



Microsoft®  
**SQL Server™ 2008**

**An Introduction to SQL Server 2008  
Integration Services**

SQL Server Technical Article

---

# Copyright

This is a preliminary document and may be changed substantially prior to final commercial release of the software described herein.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2007 Microsoft Corporation. All rights reserved.

Microsoft, Office Excel, Reporting Services, Visual Basic, Visual C#, Visual C++, and Visual Studio are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

---

# Table of Contents

<b>Introduction</b> .....	<b>1</b>
A Real-World Scenario .....	1
Data Sources .....	1
Data Consumption.....	1
Data Integration Requirements .....	1
<b>Challenges of Data Integration</b> .....	<b>2</b>
Technology Challenges .....	2
Organizational Challenges .....	3
Power Challenge .....	4
Comfort Zone Challenge.....	4
Economic Challenges .....	4
<b>SQL Server 2008 Integration Services</b> .....	<b>4</b>
SSIS Architecture .....	5
Task flow and data flow engine .....	5
Pipeline architecture .....	5
ADO.NET connectivity .....	6
Thread pooling.....	6
Persistent lookups .....	6
Integration Scenarios .....	6
SSIS for data transfer operations .....	6
SSIS for data warehouse loading .....	7
SSIS and Data Quality .....	10
Application of SSIS Beyond Traditional ETL.....	11
SSIS, the Integration Platform.....	14
<b>Making Data Integration Approachable</b> .....	<b>17</b>
<b>Conclusion</b> .....	<b>19</b>



# Introduction

The ability to transform corporate data into meaningful and actionable information is the single most important source of competitive advantage in today's business world. Harnessing the data explosion to better understand the past and get direction for the future has turned out to be one of the most challenging ventures for enterprise Information Technology departments in global organizations. There are three broad categories of issues associated with data integration:

- Technology challenges
- Organizational issues
- Economic challenges

In this paper, we will explore these challenges in detail and discuss how to address them with Microsoft® SQL Server™ 2008 Integration Services (SSIS). First you should view them in the context of a real-world scenario.

## A Real-World Scenario

A major global transportation company uses its data warehouse to both analyze the performance of its operations and to predict variances in its scheduled deliveries.

## Data Sources

The major sources of data in this company include order data from its DB2-based order entry system, customer data from its SQL Server-based customer relationship management (CRM) system, and vendor data from its Oracle-based ERP system. In addition to data from these major systems, you incorporate data from spreadsheets that track "extraordinary" events into the data warehouse, which shipping supervisors have entered by hand. Currently, you incorporate external data such as weather information, traffic status, and vendor details (for subcontracted deliveries) on a delayed basis from text files from various sources.

## Data Consumption

Not only are the sources for these data diverse, but the consumers are also diverse both in their requirements and their geographic locations. This diversity has led to a proliferation of local systems. One of the major efforts for the Information Technology department is to establish a "single version of the truth," at least for its customer data.

## Data Integration Requirements

In view of this diversity of data, business needs, and user requirements, the Information Technology department has specified the following set of data integration requirements:

- They must provide reliable and consistent historical and current data integrated from a variety of internal and external sources.
- To reduce lags in data acquisition, data from providers and vendors must be available via Web services or some other direct mechanism such as FTP.
- They need to cleanse and remove duplicate data and otherwise enforce data quality.

- Increasing global regulatory demands require that the company maintain clear audit trails. It is not enough to maintain reliable data; the data needs to be tracked and certified.

## Challenges of Data Integration

At one level, the problem of data integration in our real-world scenario is extraordinarily simple. Get data from multiple sources, cleanse and transform the data, and load the data into appropriate data stores for analysis and reporting. Unfortunately, in a typical data warehouse or business intelligence project, enterprises spend 60–80% of the available resources in the data integration stage. Why is it so difficult?

## Technology Challenges

Technology challenges start with source systems. We are moving from collecting data on transactions (where customers commit to getting, buying, or otherwise acquiring something) to collecting data on pre-transactions (where mechanisms such as Web clicks or RFID tags track customer intentions). Data is now not only acquired via traditional sources and formats, such as databases and text files, but is increasingly available in a variety of different formats (ranging from proprietary files to Microsoft Office documents to XML-based files) and from Internet-based sources such as Web services and RSS (Really Simple Syndication) streams. The most pertinent challenges are:

- Multiple sources with different formats.
- Structured, semi-structured, and unstructured data.
- Data feeds from source systems arriving at different times.
- Huge data volumes.

In an ideal world, even if you somehow manage to get all the data we need in one place, new challenges start to surface, including:

- Data quality.
- Making sense of different data formats.
- Transforming the data into a format that is meaningful to business analysts.

Suppose that you can magically get all of the data that you need and that you can cleanse, transform, and map the data into a useful format. There is still another shift away from traditional data movement and integration. That is the shift from fixed long batch-oriented processes to fluid and shorter on-demand processes. Most organizations perform batch-oriented processes during “downtimes” when users do not place heavy demands on the system. This is usually at night during a predefined batch window of 6–8 hours, when no one is supposed to be in the office. With the increasing globalization of businesses of every size and type, this is no longer true. There is very little (if any) downtime and someone is always in the office somewhere in the world.

As a result you have:

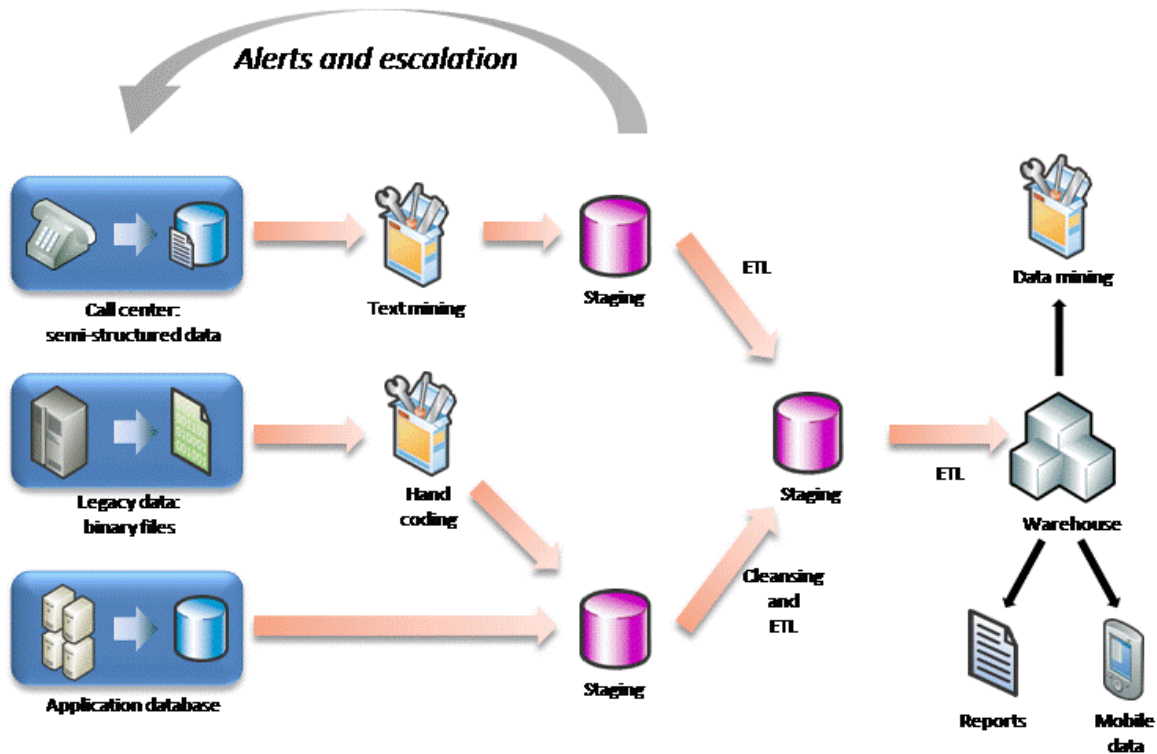
- Increasing pressure to load the data as quickly as possible.
- The need to load multiple destinations at the same time.
- Diverse destinations.

