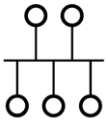

Orbix®

First Northern Bank Tutorial

Version 6.1, December 2003

Making Software Work Together™

Updated: 30-Jan-2004

Contents

List of Figures

CHAPTER 1

Introduction

This tutorial describes how to run the First Northern Bank demo, which is a cross-platform demo application. This chapter introduces the overall demo architecture, and the CORBA, J2EE, Web services, and .NET components.

In this chapter

This chapter discusses the following topics:

First Northern Bank Architecture

Overview

The new First Northern Bank (FNB) system performs a range of banking functions across a wide range of platforms (for example, CORBA, J2EE, Web services, and .NET). The FNB system is implemented using a wide variety of programming languages (for example, Java, C#, and Visual Basic).

This section describes the high-level architecture of the new system, and gives a brief overview of its components. It includes the following topics:

FNB architecture

Figure 1 shows the overall FNB demo system architecture.

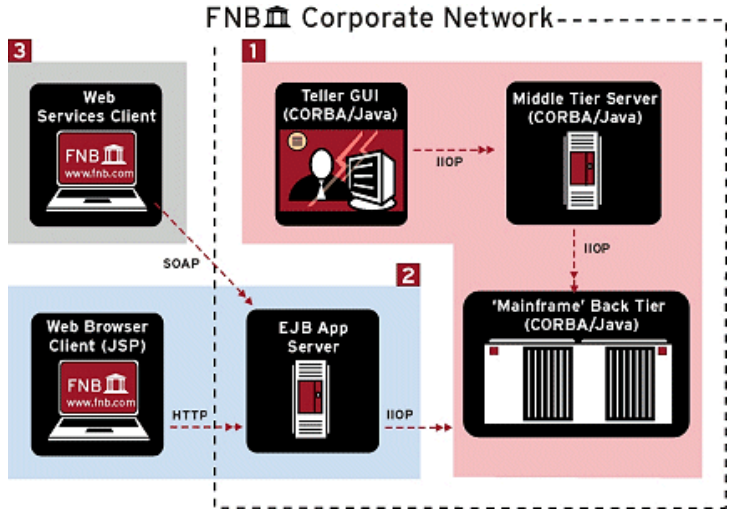

Figure 1: *FNB System Architecture*

The main components in Figure 1 are as follows:

1. CORBA core banking.
2. Visual Basic ATM.
3. Web services online purchasing.
4. .NET Remoting online purchase manager.
5. J2EE Internet banking.

CORBA banking

The CORBA banking application provides the core banking services that the bank offers to its customers. For example, opening an account, making a deposit, or making a withdrawal.

The CORBA banking application is implemented as a three-tier system, which consists of the following components:

- Bank teller client GUI (Graphical User Interface).
- Middle-tier server.
- Back-tier simulated mainframe system (a CORBA Java server).

All network communication is sent using the Internet Inter-ORB Protocol (IIOP).

Visual Basic ATM

The Automated Teller Machine (ATM) client application enables customers to view their account details and to make withdrawals.

The ATM client application is implemented as a simple Visual Basic GUI application. The GUI client is integrated with the CORBA middle-tier using Orbix COMet, which bridges between the Microsoft COM (Component Object Model) and CORBA standards.

Web services online purchasing

The Web services application provides an online purchasing system for online vendors. In this system, FNB stores a record of online purchase transactions, including all customer credit card details. Online vendors only need to store the transaction IDs, and in case of problems, they can ask the bank for details.

Figure 1 shows a Web services client application that invokes on the CORBA based-Web service. This client could be implemented in several programming languages (for example, Java, C++, or Visual Basic). The FNB example uses the generic Web services test client provided by Orbix.

Like the CORBA and J2EE systems, the Web services application is also a three-tier system. Network communications between the Web service and the client are sent using the Simple Object Access Protocol (SOAP) over HTTP.

.NET Remoting online purchasing manager

FNB also provides a .NET Remoting client application, written in C#. This is a simple online purchase management application that runs internally in the FNB bank. It enables bank staff to retrieve details of merchants that are registered with the bank.

The .NET Remoting client uses the Orbix .NET Connector to bridge between .NET and the CORBA back-tier server.

J2EE internet banking

The J2EE application provides customers with Internet banking services. It provides Web browser access to customer accounts (for example, viewing an account balance, or paying a bill online).

The J2EE Internet banking application is implemented using Enterprise Java Beans (EJBs) and Java Server Pages (JSPs), which run in any application server (for example, JBoss). The application server uses Orbix Connect to communicate with the back-tier CORBA server. Orbix Connect implements Sun's Java Connector Architecture (JCA) standard.

Network communications between the application server and the browser client are sent using the Hypertext Transfer Protocol (HTTP). Those between the application server and the back-tier server are sent using IIOP.

CORBA Banking Application

Overview

This section describes the CORBA core banking application in more detail. It includes the following topics:

- "CORBA bank architecture".
- "Bank teller GUI client".
- "Middle-tier CORBA server".
- "Back-tier CORBA server".

CORBA bank architecture

Figure 2 shows the architecture of the three-tier CORBA banking application.

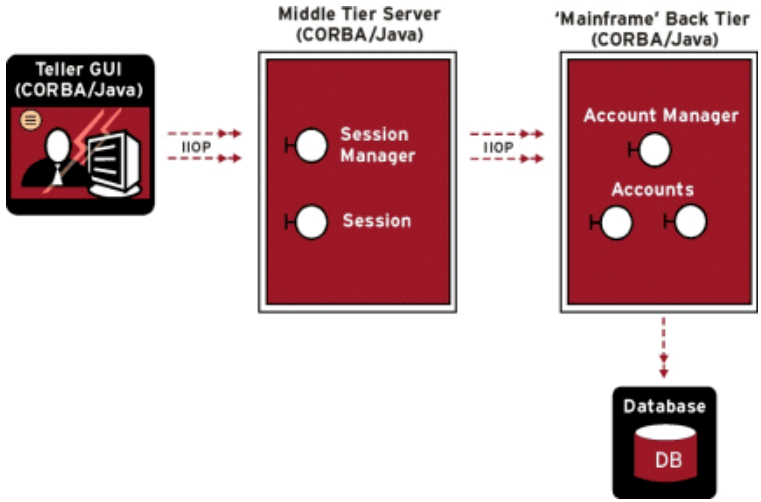

Figure 2: *FNB Banking Application*

The main components in Figure 2 are as follows:

- Front-tier client used by bank teller (Java GUI).
- Middle-tier business architecture (CORBA Java server).
- Back-tier simulated mainframe system (implemented as a CORBA Java server).
- Back-tier database (flat-file system).

Bank teller GUI client

The bank teller GUI enables tellers to open and close accounts, and to make withdrawal and lodgements to accounts.

The bank teller GUI is implemented as a Java Swing client application. The GUI interface is shown in Figure 17 on page 29.

Middle-tier CORBA server

The middle-tier CORBA Java server manages business sessions between the client and the back-tier server.

The middle-tier server implements a `BussinesSessionManager` factory object, which creates session objects to manage interaction with the client (for example, `TellerSession` and `BusinessSession` objects).

The middle-tier CORBA server is also known as the FNB Business Architecture (FNBBA).

Back-tier CORBA server

The back-tier CORBA server is the simulated mainframe system, which is responsible for managing customer accounts.This is implemented as the Java `BankObjects` server. You do not need a mainframe system to run this demo.

The `BankObjects` server implements an `AccountMgr` factory object, which creates `Account` objects (for example, `CreditCardAccount` and `CurrentAccount` objects). These objects represent all customer account information (for example, customer name, address, and account number).

The `Account` objects are stored in a flat-file database system, which stores the accounts as serialized Java objects on disk.

Visual Basic ATM Application

Overview

This section describes the Visual Basic ATM client and how it is integrated with the CORBA banking application. It includes the following topics:

- "COM/CORBA bridge architecture".
- "Visual Basic ATM client".
- "Orbix COMet".
- "CORBA middle and back-tiers".

COM/CORBA bridge architecture

Figure 2 shows the architecture of the three-tier COM/CORBA integration used by the ATM application.

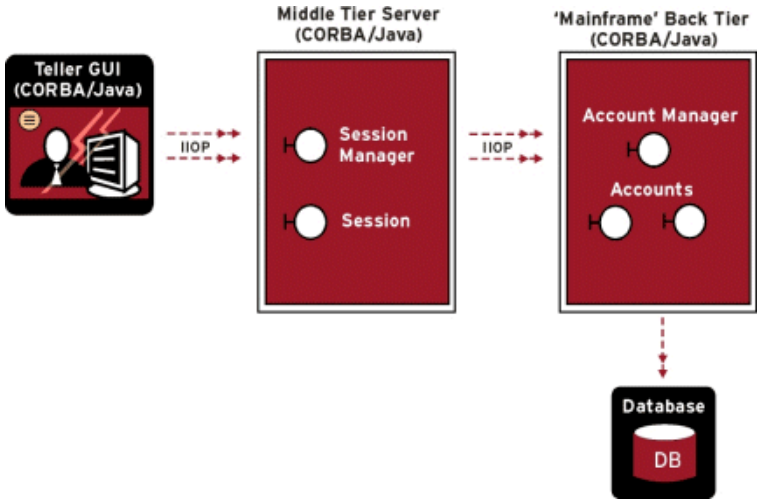

Figure 3: *ATM System Architecture*

The main components in Figure 2 are as follows:

- Front-tier client used by customer (Visual Basic GUI).
- Middle-tier business architecture (CORBA Java server).
- Back-tier simulated mainframe system (implemented as a CORBA Java server).
- Back-tier database (flat-file system).

Visual Basic ATM client

The ATM client enables customers to view their account details and make withdrawals. The ATM client GUI is implemented as a Visual Basic client application. The GUI interface is shown in Figure 17 on page 29.

Orbix COMet

Orbix COMet is a bridge between Microsoft COM (or Automation) and CORBA. Orbix COMet integrates the Visual Basic ATM client with the CORBA middle-tier server.

When a customer performs a transaction using an ATM, the client issues a request that is intercepted by COMet, which runs on the client machine. COMet then directs the request to the CORBA middle-tier server. The server processes the client operation, and sends the results of the completed operation back to the client.

CORBA middle and back-tiers

The middle-tier and back-tier systems are the same as those described in "Middle-tier CORBA server" and "Middle-tier CORBA server" on page 7.

Web Services Online Purchasing Application

Overview

This section describes the Web services online purchasing application and its components in more detail. It includes the following topics:

- "Online purchasing architecture".
- "Web services clients".
- "Middle-tier CORBA interface".
- "Back-tier CORBA server".

Online purchasing architecture

Figure 4 shows the architecture of the three-tier online purchasing application.

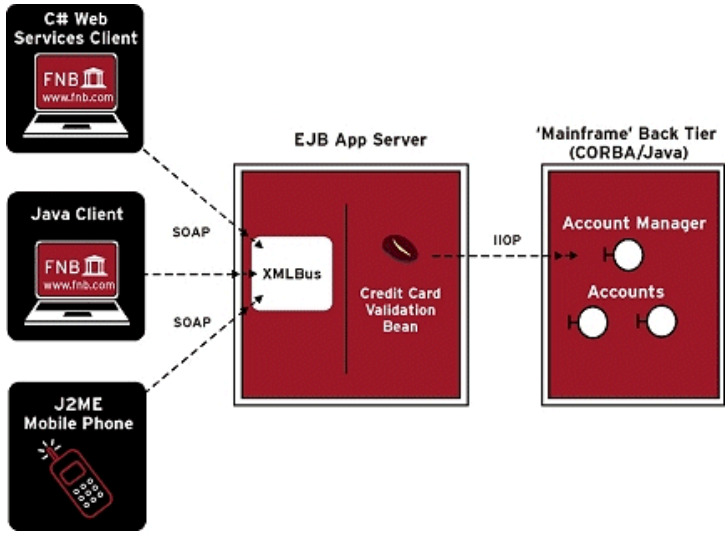

Figure 4: *Online Purchasing Application*

The main components in Figure 4 are as follows:

- Web services clients.
- Middle-tier CORBA interface.
- Orbix Web services.
- Back-tier CORBA Java server.

Web services clients

The Web services client applications provide online facilities for managing credit card purchases.

Figure 4 shows a variety of client applications. Because this is a Web service, the client could be written in several programming languages (for example, Java, C++, or Visual Basic). This tutorial demonstrates how to use a generic Web services test client provided by Orbix.

Middle-tier CORBA interface

The middle-tier server runs the `FNBOnlinePurchasing` CORBA interface, shown in Figure 4. This interface communicates with the Web services client using SOAP and HTTP.

The application server forwards the HTTP request to the Orbix Web services container, which decodes and handles the incoming SOAP message.

Back-tier CORBA server

Communication between the middle-tier CORBA interface and the back-tier server is sent using IIOP. See "CORBA Banking Application" on page 6 for more information about the back-tier CORBA server.

J2EE Internet Banking Application

Overview

This section describes the J2EE Internet banking application and its components in more detail. It includes the following topics:

- "Internet banking architecture".
- "Web browser client".
- "J2EE application server".
- "Orbix Connect".
- "Back-tier CORBA server".

Internet banking architecture

Figure 5 shows the architecture of the three-tier J2EE Internet banking application.

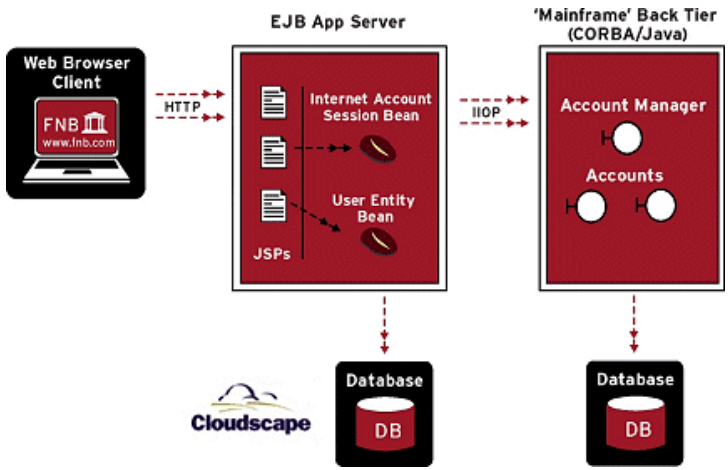

Figure 5: *Internet Banking Application*

The main components in Figure 5 are as follows:

- Web browser client.
- Middle-tier J2EE application server.
- Orbix Connect.
- Back-tier CORBA Java server.
- Back-tier database.

Web browser client

A standard Web browser provides Internet banking services to customers. Users must first register, and create a user ID and password, before logging on. Internet banking services include viewing an account balance and paying a bill online.

Network communications between the Web browser and the application server are sent using HTTP.

