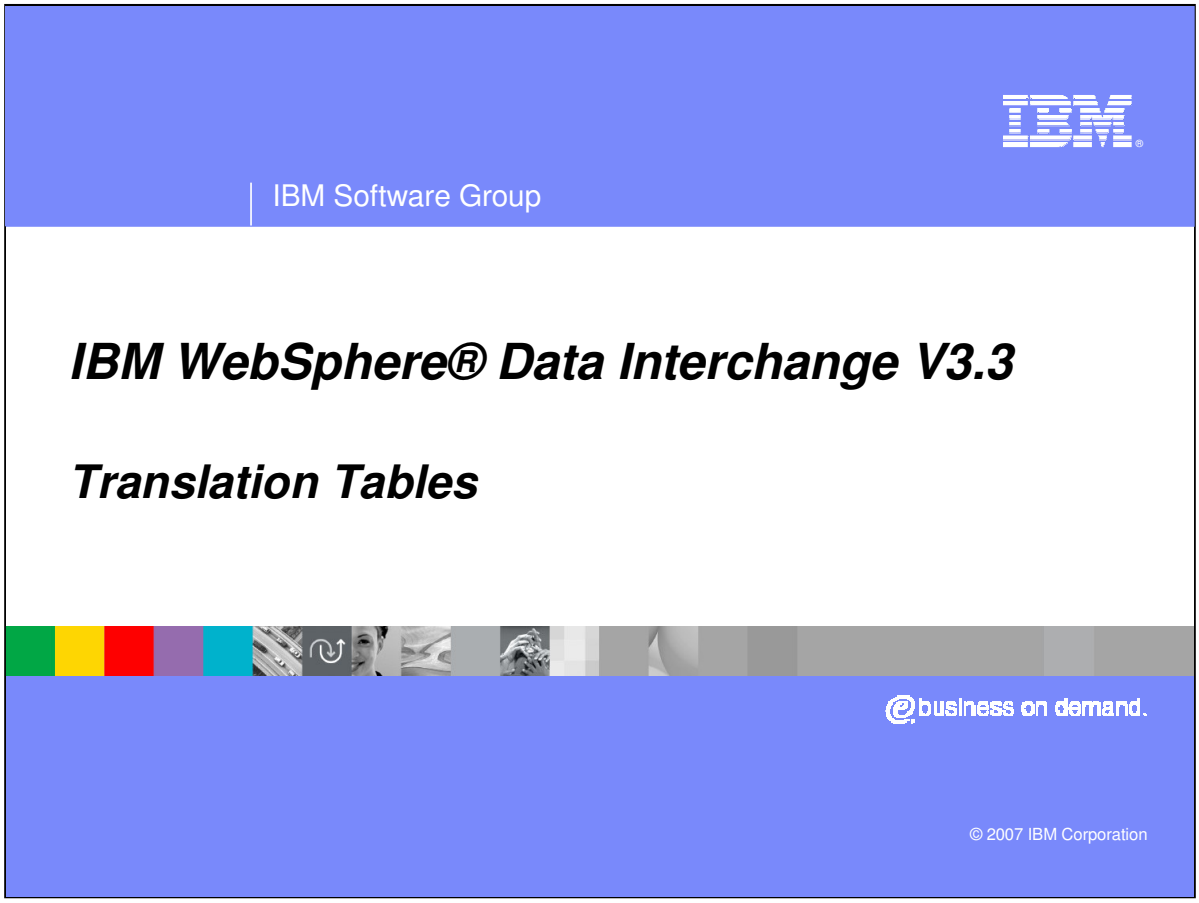


This presentation will review Translation Tables.

Agenda

- Purpose of Translation Tables
- Accessing Translation Tables
- Mapping example
- Restrictions
- Internals
- Summary

Topics to be covered in this presentation are:

What is the purpose of WebSphere Data Interchange (WDI) Translation Tables?

How are Translation Tables accessed?

What is an example of using a Translation Table in WDI Mapping?

Are there any Restrictions?

What are some Best practices?

