Using SYSERR for Multilingual Support

This section describes how Natural Construct uses the Natural SYSERR utility to dynamically translate text and messages. SYSERR contains reference numbers that reference text strings in one or more languages. This section covers the following topics:

Introduction

Natural Construct supports the dynamic translation of text and messages on many specification panels. Instead of typing text for panel headings, field prompts, error messages, etc., you can use a SYSERR reference number. At runtime, the reference number is replaced with its corresponding SYSERR text.

Maintenance

Using SYSERR references reduces your maintenance efforts. To modify a field prompt used on many panels, for example, you can change the text in SYSERR and all fields that use that reference number display the new name at runtime. It also helps maintain consistency throughout your generated applications, by ensuring that the same text is displayed in multiple locations.

Translation

For each SYSERR reference number, you can define message text in other languages. At runtime, text for the currently-selected language (the current value of the *Language system variable) is retrieved.

The text on all Natural Construct panels can be dynamically translated into any Natural-supported language.

Note:
If you only require one language, this feature can be disabled during installation. For more information, see Static (One-Language) Mode.

The default language for Natural Construct is English (*Language 1), which is always supported. Check with your local Software AG office to ensure that your language is supported.

Define SYSERR References

Each SYSERR reference number can have up to 15 distinct text entries — each one separated by a (/) slash delimiter. For information about setting up reference numbers, refer to the SYSERR utility in the Natural Utilities documentation.

To use SYSERR reference numbers in Natural Construct, the reference must follow a pattern where the first character is an * (asterisk) and the next four digits represent a valid SYSERR reference number. For example:

`*nnnn`

where * indicates the currently specified SYSERR message library and *nnnn* represents a valid reference number. To identify one of the 15 possible positions within a SYSERR reference number, use the following notation:

`*nnnn.A`

where *A* is a number from 1–9 or a letter from A–F. The numbers 1–9 represent the first nine positions and the letters A–F represent the 10th to 15th positions. For example, to reference the fifteenth position within a reference number, specify:

`*nnnn.F`

Note:
We recommend that you always specify a position value, even if there is only one occupied position in the reference number. This eliminates the need to modify SYSERR references if additional positions are occupied in the future.

Use SYSERR References

You can use SYSERR reference numbers in several ways, such as:

- On Maps (Screen Prompts)
- For Panel Headings and PF-Key Names
- In Messages
- For Text Translation
- With Substitution Values

All text members, excluding the help text members, reside within the SYSERR utility. Each text member is identified by a two-part key — a SYSERR library name and a four-digit number.

- For more information about SYSERR, refer to the Natural Utilities documentation.
- For information about using SYSERR references in help text, see *Message Numbers*, *Natural Construct Help Text*.

On Maps (Screen Prompts)

To display panels in many languages, Natural Construct uses a single map approach. Variables for all screen prompts are defined and initialized in a translation local data area (LDA) associated with each map.

Translation LDAs initialize the screen prompts with SYSERR references for the dynamic translation version or constants for the static version. All supplied LDAs use SYSERR references by default, but you can change this if desired. For more information about dynamic and static installations, refer to the installation documentation.

The one-to-one association between a map and its translation LDA is an effective method for naming and tracking panels and their prompts. Each supplied map and its translation LDA have identical names — except for the last character. The last character in a map name is “0” (zero) and the last character in a translation LDA name is “L”. For example, the second specification panel for the Menu model is CUMNMB0 and the translation LDA is CUMNMBl.

Screen prompts are typically translated prior to displaying a panel, and panels usually have more than one prompt. For this reason, Natural Construct uses the CSUTRANS utility to receive a block of text and translate all references numbers. The CSTLDA library in SYSERR is Natural Construct’s dedicated library and contains all language-independent prompt text.

For more information, see CSUTRANS Utility.

For Panel Headings and PF-Key Names

You define and maintain panel headings and PF-key names in the Administration subsystem: the first heading for a model specification panel on the Maintain Models panel and the PF-key settings on the Natural Construct control record.

Note:
When we refer to panel headings and PF-key names, we are referring to the Natural Construct panels and PF-keys and not those used by the generated applications.

You define and maintain panel headings and PF-key names in the Administration subsystem: the first heading for a model specification panel on the Maintain Models panel and the PF-key settings on the Natural Construct control record.

