

Chapter 13

Synchronizing Mobile Data¹

In the last chapter, we covered managing and displaying data on the device using ADO.NET and SQL Server CE. In this chapter, we expand outward to retrieve that data from a desktop SQL Server and to post it back to the desktop after it has been modified on the device.

[\[Comment 13cs.1\]](#)

Introduction	2
Understanding Remote Data Applications	2
RDA and Merge Replication	3
IIS Connectivity	4
Database Connectivity	5
Installing Remote Data Connectivity	6
Creating the Virtual Directory	6
Configuring Additional Components	10
Using RDA.....	11
RDA Capabilities and Overhead	11
Programming for RDA.....	12
Pulling Data.....	12
Fixing the Identity Property	14
Viewing the Pulled Schema	15
Modifying Pulled Data Locally.....	15
Push	17
SubmitSQL.....	20
Using Merge Replication	21
Avoiding Synchronization Failures Through Good Design	22
Configuring Merge Replication.....	26
Programming for Merge Replication	26
Subscribe	28
Synchronize	29
Modify Data at the Device.....	31
Summary.....	32
Choosing Between Merge Replication and RDA	33
Conclusion	33
Chapter Appendixes.....	34
The Virtual Directory Creation Wizard	34
The Create Publication Wizard	39

¹ Some material in this chapter first appeared in an article which we wrote for MSDN Magazine: *SQL Server CE: New Version Lets You Store and Update Data on Handheld Devices*, MSDN Magazine, June 2001, Volume 16, Number 6. Reprinted with permission of MSDN Magazine.

Introduction

Three things make up the core of a typical SQL Server CE application scenario: a Windows CE device, some data to be stored, and a user who is most likely – though not necessarily – mobile. While scaled down from its desktop and server counterpart, SQL Server CE supports a very capable database engine that can easily support a single, standalone database. [\[Comment 13cs.2\]](#)

In addition, the SQL Server CE database engine has built-in support for sharing data between a locally-stored database and a remote SQL Server 2000 database. This support, and how you can use that support to move data between a SQL Server CE database and a SQL Server 2000 database, is the subject of this chapter. [\[Comment 13cs.3\]](#)

By itself, SQL Server CE is a single-user database engine. But it can participate in schemes to share a central database with other users through the support of the server-side database engine, SQL Server 2000. For example, a company might equip each member of its sales force with a Windows CE device. When a salesperson is visiting customers, a SQL Server CE database could operate as a standalone database engine. When a salesperson comes back to the office (or dials into the central system via modem or remote Internet connection), the device-based database can share updates with the central database, while also receiving updates back from the central server. Two elements are central to these types of scenarios: a central data store that is maintained by a SQL Server database, and a transport mechanism for moving the data between the Windows CE device and the SQL Server database. [\[Comment 13cs.4\]](#)

In this chapter we look at the .NET Compact Framework capabilities for moving, modifying and synchronizing data between a SQL Server 2000 on a desktop machine and a SQL Server CE on a device. [\[Comment 13cs.5\]](#)

Terminology

For the remainder of this chapter, we'll use "SQL Server CE" to designate SQL Server 2000 Windows CE Edition. Unless otherwise noted, all references to SQL Server refer to the versions of SQL Server - SQL Server 6.5, SQL Server 7.0, and SQL Server 2000 - that run on Windows NT and Windows 2000. These, incidentally, are the versions of SQL Server that are compatible with SQL Server CE. [\[Comment 13cs.6\]](#)

Understanding Remote Data Applications

.NET Compact Framework applications that access common data stored in a desktop SQL Server are characterized by the four capabilities, two methods, and one connectivity mechanism that .NET Compact Framework provides; and by an overwhelming desire to avoid errors rather than handle them. [\[Comment 13cs.7\]](#)

.NET Compact Framework applications can

- 1 Retrieve data from a SQL Server database into a SQL Server CE database on a device. [\[Comment 13cs.8\]](#)
- 2 Add to, remove, or modify that data at the device *and have SQL Server CE track those changes*. [\[Comment 13cs.9\]](#)
- 3 Have SQL Server CE submit the changed data to the SQL Server and receive any exceptions resulting from failed updates on the server. [\[Comment 13cs.10\]](#)
- 4 Submit SQL statements directly from the Compact Framework application to a SQL Server database. [\[Comment 13cs.11\]](#)

