Natural Business Services

Natural Business Services Getting Started

Version 5.3.1

February 2010

Natural Business Services

This document applies to Natural Business Services Version 5.3.1.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 2006-2010 Software AG, Darmstadt, Germany and/or Software AG USA, Inc., Reston, VA, United States of America, and/or their licensors.

The name Software AG, webMethods and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA, Inc. and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Use of this software is subject to adherence to Software AG's licensing conditions and terms. These terms are part of the product documentation, located at http://documentation.softwareag.com/legal/ and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices and license terms, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". This document is part of the product documentation, located at http://documentation.softwareag.com/legal/ and/or in the root installation directory of the licensed product(s).

Table of Contents

1 Natural Business Services Getting Started 1
2 Introduction 3
    Create a Web Service for Your Business Service 4
    Use Unicode Parameters for Your Business Service 5
3 Overview of Natural Business Services 7
4 Architecture of Natural Business Services 9
    Server Components 11
    Client Components 14
5 Roles and Prerequisites 17
    Introduction 18
    Understanding Prerequisites 18
    Types of Security Available with EntireX 21
    Summary of Tasks 22
6 Supplied Demo Applications 33
    Business Service Types 34
    Features of the Demo Application 37
    Additional Features When Using Predict 40
7 Using Natural Business Services 41
    Administer Business Services 42
    Configure the Plug-in 42
    Create a New Business Service 42
    Create a Client Consumer 47
8 Installation Tips 49
    Troubleshoot EntireX and SPoD Connections 50
    Resolve IIS (Internet Information Server) and ASP.NET Issues 51
9 Related Documentation 53
A Appendix A: Glossary 55

1 Natural Business Services Getting Started

Natural Business Services Getting Started provides an overview of Natural Business Services and the architecture of the product and its components. This documentation describes the supplied demo applications and contains step-by-step instructions to create and maintain business services, client consumers of the services, and user interfaces.

This documentation is intended for developers and others who are new to Natural Business Services and want to learn how to create and maintain business services.

Natural Business Services Getting Started covers the following topics:

Introduction	Introduces Natural Business Services and provides information on other sources of information about creating and maintaining business services. It also describes the different ways to create Web services for your business services, as well as how to generate and invoke services using Unicode parameters.
Overview of Natural Business Services	Provides an overview of Natural Business Services.
Architecture of Natural Business Services	Describes the architecture of Natural Business Services and its components.
Roles and Prerequisites	Describes the roles of different users and helps new users understand which prerequisites are required for each business service activity.
Supplied Demo Applications	Describes the demo applications supplied with Natural Business Services.
Using Natural Business Services	Provides a step-by-step example of how to create a business service and provides links to additional information.
Installation Tips	Contains information on resolving problems you may encounter after installing Natural Business Services.
Related Documentation	Contains links to other documents in the Natural Business Services documentation set.

	Appendix A: Glossary	Contains definitions of terms used throughout the Natural Business Services documentation set.

Note: For more information on Natural Business Services, see Understanding Natural Business Services.

2 Introduction

- Create a Web Service for Your Business Service 4
- Use Unicode Parameters for Your Business Service 5

Natural Business Services allows you to create and maintain business services. Each business service combines a group of methods related to a common business entity, such as a customer or order. Processing for the methods is supplied by either existing or wizard-generated Natural subprograms. The definition for each service is stored in the business service repository and identifies the associated methods.

As data typically comes from different platforms and uses different character sets, each business service requires a subprogram proxy to retrieve data. All business code is contained in the subprogram that is called by the proxy.

During generation, the Business Service wizard:

- Creates the subprogram proxy
- Populates the business repository with information about the methods used by the service
- Provides the domain, service, and version specifications
- Adds the method and service descriptions to the business repository explorer

For information on how Natural Business Services creates business services, see Understanding Natural Business Services.

This section covers the following topics:

Create a Web Service for Your Business Service

Natural Business Services provides different techniques you can use to create Web services for existing business services. Depending on the product you prefer and your development requirements, you have several choices. The following table describes three techniques to create Web services:

Component/Location	Technology	Description
Eclipse plug-in	Java, Apache AXIS	The Natural Business Services Eclipse plug-in generates Web service classes and descriptor files that are understood by the Apache AXIS framework. The plug-in provides a wizard to generate a client proxy (which is used to translate Natural data formats to the appropriate language formats) and, optionally, a Web service for an existing business service. Once the proxy has been created, the features of Eclipse can be used. In addition, the Eclipse plug-in provides an option to generate default tests based on all the methods used by a business service. Once generated, you can deploy the Web service using the WS-Stack plug-in.

Component/Location	Technology	Description
Visual Studio add-in (ASMX method)	Natural Development Server (NDV) and EntireX	The Natural Business Services Visual Studio add-in generates .NET client proxy classes you can use in an ASP.NET Web service project. You can then expose the properties and methods to a Web service class. This method requires coding on your part (for an example of this method, see SAMPLES in the Natural Business Services Installation folder). This add-in provides two wizards: one allows you to select an existing business service from the Business Service repository explorer and then generate a client proxy (which is used to translate Natural data formats to the appropriate language formats) and another generates a Web service for an existing business service. Once the proxy has been created, the features of Visual Studio can be used. For example, .NET's Intellisense can provide the attributes for the business service. In addition, the .NET add-in provides an option to generate default tests based on all the methods used by a business service.
	.NET runtime and IIS (Internet Information Server)	In addition to generating business services, the Visual Studio add-in allows you to create a Web service that uses the IIS (Internet Information Server) .NET runtime to execute the Web Service Engine (WSE). Although the business service is written in Natural, you can code in .NET within user exits. The add-in also allows you to send and receive SOAP messages from a Web service and generate the basic components of a web application (for example, HTML, JavaScript, or XSL). **Note:** For more sophisticated web application development, ask about Software AG's web application development tools.

Note: Although the Natural Development Server (NDV) communication is available for the Visual Studio add-in and can be used at runtime, it is not recommended. NDV is intended for a development environment with relatively few users, while EntireX communication is intended for production applications that have many users.

Use Unicode Parameters for Your Business Service

If your Natural environment handles Unicode, the Business Service wizard can generate and invoke services that use Unicode parameters.

According to their implementation in Natural, Unicode values must be stored in fields with U format. This means that subprograms that implement your business services will have one or more U format parameters in their parameter data areas (PDAs).

Natural Business Services does not require special settings to generate business services containing Unicode format fields. However, your runtime environment must use the supplied Unicode dis-

patcher to consume Unicode services. To allow this, run the CSRLOAD utility and enter "Y" in the Unicode field. CSRLOAD creates the Natural Business Services server definitions.

Note: Although the Unicode dispatcher (implemented via SPSDISPU) can process both Unicode and non-Unicode requests and replace the standard dispatcher (implemented via SPSDISP), it will only run in a Unicode-enabled Natural environment.

For more information on the CSRLOAD utility, see:

- Mainframe platforms: Load Default Repository Data
- Unix platforms: Load Default Repository Data

For more information on Unicode support, refer to the Natural documentation.

3 Overview of Natural Business Services

A Natural business service provides a business perspective of a grouping of Natural subprograms. This perspective includes such things as the:

- Business service description
- Service methods and method descriptions

Retrieval algorithms are available to allow quick and easy searches of the business service repository. This allows a business analyst to quickly determine if a particular service currently exists or if one must be created. In addition, you can use the business service repository explorer to deploy business services from one environment to another and/or apply security at a domain, business service, and/or method level.

Note: The Natural subprograms used for the supplied business services are located in the SYSBIZDE library on the server.

Each method defined for a business service is typically associated with a different Natural subprogram. If a method does not have an associated subprogram, a default subprogram for the entire service is required. In addition, every subprogram that is exposed through a business service requires a subprogram proxy.

The attributes for each method are determined by the subprogram invoked for that method. These attributes can be used to describe the business service or as input, output, input/output, or state values for the service. They can be acted upon and/or changed based on which methods have been defined for a service. For example, the Customer service in the demo application contains the Update method and the Name and Phone Number attributes, which allows the service to be used to change the phone number for a customer.

Once a business service is created, it can be consumed by client proxies generated in Visual Studio or Eclipse. You can also use Visual Studio or Eclipse to generate Web services.

You can quickly develop business services using the Business Service wizard supplied with the Natural Business Services plug-ins. This wizard recognizes the style of service you require based on your answers to simple questions asked on the wizard panels.

Although you can create a business service for any Natural subprogram that does not contain user interface code (i.e., a Natural subprogram that was not generated using Natural Construct and does not contain user interface code), this type of subprogram would only have one method associated with it — DEFAULT, which executes the associated subprogram as it normally would in Natural. An example of this type of business service is the ErrorMessageTesting service in the DEMO domain. This service invokes the FLIPSTY subprogram proxy, which in turn executes the FLIPSTR subprogram.

For more functionality, you can wrap several subprograms into one subprogram and create additional methods. An example of this type of service is the CalculatorAdvance service in the DEMO domain. When this service was created, the Business Service wizard recognized that the developer wanted to use more than one subprogram. The wizard prompted the developer to define the additional methods based on which subprograms to call and in what order to call them. The methods were enhanced by adding user exit code between the calls to these subprograms.