Not only do you need to achieve all of these results, but also you need to achieve them as fast as possible. In extreme cases, such as online businesses, you must integrate

data on a continuous basis. There are no real batch windows and latencies cannot exceed minutes. In many of these scenarios, the decision-making process is automated with continuously running software.

Scalability and performance become more and more important as you face business needs that cannot tolerate any downtime.

Without the right technology, systems require staging at almost every step of the warehousing and integration process. As different (especially nonstandard) data sources need to be included in the Extract, Transform, and Load (ETL) process and as more complex operations (such as data and text mining) need to be performed on the data, the need to stage the data increases. As illustrated in Figure 1, with increased staging the time taken to “close the loop,” (i.e., to analyze, and take action on new data) increases as well. These traditional ELT architectures (as opposed to value-added ETL processes that occur prior to loading) impose severe restrictions on the ability of systems to respond to emerging business needs.



**Figure 1**

Finally, the question of how data integration ties into the overall integration architecture of the organization is becoming more important when you need both the real-time transactional technology of application integration and the batch-oriented high-volume world of data integration technology to solve the business problems of the enterprise.

## Organizational Challenges

There are two broad issues with data integration in a large organization; these are the “power” challenge, and the “comfort zone” challenge.

## Power Challenge

Data is power and it is usually very hard to make people think of data in terms of a real valuable shared asset of the company. For enterprise data integration to be successful, all of the owners of multiple data sources must buy into the purpose and direction of the project. Lack of cooperation from the relevant parties is one of the major reasons for the failure of data integration projects. Executive sponsorship, consensus building, and a strong data integration team with multiple stakeholders are a few of the critical success factors that can help resolve the issues.

## Comfort Zone Challenge

You can solve challenges of data integration, when analyzed in the context of an isolated need, in multiple ways. Hand coding solves about 60% of data integration. The technology used to solve similar problems can range from replication, ETL, SQL, to Enterprise Application Integration (EAI). People gravitate towards the technology with which they are familiar. Although these approaches have overlapping capabilities and can perhaps do the job in isolated cases, these technologies are optimized to solve different sets of problems. When trying to solve the problem of enterprise data integration, the lack of a sound architecture with appropriate technology choices can turn out to be a recipe for failure.

## Economic Challenges

The organizational and technology related issues previously outlined conspire together to make data integration the most expensive part of any data warehouse/business intelligence project. The major factors that add to the cost of data integration are:

- Getting the data out in the format that is necessary for data integration ends up being a slow and torturous process fraught with organizational power games.
- Cleansing the data and mapping the data from multiple sources into one coherent and meaningful format is extraordinarily difficult
- More often than not, standard data integration tools do not offer enough functionality or extensibility to satisfy the data transformation requirements for the project. This can result in the expenditure of large sums of money in consulting costs to develop special ETL code to get the job done.
- Different parts of the organization focus on the data integration problem in silos.

When there is a need to put them all together, additional costs are incurred to integrate these efforts into an enterprise-wide data integration architecture.

As the data warehousing and business intelligence needs of the organization evolve, faulty data integration architecture becomes more and more difficult to maintain and the total cost of ownership skyrockets.

## SQL Server 2008 Integration Services

The traditional ETL-centric data integration from standard data sources continues to be at the heart of most data warehouses. However, demands to include more diverse data sources, regulatory requirements, and global and online operations are quickly transforming the traditional requirements for data integration. In this fast growing and changing landscape, the need to extract value from data and the need to be able to rely on it is more important than ever before. Effective data integration has become the

basis of effective decision-making. SQL Server Integration Services provides a flexible, fast, and scalable architecture that enables effective data integration in current business environments.

In this paper, we will examine how SQL Server Integration Services (SSIS) is an effective toolset for both the traditional demands of ETL operations, as well as for the evolving needs of general-purpose data integration. We will also discuss how SSIS is fundamentally different from the tools and solutions provided by major ETL vendors so it is ideally suited to address the changing demands of global business from the largest enterprise to the smallest business.

## SSIS Architecture

### Task flow and data flow engine

SSIS consists of both an operations-oriented task-flow engine as well as a scalable and fast data-flow engine. The data flow exists in the context of an overall task flow. The task-flow engine provides the runtime resource and operational support for the data-flow engine. This combination of task flow and data flow enables SSIS to be effective in traditional ETL or data warehouse (DW) scenarios as well as in many other extended scenarios such as data center operations. In this paper we will mainly focus on the data-flow related scenarios. The use of SSIS for data center oriented workflow is a separate topic by itself.

### Pipeline architecture

At the core of SSIS is the data transformation pipeline. This pipeline has a buffer-oriented architecture that is extremely fast at manipulating rowsets of data once they have been loaded into memory. The approach is to perform all data transformation steps of the ETL process in a single operation without staging data, although specific transformation or operational requirements, or indeed hardware may be a hindrance. Nevertheless, for maximum performance, the architecture avoids staging. SSIS even avoids copying the data in memory as far as possible. This is in contrast to traditional ETL tools, which often require staging at almost every step of the warehousing and integration process. The ability to manipulate data without staging extends beyond traditional relational and flat file data and beyond traditional ETL transformation capabilities. With SSIS, all types of data (structured, unstructured, XML, etc.) are converted to a tabular (columns and rows) structure before being loaded into its buffers. Any data operation that you can apply to tabular data, you can also apply to the data at any step in the data-flow pipeline. This means that a single data-flow pipeline can integrate diverse sources of data and perform arbitrarily complex operations on this data without having to stage the data.

You should note though, that if staging is required for business or operational reasons, SSIS has good support for these implementations as well.

This architecture enables you to use SSIS in a variety of data integration scenarios, ranging from traditional DW-oriented ETL to nontraditional information integration technologies.

## ADO.NET connectivity

A significant aspect of an integration services solution is the extraction or loading of data. It is therefore important that your integration solution can connect seamlessly to a wide range of data sources to make the most of the performance and reliability benefits brought by a comprehensive data access platform. SQL Server 2008 Integration Services is optimized for ADO.NET connectivity (previous versions were optimized for OLE DB or ODBC.) The move to ADO.NET improves system integration and third party support. SQL Server 2005 Integration Services used OLE DB for important tasks such as lookups, but now you can use ADO.NET for tasks as well as source and destination components.

## Thread pooling

As an integration solution is scaled up, it often reaches a plateau beyond which performance improvements are difficult to achieve. SQL Server 2008 Integration Services breaks through this limitation by sharing threads among multiple components, which increases parallelism and reduces blocking; and therefore increases performance in large highly parallel, multiple-processor, multiple-core systems.

As well as increasing performance on most systems, thread pooling also reduces the need for manual configuration of SSIS packages to increase parallelism, and therefore increased developer productivity.

## Persistent lookups

Performing lookups is one of the most common operations in an integration solution. This is especially prevalent in data warehousing where fact records use lookups to transform business keys to their corresponding surrogates. SQL Server 2008 Integration Services increases the performance of lookups to scale to meet the largest tables.

You can configure Lookup transformations to cache some, or all, of the reference data before the input column is processed. SQL Server 2008 Integration Services can load a full cache from any source and allows the cache to be greater than 4 GB, even on a 32-bit operating system. By using a partial cache, SQL Server 2008 Integration Services pre-charges the Lookup by using the data flow. Partial caches support OLEDB, ADO.Net, and ODBC for database lookups, and they track lookup hits and misses. If you choose not to pre-cache reference data, SQL Server 2008 Integration Services supports batched database calls and case-insensitive matches.