Purpose of Translation Tables

- Translation tables
  - Used during translation
  - Referenced during mapping
  - Substitutes one value for another
  - Allows handling of differences between your data and a trading partners' data
- For example, convert part numbers a trading partner uses to those you use.

When WebSphere Data Interchange translates data from your source document to a target document, it can also substitute one value for another. WebSphere Data Interchange substitutes values through translation tables. Use translation tables to handle: v Differences between your data and your trading partners' data. For example, say your trading partner uses its own numbers for parts you sell. You can set up a translation table to convert the part numbers your trading partner uses to those you use.

Conflicts between data in your source document and data in your target document are different. For example, say the source document uses an abbreviated unit of measurement that is not the same as in the target document. You can create a translation table to substitute a different code in your target document than is in your source document.

Types of Translation Tables

- Types of Translation tables
  - Forward Translation tables
    - Converts local value in the table entry to the target value
    - For example, during Send processing or source based maps
  - Reverse Translation tables
    - Converts the target value to the local value
    - For example, during Receive processing or target based maps

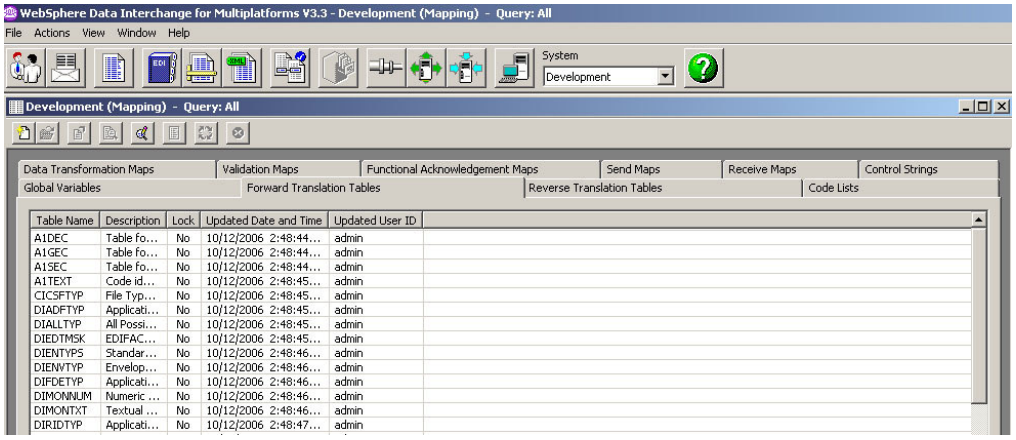
In Data Transformation Maps, Validation Maps, and Functional Acknowledgement Maps, lookup of a value is normally done against the source values in a Forward Translation Table. When a matching source value is found, the corresponding target value is substituted for the original value. There is an option that allows you to perform lookup using the target value. This allows you to use the same table to translate a value to a trading partner's equivalent value and also to translate a trading partner's value to your equivalent value. Using the target value for lookup is useful only when target values are unique. When the option to perform lookup using the target value is used, the lookup attempts to locate a target value in the translation table that matches the value in the simple element or variable. When found, the corresponding source value is substituted for the original value.

Receive Maps always perform lookup on a Forward Translation Table using the target value. In Receive Maps, the source Electronic Data Interchange (EDI) Standards value is looked up in the translation tables list of target values. When a match is found, the corresponding source value is substituted for the original value. Using Forward Translation Tables in a Receive Map is useful only when the target values are all unique. If the target values are not unique, you should use a Reverse Translation Table. Results will be unpredictable if you use a Forward Translation Table with non-unique target values in a Receive Map.

Send Maps always perform lookup on a Forward Translation Table using the source value. In Send Maps, the source Data Format value is looked up in the translation tables list of source values. When a match is found, the corresponding target value is substituted for the original value.

