WebSphere Data Interchange v3.2 Data Transformation XML Document Processing Implementation Guide

Data Transformation XML Document Processing CSD Overview

WebSphere Data Interchange(WDI) development has created this CSD in response to the immediate demand to process extremely large XML documents that contain a repeating compound element which would produce a single output message with Data Transformation processing. WDI development has also received several requests to process a single XML document source and create multiple target documents. This document is not a formal IBM publication. It is a technical document written by WDI development to help customers implement Data Transformation enhancement to XML document processing using this CSD.

Data Transformation processing:

Translating data from any EDI, XML, or data format to any other EDI, XML, or data format using a data transformation map. Any-to-any translation is a WebSphere Data Interchange feature that allows you to translate data from any supported source document type to any supported target document type. Supported document types include data formats, EDI standards, and XML data. The Utility command PERFORM TRANSFORM is used for any-to-any translation. The XML Document Processing CSD addresses splitting a single XML document into smaller XML documents for transformation with data transformation processing. **NOTE:** The new XML document processing functionality is described in this document.

XML Document processing:

The default XML document processing using data transformation processing enables mapping of a single XML document to a single target document. Many XML source documents resemble EDI data. A single XML document contains header type information, multiple messages, and trailer type information. It is desirable for example to map the XML source document to multiple EDI target documents. To achieve this, a double transformation process is needed to transform the XML document to an intermediate document, for example data format, with a second transformation process to map the intermediate document to the EDI document.

The XML document processing has been enhanced to remove the double transformation process. The single XML document will be split based on a defined XML compound element and reconstructed before the document enters the data transformation message flow. The following enhancements have been made to enable the document split:

WDI Client DTD and Schema definitions:

An enhancement to WDI Client XML DTD and Schema definitions to contain an "Overview" tab has been added to display a visual layout of the DTD or schema. You can right click on elements and use functions on the popup menu to set certain fields in the General tab page. A right click on a simple element will display a popup menu to set elements that contain the sender and receiver ID and qualifier **paths**. A right click on a compound elements will display a popup menu to set XML document split **element Ids**. **NOTE:** The split element identification is not a path, it is an element ID.

There are three elements that may be defined to split the XML document. Element identifying the header area in the XML document, element identifying the individual messages (split area), and element identifying the trailer area in the XML document. These definitions are used to split and reconstruct the XML documents before they are placed in the data transformation message flow. The element identifying the individual messages is **required** to split the source XML document. If the element identification is not defined, the source XML document will not be split. If the header area is not defined, the beginning of the XML document up to the element identifying the message will be use to construct a header area for the split document. If the trailer area is not defined, the end root element will be used as the trailer area for the split document. If the trailer area is defined and is actually a terminating element in the XML source input, then the right click to define this using WDI Client is to right click the compound element (that begins the trailer element), define the element as the trailer element, and check the box on the general tab "Element Terminator Indicates Start of Trailer Section".

WDI Client data transformation mapping:

A new XML source document property "MsgSplitCnt" is available and can be used to identify the number of documents split within each header/message/trailer split. The MsgSplitCnt property is set to zero until the last split message is processed. MsgSplitCnt property is reset with each new trailer/header identification. The MsgSplitCnt property can be used, for example, to MapChain() to a summary mapping by counting the number of source messages processed using a global variable within the source document mapping and comparing this to the MsgSplitCnt property.

The "InputMsgCnt" source document property identifies the number of input messages processed and is available in data transformation mapping. Source document property "LastMsg" identifies that WDI is processing the last message in the input file.

WDI server processing:

During processing, the XML source input message is read and stored in a buffer. The "root" element is identified (from the input message) and the WDI DTD or Schema definition is retrieved to determine if the source XML document has been defined as a split document. This is done with all XML input source messages and can be removed by specifying the PERFORM keyword XMLSPLIT(N).

During the XML document definition retrieval, if there are multiple DTD or Schema definitions defining the same root element, it may be necessary to use PERFORM keywords DICTIONARY or DOCUMENT to identify the specific DTD or Schema being used for processing.

If the XML document is to be split, the header area is identified and stored in a header area buffer, the trailer area is identified and stored in a trailer area buffer. The header area is the beginning of the XML document and the Header element identification up to the first Message element identification. The trailer area is defined as the Trailer element identification up to the next header element identification. The area between the first message element identification up to the trailer element identification are written out to the "XMLWORK" file. During reconstruction, each split message is read from the XMLWORK file. The header/message/trailer are constructed and sent through the data transformation message flow as individual documents. NOTE: The XMLWORK file must be allocated.

