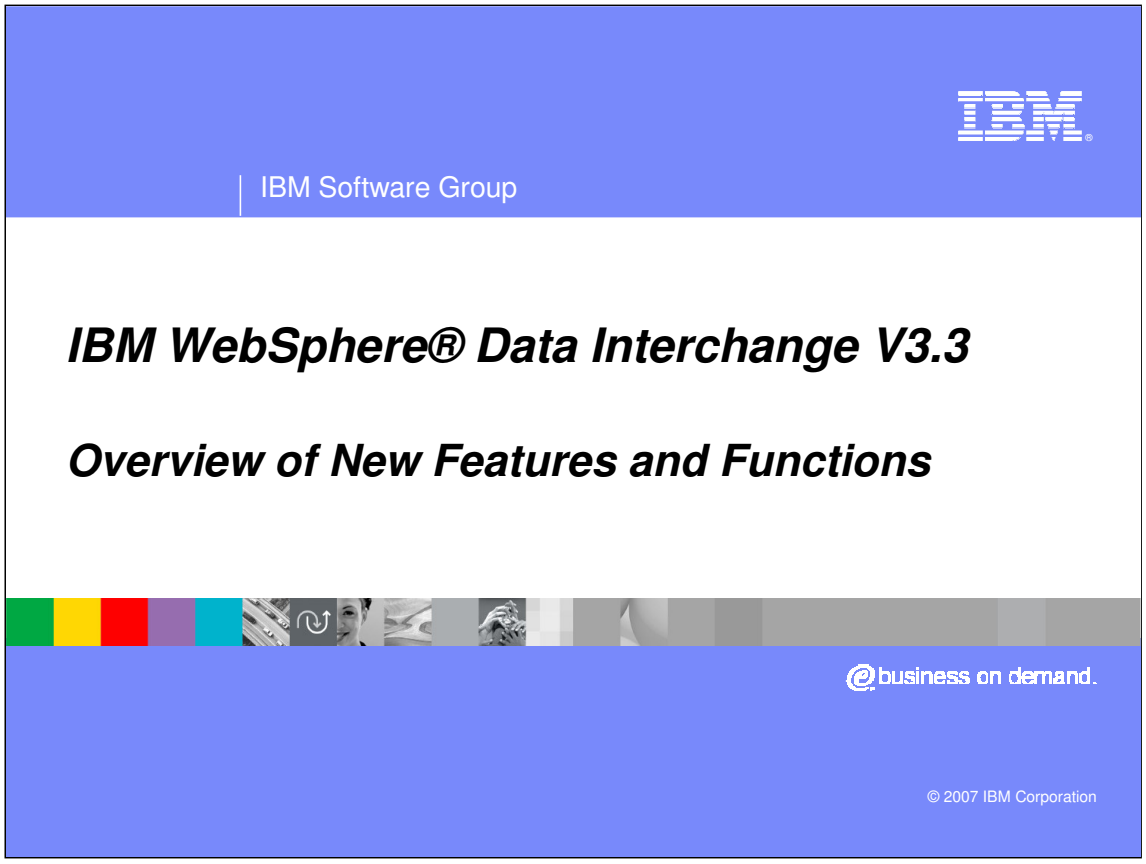


This presentation describes new features and functions included with IBM WebSphere Data Interchange version 3.3.

Agenda

- Send/Receive to Data Transformation Migration
- Infrastructure changes
- Additional Feature / Function
- Summary and references

The presentation provides a review of the new features for send and receive to Data Transformation (DT) migration and infrastructure changes.

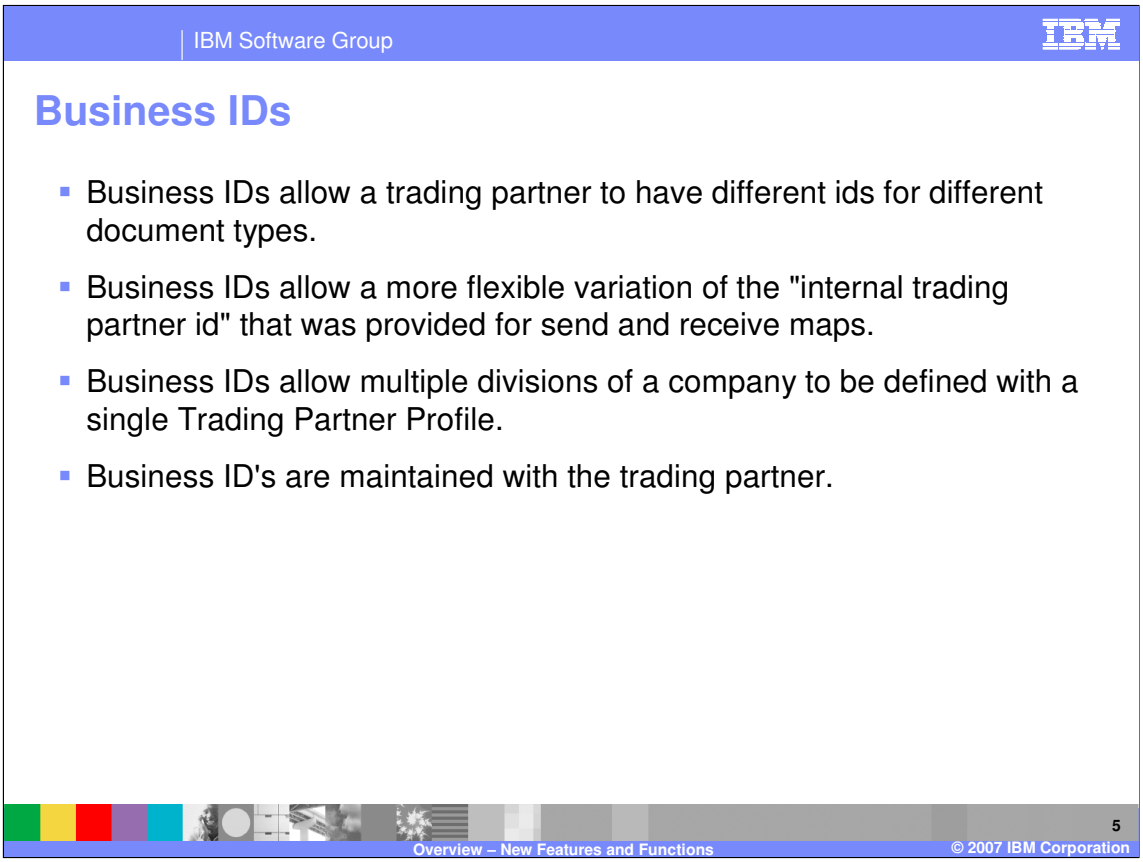
Section

Send/Receive to Data Transformation migration

Send/Receive to Data Transformation Migration

- Add Business Ids / Internal Trading Partner ID functionality for DT maps
- Add SetElementAttribute / &ZEROSIG for DT Maps
- Allow Management Reporting (MR) for DT maps - extend MR to XML and data format (DF) data
- Add XML and flat file data to DT Document Store Interface

Business Ids were added to the trading partner profile to provide the internal trading partner ID functionality for Data Transformation (DT) maps. The SetElementAttribute command was added for DT maps to provide the &ZEROSIG function. Management reporting is now available for the DT processing and has been extended to XML and data format (DF) data. The Transaction Store is now the Document Store and provides processing information for XML and flat file or application data processing.



