Natural Development Server

Natural Development Server for Windows

Version 9.3.1

February 2025

ADABAS & NATURAL

This document applies to Natural Development Server Version 9.3.1 and all subsequent releases.

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

Document ID: NDV-NDVDOC-931-20250228

Table of Contents

Preface

This documentation applies to Natural Development Server (product code NDV) for Windows.

Natural Development Server for Windows is released together with Natural for Windows. It has the same version number as Natural for Windows.

For information on changes, enhancements or new features in this version of Natural Development Server, see the *Release Notes* in the corresponding Natural for Windows documentation.

Introducing Natural Development Server	Describes purpose and functionality of Natural Development Server which is used in conjunction with NaturalONE or Natural for Windows (as a client) in a Natural Single Point of Development (SPoD) environment.
Development Server File	Describes purpose and use of the Natural Development Server file, a central dictionary file that is structurally identical to the Natural system file `FDIC`.
Installing Natural Development Server	How to install Natural Development Server on Windows.
Configuring Natural Development Server	How to configure Natural Development Server.
Operating Natural Development Server	How to operate Natural Development Server.
SPoD-Specific Limitations and Considerations	Describes the limitations which are due to the different capabilities of the user interfaces available on the client and server sides. In addition, this document includes hints which are important for the efficient use of the remote development facilities.

Related Documentation

- Natural Single Point of Development documentation
- NaturalONE documentation
- Natural for Windows documentation

1 About this Documentation

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
`Monospace font`	Identifies service names and locations in the format *`folder.subfolder.service`*, APIs, Java classes, methods, properties.
`Italic`	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
`Monospace font`	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information and Support

Product Documentation

You can find the product documentation on our documentation website at **https://documentation.softwareag.com**.

Product Training

You can find helpful product training material on our Learning Portal at **https://learn.softwareag.com**.

Tech Community

You can collaborate with Software GmbH experts on our Tech Community website at **https://techcommunity.softwareag.com**. From here you can, for example:

- Browse through our vast knowledge base.
- Ask questions and find answers in our discussion forums.
- Get the latest Software GmbH news and announcements.
- Explore our communities.
- Go to our public GitHub and Docker repositories at **https://github.com/softwareag** and **https://containers.softwareag.com/products** and discover additional Software GmbH resources.

Product Support

Support for Software GmbH products is provided to licensed customers via our Empower Portal at **https://empower.softwareag.com**. Many services on this portal require that you have an account. If you do not yet have one, you can request it at **https://empower.softwareag.com/register**. Once you have an account, you can, for example:

- Download products, updates and fixes.
- Search the Knowledge Center for technical information and tips.
- Subscribe to early warnings and critical alerts.
- Open and update support incidents.
- Add product feature requests.

Data Protection

Software GmbH products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

2 Introducing Natural Development Server

This chapter describes the purpose and the functions of Natural Development Server (product code NDV) which are used in conjunction with NaturalONE or Natural for Windows (as client) in a Natural Single Point of Development (SPoD) environment.

Purpose of Natural Development Server

Natural Development Server enables you to use NaturalONE or the Natural Studio development environment provided by Natural for Windows to develop and test Natural applications in a remote Natural Windows environment running under the operating system Windows.

For more information on NaturalONE and remote development, see

- NaturalONE documentation (describes the SPoD client side; how to manage offloaded Natural objects in the Eclipse workspace, and also how to modify them directly on a development server).
- Natural Single Point of Development documentation (general information).

For more information on Natural Studio and remote development, see

- Natural for Windows documentation or Help system (describes the SPoD client side; how to manage Natural objects directly on a development server).
- Natural Single Point of Development documentation (general information).

Remote Development Functions

Establishing a Connection between Client and Server

A connection to an active development server can be established by mapping it in the client (that is, in NaturalONE or Natural Studio). A dialog will be shown for setting up the connection in which you have to specify the following information:

Server

Host name	The host name defines the remote node name where the server is running (or the IP address of the server).
Server port	The server port defines the TCP/IP port number for the development server.
Environment name	The environment name can be used to give the addressed server a logical (descriptive) name. If this box is left blank, a default name will be created automatically.

Startup

Session parameters	If dynamic parameters are required for your development server, specify them in this text box. Otherwise, leave this text box blank.
User ID	Your user ID is automatically provided.
Password	If Natural Security is installed on the development server, specify the required password in this text box. Otherwise, leave this text box blank.

