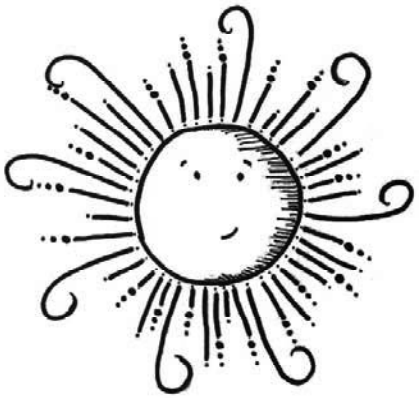

1

Introduction

by Tom Snyder

Simplicity is the ultimate sophistication.

—Leonardo da Vinci

In today's business environments, it's not unusual for application developers to build and maintain applications that draw from a diverse set of data sources. The business data that applications must process often resides in assorted places and platforms: Microsoft Excel spreadsheets, applications in the cloud, and database platforms such as IBM DB2, Microsoft SQL Server, and Oracle—to name just a few common examples.

Developers need a practical, reliable, and easily repeatable method for getting at that data: one that accesses the required data, performs the necessary database operations efficiently, and makes it available to the requesting application. Through experience, Vedish and I have come to rely on Microsoft SQL Server Information Services (SSIS) as our preferred tool to perform these tasks. We have found that SSIS enables easy, reliable data retrieval from multiple database platforms into business applications written in a variety of popular languages.

Why Use SSIS?

To write a book on a topic, you need to have a passion for that topic. We are software engineers and developers with experience in multiple environments, and

we have a passion for SSIS! When I first started working with SSIS, I was amazed at its capabilities, and we feel inspired to share this knowledge with other developers who could benefit from employing SSIS's powerful data-integration capabilities in their applications.

Why do I use SSIS? Primarily to support an ever-changing array of external applications and data formats. As a longtime programmer (Java, RPG, C#) who has written reusable code for years, what appealed to me about SSIS was its potential for reusability. Put SSIS in the mix of application development languages and tools, and you could be spending more time enhancing core systems instead of writing yet another custom system for another new format.

This book is intended for several audiences, including but not limited to developers, database administrators, data architects, reporting teams, testers, analysts and anyone involved with data warehousing, business intelligence, or repeatable data-gathering processes. During my career I've worked in various collaborative environments, but there was always a wall of knowledge between different development teams from the database-related departments. SSIS enabled us to bridge the gap and allow developers from across the organization—diverse teams working with different languages, applications, and database platforms—to enable communication among disparate data sources and applications using a common tool.

SSIS offers the following benefits for development teams:

Efficiency

By supplying the tools to graphically develop processes to perform common tasks, SSIS enables you to put together common database operations quickly. A key strength of SSIS is its ability to quickly return results by using repeatable processes that probably make up 90 percent of your work. This saves you the time and effort of having to write the same code repeatedly.

I think it's important to note that even though the SSIS method described in this book relies on the use of predefined components, understanding the low-level details of data access is still important and useful. But if you need a practical, fast way to perform data-access operations, the SSIS method will get the job done without requiring you to spend time repeating the mundane details and avoid the risk of introducing errors with new code.

Database Agnostic

SSIS supports any database that provides OLE DB and ODBC capabilities.

Support and Documentation

SSIS is a Microsoft product with enterprise-level service and support. There is an abundance of official Microsoft SSIS documentation online, and of course, now you have this book.

Availability

SSIS is part of Microsoft's SQL Server Data Tools (SSDT). Throughout my history of employment, SQL Server—which provides the components that you need to use SSIS—has always been running somewhere in the company. Thus, SSIS is widely available.

An SSIS Overview

Microsoft defines SSIS as "a platform for building enterprise-level data integration and data transformations solutions" (see *https://docs.microsoft.com/en-us/sql/integration-services/sql-server-integration-services*). It is a graphical extract, transform, and load (ETL) tool with predefined tasks that can be extended with your own code.

Mechanically, when you create an SSIS package you are simply creating an XML file that contains a list of instructions to be executed by the SSIS engine, just as Java or C# code is simply byte code that contains instructions to be executed by the Java Virtual Machine (JVM) or Common Language Runtime (CLR), respectively.

SSIS allows you to execute packages in different ways: using `DTExecUI`, `DTExec` command line, external execution utilities that use `DTExec`, SQL Server Agent, and SSDT.

DB2, Oracle, and Microsoft are the databases used for the examples in this book. However, you could also use any other database that supports OLE DB and/or ODBC, such as MariaDB, MySQL, and PostgreSQL.

OLE DB vs. ODBC

Open Database Connectivity (ODBC) was originally developed by Microsoft with Progress DataDirect. This was intended to make applications database agnostic by translating all the database-specific operations with the ODBC driver when communicating with the database. ODBC was adopted as the standard data-access interface, and most databases provide ODBC capabilities.

Object Linking and Embedding, Database (OLE DB) was also originally designed by Microsoft. OLE DB was intended to be the successor to ODBC to extend its capabilities beyond SQL and include additional enhancements.

OLE DB will be used throughout this book.