Accessing Translation Tables

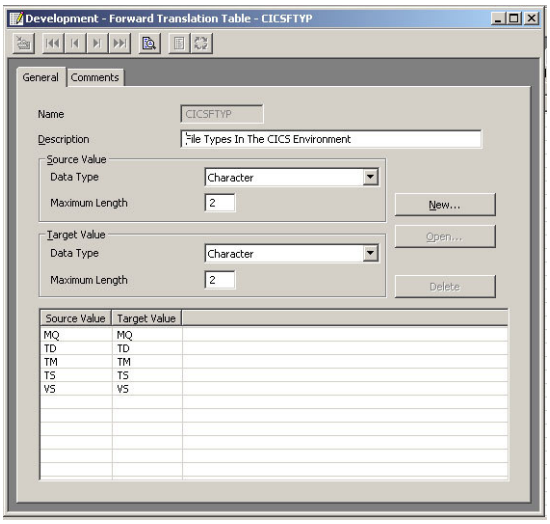
- From the Mapping Functional Area
  - Click on the Forward (or Reverse) Translation Tables tab
  - Table can be named for reference during mapping



Forward Translation Tables are located in the Mapping Functional Area. Open the Mapping Functional Area window by pressing the Mapping button on the WebSphere Data Interchange Client Navigator bar. There you will find the Forward Translation Tables List window. The list window is used to list and perform maintenance functions on Forward Translation Tables.

Accessing Translation Tables

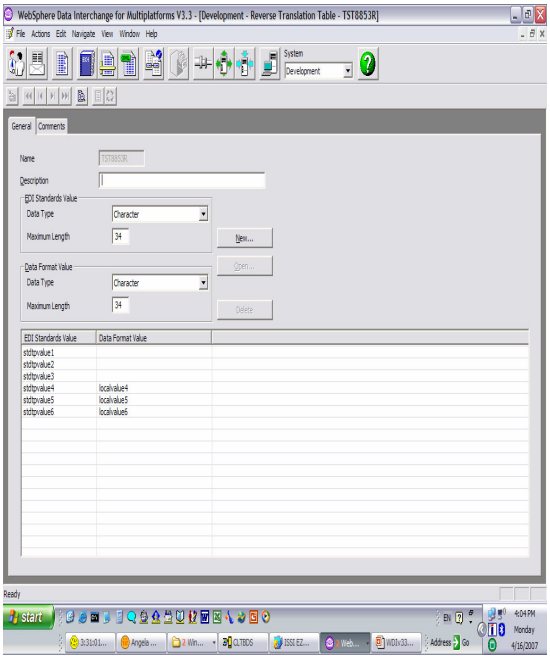
- Open a translation table editor
  - Double click on a table name
- Have two values which can be given sizes
- Values are designated as source value and target value