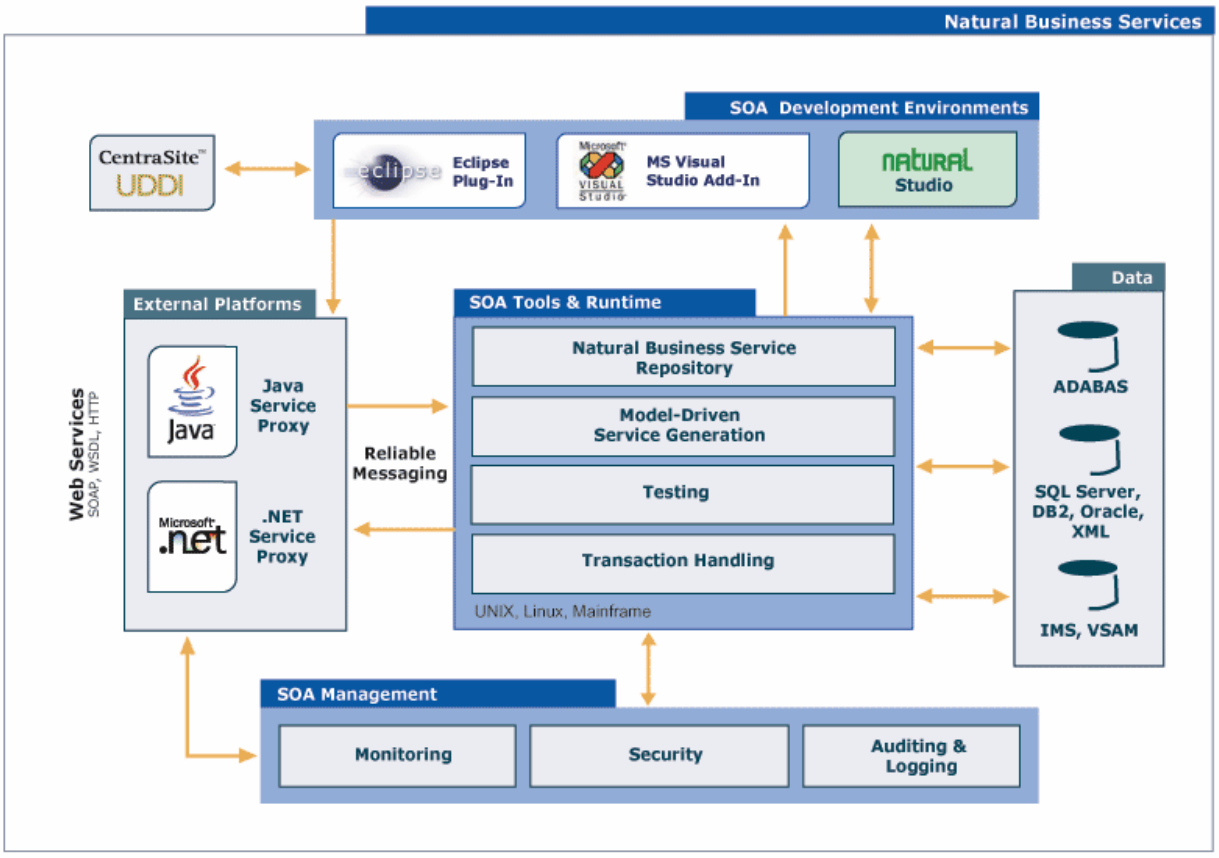
While this functionality is possible without using Natural Construct and Predict, you can enhance the functionality by generating business services based on files that have been defined in Predict. Standard methods, such as Delete, Next, Update, Browse, and FindBy*, can be automatically created based on input to the Business Service wizard and the file definitions set up in Predict.

Note: If you would like to include the additional functionality provided by Natural Construct and Predict, but your applications were created without using these products, you should consider Natural re-engineering. Re-engineering analyzes existing monolithic applications and uses Natural Construct models to convert them into client/server style applications.

4 Architecture of Natural Business Services

- Server Components 11
- Client Components 14

Using Natural Business Services, you can create all the components of a business service, including Natural object subprograms that perform maintenance and browse functions and GUI dialogs or web pages that communicate with the object subprograms. Communication between server and client components of an application is performed by a combination of EntireX and Entire Net-Work (or EntireX configured to use TCP/IP), as well as Natural Business Services middleware components: the business service client proxy and Natural Business Services servers. The middleware components encapsulate calls to EntireX on the client and server. The following diagram shows the architecture of character-based Natural applications and business service components:



This section describes these components according to the platforms on which the components run.

Server Components

This section describes the server components for Natural Business Services. The following topics are covered:

- Required for Development
- Required at Runtime
- System Functions

Required for Development

The following table lists the components required for development purposes:

Component	Description
Natural subprograms	Subprograms written in Natural that do not contain user interface code (for example, WRITE, DISPLAY, PRINT, INPUT, and REINPUT statements) or navigation code (for example, PF-key processing). They can be existing Natural subprograms or they can be wizard-generated in Natural for Windows. Existing subprograms can be wrapped together so one server subprogram accesses more than one subprogram. The Business Service wizard can wrap the subprograms it generates, as well as use Natural Construct models internally to generate subprograms that perform maintenance and browse functions on the server. The wizard chooses the appropriate model based on criteria the user has selected. These models are: Object-Browse-Subp, Object-Maint-Subp, Object-Browse-Select-Subp, and Object-Generic-Subp. **Note:** 1. The Object-Browse-Select-Subp model has the same type of functionality as the Browse-Select model, but is designed for a client/server environment where only *n* rows are processed at a time. 2. The Object-Generic-Subp generates a business service that uses more than one pre-existing subprogram. The same set of business objects can be accessed from character-based Natural applications, client/server applications, and web applications. This ensures that the integrity of business data is preserved, independent of the presentation layer, and existing code can be preserved.
Character user interface (optional; only used at sites that access the business service from a 3270 client)	Non-distributed Natural applications created with Natural Construct accessing subprograms directly (for example, subprograms generated by the Object-Maint-Dialog model).

Component	Description
Subprogram proxy	Link between a specific subprogram and the Natural Business Services dispatch server. The subprogram proxy: ■ Provides a common interface so that the dispatch server can pass the same set of parameters to any subprogram proxy ■ Issues a CALLNAT to the subprogram ■ Converts the parameter data of the subprogram into a format that can be transmitted between the client and server ■ Supports optimization of the data passed through the network so that only input parameters must be sent to the dispatch server and only output parameters must be returned to the client ■ Validates the format and length of the data received from the client ■ Supports debugging features to help uncover inconsistencies between the data sent by the client and the data expected by the subprogram proxy ■ Populates the repository with the domain name, business service name, version, and default methods during generation. The default methods are determined by the subprogram for which the proxy was created. For example, an object maintenance subprogram has default methods such as Add, Delete, and Next and wrapped Natural subprograms have their own methods in the specifications. The subprogram proxy adds these methods to the repository.

Required at Runtime

The following table lists the components required for runtime purposes:

Component	Description
Business Service Administration subsystem	Server subsystem that allows system administrators, application administrators, and developers to set up and manage system and application environments. **Note:** Although this subsystem also provides access to the Business Service repository, we recommend that you use the Eclipse or Natural plug-in to access the repository. Only system environment options will be discussed from the server perspective.
EntireX	Runtime component that transfers messages between Windows or the web server and the Natural environment. EntireX can be configured to use either native TCP/IP or Entire Net-Work as the transport layer. EntireX performs the following runtime functions: ■ Encrypt and decrypt data (set in the Broker attribute file) ■ Compress and decompress data (set in the Broker attribute file) ■ Translate data (handled automatically and has defaults you can customize)

Component	Description
	Note: 1. As this component is separate, it can be used without Natural Business Services and may already be installed. It is a required component for the Natural Business Services system environment. 2. You do not require EntireX to create or test business services. It is used to define security or to balance loads in a runtime environment.
Natural Business Services dispatch server	Server that provides a common interface and EntireX services for Natural subprograms in the application. The main functions of the Natural Business Services dispatch server are to: ■ Receive requests from the client through EntireX ■ Optionally decompress and/or decrypt and translate the request message from the client's character set (ASCII) to the server's character set (either ASCII or EBCDIC) ■ Check security to ensure that the client is permitted to issue the request ■ Determine the name of the subprogram proxy that handles the request ■ Issue a CALLNAT to the subprogram proxy, passing the received message as a parameter string ■ Optionally compress and/or encrypt the message to be returned ■ Send information received from the subprogram proxy back to the client application **Note:** For more information, see **System Functions**.
Dispatch server data	Information that is defined and maintained in the Business Service Administration subsystem and accessed by Natural Business Services dispatch servers anywhere on the network using EntireX.
Security server	Server used to check security settings in the Business Service Administration subsystem to determine whether to grant client requests. This stand-alone server operates independently of any one Natural Business Services dispatch server, which allows the server to centrally process the requests of several dispatch servers located on nodes throughout the network.
Attach server	Server that determines which dispatcher to use and whether other dispatchers are required. If other dispatchers are required, the attach server will start them.
Business Service repository	Directory structure containing the business service metadata, such as domains, subprogram proxies, descriptions, methods, method descriptions, as well as security access to these services and methods.

System Functions

All Natural Business Services dispatch servers defined in the Business Service Administration subsystem have access to the following common system functions:

Function	Description
Return debugging information	Ensures that all requested debugging information is generated into the source area. Debugging information is requested by setting a trace option in the subprogram proxy. The debugging information is stored as a source member that can be examined or used to initiate the request locally on the server, thereby removing the client and the network from the test.
Handle errors	Captures runtime errors and returns the errors to the client. If possible, this function also restarts the server that ended with the runtime error.
Handle messages	Returns a message string based on a message number and substitution values. This function accepts and updates the data used by the Natural Business Services dispatch server to return the message.

Client Components

The client components for Natural Business Services are:

- Natural for Windows
- Business Service Consumers

Natural for Windows

The Natural for Windows components on the client are:

Component	Description
Program generation plug-in	Interface into Natural Construct.
Business Service plug-in/add-in	Interface into the Business Service repository. This interface also provides a wizard to generate business services.
Natural Development Server (NDV)	Middleware component used to connect the client to a Natural server. EntireX is not required in a Natural for Windows environment.
Business Service repository explorer	Tree view of the business repository. Using the repository explorer, you can: ■ Perform the following tasks using the context menu for the Business Services node: ■ Search for and find specific services across domains

Component	Description
	■ View the security audit window to determine if anything unusual is happening (the security audit window can also be accessed by selecting **Business Services** from the **Tools** menu) ■ Perform the following tasks using the context menu for the Domains node: ■ Invoke the Business Service wizard to create a new business service for the domain ■ Refresh the list of available business services in the domain ■ Search the business services using different search techniques ■ Deploy all the services in the domain to a specified environment (excluding the Natural modules) ■ Implement security for the services in the domain ■ Audit activities pertaining to the domain ■ Delete the domain and all services in the domain ■ Edit the domain to modify its description or steplib chain definition ■ Perform the following tasks using the context menu for a business service: ■ Test the service ■ Deploy the service to another environment ■ Audit the service to determine who changed it and when ■ Regenerate the service proxy; this is required if the service interface (PDAs) changes ■ View the subprograms used by the service and, optionally, open a subprogram in the Natural editor ■ Delete the service ■ Edit the service to modify descriptions, add or delete methods, or change subprogram proxies
Business Service wizard	Wizard used to create business services from existing Natural subprograms and/or generate new subprograms. Internally, the wizard generates Natural subprogram proxies and Natural Construct objects and populates the Business Service repository.