Microsoft publicly announced that OLE DB will no longer be supported and is deprecated as of Microsoft SQL Server 2012. (See *https://blogs.msdn.microsoft.com/sqlnativeclient/2011/08/29/microsoft-is-aligning-with-odbc-for-native-relational-data-access/*.) However, in August 2011 Microsoft also stated that the "OLE DB Provider for SQL Server is still supported on Denali for the following 7 years." (See *https://social.technet.microsoft.com/Forums/en-US/e696d0ac-f8e2-4b19-8a08-7a357d3d780f/microsoft-is-aligning-with-odbc-for-native-relational-data-access-faq?forum=sqldataaccess*.) In addition, OLE DB has been provided for Microsoft SQL 2016, which is what we are using in this book, so the standardization of everything to ODBC is still unknown.

Because of the Microsoft notifications that OLE DB was going to be deprecated, I was originally planning to use ODBC as the connection of choice throughout the book. But as I started going through developing the sample SSIS projects for the book, I quickly found that ODBC has limited capabilities in SSIS development, whereas OLE DB can do everything.

All that said, we will simply explore at what we need to develop our "database-agnostic" SSIS packages and will use for our examples the OLE DB providers that are supplied for Microsoft SQL Server, IBM DB2, and Oracle.

Technologies change rapidly, and at present, the fate of OLE DB appears uncertain. If Microsoft does truly deprecate OLE DB, it will probably be supported until the next version of Microsoft SQL Server is available. By that time, we should be writing another book on the latest version of Microsoft SQL Server. Perhaps by then, SSIS will be updated to fully support ODBC.

If you're following along with the book, you will be using SQL Server 2016 Express Edition, which is version 13. Table 1.1 shows a list of SQL Server versions since 2000.

Table 1.1: SQL Server Versions

Description	Release
SQL Server 2016	13
SQL Server 2014	12
SQL Server 2012	11
SQL Server 2008	10
SQL Server 2005	9
SQL Server 2000	8

If you are using a preexisting installation of Microsoft SQL Server and are unsure what version you are on, you could run the following SQL statement to find out:

```
select @@version
```

When I ran this statement on my machine, I received the following output. (Yours may be different depending on when you downloaded your version.)

```
Microsoft SQL Server 2016 (RTM-GDR) (KB3194716) - 13.0.1722.0 (X64)   Sep
26 2016 13:17:23   Copyright (c) Microsoft Corporation Express Edition
(64-bit) on Windows 8.1 Pro 6.3 <X64> (Build 9600: ) (Hypervisor)
```

Alternative ETL Tools

Microsoft is not the only provider of ETL tools. IBM and Oracle provide tools, and others are available. I wanted to include IBM InfoSphere DataStage (*https://www.ibm.com/us-en/marketplace/datastage*) and Oracle Data Integrator (ODI, *http://www.oracle.com/us/products/middleware/data-integration/enterprise-edition/overview/index.html*) in my initial product evaluations of existing ETL tools. However, neither vendor could provide me an acceptable express or evaluation version, whereas Microsoft makes SSIS available as a free download for anyone.

Goals of This Book

Our high-level objective with this book is to promote peace, love, and happiness among different development teams, with a data-access solution that fosters developer harmony and satisfies the needs of users and business requirements.

Teams and individual developers usually have strong preferences for languages, tools, and development methods. Users, too, have their preferred tools and methods that they use to do their jobs. Finally, the business' priorities must factor into whatever technology solution you choose. Is the cost of the software and resources most important? Is it the level of support? Or is it the solution's performance?

The SSIS solution explained in this book can satisfy all these needs in a pragmatic way. Rather than use a single development language over standardized database technologies—an ideal but utopian solution, our SSIS solution works in a typical real-world business environment where every application is written in a different language on a different database.

In this book, Vedish and I explain the components of the SSIS-based data-access solution, and how and why to use them. We're not employees of Microsoft, IBM, or Oracle; we're simply developers who are sharing the knowledge we've gained through years of experience. Just as Sir Isaac Newton did not create gravity, but simply observed its behavior and documented it, we are sharing our real-world experiences (both successes and less-than-successful efforts) with you, so you get the most value from our experiences and can apply them successfully in your own environment.

Summary

In summary, Microsoft SSIS is just another tool to have on hand for the right job at the right time. Which ties into the philosophy that if you have a chainsaw you could cut through almost anything, but should you? You wouldn't use a chainsaw to cut grass, but you could, I guess. Perspective is important.

For example, if performance is most important to you as a developer, you may take a different approach to implementing your data-access solution than this book describes, by writing all the ETL code in a high-level language (HLL) such as C# or Java. However, this method depends on the support of a developer who knows that HLL, whereas if you use SSIS, anyone from a novice to a karate-

kicking SSIS guru could provide your ETL solution, which could well be a much wiser use of your resources.

Further, using the SSIS method saves time and developer resources by creating a reusable solution. You simply click and drag repeatable tasks into easy-to-maintain SSIS packages with all the bells and whistles already built in, such as exception handling and logging, with a few clicks of the mouse.

As useful as SSIS is, we don't want to turn it into your "chainsaw"! Rather, we will show you how to use it in specific cases suitable for your appropriate ETL projects. Let's get started!