Activities #1 through #3 are related; activity #4 is independent of the other three. One method that supports activity #4, the ADO.NET provider classes for SQL Server, was covered in the previous chapter; a second, which uses the same classes and connectivity as replication will be covered in this chapter. [\[Comment 13cs.12\]](#)

An application that synchronizes data between SQL Server and SQL Server CE does so by retrieving data from SQL Server, accepting data from the user, having SQL Server CE determine what data needs to be sent back to SQL Server, and having SQL Server CE send that data. It is the SQL Server CE Client Agent component running on the device, in conjunction with the SQL Server CE Server Agent and the SQL Server engine running at the server, which retrieves data from SQL Server and subsequently determines which user entered data represents changes to the original and propagates those changes back to SQL Server. [\[Comment 13cs.13\]](#)

The two methods provided by .NET Compact Framework in conjunction with the Win CE and SQL Server for doing this transfer of data are *Remote Data Access* (RDA), which provides all four capabilities listed above and connects the SQL Server versions 2000, 7.0, and 6.5; and *Merge Replication*, which does not support direct submission of SQL statements and only connects with SQL Server 2000. [\[Comment 13cs.14\]](#)

Regardless of the method used, the connection between SQL Server CE and the desktop SQL Server is established using Internet-standard protocols. The server-side of the connection requires that you configure a website and install a SQL Server CE server side component at that website. That component sends/receives HTTP, translates the HTTP into SQL Server commands, connects to the SQL Server, submits the commands, and returns the results to SQL Server CE on the device. [\[Comment 13cs.15\]](#)

The device-side of the connection can be established using any of several methods, providing that the end result is a TCP/IP connection to the website that hosts the connection to the central database. For example, a device could be connected to a desktop system using ActiveSync 3.5 which has built-in support for providing network access to a connected device. With earlier versions of ActiveSync, SQL Server CE provides its own bridging through a component called *SQL Server CE Relay*. A device can also connect to a server using a dedicated wired or wireless network card. A connection can be accomplished using a modem for servers which are visible through the Internet, or a modem with a virtual private network (VPN) connection for servers that are protected behind a network firewall. [\[Comment 13cs.16\]](#)

Throughout this book we have talked about handling errors. But in the remote data environment it is always best to avoid errors rather than handle them. In a sense you cannot handle them, for “what SQL Server 2000 says, goes”. If SQL Server 2000 rejects your changes, you must undo those changes in the SQL Server CE database or re-synch with SQL Server 2000, and notify the user. When this happens, you need to make new changes to the data - ones that can be accepted by the server – and resubmit those changes. We address the topic of avoiding synchronization errors later in this chapter. [\[Comment 13cs.17\]](#)

RDA and Merge Replication

Two separate mechanisms are provided to move data and schema between SQL Server CE and SQL Server: Remote Data Access (RDA) and Merge Replication. Both are used in conjunction with a web server based on Microsoft’s web server technology, *Internet Information Services* (IIS). The complex part of any SQL Server CE application is the transferring of data

between SQL Server CE and SQL Server, especially if you have never used replication in your development, or developed IIS-based applications. [\[Comment 13cs.18\]](#)

One of the design benefits of both RDA and Merge Replication is that they move both data and also schema. This transfer of data and schema can be as broad as an entire database, or as narrow as a subset of a table. This control of schema resides on the SQL Server, and in most cases eliminates the need for explicit data definition statements in SQL Server CE applications altogether. [\[Comment 13cs.19\]](#)

The primary benefit of RDA is that it has a smaller footprint on the device and is somewhat faster due to the reduced functionality and tracking. The key disadvantages of RDA are that it: [\[Comment 13cs.20\]](#)

- 1 Requires more code; both ADO.NET code as well as SQL code. [\[Comment 13cs.21\]](#)
- 2 Provides little support for handling conflicts. [\[Comment 13cs.22\]](#)
- 3 Is not an established database interface, being used only in SQL Server CE. [\[Comment 13cs.23\]](#)

Merge replication, on the other hand, can be setup by a database administrator, requires very little code, and allows for the establishment of database update rules, including rules for conflict resolution when synchronizing new data into an existing SQL Server 2000 database. While, RDA is supported for all versions of SQL Server, Merge Replication requires SQL Server 2000. Table 13-1 summarizes the data replication support provided for different versions of SQL Server. [\[Comment 13cs.24\]](#)

Table 13-1 Server Data Replication Support [\[Comment 13cs.25\]](#)

