Ascential™ Software

DataStage® XE Architectural Overview

I. Executive Summary

In today's information age, organizations have grown beyond the question "Should we build a data warehouse" to "how should we build it and what should we build it with?" Unfortunately once a warehouse has been built, it attracts more users, which in turn increases the demand on the warehouse. More users translate into more requirements in terms of performance, features, and overall support. Very quickly developers find themselves having to incorporate more data sources, targets, and transformations into an existing warehouse application. In reality, the data warehouse project never ends.

How an organization chooses to build the warehouse has an impact on costs in terms of time, money, and the resources required to expand and maintain the warehouse environment. If an organization chooses the wrong tool for their environment they may find that they have to rebuild the warehouse from scratch. Writing custom code is often the simplest approach at first, but quickly exhausts the organization's ability to keep up with demand. Change is very difficult to manage when the transformation rules exist buried in procedural code. Purchasing a full solution from an RDBMS vendor can be an attractive initial choice but it requires a commitment to their database and tool suite.

Ideally, the best approach should allow flexibility in the warehouse architecture so that the organization can easily add other best-of-breed tools over time. The most flexible and cost effective choice over the long term is to equip the warehouse team with a tool powerful enough to do the job as well as allow them to maintain and expand the warehouse.

That choice is DataStage® XE.

With DataStage XE, warehouse administrators and developers will have the flexibility to:

- Provide data quality and reliability for accurate business analysis
- Collect, integrate, and transform data from diverse and sometimes obscure sources with structures ranging from simple to highly complex, for building and maintaining the warehouse infrastructure without overloading available resources
- Integrate meta data across the warehouse environment crucial for maintaining consistent analytic interpretations
- Capitalize on mainframe power for extracting and loading legacy data as well as maximize network performance

Going one step further, Ascential Software has long supported developers in offering a warehouse development environment that:

- Works intuitively thereby reducing the learning curve and maximizing development resources
- Reduces the development cycle by providing hundreds of pre-built transformations as well as encouraging code reuse via APIs
- Helps developers verify their code with a built-in debugger thereby increasing application reliability as well as reducing the amount of time developers spend fixing errors and bugs
- Allows developers to quickly "globalize" their warehouse applications by supporting single byte character sets and multi-byte encoding
- Offers developers the flexibility to develop warehouse applications on one platform and then package and deploy them across the warehouse environment

Overall, building a data storage container for business analysis is not enough in today's fast-changing business climate. Organizations that recognize the true value in maintaining and evolving their transaction and warehouse architecture, along with the attendant complexities of that process, have a better chance of withstanding and perhaps anticipating significant business shifts. Choosing the right tool for the job can make the difference between being first to market and capturing large market share or just struggling to survive.

Table of Contents

II. Introduction

In today's information age, organizations have grown beyond the question "Should we build a data warehouse" to "how should we build it and what should we build it with?" Unfortunately the headlong rush to construct a useful warehouse has resulted in headaches and unforeseen problems and inefficiencies. How were these problems created and better yet, how can they be avoided?

Starting at the beginning, the primary requirement of the warehouse is to provide easy and efficient transfer of usable data ready for analysis, from the transaction applications to the data consumers. Warehouse developers have three options in building the data warehouse:

- Write custom code
- Purchase a packaged solution from an RDBMS vendor
- Equip the warehouse designers with powerful tools

Just choosing a development option without considering the organization's data requirements after the warehouse has been completed is the first pitfall. From there, the problems start to build up because organizations grow, data consumers become more sophisticated, business climates change, and systems and architectures evolve. Before building or updating their data warehouse, developers and managers need to ask:

- How will the development options and warehouse architecture affect overall performance?
- How will the warehouse handle data complexities such as handing and transforming unstructured data?
- How will we correct invalid or inaccurate data going into the warehouse?
- How will we integrate meta data especially when it comes from obscure sources?
- How will we manage the warehouse environment as the organization expands and changes?

Simply building a storage container for the data is not enough. Ultimately, the warehouse should have accurate data that is "analysis ready".

Getting Ready for Analysis

Exactly how does the stored transaction data become suitable for mining, query, and reporting? First, the details need to be consolidated into classifications and associations that turn them into meaningful, accessible information. In addition, since the data coming from online systems is notoriously "dirty" and full of inaccuracies it has to be "cleansed" as it is moved to the warehouse environment. In fact, data quality is the number one reason for data warehouse project failure.

Further, users often use the systems incorrectly. Sometimes they become creative with data when the system doesn't accommodate their needs—e.g. "put the customer's previous address in the last line of the 'comment' field". Sometimes faulty design leads to creation of inscrutable data—e.g. " when calculating taxes, a state code of 'MH' means the employee lives in Massachusetts, but works in New Hampshire". Company mergers and acquisitions bring diverse operational systems together to compound the problem. Data needs to be transformed in order to aggregate detail, expand acronyms, identify redundancies, calculate fields, change data structures, and to impose standard formats for currency, dates, names, and the like. These standard data definitions, or meta data, must accompany the data to the warehouse in order for users to recognize the full value of the information. Realizing the data's full potential means that the users must have a clear understanding of what the data actually means, where it originated, when it was extracted, when it was last refreshed, and how it has been transformed.

Warehouse Pitfalls

Unfortunately once a warehouse has been built, it attracts more users, which in turn increases the demand on the warehouse. More users translate into more requirements in terms of performance, features, and overall support. Very quickly developers find themselves having to incorporate more data sources, targets, and transformations into an existing warehouse application. In reality, the data warehouse project never ends.

How an organization chooses to build the warehouse has an impact on costs in terms of time, money, and the resources required to expand and maintain the warehouse environment. If an organization chooses the wrong tool for their environment they may find that they have to rebuild the warehouse from scratch. Writing custom code is often the simplest approach at first, but quickly exhausts the organization's ability to keep up with demand. Change is very difficult to manage when the transformation rules exist buried in procedural code. Purchasing a full solution from an RDBMS vendor can be an attractive initial choice but it requires a commitment to their database and tool suite.

Ideally, the best approach should allow flexibility in the warehouse architecture so that the organization can easily add other best-of-breed tools over time. The most flexible and cost effective choice over the long-term is to equip the warehouse team with a tool powerful enough to do the job as well as allow them to maintain and expand the warehouse. The most current ETL tools employ engine-based technology to address the issues around ease of use and maintenance.

Not All Tools are Created Equal

The successful power tool for building the warehouse will have a relentless focus on performance, minimize the trade off between integration and flexibility, and have a clearly demonstrable return-on-investment. In order to accomplish these goals, warehouse designers and developers need a tool that maximizes their performance by:

- Handling architecture complexity in terms of integrating distributed and non-distributed systems, synchronization, mainframe and client/server sources
- Providing the choice to harness the power of mainframe systems for 'heavy-duty' extract, transform, load processing
- Dealing effectively with transformation complexity such as structured and unstructured data sources, integrating web data, and changed data
- Providing extensive built-in transformation functions
- Permitting the reuse of existing code through APIs thereby eliminating redundancy and retesting of established business rules
- Offering a full debugging environment for testing and exit points for accessing specialized external or third-party tools
- Bridging the meta data gap between design and business intelligence tools
- Supplying an easy deployment mechanism for completed transformation applications
- Making data quality assurance an inherent part of the data warehouse environment

In addition, the tool must offer high performance for moving data at run-time. It should pick the data up once and put it down once, performing all manipulations in memory and using parallel processing wherever possible to increase speed. Since no tool stands alone, it must have an integration architecture that provides maximum meta data sharing with other tools without imposing additional requirements upon them. Indeed, each tool should be able to retain its own meta data structure yet play cooperatively in the environment. Finally, the tool's financial return must be clearly measurable. Companies using a tool that fulfills these requirements will be able to design and run real-world data warehouses and marts in a fraction of the average time.

Is there a tool today that can fulfill all the requirements? The answer is yes. DataStage XE from Ascential Software offers the flexibility and technology so that developers can both build and maintain complex data warehouses cost-effectively.

III. What is DataStage XE?

Using DataStage XE, warehouse developers can take data from diverse sources and complex data forms such as legacy data, B2B and web environments, as well as enterprise applications such as SAP and Siebel. They can transform this data, load it into a warehouse, data mart or business intelligence application for analysis. By managing the meta data, DataStage XE completely integrates meta data with the most commercially popular data modeling and data access tools. Finally, the quality assurance component enables warehouse administrators to audit, monitor, and manage the quality of the data as the warehouse expands and evolves.

Specifically, DataStage XE is an integrated set of software components consisting of:

- Quality Manager for data quality assurance critical for accurate business analysis
- MetaStage for meta data integration in order to maintain consistent analytic interpretations as well as track changes to the data warehouse
- DataStage for data collection and integration from diverse sources for complete "snapshots" and data movement and transformation for system and end-user productivity
- DataStage XE/390 for extracting legacy data while using the power of the mainframe infrastructure

Quality Manager

As part of DataStage XE, Quality Manager gives development teams and business users the ability to audit, monitor, and certify data quality at key points throughout the data integration lifecycle. Further they can identify a wide range of data quality problems and business rule violations that can inhibit data migration efforts as well as generate data quality metrics for projecting financial returns.

By improving the quality of the data going into DataStage transformations, organizations also improve warehouse performance and the data quality of the resultant target data. The end result is validated data and information for making smart business decisions and a reliable, repeatable and accurate process for making sure information maintains its superior quality over time.

MetaStage

A critical component of DataStage XE is MetaStage, Ascential's solution for meta data management across data warehouse environments. Most data warehouses and marts are created using a wide variety of tools that cannot exchange meta data. As a result, business users are unable to understand and leverage enterprise data because the contextual information, or meta data, required is unavailable or unintelligible. Based on patented technology, MetaStage offers broad support for sharing meta data between third-party data environments. MetaStage uses MetaBrokers™ to ensure the complete exchange of all related meta data, regardless of source type.

MetaStage's hub and spoke architecture allows the sharing of meta data among data modeling, data movement, date quality, and business intelligence products. MetaBrokers translate the form, meaning and context of meta data of meta data from tool to tool—providing a basis for understanding your strategic information environment, including cross-tool impact analysis and data lineage. The MetaHub™ Directory maintains a complete catalog of the organization's meta data, including physical, technical, business and process meta data. Without a well-maintained meta data store, users risk misinterpreting their data analysis resulting in poor business decisions.

