TATA CONSULTANCY SERVICES
Experience certainty.

DataStage Basics

16/12/2015

Training Material

Vodafone Hutchison Australia

Kumar S

Telecom

kumar.s4@tcs.com

knowmax
Our Intelligence Hub

Confidentiality Statement

Include the confidentiality statement within the box provided. This has to be legally approved

Confidentiality and Non-Disclosure Notice

The information contained in this document is confidential and proprietary to TATA Consultancy Services. This information may not be disclosed, duplicated or used for any other purposes. The information contained in this document may not be released in whole or in part outside TCS for any purpose without the express written permission of TATA Consultancy Services.

Tata Code of Conduct

We, in our dealings, are self-regulated by a Code of Conduct as enshrined in the Tata Code of Conduct. We request your support in helping us adhere to the Code in letter and spirit. We request that any violation or potential violation of the Code by any person be promptly brought to the notice of the Local Ethics Counselor or the Principal Ethics Counselor or the CEO of TCS.   All communication received in this regard will be treated and kept as confidential.

Table of Content

1. Introduction

Data Stage is a powerful tool that can be used for ETL Process. IBM InfoSphere DataStage is an ETL tool and part of the IBM Information Platforms Solutions suite and IBM InfoSphere. It uses a graphical notation to construct data integration solutions and is available in various versions such as the Server Edition, the Enterprise Edition, and the MVS Edition.

IBM InfoSphere Information Server is a software platform that helps organizations derive more value from the complex, heterogeneous information that is spread across their systems. It provides breakthrough collaboration, productivity, and performance for cleansing, transforming, and moving this information consistently and securely throughout the enterprise. It can then be accessed and used in new ways to drive innovation, increase operational efficiency, and lower risk.

As a component of IBM InfoSphere Information Server, IBM InfoSphere DataStage integrates data across multiple, high-volume data sources and target applications. It integrates data on demand with a high-performance parallel framework, extended metadata management, and enterprise connectivity.

DataStage supports the collection, integration, and transformation of large volumes of data, with data structures ranging from simple to highly complex. DataStage can manage data arriving in real time, as well as data received on a periodic or scheduled basis, which enables companies to solve large-scale business problems through high-performance processing of massive data volumes.

By making use of the parallel processing capabilities of multiprocessor hardware platforms, IBM InfoSphere DataStage Enterprise Edition can scale to satisfy the demands of ever-growing data volumes, stringent real-time requirements, and ever-shrinking batch windows. Along with these key components, establishing consistent development standards helps to improve developer productivity and reduce ongoing maintenance costs. Development standards can also make it easier to integrate.

2. Key features of DataStage

DataStage provides these features and benefits:

- Powerful, scalable ETL platform—supports the collection, integration and transformation of large volumes of data, with data structures ranging from simple to complex.
- Support for big data and Hadoop—enables you to directly access big data on a distributed file system, and helps clients more efficiently leverage new data sources by providing JSON support and a new JDBC connector.
- Near real-time data integration—as well as connectivity between data sources and applications.
- Workload and business rules management—helps you optimize hardware utilization and prioritize mission-critical tasks.
- Ease of use—helps improve speed, flexibility and effectiveness to build, deploy, update and manage your data integration infrastructure.
- Rich support for DB2Z and DB2 for z/OS—including data load optimization for DB2Z and balanced optimization for DB2 on z/OS

DataStage provides some high end features and benefits as listed below

- Powerful
- Big Data & Hadoop support
- Real time data integration
- Work load management
- Ease of use
- Security Controls
- Web Interface
- Data Repository

3. DataStage

DataStage provides a means of quickly creating usable data warehouses or data marts. It is an integrated set of tools for designing, developing, compiling, running, and administering applications that extract data from one or more data sources, perform complex transformations of the data, and load one or more target files or databases with the resulting data.

Solutions developed with DataStage are open and scalable; you can, for example, readily add data sources and targets, or handle increased volumes of data.

Projects

Whenever you start a DataStage client, you are prompted to attach to a DataStage project. Each complete project may contain:

• DataStage jobs. A set of jobs for loading and maintaining a data warehouse.

• Built-in components. Predefined components used in a job.

• User-defined components. Customized components created using the

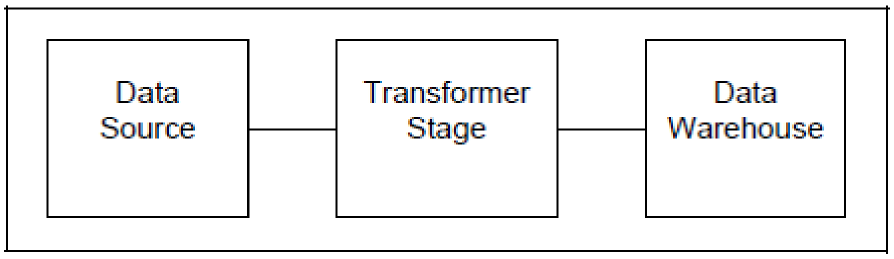
DataStage Manager. Each user-defined component performs a specific task in a job.

Jobs

A DataStage job consists of a series of individual stages, linked together to describe the flow of data from a data source to a final data warehouse or data mart. Each stage describes a particular database or process.

For example, one stage extract data from a data source, while another transforms it. Stages are added to a job and linked together using the DataStage Designer.

The following diagram represents the simplest job you could have: a data source, a Transformer (conversion) stage, and the final database.