In the area of Send map and Receive map migration to Data Transformation (DT) mapping, a couple of additional capabilities that were in Send and Receive mapping are included that did not previously exist. The first of these is the Trading Partner Business ID capability for DT maps. This is functionally equivalent to the Internal Trading Partner capability of Send mapping and Receive mapping.

&ZEROSIG or SetElementAttribute Functionality

- Set Element Attribute command
  - Created to support significant zeros in optional EDI data elements with Data Transformation processing, i.e. &ZEROSIG
  - Controls formatting of elements
    - WDI 3.2 DT maps automatically format values on output
      - EDI numeric elements will trim leading and trailing zeros.
      - When the value is zero and the element is optional, the value is suppressed in the EDI output.
    - WDI 3.3 SetElementAttribute mapping command
      - Allows customers to control this type of formatting for
        - zeros
        - leading and trailing blanks
        - left and right justification
        - empty elements

There is a particular need for customers to control zero values for an EDI optional element in Data Transformation (DT) mapping. This was handled with the &ZEROSIG mapping command for Send mapping. WebSphere Data Interchange (WDI) has created the Set Element Attribute command to support handling of elements with significant zeros in optional EDI data elements with Data Transformation processing.

The purpose of the new mapping command is to control formatting of elements. WDI currently does automatic formatting of values on output. For example, EDI numeric elements will trim leading and trailing zeros. If the value is zero and the element is optional, the value is suppressed in the EDI output. With the SetElementAttribute mapping command customers can control this type of formatting including how zeros, leading and trailing blanks, left and right justification, and empty elements are handled for output. SetElementAttribute only applies to DT mapping.

&ZEROSIG or SetElementAttribute Functionality

- SetElementAttribute command
- Placed anywhere in a Data Transformation map
- Target path specification in the command
  - target path identifies the scope of the command execution.
  - scope can be for the
    - Entire document,
    - Specific compound elements (including loops, segments and records, and composite elements and structures),
    - Specific simple elements
- Allows customers to apply the formatting control at specific levels in the mapping without the need of a mapping command at the simple element level for all elements involved.

The SetElementAttribute command can be placed anywhere in a Data Transformation map. The target path specification in the command identifies the scope of the command execution. The scope can be for the entire document, specific compound elements (including loops, segments and records, and composite elements and structures), or, specific simple elements.

The scope capability of this command allows customers to apply the formatting control at specific levels in the mapping without the need of individual mapping command at the simple element level for all elements involved.

SetElementAttribute Command Syntax:

- **SetElementAttribute** (*path element_spec, string attribute_name, string value*)
- Does not apply to the following element types:
  - XML sequence node (SQnnnn)
  - XML choice node (CHnnnn)
  - XML attribute list node (ATTLIST)
  - EDI Table node.
- element_spec– specifies the element on which to assign the property. Valid values are:
  - path - The path to a target element.
  - this – The special keyword "this", which indicates the current source or target element.
  - targetRoot – The special keyword "targetRoot", which refers to the root element of the target document.

The syntax of the SetElementAttribute command is shown in this slide.

This command does not apply to the following element types: XML sequence node (SQnnnn), XML choice node (CHnnnn), XML attribute list node (ATTLIST), and EDI Table node. Since these are not real nodes in the input data, they cannot have attributes associated with them at execution time.

The “element_spec” specifies the element on which to assign the property. Valid values are “path”, “this”, and “targetRoot”. Path means the path to a target element. The target element can be an XML element, attribute, or value, an EDI loop, segment, composite element, or simple element or an data format loop, record, structure, or field.

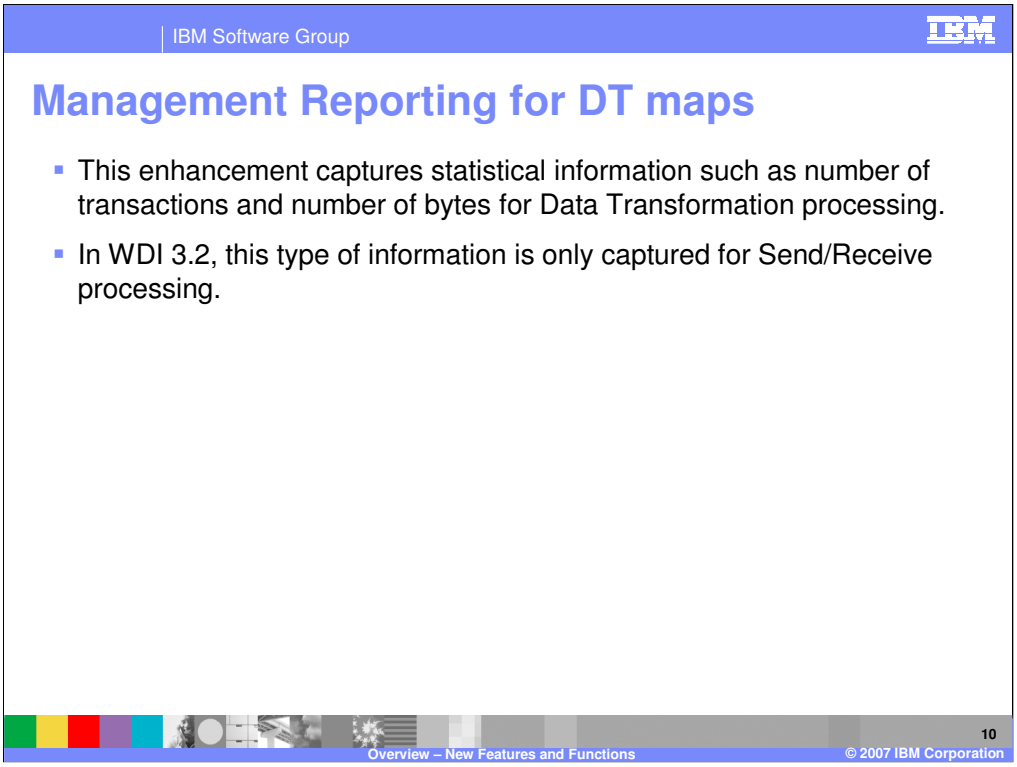
The special keyword “this” indicates the current source or target element.