If desired, you can use the CST-Modify model to generate a maintenance subprogram for the model that can override these defaults. Maintenance subprograms reference the #HEADER1 and #HEADER2 internal variables to display panel headings. If these headings are not overridden by the maintenance subprograms, Natural Construct automatically uses the defaults supplied by the nucleus (in the CU—PDA.#HEADER1 and CU—PDA.#HEADER2 variables). For more information about overriding panel headings, see Standard Parameters Panel.

All Natural Construct panel headings and PF-key names support text or SYSERR references (the **nnnn.A* notations). For example, to name a PF-key “main” on the control record, enter one of the following:

- "*0033.5" (which corresponds to “main” in SYSERR)
- "main" (which disables the dynamic translation feature)

All heading and PF-key text is saved in the same SYSERR library as the prompt text (CSTLDA).

In Messages

All Natural Construct messages also support dynamic translation. Messages have action properties (verbs), whereas screen prompts have descriptive properties (adjectives). For this reason, the message and prompt text is stored in separate SYSERR libraries and use separate translation utilities:

- Messages are stored and maintained in the CSTMSG library and are accessed via the CNUMSG single message utility.
- Screen prompts are stored and maintained in the CSTLDA library and are accessed via the CSUTRANS utility.

If you change the supplied screen prompt text, ensure that the screen prompt and message text are consistent. If the message text references a different SYSERR number than the screen prompt, the message may be confusing.

With modules for which source is not supplied, Natural Construct uses the text substitution feature supported by the CNUMSG utility (where :1::2::3: are place holders for potential substitution values). For example, if the screen prompt is "Module name" and the message is ":1::2::3:is required", the message is displayed as: Module name is required.

This message substitution feature provides many benefits, including:

- Consistent use of panel and message text
- Reuse of common messages, such as “is required”
- Reduced volume of message translation
- Consistent wording between modules
- Support for a cleaner and crisper look

The following example shows a typical message and how it is coded:

```
ASSIGN CNAMSG.MSG-DATA(1) = CU—MAL.#GEN-PROGRAM
INCLUDE CU—RMSG ’2001’
        ’’’:1::2::3:is required’’’
        ’#PDA-PROGRAM-NAME’
```

This assignment transfers the contents of the corresponding prompt variable into the first (of a possible three) substitution data member: `CNAMSG.MSG-DATA(1)`. The members are then transferred into an INCLUDE member that calls the CNUMSG utility.

In the preceding code example, CU—MAL is the translation LDA for the CU—MA0 map and CU—MAL. #GEN-PROGRAM is the prompt variable containing the initialized text (either “Module” or the SYSERR number that references “Module”). The 2001 on the INCLUDE line represents the SYSERR reference number that points to the message: “:1::2::3:is required”. The “:1::2::3:is required” text below the INCLUDE code is used as an internal default should the text not be found.

You can use the Natural Construct messaging infrastructure to override the message lookup and force the CNUMSG utility to disregard the SYSERR reference number and use the text (:1::2::3:is required) instead. This feature is useful during model development because you can enter message text in the source code or test the code without calling the SYSERR utility. To do this for a single module, add a single line before the previous code example as follows:

```
ASSIGN CNAMSG.INSTALL-LANGUAGE = *LANGUAGE
```

To do this for an application, change the initial value for the CNAMSG.INSTALL-LANGUAGE variable and recompile all the Natural Construct model subprograms.

The following INCLUDE code members all retrieve message text, but process the text in different ways:

INCLUDE Code Member	Description
INCLUDE CU—RMSG	Retrieves and displays messages on current panel.
INCLUDE CU—SERR	Retrieves and sets error code messages and then exits current module.
INCLUDE CU—GMSG	Retrieves messages and continues processing (typically used for warning messages).
INCLUDE CU—GTXT	Retrieves messages and continues processing, but does not transfer the text to the CSASTD structure (typically used to perform initializations without corrupting the messaging data in CSASTD).

For Text Translation

You can translate text in one of two ways: mass translation from within the SYSERR utility or context translation from within Natural Construct, which uses the SYSERR utility to store text for all supported *Language values. English is the default language; it is always supplied and supported.