The links between the stages represent the flow of data into or out of a stage. You must specify the data you want at each stage, and how it is handled.

Data properties are defined by:

- **Table definitions**

  These specify the data you want. Each table definition contains:

  - Information about the table or file that holds the data records.
  - A description of the individual columns.

- **Data elements**

  Each data element describes one type of data that can be stored in a column. The data element associated with a column defines the operations that can be carried out on that column. DataStage has numerous predefined data elements representing commonly required data types (such as date, time, number, and string). You can also define your own special data elements.

- **Transforms**

These convert and cleanse your data by transforming it into a format you want to save and use in your final warehouse. DataStage provides a large library of built-in transforms. Together, these properties determine what occurs at each stage of a DataStage job. The properties are set up project-wide and are shared by all the jobs in a project. Compilation of a job creates an executable that is scheduled by the DataStage
Director and run by the DataStage Server. When the job is run, the processing stages described in the job design are performed using the data properties you defined. Executable jobs can be packaged for use on other Datastage systems.

Stages

A stage can be passive or active. A passive stage handles access to databases for the extraction or writing of data. Active stages model the flow of data and provide mechanisms for combining data streams, aggregating data, and converting data from one data type to another.
There are two types of stage:

- Built-in stages. Supplied with DataStage and used for extracting, aggregating, transforming, or writing data. These stages can be either passive or active.
- Plug-in stages. Additional stages defined in the DataStage Manager to perform tasks that the built-in stages do not support.

A stage usually has at least one data input and one data output. However, some stages can accept more than one data input, and output to more than one stage.
Stages and links can be grouped together to form a container. A container is represented by a Container stage. The links to and from a container are represented by Container Input and Container Output stages.

National Language Support

DataStage can be installed with built-in National Language Support (NLS). NLS enables DataStage to:

- Process data in a wide range of languages
- Accept data in almost any character set into most DataStage fields
- Use local formats for dates, times, and money
- Sort data according to local rules
- Convert data between different encodings of the same language (for example, for Japanese it can convert JIS to EUC)

This remarkable capability is made possible by DataStage using an internal character set based on the international standard UNICODE format. It invisibly translates data to and from this character set, as required.

4. DataStage Components

DataStage has two components:

- Server Components
- Client Components

Server Components

DataStage has three server components:

- **Repository.** A central store that contains all the information required to build a data mart or data warehouse.
- **DataStage Server.** Runs executable jobs, under the control of the DataStage Director, that extract, transform, and load data into a data warehouse.
- **DataStage Package Installer**. A user interface used to install packaged DataStage jobs and Plug-ins.

Client Components

DataStage has four client components, which are installed on any PC running Windows 98 or Windows NT 4.0 or later:

- **DataStage Manager.** A user interface used to view and edit the contents of the Repository.
- **DataStage Designer.** A graphical tool used to create DataStage jobs.
- **DataStage Director.** A user interface used to validate, schedule, run, and monitor DataStage jobs.
- **DataStage Administrator.** A user interface used to set up DataStage users, control purging criteria, and install NLS maps and locales.

The exercises in this tutorial center on the clients. The server components require little interaction, although the exercises in which you use the DataStage Manager also give you the opportunity to examine the Repository.

The following sections briefly introduce the DataStage Manager, Designer, Director, and Administrator. We return to these tools during the exercises, when you learn how to use them to accomplish specific tasks. In doing so you gain some familiarity with each tool.

DataStage Manager

The DataStage Manager is a graphical tool that enables you to view and manage the contents of the DataStage Repository.

You use it to browse, import, and edit Meta data about data sources, targets, and transformations. It also allows you to define or install DataStage components such as data elements, custom plug-in stages, custom transforms, and routines.

DataStage Designer

DataStage Designer to build jobs by creating a visual design that models the flow and transformation of data from the data source through to the target warehouse. The DataStage Designer graphical interface lets you select stage in the DataStage Designer, you define the required actions and processes for each stage and link. You compile your jobs from the Designer, which also provides a job debugger.

A job created with the DataStage Designer is easily scalable. This means that you can easily create a simple job, get it working, then insert further processing, additional data sources, and so on.

As you work through the exercises in this tutorial, you use the DataStage Designer to create a variety of jobs. These jobs introduce you to the input and output stages, and demonstrate some of the ways in which you can use the Transformer stage.

You also do an exercise that demonstrates how to run the debugger. By the time you complete the exercises you will be able to appreciate both the power and ease of use of the DataStage Designer.

