Experience Using SQL Server CE to Support Mobile Collection and Analysis of Glucose Data

Rajendra Penugonda
Department of Computer & Information Sciences
Minnesota State University, Mankato
rajendra.penugonda@mnsu.edu

Steven Case
Department of Computer & Information Sciences
Minnesota State University, Mankato
steven.case@mnsu.edu

Abstract

Providing access to data in a timely manner is an important factor in the workplace. This is particularly true in the medical community. New methods are being developed to make the best use of the time available for a specific task. The use of clipboards, paper and pencil is getting obsolete for keeping track of patient information. Consequently, maintaining a consistency of data for remote use is increasingly becoming a necessity. This is readily apparent in the need for accurate monitoring of medical data related to acute medical care.

The authors have been involved in the development of a mobile application for the collection and analysis of glucose data relative to the treatment of diabetic patients. The system implements the Staged Diabetes Management system developed by the International Diabetes Center. The mobile application is supported on the Pocket PC platform and communicates to a back-end server using a wireless LAN – although the system is intended to operate seamlessly with any TCP/IP network.

In order to reduce the challenges associated with maintaining and accessing data in remote servers, SQL Server Windows 2000 CE Edition (SQL Server CE) was selected. SQL Server CE provides the flexibility in development along with the database performance expected for the mobile system. SQL Server CE provides the capabilities for efficient database management in mobile applications on wireless devices.

The mobile application is integrated with back-end server software using IIS and SQL Server 2000. The usage of IIS server provides different methods for authentication, authorization and encryption. This architecture provides the capability for data synchronization through merge replication along with continuous and efficient access to data to data anytime the user has access to the wireless network.

This paper will discuss the developed Stage Diabetes Management system. The paper will elaborate on the benefits experienced using SQL Server CE and the Pocket PC development tools in the development of this mobile application.

Introduction

As mobile computing and wireless technology have been gaining popularity in the past few years, the healthcare industry has been adopting various tools to provide mobile access to clinical and administrative data [5]. Healthcare personnel find their schedules to be busier and more stressful than in the past [7]. While treating patients, it is cumbersome to carry a clipboard with the details of the patient, the different lab test reports and (possibly) reference books for guidance. It is also not feasible for these personnel to be limited to a specific room and computer where they can access and update information about the patient. A time lapse between the actual medical process and its documentation may also lead to the inadvertent omission of an important detail.

Various medical device manufacturers are designing and developing thermometers, heart monitors, glucose meters and other medical devices to report results to a central repository as soon as a measurement is taken [5]. In order to eliminate the inefficiencies resulting in the gaps in location and time, wireless applications are being developed with a high level of security, scalability and reliable access to data. These applications focus on specific processes related to clinical and administrative tasks. By implementing these applications, companies have claimed to gain an increase in revenue, improvement in productivity, savings in costs and a high level of customer satisfaction. The different applications benefiting from these improvements can be broadly classified into point-of-care applications, clinical trial data collection, disease state management and pharmaceutical sales force automation.

Point-of-care applications allow the physicians to spend more time with the patient, give them timely access to critical data and help to reduce errors. Specifically, these applications can include tracking of patient referrals, enabling patient records to be viewed on a mobile device and accessing medical references stored in a central database [8]. A cumbersome paper-based system can result in problems analyzing results after data compilation and cause undesirable inconsistencies. The clinical trial data collection applications can capture data at the point of origin, track the data over a period of time and monitor symptoms by collecting and storing patient information such as blood sugar levels, blood pressure and cholesterol using a mobile device. The disease state management applications can retrieve information about a patient's medical condition, detect if a specific drug has any adverse side effects when combined with current medications and provide notifications to the patient to take prescribed medications. The pharmaceutical sales force automation applications are intended for sales representatives to track a specific medication's prospects, the inventory level of an existing medicine and access to personal information management software [1].