SQL Server Version	Remote Data Access (RDA)	Merge Replication
SQL Server 6.5	X	
SQL Server 7.0	X	
SQL Server 2000	X	X

IIS Connectivity

There were several reasons that the SQL Server CE team chose to use the Internet Information Server as its primary conduit. A very important segment of the user base consists of mobile users. It was important to allow the connectivity options to be as widespread as possible. By building SQL Server CE around HTTP connectivity, wherever someone can gain access to an Internet connection to be able to browse the web, they can also connect to their SQL Server home world. In addition to its widespread availability, an IIS connection also provides authentication and authorization so that a user's data can get through a corporate firewall. And finally, IIS provides the ability to encrypt data when using a certificate – so private data can remain private. [\[Comment 13cs.26\]](#)

When data is transmitted between SQL Server CE and SQL Server, it is automatically compressed for its trip between the CE device and the HTTP site. In one benchmark, using the infamous Northwind database, the amount of compression observed was approximately 8x. [\[Comment 13cs.27\]](#)

While compression is always enabled, encryption is not. You can, however, enable encryption of the data being transferred between SQL Server CE and the IIS Server by using the standard encryption capability of IIS. This encryption is entirely transparent to the two database engines. [\[Comment 13cs.28\]](#)

ActiveSync and SQL Server CE

No discussion of connectivity to a Windows CE device would be complete without some mention of ActiveSync. ActiveSync describes a mechanism for moving data between a Windows workstation and a Windows CE device. For example, it is the primary mechanism for moving email messages, appointments, task lists, and contact information between a Windows CE device and a desktop. It's important to be aware that SQL Server CE does not use ActiveSync for the transmission of data. But when installing SQL Server CE on a device like a PocketPC or a Handheld PC, ActiveSync must be present to assist in this operation. [\[Comment 13cs.29\]](#)

Database Connectivity

While IIS is used for data transmission, SQL Server CE components are used at both ends of the transmission for database connectivity. SQL Server CE has two components that are used for this: *SQL Server CE Client Agent* runs on the device, and *SQL Server CE Server Agent* runs on the desktop. Thus, five components are involved in the transfer of data between SQL Server and SQL Server CE. (See figure 13-1) [\[Comment 13cs.30\]](#)

On the device
 SQL Server CE Engine
 SQL Server CE Client Agent

On the server(s)
 SQL Server CE Server Agent
 IIS Server
 SQL Server

The three server-side components can be deployed with some amount of flexibility. In particular, IIS and SQL Server 2000 can either be deployed on the same server system, or on separate machines. The SQL Server CE Server Agent component, however, must be deployed on the same system with IIS. [\[Comment 13cs.31\]](#)

Figure 13-1 – The Five Components of Remote Data Access

[Hand drawing. To be submitted. Dave.]

The SQL Server CE Client Agent is a device-side component that tracks changes made to data on the device and, when instructed to do so, submits them to the SQL Server CE Server Agent. Note that the track change support of SQL Server CE is not the same as the track change support for ADO.NET data sets. [\[Comment 13cs.32\]](#)

The SQL Server CE Server Agent is a server-side component that retrieves changes from the client agent, and connects to the SQL Server 2000 database on behalf of the device. The server agent combines three different roles within one component. [\[Comment 13cs.33\]](#)

1. An ISAPI² extension DLL, an extension to a web server running Microsoft's IIS [\[Comment 13cs.34\]](#)
2. Its name is part of a URL, which provides the means by which the client agent and the server agent are able to find each other [\[Comment 13cs.35\]](#)
3. A SQL Server client program, because it connects to and interacts with a SQL Server 2000 database [\[Comment 13cs.36\]](#)

² ISAPI stands for Internet Server Applications Programming Interface, a mechanism for extending the functionality of IIS.