DataStage

DataStage is a client/server development tool for building and supporting data migration applications. For companies that require dynamic data access such as information from their e-business or enterprise application processes, Ascential Software offers options such as XML Pack, Enterprise Application Packs, and the MQ Series Plug-in.
On the server side, DataStage has a transformation engine that enables complex processing while providing ease of

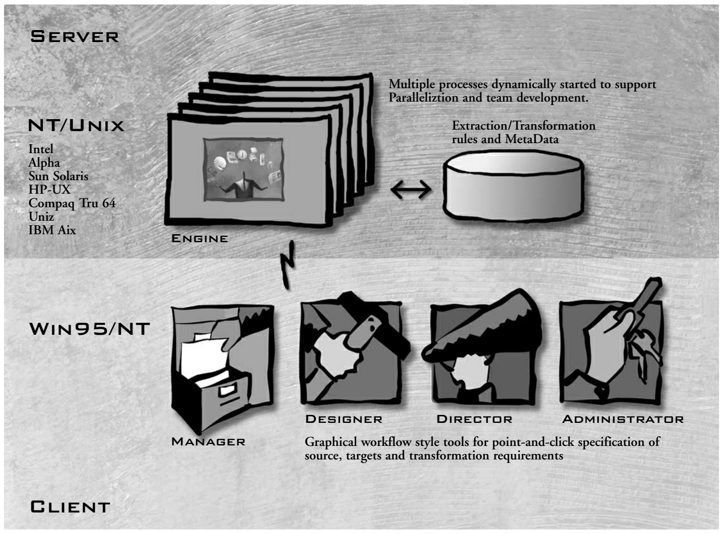
use, management control and maximum performance. The DataStage client is a graphical tool with the following major components: Manager, Designer, Director, and Administrator. The DataStage Manager supports the import/export of meta data, as well as the central control of shared transformation objects. The Designer is the tool that visually represents the data transformation process with an intuitive easy-to-use graphical engine. The Director, as its name implies, supports the scheduling and execution of completed transformations, and the Administrator provides for housekeeping and security functions. Data warehousing professionals use the DataStage client to interact with the DataStage Server, the workhorse that processes the transformations and moves data at run-time.

DataStage XE/390

As an option for the DataStage XE tool suite, DataStage XE/390 extends the suite's capabilities to the industry-leading OS/390 mainframe environment. DataStage XE/390 is the only data integration tool that allows developers to perform all the vital functions necessary to extract, transform, and load data from client-server and OS/390 mainframe data sources using a single toolset interface. Further, it is the only data integration tool on the market today that allows users to determine where to execute jobs—at the source, the target, or both. When used with other DataStage components such as MetaStage and Quality Manager, DataStage XE/390 provides developers with extensive meta data management as well as ensuring that the processed data is reliable and solid for business analysis.

IV. DataStage Components

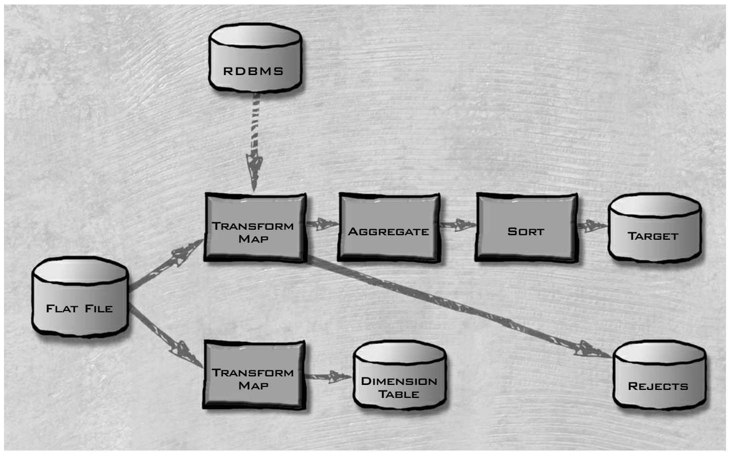
DataStage employs a client/server architecture that supports team development and maximizes the use of hardware resources. Sites can select the best operating platform to meet their needs for transformation processing and throughput. The DataStage client is an integrated set of graphical tools that permit multiple designers to share server resources and design and manage individual transformation mappings. The DataStage client runs on Windows 32-bit platforms and comprises the following four tools: Designer, Manager, Administrator and Director. The Server component of DataStage is the actual data movement workhorse where parallel processes are controlled at runtime to migrate data between multiple disparate sources and targets. The Server can be installed in either NT or UNIX environments and tuned to exploit investments in multiple CPUs and real memory. The Server also stores all the transformation meta data thereby providing a shared resource for the development team.


Server
Multiple processes dynamically started to support Paralleliztion and team development.
NT/Unix
Intel
Alpha
Sun Solaris
HP-UX
Compaq Tru 64
Uniz
IBM Aix
Extraction/Transformation rules and MetaData
Engine
Win95/NT
Designer
Director
Administrator
Manager
Graphical workflow style tools for point-and-click specification of source, targets and transformation requirements
Client

DataStage Architecture

The DataStage Designer

Ascential Software's DataStage was the first ETL tool to pioneer a visual design model for graphically outlining a data migration process. Following a workflow style of thinking, designers select source, target, and process icons and drop them onto a "drafting table" template that appears initially as an empty grid. These icons called "stages" exist for each classic source/target combination consisting of flat files, Oracle, Informix, Sybase, and bulk loader technologies. The Designer connects these representative icons via arrows called "links" that illustrate the flow of data and meta data when execution begins. Further, DataStage uses the graphical metaphor to build table lookups, keyed relationships, sorting and aggregate processing.



DataStage Job Design

This same metaphor is extended in "drill down" fashion to the individual field level, where column-to-column mappings are created visually. The data warehouse developer draws lines that connect individual columns of each source table, row, or answer set definition to their corresponding targets. Developers can specify complex Boolean logic, string manipulation, and arithmetic functions within the Designer by using DataStage's extensive list of built-in functions listed in a drop-down menu. Developers construct transformation expressions in a point-and-click fashion. If the developer needs to use external user-written routines, they too can be selected from the same menu. Filters and value-based constraints that support the splitting of rows to alternative or duplicate destinations are also designed and illustrated using this model.

If desired, a DataStage job can be executed from within the integrated Debugger. The developer starts a process from inside of the Designer tool, setting breakpoints that pause the job at specific logical and field content boundaries. Developers can validate transformations as the results are queried in a pop-up "Watch Window." This feature speeds up the development process by eliminating the need to print or browse tedious trace files and remotely written log structures.

The DataStage Designer helps take the complexity out of long-term maintenance of data migration applications. Team members can share job logic and those who inherit a multi-step transformation job will not be discouraged or delayed by the need to wade through hundreds of lines of complex programming logic.

The DataStage Manager

The DataStage Manager catalogs and organizes the building blocks of every DataStage project. Managed objects include table definitions, centrally coded transformation routines, and meta data connections to other tools in the data warehouse suite. Through Ascential's patented MetaBroker™ technology, DataStage exchanges meta data with the warehouse design tool to obtain initial warehouse and design schemas, then later with the organization's business intelligence tools to make transformation and business design meta data available to end users.

The DataStage Manager gathers column and table information from relational catalogs via standard RDBMS APIs. Further, it examines flat files for their required column/offset combinations and parses COBOL definitions into fields as outlined in their original PICTURE clauses.

The DataStage Manager is also the tool that promotes the sharing of DataStage building blocks among different Server instances. This feature supports not only the deployment of production applications throughout the enterprise but also facilitates sharing transformation logic with Ascential Software's Customer Support Specialists.

The DataStage Administrator

The DataStage Administrator provides controls for the DataStage Server environment. Based on their role in the system, DataStage developers are restricted from seeing particular projects on the server or may be only permitted to execute (not edit or develop) new or existing jobs. The Administrator also controls performance aspects of the Server, such as trace files, inactivity timeout periods, and cleanup/purging of server log resources.

The DataStage Director

The DataStage Director is the interactive tool that allows the starting, stopping, and execution monitoring of DataStage jobs. The status of each job (whether running, aborted, compiled, or not compiled) is immediately shown in the Director's start up window. From the primary status listing, jobs can be started or scheduled for future execution. Jobs are easily executed with only a limited set of rows, allowing developers to test transformation routines without running through an entire data set. In addition, the Director validates a job, certifying that all connections, user-ids, passwords, etc, will be valid when the job is executed in its true production window. Validating a job saves time and reduces frustration during development.

Developers can closely observe the running jobs in the Monitor Window to provide run-time feedback on user-selected intervals. The powerful process viewer estimates rows-per-second and allows developers to pinpoint possible bottlenecks and/or points of failure.

Using the Director, the developer can browse detailed log records as each step of a job completes. These date and time stamped log records include notes reported by the DataStage Server as well as messages returned by the operating environment or source and target database systems. DataStage highlights log records with colored icons (green for informational, yellow are warnings, red for fatal) for easy identification.

The DataStage Director provides a graphical scheduler that, depending on the operating system, interfaces with the NT scheduling service or generates CRON scripts. Associated closely with the Director are the DataStage Job Control API and Command Language. This interface allows any remote C program or command shell to initiate jobs, query their results or program a more complex job execution sequence. The Job Control API and Command Language facilitate the interaction of DataStage with popular enterprise server management and scheduling tools.

The DataStage Server
The DataStage Server runs as an NT Service or UNIX daemon, awaiting requests to either create new processes for job execution or to support the logon of another data warehouse developer. At any one time there may be processes for multiple jobs running alongside processes supporting the team of developers. The DataStage Server is responsible for all interactions between indicated sources and targets, and controls the direct API interfaces to relational databases and remote platforms. The Server also keeps track of all transformation meta data that developers enter, import, control and manage via the DataStage client.

The Server analyzes jobs at compile-time. Where possible, specialized control code tells the DataStage Server to spawn multiple processes in parallel. Increased throughput is the result, as parallel processing helps ensure that job streams are completed as efficiently as possible. In addition the DataStage Server uses instructions from the designer to allocate memory for hashed lookup tables, value resolution and support for complex transformation routines.

DataStage Options
Some warehouse environments have data sources whose structure or source is complex and requires significant processing in order for it to be analyzed by a business intelligence tool. Without this processing, data consumers would not be able to "compare apples to apples." Ascential offers options to DataStage to address specific complex data requirements. These requirements consist of support for XML, ERP data and MQSeries messages. These options extend DataStage’s functionality without adding complexity to the development or the analysis environments.

XML Pack
XML Pack™ provides support for business data exchange using XML. With XML Pack, users can read XML-formatted text and write out data into an XML format, then exchange both business-to-business and application-to-application data and meta data.

Specifically, the Pack’s XML Writer and XML Reader enable companies to leverage their legacy investments for exchanging infrastructures over the Internet without the need for writing specific point-to-point interchange programs and interfaces. Using XML Writer, DataStage can read data from legacy applications, ERP systems, or operational databases, transform or integrate the data with other sources, and then write it out to an XML document. The XML Reader allows the reading or import of XML documents into DataStage. Further, DataStage can integrate data residing in the XML document with other data sources, transform it, and then write it out to XML or any other supported data target.

Users can use XML not only as a data encapsulation format but also as a standard mechanism for describing meta data. By means of an XML Pack Document Type Definition (DTD) for extraction, transformation, and transport (ETT) of meta data, DataStage supports the exporting and importing of all meta data related to the ETT process. When exporting the ETT meta data into an XML document, DataStage supplies the DTD, enabling other tools to understand the structure and contents of the exported document. In addition, XML Pack enables users to export XML meta data from any report, including catalogs, glossaries, and lineage reports, to an XML document. Reports can then be displayed in an XML-capable browser or imported into any other application that supports XML.

Enterprise Application PACKs
For many organizations, enterprise application (EA) systems provide critical data sources for business analysis. DataStage XE provides full integration with leading enterprise applications including SAP, Siebel, and PeopleSoft. The DataStage Extract PACKs for SAP R/3, Siebel and PeopleSoft, and the DataStage Load PACK for SAP BW enable warehouse developers to integrate this data with the organization's other data sources.

