

Natural

Natural Web I/O Interface

Version 9.1.1

April 2019

This document applies to Natural Version 9.1.1 and all subsequent releases.

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

Copyright © 1979-2019 Software AG, Darmstadt, Germany and/or Software AG USA, Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

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

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at <http://softwareag.com/licenses>.

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://softwareag.com/licenses/> 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, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at <http://softwareag.com/licenses> and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

Document ID: NATMF-NNATWEBIO-911-20211014

Table of Contents

Preface	vii
1 About this Documentation	1
Document Conventions	2
Online Information and Support	2
Data Protection	3
I Introduction	5
2 Introduction	7
What is the Natural Web I/O Interface?	8
Components of the Natural Web I/O Interface	8
Executing a Natural Application in a Web Browser	9
Client-Server Compatibility	10
Terminology	10
Differences in a SPoD Development Environment	11
Restrictions When Using the Natural Web I/O Interface with Natural Applications	11
Differences between the Natural Web I/O Interface Client and Terminal Emulation	13
II Introducing the Natural Web I/O Interface Server CICS Adapter	15
3 Introducing the Natural Web I/O Interface Server CICS Adapter	17
Purpose of the Natural Web I/O Interface Server CICS Adapter	18
CICS Support	18
Product Interaction	19
III Introducing the Natural Web I/O Interface Server IMS Adapter	21
4 Introducing the Natural Web I/O Interface Server IMS Adapter	23
Purpose of the Natural Web I/O Interface Server IMS Adapter	24
IMS TM Support	24
Product Interaction	25
IV Installing and Configuring the Natural Web I/O Interface Server	27
5 Natural Web I/O Interface Server Concept and Structure	29
Web I/O Interface Server Concept	30
Natural Web I/O Interface Server under SMARTS on BS2000	30
Front-End Stub NATRNWO	31
Front-End	32
Server Monitor	33
6 Prerequisites	35
General Prerequisites for Web I/O Interface Server Installation	36
Prerequisites for the Web I/O Interface Server for z/OS	36
Prerequisites for the Web I/O Interface Server under SMARTS on z/VSE	36
Prerequisites for the Web I/O Interface Server under SMARTS on BS2000	37
7 Installing the Natural Web I/O Interface Server under z/OS	39
Prerequisites	40

Content of the Web I/O Interface Server Distribution Medium	40
Installation Procedure	40
8 Installing the Natural Web I/O Interface Server under SMARTS on z/VSE	45
Prerequisites	46
Content of the Web I/O Interface Server Distribution Medium	46
Installation Procedure	47
Allocating and Configuring a SMARTS Portable File System	49
9 Installing the Natural Web I/O Interface Server under SMARTS on BS2000	53
Prerequisites	54
Installation Medium	54
Installation Procedure	55
Installation Verification	56
SMARTS Portable File System on BS2000	57
10 Configuring the Natural Web I/O Interface Server	61
Configuration Requirements for z/OS	62
Configuration Requirements for z/VSE	69
SMARTS Configuration File for BS2000	71
Web I/O Interface Server Configuration File for z/OS and z/VSE	80
Web I/O Interface Server Configuration Parameters	80
Web I/O Interface Server Configuration File Example	92
Web I/O Interface Server Data Sets for z/OS, z/VSE	93
Web I/O Interface Server User Exits	93
11 Installing the Natural Web I/O Interface Server CICS Adapter under z/OS	95
Prerequisites	96
Installation Procedure	96
12 Installing the Natural Web I/O Interface Server CICS Adapter under SMARTS on z/VSE	101
Prerequisites	102
Installation Procedure	102
13 Configuring the Natural Web I/O Interface Server CICS Adapter	105
Configuration File	106
Configuration Parameters	106
14 Installing the Natural Web I/O Interface Server IMS Adapter	111
Prerequisites	112
Example Jobs	112
Installation Procedure	112
15 Configuring the Natural Web I/O Interface Server IMS Adapter	117
Configuration File	118
Configuration Parameters	118
V Installing the Natural Web I/O Interface Client	121
16 Prerequisites	123
Servlet Container	124
Natural for Mainframes	124
Natural for UNIX	124
Natural for OpenVMS	125

Natural for Windows	125
Browser Prerequisites	125
17 Installing the Natural Web I/O Interface Client on Apache Tomcat	127
Installation Steps	128
Installation Verification	130
18 Migrating the Natural Web I/O Interface Client from IIS to Apache Tomcat	131
Before You Install the Natural Web I/O Interface Client	132
Installing the Natural Web I/O Interface Client on Apache Tomcat	133
Configuring the Natural Web I/O Interface Client on Apache Tomcat	133
VI Configuring the Client	137
19 About the Logon Page	139
Starting a Natural Application from the Logon Page	140
Examples of Logon Pages	140
Dynamically Changing the CICS Transaction Name when Starting a Session	141
Specifying a Password in the Logon Page	142
Changing the Password in the Logon Page	142
Browser Restrictions	143
20 Natural Client Session Configuration	145
General Information	146
Name and Location of the Configuration File	146
21 Natural Client Configuration Tool	147
Invoking the Configuration Tool	148
Session Configuration	149
Logging Configuration	157
Logon Page	157
Logout	158
22 Natural Web I/O Style Sheets	159
Name and Location of the Style Sheets	160
Editing the Style Sheets	160
Modifying the Position of the Main Output and of the PF Keys	160
Modifying the Font Size	162
Modifying the Font Type	163
Defining Underlined and Blinking Text	163
Defining Italic Text	164
Defining Bold Text	164
Defining Different Styles for Output Fields	165
Modifying the Natural Windows	165
Modifying the Message Line	166
Modifying the Background Color	166
Modifying the Color Attributes	167
Modifying the Style of the PF Key Buttons	168
XSLT Files	168
23 Starting a Natural Application with a URL	171

24	Configuring Container-Managed Security	173
	General Information	174
	Name and Location of the Configuration File	174
	Activating Security	174
	Defining Security Constraints	175
	Defining Roles	175
	Selecting the Authentication Method	176
	Configuring the UserDatabaseRealm	176
25	Configuring SSL	177
	General Information	178
	Creating Your Own Trust File	178
	Defining SSL Usage in the Configuration File	179
26	Logging	181
	General Information	182
	Name and Location of the Configuration File	182
	Invoking the Logging Configuration Page	182
	Overview of Options for the Output File	184
VII	Operating and Monitoring the Natural Web I/O Interface Server	185
27	Operating the Natural Web I/O Interface Server	187
	Starting the Natural Web I/O Interface Server	188
	Terminating the Natural Web I/O Interface Server under z/OS, z/VSE	190
	Terminating the Natural Web I/O Interface Server under BS2000	190
	Changing the SYSOUT File Assignment of the FSIO Task under BS2000	191
	Monitoring the Natural Web I/O Interface Server	191
	Runtime Trace Facility	193
	Trace Filter	194
28	Monitor Client NATMOPI	197
	Introduction	198
	Prerequisites for NATMOPI Execution on BS2000	198
	Command Interface Syntax	198
	Command Options Available	199
	Monitor Commands	199
	Directory Commands	199
	Command Examples	200
29	HTML Monitor Client	203
	Introduction	204
	Prerequisites for HTML Monitor Client	204
	Server List	204
	Server Monitor	206
	Index	209

Preface

This documentation is organized under the following headings:

Introduction	What is the Natural Web I/O Interface?
Introducing the Natural Web I/O Interface Server CICS Adapter	Special information which applies if you want to use the Natural Web I/O Interface server in a CICS environment under z/OS or z/VSE.
Introducing the Natural Web I/O Interface Server IMS Adapter	Special information which applies if you want to use the Natural Web I/O Interface server in an IMS TM environment under z/OS.
Installing and Configuring the Natural Web I/O Interface Server	How to install and configure the Natural Web I/O Interface server in a mainframe environment.
Installing the Natural Web I/O Interface Client	How to install the Natural Web I/O Interface client on an application server or in a servlet container so that it can be used with the Natural Web I/O Interface server.
Configuring the Client	How to define the information that is to appear in the logon page.
Operating and Monitoring the Natural Web I/O Interface Server	How to operate the Natural Web I/O Interface server in a mainframe environment, and how to monitor it using the Monitor Client NATMOP I or the HTML Monitor Client.

 **Note:** This documentation only explains how to install the Natural Web I/O Interface server in a mainframe environment. For information on how to install it in a UNIX, OpenVMS or Windows environment, see the Natural documentation for the appropriate platform.

1 About this Documentation

- Document Conventions 2
- Online Information and Support 2
- Data Protection 3

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
Monospace font	Identifies service names and locations in the format <i>folder.subfolder.service</i> , APIs, Java classes, methods, properties.
<i>Italic</i>	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

Software AG Documentation Website

You can find documentation on the Software AG Documentation website at <https://documentation.softwareag.com>.

Software AG Empower Product Support Website

If you do not yet have an account for Empower, send an email to empower@softwareag.com with your name, company, and company email address and request an account.

Once you have an account, you can open Support Incidents online via the eService section of Empower at <https://empower.softwareag.com/>.

You can find product information on the Software AG Empower Product Support website at <https://empower.softwareag.com>.

To submit feature/enhancement requests, get information about product availability, and download products, go to [Products](#).

To get information about fixes and to read early warnings, technical papers, and knowledge base articles, go to the [Knowledge Center](#).

If you have any questions, you can find a local or toll-free number for your country in our Global Support Contact Directory at https://empower.softwareag.com/public_directory.aspx and give us a call.

Software AG Tech Community

You can find documentation and other technical information on the Software AG Tech Community website at <https://techcommunity.softwareag.com>. You can:

- Access product documentation, if you have Tech Community credentials. If you do not, you will need to register and specify "Documentation" as an area of interest.
- Access articles, code samples, demos, and tutorials.
- Use the online discussion forums, moderated by Software AG professionals, to ask questions, discuss best practices, and learn how other customers are using Software AG technology.
- Link to external websites that discuss open standards and web technology.

Data Protection

Software AG 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.

I Introduction

2 Introduction

- What is the Natural Web I/O Interface? 8
- Components of the Natural Web I/O Interface 8
- Executing a Natural Application in a Web Browser 9
- Client-Server Compatibility 10
- Terminology 10
- Differences in a SPoD Development Environment 11
- Restrictions When Using the Natural Web I/O Interface with Natural Applications 11
- Differences between the Natural Web I/O Interface Client and Terminal Emulation 13

This chapter describes the purpose and the functions of the Natural Web I/O Interface.



Note: This introduction mainly describes how the Natural Web I/O Interface works in a runtime (production) environment. The section *Differences in a SPoD Development Environment* briefly explains the special version that is used in a SPoD development environment.

What is the Natural Web I/O Interface?

The Natural Web I/O Interface is used to execute Natural applications in a web browser. It fully supports the following:

- The display and input of Unicode characters. See *Unicode Input/Output Handling in Natural Applications* in the *Unicode and Code Page Support* documentation.
- Rich internet applications developed with Natural for Ajax.

Components of the Natural Web I/O Interface

The Natural Web I/O Interface consists of a server and a client.

Server

The Natural Web I/O Interface server enables you to use a browser as the I/O device for Natural applications. The server does the user authentication, creates the Natural session and handles the I/O between Natural and the client. The Natural Web I/O Interface server is installed on the same machine as the Natural application.

Client

The client handles the communication between the user's web browser and the Natural Web I/O Interface server. It converts the output from the Natural application to web pages, and returns the user input to Natural.

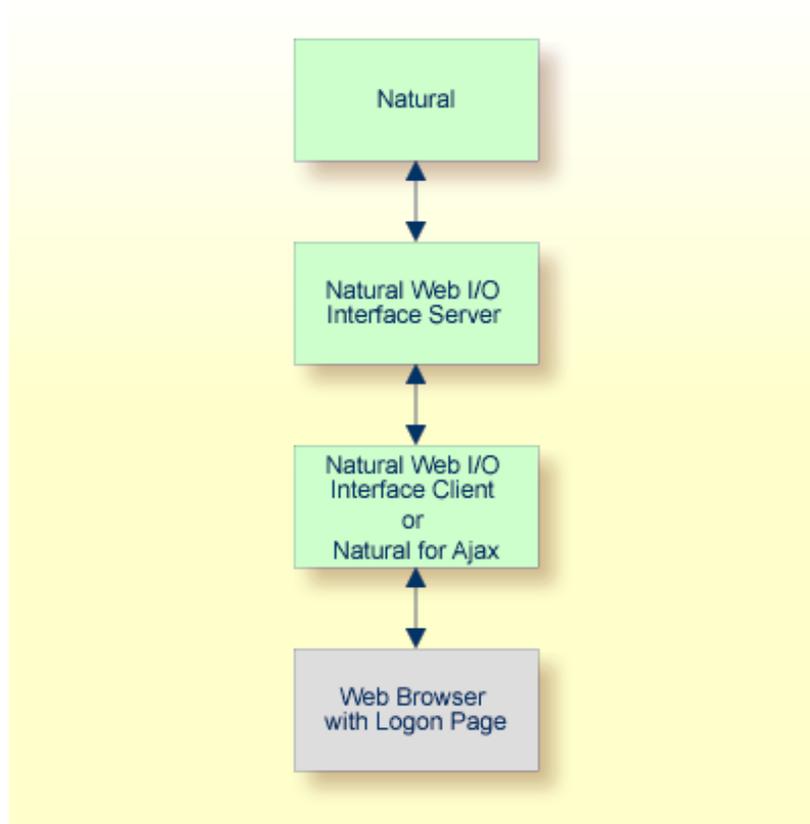
Two types of client are supported:

- Natural Web I/O Interface client for displaying character-based applications in the web browser. Maps with GUI controls are not supported in this case.
- Natural for Ajax for displaying rich internet applications in the web browser. For further information on this type of client, see the Natural for Ajax documentation.

The client is installed on a web/application server. This can be done on any machine in the network.

Executing a Natural Application in a Web Browser

The Natural Web I/O Interface receives data from a Natural application and delivers web pages to the user's web browser. This is illustrated in the following graphic:



The communication steps for executing a Natural application in the web browser are:

1. The user enters the address (URL) of a logon page in the web browser. The client then displays the logon page in the web browser.



Note: For information on how to invoke and configure the logon page, see [Configuring the Client](#).

2. The user enters all required information for starting a Natural application into the logon page. This information is sent to the client.
3. The client asks the Natural Web I/O Interface server to start the requested Natural application for this user.
4. The Natural Web I/O Interface server checks the supplied user ID and password, creates a Natural session for the user and starts the Natural application.

5. The Natural application returns the first application screen which is then transferred via the Natural Web I/O Interface server to the client and finally as a web page to the web browser.

Different web browsers are supported. Note that cookies and JavaScript must be enabled in the web browser. For a list of the currently supported web browsers, see the browser prerequisites for the type of client that you are using.

Client-Server Compatibility

The following rules apply:

- The Natural Web I/O Interface server can work with any client that has the same or a higher protocol version.

If the server detects that the client is using a version that is lower than the server version, the server replies that the client is too old and the connection is closed.

- The client can work with any server that has the same or a lower protocol version.

If the client detects that the server is using a version that is lower than the client version, the client switches to the server version. However, new client functionality is not supported in this case.

- The Natural Web I/O Interface server must have the same protocol version as the Natural process that is started by the server. If Natural detects that the server is using a different protocol version, an error message is sent to the user and the connection is closed.

Terminology

On the different Natural platforms for which the Natural Web I/O Interface is supported, different techniques are used for implementing the server part of the Natural Web I/O Interface. On Natural for UNIX and Natural for OpenVMS, it is implemented as a daemon. On Natural for Windows, it is implemented as a service. On the mainframe, it is implemented as a server. In this documentation, the general term “server” is therefore used for all different kinds of implementation.

Differences in a SPoD Development Environment

The previous sections of this introduction have described how the Natural Web I/O Interface works in a runtime (production) environment. This section briefly explains the differences in a SPoD development environment.

A special version of the Natural Web I/O Interface is used when working in a remote development environment with Natural for Windows (SPoD). In this case, the Natural Web I/O Interface is an integrated component which does not require a separate installation. The server is part of the Natural Development Server (NDV), and the client is part of Natural Studio. Other than in the runtime environment, the screen is not displayed in a browser but in a normal window. Rich GUI pages created by Natural for Ajax are not supported in the development environment.

It is important that I/O via the Natural Web I/O Interface has been enabled on the Natural host. Otherwise, the Natural Web I/O Interface cannot be invoked. See also *Unicode Input/Output Handling in Natural Applications* in the *Unicode and Code Page Support* documentation.

Restrictions When Using the Natural Web I/O Interface with Natural Applications

There are several restrictions when using the Natural Web I/O Interface with Natural applications on UNIX, OpenVMS, mainframe or Windows hosts.



Note: The term “application” refers to application software. It does not refer to system software or software for development.

The following restrictions apply:

- **GUI controls**

GUI controls are not supported: dialogs, buttons, radio buttons, list boxes, list views, check boxes etc. The Natural Web I/O Interface only supports Natural applications developed without GUI controls.

- **File transfer**

File transfer (for example, with the `DOWNLOAD` statement) is not supported by the Natural Web I/O Interface.

- **Runtime errors**

This restriction applies to older Natural versions on UNIX and Windows. As of version 6.3.3, this restriction no longer applies.

Runtime errors in Natural applications are not handled by the Natural Web I/O Interface. This leads to a loss of the session. Bypass: use the Natural system variable *ERROR-TA to handle the error. Sample Natural error transaction:

```
DEFINE DATA
LOCAL
1 ERR_INFO
  2 ERR_NR(N5)
  2 ERR_LINE(N4)
  2 ERR_STAT(A1)
  2 ERR_PNAM(A8)
  2 ERR_LEVEL(N2)
END-DEFINE
INPUT ERR_INFO
DISPLAY ERR_INFO
TERMINATE
END
```

■ **Terminal commands**

Terminal commands are not supported. They do not work when entered in the Natural Web I/O Interface client.

■ **Natural system variable *INIT-ID**

When using the Natural Web I/O Interface client with Natural applications on UNIX, OpenVMS, mainframe or Windows hosts, the Natural system variable *INIT-ID will not be filled with a value for the terminal type. On UNIX, OpenVMS and Windows, it will contain the value "notty". On mainframes, it will contain a session ID that is unique on that server.

The following restrictions apply to Natural on UNIX, OpenVMS and Windows hosts (the mainframe does not have these restrictions):

■ **Return to the Natural main screen**

You must not use Natural applications that return to the Natural main screen as this leads to wrong screen display and a loss of the session.

■ **Natural editors and utilities**

You must not use Natural utilities such as SYSMAIN or SYSDDM and editors such as the program editor as this leads to wrong screen display and a loss of the session.

■ **Natural system commands**

You must not use any Natural system command such as CATALL, FIND, GLOBALS, HELP, KEY, LIST, RETURN, SCAN, SETUP or XREF as this leads to wrong screen display and a loss of the session.

Differences between the Natural Web I/O Interface Client and Terminal Emulation

The Natural Web I/O Interface client runs as an HTML terminal emulator inside a browser control. The look and feel of the Natural Web I/O Interface client display is quite similar to that of the regular terminal (emulation), but there are some differences due to browser functionality:

- A double-click with the mouse pointer on any field simulates the `ENTER` key.
- It is not possible to position the cursor outside the range of input and output fields.
- The cursor can be moved with the left and right arrow keys within one input field. It is also possible to jump from one input field to another using the left, right, up and down arrow keys.
- The insert mode can be switched on and off using the `INSERT` key.
- For Unicode character sets (type `U`; for example, Chinese), one character may require more space than an ordinary alphanumeric character, because the Unicode character representation is proportional. The application design must take this into account, because Natural is based on characters with fixed width. For input fields it is possible to scroll within the field, but for output fields there may not be sufficient space to display the Unicode characters. The display length for a field can be controlled by the session parameter `DL`.
- Type-ahead mode is not supported.
- Paste in overwrite mode is not supported.
- Key schemes are fixed; keys such as the right `CTRL` key and the `ENTER` key on the numeric pad are no longer definable.
- Screen update is slower since the complete screen is sent rather than updates.
- The blink attribute is not supported in Internet Explorer.
- The keys `PF1` through `PF12` are simulated by the key combinations `F1` through `F12`.
- The keys `PF13` through `PF24` are simulated by the key combinations `SHIFT+F1` through `SHIFT+F12`.
- The keys `PF25` through `PF36` are simulated by the key combinations `CTRL+F1` through `CTRL+F12`.
- The keys `PF37` through `PF48` are simulated by the key combinations `ALT+F1` through `ALT+F12`.
- The program attention keys (`PA1`, `PA2` and `PA3`) are simulated by the key combinations `CTRL+SHIFT+F1`, `CTRL+SHIFT+F2`, `CTRL+SHIFT+F3`.
- The clear key is simulated by `CTRL+SHIFT+F4`.

IBM Mainframes Only

- The terminal screen size is controlled by the Natural profile parameter `TMODEL`. The default setting `TMODEL=0` means 24 lines and 80 columns.
- There is no `ATTN` (attention interrupt) key, no `RESET` key and no `EEOF` (erase end of file) key.

VT Only

- The I/O occurs in block mode. Therefore, the Natural program will only react when a function key is pressed.

II Introducing the Natural Web I/O Interface Server CICS Adapter

3 Introducing the Natural Web I/O Interface Server CICS

Adapter

- Purpose of the Natural Web I/O Interface Server CICS Adapter 18
- CICS Support 18
- Product Interaction 19

This chapter describes the purpose and the functions of the Natural Web I/O Interface Server CICS Adapter.

Purpose of the Natural Web I/O Interface Server CICS Adapter

The Natural Web I/O Interface Server CICS Adapter is designed for a mainframe Natural context where it enables the use of a Natural Web I/O Interface server, running under z/OS in batch mode or under SMARTS on z/VSE within a CICS TP monitor environment.

CICS Support

The CICS support is not implemented within the front-end stub NATRNWO. For dispatching the Natural sessions in CICS, the Web I/O Interface server continues to run in batch mode or under SMARTS. But it uses the remote front-end NATCSRFE that is delivered with the Natural Web I/O Interface server to dispatch the Natural sessions in CICS. That is, depending on the installed front-end, a server dispatches the sessions locally (NCFNUC for SMARTS, NATMVS for batch mode) or remotely (NATCSRFE for CICS).

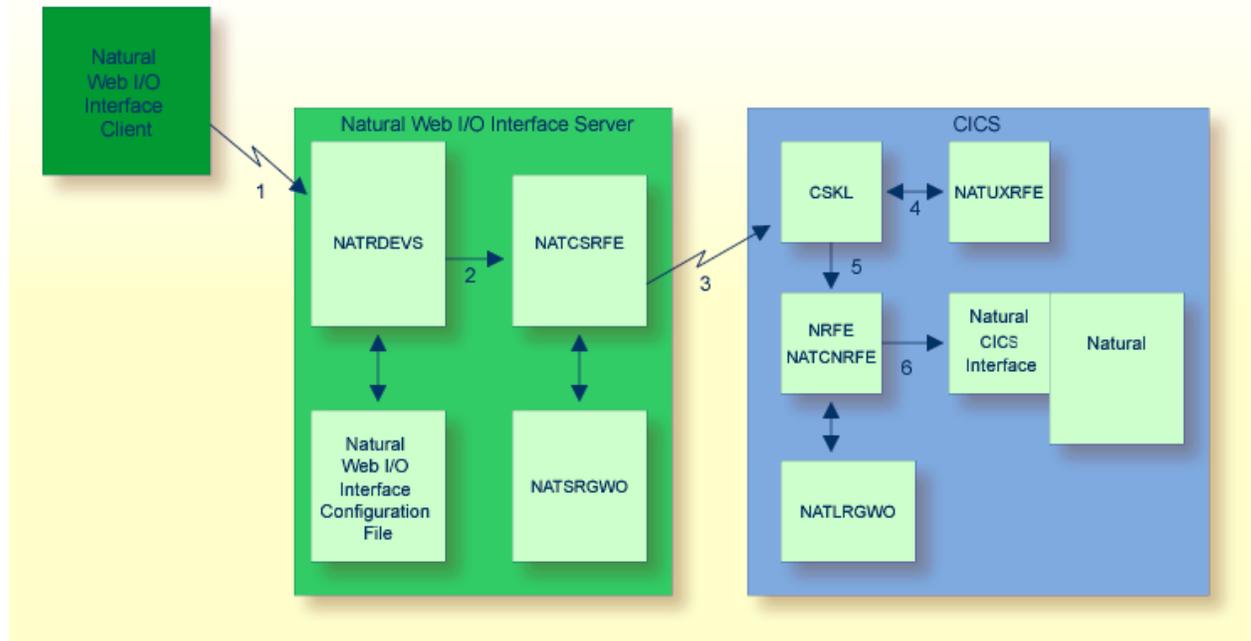
NATCSRFE in turn accepts the Natural request from NATRNWO and transfers it to a configured CICS environment using the CICS Socket Interface. Within the CICS environment, a CICS Natural transaction is launched that processes the Natural request and returns the result. Thus it is not necessary to execute the entire Web I/O Interface server under CICS. Only if Natural is requested to run the Natural application, control is transferred to CICS for execution.

The Natural Web I/O Interface Server CICS Adapter comprises the following components:

NATCSRFE	The remote front-end called by the Natural Web I/O Interface server to dispatch a Natural request. It is loaded into the Web I/O Interface server's address space.
NATCNRFE	The counterpart of NATCSRFE. NATCNRFE runs in the CICS address space. It is started by the IBM-provided standard listener of the CICS Socket Interface (refer also to <i>TCP/IP V3R1 for MVS: CICS TCP/IP Socket Interface Guide</i> and <i>TCP/IP for z/VSE V1R5 IBM Program Setup and Supplementary Information</i>).
NATSRGWO/NATLRGWO	Transmits the data relevant for Natural Web I/O Interface server between Natural Web I/O Interface server and the Natural session running in CICS. NATSRGWO must be loaded into the Natural Web I/O Interface server's address space and NATLRGWO into the CICS address space.
NATUXRFE	This user exit obtains the client credentials from the Natural Web I/O Interface server and authenticates then with a CICS VERIFY PASSWORD request. If the request succeeds, the CICS listener launches the NWO transaction under the client account (impersonation).

Product Interaction

The following figure illustrates the interaction between the Natural Web I/O Interface server and the CICS environment involved.



1. The Web I/O Interface (NWO) client sends a request to the Natural Web I/O Interface server using the port number specified with the Natural Web I/O Interface server configuration variable `PORT_NUMBER`.
2. The Natural Web I/O Interface server dispatches the Natural session using the Natural front-end you have specified with the Natural Web I/O Interface server configuration variable `FRONTEND_NAME`. Specify `NATCSRFE` in order to use the Natural Web I/O Interface Server CICS Adapter.
3. `NATCSRFE` transmits the request to the host/port specified with the Natural Web I/O Interface server configuration variable `RFE_CICS_TA_HOST / RFE_CICS_TA_PORT`. You must configure the CICS-supplied standard listener `CSKL (z/OS)` or `EZAL (z/VSE)` to listen at this port.
4. If the Natural Web I/O Interface server is configured to perform remote impersonation (`SECURITY_MODE=IMPERSONATE/IMPERSONATE_REMOTE`), `NATUXRFE` is called to authenticate the client. If the authentication succeeds, `CSKL` launches the CICS transaction `NRFE` under the account of the client (impersonated).
5. `CSKL` launches the CICS transaction you have specified with the Natural Web I/O Interface server configuration parameter `RFE_CICS_TA_NAME` (`NRFE` in this example). This transaction must be defined to use the program `NATCNRFE`.

