Informatica B2B Data Transformation (Version 9.1.0)

COBOL Processing Guide
This Software is protected by U.S. Patent Numbers 5,794,246; 6,014,670; 6,016,501; 6,029,178; 6,032,158; 6,035,307; 6,044,374; 6,092,086; 6,208,990; 6,339,775; 6,640,226; 6,789,096; 6,820,077; 6,823,373; 6,850,947; 6,895,471; 7,117,215; 7,162,643; 7,254,590; 7,281,001; 7,421,458; 7,496,588; 7,523,121; 7,584,422; 7,720,842; 7,721,270; and 7,774,791, international Patents and other Patents Pending.

DISCLAIMER: Informatica Corporation provides this documentation "as is" without warranty of any kind, either express or implied, including, but not limited to, the implied warranties of non-infringement, merchantability, or use for a particular purpose. Informatica Corporation does not warrant that this software or documentation is error free. The information provided in this software or documentation may include technical inaccuracies or typographical errors. The information in this software and documentation is subject to change at any time without notice.

NOTICES

This Informatica product (the "Software") includes certain drivers (the "DataDirect Drivers") from DataDirect Technologies, an operating company of Progress Software Corporation ("DataDirect") which are subject to the following terms and conditions:

1. THE DATADIRECT DRIVERS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT.

2. IN NO EVENT WILL DATADIRECT OR ITS THIRD PARTY SUPPLIERS BE LIABLE TO THE END-USER CUSTOMER FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL OR OTHER DAMAGES ARISING OUT OF THE USE OF THE ODBC DRIVERS, WHETHER OR NOT INFORMED OF THE POSSIBILITIES OF DAMAGES IN ADVANCE. THESE LIMITATIONS APPLY TO ALL CAUSES OF ACTION, INCLUDING, WITHOUT LIMITATION, BREACH OF CONTRACT, BREACH OF WARRANTY, NEGLIGENCE, STRICT LIABILITY, MISREPRESENTATION AND OTHER TORTS.

Part Number: DT-CBL-91000-0001
# Table of Contents

## Preface ........................................................................... ii

Informatica Resources. .......................................................... ii
  Informatica Customer Portal. .............................................. ii
  Informatica Documentation. .............................................. ii
  Informatica Web Site. ....................................................... ii
  Informatica How-To Library. ............................................... ii
  Informatica Knowledge Base. ........................................... iii
  Informatica Multimedia Knowledge Base. ........................ iii
  Informatica Global Customer Support. ............................... iii

## Chapter 1: Transforming COBOL Data.............................................. 1

COBOL Support Overview. .................................................... 1
  Importing a COBOL Data Definition. ............................... 1
    Supported COBOL Features. ........................................... 2
    Example of COBOL Import. .......................................... 2

## Chapter 2: Working with COBOL Projects. ...................................... 5

Test Procedures. .......................................................... 5
  Testing a COBOL Parser. ................................................. 5
  Testing a COBOL Serializer. ........................................... 6

Editing COBOL Projects. ..................................................... 7
  Integrating with Other Transformations. ......................... 7
  Deploying COBOL Projects as Services. ....................... 8

## Index. ................................................................ 9
Preface

The *Data Transformation COBOL Processing Guide* is written for software developers who want to transform data to or from COBOL data structures. The book explains how to generate and run transformations for structures that conform to COBOL data definitions.

Informatica Resources

Informatica Customer Portal

As an Informatica customer, you can access the Informatica Customer Portal site at [http://mysupport.informatica.com](http://mysupport.informatica.com). The site contains product information, user group information, newsletters, access to the Informatica customer support case management system (ATLAS), the Informatica How-To Library, the Informatica Knowledge Base, the Informatica Multimedia Knowledge Base, Informatica Product Documentation, and access to the Informatica user community.

Informatica Documentation

The Informatica Documentation team takes every effort to create accurate, usable documentation. If you have questions, comments, or ideas about this documentation, contact the Informatica Documentation team through email at infa_documentation@informatica.com. We will use your feedback to improve our documentation. Let us know if we can contact you regarding your comments.

The Documentation team updates documentation as needed. To get the latest documentation for your product, navigate to Product Documentation from [http://mysupport.informatica.com](http://mysupport.informatica.com).

Informatica Web Site

You can access the Informatica corporate web site at [http://www.informatica.com](http://www.informatica.com). The site contains information about Informatica, its background, upcoming events, and sales offices. You will also find product and partner information. The services area of the site includes important information about technical support, training and education, and implementation services.