MQSeries Stage
Many enterprises are now turning to message-oriented, "near" real-time transactional middleware as a way of moving data between applications and other source and target configurations. The IBM MQSeries family of products is currently one the most popular examples of messaging software because it enables enterprises to integrate business processes.

The DataStage XE MQSeries Stage treats messages as another source or target in the warehouse environment. In other words, this plug-in lets DataStage read from and write to MQSeries message queues. The MQSeries Stage can be used as:

- An intermediary between applications, transforming messages as they are sent between programs
- A conduit for the transmission of legacy data to a message queue
- A message queue reader for transmission to a non-messaging target

Message-based communication is powerful because two interacting programs do not have to be running concurrently as they do when operating in a classic conversational mode. Developers can use DataStage to transform and manipulate message contents. The MQSeries Stage treats messages as rows and columns within the DataStage engine like any other data stream. It can receive rows in standard row and column format, depending on whether it is reading or writing the message queue. As a message writer, the stage writes only datagram messages. As a message reader, the stage accepts all message types. All reads are browse reads, so the message remains in the queue. If the stage reads a "request" type message, the developer must ensure that another application reads and responds to it as a "request" type message.

The MQSeries Stage is the first step in providing real-time data warehousing and business intelligence. It supports multiple input and output links but does not support reference links because message data cannot be guaranteed to be persistent and lookups are not key-based. With the MQSeries Stage, developers can apply all of the benefits of using an ETL tool for data warehousing to application integration.

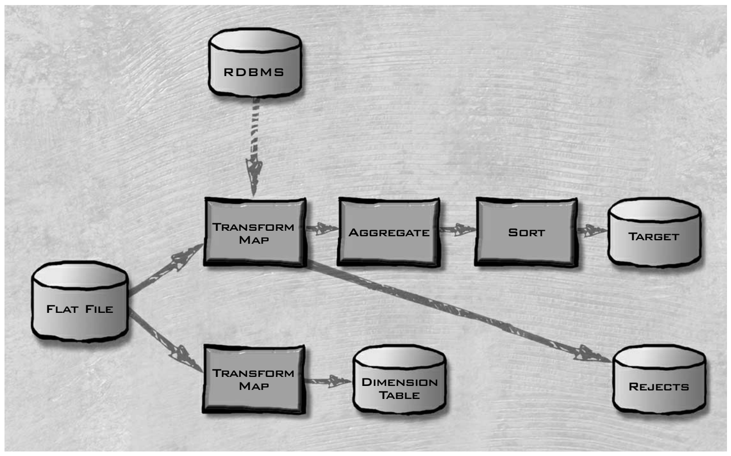
Platforms
Refer to Appendix A for a complete list of supported platforms.

Sources and Targets
Refer to Appendix A for a complete list of supported Source and Target systems.

V. Ascential’s Approach

Ascential’s approach to data warehousing is a reflection of the company’s extensive experience with mission critical applications. Supporting such systems requires an intimate understanding of operating environments, performance techniques, and coding methodologies. Users of these applications demand performance as well as functionality. Extending those qualities to extraction/transformation is the cornerstone of Ascential’s data movement strategy. DataStage XE meets the goal of maximizing throughput without sacrificing control and usability. Every aspect of its transformation engine is optimized to move the maximum number of rows per second through each process.

DataStage jobs, by default, are designed to pick up data just once from one or more sources, manipulate it as necessary, and then write it once to the desired target. Consequently DataStage applications do not require intermediate files or secondary storage locations to perform aggregation or intermediate sorting. Eliminating these steps avoids excessive I/O, which is one of the most common performance bottlenecks. Data does not have to be loaded back into a relational system to be reprocessed via SQL. As a result, DataStage jobs complete in a shorter period of time.



DataStage Job Design

If multiple targets are critical to the application, DataStage will split rows both vertically and horizontally and then send the data through independent paths of logic—each without requiring a temporary staging area. DataStage makes it a priority to do as much as possible in memory, and to accomplish complex transformations in the fewest possible passes of the data. Again, the benefit of this approach is fewer input/output operations. The less disk activity performed by the job, the faster the processes will complete. Target systems can be online and accessible by end users in a shorter period of time. The following sections describe the core concepts within DataStage that support this methodology.

Multiple Sources

DataStage makes it possible to compare and combine rows from multiple, heterogeneous sources. Without writing low-level code, the DataStage developer constructs data flow diagrams that relate multiple input files for full-scale joins or reference lookup purposes. Multiple sources might be flat files, whose values are compared in order to isolate changed data, or heterogeneous tables joined for lookup purposes. DataStage issues the appropriate low-level SQL statements, compares flat files, and handles connectivity details. Most critical for performance are file comparisons that are performed "in stream," and rows of data meeting selected join characteristics that are continuously fed into the rest of the transformation process.

Multiple Targets
DataStage's design supports the writing of multiple target tables. In the same job tables concurrently can:

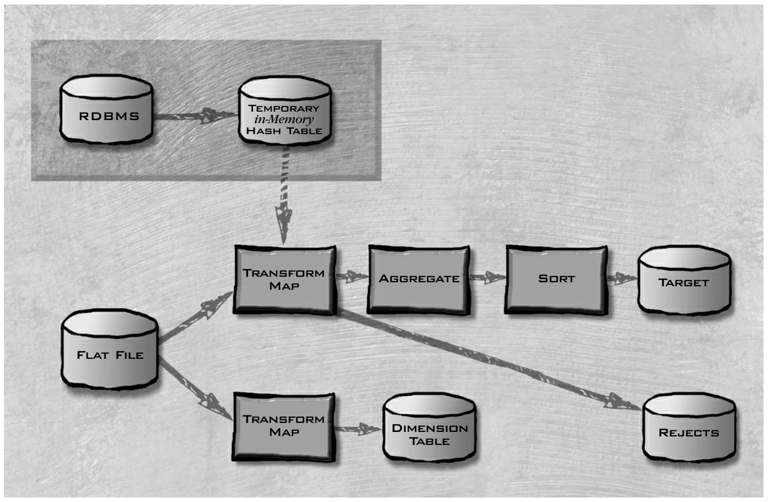
- Be of mixed origin (different RDBMS or file types)
- Have multiple network destinations (copies of tables sent to different geographic regions)
- Receive data through a variety of loading strategies such as bulk loaders, flat file I/O, and direct SQL

By design, rows of data are selectively divided or duplicated when the developer initially diagrams the job flows. DataStage can optionally replicate rows along multiple paths of logic, as well as split rows both horizontally according to column value, and vertically with different columns sent to each target. Further, DataStage can drive separate job flows from the same source; the DataStage engine, from one graphical diagram, can split single rows, while in memory, into multiple paths, or invoke multiple processes in parallel to perform the work. There are no intermediate holding areas required to meet this objective. The resulting benefit is fewer passes of the data, less I/O, and a clear, concise illustration of real-world data flow requirements.

Sorting and Aggregation
DataStage further enhances performance and reduces I/O with its built-in sorting and aggregation capabilities. The Sort and Aggregation stages of DataStage work directly on rows as they pass through the engine, while other tools rely on SQL and intermediate tables to perform these critical set-oriented transforms. DataStage allows the Designer to place sort functionality where needed in the job flow. This placement may be before or after calculations, before or after aggregation, and if necessary, along the independent paths of logic writing to multiple targets. For example, the warehouse application can perform Aggregation immediately following the generation of new key or grouping values and prior to INSERT into the RDBMS. Developers can set properties in the sort stage to fully utilize available real memory. If the data is already in sorted order, the Aggregation stage will deliver rows to the target, as each aggregate group is complete. As long as sufficient resources are available, the Sort and Aggregation stages perform their work in memory. Once again, this approach reduces physical I/O and helps ensure that DataStage jobs complete in the shortest possible timeframe.

Linked Transformations
DataStage performs mapping and transformation operations in memory, as rows pass through the process. The DataStage engine exploits available memory to reduce I/O and provide faster job execution. The DataStage Transformation Stage provides the user with the ability to graphically illustrate column-to-column mappings, to build arithmetic or string manipulation expressions, and to apply built-in or user-written functions. Sometimes it is necessary to incorporate dependent or "linked" transformations into a data migration process. The results of intermediate column calculations are often reused in subsequent statements. The engine processes such linked transformations as regular subroutines, fired in iterative order. No rows are written to disk. Rows flow continuously through one or many transformation stages as DataStage ensures a seamless flow of data and increased overall throughput. In addition, it allows the developer to graphically link transformations, making order of operations crystal clear to anyone who needs to support the job in the future.

In-Memory Hash Tables
Built into every DataStage Server is the ability to automatically create and load dynamic hash tables that dramatically increase the performance of lookups. Software algorithms use lookups for comparing keys and bringing back dimension or fact data such as prices and descriptions, or to validate data such as determining if a key exists within a reserved corporate listing. DataStage developers load hash tables directly from source information and then select the "pre-load to memory" option to gain optimal performance. The hash tables can be loaded directly within the same job as the required lookup or at an earlier time and saved for use within many jobs of the application. Performance gains over disk-intensive operations are significant, especially when there are many different fields requiring comparison. Other tools typically perform lookups by issuing PREPAREd SQL statements back into a relational database, a workable but tedious mechanism.



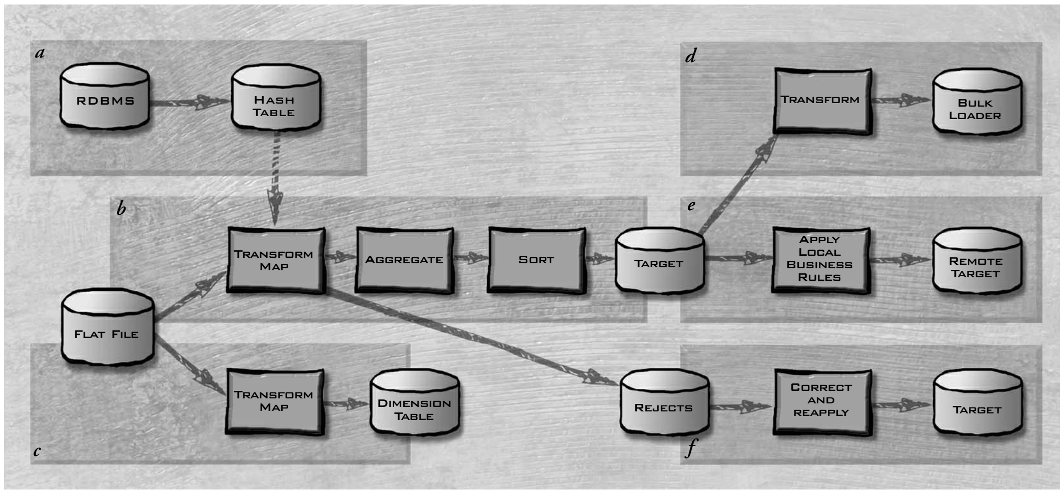
Enhancing Performance with in-Memory Hash Tables

Complex Flat Files
DataStage XE users are often confronted with the need to extract data from flat files with complex internal structures. These legacy file formats, generally designed and constructed using COBOL applications on the mainframe, contain repeating groups (OCCURS) of information, record types, and mixed (character and binary) data types. The DataStage Complex Flat File Stage enables users to easily integrate these file types into their data movement applications. The Complex Flat File Stage first interprets the meta data stored in COBOL copybooks or other forms of COBOL source code. This step provides DataStage XE with the column names and field offsets required for correct interpretation of the data. At run time, the Complex Flat File Stage performs automatic data type conversions; checks record type identifiers at the time of retrieval, and normalizes repeating "arrays" of information. The processing of the Complex Flat File Stage allows developers to concentrate on business rules without having to worry about items such as the transformation of mainframe PACKED fields.

