

Natural Business Services

Common Model Specifications and Development Tasks

Version 5.3.1

February 2010

This document applies to Natural Business Services Version 5.3.1.

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

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

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

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

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

Table of Contents

1 Common Model Specifications and Development Tasks	1
2 Common Fields on the Standard Parameters Panel	3
3 Set Up a Password File	7
4 Change the Default Window Settings	9
Test the Modified Window Settings	11
5 Determine Which Condition Codes are Set	13

1 Common Model Specifications and Development Tasks





Common Model Specifications and Development Tasks describes the common specifications for the Natural Construct models and how to perform common development tasks.



Notes:

1. Although the screen examples used in this documentation are from a mainframe environment, the information applies to all server environments.
2. For information about the additional specification panels for a specific model, see the documentation in which that model is described.

Common Model Specifications and Development Tasks covers the following topics:

 Common Fields on the Standard Parameters Panel	Describes the common fields on the Standard Parameters panel.
 Set Up a Password File	Describes how to set up a password file to implement password checking for generated modules.
 Change the Default Window Settings	Describes how to change the default window settings for your generated modules, such as the size and position of the window and whether it uses a frame (border). Includes information on testing the new window settings.
 Determine Which Condition Codes are Set	Explains how to verify which condition codes have been set for your generated module.

2 Common Fields on the Standard Parameters Panel

The Standard Parameters panel is similar for all program models; it is the first and sometimes the only specification panel. The following example shows the Standard Parameters panel for the Object-Maint-Dialog model:

```
CUOMMA                               Object-Maint-Dialog Program          CU--MA0
Sep 16                               Standard Parameters                    1 of 4

Module ..... _____
System ..... CST52S_____
Global data area ... CDGDA___ *
With block ..... _____

Title ..... Object Dialog..._____
Description ..... This program is used to maintain the..._____
_____
_____
_____

First header ..... _____
Second header ..... _____

Command ..... _
Message numbers .... _
Password ..... _

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
right help  retrn quit                               right main
```



Note: The Standard Parameters panel for other models may not contain all the fields shown.

The fields on the Standard Parameters panel are:

Field	Description
Module	<p>Name of the module (by default, the name specified in the Module name field on the Generation main menu). This is a required field.</p> <p>The module name must be alphanumeric, a maximum of eight characters in length, and cannot contain blanks.</p>
System	<p>Name of the system (by default, the name of the current library). This is a required field.</p> <p>The system name must be alphanumeric, not exceed 32 characters in length, and does not have to be associated with a Natural library ID. (The combination of the module name and system name is used as a key to access help information for the generated module.)</p>
Global data area	<p>Name of the global data area (GDA) used by the generated module. By default, "CDGDA" is displayed. This is a required field.</p> <p>To allow inter-program communication, generated modules require a small number of global variables. The CDGDA global data area contains the global variables required to test a generated module.</p> <p>Before creating a new application, copy this GDA from the SYSTEM library and rename it to match your naming conventions. Then add any additional global variables your application may require.</p>
With block	<p>Name of the GDA block used by the generated module. This is an optional field.</p> <p>To use a GDA block, you need only specify the lowest level block name; the corresponding path name is determined automatically. For more information about GDA blocks, see the Natural documentation.</p>
Title	<p>Title for the generated module. The title is used internally to identify modules for the List Generated Modules function on the Generation main menu. This is a required field.</p>
Description	<p>Brief description of what the generated module does. This is an optional field and is used internally for documentation purposes.</p>
First header	<p>First header (heading) for the generated module. This header is centered at the top of the generated panel and intensified. This is a required field.</p>
Second header	<p>Second header (heading) for the generated module. This header is centered under the first header and intensified. This is an optional field.</p>
Command	<p>If this field is marked, the generated module supports direct command processing. If this field is blank, the generated module does not support direct commands.</p>
Message numbers	<p>If this field is marked, the generated module uses message numbers rather than message text for all REINPUT and INPUT messages.</p> <p>Note: Use the same technique consistently throughout your application, since passing messages between modules using different techniques will not always produce the desired results.</p>

Field	Description
Password	<p data-bbox="464 243 1471 306">If this field is marked, the generated module performs password checking. If this field is blank, the generated module does not perform password checking.</p> <p data-bbox="464 338 1471 401">To include password checking, you must set up a password file. For information, see Set Up a Password File.</p>

3 Set Up a Password File

You can specify password checking for many of the generated modules. Natural Construct builds the mechanism for password checking into your module by including the CCPASSW copycode member. Within this copycode, the CDPASSW subprogram is invoked and passed the module and system names.