Business Service Consumers

You can create business services in the Visual Studio and Eclipse development environments, as well as consume the services. Use the Natural Business Services Visual Studio add-in or Eclipse plug-in to create classes and Web services from business services.

The business service consumer components on the client are:

Component	Description
Business Service add-in or plug-in	Link to Natural Business Services from Visual Studio or Eclipse. Using the add-in or plug-in, you can configure the business service connections, search for business services, and invoke the client proxy and/or Web Service wizard to generate classes and/or Web services (.NET or C# in Visual Studio or Java in Eclipse).
Business Service menu	Menu used to perform the following tasks: ■ Open the Business Service repository explorer ■ Search for services in the Business Service repository **Note:** The connection used for this service is from the Visual Studio configuration settings, not the connection selected in the repository explorer. In Eclipse, these features are available through the context menus in the repository explorer. ■ Modify the configuration settings ■ Regenerate all classes in the current project that were created by the Business Service wizard
Business Service repository explorer	Tree view of the business repository. Using the repository explorer, you can: ■ View the current business services and domains ■ Search services across domains to find particular services ■ Invoke wizards to create a client proxy or Web service
Client Class wizard	Wizard used to generate client proxies and/or Web services (depending on the context) that are specific to a particular business service.
Web Service wizard	Wizard used to generate Web services that use IIS (Internet Information Server) and a .NET runtime component. This wizard is invoked from the Web Services node in the explorer window. After generating the Web service, open the context menu for the Web service and select **Test** to submit SOAP messages and test these services.
Web Application wizard	Wizard used to generate a web application based on previously generated Web services. This wizard is invoked from the Web Applications node in the explorer window. During generation, the wizard sets up folders for the web application and creates the support files. After generating the web application, you can open the Web Applications node and select the context menu for Pages to generate a web page or select the context menu for Menu to generate a menu for the web application.
Configuration utility	Utility used to define configuration settings for Web services and applications. **Note:** Web applications use the connection specified in the Web.config file in the Web Services directory in inetpubs.

5 Roles and Prerequisites

- Introduction 18
- Understanding Prerequisites 18
- Types of Security Available with EntireX 21
- Summary of Tasks 22

This section describes the roles of different users and helps new users understand which prerequisites are required for each business service activity. The following topics are covered:

Introduction

With all the functionality and components of Natural Business Services, it is important to understand which components are required for your scenario. Every development activity requires the Business Service repository and the Administration subsystem and every consumer requires an existing business service. All business services are created using the Natural Business Services Natural plug-in. As this plug-in does not require EntireX, you can use Natural Development Server (NDV) for development. EntireX is required for runtime environments.

At a high level, the minimum tasks required to produce a viable business service are:

1. Install Natural Business Services on the client and the server.
2. Ensure connectivity exists through NDV (for development activities) and EntireX (for development and/or runtime activities).
3. Ensure security is set for users and the development environments.
4. Create the business service.
5. Consume the business service through one of the following options:
   - Java (via Eclipse)
   - Visual Studio
6. Invoke the business service through one of the consumers.

Understanding Prerequisites

This section describes the prerequisites required for each business service activity. The following topics are covered:

- Create Business Services with the Natural Plug-in
- Create Business Services with the Eclipse Plug-in
- Create Web Services Without a Client SDK
- Create Java Applications or Web Services
- Create .NET Applications or Web Services
- Execute Web Applications Created Without a Client SDK
- Execute Java Applications
- Execute Java Web Services
- Execute .NET Applications

- Execute .NET Web Services

Create Business Services with the Natural Plug-in

The following components are required to create a business service with the Natural Business Services Natural plug-in:

Natural Business Services	Natural for Windows	Eclipse (Java)	Visual Studio .NET	Communication Tool	IIS	.NET Runtime	Java Runtime
	X			NDV		X	

Create Business Services with the Eclipse Plug-in

The following components are required to create a business service with the Natural Business Services Eclipse plug-in:

Natural Business Services	Natural for Windows	Eclipse (Java)	Visual Studio .NET	Communication Tool	IIS	.NET Runtime	Java Runtime
		X		EntireX			X

Create Web Services Without a Client SDK

The following components are required to create a Web service to be used with IIS (Internet Information Server) and without a client SDK, such as Visual Studio or Eclipse:

Natural Business Services	Natural for Windows	Eclipse (Java)	Visual Studio .NET	Communication Tool	IIS	.NET Runtime	Java Runtime
X				EntireX	X	X	

Tip: Use these Web services to generate web applications created with HTML, JavaScript, and XSL. For more information, see Build a Web Application.

Create Java Applications or Web Services

The following components are required to create a Java application or Web service:

Natural Business Services	Natural for Windows	Eclipse (Java)	Visual Studio .NET	Communication Tool	IIS	.NET Runtime	Java Runtime
X		X		EntireX			

Create .NET Applications or Web Services

The following components are required to create a .NET application or Web service:

Natural Business Services	Natural for Windows	Eclipse (Java)	Visual Studio .NET	Communication Tool	IIS	.NET Runtime	Java Runtime
X			X	NDV or EntireX		X	

Execute Web Applications Created Without a Client SDK

The following components are required to execute a web application for Web services created without using an SDK:

Natural Business Services	Natural for Windows	Eclipse (Java)	Visual Studio .NET	Communication Tool	IIS	.NET Runtime	Java Runtime
X				EntireX	X	X	

Execute Java Applications

The following components are required to execute a Java application:

Natural Business Services	Natural for Windows	Eclipse (Java)	Visual Studio .NET	Communication Tool	IIS	.NET Runtime	Java Runtime
X				EntireX			X

Execute Java Web Services

The following components are required to execute a Java Web service:

Natural Business Services	Natural for Windows	Eclipse (Java)	Visual Studio .NET	Communication Tool	IIS	.NET Runtime	Java Runtime
X				EntireX			X

Execute .NET Applications

The following components are required to execute a .NET application:

Natural Business Services	Natural for Windows	Eclipse (Java)	Visual Studio .NET	Communication Tool	IIS	.NET Runtime	Java Runtime
X				EntireX		X	

Execute .NET Web Services

The following components are required to execute a .NET Web service that works with ASP.NET:

Natural Business Services	Natural for Windows	Eclipse (Java)	Visual Studio .NET	Communication Tool	IIS	.NET Runtime	Java Runtime
X				EntireX		X	

Types of Security Available with EntireX

The following types of security are available when using the EntireX communication middleware:

Security Type	Description
NONE	Uses no security.
NSC	Uses Natural Security to validate users and check library access. (Natural Business Services definitions are used to check authorizations.)
APPL	Uses Natural Business Services to validate users and check authorization.
SAF	Uses SAF to validate users and check authorizations.
SAF-NSC	Uses SAF for user authentication, Natural Business Services for user, group, service, and method validation.
SAF-APPL	Uses SAF for user authentication, Natural Security for user and group validation, and Natural Business Services for service and method validation.

Note: Depending on your configuration, use either NSC or APPL with Natural Development Server (NDV).

As most sites have both a production and development environment, the decision about which type of security to use can be postponed for a while (although you must eventually choose a security type).

For more information on security, see Defining Users and Security Groups.

Summary of Tasks

Once you know which components are required for your site, you can determine which tasks must be done and how to verify that a task is completed. This section covers the following topics:

- Install Natural Business Services
- Set up, Deploy, and Monitor the Development Environment
- Create a Business Service
- Configure the Client
- Set Up Internet Information Server
- Create a Client Proxy
- Create a Web Service

Install Natural Business Services

This section describes the products and procedures required to install Natural Business Services on the server and on the client.

On the Server

The following products are required to install Natural Business Services on the server:

- Natural Development Server (NDV)
- Natural Business Services (NBS)

Note: Natural Security and EntireX are optional.

For information on installing Natural Business Services on the server, see:

- Natural Business Services Installation on Mainframes
- Natural Business Services Installation on Unix

To verify that Natural Business Services is installed on the server:

1 Ensure that Natural for Windows can connect to the server via NDV.

2 Ensure that users and groups are defined in NBS and, optionally, Natural Security.

3 Optionally, ensure that the attach, dispatch, and security servers can be initiated and pinged.

On the Client

The following products are required to install Natural Business Services on the client:

- Natural for Windows
- Business Service plug-in (repository explorer)

For information on installing Natural Business Services on the client, see Natural Business Services Installation on Windows.

To verify that the plug-in is installed and activated on the client:

1 Access the Business Service repository explorer.

2 Select **Domains** > **DEMO**.

3 Open the context menu for Customer.

4 Select **Test** on the submenu.

5 Select the Browse method.

6 Select **Run**.

If data comes back, there is connectivity to a business service.

Set up, Deploy, and Monitor the Development Environment

This section describes the products and procedures required to set up, deploy, and monitor the development environment. The following products are required for these tasks:

- Natural for Windows
- Business Service plug-in (repository explorer)

To verify that the products are operational:

1 Add a new domain.

2 Define a new steplib chain that includes all your development libraries required for the new domain.

3 Associate the new steplib chain with the new domain.

4 Define the domain, service, and method-level security.

5 Perform audit activities (as required).

6 Deploy services (as required).

- For information on the Eclipse plug-in, see Natural Business Services Eclipse Plug-in
- For information on the Natural plug-in, see Natural Business Services Natural Plug-in

Create a Business Service

This section describes the products and procedures required to create a business service using the Eclipse and Natural plug-ins.