The special keyword “targetRoot” refers to the root element of the target document. It would apply to the entire target document, unless the attribute is overridden at a lower level.

SetElementAttribute Command Syntax:

- string attribute_name - WDI attribute names.
- Valid values are:
  - TrimLeadingZeros
  - TrimTrailingZeros
  - TrimLeadingBlanks
  - TrimTrailingBlanks
  - SuppressZeroValues
  - SuppressEmptyElements
  - CharValueTooShort
  - NumValueTooShort

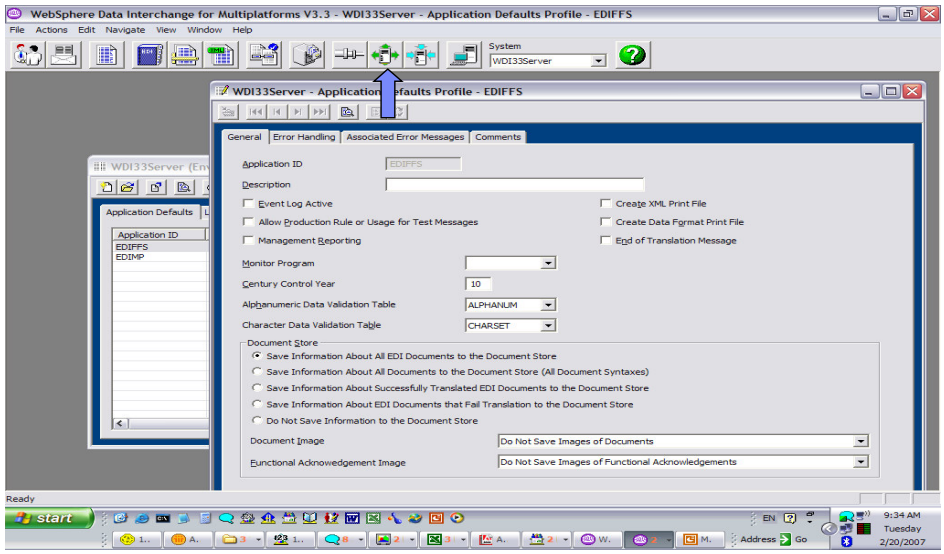
A list of WDI attribute names and valid values is shown on this slide.



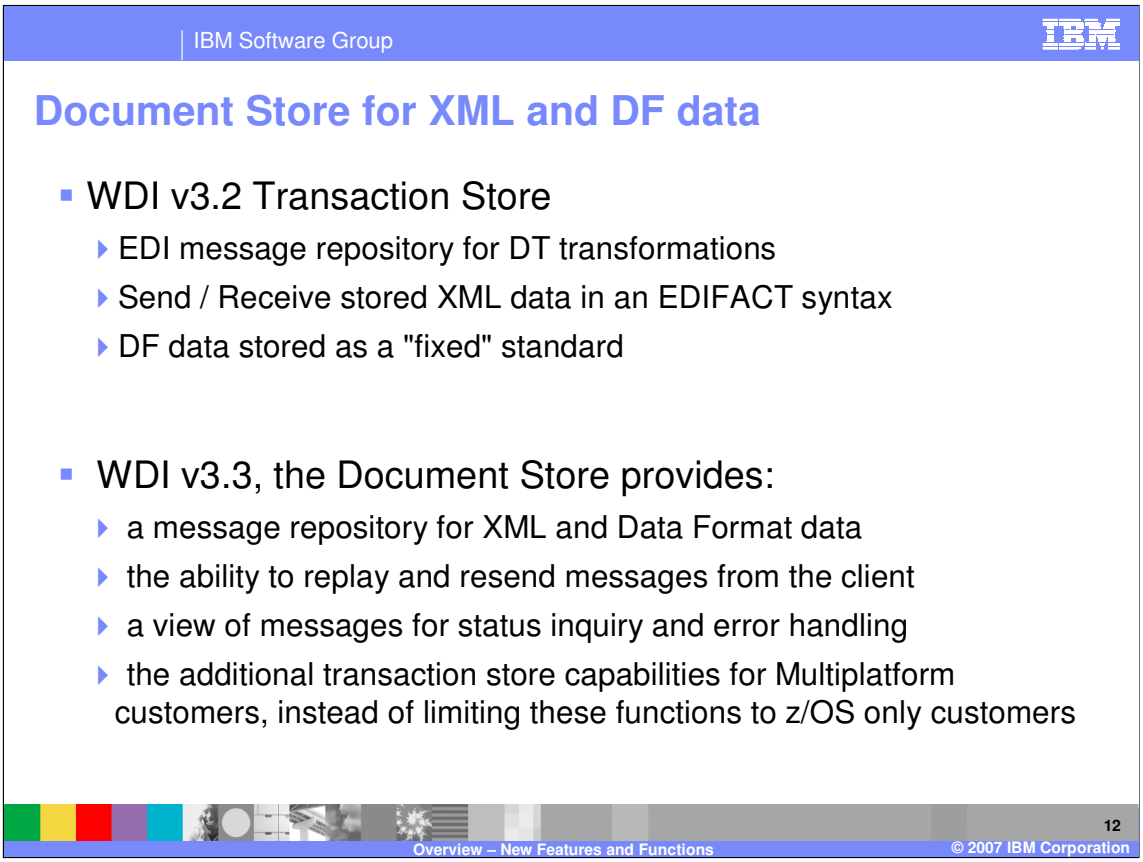
The Management Reporting subsystem of WDI captures statistical information such as number of transactions and number of bytes. In WDI version 3.2, this type of information was only captured for Send map and Receive map processing.

In WDI version 3.3 this functional can also be captured and reported for Data Transformation map processing.

Management Reporting



Management Reporting can be activated using the Application Defaults Profile located in the Environment functional area. When the checkbox is selected, Management Reporting is collected during translation.