J2EE application server

An application server (for example, JBoss) provides the middle-tier J2EE infrastructure. It runs the Java Server Pages (JSPs) that serve up the Internet banking Web pages in the browser. The application server also runs the Enterprise Java Beans (EJBs) that communicate with a database and the back-tier CORBA server.

For example, the `User` entity bean handles the customer information stored in the database; while the Internet account session bean (`InetAccount`) handles browser sessions with the back-tier server.

Orbix Connect

Orbix Connect is used by the application server to communicate with the back-tier CORBA server.

Orbix Connect implements Sun's Java Connector Architecture (JCA) standard. It works within popular J2EE application servers (for example, JBoss) to automate configuration, packaging and deployment of connections to CORBA servers. CORBA details are hidden from developers, which reduces time-to-deployment.

Back-tier CORBA server

Communications between the application server and the back-tier server are sent using IIOP. See "CORBA Banking Application" on page 6 for more information about the back-tier CORBA server.

Note: For more detailed information on the FNB system architecture and how the system was developed, see the *First Northern Bank Developer's Introduction*. This is available online at:

`http://www.iona.com/support/docs/orbix/6.1/getting_started.xml`

CHAPTER 2

Running the Banking Application

This chapter explains how to set up your environment for the FNB demo, and how to run the core CORBA banking application. You should perform these steps first, before proceeding to any others. This chapter also shows how to run an optional integration with Microsoft COM.

In this chapter

This chapter includes the following topics:

Setting Up your Environment

Overview

This section describes the steps that you should perform before running the First Northern Bank demo. It includes the following steps:

Step	Action
1	"Install the required software".
2	"Check your Java configuration".
3	"Create a configuration domain".
4	"Start IONA Central".
5	"Customize IONA Central".
6	"Set your configuration domain".
7	"Start the Orbix services".

Install the required software

To run the demo, you must have the following software installed:

- Orbix 6.1 Standard or Enterprise.
- Java Development Kit 1.4.1_01 or higher.

.NET To run the .NET online purchasing manager, you must have the .NET 1.1 runtime installed. Available from Microsoft Windows Update center or with Visual Studio .NET 2003 (VS 7.1).

Visual Basic To run the Visual Basic ATM client, you must have the Visual Basic 6 runtime installed.

J2EE To run the J2EE application, you must have the following installed:

- Orbix Connect 1.0
- a J2EE application server supported by Orbix Connect (For example, JBoss 3.2.x)

UNIX: If you are using a shared Orbix installation, ensure that the following directories are not write protected:

- *<install-dir>*/etc
- *<install-dir>*/var

Check your Java configuration

Ensure that your JAVA_HOME environment variable specifies the directory where you installed your Java Development Kit; for example:

```
c:\j2sdk1.4.1_01
```

Note: If your Java environment is listed on your CLASSPATH, it must be the same version as that specified by your PATH environment variable.

Check your Orbix environment

Ensure that your environment variables are set as follows:

- IT_PRODUCT_DIR should be set to your Orbix installation directory (for example, c:\orbix).
- PATH should include the following directory:

  <install-dir>\asp\6.1\bin

Create a configuration domain

Orbix supplies all the infrastructure services required to support the deployment of enterprise distributed applications. An Orbix configuration domain specifies which services are deployed, and how they are configured within a particular environment.

To create a configuration domain, perform the following steps:

1. Enter the following command:

   ```
   <install-dir>\asp\6.1\bin\itconfigure
   ```

   Alternatively, on Windows, you can also select **Orbix Configuration** from the **Start** menu.

   This displays the **Orbix Configuration** tool, shown in Figure 6.
2. Click the **Create** button to set the domain name and type. Accept the default settings.

3. Click **Next** and follow the remaining steps in the wizard, accepting all the default settings. This creates and deploys a configuration domain named `sample-domain`. It also generates the following scripts:

   `sample-domain_env`

   `start_sample-domain_services`

   These scripts are used to set your environment to use `sample-domain`, and to start its services. They are created in your *<install-dir>*`\etc\bin` directory.

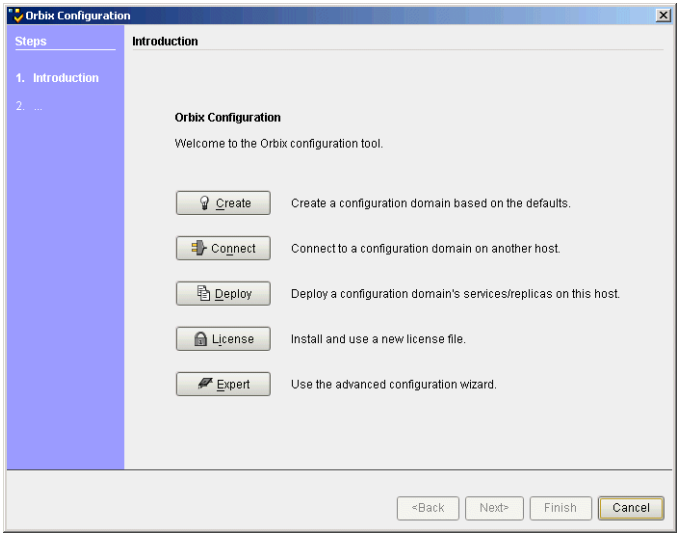

Figure 6: *Orbix Configuration Tool*

Note: For full details of using the **Orbix Configuration** tool, see the *Orbix Administrator's Guide*.

Start IONA Central

The **IONA Central** toolbar enables you to start up Orbix tools in your environment. It also enables you to run the various FNB demo applications.

To start **IONA Central**, enter the following command:

```
<install-dir>\asp\6.1\bin\itcentral
```

Alternatively, on Windows, you can also select **IONA Central** from the **Start** menu.

IONA Central detects the currently configured domain when it starts up (in Figure 7, this is `sample-domain`, displayed in the title bar).

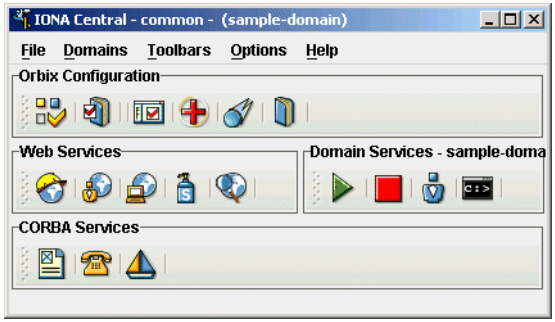

Figure 7: *IONA Central*

Customize IONA Central

To customize **IONA Central** for use with the FNB demo, perform the following steps:

1. Select **File**|**Open Project.** This launches an **Open** file dialog box.
2. Navigate to the following directory:

   ```
   <install-dir>\asp\6.1\demos\common\
   ```
3. Select the `fnb` directory, and click **Open**. This adds custom FNB buttons to the toolbar, shown in Figure 8.

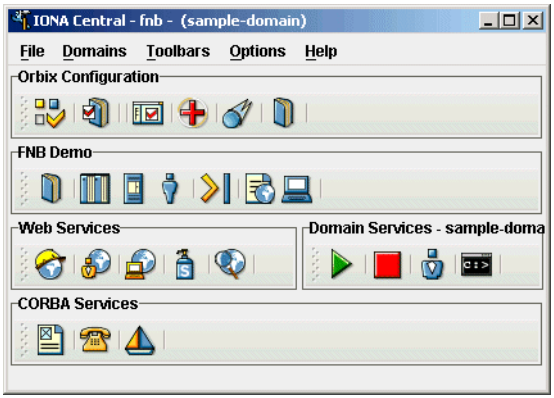

Figure 8: *Customized IONA Central*

Set your configuration domain

You should ensure that the default `sample-domain` is set. You can set your domain using the **IONA Central** menu. Select **Domains**|**Set Active Domain**, and select **sample-domain**.

Alternatively, you can do the following:

Windows

1. Change to the `<install-dir>\etc\bin` directory.
2. Run `sample-domain_env`.

UNIX

1. Change to the `<install-dir>/etc/bin` directory. For example, this may be `/etc/opt/iona/bin`, depending on your environment.
2. Source `sample-domain_env`.

Start the Orbix services

To start the Orbix services, click the **Start Domain Services** button:

Alternatively, you can run the following command:

```
start_sample-domain_services
```

This starts the Orbix services needed to run the demo. Wait until you see a `Finished` message. You are now ready to run the demo.

Running the CORBA Application

Overview

The First Northern Bank demo is pre-built and ready to run. This section explains how to run the core CORBA banking application. It includes the following steps:

Step	Action
1	"Start the back-end server".
2	"Start the middle-tier server".
3	"Start the GUI console".
4	"Logon to the system".
5	"Create a bank account".
6	"Lodge funds into the current account".

Start the back-end server

To start the back-end CORBA server, perform the following steps:

1. In the **FNB Demo** pane of **IONA Central**, click **Start the CORBA backend server**:

   

   Alternatively, enter the following command from your `<install-dir>\asp\6.1\demos\common\fnb` directory:

Windows	`start itant start_backend`
UNIX	`itant start_backend&`

   This starts the `BankObjects` Java server, which acts as a repository for the account objects stored in the system.

2. Wait until you see the following message:

```
start_backend:
     [exec] Removing Federating Naming Service...
     [exec] Unbinding Mainframe from NS
     [exec] Done
     [echo] Starting the Bankobjects server
     [java] Initialising ORB...
     [java] Registering mbean...
     [java] Bankobjects server active....
```

Start the middle-tier server

To start the middle-tier CORBA server, perform the following steps:

1. In the **FNB Demo** pane of **IONA Central**, click **Start the FNBBA CORBA server**:

   

   Alternatively, enter the following command:

Windows	`start itant start_fnbba`
UNIX	`itant start_fnbba&`

   This starts the middle-tier server architecture used by the GUI client application.

2. Wait until you see the following message:

```
start_fnbba:
     [echo] Starting the FNBBA server
     [java] Initialising ORB...
     [java] Creating POA's....
     [java] After registering BusinessSessionManager...
     [java] Registering mbean...
     [java] Context is Mainframe
     [java] fnbba server running...
```

Start the GUI console

To run the Java GUI console, perform the following steps:

1. In the **FNB Demo** pane of **IONA Central**, click **Start the teller GUI**:

   

   Alternatively, enter the following command:

   Windows `start itant start_gui`

   UNIX `itant start_gui&`

2. Wait until you see the following message:

```
start_gui:
     [echo] Starting the GUI window
```

The **Logon Screen** dialog is displayed, as shown in Figure 9.

Figure 9: *The Logon Screen*

Logon to the system

To logon to the bank system, perform the following steps:

1. Enter your name in the **Teller ID** field.
2. Click **OK** to confirm your teller name.

The **FNB Teller Application** starts up. This is a simple GUI console for bank tellers, shown in Figure 10. You can use this console to open and close bank accounts, and to lodge and withdraw money.

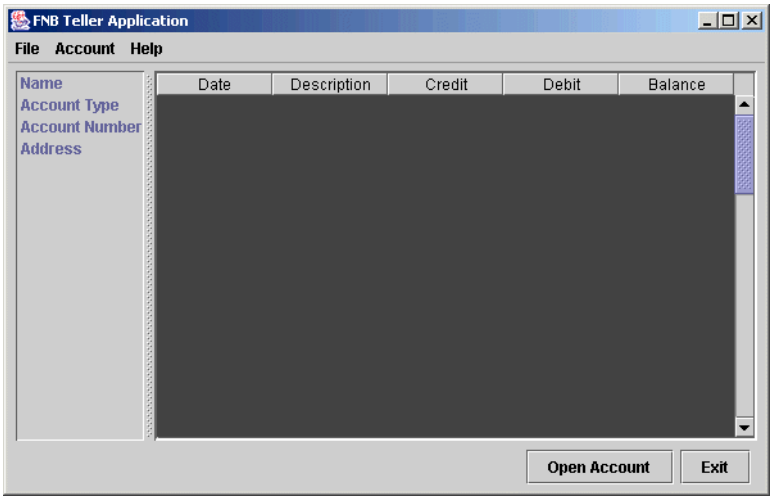

Figure 10: *FNB Teller Application*

Create a bank account

To create bank account, perform the following steps:

1. Select **Account**|**New Account**. The **New Account Details** dialog is displayed, as shown in Figure 11.
2. Select the **Type of Account to Open** (**Credit Card** or **Current Account**).
3. Enter the customer details in the appropriate text fields.
4. Click **OK** to save the customer details. The **Account Created** dialog is displayed, as shown in Figure 12.

To run the Web services application later, you should create at least two bank accounts—a credit card account for the online purchaser (shown in Figure 11), and a current account (shown in Figure 13) for the online retailer. Be sure to enter a **Credit Limit** for your credit card account.

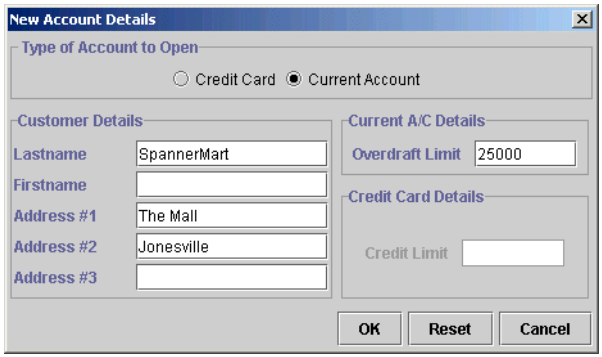

Figure 11: *Current Account Details*

Figure 12: *Current Account Created*

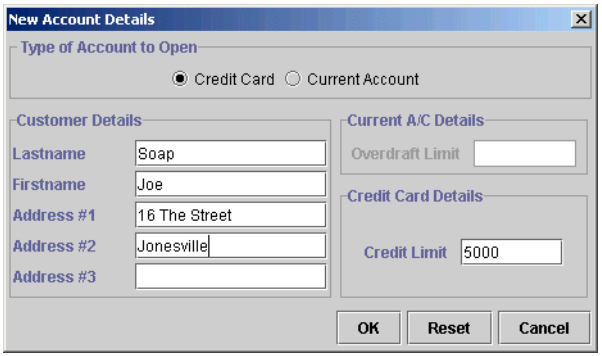

Figure 13: *Credit Card Account Details*

Note: Make a note of the account numbers that you created; you will use them later for online transactions.

Figure 14: *Credit Card Account Created*

Lodge funds into the current account

To lodge funds into an existing open account, perform the following steps:

1. Select **Account|Lodge Funds.** The **Lodge Funds** dialog is displayed, as shown in Figure 15.
2. Enter the amount in the text box.
3. Click **OK** to save the lodgement and return to the **FNB Teller Application**.