Named Pipe Stage
DataStage's Named Pipes capability makes it easy for developers to transfer data between two or more DataStage jobs or between a DataStage job and an external program. A feature of many operating systems, the named pipes mechanism allows the transparent movement of data through memory between critical processes of an ETL application. Named pipes are powerful but sometimes difficult to enable. With DataStage, developers simply select the named pipe option, and DataStage takes care of any lower level operating system details. Further, DataStage can seamlessly share data via the named pipe support of other tools such as specialized bulk loaders. Large jobs can be split into a number of more manageable jobs, or can be shared among a number of developers on the data warehousing team. At execution, individual job streams run in concurrent but integrated processes, passing rows between one another. Using named pipes optimizes machine resources and reduces design and development time. Also, staff members can easily address separate yet related parts of an extraction transformation problem and avoid platform specific issues surrounding named pipe configurations.

Parallelism
The DataStage Engine exploits multiple CPU environments by automatically distributing independent job flows across multiple processes. This feature ensures the best use of available resources and speeds up overall processing time for the application. The DataStage Server analyzes jobs at the start of execution, highlighting dependent operations and separately flagging independent flows of data. For a given job, the DataStage Engine will potentially spawn many parallel processes at run-time. These processes will run concurrently and have their labor divided among multiple processors

on a machine with multiple CPUs. In the diagram below, six (6) processes are ultimately spawned for this single job. Processes (a) and (c) start immediately; process (b) has to wait until the lookup table is entirely loaded into memory before getting started. Processes (d), (e) and (f) are kicked off simultaneously after the loading of the first table in the star schema has reached completion.



DataStage Invokes Parallel Processes

Native Support for Popular Relational Systems

DataStage reduces network overhead by communicating with popular databases using their own dedicated application programming interfaces. Relational databases are a key component of just about every decision support application. Usually, they are the medium for storing the warehouse or data mart. In an increasing number of scenarios, they are also the source of business information. Interacting with these database systems often requires the use of SQL for retrieving data from source structures or for writing results to a target. Even though bulk loaders typically offer greater performance, an extraction/load routine may need to use dynamic SQL for record sequencing, updates, or other complex row processing. It is critical that tools in the extraction/transformation arena support the most efficient method for communicating with these databases and the vast array of vendor configurations. ODBC is the most flexible vehicle for providing this type of interaction, but may not always provide the best performance with the least amount of overhead. DataStage supports ODBC and provides drivers for many of the popular RDBMS. It also concentrates on delivering direct API access allowing the DataStage developer to bypass ODBC and "talk" natively to the source and target structures using direct calls. Wherever possible, DataStage exploits special performance settings offered by vendor APIs and avoids the management overhead associated with the configuration and support of additional layers of software.

A complete listing of DataStage Stages is contained in Appendix A.

Bulk Loaders

Ascential Software is committed to supporting the high speed loading technologies offered by our partners and other leading vendors in the data warehousing industry. Bulk loaders or "fast load" utilities permit high-speed insertion of rows into a relational table by typically turning off logging and other transactional housekeeping performed in more volatile online applications. DataStage supports a wide variety of such bulk load utilities either by directly calling a vendor's bulk load API or generating the control and matching data file for batch input processing. DataStage developers simply connect a Bulk Load Stage icon to their jobs and then fill in the performance settings that are appropriate for their particular environment. DataStage manages the writing and rewriting of control files, especially when column names or column orders are adjusted. In the case of bulk API support, DataStage takes care of data conversion, buffering, and all the details of issuing low-level calls. The developer gets the benefit of bulk processing without having to incur the learning curve associated with tricky coding techniques.

A complete listing of Bulk Loader Stages is contained in Appendix A.

FTP Stage

Preparing data for loading to a data mart or data warehouse often involves the use of file transfer protocol to move flat files between machines in the corporate network. This process is supported generally by shell programs and other user written scripts and requires that there be enough disk space on the target machine to retain a duplicate copy of the file. While this process works, it does not efficiently use available storage. DataStage provides FTP support in a stage that uses file transfer protocol, but skips the time consuming I/O step needed when executing a stand-alone command program. While blocks of data are sent across the network, DataStage pulls out the pre-mapped rows, moving them directly into the transformation process. Also, DataStage can use its FTP support to write flat files to remote locations. DataStage's FTP Stage provides great savings in time as no extra disk I/O is incurred prior to or after DataStage execution.

Distributed Repository

Corporate executives need reliable, trustworthy, global information about their business. To support this requirement, warehouse managers bear the burden of providing consistent data and meta data that meet corporate standards while maintaining flexibility. In other words, global transformation meta data, model definitions, and business rules all need to be kept in-synch while local company sites need to retain autonomy to develop rules that are region or division specific. DataStage XE provides warehouse developers with a central hub that manages meta data at the tool-integration level. Remote sites can subscribe to a set of meta data objects within the warehouse application. These sites are notified via email when meta data changes occur within their subscription.

Further, DataStage XE offers version control, which saves the history of all the ETL development. It preserves application components such as table definitions, transformation rules, and source/target column mappings within a 2-part numbering scheme. Developers can review older rules and optionally restore entire releases that can then be moved to distributed locations. DataStage allows developers to augment distributed objects locally as they are needed. The objects are moved in a read-only fashion so as to retain their "corporate" identity. Corporate IT staff can review local changes and "version" them at a central location. Version control tracks source code for developers as well as any ASCII files such as external SQL scripts, shell command files and models. Version control also protects local read-only versions of the transformation rules from inadvertent network and machine failures, so one point of failure won't bring down the entire worldwide decision support infrastructure.

Handling Changed Data Capture

Handling changed data is another item critical to the performance of extraction transformation applications. Decision support applications require updating on a periodic basis. The load time required for such an update is determined by variables such as the:

- Frequency of update
- Availability of the target
- Ability to access sources
- The anticipated volume of changed data
- The technical structure of the OLTP database

Network overhead and lengthy load times are unacceptable in volume-intensive environments. DataStage's support for Changed Data Capture minimizes the load times required to refresh your data marts and data warehouses.

Changed Data Capture describes the methodologies and tools required to support the acquisition, modification, and migration of recently entered or updated records. Ascential Software realizes the inherent complexities and pitfalls of Changed Data Capture and has created a multi-tiered strategy to assist developers in building the most effective solution based on logs, triggers, and native replication. There are two significant problems to solve when dealing with changed data:

- The identification of changed records in the operational system
- Applying the changes appropriately to the warehouse

To assist in the capture of changed data, Ascential Software provides tools in DataStage that obtains new rows from the logs and transactional systems of mainframe and popular relational databases such as DB2, IMS, and Oracle. In fact, each version of Changed Data Capture is a unique product that reflects the optimal method of capturing changes given the technical structure of the particular database. The Changed Data Capture stage specifically offers properties for timestamp checking and the resolution of codes that indicate the transaction type such as insert, change, or delete. At the same time, its design goals seek to select methods that leverage the native services of the database architecture, adhere to the database vendor's document formats and APIs as well as minimize the invasive impact on the OLTP system.

Once changed data is isolated and retrieved, it must then be applied to the warehouse. How should this task be accomplished? Consider the situation where an "update" is really a set of new line items inserted into an existing invoice record. Because decision support applications often have data models that look nothing like the original transaction system, this structure can lead to a serious dilemma. How do new line items get applied to a data warehouse that no longer contains invoice detail? The decision support tables may be roll-ups of quarterly revenue by product number. Changed Data Capture captures changes to the operational data and produces Delta Store files. DataStage XE uses these files to update the data warehouse. From a workflow perspective, the warehouse developer defines a Delta Data Store file as an input table within one of the DataStage XE products on a Windows 95/NT platform. DataStage XE generates a program and transfers it to the target platform. This program extracts from the Delta Data store, performs transformations, and then applies the changes to the target data warehouse.

The uniqueness of Ascential's approach lies in the way DataStage XE handles the Delta Data Store. Changed Data Capture generates a Delta Data Store file for each table processed by DataStage XE. It allows warehouse administrators to group related tables together so that all the required Delta Data Store files for a particular warehouse are created easily. Further, Changed Data Capture allows the administrator to specify the extraction criteria, data and control file information, and the method of extracting data (either Automated or On-request). By specifying the extraction criteria, administrators can identify the tables for which changed records need to be extracted and processed into Delta Data Store files.

Further, Changed Data Capture normally operates in Automated mode to provide ongoing capture of the changed operational data on a regularly scheduled basis. Warehouse administrators can schedule this mode to run daily, weekly, or monthly, without any further intervention after it is initiated. Once set up, each successive run automatically captures the changes that have occurred since the last time data was captured. Conversely, On-Request mode provides an alternate one-time capture of the changed data over a specified period of time. Administrators specify the start and end transactions for the capture. Changed Data Capture then captures all changes that have occurred during that period of time. Regardless of the mode, Changed Data Capture will only record changes made by committed transactions. Transactions that are in-flight at run-time and commit later will be captured by the next Changed Data Capture run. In addition, Ascential provides customers with a formal methodology for changed data capture that fully describes the issues surrounding this complex process and suggests how application of changed rows is best accomplished with DataStage. Ascential professionals with field experience in applying these concepts wrote these guidelines. DataStage users interested in Changed Data Capture get an integrated solution along with proven pointers for making it successful. See Appendix A for a list of supported environments for Changed Data Capture.

VI. Commitment to Developers

Ascential Software has a long history of supporting developers, including end-users, value-added resellers, system integrators and specialty consulting firms. DataStage embodies this ideal. Many of Ascential's OEM partners include DataStage technology in their data warehouse solutions suite of products. Others embed it "under the covers" as a transformation engine within their application. DataStage has many features that are designed exclusively to make the data migration process more intuitive, easier to understand and maintain, and faster to deploy. The features described below outline Ascential's commitment to usability and integration with various platforms and third party tools.

Usability

The DataStage Designer was the first tool in the extraction/transformation market to graphically illustrate the actual flow of data from source to target. The graphical palette is the starting place for design and it allows developers to easily diagram the movement of data through their environment at a high level. Designing jobs in this fashion leads to faster completion of the overall application and simpler adoption of logic by other members of the development team thereby providing easier long-term maintenance. Developers drag-and-drop icons representing data sources, data targets, and intermediate processes into the diagram. These intermediate processes include sorting, aggregating, and mapping individual columns. Further, developers can add details such as table names and sort keys via this same graphical drag and drop interface. DataStage pictorially illustrates lookup relationships by connecting primary and foreign keys, and it makes function calls by simply selecting routine names from a drop down menu after clicking the right mouse button. Developers can select user-written routines from a centrally located drop down list. Sites may import pre-existing ActiveX routines or develop their own in the integrated editor/test facility. At run-time DataStage does all of the underlying work required to move rows and columns through the job, turning the graphical images into reality.