Informatica How-To Library

As an Informatica customer, you can access the Informatica How-To Library at [http://mysupport.informatica.com](http://mysupport.informatica.com). The How-To Library is a collection of resources to help you learn more about Informatica products and features. It includes articles and interactive demonstrations that provide solutions to common problems, compare features and behaviors, and guide you through performing specific real-world tasks.
Informatica Knowledge Base

As an Informatica customer, you can access the Informatica Knowledge Base at http://mysupport.informatica.com. Use the Knowledge Base to search for documented solutions to known technical issues about Informatica products. You can also find answers to frequently asked questions, technical white papers, and technical tips. If you have questions, comments, or ideas about the Knowledge Base, contact the Informatica Knowledge Base team through email at KB_Feedback@informatica.com.

Informatica Multimedia Knowledge Base

As an Informatica customer, you can access the Informatica Multimedia Knowledge Base at http://mysupport.informatica.com. The Multimedia Knowledge Base is a collection of instructional multimedia files that help you learn about common concepts and guide you through performing specific tasks. If you have questions, comments, or ideas about the Multimedia Knowledge Base, contact the Informatica Knowledge Base team through email at KB_Feedback@informatica.com.

Informatica Global Customer Support

You can contact a Customer Support Center by telephone or through the Online Support. Online Support requires a user name and password. You can request a user name and password at http://mysupport.informatica.com.

Use the following telephone numbers to contact Informatica Global Customer Support:

<table>
<thead>
<tr>
<th>North America / South America</th>
<th>Europe / Middle East / Africa</th>
<th>Asia / Australia</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Toll Free</strong></td>
<td><strong>Toll Free</strong></td>
<td><strong>Toll Free</strong></td>
</tr>
<tr>
<td>Brazil: 0800 891 0202</td>
<td>France: 00800 4632 4357</td>
<td>Australia: 1 800 151 830</td>
</tr>
<tr>
<td>Mexico: 001 888 209 8853</td>
<td>Germany: 00800 4632 4357</td>
<td>New Zealand: 1 800 151 830</td>
</tr>
<tr>
<td>North America: +1 877 463 2435</td>
<td>Israel: 00800 4632 4357</td>
<td>Singapore: 001 800 4632 4357</td>
</tr>
<tr>
<td><strong>Standard Rate</strong></td>
<td>Italy: 800 915 985</td>
<td><strong>Standard Rate</strong></td>
</tr>
<tr>
<td>North America: +1 650 653 6332</td>
<td>Netherlands: 00800 4632 4357</td>
<td>India: +91 80 4112 5738</td>
</tr>
<tr>
<td></td>
<td>Portugal: 800 208 360</td>
<td><strong>Standard Rate</strong></td>
</tr>
<tr>
<td></td>
<td>Spain: 900 813 166</td>
<td>United Kingdom: 0800 4632 4357 or 0800 023 4632</td>
</tr>
<tr>
<td></td>
<td>Switzerland: 00800 4632 4357 or 0800 4632 200</td>
<td><strong>Standard Rate</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Australia: 1 800 151 830</td>
</tr>
<tr>
<td><strong>Standard Rate</strong></td>
<td></td>
<td>New Zealand: 1 800 151 830</td>
</tr>
<tr>
<td>France: 0805 804632</td>
<td></td>
<td>Singapore: 001 800 4632 4357</td>
</tr>
<tr>
<td>Germany: 01805 702702</td>
<td></td>
<td><strong>Standard Rate</strong></td>
</tr>
<tr>
<td>Netherlands: 030 6022 797</td>
<td></td>
<td>United Kingdom: 0800 4632 4357 or 0800 023 4632</td>
</tr>
</tbody>
</table>

Preface iii
CHAPTER 1

Transforming COBOL Data

This chapter includes the following topics:

♦ COBOL Support Overview, 1
♦ Importing a COBOL Data Definition, 1

COBOL Support Overview

Data Transformation offers built-in support for transforming COBOL data to and from an XML representation. Via the XML representation, you can transform the COBOL data to and from any other data format. This capability enables you to integrate legacy COBOL applications easily with applications that use other formats to represent the data.

In Data Transformation Studio, you can import a COBOL data definition from a copybook. This generates a Data Transformation project containing:

♦ An XSD schema file that defines an XML representation of the COBOL data structure.
♦ A parser that transforms input data conforming to the COBOL data definition to the XML representation.
♦ A serializer that transforms data in the XML representation back to the native COBOL representation.

You can deploy the parser and serializer as Data Transformation services that process COBOL data, or you can incorporate them in other projects and services that integrate with any desired data formats.

Importing a COBOL Data Definition

1. Click File > New > Project.
   The New Project wizard appears.
   Note: You must use File > New > Project for this purpose. Do not use File > Import, which imports existing Data Transformation or Eclipse projects.
2. Under the Data Transformation node, select Import Project.
3. On the next wizard page, enter a name for the project, and select the storage location.
   The default location is the Data Transformation Studio workspace folder.
4. On the next page, browse to the COBOL data definition file, in other words, to a COBOL copybook.
5. Click Finish.