Because it is an ISAPI extension DLL, the SQL Server CE Server Agent must be copied to a website and also registered with that website. Because the name of the server agent is part of a URL, you need to specify that URL in your application. Because it is a SQL Server Client program, you may need to provide SQL Server logon information in your application. And because the transfer of data to and from SQL Server is done via IIS, your request must pass through both IIS security and SQL Server security. All of this has an impact on how you configure the components on the server side. [\[Comment 13cs.37\]](#)

Enhanced Security and SQL Server CE Server Agent

As part of the effort to improve security, version 6.0 of Internet Information Services has a different set of default settings from previous versions. This version, which ships with Windows Server 2003, starts up with ISAPI extensions disabled. For the SQL Server CE Server Agent to operate properly, you must enable these extensions. This is done from a logon account with administrator privileges through the server's Administrative Tools. Open Internet Information Services and open the *Web Service Extensions* node. [\[Comment 13cs.38\]](#)

During this chapter we cover how to install and configure the various components required for remote data access. It is not the intent of this chapter to make you an IIS administrator or a SQL Server administrator. It is, instead, the intent of this chapter to provide you with enough setup information so that you can setup a development environment on your desktop machine at work or at home. [\[Comment 13cs.39\]](#)

It might seem strange that you need to install SQL Server CE for this chapter, but did not need to do so for the previous chapter. You do not need to install SQL Server CE for local data access because SQL Server CE is automatically installed on a device along with an application whenever an application contains a reference to the `System.Data.SqlServerCe.dll` assembly. But SQL Server CE Server Agent is not automatically installed on your IIS web server system. You, or an administrator, must do that installation yourself; a subject that we cover next. [\[Comment 13cs.40\]](#)

Installing Remote Data Connectivity

People familiar with IIS recognize that the device-to-IIS connection can be made in Anonymous, Basic Authentication, or Integrated Windows Authentication mode; people familiar with SQL Server 2000 know that the IIS to SQL Server connection can be made in Windows Authentication or SQL Server authentication mode. All this results in a connectivity / security chain that involves your SQL Server CE application, the network, the operating system, IIS, NTFS, SQL Server, and, in the case of merge replication, a shared directory. [\[Comment 13cs.41\]](#)

Creating the Virtual Directory

As stated above, there are three security contexts that IIS exposes that can be used with SQL Server CE (Anonymous, Basic and Integrated). Of the three, Integrated is the most restrictive when using SQL Server CE because it requires having IIS and SQL Server on the same machine. This is because NTLM does not support proxying security credentials through machines. Kerberos does support this in Windows 2000, but support for Kerberos is not supported on Windows CE devices. Like any web based application, if Basic authentication is chosen, it is recommended that SSL be used to prevent the username and password from being deciphered from the device and the IIS box. [\[Comment 13cs.42\]](#)

Before you begin to code the application, there are a number of standard web development issues to address. You must decide the following: Where will the ISAPI EXTENSION DLL be located? Where will the SQL Server be located? Will there be a firewall or proxy server between them? What OS user should the Server Agent execute as? How do you ensure that that OS user is a valid SQL Server login? Should there be multiple sites, each with their own security; or should there be just one site, one connection point for all the applications?

[\[Comment 13cs.43\]](#)