Our Application

The mobile application that we are developing is a software implementation of the Staged Diabetes Management System designed by the International Diabetes Center [3]. This system provides- a quick guide to detection and treatment of diabetes and has been

developed in accordance with the current standards of diabetes management. The guide provides essential information for starting and adjusting diabetes therapies. In our application, we are implementing only a portion of this guide to demonstrate the use of different tools and software we are using in the development process. This paper will provide an overview of the features provided by SQL Server CE (SSCE) and how it can be used in an application developed with eMbedded Visual Tools.

SSCE Environment

The SSCE platform offers the features of a relational database management system and its developmental model is similar to the SQL Server family. It supports the Structure Query Language (SQL) within applications developed for mobile devices. It also offers flexibility in data access whether it is connected to the remote SQL Server 2000 machine or it is in a disconnected mode. SSCE can be used in a combination of three different environments as shown in Figure 1.

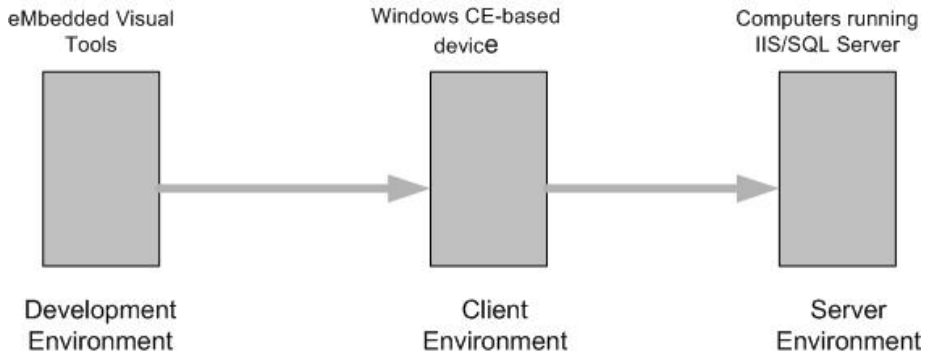

Figure 1: SSCE Environment [6]

The development environment includes a computer having either Microsoft Visual Studion 6.0, Microsoft Visual Studio .NET or Microsoft eMbedded Visual Tools installed [6]. The Microsoft eMbedded Visual Tools installation must include one of the software development kits (SDK) installed, namely, Handheld PC 2000 SDK or Pocket PC SDK or Windows Powered Pocket PC 2002 SDK. In our application we are using eMbedded Visual Basic, a component of Microsoft eMbedded Visual Tools for development. Microsoft eMbedded Visual Basic (eVB) is used for programming applications and uses SQL syntax that is specific to SSCE.

The SSCE environment provides support for two object models to represent the relational database. The Microsoft ActiveX Data Objects for Windows CE (ADOCE) and Microsoft ActiveX Data Objects Extensions for Data Definition language and Security (ADOXCE) are the two supported object models with both providing support for access to the underlying database and the database schema. The different operations supported by the database object modules include the addition of records to the tables, retrieval of records, creation of databases, tables and indexes. The ADOXCE object model is used

from eVB and interacts with a generic OLEDB Application Programming Interface, which accesses SSCE through an OLEDB provider [4]. The OLE DB provider is the lowest level interface for accessing data in SSCE.

The ADOCE library is used to manipulate the data within the database. The ADOXCE library extends ADOCE and is used to create databases (known as catalogs), tables, and indexes. ADOXCE also provides interfaces to manipulate the schema objects. A more detailed view of the development environment can be seen in Figure 2.

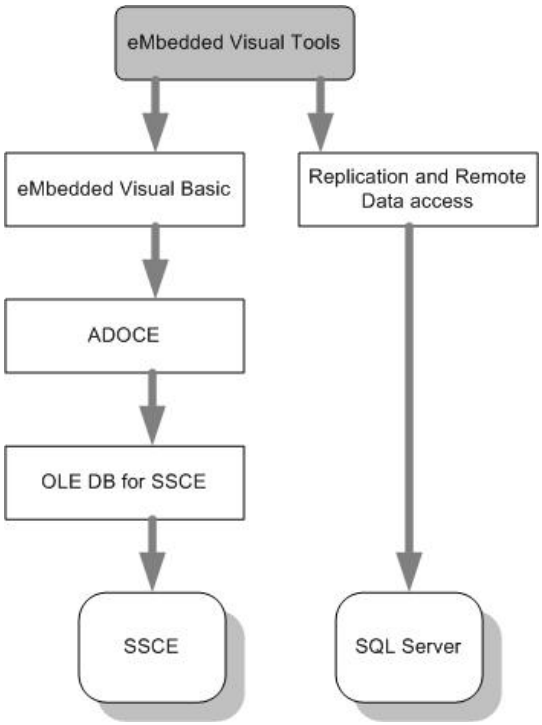

Figure 2: Development Environment [6]