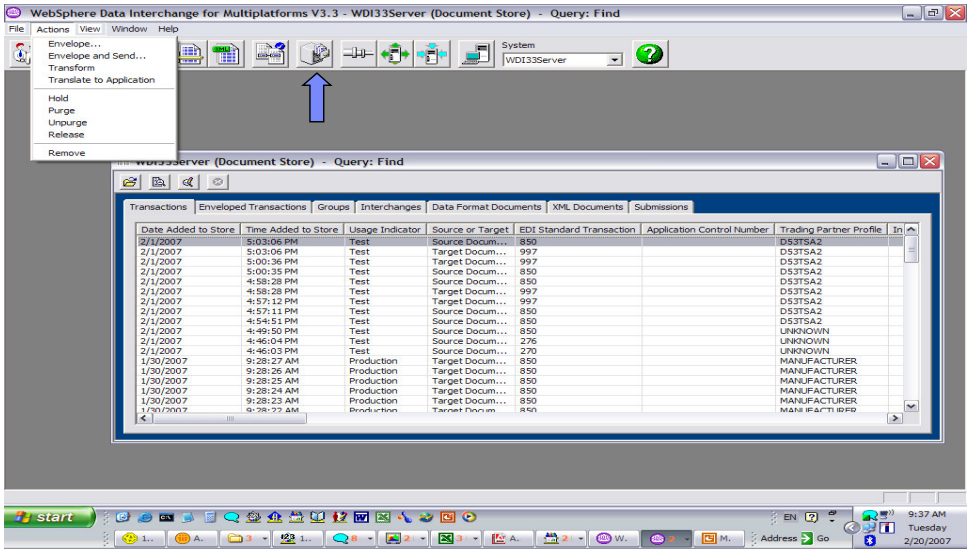


The WDI version 3.2 Transaction Store was an EDI message repository for DT transformations and Send and Receive translations. Send and Receive XML data was stored in an EDIFACT syntax. Data Format data was stored as a "fixed" standard.

In WDI version 3.3, the Document Store is introduced. It provides

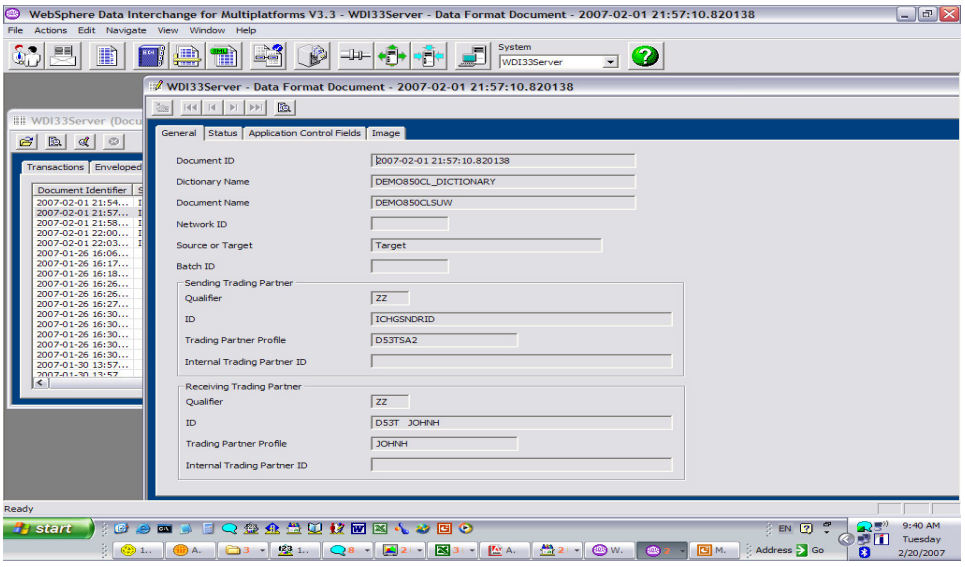
1) a message repository for XML and Data Format data
2) the ability to replay and resend messages from the client
3) a view of messages for status inquiry and error handling
4) the additional transaction store capabilities for Multiplatform customers, instead of limiting these functions to z/OS only customers

Document Store



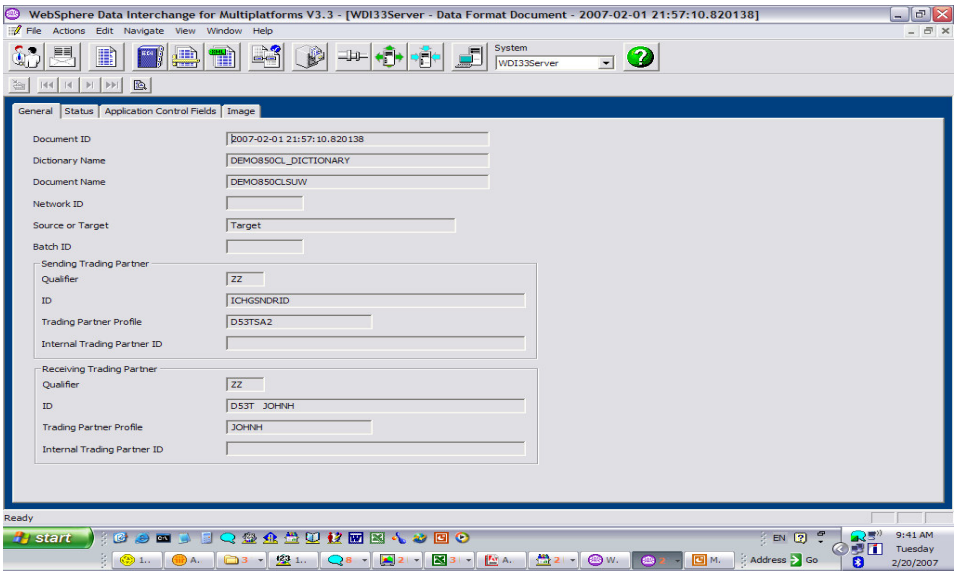
Groups of documents in the Document Store can be selected and have a number of actions that can be used to process the selected transactions. These are show in the screen shots above.

Document Store



In this example, typical information available for a document is shown.

Document Store



Here is another view of the Data collected.