DataStage Director

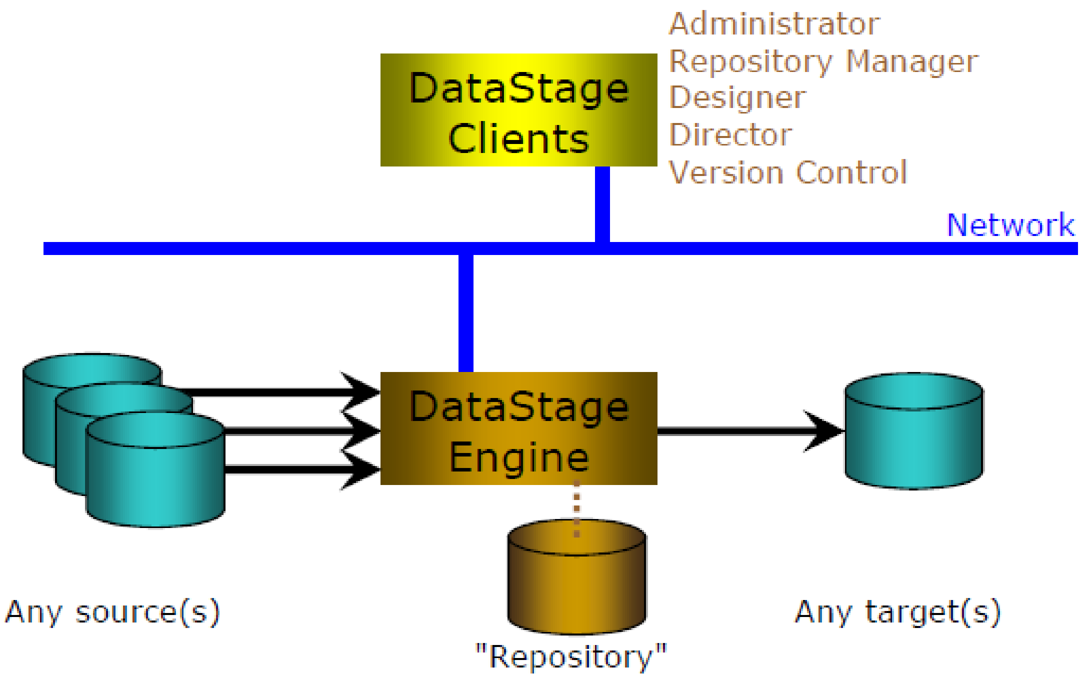
The DataStage Director enables you to run, monitor, and control jobs built in the DataStage Designer. When you run a job, the Director prompts for any run-time parameters. It lets you monitor run-time performance and events such as error conditions. You can run jobs on an ad hoc basis or schedule them.

DataStage Administrator

The DataStage Administrator enables you to set up DataStage users, control the purging of the Repository, and, if NLS is installed, install and manage maps and locales. In Chapter 9 you use the DataStage Administrator to install various maps and locales which are used in the exercises to demonstrate the power of NLS.

5. DataStage Architecture

DataStage server and a local area network on which one or more DataStage client machines may be connected. When clients are remote from the server, a wide area network may be used or some form of tunnelling protocol (such as Citrix MetaFrame) may be used instead.

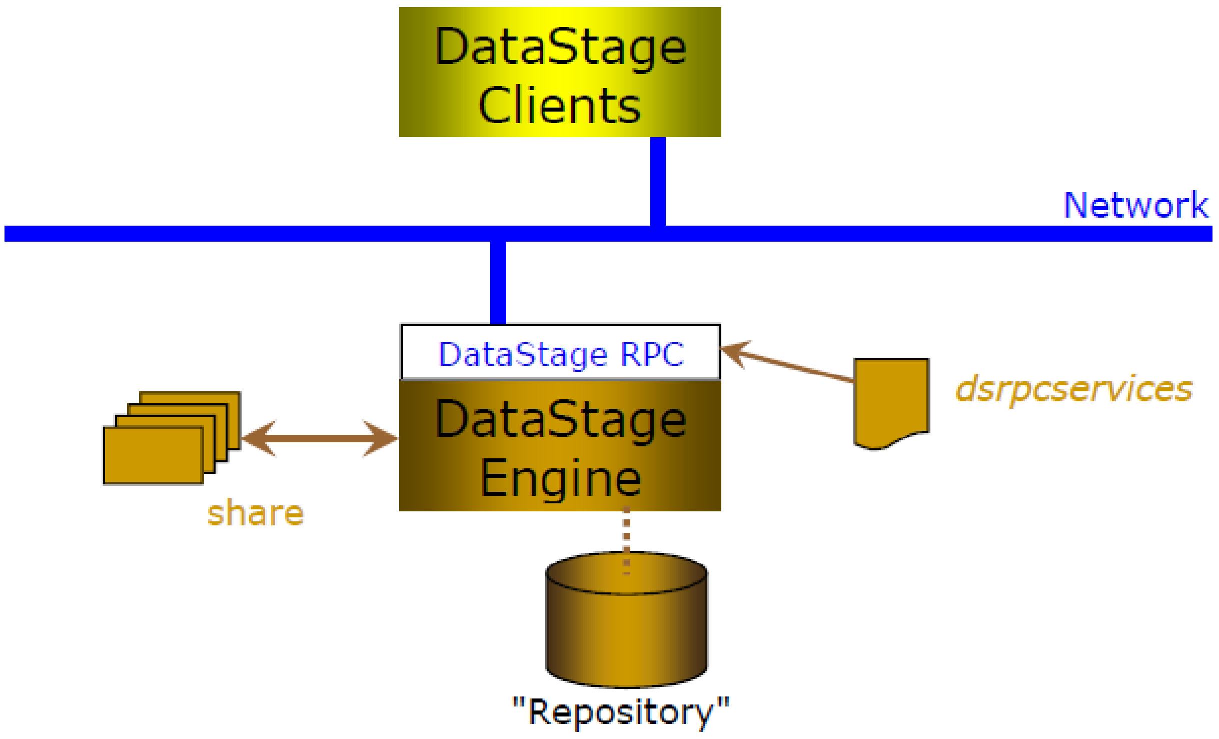


Note that the Repository Manager and Version Control clients do not exist in version 8.0 and later. Different mechanisms exist, which will be discussed later.

Connection to data sources and targets can use many different techniques, primarily direct access (for example directly reading/writing text files), industry-standard protocols such as ODBC, and vendor specific APIs for connecting to databases and packages such as Siebel, SAP, Oracle
Financials, etc.