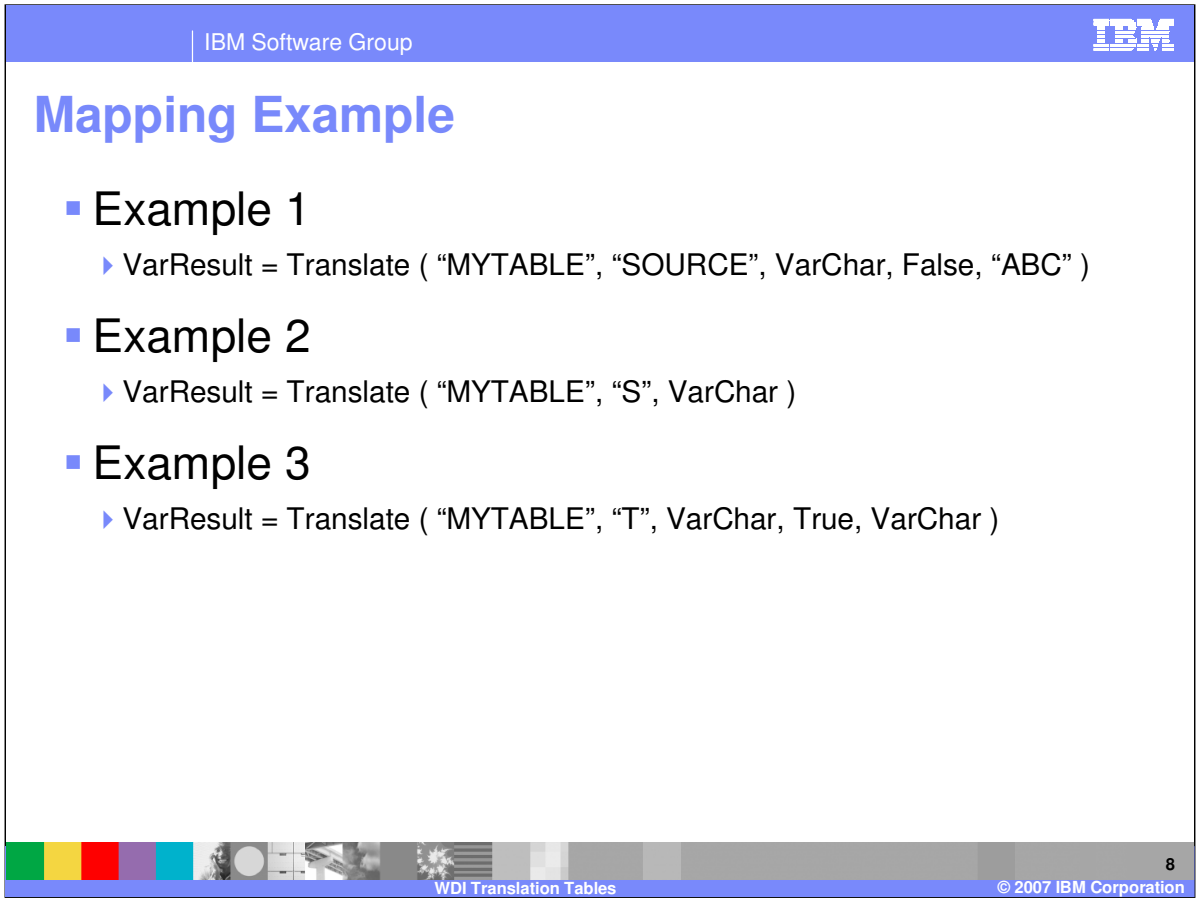
Use the Forward Translation Table Editor to add, view and change information about a Forward Translation Table. The Forward Translation Table Editor is accessed through the Forward Translation Tables List window. With a send map the application value is the source value in the table definition and the EDI value is the target value in the table definition. With a receive map the EDI source value in input data is used to locate the target value in the table definition and the source value in the table definition is used as the application target or output value. With Data Transformation maps the mapping command controls if the source or target value in the table should be used for the search.

Accessing Translation Tables

- Reverse Translation Table
  - Used for Receive
- Have two values which can be given sizes
- Values are designated as EDI Standards and Data Format value



A Reverse translation table is a special table used for the Receive process. With a receive map the EDI source value in input data is used to locate the target value in the table definition and the source value in the table definition is used as the application target or output value. With a reverse translation table the EDI source value in the input data is used to locate the EDI Standards value in the table definition and the Data Format value in the table definition is used as the application target or output value.



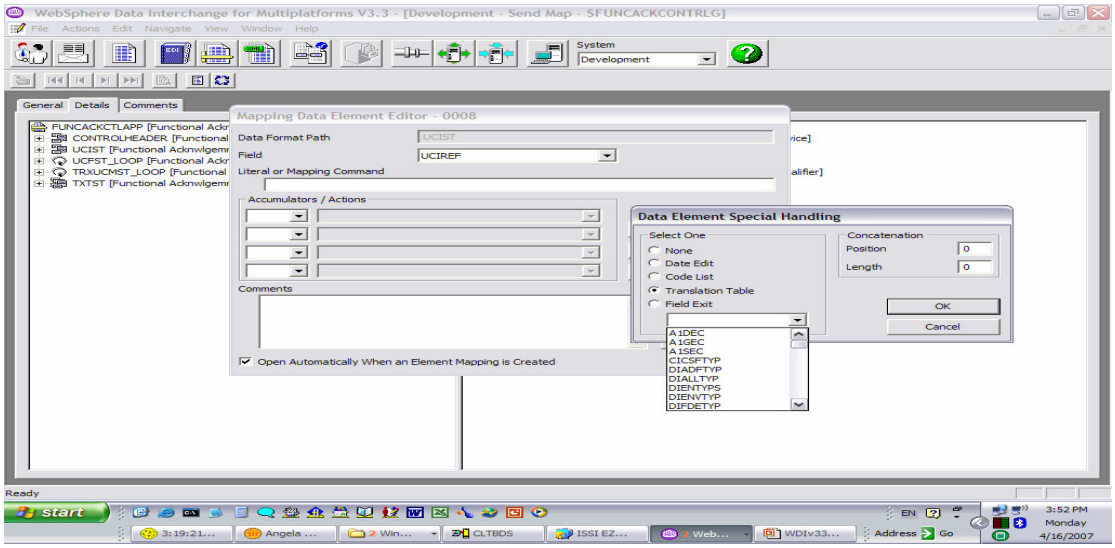
With Data Transformation maps, you associate translation tables with maps through the use of the Translate function.

