Business Services and the Natural Construct Models

This section describes the different types of business services and how they map to the Natural Construct models. All examples used in this section are from the demo application.

Note:
For more information on the demo application, see Supplied Demo Applications.

The following table lists the service types and how they relate to the Business Service wizard and Natural Construct models:

Service Type	Business Service Wizard Option	Natural Construct Model Used
Arbsub (arbitrary subprogram)	• Based on existing subprograms/Directly enable subprogram	• Subprogram-Proxy
	• Create an empty service skeleton	• Subprogram-Proxy (and the wizard creates the skeleton subprogram)
Traditional	• Based on existing subprograms/Use traditional defaults	• Subprogram-Proxy for Object-Browse or Object-Maint (or both if they access the same file)
	• Generate new subprograms for data access/Generate compound data access service	• Object-Browse and/or Object-Maint, as well as Subprogram-Proxy for Object-Browse and/or Object-Maint
Object-Browse-Select	• Based on existing subprograms/FindBy methods	• Object-Browse-Select and Subprogram-Proxy
	• Generate new subprograms for data access/Generate single view data access service	• Object-Maint (optional and not exposed to the client), Object-Browse (not exposed to the client), Object-Browse-Select, and Subprogram-Proxy
Object-Generic	• Based on existing subprograms	• Subprogram-Proxy and Object-Generic-Subp

Notes:

1. For more information on business service types, see Business Service Types.
2. For information on subprogram proxies, see Natural Business Services Subprogram-Proxy-Client Model.
3. For information on the Object models, see Natural Construct Object Models.

This section covers the following topics:

- Parameters Exposed for Each Natural Construct Model

- Parameters Available to All Services

Parameters Exposed for Each Natural Construct Model

The following table lists each Natural Construct model used to create business services and the parameters exposed for each one:

Model	Parameters Exposed
Subprogram-Proxy	+METHOD - Parameters for the subprogram that is called by the proxy (these parameters are described with each model)
Object-Maint	- Object PDA (M*5char*A) - Key PDA (M*5char*A) - Restricted PDA (M*5char*R) - CDAOBJ2 PDA - Message parameters (CDPDA-M) - Additional parameters as specified by the developer
Object-Browse	- Key PDA (A*5char*K) - Row PDA (A*5char*D) - Restricted PDA (A*5char*P) - Standard browse parameters (CDBRPDA) - Message parameters (CDPDA-M) - Additional parameters as specified by the developer

Model	Parameters Exposed
Object-Browse-Select	- Row PDA (B*5char*NE1) - Additional row parameters as specified by the developer - Service state parameters (CDBUPDA) - Business service transaction parameters (CDBUINFO) - Business service message parameters (CDBUINFO) - Restricted PDA (A*5char*P) - Key PDA (A*5char*K) - Additional service parameters as specified by the developer
Object-Generic-Subp	#BIZ-INPUTS #BIZ-OUTPUTS #BIZ-STATE #BIZ-INPUT-OUTPUTS

Parameters Available to All Services

This section describes the parameters available to all supplied business services. The following topics are covered:

- +METHOD

- User-Defined Parameters

- Parameters Common to Object-Browse and Object-Maint Models

- Parameters Specific to the Object-Maint Model

- Parameters Specific to the Object-Browse Model

- Parameters Specific to the Object-Browse-Select Model

- Parameters Specific to the Object-Generic-Subp Model

+METHOD

Every business service requires a subprogram proxy. This subprogram:

- Converts data from the communication layer (for example, EntireX send/receive buffers, SPoD message buffer) into Natural formats

● Creates an independent variable called +METHOD containing the method that was used

If the client requests the Add method, for example, the +METHOD variable contains the value "Add" by the time the processing gets to the subprogram containing the business logic. This can be particularly useful when a subprogram handles more than one method.