DataStage Client/Server Connectivity

Connection from a DataStage client to a DataStage server is managed through a mechanism based upon the UNIX remote procedure call mechanism. DataStage uses a proprietary protocol called DataStage RPC which consists of an RPC daemon (dsrpcd) listening on TCP port number 31538 for connection requests from DataStage clients. Before dsrpcd gets involved, the connection request goes through an authentication process. Prior to version 8.0, this was the standard operating system authentication based on a supplied user ID and password (an option existed on Windows-based DataStage servers to authenticate using Windows LAN Manager, supplying the same credentials as being used on the DataStage client machine – this option was removed for version 8.0). With effect from version 8.0 authentication is handled by the Information Server through its login and security service.



Each connection request from a DataStage client asks for connection to the dscs (DataStage Common Server) service.
The dsrpcd (the DataStage RPC daemon) checks its dsrpc services file to determine whether there is an entry for that service and, if there is, to establish whether the requesting machine's IP address is authorized to request the service. If all is well, then the executable associated with the dscs service (dsapi_server) is invoked.

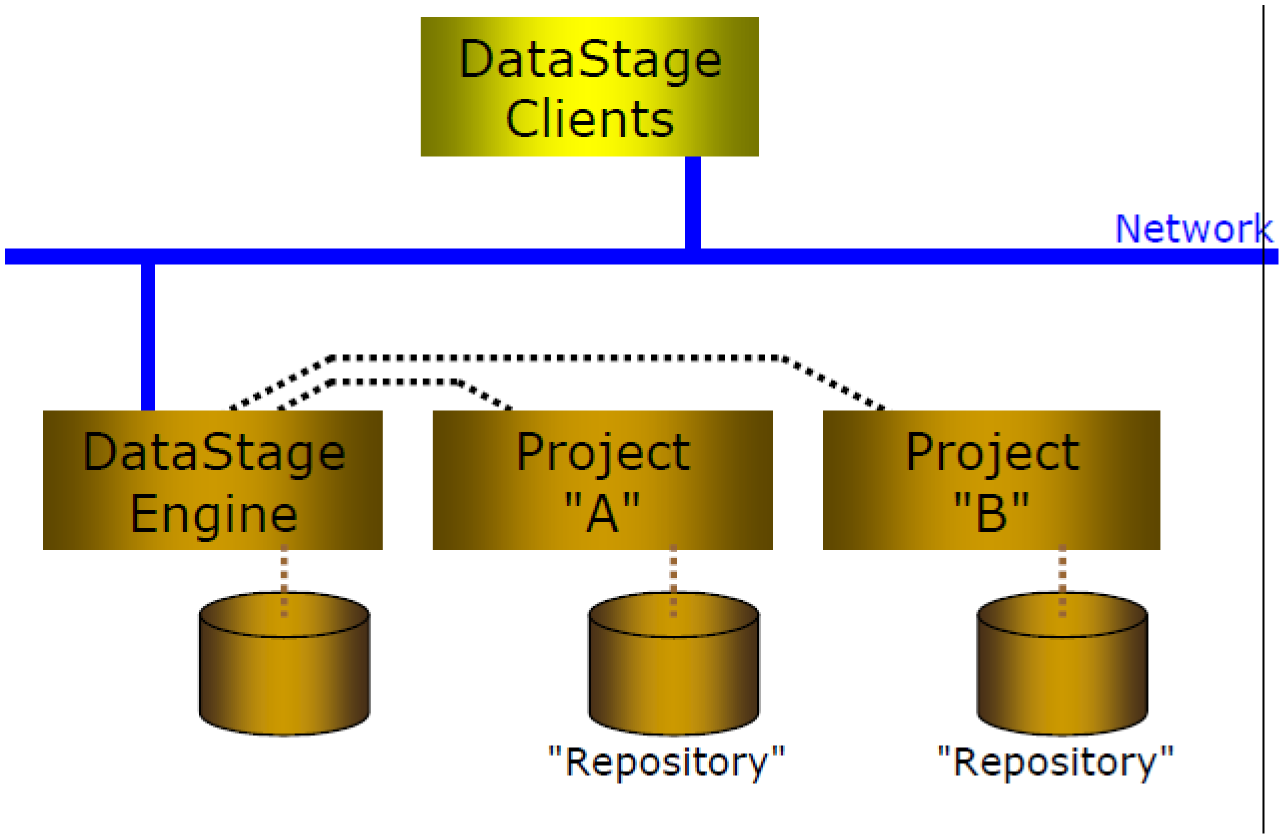
DataStage Processes and Shared Memory

Each dsapi_server process acts as the "agent" on the DataStage server for its own particular client connection, among other things managing traffic and the inactivity timeout. If the client requests access to the Repository, then the dsapi_server process will fork a child process called dsapi_slave to perform that work.
Typically, therefore, one would expect to see one dsapi_server and one dsapi_slave process for each connected DataStage client. Processes may be viewed with the ps -ef command (UNIX) or with Windows Task Manager.

Every DataStage process attaches to a shared memory segment that contains lock tables and various other inter-process communication structures. Further each DataStage process is allocated its own private shared memory segment. At the discretion of the DataStage administrator there may also be shared memory segments for routines written in the DataStage BASIC language and for character maps used for National Language Support (NLS). Shared memory allocation may be viewed using the ipcs command (UNIX) or the shrdump command (Windows). The shrdump command ships with DataStage; it is not a native Windows command.