Eclipse Plug-in

The following products are required to create a business service in the Eclipse plug-in:

- Eclipse SDK
- Natural Business Services Eclipse plug-in

To verify that the plug-in is operational:

1 Invoke the wizard and create a new business service.

2 Use the Service Repository Explorer to test the business service.

For information, see Natural Business Services Eclipse Plug-in.

Natural Plug-in

The following products are required to create a business service in the Natural plug-in:

- Natural for Windows
- Natural Business Services Natural plug-in

To verify that the plug-in is operational:

1 Invoke the wizard and create a new business service.

2 Use the Business Service repository explorer to test the business service.

For information, see Natural Business Services Natural Plug-in.

Configure the Client

This section describes the products and procedures required to configure the client in the Eclipse and Natural plug-ins, and in the Visual Studio add-in.

Eclipse Plug-in

The following product is required to configure the client in the Eclipse plug-in:

- Natural Business Services Eclipse plug-in

To verify that the plug-in is operational:

1. Open the context menu for **Default Broker** in the repository explorer.
2. Select **Configure** on the submenu.

   The **Connection Configuration Wizard** is displayed.
3. Select **Default Broker** from **Connection ID**.
4. Type the broker ID for the default broker in **Broker ID**.
5. Type the port number for the default broker in **Port**.
6. Select **Test** and verify that there is connectivity.

For information, see Configure Connections.

Natural Plug-in

The following product is required to configure the client in the Natural plug-in:

- Natural Business Services Natural plug-in

Note: The NBS attach server must be available to complete this test.

To verify that the plug-in is operational:

1. Select **Configuration** from the **Business Services** menu.
2. Select **Default Broker** from **Connection ID**.
3. Type the broker ID for the default broker in **Broker ID**.
4. Type the port number in **Port**.
5. Select **Test** and verify that there is connectivity.

For information, see Access Business Services.

Visual Studio Add-in

The following product is required to configure the client in the Visual Studio add-in:

- Natural Business Services Natural plug-in

Note: The NBS attach server must be available to complete this test.

To verify that the plug-in is operational:

1. Select **Configuration** from the **Business Services** menu.
2. Select **Default Broker** from **Connection ID**.
3. Type the broker ID for the default broker in **Broker ID**.
4. Type the port number in **Port**.
5. Select **Test** and verify that there is connectivity.

For information, see Configure Connections.

Set Up Internet Information Server

This section describes the products and procedures required to set up the Internet Information server in the Natural plug-in and in the Visual Studio add-in.

Natural Plug-in

The following products are required to set up the Internet Information server in the Natural plug-in:

- Internet Information Server (IIS)
- Natural Business Services Natural plug-in

To verify that IIS is operational:

1. Open the Business Service explorer window in the Natural plug-in.

   For information, see Using the Business Service Repository.
2. Expand the **Business Services** node.

   The **Domains** and **Configuration** nodes are displayed.
3. Expand the **Domains** node.

   The server retrieves the list of business service domains for the specified connection.
4. Select the **Demo** domain.

The server retrieves the list of business services in the **Demo** domain.

5 Open the context menu for OrderTraditional.

6 Select **SOAP Client Tests** on the submenu.

7 Test the Browse method.

If the method cannot be tested, the reason may be that IIS has not been set up correctly or that the Web.config file in the demo folder (under wwwroot) is using a different connection ID than the active one in the current configuration..

Visual Studio Add-in

The following products are required to set up the Internet Information server in the Visual Studio add-in:

- Internet Information Server (IIS)
- Natural Business Services Visual Studio add-in

To verify that IIS is operational:

1 Open the Business Service explorer window in the Visual Studio add-in.

For information, see Use the Business Service Explorer.

2 Expand the **Business Services** node.

The **Domains** and **Configuration** nodes are displayed.

3 Expand the **Domains** node.

The server retrieves the list of business service domains for the specified connection.

4 Select the **Demo** domain.

The server retrieves the list of business services in the **Demo** domain.

5 Open the context menu for OrderTraditional.

6 Select **SOAP Client Tests** on the submenu.

7 Test the Browse method.

If the method cannot be tested, the reason may be that IIS has not been set up correctly or that the Web.config file in the demo folder (under wwwroot) is using a different connection ID than the active one in the current configuration..

Create a Client Proxy

This section describes the products and procedures required to create a client proxy in the Eclipse plug-in and in the Visual Studio add-in.

Eclipse Plug-in

The following products are required to create a client proxy in the Eclipse plug-in:

- Eclipse SDK
- Natural Business Services Eclipse plug-in

To verify that the plug-in is operational:

1. Open the Business Services repository explorer.
2. Open the context menu for a business service.
3. Select **Create class**.

   The Create Class wizard is displayed.
4. Select **Finish** to generate the proxy class using the defaults.

Note: For more information, see Generate a Client Proxy Class.

Visual Studio Add-in

The following products are required to create a client proxy in the Visual Studio add-in:

- Visual Studio 2005 or higher
- Natural Business Services Visual Studio add-in

To verify that the product is operational:

1. Open the Business Services repository explorer.
2. Define a connection to a remote Natural Business Services environment.
3. Use the Create Class wizard to generate a class for a business service.

Note: For more information, see Creating a Client Proxy Class.

Create a Web Service

This section describes the products and procedures required to create a Web service in the Eclipse plug-in and in the Visual Studio add-in (to generate in Visual Basic .NET).

Eclipse Plug-in

The following products are required to create a Web service in the Eclipse plug-in:

- Eclipse SDK
- Natural Business Services Eclipse plug-in

To verify that the plug-in is operational:

1 Open the context menu for a business service in the **NBS Repositories** view.

2 Select **Create class**.

The **Configure generated class** panel for the Create Class wizard is displayed.

3 Select **Generate Web service class**.

4 Select **Finish** to generate the proxy class using the defaults.

Note: For more information, see Generate a Client Proxy Class.

Visual Studio Add-in

The following products are required to create a Web service in the Visual Studio add-in for Visual Basic .NET:

- Visual Basic .NET
- Natural Business Services Visual Studio add-in

To create a Web service:

1 Create an ASP.NET Web services project.

2 Write code to access and expose the methods for a previously generated Web service.

Note: For more information, see SAMPLES in the Natural Business Services Installation folder.

Test a Web Service

To test a Web service:

1 Open the context menu for a Web service in the explorer.

2 Select **Test**.

3 Select a SOAP method to test.

Note: For more information, see Test a Web Service.

Create a Web Application

To create a web application:

1 Open the context menu for the Web Applications node in the explorer.

Tip: Typically the word “localhost” is used in the node name.

2 Select **New Web Application**.

The Web Application wizard panel is displayed. This wizard will set up an environment for your new web application.

Note: For more information, see Create the Web Application Project.

Create a Web Page

To create a web page:

1 Select the node for a web application listed in the explorer.

2 Open the context menu for the Pages node.

3 Select **New Page**.

The Web Page wizard panel is displayed.

4 Create a web page.

Note: For more information, see Generate a Web Page.

Create a Menu

To create a menu:

1 Select the node for a web application listed in the repository explorer.

2 Open the context menu for Menu.

3 Select **Show Wizard**.

The Menu wizard panel is displayed.

4 Create the menu.

Note: For more information, see Generate a Menu.

6 Supplied Demo Applications

- Business Service Types 34
- Features of the Demo Application 37
- Additional Features When Using Predict 40

Natural Business Services provides sample business services in the DEMO domain. All the Natural modules for these services are located in the SYSBIZDE library on the server. For more information about these services, see DEMO Domain.

Note: If the sample services are not listed in the Business Service explorer, have your Natural Business Services administrator run the CSRLOAD program to load the repository.

In addition, client consumers (for example, Web services or client proxy classes) have been generated to demonstrate executing these services using Visual Studio and Java. Each business service was used to generate a Web service and then the Web services were used to generate the demo web application (web pages and menus).

This section describes the various business service types used in the demo application and highlights several features of the subprograms and proxies supplied in SYSBIZDE. The following topics are covered:

Business Service Types

Each business service has a type, which is determined by the wizard based on user input. On the client, the type is hidden from the user. On the server, it determines which Natural Construct model is used to generate the service. When the subprogram proxy is generated, the business service type is also stored in the business repository.

The business service types are:

Type	Service
blank	Arbsub (arbitrary subprogram)
1	Traditional
2	Object-Browse-Select without maintenance
3	Object-Browse-Select with maintenance
4	Object-Generic

A user may be able to identify which type a business service uses by its methods. The business service types are described in the following sections:

- Arbsub (Arbitrary Subprogram)
- Traditional
- Object-Browse-Select (Without Maintenance)
- Object-Browse-Select (With Maintenance)

- Object-Generic

Arbsub (Arbitrary Subprogram)

This business service (Type blank) assumes that the subprogram associated with it was not generated by Natural Construct. As Natural Business Services has no knowledge of the methods or required input and output parameters, all parameters are exposed to the client and the DEFAULT method is created. You can rename this method, as well as create other methods, by modifying the business repository information for this service.

An example of this type is the Calculator service, which has four methods: Add, Divide, Multiply and Subtract. As these methods would have behaved the same way as the DEFAULT method, overrides were added when the Web service was generated; the #FUNCTION parameter required a different value for each of the four new methods. For example, the Add #FUNCTION method required an override of “Add”.

A developer determines the value of #FUNCTION and decides what the subprogram will require to perform the function. For example, the Calculator service in the business repository uses the CALCY subprogram proxy in the SYSBIZDE library to call the CALC subprogram. If #FUNCTION is Add, CALC executes the appropriate code.

The main drawback to the Arbsub business service type is the amount of control given to the client. For example, if a Web service developer accidentally set the #FUNCTION override to Divide for the Add method, a user who only had permission to perform the Add method could inadvertently perform the Divide function.

Tip: If this is a concern, use an Object-Generic business service type.

Traditional

This business service (Type 1) has two subprograms associated with it: an object maintenance subprogram (generated by the Object-Maint-Subp model) and an object browse subprogram (generated by the Object-Browse-Subp model).

