

Natural Construct Version 3.3.3

Release Notes for OpenVMS

Manual Order Number: CST333-008VMS

This document applies to Natural Construct Version 3.3.3 for OpenVMS and to all subsequent releases.

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

Readers' comments are welcomed. Comments may be addressed to the Documentation Department at the address on the back cover or to the following e-mail address:

Documentation@softwareag.com

© July 1999, SAGA SOFTWARE, Inc. and Software AG

All rights reserved

Printed in the Federal Republic of Germany

Software AG and/or all Software AG products are either trademarks or registered trademarks of Software AG. Other products and company names mentioned herein may be the trademarks of their respective owners.

TABLE OF CONTENTS

RELEASE NOTES

New Features	6
Natural Compatibility	6
Action Helproutine Enhancements	6
Defining Default Specifications	7
Modifying or Creating Model PDAs	9
Support for Natural Construct Client	9
Support for the Map Model	9
Remote-Procedure-Call-Stub Model	10
Enhancements to Existing Features	11
CDKEYLDA Adjustment	11
CDFLIP Map Adjustments	11
Model Enhancements to Support SQL-Based DDMs	11
Regeneration of Object PDAs	12
Various Minor Corrections	12
Batch Model	12
Browse* Models	12
Driver Model	13
Maint Model	13
Menu Model	14
Object* Models	15
View Model	15
Multiple Models	15
Predict	15
Modules	16
Utilities	16
Editor/Editing Problems	16
Generation Problems	17
Miscellaneous	17

RELEASE NOTES

These release notes apply to Natural Construct V3.3.3 for OpenVMS. They contain information that is not included elsewhere in the Natural Construct documentation set.

The following topics are covered:

- **New Features**, page 6
- **Enhancements to Existing Features**, page 11

To install this product in your environment, see *Natural Construct Installation and Operations Manual for OpenVMS*.

New Features

The following features are included in the version 3.3.3 release of Natural Construct.

Natural Compatibility

Natural Construct V3.3.3 for OpenVMS is compatible with the following products:

- Natural for OpenVMS version 3.1.1 or higher
- Predict for OpenVMS version 3.4.2 or higher
- Adabas for OpenVMS 3.2.2 or higher

Action Helproutine Enhancements

A new Action helproutine, CDACTHL2, is introduced, which fully supports action code of up to three characters, as defined in the CDACTA parameter data area. Changes to the Maint, Object-Dialog-*, and Browse-Select-* models have been made to take advantage of this.

When Maint and Object-Dialog-* model programs that use inline maps are regenerated (or new ones created), the CDACTHL2 helproutine is generated into the programs in place of CDACTHLP. References to CDACTHLP from the Action fields on external maps can be replaced with CDACTHL2 at your discretion. CDACTHLP is still available to support existing applications.

CDACTHL2 is also generated into the Action columns of inline maps in Browse-Select-* model programs. CDACTHL2 can be attached to the Action columns of external maps at your discretion.

The default positioning of the #ACTION and #VAL-ACT variables on inline maps for Maint and Object-Dialog-* model programs was adjusted to provide room for potentially large lists of valid actions to expand freely. A new variable, #VAL-ACT2, holds this longer list of valid actions. References to #VAL-ACT on external maps can be replaced with #VAL-ACT2 at your discretion. #VAL-ACT is still generated into the programs to support existing applications.

The eight standard Action help text members keyed with a one-character action code as the minor component (as in F/NCST-ACTION/A, for example) were copied and keyed with a three-character minor component (for example, F/NCST-ACTION/ADD). The longer, three-character minor component provides greater scope for maintaining unique help text member keys as the number of actions defined in applications grows.

The original set of Action help text members is still supplied, and the original key structure is still used by CDACTHLP; the new key structure is used by CDACTHL2. If you want CDACTHL2 to find help text members for additional actions you define, you must save them with a three-character minor component that corresponds to the first three characters of the action name, as defined in CSTAPPL (in SYSERR).

Defining Default Specifications

Natural Construct V3.3.3 provides a facility to define default specifications and use these to further automate the generation process.

Note: This facility does not apply to the statement models or the Object-PDA models.

- To define default specifications for individual models on an application (library) level:
 - 1 Invoke the Modify function on the Generation main menu.
 - 2 Type “DEFAULT” in the Specification/Program field.
 - 3 Type a valid model name in the Model field.
 - 4 Define your defaults on the model specification panels as desired.
 - 5 Invoke the Save function to save your defaults.
- To modify previously-defined defaults:
 - 1 Invoke the Read function on the Generation main menu.
 - 2 Type “DEFAULT” in the Specification/Program field.
 - 3 Type the model name for which the defaults apply in the Model field.
 - 4 Invoke the Modify function.