## Integration Scenarios

### SSIS for data transfer operations

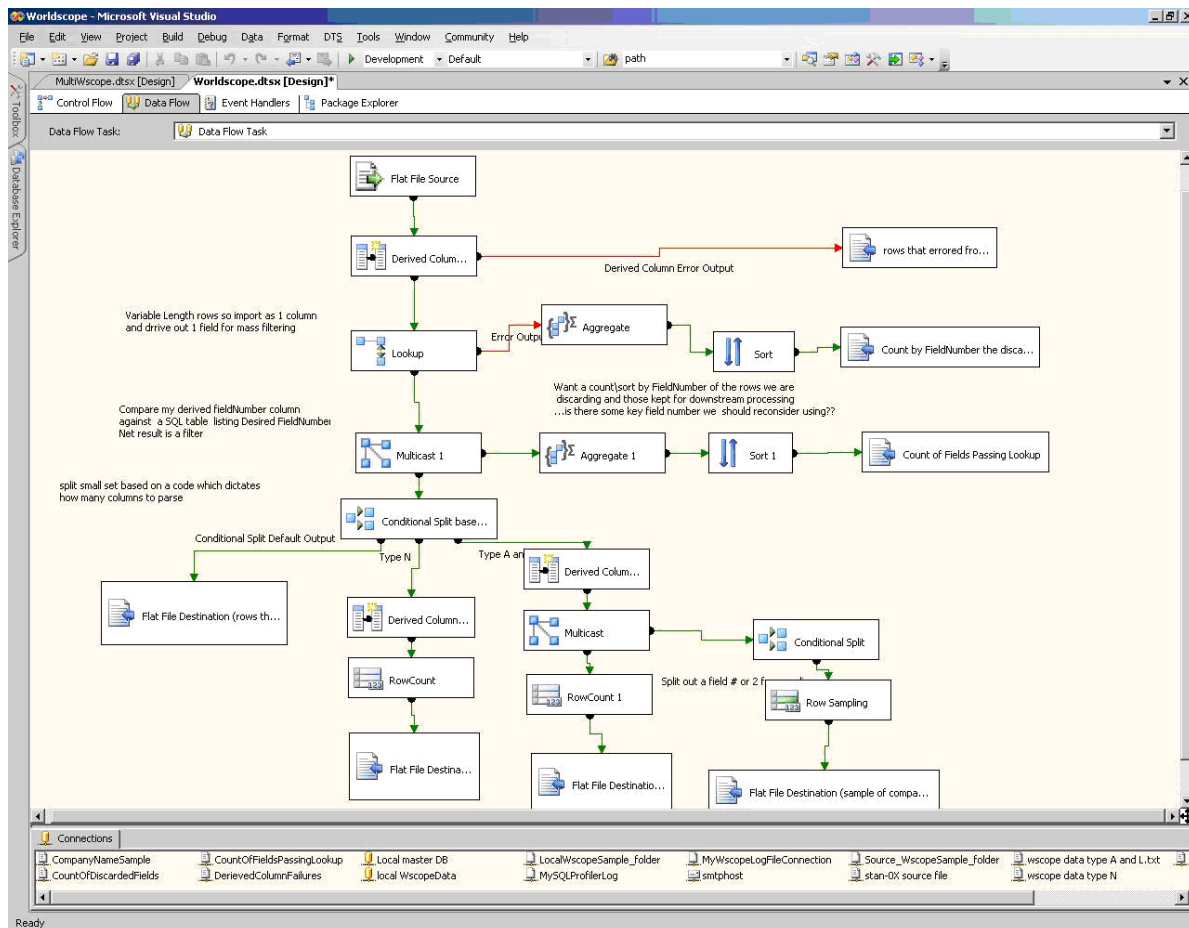
Although SQL Server 2005 Integration Services was a much more functional and powerful product than previous versions, many users found that simple data transfers by using the wizard was more complex and less functional. SQL Server 2008 Integration Services has an improved wizard that uses ADO.NET, has an improved user interface, performs automatic data type conversions, and is more scalable than previous versions.



**Figure 2**

## SSIS for data warehouse loading

At its core, SSIS is a comprehensive, fully functional ETL tool. Its functionality, scale, and performance compare very favorably with high-end competitors in the market at a fraction of their cost. The data integration pipeline architecture allows it to consume data from multiple simultaneous sources, perform multiple complex transformations, and then land the data to multiple simultaneous destinations. This architecture allows SSIS to be used not only for large datasets, but also for complex data flows. As the data flows from source(s) to destination(s), you can split, merge, and combine the stream of data with other data streams, and otherwise manipulate it. Figure 3 shows an example of such a flow.



**Figure 3**

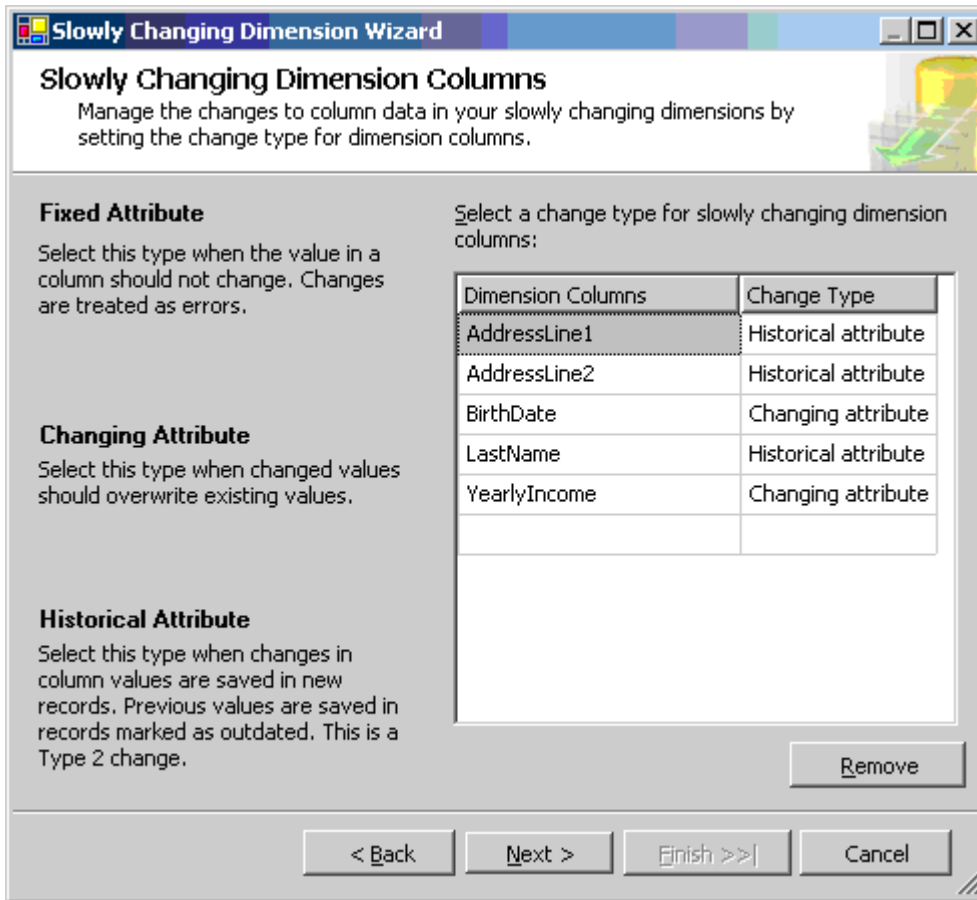
SQL Server 2008 includes support for Change Data Capture (CDC), which you can use to record insert, update, and delete activity in SQL Server tables, and make the details of the changes available in an easily consumed relational format. You can take advantage of CDC when implementing an ETL solution with SQL Server 2008 Integration Services to ensure that only changed data is included in the extraction process, which eliminates the overhead of performing a full data refresh that includes unchanged data in each ETL operation.