6. NATCNRFE finally dispatches the Natural session using the Natural CICS front-end you have specified with the Natural Web I/O Interface server configuration parameter `RFE_CICS_FE_NAME`.

III

Introducing the Natural Web I/O Interface Server IMS

Adapter

4 Introducing the Natural Web I/O Interface Server IMS Adapter

- Purpose of the Natural Web I/O Interface Server IMS Adapter 24
- IMS TM Support 24
- Product Interaction 25

This chapter describes the purpose and the functions of the Natural Web I/O Interface Server IMS Adapter.

Purpose of the Natural Web I/O Interface Server IMS Adapter

The Natural Web I/O Interface Server IMS Adapter is designed for a mainframe Natural context where it enables the use of a Natural Web I/O Interface server running under z/OS in batch mode within an IMS TM environment.

IMS TM Support

The IMS TM support is not implemented within the front-end stub NATRNWO. For dispatching the Natural sessions in IMS TM, the Web I/O Interface server continues to run in batch mode. But it uses the remote front-end NATISRFE that is delivered with the Natural Web I/O Interface server to dispatch the Natural sessions in IMS TM. That is, depending on the installed front-end, a server dispatches the sessions locally (NATMVS for batch mode) or remotely (NATISRFE for IMS TM).

NATISRFE in turn accepts the Natural request from NATRNWO and transfers it to a configured IMS TM environment using the IMS installation provided BPM listener. The IMS listener launches in a dedicated MPP region a non conversational Natural transaction that processes the Natural request and returns the result. Thus it is not necessary to execute the entire Natural Web I/O Interface server under IMS TM. Only small working units (Natural requests such as “save source” or “get library list”) are transferred to IMS TM for execution.

The Natural Web I/O Interface Server IMS Adapter comprises the following components:

NATISRFE	The remote front-end called by the Natural Web I/O Interface server to dispatch a Natural request. It is loaded into the Web I/O Interface server's address space.
NATINRFE	The counterpart of NATISRFE. NATINRFE runs in an MPP region. It is launched by the IBM-provided IMS Listener (refer also to z/OS Communications Server IP IMS Socket Guide).
NATSRGWO/NATLRGWO	Transmits the data relevant for Natural Web I/O Interface server between Natural Web I/O Interface server and the Natural session running in IMS TM. NATSRGWO must be loaded into the Natural Web I/O Interface server's address space, and NATLRGWO into the MPP region.

Product Interaction

The following description explains the interaction between the Natural Web I/O Interface server and the IMS TM environment involved.

1. The Web I/O Interface (NWO) client sends a request to the Natural Web I/O Interface server using the port number specified with the Natural Web I/O Interface server configuration variable `PORT_NUMBER`.
2. The Natural Web I/O Interface server dispatches the Natural session using the Natural front-end you have specified with the Natural Web I/O Interface server configuration variable `FRONTEND_NAME`. Specify `NATISRFE` in order to use the Natural Web I/O Interface Server IMS Adapter.
3. `NATISRFE` transmits the request to the host/port specified with the Natural Web I/O Interface server configuration variable `RFE_IMS_TA_HOST / RFE_IMS_TA_PORT`. You must configure the IBM provided BMP Listener to listen at this port.
4. The BMP Listener launches the IMS TM transaction you have specified with the Natural Web I/O Interface server configuration parameter `RFE_IMS_TA_NAME`. This transaction must be specified in the configuration of the IMS Listener.
5. The server transaction first retrieves the transaction initialization message. `TIM` contains the necessary information to do the `TAKESOCKET` and passes it to `NATINRFE`. `NATINRFE` does the `TAKESOCKET` and establishes the TCP/IP conversation with `NATISRFE`.
6. `NATINRFE` finally dispatches the Natural IMS front-end, which establishes a Natural server session.

IV Installing and Configuring the Natural Web I/O Interface

Server

The Natural Web I/O Interface server is available on z/OS and (under SMARTS) on z/VSE and BS2000.



Note: SMARTS is an acronym for “Software AG Multi-Architecture Runtime System”. It constitutes a runtime layer that allows POSIX-like applications to run on mainframe operating systems. Software AG products communicate with the operating system through the SMARTS layer.

This part covers the following topics:

[Natural Web I/O Interface Server Concept and Structure](#)

[Prerequisites](#)

[Installing the Natural Web I/O Interface Server under z/OS](#)

[Installing the Natural Web I/O Interface Server under SMARTS on z/VSE](#)

[Installing the Natural Web I/O Interface Server under SMARTS on BS2000](#)

[Configuring the Natural Web I/O Interface Server](#)

Natural Web I/O Interface Server CICS Adapter

The following topics apply in addition if you want to use the Natural Web I/O Interface server in a CICS environment under z/OS or z/VSE:

[Installing the Natural Web I/O Interface Server CICS Adapter under z/OS](#)

[Installing the Natural Web I/O Interface Server CICS Adapter under SMARTS on z/VSE](#)

[Configuring the Natural Web I/O Interface Server CICS Adapter](#)

Natural Web I/O Interface Server IMS Adapter

The following topics apply in addition if you want to use the Natural Web I/O Interface server in an IMS TM environment under z/OS:

[Installing the Natural Web I/O Interface Server IMS Adapter](#)

[Configuring the Natural Web I/O Interface Server IMS Adapter](#)

Notation *vrs* or *vr*

When used in this documentation, the notation *vrs* or *vr* represents the relevant product version (see also *Version* in the *Glossary*).

5

Natural Web I/O Interface Server Concept and Structure

- Web I/O Interface Server Concept 30
- Natural Web I/O Interface Server under SMARTS on BS2000 30
- Front-End Stub NATRNWO 31
- Front-End 32
- Server Monitor 33

This chapter describes the concept and the structure of the server for the Natural Web I/O Interface which is designed for use on z/OS and (under SMARTS) on z/VSE or BS2000.

Web I/O Interface Server Concept

A Natural Web I/O Interface server is a multi-user, multi-tasking application. It can host Natural sessions for multiple users and execute their applications concurrently.

The concept is based on the “serverized” Natural runtime system. Its architecture comprises a server front-end stub (Web I/O Interface server stub NATRNWO) that uses the Natural front-end to dispatch Natural sessions and to execute applications within these sessions.

The Natural Web I/O Interface server architecture basically consists of:

- SMARTS runtime environment (only on z/VSE and BS2000)

SMARTS is used to implement a server runtime environment for the execution of the Web I/O Interface server.

- **Front-end stub**

The stub NATRNWO is launched to initialize a Natural Web I/O Interface server. It listens for incoming connection requests and launches a Natural session for executing the application.

- **Front-end**

The front-end is called (together with the Natural runtime system) by the front-end stub for session initialization/termination, application execution and session roll-in/roll-out.

- **Server monitor**

A monitor task allows the administrator to control the server activities, to cancel particular user sessions or to terminate the entire server, etc.

Natural Web I/O Interface Server under SMARTS on BS2000

SMARTS is an acronym for “Software AG Multi-Architecture Runtime System”. It constitutes a runtime layer that allows POSIX-like applications to run on mainframe operating systems. Software AG products communicate with the operating system through the SMARTS layer.

SMARTS on BS2000 Basics

SMARTS implements a server runtime environment for the execution of the Web I/O Interface server. Technically, SMARTS represents a C runtime system and implements a nearly full-blown POSIX system. It drives a family of tasks which either process dedicated functionality or process the application payload in parallel-executed worker tasks. The tasks with a dedicated functionality are the main or oc task, the system thread loop task (started as second task), the socket communication task, the pfs task and the sequential file I/O task. The pfs task is optional; it processes all I/O operations on the POSIX file system (PFS). The PFS is used for Web I/O Interface server work/print file access method, thus allowing the testing of programs which execute access to work or print files.

SMARTS offers a configurable set of resources to process the application workload. These resources are mainly threads and (worker-) tasks. The actual workload is scheduled by the SMARTS kernel using these resources. In case of momentary shortages of one or the other resource, SMARTS is able to queue incoming requests and to roll-out inactive threads.

All data, processed by SMARTS or a SMARTS application, is located in one common memory pool, the data pool. All code modules, except for some smaller bootstrap routines, are loaded as shared code into another common memory pool, the code pool.

The worker-tasks are the processes (TSNs) by which the Natural runtime is executed. The amount of storage requested by Natural is located in the SMARTS threads within the data pool during the execution of a transaction. If the number of sessions to be processed exceeds the number of threads defined, an internal facility is invoked for rolling the threads. Threads rolled out are placed in compressed format in a so-called roll buffer pool which resides in the data common memory pool as well.

Front-End Stub NATRNWO

The multi-user, multi-tasking, front-end stub NATRNWO is launched to initialize a Natural Web I/O Interface server.

The following topics are covered below:

- [Stub Description](#)

- **Natural System Variables Used**

Stub Description

The task executing the server initialization (TMain) basically is the main listener which waits for incoming requests from the Web I/O Interface client. It owns a session directory to manage multiple clients (users) and their corresponding remote Natural sessions. TMain has the task to accept all incoming requests and to dispatch them to other subtasks (TWork). The process is as follows:

- First, a server connection issued by the user on the client side (the Login button of the Web I/O Interface client) connects to TMain to establish a connection.
- Next, TMain inserts the client into its session directory, attaches a new TWork subtask and passes the connection to TWork.
- TWork initializes a new Natural session and starts the specified Natural application program.
- While the application performs I/O requests, TWork intercepts the I/O data and passes them to the Web I/O Interface client for processing the I/O. The I/O reply is sent back to the server and the server continues the application.
- If the application terminates (reaches the NEXT mode), TWork terminates the Natural session and drops the connection to the Web I/O Interface client.

That is, each client owns one subtask TWork on the Natural Web I/O Interface server. This subtask runs a Natural session (and within the Natural session, a Natural application) and remains active as long as the application is running.

Natural System Variables Used

Within a Natural Web I/O Interface server session, the following Natural system variables are used:

- *TPSYS contains SERVSTUB,
- *DEVICE contains BROWSER,
- *SERVER-TYPE contains WEBIO.

Front-End

Under z/OS, the Natural front-end required for a Natural Web I/O Interface server is a Natural batch front-end driver, which should be LE enabled. See sample installation jobs for details.

Under , the front-end is called (together with the Natural runtime system) by the front-end stub for session initialization/termination, request execution and session roll-in/roll-out.

Under z/VSE and BS2000, the Natural front-end required for a Natural Web I/O Interface server is the Natural Com-plete driver NCFNUC that is delivered with the corresponding Natural Version for Mainframes.

The Natural front-end required for executing the Natural sessions under control of CICS is the Natural remote front-end NATCSRFE that is delivered with the Natural Web I/O Interface server. For more information, refer to the *Natural Web I/O Interface Server CICS Adapter* documentation.

Server Monitor

To enable the administrator to monitor the status of the Natural Web I/O Interface server, a monitor task is provided which is initialized automatically at server startup. Using the monitor commands, the administrator can control the server activities, cancel particular user sessions, terminate the entire server, etc. See [Operating the Web I/O Interface Server](#).

6 Prerequisites

- General Prerequisites for Web I/O Interface Server Installation 36
- Prerequisites for the Web I/O Interface Server for z/OS 36
- Prerequisites for the Web I/O Interface Server under SMARTS on z/VSE 36
- Prerequisites for the Web I/O Interface Server under SMARTS on BS2000 37

This chapter describes the prerequisites that apply when you install a Natural Web I/O Interface server on a mainframe computer.

General Prerequisites for Web I/O Interface Server Installation

The following general prerequisites apply:

- The currently applicable version of Natural for Mainframes must be installed.
- The International Components for Unicode for Software AG (ICS) must be installed with the Web I/O Interface server; see *International Components for Unicode for Software AG* in the *Unicode and Code Page Support* documentation.



Note: For further information, refer to the products and versions specified under *Software AG Product Versions Supported by Natural* in the current *Natural Release Notes*.

Prerequisites for the Web I/O Interface Server for z/OS

In addition to the general prerequisites described above, the following operating-system-specific prerequisites apply:

- z/OS must be installed.
- To prevent the formation of endless loops in user programs running under the Web I/O Interface, specify a reasonable value for Natural profile parameter MT (maximum CPU time).

Prerequisites for the Web I/O Interface Server under SMARTS on z/VSE

In addition to the [general prerequisites](#) described above, the following operating-system-specific prerequisites apply:

- z/VSE must be installed.
- SMARTS must be installed (product code APS).
- The Natural Com-plete/SMARTS Interface must be installed (product code NCF).
- To prevent the formation of endless loops in user programs running under NWO, specify a reasonable value for the CPU time limit in the SMARTS parameter THREAD-GROUP.
- As a prerequisite for using the client impersonation feature (parameter SECURITY_MODE), Natural Security must be installed.

Prerequisites for the Web I/O Interface Server under SMARTS on BS2000

In addition to the [general prerequisites](#) described above, the following operating-system-specific prerequisites apply:

- BS2000 must have been installed on the server mainframe.
- SMARTS must be installed (product code APS).
- The Natural Com-plete/SMARTS Interface must be installed (product code NCF)

7 Installing the Natural Web I/O Interface Server under z/OS

- Prerequisites 40
- Content of the Web I/O Interface Server Distribution Medium 40
- Installation Procedure 40

This chapter describes how to install a server for the Natural Web I/O Interface (product code NWO) under the operating system z/OS.

The installation of the Web I/O Interface server is performed by installation jobs. The sample jobs are contained in the data set `NW0vrs.JOBS` (where *vrs* represents the relevant product version) and are prefixed with `NW0`, or generated by System Maintenance Aid (SMA).

Notation *vrs* or *vr*

When used in this documentation, the notation *vrs* or *vr* represents the relevant product version (see also *Version* in the *Glossary*).

Prerequisites

For details, refer to the section [Prerequisites](#).

Content of the Web I/O Interface Server Distribution Medium

The installation medium contains the data sets listed in the table below. The sequence of the data sets and the number of library blocks needed are shown in the *Software AG Product Delivery Report*, which accompanies the installation medium.

Data Set Name	Contents
<code>NW0vrs.OBJS</code>	Contains the object modules of the server.
<code>NW0vrs.JOBS</code>	Example installation jobs.

Installation Procedure

Step 1: Allocate the Web I/O Interface Server LOAD library

(Job I008, Step 9410)

Step 2: Create a Web I/O Interface Server configuration file and sample Clist

(Job I009 / Step 9410, 9420, 9430)

Step 9410 creates a sample `NWOCONFIG` for the batch server.

Step 9420 creates a sample `Clist` to ping and terminate a Web I/O Interface server.

Step 9430 creates a sample batch job to ping and terminate a Web I/O Interface server.

The following parameters of the configuration file have to be defined. See [Configuring the Natural Web I/O Interface Server](#). For the other parameters, the default values may be used:

FRONTEND_NAME	Specify the name of the Web I/O Interface server front-end module you will generate in one of the following steps.
PORT_NUMBER	Specify the TCP/IP port number under which the server can be connected.

Step 3: Link the object modules into the NWO load library

(Job I054, Step 9410)

The NWO object modules must be linked with the necessary runtime extensions of your batch installations into executable load modules.



Note: The module `NATCNRF` applies to two different products, the Natural Web I/O Interface Server (NWO) and the Natural Development Server (NDV). So if you have already installed NDV, the module `NATCNRF` might already be there. However, it does not matter if you re-install `NATCNRF` with NWO because the resulting module from either installation is the same.

See sample job `NWOI054` on data set `NWOvrs.JOBS`.

Step 4: Create the Natural parameter module and NWO server front-end module

(Job I060, Steps 9410, 9420, 9430)

- Job I060, Step 9410 starts the batch program `IEBUPDATE` to store the parameter module `NWOPARM`.
- Job I060, Step 9420 assembles and links the parameter module `NWOPARM`.
- Job I060, Step 9430 links the NWO server front-end module.

The reentrant `ADALINK` module `ADALNKR` must be used.



Note: If you have Natural subproducts for IBM database access installed, for example, Natural for DB2, Natural for DL/I, Natural for VSAM, you need to adjust the link job to include `ATTRCSS` from `SYS1.CSSLIB` in order to enable the use of `SYNCPPOINT/ROLLBACK` functionalities.

Step 5: Create server startup JCL

(Job I200, Step 9415)

Described in the section [Configuring the Natural Web I/O Interface Server](#). See sample member NWOSTART on data set NW0vrs.JOBS.

Step 9415 creates a startup procedure for the batch server.

Sample:

```
//          PROC SRV=SAGNWO
//NWO      EXEC PGM=NATRWO ,
// REGION=4000K,TIME=1440,PARM=' POSIX(ON),TRAP(ON,NOSPIE)/&SRV '
//STEPLIB DD  DISP=SHR,DSN=NW0vrs.LOAD
//          DD  DISP=SHR,DSN=SAGLIB.SMALOAD
//SYSUDUMP DD  SYSOUT=X
//CEEDUMP DD  SYSOUT=X
//CMPRINT DD  SYSOUT=X
//STGCONFIG DD  DISP=SHR,
//          DSN=NW0.CONFIG(&SRV)
//STGTRACE DD  SYSOUT=X
//STGSTDO DD  SYSOUT=X
//STGSTDE DD  SYSOUT=X
//SYSOUT   DD  SYSOUT=X
```



Note: The Web I/O Interface server account must be defined in the z/OS UNIX System Services (OE segment). If the server account is not defined, the server ends with U4093 and system message CEE5101C in the trace file.

Step 6: Web I/O Interface clients must be defined to Natural Security

If Natural Security (NSC) is installed:

- The Web I/O Interface initial user ID (default ID is STARGATE) must be defined in Natural Security with a valid default library. Refer also to Web I/O Interface server configuration parameter [INITIAL_USERID](#). Alternatively, you can define the Natural profile parameter `AUTO=OFF` (automatic logon) for the Web I/O Interface.
- Each client user ID must be defined in Natural Security.

If the Web I/O Interface initial user ID is not defined, the Web I/O Interface server initialization aborts with a NAT0856.

If a Web I/O Interface client is not defined, the server connection returns an NSC error.

If you connect to the server from a Web I/O Interface client, make sure that the user who is defined in Natural Security has a default library or a private library defined. Otherwise, error message NAT0815 will occur.

Step 7: Web I/O Interface clients must be defined to the server host

If you configure the Web I/O Interface server to use an external security system (see Web I/O Interface server configuration parameter [SECURITY_MODE](#)), the Web I/O Interface clients must be defined to the external security system.

8 Installing the Natural Web I/O Interface Server under SMARTS

on z/VSE

▪ Prerequisites	46
▪ Content of the Web I/O Interface Server Distribution Medium	46
▪ Installation Procedure	47
▪ Allocating and Configuring a SMARTS Portable File System	49

This chapter describes how to install a server for the Natural Web I/O Interface (product code NWO) under the runtime environment SMARTS on z/VSE.

The installation of the Web I/O Interface server is performed by installation jobs. The sample jobs are contained in the data set `NW0vrs.LIBJ`, or generated by System Maintenance Aid (SMA).

Notation *vrs* or *vr*

When used in this documentation, the notation *vrs* or *vr* represents the relevant product version (see also *Version* in the *Glossary*).

Prerequisites

For details, refer to the section [Prerequisites](#).

Content of the Web I/O Interface Server Distribution Medium

The installation medium contains the data sets listed in the table below. The sequence of the data sets and the number of library blocks needed are shown in the *Software AG Product Delivery Report*, which accompanies the installation medium.

Data Set Name	Contents
<code>APSVrs.LIBR</code>	Contains the load modules of the SMARTS server.
<code>NW0vrs.LIBR</code>	Contains the load modules of the Web I/O Interface server. See Web I/O Interface Server Concept and Structure .
<code>NW0vrs.LIBJ</code>	Contains Installation Job Control for customers who install without using System Maintenance Aid.

Installation Procedure

Step 1: Create a Web I/O Interface server configuration file

(Job I009, Step 9400)

Define sublibrary for NWO environment.

(Job I009, Step 9410)

Catalogs the configuration file of the Web I/O Interface server. For a description of the parameters, refer to [Configuring the Natural Web I/O Interface Server](#).

The following parameters of the configuration file must be defined. For the other parameters, the default values may be used:

FRONTEND_NAME	Specify the name of the server front-end module you will create in one of the following steps.
PORT_NUMBER	Specify the TCP/IP port number under which the server can be connected.

Step 2: Create a SMARTS SYSPARM file

(Job I009, Step 9420)

Catalogs the configuration file SYSPARM for SMARTS.

For detailed information on the SMARTS configuration file, see *Configuration of the SMARTS Environment* in the SMARTS documentation.

(Job I009, Step 9430)

Create dummy member for NWO.

Step 3: Link the object modules into the NWO load library

(Job I054, Step 9410)

The NWO object modules must be linked with the necessary runtime extensions of your batch installations into executable load modules.

Step 4: Assemble and link reentrant ADALNK

Job I055, Step 9401, assemble and link reentrant ADALNK phase.

The server environment requires a reentrant ADALNK phase.

Link ADALNK using the ADALNKR module.

Step 5: Assemble and catalog the Web I/O Interface server module

- Job I055, Step 9410, assemble and catalog the NCFNWOPM module.

Step 6: Create the Web I/O Interface server front-end module

(Job I060, Steps 9410, 9420)

- Job I060, Step 9410, assemble and catalog the Natural parameter module for NWO.
- Job I060, Step 9420, link the NWO front-end.

Step 7: Catalog the SMARTS startup job

(Job I200, Step 9415)

Extend your SMARTS startup job by NWO-specific definitions described in the section *Configuring the Natural Web I/O Interface Server*.

VSE Sample:

```
* $$ JOB JNM=NWOSRV,CLASS=C,DISP=L,LDEST=(,UID)
* $$ LST CLASS=A,DISP=H
// JOB NWOSRV --- NWO SERVER STARTUP ---
// OPTION PARTDUMP,NOSYSDMP,LOG
/* NWO server data sets -----
// DLBL NWOSRVT,'SYSLST'
// DLBL NWOSRVC,'/SAGLIB/NWOCNFG/NWOSRV.P' location of NWO configuration file
// DLBL NWOSRVE,'SYSLST' NWO error output directed to job ←
output
// DLBL NWOSRVO,'SYSLST'
// DLBL SYSPARM,'/SAGLIB/NWOCNFG/MSGQ.DMY' location of NWO Dummy file
// DLBL STDOUT,'CONSOLE' STDOUT directed to VSE console
// DLBL STDERR,'CONSOLE'
/* Libdef's -----
/*
// LIBDEF PHASE,SEARCH=(SAGLIB.NWOCNFG, +
SAGLIB.APSvrs, +
SAGLIB.ADAvrs)
/* *****
```

```
// UPSI    00000000
// EXEC    TLINSP,SIZE=AUTO
* $$ SLI MEM=RJANPARM.P,S=SAGLIB.APSvrs
* $$ SLI MEM=PXANCONF.P,S=SAGLIB.APSvrs
* $$ SLI MEM=SYSPARM.P,S=SAGLIB.NWOCNFG           NWO specific SMARTS configuration ←
file
/*
// EXEC    LISTLOG
```

Step 8: Web I/O Interface clients must be defined to Natural Security

If Natural Security (NSC) is installed:

- The Web I/O Interface initial user ID (default ID is STARGATE) must be defined in Natural Security with a valid default library. Refer also to Web I/O Interface server configuration parameter `INITIAL_USERID` in the section [Configuring the Natural Web I/O Interface Server](#). Alternatively, you can specify the Natural profile parameter `AUTO=OFF` (automatic logon) for the Web I/O Interface.
- Each client user ID must be defined in Natural Security.

If the Web I/O Interface initial user ID is not defined, the Web I/O Interface server initialization aborts with a NAT0856 error message.

If a Web I/O Interface client is not defined, the **Map Environment** dialog returns an NSC error.

If you connect to the server from a Web I/O Interface client, make sure that the user who is defined in Natural Security has a default library or a private library defined. Otherwise, the error message NAT0815 will be displayed.

Step 9: Web I/O Interface clients must be defined to the server host

If you configure the Web I/O Interface server to use an external security system (see Web I/O Interface server configuration parameter `SECURITY_MODE`), the Web I/O Interface clients must be defined to the external security system.

Allocating and Configuring a SMARTS Portable File System

The Natural server uses the SMARTS portable file system (PFS) as a data container for Natural work files, print files, temporary sort files and the editor work file. The SMARTS PFS is the only storage medium available for those files under SMARTS.

In order to be able to use the PFS for Natural files, you have to configure Natural accordingly:

- For work or print files, specify the access method `AM=SMARTS`, using the Natural profile parameters `WORK` and/or `PRINT`.

- For temporary sort files, specify the type of storage medium `STORAGE=SMARTS`, using the Natural profile parameter `SORT`.
- And to allocate the editor work file in the PFS, specify the work file mode `FMODE=SM`, using the Natural profile parameter `EDBP`.

If you use one of these options, you have to configure your SMARTS to use a PFS.

Step 1: Allocate a PFS

Allocate a z/VSE file (`LRECL=4096`) with the estimated size (the file must start at cylinder boundary) to store your PFS, and completely initialize the container to contain `x'00's`.

The initialization can be done using the following job (runs until no more extents are available in the file and requires extra operator intervention to continue the job after the WTOR message for the extent overflow):

```
// JOB PFSFMT
// OPTION NODUMP
// DLBL CMWKF01,'your.pfs.file.name',0,SD
// EXTENT SYS001,volume,...
// ASSGN SYS001,DISK,VOL=volume,SHR
// ASSGN SYS000,READER
// UPSI 00000011
// LIBDEF PHASE,SEARCH=(SAGLIB.NATvrs)
// EXEC NATBATvr,SIZE=AUTO,PARM='SYSRDR'
IM=D,MADIO=0,MT=0,OBJIN=R,ID=',',INTENS=1,
WORK=((1),AM=STD,RECFM=FB,BLKSIZE=4096,LRECL=4096)
EDT
DEFINE DATA LOCAL
1 B(B1) INIT<H'00'>
END-DEFINE
DEFINE WORK FILE 1 'CMWKF01'
REPEAT
WRITE WORK 1 B
WHILE 1=1
END
.E
RUN
FIN
/*
```

Step 2: Configure SMARTS

In the SMARTS startup JCL, add the following statements:

```
// DLBL APSPFS1,'your.pfs.file.name',0,DA
// EXTENT SYSvrs,volume,...
// ASSGN SYSvrs,DISK,VOL=volume,SHR
```

In the SMARTS SYSPARM file, add the following statements (where *vrs* or *vr* represents the relevant product version):

```
RESIDENTPAGE=NCFvrAPS
CDI_DRIVER=('CIO,PAANCIO')
CDI=('PFS1,PAANPFS,CACHESIZE=2000,LRECL=4096,CONTAINER=CIO://DD:APSPFS1')
MOUNT_FS=('PFS1://','/usr/')
ENVIRONMENT_VARIABLES=/SAGLIB/APSvrs/ENVARS.P
```

Create a member `ENVARS.P` under `SAGLIB.APSvrs` containing the following lines:

```
NAT_WORK_ROOT=/usr/natural/etc/work_file
NAT_PRINT_ROOT=/usr/natural/etc/print_file
NCFWFAPS_TRACE_LEVEL=0
```

`NCFWFAPS_TRACE_LEVEL=31` may be used for diagnostic purposes. It causes all PFS accesses to be traced by Natural and to be written to the Web I/O Interface server data set `STDOUT`. It is recommended to set the `NCFWFAPS_TRACE_LEVEL` parameter value to zero.