Data Transformation Studio generates a project containing an XSD schema file, a parser, and a serializer for the COBOL data.

Supported COBOL Features

The COBOL import supports data definitions of any complexity. For example, the data definitions can use the packed decimal (COMP-3), binary (COMP-1, COMP-2, or COMP-4), and logical decimal point (999999) data types. They can contain features such as REDEFINES, OCCURS, and OCCURS DEPENDING ON clauses.

The data definition must comply with the following specifications:

- No more than 72 characters per line, and no text beyond column 72
- The first line must be a remark, with a * in column 7, or it must start with a level number
- The first level number must be in column 1 or 8.

The following COBOL features are not currently supported:

- The special level numbers 66, 77, and 88
- USAGE clauses at a group level
- INDEXED BY clauses
- POINTER and PROCEDURE-POINTER

Example of COBOL Import

In this example, we imported the following COBOL data definition file:

```
000100* ---------------------------------------------------------------
001300* INPUT SAMPLE
001600* ---------------------------------------------------------------
001700   01 SAMI001.
001800*
003000   05 SAI-KEY-LNAME   PIC X(10).
003100*   ------------------------ KEY-LNAME
003800   05 SAI-FNAME       PIC X(10).
003810*   ------------------------ FNAME
006000   05 SAI-SALARY      PIC X(8).
006100*   ------------------------ SALARY
006600   05 SAI-ID          PIC X(17).
006700*   ------------------------ ID
007800   05 SAI-BDATE       PIC 9(8).
007810*   ------------------------ BIRTHDATE (JJJJMMDD)
```
We followed the procedure for importing a COBOL data definition, assigning a project name of COBOL_DEMO. The procedure automatically generated a schema, parser, and serializer, which are illustrated below.

Figure 1. Schema Auto-Generated by the COBOL Import Procedure

```
<xml version="1.0" encoding="windows-1252">
  <xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="SAM001">
      <xs:complexType>
        <xs:sequence>
          <xs:element default="" name="SAI-KEY-LNAME" type="xs:string"/>
          <xs:element default="" name="SAI-FNAME" type="xs:string"/>
          <xs:element default="" name="SAI-SALARY" type="xs:string"/>
          <xs:element default="" name="SAI-DATE" type="xs:string"/>
        </xs:sequence>
        <xs:complexType name="SAM001_variable_type">
          <xs:element name="skip"/>
          <xs:element name="numericDefault"/>
          <xs:element name="stringDefault"/>
        </xs:complexType>
      </xs:complexType>
    </xs:element>
  </xs:schema>
```

