

General Information

The aim of the Con-nect Application Programming Interface is to provide you with an alternative method of accessing information within Con-nect. You may need such access for any number of reasons; for example, to modify the company address, phone number or zip code.

To achieve a flexible but safe approach, Con-nect offers a series of parameter-driven API subprograms; each with a specific purpose. Using the API subprograms, you can create new objects, update existing objects, or simply retrieve information about existing objects.

Although each API subprogram has a special purpose, the ways in which an API subprogram can be used is almost unlimited. This provides you with a great deal of control and flexibility, dependent only upon how best you wish to utilize these facilities, incorporating them as desired into your own programs.

This chapter covers the following topics:

- The API Subprograms
 - Subprograms Called By An API Subprogram
 - API Data Area
 - Local Data Areas
 - Return Codes
 - Parameters
 - Password
 - Security Aspects
 - Transaction Handling
 - Extended Files
 - Time Specifications
-

The API Subprograms

The API subprograms are supplied in the library SYSCNT2. They begin with the prefix "Z-" (as in Z-GET11C).

The number in the name of an API subprogram refers to the object number. For example, 10 is the object number for distribution lists. Thus, Z-ADD10 is used to ADD a distribution list. See *Object Numbers* for a list of all object numbers.

Note:

A user can be disallowed to access TRS objects from a user-written program using the API subprograms. See *Con-nect Administration*, section *Con-nect Text Retrieval Administration* for information on the system defaults.

Subprograms Called By An API Subprogram

The subprograms that are called by the API subprograms are also supplied in the library SYSCNT2. They also begin with the prefix "Z-" (as in Z-120).

Each description of an API subprogram provides a list of subprograms that are called by the API program (including all subprograms that are then again called by the subprograms). For example:

Z-120
Z-122
Z-123
Z-1200&0

The language-specific map Z-1200&0 must be in the same library or steplib as the API subprograms. This map is invoked when a cabinet has been locked due to too many access attempts with an incorrect password. A locked cabinet can only be unlocked by a cabinet or system administrator. Natural automatically replaces the ampersand (&) in Z-1200&0 with the current value of *LANGUAGE. Thus, the English name for this map is Z-120010.

If Con-nect Text Retrieval and/or the transport service is installed, additional subprograms are called from most API subprograms.

- **Con-nect Text Retrieval**

The following subprograms are called (in addition to those listed with the description of an API subprogram):

Z-283
Z-284

- **Transport Service**

The following subprograms are called (in addition to those listed with the description of an API subprogram):

X-AMAIL
YA-CLOSE
YA-GETDA
YA-OPCR
YA-PUTDA

YA-PUTRE
YC-GET
YC-PUT
YCCORDAT
YCDMPSDE
YCLOG
YCSETUP
YCTCDAT
YFN5
YH-BCALL
YH-GETID
YH-GMT
YH-RMOBJ
YL-01
YL-02

If an application is not executed in the library SYSCNT2, the API subprograms and all subordinate subprograms should be copied to the appropriate steplib prior to execution.

API Data Area

The most important profile data such as the cabinet name or the current time zone are internally stored in an API data area which is available to all API subprograms. For these profile data, the database is not accessed each time an API subprogram is called. Rather, it is accessed only if a different cabinet is used while calling several API subprograms. Thus, performance is significantly improved.

If for any reason, the above-mentioned profile data are not available, they are automatically restored by the API subprogram.

The prerequisite for this is that Con-form has been linked to Con-nect.

Local Data Areas

Local data areas for all API subprograms are supplied in the library SYSCNT2. They are provided for your convenience.

Each local data area begins with the prefix "L-" and is followed by the corresponding API subprogram name. Thus, L-ADD05 is the local data area for the API subprogram Z-ADD05.

The fields in a local data area have the same names as the parameters that have been used in the English documentation.

The level structure is the same in all local data areas. Thus, it is possible to use more than one local data area at a time.

Example:

```

1 LOCAL-ADD05
2 RETURN-CODE      N    2
2 CABINET          A    8
2 PASSWORD         A    8
2 CABINET-NAME     A    8
2 CABINET-TYPE     N    1
2 LAST-NAME        A   32
2 FIRST-NAME       A   32
2 INITIAL          A    1
2 DISALLOW-MAIL    A    1
2 ISN              P    8

```

To call a local data area, you just have to use its name with the DEFINE DATA LOCAL statement in your API subprogram. Example:

```

DEFINE DATA
LOCAL USING L-ADD05
END-DEFINE

```

Detailed examples are provided in the descriptions of Z-ADD05 and Z-MODADR.

Return Codes

When an API subprogram terminates successfully, either the value 0 (success) or 77 (end of list) is returned in the parameter Return-code.

When an error has occurred, the appropriate value is returned in the parameter Return-code. Each description of an API subprogram provides a list of possible return codes. See *Return Codes* for a list of all possible return codes.

Note:

You can also use the parameter Return-code to control transaction handling. See *Transaction Handling*.