Step 3: Verify PFS Configuration

SMARTS now should protocol the following line in `SYSLST` during startup:

```
APSPSX0050-SYSNAME CDI PFS1 PROTOCOL INITIALIZED
```

Once your Natural Web I/O Interface server installation has been completed, issue a **Map Environment** command to the Natural Web I/O Interface server, using the following session parameter:

```
WORK=((1),AM=SMARTS)
```

The following Natural program should run and display Record 01:

```
0010 DEFINE DATA LOCAL
0020 1 A(A20)
0030 END-DEFINE
0040 A := 'Record 01'
0050 WRITE WORK 1 A
0060 CLOSE WORK 1
0070 READ WORK 1 A
0080 WRITE A
0090 END-WORK
0100 END
```

9 Installing the Natural Web I/O Interface Server under SMARTS on BS2000

▪ Prerequisites	54
▪ Installation Medium	54
▪ Installation Procedure	55
▪ Installation Verification	56
▪ SMARTS Portable File System on BS2000	57

This chapter describes how to install a server for the Natural Web I/O Interface (product code NWO) in the runtime environment SMARTS on BS2000.

Notation *vrs* or *vr*

When used in this documentation, the notation *vrs* or *vr* represents the relevant product version (see also *Version* in the *Glossary*).

Prerequisites

For details, refer to the section [Prerequisites](#).

Installation Medium

Content of the Web I/O Interface Server Distribution Medium

The installation medium contains the data sets listed in the table below. The sequence of the data sets and the number of library blocks needed are shown in the *Software AG Product Delivery Report*, which accompanies the installation medium.

Data Set Name	Contents
APSVrs.LIB	Contains the load modules and the procedures of the SMARTS server.
APSVrs.L0nn	Fix level of SMARTS (product code APS). The content of this library must be copied into the library APSVrs.LIB.
NW0vrs.MOD	Contains the load modules of the Natural Web I/O Interface server.
NW0vrs.JOBS	Contains the Installation Job Control for those customers who want to install without using System Maintenance Aid (SMA).
NCFvrs.MOD	Contains the load modules of the Natural Complete Interface (required for SMARTS).

where *vrs* represents the relevant product version, and *nn* in a data set name represents the fix level of the product.

Content of the Web I/O Interface Server JOBLIB

Naming Conventions

In the following text, the library name JOBLIB stands for

- the example job library (NWO *vs.* JOBS), if you are not using SMA, or
- the SMA job library (see SMA parameter JOBLIB in SMA Parameter Group BASIC), if you are using SMA.

Software AG uses the following naming conventions for source elements in the library JOBLIB:

A<*product-code*><*function*> = Assembler sources

L<*product-code*>< *function*> = Instruction for TSOSLNK/BINDER

Important Elements of the Job Library

Element	Description
NWO-SYSPARM	SMARTS parameters for NWO.
NWO-CONFIG	NWO configuration parameters.
NWO-ADAPARM	ADALNK parameter for NWO.
ANWOPARM	Assembler source - Natural parameter module for NWO.
LNATSHAR	Link instructions to link the Natural nucleus.
LNWOFRNT	Link instructions to link the NWO front-end module.
START-NWO	Procedure to start the NWO server.
STOP-NWO	Procedure to stop the NWO server.
SHOW-SYSOUT	Show SYSOUT file.

Installation Procedure

To install the Natural Web I/O Interface server in a SMARTS environment on BS2000, perform the following steps:

Step 1: Assemble the Natural parameter module

(Job I055, Step 9420)

Assemble the source ANWOPARM which is contained in the library JOBLIB.

Step 2: Link the NWO front-end module

(Job I060, Step 9410)

Link the NWO front-end module by using the source LNWOFRNT which is contained in the library JOBLIB.

Step 3: NWO clients must be defined to Natural Security

If Natural Security (NSC) is installed:

- The Web I/O Interface initial user ID (default ID is STARGATE) must be defined in Natural Security with a valid default library. Refer also to Web I/O Interface server configuration parameter [INITIAL_USERID](#). Alternatively, you can define the Natural profile parameter AUTO=OFF (automatic logon) for the Web I/O Interface.
- Each client user ID must be defined in Natural Security.

If the Web I/O Interface initial user ID is not defined, the Web I/O Interface server initialization aborts with a NAT0856.

If a Web I/O Interface client is not defined, the server connection returns an NSC error.

If you connect to the server from a Web I/O Interface client, make sure that the user who is defined in Natural Security has a default library or a private library defined. Otherwise, error message NAT0815 will occur.

Installation Verification

Check Messages

BS2000 Operator Console

```
APSSVR0026-* Server NCFNATvr started
APSOPC0000-* SMARTServer is initialized
```

Where *vr* is the current product version and release number.

SYSOUT of FSIO-Task

The following messages are displayed when parameter `TRACE_LEVEL=31` was set:

```
Template session initialized
Template runtime connected
Template runtime terminated
Template ready for clone
Listener thread launched 01625028
*****
***** Server is up *****
*****
Waiting for terminate event...
```

SMARTS Portable File System on BS2000

The following topics are covered below:

- [PFS Description](#)
- [Allocating and Configuring a SMARTS Portable File System](#)

PFS Description

SMARTS PFS (Portable or POSIX File System) implements a file system, known from UNIX systems, in a mainframe environment. Basically, it consists of a container file, which comprises all (UNIX) files and a corresponding (logical) access method, which processes all requested I/O operations.

The container file has to be allocated and preformatted using the BS2000 procedure `CREATE-PFS` described below.

The PFS maps all file names to a node of a directory tree within the physical container file. In the case of BS2000, this container file is a PAM file with a block size of 4 KB (STD, 2).

Within Natural, the actual path is specified by a corresponding `DEFINE WORK FILE` statement in the program which executes work file or print file access.

Each node (subdirectory) is separated by a slash (/) from its parent. The highest level qualifies the file name.

Example 1:

```
DEFINE WORK FILE 1 '/MISC/USER1/TESTFILE/'
```

Specifies: ROOT' => MISC => 'USER1 => 'TESTFILE

```
://MISC/USER1/TESTFILE
```

Example 2:

```
DEFINE WORK FILE 1 'TESTFILE2.W01'
```

Specifies: ROOT' => TESTFILE2.W01

Allocating and Configuring a SMARTS Portable File System

The Natural server uses the SMARTS portable file system (PFS) as a data container for Natural work files, print files, temporary sort files and the editor work file. The SMARTS PFS is the only storage medium available for these files under SMARTS.

In order to be able to use the PFS for Natural files you have to configure Natural accordingly:

- For work files or print files, specify the access method `AM=SMARTS`, using the Natural profile parameters `WORK` (Work-File Assignments) and/or `PRINT` (Print File Assignments).
- For temporary sort files, specify the type of storage medium `STORAGE=SMARTS`, using the Natural profile parameter `SORT` (Control of Sort Program).
- And to allocate the editor work file in the PFS, specify the work file mode `FMODE=SM`, using the Natural profile parameter `EDBP` (Software AG editor buffer pool definitions).

If you use one of these options, you have to configure your SMARTS to use a PFS.

Step 1: Allocate a PFS

The file `PFS.TST` should not exist before the allocation procedure is executed, otherwise DMS error `DMS0683` will occur.

The initialization can be done by using the following procedure:

```
/CLP FROM-FILE=*LIBRARY-ELEMENT(LIBRARY=APSVrs.LIB,ELEMENT=CREATE-PFS, -  
/   TYPE=SYSJ),PROCEDURE-PARAMETERS=(FILE-NAME=PFS.TEST,SIZE=4096K, -  
/   APS-LIB=APSVrs.LIB)
```

Step 2: Configure SMARTS

In the SMARTS SYSPARM configuration file, add the following lines:

```
CDI_DRIVER=( 'CIO,PAAQBIO,PFSTSK=PFSTASK,TRACE=N' )  
CDI_DRIVER=( 'TESTPFS,PAANPFS,LRECL=4096,CONTAINER=CIO:PFS/TEST/' )  
MOUNT_FS=( 'TESTPFS://','/' )
```


10

Configuring the Natural Web I/O Interface Server

▪ Configuration Requirements for z/OS	62
▪ Configuration Requirements for z/VSE	69
▪ SMARTS Configuration File for BS2000	71
▪ Web I/O Interface Server Configuration File for z/OS and z/VSE	80
▪ Web I/O Interface Server Configuration Parameters	80
▪ Web I/O Interface Server Configuration File Example	92
▪ Web I/O Interface Server Data Sets for z/OS, z/VSE	93
▪ Web I/O Interface Server User Exits	93

This chapter describes how to configure a Natural Web I/O Interface server.

Where applicable, specific information for z/OS, z/VSE or BS2000 is provided.

Configuration Requirements for z/OS

The following topics are covered:

- [Language Environment Parameter Settings](#)
- [External Security Configuration](#)
- [SSL Support](#)

Language Environment Parameter Settings

A Natural Web I/O Interface server requires the following z/OS language environment parameter configuration:

Parameter	Definition				
POSIX(ON)	Enables a Natural Web I/O Interface server to access the POSIX functionality of z/OS. If you start a Natural Web I/O Interface server server with POSIX(OFF), it terminates immediately with a user abend U4093 and the system message EDC5167. IBM supplies the default "OFF".				
TRAP(ON,NOSPIE)	<p>Defines the abend handling of the LE/370 environment:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 20%;">ON</td> <td>Enables the Language Environment condition handler.</td> </tr> <tr> <td>NOSPIE</td> <td>Specifies that the Language Environment will handle program interrupts and abends via an ESTAE, that is, the Natural abend handler will receive control to handle program interrupts and abends.</td> </tr> </table> <p>If you do not specify TRAP(ON,NOSPIE), the Natural abend handling does not work properly. IBM supplies the default (ON,SPIE).</p>	ON	Enables the Language Environment condition handler.	NOSPIE	Specifies that the Language Environment will handle program interrupts and abends via an ESTAE, that is, the Natural abend handler will receive control to handle program interrupts and abends.
ON	Enables the Language Environment condition handler.				
NOSPIE	Specifies that the Language Environment will handle program interrupts and abends via an ESTAE, that is, the Natural abend handler will receive control to handle program interrupts and abends.				
TERMTHDACT(UADUMP)	Defines the level of information that is produced in case of an abend. The option UADUMP generates a Language Environment CEEDUMP and system dump of the user address space. The CEEDUMP does not contain the Natural relevant storage areas. IBM supplies the default (TRACE).				
ENVAR(TZ=...)	<p>The ENVAR option enables you to set UNIX environment variables. The only environment variable applicable for the Natural Web I/O Interface server is TZ (time zone). This variable allows you to adjust the timestamp within the Natural Web I/O Interface server's trace file to your local time.</p> <p>Example:</p>				

Parameter	Definition
	ENVAR(TZ=CET-1DST) CET - 1 hour daylight saving time

You can set the z/OS language environment parameters:

- With the `PARM` parameter specified in the `EXEC` card of the Natural Web I/O Interface server startup job. The length of the options is limited by the maximum length of the `PARM` parameter.
- Assemble an LE/370 runtime option module `CEEUOPT` and link it to the Natural Web I/O Interface server load module.

External Security Configuration

If you configure the Web I/O Interface server to impersonate the Web I/O Interface clients in the server (Web I/O Interface server configuration parameter `SECURITY_MODE=IMPERSONATE` or `IMPERSONATE_LOCAL`), the Web I/O Interface server must run “program-controlled”. Under RACF, the following definitions are required for the Web I/O Interface server:

- The resource `BPX.SERVER` must be defined and the Web I/O Interface server account must have `READ` access to this resource.
- The `LOAD` data sets defined in the Web I/O Interface server startup job definition must be defined to the program class `"**"`.

```
ralt program ** addmem('natural load library') uacc(read)
ralt program ** addmem('NWO load library'//NOPADCHK) uacc(read)
ralt program ** addmem('user load library'//NOPADCHK) uacc(read)
```

- `SETR WHEN(PROGRAM) REFRESH`

Additionally, each client connecting to the server must be defined in RACF and must be granted to use the z/OS Unix System Services.

SSL Support

- [SSL over AT-TLS](#)
- [Maintenance of Certificates under z/OS](#)
- [Using RACF Key Rings](#)
- [Using Key Databases](#)
- [How to configure TCP/IP for AT-TLS?](#)
- [How to Verify AT-TLS Configuration?](#)
- [Frequently Asked Questions](#)

- [Generation of a Natural Web I/O Interface Server Certificate](#)

SSL over AT-TLS

SSL support for the Natural Web I/O Interface server is based on the z/OS Communication Server component AT-TLS (Application Transparent-Transport Layer Security).

AT-TLS provides TLS/SSL encryption as a configurable service for sockets applications. It is realized as an additional layer on top of the TCP/IP protocol stack, which exploits the SSL functionality in nearly or even fully transparent mode to sockets applications. AT-TLS offers three modes of operation. See *z/OS Communications Server, IP Programmer's Guide and Reference*, Version 1, Release 9, Chapter 15, IBM manual SC31-8787-09.

These modes are:

■ Basic

The sockets application runs without modification in transparent mode, unaware of performing encrypted communication via AT-TLS. Thus legacy applications can run in secured mode without source code modification.

■ Aware

The application is aware of running in secured mode and is able to query TLS status information.

■ Controlling

The sockets application is aware of AT-TLS and controls the use of AT-TLS encryption services itself. This means, the application is able to switch between secured and non secured communication.

Natural Web I/O Interface server uses the Basic mode for its SSL implementation. That is, a server configured as SSL server rejects requests from non-secured clients.

Maintenance of Certificates under z/OS

Certificates, which are to be used with AT-TLS, can be maintained in two ways under z/OS. They are stored either in RACF key rings or in key databases, which are located in the z/OS UNIX file system. Which of these proceedings actually applies is defined in the AT-TLS Policy Agent Configuration file for the z/OS TCP/IP stack, which is used by the Natural HTTPS client.

IBM delivers a set of commonly used CA root certificates with each z/OS system delivery. If key rings are going to be used to hold server certificates, those root certificates must be manually imported into the key rings by the system administrator. If IBM delivers newer replacements for expired root certificates, all affected key rings have to be updated accordingly.

Unlike key rings, key databases contain the current set of root certificates automatically after they have been newly created. However, the need for maintaining always the latest set of root certificates applies to the key database alternative as well.

Using RACF Key Rings

In RACF, digital certificates are stored in so-called key rings. The RACF command `RACDCERT` is used to create and maintain key rings and certificates, which are contained in those key rings.

See *z/OS Security Server RACF Security Administrator's Guide*, IBM manual SA22-7683-11, and *z/OS Security Server RACF Command Language Reference*, IBM manual SA22-7687-11.

Using Key Databases

Alternatively to RACF, certificates can be kept in key databases, which reside in the z/OS UNIX services file system. For the creation and maintenance of key databases, the `GSKKYMANT` utility has to be used.

See *z/OS Cryptographic Services PKI Services Guide and Reference*, IBM manual SA22-7693-10.

How to configure TCP/IP for AT-TLS?

Proceed as follows:

1. In the TCP/IP configuration file, set the option `TTLS` in the `TCPCONFIG` statement.
2. Configure and start the AT-TLS Policy Agent. This agent is called by TCP/IP on each new TCP connection to check if the connection is SSL.
3. Create the Policy Agent file containing the AT-TLS rules. The Policy Agent file contains the rules to stipulate which connection is SSL.

See also *z/OS Communications Server: IP Configuration Guide*, Chapter 18 *Application Transparent Transport Layer Security (AT-TLS) data protection*.

The Sample Policy Agent file defines the server with the job name starting with `NWODEV` and listening at port 4843 to use SSL.

The sample expects the certificate database on the HFS file `/u/admin/CERT.kdb`.

```
TTLSRule ConnRule01~1
{
  LocalAddrSetRef      addr1
  RemoteAddrSetRef     addr1
  LocalPortRangeRef   portR1
  RemotePortRangeRef  portR2
  Jobname              NWODEV*
  Direction            Inbound
  Priority              255
}
```

```

    TTLSGroupActionRef      gAct1~NW0_Server
    TTLSEnvironmentActionRef eAct1~NW0_Server
    TTLSConnectionActionRef cAct1~NW0_Server
}
TTLSGroupAction           gAct1~NW0_Server
{
    TTLSEnabled            0n
}
TTLSEnvironmentAction     eAct1~NW0_Server
{
    HandshakeRole          Server
    EnvironmentUserInstance 0
    TTLSKeyringParmsRef    keyR1
}
TTLSConnectionAction      cAct1~NW0_Server
{
    HandshakeRole          Server
    TTLSCipherParmsRef     cipher1~AT-TLS__Silver
    TTLSConnectionAdvancedParmsRef cAdv1~NW0_Server
}
TTLSConnectionAdvancedParms cAdv1~NW0_Server
{
    CertificateLabel       NDV_TEST_CERT
}
TTLSKeyringParms          keyR1
{
    Keyring                 /u/admin/CERT.kdb
    KeyringStashFile        /u/admin/CERT.sth
}
TTLSCipherParms           cipher1~AT-TLS__Silver
{
    V3CipherSuites         TLS_RSA_WITH_DES_CBC_SHA
    V3CipherSuites         TLS_RSA_WITH_3DES_EDE_CBC_SHA
    V3CipherSuites         TLS_RSA_WITH_AES_128_CBC_SHA
}
IpAddrSet                 addr1
{
    Prefix                  0.0.0.0/0
}
PortRange                  portR1
{
    Port                    4843
}
PortRange                  portR2
{
    Port                    1024-65535
}

```

How to Verify AT-TLS Configuration?

Check Policy-Agent job output JESMSG LG for:

```
EZZ8771I PAGENT CONFIG POLICY PROCESSING COMPLETE FOR <your TCP/IP address space>: ←
TTLS
```

This message indicates a successful initialization.

Check Policy-Agent job output JESMSG LG for:

```
EZZ8438I PAGENT POLICY DEFINITIONS CONTAIN ERRORS FOR <your TCP/IP address space>: ←
TTLS
```

This message indicates errors in the configuration file. Check the *syslog.log* file for further information.

Does the configuration rule cover the server?

Try to connect the server and check *syslog.log* for:

```
EZD1281I TTLS Map CONNID: 00018BED LOCAL: 10.20.91.61..4843 REMOTE: ←
10.20.160.47..4889 JOBNAME: NWODEVvr USERID: NWOSRV TYPE: InBound STATUS: Enabled ←
RULE: ConnRule01~1 ACTIONS: gAct1 eAct1~NWO_Server cAct1~NWO_Server
```

The above entry indicates that the connection to Port 4843 is SSL enabled.

Frequently Asked Questions

Is there more information about problem determination?

See also *z/OS V1R8.0 Comm Svr: IP Diagnosis Guide: 3.23, Chapter 29 Diagnosing Application Transparent Transport Layer Security (AT-TLS)*

How to switch on P-agent trace?

See *Comm Svr: IP Configuration Reference, Chapter 20 Syslog daemon and Comm Svr: IP Configuration Guide, Chapter 1.5.1 Configuring the syslog daemon (syslogd)*

Error at connection establishment

Find return code RC and corresponding GSK_ function name in P-agent trace.

See *System SSL Programming* and locate the RC in Chapter 12.1 *SSL Function Return Codes*.

Sample trace with `trace=255`:

```
EZD1281I TTLS Map CONNID: 00002909 LOCAL: 10.20.91.61..1751 REMOTE: ←
10.20.91.117..443 JOBNAME: KSP USERID: KSP TYPE: OutBound STATUS: A
EZD1283I TTLS Event GRPID: 00000003 ENVID: 00000000 CONNID: 00002909 RC: 0 ←
Connection Init
EZD1282I TTLS Start GRPID: 00000003 ENVID: 00000002 CONNID: 00002909 Initial ←
Handshake ACTIONS: gAct1 eAct1 AllUsersAsClient HS-Client
EZD1284I TTLS Flow GRPID: 00000003 ENVID: 00000002 CONNID: 00002909 RC: 0 Call ←
GSK_SECURE_SOCKET_OPEN - 7EE4F718
EZD1284I TTLS Flow GRPID: 00000003 ENVID: 00000002 CONNID: 00002909 RC: 0 Set ←
GSK_SESSION_TYPE - CLIENT
EZD1284I TTLS Flow GRPID: 00000003 ENVID: 00000002 CONNID: 00002909 RC: 0 Set ←
GSK_V3_CIPHER_SPECS - 090A2F
EZD1284I TTLS Flow GRPID: 00000003 ENVID: 00000002 CONNID: 00002909 RC: 0 Set ←
GSK_FD - 00002909
EZD1284I TTLS Flow GRPID: 00000003 ENVID: 00000002 CONNID: 00002909 RC: 0 Set ←
GSK_USER_DATA - 7EEE9B50
EZD1284I TTLS Flow GRPID: 00000003 ENVID: 00000002 CONNID: 00002909 RC: 435 Call ←
GSK_SECURE_SOCKET_INIT - 7EE4F718
EZD1283I TTLS Event GRPID: 00000003 ENVID: 00000002 CONNID: 00002909 RC: 435 ←
Initial Handshake 00000000 7EEE8118
EZD1286I TTLS Error GRPID: 00000003 ENVID: 00000002 CONNID: 00002909 JOBNAME: KSP ←
USERID: KSP RULE: ConnRule01 RC: 435 Initial Handshake
EZD1283I TTLS Event GRPID: 00000003 ENVID: 00000002 CONNID: 00002909 RC: 0 ←
Connection Close 00000000 7EEE8118 ←
```

Generation of a Natural Web I/O Interface Server Certificate

Under z/OS, SSL certificates can be produced with the UNIX System Services utility GSKKYPAN. The following steps have to be executed for the production of a new certificate, which is to be used for the SSL secured communication between Natural Web I/O Interface client and server:

1. Start a shell session out of TSO or connect via telnet to the z/OS UNIX shell.
2. Start GSKKYPAN.
3. Create a new – or open an existing key database.
4. Create a self-signed certificate, for example, of type “User or server certificate with 2048-bit RSA key”.
5. Export the certificate to a (HFS) file. Choose Base64 ASN.1 DER as export format.

This generated file can now be copied to the Natural Web I/O Interface client(s) using FTP with ASCII transfer format. On the client side, the received file should be stored with the file name suffix .CER. The certificate can now be used by the Natural Web I/O Interface client.

If certificates are kept in a RACF key ring, the generated certificate has to be imported into the appropriate key ring using the RADCERT command.

Certificates, which are produced on a different platform, for example, on a Windows PC, can be imported into a RACF key ring or into a key database as well.

Detailed information about the use of the `GSKKYM` utility can be found in the IBM Communications server documentation, e.g in the following manuals:

z/OS Communications Server IP Configuration Guide Version 1 Release 2 (IBM manual SC31-8775-01

or

z/OS Communications Server Cryptographic Services System Secure Sockets Layer Programming (IBM manual SC24-5901-04).

For the generation of certificates under Windows, a free downloadable utility named Ikeyman is available on several websites. Ikeyman is an IBM product as well and maps the functionality of `GSKKYM` to the Windows platform.

Configuration Requirements for z/VSE

The following topics are covered below:

- [SMARTS SYSPARM Parameters](#)
- [SYSPARM Example for the Natural Web I/O Interface Server](#)

SMARTS SYSPARM Parameters

The Natural Web I/O Interface server requires the following SMARTS SYSPARM parameters:

Parameter	Definition	
RESIDENTPAGE	The following members must be defined in the SMARTS resident area: NATRNWO, NATMONI, NATDSSEC, Natural front-end (NCFNUC) and Natural nucleus (if you run using a split nucleus).	
SECSYS	The installed external security system (RACF ACF2 TOPSECRET).	
SERVER	The following SERVER definitions are required for the Natural Web I/O Interface server:	
	SERVER=(OPERATOR , TLINOPER , TLSPOPER)	The Operator Communications Server.
	SERVER=(POSIX , PAENKERN)	The POSIX Server.
	The Natural local buffer pool definition.	For details, refer to the Natural Com-plete interface documentation.
CDI_DRIVER	CDI_DRIVER=('TCPIP , PAACSOCK , MINQ=10 , MAXQ=20 ')	The SMARTS TCP/IP Socket Driver for Connectivity Systems TCP/IP stack on z/VSE. MINQ/MAXQ define the

Parameter	Definition	
		number of TcpIP listener tasks.
THSIZEABOVE	THSIZEABOVE=1024	The storage above 16 MB that is available for each Natural Web I/O Interface server subtask. This size must be large enough to keep the Natural tread, heap and stack of the Natural Web I/O Interface server subtasks. A certain headroom (20% or more, depending on your environment) is recommended. If the Natural Web I/O Interface server initialization fails with NAT9915 GETMAIN for thread storage failed, this parameter must be increased.
ADASVC	ADASVC=nnn	The Adabas SVC number of your Adabas installation.

You can set the SMARTS SYSPARM parameters in the file SMARTS.CONFIG which must reside on one of your accessed disk.

SYSPARM Example for the Natural Web I/O Interface Server

For z/VSE:

```

* ----- ADABAS PARMS -----*
ADACALLS=20 CALLS BEFORE ROLL
ADASVC=47 ADABAS SVC NUMBER
* ----- BUFFERPOOL PARMS -----*
BUFFERPOOL=(064,030,20,ANY)
BUFFERPOOL=(128,064,64,ANY)
BUFFERPOOL=(256,010,10,ANY)
BUFFERPOOL=(512,032,10,ANY)
BUFFERPOOL=(1K,032,32,ANY)
BUFFERPOOL=(6K,005,02,ANY)
BUFFERPOOL=(8K,016,16,ANY)
* ----- ROLLING PARMS -----*
ROLL-BUFFERPOOL=(048K,04,04,DS) ESA DATA SPACE
ROLL-BUFFERPOOL=(064K,04,04,DS) ESA DATA SPACE
ROLL-BUFFERPOOL=(128K,04,04,DS) ESA DATA SPACE
ROLL-BUFFERPOOL=(256K,04,04,DS) ESA DATA SPACE
    
```

```

ROLL-BUFFERPOOL=(800K,02,02,DS) ESA DATA SPACE
*
* ----- NWO Server to launch at startup -----*
* STARTUPPGM='NATRNWO NWOS1'
*
*
TASK-GROUP=(DEFAULT,6)
THREAD-GROUP=(DEFAULT,(DEFAULT,252,06,15,28,N))
*
THSIZEABOVE=1024
*
SERVER=(NCFNATvr,NCFNATvr) NATvr BUFFER POOL
*
CDI_DRIVER=( 'TCPIP,PAACSOCK,MINQ=10,MAXQ=20' )
*
RESIDENTPAGE=NATRNWO
RESIDENTPAGE=NWONCFvr
RESIDENTPAGE=NATNUCvr
RESIDENTPAGE=NATMONI

```

Where *vr* is the current product version and release number.

Where *vr* or *vrs* is the current product version and release number.

SMARTS Configuration File for BS2000

The Natural Web I/O Interface server reads its configuration parameters from a file which resides as an S-type member in the `NWO-JOBS` library.