DataStage Projects

Talking about the Repository is a little misleading. As noted earlier, DataStage is organized into a number of work areas called "projects". Each project has its own individual local Repository in which its own designs and technical and process metadata are stored.



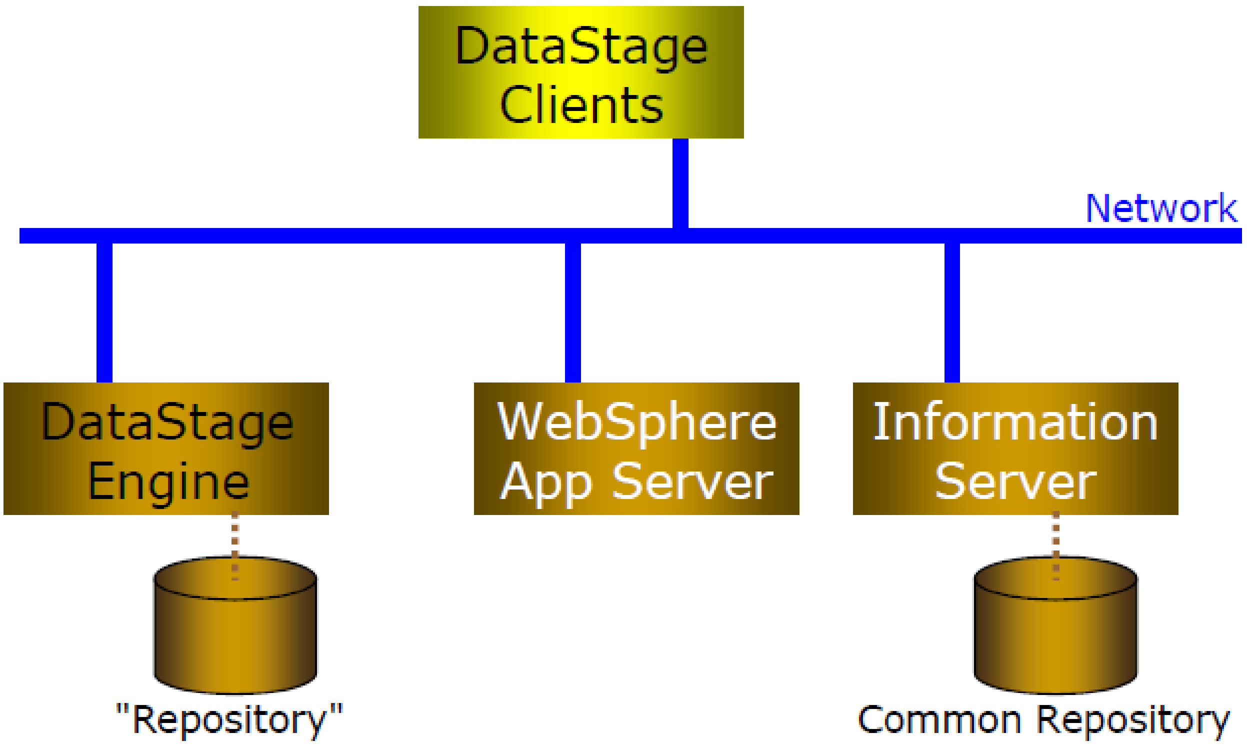
Each project has its own directory on the server, and its local Repository is a separate instance of the database associated with the DataStage server engine. The name of the project and the schema name of the database instance are the same. System tables in the DataStage engine record the existence and location of each project. Location of any particular project may be determined through the Administrator client, by selecting that project from the list of available projects. The pathname of the project directory is displayed in the status bar.

When there are no connected DataStage clients, dsrpcd may be the only DataStage process running on the DataStage server. In practice, however, there are one or two more.

The DataStage deadlock daemon (dsdlockd) wakes periodically to check for deadlocks in the DataStage database and, secondarily, to clean up locks held by defunct processes – usually improperly disconnected
DataStage clients.
Job monitor is a Java application that captures "performance" data (row counts and times) from running DataStage jobs. This runs as a process called JobMonApp.

Changes in Version 8.0

In version 8.0 all of the above still exists, but is layered on top of a set of services called collectively IBM Information Server. Among other things, this stores metadata centrally so that the metadata are accessible to many products, not just DataStage, and exposes a number of services including the metadata delivery service, parallel execution services, connectivity services, administration services and the already-mentioned login/security service. These services are managed through an instance of a WebSphere
Application Server.