Parameters

Parameter List

Each description of an API subprogram provides a list of all required parameters. For example:

Parameter	Format	In	Out	Remarks
Return-code	N2	O	X	Input -1: no ET.
Cabinet	A8	R		The cabinet which contains the wanted document.
Password	A8	R		The password of the above cabinet.
Document-name	A32	R*		The name of the document for which you want to return the description. Either Document-name or ISN, not both.
ISN	P8	R*		The ISN of the document for which you want to return the description.
Description	A60/1:4		X	The description of the document.

Note:

All parameter names which occur in the description of an API subprogram are written with an initial upper-case letter. If a parameter name consists of several words, it is hyphenated.

The "In" Column

The parameters that are marked in the "In" column of a parameter list are supplied by the user. The letter R (required) indicates that you must assign a value to the parameter. The letter O (optional) indicates that you can determine whether you want to assign a value to the parameter, or not.

The notation R* always appears pairwise in the "In" column. This means that you must specify only one of the two required parameters indicated by R* - you must not specify both parameters.

The notation O* also always appears pairwise in the "In" column. This means that you must specify only one of the two optional parameters indicated by O* - you must not specify both parameters.

The notation R/O indicates that you must specify at least one of the parameters marked with R/O. For example, when three parameters are marked with R/O, you must assign a value to one of these parameters; the other two parameters are optional.

The "Out" Column

The parameters that are marked with X in the "Out" column of a parameter list are returned by the API subprogram.

Specifying an Object Name

It is not possible to identify an object using its description (which is usually shown in parentheses). To identify an object, you must always specify its name (or ISN).

Objects in the Inbasket and Outbasket do not have a name. They are identified by their subjects.

An object name is indicated as follows in the parameter list:

Parameter	Format	In	Out	Remarks
Document-name	A32	R*		The name of the document for which you want to return the description. Either Document-name or ISN, not both.

Returning a List of Objects

The parameter Number is used in several API subprograms. It determines the maximum number of objects that are returned with each call. The default value is 20. You can define a lower value.

The parameter Number is indicated as follows in the parameter list:

Parameter	Format	In	Out	Remarks
Number	N2	O		The maximum number of objects that are returned with each call. When you do not specify a value, or when you specify a value greater than 20, the default value 20 is used.

A subprogram which is used to return a list of objects should be invoked iteratively until the return code 77 indicates the end of the list. For example:

```
REPEAT UNTIL RETURN-CODE = 77
  CALLNAT 'Z-DIS23'
  RETURN-CODE
  CABINET
  START-VALUE
  NUMBER
  PROFILE-TABLE (*)
  WORK-PARAMETER
UNTIL RETURN-CODE NE 0
END-REPEAT
```

The Work-Parameter

The Work-parameter is used in several API subprograms. It functions as a buffer if an API subprogram is called iteratively.

The Work-parameter is for internal use only. Thus, you cannot modify it while processing a specific object. Since the parameter is filled by every call, you must reset it if another object is to be processed by the same API subprogram.

The Work-parameter is indicated as follows in the parameter list:

Parameter	Format	In	Out	Remarks
Work-parameter	A46			For internal use.

Password

Private Cabinet

The "Remarks" column of the parameter list always indicates who is authorized to use an API subprogram. As a rule, any user who knows the password of a private cabinet is allowed to work in that cabinet.

Parameter	Format	In	Out	Remarks
Cabinet	A8	R		The cabinet in which the appointment is to be added.
Password	A8	R		The password of the above cabinet.

When the cabinet ID you specify is the same as the user ID with which you are currently logged in, a password is not required. When you specify a password, it is not validated.

When the cabinet ID you specify is not the same as the user ID with which you are currently logged in (i.e. you want to access a cabinet other than your own), the password is required and validated. It is not checked whether you have been authorized to use that cabinet.

Cabinet SYSCNT

The cabinet SYSCNT can only be accessed by a system administrator or supervisor.

When you specify SYSCNT as the cabinet ID and the system recognizes you as a system administrator or supervisor (as specified with the user ID and password with which you are currently logged in), a password is not required. When you specify a password, it is not validated.

Administrator Status

When an API subprogram can only be used by an administrator, this is indicated as follows in the "Remarks" column of the parameter list:

Parameter	Format	In	Out	Remarks
Cabinet	A8	R		The cabinet ID of a supervisor or system administrator.
Password	A8	R		The password of the above cabinet.

When the cabinet ID you specify is the same as the user ID with which you are currently logged in and the system recognizes you as an administrator, a password is not required. When you specify a password, it is not validated.

In this case, the parameter Cabinet is simply used to check whether the user who invokes the API subprogram has the required administrator status.

Security Aspects

Private Objects

All objects marked as private within a cabinet are ignored by the API subprograms if the private cabinet is accessed by a user other than the user to whom the cabinet has been assigned.

However, you must take care in the case of a folder which has been marked as private. The objects stored in this folder are not automatically private objects. A private folder does not appear in a list of folders, when another user works in your cabinet. However, when the user displays, for example, a list of all documents in your cabinet, the documents which are stored in a private folder are also shown in this list.