Notes:

1. Translation tables are used to convert characters when sending or receiving data to or from a host while working with a terminal emulation, see *Adapting Translation Tables for Natural Web I/O Interface Server under BS2000* in the Natural for Windows (Natural Studio) documentation.
2. The SMARTS configuration is contained as an S-type member which resides in the `NWO-JOBLIB`. It has to be passed to the system in the startup procedure parameter `NWO-SYSPARM`.

The following topics are covered below:

- [SMARTS SYSPARM Parameters](#)
- [SMARTS Server Environment Configuration Parameters](#)

▪ SMARTS Sample Configuration

SMARTS SYSPARM Parameters

The Natural Web I/O Interface server requires the following SMARTS SYSPARM parameters:

Parameter	Definition	
RESIDENTPAGE	This parameter specifies one or more programs which are loaded into the SMARTS load-pool during system startup. The following members must be defined in the SMARTS resident area: NATRNWO, NATMONI, Natural front-end (NCFNUC) and Natural nucleus.	
SERVER	The following SERVER definitions are required for the Natural Web I/O Interface server:	
	SERVER=(POSIX, PAENKERN, CONF=PAANCONF)	The POSIX server.
	SERVER=(NCFNATnn, NCFNATnn)	The Natural buffer pool server. The size of the various Natural buffer pools is configured in the parameter string of the NWO configuration parameter SESSION_PARAMETER .
	SERVER=(NWOSEV, PAENAPPS, NATRNWO, 'SERVERID')	The NWO server <i>SERVERID</i> .
CDI_DRIVER	The CDI (Communication Driver Interface) drivers specify the various I/O routines and/or tasks of SMARTS as well as the physical access paths for sequential files and PFS-container files. If necessary for technical reasons, these routines are implemented as separate tasks (socket communication, physical file I/O, etc.). The others are executed in the oc-task (main task) or in one of the worker-tasks (e.g. console I/O).	
	CDI_DRIVER=('console, PAANCNIO')	The console server.
	CDI_DRIVER=('file, PA2SFIO, SIOTSK=JOBNAME')	The file I/O task <i>JOBNAME</i> (SYSOUT, traces, etc.).
	CDI_DRIVER=('tcpip, PAAZSOCK, SOCKETSK=JOBNAME')	The socket task <i>JOBNAME</i> .
	CDI_DRIVER=('CIO, PAAQBIO, PFSTSK=')	The PFS task <i>JOBNAME</i> running the CDI driver CIO.
	CDI_DRIVER=('PFS, PAANPFS, LRECL=4096, CONTAINER=CIO:AAA/BBB/CC')	Defines the container file \$USERID.AAA.BBB.CC for the Portable File System (PFS); see also <i>Using SMARTS PFS under BS2000</i> .
MOUNT_FS	MOUNT_FS=('PFS://', '/') Mapping of (PFS) file names to the appropriate physical BS2000 container file; see also <i>Using SMARTS PFS under BS2000</i> . This parameter maps (POSIX) filenames to a physical BS2000 container file for the specified PFS.	

Parameter	Definition	
DATA-MAXIMUM	DATA-MAXIMUM= <i>nnnn</i>	Maximum possible CMP size for DATA in MB (over all sessions). This parameter limits the maximum size of the data common memory pool. At SMARTS startup, the data CMP is enabled with the specified size, however, real storage allocations within the pool are done on demand only in the actually requested size (For more information, please refer to the BS2000 executive macros manual: <i>ENAMP / REQMP macros</i>).
CODE-MAXIMUM	CODE-MAXIMUM= <i>nn</i>	Maximum possible CMP size for (shared) Code in MB. This parameter limits the total size of all routines loaded, such as the SMARTS kernel itself, NWO and the Natural nucleus.
THREAD-GROUP	THREAD-GROUP=(DEFAULT, (DEFAULT, 0, <i>n</i>))	Establishes the default thread-group with <i>n</i> threads. Only the <code>num</code> subparameter is of importance. This value defines the number of sessions that can be active and kept in storage in parallel; see also the WORKLOAD and TASK-GROUP parameters.
TASK-GROUP	TASK-GROUP=(DEFAULT, <i>n</i>)	Specifies the number of active worker-tasks. The value <i>n</i> should correspond to the specified number of threads. Only the <code>num</code> subparameter is of importance. This value defines the number of worker-tasks (BS2000 tasks) that can execute NWO Natural sessions in parallel. Evidently, this parameter value should correspond to the number of available threads as defined by the THREAD-GROUP definition!
THSIZEABOVE	THSIZEABOVE= <i>nnnn</i>	Specifies the SMARTS thread size (in KB) which has to contain all buffers of one Natural session. This parameter specifies the size of the SMARTS threads which have to contain the Natural thread

Parameter	Definition	
		(NTHSIZE subparameter of Natural profile parameter COMP or macro NTCOMP). A certain headroom (20% or more, depending on your environment) is recommended. If the Natural Web I/O Interface server initialization fails with NAT9915 GETMAIN for thread storage failed, this parameter must be increased.
WORKLOAD-AVERAGE	WORKLOAD-AVERAGE= <i>n</i>	The average number of sessions active in parallel. This parameter defines the expected average number of Natural sessions (not users, since one user can start more than one session!) which are to be executed by the server. This parameter must not be confused with the THREAD-GROUP parameter, because it represents the sum of all active and inactive sessions.
WORKLOAD-MAXIMUM	WORKLOAD-MAXIMUM= <i>nn</i>	The maximum possible number of sessions active in parallel. This parameter defines the maximum possible number of Natural sessions.
ADASVC	ADASVC=249	The Adabas SVC number must not be changed.

 **Note:** The parameter values printed in italics (*SERVERID*, *JOBNAME*, *PFS*, *nnn*, etc.) are to be specified by the user.

SMARTS Server Environment Configuration Parameters

The following general parameter descriptions are adapted excerpts from the original SMARTS documentation. The text is provided for background information only. Therefore, not all of the information contained therein applies to the Natural Web I/O Interface server under SMARTS on BS2000.

- ADASVC
- RESIDENTPAGE
- SERVER
- TASK-GROUP
- THREAD-GROUP
- THSIZEABOVE

- WORKLOAD-AVERAGE, WORKLOAD-MAXIMUM

ADASVC

This parameter is for internal use only. Do not change the Adabas SVC number.

RESIDENTPAGE

Sysparm	Use	Possible Values	Default
RESIDENTPAGE	The name of a program to be loaded and made resident when SMARTS is initialized.	<i>program-name</i>	none

All modules are assumed to be reentrant, and are loaded into the address space automatically at first reference.

The program must be fully reentrant. If it is not marked reentrant, a warning message is issued on the operator's console at SMARTS initialization time.

The program must reside in the APS-LIB or in the NWO-MOD library of the SMARTS initialization procedure.

SERVER

Sysparm	Use	Possible Values	Default
SERVER	Information that identifies a server to SMARTS.	<i>server-information</i>	none

The *server-information* has the format

(serv-id,init-mod,p1,p2,pn)

- where

<i>serv-id</i>	is the ID for this server (1-8 characters).
<i>init-mod</i>	is the name of the initialization/termination routine.
<i>p1,p2,pn</i>	are parameters to be passed to the initialization routines.

Specifying the SERVER parameter causes SMARTS to build a server directory entry (SDE) for the specified server and pass control to the initialization routine specified to initialize the server.

TASK-GROUP

Sysparm	Use	Possible Values	Default
TASK-GROUP	A group comprising one or more tasks, available when SMARTS is started.	(<i>grp, num, priority, maxq</i>)	(DEFAULT, <i>num</i>)

- where

<i>grp</i>	Required. The name of the task group being defined. The default task group is DEFAULT.
<i>num</i>	Required. The number of tasks to be allocated in the task group. The default number of tasks is calculated dynamically based on the size of the installation.
<i>priority</i>	Not supported under BS2000.
<i>maxq</i>	The maximum number of TIBs (default 16) expected on this task group's work queue at the same time. Under normal circumstances, the default should be adequate. When there are problems and it is not, a secondary Last In First Out (LIFO) queue is used so that no work is lost. The normal queue is First In First Out (FIFO), which ensures that work is done in the order in which it is received. This is why the LIFO queue is only used as a secondary backup.



Important: For SMARTS, only the TASK-GROUP DEFAULT is available. Software AG strongly recommends that you use the default definition. If other products running on SMARTS require changes to the defaults or allow the definition of their own TASK-GROUPS, that will be indicated in the relevant documentation.



Notes:

1. A maximum of 8 task groups may be defined.
2. Task-group names are converted to uppercase prior to being processed; therefore, a parameter entered in lowercase is treated as, and appears in, uppercase letters.
3. If more than one specification appears for a task group, the last valid specification is used.
4. The task group DEFAULT must always exist in the system. If it is not explicitly defined by the installation, the task group is built by the system with the default values.
5. Note that the total number of tasks to be attached must not exceed the MAXTASKS specification. This is not checked until the task groups are being built; however, exceeding the value leads to task-group allocation errors as against parameter errors.

THREAD-GROUP

Sysparm	Use	Possible Values	Default
THREAD-GROUP	A thread group containing one or more thread subgroups and threads, to be available when SMARTS is started.	see below	see below

The format for the value is

```
(grp,(sub,size,num,cpu,real,key),...,(sub,size,num,cpu,real,key))
```

- where

<i>grp</i>	Required. The name of the task group being defined.
<i>sub</i>	The name of the subgroup being defined. If a subgroup name is specified more than once for the same group, the last valid specification is used when parameter processing has been completed.
<i>size</i>	Required. The amount of storage in kilobytes to be allocated for each thread below the line. A valid value is between 8 kilobytes and 1 megabyte.
<i>num</i>	The number of threads to be allocated in the thread subgroup. The value must be greater than 1 and less than 4096. Generally, this subparameter is required. It can be omitted for one (and only one) thread subgroup in the address space; in this case, the number of threads to be allocated for the subgroup is calculated dynamically by SMARTS, based on the size of the installation.
<i>cpu</i>	The CPU time in seconds (default 0.00) that a user program can use in the thread subgroup for one SMARTS transaction. This value may be entered as an integer or to a level of hundredths of seconds using the <i>n.nn</i> format. If a 0 is provided as the CPUTIME for a thread subgroup, no CPU limit is placed on programs running in the associated threads.
<i>real</i>	The wait time in seconds (default 0.00) for the thread subgroup, after which a message is issued to the console if the user program has not given up control of its thread. This value may be entered as an integer or to a level of hundredths of seconds using the <i>n.nn</i> format. If 0 is specified, elapsed time is not checked for the thread subgroup.
<i>key</i>	The key (default M) in which the threads within the subgroups are allocated: M - The thread keys are a mixture of user keys excluding the key in which SMARTS is running. N - No storage protection is implemented and all threads run in the same key as SMARTS.



Note: The user may also specify a value in the range 1 to 15 inclusive to allocate a thread to that key explicitly.

The default value is

```
THREAD-GROUP=(DEFAULT,($DEFAULT,8,num))
```

- where *num* is calculated dynamically based on the size of the installation.



Important: For SMARTS, only `THREAD-GROUP DEFAULT` is available. Software AG strongly recommends that you use the default definition. If other products running on SMARTS re-

quire changes to the defaults or allow the definition of their own `THREAD-GROUPS`, that will be indicated in the relevant documentation.



Notes:

1. A maximum of 8 thread groups may be defined.
2. A maximum of 8 subgroups can be allocated per thread group. The subgroups may be defined on one line or on different lines. When a second `THREAD-GROUP` statement is used, the same group name must be specified to relate the subgroup entries.
3. Thread group and subgroup names are converted to uppercase prior to being processed; therefore, a parameter entered in lowercase is treated as, and appears in, uppercase letters.
4. If more than one specification appears for a thread subgroup of a thread group, the last valid specification is used.
5. The amount of storage specified on the `THSIZEABOVE` parameter is allocated above the line for each thread defined as a result of the `THREAD-GROUP` parameter.
6. The thread group `DEFAULT` must always exist in the system. If it is not explicitly defined by the installation, the thread group is built by the system with the default values. If it is defined, the system ensures that a thread subgroup with a thread size at least as large as that required by `DEFAULT` is allocated. If not, the system allocates an additional subgroup for the group. If too many subgroups have been defined, the last one defined is overwritten to allow for the default specification.
7. The keyword data is processed from left to right. If more than one thread subgroup is defined on one line and the line contains an error, even if an error message is issued for the line, any subgroups processed up to the error are still accepted. That is to say, if the first subgroup is correct and the second is not, an error message is issued but the first thread subgroup is defined while the second and subsequent specifications in the same statement are ignored.

THSIZEABOVE

Sysparm	Use	Possible Values	Default
THSIZEABOVE	The amount of storage above the 16 MB line, in multiples of 1024 bytes, to be allocated to each thread.	<i>n</i>	1024

WORKLOAD-AVERAGE, WORKLOAD-MAXIMUM

The WORKLOAD-AVERAGE parameter specifies a normal workload value, and the WORKLOAD-MAXIMUM parameter specifies a maximum workload value. SMARTS uses these values together with the region sizes above and below the 16 MB line to configure itself.

These parameters are not required, but tuning them may improve performance.

Sysparm	Use	Possible Values	Default
WORKLOAD-AVERAGE	The average number of parallel processes expected to run in SMARTS.	1-32767	WORKLOAD-MAXIMUM divided by 4.
WORKLOAD-MAXIMUM	The maximum number of parallel processes expected to run in SMARTS.	1-32767	50 if WORKLOAD-AVERAGE is not specified, otherwise WORKLOAD-AVERAGE times 4.

SMARTS Sample Configuration

```

ADASVC=249
DATA-MAXIMUM=160
CODE-MAXIMUM=34
THSIZEABOVE=4096
THREAD-GROUP=(DEFAULT,(DEFAULT,0,4))
TASK-GROUP=(DEFAULT,2)
WORKLOAD-AVERAGE=8
WORKLOAD-MAXIMUM=40
RESIDENTPAGE=NATSOCK
RESIDENTPAGE=NATMONI
RESIDENTPAGE=NATRNO
RESIDENTPAGE=NCFSERV
RESIDENTPAGE=NvrLPRRB
*****SERVERS *****
SERVER=(POSIX,PAENKERN,CONF=PAANCONF)
SERVER=(NCFNATvr,NCFNATvr,1,2048,2,100,4,2048)
SERVER=(NWNATvr,PAENAPPS,NATRNO,'NWOS01')
*****CDI-DRIVERS*****
CDI_DRIVER=('file,PA2SFIO,SIOTSK=HSFSIO')
CDI_DRIVER=('console,PAANCNIO')
CDI_DRIVER=('tcpip,PAZSOCK,SOCKTSK=HSSOCTA2,TRACE=1')
CDI_DRIVER=('CIO,PAQBIO,PFSTSK=HSPFS,TRACE=N')
CDI_DRIVER=('PFS,PAANPFS,LRECL=4096,CONTAINER=CIO:NWO/ROOT/SERVER1/')
MOUNT_FS=('PFS://','/')
    
```

Web I/O Interface Server Configuration File for z/OS and z/VSE

A configuration file is allocated to the name `<serverid>C` (for example, `NW0S1C`) or `STGCONFIG` alternatively.

The configuration file is a text file located on a data set or on an HFS file under z/OS and a librarian member under z/VSE.

The configuration file contains the server configuration parameters in the form of a `keyword=value` syntax. In addition, it may contain comments whose beginning is marked with a hash symbol (#).

See also the [Web I/O Interface Server Configuration File Example](#) shown below.

Web I/O Interface Server Configuration Parameters

The following Web I/O Interface server configuration parameters are available:

- `COMPATIBILITY_MODE`
- `FORCE_IPV4`
- `FRONTEND_NAME`
- `FRONTEND_OPTIONS`
- `FRONTEND_PARAMETER`
- `HANDLE_ABEND`
- `HOST_NAME`
- `HTPMON_ADMIN_PSW`
- `HTPMON_PORT`
- `HOST_NAME`
- `IGNORE_PRESENT_SERVER`
- `INITIAL_USERID`
- `KEEP_TCB`
- `O4I`
- `PASSWORD_MIXEDCASE`
- `PORT_NUMBER`
- `SECURITY_MODE`
- `SESSION_PARAMETER`
- `SESSION_TIMEOUT`
- `THREAD_NUMBER`
- `THREAD_SIZE`
- `TRACE_FILTER`
- `TRACE_LEVEL`

- `UPPERCASE_SYSTEMMESSAGES`

COMPATIBILITY_MODE

The current version of NWO presumes to run with the most recent version of Natural. An error `NAT7729 NWO and Natural version do not agree` is issued when running with older Natural versions. This is because NWO must negotiate a subset of functionality with the client at a time when the involved Natural version is not already known.

If you want to run NWO with a previous version of Natural, you can set this parameter to `YES`. It is recommended that you leave this parameter at its default value if you intend to run your NWO with the most recent version of Natural, because in this case `COMPATIBILITY_MODE=YES` would unnecessarily limit the functionality.

Value	Explanation
YES	Accept also older versions of Natural. Results in a limitation of the functionality documented with the most recent version.
NO	Presume to run with the most recent version of Natural. This is the default value.

Example:

```
COMPATIBILITY_MODE=YES
```

FORCE_IPV4

This parameter is only available with Natural Web I/O Interface Version 8.3 or above and applies to z/OS and z/VSE.

This parameter allows you to enforce the use of communication method `IPV4`.

Value	Explanation
YES	Enforce the use of communication method <code>IPV4</code> .
NO	First try communication method <code>IPV6</code> . If this fails give an error message and use communication method <code>IPV4</code> . This is the default value.

FRONTEND_NAME

This configuration parameter specifies the name of the Natural front-end to be used to start a Natural session. The front-end resides on a PDS member.

Value	Explanation
<i>frontend-name</i>	Natural front-end to be used. Maximum length: 8 characters.

No default value is provided.

Example:

```
FRONTEND_NAME=NATvrsv
```

FRONTEND_OPTIONS

This configuration parameter applies to z/OS only.

The values of this configuration parameter may be used to specify additional options for the Natural front-end.

Value	Explanation
01	Do not use the Roll Server. This is the default value.
02	Clean up roll file at server termination.
04	Write GTF trace.
08	Write ETRACE.
10	Front-end automatic termination.
20	Write console information.

You may combine the above options as desired in that you add their values and set the result as shown in the example below.

Example:

```
FRONTEND_OPTIONS=07
```

The setting in this example enables the Options 01, 02 and 04.

FRONTEND_PARAMETER

This configuration parameter applies to z/OS only.

This optional configuration parameter contains additional Natural front-end parameters as specified in the Startup Parameter Area.

Value	Explanation
<i>parameter-name</i>	<p>You can define multiple parameters. Each parameter specification is a pair of 8-character strings, the first containing the parameter keyword and the second the parameter value, for example:</p> <pre>FRONTEND_PARAMETER = 'MSGCLASSX '</pre>

No default value is provided.

For further information, refer to the section *Natural in Batch Mode* in the *Natural Operations for Mainframe* documentation.

Example:

```
FRONTEND_PARAMETER='MSGCLASSX      '
```

The setting in this example specifies that the default output class for CMPRINT is "X".

HANDLE_ABEND

If an abend occurs in the server processing outside the Natural processing the abend is not trapped by the Natural abend handling. For this reason the NWO server has its own abend recovery.

It is recommended that you leave this parameter on its default value in order to limit the impact of an abend to a single user. If you set the value of this parameter to NO, any abend in the server processing terminates the complete server processing. That is, it affects all users running on that server.

Value	Explanation
YES	Trap abends in the server processing, write a snap dump and abort the affected user. This is the default value.
NO	Suspend the server abend handling.

Example:

```
HANDLE_ABEND=NO
```

HOST_NAME

This configuration parameter applies to z/OS and z/VSE.

This optional configuration parameter is necessary only if the server host supports multiple TCP/IP stacks.

Value	Explanation
<i>host-name</i>	If HOST_NAME is specified, the server listens on the particular stack specified by HOST_NAME, otherwise the server listens on all stacks.

No default value is provided.

Example:

```
HOST_NAME=node1
```

or

```
HOST_NAME=157.189.160.55
```

HTPMON_ADMIN_PSW

This configuration parameter applies to z/OS, z/VSE and BS2000.

This configuration parameter defines the password required for some monitor activities (e.g. Terminate Server) performed by the HTML Monitor Client.

Value	Explanation
any character string	The password to be entered at the HTML Monitor Client for some monitor activities.

No default value is provided.

Example:

```
HTPMON_ADMIN_PSW=GHAU129B
```

HTPMON_PORT

This configuration parameter applies to z/OS, z/VSE and BS2000.

A Web I/O Interface server can be configured to host an HTTP monitor task which serves the HTML Monitor Client running in a web browser. It is not required to run this monitor task on each server. A single task allows you to monitor all servers running at one node.

This configuration parameter defines the TCP/IP port number under which the server monitor task can be connected from a web browser.

Value	Explanation
1 - 65535	TCP/IP port number.

No default value is provided.

Example:

```
HTPMON_PORT=3141
```

HOST_NAME

This configuration parameter applies to z/VSE and BS2000.

This configuration parameter defines the host name of the Web I/O Interface server.

IGNORE_PRESENT_SERVER

This configuration parameter applies to z/OS and z/VSE in conjunction with the Web I/O Interface server CICS Adapter.

A Web I/O Interface (NWO) server allocates a so-called "server environment" which contains the server dependent common resources.

This environment is unique for each NWO server and relates to the server name. If an NWO server with Web I/O Interface Server CICS Adapter ends abnormally, it might leave a stuck NWO server environment within the CICS region. This causes that a restart of the server fails with error message NAT9913.

If you start an NWO server with `IGNORE_PRESENT_SERVER=YES`, it might damage an already running server which is using the same server name and the same CICS region.

Value	Explanation
YES	Terminate existing CICS server environment.
NO	Abort server initialization if a CICS server environment already exist. This is the default value.

Example:

```
IGNORE_PRESENT_SERVER=YES
```

INITIAL_USERID

At server initialization, the Natural Web I/O Interface server creates a temporary Natural session to obtain the properties of the installed Natural environment.

This configuration parameter specifies the user ID to be used for this Natural session.

Value	Explanation
<i>userid</i>	The specified value must not exceed 8 characters, otherwise it is truncated.
STARGATE	This is the default value.

Example:

```
INITIAL_USERID=NWOINITU
```

See also *Web I/O Interface clients must be defined to Natural Security* in the operating-system-specific Natural Web I/O Interface server *Installation* section.

KEEP_TCB

This configuration parameter applies to z/OS only.

By default, the remote Natural session of a mapped environment terminates its TCB whenever you switch the focus within Natural Studio to a different mapped environment. If you toggle the focus back, the remote session is dispatched using a different TCB.

The maximum number of active TCBs is equal to the number of connected clients.

The configuration parameter `KEEP_TCB` specifies whether the remote Natural session should use the same TCB during its entire lifetime. This is required if you use Adabas and the Adabas parameter `ADANAME` is set to `ADAUSER` or if you want to access DB2. It could also be required if you access 3GL programs which need to be executed under the same TCB for successive calls.

Value	Explanation
YES	The remote Natural session uses the same TCB during its entire lifetime.
NO	This is the default value.

Example:

```
KEEP_TCB=YES
```

04I



Note: This parameter is only available with Natural Web I/O Interface Version 8.3 or above.

This parameter allows you to collect server data for Optimize for Infrastructure.

Value	Explanation
YES	Collect server data for Optimize for Infrastructure.
NO	Do not collect server data for Optimize for Infrastructure. This is the default.

Example:

```
04I=YES
```

PASSWORD_MIXEDCASE

This configuration parameter applies to z/OS and z/VSE.

This parameter allows you to define whether passwords specified in the connection dialog are translated into upper case or not.

This parameter does only apply with `SECURITY_MODE=IMPERSONATE`, `IMPERSONATE_LOCAL` or `IMPERSONATE_REMOTE`.

Value	Explanation
YES	Passwords remain in mixed case.
NO	Passwords are translated into upper case. This is the default.

Example:

PASSWORD_MIXEDCASE=YES

PORT_NUMBER

This configuration parameter defines the TCP/IP port number under which the server can be connected.

Value	Explanation
1 - 65535	TCP/IP port number.

No default value is provided.

Example:

PORT_NUMBER=3140

 **Note:** Under BS2000, some port numbers are privileged and reserved for certain system services. Ask your BS2000 system administrator for the port number range available to you.

SECURITY_MODE

This configuration parameter applies to z/OS and z/VSE.

The Natural Web I/O Interface server offers a security concept that also covers the operating system resources. The client credentials are validated at the operating-system-depending security system and the client request is executed under the client's account data.

Using the SECURITY_MODE parameter, you can specify at which rank (in batch mode z/OS, z/VSE or under CICS) you want to impersonate the activities of a Web I/O Interface client.

Value	Explanation
IMPERSONATE_LOCAL	Impersonation is done within the Natural Web I/O Interface server environment. If the session is dispatched in a remote TP environment (e.g. in CICS using the NWO CICS Adapter), it is still executed anonymous. The client must be defined in the security system of the Web I/O Interface server. It is not required to define the client in a remote TP environment. For z/OS, see also External Security Configuration . For z/VSE, see also SYSPARM Parameter SECSYS .
IMPERSONATE_REMOTE	No impersonation is done within the Natural Web I/O Interface server environment. If the session is dispatched in a remote TP environment, the client is impersonated. The client must be defined in the security system of the remote TP environment. See also Web I/O Interface server security exit NATUXRFE and the section Product Interaction in the <i>Web I/O Interface Server CICS Adapter</i> documentation. Note: Please verify the correct installation of NATUXRFE. A Map Environment attempt with a valid user ID and an invalid password should fail with a NAT0873 error.

Value	Explanation
IMPERSONATE	Impersonation is done within the Natural Web I/O Interface server environment and in a remote TP environment. The client must be defined in the security system of the Natural Web I/O Interface server and in the remote TP environment.

No default value is provided.

Example:

```
SECURITY_MODE=IMPERSONATE
```

SESSION_PARAMETER

This optional configuration parameter defines session parameters that precede the parameter string specified in the connection dialog of the Natural Web I/O Interface client.

Value	Explanation
<i>parameter-string</i>	This string may extend across several lines. A + sign at the end of a string line denotes that another line follows.

No default value is provided.

Example 1:

```
SESSION_PARAMETER='NUCNAME=NATNUCvr' +
'PROFILE=(NWOPARM,18006,48),ADAMODE=0,' +
'BPI=(TYPE=NAT,SIZE=6044),BPI=(TYPE=EDIT,SIZE=2048)', +
'BPI=(TYPE=SORT,SIZE=1024)'
```

Example 2:

```
SESSION_PARAMETER=FNAT=(10,930)
```

The setting in the second example defines that every session on this Natural Web I/O Interface server is started with the session parameter FNAT=(10,930) appended to the user-specified parameters or the definitions in the configuration parameter DEFAULT_PROFILE.

SESSION_TIMEOUT

Cancel inactive sessions when the `SESSION_TIMEOUT` parameter is met. Check for sessions inactive longer than *n* minutes once a day at `HH:MM` (24 hours) or every *n* minutes.

The server will not start if an invalid `SESSION_TIMEOUT` parameter is given.