The DataStage Director extends ease of use at run-time by providing tools that limit resources and allow validation of jobs to check for subtle syntax and connectivity errors. It immediately notifies developers that passwords have changed rather than discovering such failures the following morning when the batch window has closed. The Director also provides the Monitor window that provides real-time information on job execution. The Monitor displays the current number of rows processed and estimates the number of rows per second. Developers often use the Monitor for determining bottlenecks in the migration process.

Transformation Details
DataStage includes an extensive library of functions and routines for column manipulation and transformation. Developers may apply functions to column mappings directly or use them in combination within more complicated transformations and algorithms. More than 120 granular routines are available for performing everything from sub-string operations through character conversions and complex arithmetic. The DataStage scripting language for transformations is an extended dialect of BASIC that is well suited for data warehousing as it contains a large number of functions dedicated to string manipulation and data type conversion. Further, this language has a proven library of calls and has been supporting not only data migration applications but also on-line transaction processing systems for more than 15 years.

Here is a partial list of granular functions that are available within DataStage:

Abs	ConvertQuarter	Left	Sin
Acos	ConvertTag	Len	Soundex
Asin	ConvertWeek	Ln	Space
Atan	ConvertYear	Locate	Splice
Alpha	Cos	Lower	Squote
Ascii	Count	Maximum	Sqrt
Bitand	Date	Minimum	Substring
Bitnot	Dcount	Mod	Tan
Bitor	Div	Neg	Time
Bitset	Downcase	Not	Timedate
Byte	Dquote	Num	TimeStamp
Bytelen	Ebcdic	Quote	Trim
Byteval	ExecDos	QuarterTag	Upcase
Change	ExecSH	Pwr	Unichar
Char	Exp	Real	Uniseq
Checksum	Findstr	Remove	WeekTag
Compare	Insert	Replace	Xtd
Convert	Int	Rnd	
ConvertMonth	Isnull	Right	

In addition, DataStage has 200 more complex transformations that cover functional areas such as data type conversions, data manipulation, string functions, utilities, row processing, and measure conversions, including distance, time, and weight.

After applying functions and routines to desired column mappings, the data warehouse developer clicks on the DataStage "compile" icon. As noted earlier, this feature first analyzes the job and then determines how to best manage data flows. Next, the compiler creates object code for each of the individual transformation mappings and uses the most efficient method possible to manipulate bytes of data as they move to their target destinations.

It is important to note that developers can reuse transformation code developed in DataStage within the same project and by other DataStage Servers. Further, they can easily share routines through various mechanisms from formal export to integration with DataStage's central warehouse control facility.

Special Merge Stage
DataStage's Merge Stage allows developers to perform complex matching of flat files without having to first load data into relational tables or use cryptic SQL syntax. The Merge Stage supports the comparison of flat files with the option of selecting up to seven different Boolean set matching combinations. More versatile than most SQL joins, the Merge Stage allows developers to easily specify the intersection of keys between the files, as well as the resulting rows they wish to have processed by the DataStage job. Imagine comparing an aging product listing to the results of annual revenue. The Merge Stage lets us ask questions such as "which products sold in what territories and in what quantities,"

and "which products have not sold in any territory?" The Merge Stage performs a comparison of flat files, which is often the only way to flag changed or newly inserted records, to answer questions such as "Which keys are present in today's log that were not present in yesterday's log?" When flat files are the common sources, the Merge Stage provides processing capability without having to incur the extra I/O of first loading the structures into your relational system.

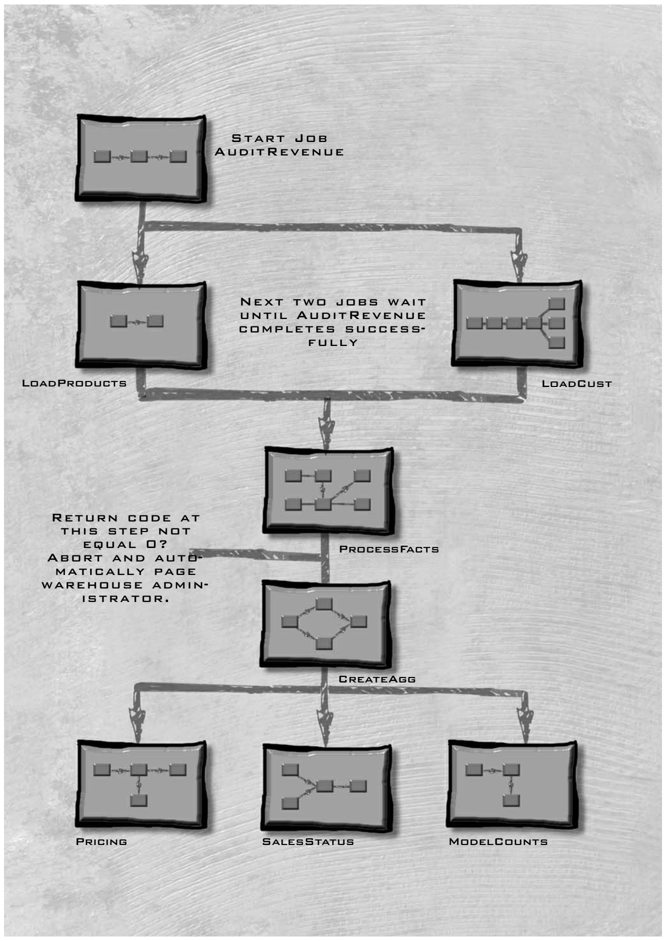
The DataStage Debugger

Fully integrated into the DataStage Designer is the DataStage Debugger. The Debugger is a testing facility that increases productivity by allowing data warehouse developers to view their transformation logic during execution. On a row-by-row basis, developers set break points and actually watch columns of data as they flow through the job. DataStage immediately detects and corrects errors in logic or unexpected legacy data values. This error detection enhances develop productivity especially when working out date conversions or when debugging a complex transformation such as rows that belong to a specific cost center or why certain dimensions in the star schema are coming up blank or zero. Laboring through extensive report output or cryptic log files is unnecessary because the DataStage Debugger follows the ease-of-use metaphor of the DataStage Designer by graphically identifying user selected break points where logic can be verified.

Operating System Integration

A successful data migration application usually consists of more than just one job. In the typical star schema, there may in fact be 10, 20, or even more. It is very important that any ETL tool be able to tie each instance of transformation activity together into a cohesive application. The application needs to have a single point of entry, be able to "container" different jobs into function and operational process groups, and interact with the operating system. Proper job management makes administration easier not only for the data warehouse developer, but also for the systems personnel who will be supporting the application when it is released into production. The tool must be able to pass return codes back to system shell scripts and be able to invoke or be invoked by other tools and utilities.

An important feature in DataStage is the Job Control API and Command Language. The Command Language allows any shell program or interactive user to control the DataStage Server. The language includes commands to start, monitor, and stop jobs, query the server for the results of a previously run job, and to force a wait status on any particular job. Control of job sequencing is important because it supports the construction of a job hierarchy complete with inter-process dependencies. For instance in one particular data warehousing application, "AuditRevenue" may be the first job executed. No other jobs can proceed until (and unless) it completes with a zero return code. When "AuditRevenue" finishes, the Command Language signals DataStage to run two different dimension jobs simultaneously. A fourth job, "ProcessFacts" waits on the two dimension jobs, as it uses the resulting tables in its own transformations. These jobs would still function correctly if run in a linear fashion, but the overall run time of the application would be much longer. The two dimension jobs run concurrently and in their own processes. On a server box configured with enough resources they will run on separate CPUs. This sequence is only the start of a more complicated sequence of jobs as shown in the figure:


Start Job AuditRevenue
Next two jobs wait until AuditRevenue completes successfully
LoadProducts
LoadCust
Return code at this step not equal 0? Abort and automatically page warehouse administrator.
ProcessFacts
CreateAgg
Pricing
SalesStatus
ModelCounts

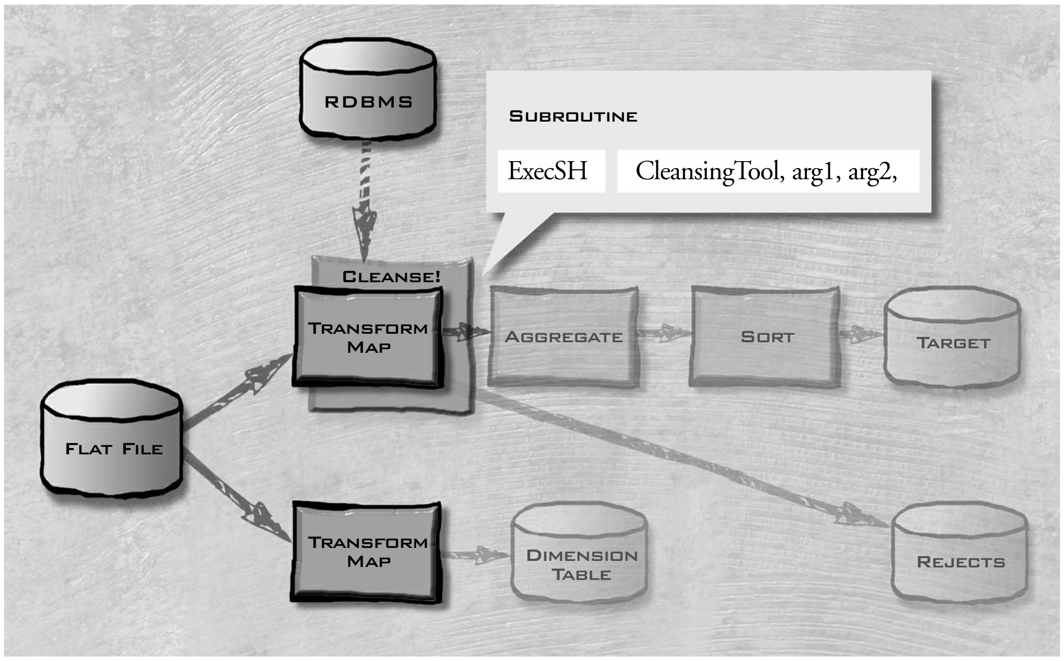
Complex Job Control

A hierarchy of jobs in DataStage is referred to as a "batch" and can not only execute other jobs but can also invoke other batches. DataStage provides applications with a single point of control that supports the integration of multiple subject area transformation mappings.

Another significant benefit of this feature is the ability to have any other tool invoke a DataStage application. Most industry-standard scheduling and server management tools can exploit this feature and issue commands for the DataStage server. Warehouse developers can use the Job Control API and Command language from shells and other operating system command files or from any C program via API calls.