Since translation is typically performed once shortly after installation (or not at all if the product is delivered with the text translated), Natural Construct provides a special translation mode that is invoked via a command you can secure. This command, menut, accesses the Administration subsystem in translation mode with all translatable prompts and headings highlighted for easy identification.

Mass Translation

All Natural Construct text is available in SYSERR. The combination of the SYSERR library name and a four-digit number is the unique key or pointer to a particular text member. For example, the “:1::2::3:is required” message is stored in the CSTMSG library and its four-digit number is 2001; the “Module” screen prompt is stored in the CSTLDA library and its four-digit number is 1000.

In SYSERR, you can translate many messages one after the other (mass translation). This mechanism is fine for messages where the context is not critical. For example, the “:1::2::3:is required” message is universal and used frequently by all types of modules.

Screen prompts are more context sensitive; they may belong in a particular group or depend on a heading for meaning. To translate screen prompts, it is a good idea to perform a mass translation first and then check each panel individually for context. This is the most efficient way to translate a large number of text members, as this translation can be accomplished by less experienced Natural Construct users or a translation service.

Context Translation

Natural Construct's context (cursor-sensitive) translation provides a simple but effective method to check or change the results of a mass translation. It allows you to display a panel, place your cursor on highlighted text, press Enter, and be presented with a window in which you can change or translate the text. For example:

```
CSUTLATE                    Natural Construct
Jul 04                  Translate Short Message                    1 of 1

Language Short Message ( CSTLDA2101 )
——— ....+....1....+....2....+....3....+....4....+....5....+....6....+

English  Module/Model/Maps                                          /+20
```

This feature is even more convenient on a PC using Entire Connection, in which case you can double-click any prompt to perform the translation.

Notes:

1. You can also use the context translation mechanism to perform the original translation (instead of mass translation).
2. Because messages are displayed one at a time, they do not require context translation.

Since translation is typically performed once shortly after installation (or not at all if the product is delivered with the text translated), Natural Construct provides a translation mode command that you can secure. This command, menut, accesses the Administration subsystem in translation mode with all translatable prompts and headings highlighted for easy identification.

Unlike messages, which all use the same byte length, screen prompts vary in length depending on panel design and available space. For performance and space considerations, multiple screen prompts may share the same SYSERR location. For example, SYSERR number 2000 corresponds to the following text:

```
CSTLDA2000 Module/System/Global data area                    /+20
```

where `CSTLDA2000` indicates the SYSERR library and the four-digit number that identifies the values: Module, System, and Global data area (delimited by a "/"). Decimal numbers indicate which text is retrieved (for example, 2000.1 for Module, 2000.2 for System, and 2000.3 for Global data area). Since prompts can be different lengths, the /+20 notation indicates that each of these prompts can occupy up to 20 bytes on any panel they are used.

With Substitution Values

Substitution values are additional data that can be displayed with message text at runtime. For example, you can specify that Menu (the substitution value) be displayed with Main (the message text). The actual substitution value can be either text or another reference number. Most areas in Natural Construct that support reference numbers also support data substitution. For information about supported areas, see Supported Areas in Natural Construct.

To use substitution values with a reference number, the reference number must be defined in the SYSERR utility with the :1::2::3: place holders. For more information, refer to *REINPUT Statement*, *Natural Statements* documentation.

To specify substitution values for a reference number that contains place holders, type the reference number (**nnnn.A* format), followed by a comma (,) delimiter, and up to three substitution values. For example, if you enter:

```
0200.1,Menu,Model
```

where `0200.1` corresponds to the message text `:1::2::3:Program`, and Menu and Model are the substitution values. At runtime, the following text is displayed:

```
Menu Model Program
```

In this example, Menu replaced the first place holder and Model replaced the second.

Note:
If no substitution values are defined, the place holders are ignored.

You can enter text, or reference numbers, or both as substitution values. For example, if you enter:

```
0200.1,Menu,0502.4
```

where `Menu` is the first substitution value and `0502.4` is the second substitution value (which corresponds to the message text “Model”). At runtime, the following message is displayed:

```
Menu Model Program
```

Format SYSERR Message Text

In some areas where SYSERR references are used, you can specify how the retrieved message text is formatted at runtime. The following table describes the formatting characters:

Character	Description
,	Separates the **nnnn.A* notation from the format characters.
.	Fills the remaining blanks.
+	Centers the retrieved text.
<	Left-justifies the retrieved text. Typically, you will not use this character because retrieved text is left-justified by default.
>	Right-justifies the retrieved text.
/	Indicates the end of format characters and the beginning of the field length override. For example, “+/30” indicates that the first 30 characters of returned text are centered. Any additional characters are truncated. This character is used with alignment characters (such as +, <, or >).
NN	Indicates the field length override value. Using the example above (+/30), the field length override is 30 characters.