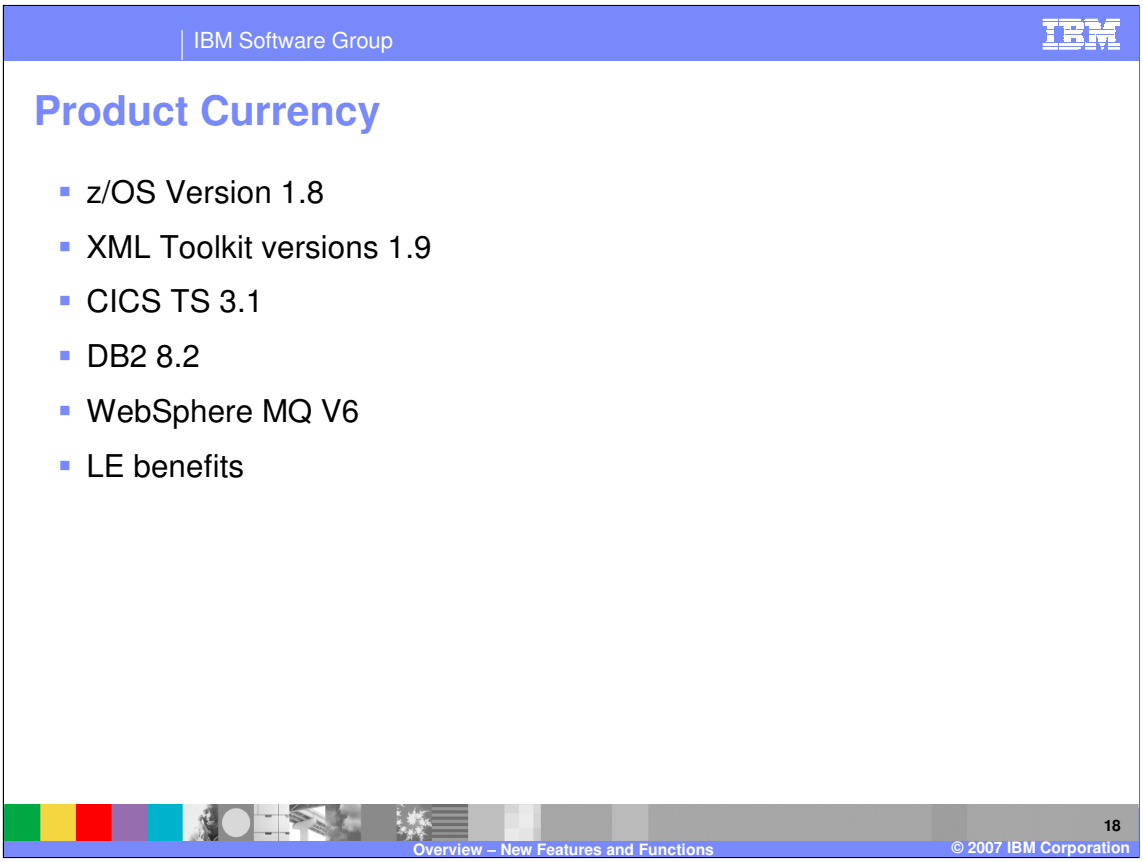
Section

Infrastructure changes

Infrastructure changes

- Remove z/OS Admin Facility User Interface
- Upgrade to z/OS C/C++ Compiler from VM C/C++ compiler
- Upgrade version of XML Toolkit for z/OS
- Upgrade version of DB2, WMQ
- Convert print reports to HTML and eliminate Crystal Reports

Infrastructure changes include the removal of the z/OS administration facility. Upgrades were made for the C/C++ compiler, the XML Toolkit for z/OS, DB2, and WebSphere MQ. The Client reports were converted to HTML format.



The internal changes were made to update the product base and platform support. WDI version 3.3 now supports z/OS version 1.8, the XML Toolkit used is version 1.9, and Language environment (LE) parameters can now be specified via JCL statements. Platform support has been extended to allow WDI 3.3 to run using CICS TS version 3.1, DB2 version 8.2, and WebSphere MQ version 6.

Section

Additional Features and Functions

Additional Feature/Function

- Allow multiple, disjoint record ids (improves SAP support)
- Add X12 999 FA support
- Provide a Java API similar to the C++ API
- Globalization improvements (allow encoding to be specified for all source/target syntaxes)

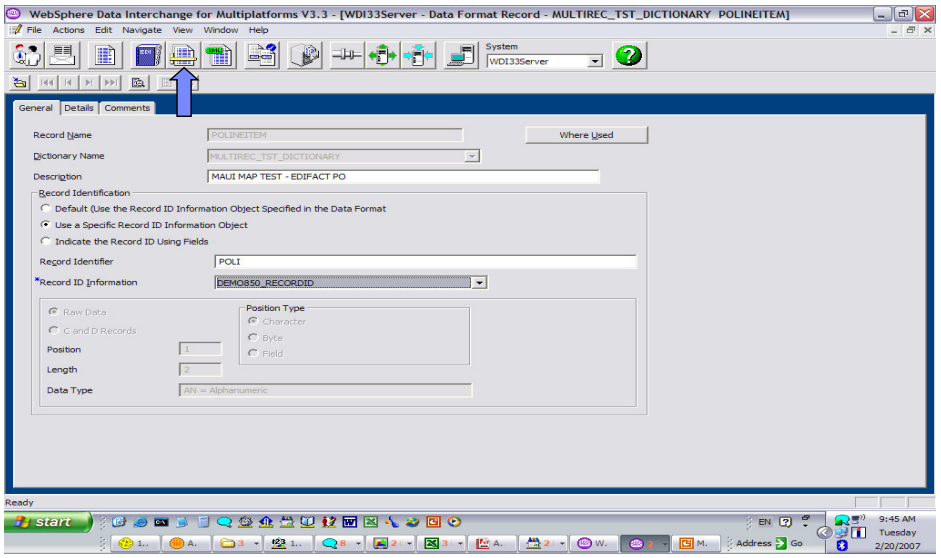
Additional features and functions were added for data format record identification and support for the X12 999 functional acknowledgement. A Java application programming interface (API) similar to the existing C++ API is provided. All source and target syntax types now provide character encoding options.

Allow Multiple Record IDs

- SAP has an extended key length
  - multiple values with a size greater than 16 bytes
  - allows for a 16 byte "floating" key
- WDI now has
  - primary record key of up to 64 bytes
  - additional record keys,
    - dependent upon the value of the primary key
  - keys need not be consecutive or adjoining

The application data record identification enhancement extends the key length and allows records to be identified on a per record basis. For example, SAP has an extended key length which means multiple key values with a total field size greater than 16 bytes. WDI now allows for a 64 byte "floating" key. This change allows WDI to have a primary record key of up to 64 bytes, and additional record keys, dependent upon the value of the primary key. The keys need not be consecutive or adjoining.