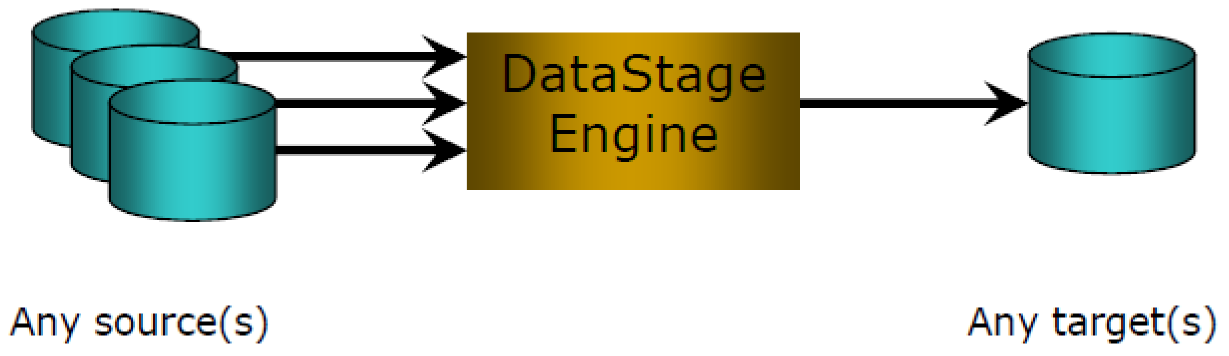


The common repository, by default called XMETA, may be installed in DB2 version 9 or later, Oracle version 10g or later, or Microsoft SQL Server 2003 or later.
The DataStage server, WebSphere Application server and Information Server may be installed on separate machines, or all on the same machine, or some combination of machines. The main difference noticed by
DataStage users when they move to version 8 is that the initial connection request names the Information Server (not the DataStage server) and must specify the port number (default 9080); on successful authentication they are

then presented with a list of DataStage server and project combinations being managed by that particular Information Server.

Run-Time Architecture

Now let us turn our attention to run-time, when DataStage jobs are executing. The concept is a straightforward one; DataStage jobs can run even though there are not clients connected (there is a command line interface (dsjob) for requesting job execution and for performing various other tasks).



However, server jobs and parallel jobs execute totally differently. A job sequence is a special case of a server job, and executes in the same way as a server job.

Server Job Execution

Server jobs execute on the DataStage server (only) and execute in a shell called uvsh (or dssh, a synonym). The main process that runs the job executes a DataStage BASIC routine called DSD.RUN – the name of this program shows in a ps –ef listing (UNIX). This program interrogates the local Repository to determine the runtime configuration of the job, what stages are to be run and their interdependencies. When a server job includes a Transformer stage, a child process is forked from uvsh also running uvsh but this time executing a DataStage BASIC routine called DSD. Stage Run. Server jobs only ever have uvsh processes at run time, except where the job design specifies opening a new shell (for example sh in UNIX or DOS in Windows) to perform some specific task; these will be child processes of uvsh.

Parallel Job Execution

Parallel job execution is rather more complex. When the job is initiated the primary process (called the "conductor") reads the job design, which is a generated Orchestrate shell (osh) script. The conductor also reads the parallel execution configuration file specified by the current setting of the APT_CONFIG_FILE environment variable. Based on these two inputs, the conductor process composes the "score", another osh script that specifies what will actually be executed.

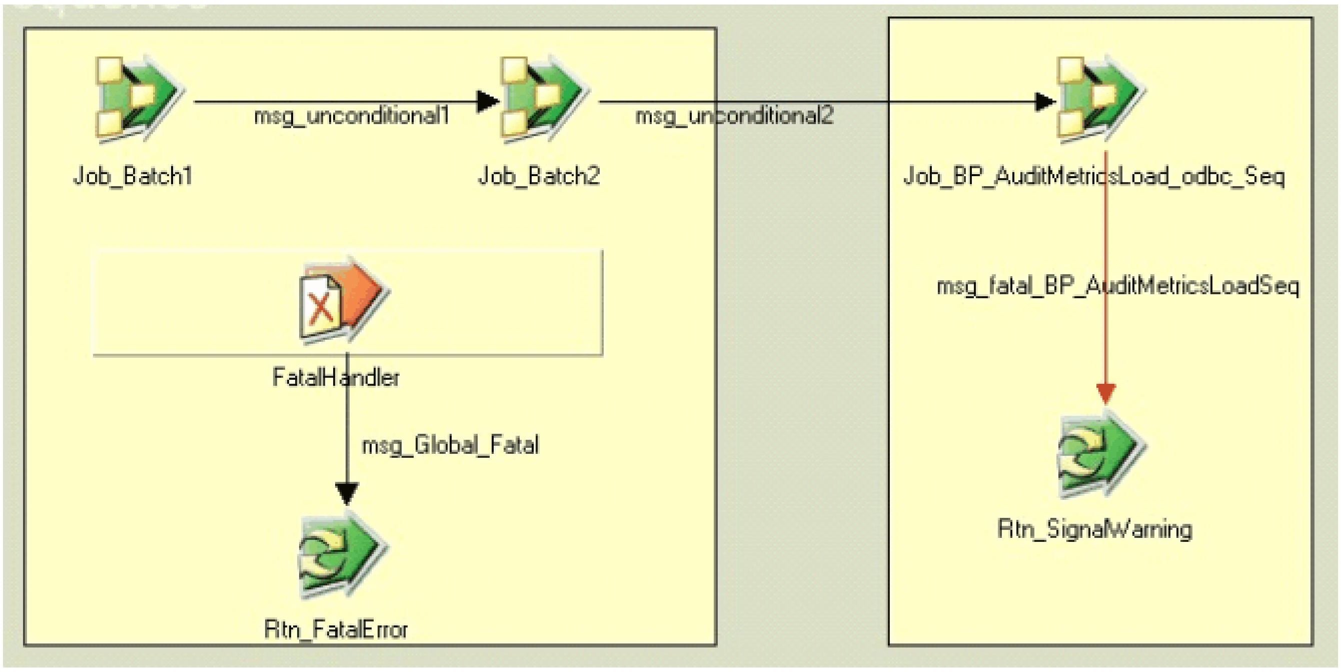
6. DataStage Jobs

Work performed by data integration jobs fall into four general categories:

- Reading input data, including sequential files, databases and DataStage (DS) Parallel Datasets
- Performing row validation to support data quality
- Performing transformation from data sources to data targets
- Provisioning data targets

Job sequences

It is built from individual parallel jobs assembled in IBM InfoSphere DataStage, controlled as modules from master DataStage Sequence jobs



These job sequences control the interaction and error handling between individual DataStage jobs, and together form a single end-to-end module in a DataStage application.