The following examples show different methods of formatting the text for SYSERR reference number 0210.1 (which references the text, “Field Help”):

Format Specified	Result
*0210.1,+/24	Centers text in 24 bytes. At runtime, text is displayed as: `|  Field Help  |`
*0210.1,>/24	Right-justifies text. At runtime, text is displayed as: `|  Field Help|`
*0210.1,/24 or *0210.1,</24	Left-justifies text (the default). At runtime, text is displayed as: `|Field Help  |`
*0210.1,./24	Left-justifies text and fills the remaining blank spaces with periods. At runtime, text is displayed as: `|Field Help.....|`

Supported Areas in Natural Construct

The following table lists the areas where you can use SYSERR references. The Substitutions column indicates whether substitution values are supported for the corresponding panel; the Formatting column indicates whether formatting is supported.

Location	Panel Element	Substitutions	Formatting
Maintain Control Record panel	PF-key names	No	No
	Panel indicators	No	No
Maintain Models panel	Description	Yes	No
Maintain Subprogram panel	Description	Yes	No
	PF-key names	No	No
Standard Parameters panel (CST-Modify model)	Header 1	Yes	Text centering only
	Header 2	Yes	Text centering only
	PF-key names	No	No
Translation local data areas (LDAs)	CNUMSG utility	Yes	Partial support
	CSUTRANS utility	Yes	Yes
Help Text editor	Header 1	Yes	No
	Header 2	Yes	No
	Hotlinks	Yes	No
	Body of help text	Yes	Yes

- For information on substitution values, see With Substitution Values.

- For information on formatting, see Format SYSERR Message Text.

The following table lists sections where you can find more information about each of the Natural Construct functions and utilities in which SYSERR reference numbers are supported:

To Learn More About	Refer To
Maintain Control Record panel	Maintain Control Record Function
Maintain Models panel	Maintain Models Function
Maintain Subprogram panel	Maintain Subprograms Function
CST-Modify model Standard Parameters panel	Parameters for the CST-Modify Model
Translation LDA utilities (CNUMSG and CSUTRANS)	• CNUMSG Subprogram • CNUMSG Utility • CSUTRANS Subprogram • CSUTRANS Utility
Help Text editor	*Editing Help Text*, *Natural Construct Help Text*

CSUTRANS Utility

Natural Construct translates screen prompts before they are displayed. As most panels have multiple prompts, Natural Construct incorporates the CSUTRANS utility to receive a block of text and translate all references to SYSERR numbers into the appropriate *Language text.

CSUTRANS translates 1:V data structures and is used extensively for dynamic translation. The utility reads through a supplied local data area, looking for one of two patterns: **nnnn* or **nnnn.A*.

The **nnnn* pattern returns all text for that SYSERR number, whereas the **nnnn.A* pattern returns only the text in the specified position (delimited by a /, such as **nnnn*.1 for the first position, **nnnn*.2 for the second, **nnnn*.A for the 10th, etc.). The extension in the **nnnn.A* pattern is alphanumeric; valid values range from 1–9 and A–F, for a total of 15 possible positions.

To retrieve a valid message, you must also specify the SYSERR library name (CSTLDA, by default).

Note:
To change the library name, use the #MESSAGE-LIBRARY variable.