In example 1, an attempt is made to locate the string contained in variable VarChar in the Source Value column of Forward Translation Table “MYTABLE”. If the value is not found in the translation table, a warning message is not issued and variable VarResult will be set to “ABC”. If the value is found in the translation table, variable VarResult will be set to the value in the corresponding Target Value column.

In example 2, an attempt is made to locate the string contained in variable VarChar in the Source Value column of Forward Translation Table “MYTABLE”. If the value is not found in the translation table, a warning message is issued and variable VarResult will be set to an empty string (“”). If the value is found in the translation table, variable VarResult will be set to the value in the corresponding Target Value column.

In example 3, an attempt is made to locate the string contained in variable VarChar in the Target Value column of Forward Translation Table “MYTABLE”. If the value is not found in the translation table, a warning message is issued and variable VarResult will be set to the string contained in variable VarChar. If the value is found in the translation table, variable VarResult will be set to the value in the corresponding Source Value column.

Mapping Example



With Send and Receive maps, you associate translation tables with maps through the use of the special handling or you can use an expression on the literal or command line.

Restrictions

- Table value sizes are limited
- The maximum length of the source values can be from 1 to 35.
- The combined maximum length of the source values and target values cannot exceed 68.

A translation table has two fields – the source and target. Each field is limited to a size of 35 bytes. In addition, the combined length of the sizes of the source and target fields can be no more than 68 bytes.

Internals

- DB2 Tables are EDIPSTD and EDIPSTV
  - EDIPSTD has table definitions for entry size and type
  - One row exists for each Table
  - Table name is the major key in both tables
  - Each code entry is a unique row in EDIPSTV
- Export / Import records are 8B1 and 8B2
  - 8B1 records contain EDIPSTD information
  - 8B2 records contain code values – VAR1 and VAR2

There are two DB2 Tables in WDI that house the Translation Tables as well as Code Lists. The EDIPSTD Table identifies each logical table. There is one row for each table. The key is the table name. Characteristics of the table, like size and type of the source element, size and type of the target element, are fields in the row. The EDIPSTV table contains the values of each table. Each value in the table is stored in its own row. Duplicate source values are not permitted (duplicate key) in Forward Translation Tables, but can exist as a Reverse Translation table.

The Export / Import subsystem creates a 8B1 record for each table based on the EDIPSTD table, and also one 8B2 record for each row in the EDIPSTV table.

Summary

- Translation tables allow the user to convert one value to another
- Primarily implemented to allow conversion of elements from the external TP values to your values for the same element
- Tables can exist for every element and are named
- Variable sizes can themselves vary by table
- Translate mapping command implements usage of a translation table for DT maps

In summary, Translation tables allow the user to convert one value to another. They are primarily implemented to allow conversion of elements from the external Trading Partner (TP) values to your values for the same element. A Translation table can exist for a number of elements. Tables are named and can be referenced in mapping commands. Each entry for a table is the same size, but each table can have different sizes for the source and target elements. The Translate mapping command implements the ability to use a translation table for Data Transformation (DT) maps.

Template Revision: 04/25/2006 11:09 AM

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM	CICS	IMS	WMQ	Tivoli
IBM(logo)	Cloudscape	Informix	OS/390	WebSphere
e(logo)business	DB2	iSeries	OS/400	xSeries
AIX	DB2 Universal Database	Lotus	pSeries	zSeries

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds.

Other company, product and service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or program(s) described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (e.g., IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.