These settings are transferred to the selected Natural Development Server and evaluated to create an exclusive Natural session that is responsible for executing all development requests for that environment. Once you have sucessfully mapped a development server, the Natural objects of the connected remote development environment are shown in NaturalONE or Natural Studio.

Using the Remote Development Functionality

You can use the entire functionality of NaturalONE or Natural Studio to create, edit, stow or execute Natural objects on the remote Natural environment. You can map to multiple environments from one NaturalONE or Natural Studio. Each mapped environment owns a Natural session on the Natural Development Server, even if you map multiple environments on the same server.

When you are working with NaturalONE, it is recommended that you work in the so-called "local mode". In local mode, the sources are no longer stored or modified directly on the development server. The central place for keeping the sources is now the Eclipse workspace which is connected to a version control system.

3 Development Server File

This chapter describes purpose and use of the Natural Development Server file, a central dictionary file that is structurally identical to the Natural system file FDIC.

The use of the Natural Development Server file FDIC is optional. If you do not use the FDIC file, object locking will be based on operating system means. If you are using the FDIC file, object locking will also take place controlled by the operating system, but the object locking information will in addition be held in the FDIC file.

Purpose of the Development Server File

As Natural stores its data in system files, Natural Development Server stores its data in the system file that is assigned to the Natural parameter FDIC, a logical system file which is called the “development server file”.

The development server file is used as a central dictionary file for storing Natural applications and the links to objects making up an application. It also holds object locking information. This information is not bound to certain groups of application developers, but has an impact on the entire application development of an enterprise. Therefore, this file should be available only once, to ensure that the application definitions and locking states are kept consistent.

Note: The documentation of a Predict Program object, with the object's implementation pointer completely filled (i.e. Natural member name, library name, user system file number, and user system file database id), is deleted when the Natural member is deleted in an NDV environment. By default, the switch "**Delete documentation with deletion of Natural member**" is initialized with "Y", i.e. deleting the existing documentation. If you wish to keep the documentation set this new switch to "N".

Relations between FDIC and the Development Server File

The development server file layout corresponds to the file layout of the Natural system file FDIC used by Predict. This means that the central dictionary file can also be used to hold Predict data, but Predict is not a prerequisite for using the development server file. This enables you to use your existing application documentation in the application definitions of the remote development environment.

Unique Development Server File

It is of vital importance that the various remote development environments that can be mapped use a common and unique development server file.

Non-compliance with this requirement may give rise to inconsistencies in object locking and in the applications existing in the application workspace.

Under Natural Security

In a Natural Development Server that is protected by Natural Security, the use of another `FDIC` file in the application workspace is prevented if the application security profiles are activated. See also *Application Protection* in the *Natural Security* documentation.

4 Installing Natural Development Server under Windows

Natural Development Server is delivered with the applicable version of Natural for Windows.

For information on how to install Natural Development Server, see the *Installation* documentation for Natural for Windows.

5 Configuring Natural Development Server under Windows

NDV User Exit Routine

Natural Single Point of Development provides the following user exit:

NDV-UX01	This exit is invoked before a Natural source object or a DDM is edited. It can be used to reject editing of certain sources. The source code of this exit is delivered in the library SYSLIB and named NDV-SX01 *).

*) The sources of these user exit routines are named NDV-SX*nn*, where *nn* denotes the number of the user exit routine.

To make a user exit routine available

1 Copy the source code from SYSLIB into a user library.

2 Catalog it under the name NDV-UX*nn*.

3 Copy it back into the Natural system library SYSLIB.

The name of each user exit source is different from the name of the corresponding cataloged object. This guarantees that the object is not affected if the user exit source is overwritten by an installation update.

For further details, see the source code of the user exit routines NDV-SX*nn* in the Natural system library SYSLIB.

6 Operating Natural Development Server

This chapter describes how to operate Natural Development Server in a Windows-based environment.

Introduction

Natural Development Server for Windows represents a listener process which waits for incoming Natural clients to connect. For every connection request, the listener starts a Natural process which will then communicate with the corresponding client. Natural Development Server can only be run in a valid Natural environment. For the currently applicable version of Natural for Windows, refer to Empower at ***https://empower.softwareag.com/***.

