**WTX**

WebSphere Transformation Extender has a unique many-to-many model of transforming and processing data. With this model, WebSphere Transformation Extender can run all transforms, lookups, and data enrichments with only one pass at the data, making it one of the best

performing transformation engines on the market. Data is validated to content rules and context usages as part of the transformation process. It is not necessary to write separate logic or have

separate executions to provide extremely rich data validation. There is no “language” to WebSphere Transformation Extender. The transforms and data process are all maintained within the spreadsheet-type GUI. It is not necessary to write code to handle complex transforms. WebSphere Transformation Extender uses self-describing data models to handle data in its native format and has a unique mechanism for describing data in its native form.

**Problems that WebSphere Transformation Extender solves**

• Connects applications, databases, processes within the enterprise

• Partners processes across the enterprise

• Powerful transformation and routing

• No coding

• Consistent approach to multiple types of integration

• Reusable objects

• Easy design and implementation

• Fast maintenance without coding

• Rapid adaptation to change

• Uses the existing IT infrastructure

• Fully supports existing IT investments such as databases and messaging middleware

• Non-invasive integration

• Application and industry packs

**WebSphere Transformation Extender Design Studio**

The WebSphere Transformation Extender Design Studio components are the design-time applications that are used to model, develop, manage, and test data transformations, application integration maps, and systems in a development environment. Design Studio provides the means for developing event-driven application-to-application integration, business-to-business integration, and consumer-to-business application integration. Integration developers and data architects use Design Studio to visualize both simple and complex data types.

You can use Design Studio to complete the following tasks:

\_ Define your data

\_ Create maps for your data

\_ Manage the systems of maps that you create

\_ Develop and test maps and systems in a Microsoft® Windows® environment

Design work can be centralized or distributed. A single Design Studio can serve

multiple servers, or multiple Design Studios can be used to create interfaces for a

single production environment.

Design Studio includes several components:

**Type Tree Editor** The modeling component that is used to define data objects, including source and target data structures.

**Map Editor** The modeling component that is used to develop maps that define your data transformation logic.

**Database Interface Designer**

The modeling component that is used to import metadata about queries, tables, and stored procedures for data stored in relational databases. The Database Interface Designer identifies characteristics, such as update keys and database triggers, of those objects to meet mapping and execution requirements.

**Integration Flow Designer**

The modeling component that is used to define and manage data integration processes. Use the Integration Flow Designer to define interactions among maps and systems of maps, to validate the logical consistency of workflows, and to prepare systems to run. The Type Tree Editor and Map Editor Are Eclipse-based tools. The Eclipse workbench delivers a common, standards-based user experience across the WebSphere portfolio, and is supported by many independent software vendors.

**XML schemas:** The Design Studio has strong support for XML schemas. This support includes the use of an XML schema directly in a map, switchable on/off XML schema validation on a per-card basis. It also includes support for the XSD Union type and for the subtype attribute

Transformations and validations can be fully developed, analyzed, and tested on a developer’s workstation by using Design Studio, before generating a platform-specific compiled format for use with any WebSphere Transformation Extender edition engine. As a file-based solution, no additional database repository is required to manage the transformation artifacts. Any preferred Eclipse-supported source- code management solution can be used to manage the assets between development teams.

**Type Tree Editor**

The Type Tree Editor is the design component that is used to specify, define, and manage type definitions in the form of type trees. A *type tree* is a data dictionary that defines how types are classified. The type tree files that you create in the Type Tree Editor are data definition files. Each object identified in your data is defined as a *type* in your type tree. Type trees can also be created using the Type Tree Maker, the Importer Wizard, and the Database Interface Designer.

You can use the Type Tree Editor to define the following information:

\_ Properties for data structures

\_ Data validation rules

\_ Data as text or binary

\_ Different character sets

A type tree describes the syntax, structure, and semantics of your data. The syntaxof data refers to its format. The format of data includes tags, delimiters, terminators, and other characters that separate or identify data breaks. The *structure* of data refers to its composition, including repeating substructures, nested groupings, sequences, and choices. The *semantics* of data refer to the

meaning of the data, including rules for data values, relationships among parts of large data object, and error detection and recovery. The Type Tree Editor includes an importer tool for automatically generating type trees for data that is described in XML, COBOL, and other formats or application

data structures.

**XML schemas:** The Design Studio has strong support for XML schemas. This support includes the use of an XML schema directly in a map, switchable on/off XML schema validation on a per-card basis. It also includes support for the XSD Union type and for the xsi: type attribute.

**Map Editor**

The Map Editor is an application in Design Studio that is used to specify data transformation logic in the form of map rules. The Map Editor uses the definition of data objects that are created in the Type Tree Editor as inputs and outputs. The Map Editor provides the functionality to specify rules for the transformation and routing of data. Maps are analyzed, compiled, and tested by using the Map

Editor. This process is facilitated by convenient “from” and “to” windows, drag-and-drop techniques, and spreadsheet-type rules. Mapping rules are added by anagogical statements. The Map Editor provides arch set of more than 100predefined functions for operations such as conditional testing, table lookups,

mathematical functions, character string parsing, date and time handling, and data extraction.

The Map Editor is used to perform the following tasks:

\_ Create maps to specify the logic necessary for transforming the input data to

the desired output data

\_ Identify the source and data objects of the input data

\_ Identify the target and data objects of the output data

\_ Specify and build the output data according to the map rules

\_ Provide information about data validation by generating trace files

\_ View the run results of the map execution

You build and run maps by using the Map Editor on a Windows platform. You can also use the Map Editor to build a map to be run on another platform. Each map can be built for a specific platform and then executed on that platform to perform the transformation of the data.

**1.2.3 Database Interface Designer**

The Database Interface Designer is used to import metadata about queries,

tables, and stored procedures for data that is stored in relational databases. Use

the Database Interface Designer to identify characteristics of those objects to

meet mapping and execution requirements such as update keys and database

triggers.

Use the Database Interface Designer to perform the following tasks:

\_ Specify the databases to use for a source or target

\_ Define query statements

\_ Automatically generate type trees for queries or tables