Example 1:

The sample XML document below contains information about three companies. The expanded elements (those proceeded by a dash) show the first company element contains information about the company and information about four employees of the company. If each employee element needs to be translated into its own document, then the "employee" element would be listed as the Message Element on the General tab page of the Schema Editor. The Header Element would be the "company" element. The Trailer Element would be the " employee-list" element with the Element Terminator Indicates Start of Trailer Section check box set.

```
<? Xml version="1.0" encoding="UTF-8" ?>
- <root-element xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                                xsi:noNamespaceSchemaLocation="ThisDoc.xsd">
    - <company>                                    <=== Header Element
        - <company-details>
              + <campany-name>
              + <campany-address>
          </company-details>
         - <employee-list>
              + <employee>                         <==== Message Element (Split here)
              + <employee>
              + <employee>
              + <employee>
           </employee-list>                        <=== Trailer Element (Element Terminator)
      </company>                                   (Note: end of header area)
   + <company>
    + <company>
</root-element>
```

In the result would be one document like the following for each "employee" contained in the source document.

```
<? Xml version="1.0" encoding="UTF-8" ?>
- <root-element xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                    xsi:noNamespaceSchemaLocation="ThisDoc.xsd">
    - <company>
        - <company-details>
              <campany-name>
            + <campany-address>
          </company-details>
         - <employee-list>
              + <employee>
          </employee-list>
```

```
        </company>
</root-element>
```

Example 2:

The sample XML document below contains information about three companies. The expanded elements (those proceeded by a dash) show the first company element contains information about the company and information about four employees of the company. If each employee element needs to be translated into its own document, then the "employee" element would be listed as the Message Element on the General tab page of the Schema Editor. The Header Element would be the "company" element. The Trailer Element would be the " employee-list" element with the Element Terminator Indicates Start of Trailer Section check box set.

```
<? Xml version="1.0" encoding="UTF-8" ?>
- <root-element xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                              xsi:noNamespaceSchemaLocation="ThisDoc.xsd">
    - <company>                                  <=== Header Element
        - <company-details>
              + <campany-name>
              + <campany-address>
          </company-details>
      </company>                                 (Note: end of header area)
    - <employee-list>
          + <employee>                           <==== Message Element (Split here)
          + <employee>
          + <employee>
          + <employee>
       </employee-list>                          <=== Trailer Element (Element Terminator)
   + <company>
    + <company>
</root-element>
```

In the result would be one document like the following for each "employee" contained in the source document.

```
<? Xml version="1.0" encoding="UTF-8" ?>
- <root-element xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                    xsi:noNamespaceSchemaLocation="ThisDoc.xsd">
    - <company>
        - <company-details>
              <campany-name>
            + <campany-address>
          </company-details>
      </company>
```

```
    - <employee-list>
        + <employee>
    </employee-list>
</root-element>
```

Example 3:

The sample XML document below contains information about three companies. The expanded elements (those proceeded by a dash) show the first company element contains information about the company and information about four employees of the company. If each employee element needs to be translated into its own document, then the "employee" element would be listed as the Message Element on the General tab page of the Schema Editor. The Header Element would be the "company" element. There is no real trailer element to be defined and the trailer element identification will default to the end root element. **NOTE:** With no trailer area, only the first header element occurrence can be identified. All employee elements will be split under the first company occurrence. MsgSplitCnt property will be a total and set with the last employee split processed.

```
<? Xml version="1.0" encoding="UTF-8" ?>
- <root-element xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                        xsi:noNamespaceSchemaLocation="ThisDoc.xsd">
    - <company>                              <=== Header Element
        - <company-details>
            + <campany-name>
            + <campany-address>
          </company-details>
      </company>                             (Note: end of header area)
    + <employee>                         <==== Message Element (Split here)
    + <employee>
    + <employee>
    + <employee>
    + <company>
    + <company>
</root-element>
```

In the result would be one document like the following for each "employee" contained in the source document.

```
<? Xml version="1.0" encoding="UTF-8" ?>
- <root-element xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                    xsi:noNamespaceSchemaLocation="ThisDoc.xsd">
    - <company>
        - <company-details>
            <campany-name>
          + <campany-address>
```

```
        </company-details>
      </company>
    + <employee>
</root-element>
```

WDI Utility Interface

New keywords for PERFORM TRANSFORM

The following keywords have been added to the PERFORM TRANSFORM command:

- XMLSPLIT(Y/N) - default(Y)
  For data transformation and XML source message processing. Overrides the XML DTD or Schema definition retrieval and processes the XML input message using default processing.

- FILTERMSGS(UTxxxx, Upxxxx, etc).
  For data transformation processing. Filters messages in the list, if severity is less than 8. Maximum length is 80. Up to 11 individual messages may be filtered. See also IGNOREWARN and IGNOREINFO keywords.

- IGNOREINFO(Y/N) - default N
  For data transformation processing. Filters all informational messages.

- IGNOREWARN(Y/N) - default N
  For data transformation processing. Filters all warning messages.

WDI Client Interface

New Source Document Properties available during mapping execution

MsgSplitCnt
Identifies the number of XML documents split within each header/message/trailer split. The MsgSplitCnt property is set to zero until the last split message is processed. MsgSplitCnt property is reset with each new trailer/header identification.

Example:

```
TotalNumEmployees = GetProperty ("MsgSplitCnt")
EmployeeCnt = EmployeeCnt + 1
If (EmployeeCnt = TotalNumEmployees)
   MapChain ("SUMMARYMAP")
```

InputMsgCnt
Identifies the number of input messages processed. This is a running total.

LastMsg
A value of ‘Y’ indicates the current message is the last message being processed in the message flow.