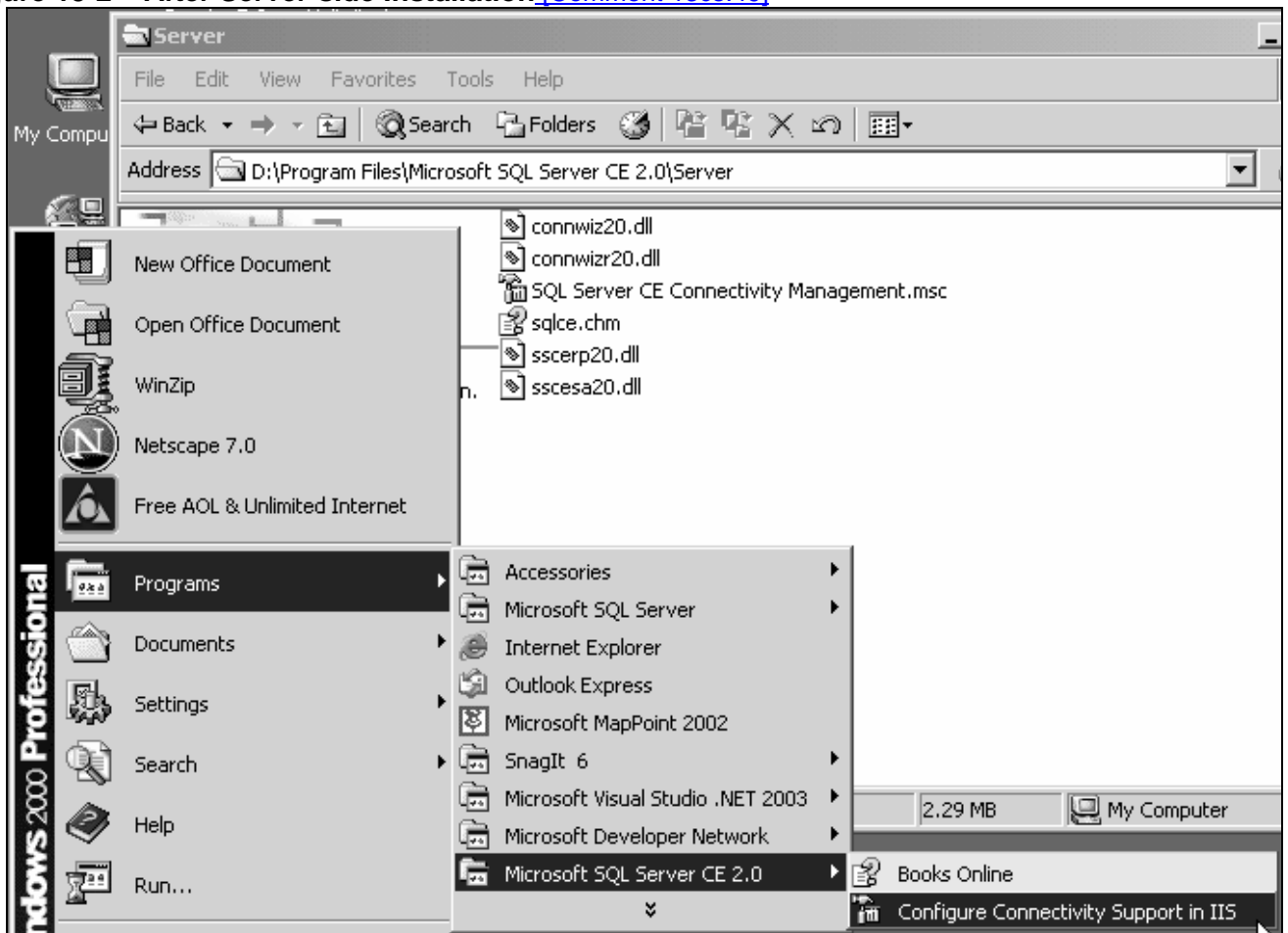
These questions include the typical setup procedures for doing any web based application that also has backend database support (setting up a virtual directory, installing the server side components in the virtual directory, choosing the authentication modes to the virtual directory, assigning user rights and setting up the necessary security and access rights to gain access to SQL Server). It is not our intent to repeat the details for doing this that can be found in the SQL Server CE Books Online. Rather, we'll take a high level look at the SQL Server CE application to IIS to SQL Server chain. [\[Comment 13cs.44\]](#)

Fortunately, SQL Server CE provides a tool for creating and configuring the web server to be accessed by client devices. And that means that you must install the server-side component of SQL Server CE and then use it to install the IIS site(s). Books Online refers to this as "Installing SQL Server CE." But it is really installing the Server-side components of SQL Server CE. [\[Comment 13cs.45\]](#)

If you are doing your initial development in a single machine environment at work or at home, and if you have already installed Visual Studio .NET on your machine, then you already have the SQL Server CE Server Agent on that machine. It is not yet installed, but it is present on your system. To install it, go to the Visual Studio install folder, by default `\Program Files\Microsoft Visual Studio .NET 2003\CompactFrameworkSDK\vN.N.N`, in the `Windows CE` subdirectory you can find the self extracting Zip file `sqlce20sql2ksp2.exe`. (The name of the file denotes the SQL Server 2000 service pack that must be installed for this version to work correctly.) Run it and it installs the SQL Server CE Server Agent and some tools. [\[Comment 13cs.46\]](#)

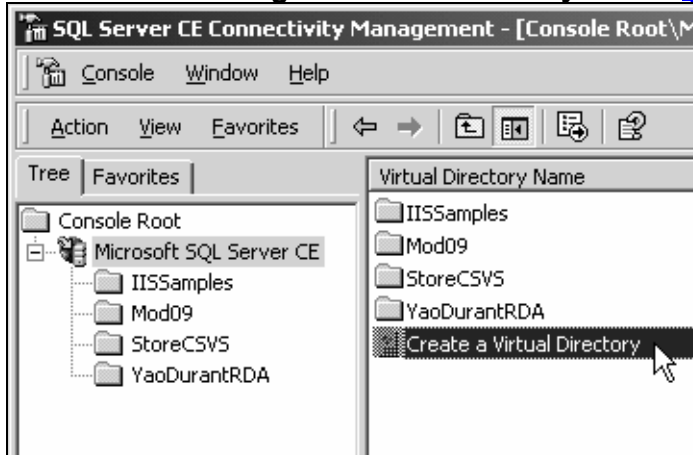
If you do not have Visual Studio .NET installed then you can install the SQL Server CE Client Agent from a SQL Server CE CD. On the SQL Server CE CD that came with our MSDN subscription, for instance, `setup.exe` is located at `\ENGLISH\SQLCE11`. [\[Comment 13cs.47\]](#)