The typical methods associated with this type are:

- Delete
- Exists
- Former
- Get
- Initialize
- Next
- Store

- Update
- Browse

An example of this type is the Customer business service, version 010101. The Browse method calls the ACUSTY proxy, which calls the ACUSTN browse subprogram. The other methods call the MCUSTY proxy, which calls the MCUSTN maintenance subprogram.

Object-Browse-Select (Without Maintenance)

This business service (Type 2) is created using the Object-Browse-Select-Subp model and behaves in the same manner as a Browse-Select subprogram. Internally, it is different since you only want to pass a limited number of rows across the wire at one time in a client/server environment.

The object browse select subprogram requires an object browse subprogram. Based on the keys for the object browse subprogram, the Object-Browse-Select-Subp model creates the FindBy… methods. The method names can be modified during generation and methods can be removed if they are not required. If the HISTOGRAM option is selected for a browse key in the object browse subprogram, the object browse select subprogram defines count methods for the key. The key values and a count are provided, instead of all the data for the record (using FindBy… methods).

An example of the Object-Browse-Select business service type is Order, version 020101. The FindByOrderWarehouseId method returns all data in Warehouse ID order. A related method, OrderWarehouseIdCount, returns only the specified warehouse ID and a count of the number of orders for that warehouse. This service uses the BORDY subprogram proxy, which accesses the BORDN object subprogram.

Object-Browse-Select (With Maintenance)

This business service (Type 3) is also created using the Object-Browse-Select-Subp model and takes advantage of more features of the model. The CustomerWithContactData, version 020101, business service calls both the object browse and object maintenance subprograms from the same object browse select subprogram. To allow this functionality, an object maintenance subprogram was added to the Object-Browse-Select-Subp model specifications. This created four new methods: MultiMaint, Update, Delete and Store, which can be applied to the entire business service (i.e., a group of rows).

The action applied to each row is indicated by the row state, for example: U for Update, D for Delete, and A for Add. Each row sent to the server for maintenance requires a row state. For the server to apply these actions, the business method must be MultiMaint, Update, Delete or Store.

In general, the client never needs to use the Update, Delete or Store methods because the MultiMaint method handles all of them. The methods are supplied for security purposes. For example, if an administrator applied security at the method level to revoke Delete privileges for a user, the business service will not allow the user to use a D (for Delete) row state.

The CustomerWithContactData business service, version 020101, uses the BCUST2Y subprogram proxy, which calls the BCUST2N object subprogram. The Object-Browse-Select service uses the ACUST2N object browse subprogram and the MCUST2N object maintenance subprogram.

Object-Generic

This business service (Type 4) allows the business service to access up to 10 subprograms and create up to 20 different methods. Each method can be clearly defined on the server by hard-coding certain values. This prevents security from being breached and allows one client interface to be used for multiple subprograms.

An example of this type is the CalculatorAdvance business service. This service invokes the BNUMY proxy, which calls the BNUM subprogram. BNUM accesses two subprograms: CALC and GCDN. As the data for these subprograms is similar, the exposed variables have been reduced (this is why some of the parameters are commented out in the generated PARAMETER-DATA user exit code). Before either subprogram is called, data must be moved from the exposed data area to the local data areas used to pass data into these subprograms. This is done in the generated MOVE-TO user exit code. Similarly, data must be moved back to the exposed data area before control is returned to the client. This is done in the generated MOVE-BACK user exit code.

While defining methods for the CalculatorAdvance service, the developer specified which subprograms to execute for each method, the execution order of these subprograms, and whether code should be executed before and/or after the subprogram. For example, the SolutionWithLowerNumbers method has code that is executed after the GCDN subprogram is executed. This method also executes the CALC subprogram. (Refer to the AFTER-CODE subroutine in the BNUM subprogram.) This code reduces the first and second number based on the greatest common denominator and then calculates the division between the two numbers.

The Subtract method executes code before the CALC subprogram is executed. Refer to the BEFORE-CODE subroutine to see how #FUNCTION is assigned. This process is similar to providing overrides for the Calculator service, except the Web service cannot change #FUNCTION for the Subtract method because it will always be overridden in the BNUM subprogram.

Features of the Demo Application

This section describes the features of the demo application, SYSBIZDE. The following topics are covered:

- Subprogram Proxies
- Web and Business Services

- SOAP Client

Subprogram Proxies

As data typically comes from a different platform and uses a different character set, each business service requires a subprogram proxy to retrieve data off the wire. Although the proxy is required, all business code is contained in the subprogram that is called by the proxy.

During generation of the subprogram proxy, the business repository is populated with the methods used by the service, as well as the domain/service/version specifications.

Web and Business Services

The SYSBIZDE demo application includes the following Web and business services:

Web Service Name (or business service name when different)	Type	Proxy	Subprogram
Calculator	Arbsub	CALCY	CALC
CalculatorAdvance	Object-Generic	BNUMY	BNUM CALC GCDN
CustomerCreditAnalysis	Traditional	MCUST3Y	MCUST3N
CustomerPlain Customer, version 010101	Traditional With browse	MCUSTY ACUSTY	MCUSTN ACUSTN
CustomerWithContactData	Object-Browse-Select With browse With maintenance	BCUST2Y	BCUST2N ACUST2N MCUST2N
CustomerWithContactDataTraditional CustomerWithContactData, version 010101	Traditional With browse	MCUST2Y ACUST2Y	MCUST2N ACUST2N
ErrorMessageTesting	Basic Arbsub	FLIPSTY	FLIPSTR
FlipString	Arbsub with defined method	FLIPSTY	FLIPSTR
GreatestCommonDenominator (also used with CalculatorAdvance)	Arbsub	GCDY	GCDN
Order	Object-Browse-Select With browse	BORDY	BORDN AORDN

Web Service Name (or business service name when different)	Type	Proxy	Subprogram
OrderTraditional Order, version 010101	Traditional With browse	MORDY AORDY	MORDN AORDN
Product	Traditional With browse	MPRODY APRODY	MPRODN APRODN
StringManipulation	Object-Generic	BSTRINGY	BSTRINGN FLIBSTR CSUCASE
Warehouse	Traditional With browse	MWHY AWHY	MWHN AWHN

SOAP Client

The easiest way to test a Web service is to create a SOAP message that uses the service. To do this, open the context menu for the Web service and select the SOAP Client testing tool. The SOAP Client provides a list of the available methods. After you select a method, the SOAP Client fills in the URL, method name, and a simple SOAP Request. If you require specific input values, you can modify the SOAP message to reflect this.

In some cases, you can use one method to create output for another method to use as input. For example, the output for a FindBy… method is typically the input for a MultiMaint method. In this case, you can indicate the rows to be modified by typing a value in the appropriate row states (for example, U for Update).

Natural Business Services uses services exposed by the SOAP Client. For example:

- GetMetaData, which retrieves the metadata for a service
- GetServiceList, which determines which services are used in a Web service node
- Login, which creates a security token for a user ID and password combination

To view an example of input that has been modified, open the Soap Client example called CalculatorAdvanceSolutionWithLowerNumbers (located in the SoapClient directory under inetpubs/wwwroot/NBSDemos). CalculatorAdvanceSolutionWithLowerNumbers divides the first and second numbers by their greatest common divisor. This produces the lowest numbers that equal the same result when they are divided. The greatest common divisor is also shown. For example, 30 divided by 24=1.25. The greatest common divisor is 6, so 30 can be reduced to 5 and 24 can be reduced to 4 and the result is 1.25.

Additional Features When Using Predict

This section describes additional features of the SYSBIZDE demo application when using the Predict data dictionary.

BDT Options

When data is displayed, formatting should be automatically performed. To do this, you can define BDTs (business data types) for fields in the Predict file definition. If a BDT is not assigned in Predict, the Web Service wizard "guesses" which BDT to use. For example, if the field format is numeric with two decimal places, the wizard guesses that the BDT type is a currency amount. An example of using this BDT is the Result field for the Calculator service in the DEMO domain. A "$" is automatically placed at the beginning of this field.

With this rule you may think the FIRST-NUM and SECOND-NUM fields should also have "$" in front of them. By default they do, but the data type was changed from BDTCurrency to BDTNumeric when the Web service was generated.

Tip: To remove the "$" from the Result field, you can assign BDTNumeric to this field and regenerate the Web service.

ALLOW-LOWER-CASE Option

By default, an object browse subprogram converts all input data into upper case. The ALLOW-LOWER-CASE option allows users to enter data in lower case. It is useful for a field like Business Name, where the name is stored in mixed case.

To specify the lower case option:

1. Associate the ALLOW-LOWER-CASE keyword with the definition for the input field in Predict.
2. Regenerate the object browse subprogram.

For more information, see Natural Construct Object Models.

7 Using Natural Business Services

- Administer Business Services 42
- Configure the Plug-in 42
- Create a New Business Service 42
- Create a Client Consumer 47

This section describes the step-by-step procedure to create a business service and provides links to additional information. The following topics are covered:

Administer Business Services

Administrators can use the Business Service repository explorer in the Natural Business Services plug-in to deploy business services and to apply security to services.

- For information on administering business services in the Eclipse plug-in, see Administering Business Services.
- For information on administering business services in the Natural plug-in, see Administering Business Services.

Configure the Plug-in

Before you can create a business service, you must confirm that the environments are set up correctly.

- For information on configuring the Eclipse plug-in, see Configure the Eclipse Plug-in.
- For information on configuring the Natural plug-in, see Configure the Natural Plug-in.

Create a New Business Service

A business service consists of a collection of methods related to a common business entity. The best way to create a business service is to use the Business Service wizard in the Natural Business Services Eclipse or Natural plug-in. Each plug-in provides a repository explorer you can use to select and test a service.

During generation, the Business Service wizard creates a subprogram proxy to translate data (including Unicode formats) and then adds an entry in the repository explorer. It also generates additional subprograms as required.

This section describes how to use the Business Service wizard to create a business service based on existing Natural subprograms. You can also use the wizard to create business services by generating new subprograms or by creating an empty service skeleton.

Notes:

1. For information on these and other options in the Eclipse plug-in, see Developing Business Services/Create a Business Service. For information on these and other options in the Natural plug-in, see Developing Business Services/Create a Business Service.
2. The examples used in this section are taken from the Natural Business Services Eclipse plug-in. Some options may not be available with other plug-ins.