Figure 2. Auto-Generated Parser that Transforms COBOL Data to XML

```
COBOL демо_parsers-schema.xsd

? FIXED-FORMAT PARSER = Parser >>
    example_source = Text()
    format = BinaryFormat()
    remark = "created from COBOL copybook"
    contains
      RepeatingGroup >>
      separator_position = between
      separator =
      target = Locator(/SAM001)
      contains
        Group >>
        target = Locator(/SAM001)
        contains
          Content<allow_empty_values>(OffsetSearch(10), /SAM001/*/SAI-KEY-LNAME)
          Content<allow_empty_values>(OffsetSearch(10), /SAM001/*/SAI-FNAME)
          Content<allow_empty_values>(OffsetSearch(10), /SAM001/*/SAI-SALARY)
          Content<allow_empty_values>(OffsetSearch(8), /SAM001/*/SAI-ID)
          Content<allow_empty_values>(OffsetSearch(8), /SAM001/*/SAI-DATE)
          ...
      ...

  SAM001_Variables = Variable >>
  val_type = SAM001_variable_type
```
Figure 3. Auto-Generated Serializer that Transforms XML to COBOL

```cobol
COBOL_DEMO_serializer.tpp

SAM001_FixedFormatSerializer = Serializer >>
   root_tag = "ContentMaster"
   remark = "created from Cobol copybook"
   contains
   SetValue("", SSAM001_Variables*/s/stringsDefault)
   SetValue("0", SSAM001_Variables*/s/numericDefault)
   RepeatingGroupSerializer >>
      separator_position = before
      source = [Locator/(SAM001)]
      contains
      GroupSerializer >>
         source = [Locator/(SAM001)]
         contains
         ContentSerializer=allow_empty_values:/.SAM001*/s/SALARY, .[Resize()]
         ContentSerializer=allow_empty_values:/.SAM001*/s/SALDATE, .[FormatNumber(), Resize()]
         ...
CHAPTER 2

Working with COBOL Projects

This chapter includes the following topics:

- Test Procedures, 5
- Editing COBOL Projects, 7

Test Procedures

This section briefly describes procedures for testing and working with COBOL transformations. For more information, see the Data Transformation Getting Started Guide and the Data Transformation Studio User Guide.

Testing a COBOL Parser

You can test the COBOL parser by transforming sample COBOL data to XML and verifying the output. To do this, you need an input file that contains sample COBOL data. The data structure must conform to the data definition that you imported.

In the following example, we use a COBOL data file corresponding to the data definition that we imported in the preceding chapter. For more information about the data structure, see “Example of COBOL Import” on page 2.

Figure 4. Sample COBOL Input File

To test a COBOL parser:

1. In the Data Transformation Explorer view, double-click the TGP script file of the parser. The parser appears in an IntelliScript editor.
2. Right-click the parser name, which is usually in the first line of the IntelliScript, and click Set as Startup Component.
3. Expand the IntelliScript tree, and edit the example_source property of the parser. Change its value from Text to LocalFile.
The auto-generated COBOL parser is configured in a way that does not require an example source document. When you finish testing, you can remove the example source if you wish. Leaving the example source has no effect on the transformation at runtime.

4. Nested within the `LocalFile` component, assign the `file_name` property. Browse to the input file, which contains the sample COBOL data.

![Figure 5. Assigning the Input File to the example_source Property of the Parser](image)

5. The document appears in the example pane of the IntelliScript editor. If the example pane is not visible, click `IntelliScript > Both`. If the document is not displayed, right-click the parser name in the IntelliScript, and click `Open Example Source`.

6. Optionally, click `IntelliScript > Mark Example` to highlight the data that the parser finds in the document. If the data correctly conforms to the data definition that you imported, the parser should find all the data in the document. Therefore, the `Mark Example` command should highlight the entire document. For more information about the color coding, see the *Data Transformation Studio Editing Guide*.

7. Click `Run > Run` to activate the parser.

8. Examine the execution log in the `Events` view. Confirm that the parser ran without error. For more information about interpreting the log, see “Running and Testing Projects” in the *Data Transformation Studio User Guide*.

9. To view the parser output, double-click the `Results\output.xml` file in the Data Transformation Explorer.

![Figure 6. XML Output of the Parser](image)

**Testing a COBOL Serializer**

After you test a COBOL parser, you can run the COBOL serializer on the output of the parser.

To test a COBOL serializer:

1. In the Data Transformation Explorer, double-click the TGP script file of the serializer. The serializer appears in an IntelliScript editor.

2. Right-click the serializer name, and click `Set as Startup Component`.

3. Click `Run > Run` to activate the serializer. At the prompt, browse to the `Results\output.xml` file, which you generated when you ran the parser.

4. Examine the execution log in the `Events` view. Confirm that the serializer ran without error.
5. To view the serializer output, double-click the Results\output.xml file in the Data Transformation Explorer. The display should be the same as the original input, on which you ran the parser. For example, if you run on the sample parser output illustrated above, the serializer output is:

Figure 7. Output of the Serializer, Identical to the Original COBOL Data

| JONES  | BILL  | 1452 | ZY9335787819560517 |

Editing COBOL Projects

You can edit an auto-generated COBOL project in Data Transformation Studio. If you do this, document your editing. The documentation might be essential if you later revise the COBOL data definition, re-import it to a new project, and need to reproduce your editing.

Integrating with Other Transformations

You can integrate auto-generated COBOL transformations with other transformation components. The following example illustrates a typical procedure. The example illustrates how to integrate a legacy COBOL application with a database system.

Storing COBOL Data in the Database

To store COBOL data in the database:

1. Use the auto-generated COBOL parser to convert the native COBOL data to the XML representation.
2. Use a mapper to transform the XML representation to the corresponding database fields.
3. Within the mapper, use an ODBCAction component to store the result in the database.

Retrieving COBOL Data from the Database

To retrieve COBOL data from the database:

1. Use a second mapper, containing an ODBCAction component, to retrieve the data from the database.
2. The mapper transforms the database fields to the XML representation of the COBOL data.
3. Use the auto-generated COBOL serializer to convert the XML to the native COBOL representation.

Implementation

You can chain the auto-generated COBOL transformations together with other transformation components in a single Data Transformation project. For example, you can use the RunParser, RunMapper, and RunSerializer actions to activate the various components described in the above example. For more information about these actions, see the Data Transformation Studio User Guide.

Alternatively, you can deploy the components in separate Data Transformation services, which you activate in sequence. For more information about activating the services, see the Data Transformation Engine Developer Guide.
Deploying COBOL Projects as Services

When you finish editing and testing, you can deploy a COBOL project as a service that runs in Data Transformation Engine. For more information, see “Deploying Data Transformation Services” in the *Data Transformation Studio User Guide*. 
INDEX

C
COBOL
  data definition file 2
  deploying projects as services 8
  importing data definition 1
  integration applications 7
  supported features 2
  testing parser 5
  testing serializer 6

D
data definition file
  COBOL 2

P
parser
  for COBOL data 1

S
schema
  for COBOL data 1
  serializer
  for COBOL data 1