To access the value in +METHOD within a subprogram, add +METHOD to the DEFINE DATA statement as an independent variable. For example:

```
* Methods for current user
   INDEPENDENT
   01 +METHOD (A32)                      /* Business Method
```

Note:
For an example of how this strategy is used, refer to the BNUM subprogram in the SYSBIZDE library. Although this subprogram was created using the Object-Generic-Subp model, the same strategy can be used with other models.

User-Defined Parameters

All services allow you to include additional parameters that are unique to your business requirements. You must add these parameters within user exits for the appropriate subprograms on the server.

Note:
If you change the parameter data area in the program editor, you must regenerate the subprogram proxy to reflect the change. To do this, open the context menu for the business service in the repository tree and select **Regenerate the proxy**.

Parameters Common to Object-Browse and Object-Maint Models

The following PDA is common to both the object browse and object maintenance subprograms:

Name	Default Natural PDA Name	Default Level 1 Name	MCUSTN Example	Description
Message	CDPDA-M	MSG-INFO	CDPDA-M	Messages from the server indicating which action took place (for example, Order added successfully).

The MSG-INFO level 1 structure in the CDPDA-M PDA contains the following variables:

Variable	Value	Format	Purpose
##MSG	Text	A79	Contains message data.
##MSG-NR	A number	N4	Contains message number in SYSERR.
##MSG-DATA	Data	A32/1:3	Contains message data substituted for the message number :*n*: notation. Up to three values can be used (:1: :2: and :3:).
##RETURN-CODE	- Blank - W - E	A1	- Information - Warning - Fatal error
##ERROR-FIELD	A field name	A32	Contains the name of the field in error; this variable can be used to highlight the appropriate field in the user interface.
##ERROR-FIELD-INDEX1, ##ERROR-FIELD-INDEX2, and ##ERROR-FIELD-INDEX3	Occurrence numbers	P3	Indicates which occurrence in an array is causing a problem. Since the error field can be an array, this variable provides the opportunity to indicate which occurrence is causing the problem.

Parameters Specific to the Object-Maint Model

Every object maintenance subprogram contains a minimum of four PDAs, which correspond to five level 1 structures.

Note:
For an example of an object maintenance subprogram, refer to the MCUSTN subprogram in the SYSBIZDE library. This Customer service is located in the DEMO domain, version 010101.

These PDAs are:

Name	Default Natural PDA Name	Default Level 1 Name	MCUSTN Example	Description
Object PDA	M*5char*A	M*5char*A	MCUSTA	An instance of data from the file that is being maintained (for example, Customer Number, Business Name).
Key PDA	M*5char*A-ID	M*5char*A	MCUSTA	The key value used to retrieve the instance of data being maintained (for example, Customer Number). Since the object maintenance subprogram moves data from the object PDA to the key PDA, the key PDA should be reserved for internal use only.
Restricted PDA	M*5char*R	M*5char*R	MCUSTR	State information that allows the server to "remember" the last object retrieved.
Message parameters	CDPDA-M	MSG-INFO	CDPDA-M	Messages from the server indicating what action took place (for example, Order added successfully).
Maintenance parameters	CDAOBJ2	CDAOBJ2	CDAOBJ2	Parameters specific to the object maintenance subprogram.

Note:
The Business Service wizard assigns up to five characters to identify the service (*5char*).

The advantage of using Natural Construct-generated code is that the generator can recognize and write the appropriate code for a Natural Construct object. The CDAOBJ2 parameter data area provides options to control the functionality of the object maintenance subprogram. The object PDA contains the key PDA as a separate level 1 structure. The object PDA includes some of the same fields as the key PDA. If the key was Customer Number, for example, the Customer Number variable is contained in the object PDA twice, once under each level 1 structure. As the key PDA is used internally, it should not be modified.

The CDAOBJ2 PDA contains the following variables:

Variable	Value	Format	Purpose
2 INPUTS			Parameter group that quickly identifies possible inputs for the object maintenance subprogram.