- 5 Modify the defaults on the model specification panels as desired.
- 6 Invoke the Save function to save your modified defaults.

When you are defining default parameters, specification edits are not invoked. Defaults are read in for a model whenever the model's clear subprogram is invoked. This occurs automatically whenever the Clear function is invoked while a model name is specified, or the Modify function is invoked for a new model name.

The names of the modules that store the default model specifications in the application library are derived from the names of the model clear subprograms. For example, if the Maint model's clear subprogram is CUFMC, the default specifications module is called C@FMC.

To maintain unique default specification modules per model, the supplied models (which shared common clear subprograms) now have their own clear subprograms. If you created models at your site that share common clear subprograms, you should make copies of these and ensure that each model has a clear subprogram with a unique name.

One advantage of using the DEFAULT keyword when maintaining default model specifications is that it allows Natural Construct to shield developers from having to know or care about the naming convention used for storing the defaults.

Model specification defaulting at the application level overrides the PROVIDE-DEFAULT-VALUES user exit for a model's clear subprogram (i.e., if a default module for a model exists in the current library, these defaults are used rather than the defaults in the user exit). If no default module exists in the current library for a model, Natural Construct uses the defaults in the user exit. Model specification defaulting supports the Read, Modify, Save, List, and Clear functions. The User Exit, Generate, Test, Edit, and Stow functions are not supported.

If the DEFAULT keyword is inconvenient for your site, you can modify it by altering the value of the #DEFAULTING-KEYWORD variable in the CU--MDLL data area. After modifying the value of this variable, you must recatalog the modify subprograms for your models, as well as the CU--MDL subprogram, and use the SYSMAIN utility to move the object code into SYSTEM in your FNAT.

Note: Additional logic was added to the modify subprograms of the supplied models to allow them to recognize when default specifications are being defined and to bypass the specification edits. Modify subprograms you created or customized will support the new default functionality without modification, except that they enforce the specification edits when defaults are defined until such time as you modify them as per the supplied modify subprograms.

Modifying or Creating Model PDAs

During generation, Natural Construct reads information from the parameter data area (PDA) for each model. Previous versions of Natural Construct read the model PDA from the SYSLIB library. This version reads the model PDA from the SYSCST library. If you create new models, ensure that a copy of their model PDA is available in the SYSCST library.

Note: If you extend the list of condition codes for a model, add the new #PDAC- variables to the end of the list of current #PDAC- variables.

Support for Natural Construct Client

You can use Natural Construct V3.3.3 for OpenVMS as a host for Natural Construct Client. The Model Definition Maintenance function in the Administration subsystem was enhanced to support the definition of specification panels for use with Natural Construct Client. The model load, unload, and compare utilities were also enhanced to support Natural Construct Client.

Support for the Map Model

Natural Construct V3.3.3 supports the Map model, which helps generate maps quickly to use with other supplied models (such as the Maint and Object-Maint-Dialog* models). For information about the Map model, see *Natural Construct Generation User's Manual*.

Note: Currently, you cannot specify a layout map. Consequently, the Layout Map field on the first specification panel for the Map model is protected.

Remote-Procedure-Call-Stub Model

This model generates the required Entire Net-Work RPC CALLNAT stubs on the client that define parameters passed to a remote subprogram. It accepts the name of a remote subprogram as input, builds the list of parameters to be passed to the subprogram, and then calls NATCLT.

The format of the data defined in the subprogram must conform to the following standards:

- Reference each external parameter data area (PDA) on a separate line. For example:

```
PARAMETER USING PDA1  
PARAMETER USING PDA2
```

- Precede inline parameter data with the PARAMETER keyword on a separate line (see following example).
- Reference each inline variable declaration on a separate line. For example:

```
PARAMETER  
01 #FIELD1 (A2)  
01 #FIELD2 (N3)
```

The Remote-Procedure-Call-Stub model does not support user exit processing.

Enhancements to Existing Features

CDKEYLDA Adjustment

The format and length of the #KEY-NAME array defined in the CDKEYLDA local data area was increased from A5 to A10 so that, by default, applications generated with Natural Construct support PF-key names up to 10 characters in length. This change provides greater flexibility when translating PF-key names into other languages, especially those requiring double-byte character support where Shift-out/Shift-in characters have to be accommodated. While the ISA PF=key format displays a maximum of five characters, the SAA and PC formats and the “Actions as commands” format (all accessible through CDFLIP) display up to 10 characters.