Job sequences also provide the recommended level of integration with external schedulers (such as AutoSys, Cron, CA7, and so forth). This provides a level of granularity and control that is easy to manage and maintain, and provides an appropriate use of the respective technologies.

In most production deployments, job sequences require a level of integration with various production automation technologies (such as scheduling, auditing/capture, and error logging).

7. Job types

Nearly all data integration jobs fall into three major types:

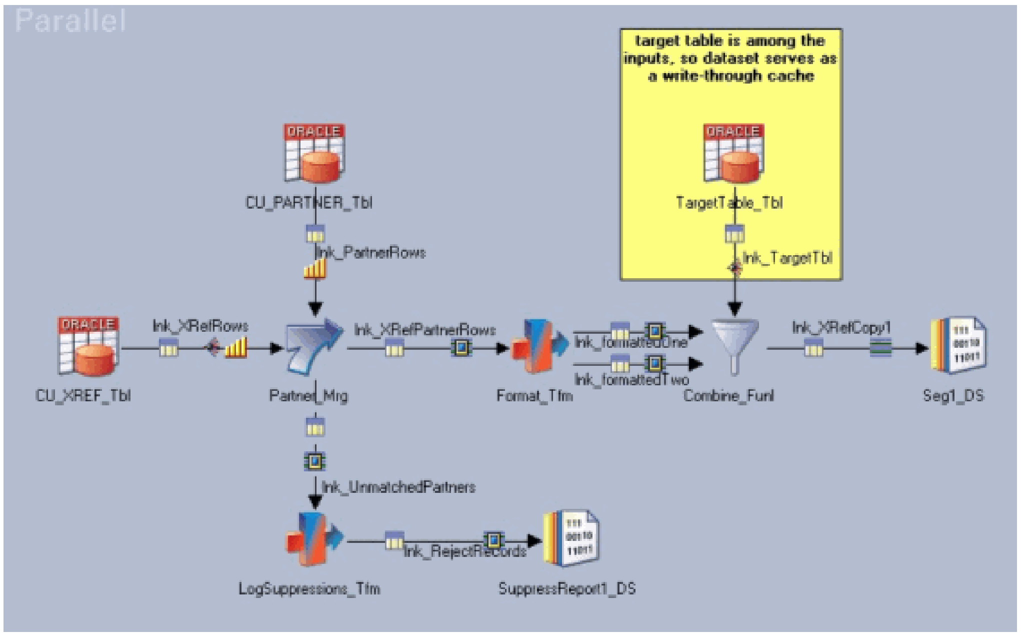
- Transformation jobs
- Provisioning jobs
- Hybrid jobs

Transformation jobs

In transformation jobs, data sources, which might be write-through cache datasets, are processed to produce a load-ready dataset that represents either the entire target table or new records to be appended to the target table.

If the entire target table is regenerated with each run, and no other external or subsequent processing alters the contents of the target table, the output dataset qualifies as a write-through cache that can be used by subsequent DataStage jobs instead of reading the entire target table.

When we say "subsequent jobs" we mean any jobs executed as part of the same transformation cycle, until the target table is updated in the target database by a provisioning job. A transformation cycle might correspond, for instance, to daily, weekly, or monthly batch processes.



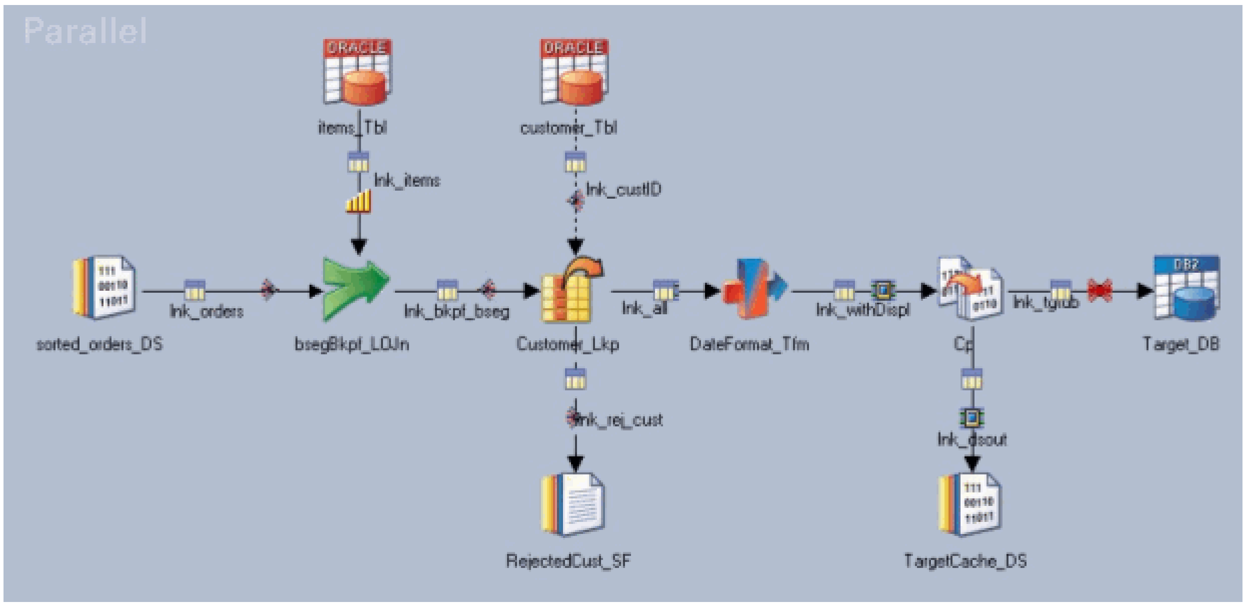
As a transformation cycle progresses, any real-time updates to the target tables in the target database (by online or real-time applications, for instance), must cease, so as not to yield invalid results at the end of the processing cycle. Otherwise, at the final steps of the processing window, provisioning jobs would invalidate or overwrite modifications performed by the online applications. The target table would then become inconsistent.