Figure 15: *Lodge Funds Dialog*

Figure 16 shows the **FNB Teller Application,** and the current account lodgements made:

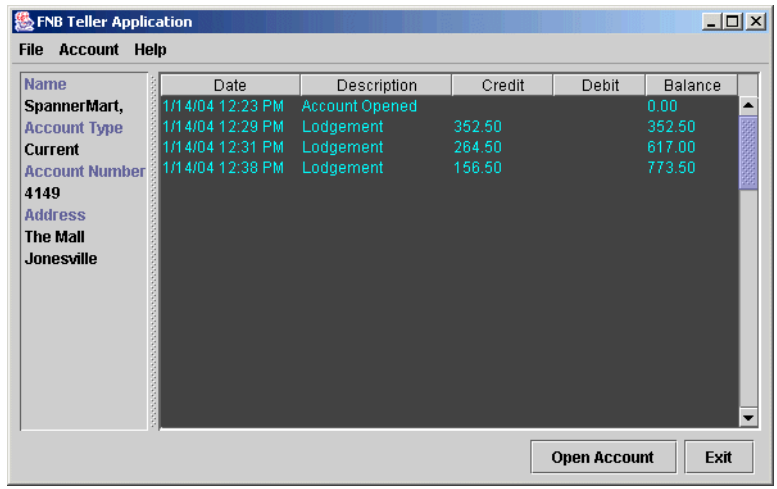

Figure 16: *Viewing Lodgements*

Running the Visual Basic ATM Application

Overview

This section explains how to run the Visual Basic ATM application. This is a Windows client application that uses Orbix COMet to bridge between Microsoft COM and CORBA. This section includes the following steps:

Step	Action
1	"Populate the IFR".
2	"Start the ATM client".
3	"Enter your PIN number".
4	"View your account details".
5	"Make a withdrawal".

Note: The VB ATM is an optional Windows client application. You do not need to run this application to run other applications in the FNB demo (for example, the online purchasing Web service).

Requirements

To run the Visual Basic ATM client, you must have the Visual Basic 6 runtime installed.

Populate the IFR

In the **FNB Demo** pane of **IONA Central**, click **Put the IDL into the IFR**:

Alternatively, enter the following command:

```
itant populate_ifr
```

This populates the CORBA Interface Repository with the Interface Definition Language (IDL) used by the FNB demo.

Start the ATM client

In the **FNB Demo** pane of **IONA Central**, click **Visual Basic ATM Client**:

Alternatively, enter the following command:

```
<install-dir>\asp\6.1\demos\common\fnb\atm\atm
```

This launches the ATM login screen, shown in Figure 17.

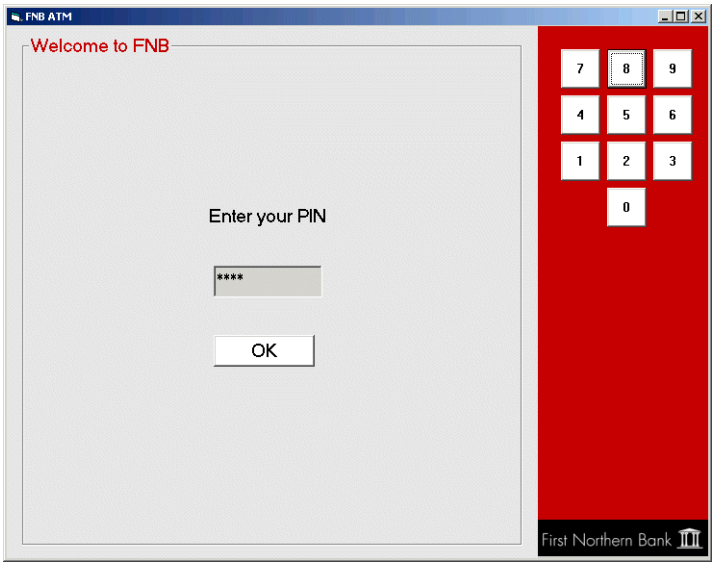

Figure 17: *ATM Login Screen*

This ATM client selects one of the accounts you have created at random, so you must have created at least one account.

Enter your PIN number

Use the buttons at the top right of the screen to enter a four digit Personal Identification Number (PIN). You can enter any four digit number. Alternatively, enter a number directly in the text box provided.

Click **OK**.

This launches the **Welcome** screen, shown in Figure 18.

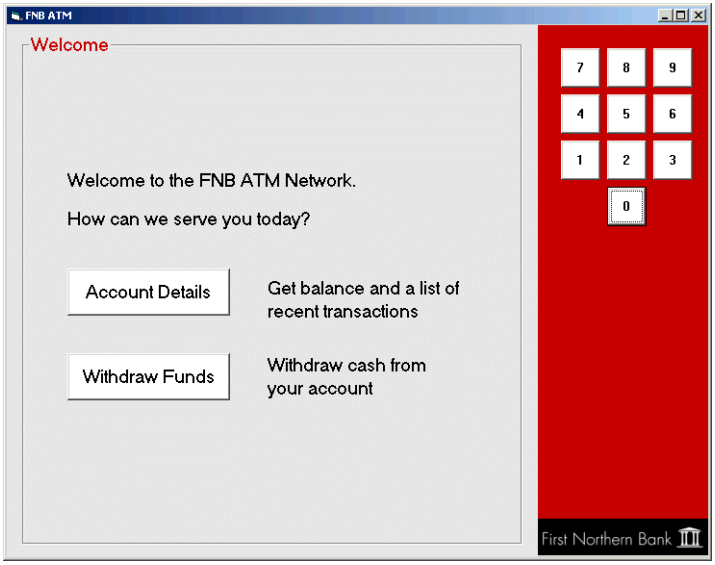

Figure 18: *ATM Welcome Screen*

View your account details

Click the **Account Details** button. This launches the **Account Details** screen, which displays your balance and a list of your recent transactions, shown in Figure 19.

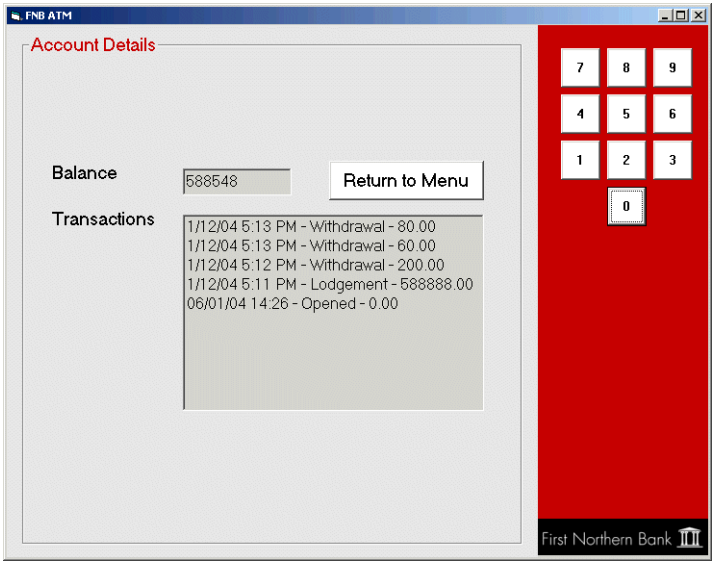

Figure 19: *ATM Account Details*

Make a withdrawal

To make a withdrawal, perform the following steps:

1. Click the **Return to Menu** button to return to the **Welcome** screen.
2. Click the **Withdraw Funds** button. This launches the **Withdraw Funds** screen, shown in Figure 20.
3. Click the amount you wish to withdraw. If you open the **FNB Teller Application,** your withdrawal will be displayed in the list of transactions.

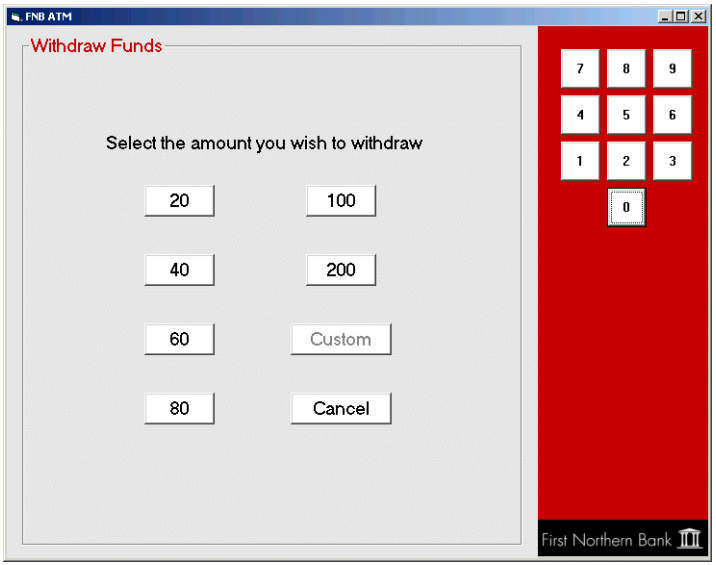

Figure 20: *ATM Withdraw Funds*

CHAPTER 3

Running the Online Purchasing Application

This chapter explains how to run the FNB online purchasing application. This sample application shows CORBA, Web services, and .NET components, working together, in an integrated example.

In this chapter

This chapter discusses the following topics:

Creating the Web Service

Overview

This section describes how to create the FNB Web service application. It explains how to enable a online purchasing Web service for the CORBA banking application. It includes the following steps:

Step	Action
1	"Populate the IFR".
2	"Start the Create Services wizard".
3	"Specify the application name".
4	"Specify available resources".
5	"Configure the client ORB".
6	"Specify the target object reference".
7	"Select the Interface Repository ID".
8	"Select the Web service methods".
9	"Specify the Output XAR".
10	"Close the Create Service wizard".
11	"Deploy the Web service application".

Populate the IFR

If you did not run the Visual Basic ATM client, you will need to populate the IFR. In the **FNB Demo** pane of **IONA Central**, click **Put the IDL into the IFR**:

Alternatively, enter the following command:

```
itant populate_ifr
```

This populates the CORBA Interface Repository with the Interface Definition Language (IDL) used by the FNB demo.

Start the Create Services wizard

To start the **Create Services** wizard, perform the following steps:

1. In the **Web Services** pane of **IONA Central**, click the **Web Services Builder** button:

   

   Alternatively, enter the following command:

   ```
   itws_builder
   ```

   The **Web Service Builder** is displayed in Figure 21.

2. In the left-hand **PROJECTS** pane, select the project that you wish to use. This example uses the default **My Project.** If you wish to create a new project, select **Project|New**.

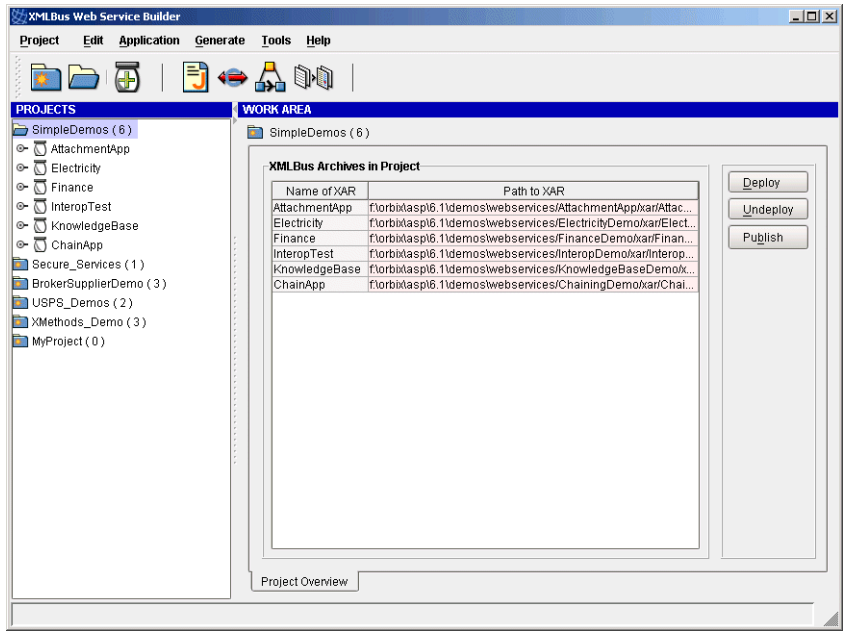

Figure 21: *Web Service Builder*

3. From the toolbar, select the **Create a Web Service based on a CORBA Resource** button:

Alternatively, from the menu, select **Application**|**Create Web Service**|**From CORBA IDL**.

Figure 22 shows the **Create Service (From CORBA IFR)** wizard that takes you through the required steps. This example uses CORBA IDL, but you can also build a Web service from a Java class, an XML schema map, or process flow. See the *Orbix Web Services Development Tools* guide for more details.

Specify the application name

To specify the Web service name information, perform the following steps:

1. Enter the **Application Name** (in this example, `FNBOnlinePurchasing`, shown in Figure 22).

   The **Service Name** (`FNBOnlinePurchasingService`) and **Port Name** for your application (`FNBOnlinePurchasingPort`) are filled automatically when **AutoFill** is checked. If this is not checked, you must also specify these values.

2. Click **Next**.

All this information is used to identify the Web service application in the Web services container.

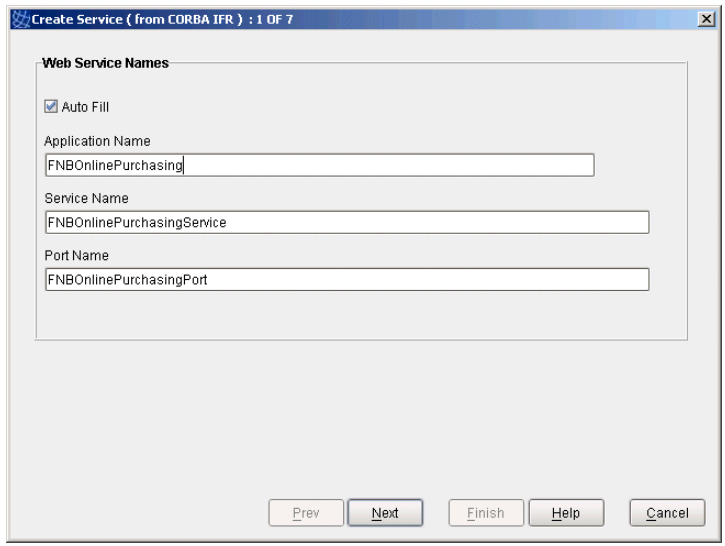

Figure 22: *Specifying Web Service Names*

Specify available resources

You can explicitly add more resources to your Web service (for example, Java classes or EAR files) using the **Available Resource** screen, shown in Figure 23. This is not necessary in this example.

Click **Next**.