Invoking Specialized Tools

Throughout a job, the DataStage developer can define specific exit points. These exit points are used to make calls to the operating system or to invoke existing code and utilities. Powerful cleansing tools such as FirstLogic and Trillium are often called in this manner. These cleansing tools perform data clean-up operations, such as address parsing and company/product name pattern matching. The developer drags and drops the Stage icons that contain the definition of these exit points onto the Designer canvas. DataStage provides unlimited flexibility, as there is essentially no limit to the number of exit points that a user might include in a particular job. Some of the benefits of calling external programs are the ability to perform clean up operations, issue an email message, or send a page based on some event within the application.



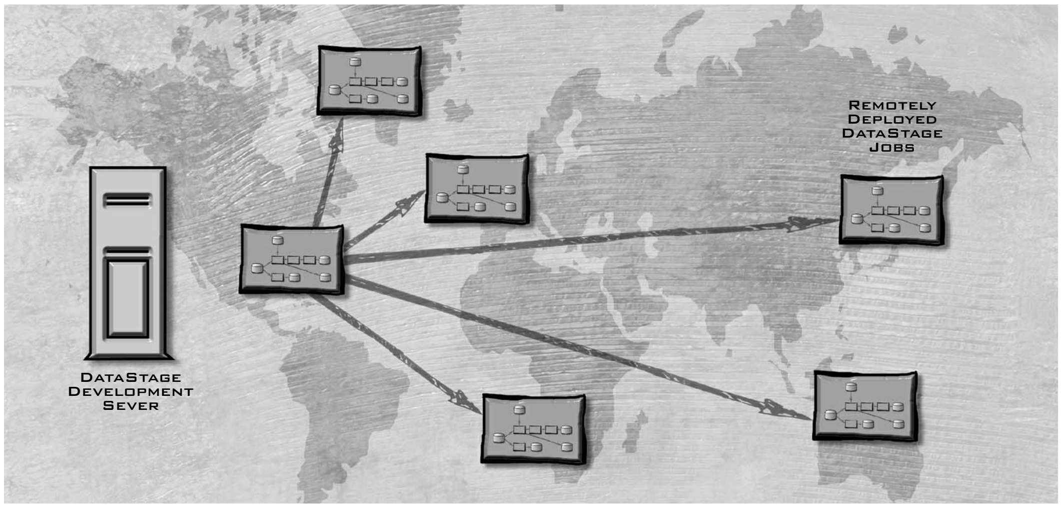
Invoking Specialized Tools

The Plug-In API

Within DataStage, each Stage icon contains a dialog box that has been designed exclusively for the technology addressed by the Stage. For example when a designer clicks on the Sort stage, the software requests a list of properties for the developer to define such as sort keys, order, and performance characteristics. These properties appear as the result of DataStage calling a lower level program that is written using the DataStage Plug-In API. Well-documented and fully supported, this API gives subject and technical matter experts the facility for creating their own extensions to the product. Building custom Plug-Ins provides the opportunity for integrators, resellers, and large MIS shops to extend DataStage for their own proprietary or heavily invested interfaces. Since Plug-In Stages are not dependent on any one release of the DataStage engine, developers can download them from the web and install them separately. Ascential Software frequently develops new Stages and distributes them to existing customers in this fashion.

Job Parameters
Built into DataStage are several features exclusively designed to support the packaging and deployment of completed data migration applications. The DataStage export feature makes it possible to develop extract transformation jobs on the most appropriate or most accessible platform. Remote sites can develop transformation logic on a laptop and then easily deploy the jobs on either NT or UNIX. DataStage's deployment features provide flexibility in selecting the development platform and make it possible to create transformation templates that can be shared throughout the company. An invaluable component of job packaging is DataStage's support of Job Parameters. Job Parameters allow variable string replacement anywhere in the transformation process. Developers can supply pathnames, user-ids, passwords, WHERE clauses and even arguments to a subroutine at run-time when Job Parameter values are populated by the DataStage engine. Also, DataStage gives developers the option of supplying Job Parameters on the command line, through prompts given by the DataStage Director, or by giving site level defaults. DataStage's Release mechanism extends deployment functions by "freezing" a read-only version of the job and all of its components. The "released" version of a job is then formally packaged by simply clicking on the DataStage Packager. An industry standard installation script "wraps " DataStage "Packages" for easier installation and automated configuration. Industry standard packaging makes it simpler to deploy data warehousing jobs to remote sites where administrative personnel may not be as familiar with the environment.

Ascential also offers a special licensing and deployment option that provides additional flexibility to VARs, integrators and large IS shops that wish to distribute finished applications to remote sites. DataStage's run-time option leaves out



Deploying Production DataStage Applications

the development tools but includes the DataStage Server and the Director for job scheduling and execution. Independent software vendors make extensive use of the packaging and deployment features when embedding DataStage in their applications.

National Language Support
For the actual transformation and movement of data, DataStage supports not only the standard single byte character sets but also allows for the use of multi-byte encoding. In addition to the complex string handling capabilities that come with multi-byte functionality, (e.g., a string containing six Japanese characters will have a length of 12 bytes), DataStage also supports the definition and use of national locales and conventions which encompass such varied attributes as:

- National currency rules (e.g. using appropriate currency symbols)
- Numeric representation (e.g. 1.000.000,00 for a decimal number)

- Date and time display formats (e.g. DD/MM/YY or YYYY-MM-DD)
- Collating sequences (e.g. does "ç" get sorted before or after "c", or "na" get sorted after "sa" in Japanese)

Should a set of multi-byte characters need to be extended or modified, DataStage provides support for customization. The definition of a new character set (be it single or multi-byte) can be as simple as modifying an existing map or can include entering character sorting weights or display exceptions.

DataStage allows Project, Job, Stage, and Column levels to control character set mapping so that users can customize mapping to their exact requirements. For example, column or field input may come in as a variant of EBCDIC whereas the output can be in the standard Taiwanese character set BIG-5. In fact, in some implementations DataStage has been used as a conversion engine to simply translate files from one character set to another.

In addition to run-time character conversion, DataStage supports localization for Japanese and can be extended to support other languages. Localization provides a GUI interface in a local language for easier understanding and increased market penetration in non-English speaking locales.

National Language Support makes DataStage a logical choice for worldwide enterprises. It supports the extraction, transformation, and loading of data across the globe, dealing easily with specialized data movement requirements and the use of local languages throughout the world.

VII. Mainframe Integration

DataStage XE/390 provides warehouse developers with the option of harnessing the power and the investment made in their mainframe systems. Using the same DataStage user interface, developers can generate native COBOL and JCL programs to execute ETL tasks in the mainframe environment. The end result is an efficient use of the network environment and reduced data traffic because only the "transformed" data is loaded into the warehouse.

DataStage XE/390 provides data warehouse developers with a capability unmatched by competitive offerings. No other tool provides the same quick learning curve and common interface to build ETL processes that will run on both the Mainframe and the Server.

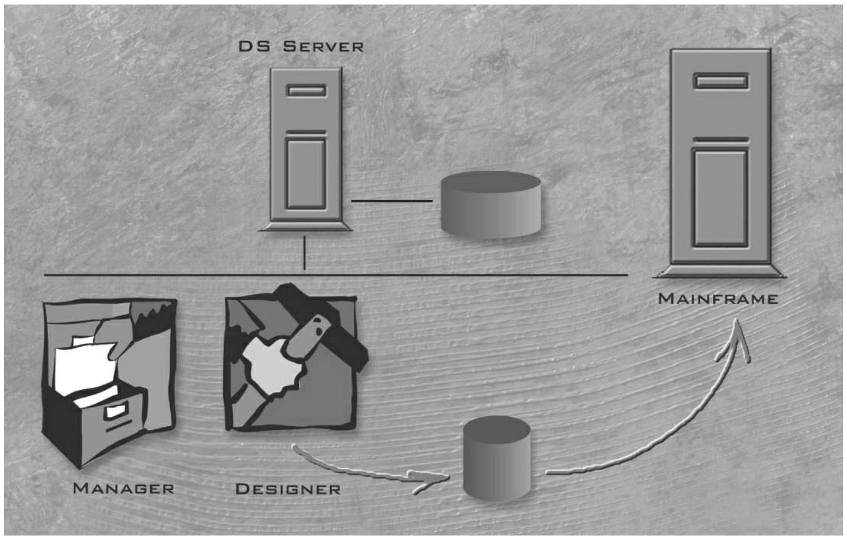
Once the job design is completed, DataStage XE/390 will generate three files that are moved onto the Mainframe, a COBOL program, and two JCL jobs. One JCL job compiles the program and the other runs the COBOL program.

Typically, an organization using mainframes will follow one of the two deployment philosophies:

- Both their transaction system and warehouse will be on the mainframe or
- Minimum processing is done on their mainframe based transactional systems and the warehouse is deployed on a Unix or NT server.

DataStage XE/390 excels in both instances. In the first instance, all processing is kept on the mainframe while ETL processes are created using the DataStage GUI to implement all the complex transformation logic needed.

In the second instance, DataStage XE/390 can be used to select only the rows and columns of interest. These rows can have complex transformation logic applied to them before being sent via FTP to the UNIX or NT server running DataStage. Once the file has landed, the DataStage XE server will continue to process and build or load the warehouse as required.



DataStage XE/390 Architecture

VIII. Data Quality Assurance

Often, data quality is confused with data cleansing. Data cleansing consists of data clean-up operations such as address parsing and company/product name pattern matching. The success of the data warehouse relies directly on the quality of the loaded data. Ascential's Quality Manager offers a structured, repeatable methodology for data quality analysis. Specifically, Quality Manager provides a framework for developing a self-contained and reusable Project which consists of business rules, analysis results, measurements, history and reports about a particular source or target environment. This Project becomes the dashboard for warehouse developers and administrators to improve data quality and measure those improvements over time.

First, through built-in automation features the developers can quickly discover the values within the data and can answer the question: Is the data complete and valid? From meta data derived in this initial discovery process, the developers can then move to build simple or sophisticated tests known as Filters against specific combination of data fields and sources to ensure that data complies with business rules and database constraints. They can run a series of Filters as a job stream or using the Macro feature, can also schedule the job stream within or outside Quality Manager. Quality Manager stores the incidents and history of defective data and allows developers to export this information to other tools, send it using email or post it on intranet sites. The automated meta data facility reviews and updates meta data information for further analysis and for generating DataStage transformations. Throughout, Quality Manager offers Metric capabilities to measure, summarize, represent and display the business cost of data defects. The developer can produce in a variety of formats including HTML and XML reports, analysis charts for showing the relative impact and cost of data quality problems and trend charts that illustrate the trend in data quality improvement over time – important for on-going management of the data environment.

Depending on the data environment, ensuring data quality can range from the very simple to the very complex. For example, companies dealing with global data need to be localization-ready. Quality Manager enables data quality analysis of double-byte data. Others require production mode processing. Quality Manager allows users to run a server version of the data quality SQL-based analysis in a monitoring environment. Further, Quality Manager works with complimentary products such as Trillium and FirstLogic for name and address scrubbing.