Variable	Value	Format	Purpose
3 #FUNCTION	GET, FORMER, NEXT, UPDATE, DELETE, STORE, EXISTS, INITIALIZE and any other values that have been coded	A15	Contains the function for an object maintenance subprogram. Typically, there is a direct relationship between the business service method and #FUNCTION. (For an example of this, refer to the methods for the Customer business service in the DEMO domain, version 010101.) The power of a Natural Construct-generated object maintenance subprogram is greatest when there are intra-object relationships between files. In an intra-relationship, lower level file records must be deleted when a higher level file record is deleted (when an order header is deleted, for example, all order lines must also be deleted). You can define intra-object relationships in Predict and Natural Construct generates the appropriate code to handle the complexity of these relationships. When an order is updated, for example, an order line is added, deleted, or updated. A Natural Construct-generated object recognizes these relationships and handles this complexity automatically.
3 FLAGS-IN	True/False	L/1:10	Indicates the number of input flags. A standard object maintenance subprogram uses five input flags, although 10 flags are available.
3 REDEFINE FLAGS-IN			Contains additional input flags. If you include additional input flags in user exit code, you can add them in this FLAGS-IN redefinition.

Variable	Value	Format	Purpose
4 #CLEAR-AFTER-UPDATE	True/False		Indicates whether to reset the object PDA after an Update action. If you want the object PDA to be reset after an Update action, set this variable to True.
4 #RETURN-OBJECT	True/False		Indicates alternate processing. Although this variable is not used in generated code, it can be used in user exit code. For example, if this variable is True and the client executes the EXISTS method, you can write user exit code to execute the GET method instead.
4 #ET-IF-SUCCESSFUL	True/False		Indicates the successful processing of an Update action. The object maintenance subprogram will only issue an END TRANSACTION statement if this variable is True and an update was performed.
4 #USE-ISN	True/False		Indicates the use of Adabas ISN values to retrieve data. If this variable is True, the ISN (rather than the key value) can be used to retrieve data. Use the USE-ISN variable when an object-browse-select subprogram retrieves data via a non-unique key. When the data is modified, the ISN can be used as the unique key to retrieve the data. For more information, see Use *ISN as the Unique Primary Key for Maintenance.
4 #BACKOUT-ISSUED	True/False		Indicates that a BACKOUT TRANSACTION statement was issued. If a BACKOUT TRANSACTION statement was issued, this variable is set to True.
2 OUTPUTS			

Variable	Value	Format	Purpose
3 FLAGS-OUT	True/False	L/1:10	Indicates the number of output flags. A standard object maintenance subprogram uses five output flags, although 10 flags are available.
3 REDEFINE FLAGS-OUT			Contains additional output flags. If you include additional output flags in user exit code, you can add them in this FLAGS-OUT redefinition.
4 #OBJECT-CONTAINS-DERIVED-DATA	True/False		Indicates whether a field is derived (i.e., is not directly from the database). If a field is derived, this variable can be set to True in user exit code.
4 #EXISTS	True/False		Indicates whether an object exists in the database after a GET, NEXT, FORMER, or EXISTS method is issued. If the object exists in the database after one of these methods is issued, this variable is set to True.

Parameters Specific to the Object-Browse Model

Every object browse subprogram contains a minimum of five PDAs.

Note:
For an example of an object browse subprogram, refer to the ACUST2N subprogram in the SYSBIZDE library. This Customer business service is located in the DEMO domain, version 010101.

These PDAs are:

Name	Default Level 1 Name	ACUST2N Example	Description
Row PDA	A*5char*D	ACUST2D	*N* rows of data from the file being browsed (for example, 20 rows containing Customer Number, Business Name, Phone Number, etc.).
Key PDA	A*5char*K	ACUST2K	All elementary components that make up the specified keys (for example, Customer Number, Business Name).
Restricted PDA	A*5char*P	ACUST2P	State information for a stateless environment. This allows the server to "remember" the position of the last call and retrieve the next *n* rows of data.
Message parameters	CDPDA-M	CDPDA-M	Messages from the server indicating what action took place (for example, Orders browsed successfully).
Browse parameters	CDBRPDA	CDBRPDA	Parameters specific to the object browse subprogram.