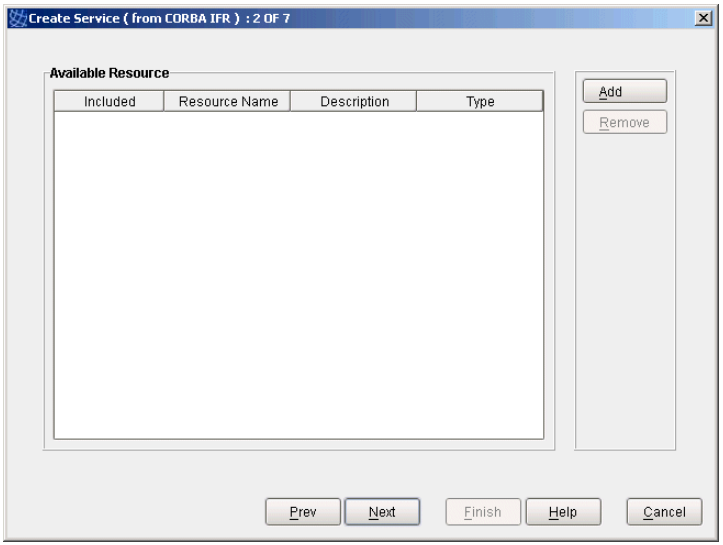

Figure 23: *Specifying Available Resources*

Configure the client ORB

To select the client-side ORB configuration, click **Next**. This specifies the default, which is **Use Orbix services** (shown in Figure 24).

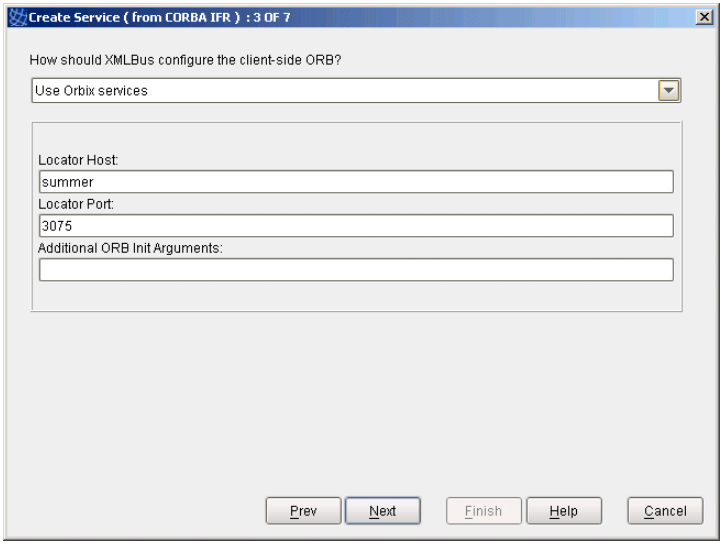

Figure 24: *Configuring the Client ORB*

Specify the target object reference

To choosing how to specify target object reference, perform the following steps:

1. Select **As a Naming Service entry that will be resolved at runtime** (shown in Figure 25).

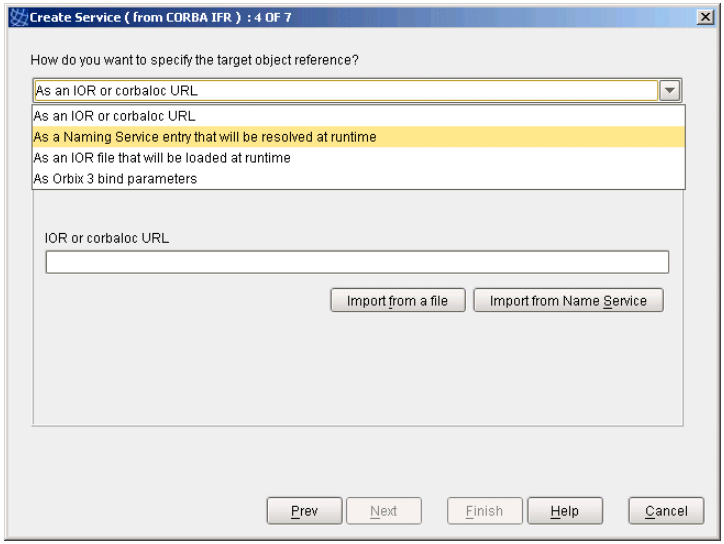

Figure 25: *Specifying the Object Reference*

2. Click **Import from Name Service**. This displays the **Naming Service Browser** dialog, shown in Figure 26.

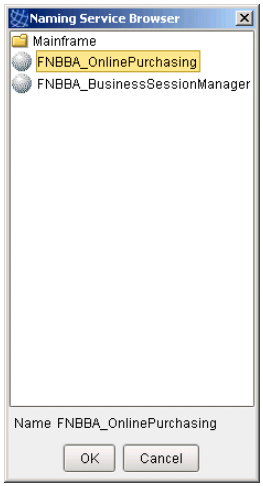

Figure 26: *Naming Service Browser*

3. Select **FNBBA_OnlinePurchasing**.
4. Click **OK**.
5. Click **Next**.

Select the Interface Repository ID

To select the Interface Repository ID, perform the following steps:

1. Click **Browse IFR**.

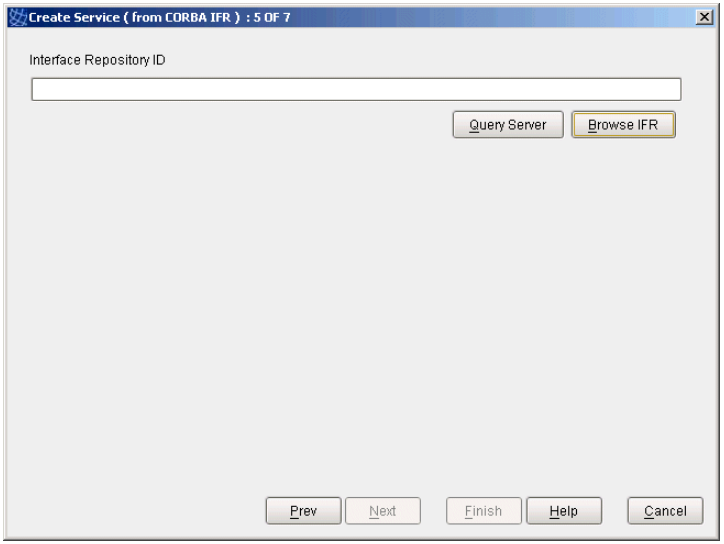

Figure 27: *Selecting the IFR ID*

This launches the **Interface Repository Browser** dialog, shown in Figure 28.

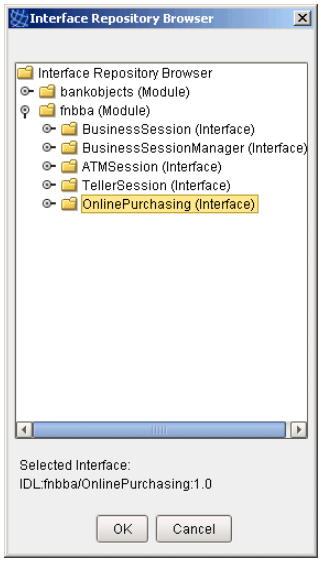

Figure 28: *Interface Repository Browser*

2. Select the **OnlinePurchasing** interface.
3. Click **OK**.
4. Click **Next**.

Select the Web service methods

To select the Web service methods, perform the following steps:

1. In the **METHOD SELECTION** pane, shown in Figure 29, check the following methods

   `makePurchase()`

   `registerMerchant()`

   The unselected methods (for example, `listMerchants()`) will not be exposed a Web service. These will remain for internal use within the banking system (see "Viewing merchants" on page 62.

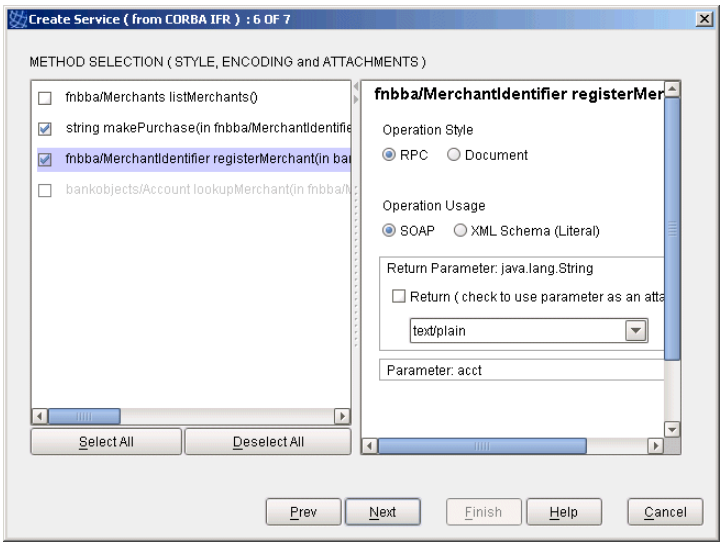

Figure 29: *Selecting Web Service Methods*

2. Click **Next**.

Specify the Output XAR

In the **Output XAR File** field, you must specify the full path to where you wish the XAR file to be created (for example, `c:\temp\FNBOnlinePurchasing.xar`).

You can use the default settings for **Schema Namespace** and **Target Namespace**, which are sufficient for this example.

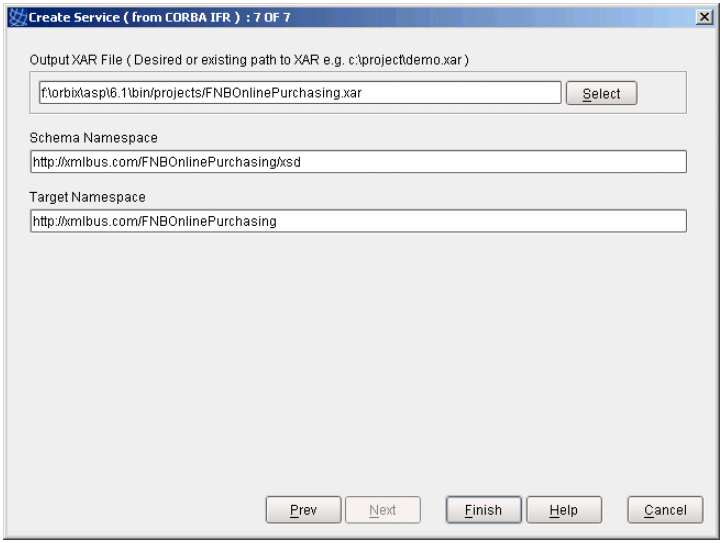

Figure 30: *Specifying the Output Details*

Click **Finish**. The **Web Service Builder** creates the information needed to deploy your Web service, and stores it in the XAR you specified. Figure 31 shows the dialog displayed when the XAR file is created.

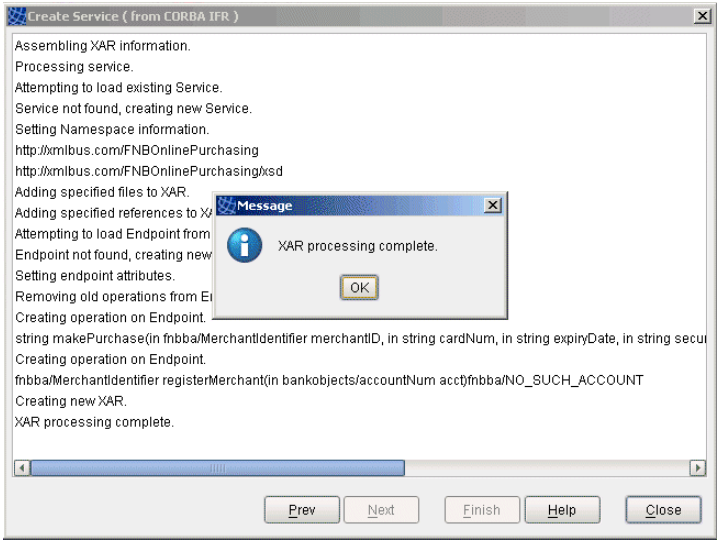

Figure 31: *Completed XAR*

Close the Create Service wizard

To close the **Create Service** wizard, perform the following steps:

1. Click **OK** to close the **Message** dialog.
2. Click **Close** to close the **Create Service** wizard.

This returns you to the **Web Service Builder**. Figure 32 displays the information for the `FNBOnlinePurchasing.xar` Web service in the **Web Service Builder**.

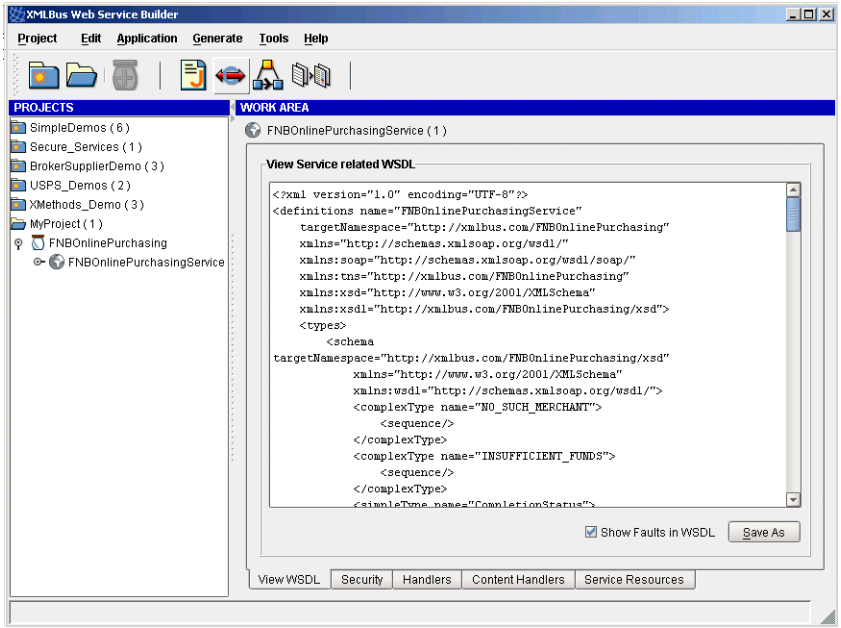

Figure 32: *FNB Web Service Information*

Deploy the Web service application

To deploy the Web service application, perform the following steps:

1. Select your FNB project in the **PROJECTS** pane of the **Web Service Builder**, shown in Figure 33.

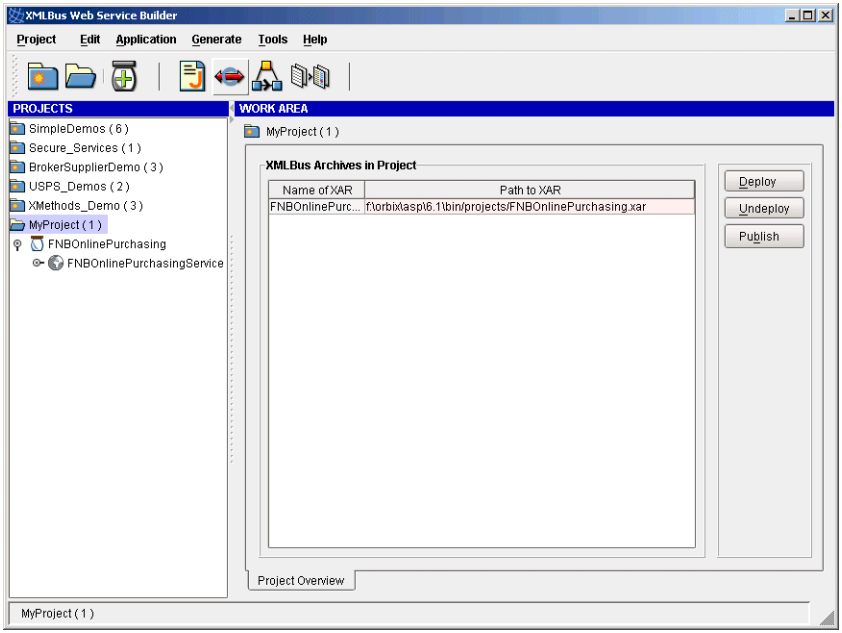

Figure 33: *Selecting the Web Service*

2. From the menu, select **Application**|**Deploy**. The **Deploy a XAR** dialog is displayed, as shown in Figure 34.

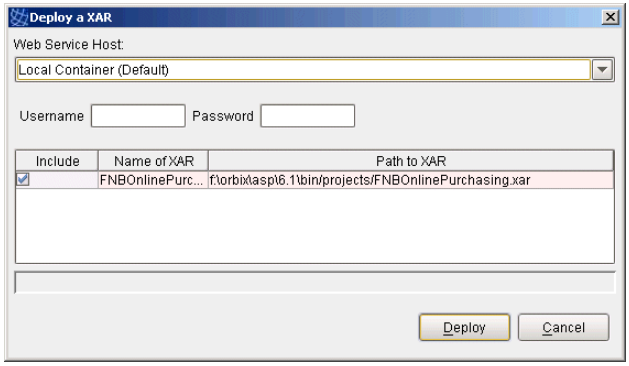

Figure 34: *Deploy a XAR Dialog*

3. Click **Deploy**. This deploys the Web service application into the Web services container.

Running the Web Service

Overview

This section describes how to run the First Northern Bank's Web service application. It includes the following steps:

Step	Action
1	"View the deployed application".
2	"View the Web service details".
3	"Test the Web service".
4	"Invoke the registerMerchant() operation".
5	"Invoke the ConfirmPurchase operation".
6	"View the purchase in the bank".

View the deployed application

You can verify that the Web service application has been deployed by viewing it in the **Web Services Manager**.

In the **Web Services** pane of **IONA Central**, select the **Web Services Manager** button:

Alternatively, enter the following URL in your browser:

```
http://localhost:53205/xmlbus/
```

This displays the **Web Services Manager**, shown in Figure 35. In this example, the deployed `FNBOnlinePurchasing` application is selected.

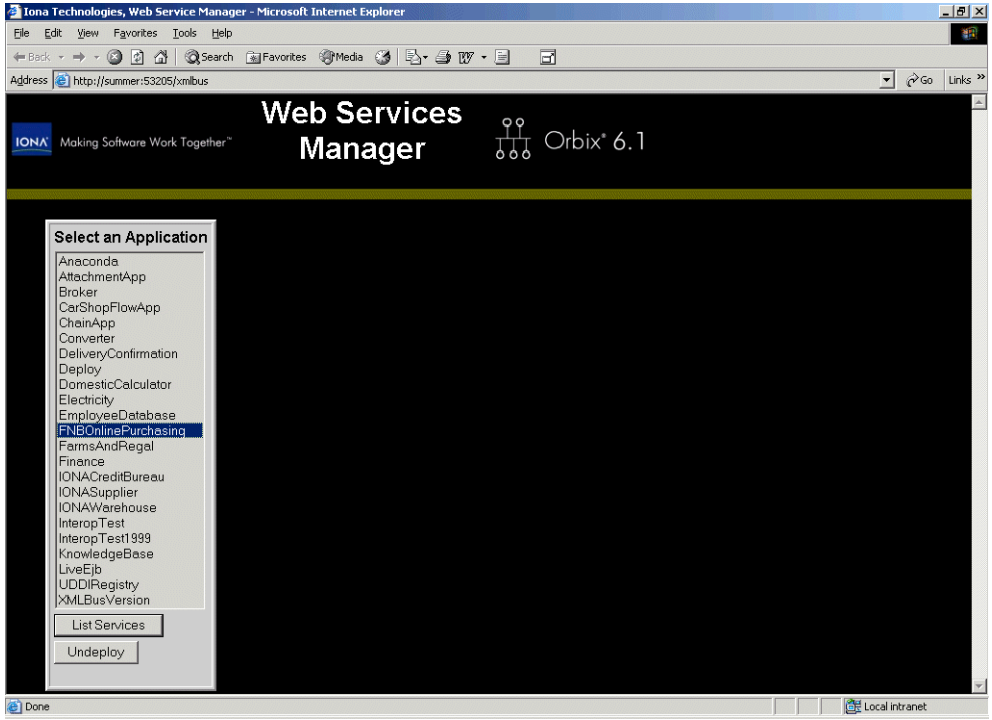

Figure 35: *Web Services Manager*

View the Web service details

To view the URL for the Web service, perform the following steps:

1. Ensure that the `FNBOnlinePurchasing` application is selected.
2. Click the **List Services** button. This displays the Web services for that application. In this example, there is a single service named `FNBOnlinePurchasingService`, shown in Figure 36.

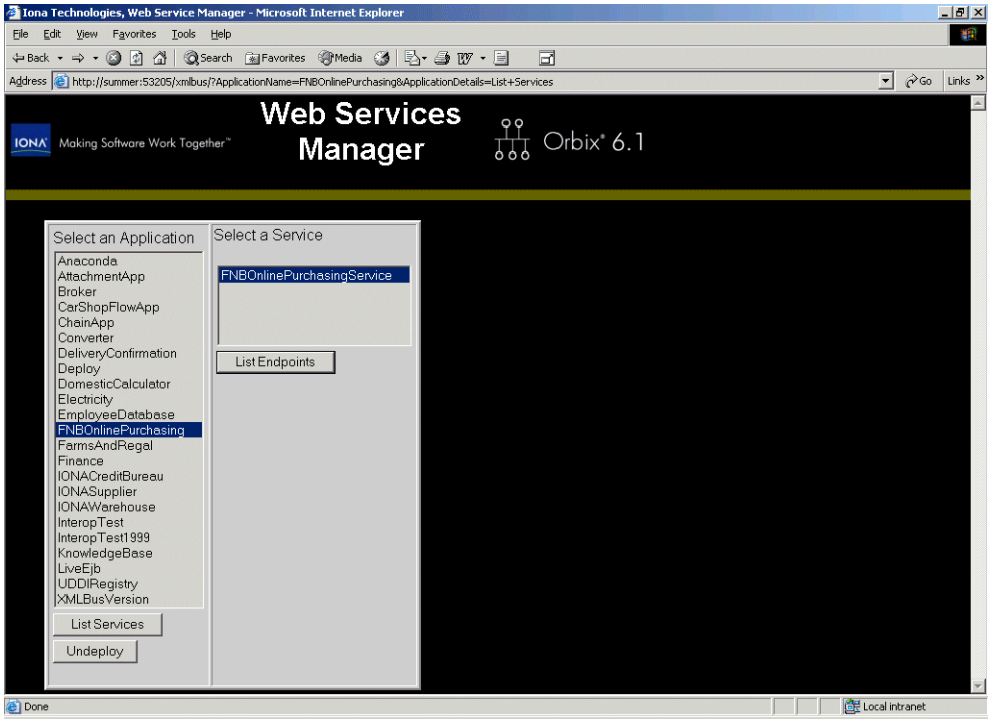

Figure 36: *Listed Web Services*

3. Select `FNBOnlinePurchasingService`, and click the **List Endpoints** button. This displays the endpoint information for the Web service, shown in Figure 37.

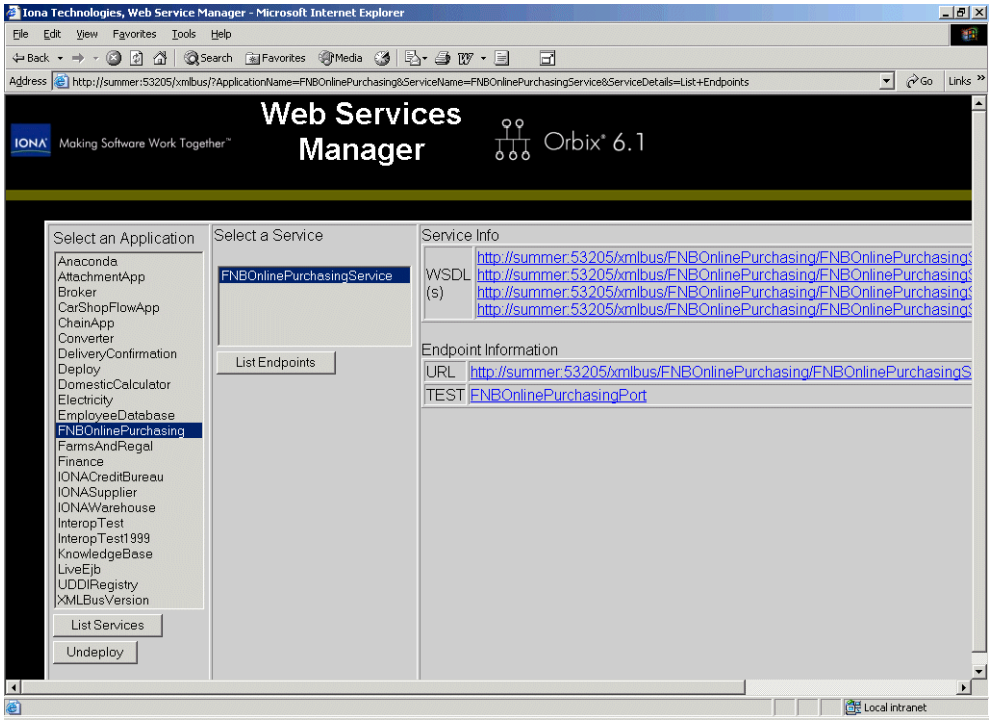

Figure 37: *Web Service Endpoint Information*

Test the Web service

The **Web Services Manager** also enables you to use the **WSDL Dynamic Test Client** to connect to your deployed Web service application, and invoke its operations.

1. In the **TEST** field of the In the **Endpoint Information** table, click the **FNBOnlinePurchasingPort** link, as shown in Figure 37. This displays the **WSDL Dynamic Test Client**, as shown in Figure 38.

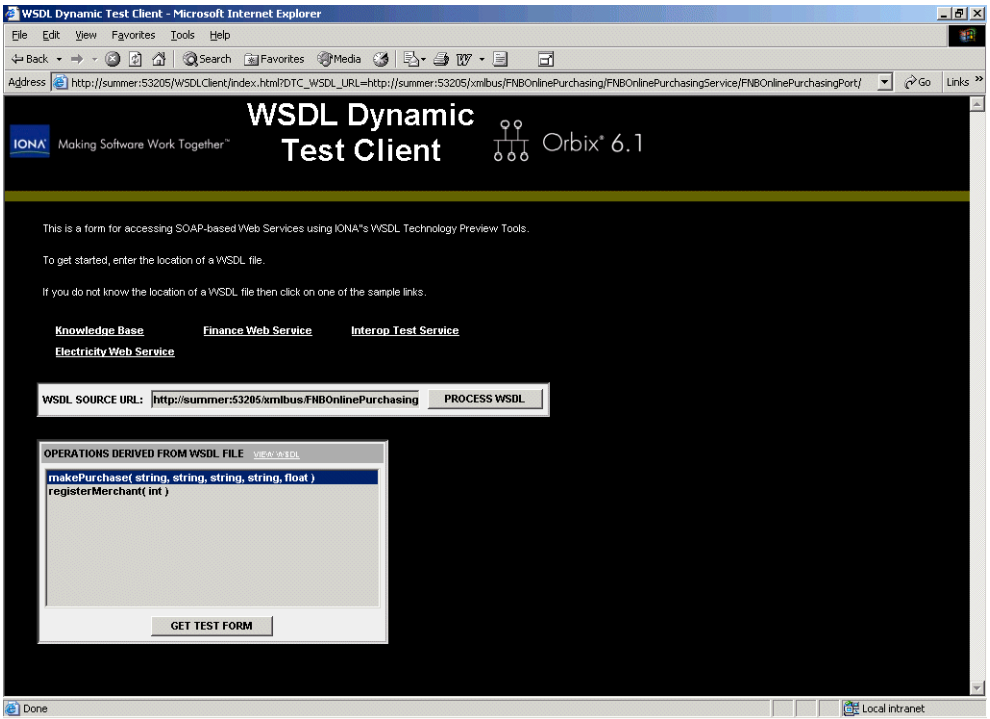

Figure 38: *FNB Web Service Operations*

Invoke the registerMerchant() operation

To invoke the `registerMerchant()` operation, perform the following steps:

1. Select the `registerMerchant()` operation in the test client. This operation registers online retailers to be the beneficiaries of online customer purchases.
2. Click the **GET TEST FORM** button. This displays the test form for the `registerMerchant()` operation, shown in Figure 39.

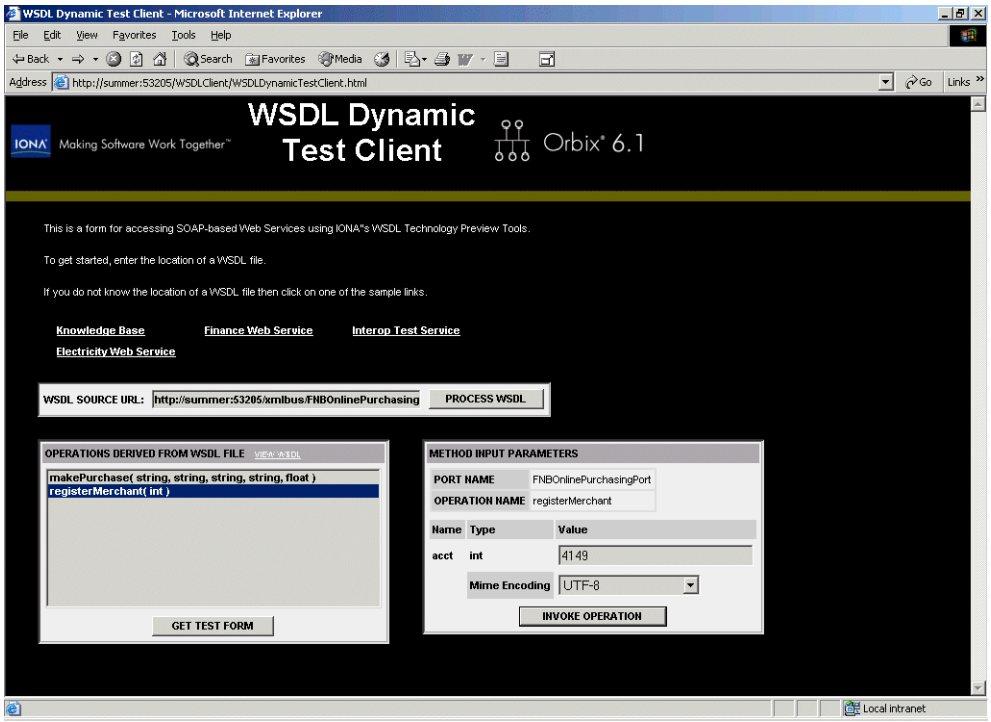

Figure 39: *registerMerchant Operation*

3. Enter the merchant's account current number, which you created earlier, in the field for the first parameter (4149 in Figure 39).
4. Click the **INVOKE OPERATION** button. This returns a merchant ID as the **Return Value** (`390` in Figure 40). Make a note of the value displayed.