To create a new business service:

1 Select the connection for the business service in the **NBS Repositories** view.

If required, enter the user ID and password for the connection.

Note: You must use a SPoD connection to create a business service.

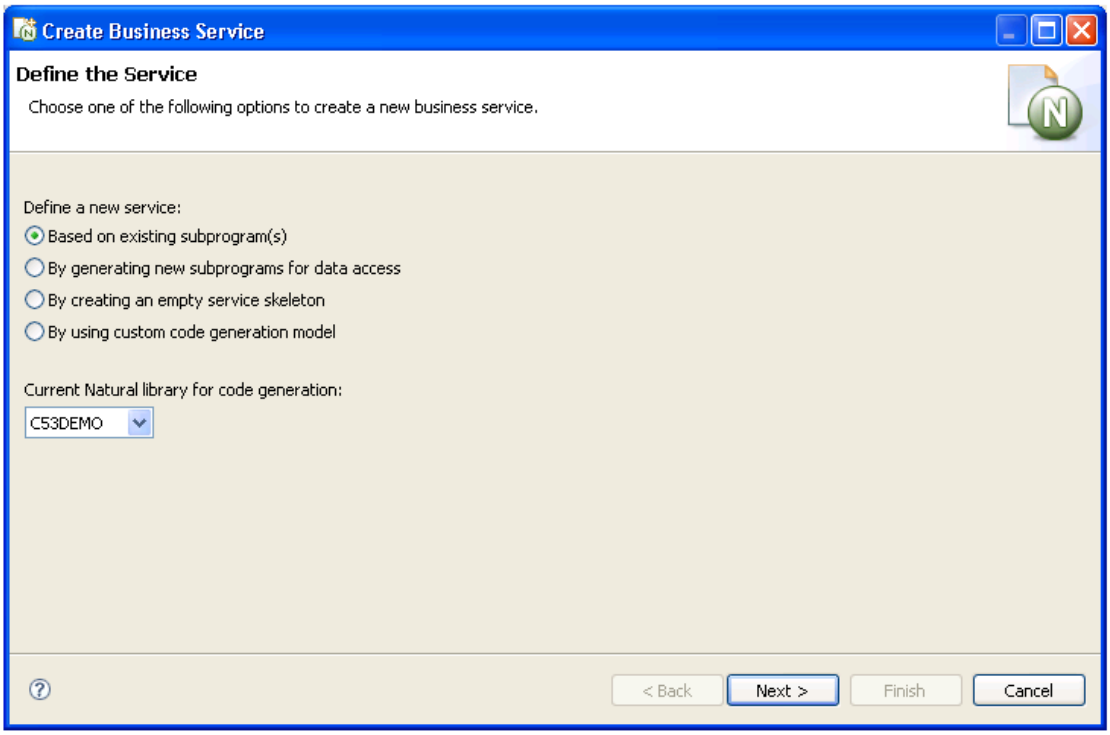
2 Expand the **Domains** node.

The list of domains is displayed.

3 Open the context menu for the domain in which you want to generate the service.

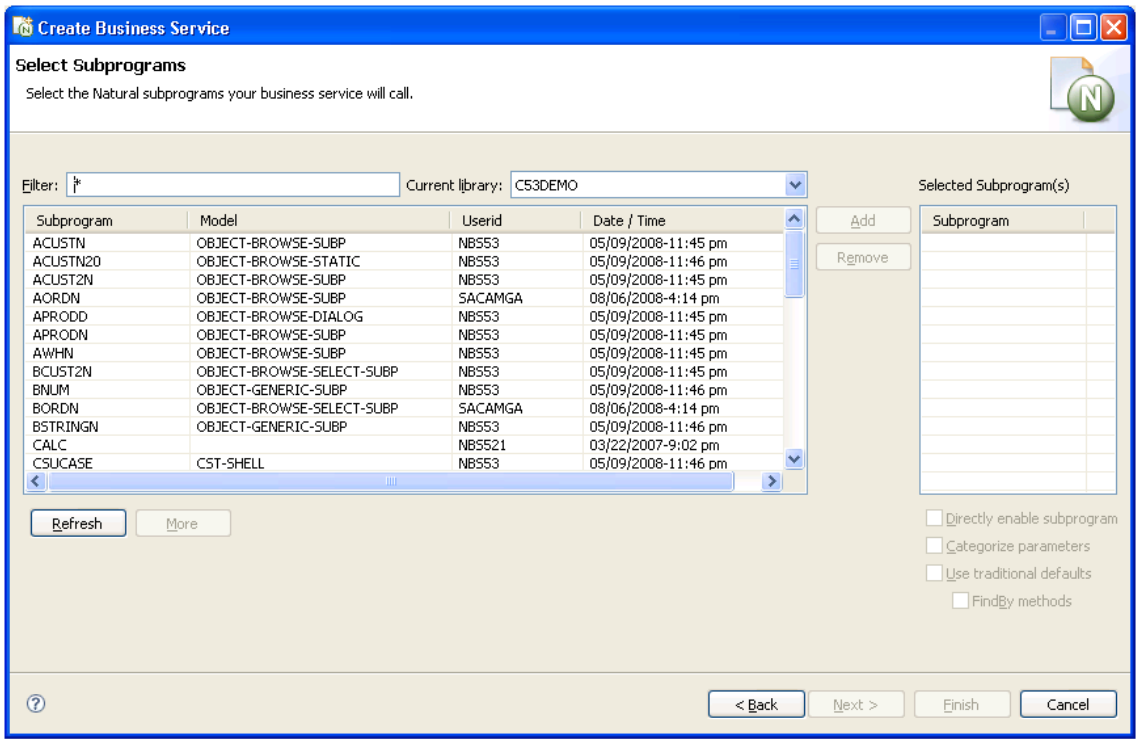
4 Select **Create business service**.

The **Define the Service** panel is displayed. For example:



5 Select **Based on existing subprogram(s)**.

The **Select Subprograms** panel is displayed. For example:



6 Select **Refresh**.

A list of available subprograms for the current library is displayed.

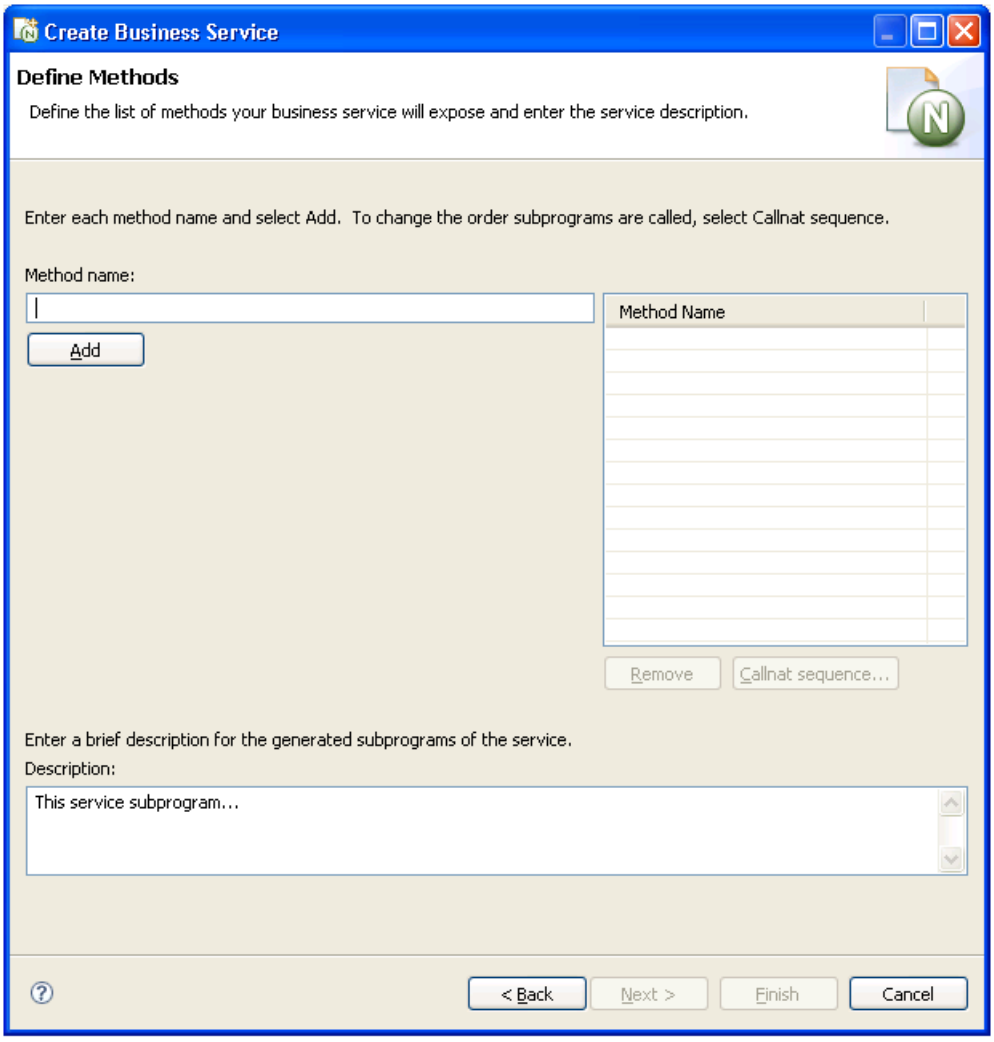
7 Select the subprogram you want to use.

8 Select **Add**.

The selected subprogram is listed in **Selected subprogram(s)**.