The following example transformation job demonstrates the use of write-through cache parallel datasets: The source Oracle stages read data from a couple of source tables, which are joined and updated by a Parallel Transformer. This transformed join result is funneled with the content of the target table (extracted by the "TargetTable_tbl" stage). The result of the Funnel is saved to the Seg1_DS parallel dataset.

The content of the Seg1_DS parallel dataset can be used by subsequent jobs, in the same transformation cycle, as input to other joins, lookup, or transform operations, for instance. Those jobs can thus avoid re-extracting data from that table unnecessarily. These operations might result in new versions of this dataset. Those new versions replace Seg1_DS (with different names) and are used by other subsequent downstream jobs.

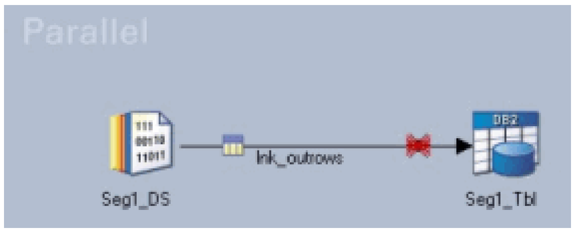
Hybrid jobs

The sample hybrid job depicted in Figure 2-5 demonstrates a job that transforms and combines source data, creating a dataset of target rows, and loading the target table. Because the target table is not sourced, this is an insert of new rows instead of creating a complete write-through cache.



Provisioning jobs

The sample provisioning job depicted in Figure 2-6 demonstrates the straightforward approach to simple provisioning tasks. In general, it is a good idea to separate the loading of a database from the ETL process, as a database load can often fail due to external reasons. Creating the load-ready dataset allows the provisioning job to be rerun without reprocessing the source extracts and transformations.



8. DataStage Advantages

- Development Partnerships
- Single vendor solution for bulk data transfer and complex transformations
- Transparent and wide range of licensing options
- Single interface to integrate heterogeneous applications
- Flexible development environment
- It helps get most out of hardware investment and resources.
- The DataStage server performs very well on both Windows and Unix servers
- It is able to integrate data coming from all parts of the company
- It helps to understand the new and already existing clients
- It handles all company data and adapts to the needs
- It offers the possibility for the organization of a complex business intelligence
- Flexible and scalable
- It accelerates the running of the project
- Easily implementable.
- It's Faster
- Transformer Looping
- Transformer Remembering
- Easy to Install
- Check In and Check Out Jobs
- High Availability Easier than ever
- New Information Architecture Diagramming Tool
- Vertical Pivot
- Z/OS File Stage
- Balanced Optimizer Comes Home

9. DataStage Disadvantages

- Big architectural differences in the Server and Enterprise edition which results in the fact that migration from server to enterprise edition may require vast time and resources effort.
- There is no automated error handling and recovery mechanism - for example no way to automatically time out zombie jobs or kill locking processes. However, on the operator level, these errors can be easily resolved.
- No Unix Datastage client - the Client software available only under Windows and there are different clients for different DataStage versions. The good thing is that they still can be installed on the same windows pc and switched with the Multi-Client Manager program.
- Might be expensive as a solution for a small or mid-sized company.
- Data from different sources would be different, poorly documented and dirty.
- Name and address standardization is not very easy
- Specialized software for standardization – specific to geography
- Perfecting the data is costly

Thank You

Contact

For more information, contact **kumar.s4@tcs.com** (Email Id of ISU)

About Tata Consultancy Services (TCS)

Tata Consultancy Services is an IT services, consulting and business solutions organization that delivers real results to global business, ensuring a level of certainty no other firm can match. TCS offers a consulting-led, integrated portfolio of IT and IT-enabled infrastructure, engineering and assurance services. This is delivered through its unique Global Network Delivery Model^{TM}, recognized as the benchmark of excellence in software development. A part of the Tata Group, India's largest industrial conglomerate, TCS has a global footprint and is listed on the National Stock Exchange and Bombay Stock Exchange in India.

For more information, visit us at **www.tcs.com**.

IT Services
Business Solutions
Consulting

All content / information present here is the exclusive property of Tata Consultancy Services Limited (TCS). The content / information contained here is correct at the time of publishing. No material from here may be copied, modified, reproduced, republished, uploaded, transmitted, posted or distributed in any form without prior written permission from TCS. Unauthorized use of the content / information appearing here may violate copyright, trademark and other applicable laws, and could result in criminal or civil penalties. Copyright © 2011 Tata Consultancy Services Limited