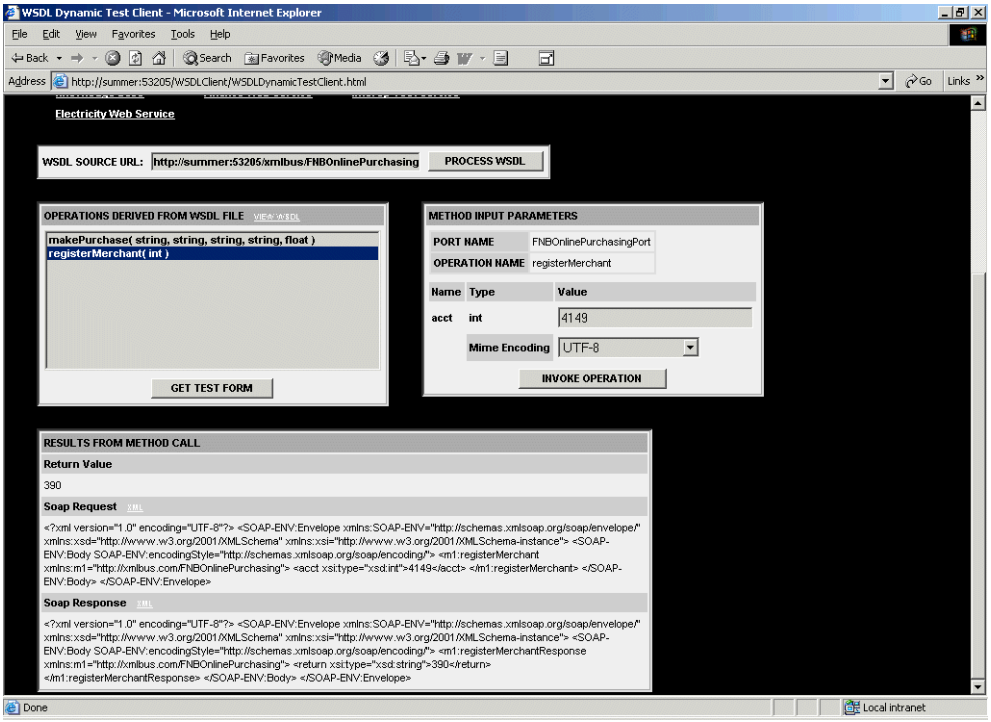

Figure 40: *Returned Merchant ID*

Invoke the ConfirmPurchase operation

To invoke the `makePurchase()` operation, perform the following steps:

1. Select the `makePurchase()` operation.
2. Click the **GET TEST FORM** button. This displays the test form for the `makePurchase()` operation, shown in Figure 41.

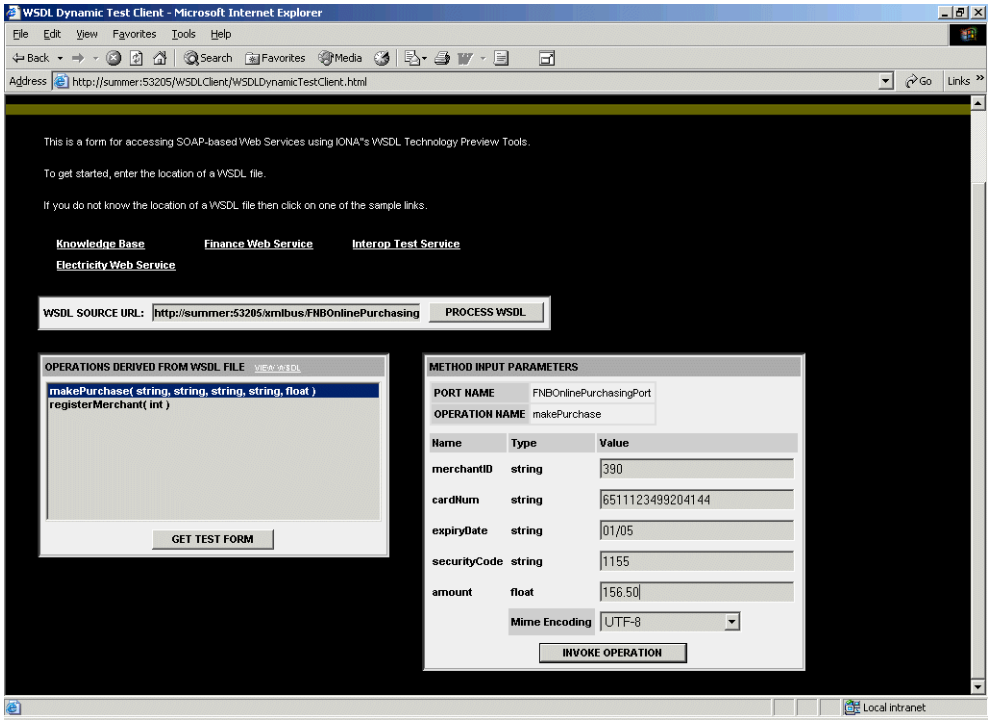

Figure 41: *makePurchase Operation*

3. Enter the following details:
   - merchant ID
   - customer credit card account number (no spaces)
   - expiry date as the third parameter (enter any date)
   - security code (enter any number)
   - amount

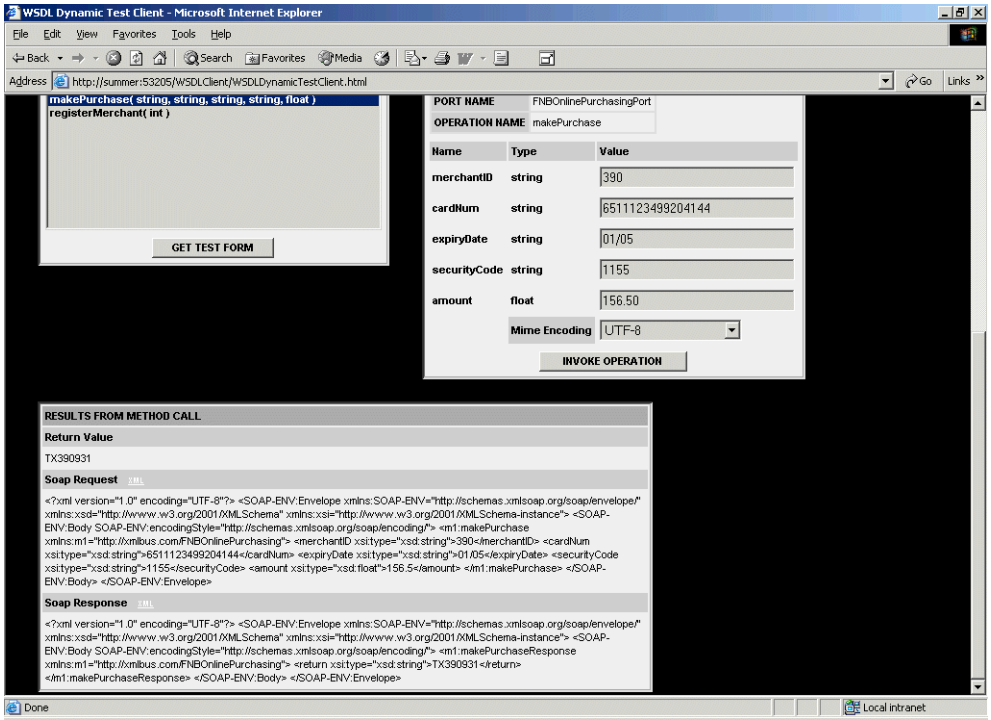

Figure 42: *Purchase Transaction Complete*

4. Click the **INVOKE OPERATION** button. If the operation is successful, this returns a transaction ID (`TX390931` in Figure 42).

   The transaction ID is the only reference stored by the merchant. The bank stores the full details for all transactions (for example, customer credit details, and so on). In case of any problems, the merchant can ask the bank for details.

   This completes invoking on the Web service.

View the purchase in the bank

When the purchase has been confirmed, you can then view the purchase transaction in the **FNB Teller Application**. Figure 43 shows confirmation of the purchase in the customer's credit card account.

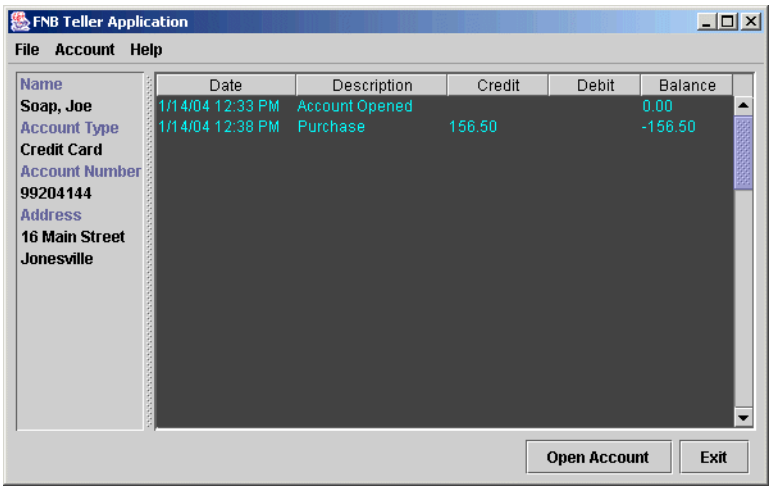

Figure 43: *Confirmation of Purchase*

Figure 44 shows confirmation of the lodgement into the merchant's current account.

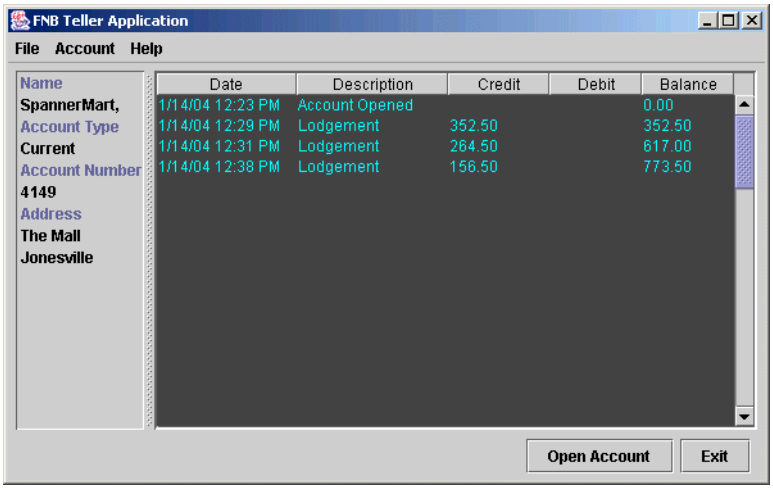

Figure 44: *Confirmation of Lodgement*

Running the C# .NET Remoting Client

Overview

This section describes how to run the C# .NET Remoting client application. This is a simple online purchase management application that runs internally in the FNB bank. It enables bank staff to retrieve details of merchants that are registered with the bank.

This facility to browse merchant details is internal-only feature in the bank. It is not exposed in the FNBOnlinePurhchasing Web service (see "Select the Web service methods" on page 44).

Requirements

To run the C# .NET Remoting client, you must have the .NET 1.1 runtime installed. This is available from Microsoft Windows Update center or with Visual Studio .NET 2003 (VS 7.1).

Populate the IFR

If you did not run the Visual Basic ATM client, you will need to populate the IFR. In the **FNB Demo** pane of **IONA Central**, click **Put the IDL into the IFR**:

Alternatively, enter the following command:

```
itant populate_ifr
```

This populates the CORBA Interface Repository with the Interface Definition Language (IDL) used by the FNB demo.

Starting the .NET client

To start the .NET client, in the **FNB Demo** pane of **IONA Central**, click **C# .NET Remoting Online Purchasing Manager:**

Alternatively, change to the following directory:

```
...common\fnb\onlinepurchasingmanager\onlinepurchasingmanager\bin
   \Release
```

And enter the following command:

```
onlinepurchasingmanager.exe
```

Figure 45 shows the **FNB Online Purchasing Manager** GUI.

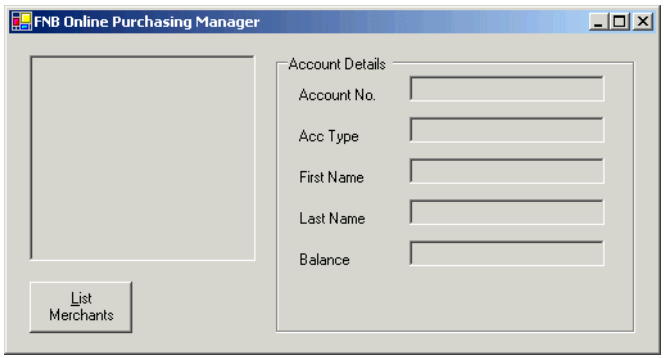

Figure 45: *C# .NET Remoting Client*

Viewing merchants

To view a list of registered merchants, click the **List Merchants** button. This shows a list of merchant IDs.

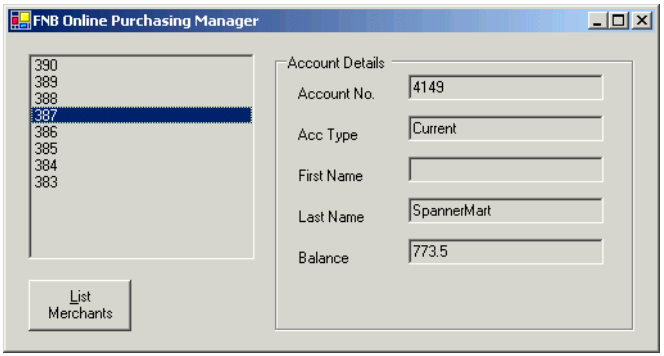

Figure 46: *List Merchants*

CHAPTER 4

Running the Internet Banking Application

This chapter explains how to run the FNB Internet banking application. This sample application shows CORBA and J2EE, components, working together, in an integrated online banking example.

