

Product Overview: *etlmessenger*

In August 2007 The Data Warehouse Institute, www.tdwi.org, reported that 45% of Extract-Transform-Load, "ETL", projects are hand-coded, down from 60% in their 2005 report, and they rated ETL issues (graphic).

The top four reasons for hand-coding were:

1. Cost of ETL tools
2. Time taken to get budget and resources allocated
3. Pre-existing hand-coding
4. ETL tools are environment specific

1 Introduction

1.1 Overview

Data Architecture Strategy and, as an integral part of that, Data Synchronisation, is a key challenge for all enterprises; government and corporate, small and large.

A high priority in Data Architecture Strategy is the transference and synchronisation of structured data from one environment to another. Yet, to date, it is a non-trivial exercise for most enterprises. Typical solutions range from Enterprise Wide Integration Busses, through to customised scripting processes. These solutions are often complex, expensive, time consuming, and prone to error.

etlmessenger represents a unique and generic solution to this problem. It enables the transfer and synchronisation of structured data from one data source to an alternate data source utilising a controlled repeatable process, whilst at the same time allowing for data manipulation, transformation and validation.

Using *etlmessenger*, it is now much easier to keep a database from one vendor (e.g. Oracle 7) synchronised, in real time, with that of another vendor (e.g. MS SQL Server) or a structured data file (e.g. XML), notwithstanding any differences in data structure.

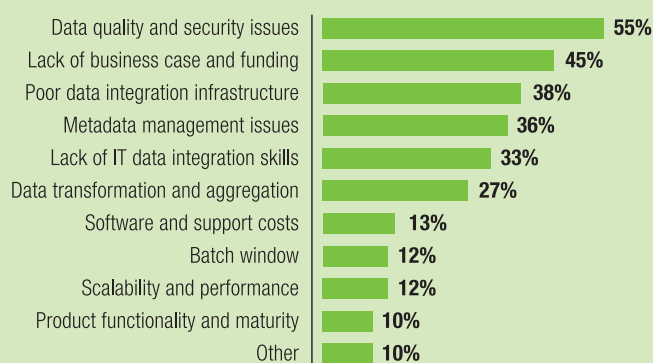
1.2 Background

etlmessenger was initially designed in 2001 based on a number of software inventions, and a provisional patent application was filed in July, 2001. Patent protection in the United States has already been advised for core inventions, and further patents are pending in the United States and Australia. This patent protection seeks to protect the technology which provides *etlmessenger* with its unique advantages.

etlmessenger has been successfully utilised, for example, to prove the feasibility of large and complex XML projects in the Australian Banking industry where a distributed enterprise environment required synchronisation, transformation, and validation of data between multiple different structured data sources in multiple enterprises.

1.3 Problem Description

etlmessenger solves the problem of data synchronisation, transformation, and validation across multiple disparate data sources through the use of a patent pending ETL process. Data can then also be transmitted via JMS, HTTPS, SMTP or FTP.



Source: The Data Warehouse Institute, www.tdwi.org

2 Product Description

2.1 Overview

etlmessenger is a Java based application which essentially performs the following key steps:

- Source Data Extraction
- Source Data Validation
- Source Data Transformation
- Intermediate Source Data Storage
- Data Transmission
- Intermediate Destination Data Storage
- Destination Data Transformation
- Destination Data Validation
- Destination Data Insertion/Update

The process is repeatable, and reversible, thus allowing a single solution to cover multiple different data sources and transmission platforms.

The transformation and validation processes are configured via XML mapping files, while the transmission is typically via JMS (though even this is configurable).

This allows the solution to transfer data from any structured data source, equally. Thus keeping an XML file synchronised with a database is no more complex than keeping two databases synchronised. The only limitation is that the database must support a JDBC (or ODBC) interface.

Internally *etlmessenger* is constructed with four key modules:

- Provider Service
- Adapter Service
- System Administrator
- Data Mapper

2.2 Key Components

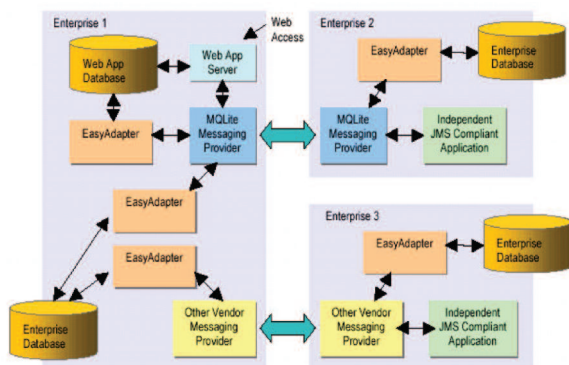
2.2.1 Adapter Service

The Adapter Service is the core component of *etlmessenger*. The Adapter Service performs the data extraction (or persistence), validation and transformation in accordance with predefined XML mapping files. A single Adapter Service operates in conjunction with a Provider service to implement one half of the solution.