9 Select **Next**.

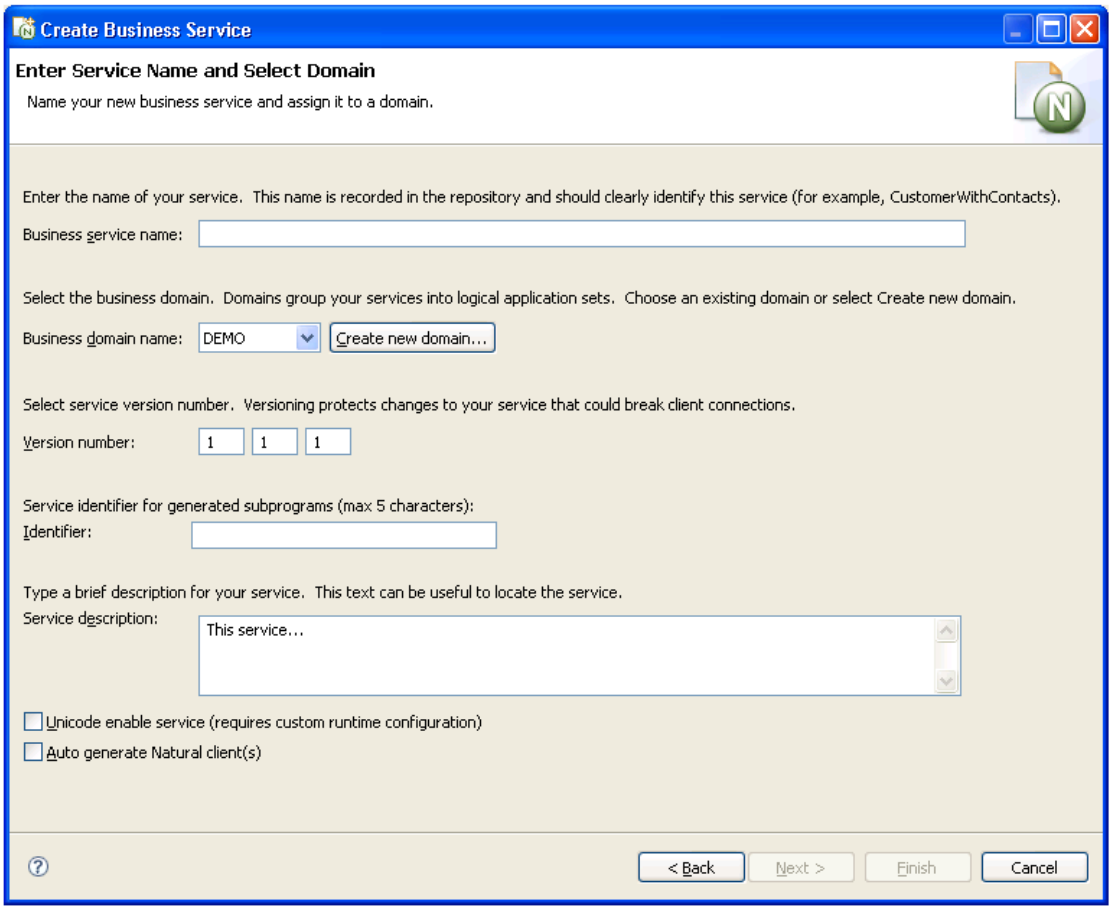
The **Define Methods** panel is displayed. For example:



Use this panel to define the methods the business service will expose to the consumer. Each time you type a method name and select **Add**, the method is listed in **Method Name**. Also use this panel to provide a brief description of the subprograms to be generated.

10 Select **Next**.

The **Enter Service Name and Select Domain** panel is displayed. For example:



11 Enter the name of your service in **Business service name**.

This name should clearly identify the service (for example, CustomerWithContacts).

12 Select the business domain from **Business domain name**.

13 Verify the version number in **Version number**.

Version numbers help protect changes to the service that can disrupt client communications. Use version 1.1.1 (default) for a new service.

14 Type the service identifier for generated subprograms in **Identifier**.

The service identifier can be up to five characters in length and will be used with a wizard-generated prefix and suffix to identify the generated subprograms used for this service.

15 Type a brief description for your service in **Service description**.

16 Select **Finish** to generate the business service.

The **Generation Status** window is displayed, showing the progress of the generation. When generation is completed, the window shows the names of the files generated for the business service.

The Results column indicates that the generated files have been generated for the first time (New).

You have successfully generated and saved a new business service. For information on testing your business service, see Test a Business Service.

Note: For information on testing a business service with the Natural plug-in, see Test a Business Service.

Create a Client Consumer

A client consumer is anything that uses a business service. You can create client consumers in several different ways. For example, you can create client proxies classes (and, optionally, Web services) in Java or .NET to consume business services. For information:

Plug-in/Add-in	Described In
Eclipse plug-in	Developing Client Proxy Classes and Web Services
Visual Studio add-in	Creating a Client Proxy Class and Creating a Web Service.

8 Installation Tips

- Troubleshoot EntireX and SPoD Connections 50
- Resolve IIS (Internet Information Server) and ASP.NET Issues 51

This section contains information on resolving problems you may encounter after installing Natural Business Services. The following topics are covered:

Troubleshoot EntireX and SPoD Connections

There are only two types of connections: EntireX or SPoD. If you installed Natural Business Services using the defaults, you should only need to supply the Broker ID or Host name to configure these connections. Select **Test** to confirm that the connection is working. If the connection is not working, verify the following:

1. Try to ping the Broker ID from the command line in Windows (for example, `Ping MyBrokerId`). If you still cannot establish the connection, modify the hosts file in *C:\WINDOWS\system32\drivers\etc.* or contact the network administrator.
2. Ask your Natural Business Services administrator to confirm that the attach server can be pinged from the server in the Business Service Administration subsystem. If you still cannot establish a connection, the attach server may not be initiated, or when it was initiated, the JCL may have failed. After issuing the INITIATE command, a message indicates which batch job was started. Ask the administrator to review the output of this batch job.

   Note: If the attach server is being initiated successfully, the batch job will run continuously.

   Note: The batch job is created using a JCL template supplied in the SYSBIZ library.

   You can determine which template was used by displaying the attach server settings in System Administration. Display the Maintain Servers panel and edit the attach server. The JCL template should be listed in the Server Start Parameters section (for example, JCL=mytemplate). Next, logon to SYSBIZ and edit the template (for example, mytemplate).

   Natural Business Services supplies default templates (BATCHJCL, for example). These templates must be customized for each site. We recommend that you copy BATCHJCL to another text module, modify it appropriately, and then reference the modified text member.

   Tip: The easiest way to determine what the template should look like is to find an existing Natural batch job and customize the template to have similar references.

3. Ensure that the EntireX attribute file has the same class/server/service references as the server definitions in the Business Service Administration system.
4. Natural Business Services initiates the dispatch and security servers as they are required. If there are problems, it is also helpful to have the administrator initiate these servers from the Business Service Administration server. If this is successful, both servers will be initiated and can be pinged from within the Business Service Administration system. If not, possible reasons are:

- The dispatch server is defined as a subtask, but the Natural nucleus being used has not been set up for subtasking (for information on how to do this, see Activate the Business Service Administration Subsystem). If the nucleus is set up for subtasking, start the dispatch server as a batch job instead of a subtask. It is easier to find the dispatch errors in a batch job.
- The user ID to bring the dispatch server up has not been defined to security.
- The dispatch server listed on the second Maintain Servers panel points to the security server that will be used. Ensure that the security server is the one the dispatch server is pointing to.

5. If the attach, dispatch, and security servers can be pinged on the server and the Broker ID can be pinged on the client, the EntireX settings may not be set up correctly. For example, is the user ID set up for EntireX security. Is EntireX security turned on?

Resolve IIS (Internet Information Server) and ASP.NET Issues

If your ASP.NET applications will not run (including the Web Service Engine), verify the following:

- ASP.NET is installed

  Select **Home directory > Configuration** from **IIS manager** and verify that the .ASPX extension is in list.

- ASP.NET permissions are allowed on Windows 2003

  Select **IIS admin > Web Service Extensions**.

- IIS was installed after .NET

  Run *C:\WINDOWS\Microsoft.NET\Framework\v1.1.4322\aspnet_regiis.exe* to install ASP.NET.

- If you are trying to debug ASP.NET, ensure that integrated Windows security is turned on

  Select **directory security** from **IIS manager**.

9 Related Documentation

The following table lists other sources of information about Natural Business Services:

Document	Description
Natural Business Services Administration	Describes how to set up and manage environments for Natural Business Services applications. This guide is intended for: ■ Administrators who use the Business Service Administration subsystem to manage network communications and business services ■ Developers who create and maintain business services
Understanding Natural Business Services	Provides information on how Natural Business Services creates business services.
Natural Business Services Natural Plug-in	Describes the Natural plug-in supplied by Natural Business Services. This guide is intended for developers who are familiar with Natural for Windows and the Natural Development Server (NDV) and want to use the Natural Business Services plug-in to create and maintain business services.
Natural Business Services Eclipse Plug-in	Describes the Natural Business Services plug-in for Eclipse. This guide is intended for developers who are familiar with Eclipse and want to use the Natural Business Services Eclipse plug-in to create Natural business services, Web services, web applications, and/or generate client proxy classes to access business services.
Natural Business Services Visual Studio Add-in	Describes the Natural Business Services add-in for Visual Studio. This guide is intended for developers who are familiar with Visual Studio and want to use the Natural Business Services Visual Studio add-in to create Web services, web applications, and/or generate client proxy classes to access Natural business services.
Natural Business Services Tools and Utilities	Describes the development tools and utilities provided by Natural Business Services: the Broker Driver program, NBS Tester, and Server Monitor.

Document	Description
Natural Business Services Subprogram-Proxy-Client Model	Describes the Natural Business Services Subprogram-Proxy-Client model. This model generates a subprogram proxy to access a Natural subprogram.
Natural Construct Administration and Modeling	Describes how to access and use the Administration subsystem of Natural Construct.
Natural Construct Object Models	Describes the Natural Construct Object models. These models generate the Natural subprograms used by Natural Business Services.
Natural Construct Transform-Browse Model	Describes the Natural Construct Transform-Browse model. This model transforms an existing browse module (generated by the Browse, Browse-Select, Browse-Select-Subp, or Browse-Subp model) into object-browse modules (generated by the Object-Browse series of models).