In this chapter

This chapter discusses the following topics:

Setting up your J2EE Environment

J2EE prerequisites

Before running the J2EE application, you must have the following J2EE software installed:

- JDK 1.4.1 or higher.
- Orbix Connect 1.0.
- an application server that works with Orbix Connect (for example, JBoss 3.2.x).

Install Orbix Connect

Orbix Connect implements Sun's Java Connector Architecture (JCA) standard. Orbix Connect is used by the application server to communicate with the back-tier CORBA server. For installation details, see the *Orbix Connect Installation Guide*.

Configure Orbix Connect

To configure Orbix Connect, set your `ORBIXCONN_HOME` environment variable to the location of your Orbix Connect installation, for example:

```
SET ORBIXCONN_HOME=c:\orbix\connect
```

If you have the `IT_LICENSE_FILE` environment variable set, you must also add the contents of your Orbix Connect license file to your Orbix 6.1 license file. You do not need to do this if you do not have `IT_LICENSE_FILE` set.

Install an application server

You must install J2EE 1.3 compliant application server that has been tested with Orbix Connect, for example:

- JBoss 3.2.2
- BEA WebLogic 8.1 SP1
- IBM WebSphere 5.0.2

For installation instructions, see your application server documentation (for example, see http://www.jboss.org/).

Configure your application server

You must ensure that your application server environment is configured correctly. For example, if you are using JBoss, your `JBOSS_HOME` environment variable must be set to the location of your JBoss installation:

```
SET JBOSS_HOME=c:\Program Files\jboss-3.2.3
```

Running the J2EE Application

Overview

This section explains how to run the FNB Internet banking application with JBoss. This is a J2EE web application, named `AllDayBanking`. This section includes the following steps:

Step	Action
1	"Build the J2EE demo files".
2	"Run the application server".
3	"Deploy the J2EE application".
4	"View account details online".
5	"Pay a credit card bill online".

Build the J2EE demo files

To build the J2EE demo files, perform the following steps:

1. Change to the following directory:

   `<install-dir>\asp\6.1\demos\common\fnb\ibank`
2. Enter the following command:

   ```
   itant build
   ```
3. Wait until you see the following message:

   ```
   BUILD SUCCESSFUL
   ```

Run the application server

To run the JBoss application server, perform the following steps:

1. Change to the following directory:

   `<jboss-dir>\bin`
2. Run the following command:

   ```
   run.bat
   ```

Deploy the J2EE application

To deploy the `AllDayBanking` J2EE application, perform the following steps:

1. Enter the following command from your `...\ibank` directory:

   ```
   itant deploy_jboss
   ```

2. Wait until you see the following message:

   ```
   BUILD SUCCESSFUL
   ```

This deploys the `AllDayBanking` application.

View account details online

To log on to the web application, perform the following steps:

1. In the **FNB Demo** pane of **IONA Central**, click **View FNB Web Application Home Page**:

   

   Alternatively, specify the following URL in your web browser:

   ```
   http://localhost:8080/AllDayBanking/
   ```

   The default port number for JBoss is `8080`. This URL displays the logon page for the J2EE application, shown in Figure 47.

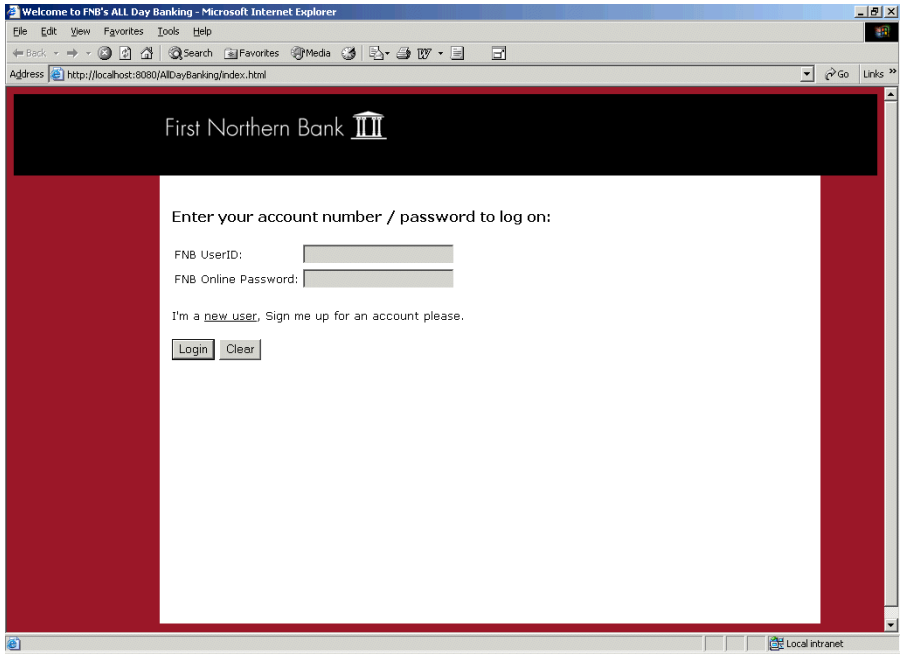

Figure 47: *FNB Web Logon Page*

2. You must create your online user ID the first time you run the internet banking application. To create a user ID, click the **new user** link. This displays the **New User Registration** page, shown in Figure 48.

3. Enter the customer details and account numbers of a customer that you created in "Create a bank account" on page 24.

   Note: Complete all fields in this form. You do not need to enter the first eight digits of the credit card account number.

4. Enter your preferred user ID and online password.
5. Click **Submit**.

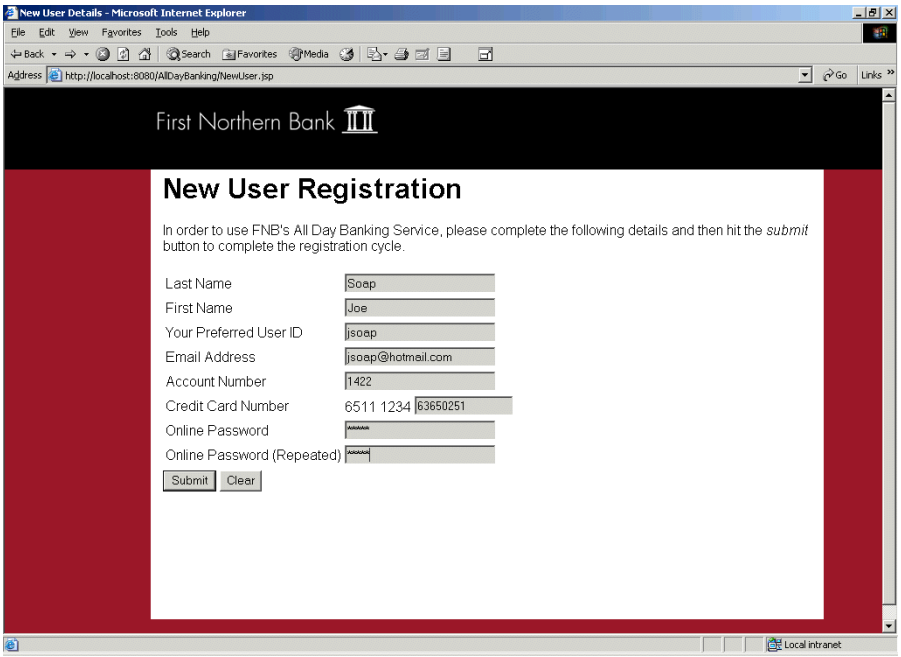

Figure 48: *New User Registration*

6. Click the **AllDayBanking** link to return to the logon web page, shown in Figure 47 on page 67.
7. Logon, using your new user ID and password. This displays your customer accounts, shown in Figure 49. Click an account to view details of the account transactions.

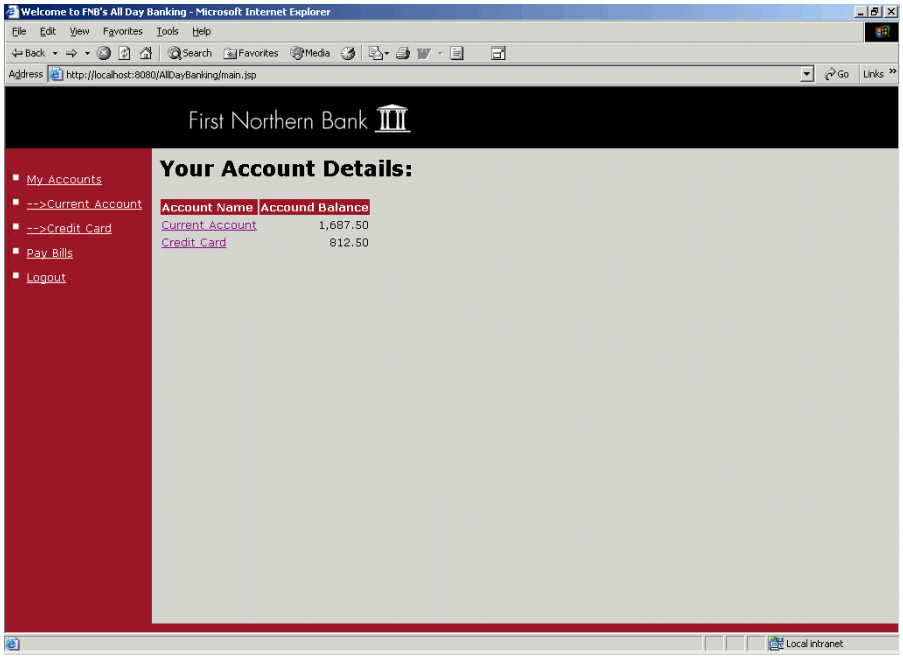

Figure 49: *Online Account Details*

Pay a credit card bill online

To pay a credit card bill online, perform the following steps:

1. Click the **Pay Bills** link on the left of the page. This displays the **Credit Card Bill Payment** page, shown in Figure 50.
2. Enter the amount you wish to pay.
3. Click the **Pay Bill** button.

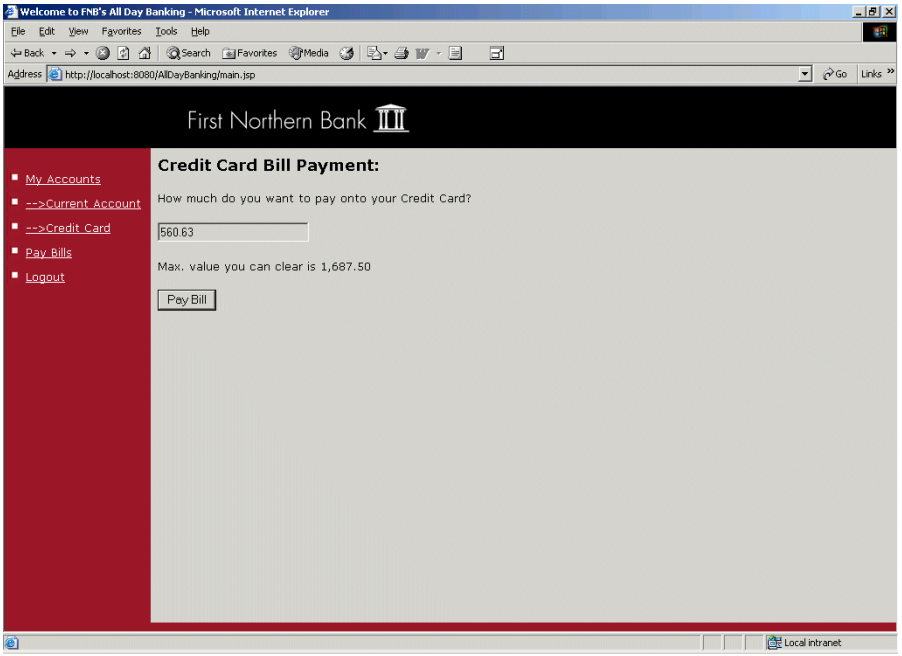

Figure 50: *Online Bill Payment*

Using other application servers

This chapter uses JBoss as an example application server. See the *Orbix Connect User's Guide* for details of how to use other application servers (for example, IBM Websphere or BEA WebLogic).

CHAPTER 5

Managing the FNB Demo

The First Northern Bank demo comes with application management features built-in. This chapter explains how to manage the demo using the IONA Administrator tool.

In this chapter

This chapter discusses the following topics:

Starting IONA Administrator

Overview

This section describes how to start the IONA Administrator Web Console from your web browser, and how to log in as an administrator.

Starting from IONA Central

To start the IONA Administrator Web Console from **IONA Central**, do the following

Step	Action
1	If **IONA Central** is not already running, type the following command to start it up: `itcentral` Alternatively, you can also start **IONA Central** from the Windows **Start** menu.
2	In the **Domain Services** toolbar of **IONA Central**, click the **Administrator Web Console** button: 

Starting from your browser

To start the IONA Administrator Web Console from your browser, type the following URL in the **Address** field of your web browser:

```
http://localhost:53185/admin
```

You can start the web console by specifying the address of any management service host in your browser. To start the web console, use the following URL:

```
http://host:port_number/admin
```

The variable *host* is the name or IP address of the host that the domain's management service is running. The variable *port_number* is the port number of the web server configured for this domain. The default port number is 53185.

Example addresses are:

```
http://localhost:53185/admin
http://hamlet.myco.com:53185/admin
http://192.165.146.12:53185/admin
```

Troubleshooting the web console

The management service requires the following configuration setting for the web console:

```
iona_services{
    management{
        policies:well_known_addressing_policy:http:addr_list =
   ["my-host:53185", "localhost:53185"];
    };
};
```

The variable *my-host* refers to your hostname.

The login dialog

When you start up the IONA Administrator Web Console, the **Enter Network Password** dialog appears. This dialog is shown in Figure 51.

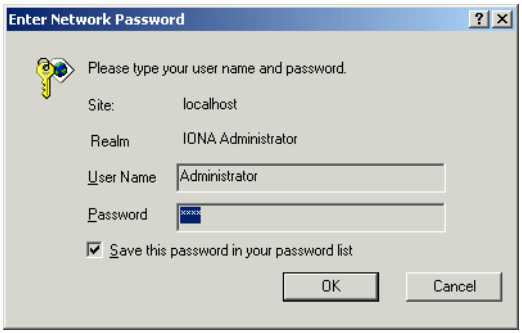

Figure 51: *The Login Dialog*

Logging in as administrator

To login as an administrator, perform the following steps:

Step	Action
1	In the **User Name** field, type `Administrator`.
2	In the **Password** field, type `IONA`.
3	Press the **OK** button.