To include password checking, you must set up a password file. The file is keyed on the module name used to catalog the module and the system name used to generate the module.

The password file can be a view of any file with Natural-Construct-Password as the data definition module name. The view must contain the following fields:

Field	Format
PASSWORD-KEY	A40 (32-character system name, plus an 8-character module name)
PASSWORD	A8 (8-character password)

When a user attempts to invoke the module, the CDPASSW subprogram reads the password file. If the module/system name combination exists in the file and does not have a password, the user can invoke the module. If the module/system name combination exists and has a password, the user must enter the correct password before the module is invoked. If a user enters five incorrect passwords, execution is aborted.

If you specify password checking, you must modify the CDPASSW subprogram to include a valid password view and any final processing you want to perform and then catalog the modified subprogram. For more specific password checking, you can modify the CCPASSW copycode member (to call a different subprogram) or modify the CDPASSW subprogram (to refine your security standards).

4 Change the Default Window Settings

- Test the Modified Window Settings 11

▶ **To change the default window settings for your generated module:**

- 1 Press PF5 (windw).

The Window Parameters window is displayed. For example:

```

CU--DWM          Natural Construct          CU--DWM0
Oct 21           Window Parameters          1 of 1

      Size ..... Height ..... ___
                Width ..... ___

      Position .... Line ..... ___
                Column ..... ___

      Frame OFF .... _
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---
      help retrn quit test
    
```

- 2 Use the following fields to build the DEFINE WINDOW statement:

Field	Description
Size	<ul style="list-style-type: none"> ■ Height Number of lines the window spans. ■ Width Number of columns the window spans.
Position	<ul style="list-style-type: none"> ■ Line Number of lines from the top of the panel to the top of the window. ■ Column Number of columns from the left side of the panel to the left side of the window. The line and column values form the top left corner of the window.
Frame OFF	If this field is marked, the window is displayed without a border (frame).

- 3 Press Enter to confirm the changes.

Test the Modified Window Settings

You can view a test version of the window with the characteristics specified in the Window Parameters window.

▶ **To test the modified window settings:**

- Press PF4 (test) in the Window Parameters window.

5

Determine Which Condition Codes are Set

▶ **To determine which condition codes are set:**

- 1 Mark the Condition codes field in the Optional Parameters window.
- 2 Press PF2 (retrn).

The Generation main menu is displayed.

- 3 Generate the module.

After generation is complete, the Condition Codes Trace window is displayed, showing the condition codes set for the current module. For example:

CSGCTRAC	Natural Construct	CSGTRCO
Jan 13	Condition codes Trace	1 of 1
Condition codes	Condition codes	
-----	-----	
1 X MULTIPLE-ENTITIES	14 DB2-TIMESTAMP	
2 X KEY-NOT-DEFINED-AS-UQ	15 CONFINED-KEY-PREFIX	
3 KEY-IS-ALPHA	16 RESET-REDEFINES	
4 KEY-IS-LOGICAL	17 LOGGING-UPDATES	
5 HOLD-FIELD-EQ-TIME	18 DB2-REFERENTIAL-INTEGRITY	
6 X PRIME-FILE-IS-ADABAS	19 DB2-INTRA-CASCADE-DELETE	
7 PRIME-FILE-IS-DB2	20 KEY-IS-REDEFINED	
8 CLAUSE-IS-DB2-SPECIFIC	21 DERIVED-DATA	
9 PRIME-FILE-IS-IMS	22 NAT-REFACTORING	
10 PRIME-FILE-IS-VSAM	23 X GET-BY-ISN	
11 KEY-IS-A-SUPER	24 X HASH-LOCKING	
12 X USE-MSG-NR	25 NAT4-OR-HIGHER	
13 KEY-IS-PRIME		



Tip: If the condition code is not marked, ensure that the code is being generated in a Natural V4 (Natural V6 on open systems) or higher environment and that the option is set to True

in the CSXDEFLT subprogram. CSXDEFLT is installed in the SYSCSTX library. To use the code, compile the subprogram in the SYSCST library and then copy the object code to the SYSLIBS library. For information on CSXDEFLT, see Use CSXDEFLT Overrides.