SSIS can consume data from (and load data into) a variety of sources including managed (ADO.NET), OLE DB, ODBC, flat file, Microsoft Office Excel®, and XML by using a specialized set of components called adapters. SSIS can even consume data from custom data adapters (developed in-house or by third parties) so you can wrap legacy data loading logic into a data source that you can then seamlessly integrate into the SSIS data flow. SSIS includes a set of powerful data transformation components that allow data manipulations that are essential for building data warehouses. These transformation components include:

- **Aggregate.** Performs multiple aggregates in a single pass.
- **Sort.** Sorts data in the flow.
- **Lookup.** Performs flexible cached lookup operations to reference datasets.
- **Pivot and UnPivot.** Two separate transformations pivot and unpivot data in the flow.

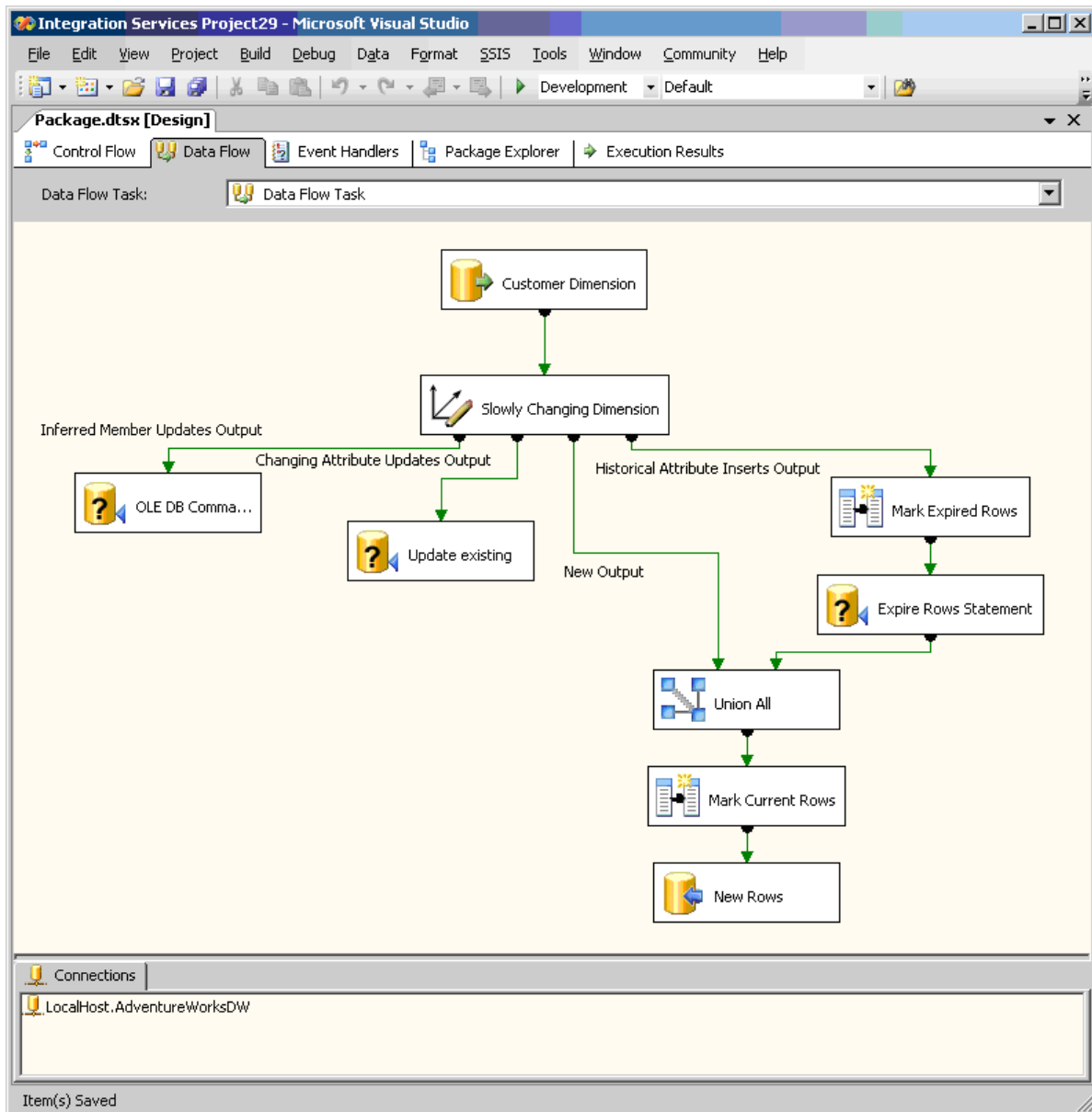
- **Merge, Merge Join, and UnionAll.** Can perform join and union operations.
- **Derived Column.** Performs column-level manipulations such as string, numeric, and date/time operations, and code page translations. This one component wraps what other vendors might break up into many different transformations.
- **Data Conversion.** Converts data between various types (such as numeric and string).
- **Audit.** Adds columns with lineage metadata and other operational audit data.

In addition to these core data warehousing transformations, SSIS includes support for advanced data warehousing needs such as Slowly Changing Dimensions (SCD). The SCD Wizard in SSIS guides users through specifying their requirements for managing slowly changing dimensions and, based upon their input, generates a complete data flow with multiple transformations to implement the slowly changing dimension load. Support for standard Type 1 and 2 SCD along with two new SCD types (Fixed Attributes and Inferred Members) is provided. Figure 4 shows a page from the SCD Wizard.



**Figure 4**

Figure 5 shows the data flow that is generated by this Wizard.



**Figure 5**

You can also use SSIS to load Analysis Services multidimensional OLAP (MOLAP) caches directly from the data-flow pipeline. This means that not only can you use SSIS to create relational data warehouses, but you can also use it to load multidimensional cubes for analytical applications.