2.2.2 Provider Service

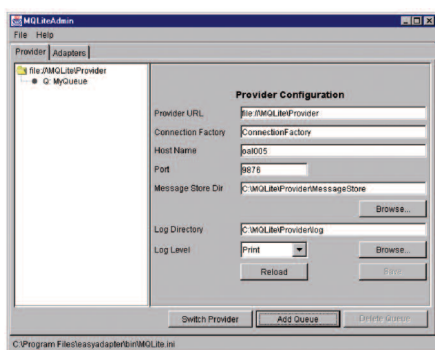
The Provider Service is the transmission service, allowing one Adapter

Service to communicate with another Adapter Service using a guaranteed delivery mechanism. Typically the Provider Service is implemented in accordance with the JMS standard. **etlmessenger** is packaged with MQLite, however this can be substituted with any existing or preferred JMS compliant messaging system. Provider transmission is configured via the GUI and stored as an XML navigation file.



2.2.3 System Administration

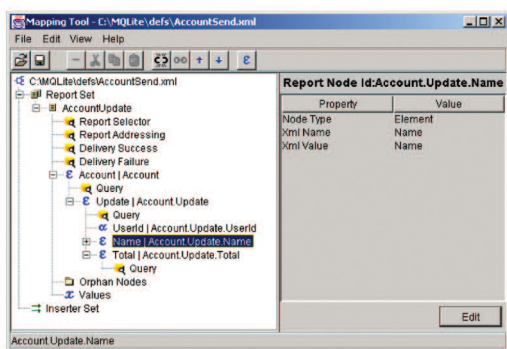
Configuration of the Provider Service and Adapter Service is via the System Administration tool. This is a graphical set-up tool where the administrator can specify where and how the Provider Service operates, where the Adapter Service is located, and what to do under error conditions.



The System Administration tool stores the settings in several external configuration files, which can (if so desired) be manually modified and managed to allow version control and management of the configuration.

2.2.4 Data Mapper

The Data Mapper is a graphical modelling tool which allows the administrator to interrogate the source (or destination) data source and, easily and quickly, build rules in relation to it to allow for simple and complex validation and transformation processes.



Mapping and transformation, including data manipulation, is configured via a GUI utilizing SQL commands, which are then stored in an XML configuration document. Rather than map directly from one structured data source to another, the patent pending technique utilized by **etlmessenger** involves data being initially read to an intermediate virtual data structure where it can be manipulated and transformed on the fly.

3 Technical Specification

etlmessenger is implemented as a J2SE application, and operates on any J2SE 1.4 or above installation.

etlmessenger operates in real time, polling the source data structure for events, such as a new or changed record, upon which the ETL and transmission processes are automatically invoked.

The use of the patent pending intermediate virtual data structure in **etlmessenger** provides for significant throughput performance over competitive approaches.

Utilization of XML allows validation for data quality.

Deploying common industry standards and technology such as Java, SQL, XML, JMS, the well known Xerces library to facilitate XML validation and parsing, and CyberNeko for HTML parsing, reduces learning time significantly and increases the speed of deployment.

3.1 System Requirements

As a J2SE application, **etlmessenger** simply requires the installation of any J2SE 1.4 RTE (or above).

etlmessenger makes use of a JNDI implementation which must be installed prior to operation. Typically this can be provided by an existing J2EE installation (though any JNDI implementation will suffice).

etlmessenger utilises a JMS implementation for data distribution. An inbuilt JMS implementation is provided with **etlmessenger**, although any existing or alternate JMS implementation can equally be utilised.

For Error Logging / Notification **etlmessenger** makes use of an email system (SMTP) to notify administrators of any transformation, validation or transmission errors. Thus access to an existing SMTP mail system is required to enable the error reporting functionality.

3.2 Database Requirements

There are no specific database requirements. The only database related requirement for **etlmessenger** is that the database is JDBC (or ODBC) compliant, and installed. As **etlmessenger** is a generic tool, equally applicable to most vendor databases, it does not ship with JDBC or ODBC drivers.

3.3 File System Requirements

etlmessenger is a J2SE based system and as such is equally capable of operation on a Windows Platform as it is a Unix Platform (or other compliant J2SE platform). All that is required is the appropriate permissions to write to the file system.

4 Conclusion

etlmessenger provides a unique solution to the problem of data synchronisation between different distributed data sources. It allows two Oracle databases to be kept in complete synchronisation just as easily and simply as it does a MS SQL Server and IBM DB2 database to be kept synchronised, or indeed a structured XML file.

etlmessenger is sophisticated, yet easy to deploy and configure. It performs complex transformations, but is not complex to use. It allows for source and target data structure change without the need to reconfigure. It is robust and continues its function when data errors are detected by validation, and error notification has occurred. It is rules based, avoiding the need for custom coding and/or complex and expensive scripting solutions.

Best of all **etlmessenger** has the lowest total cost of ownership in the ETL vendor industry without sacrificing any of the needs of large enterprises for a robust, standards based solution that provides appropriate data management functions.

Synchronisation and data quality can now be achieved quickly, simply and repeatedly using a single tool, **etlmessenger**.

So if you need

1. ease and speed of deployment (hours, not days)
2. ease and speed of mapping (days, not weeks; patent pending data transformation technique)
3. high throughput speed (patent pending data transformation technique)
4. multi-mode inbuilt messaging (via JMS, HTTPS, SMTP and FTP)
5. real-time or batch automated (rules and event based)

then **etlmessenger** may be a great solution for your enterprise.