IX. Meta Data Management

Tool Integration

In order to support the enterprise-level data warehouse, software tool vendors need to collaborate with upstream and downstream partners as well as integrate the tools within their own product suites. However, meta data—the key to information sharing—exists in a patchwork of repositories and proprietary meta data stores. DataStage XE integrates meta data with other tools using MetaBroker products. MetaBrokers use patented technology to integrate the meta data among disparate tools in a way that eliminates many of the common barriers and problems associated with tool integration today. MetaBrokers provide a way to share meta data among software tools that:

- Minimizes the impact on tools, especially their data structures
- Manages the impact of change within shared environments
- Eliminates meta data loss during sharing
- Maximizes the meta data that is available to each tool
- Retains and uses information about the shared environment itself

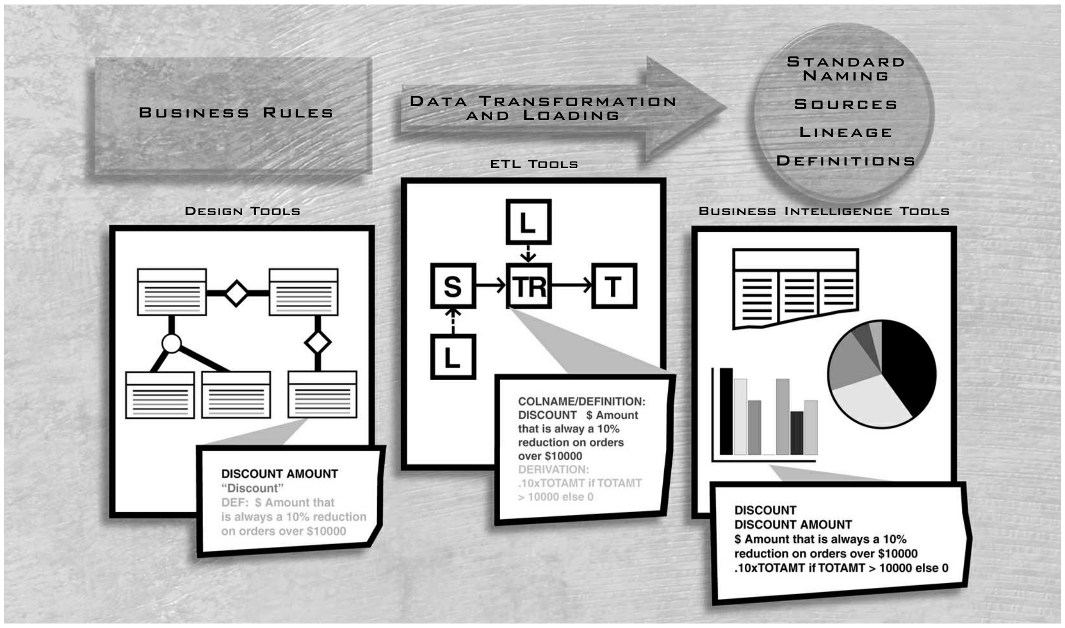
Currently, warehouse developers employ two common methods to integrate meta data. The so-called "quickest" and "easiest" method is to write a bridge from one tool to another. In using this approach, a programmer must understand the schema of the source tool and the schema of the target tool before writing code to extract the meta data from the source tool’s format, transform it and put it into the target tool’s format. If the developer wants the exchange to be bi-directional then the process must be done in reverse. Unfortunately, this approach breaks down as the number of integrated tools increases. Two bridges support the exchange between two tools. It takes six to integrate three tools, twelve to integrate four tools, and so forth. If one of the four integrated tools changes its schema then the developer must change six integration bridges to maintain the integration. Writing and maintaining tool-to-tool bridges is an expensive and resource-consuming proposition over time for any IS organization or software vendor.

Another integration method is the common meta data model, which generally takes one of two forms. The first form, characterized as a "least common denominator" model, contains a set of meta data that a group of vendors agree to use for exchange. However, this exchange is limited to those items described in the common model. The more vendors involved in the agreement, the smaller the agreed set tends to become. Alternatively, a vendor can establish their own meta data model as "the standard" and require all other vendors to support that model in order to achieve integration. This method benefits the host vendor’s product, but typically doesn’t comprise all the meta data that is available to be shared within tool suites. Sometimes a common model will have extension capabilities to overcome the shortcomings of this approach. But, the extensions tend to be private or tool-specific. They are either not available to be shared, or shared via a custom interface to access this custom portion of the model, i.e. by writing custom bridges to accomplish the sharing.

With MetaBrokers, Ascential takes a different approach to meta data sharing. The core components of the technology are the MetaHub, a patented transformation process, and the MetaBroker production process. The MetaHub is the integration schema and is vendor-neutral in its approach. It is composed of granular, atomic representations of the meta data concepts that comprise the domain of software development. As such, it represents a union of all of the meta data across tools, not a least common denominator. To use an analogy from the science of chemistry, it could be considered a "periodic table of elements" for software development. It is this type of model that allows all the meta data of all tools to be expressed and shared.

Software tools express their meta data not at this kind of granular level, but at a more "molecular" level. They "speak" in terms of tables, columns, entities, and relationships, just as we refer to one of our common physical molecules as salt, not as NaCL. In an ideal integration situation the tools do not need to change their meta data representation in any way, nor be cognizant of the underlying integration model. The MetaBroker transformation process creates this result. The MetaBroker for a particular tool represents the meta data just as it is expressed in the tool's schema. It accomplishes the exchange of meta data between tools by automatically decomposing the meta data concepts of one tool into their atomic elements via the MetaHub and recomposing those elements to represent the meta data concepts from the perspective of the receiving tool. In this way all meta data and their relationships in the integrated suite are captured and retained for use by any of the tools.

Ascential's MetaBroker production process is a model-based approach that automatically generates the translation process run-time code – a translation engine (.dll) that converts tool objects into MetaHub objects and vice versa for the purpose of sharing meta data. It uses a point-and-click graphical interface to create granular semantic maps from the meta data model of a tool to the MetaHub. This powerful process allows for easy capture, retention, and updating of the meta data rules that are encoded in the MetaBroker thus providing huge timesavings in creating and updating data sharing solutions. The current set of MetaBrokers facilitates meta data exchange between DataStage and popular data modeling and business intelligence tools. See Appendix A for a listing of specific MetaBrokers provided by Ascential Software. Further, MetaStage's Custom MetaBroker Development facility allows the Ascential Services organization to deliver customer-specified MetaBrokers quickly and reliably.

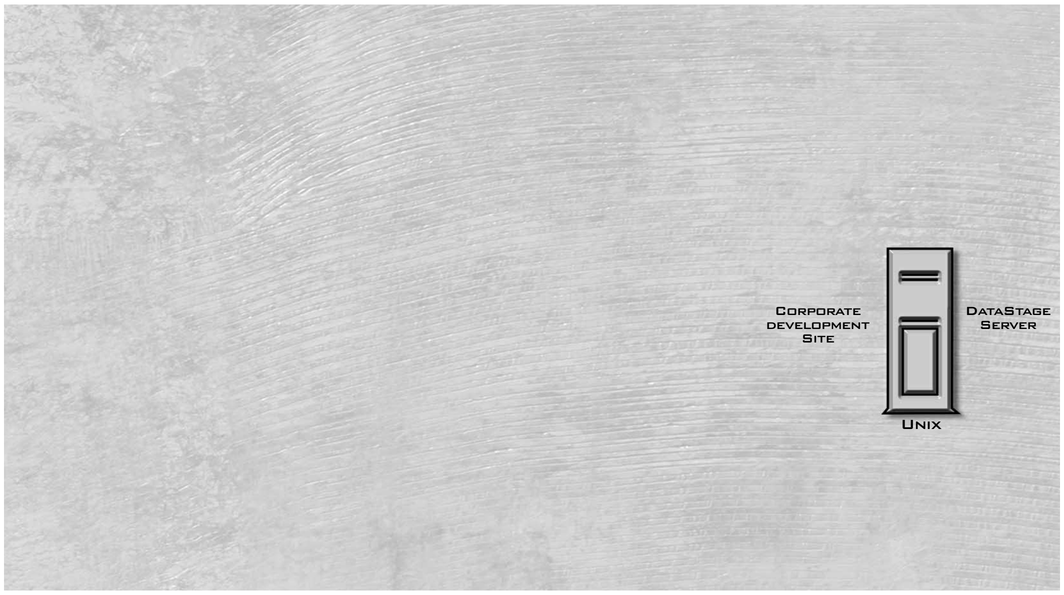

Business Rules
Data Transformation and Loading
Standard Naming
Sources
Lineage
Definitions
ETL Tools
Design Tools
Business Intelligence Tools
L
S
TR
T
L
DISCOUNT AMOUNT
"Discount"
DEF: $ Amount that is always a 10% reduction on orders over $10000
COLNAME/DEFINITION:
DISCOUNT $ Amount that is alway a 10% reduction on orders over $10000
DERIVATION:
.10xTOTAMT if TOTAMT > 10000 else 0
DISCOUNT
DISCOUNT AMOUNT
$ Amount that is always a 10% reduction on orders over $10000
.10xTOTAMT if TOTAMT > 10000 else 0

MetaBrokers™ for Meta Data Sharing

Central Control and Monitoring

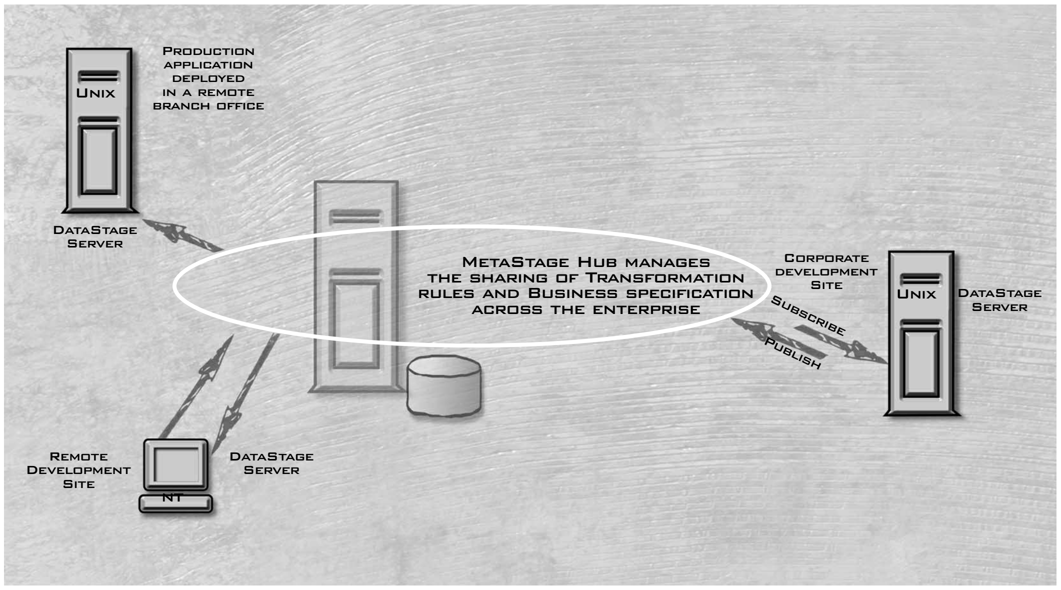
Control and management are critical to the successful deployment of an extraction/transformation tool throughout an organization. Regardless of the overall approach taken to data warehousing (centralized sources of extracted operational data, distributed data marts, or combinations of these methodologies), common business definitions, column relationships, and transformation routines must be shared across all divisions within the corporation. DataStage XE combines powerful meta data sharing capabilities with event management, process control, and analytic reporting to boost productivity and prevent the proliferation of random islands of decision support architectures.