Record Keys

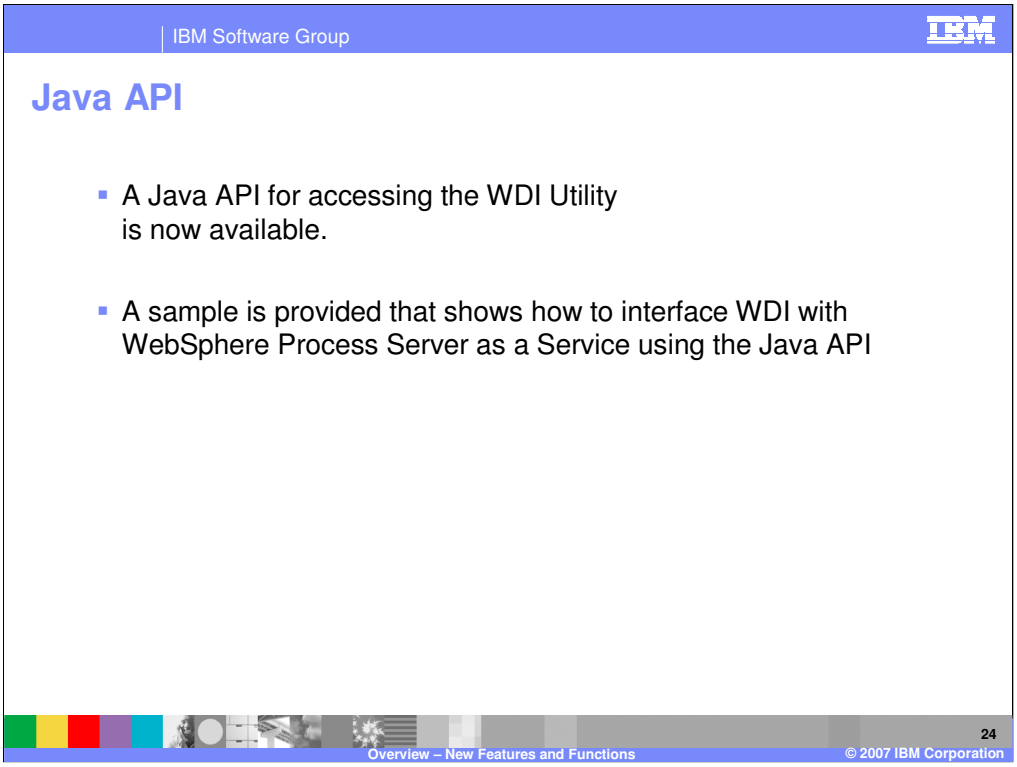


Record keys can be defined on the General tab of the Data Format panel.

ANSI X12 999 Transaction Set Support

- ANSI ASC X12 has implemented the Acknowledgment Transaction Set 999.
  - purpose is to report validation errors when a message is validated against an implementation guideline
  - different from the 997 which report errors validating against the message published by X12
  - 999 replaces the 997 when used.
- 999
  - compliments the validation map feature of WDI.
  - validation map validates the implementation guideline and feeds error information into the 999 transaction
  - validation component validates against the base message definition and feeds the 997 transaction.

ANSI ASC X12 has implemented the Acknowledgment Transaction Set 999. The purpose of the 999 is to report validation errors when a message is validated against an implementation guideline. This is different from the 997 that is used to report errors validating against the definition of the message published by the X12 standards bodies. The 999 would replace the 997 when used.



A Java application programming interface (API) for accessing the WDI Utility is now available.

A sample is provided that shows how to interface WDI with WebSphere Process Server as a Service using the Java API.

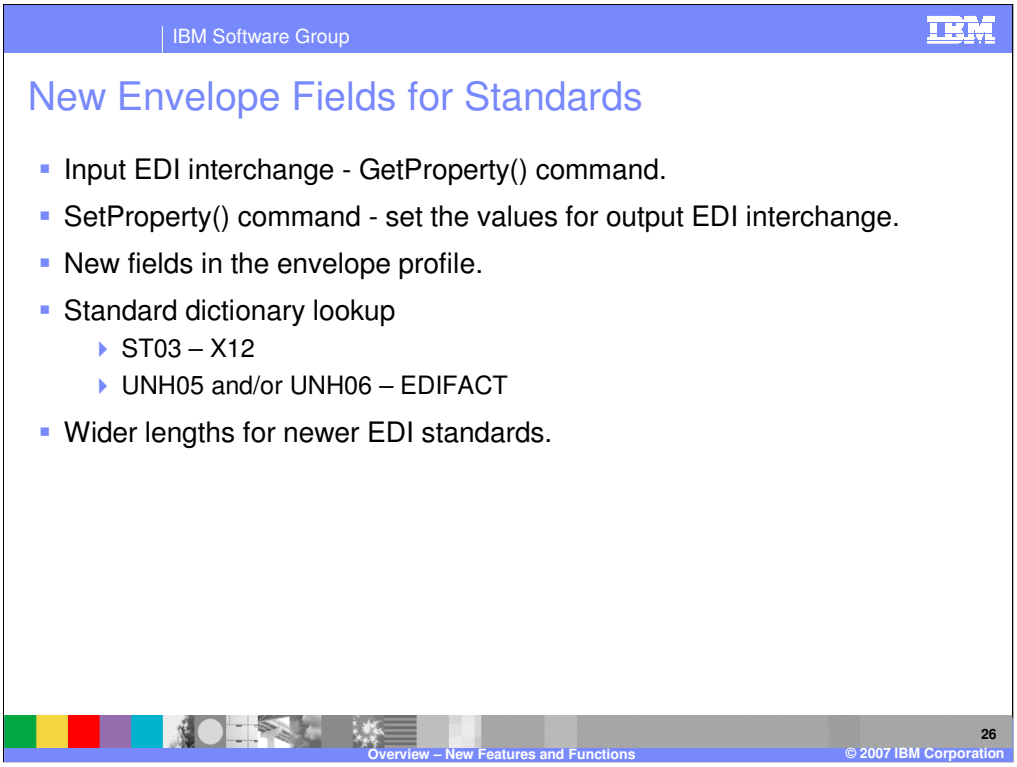
Globalization Data Improvements

- WDI v3.2 enhanced the XML serializer to support a
- user controlled encoding for data output. This is now extended to EDI output and ROD output as well.

- Double byte character set support is now available
- during translation and as part of the WDI database

WDI version 3.2 enhanced the XML serialization to support a user controlled encoding for data output. This is now extended to EDI output and record oriented data (ROD) output as well.

Double byte character set support is now available during translation and as part of the WDI database.



Users can access and set the new envelope fields as follows:

When the new fields are specified in an input EDI interchange, users can get the values in their map using the GetProperty() command.