This change will not affect existing applications compiled against the previous version of CDKEYLDA in any way. As modules from these applications are recompiled against the new CDKEYLDA, these modules inherit the 10 character support seamlessly.

CDFLIP Map Adjustments

The map names referenced within CDFLIP were adjusted to make use of the “&” notation. This change allows CDFLIP to automatically support maps in multiple languages without customization. The CDFLIP11 and CDFLIP21 English layout maps are the only ones supplied. To have CDFLIP support your language(s) of choice, copy these maps, translate them, and replace the last character of the map names with the appropriate language code.

Model Enhancements to Support SQL-Based DDMs

All models that generate file accesses can now specify any of the new file types supported by Predict V3.3 (Adabas D, Oracle, Sybase, Informix, and Ingres).

Note: This support depends on the feature being supported by Predict and Natural.

Regeneration of Object PDAs

The Object-PDA and Object-PDA-R models were enhanced to allow the object parameter data areas (PDAs) to be regeneratable. Model specifications are now stored at the top of these PDAs at generation time to support the regeneration feature.

The NCSTBGEN and CSTBGEN mass generation utilities were adapted to ensure that data areas are regenerated and cataloged before other objects.

Note: Existing object PDAs will not be recognized as generated by Natural Construct.

Various Minor Corrections

Batch Model

- A statement in the CUBAGPS module for the Batch model that compares the value of *LANGUAGE to the elements of an array was adjusted because comparison of a system variable to an array is no longer supported by Natural.
- The CUBAGPS module for the Batch model was adjusted to prevent NULL lines from persisting in the edit buffer (under certain conditions) after generation of TOP-OF-PAGE user exits.
- The correct CUBAHELP help routine is now attached to the Additional Input Parameters panel for the Batch model.

Browse* Models

- Field headings generated into the WRITE-FIELDS user exit for Browse* model programs now correspond to field name synonyms rather than the original field names (whenever synonyms are defined and when there is no field heading information defined in Predict).
- When an alternate index that is redefined in a VSAM file is used as the primary key in a Browse* model program, generation no longer results in an abnormal termination.
- The BEFORE-STANDARD-KEY-CHECK user exit, which was missing from the Browse* models (i.e., the non-Select ones), is now included in these models.

- When generating code for the WRITE-FIELDS user exit, the Browse* models now check whether the DDM for the file involved was generated with a prefix and, if so, generate field names that include the DDM prefix.
- The Prototype function (PF12) on the Build Report Layout panels for Browse* models now supplies default position values for fields with an “*” (asterisk) in the Panel or * field.
- The Browse* and Browse-Select* models no longer generate duplicate “Additional (non key) input field” clauses when additional user fields are specified in the generation parameters and the primary key is a superdescriptor.
- The CUSLC3 code frame for the Browse-Select* models was adjusted to suppress the generation of an IF #ACTION = 'A' clause when the format of the action column is Logical. Furthermore, when this clause is generated into programs, the reference to “A” is replaced with a reference to “#ADD(*)”.
- The End of Data message is now displayed intensified in the Browse-Select* models, just as in the Browse* models.
- Pressing PF5 on the Action Column Definition panel for the Browse-Select model when actions with Logical format are used on an external map no longer creates an indexing error.

Driver Model

- The Driver model was adjusted to properly initialize the components of compound keys to their respective null values.

Maint Model

- The Maint model now generates correctly against IMS files containing multiple re-defined keys extracted from the I-segment.
- When generating inline maps, the Maint model bypassed fields whose names were embedded within the name of the key field. This was corrected.
- In the Maint model, compound keys with packed elements are now reset correctly when the key is null.
- The WHERE clauses in Select statements for the Next action are now generated correctly by the Maint model for DB2 tables whose keys are redefined in Predict.
- A new edit was added to the Maint model parameter specification panels to ensure that the secondary file, if specified, is not defined in a referential integrity relationship with the primary file within Predict.

- Field prompts generated into inline INPUT statements in Maint model programs now correspond to field name synonyms rather than the original field names (whenever synonyms are defined and when there is no field heading information defined in Predict).
- The LINE-CHECK subroutine for Maint model programs now compares binary format fields against a value of H'00' instead of "0" when checking for null values.
- The CUFMGREF module for the Maint model was adjusted so that Maint programs generated without a Purge action now STOW properly.
- The CUFMC3 code frame for the Maint model was adjusted to avoid situations where secondary file records were stored during Add actions even though occurrences of the secondary file array were blank.