The client environment consists of the device on which the application will be deployed. The device connects to the server directly through a network connection. Presumably, in order to support greater mobility, the network connection will be a wireless connection to an always-on wireless network. This enables access to current data at any time some information is needed from the database. Due to the mobility of the application and the potential for data to be locally stored on the mobile device, synchronization and conflict resolution is an ongoing process in this type of connection. Another type of connection can be made using Microsoft ActiveSync. In this mode the mobile device retains a local subset of the remote database. The mobile device will have to be docked with a desktop platform to perform synchronization of the databases – remote and local.

The server environment can consist of a single or multiple servers with at least one instance of IIS and one instance of SQL Server running within the system. Whenever a remote data access occurs, the communication to SQL Server occurs through the IIS.

The different components in the client and server environments and their relationships are shown in Figure 3. The client environment has a SSCE database engine that is responsible for managing the SSCE data on the Windows CE device. It keeps track of the database transactions and maintains some information in the database records about the tracking. The SSCE client agent in this environment is primarily responsible for connectivity and implements objects that can establish connections to the SQL Server through applications. Any requests made by the client agent are sent to the SSCE server agent through HTTP. The server agent communicates with the SQL Server and sends the results back to the client agent.

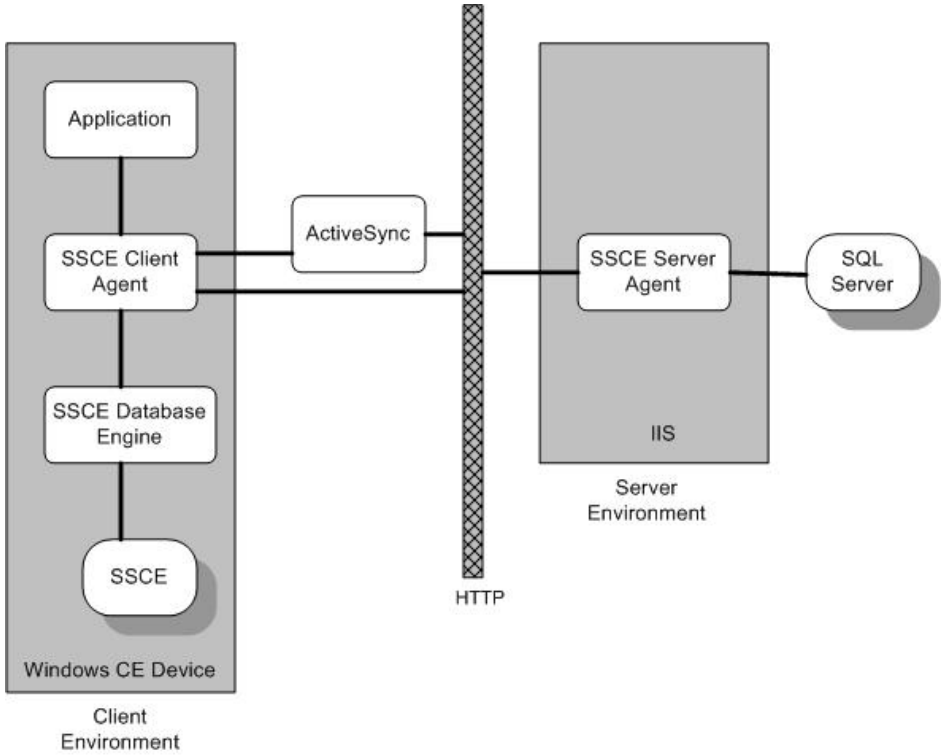

Figure 3: Client and Server Environment [6]

SSCE Installation

There are two options to configure a SSCE environment – single and multi-server environments. As the name indicates, a single-server environment will have the IIS and the SQL Server on the same computer. In the multi-server environment, the IIS and SQL Server are on different computers. The multi-server environment is more common in the corporate applications and hence our application is based on a multi-server environment.

