



Data Warehouse Testing: Pharma

Jeff Bocarsly, PhD
Director
Functional Test Automation Group
RTTS

Data Warehouse Testing

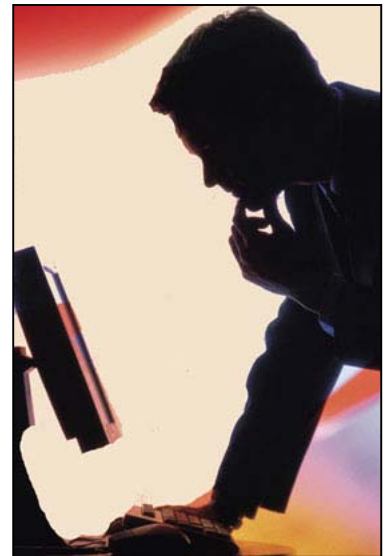
A data warehouse is a repository of transaction data that has been extracted from original electronic sources and transformed so that query, analysis and reporting on trends within historic data is both possible and efficient. The analyses provided by data warehouses may support an organization's strategic planning, decision support, or monitoring of outcomes of chosen strategies. Typically, data that is loaded into a data warehouse is derived from diverse sources of operational data, which may consist of data from databases, feeds, application files (such as office productivity software files) or flat files. The data must be extracted from these diverse sources, transformed to a common format, and loaded into the data warehouse. Extraction, transformation and loading (ETL) is a critical step in any data warehouse implementation, and continues to be an area of major importance over the life of the warehouse due to recurrent warehouse updating. Once a data warehouse is populated, front-end systems must be built to facilitate querying, analysis and reporting. The data warehouse front-end may provide a simple presentation of data based on queries, or may support sophisticated statistical analysis options. Data warehouses may have multiple front-end applications, depending on the various profiles within the user community. Figure 1 shows a simplified data warehouse scheme.

An effective data warehouse testing strategy focuses on the main structures within the data warehouse architecture:

- 1) The ETL layer
- 2) The data warehouse itself,
- 3) The front-end data warehouse applications

Each of these units must be treated separately and in combination, and since there may be multiple components in each (multiple feeds to ETL, multiple databases or data repositories that constitute the warehouse, and multiple front-end applications), each of these subsystems must be individually validated.

An example of the complexities that may occur comes from the Pharmaceutical industry, where a broad variety of data types and sources may be fed into a data warehouse. In addition to general Pharma-specific information exchange formats (e.g., HL7, CDISC), organizations may have multiple proprietary and internal data formats, which may have been acquired in the process of industry consolidation. Thus, pharmaceutical data warehouses often have to support a broader range of input formats to their ETL layers than data warehouses in industries where less consolidation has occurred.



Pre-Deployment Phase

Strategic Approach

The recommended pre-deployment strategy is to build test automation (both functional and performance) for every test entry point in the system (feeds, databases, internal messaging, front-end transactions). The goal of the strategy is to provide automated tools for rapid localization of data issues between test entry points (see Figure 1).

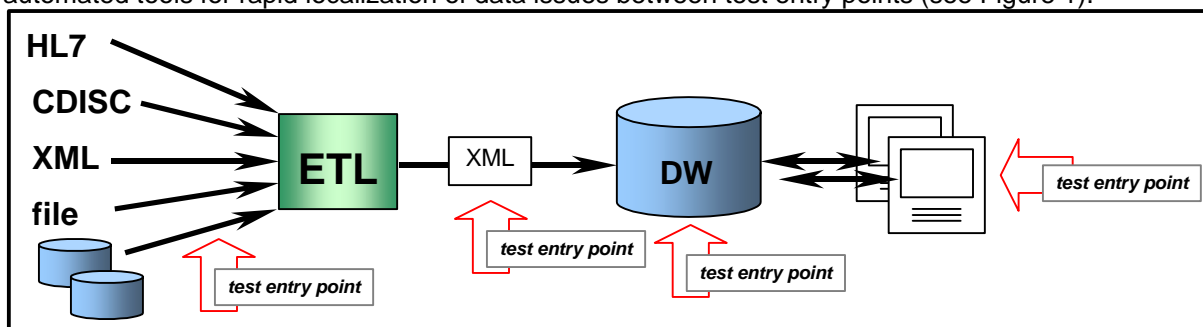


Figure 1. RTTS' Data Warehouse Pre-Deployment Testing Strategy

In the example shown in Figure 1, data from a variety of sources is transformed by the ETL component into XML messages that utilize a custom XML grammar designed specifically around company needs. The XML messages are

then parsed and stored in the data warehouse. Front-end applications access the data warehouse in order to provide historical and statistical analyses of company data.

As indicated in Figure 1, there are five test entry points to the system. In functional automation, the emphasis at each test entry point is the validation of data integrity. Using test automation, data can be tracked from the input layer, through the ETL processing and into the XML messages produced, on to storage in the data warehouse, to the front-end applications. So, for example, if corrupt data is found in a front-end application, the execution of automated suites can rapidly determine whether the problem is located in the data source, an ETL process, in a data warehouse database, or in the front-end itself. On the performance side the same test entry points are utilized, to focus on characterizing subsystem response under load. A similar strategy to that used for data integrity validation is used for determining the origin of performance issues. In other words, subsystem performance can be measured at any of the identified test entry points. This is especially significant with regard to the test entry point located at the data warehouse component itself. This entry point tests one of the key architectural features of a data warehouse: its optimization of queries. Data warehouses commonly use strategies such as aggregate tables, partitioning and B-tree indexing in order to speed up queries, and the success of the architecture used should be directly verified. The emphasis on rapid localization of either data or performance problems in complex data warehouse architectures provides a key tool for promoting efficiencies in development, and for shortening build cycles and meeting release targets.