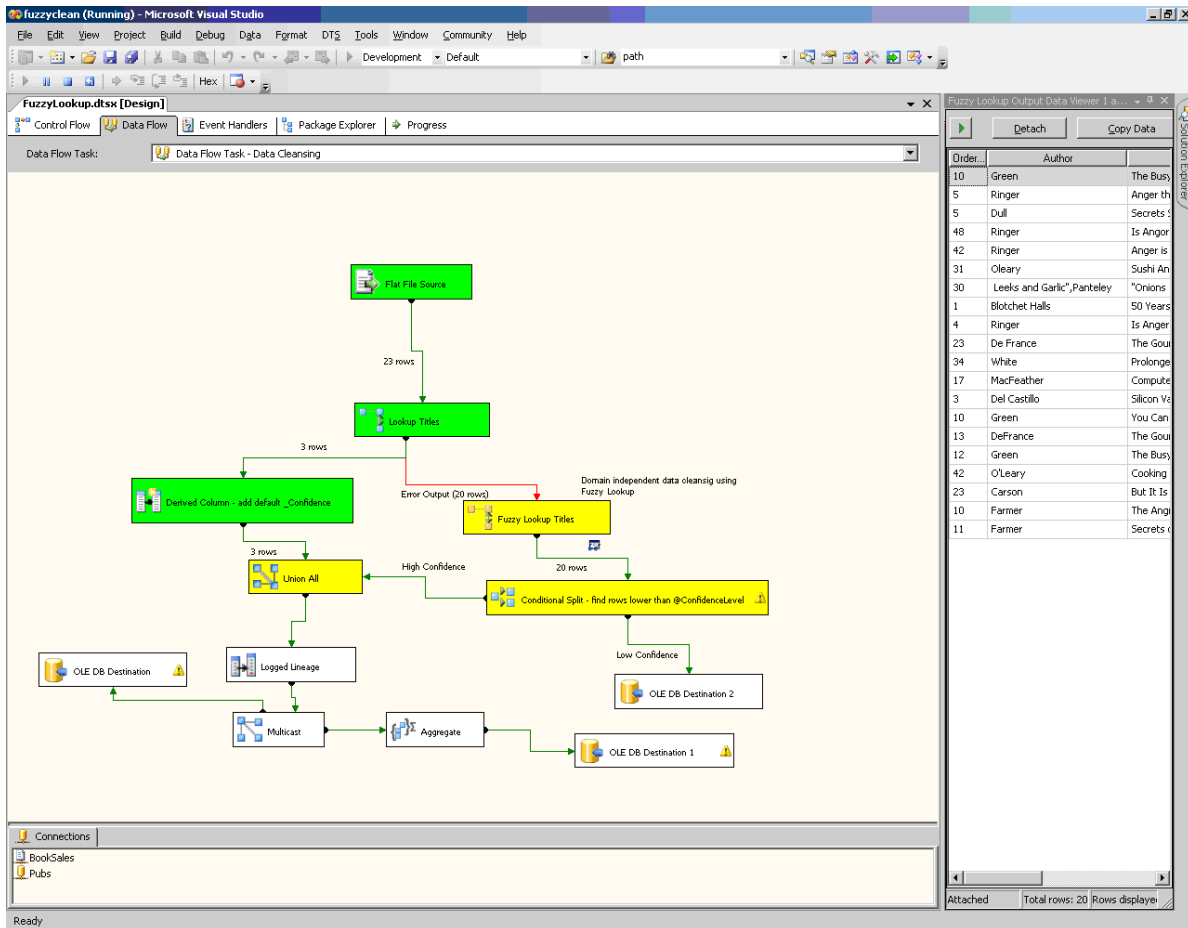
## SSIS and Data Quality

One of the key features of SSIS, as well as its ability to integrate data, is its ability to integrate different technologies to manipulate the data. This has allowed SSIS to include innovative “fuzzy logic”-based data cleansing components. The Microsoft Research labs developed these components and they represent the latest research in this area. The approach taken is a domain independent one and does not depend upon

any specific domain data, such as address/zip reference data. This allows you to use these transformations for cleansing most types of data, not just address data.

SSIS deeply integrates with the data mining functionality in Analysis Services. Data mining abstracts out the patterns in a dataset and encapsulates them in a mining model. You can then use this mining model to make predictions on what data belongs to a dataset and what data may be anomalous. So you can use data mining as a tool for implementing data quality.

Support for complex data routing in SSIS helps you to not only identify anomalous data, but also to automatically correct it and replace it with better values. This enables “closed loop” cleansing scenarios. Figure 6 shows an example of a closed loop cleansing data flow.



**Figure 6**

In addition to its built-in data quality features, SSIS can be extended to work closely with third-party data-cleansing solutions.

## Application of SSIS Beyond Traditional ETL

The ability of the data-flow pipeline to manipulate almost any kind of data, the deep integration with Analysis Services, the support for extending it with a large variety of data manipulation technologies, and the inclusion of a rich work-flow engine allow SSIS to be used in many scenarios that are not traditionally thought of as ETL

## Service Oriented Architecture

SSIS includes support for sourcing XML data in the data-flow pipeline, including data both from files on disk as well as URLs over HTTP. XML data is “shredded” into tabular data, which then can be easily manipulated in the data flow. This support for XML can work with the support for Web services. SSIS can interact with Web services in the control flow to capture XML data.

You can capture XML from files, from Microsoft Message Queuing (MSMQ), and over the Web via HTTP. SSIS enables the manipulation of the XML with XSLT, XPATH, diff/merge, etc. and can stream the XML into the data flow.

This support enables SSIS to participate in flexible Service Oriented Architectures (SOA).

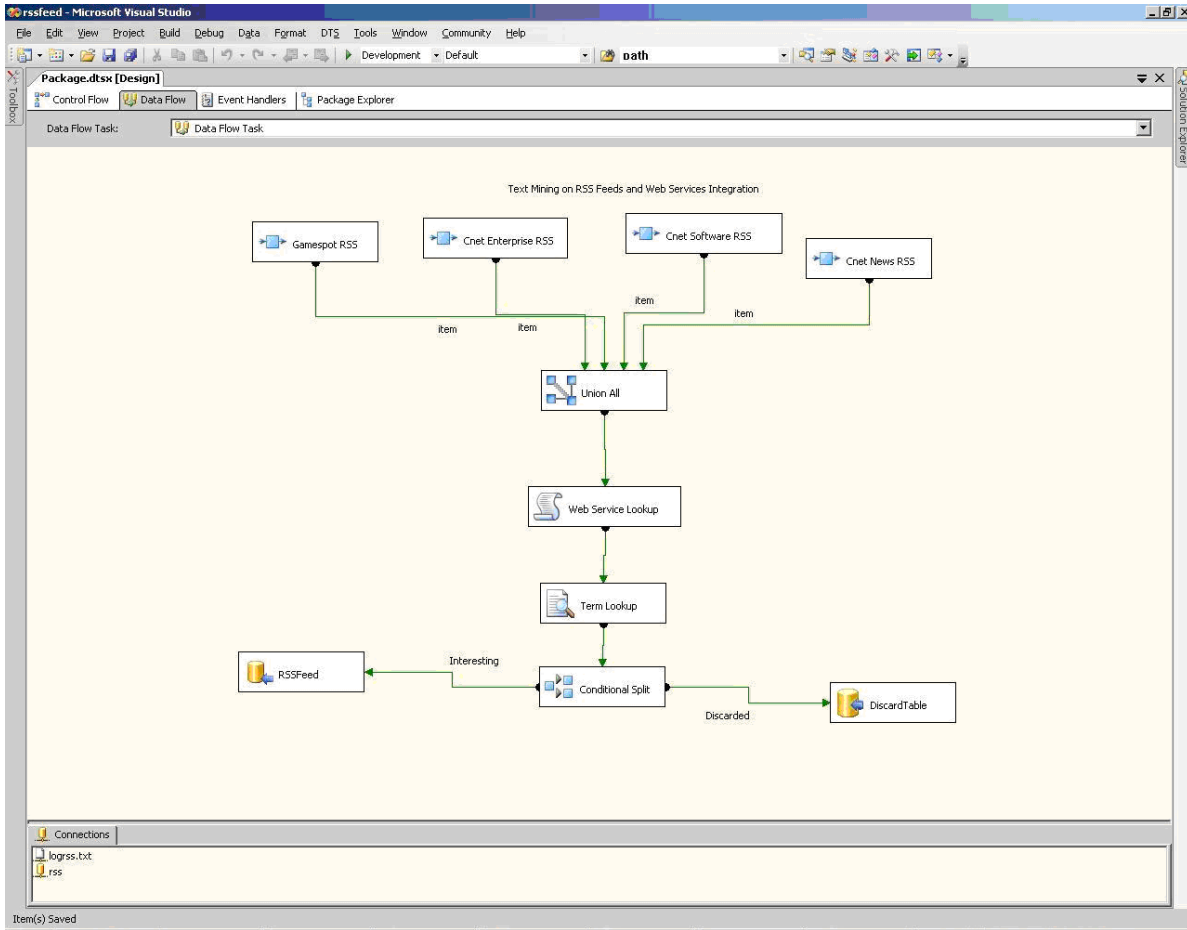
## Data and Text mining

SSIS not only has deep integration with the data mining features from Analysis Services, but it also has text mining components. Text mining (also referred to as text classification) involves identifying the relationship between business categories and the text data (words and phrases). This allows the discovery of key terms in text data and, based upon this, to identify text that is “interesting” automatically. This in turn can drive “closed-loop” actions to achieve business goals such as increasing customer satisfaction and enhancing the quality of the products and services.