A Appendix A: Glossary

The following terms are used throughout the Natural Business Services documentation set. Each term is listed with its meaning:

Term	Description
add-in	Wizard or utility that expands the functionality of an IDE (integrated development environment).
application library	Natural library containing the server application components of a client/server application.
assembly	Logical grouping of functionality in Visual Studio. Can be one or more executable or DLL files.
attach server	Middleware component that determines which dispatcher to use and whether other dispatchers are required. If other dispatchers are required, the attach server will start them.
business data types (BDTs)	Type validation on the client that applies business semantics to a field. Typically, BDTs are used to format field data specified by the user. For example, if an application has an input field to enter a phone number, you can associate a BDT with the field to reformat the number with hyphens. A user can enter "7053332112". When the user moves to the next field or performs another action, the number is automatically reformatted as 705-333-2112. Natural Business Services supplies standard BDTs, which you can customize, or you can create your own. Default field BDTs are added to the keyword components of a field in Predict.
business service	Group of methods related to a common business entity, such as Customer, Order, or Department.
Business Service Administration subsystem	Server subsystem used to maintain and query tables that define business services and security data. This subsystem is commonly known as the Natural Business Services repository. Use the Natural Business Services Natural plug-in to modify the repository data. **Note:** You must have security privileges to access this subsystem.

Term	Description
	Note: You can also enter "menu" from the SYSBIZ library to access the Administration subsystem.
business service definition	Definition in the Natural Business Services repository that identifies the methods exposed by a subprogram. Use the Natural SPoD add-in to modify these settings.
business service explorer (BSE)	Displays business services on a server organized by Connection/Domain/Service Name.
BDT class	Collection of all BDT procedures.
BDT modifier	Additional logic users supply to modify the formatting or validation rules for a BDT. For example, BTD_NUMERIC ensures that only numeric values are entered in a field. You can also add a modifier to round numeric values. To increase flexibility, each BDT defines its own modifiers.
cache	See security cache.
cardinality	Number of dimensions of information. Information with the same number of dimensions has the same cardinality.
class	Object representation of a business service in Visual Basic, C#, or Java.
complex redefine	Redefinition of a data area containing multiple data types, multiple redefinitions of a data field, or multiple levels of redefined fields.
compression	Reduce the byte size required to transmit data to and from the client and server. Data is compressed when it is sent and then decompressed when it reaches its destination. This reduces the size of data transmissions and improves network performance.
consumer	Anything that uses a Natural business service. For example, the generated client proxy classes in Visual Studio or Java consume business services. Web services can also consume business services.
context menu	Task-based submenu that is displayed by right-clicking an item in the repository tree or pressing the context menu key on the keyboard, for example.
control record	Record that controls whether Natural Security is used.
database record	Logical view of database information. A database record can be comprised of one or more logically related database files or tables. Natural Business Services represents database information in parameter data areas (PDAs).
DBID	Acronym for database ID, which is the number identifying the server database containing application components.
deployment	Movement of an application from a source environment to a target environment (for example, moving an application from development to QA).
dialog	GUI form (window) running on the client.
Dispatch Client	Component that provides the Natural Business Services data exchange, which facilitates calls from a client to Natural subprograms running on a server.
dispatch server	Middleware component that encapsulates broker calls on the server, provides directory services, enforces security, and invokes back end Natural services.
domain	Entity that defines a collection of related business services (for example, Test, Admin, and Sales).

Term	Description
encapsulation	Technique in object-oriented programming in which the internal implementation details of an object are hidden from users of the object. Methods control how the object data is manipulated. Encapsulation allows internal implementations to change without affecting the way an object is used externally.
encryption	Encoding data so it is unusable for individuals without access to the decryption algorithms. Natural Business Services allows you to encrypt sensitive data, such as payroll information, during network transmission. Data is decrypted when it reaches its destination.
EntireX Broker service settings	Collection of Broker-related parameters, including Broker ID, server class, server name, and service.
EntireX Broker stub	Broker DLL on a Windows platform.
event	Action recognized by an object, such as pressing a key or clicking a mouse. You write code to respond to events.
event handler	Custom code that reacts to events raised within the Web Service Engine (WSE).
FNR	Acronym for the file number that identifies a specific server database file containing application components.
framework template	Set of templates supplied for applications. These customizable templates include header, footer, navigation bar, messages area, and constants.
generate	Process of producing code from specifications.
generated module	Generated component for either the client or server portion of an application. Generated server modules include Natural subprograms, subprogram proxies, and parameter data areas. Generated client modules include Visual Studio classes, XSL, HTML, etc.
group	Collection of users defined in the Business Service repository.
GUI control override	Use Predict keywords to force a GUI control derivation. See also keyword.
HTML template	HTML that may contain replacement tags, which are dynamically exchanged for content or nested HTML templates at runtime.
IIS	Internet Information Server development tool in Windows.
instantiation	Process of creating an instance of a class. The result is an object.
internationalization	Adapting an application to make it easy to localize. See also localization.
keyword	Predict metadata type that acts as a label, identifier, or specification for generation (such as BDTPhone or ALLOW-LOWER-CASE).
level 1 data block	Level 1 field or structure and its subfields in a Natural parameter data area (PDA).
library image file (LIF)	File that contains Natural PDA definitions used by the Dispatch Client.
localization	Process of translating and adapting a software product for use in a different language or country.
lookups	Return descriptive information when a user requests a browse window or enters a value in a foreign key field in a maintenance window. For example, assume Warehouse Number is a foreign key field in the Order window and Warehouse Name is a descriptive field attached to the foreign key value. When a user enters

Term	Description
	a valid warehouse number, the lookup returns the name of the warehouse to display.
metadata	Information about data. Metadata describes how physical data is formatted and interrelated. It includes descriptions of data elements, data files, and relationships between data entities. Typically, metadata is maintained in a repository known as a data dictionary, such as Predict.
method	Procedure that operates on an object and is implemented internally by the object. For example, the Update method updates a Customer Order object after changes to the order information.
model	Template used to generate modules. Each model contains one or more specification panels. Using these panels, you can specify parameters for a desired module and then generate the corresponding code. Natural Construct provides numerous models, including the Object-Maint-Subp and Object-Browse-Subp models.
module	Single application component, such as a hand-coded Natural program, subprogram, or data area or a Natural Construct-generated program, subprogram, or data area.
multi-level security	Security you can define at a high level or at a detailed level affecting many objects. For example, you can apply multi-level security at a domain, business service, or method level.
Natural Business Services (NBS)	Services stored on the server (similar to a subprogram proxy).
Natural Construct nucleus	Sophisticated driver program that invokes the model subprograms at the appropriate time in the generation process and performs functions common to all models, such as opening windows and performing PF-key functions. The nucleus communicates with the model subprograms through standard parameter data areas (PDAs). These PDAs contain fields assigned by Natural Construct, as well as fields required by a model.
Natural Debugging utility	Utility available in a Natural environment to help you locate and analyze logic errors. This utility is not available in the client environment, but you can access it on the server with client input by invoking the Invoke Proxy function in the Business Service Administration subsystem. The subprogram proxy sets up an online environment that simulates the client/server environment and allows you to use all the features of the Natural Debugging utility.
node	■ Individual computer or, occasionally, another type of machine in a network ■ An item on a menu tree or navigation bar ■ A setting in an XML file
nucleus	See Natural Construct nucleus.
object	Any application component, such as a form or record.
ping	Request sent to a service to determine whether the service is running.

Term	Description
platform	Piece of equipment that, together with its operating system, serves as a base on which you can build other systems. For example, a Unix box can serve as a platform for an accounting system.
regular expression	String-searching criteria to scan for a specified text string and, optionally, substitute another string. For example, a regular expression can search for "Natural Business Services" and replace each occurrence with "Business Services" alone.
remote call	Communication with an object residing in a different location.
resource	Text or binary value that can be localized. See also localization.
security cache	File used to store recently-accessed security data.
security server	Component of the Business Service Administration subsystem that controls access to application libraries, objects, and methods.
server	Computer that provides services to another computer (called a client) and responds to requests for services. On multitasking machines, a process that provides services to another process is called a server.
server application	Application that runs on a server machine.
Server Manager	Client tool supplied with Natural Business Services that allows you to specify which servers the client uses to communicate with the server.
server settings	Collection of parameters used to configure a server.
service proxy	See subprogram proxy.
shutdown	Command sent to terminate a server.
Simple object access protocol (SOAP)	XML-based standard communication protocol to interact with Web services.
SPoD	Natural for Windows Single Point of Development product.
steplib chain	Hierarchy of Natural libraries that determines the location from which modules are executed.
subprogram proxy	Provides the link between a specific subprogram and the Natural Business Services dispatch server.
template parser	Class used to parse HTML or other templates.
token (Natural Business Services security)	Unique, system-generated, identification number used in place of a user ID and password for multiple calls. Users can logon with their user ID and password and then request a token to use for later calls.
trace options	Options that specify how to trace messages sent between the client and server.
verification rule	Predict-defined business rules that are implemented in the object maintenance subprogram on the server. They also provide default values for derived fields represented by GUI controls, such as check boxes, option buttons, or drop-down combo boxes. You can use verification rules to force users to make a selection based on one or more choices. For example, if an application has an input field for the state name, you can attach a verification rule to the field in Predict so that only valid state names are accepted.

Term	Description
web application	Application created using the Natural Business Services wizards and add-ins. It allows users to access business services and data from a web browser.
web framework	Group of Visual Basic modules and classes that collaborate to dynamically generate web pages.
Web service	Service to expose data or functionality to the intra/internet.
Web Service Engine (WSE)	Core DLL supplied with Natural Business Services that handles Web service requests (*.sws requests) from IIS (Internet Information Server).
Web service root	Main/root directory (folder) for Web services. By default, the name of the Web services root directory is NBS/WebServices. For example: *http://localhost/NBS/WebServices* *C:\inetpub\wwwroot\NBS\WebServices* **Note:** For examples used in this documentation, assume the Web services root is `NBS/WebServices`.
WSDL (Web Service Definition Language)	XML document used to describe a Web service.
XML (Extensible Markup Language)	Industry-accepted standard language used to transmit data.
XML extract	Extract information from Predict and other sources, which is stored on the client as metadata in XML format. This includes information about business services, as well as the formatting used by wizards to build application components. See also metadata.