Value	Explanation
<code>hh:mm,n</code> <numeric value greater than 0> or <code>m</code> <numeric value greater than 0>, <code>n</code> <numeric value>0>	If format is <code>hh:mm</code> , check once a day at <code>hh:mm</code> for sessions more than <i>n</i> minutes inactive. or If format is a numeric value, check every <i>m</i> minutes for sessions more than <i>n</i> minutes inactive.

Examples:

```
SESSION_TIMEOUT=19:30,480
```

Every day at 19:30 cancel sessions more than 480 minutes inactive.

```
SESSION_TIMEOUT=360,480
```

Every 360 minutes cancel sessions more than 480 minutes inactive.

THREAD_NUMBER

This configuration parameter applies to z/OS only.

This configuration parameter specifies the number of physical storage threads to be allocated by the Natural front-end, that is, the number of sessions that can be executed in parallel.



Note: This parameter is obsolete when the Natural Web I/O Interface Server CICS Adapter or Natural Web I/O Interface Server IMS Adapter is used.

Value	Explanation
<code>thread-number</code>	Number of physical storage threads to be allocated. Note: This number does not limit the number of sessions within the server, but the number of sessions which can be in execution status concurrently. The number of sessions is limited by the size of the Natural swap medium.
3	This is the default value.

Example:

```
THREAD_NUMBER=5
```

THREAD_SIZE

This configuration parameter applies to z/OS only.

This configuration parameter specifies the size of each physical storage thread which contains the Natural session data at execution time.



Note: This parameter is obsolete when the Natural Web I/O Interface Server CICS Adapter or Natural Web I/O Interface Server IMS Adapter is used.

Value	Explanation
<i>thread-size</i>	Size (in KB) of each physical storage thread.
500	This is the default value.

Example:

```
THREAD_SIZE=800
```

TRACE_FILTER

This optional configuration parameter enables you to restrict the trace by a logical filter in order to reduce the volume of the server trace output, for example:

```
TRACE_FILTER="Client=(KSP P*)"
```

Each request of the user ID "KSP" and each request of the user IDs starting with a "P" are traced.

See [Trace Filter](#) in the section *Operating the Natural Web I/O Interface Server*.

TRACE_LEVEL

Value	Explanation
<i>trace-level</i>	See Trace Level in the section <i>Operating the Natural Web I/O Interface Server</i> .
0	This is the default value.

Example:

```
TRACE_LEVEL=0x00000011
```

or alternatively

```
TRACE_LEVEL=31+27
```

The setting in the example switches on **Bits 31 and 27**.

UPPERCASE_SYSTEMMESSAGES

This configuration parameter is used to enable or disable the translation of all NWO error messages and trace outputs to uppercase. This feature is for customers who are using character sets with no lowercase characters defined.

Value	Explanation
YES	Enable uppercase translation.
NO	Disable uppercase translation. This is the default value.

Web I/O Interface Server Configuration File Example

For z/OS:

```
# This is a comment
SESSION_PARAMETER=profile=(stgqa,10,930) fuser=(10,32) CFICU=ON
THREAD_NUMBER=2
THREAD_SIZE=700
FRONTEND_NAME=NATOSvrL      # and another comment
PORT_NUMBER=4811
```

Where *vr* is the current product version and release number.

For z/VSE:

```
# This is a comment
SESSION_PARAMETER=profile=(stgqa,10,930) fuser=(10,32) CFICU=ON
DEFAULT_PROFILE=DEFPROF
FRONTEND_NAME=NATNCF      # and another comment
PORT_NUMBER=4711
```

For BS2000:

```

SESSION_PARAMETER = 'NUCNAME=NvrLPRRB' +
  ' PROFILE=(NWOPARM,18006,58),ADAMODE=0,' +
  ' FNAT=(18006,58),FUSER=(18006,19),FDIC=(18006,11),' +
  ' FSPPOOL=(18006,58),FSEC=(18006,58),CFICU=ON,MENU=OFF,CP=EDF03IRV'
THREAD_NUMBER = 3
THREAD_SIZE = 900
FRONTEND_OPTIONS = 0X01
FRONTEND_NAME = NCFSERV
PORT_NUMBER = 4811
MONITOR=Y

```

Where *vr* is the current product version and release number.

Web I/O Interface Server Data Sets for z/OS, z/VSE

The Natural Web I/O Interface server requires the following data sets:

STGCONFG	Defines the server configuration file.
STGTRACE	The server trace output.
STGSTDO	The stdo data set.
STGSTDE	The stde error output.

Alternately, you can qualify each data set name by the server ID. Under z/VSE, this is necessary if you want to start different Natural Web I/O Interface servers under a single SMARTS address space.

NWOS1C	Defines the server configuration file for the server NWOS1.
NWOS1T	The server trace output for the server NWOS1.
NWOS1O	The stdo data set for the server NWOS1.
NWOS1E	The stde error output for the server NWOS1.

Web I/O Interface Server User Exits

The Natural Web I/O Interface server offers the following user exit:

User Exit NSECUX01

This user exit is applicable only when the parameter `SECURITY_MODE` is set to `IMPERSONATE_LOCAL` or `IMPERSONATE`.

This user exit allows you to adapt the user ID used for the RACF login. It is useful if the RACF user IDs and the user IDs used in Natural differ according to a standardized rule. For example, each RACF user ID is the corresponding Natural user ID preceded by two dollar signs (\$\$).

If the exit (the load module `NSECUX01`) is found in the NWO load library concatenation, it is called using standard linkage conventions (direct branch using a `BASR` instruction) before the user is validated against RACF.

The following parameters are passed to the exit:

Name	Format	In/Out	Description
<code>sUId</code>	<code>CL64</code>	I/O	User ID to be modified for RACF login.

The exit is called using standard linkage conventions.

Sample user exit implemented in C:

```
#include <string.h>
#include <stdio.h>

# pragma linkage (NSECUX01, FETCHABLE)

void NSECUX01(char sUId[64])
{
    char sUIdTemp[64];

    printf("Uex got usid:%s\n", sUId);
    strcpy(sUIdTemp, sUId);
    sprintf(sUId, "$$%s", sUIdTemp);
    printf("Uex ret usid:%s\n", sUId);
    return;
}
```

The exit above extends each user ID by two preceding dollar signs (\$\$) when it is used for RACF login.

11 Installing the Natural Web I/O Interface Server CICS Adapter under z/OS

▪ Prerequisites	96
▪ Installation Procedure	96

This chapter describes how to install the CICS connection for a Natural Web I/O Interface server (product code NWO) running under z/OS in batch mode.

Prerequisites

For details, refer to the section *Prerequisites*.

Installation Procedure

To install the Natural Web I/O Interface Server CICS Adapter, perform the following steps:

Step 1: Customize CICS

(Job I005, Steps 9405, 9406, 9410, 9411)

The Natural Web I/O Interface server load library must be defined in the CICS `DFHRPL` concatenation.

The CICS TCP/IP environment can be customized using the configuration macro `EZACICD` in the `EZACONFG` configuration data set or the configuration transaction `EZAC`. This section describes the usage of the transaction `EZAC`.

Customize the standard listener `CSKL` of the CICS socket interface using the CICS transaction `EZAC`, `ALTER`, `LISTENER` and, on the second screen, define `NATUXRFE` in the `SECEXIT`.

For impersonation with pass phrases an enhanced listener is required. Customize an existing enhanced listener or define a new one using the CICS transaction `EZAC`, `DEFINE`, `LISTENER` with `TRANID` `CSEL` (or another name of your choice) and `FORMAT` `ENHANCED`. Customize this enhanced listener by altering the settings on the second screen as follows:

```
CSTRANid    ===> NRFE
CSSTTYPe    ===> KC
CSDELAY     ===> 000000
MSGLENgth   ===> 000
PEEKDATA    ===> NO
MSGFORMat   ===> EBCDIC
USEREXIT    ===> NATUXRFE
GETTID      ===> NO
USERID      ===>
```

The definition of `SECEXIT=NATUXRFE` is mandatory when an enhanced listener is used or when the Natural Web I/O Interface server is started with impersonation (parameter `SECURITY_MODE`).

Start the listener using the CICS transaction `EZA0`.

Specify `AF=INET6` if IPv6 (Internet Protocol Version 6) is to be used.

 **Note:** Using IPv6 is only possible with Natural Web I/O Interface Server Version 8.3.2 or above

The following CICS resource definitions are required:

1. Define the CICS transaction for the remote front-end. This transaction name is an arbitrary name which must be defined in the NWO configuration parameter `RFE_CICS_TA_NAME`. This document uses the transaction name `NRFE`.

```
DEFINE TRANSACTION(NRFE) GROUP(NWOgroup)
PROGRAM(NATCNRFE)
TWSIZE(128)
RESTART(NO)
TASKDATAKEY(USER)
TASKDATALOC(ANY)
```

2. Define the programs `NATCNRFE`, `NATLRGWO` and `NATUXRFE`:

```
DEFINE PROGRAM(NATCNRFE) GROUP(NWOgroup)
LANGUAGE(C) DATALOCATION(ANY) EXECKEY(USER)
```

```
DEFINE PROGRAM(NATLRGWO) GROUP(NWOgroup)
LANGUAGE(C) DATALOCATION(ANY) EXECKEY(USER)
```

```
DEFINE PROGRAM(NATUXRFE) GROUP(NWOgroup)
LANGUAGE(LLE370) DATALOCATION(ANY) EXECKEY(CICS)
```

`NATUXRFE` must be defined with `EXECKEY(CICS)` because the transaction `CSKL` is defined with `TASKDATAKEY(CICS)` and the program `NATUXRFE` is part of the calling chain initiated by `CSKL`. This also applies if another transaction defined with `TASKDATAKEY(CICS)` is used to invoke `NATUXRFE`.

In addition, when using the CICS open transaction environment (OTE), set the following parameters for `NATCNRFE` and `NATUXRFE`:

```
API(OPENAPI)
CONCURRENCY(THREADSAFE)
```

This is required, for example, if the parameter `OTE=YES` is set for the configuration macro `EZACICD` with `TYPE=CICS` for the CICS region, or if the parameter is set with the transaction `EZAC,ALTER,CICS`.

The values of `API`, `CONCURRENCY` and `EXECKEY` for `NATCNRFE` must be the same as for the environment-dependent nucleus of Natural because Natural is called by `NATCNRFE` using standard

linkage conventions (direct branch using a BASR instruction) instead of an EXEC CICS LINK command.

You need not adapt the program definition of NATLRGW0 for the CICS OTE.

3. For DB2 access, a DB2 plan name must be defined. If you have not specified a DB2 plan name for pool threads in the DB2CONN resource definition, the transaction specified in RFE_CICS_TA_NAME and its associated DB2 plan name must be defined to CICS with a DB2TRAN and/or DB2ENTRY resource definition.



Note: The dynamic plan selection provided by the Natural for DB2 interface must not be used.

4. For DB2 access, the authorization ID under which the NWO CICS transaction is accessing DB2 must have the necessary privileges for DB2 access. The authorization ID to be used is specified in the DB2ENTRY resource definition. If you choose the USERID option, the user ID of the CICS system will be used because the NWO CICS transactions are running under the user ID of the CICS system.

The sample JCL containing the following members defines all necessary CICS entries:

- NWOCNF

Step 2: Create member for CICS server

- Job I009, Step 9411, create member NWOCNFC for CICS server

Step 3: Link the object modules into the NWO load library

(Job I054, Step 8420)

The NWO object modules must be linked with the necessary runtime extensions of your CICS installations into executable load modules.

See sample job NW0I054 on data set NW0vrs.JOBS.

Step 4: Create startup procedure

- Job I200, Step 9416, create startup procedure for CICS server

Step 5: Customize the Natural Web I/O Interface Server

In order to dispatch the NWO Natural sessions in CICS, you must adapt the configuration file of your Natural Web I/O Interface server running under z/OS in batch mode. For this purpose, two sample JCL members (NW0I009C and NW0CONF0) are available.

Refer to [Configuring the Natural Web I/O Interface Server CICS Adapter](#) and to [Configuring the Natural Web I/O Interface Server](#).

12 Installing the Natural Web I/O Interface Server CICS

Adapter under SMARTS on z/VSE

■ Prerequisites	102
■ Installation Procedure	102

This chapter describes how to install the CICS connection for a Natural Web I/O Interface server (product code NWO) running under SMARTS on z/VSE.

Prerequisites

For details, refer to the section [Prerequisites](#).

Installation Procedure

To install the Natural Web I/O Interface Server CICS Adapter, perform the following steps:

Step 1: Customize CICS

(Job I005, Step 9405)

The Natural Web I/O Interface server sublibrary must be defined in the CICS Libdef search chain.

Customize the standard listener EZAL of the CICS socket interface using the CICS transaction EZAC, DISPLAY, LISTENER and, on the second screen, define NATUXRFE in the SECEXIT field of EZAL.

As of z/VSE Version 4.1, the CICS *task related user exit* must be active (transaction EZAT).

Start the standard listener using the CICS transaction EZA0.

The following CICS resource definitions are required:

1. Define the CICS transaction for the remote front-end. This transaction name is an arbitrary name which must be defined in the NWO configuration parameter RFE_CICS_TA_NAME. This document uses the transaction name NRFE.

```
DEFINE TRANSACTION(NRFE) GROUP(NW0group)
    PROGRAM(NATCNRFE)
    TWASIZE(128)
    RESTART(NO)
    TASKDATAKEY(USER)
    TASKDATALOC(ANY)
```

2. Define the programs NATCNRFE, NATLRGWO and NATUXRFE:

```
DEFINE PROGRAM(NATCNRFE) GROUP(NW0group)
LANGUAGE(C) DATALOCATION(ANY) EXECCKEY(USER)
```

```
DEFINE PROGRAM(NATLRGW0) GROUP(NW0group)
LANGUAGE(C) DATALOCATION(ANY) EXECCKEY(USER)
```

```
DEFINE PROGRAM(NATUXRFE) GROUP(NW0group)
LANGUAGE(C) DATALOCATION(ANY) EXECCKEY(USER)
```

NATUXRFE must be defined with EXECCKEY(CICS) because the transaction CSKL is defined with TASKDATAKEY(CICS) and the program NATUXRFE is part of the calling chain initiated by CSKL. This also applies if another transaction defined with TASKDATAKEY(CICS) is used to invoke NATUXRFE.

In addition, when using the CICS open transaction environment (OTE), set the following parameters for NATCNRFE and NATUXRFE:

```
API(OPENAPI)
CONCURRENCY(THREADSAFE)
```

This is required, for example, if the parameter OTE=YES is set for the configuration macro EZACICD with TYPE=CICS for the CICS region, or if the parameter is set with the transaction EZAC,ALTER,CICS.

The values of API, CONCURRENCY and EXECCKEY for NATCNRFE must be the same as for the environment-dependent nucleus of Natural because Natural is called by NATCNRFE using standard linkage conventions (direct branch using a BASR instruction) instead of an EXEC CICS LINK command.

You need not adapt the program definition of NATLRGW0 for CICS OTE.

Step 2 : Customize the Natural Web I/O Interface Server

In order to dispatch the NWO Natural sessions in CICS, you must adapt the configuration file of your Natural Web I/O Interface server running under SMARTS on z/VSE. For this purpose, one sample JCL member (SMAI009 Step 9410) is available.

Refer to [Configuring the Natural Web I/O Interface Server CICS Adapter](#) and to [Configuring the Natural Web I/O Interface Server](#).

Step 3: Link the object modules into the NWO load library

(Job I054, Step 9420)

The NWO object modules must be linked with the necessary runtime extensions of your CICS installations into executable load modules.

13 Configuring the Natural Web I/O Interface Server CICS

Adapter

■ Configuration File	106
■ Configuration Parameters	106

This chapter describes how to configure the CICS connection for a Natural Web I/O Interface server (product code NWO) running on z/OS or under SMARTS on z/VSE.

Configuration File

After the installation of the Natural Web I/O Interface Server CICS Adapter is complete, the configuration of the Natural Web I/O Interface Server CICS Adapter has to be done in the configuration file of the corresponding Natural Web I/O Interface server.

To enable the CICS Adapter, specify the remote front-end module in the NWO configuration parameter `FRONTEND_NAME` (`FRONTEND_NAME=NATCSRFE`).

Configuration Parameters

The following CICS-relevant configuration parameters exist:

- `RFE_CICS_TA_NAME`
- `RFE_CICS_FE_NAME`
- `RFE_CICS_TA_HOST`
- `RFE_CICS_TA_PORT`
- `RFE_CICS_TA_INIT_TOUT`
- `RFE_CICS_KEEP_TA`
- `RFE_CICS_TRACE`

RFE_CICS_TA_NAME

This configuration parameter specifies the CICS transaction to be used for starting the remote front-end in CICS. This transaction must be defined in CICS and must refer to the program `NATCNRFE`. See also [Installing the Natural Web I/O Interface Server CICS Adapter under z/OS](#) or [Installing the Natural Web I/O Interface Server CICS Adapter under SMARTS on z/VSE](#).



Note: At logon, this transaction name can be overridden by the user in order to switch to a different CICS transaction on a mainframe. See the section *Dynamically Changing the CICS Transaction Name when Starting a Session* in the client documentation for the Natural Web I/O Interface server.

Default Value	none
Example	RFE_CICS_TA_NAME=NRFE

RFE_CICS_FE_NAME

This configuration parameter specifies the Natural CICS nucleus you have installed with the applicable Natural for Mainframes installation under CICS. This program must be defined in CICS.

Default Value	none
Example	RFE_CICS_FE_NAME=NCIvrNUC

See also the Natural *Installation for Mainframes* documentation, *Installing the Natural CICS Interface, Customize CICS*.

RFE_CICS_TA_HOST

This configuration parameter specifies the TCP/IP address of the host the desired CICS is running. This parameter can be omitted if the Web I/O Interface server and CICS are running on the same TCP/IP node.

Default Value	The host address of the server.
Example	RFE_CICS_TA_HOST=node1 or RFE_CICS_TA_HOST=157.189.160.55

RFE_CICS_TA_PORT

This configuration parameter specifies the TCP/IP port of the CICS supplied listener.

You can acquire this port number using the CICS supplied transaction EZAC. The CICS command `EZAC DISPLAY LISTENER` shows the definitions of the CICS standard listener.

 **Note:** This port number is not used in the Web I/O Interface client to connect to a Web I/O Interface server. This port number (and the `RFE_CICS_TA_HOST` definition) is used internally by the Web I/O Interface server to communicate with the CICS region.

Possible Values	1 - 65535
Default Value	none
Example	RFE_CICS_TA_PORT=3010

RFE_CICS_TA_INIT_TOUT

If the Web I/O Interface client sends a request to a Natural Web I/O Interface server that is configured to use the CICS remote front-end, the remote front-end launches a CICS transaction (NRFE) for processing the request. The CICS transaction in turn listens to the TCP/IP to receive the data from the Web I/O Interface server required for processing the request.

This configuration parameter specifies the timeout value (in seconds) a launched transaction waits until the expected request data arrive from the server. If this timeout expires, the request aborts with a NAT9940 error.

Default Value	5
Example	RFE_CICS_TA_INIT_TOUT=20



Note: Do not define a value below 5.

RFE_CICS_KEEP_TA

For each request sent by Natural, the Natural Web I/O Interface server opens a TCP/IP connection to the CICS region and launches a CICS transaction (NRFE) for processing the request. With `RFE_CICS_KEEP_TA=YES`, the CICS transaction remains active for processing further requests of the same client. This saves the overhead for creating the TCP/IP connection and transaction initialization for successive requests, but consumes more resources within the CICS region due to waiting transactions.

The transaction wait time (for successive requests) is limited by `RFE_CICS_TA_INIT_TOUT`. That is, if the time slice between two successive requests exceeds the time specified by `RFE_CICS_TA_INIT_TOUT`, the CICS transaction and the TCP/IP connection is terminated independent of the `RFE_CICS_KEEP_TA` definition.

`RFE_CICS_TA_INIT_TOUT=5` is a reasonable value to reuse transactions for multiple requests initiated by a single action in Natural Studio and to save CICS resources if Natural Studio waits for the next action of the user.

Default Value	NO
Example	RFE_CICS_KEEP_TA=YES

RFE_CICS_TRACE

This configuration parameter specifies the trace level for the remote front-end.

The trace level is similar to the trace implemented for the Web I/O Interface server. It is a bit string where each bit is responsible for a certain trace information:

Bit 31	Trace main events (transaction initialization/termination, request processing).
Bit 30	Detailed functions.
Bit 29	Dump internal storage areas.
Bit 27	Dump buffer header exchanged between Web I/O Interface server and CICS.
Bit 26	Dump entire buffer exchanged between Web I/O Interface server and CICS.
Bit 25	Dump the Natural Web I/O Interface server relevant buffer only (remote gateway buffer).
Bit 23	Trace error situations only.
Bit 07	Activate trace in the Web I/O Interface server region.
Bit 06	Activate trace in the CICS region.
Bit 00	Reserved for trace-level extension.

The trace destination is the data set defined for STDOUT.

Default Value	0
Example	RFE_CICS_TRACE=31+27+7 Dump main events and buffer header in the CICS region (Bits 31 + 27 + 7).

The following is a sample server configuration file using the Natural Web I/O Interface Server CICS Adapter:

```
# The Web I/O Interface Server parameter.
SESSION_PARAMETER=PROFILE=(NWO,10,930)
FRONTEND_NAME=NATCSRFE           # Use the CICS Adapter front-end.
PORT_NUMBER=4711                 # The port number used by the Web I/O Interface Client.

# The CICS Adapter parameter.
RFE_CICS_TA_NAME=NRFE           # The CICS transaction for remote front-end.
RFE_CICS_TA_PORT=3010           # The port of the CICS listener.
                                # No RFE_CICS_TA_HOST is defined. This requires
                                # that CICS runs on the same node as the server.
RFE_CICS_FE_NAME=NCIvrNUC       # The name of the installed Natural CICS nucleus.
RFE_CICS_TA_INIT_TOUT=20        # Transaction timeout is 20 seconds.
```



Note: The server parameters `THREAD_NUMBER` and `THREAD_SIZE` are obsolete when the Natural Web I/O Interface Server CICS Adapter is used.

14

Installing the Natural Web I/O Interface Server IMS Adapter

■ Prerequisites	112
■ Example Jobs	112
■ Installation Procedure	112

This chapter describes how to install the IMS TM connection for a Natural Web I/O Interface server (abbreviated: “NWO server”) running under z/OS in batch mode.

Notation *vrs* or *vr*:

When used in this document, the notation *vrs* or *vr* represents the relevant product version (see also Version in the *Glossary*).

Prerequisites

For details, refer to the section *Prerequisites*.

Example Jobs

The example installation jobs are contained in the `NW0vrs.JOBS` data set and are prefixed with the product code. The data set is provided on the installation medium for Natural Web I/O Interface.

Installation Procedure

Perform the following steps:

Step 1: Customize the IMS Listener

See *z/OS Communications Server IP IMS Socket Guide* for the description of

- how to set up and operate the IMS Listener,
- how to use the IMS TCP/IP control statements.



Note: The port number of the IMS Listener is to be used later on in the NWO configuration parameter `RFE_IMS_TA_PORT`.

Step 2: Install the Server Transaction

It is assumed that the Natural IMS TM Interface is already installed. Thus, this step describes the additional steps which are necessary to create the Natural Web I/O Interface server environment.

Step 2.1: Adapt the Transaction Code Table

(Job I055, Steps 2555, 2556)

- In the NTIMSPT macro of the Natural parameter module, add the following entry:

```
NTIMSPT TRAN=NAT vrsAD ,TYPE=SFE , X
        ENVPID=ENVNW000 , X
        PSB=NII vrsAD
```

- Assemble and link the parameter module.

Step 2.2: Adapt the Environment Parameters

(Job I055, Steps 2580, 2581)

- In the NTIMSPE macro of the Natural parameter module, add a new entry:

```
NTIMSPE ENTRYNM=ENVNW000 , X
        THBELOW=OFF , X
        THSIZE=1536000 (1500K)
```

- Assemble and link the parameter module.

Step 2.3: Relink the Natural IMS TM Interface Module

(Job I055, Step 2582)

Relink the Natural IMS TM interface module NII vrsIF.

Step 2.4: Natural IMS TM Parameter Module

(Job I080, Steps 2500, 2510)

- Build the Natural IMS TM parameter module.
- Assemble and link the Natural IMS TM parameter module.

Step 2.5: Link the Server Front-End

(Job I080, Step 2586)

Link the server front-end.

Ensure that the name of the server front-end matches the value specified with the PSB keyword subparameter of the NTIMSPT macro in installation step 2.1 above.

Step 3: Customize IMS TM

- Define the PSB for the server transaction.

As a minimum requirement you need a TP PCB. In order to access DL/I databases from the server transactions, the specific DB PCBs have to be added.

Define the application of the server transaction in the stage1 input. The transaction is a single mode, single segment non-conversational transaction.

Example:

```
APPLCTN PSB=NII vrsAD,PGMTYPE=TP,SCHDTYP=PARALLEL
TRANSACT CODE=NAT vrsAD,MODE=SNGL, X
MSGTYPE=(SNGLSEG, NONRESPONSE,4)
```

- Adapt the MPP region.

In the JCL of the MPP region, add the SYSTPCD DD statement.

See *z/OS Communications Server IP IMS Socket Guide* for the description on how to configure TCPIP.DATA.

- Concatenate the data set containing the NWO load library to the STEPLIB DD statement.

Before starting the Natural Web I/O Interface Server IMS Adapter installation, set the SMA parameter NWO-SRV-IMS to Y (yes).

To install the Natural Web I/O Interface Server IMS Adapter, perform the following steps:

Step 4: Create Member for NWO Server

(Job I009, Step 9412)

- Create member `NWOCONF1` for the NWO server.

Step 5: Link the Object Modules into the NWO Load Library

(Job I054, Step 9440)



Notes:

1. The module `NATINRFE` applies to two different products: the Natural Development Server (NDV) and the Natural Web I/O Interface (NWO server). So if you have already installed NDV, the module `NATINRFE` might already be there. However, it does not matter if you reinstall `NATINRFE` with NWO because the resulting module from either installation is the same.
2. The module `NATISRFE` (referenced in config via `FRONTEND_NAME=NATISRFE`) has already been linked in the prior steps of batch installation.

Step 6: Create Startup Procedure

(Job I200, Step 9417)

- Create the startup procedure for the NWO server.

Step 7: Customize the Natural Web I/O Interface Server

In order to dispatch the NWO Natural sessions in IMS, you must adapt the configuration file of your development server running under z/OS in batch mode. For this purpose, two sample JCL members (`NWOI009I` and `NWOCONF1`) are available.

Refer to [Configuring the Natural Development Server IMS Adapter](#) and to [Configuring the Natural Web I/O Interface Server](#).

15 Configuring the Natural Web I/O Interface Server IMS

Adapter

■ Configuration File	118
■ Configuration Parameters	118

This chapter describes how to configure the IMS TM connection for a Natural Web I/O Interface server (product code NWO) running on z/OS.

Configuration File

After the installation of the Natural Web I/O Interface Server IMS Adapter is complete, the configuration of the Natural Web I/O Interface Server IMS Adapter has to be done in the configuration file of the corresponding Natural Web I/O Interface server.