Regardless of how you install SQL Server CE on the server machine, the `\Program Files\Microsoft SQL Server CE 2.0\Server` directory will contain a file named `sscesa20.dll`. This file is the SQL Server CE Server Agent. It is shown in figure 13-2, along with the Start menu entry that was added by the installation of SQL Server CE. [\[Comment 13cs.48\]](#)

Figure 13-2 – After Server-side Installation [\[Comment 13cs.49\]](#)

But installing the Server Agent does not yet mean that you can transfer data between the device and the desktop. Because this transfer is done via the Internet, you must establish one or more sites on your IIS machine for use by SQL Server CE, and into each site you must install the `sscesa20.dll`, and you must set the appropriate permissions for that site. [\[Comment 13cs.50\]](#)

The program that helps you do this, the SQL Server Connectivity Management program, is reached from the Start Menu entry shown in Figure 13-2. This management application then allows you to reach a wizard, the *Virtual Directory Creation Wizard* which helps you configure the site. To configure a new site from the SQL Server Connectivity Management program, select Microsoft SQL Server CE in the left panel and click on Create a Virtual Directory in the right panel, as shown in figure 13-3. To modify an existing directory configuration, select the directory from the right panel. [\[Comment 13cs.51\]](#)

Figure 13-3 - Launching the Virtual Directory Wizard [\[Comment 13cs.52\]](#)

The wizard walks you through five dialog boxes. These dialog boxes are quite self explanatory. We have chosen to place them at the end of this chapter, in the *Virtual Directory Creation Wizard* section, as they require several pages and would interrupt the flow of the chapter if shown here. But we do want to make them available for readers who are visually oriented and who “learn by seeing”. [\[Comment 13cs.53\]](#)

Through the wizard you specify the name and location of the virtual directory; chose the authentication method; and optionally configure and share the snapshots directory that is needed for merge replication. The wizard creates the web site and virtual directory, copies the `sscesa20.dll` file to it, and registers `sscesa20.dll`. It does not configure NTFS permissions, add domain users and groups, add SQL Server logons, or generate code. Those tasks must be done by you, or an appropriate administrator, after analyzing the needs of the site. [\[Comment 13cs.54\]](#)

For instance, the need to provide read-only access to public data usually results in: [\[Comment 13cs.55\]](#)

1. A site that allows anonymous access [\[Comment 13cs.56\]](#)
2. A SQL Server logon for the IUSR_MachineName user [\[Comment 13cs.57\]](#)
3. A SQL Server that allows SQL Server logon authentication [\[Comment 13cs.58\]](#)
4. Application code that does not specify and of the change-tracking options. [\[Comment 13cs.59\]](#)

While authenticated access to corporate data might be accomplished with: [\[Comment 13cs.60\]](#)

1. A site that uses Integrated Windows authentication and SSL [\[Comment 13cs.61\]](#)
2. A new SQL Server logons group [\[Comment 13cs.62\]](#)

When you have finished configuring a website you can test it by opening your browser and navigating to `HTTP://server name/virtual directory name/sscesa20.dll`, as shown in Figure 13-4. If you receive the response shown in Figure 13-4 then your site installation was successful. [\[Comment 13cs.63\]](#)

Figure 13-4 - After Configure Test [\[Comment 13cs.64\]](#)

If you specified a security model that required authentication and if you were not currently logged on when you navigated to the site, you were asked to enter your user name and password. Whatever URL provides access to the server agent, like that shown in Figure 13-4, is the URL that you specify in your application code. Whatever user name and password you specified to log on is the user name and password that you specify in your application code. [\[Comment 13cs.65\]](#)

If you are unable to browse to the site from your device, try browsing from the server machine, using `localhost` as the server name. If this local access attempt fails, then the problem lies in the web-server configuration. Return to the SQL Server Connectivity Management program and check the properties and spellings of your configuration. [\[Comment 13cs.66\]](#)