Consider how tools such as DataStage are deployed throughout an organization. One group is successful in deploying a prototype or initial application.



Initial Implementation

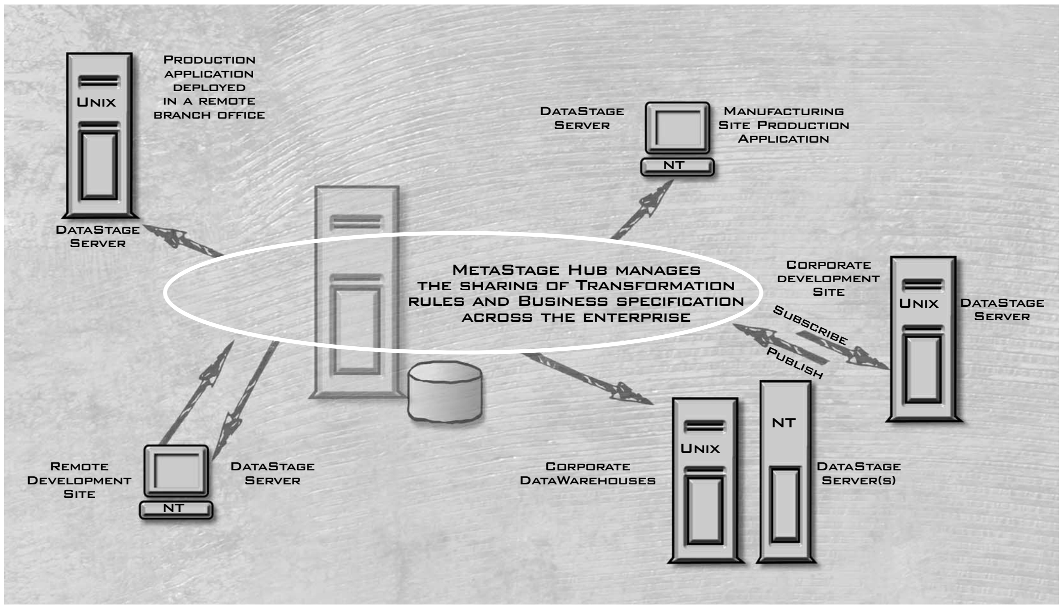
This success influences other units to move in the same direction. The group adds additional development sites and then the application is formally delivered to multiple remote offices. Within a short time, many divisions of the corporation are experiencing the benefits of a well-managed decision support system.


Unix
Production application deployed in a remote branch office
DataStage Server
MetaStage Hub manages the sharing of Transformation rules and Business specification across the enterprise
Corporate development Site
Unix
DataStage Server
Subscribe
Publish
Remote Development Site
NT
DataStage Server

Success leads to more implementation.

Achieving widespread success is positive, but will be short-lived without the formal control and infrastructure offered by MetaStage. MetaStage allows the corporation to reap the benefits of targeted problem solving without losing control of business definitions, algorithms and established standards.

Via MetaStage, DataStage XE developers are able to "publish" centrally controlled pieces of the transformation application, such as table definitions, routines, and customized functions. These pieces are then pushed to the remote DataStage sites that receive them into their own native environment.


Production application deployed in a remote branch office
Unix
DataStage Server
DataStage Server
NT
Manufacturing Site Production Application
MetaStage Hub manages the sharing of Transformation rules and Business specification across the enterprise
Corporate development Site
Unix
DataStage Server
Subscribe
Publish
Remote Development Site
NT
DataStage Server
Corporate DataWarehouses
Unix
NT
DataStage Server(s)

Enterprise Scalability

Local DataStage Servers exploit the best platform for performance or adhere to administrative policy and still get the benefit of shared logic and remain under control of a centralized administrator. MetaStage uses this publish and subscribe process to move standard, published definitions among the various tools in the environment. Data modeling, ETL, and business intelligence tools can share meta data seamlessly after it is brought into MetaStage. MetaStage uses email to automatically notify subscribers of changes occur to any of the published objects. This flexible layer of management encourages definition reuse while preventing accidental overwriting or erasure of critical business logic.

Impact Analysis

Within the MetaStage hub, developers can traverse all the relationships associated with an object represented in any of the MetaBroker models. This feature makes it possible to assess the impact of changes across the integrated environment even before a change occurs. For example, let us look at the effect of a change to an existing table, one that is used as the schema for multiple warehouses and marts. The warehouse designer will ALTER the Sales-Person TABLE to add a new column, Territory_Key, and this column will be defined as NOT NULL. Because it will be defined as NOT NULL, the SQL statements in all the DataStage jobs that create this table will fail if they don't incorporate the change. Impact analysis reporting allows the designer to quickly navigate from the table to the stages within the affected jobs to assess the impact of this change and proceed with the sized maintenance effort. Impact analysis shows its power again when the developer changes the definition of standard corporate calculations such as profit or cost-of-goods-sold. Starting from the transformation routine definition, the designer can quickly identify which columns within stage within jobs use the routine thereby allowing developers to implement the change quickly.

Data Lineage

MetaStage integrates Data Lineage reporting with the same underlying mechanism as Impact Analysis but to answer different questions. Data lineage pertains to events that occur against data. When a DataStage job runs, it reads and writes data from source systems to target data structures and sends data about these events to MetaStage automatically. Within MetaStage, event meta data connects with its job design meta data. Data Lineage reporting answers questions about the warehouse production processes. What tables are being read from or written to, and from what databases on whose servers? What directory was used to record the reject file from last night's run? Or, exactly which version of the FTP source was used for the load? Data lineage reporting allows managers to have an overall view of the production environment from which they can optimize the creation of warehouses and data marts.

Meta Data Reporting and Analysis

Superior, powerful meta data reporting and analysis is a logical by-product of MetaStage's architecture. The MetaBroker production process retains a tremendous amount of knowledge about the shared environment. The data about how tools are mapped to the MetaHub (and therefore indirectly to each other) is a rich source of meta data that allows Ascential to provide advanced meta data management and reporting capabilities like Impact Analysis and Data Lineage.

Equally important to shared data assets is the ability to monitor data warehousing activity throughout the company. Remote DataStage servers transparently report their operational results back to MetaStage. From one single point of control, decision support managers can review individual transformation jobs and the tables they are reading and writing. This information provides advance notification of bottlenecks and errors in processing. As a result, organizations avoid resource conflicts and waste fewer hours while trying to contact field personnel to research problems or facilitate status reporting.

X. Conclusion

Rushing headlong to build a useful data warehouse has caused many warehouse developers and administrators to deal with unanticipated problems and headaches. Unfortunately a successful warehouse project will attract more users who will add and change data requirements. More important, the business climate will change and systems and architectures will evolve leaving developers to figure out how they are going to "make it all work" with the resources they have.

Warehouse developers who consider key factors such as performance, network traffic, structured and unstructured data, data accuracy, integrating meta data, and overall warehouse maintenance will stay one step ahead. But, is there a warehouse tool that can help them deal with these factors without creating long learning curves or diminishing productivity?

DataStage XE can.

With DataStage XE, warehouse administrators and developers will have the flexibility to:
- Provide data quality and reliability for accurate business analysis
- Collect, integrate, and transform complex and simple data from diverse and sometimes obscure sources for building and maintaining the warehouse infrastructure without overloading available resources
- Integrate meta data across the warehouse environment crucial for maintaining consistent analytic interpretations
- Capitalize on mainframe power for extracting and loading legacy data as well as maximize network performance

Going one step further, Ascential Software has long supported developers in offering a warehouse development environment that:
- Works intuitively thereby reducing the learning curve and maximizing development resources
- Reduces the development cycle by providing hundreds of pre-built transformations as well as encouraging code reuse via APIs
- Helps developers verify their code with a built-in debugger thereby increasing application reliability as well as reducing the amount of time developers spend fixing errors and bugs
- Allows developers to quickly "globalize" their warehouse applications by supporting single byte character sets and multi-byte encoding
- Offers developers the flexibility to develop warehouse applications on one platform and then package and deploy them across the warehouse environment

Overall, building a data storage container for business analysis is not enough in today's fast-changing business climate. Organizations that recognize both the true value and complexities involved in maintaining and evolving their transaction and warehouse architecture have a better chance of withstanding and perhaps anticipating significant business shifts. Choosing the right tool for the job can make the difference between being first to market and capturing a large market share or just struggling to survive.

XI. Appendix A

Platforms

Client

Windows 32-bit platforms

Server

Windows NT (Intel and Alpha platforms)

Unix AIX

HP-UX

Sun Solaris

COMPAQ Tru64

Stages

Direct SQL

Oracle OCI (for Oracle releases 7 and 8)

Sybase Open Client

Informix CLI

OLE/DB for Microsoft SQL Server 7

ODBC

Bulk Loaders

Oracle

Informix ADO/XPO High Performance

Loader

Sybase Adaptive Server

Sybase Adaptive Server IQ

Microsoft SQL Server 7 via OLE/DB

Microsoft SQL Server 6.5 via BCP

Informix Redbrick

UDB

Ascential Databases

UniVerse

Unidata

Process

Aggregation

Sort

Merge

Additional Options

Changed Data CaptureFTP

Named Pipes

MetaBrokers

Design Tools

Design Tools

CA-ERwin

Embarcadero ER/Studio

Oracle Designer

Sybase PowerDesigner

Extraction/Transformation/Loading Tools

DataStage

Business Intelligence Tools

Cognos Impromptu

Business Objects

Other

ODBC

Custom MetaBroker Development – offered as a Service

Changed Data Capture Support Environments

Changed Data Capture sources include:

DB2 for MVS or OS/390

DB2 UDB for MVS or OS/390

IMS for MVS or OS/390

Oracle for AIX

Oracle for HP-UX

Oracle for Solaris

Oracle for Windows/NT

About Ascential Software

Ascential Software Corporation is the leading provider of Information Asset Management solutions to the Global 2000. Customers use Ascential products to turn vast amounts of disparate, unrefined data into reusable information assets that drive business success. Ascential's unique framework for Information Asset Management enables customers to easily collect, validate, organize, administer and deliver information assets to realize more value from their enterprise data, reduce costs and increase profitability. Headquartered in Westboro, Mass., Ascential has offices worldwide and supports more than 1,800 customers in such industries as telecommunications, insurance, financial services, healthcare, media/entertainment and retail. For more information on Ascential Software, visit http://www.ascentialsoftware.com.

Ascential™ Software

50 Washington Street
Westboro, MA 01581
Tel. 508.366.3888
www.ascentialsoftware.com

ASCENTIAL SOFTWARE REGIONAL SALES OFFICES

North America	800 486 9636	Japan	81 3 5562 4321
	508 366 3888	Asia	852 2824 3311
Northern Europe	44 20 8818 0700	South Africa	27 11 807 0313
Southern Europe	33 (0) 1 4696 37 37	Australia/New Zealand	612 9900 5688
Central & Eastern Europe	49 89 20707 0	Latin America	55 11 5188 1000

WP-3001-0901

Printed in USA 09/01.