To enable the Natural Web I/O Interface Server IMS Adapter, specify the remote front-end module in the NWO configuration parameter `FRONTEND_NAME`:

```
FRONTEND_NAME=NATISRFE.
```

Configuration Parameters

The following IMS-relevant configuration parameters exist:

- `RFE_IMS_TA_NAME`
- `RFE_IMS_TA_HOST`
- `RFE_IMS_TA_PORT`
- `RFE_IMS_TRACE`

RFE_IMS_TA_NAME

This configuration parameter specifies the IMS transaction to be used for starting the remote front-end in IMS TM. This transaction must be defined in IMS TM and must refer to the program NATINRFE. See also *Installing the Natural Web I/O Interface Server IMS Adapter under z/OS*.

Default Value	none
Example	RFE_IMS_TA_NAME=NATvrsAD

RFE_IMS_TA_HOST

This configuration parameter specifies the TCP/IP address of the host the desired IMS TM is running. This parameter can be omitted if the Web I/O Interface server and IMS TM are running on the same TCP/IP node.

Default Value	The host address of the server.
Example	RFE_IMS_TA_HOST=node1 or RFE_IMS_TA_HOST=157.189.160.55

RFE_IMS_TA_PORT

This configuration parameter specifies the TCP/IP port of the IMS supplied listener.



Note: This port number is not used in the Web I/O Interface client to connect to a Web I/O Interface server. This port number (and the RFE_IMS_TA_HOST definition) is used internally by the Web I/O Interface server to communicate with the IMS region.

Possible Values	1 - 65535
Default Value	none
Example	RFE_IMS_TA_PORT=3010

RFE_IMS_TRACE

This configuration parameter specifies the trace level for the remote front-end.

The trace level is similar to the trace implemented for the Web I/O Interface server. It is a bit string where each bit is responsible for a certain trace information:

Bit 31	Trace main events (transaction initialization/termination, request processing).
Bit 30	Detailed functions.
Bit 29	Dump internal storage areas.
Bit 27	Dump buffer header exchanged between Web I/O Interface server and IMS TM.
Bit 26	Dump entire buffer exchanged between Web I/O Interface server and IMS TM.
Bit 25	Dump the Natural Web I/O Interface server relevant buffer only (remote gateway buffer).
Bit 23	Trace error situations only.
Bit 07	Activate trace in the Web I/O Interface server region.
Bit 06	Activate trace in the IMS MPP region.
Bit 00	Reserved for trace-level extension.

The trace destination is the data set defined for STDOUT.

Default Value	0
Example	RFE_IMS_TRACE=31+27+7 Dump main events and buffer header in the IMS MPP region (Bits 31 + 27 + 7).

The following is a sample server configuration file using the Natural Web I/O Interface Server IMS Adapter:

```
# The Web I/O Interface Server parameter.
SESSION_PARAMETER=PROFILE=(NWO,10,930)
FRONTEND_NAME=NATISRFE           # Use the IMS Adapter front-end.
PORT_NUMBER=4711                 # The port number used by the Web I/O Interface Client.

# The IMS Adapter parameter.
RFE_IMS_TA_NAME=NATvrsAD         # The IMS transaction for the remote front-end.
RFE_IMS_TA_PORT=3020             # The port of the IMS listener.
                                  # No RFE_IMS_TA_HOST is defined. This requires
                                  # that IMS runs on the same node as the server.
```



Note: The server parameters `THREAD_NUMBER` and `THREAD_SIZE` are obsolete when the Natural Web I/O Interface Server IMS Adapter is used.

V

Installing the Natural Web I/O Interface Client

This part explains how to install the Natural Web I/O Interface client on Tomcat so that it can be used with the server part of the Natural Web I/O Interface that is running in a Natural for Mainframes, Natural for UNIX, Natural for OpenVMS or Natural for Windows runtime environment.

The following topics are covered:

[Prerequisites](#)

[Installing the Natural Web I/O Interface Client on Apache Tomcat](#)

[Migrating the Natural Web I/O Interface Client from IIS to Apache Tomcat](#)

16 Prerequisites

▪ Servlet Container	124
▪ Natural for Mainframes	124
▪ Natural for UNIX	124
▪ Natural for OpenVMS	125
▪ Natural for Windows	125
▪ Browser Prerequisites	125

Servlet Container

The following servlet container is supported. The servlet container is not delivered with the Natural Web I/O Interface. It can be obtained from the location indicated below, according to its license terms.

- **Apache Tomcat 7 and 8**
See <http://tomcat.apache.org/>.

Natural for Mainframes

If you want to use the Natural Web I/O Interface client with Natural for Mainframes, the following must be installed:

- Natural for Mainframes Version 8.2.5 or above, and
- the Natural Web I/O Interface server.

For detailed information, see:

- the *Installation* documentation for the different operating systems which is provided for Natural for Mainframes;
- the section *Installing and Configuring the Natural Web I/O Interface Server* in the version of this *Natural Web I/O Interface* documentation which is provided for Natural for Mainframes.

Natural for UNIX

If you want to use the Natural Web I/O Interface client with Natural for UNIX, the following must be installed:

- Natural for UNIX Version 8.3 or above, and
- the Natural Web I/O Interface server (which is implemented as a daemon).

For detailed information, see:

- the *Installation* documentation which is provided for Natural for UNIX;
- the section *Installing and Configuring the Natural Web I/O Interface Server* in the version of this *Natural Web I/O Interface* documentation which is provided for Natural for UNIX.

Natural for OpenVMS

If you want to use the Natural Web I/O Interface client with Natural for OpenVMS, the following must be installed:

- Natural for OpenVMS Version 6.3.4 or above, and
- the Natural Web I/O Interface server (which is implemented as a daemon).

For detailed information, see:

- the *Installation* documentation which is provided for Natural for OpenVMS;
- the section *Installing and Configuring the Natural Web I/O Interface Server* in the version of this *Natural Web I/O Interface* documentation which is provided for Natural for OpenVMS.

Natural for Windows

If you want to use the Natural Web I/O Interface client with Natural for Windows, the following must be installed:

- Natural for Windows Version 8.3 or above, and
- the Natural Web I/O Interface server (which is implemented as a service).

For detailed information, see:

- the *Installation* documentation which is provided for Natural for Windows;
- the section *Installing and Configuring the Natural Web I/O Interface Server* in the version of this *Natural Web I/O Interface* documentation which is provided for Natural for Windows.

Browser Prerequisites

Supported browsers in this version are:

Internet Explorer 11 ⁽¹⁾

Microsoft Edge

Mozilla Firefox Extended Support Release 38 and 45 ⁽²⁾

Google Chrome ⁽³⁾

Notes:

⁽¹⁾ The Natural Web I/O Interface client Version 1.3.15 is the last version that supports Internet Explorer 8.

⁽²⁾ Only the Extended Support Releases of Mozilla Firefox are explicitly supported. Due to frequent upgrades of the Mozilla Firefox consumer release, the compatibility of the Natural Web I/O Interface client with future versions of Mozilla Firefox cannot be fully guaranteed. Possible incompatibilities will be removed during the regular maintenance process of the Natural Web I/O Interface client.

⁽³⁾ The Google Chrome support is based on Google Chrome Version 31. Due to frequent version upgrades of Google Chrome, compatibility of the Natural Web I/O Interface client with future versions of Google Chrome cannot be fully guaranteed. Possible incompatibilities will be removed during the regular maintenance process of the Natural Web I/O Interface client.



Important: Cookies and JavaScript must be enabled in the browser.

17 Installing the Natural Web I/O Interface Client on Apache

Tomcat

- Installation Steps 128
- Installation Verification 130

If you want to use the Natural Web I/O Interface client with Apache Tomcat, you must proceed as described in this chapter.

Installation Steps

The Natural Web I/O Interface client is installed using the Tomcat Manager.

The following is assumed:

- `<install-dir>` is the path to Software AG's main installation directory. By default, this is `C:\SoftwareAG` on Windows and `/opt/softwareag` on UNIX.
- `<host>` is the name of the machine on which Apache Tomcat is installed.
- `<port>` is the name of the port where Apache Tomcat is installed. In a default installation, this is port 8080.
- `<tomcat>` is the path to the directory in which Apache Tomcat is installed.

The following topics are covered below:

- [First-time Installation](#)
- [Update Installation](#)

First-time Installation

➤ To install the Natural Web I/O Interface client

- 1 Natural for Windows and UNIX: Copy the complete contents of the `<install-dir>/natural/INSTALL/nwoclient/j2ee/v<nnnn>/tomcat` directory to a directory of your choice on your hard disk.

Or:

Download the Natural Web I/O Interface client for Apache Tomcat from Empower (<https://empower.softwareag.com/>) and unzip the contents to a directory of your choice on your hard disk.

- 2 On UNIX platforms: Dearchive the TAR file using the following command:

```
tar -xvf nwonnnn.tar
```

- 3 Make sure that Apache Tomcat is running.
- 4 Open your web browser and enter the following URL:

```
http://<host>:<port>/manager/html
```

This opens the Tomcat Manager.

- 5 Deploy the web application file *natuniweb.war*:
 - Under **Select WAR file to upload** select the path to the file *natuniweb.war*.
 - Choose **Deploy**.
- 6 In the Tomcat Manager, look for the application **Natural Web I/O Interface Client** and choose **Reload**.

Update Installation

➤ To update the Natural Web I/O Interface client

- 1 Natural for Windows and UNIX: Copy the complete contents of the `<install-dir>/natural/INSTALL/nwoclient/j2ee/v<nnnn>/tomcat` directory to a directory of your choice on your hard disk.

Or:

Download the Natural Web I/O Interface client for Apache Tomcat from Empower (<https://empower.softwareag.com/>) and unzip the contents to a directory of your choice on your hard disk.

- 2 On UNIX platforms: Dearchive the TAR file using the following command:

```
tar -xvf nwonnnn.tar
```

- 3 Shut down Apache Tomcat.
- 4 Create a backup copy of your *sessions.xml* file, which is located in `<tomcat>/webapps/natuniweb/WEB-INF`.
- 5 Start Apache Tomcat.
- 6 Open your web browser and enter the following URL:

```
http://<host>:<port>/manager/html
```

This opens the Tomcat Manager.

- 7 Select *natuniweb.war* in the list of installed applications.
- 8 Choose **Undeploy**.
- 9 Deploy the new version of the Natural Web I/O Interface client as in a first-time installation.
- 10 Restore the *sessions.xml* file that you have backed up previously.

Installation Verification

It is assumed that `http://<host>:<port>` is the URL of your application server.

➤ To verify the installation

- Enter the following URL in your web browser:

```
http://<host>:<port>/natuniweb/natural.jsp
```

For example:

```
http://myhost:8080/natuniweb/natural.jsp
```

The Natural Web I/O Interface client is now started in your browser. The entries which appear in the resulting logon page depend on the settings in your configuration file. For further information, see [Configuring the Client](#).

18 Migrating the Natural Web I/O Interface Client from IIS to Apache Tomcat

- Before You Install the Natural Web I/O Interface Client 132
- Installing the Natural Web I/O Interface Client on Apache Tomcat 133
- Configuring the Natural Web I/O Interface Client on Apache Tomcat 133

Microsoft Internet Information Services (IIS) is no longer supported. If you are currently using the Natural Web I/O Interface client on IIS, you have to move to Apache Tomcat.



Note: JBoss Application Server and Oracle GlassFish Server are also no longer supported. If you are currently using one of these application servers, you also have to move to Apache Tomcat. In this case, however, you can reuse your previous settings (that is, the URL for logon page and the configuration file *sessions.xml*).

The most simple solution is to migrate the Natural Web I/O Interface client from IIS to Apache Tomcat. Therefore, this chapter gives IIS administrators a quick introduction to a Tomcat installation and describes the migration steps.

Before You Install the Natural Web I/O Interface Client

If Apache Tomcat is not yet installed, proceed as described in the topics below:

- [Installing Tomcat](#)
- [Installing Java](#)
- [Starting the Tomcat Server](#)

Installing Tomcat

Go to <http://tomcat.apache.org/download-60.cgi> and download Tomcat 6 as a zip file.

For Microsoft Windows users: download either the 32-bit or the 64-bit Windows zip file.

Unzip the downloaded zip file to a directory of your choice.

Installing Java

Tomcat is based on Java. Therefore, you have to make sure that a Java Runtime Environment (JRE) or a Java Development Kit (JDK) is installed. The version of the Java runtime should be at least Java 6 update 24. This is the minimum version that is required for the Natural Web I/O Interface client on Tomcat.

You can download the Java JRE or JDK from the Oracle website at <http://www.oracle.com/technetwork/java/javase/downloads/index.html>.

If Java is installed on your system, make sure that the environment variable `JAVA_HOME` is set to the Java home directory.

Starting the Tomcat Server

When Tomcat and the appropriate Java version have been installed, you can start Tomcat.

To start Tomcat, execute the *startup.bat* file from the *bin* directory of your Tomcat installation. To check whether Tomcat is running, enter the following URL:

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

This should display Tomcat's default home page.

Installing the Natural Web I/O Interface Client on Apache Tomcat

When Apache Tomcat has been installed, install the Natural Web I/O Interface client as described in [Installing the Natural Web I/O Interface Client on Apache Tomcat](#).

Configuring the Natural Web I/O Interface Client on Apache Tomcat

When the Natural Web I/O Interface client has been installed, proceed as described in the topics below:

- [Invoking the Logon Page](#)
- [Changing the Tomcat HTTP Port](#)
- [Using the Settings from Your IIS Configuration File](#)
- [Using the Configuration Tool](#)
- [Protecting the Configuration Tool Against Unauthorized Access](#)
- [Displaying the Logon Page by Default](#)

Invoking the Logon Page

Enter the following URL to invoke the logon page (this is different from the URL that was used with IIS):

```
http://localhost:8080/natuniweb/natural.jsp
```

Changing the Tomcat HTTP Port

IIS usually runs on the default port 80. If you want Tomcat to work with the same port, edit the file *server.xml* which is located in Tomcat's *conf* subdirectory and then search for the following text:

```
<Connector port="8080" protocol="HTTP/1.1"
```

Change the port number so that it looks as follows:

```
<Connector port="80" protocol="HTTP/1.1"
```

Using the Settings from Your IIS Configuration File

With Tomcat, you can reuse the *settings.xml* configuration file of IIS, but you have to rename the file to *sessions.xml*. Proceed as follows:

1. Copy the *settings.xml* file from your IIS installation to the following directory of your Tomcat installation:

```
webapps/natuniweb/WEB-INF
```

2. Either rename the *sessions.xml* file which comes with the Natural Web I/O Interface client installation on Tomcat (for example, to *sessions-original.xml*) or delete it.
3. Rename the *settings.xml* file to *sessions.xml*.

Using the Configuration Tool

When the Natural Web I/O Interface client runs on Tomcat, it is no longer necessary to edit the configuration file manually. Instead, you can use the configuration tool. Using this tool has the advantage that it is not possible for you to create invalid XML code and thus damage the XML file. See [Using the Configuration Tool](#) for further information.

The IIS-specific entries in the renamed configuration file will be ignored. These are:

```
natural_parameter visible  
theme  
screen top  
screen left  
screen size  
screen pfkeypos
```

You can still edit the configuration file manually. However, this is no longer recommended.

Protecting the Configuration Tool Against Unauthorized Access

It is possible to protect the configuration tool against unauthorized access. See [Configuring Container-Managed Security](#) for detailed information.

For detailed information on the necessary realm configuration for Tomcat, see <http://tomcat.apache.org/tomcat-6.0-doc/realm-howto.html>.

Displaying the Logon Page by Default

When you enter the URL `http://localhost:8080/natuniweb`, Tomcat shows the default page of the Natural Web I/O Interface client which allows you to access either the **logon page** or the **configuration tool** of the Natural Web I/O Interface client.



Note: If you have defined a different port (for example, 80), make sure to use that port number in the URL.

This behavior is different from IIS which displays the logon page by default. If you also want Tomcat to display the logon page by default, edit the file `web.xml` which is located in Tomcat's `webapps\natuniweb\WEB-INF` directory and search for the following entry:

```
<welcome-file-list>
  <welcome-file>
    index.html
  </welcome-file>
</welcome-file-list>
```

Change the name of the welcome file to `natural.jsp` as shown in the following example:

```
<welcome-file-list>
  <welcome-file>
    natural.jsp
  </welcome-file>
</welcome-file-list>
```

VI

Configuring the Client

This part explains how to configure the Natural Web I/O Interface client so that it can be used in a Natural runtime environment. The following topics are covered:

[About the Logon Page](#)

[Natural Client Session Configuration](#)

[Natural Client Configuration Tool](#)

[Natural Web I/O Style Sheets](#)

[Starting a Natural Application with a URL](#)

[Configuring Container-Managed Security](#)

[Configuring SSL](#)

[Logging](#)

19

About the Logon Page

- Starting a Natural Application from the Logon Page 140
- Examples of Logon Pages 140
- Dynamically Changing the CICS Transaction Name when Starting a Session 141
- Specifying a Password in the Logon Page 142
- Changing the Password in the Logon Page 142
- Browser Restrictions 143

Starting a Natural Application from the Logon Page

When you start the Natural Web I/O Interface client in the browser, a logon page appears. The entries in this logon page depend on the settings in your configuration file (see [Natural Client Session Configuration](#)).

In order to start a Natural application from the logon page, you enter the following URL inside your browser:

```
http://<host>:<port>/natuniweb/natural.jsp
```

where *<host>* and *<port>* are the host name and port number of your application server.

Examples of Logon Pages

For each session definition that has been configured in the configuration file, an entry appears on the logon page. If the user selects the corresponding entry, only those parameters that were not preconfigured in the configuration file need to be specified in the logon page in order to start the application. Usually, you will preconfigure all connection parameters except user name and password.

The following example shows part of a logon page which results from a configuration file in which no special entries are defined for a session:

The screenshot shows a web form titled "Enter connection details:". It contains the following elements:

- Session ID:** A dropdown menu with "Connect to Natural" selected.
- Host name:** A text input field.
- Port number:** A text input field.
- User name:** A text input field.
- Password:** A text input field.
- Application:** A text input field.
- Natural parameters:** A text input field.
- Buttons:** "Connect" and "Change password" buttons are located to the right of the Session ID dropdown.

The following example shows part of a logon page which results from a configuration file in which many settings are already predefined (including user ID and password):

Enter connection details:

Session ID:

To log on to a session, you have to specify all required information in the logon page (for example, you select a session from the corresponding drop-down list box). When you choose the **Connect** button, the screen for the selected session appears.

Dynamically Changing the CICS Transaction Name when Starting a Session

The following description applies if you want to switch to a different CICS transaction on a mainframe.

You specify the CICS transaction name in the same text box in which you also specify the dynamic parameters for the Natural environment. So that the CICS transaction name can be evaluated, it is important that you specify it before any Natural parameters, using the following syntax:

```
<TA_NAME=name>
```

where *name* can be 1 to 4 characters long. This must be the name of an existing CICS transaction which applies to a CICS Adapter. It will override the transaction name which is currently defined in the configuration file for the CICS Adapter on the Natural Web I/O Interface server (NWO). Ask your administrator for further information.

Make sure to put the entire definition in angle brackets. When this definition is followed by a Natural parameter, insert a blank before the Natural parameter. Example:

```
<TA_NAME=NA82> STACK=(LOGON SYSCP)
```

If the specified CICS transaction name cannot be found, an error message occurs and the session cannot be started.



Note: The above definition for the CICS transaction name can also be specified in the [configuration tool](#), in the same place where you also specify the Natural parameters, and together with the [URL parameter](#) natparam.

Specifying a Password in the Logon Page

The following information applies when the field for entering a password appears on the logon page. This field does not appear when a password has already been defined in the configuration file.

Under Windows, UNIX and OpenVMS, you always have to enter the operating system password, even if Natural Security is active.

On the mainframe, this is different: When Natural Security is not active, you have to enter the operating system password. When Natural Security is active, you have to enter the Natural Security password.

Changing the Password in the Logon Page

Currently, this functionality is only available for Natural for UNIX, Natural for OpenVMS and Natural for Windows.

The following information applies when the fields for entering a user ID and a password appear on the logon page. These fields do not appear when user ID and password have already been defined in the configuration file; in this case, it is not possible to change the password in the logon page.

When your password has expired, you are automatically asked for a new password. When you try to log on with your current password, an error message appears and input fields for changing the password are shown.

➤ To change the password

- 1 Choose the **Change password** button in the logon page.

The name of this button changes to **Don't change password** and the following two input fields are shown in the logon page:

- **New password**
- **Repeat new password**

- 2 Enter your user ID and your current password as usual.
- 3 Enter the new password in the two input fields.
- 4 Choose the **Connect** button to change the password.

Or:

If you do not want to change your password, choose the **Don't change password** button. The two input fields will then disappear.

Browser Restrictions

The browser's "Back" and "Forward" buttons do not work with the Natural Web I/O Interface client and should therefore not be used.

If you want to run two Natural sessions in parallel, you have to start a new instance of the browser (for example, by choosing the corresponding icon in the Quick Launch toolbar of Windows). You must not use the browser's "New Window" function. This would result in one session running in two browsers, which is not allowed.

20 Natural Client Session Configuration

- General Information 146
- Name and Location of the Configuration File 146

General Information

The configuration file is an XML file which is required to define the sessions that can be invoked from the logon page.

To edit the configuration file, you use the configuration tool. Using this tool has the advantage that it is not possible for you to create invalid XML code and thus damage the XML file. See [Natural Client Configuration Tool](#) for further information.

Name and Location of the Configuration File

The name of the configuration file is *sessions.xml*. It can be found in the *WEB-INF* directory.

`<tomcat-install-dir>/webapps/natuniweb/WEB-INF`

21 Natural Client Configuration Tool

▪ Invoking the Configuration Tool	148
▪ Session Configuration	149
▪ Logging Configuration	157
▪ Logon Page	157
▪ Logout	158

Invoking the Configuration Tool

The Natural Web I/O Interface client offers a configuration tool. The configuration tool is used to create the session configurations which are then available in the logon page. It can also be used for logging purposes in case of problems; however, this should only be done when requested by Software AG support.

The configuration tool is automatically installed when you install the Natural Web I/O Interface client.

➤ To invoke the configuration tool

- Enter the following URL in your browser:

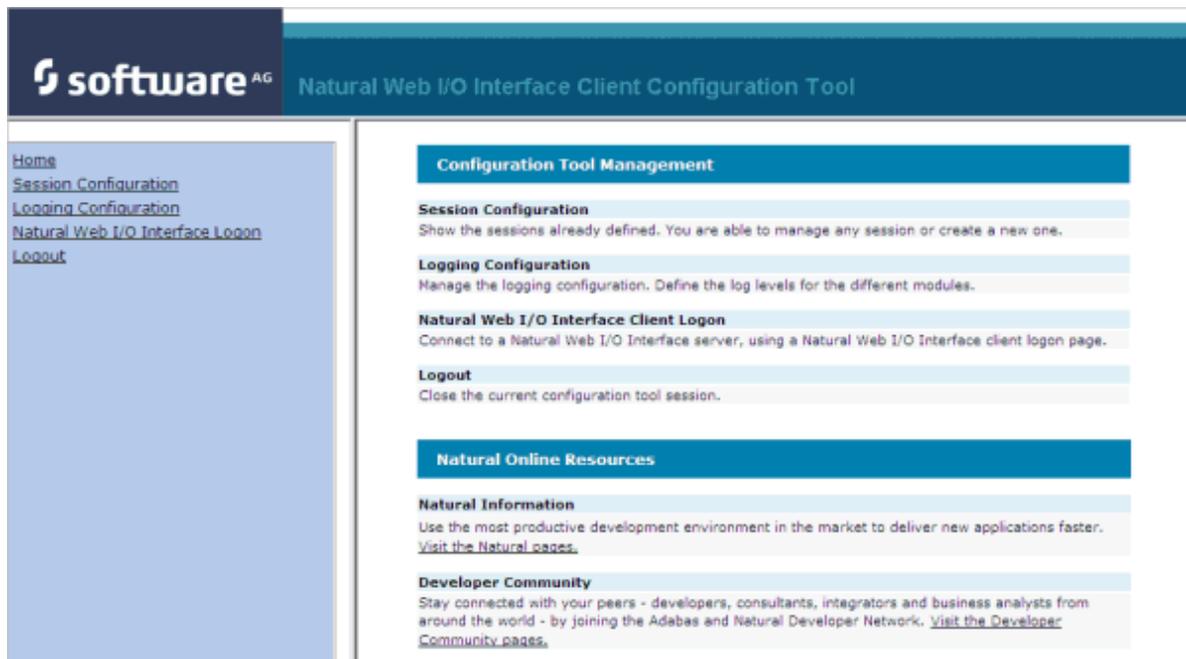
```
http://<host>:<port>/natuniweb/conf_index.jsp
```

where *<host>* and *<port>* are the host name and port number of your application server.



Note: You might wish to protect the configuration tool against unauthorized access. See [Configuring Container-Managed Security](#) for information on how to restrict the access to sensitive areas of the application server environment. If you have restricted access to the configuration tool, an authentication dialog appears. The appearance of this dialog depends on the authentication model you have chosen.

The configuration tool appears.



The configuration tool has two frames.

The home page of the configuration tool is initially shown in the right frame. It provides brief descriptions for the links provided in the left frame. It also provides links to several Software AG pages on the web.

When you have invoked a function (for example, when you are currently viewing the session configuration), you can always choose the **Home** link in the left frame to return to the home page of the configuration tool.

The functions that are invoked by the other links in the left frame are described below.

Session Configuration

This section explains how to manage the content of the configuration file for the sessions. It covers the following topics:

- [Invoking the Session Configuration Page](#)
- [Global Settings](#)
- [Adding a New Session](#)
- [Editing a Session](#)
- [Overview of Session Options](#)
- [Duplicating a Session](#)
- [Deleting a Session](#)
- [Adding a New User](#)
- [Saving the Configuration](#)

Invoking the Session Configuration Page

The content of the configuration file for the sessions is managed using the **Session Configuration** page.

➤ To invoke the Session Configuration page

- In the frame on the left, choose the **Session Configuration** link.

The **Session Configuration** page appears in the right frame. It shows the global settings and lists all sessions and users that are currently defined. For a session, some of the configuration file information is shown. Example:

Session Configuration

Global Settings

Last activity timeout (n seconds):

Trace directory:

SSL trust file path:

SSL trust file password:

Sessions

Session ID	Host Name	Port Number	Application	Natural Parameters	Edit	Duplicate	Delete
Connect to Natural					Edit	Duplicate	Delete
localtestserver	localhost	6640			Edit	Duplicate	Delete

Users

User ID	Edit	Duplicate	Delete
user1	Edit	Duplicate	Delete

Global Settings

The global settings apply for all defined sessions. You can define the following global settings in the configuration file:

Option	Description
Last activity timeout (n seconds)	The number of seconds that the client waits for the next user activity. When the defined number of seconds has been reached without user activity, the session is closed. The default is 3600 seconds.
Trace directory	<p>Optional. Location of a different trace directory.</p> <p>When a different trace directory is not defined, the trace files are written to the default trace directory. By default, the trace files are written to the directory which has been set by the Java property <code>java.io.tmpdir</code>. On Windows, this is normally the environment variable <code>TMP</code> for the user who started the application server. On UNIX, this is normally <code>/tmp</code> or <code>/var/tmp</code>.</p> <p>You can also set this property in the start script for the application server.</p>

Option	Description
	Tracing can be enabled individually for each session (see Overview of Session Options below). However, it should only be enabled when requested by Software AG support.
SSL trust file path	Optional. The path to your trust file. See Configuring SSL for further information.
SSL trust file password	If your trust file is password-protected, you have to specify the appropriate password. When you do not specify the password for a password-protected trust file, the trust file cannot be opened and it is thus not possible to open an SSL session. When your trust file is not password-protected, you should not specify a password.