In our application, the development tools, IIS and SQL Server are installed on separate computers. The development machine has eMbedded Visual Tools 3.0 and we are using eMbedded Visual Basic 3.0. This machine also has ActiveSync 3.5 and SSCE client development tools installed on it. The machine with IIS configuration has the SSCE server tools installed on it. These tools are needed in order to facilitate remote data access and replication. There is no need for any specific installation of any SSCE tools on the machine running SQL Server 2000.

The Windows CE based device also needs to have an installation of SSCE. This can be done either by an automated process with eMbedded Visual Basic or by a manual download of SSCE. We have preferred the use of eMbedded Visual Basic for the installation. In order to include SSCE in our application, we added Microsoft CE SQL Server Control 2.0, Microsoft CE ADO Control 3.1 and Microsoft CE ADO Ext. 3.1 for DDL in the eMbedded Visual Basic Project References [4].

Features

The major advantage offered to application developers by SSCE is the compact footprint. The SSCE database engine occupies 700k for Intel StrongARM x86 architecture CPUs and approximately 1.2MB to 1.6MB of disk space depending on the other processors on the target device. Though SSCE is built on the same model as SQL Server 2000, it does not include all the features provided in SQL Server 2000. This made it possible to keep SSCE small by eliminating some SQL Server 2000 functionalities that are not needed for the Windows CE environment. SSCE supports only Unicode character types and doesn't support some data types such as *smalldatetime* because it is converted to the *datetime* data type [6].

For mobile professionals, SSCE offers the advantage of installing portions of SQL Server 2000 database on the Pocket PC device. Remote Data Access (RDA) in SSCE provides features that enable an application to access data from a remote database and then store the data in a local database. Using RDA, data can be read from and modified on a remote server. SSCE also provides methods for tracking the changes made to the database whenever the records are being updated to the remote database. This is made feasible using the PULL and PUSH methods, which can be used by an application to access a remote database.

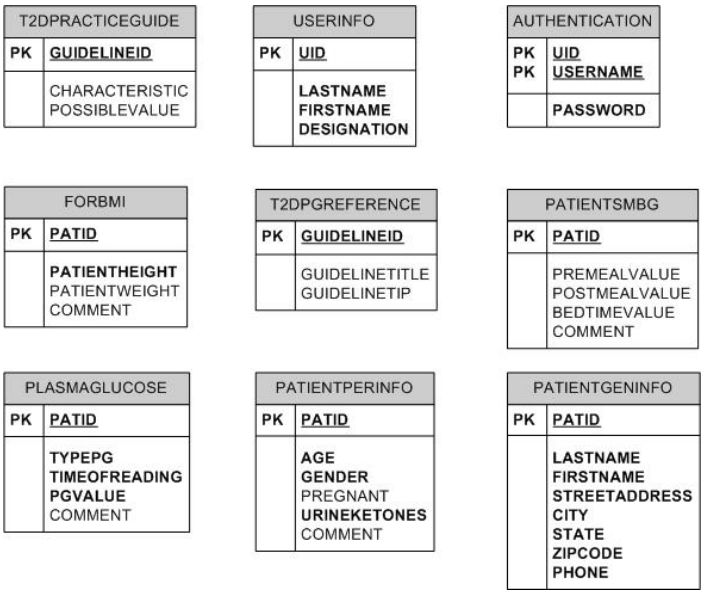

Figure 4: Database Schema for the application

Figure 4 shows the database schema used for our application. The foreign key relationships are not carried over from the SQL Server 2000 database but have to be created in the application using the objects provided by SSCE.