You can also use SYSERR numbers to assign the INIT values for fields in the translation LDAs. These LDAs are passed through the CSUTRANS utility, which expects a certain data structure. The following example illustrates this structure for the Standard Parameters panel for the Batch model:

```
***SAG TRANSLATION LDA
***used by map CUBAMA0.
   1 CUBAMAL
   2 TEXT                                /* Corresponds to SYSERR message
   3 #GEN-PROGRAM               A   20 INIT<'*2000.1,.'>
   3 #SYSTEM                    A   20 INIT<'*2000.3,.'>
   3 #GDA                       A   20 INIT<'*2000.2,.'>
   3 #TITLE                     A   20 INIT<'*2001.3,.'>
   3 #DESCRIPTION               A   20 INIT<'*2001.2,.'>
   3 #GDA-BLOCK                 A   20 INIT<'*2001.1,.'>
 R 2 TEXT
   3 TRANSLATION-TEXT
   4 TEXT-ARRAY                 A    1 (1:120)
   2 ADDITIONAL-PARMS
   3 #MESSAGE-LIBRARY           A    8 INIT<'CSTLDA'>
   3 #LDA-NAME                  A    8 INIT<'CUBAMAL'>
   3 #TEXT-REQUIRED             L      INIT<TRUE>
   3 #LENGTH-OVERRIDE           I    4 /* Explicit length to translate
```

Some of the important structural elements in this LDA are:

- The first comment line (`**SAG TRANSLATION LDA`) indicates that this is a translation LDA. During a Static install, Natural Construct scans for this comment line and replaces the SYSERR numbers with the appropriate text.
- The `CUBAMAL` level 1 structure name is typically the LDA name. You should use this qualifier to reference the variables.
- The level 3 variables (`#GEN-PROGRAM`, `#SYSTEM`, `#GDA-BLOCK`, etc.) are the screen prompts, which are initialized with a SYSERR number. All SYSERR numbers use the **nnnn.A* notation and are listed in sequential order (so that CSUTRANS does not retrieve SYSERR *2000, then *2001, and then *2000 again).

  Note:
  The sequence order does not apply to the **nnnn.A* notation extensions (.A). For example, you can list *2000.2 before *2001.1.
- The `TEXT-ARRAY` value must match the total number of bytes in all screen prompt variables to be translated.
- The `#MESSAGE-LIBRARY` value indicates the SYSERR library name used to retrieve text.
- The `#TEXT-REQUIRED` logical variable indicates whether translation is required for Natural Construct modules. If translation is required, `#TEXT-REQUIRED` ensures that translation is only performed once.

The SYSERR INIT values have the following format:

Position	Format
Byte 1	Must be an asterisk (*).

Position	Format
Bytes 2–5	Must be numeric and represent a valid SYSERR number. The first five bytes are mandatory. These values are used to retrieve the text associated with the corresponding SYSERR number and the current value of *Language. If the text for the current language is not available, CSUTRANS follows a modifiable hierarchy of *Language values until text is retrieved (you define this hierarchy in the DEFAULT-LANGUAGE field within the CNAMSG local data area). As the original development language, English (*Language 1) should always be available. **Note:** CSUTRANS does not perform substitutions (using :1::2::3:). To perform substitutions, call the CNUMSG subprogram. For information, see CNUMSG Subprogram.
Byte 6	Can be a period (.), which indicates that the next byte is a position value.
Byte 7	Can be a position value. Valid values are 1–9, A (byte 10), B (byte 11), C (byte 12), D (byte 13), E (byte 14), F (byte 15), and G (byte 16). For example, *2000.2 identifies the text for SYSERR number 2000, position 2 (as delimited by a / in SYSERR). If the message for SYSERR number 2000 is Module/System/Global data area, only System is retrieved. If you reference the same SYSERR number more than once in a translation LDA, define the INIT values on consecutive lines to reduce the number of calls to SYSERR. (The position values for a SYSERR number can be referenced in any order.) **Tip:** To minimize confusion, we recommend that you use the *.A* extension even when there is only one position defined for the SYSERR number.
Byte 8	Can be a comma (,), which indicates that the next byte or bytes contain special format characters. Values specified before the comma (,) indicate what text to retrieve; values specified after the comma indicate how the text is displayed. **Note:** Although you can use a comma in byte 6 (instead of a period), use the *.A* extension in bytes 6 and 7.

Position	Format
Byte 9	After the comma, can be one of the following: ● . (period) Indicates that the first position after the field name is blank and the remainder of the field prompt is filled with periods (`Module .........:`, for example). ● + Indicates that the text is centered using the specified field length override (see description of Byte 10). If you do not specify the override length, Natural Construct uses the actual field length. ● < Indicates that the text is left-justified (this is the default). ● > Indicates that the text is right-justified. ● / Indicates that a length override value follows. This character is placed after the alignment character (+,< or >). For example, /+20 indicates that the text is centered within 20 bytes.
Bytes 10–16	After the / (override length indicator), indicates the override length in bytes.