Adding a New Session

You can add a new session to the configuration file.

➤ To add a new session

- 1 Choose the **Add New Session** button.

The **Edit Session** page appears.

- 2 Specify all required information as described below in the section [Overview of Session Options](#).
- 3 Choose the **OK** button to return to the **Session Configuration** page.

The new session is not yet available in the configuration file.

- 4 Choose the **Save Configuration** button to write the new session to the configuration file.

Editing a Session

You can edit any existing session in the configuration file.

➤ To edit a session

- 1 Choose the **Edit** link that is shown next to the session that you want to edit.

The **Edit Session** page appears.

- 2 Specify all required information as described below in the section [Overview of Session Options](#).
- 3 Choose the **OK** button to return to the **Session Configuration** page.

The modifications are not yet available in the configuration file.

- 4 Choose the **Save Configuration** button to write the modifications to the configuration file.

Overview of Session Options

The **Edit Session** page appears when you

- **add** a new session, or
- **edit** an existing session.

Example:

Edit Session

Session ID:	<input type="text"/>
Type:	Undefined <input type="text"/>
Host name:	<input type="text"/>
Port number:	<input type="text"/>
Use SSL	<input type="radio"/> Yes <input checked="" type="radio"/> No
User name:	<input type="text"/>
User name in upper case:	<input type="radio"/> Yes <input checked="" type="radio"/> No
Password:	<input type="text"/>
Application:	<input type="text"/>
Natural parameters:	<input type="text"/>
Double-click behavior:	Enter <input type="text"/>
Screen rows:	24 <input type="text"/>
Screen columns:	80 <input type="text"/>
Show function key numbers:	<input type="radio"/> Yes <input checked="" type="radio"/> No
Check for numeric input:	<input checked="" type="radio"/> Yes <input type="radio"/> No
Trace:	<input type="radio"/> Yes <input checked="" type="radio"/> No
Timeout (in seconds):	60 <input type="text"/>
Filler character:	<input type="text"/>

The **Edit Session** page provides the following options:

Option	Description
Session ID	Mandatory. A session name of your choice. On the logon page, the session name is provided in a drop-down list box.
Type	<p>The platform on which user ID and password are authenticated. You can select the required setting from the drop-down list box.</p> <ul style="list-style-type: none"> ■ Undefined Default. User ID and password can have a maximum of 32 characters. See also the description for Natural for Windows, UNIX or OpenVMS below. ■ Natural for Mainframes User ID and password can have a maximum of 8 characters. ■ Natural for Mainframes with Natural Security User ID and password can have a maximum of 8 characters. The user ID must comply with the Natural naming conventions for library names. ■ Natural for Windows, UNIX or OpenVMS User ID and password can have a maximum of 32 characters. When a domain is required, you have to specify it together with the user ID (in the form "<i>domain\user-ID</i>").
Host name	The name or TCP/IP address of the server on which Natural and the Natural Web I/O Interface server are running. When this is specified, the corresponding field does not appear on the logon page.
Port number	The TCP/IP port number on which the Natural Web I/O Interface server is listening. When this is specified, the corresponding field does not appear on the logon page.
Use SSL	<p>If set to Yes, a secure connection is established between the Natural Web I/O Interface client on the application server and the Natural Web I/O Interface server.</p> <p>Important: If you want to use SSL with Natural for Mainframes, one of the corresponding mainframe types must be selected; the type must not be Undefined or Natural for Windows, UNIX or OpenVMS. The other way around, if you want to use SSL with Natural for Windows, UNIX or OpenVMS, you must not select one of the mainframe types; the type may also be Undefined in this case.</p>
User name	Optional. A valid user ID for the current machine. When this is specified, the corresponding field does not appear on the logon page.
User name in upper case	If selected, the input field for the user ID is in upper-case mode.
Password	<p>Optional. A valid password for the above user ID.</p> <p>Under Windows, UNIX and OpenVMS, this is always the operating system password of the user, even if Natural Security is active.</p> <p>On the mainframe, this is different: When Natural Security is not active, this is the operating system password of the user. When Natural Security is active, this is the Natural Security password.</p>

Option	Description
	<p>When a password is specified, the corresponding field does not appear on the logon page. The configuration tool saves the password in encrypted form.</p>
<p>Application</p>	<ul style="list-style-type: none"> ■ Natural for Mainframes The name of the Natural program or a command sequence that starts your application as you would enter it on the NEXT prompt. Example: <code>TEST01 data1,data2</code> ■ Natural for UNIX The name of the UNIX shell script for starting the Natural application (a file similar to <i>nwo.sh</i>). ■ Natural for OpenVMS The name of the Natural image file (for example, <i>natural<version></i> or <i>natural<version>.exe</i>). ■ Natural for Windows The name of the Windows command file (<i>.bat</i>) for starting the Natural application. <p>When this is specified, the corresponding field does not appear on the logon page.</p>
<p>Natural parameters</p>	<p>Optional. Parameters for starting the Natural application. This can be stack parameters, a parameter file/module or other Natural-specific information.</p> <ul style="list-style-type: none"> ■ Natural for Mainframes Used to pass dynamic Natural profile parameters to the session, for example: <code>SYSPARM=(MYPARMS) STACK=(LOGON MYAPPL)</code> <p>Note: It is recommended to specify the Natural program that starts the application with the option Application instead of passing it with the profile parameter <code>STACK</code>.</p> <ul style="list-style-type: none"> ■ Natural for UNIX and Natural for Windows Used when the above shell script (UNIX) or command file (Windows) uses the parameter \$5 after "natural", for example: <code>PARM=MYPARM STACK=(LOGON MYLIB;MENU)</code> ■ Natural for OpenVMS Used for starting a Natural application, for example: <code>BP=BPnode-name NLDCHK WEBIO=ON "STACK=(LOGON SYSEXT;MENU)"</code>
<p>Double-click behavior</p>	<p>The key that is to be simulated when double-clicking an output field. By default, this is the ENTER key.</p> <p>It is possible to disable the double-click behavior, or to define a function key (PF1 through PF12).</p> <p>You can select the required setting from the drop-down list box.</p>

Option	Description
	<p>Tip: When context-sensitive help has been defined for the output fields, it may be useful to define PF1. The help function will then be invoked when the user double-clicks an output field.</p>
Screen rows	<p>The number of rows in the output window. Possible values: minimum 24, no upper limit. Default: 24.</p> <p>Not used by Natural for Mainframes which uses the profile parameter TMODEL instead.</p>
Screen columns	<p>The number of columns in the output window. Possible values: minimum 80, no upper limit. Default: 80.</p> <p>Not used by Natural for Mainframes which uses the profile parameter TMODEL instead.</p>
Show function key numbers	<p>If set to Yes, the PF key numbers are shown next to the PF keys.</p>
Trace	<p>Should only be set to Yes when requested by Software AG support.</p>
Check for numeric input	<p>If set to Yes (default), numeric input fields are validated. In this case, only the following characters are allowed in numeric input fields (in addition to the numbers "0" through "9"):</p> <p><i>blank</i> + (plus) - (minus) _ (underscore) , (comma) . (period) ? (question mark)</p> <p>If set to No, numeric input fields are not validated.</p>
Timeout (in seconds)	<p>The number of seconds that the client waits for a response after an updated page was sent to the Natural session. When the defined number of seconds has been reached without response, the session is closed. The default is 60 seconds. Normally, you need not change this value.</p>
Filler character	<p>Optional. The filler character that is to be removed from the input fields. An application can define, for example, an underscore (_) as the filler character. Trailing filler characters will be removed from the input fields, and leading filler characters will be replaced with blanks.</p>

Duplicating a Session

You can add a copy of any existing session to the configuration file.

➤ To duplicate a session

- 1 Choose the **Duplicate** link that is shown next to the session that you want to duplicate.

A new entry is shown at the bottom of the list of sessions. Its name is "Copy of *session-ID*". The duplicated session is not yet available in the configuration file.
- 2 **Edit** and save the duplicated session as described above.

Deleting a Session

You can delete any existing session from the configuration file.

➤ To delete a session

- 1 Choose the **Delete** link that is shown next to the session that you want to delete.

The session is deleted from the list of sessions. It is not yet deleted in the configuration file.
- 2 Choose the **Save Configuration** button to delete the session from the configuration file.

Adding a New User

You can predefine Natural users and their passwords in the configuration file.

When a Natural page is opened with a URL that specifies a user in the URL parameter `natuser`, the specified user is matched against the list of users in the configuration file. When the specified user is defined in the configuration file, the corresponding password is used to authenticate the user when the Natural session is started. See also [Starting a Natural Application with a URL](#).

Example - when the following URL is used, the password defined for "user1" is used:

`http://myhost:8080/natuniweb/natural.jsp?natuser=user1...`

➤ To add a new user

- 1 Choose the **Add New User** button.

The **Edit User** page appears.
- 2 Specify a user name and password
- 3 Choose the **OK** button to return to the **Session Configuration** page.

The new user is not yet available in the configuration file.

- 4 Choose the **Save Configuration** button to write the new user to the configuration file.



Note: You edit, duplicate and delete a user in the same way as a session (see the corresponding descriptions above).

Saving the Configuration

When you choose the **Save Configuration** button, all of your changes are written to the configuration file. The server picks up the new settings automatically the next time it reads data from the configuration file.



Caution: If you do not choose the **Save Configuration** button but log out instead or leave the configuration tool by entering another URL, the new settings are not written to the configuration file.

Logging Configuration

The content of the configuration file for logging is managed using the **Logging Configuration** page. See the section [Logging](#) for detailed information.

Logon Page

The configuration tool provides the following link in the left frame:

- **Natural Web I/O Interface Logon**

This link opens the logon page in the right frame.

The logon page uses the current settings in the configuration file. When you select a session from the drop-down list box, you can check whether the connection details are shown as desired. If not, you can go back to the session configuration and modify the settings of the corresponding session.

See also [About the Logon Page](#).

Logout

When the configuration tool is protected against unauthorized access and you log out of the configuration tool, you make sure that no other user can change the client configuration when you leave your PC unattended for a while.

> To log out

- In the frame on the left, choose the **Logout** link.

When the configuration tool is protected against unauthorized access, the authentication dialog is shown again.

When it is not protected, the home page is shown again.

22

Natural Web I/O Style Sheets

- Name and Location of the Style Sheets 160
- Editing the Style Sheets 160
- Modifying the Position of the Main Output and of the PF Keys 160
- Modifying the Font Size 162
- Modifying the Font Type 163
- Defining Underlined and Blinking Text 163
- Defining Italic Text 164
- Defining Bold Text 164
- Defining Different Styles for Output Fields 165
- Modifying the Natural Windows 165
- Modifying the Message Line 166
- Modifying the Background Color 166
- Modifying the Color Attributes 167
- Modifying the Style of the PF Key Buttons 168
- XSLT Files 168

Name and Location of the Style Sheets

Several aspects on a page (such as font, font style or color) are controlled by a style sheet (CSS file).

The Natural Web I/O Interface client is delivered with the style sheet `3270.css`, which is located in:

`../natuniapp.ear/natuniweb.war/resources`



Note: For more information on style sheets, see <http://www.w3.org/Style/CSS/>.

Editing the Style Sheets

It is recommended that you have a basic understanding of CSS files.

You can edit the predefined style sheets or create your own style sheets.

It is recommended that you work with backup copies. When a problem occurs with your style sheet, you can thus always revert to the original state.

To see your changes in the browser, you have to

1. delete the browser's cache, and
2. restart the session.

Modifying the Position of the Main Output and of the PF Keys

Applies when only the named PF keys are displayed. This feature cannot be used when all PF keys are displayed, since they are always displayed at the same position. See also [Overview of Session Options](#).

The following elements are available:

Element Name	Description
<code>.mainlayer</code>	Controls the position of the main output in the output window. Used for languages that are written from left-to-right (LTR).
<code>.mainlayer_rtl</code>	Controls the position of the main output in the output window. Used for languages that are written from right-to-left (RTL).
<code>.pfkeydiv</code>	Controls the position of the PF keys in the output window. Used for languages that are written from left-to-right (LTR).

Element Name	Description
.pfkeydiv_rtl	Controls the position of the PF keys in the output window. Used for languages that are written from right-to-left (RTL).

The *_rtl elements are only used if Natural sends the web I/O screen with a right-to-left flag (SET CONTROL 'VON'). In the browser, the screen elements are then shown on the right side (instead of the left side).

For web I/O in applications where only the left-to-right orientation is used, the *_rtl elements are not required.

If the PF keys are to appear at the bottom, define the elements as shown in the following example:

```

/* Defines the main screen position */
.mainlayer {
    top: 5px;
    left: 0px;
    height: 550px;
}

/* Defines the main screen position for right-to-left */
.mainlayer_rtl{
    top: 5px;
    right: 30px;
    height: 550px;
}

/* Defines the PF keys screen position */
.pfkeydiv {
    height: 70px;
    left: 0px;
    top: 580px;
    width: 100%;
}

/* Defines the PF keys screen position for right-to-left */
.pfkeydiv_rtl {
    height: 70px;
    right: 30px;
    top: 580px;
    width: 100%;
}

```

Modifying the Font Size

Depending on the screen resolution, one of the following style sheets for defining the font size is used in addition to the default style sheet:

- *model2.css*
- *model3.css*
- *model4.css*
- *model5.css*

These style sheets are located in the *tmodels* subdirectory of the *resources* directory in which all style sheets are located.

Depending on what comes closest to the standard 3270 screen model, the corresponding style sheet from the *tmodels* subdirectory is automatically used. It is selected according to the following criteria:

Standard 3270 Screen Model	Criteria	Style Sheet
Model 2 (80x24)	30 rows or less.	<i>model2.css</i>
Model 3 (80x32)	Between 31 and 40 rows.	<i>model3.css</i>
Model 4 (80x43)	41 rows or more.	<i>model4.css</i>
Model 5 (132x27)	30 rows or less, and more than 100 columns.	<i>model5.css</i>

The font sizes in the above style sheets can be adjusted. Example for *model4.css*:

```
body {  
    font-size: 10px;  
}
```

The default font sizes for the above 3270 screen models are:

Standard 3270 Screen Model	Default Font Size
Model 2	16px
Model 3	14px
Model 4	10px
Model 5	12px

Modifying the Font Type

As a rule, you should only use monospace fonts such as Courier New or Lucida Console. With these fonts, all characters have the same width. Otherwise, when using variable-width fonts, the output will appear deformed.

If you want to define a different font type, you should define the same font type for the body, the output fields and the input fields as shown in the following example:

```
body {
  background-color: #F3F5F0;
  font-family: Lucida Console;
}

.OutputField {
  white-space:pre;
  border-width:0;
  font-family: Lucida Console;
  font-size: 100%;
}

.InputField {
  background-color: white;
  font-family: Lucida Console;
  border-width: 1px;
  font-size: 100%;
  border-color: #A7A9AB;
}
```

Defining Underlined and Blinking Text

The following elements are available:

Element Name	Description
.natTextDecoUnderline	Defines underlined text.
.natTextDecoBlinking	Defines blinking text.
.natTextDecoNormal	Defines normal text (no underline, no blinking).

Example:

```
/* Text decoration */  
.natTextDecoUnderline { text-decoration:underline; }  
.natTextDecoBlinking {text-decoration:blink; }  
.natTextDecoNormal {text-decoration:normal;}
```

Blinking text is not supported by the Internet Explorer.

Defining Italic Text

The following elements are available:

Element Name	Description
.natFontStyleItalic	Defines italic text.
.natFontStyleNormal	Defines normal text (no italics).

Example:

```
/* font style */  
.natFontStyleItalic {font-style:italic;}  
.natFontStyleNormal {font-style:normal;}
```

Defining Bold Text

The following elements are available:

Element Name	Description
.natFontWeightBold	Defines bold text.
.natFontWeightNormal	Defines normal text (not bold).

```
/* Font weight */  
.natFontWeightBold {font-weight:bolder;}  
.natFontWeightNormal {font-weight:normal;}
```

When you define bold text (`{font-weight:bolder;}`) for the default font Courier New, your text always has the same width as with normal text (`{font-weight:normal;}`).

However, when you define bold text for Courier or Lucida Console, the bold text will be wider than the normal text and your output may thus appear deformed. It is therefore recommended that you switch off bold text for Courier and Lucida Console:

```
.natFontWeightBold {font-weight:normal;}
```

Defining Different Styles for Output Fields

The following elements are available:

Element Name	Description
.FieldVariableBased	Defines the style for output fields that are based on a variable.
.FieldLiteralBased	Defines the style for output fields that are based on a literal.

Example:

```
.FieldVariableBased {
  /* font-style:italic; */
}

.FieldLiteralBased {
  /* font-style:normal; */
}
```



Note: In the above example, as well as in the standard CSS files delivered by Software AG, the variable-based output fields are defined as italic, but are commented out.

Modifying the Natural Windows

The following elements are available:

Element Name	Description
.naturalwindow	Controls the rendering of the Natural windows.
.wintitle	Controls the rendering of the titles of the Natural windows.

Example:

```
.naturalwindow {
  border-style: solid;
  border-width: 1px;
  border-color: white;
  background-color: black;
}

.wintitle {
  left: 0px;
```

```
top: 1px;
height: 17px;
width: 100%;
color: black;
font-size: 100%;
font-weight: bold;
background-color: white;
text-align: center;
font-family: Verdana;
border-bottom-style: solid;
border-bottom-width: 2px;
}
```

 **Note:** In a mainframe environment, you have to set the Natural profile parameter `WEBIO` accordingly to enable this feature. See *WEBIO - Web I/O Interface Screen Rendering in the Parameter Reference* which is provided with Natural for Mainframes.

Modifying the Message Line

The rendering of the message line is controlled by the `.MessageLine` element.

Example:

```
.MessageLine {
  color: blue;
}
```

 **Note:** In a mainframe environment, you have to set the Natural profile parameter `WEBIO` accordingly to enable this feature. See *WEBIO - Web I/O Interface Screen Rendering in the Parameter Reference* which is provided with Natural for Mainframes.

Modifying the Background Color

The background color is defined in the `body` element.

Example:

```
body {  
  background-color: #F3F5F0;  
  font-family: Lucida Console;  
}
```

Modifying the Color Attributes

You can define different colors for all Natural color attributes. These are:

- Red
- Green
- Blue
- Yellow
- White
- Black
- Pink
- Turquoise
- Transparent

You can define these color attributes for input fields and output fields, and for normal output and reverse video.

The following examples show how to define the color attribute “Red”.

Define the color for a normal output field:

```
.natOutputRed {color: darkred;}
```

Define the foreground and background colors for an output field with reverse video:

```
.reverseOutputRed {background-color: darkred; color:#F3F5F0;}
```

Define the color for a normal input field:

```
.natInputRed {color: darkred;}
```

Define the foreground and background colors for an input field with reverse video:

```
.reverseInputRed {background-color: darkred; color:#F3F5F0;}
```

Modifying the Style of the PF Key Buttons

The following elements are available:

Element Name	Description
.PFButton	Controls the style for normal rendering.
.PFButton:hover	Controls the style that is used when the mouse hovers over a PF key button.

Example:

```
.PFButton {
  text-align: center;
  width: 90px;
  border-style: ridge;
  border-width: 3px;
  padding: 2px;
  text-decoration: none;
  font-family: Verdana;
  font-size: 12px;
  height: 22px;
}

.PFButton:hover {
  color: #FFFF00;
  background-color: #222222;
}
```



Note: In a mainframe environment, you have to set the Natural profile parameter `WEBIO` accordingly to enable this feature. See *WEBIO - Web I/O Interface Screen Rendering in the Parameter Reference* which is provided with Natural for Mainframes.

XSLT Files

In addition to the CSS files described above, the Natural Web I/O Interface client uses XSLT files with specific names for the conversion of the Natural Web I/O Interface screens from the internal XML format to HTML. The following elements are affected:

- Input text is placed into the HTML element `<input>`.
- Output text is placed into the HTML element `<input>` (with attribute `readonly="readonly"`).
- A message line is placed into the HTML element ``.

- PF keys are embedded in an XML island and then rendered with JavaScript.
- Window elements are embedded in an XML island and then rendered with JavaScript.



Note: The JavaScript file which is part of the above conversion is *natunicript.js*. It is located in the *scripts* directory which can be found in the `<installdir>/natuniweb` directory.

The name of the default XSLT file is:

- *transuni.xsl* for all supported browsers.

The default XSLT file can be found in the following directory:

`<installdir>/natuniweb/web-INF`

The XSLT file is only read once when the server is started.



Important: It is recommended that you do not change the above XSLT file. Software AG may change or correct the original XSLT transformations in new versions or service packs of the product.

You can copy your own XSLT file into the above directory. In this case, the file must have the following name:

- *usertransuni.xsl* for all supported browsers.

If this user file can be found when the server is started, it is read instead of the default XSLT file.

When you make changes to this file, you have to restart the server so that your changes become effective.

23 Starting a Natural Application with a URL

The connection parameters available in the configuration file for the session and on the logon page can also be specified as URL parameters of the logon page URL. This allows bookmarking the startup URL of a Natural application or starting an application by clicking a hyperlink in a document.

The URL parameters overrule the definitions in the configuration file, with the exception described in the table below.

The following URL parameters are available for the logon page:

URL Parameter	Corresponding Option in the Session Configuration
<code>natsession</code>	Session ID
<code>natserver</code>	Host name
<code>natport</code>	Port number
<code>natuser</code>	User name
<code>natprog</code>	Application
<code>natparam</code>	Natural parameters
<code>natparamext</code>	Natural parameters The URL parameter <code>natparamext</code> extends an existing Natural parameter definition in the configuration file. The extension works in the following way: the Natural parameters defined in the configuration file come first. Then, the Natural parameters defined in the URL parameter <code>natparamext</code> are added, separated by a space character. If you want to overrule the definition in the configuration file, use the URL parameter <code>natparam</code> instead.
<code>nattimeout</code>	Timeout (n seconds)



Important: All parameter values must be URL-encoded.

Example: In order to start the Natural program `dump`, while your application server is running on `myhost:8080` and your Natural Web I/O Interface server is running on `myserver1:4811`, you can use the following URL:

`http://myhost:8080/natuniweb/natural.jsp?natserver=myserver1&natport=4811&natprog=dump&natuser=my-username`

24

Configuring Container-Managed Security

- General Information 174
- Name and Location of the Configuration File 174
- Activating Security 174
- Defining Security Constraints 175
- Defining Roles 175
- Selecting the Authentication Method 176
- Configuring the UserDatabaseRealm 176

General Information

The Natural Web I/O Interface client comes as a Java EE-based application. For the ease of installation, the access to this application is by default not secured. You might, however, wish to restrict the access to certain parts of the application to certain users. An important example is the **configuration tool**, which enables you to modify the Natural session definitions and the logging configuration of the Natural Web I/O Interface client. Another example is the Natural logon page.

This section does not cover the concepts of JAAS-based security in full extent. It provides, however, sufficient information to activate the preconfigured security settings of the Natural Web I/O Interface client and to adapt them to your requirements.

Name and Location of the Configuration File

Security is configured in the file *web.xml*. This file is located in the following directory:

```
<tomcat-install-dir>/webapps/natuniweb/WEB-INF
```

Activating Security

Great care must be taken when editing and changing the configuration file *web.xml*. After a change, the application server must be restarted.

Edit the file *web.xml* and look for the section that is commented with "Uncomment the next lines to add security constraints and roles.". Uncomment this section by removing the comment marks shown in boldface below:

```
<!-- Uncomment the next lines to add security constraints and roles. -->
<!--
<security-constraint>
  <web-resource-collection>
    <web-resource-name>Configuration Tool</web-resource-name>
    <url-pattern>/conf_index.jsp</url-pattern>
    <url-pattern>/faces/*</url-pattern>
  </web-resource-collection>
  ...
<security-role>
  <description>Administrator</description>
  <role-name>nwoadmin</role-name>
</security-role>
-->
```

Defining Security Constraints

The security constraints defined by default are just examples. A `<security-constraint>` element contains a number of `<web-resource-collection>` elements combined with an `<auth-constraint>` element. The `<auth-constraint>` element contains a `<role-name>`. The whole `<security-constraint>` element describes which roles have access to the specified resources.

Example - the following definition specifies that only users in the role "nwoadmin" have access to the configuration tool:

```
<security-constraint>
  <web-resource-collection>
    <web-resource-name>Configuration Tool</web-resource-name>
    <url-pattern>/conf_index.jsp</url-pattern>
    <url-pattern>/faces/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>nwoadmin</role-name>
  </auth-constraint>
</security-constraint>
```

In the following section, you will see where and how the roles are defined.

Defining Roles

A few lines below in the file *web.xml*, there is a section `<security-role>`. Here, the roles that can be used in `<security-constraint>` elements are defined. You can define additional roles as needed. The assignment of users to roles is done outside this file and will often be done in a user management that is already established at your site.

Example:

```
<security-role>
  <description>Administrator</description>
  <role-name>nwoadmin</role-name>
</security-role>
```

Selecting the Authentication Method

In the file *web.xml*, there is a section `<login-config>`. The only element that should possibly be adapted here is `<auth-method>`. You can choose between the authentication methods "FORM" and "BASIC". Form-based authentication displays a specific page on which users who try to access a restricted resource can authenticate themselves. Basic authentication advises the web browser to retrieve the user credentials with its own dialog box.

Example:

```
<login-config>
  <auth-method>FORM</auth-method>
  ...
</login-config>
```

Configuring the UserDatabaseRealm

In the *tomcat-users.xml* file (which is located in the *conf* directory), specify the role "nwoadmin" for any desired user name and password. For example:

```
<user username="pepe" password="pepe123" roles="nwoadmin"/>
```

For detailed information on the necessary realm configuration for Tomcat, see <http://tomcat.apache.org/tomcat-6.0-doc/realms-howto.html#UserDatabaseRealm>.

25

Configuring SSL

- General Information 178
- Creating Your Own Trust File 178
- Defining SSL Usage in the Configuration File 179

General Information

Trust files are used for a secure connection between the Natural Web I/O Interface server and the Natural Web I/O Interface client. Server authentication cannot be switched off. A trust file is always required.

A trust file contains the certificates that you trust. These can be certificates of a CA (Certificate Authority) such as VeriSign, or self-signed certificates.

For information on the steps that are required on the Natural Web I/O Interface server and how to generate a self-signed certificate which needs to be imported to the client, see [SSL Support](#).

To establish a secure connection, you have to proceed as described in the topics below.

Creating Your Own Trust File

To create your own trust file, you can use, for example, Sun's keytool utility which can be found in the *bin* directory of the Java Runtime Environment (JRE). Here are some helpful examples:

- Create an empty, password-protected trust file:

```
keytool -genkey -alias foo -keystore truststore.jks -storepass "your-password"  
keytool -delete -alias foo -keystore truststore.jks
```

- Import a certificate:

```
keytool -import -alias "name-for-ca" -keystore truststore.jks -storepass ←  
"your-password" -file server.cert.crt
```

You should use a meaningful name for the alias.