If local access succeeds, then the problem lies in configuring your device. From your device, try to browse to a known site, such as `www.microsoft.com`. If that fails, you have a general network connectivity problem. If it succeeds, then the problem lies in connecting your device specifically to your server. [\[Comment 13cs.67\]](#)

Configuring Additional Components

The IIS authentication configuration that you specify determines the user name that SQL Server CE Server Agent uses when it connects to SQL Server and when it accesses the shared directory required by Merge Replication. This user must be able to log onto SQL Server and to access the shared directory, which in turn must be shared. Your application is ready to connect when you have accomplished all of the following: [\[Comment 13cs.68\]](#)

- All OS users/groups, if any, have been added [\[Comment 13cs.69\]](#)
- All SQL Server logons/groups have been added [\[Comment 13cs.70\]](#)
- The directory has been shared [\[Comment 13cs.71\]](#)
- Share permissions have been granted [\[Comment 13cs.72\]](#)

Using RDA

RDA Capabilities and Overhead [\[Comment 13cs.73\]](#)

RDA enables a SQL Server CE application to: [\[Comment 13cs.74\]](#)

- 1 Retrieve data and schema from a SQL Server database into a SQL Server CE database on a device. [\[Comment 13cs.75\]](#)
- 2 Add to, remove, or modify that data at the device and have SQL Server CE track those changes. [\[Comment 13cs.76\]](#)
- 3 Have SQL Server CE submit the changed data to the SQL Server and receive any exceptions resulting from failed updates on the server. [\[Comment 13cs.77\]](#)
- 4 Submit SQL statements directly from the Compact Framework application to a SQL Server database. [\[Comment 13cs.78\]](#)

RDA is simpler than merge replication, but it requires that you write more code; both C# code and SQL code. It also requires that you use an ADO.NET class that was not needed in the previous chapter when we accessed a SQL Server database: the `SqlCeRemoteAccess` class. [\[Comment 13cs.79\]](#)

RDA uses a concurrency model that the online documentations call “Optimistic Concurrency.” The meaning of optimistic concurrency in RDA is different from the meaning that a lot of veteran SQL Server programmers are used to. In RDA, Optimistic Concurrency means: [\[Comment 13cs.80\]](#)

1. Pull data from the server to the device [\[Comment 13cs.81\]](#)
2. Modify the data on the device [\[Comment 13cs.82\]](#)
3. Submit data changes to the server [\[Comment 13cs.83\]](#)
4. Changes are made even if the data on the server has been modified since it was pulled to the device. [\[Comment 13cs.84\]](#)

So, if you are used to Optimistic Concurrency meaning: [\[Comment 13cs.85\]](#)

1. Retrieve data into the application [\[Comment 13cs.86\]](#)
2. Modify data within the application [\[Comment 13cs.87\]](#)
3. Submit changes to the server [\[Comment 13cs.88\]](#)
4. Changes are *not* made if the data on the server has been modified since it was read by the application. [\[Comment 13cs.89\]](#)

Adopt a new mindset. [\[Comment 13cs.90\]](#)

For instance, the following RDA scenario causes one update to overlay another. [\[Comment 13cs.91\]](#)

08:00	Client A pulls data from the server
08:00	Client D pulls data from the server
12:00	Client A updates row X on device A
12:00	Client B updates row X on device B
18:00	Client A pushes updates to server
19:00	Client B pushes updates to server

Thank You

Thank you for taking the time to read this preview chapter. We hope it has provided you insights and tips to help with your Compact Framework programming project. You can help us create a better book by clicking the comment link at the end of each paragraph, and sending your comments and suggestions on our review web site.

Preview Chapter Text

Our public review site provides the complete table of contents for the Compact Framework book at this link: <http://www.paul Yao.com/cfbook.htm>. That table of contents contains links to all the preview chapters. The preview chapter provides the complete outline of topics covered in a chapter, and also the first section or two from each chapter.

Complete Chapter Text

You can get the complete text for each chapter, available to readers who register at our web site. Registration is simply and easy – we only ask for an email address. To register, click on this link: <http://www.paul Yao.com/ReaderFeedback/Logon.aspx>.

When you register, you can download the available chapters from the full-text Table of Contents, available at this link: <http://www.paul Yao.com/ReaderFeedback/default.aspx>. We notify registered readers of new chapters – and chapter updates – as they become available.