Starting the Development Server

The development server can be started as follows:

```
natdvsrv [start-options] [Natural-parameters]
```

where

`-p[ort][=]<port>`	Mandatory start option. Starts the server on the specified port.	
`-c[lient][=]<type>`	Optional start option. Defines the types of clients which are permitted to connect to the server. `<type>` can be one of the following:	
	`NAT`	Only Natural for Windows (Natural Studio) clients are permitted to connect to this server.
	`ONE`	Only NaturalONE clients are permitted to connect to this server.
	`ANY`	Both Natural for Windows (Natural Studio) and NaturalONE clients are permitted to connect to this server.
	If this start option is not specified, the server will use `ANY` as the default. If a client which is not permitted tries to connect to the server, Natural error `NAT7742` will be issued, indicating that the current IDE is not permitted. **Note:** If Natural Security is installed, the types of clients permitted to connect to the server can be controlled by setting the Natural Security option **Natural client access**.	

`-s[tart][=]<image>`	Optional start option. The Natural image to be used as the server session. Any valid Natural image can be used. If this start option is not specified, the server will use "natural" as the default session image. For the currently applicable version of Natural for Windows, refer to Empower at *https://empower.softwareag.com/*.
`[Natural-parameters]`	Optional. The Natural parameters to be used in conjunction with the session image. The syntax is identical to starting a Natural session with Natural parameters.

Examples

```
natdvsrv -port=1234
```

starts the listener on port 1234, (implicitly) uses the image "natural" as the server session, and (implicitly) permits all types of clients to connect to the server.

```
natdvsrv -p=1234 -c=ONE
```

starts the listener on port 1234, (implicitly) uses the image "natural" as the server session, and permits only NaturalONE to connect to the server.

```
natdvsrv -p1234 -cNAT parm=myparm stack=(LOGON mylib)
```

starts the listener on port 1234, (implicitly) uses the image "natural" as the server session, and permits only Natural for Windows (Natural Studio) to connect to the server. When a client session connects, the listener starts Natural with the parameter module "myparm" and applies a logon to "mylib".

```
natdvsrv -p=1234 -s=naturalsql
```

starts the listener on port 1234, uses the image "naturalsql" as the server session, and (implicitly) permits all types of clients to connect to the server.

Determining the Screen Size for Web I/O

By default, a screen size of 80x24 is used for web I/O with the Natural Web I/O Interface. This is determined by the following variables:

Variable Name	Description
NWO_NCOLUMNS	The number of columns to be used. Default: 80.
NWO_NROWS	The number of rows to be used. Default: 24.

If a different screen size is to be used, you have to set these variables *before* starting the development server.

Example

Define the above variables as system variables in the **Environment Variables** dialog box. Or execute the SET command (for example, using the **Command Prompt** window), using the above variables.

```
SET NWO_NROWS=50
```

When the current screen size is 80x24, the above SET command changes the screen size to 80x50.

Terminating the Development Server

The development server can be terminated as follows:

```
natdvsrv -t[erminate][=]<port>
```

where <port> is the port number of the development server to be terminated.

Terminating the server will also terminate all Natural sessions started by the server.

Note: Alternatively, a development server session can be terminated via the kill <process-id> command.

Example

```
natdvsrv -t1234
```

terminates the listener and all Natural sessions launched by the listener.

Getting Help

```
natdvsrv [-h[elp]]
```

displays the help information on the available options.

NDV Configuration File

When a new development server is started, the file *NDVSERVER.PRU* located in your installation directory, in the subdirectory *Natural\Prof*, will be accessed to add information about the server.

When a server is terminated, the following steps are performed:

1. A search of the specified port takes place in order to find the server's process ID. This process ID is then used to terminate the server.
2. The entry of the terminated server in the configuration file will then be removed.

SSL (TLS) Support

Although we are always talking about the commonly known phrase SSL (Secure Sockets Layer) the technology actually used is TSL (Transport Layer Security). SSL and TLS basically do the same thing. They are encrypted protocols for data transfer.

The development server is using SSL/TLS to enable a secured data transfer with the involved NDV clients, such as the NaturalONE clients. Server authentication cannot be switched off. Due to the fact that asymmetrical encryption is used, a certificate (including a public key) and a private key is always required on the server.

To establish an SSL connection, you have to proceed as described in the following topics:

- Creating an SSL Certificate and a Private Key
- Start the Natural Development Server

- Configuring the Client

Creating an SSL Certificate and a Private Key

For sensitive public connections it is always a good idea to get an SSL certificate from a Certificate Authority (CA) such as Verisign. However, when using servers in a local environment it is also possible to use self-signed certificates. The following steps explain how a self-signed certificate can be created via the openssl tool:

To create a self-signed SSL certificate and a private key on the server, using openssl:

1. Copy the example configuration file *openssl.cnf* (located in *<installdir>/common/security/openssl* to a different directory, for example to the *<yourdirectory>/bin* directory of your Natural installation.
2. Adapt the copied configuration file *openssl.cnf* to your needs.
3. Set the environment variable so that it points to the file *openssl.cnf*.