Users can use the SetProperty() command to set the values of the new fields in an output EDI interchange.

Users can specify the new fields in the envelope profile.

The ST03 value is used as part of the dictionary lookup for X12 transactions, and the UNH05 and/or UNH06 values is used as part of the dictionary and document lookup for EDIFACT transactions.

Existing data elements support the wider lengths where defined by the newer EDI standards.

New Envelope Fields for Standards

- Existing issues with the envelope profiles, deferred enveloping, and transaction store have been addressed.
- Resolve long-standing limitations, problems, and usability issues.

In addition, various other existing issues with the envelope profiles, deferred enveloping, and transaction store have been addressed.

These are not really new functions or features, but resolve long-standing limitations, problems, and usability issues.

New Envelope Fields for Standards

- Removed envelope profile values that are ignored by the translator
- EDIFACT and UN/TDI envelope profiles field displays include segment, element number, sub-element number.

Some envelope profiles values are ignored by the translator, such as control numbers, dates, times, and segment counts. Users will no longer be able to set these values in the envelope profile.

EDIFACT and UN/TDI envelope profiles were previously displayed to the users with labels such as UNBnn, even when they are actually subelements. For example, element 3, subelement 2 of the UNB segment (receiver qualifier) was displayed as "UNB07“. These have been changed to the form UNBeess, where ee is the data element, and ss is the subelement. For example, UNB0302. This makes the envelope profile label consistent with the property name used in GetProperty and SetProperty commands.

New Envelope Fields for Standards

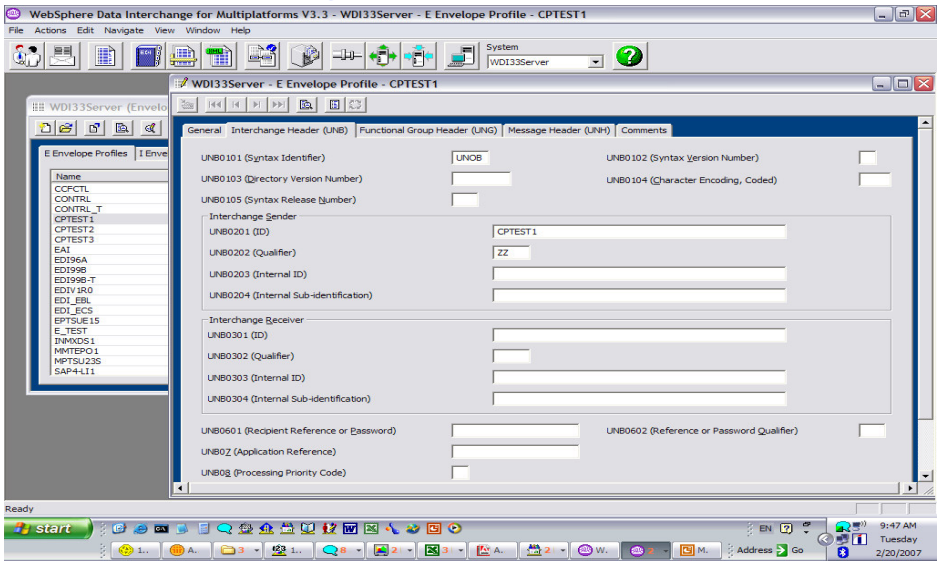
- The document store EDITSTO table and handling has been updated so all envelope properties can be stored.
- Improved duplicate detection
- The UNTDI/Tradacoms (T) envelope now sets the transaction store overrides correctly.

When doing deferred enveloping, only a subset of the envelope properties were held in the EDITSTO table. The remaining properties were discarded. This results in different envelope values depending on whether the transaction was enveloped immediately or enveloping was deferred. The EDITSTO table and handling has been updated so all envelope properties can be stored.

In certain cases, transaction store did not work correctly. If the sender TP nickname is UNKNOWN, then two interchanges with the same control number were treated as a duplicate, even if the sender id was different. This has been corrected.

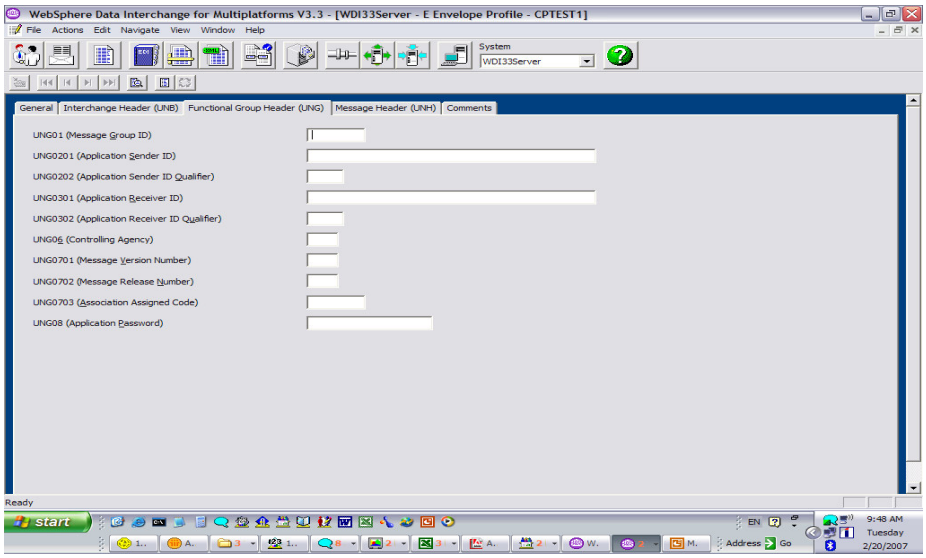
The UNTDI/Tradacoms (T) envelope now sets the transaction store overrides correctly.