Note: The **User Name** and **Password** are case sensitive.

The IONA Administrator Web Console

When you have logged in, the IONA Administrator Web Console appears in a browser window, as shown in Figure 52.

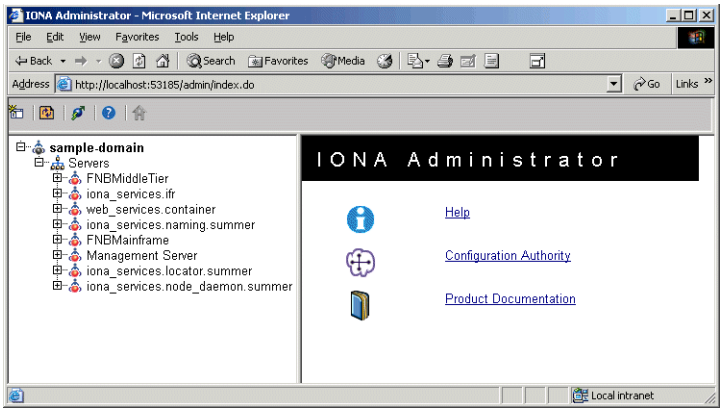

Figure 52: *IONA Administrator Web Console*

Exploring the FNB domain

The navigation tree on the left of the console displays the managed servers running in your domain, in this case `sample-domain`. The navigation tree is the starting point for exploring a distributed application.

To view the managed attributes and operations for a managed server, click on the server in the navigation tree. Details of each server is displayed in the right pane (for example, see Figure 53). You can drill down further to view all its managed data.

Managing the Back-Tier Server

Overview

This section describes the management information available for the FNB simulated mainframe or back-tier server. It includes the following topics:

- "Monitoring the back-tier server".
- "Monitoring accounts".
- "Unloading accounts".

Monitoring the back-tier server

Figure 53 shows the FNB back-tier mainframe server displayed in the console. It shows the managed attributes of this server (for example, `Name` and `State`).

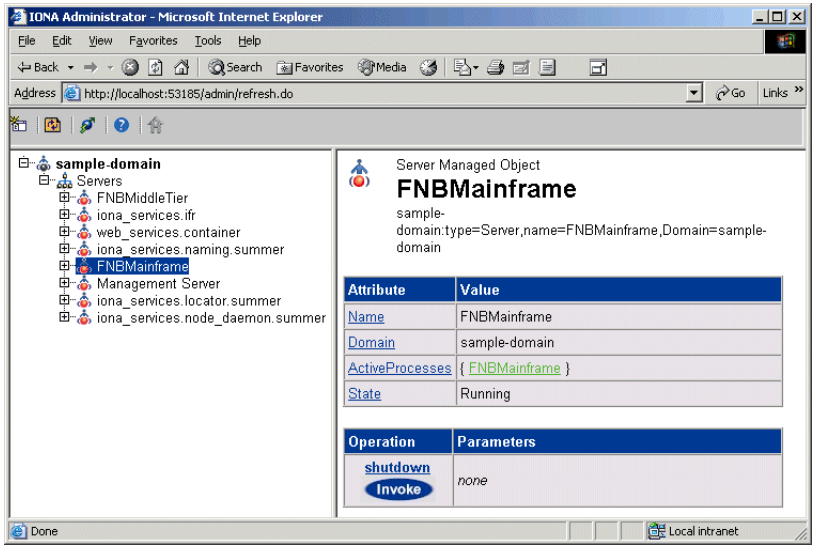

Figure 53: *FNB Mainframe Server*

Monitoring accounts

Click on the **AccountManager** icon to display details of the accounts stored in the FNB mainframe server, as shown in Figure 54.

The server attributes and operation displayed are examples of the sort of sever information that can be exposed for management (for example, the `NumberOfLoadedAccounts` attribute).

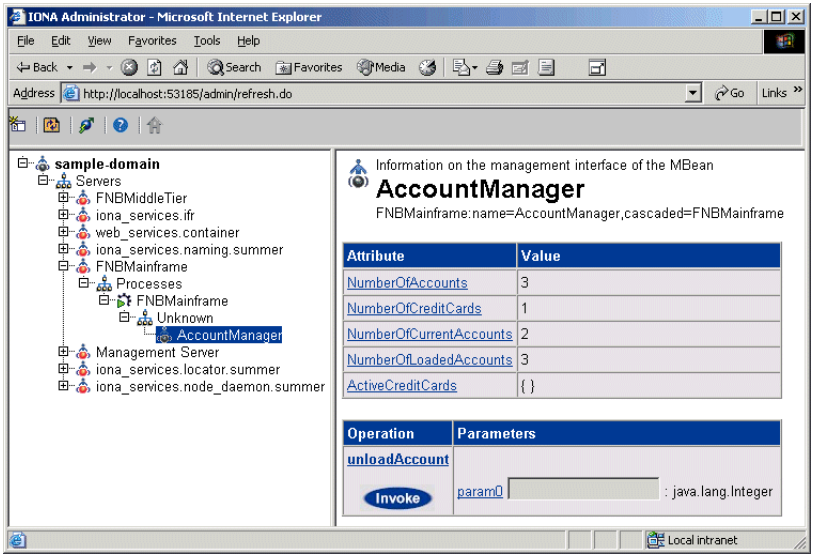

Figure 54: *FNB Account Manager*

Unloading accounts

You can unload an account from memory using the `unloadAccount()` operation, shown in Figure 54. To unload an account, perform the following steps:

1. Enter the account number to be unloaded in the **Parameters** field (for example, `1548`).
2. Click the **Invoke** button. When the account has been unloaded, the dialog is displayed as in Figure 55.
3. Click **OK** to return to the **IONA Administrator Web Console**. The **AccountManager** `NumberOfLoadedAccounts` attribute decreases in value by one.

The `unloadAccount()` operation only removes the account from memory; it will be automatically reloaded whenever the account is accessed again (for example, in the **FNB Teller Application**).

Figure 55: *Account Successfully Unloaded*

Managing the Middle-Tier Server

Overview

This section describes the management information available for the FNB middle-tier server. It includes the following topics:

- "Monitoring the middle-tier server".
- "Monitoring sessions".
- "Monitoring tellers".

Monitoring the middle-tier server

Figure 56 shows the FNB middle-tier server displayed in the console. It shows the managed attributes of this server (for example, `State`).

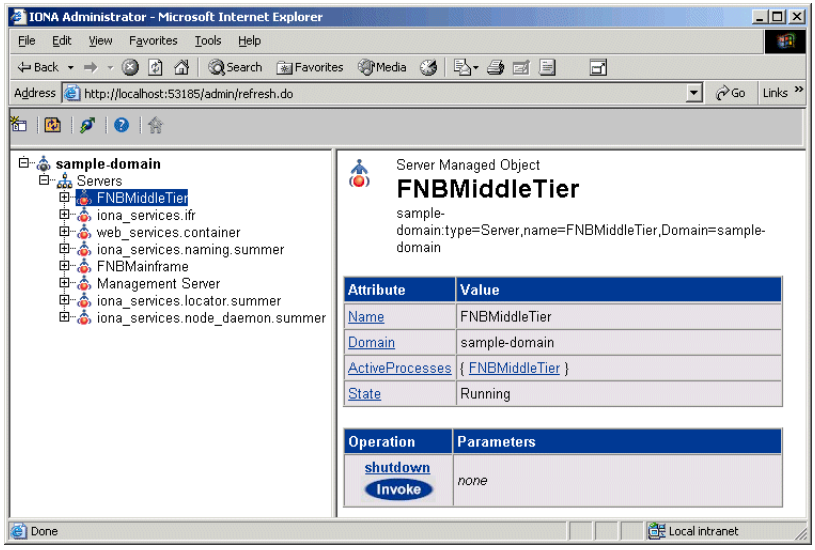

Figure 56: *FNB Middle-Tier Server*

Monitoring sessions

Click on the **BusinessSessionManager** icon to display the current number of open sessions, as shown in Figure 57. This is the number of bank tellers and online purchasing sessions that are currently logged on to the system.

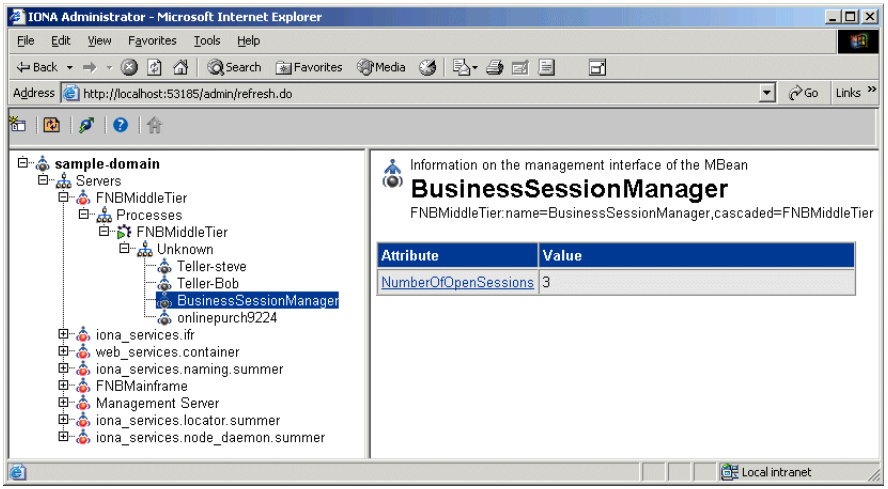

Figure 57: *FNB Business Session Manager*

Monitoring tellers

Click on a teller icon to display details of the recent transactions performed by the teller.

Figure 58 shows the display for **Teller-steve**.

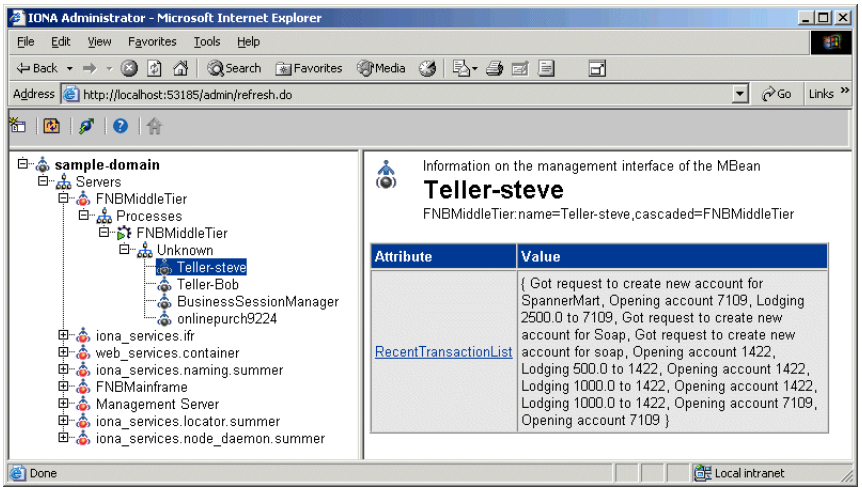

Figure 58: *FNB Teller Session*

Shutting Down the FNB Demo

Overview

This section explains how to shut down the various components of the FNB demo in reverse order to how they were started. This includes the following steps:

Step	Action
1	"Shut down the bank teller client".
2	"Shut down the middle-tier server".
3	"Shut down the back-tier server".
4	"Shut down your domain".

Shut down the bank teller client

To shut down the bank teller GUI, select **File**|**Exit** from the **FNB Teller Application**.

This shuts down the GUI client application and its background process. The teller session for the client application will no longer be displayed in the **IONA Administrator** web console**.** For example, in Figure 58, `Teller-Bob` will no longer be displayed.

Note: If you have multiple **FNB Teller Application**s running, you must shut down each application individually.

Shut down the middle-tier server

To shut down the middle-tier server, perform the following steps.

1. Select the `FNBMiddleTier` server in the IONA Administrator navigation tree.
2. Click the **Invoke** button for the **shutdown** operation in the details pane, as shown in Figure 59**.** This shuts down the FNBBA middle-tier server
3. Click the **Refresh** button in the toolbar:

   

   The server is no longer displayed in the console.

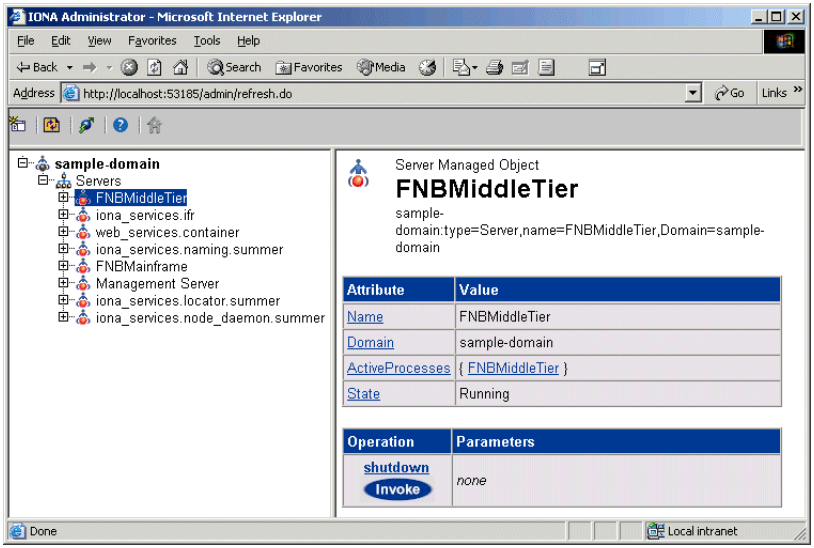

Figure 59: *Shutdown Operation*

Shut down the back-tier server

To shut down the back-tier server, perform the following steps.

1. Select the `FNBMainframe` server in the IONA Administrator navigation tree.
2. Click the **Invoke** button for the **shutdown** operation in the details pane, as shown in Figure 59**.** This shuts down the FNB back-tier server.
3. Click the **Refresh** button in the toolbar. The server is no longer displayed in the console.

Shut down your domain

To shut down your domain services, in **IONA Central** click the **Start Domain Services** button:

Alternatively, change to the `<install-dir>\etc\bin` directory, and run the following command:

```
stop_sample-domain_services
```

The console will not display server information for the `sample-domain` because the management service is no longer running.

Note: You do not need to explicitly shut down the console. To log out, simply specify a different URL in your browser.

You have now successfully shut down the all components of the FNB demo.

Further information

For more information on IONA Administrator, see the *Orbix Management User's Guide*.

For information on the business case behind the FNB system, see the *First Northern Bank Business Case.*

Finally, for information on how the FNB system was developed, see the *First Northern Bank Developer's Introduction*.

Index