   ```
   set OPENSSL_CONF=<yourdirectory>/openssl.cnf
   ```

4. Generate a certificate signing request:

   ```
   openssl req -new > ndv.server.cert.csr
   ```

   With the `openssl req -new` command we create a private key and a certificate signing request (CSR) for the root CA. You will be asked for a passphrase to protect the private key. The `openssl req` command uses the configuration parameters given in the `[req]` section of the configuration file *openssl.cnf*.
5. Generate a private RSA key issuing the following command:

   ```
   openssl rsa -in privkey.pem -out ndv.server.cert.key
   ```

6. Generate a self-signed certificate issuing the following command:

   ```
   openssl x509 -in ndv.server.cert.csr -out ndv.server.cert.crt -req -signkey ↩
   ndv.server.cert.key -days 365
   ```

   It is important that the name of the generated certificate is *ndv.server.cert.crt* and that the name of the generated private key is *ndv.server.cert.key*.

An alternative to the steps above would be to use a certificate generator within your company, creating a certificate chain with a corresponding root CA and also a private key.

Once the certificate is available, it must be copied in conjunction with the private key into the *bin* directory of your Natural installation. These two files must always have the name *ndv.server.cert.key* (representing the private key) and *ndv.server.cert.crt* (representing the self-signed certificate).

Start the Natural Development Server

Start the Natural development server with the option `-ssl`. For example:

```
natdvsrv -port=4711 -ssl
```

Configuring the Client

For further information on how to use the certificate store in NaturalONE, please refer to *SSL* under *Setting the Preferences* or *Accessing a Remote Development Environment* or *Using SSL* - all to be found in the section *Using NaturalONE* of the *NaturalONE* documentation.

Trace

A server trace can be established via the built-in trace macros.

Important: In order to generate trace files for the listener and all started sessions, a plus sign (+) must be used as the trace file name. Otherwise, a new development server session will overwrite an existing trace file.

The name of the listener trace file is constructed as shown below:

```
natdvsrv_port
```

where

port	is the assigned port number.

The name of a session trace is constructed as shown below:

```
natdvsess_ipinfo_ pid_port
```

where

ipinfo	is the client connection's IP address/name.
pid	is the process ID of the development server session.
port	is the port of the listener process.

Example

```
setenv  NCTR_STATE    1
setenv  NCTR_CONFIG   path/nctr.cfg
setenv  NCTR_OUTPUT   /tmp/+
```

After a listener process was started on port 4712 and two clients have connected, the following trace files are generated into the directory /tmp:

```
natdvsrv_4712
natdvsess_pc1_1055_4712
natdvsess_pc2_1045_4712
```

Note: Be careful when using the plus sign (+) notation. This could cause a lot of trace files to be generated.

Error Messages

When a development server is started or terminated, the following error messages can occur:

Error Number	Error Message	Action
66	The port number is missing.	Provide a valid port number.
67	The specified port number is already in use.	Use a port which is not yet occupied on the machine.
68	Invalid syntax [%*syntax*%] encountered.	The options used to start or terminate the server are used wrongly according to [%*syntax*%].
70	Invalid port number encountered.	Use a correct port number inside the valid range of 0 - 9999 and do not use other characters or digits.
71	Listen on specified port failed.	Try again with a different port number.
78	The specified server session [%*session-name*%] is not accessible.	Check that the specified session can be found via the `PATH` environment variable or that it is located inside `$NATBIN`.
79	The port number [%*number*%] exceeds upper limit (99999).	Use a port inside the valid range.
80	Invalid combination of options [%*options*%] encountered.	Check that the specified options can be used together.
81	NDV server could not be terminated (Reason: %*reason*%).	Check the reason: ■ Check that the server with the specified port is running.

Error Number	Error Message	Action
		■ Check that you have the rights to terminate the server.
82	Error accessing file 'NDVSERVER.PRU'.	Check that the file is available in your installation directory, in the subdirectory *Natural\Prof*, and that it has a valid content. If the file is corrupted, delete the file.

7 SPoD-Specific Limitations and Considerations

When you are working with Natural Single Point of Development, you will encounter a few limitations which are due to the different capabilities of the graphical user interface available on the local site and the character-based user interface that exists on the remote site. In addition, this document includes hints which are important for the efficient use of the remote development facilities.

Editor Features With SPoD

You can use Natural's Single Point of Development with different versions of Natural on a variety of platforms. Depending on the server environment you are using together with Natural for Windows (client), the editors offer different features. For further information, refer to the section *Editor Features With SPoD* in the Natural for Windows *Editors* documentation.