Pre-Deployment ETL Layer testing can be summarized as follows:

- Input synthesized feeds (flat files, HL7 feeds, CDISC feeds, other XML grammars) of known data content
- Validate output data (which may be stored in XML, database or other data formats)
- Validate data integrity at post-ETL entry points, e.g. if the processed data is persisted in a database, validate the database contents based on input data.
- Validate all data conditions for all feeds
- ETL layer performance
- ETL subsystem/component performance

Pre-Deployment Data Warehouse and Front End testing can be summarized as follows:

- Validate user queries based on a data warehouse of known data composition
- Validate user queries for all front ends in use
- Performance of data warehouse
- Performance of the front-end applications

Post-Deployment Phase

Strategic Approach

In the post-deployment area, the recommended strategy consists of application monitoring at the client, server and network levels (see Figure 2). The corresponding monitoring points are shown in Figure 2. Front-end client monitoring provides a picture of the client experience from the perspective of application performance. Ongoing monitoring of the client experience forestalls complaints from the user population about perceived application performance glitches. Server monitoring, on the other hand, gives a check on the “internal temperature” of data warehouse servers, including CPU/memory usage, log file analysis, database/SQL statistics, service/process availability, and connection up/down. Monitoring server hardware can pinpoint fundamental issues in the platform architecture. Network analysis provides a diagnostic window on how network communications contribute to application performance. Through the use of network analysis, bandwidth issues can be distinguished from network latency problems. Performance questions can be analyzed down to the level of individual SQL calls and their effects on application performance.

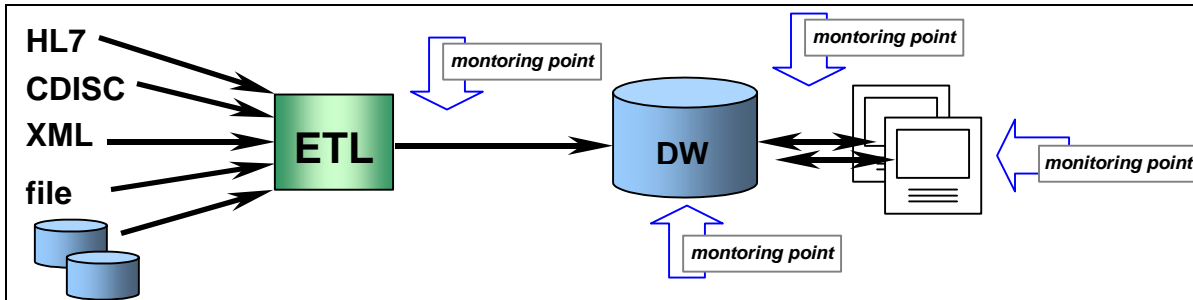


Figure 2. RTTS' Data Warehouse Post-Deployment Testing Strategy

Post-Deployment Monitoring can be summarized as follows:

- Client monitoring – monitor system response behavior at client
- Server monitoring – monitor server response behavior during normal and peak loads
- Network analysis – verify bandwidth requirements, discover application bottlenecks, tune applications for wide area networks

Bibliography

1. Inmon, W. H., Building the Data Warehouse, John Wiley & Sons; 3rd edition, 2002
2. Kimball, R.; Reeves, L.; Ross, M.; Thornthwaite, W., The Data Warehouse Lifecycle Toolkit : Expert Methods for Designing, Developing, and Deploying Data Warehouses, John Wiley & Sons, 1998

About the author:

Jeff Bocarsly, a Vice President and functional testing division manager with RTTS, has successfully implemented automated software testing projects at many Fortune 500 firms. His experience includes project in various sectors including brokerage firms, media, pharmaceutical, software, banking, insurance, and reinsurance companies. He specializes in implementing test automation and methodology solutions for software development groups.

He has recently authored a book that defines the essentials of IBM Rational® XDE™ Tester with RTTS colleague Dan Chirillo and noted software testing author, Tom Arnold, that will be released in 2004. Jeff holds a BS from UCLA, and Masters and PhD degrees from Columbia University.

About RTTS:

RTTS is a professional services organization that specializes in the testing, monitoring, diagnosing and tuning of IT applications and architecture. Serving Fortune 500 and mid-sized companies nationwide, RTTS has offices in New York, Boston, Phoenix and Orlando. RTTS draws on its expertise utilizing best-of breed products, expert test engineers and proven methodology to provide the foremost end-to-end solution that ensures application functionality, reliability, scalability and network performance.

From strategically planning your entire testing approach to tactically implementing the effort in pre-deployment and then through monitoring transactions and diagnosing bottlenecks and other post-deployment problems, RTTS provides a full-cycle iterative solution for your software quality needs.

RTTS offers full outsourcing of your entire testing needs or can provide you with individual test tool product expertise. We offer expert mentoring and education services, along with a proven game plan for providing knowledge and skills transfer.

To learn more about RTTS, visit www.rttswb.com.