## On-Demand Data Source

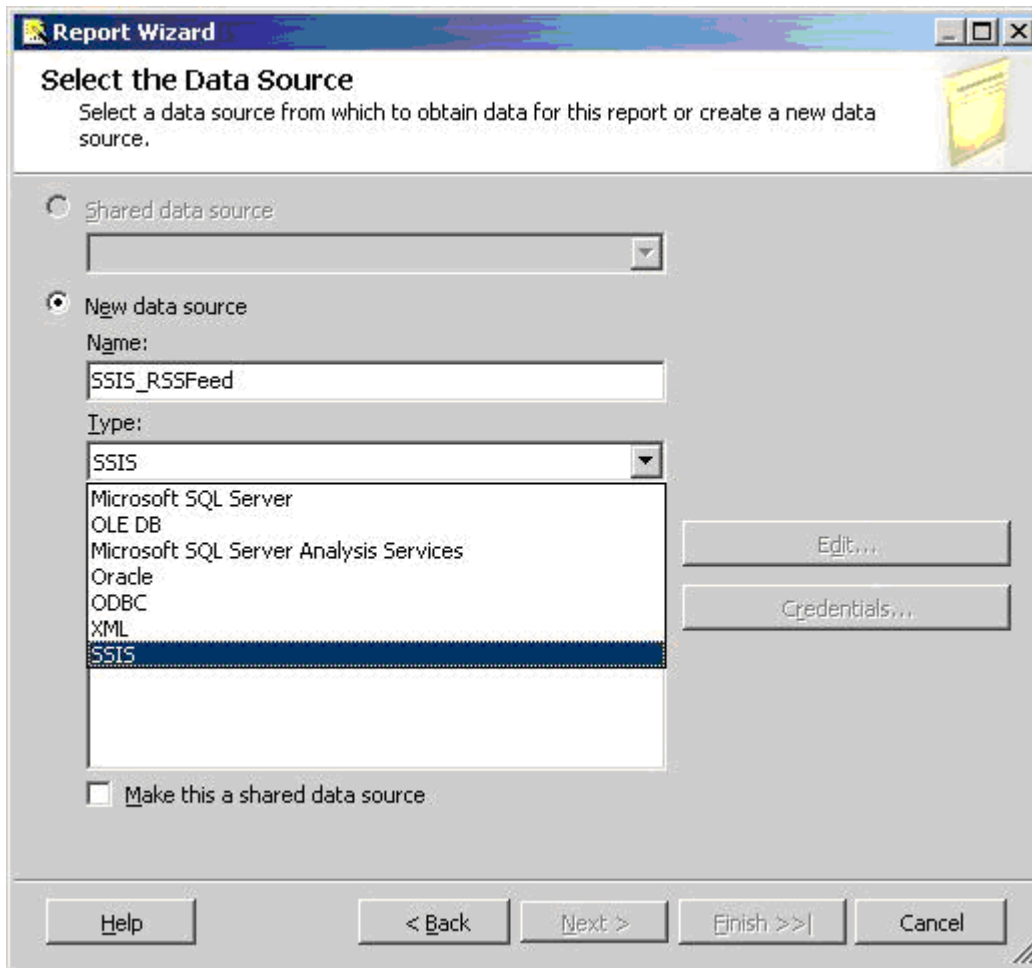
One of the unique features in SSIS is the DataReader destination, which lands data into an ADO.NET DataReader. When this component is included in a data-flow pipeline, you can use the package that contains the DataReader destination as a data source, exposed as an ADO.NET DataReader itself. So you can use SSIS not only as a traditional ETL to load data warehouses, but also as a data source that can deliver integrated, reconciled, and cleansed data from multiple sources on-demand. For example, you might use this to help Reporting Services to consume data from multiple diverse data sources by using a SSIS package as its data source.

A possible scenario that integrates all of these features consists of identifying and delivering interesting articles from RSS feeds as part of a regular report. Figure 7 shows a SSIS package that sources data from RSS feeds over the Internet, integrates with data from a Web service, performs text mining to find interesting articles from the RSS feeds, and then places the interesting articles into a DataReader destination to be finally consumed by a Reporting Services report.



**Figure 7**

Figure 8 shows the use of the SSIS package as a data source in the Report Wizard.



**Figure 8**

From an ETL tool perspective, this scenario is very unusual because there is no data extraction, transformation, or loading.

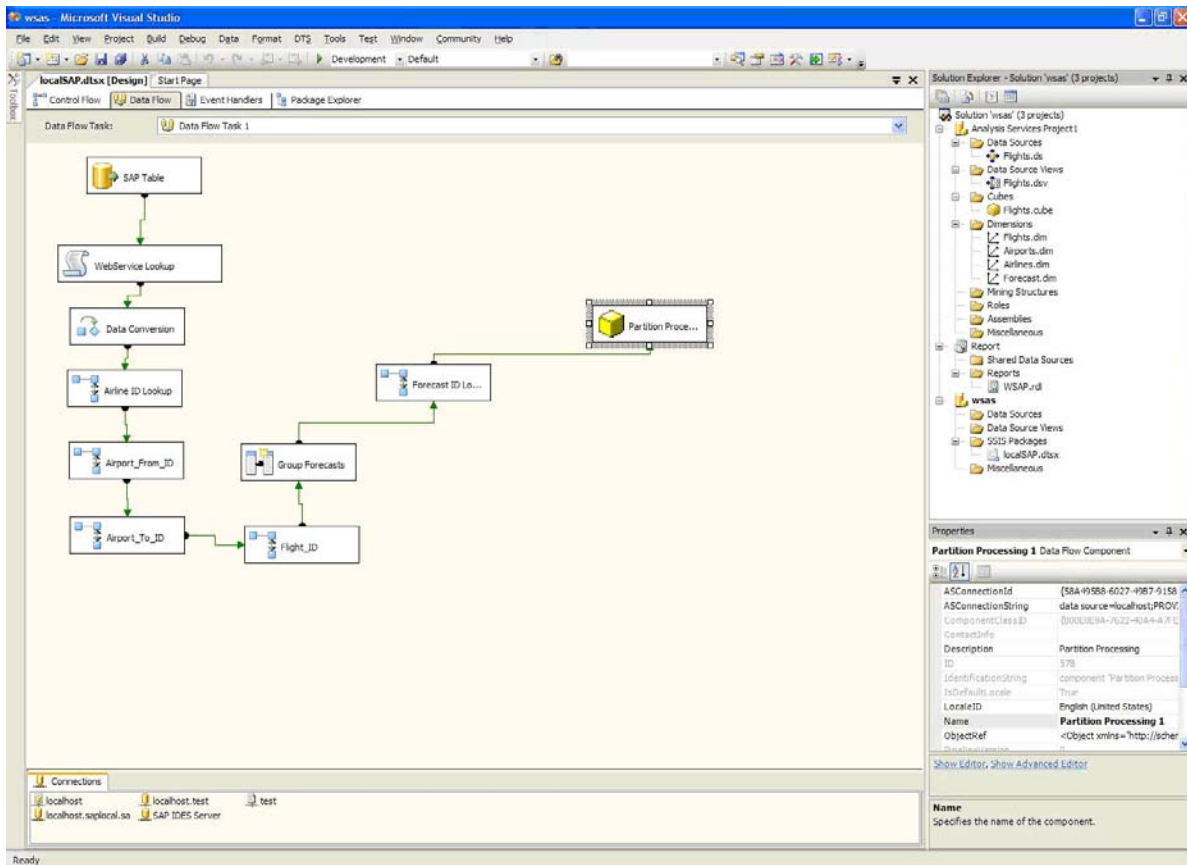
## SSIS, the Integration Platform

SSIS goes beyond being an ETL tool not only in terms of enabling nontraditional scenarios, but also in being a true platform for data integration. SSIS is part of the SQL Server Business Intelligence (BI) platform that enables the development of end-to-end BI applications.