The PULL method is used to extract data from the remote SQL Server database and store it in a local SSCE database. Some of the parameters that are needed to use this method are the local SSCE table name that the will store the records pulled from the remote SQL Server and the SQL statement that returns the rows. The OLE DB connection string used when connecting to the SQL Server Database is also required here. The PULL method also gives an option of whether to track the changes made to the pulled table and then the name of a table that will record the errors, if any, when the data is sent back to the server.

The PUSH method is called to transmit the changes to the pulled tracked SSCE table back to the originating SQL Server table. The parameters needed for the PUSH method are the name of the pulled SSCE table with updated records and the OLE DB connection string. Another option that is provided for the PUSH method is whether the updated records need to be sent back to the SQL Server individually or batched together.

RDA provides a mechanism to resolve conflicts during a push transaction. When the PULL method is invoked, SSCE provides an option of specifying an error table. SSCE supports only row-level tracking so a conflict arises whenever a row cannot be pushed to the SQL Server. The conflicting row is returned to the error table and is deleted from the local database. In the case of a batched push transaction; all the rows need to be successfully processed. A single conflicting row will result in the entire transaction's failure and in this case the local database is left unchanged.

SSCE allows application programmers to use the same syntax for SQL queries as SQL Server. This includes the CREATE, ALTER, UPDATE, INSERT, DELETE and DROP statements. SSCE also supports the GROUP BY, HAVING, ORDER BY and join clauses. Unlike SQL Server, SSCE does not support a query which depends on the results of a nested query in a FROM clause. These features allow the programmer to reuse the existing code for SQL Server and reduce the learning curve for a new skill set. SSCE also supports transactions and declarative referential integrity constraints. It also provides an option of encrypting the database with 128-bit encryption.

Replication in SSCE is based on the SQL Server merge replication. Replication can be used over local or wide area networks and the replication communication protocol is designed for wireless transports. This protocol uses compression to reduce the size of data being transmitted. It also supports the use of encryption to secure the data during transmission. The merge replication for SSCE allows data to be updated autonomously on portable devices and the server. SSCE synchronizes with SQL Server through a HTTP connection to SQL Server Publisher via IIS, which also provides authentication and authorization services. The two factors that need to be considered for configuring replication in SSCE are performance and security. Factors that affect performance include the number of devices being supported, the frequency of synchronization with the server and the connection bandwidth. Common ways of configuring SSCE replication are single server, two servers and multiple IIS systems and SQL Server republishing topologies.

Analysis and Future Work

In our application, we are not collecting the glucose data from a device but we are using the data that has been updated on the server. This means that the glucose readings are obtained when the records are pulled from the server database table. The SDM system has some algorithms and guidelines for doctors, nurses and dieticians. Our application implements some of those guidelines for the screening, detection and treatment of diabetes. The application guides the user in making decisions for screening patients through appropriate screens. Some of the screenshots of the application are shown in Figures 5-7. The user has not been given permission to modify any data that is being pulled from the server. The reason behind this approach is that the data in server is periodically updated with various test readings for the patient. The device can connect to the server seamlessly to ensure that the most current data is available for the user. By referring to the patient's medical record, the user of this application can follow the different flowcharts specified in the SDM system. One obvious extension to this work, which is planned for the future, is to create a mobile application that integrates the functionality of the glucose meters to get readings of the patient's tests.

Another planned enhancement is to extend this application to include some data mining concepts. The SDM system is based on very limited test readings for detecting and treating diabetes. A better way of dealing with this shortcoming is to monitor the blood sugar more frequently and thereby adjusting the diet intake. This might also require a

change in the physical activity and forms of exercise. If the application can provide a feature of analyzing the recent test results and provide real-time management of medicine intake or diet improvements, it will be more beneficial for the patient. This mining of data in real time will improve the management of diabetes. There will be a need for more sophisticated glucose meters for this approach and FDA approval too. We will continue to extend the application to include more functions that will enable an extensive usage of features provided by SSCE.