- List the certificates in a trust file:

```
keytool -list -v -keystore truststore.jks
```

- Delete a certificate from a trust file:

```
keytool -delete -alias "name-for-ca" -keystore truststore.jks
```

When you modify the trust file or its password, you have to restart the application server so that your modification takes effect.

Defining SSL Usage in the Configuration File

Invoke the [configuration tool](#) and proceed as follows:

1. In the global settings for all defined sessions, define the **SSL trust file path** and, if required, the **SSL trust file password**. See also [Global Settings](#) in *Natural Client Configuration Tool*.

With the server authentication, the Natural Web I/O Interface client checks whether the certificate of the Natural Web I/O Interface server is known. If it is not known, the connection is rejected.

When a trust file is not defined in the configuration tool, the Natural Web I/O Interface client tries to read the file *calist* from the *lib/security* directory of the Java Runtime Environment (JRE). The default password for this file is "changeit".

2. Define a session and set the session option **Use SSL** to **Yes**. See also [Overview of Session Options](#) in *Natural Client Configuration Tool*.

26 Logging

- General Information 182
- Name and Location of the Configuration File 182
- Invoking the Logging Configuration Page 182
- Overview of Options for the Output File 184

General Information

The Natural Web I/O Interface client uses the Java Logging API. In case of problems with the Natural Web I/O Interface client, you can enable logging and thus write the logging information to an output file. This should only be done when requested by Software AG support.

You configure logging using the [configuration tool](#).



Note: Some logging information is also written to the console, regardless of the settings in the configuration file. The console shows the information which is normally provided by the logging levels SEVERE, WARNING and INFO.

Name and Location of the Configuration File

The name of the configuration file is *natlogger.xml*, which is located in:

`<application-server-install-dir>server/default/deploy/naturalunicode.rar/log`

Invoking the Logging Configuration Page

The content of the configuration file *natlogger.xml* is managed using the **Logging Configuration** page of the [configuration tool](#).

➤ **To invoke the Logging Configuration page**

- 1 In the frame on the left, choose the **Logging Configuration** link.

The **Logging Configuration** page appears in the right frame. Example:

Logging Configuration

Specify the output log file characteristics.

- "/" : The local pathname separator
- "%t": The system temporary directory
- "%h": The value of the "user.home" system property
- "%g": The generation number to distinguish rotated logs
- "%u": A unique number to resolve conflicts
- "%%": Translates to a single percent sign "%"

File pattern name:

File type:

File size (in Kbytes; 0=unlimited):

Number of files:

File enabled: Yes No

Append mode: Yes No

Specify log levels for individual modules. The available settings are:

- SEVERE: Events that interfere with normal program execution
- WARNING: Warnings, including exceptions
- INFO: Messages related to server configuration or server status, excluding errors
- CONFIG: Messages related to server configuration
- FINE: Minimal verbosity
- FINER: Moderate verbosity
- FINEST: Maximum verbosity

Communication:

Resource adapter:

Session beans:

Message beans:

Configuration file:

Logging:

Utilities:

Natural Web I/O Interface pages:

- 2 Specify the characteristics of the output file as described below in the section [Overview of Options for the Output File](#).
- 3 Specify the log levels for individual modules by selecting the log level from the corresponding drop-down list box.

A brief description for each log level is provided on the **Logging Configuration** page.

- 4 Choose the **Save Configuration** button to write the modifications to the configuration file.



Caution: When you do not choose the **Save Configuration** button but log out instead or leave the configuration tool by entering another URL, your modifications are not written to the configuration file.

Overview of Options for the Output File

The following options are provided for specifying the characteristics of the output file:

Option	Description
File pattern name	<p>The pattern for generating the output file name. Default: "%h/nwolog%g.log".</p> <p>The default value means that an output file with the name <i>nwolog<number>.log</i> will be created in the home directory of the user who has started the application server.</p> <p>For detailed information on how to specify the pattern, see the Java API documentation.</p>
File type	<p>The format of the output file. Select one of the following entries from the drop-down list box:</p> <ul style="list-style-type: none"> ■ Text format Output in simple text format (default). ■ XML format Output in XML format. <p>The corresponding formatter class is then used.</p>
File size	<p>The maximum number of bytes that is to be written to an output file. Zero (0) means that there is no limit. Default: "0".</p>
Number of files	<p>The number of output files to be used. This value must be at least "1". Default: "10".</p>
File enabled	<p>If set to Yes (default), the file handler is enabled. If set to No, the file handler is disabled.</p>
Append mode	<p>If set to Yes, the logging information is appended to the existing output file. If set to No (default), the logging information is written to a new output file.</p>

VII Operating and Monitoring the Natural Web I/O Interface

Server

This part covers the following topics:

[Operating the Natural Web I/O Interface Server](#)

[Monitor Client NATMOPI](#)

[HTML Monitor Client](#)

Notation *vrs* or *vr*

When used in this documentation, the notation *vrs* or *vr* represents the relevant product version (see also *Version* in the *Glossary*).

27

Operating the Natural Web I/O Interface Server

- Starting the Natural Web I/O Interface Server 188
- Terminating the Natural Web I/O Interface Server under z/OS, z/VSE 190
- Terminating the Natural Web I/O Interface Server under BS2000 190
- Changing the SYSOUT File Assignment of the FSIO Task under BS2000 191
- Monitoring the Natural Web I/O Interface Server 191
- Runtime Trace Facility 193
- Trace Filter 194

This chapter describes how to operate a Natural Web I/O Interface server. Unless otherwise noted, the information below applies to all operating systems.

Starting the Natural Web I/O Interface Server

Under z/OS:

The Web I/O Interface server can be started as a “started task”:

```
//NWOSRV  PROC
//KSPSRV  EXEC PGM=NATRWO,REGION=4000K,TIME=1440,
// PARM=('POSIX(ON)/NWOSRV1')
//STEPLIB DD DISP=SHR,DSN=NWOvrs.LOAD
//        DD DISP=SHR,DSN=NATvrs.LOAD
//CMPRINT DD SYSOUT=X
//STGCONF DD DISP=SHR,DSN=NWOvrs.CONFIG(SRV1)
//STGTRACE DD SYSOUT=X
//STGSTDO DD SYSOUT=X
//STGSTDE DD SYSOUT=X
```

where *vrs* represents the relevant product version of NWO or Natural.



Note: PARM=('POSIX(ON)/NWOSRV1') - POSIX(ON) is required for a proper LE370 initialization, and NWOSRV1 is the name of the server for the communication with the monitor client.

The name of the started task must be defined under RACF and the z/OS UNIX System Services.

Under z/VSE:

Under z/VSE, a prerequisite is a running SMARTS address space that is configured to run the Natural Web I/O Interface server (see *Installing the Web I/O Server under z/VSE*).

```
<msg-id> NATRWO <server-id>
```

- where

msg-id is the message identifier assigned to the SMARTS partition, and

server-id is the name of your Natural Web I/O Interface server.

Example for z/VSE:

```
141 NATRNWO NWOS1
```



Note: If you qualify the Natural Web I/O Interface server data sets by *server-id*, the server ID is restricted to a maximum length of 6 characters.

Alternatively you can automatically start Natural Web I/O Interface servers during SMARTS initialization by using the SMARTS SYSPARM parameter *STARTUPPGM*. In the SMARTS SYSPARM file specify:

```
STARTUPPGM='NATRNO <server-id>'
```

Example:

```
STARTUPPGM='NATRNO NWOS1'
```

Under BS2000:

Under BS2000, start the Natural Web I/O Interface server with the SDF command:

```
/ENT-PROCSTART-NWO
```

The SDF procedure *START_NWO* has to be supplied with the following parameters:

Parameter	Definition	Default Value
NWO-JOBS	The NWO (SMA) job library.	NW0vrs.JOBS
ENV-MOD	The NWO environment-specific module library. This library contains the linked Natural nucleus module.	NW0vrs.JOBS
NWO-MOD	The NWO module library.	\$SAG.NW0vrs.MOD
NCF-MOD	The Natural Com-plete interface module library.	\$SAG.NCFvrs.MOD
APS-LIB	The SMARTS library (modules and procedures).	\$SAG.APSvrs.LIB
ADA-MOD	The Adabas module library.	ADAvrs.MOD
PROC-NAME	The name of the NWO <i>START</i> procedure. The procedure must reside in the NWO-JOBS library.	START-NWO
NWO-CONFIG	The NWO configuration file. It must reside in the NWO-JOBS library (Type S).	NWO-CONFIG
NWO-SYSPARM	The SMARTS configuration file. It must reside in the NWO-JOBS library.	NWO-SYSPARM
NWO-ADAPARM	The ADALNK parameter file (IDTNAME, etc). It must reside in the NWO-JOBS library.	NWO-ADAPARM
LOG-FILE-PREFIX	The log-file prefix for the SYSOUT files of all SMARTS tasks.	L.NWO.
WORKER-JOB-NAME	The job name of the worker-tasks.	NWOWORK

Parameter	Definition	Default Value
WORKER-JOB-CLASS	The job class of the worker-tasks.	*STD
WORKER-CPU-LIMIT	The CPU time limit for the SMARTS worker tasks.	*NO
LOGGING	Switches logging for diagnostic purposes.	*NO
MAIN-TASK	For internal use only. Do not modify!	

 **Caution:** Do not modify the variable names and parameter names that are used in the procedure `START_NWO`. This procedure is called recursively to attach the worker-tasks. For this purpose, the `ENTERPARM` string is internally executed and several variables are read from the system, using the `GETVAR` function of `SDF-P`.

Example procedure for entering the `START-NWO` procedure:

```
/ENT-PROC      ($SAG.APSvrs.LIB,START-NWO,J),(
/              NWO-JOBS      = $SAG.NWOvrs.JOBS,
/              NWO-MOD       = $SAG.NWOvrs.MOD,
/              NCF-MOD       = $SAG.NCFvrs.MOD,
/              APS-LIB       = $SAG.APSvrs.LIB,
/              ADA-MOD       = $SAG.ADAvrs.MOD.NWOvr,
/              NWO-SYSPARM   = NWO-SYSPARM,
/              NWO-CONFIG    = NWO-CONFIG,
/              NWO-ADAPARM   = NWO-DDLNKPAR,
/              LOG-FILE-PREFIX = L.NWO.OUT.NWOS01.)
```

Terminating the Natural Web I/O Interface Server under z/OS, z/VSE

Under z/OS and z/VSE, the Natural Web I/O Interface server can be terminated from within the Monitor Client `NATMOPI`, see [Monitor Commands](#) below.

Terminating the Natural Web I/O Interface Server under BS2000

The Natural Web I/O Interface server can be terminated with the console command:

```
/INTR <TSN smarts-main-task>,EOJ
```

Changing the SYSOUT File Assignment of the FSIO Task under BS2000

The central logical system file `SYSOUT` written by the `FSIO` task can be reassigned at SMARTS server execution time, using the SDF procedure `SHOW-SYSOUT`.

Thus it is possible to look up trace outputs, error reports, etc., without having to terminate the server.

The procedure `SHOW-SYSOUT` has to be called with the following parameters:

APS-LIB	The SMARTS module library.
TSN	The TSN of the SMARTS server main-task.

As a result of the `SHOW-SYSOUT` execution, the logical system file `SYSOUT` of the `FSIO` task will be reassigned to a new file and the previous one will be opened with the `SHOW-FILE` command. The new logical system file `SYSOUT` is built by appending a numerical suffix to the file name. The value of the suffix is incremented by 1, each time `SHOW-SYSOUT` is executed.

Example:

```
/CLP FROM-FILE=*LIBRARY-ELEMENT(LIBRARY=APSvrs.LIB,ELEMENT=
SHOW-SYSOUT),PROCEDURE-PARAMETERS=(APS-IB=APSvrs.LIB,TSN=7445),
LOGGING=*PARAMETERS"
```

Monitoring the Natural Web I/O Interface Server

To enable the administrator to monitor the status of the Natural Web I/O Interface server, a monitor task is provided which is initialized automatically at server startup. Using the monitor commands described below, the administrator can control the server activities, cancel particular user sessions, terminate the entire server, etc.

The following topics are covered below:

- [Monitor Communication](#)

- [Monitor Commands](#)

Monitor Communication

To communicate with the monitor, you can use the monitor client `NATMOPI`; see *Monitor Client NATMOPI*. Or you can use the HTML Monitor Client that supports standard web browser, see *HTML Monitor Client*.

Under z/OS, you can alternatively use the operator command `MODIFY` to execute the monitor commands described below in the section *Monitor Commands*. The output of the executed monitor command will be written to the system log.

Example:

```
F jobname,APPL=ping
```

sends the command `ping` to the Web I/O Interface server running under the job `jobname`.

Monitor Commands

The Natural Web I/O Interface server supports the following monitor commands:

Monitor Command	Action
<code>ping</code>	Verifies whether the server is active. The server responds and sends the string <code>I'm still up</code>
<code>terminate</code>	Terminates the server.
<code>abort</code>	Terminates the server immediately without releasing any resources.
<code>set configvariable value</code>	With the <code>set</code> command, you can modify server configuration settings. For example, to modify <code>TRACE_LEVEL</code> : <code>set TRACE_LEVEL 31+30+15</code>
<code>list sessions</code>	Returns a list of active Natural sessions within the server. For each session, the server returns information about the user who owns the session, the session initialization time, the last activity time and an internal session identifier (<i>session-id</i>).
<code>cancel session session-id</code>	Cancels a specific Natural session within the Natural Web I/O Interface server. To obtain the session ID, use the monitor command <code>list sessions</code> . Canceling active batch server sessions in Natural requires authorized services provided by the Authorized Services Manager (ASM). If the ASM is not started, those sessions cannot be canceled.
<code>cleanup</code>	Cancel sessions that are inactive longer than specified in configuration parameter <code>SESSION_TIMEOUT</code> .

Monitor Command	Action
help	Returns help information about the monitor commands supported.

Runtime Trace Facility

For debugging purposes, the server code has a built-in trace facility which can be switched on, if desired.

The following topics are covered below:

- [Trace Medium](#)
- [Trace Configuration](#)
- [Trace Level](#)

Trace Medium

Under z/OS and z/VSE, the Natural Web I/O Interface server writes its runtime trace to the logical system file `SYSOUT` of the `FSIO` task.

Under z/OS, z/VSE and BS2000, the Natural Web I/O Interface server writes its runtime trace to the logical system file `SYSOUT` of the `FSIO` task.

Trace Configuration

The trace is configured by a [trace level](#) which defines the details of the trace. Once a trace is switched on, it can be restricted to particular clients or client requests by specifying a [trace filter](#), see also Web I/O Interface server configuration parameter `TRACE_FILTER`.

Every Natural session is provided with a 32-bit trace status word (TSW) which defines the trace level for this session. The value of the TSW is set in the Web I/O Interface server configuration parameter `TRACE_LEVEL`. A value of zero means that the trace is switched off.

Trace Level

Each bit of the TSW is responsible for certain trace information. Starting with the rightmost bit:

Bit 31	Trace main events (server initialization/termination, client request/result).
Bit 30	Detailed functions (session allocation, rollin/rollout calls, detailed request processing).
Bit 29	Dump internal storage areas.
Bit 28	Session directory access.
Bit 27	Dump send/reply buffer.
Bit 26	Dump send/reply buffer short. Only the first 64 bytes are dumped.

Bit 25	Dump I/O buffer.
Bit 24	Dump I/O buffer short. Only the first 64 bytes are dumped.
Bit 23	Call back gateway event.
Bit 22-17	Free.
Bit 15	Trace error situations only.
Bit 14	Apply trace filter definitions.
Bit 13	Trace start and termination of the server only.
Bit 12	Trace start and termination of the client sessions only. Even if bit 13 is set.
Bit 11-01	Free.
Bit 00	Reserved for trace-level extension.



Note: Using trace levels 12 and 13 is only possible with Natural Web I/O Interface Server Version 8.3.2.

Trace Filter

It is possible to restrict the trace by a logical filter in order to reduce the volume of the server trace output.

- The filter can be set with the configuration parameter `TRACE_FILTER`.
- The filter may consist of multiple `keyword=filtervalue` assignments separated by spaces.
- To activate the filter definition, the trace bit 14 in the trace status word (see *Trace Level*) must be set.

The filter keyword is:

<code>Client</code>	Filters the trace output by specific clients.
---------------------	---

The following rules apply:

- If a keyword is defined multiple times, the values are cumulated.
- The value must be enclosed in braces and can be a list of filter values separated by spaces.
- The values are not case sensitive.
- Asterisk notation is possible.

Example:

```
TRACE_FILTER="Client=(KSP P*)"
```

Each request of the user ID "KSP" and each request of the user IDs starting with a "P" are traced.

28

Monitor Client NATMOPI

- Introduction 198
- Prerequisites for NATMOPI Execution on BS2000 198
- Command Interface Syntax 198
- Command Options Available 199
- Monitor Commands 199
- Directory Commands 199
- Command Examples 200

Introduction

The Monitor Client NATMOPI is a character-based command interface for monitoring the various types of servers that are provided in a mainframe Natural environment. Each of these servers has its own set of monitor commands which is described in the corresponding server documentation. In addition, a set of directory commands is available which can be used independent of the server type. One NATMOPI can be used to monitor different server types.

Prerequisites for NATMOPI Execution on BS2000

Execute NATMOPI with the following SMARTS console command:

```
/INTR <SMARTS-tsn>,NATMOPI <mopi-command>
```

where *SMARTS-*tsn** is the TSN of your SMARTS main task.

Example:

```
/INTR 4711,NATMOPI -dls
```

The output is written to the SYSOUT file of the FSI0 task.



Note: The server to be monitored must be running in the same SMARTS environment as NATMOPI.

Command Interface Syntax

Basically the syntax of the command interface consists of a list of options where each option can/must have a value. For example:

```
-s <server-id> -c help
```

where *-s* and *-c* are options and *<server-id>* and *help* are the option values.

It is possible to specify multiple options, but each option can have only one value assigned.

The command options available are listed below.

Command Options Available

Words enclosed in $\langle \rangle$ are user supplied values.

Command Option	Action
-s $\langle server-id \rangle$	Specify a server ID for sending a monitor command . If the server ID is not unique in the server directory, NATMOPI prompts the user to select a server.
-c $\langle monitor command \rangle$	Specify a monitor command to be sent to the server ID defined with the -s option
-d $\langle directory command \rangle$	Specify a directory command to be executed.
-a	Suppress prompting for ambiguous server ID. Process all servers which apply to the specified server ID.
-h	Print NATMOPI help.

Monitor Commands

These are commands that are sent to a server for execution. The monitor commands available depend on the type of server, however, each server is able to support at least the commands `ping`, `terminate` and `help`. For further commands, refer to [Operating the Web I/O Interface Server](#) where the corresponding server commands are described.

Directory Commands

Directory commands are not executed by a server, but directly by the monitor client NATMOPI.

You can use the directory commands to browse through the existing server entries and to remove stuck entries.

The following directory commands are available. Words enclosed in $\langle \rangle$ are user supplied values and words enclosed in [] are optional.

Directory Command	Action
ls [$\langle server-id \rangle$]	List all servers from the server directory that apply to the specified server ID. The server list is in short form.
ll [$\langle server-id \rangle$]	Same as <code>ls</code> , but the server list contains extended server information.

Directory Command	Action
rs [<i><server-id></i>]	Remove server entries from server directory. Note: If you remove the entry of an active server, you will lose the ability to monitor this server process.
cl [<i><server-id></i>]	Clean up server directory. This command pings the specified server. If the server does not respond, its entry will be removed from the directory.
ds	Dump the content of the server directory.
lm	List pending IPC messages.

Command Examples

Example: Ping a Server in Different Environments

Server in z/OS (SMARTS/Com-plete), output in STDOUT:

- System operator:

```
/F complete-jobname,NATMOPI -sServerName -cPING
```

- Com-plete online command:

```
NATMOPI -sServerName -cPING
```

Server in z/OS (started task or batch mode):

- Execute NATMOPI in batch job:

```
NATMOPI,PARM=(' -sServerName -cPING')
```

Sample job:

```
//SAGMOPI JOB SAG,CLASS=K,MSGCLASS=X
//NATEX EXEC PGM=NATMOPI,REGION=3000K,
// PARM=(' -Sname -CPING')
//* PARM=(' -H')
//STEPLIB DD DISP=SHR,DSN=NATURAL.XXXvr.LE.LOAD
// DD DISP=SHR,DSN=CEE.SCEERUN
//SYSOUT DD SYSOUT=X
//SYSPRINT DD SYSOUT=X
//*
```

Where *XXX* is the Web I/O Interface server product code (NWO) and *vr* is the two-digit product version.

- Execute NATMOPI in TSO (Command):

```
NATMOPI -sServerName -cPING
```

The NWO load library must be included in the steplib of TSO.

Server in z/VSE (SMARTS/Complete):

- System operator:

```
(vse-replid) NATMOPI -sServerName -cPING
```

- Com-plete online command:

```
NATMOPI -sServerName -cPING
```

Further Command Examples:

<code>natmopi -dls</code>	List all servers registered in the directory in short format.
<code>natmopi -dcl TST -l s TST</code>	Clean up all servers with ID TST* (ping server and remove it, if it does not respond), and list all servers with ID TST* after cleanup.
<code>natmopi -sSRV1 -cping -sSRV2 ↵ -sSRV3 -cterminate</code>	Send command ping to SRV1. Send command terminate to SRV2 and SRV3.
<code>natmopi -cterminate -sSRV1 ↵ -cping -sSRV2 -sSRV3</code>	Is equivalent to the previous example. That is, NATMOPI sends the command following the -s option to the server. If no -c option follows the -s option, the first -c option from the command line will be used.
<code>natmopi -sSRV1 -cterminate -a</code>	Send command terminate to SRV1. If SRV1 is ambiguous in the server directory, send the command to all SRV1 servers without prompting for selection.

29 HTML Monitor Client

- Introduction 204
- Prerequisites for HTML Monitor Client 204
- Server List 204
- Server Monitor 206

Introduction

The HTML Monitor Client is a monitor interface that supports any web browser as a user interface for monitoring the various types of servers that are provided in a mainframe Natural environment. Each of these servers has its own set of monitor commands which are described in the corresponding server documentation. The HTML Monitor Client enables you to list all existing servers and to select a server for monitoring.

Prerequisites for HTML Monitor Client

To run the HTML Monitor Client, any server must host an HTTP Monitor Server. The HTTP Monitor Server is a subtask that can run in any Web I/O Interface server address space. It is configured with the NWO server configuration parameter `HTPMON_PORT` and `HTPMON_ADMIN_PSW`. An HTTP Monitor Server is accessible through a TCP/IP port number and can monitor all servers running on the current node (for SMARTS: running within the current SMARTS). Although it is not necessary, you can run multiple HTTP Monitor Servers on one node. But each one needs an exclusive port number.

Server List

Open your web browser and connect the HTTP Monitor Server using the following URL:
`http://nodename:port`, where *nodename* is the name of the host on which the Natural Development Server hosting the monitor is running. And *port* is the port number the administrator has assigned as the monitor port in the NDV configuration file.

Example

Natural Server List

Refresh

	Server ID	Pid	Started	Config Parameters	Session Parameters
NWO <input type="button" value="Select"/>	QAWO4841	197	2017/08/27 20:04:19	PORT_NUMBER = 4841 FRONTEND_OPTIONS = 0X01 FRONTEND_NAME = NATCSRFE TRACE_LEVEL = 30+31+15+13+12	Version:NAT8206 TMODEL=2 SORT=(EXT=OFF) DBGAT=(ACTIVE=OFF,TRACE=00000097) RCA=(NATATDBG) RCALIAS=(NATATDBG,NCIAD833) Connection = SOC:daef:4841 Security = No
NWO <input type="button" value="Select"/>	QAWO4842	198	2017/08/27 20:04:29	PORT_NUMBER = 4842 FRONTEND_OPTIONS = 0X01 FRONTEND_NAME = NATBAT82 TRACE_LEVEL = 31+30+15+12	Version:NAT8206 NUCNAME=QA82NRE TMODEL=2 IM=F SORT=(EXT=OFF) DBGAT= (ACTIVE=OFF,TRACE=00000097) ZIIP=OFF RCA=SAGICU RCA=ICSdT54X CFICU=(ON,DATFILE=ICSdT54X)RCA=(NATATDBG) RCALIAS=(NATATDBG,NATAD834) Connection = SOC:daef:4842 Security = No
NWO <input type="button" value="Select"/>	QAWO4843	199	2017/08/27 20:04:37	PORT_NUMBER = 4843 FRONTEND_OPTIONS = 0X01 FRONTEND_NAME = NATISRFE TRACE_LEVEL = 15+30	Version:NAT8206 TMODEL=2 SORT=(EXT=OFF) Connection = SOC:daef:4843 Security = No

The server list consists of green and red entries. The red ones represent potentially dead server entries which can be deleted from the server directory by choosing the attached **Remove** button. The **Remove** button appears only for the red entries. “Potentially dead” means, that the HTTP Monitor Server “pinged” the server while assembling the server list, but the server did not answer within a 10 seconds timeout. Thus, even if you find a server entry marked red, it still might be active but could not respond to the ping. Choosing the **Remove** button does not terminate such a server but removes its reference in the monitor directory. Hence, it cannot be reached by the monitor anymore.

Choosing the **Select** button opens a window for monitoring the selected server.

Server Monitor

Monitor server QAWO4842 198

Ping

Terminate

Abort

ListSess

CancelSession

Configuration

Flush

Cleanup

Please press command key

With the buttons, you can perform the labeled monitor commands.

The selection box allows you to modify the server configuration parameters. If you select a parameter for modification, it has a predefined value. This predefined value does not reflect the setting of the server. It is just a sample value.

If you choose the **ListSess** button, a list of all Natural sessions appears in the window, for example:

Monitor server QAWO4842 198

Ping

Terminate

Abort

ListSess

CancelSession

Configuration

Flush

Cleanup

Reply for server pid 44:

	UserId	SessionId	InitTime	LastActivity	St
1	CF	D2ECEC17EFD90D46	02 07:24:01	02 10:19:32	I
2	QFSTEST	D2ECCDCEC74B4142	02 05:08:31	02 05:40:08	I
3	QFSTEST	D2ECCDB50A4CC242	02 05:08:04	02 05:18:10	I
4	QFSTEST	D2ECB50CA40AFF43	02 03:17:45	02 03:23:08	I
5	QFSUSER	D2ECBF3C8AEC6344	02 04:03:20	02 04:41:43	I
6	QFSUSER	D2ECBBFC020A4B43	02 03:48:47	02 03:52:22	I
7	QFTTEST	D2ECEB1DB7DA7C43	02 07:19:39	02 07:23:36	I
8	STARGATE	D2EC53BA2E7B8A46	01 20:02:21	01 20:02:23	

You can cancel sessions by selecting the session ID in the **SessionId** column and choosing the **CancelSession** button.

Index

A

application
start with URL, 171

C

configuration tool, 147

D

differences
Natural Web I/O Interface, 13

L

logging, 181
logon page
Natural Web I/O Interface, 139

R

restrictions
Natural Web I/O Interface, 11

S

SSL, 177

T

trust files, 177

U

URL to start application, 171

W

Web I/O Interface (see set up client)
configuration tool, 147
differences, 13
logon page, 139
restrictions, 11