Limitations

System Commands

System Commands Entered Directly on the Development Server

All system commands which are not entered in the user interface of Natural Studio are executed directly by the Development Server without control of Natural Studio. As a result, the character-based representation of the corresponding command appears in the Natural Web I/O Interface client window. This is the case when the `STACK TOP COMMAND` mechanism is used or when a system command is directly entered inside the Natural Web I/O Interface client window.

During the mapping phase any `STACK` commands entered in the text box **Session Parameters** are processed within Natural Studio and the corresponding Natural Studio windows are used.

The commands `HELLO` and `MAINMENU` do not cause a screen output on the development server side, since this would not make any sense in the SPoD environment. Instead of the menu-driven user interface, the dialogs provided in Natural Studio are used.

Terminal Commands

Using terminal commands in a SPoD environment is only possible within the Natural Web I/O Interface client window. Entering terminal commands in the command line of Natural Studio is not possible.

Moving/Copying Error Messages

Moving and copying of error messages is different in remote and local environments:

- When error messages are moved or copied within the remote environment or are moved or copied from the local to the remote environment or vice versa: the error messages involved are merged, that is,
  - error messages which already exist in the target environment are replaced,
  - messages which do not exist in the source library are kept in the target library,
  - messages which do not exist in the target library are added.
- When error messages are moved or copied within the local environment, the messages involved are handled on file level, that is,
  - all error messages (that is, files) of a language are deleted and
  - the file from the source library is created anew in the target library.

Field Sensitive Maps

For these maps, the consistency check for a map field is made as soon as the user input has been entered. Field sensitive maps can be moved or copied from the local environment to a remote environment. However, a field sensitive map cannot be tested or executed on a remote mainframe environment.

Data Input/Output

Data input and output takes place *only* via the Natural Web I/O Interface, irrespective of whether the dynamic parameter `WEBIO` is set to `ON` or not.

Dialogs

Dialogs can be edited, but cannot be executed directly on the server.

Dependencies between XRef Evaluation and Predict

If you are using dynamic language assigned when calling other objects such as `INPUT USING MAP 'MAP1&'`, the connection between caller and called object cannot be retrieved by using XRef Evaluation.

Natural on the mainframe supports case-sensitive calls to other objects such as `PERFORM SUBROUTINE`. With the current version of SPoD, this may lead to strange results when, in XRef Evaluation, trees are expanded and it is not possible to request case-sensitive calls with the filter dialog.

Performance Considerations

Progress Information

The working situation displayed in the library workspace of Natural Studio is based on the representation of the entire user system files.

The tree view window opens when the user connects to the Natural Development Server. For this, the entire system file has to be analyzed and the corresponding information has to be transferred from the Natural Development Server to the Natural Studio client. In the case of very large system files, the build-up of the tree view window can be very time consuming. Status information displayed in the status bar keeps the user informed about the progress of the screen build-up operation. This is to avoid the impression that the connection to the Natural Development Server might be interrupted.

Tip: Switch on the status bar using the **View > Status Bar** function of the menu bar. Make sure that the transfer rate of your network is 100 Mbit/s at minimum.

Filter Definition

Another possibility to reduce the amount of data read while mapping the environment is to supply filter definitions on system file or library level.

Tip: In the context menu of a system file and library node it is possible to apply filter definitions. Using these definitions on the client side, you can limit the number of libraries/objects displayed in the tree view.

Refresh Options

In the default configuration of Natural Studio, all operations which result in a modification of the system file, for example, moving or copying objects, but also a `SAVE` or `STOW` command, will cause the tree view window contents to be refreshed, which can be a very time consuming process in the case of very large system files.

Tip: By default, the **Refresh** function is set to **Full automatic refresh**. Change the automatic refresh function by choosing **Optimized automatic refresh** or **No automatic refresh** in the context menu.

Since the tree view of the application workspace displays only the objects that are linked to the application, the build-up of its tree view screen is consequently considerably faster, which is another advantage of using the application workspace.

Object Lists on Mainframes

In mainframe environments, libraries may contain a huge number of objects. Expanding such a library in a tree node in the NaturalONE Server view or in the Natural Studio views can take a long time.

Tip: Install the hyperdescriptor as described in *NaturalONE in a Nutshell > Performance Aspects* in the NaturalONE documentation. The hyperdescriptor is also used by Natural Studio. It can significantly improve the database access required for reading object names the Natural system file.