If you want to use the override length notation (*0200.4,+/6, for example) and the LDA field is too small (A6, for example), define a larger field, redefine it using a shorter display value, and then use the override length notation. For example:

```
01  #FIELD-NAME                   A 12 INIT<'*0200.4+/6'>
01  Redefine #FIELD-NAME
  02  #SHORT-FIELD-NAME           A  6
```

CNUMSG Utility

Unlike CSUTRANS, the CNUMSG utility only retrieves text for one message at a time. It is typically used to retrieve warning or error messages, and sometimes to retrieve text for initialization.

The CNUMSG utility retrieves message text in one of two ways. If a reference number is specified (CNAMSG.MSG-NR), CNUMSG uses that number to retrieve the SYSERR message text. If a reference number is not specified, CNUMSG checks the message text (CNAMSG.MSG) for the **nnnn* or **nnnn.A* notation and uses the specified notation to retrieve the SYSERR message text.

CNUMSG can also substitute values in the text it retrieves (up to a maximum of three substitution values). CNUMSG retrieves the message from SYSERR and checks to see whether the message has any substitution place holders. If it does, then the substitution text data members (CNAMSG.MSG-DATA(*)) are substituted into the appropriate place holder. If the data member is another SYSERR reference, it is

retrieved and substituted. All unused substitution place holders are removed. By default, CNUMSG uses the CSTMSG SYSERR library for messages and the CSTLDA SYSERR library for substitution data fields.

Examples of Using the CNUMSG Utility

For the following examples, assume you want to create the message: `ADD Action Description is required` and the available SYSERR numbers and text are:

SYSERR Reference Number	SYSERR Library	SYSERR Text
*2001	CSTMSG	:1::2::3:is required
*1116.1	CSTLDA	Action/Subprogram
*1117.1	CSTLDA	Description

Example 1: Typical Text Retrieval

```
ASSIGN #DESCRIPTION ="*1117.1"...         /* Variable with a SYSERR reference
ASSIGN CNAMSG.MSG-DATA(1) = "ADD"  ..   ../* Hardcoded text
ASSIGN CNAMSG.MSG-DATA(2) = "*1116.1"     /* SYSERR Reference
ASSIGN CNAMSG.MSG-DATA(3) = #DESCRIPTION  /* Variable reference
INCLUDE CU—GMSG "2001"
       """:1::2::3:is required"
       """ """
```

Example 2: Text Retrieval Using a Comma as the Delimiter

```
ASSIGN CNAMSG.MSG = "*2001,ADD,*1116.1,*1117.1"
INCLUDE CU—GMSG " "
       """:1::2::3:is required"
       """ """
```

Both of these examples build the same message. Example 1 is the preferred method because it is much more explicit. The method in Example 2 is useful when only the message text is available and the input must be entered in one field, such as the Description, Header, or Title fields.

Note:
Example 2 also supports centering. If you specify `+/NN` in your message text, CNUMSG uses the *NN* value as the centering length and removes the remainder of the text (the `,+/NN` pattern).

To perform a desired function, CNUMSG can also be called with a method. Natural Construct supports the following methods:

Method	Description
R	Retrieves the SYSERR message “as is” without any text substitution. This method works well for cases where substitutions are not desirable and the :1::2::3: place holders should be left intact (for example, when generating a call to CNUMSG itself).
S	Substitutes the data into the :1::2::3: place holders without retrieving the main message text. For example, you can use this method to apply substitutions to a text string that is created programmatically. This method only substitutes the available (passed) data into the place holders. Unused place holders are removed.
B	Retrieves the message text and performs the substitutions. This is the most commonly used method and is the default setting when the method is blank.
blank	Defaults to method B.

All other method settings will return a fatal error without performing any actions.

Static (One-Language) Mode

By default, Natural Construct is installed in dynamic (multilingual) mode, which allows users to display Natural Construct in any available language. If you intend to operate Natural Construct in one language only and do not require dynamic translation, you can replace all SYSERR references with text when Natural Construct is installed. During installation, Natural Construct provides a Static option that retrieves and replaces the **nnnn* references with the appropriate *Language text.