## Integrated Development Platform

SQL Server Integration Services, Analysis Services, and Reporting Services all use a common Microsoft Visual Studio® based development environment called the SQL Server Business Intelligence (BI) Development Studio. BI Development Studio provides an integrated development environment (IDE) for BI application development. This shared infrastructure enables metadata-level integration between various development projects (integration, analysis, and reporting). An example of such shared construct is the Data Source View (DSV), which is an offline schema/view definition of data sources, and is used by all three BI project types.

This IDE provides facilities such as integration with version control software (e.g., VSS) along with support for team-based features such as “check-in/check-out” and as such it fulfills the need for an enterprise-class team-oriented development environment for business intelligence applications. Figure 9 shows a BI Development Studio solution that consists of Integration, Analysis, and Reporting projects.



**Figure 9**

Not only does this provide a single place to develop BI applications, but it also can be used to develop other Visual Studio projects (using Visual C#®, Visual Basic® .NET etc.) and so can provide developers with a true end-to-end development experience.

Besides an integrated BI development environment, BI Development Studio has features for true run-time debugging of SSIS packages. These include the ability to set breakpoints and support for standard development constructs such as watching variables. A truly unique feature is the Data Viewer, which provides the ability to view rows of data as Integration Services processes them in the data-flow pipeline. This visualization of data can be in the form of a regular text grid or a graphical presentation such as a scatter plot or bar graph. In fact, it is possible to have multiple connected viewers that can display the data simultaneously in multiple formats. Figure 10 shows an example of geographic data visualized using a scatter plot and a text grid.

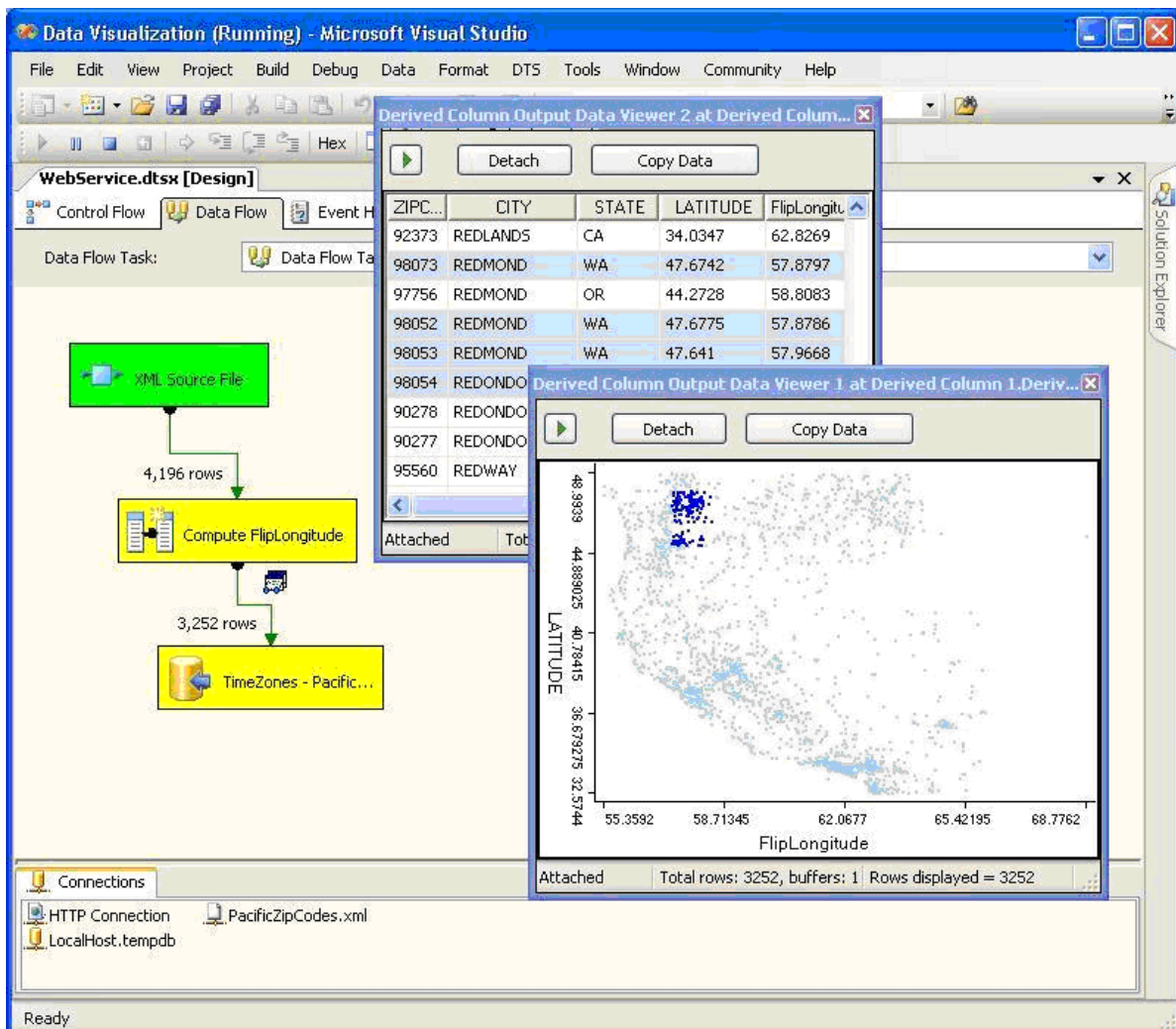


Figure 10

## Programmability

In addition to providing a professional development environment, SSIS exposes all its functionality via a set of rich APIs. These APIs are both managed (.NET Framework) and native (Win32) and allow developers to extend the functionality of SSIS by developing custom components in any language supported by the .NET Framework (such as Visual C#, Visual Basic .NET, etc.) and Visual C++. These custom components can be workflow tasks and data-flow transformations (including source and destination adapters). This allows legacy data and functionality to be easily included in SSIS integration processes, allowing you to use past investments in legacy technologies effectively. It also allows easy inclusion of third-party components.