Security Levels

The parameter Security-flags is used in several API subprograms. It determines the security levels another user must have in order to read (display), modify, copy or print an object in the specified cabinet.

Parameter	Format	In	Out	Remarks
Security-flags	A1/1:4	O		The security levels for Read, Modify, Copy and Print. Values 0 to 9.

The first character specifies the security level for Read, the second for Modify, the third for Copy and the fourth for Print. The security levels can be 0 (lowest level) to 9 (highest level).

Transaction Handling

The successful termination of an API subprogram always results in an END TRANSACTION. In this case, the value 0 is returned in the parameter Return-code.

Several API subprograms allow you to control the transaction handling. Before you call the API subprogram using the CALLNAT statement, you can move the value -1 (minus 1) to the parameter Return-code so that END TRANSACTION is not issued.

When an API subprogram allows you to move the value -1 to the parameter Return-code so that END TRANSACTION is not issued, this is indicated as follows in the "Remarks" column of the parameter list:

Parameter	Format	In	Out	Remarks
Return-code	N2	O	X	Input -1: no ET.

The API subprogram only accepts the value -1. When you move any other value to the parameter Return-code, this value will be ignored and set to 0.

See the Natural documentation for further information regarding the END TRANSACTION statement.

Exception: When you reference a new mail item which is currently stored in the Inbasket, the corresponding address list is updated so that the reception status of the object indicates that it has been read. Additionally, the mail item is moved to the file Opened. In this case, END TRANSACTION is automatically issued (i.e. if you moved the value -1 to the parameter Return-code this will be ignored in this case).

Extended Files

Since Con-nect Version 3 supports extended files, all new API subprograms will use the format P10 for the parameter ISN (instead of P8).

The parameters of those API subprograms already delivered with earlier versions of Con-nect have not been modified. However, in order to avoid a Natural error when working with extended files (for example, when the ISN does not fit in the 8 digit long output field), these API subprograms have been revised as described below.

Create an Object - Transaction Handling can be Controlled

The API subprograms listed below are used to create objects. They return the ISN of the new object. You are able to control transaction handling by moving the value -1 to the parameter Return-code.

Z-ADD01
Z-ADD01A
Z-ADD01B
Z-ADD01T
Z-ADD10
Z-ADD11
Z-ADD11C
Z-ADD17
Z-ADD27
Z-ADD79
Z-ADDOBJ
Z-MAILA
Z-TRACNF
Z-TRADCA
Z-TRAI NT

The following exception applies when the ISN exceeds the value 99,999,999:

- When you use the value -1, the object is created, the ISN is not returned and END TRANSACTION is not issued.
- When you do not use the value -1, the object is not created and BACKOUT TRANSACTION is issued.

When the ISN exceeds the value 99,999,999, the return code 17 (ISN consists of more than 8 digits) is always issued - even if the object has successfully been created.

Create an Object - Transaction Handling cannot be Controlled

The API subprograms listed below are used to create objects. They return the ISN of the new object. Successful termination of these API subprograms always results in an END TRANSACTION. You are not able to control transaction handling.

Z-ADD05
Z-ADD17A

When the ISN exceeds the value 99,999,999, you can now move the value -1 to the parameter Return-code (you still cannot control transaction handling). The following applies:

- When you use the value -1, the object is created, the ISN is not returned and END TRANSACTION is issued.
- When you do not use the value -1, the object is not created and BACKOUT TRANSACTION is issued.

When the ISN exceeds the value 99,999,999, the return code 17 (ISN consists of more than 8 digits) is always issued - even if the object has successfully been created.

Return a List of Objects

The API subprograms listed below are used to return a list of objects. For each object in the list the ISN is also returned.

Z-DIS04A
Z-DIS05
Z-DIS11
Z-DIS13A
Z-SEARCH
Z-SRCH03

The following exception applies when the ISN exceeds the value 99,999,999: the value 0 is returned in the parameter ISN. The return code is still 0 (success).

Return a Specific Object

The API subprograms listed below are used to return a specific object. You can either specify the name or the ISN of the object that you want to return.

Z-FIL11
 Z-FILOB
 Z-GET01
 Z-GET01T
 Z-GETOBJ
 Z-IN11G2
 Z-IN11U1
 Z-INOBG
 Z-INOBU
 Z-MOD01A
 Z-MOD01T

When the ISN exceeds the value 99,999,999, you can only specify the object by its name and the following exception applies: the value 0 is returned in the parameter ISN. The return code is still 0 (success).

Return the Number of Objects

Z-INBKT is used to return the number of objects in the different files of the Inbasket. The following exception applies when the number of objects in a specific Inbasket file exceeds the value 99,999,999: the value 99,999,999 is returned in the appropriate parameter. The return code is still 0 (success).

Time Specifications

With Con-nect Version 3, an error occurs if you specify one of the following time values:

24:mm	Corresponds to 0:mm on the next day.
hh:60	Corresponds to hh:00 of the next hour.

Thus, you must make sure that none of your user-written programs uses the above time values.