Notes:

1. Before using the Static option, check with your local Software AG office to ensure that your language is supported. If you are installing a static version in any language except English, which is always supported, review all messages in the CSTLDA library in SYSERR to ensure they are translated into the desired language.
2. Installing in static mode does not limit your ability to generate multilingual applications; static mode applies to the interface only.

The Static option does not replace every SYSERR reference with text; it only replaces SYSERR references in the most frequently used modules. The following table describes the areas affected and the replacements made:

Area	Replacements
Screen prompts	In all translation LDAs for which source is supplied (CU prefix), the Static option replaces references with text. To identify a translation LDA, Natural Construct checks the first comment line for `**SAG TRANSLATION LDA`.
Translation LDAs	For the most frequently used translation LDAs for which source is not supplied, you can generate static text LDAs and subprograms. For information, see Create Performance LDAs and Subprograms.
Headings and PF-key names	For all panel headings and PF-key names (which are installed with SYSERR references), you have the option of replacing the references with text.
Messages	Dynamically translated at runtime (since messages are only displayed during an error or warning condition).
Help text	Dynamically displayed at runtime (displayed on request).

Note:
Natural Construct can also use the English text supplied with each INCLUDE code member and bypass the SYSERR retrieval process (see In Messages).

There are two options for installing in static mode:

- Install Natural Construct in Static Mode
- Create Performance LDAs and Subprograms

You can specify either or both options.

Note:
If you are installing a static version in any language except English, review all messages in the CSTLDA library in SYSERR to ensure they are translated into the desired language.

Install Natural Construct in Static Mode

To install Natural Construct in static mode:

1. Log onto the SYSCST library.
2. Enter "NCSTI" (Natural Construct Install) at the Natural prompt.

   The Natural Construct Installation main menu is displayed. For example:

```
NCSTI           ***** N A T U R A L   C O N S T R U C T *****
Feb 27                    - Installation Main Menu -                  9:52 AM

               Code Function
               ---- ----------------------------------------
                S   Static Install (one language)
                L   Create Performance LDAs
                I   Create Performance Subps
                ?   Help
                .   Terminate
               ---- ----------------------------------------
          Code: _




Direct command...: _____________________________________________________________
Enter--PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12-
       help  retrn quit        flip                                    main
```

3. Enter "S" in Code.

   The Static Install (one language) window is displayed. For example:

```
INSTALL         ***** N A T U R A L   C O N S T R U C T *****
Feb 27                - Static Install (one language) -               9:57 AM

Enter the language in which you would like Natural Construct
installed (Any PF-key to quit): 1_
```

4. Enter the number for the language in which you want to install Natural Construct (for example, "2" for German, "3" for French).

   Natural Construct recreates all the LDAs for the model specification panels and replaces the SYSERR references to field prompts with the text for the language specified. The following window is displayed:

```
INSTALL         ***** N A T U R A L   C O N S T R U C T *****
Feb 27                - Static Install (one language) -              10:37 AM

All data areas have been populated with text appropriate to
language 1  .
In order to complete this process, please recompile all
Natural modules in the SYSCST library beginning with CU, CG
and copy the object code for these modules to SYSLIBS.
```

   Note:
   Set the Natural RUNTIME parameter to 40.

5. Perform a CATALL on modules beginning with “CU” or “CG” in the SYSCST library.

You need only select the subprogram and local data area (LDA) modules. In addition, mark the Catalog ALL Source-programs option to catalog all source modules. You may want to do this step in batch mode, because many modules are affected. You can use the following input:

```
LOGON SYSCST
CATALL CU*,,X,,,X,,,,X,,,,,
CATALL CG*,,X,,,X,,,,X,,,,,
FIN
```

6. Copy the object code from these modules into the SYSLIBS library.

   If you prefer to do this in batch mode, the SYSMAIN input commands are supplied below. Ensure that the IM=D parameter is set in your NATPARM. Use the following batch input:

```
LOGON SYSTEM
SYSMAIN
MENU C,C,CU*,TYPE,N,FM,SYSCST,DBID,xxx,FNR,yyy,TO,SYSLIBS,DBID,xxx,%
FNR,yyy,REP
SYSMAIN
MENU C,C,CG*,TYPE,N,FM,SYSCST,DBID,xxx,FNR,yyy,TO,SYSLIBS,DBID,xxx,%
FNR,yyy,REP
FIN
```