Menu Model

- The Menu model now generates DY= attributes that do not use hex values (if the special characters are supported on all platforms).
- The CUMN code frame for the Menu model was adjusted so that when menus issue an "Invalid code selected" message, they also reset DIALOG-INFO.##COMMAND. This removes any trailing direct command data from the system that may exist if the menu code was originally issued via a command in the Direct Command box.
- When generating inline maps that use Required/Optional parameters in Menu model programs, long lines are no longer truncated.

Object* Models

- The Object-PDA model was enhanced to handle the case where the key field used to link entities in an intra-object relationship is a redefined portion of a larger field.
- Under certain conditions of referential integrity checking, an invalid assignment of *8071 was made to ##MSG-NR in Object-Subp model subprograms. The asterisk was removed from the message number.
- The Object-Subp model no longer generates invalid field names into the referential integrity edits in the edit-object subroutine of object subprograms.
- “View-name.field-name” combinations greater than 50 characters in length are no longer truncated during the generation of object subprogram referential integrity edits.
- The CUOBGUPE module for the Object-Subp model now generates additional ELSE/END-IF statements that are required under certain conditions involving referential integrity relationships defined against sub-entities of the object.
- You can now successfully STOW Object-Maint-Dialog model programs — even if the maps they use do not reference any of the object PDA fields.

View Model

- The View model supports field redefinitions as deep as level 8 and 9.

Multiple Models

- The Next actions for Maint model programs generated against DB2 tables keyed by date or time fields no longer require a specified date/time value. Similarly, Browse model programs generated against DB2 tables keyed by date or time fields now allow scrolling to begin without first specifying a date/time key value.
- A condition that sometimes led to the selection of duplicate fields on the Build Report Layout panels for the Map, Browse*, and Batch models no longer exists.

Predict

- The Predict interface now allows models that generate file accesses to access the new file types supported in Predict V3.3 (Adabas D, Oracle, Sybase, Informix, and Ingres).

Modules

- A problem in the CSUBMIT module that occurred when submitting JCL to the internal reader was corrected.
- The CPUSUPER module was adjusted to properly handle the time components of superdescriptors defined on DB2 tables.
- The CPU-VIEW module was adjusted to allow the Maint model to properly generate against DB2 tables using AV fields.
- The CDDDB2MG2 module called the DB2SERV module using the Edit function. Under current releases of Natural DB2, this is no longer a valid call. CDDDB2MG2 now CALLNATs the NDBERR module instead.
- The PF-key assignments in the CD-HPRED module are now based on CDKEYLDA.
- The CD-HELPR module was adjusted to support values of *Language greater than nine.
- The CDENVIR module, which saves and restores screen environment parameters, was adjusted to avoid situations leading to the ESIZE of a Natural session being exceeded.
- The CDUMSG module was adjusted to properly substitute error message parameters into the :1::2::3: values of messages retrieved from SYSERR.

Utilities

- The trailing blank lines of help text pages loaded by the CSHLOAD utility are properly retained.

Editor/Editing Problems

- The program mode (Structured or Reporting) assigned to the edit buffer is now set correctly, as per the model definition, under all circumstances.
- When the Edit function for NCSTH is invoked, Natural Construct now re-reads the help text member from the database file regardless of whether the key of the member is the same as that of the help text member that was last edited.
- The index used by a referential integrity edit is no longer mistakenly carried forward into subsequent integrity edits that do not require indexes.

Generation Problems

- Longer INIT<'... '> statements are support during generation than previously.
- Group fields defined within field redefinitions no longer cause problems at generation time.
- Under certain conditions, after STOWing a generated object PDA, Natural Construct returned the user to the NEXT prompt rather than to the Generation main menu. This problem no longer occurs.

Miscellaneous

- Under certain conditions, an invalid error handling program was assigned to *ERROR-TA in the Generation subsystem. This led to a NAT0082 error (program not found) if an error condition was subsequently triggered. This was corrected.
- The field names assigned to ##ERROR-FIELD in the edit-object subroutine for object subprograms are no longer prefixed by the file name.
- Blank lines in JCL streams are now preserved during the online submission of JCL.
- User-defined arrays on dialog maps can now be named #ARRAY without confusing the “defaulting” function on Dialog model panels that process such maps to generate the map scroll region values.
- You can now issue a CLEAR command in the Direct Command box of the Generation main menu without specifying a valid object name. Additionally, executing this command no longer resets the program name assigned to the edit buffer.
- Object PDAs for objects consisting of multiple DB2 tables that contain null fields can now be generated and STOWed successfully. Null suppressed (FILLER) fields are now suffixed with a 6-digit number to ensure uniqueness within the PDA.