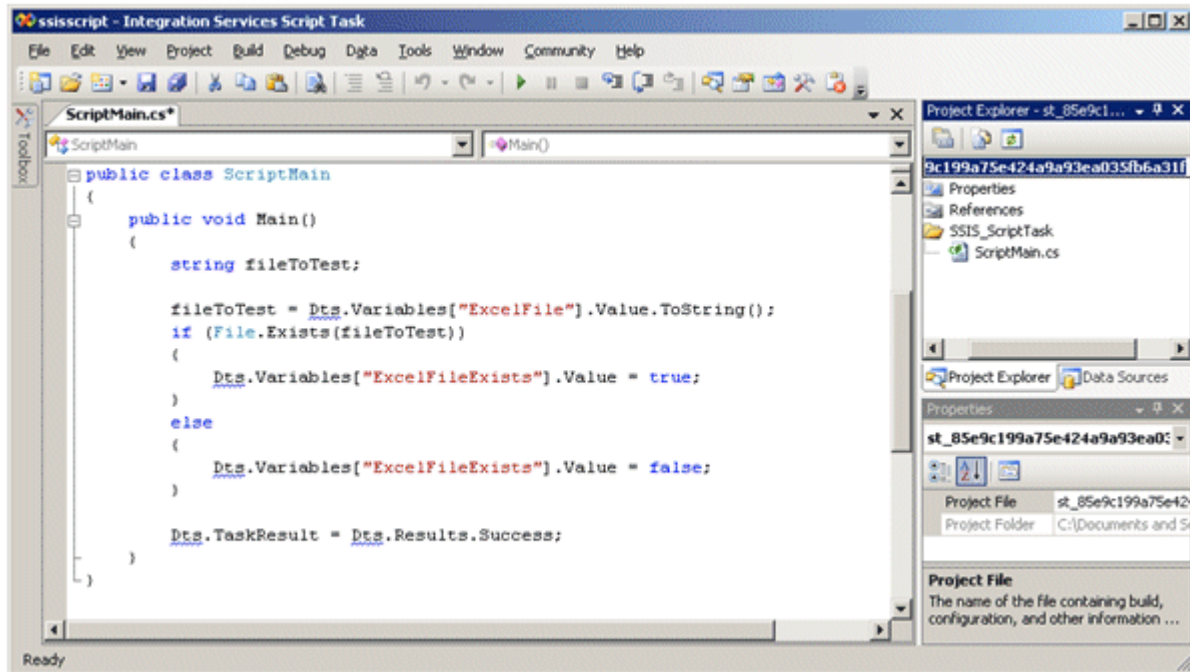
## Scripting

The extensibility previously mentioned is not only limited to re-usable custom components but also includes script-based extensibility. SSIS has script components both for task flow as well as for data flow. These allow users to write scripts in Visual

Basic .NET to add ad hoc functionality (including data sources and destinations) and to re-use any preexisting functionality packaged as .NET Framework assemblies.

SQL Server 2008 includes Visual Studio Tools for Applications, which provides a scripting environment in which you can use Visual Basic .NET, or C# to implement script components.

Figure 11 shows an example of a script that checks for the existence of an Office Excel file.

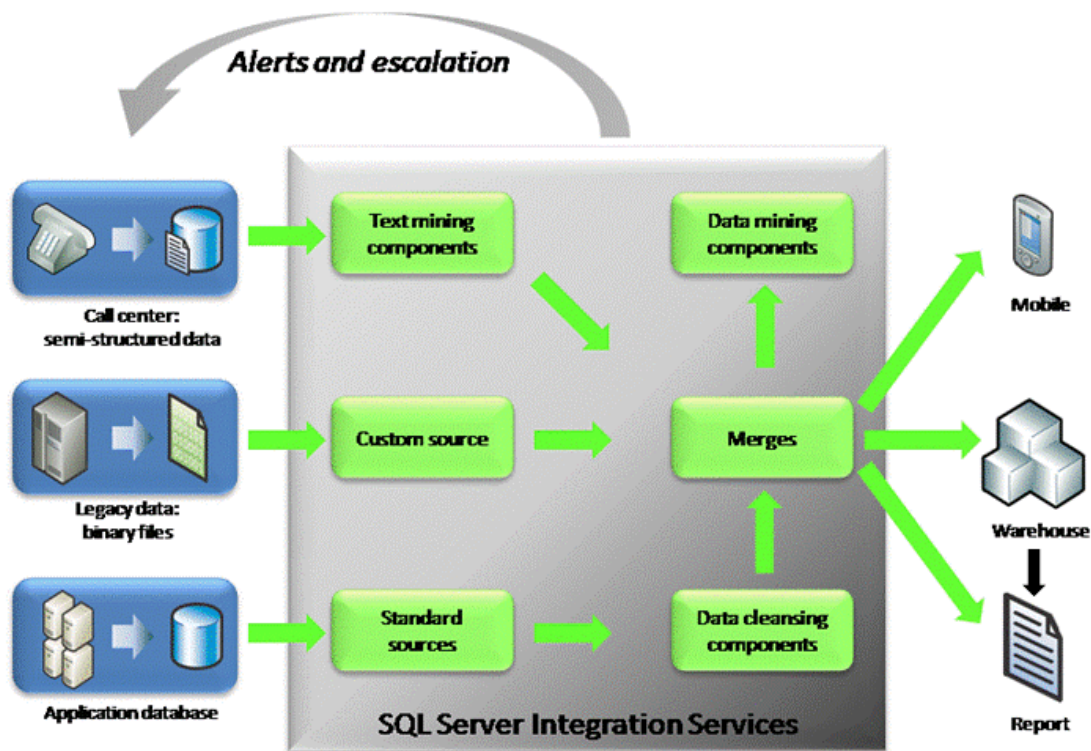


**Figure 11**

This extensibility model makes SSIS not only a data integration tool, but also an Integration Bus into which technologies like data mining, text mining, and Unified Dimensional Model (UDM) can easily be plugged in to enable complex integration scenarios involving pretty much arbitrary data manipulation and structures.

## Making Data Integration Approachable

The flexible and extensible architecture of SSIS allows it to address most of the technology challenges to data integration outlined earlier in this paper. As shown in Figure 12, SSIS eliminates (or at least minimizes) unnecessary staging. Because it performs complex data manipulation in a single pipeline operation, it is now possible to react to changes and patterns in the data quickly, in a timeframe that is meaningful for closing the loop and taking action. This is in contrast to traditional architectures that rely on data staging and that become impractical for closing the loop and taking meaningful action on data.



**Figure 12**

The extensible nature of SSIS makes it possible for organizations to leverage their existing investments in custom code for data integration by wrapping it as re-usable extensions to SSIS and by doing so to take full advantage of features such as logging, debugging, BI integration, etc. This greatly helps to overcome some of the organizational challenges outlined earlier in this paper.

The inclusion of SSIS in the SQL Server product makes the cost acquisition extremely reasonable as compared to other high-end data integration tools. Not only is the initial cost acquisition lowered, but also via tight integration with Visual Studio and the rest of SQL Server BI tools, the cost of application development and maintenance is significantly lowered in comparison to other similar tools. The extremely reasonable total cost of ownership (TCO) of SSIS (and the rest of SQL Server) makes enterprise-class data integration approachable to all segments of the market, taking it out of the exclusive domain of the largest (and richest) companies. At the same time, the architecture of SSIS is tuned to take advantage of modern hardware and to deliver performance and scale at the highest end of customer requirements. SSIS enables rich, scalable data integration to all customers, from the highest end enterprise to the small and medium business. In conjunction with the rest of the features in SQL Server, the Microsoft customer support infrastructure (ranging from broad, long beta testing, to rich online communities to premiere support contracts) and the consistency and integration with the rest of Microsoft product offerings, SSIS is truly a unique toolset that opens up new frontiers in data integration.

## Conclusion

Many businesses rely on data integration technologies to provide meaningful, reliable information to maintain a competitive advantage in the business world today. SQL Server 2008 Integration Services (SSIS) helps Information Technology departments to meet data integration requirements in their enterprises. SQL Server 2008 Integration Services meets the challenges of cleansing, transforming, and mapping multiple data sources with large volumes into a useful format. New features improve its ability to scale up and improve performance while speeding development and lowering the TCO.

**For more information:**

<http://www.microsoft.com/sql/technologies/integration/default.aspx>

Did this paper help you? Please give us your feedback. Tell us on a scale of 1 (poor) to 5 (excellent), how would you rate this paper and why have you given it this rating? For example:

- Are you rating it high due to having good examples, excellent screenshots, clear writing, or another reason?
- Are you rating it low due to poor examples, fuzzy screenshots, unclear writing?

This feedback will help us improve the quality of white papers we release. [Send feedback](#).