Create Performance LDAs and Subprograms

Regardless of whether you choose the Static Install function or not, this option will enhance performance by creating several subprograms that eliminate calls to SYSERR to build many of the frequently used screens (such as the Generation main menu). Because these programs are not supplied in source form, use the Create Performance LDAs function to create LDAs containing the text appropriate to the desired language and then use the Create Performance Subps function to create the performance subprograms. You can repeat these two steps as many times as desired, depending on how many languages you want to make available.

Note:
Natural Construct supplies the performance subprograms for English. If you are running Natural Construct in a language for which these subprograms have not been created, the English subprograms will be invoked.

To create performance LDAs and subprograms for the Natural Construct nucleus:

1. Copy the contents (source and object) of the SYSCST00 library into the SYSCST*nn* library (where *nn* is the language code for the language you want to support, such as 1 for English, 2 for German, 3 for French).

2. Log onto the SYSCST*nn* library.

3. Enter "NCSTI" (Natural Construct Install) at the Natural prompt.

   The Natural Construct Installation main menu is displayed.

   Note:
   When running NCSTI to create these LDAs and subprograms, the DC and ID characters must be set to the default (DC=. and ID=,).

4. Enter "L" in Code.

   The Create Performance LDAs window is displayed. For example:

```
INSTALL2        ***** N A T U R A L   C O N S T R U C T *****
Feb 27                   - Create Performance LDAs -                10:18 AM

NOTE: You must be in library SYSCSTnn (where nn represents
      the language number) in order to execute this function.
      This step may be repeated for as many languages
      as desired.

You are currently in library: SYSCST01
About to create performance LDAs for language: 1
Press ENTER to continue - any PF-key to stop.
```

5. Press Enter.

   A confirmation window is displayed. For example:

```
INSTALL2        ***** N A T U R A L   C O N S T R U C T *****
Feb 27                   - Create Performance LDAs -                10:21 AM

All data areas have been populated with text appropriate to
language 1  .  Please CATALL this library ( SYSCST01 )
before creating the Performance Subprograms.
```

   Note:
   You must be logged onto the SYSCST*nn* library corresponding to the language for which you are creating the LDAs. This allows multiple languages to be supported, since the LDAs are created in different libraries.

6. Press Enter.
7. Perform a CATALL on this library, ensuring that all 10 LDAs are cataloged successfully.
8. Log onto the SYSCST library.
9. Enter "NCSTI" at the Natural prompt.

   The Natural Construct Installation main menu is displayed.

10. Enter "I" in Code.

    The Create Performance Subps window is displayed. For example:

```
CSTTRANS        ***** N A T U R A L   C O N S T R U C T *****
Feb 27                   - Create Performance Subps -               10:24 AM

NOTE:  This function must be executed from library SYSCST

Enter the language number for which you would like
performance subprograms generated: __
(Press any PF-key to stop)
```

11. Enter the number of the language for which you have created performance LDAs.

    Natural Construct creates object-only performance subprograms for the specified language.

12. Copy the performance subprograms from the SYSCST library to the SYSLIBS library.

    These modules begin with “CZ” and end with the *Language value for the language in which you are installing (for example, CZHOBJ2 for German).

13. Log onto the SYSCSTX library and edit the CSXDEFLT subprogram as follows:

    - Set the PERFORMANCE default to TRUE (must be in uppercase). For example:

```
**SAG DEFINE EXIT GENERATE-CODE
*
* Your code to implement defaulting for your CST models.
DECIDE ON FIRST VALUE CSADEFLT.PARM-NAME
  VALUE 'PERFORMANCE'
    ASSIGN CSADEFLT.PARM-VALUE = 'TRUE'
  NONE
    IGNORE
END-DECIDE
**SAG END-EXIT
```

    - Save the CSXDEFLT subprogram in the SYSCSTX library.
    - Use the Natural SYSMAIN utility to copy CSXDEFLT to the SYSCST library.
    - Catalog CSXDEFLT in the SYSCST library.
    - Use the SYSMAIN utility to copy the CSXDEFLT object code to the SYSLIBS library.