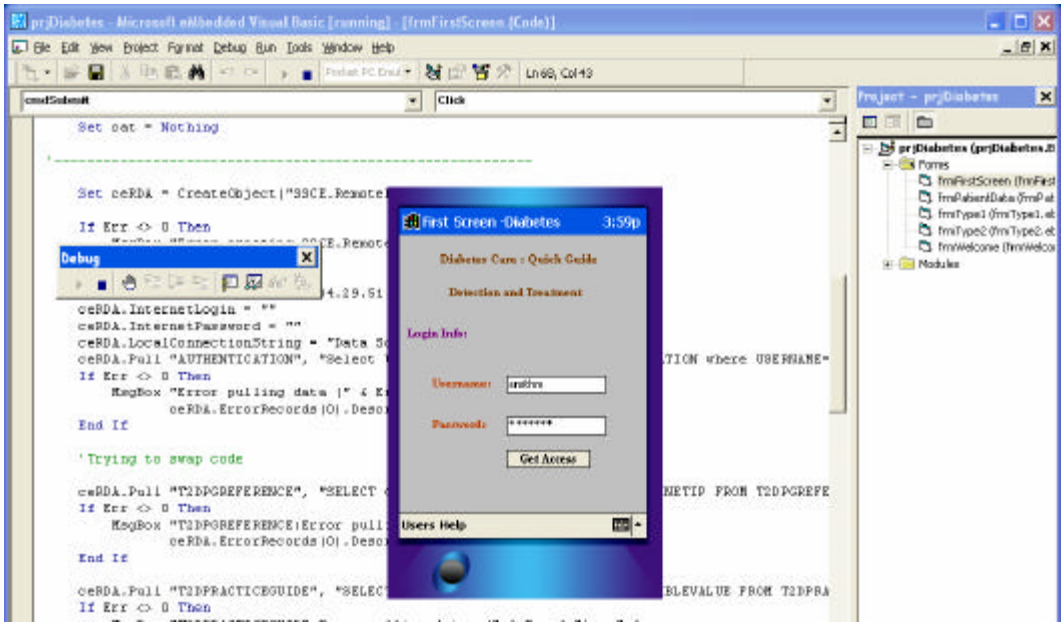

Figure 5: Login Screen in the application

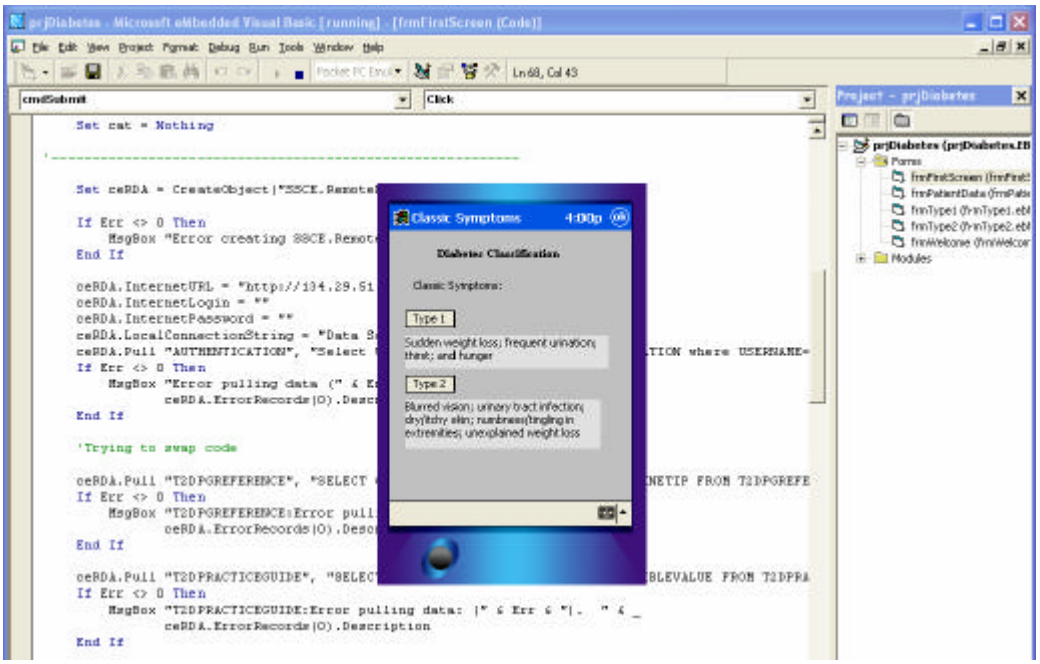

Figure 6: Screen for Selecting the type of Diabetes

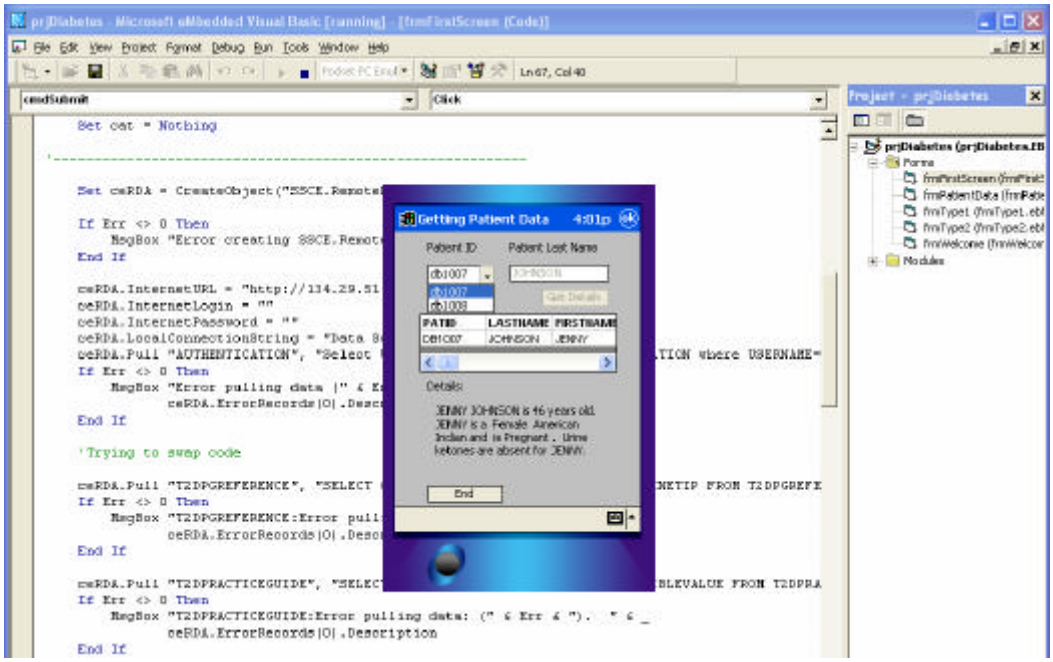

Figure 7: Screen showing the patient information extracted from the database

References

1. Air2Web, Inc. (2002). *The Possibilities of Wireless Healthcare*. Retrieved from http://www.air2web.com.
2. Article. (2002). *Healthcare System goes wireless.* Communication News. pp-40.
3. Etzwiler, Donnell. et al. (1999). *Staged Diabetes Management Detection and Treatment Quick Guide.* International Diabetes Center, Institute for Research and Education, Minneapolis.
4. Grattan, Nick. (2002). *Pocket PC, Handheld PC Developer's Guide With Microsoft eMbedded Visual Basic.* Prentice Hall.
5. Healthcare Informatics. (1999). *Wow! Wireless Works!*. Retrieved from http://www.healthcare-informatics.com/issues/1999/10_99.
6. Microsoft Corporation. (2002). *Microsoft SQL Server 2000 Windows CE Edition Books Online*. Retrieved from http://www.microsoft.com/sql/ce/techinfo/20bol.asp.
7. Turisco, F. & Steinichen, P. (2002). *Understanding the Wireless Technology and Mobile Computing in Healthcare*. Proceedings of the 2002 Annual HIMSS Conference and Exhibition.
8. Wade, J. & Stevens, T. (2002). *Web Access to Live Patient Information: Instant Knowledge, Better Care.* Proceedings of the 2002 Annual HIMSS Conference and Exhibition.