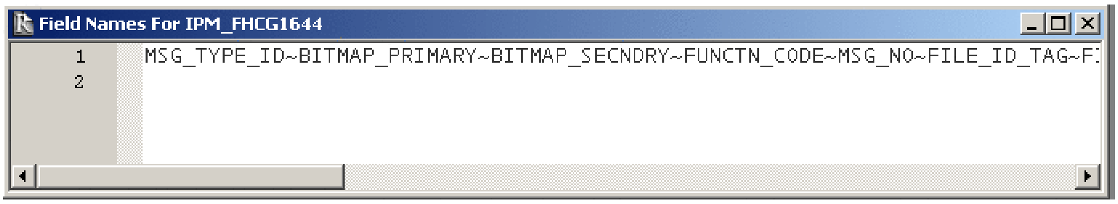
This window displays the DESCRIPTION of the given field according to the copybook.



```
Pic = 9(4), Cat type = numeric[02],
usage_allowed = default, length = 4
```

## 5.6 Exported Field Names

This window displays the names of all the fields within a given message.



```
MSG_TYPE_ID~BITMAP_PRIMARY~BITMAP_SECNDRY~FUNCTN_CODE~MSG_NO~FILE_ID_TAG~F
```

# References

[1] **objtypes**: Configuring for Object Recognition, Generation and Manipulation. CML Document CML00018-01, Code Magus Limited, July 2008. http://www.codemagus.com/documents/objtpuref.pdf .