EDIFACT Message Elements

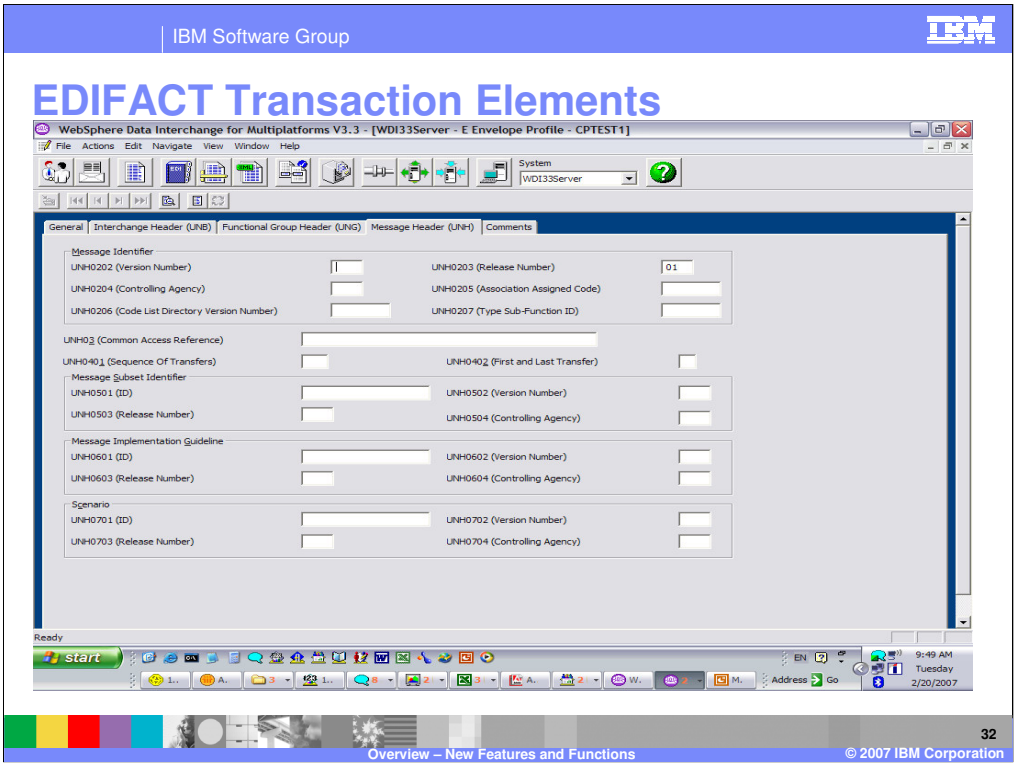


This is an example of the EDIFACT Message (UNB) elements supported.

EDIFACT Group Elements

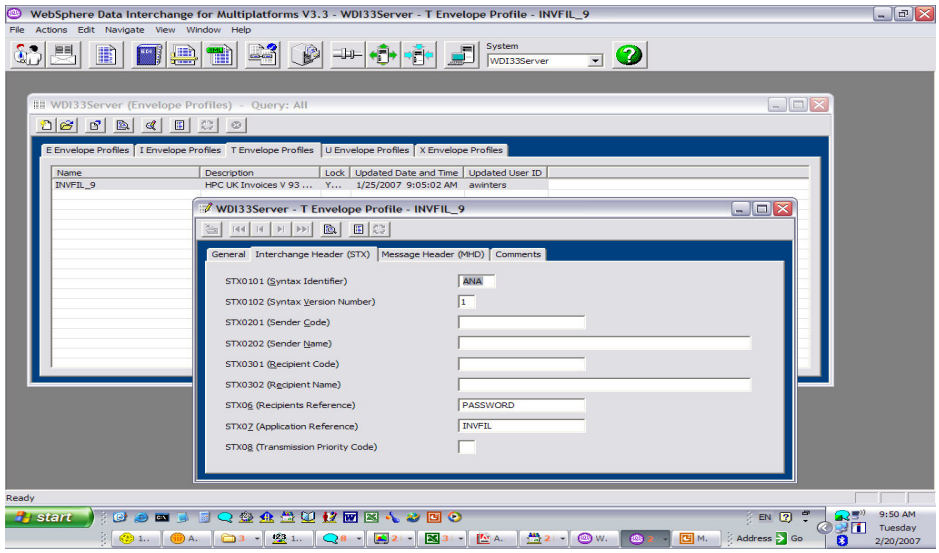


This is an example of the EDIFACT Group (UNG) elements supported.

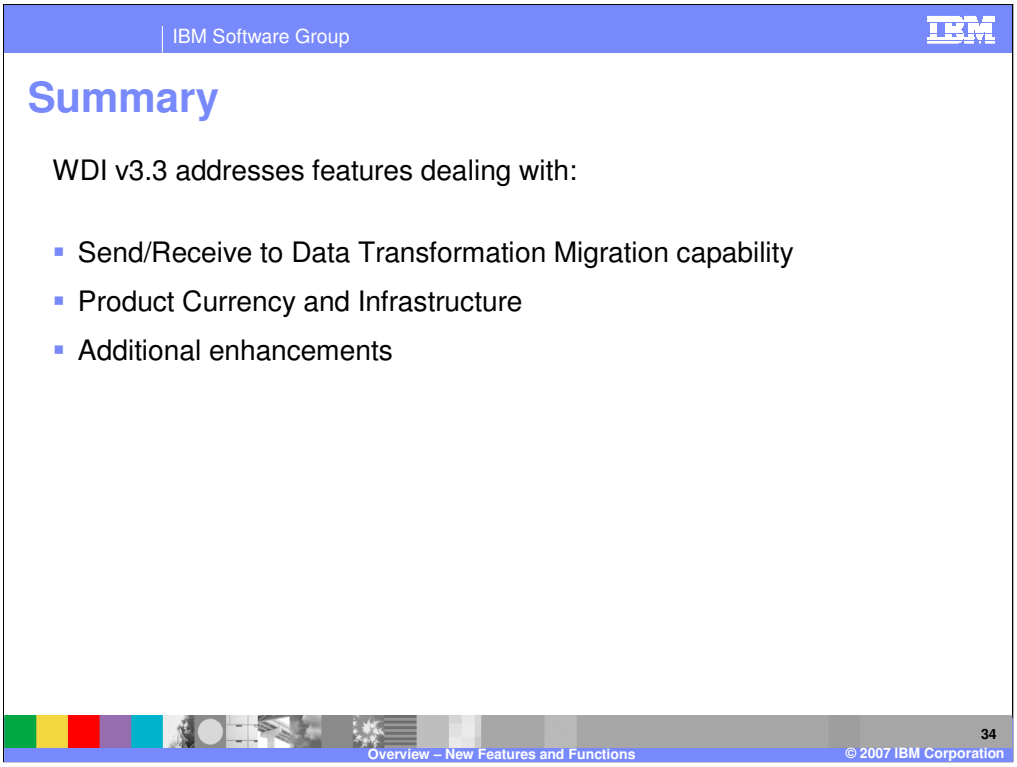


This is an example of the EDIFACT Transaction (UNH) elements supported.

TradaComs Interchange Elements



This is an example of the TradaComs (UN/TDI) Interchange (STX) elements supported.



In summary, WebSphere Data Interchange version 3.3 has addressed a number of product areas. See the related presentations about those that are of interest to you. Thank you for your attention.

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.