Note:
The Business Service wizard assigns up to five characters to identify the service (*5char*).

The advantage of using Natural Construct-generated code is that the generator can recognize and write the appropriate code for a Natural Construct object. The CDBRPDA parameter data area provides options to control the functionality of the object browse subprogram.

The CDBRPDA PDA contains the following variables:

Variable	Value	Format	Purpose
2 INPUTS			Parameter group that quickly identifies possible inputs for the object browse subprogram.
3 METHOD	0	N1	Retrieves rows. The value for this variable is generally 0, although you can use the variable to process data in a different manner if required.
3 SORT-KEY	The name of one of the keys	A32	Indicates which key should be used for this query. Since the object browse subprogram can have up to five logical keys, the data in this variable indicates which key to use. Data is returned in key order.

Variable	Value	Format	Purpose
3 HISTOGRAM	True/False	L	Indicates what is returned by the object browse subprogram. If this variable is set to True, the object browse subprogram only returns the unique values of the key and a count of how many rows contain each unique value.
3 ROWS-REQUESTED	A number	N4	Indicates the allocated number of rows assigned to an object browse subprogram. If fewer than the allocated number are required, you can enter a number in this field to specify how many rows to return. **Note:** If the number entered in this field is higher than the allocated rows, a runtime error occurs.

3 RANGE-OPTION	0, 1, 2, 3, 4, 5, 6, 7	N2	Identifies the range option. The options are:
			
			● 0 (default)
			
			   Displays the data based on the sort order. You can use wildcard symbols in the key value location (*, >, <, etc.). For example, you can use the following symbols in the Name field:
			
			   ○ A* (all names starting with A)
			
			   ○ C> (all names greater than C)
			
			   ○ M< (all names less than M)
			
			● 1 (less than)
			
			● 2 (less than or equal to)
			
			● 3 (equal to)
			
			● 4 (greater than or equal to)
			
			● 5 (greater than)
			
			● 6 (begins with)
			
			● 7 (no wildcard)

Variable	Value	Format	Purpose
3 LEADING-FIXED-COMPONENTS	A number	N2	Indicates the number of leading fixed key values for the logical key. This variable increases the default number of leading fixed key values for the logical key. All values supplied up to this number of components match the corresponding values in the returned rows.
3 USE-UNIQUE-ID	True/False	L	Indicates how to browse by a non-unique key. This variable is set when browsing by a non-unique key; it is used to simulate backwards scrolling. For example, if you are browsing by Name and want to begin at Smith, ISN number 1234, any Smith with an ISN of less than 1234 will be ignored. The value for the required unique ID must be placed in the browse key PDA. **Note:** For non-Adabas files, the primary key determines uniqueness.
2 INPUT-OUTPUTS			
3 RESTART	True/False	L	Restarts the browse from the beginning of the file. If this variable is True, the browse will start from the beginning of the file instead of from the next group of rows.
2 OUTPUTS			
3 ACTUAL-ROWS-RETURNED	A number	N4	Contains the actual number of rows returned. The number will be less than or equal to the number of rows requested.
3 END-OF-DATA	True/False	L	Indicates the end of the file. This variable is set to True when all the rows in the database have been read.

Variable	Value	Format	Purpose
3 RESTARTED	0, 1, 2, 3, 4	I1	Indicates that the browse has been restarted. A browse can be restarted if the client explicitly asks for it or when critical values change. The reason the subprogram was restarted is explained in this field based on the following values: ● 1 (explicit restart; i.e., RESTART was True) ● 2 (key information changed) ● 3 (start value changed) ● 4 (unique ID changed)

Parameters Specific to the Object-Browse-Select Model

Every object browse-select subprogram contains a minimum of five PDAs, which correspond to six level 1 structures.

Note:
For an example of an object browse-select subprogram, refer to the BCUST2N subprogram in the SYSBIZDE library. This CustomerWithContacts business service is located in the DEMO domain, version 020101.

These PDAs are:

Name	Default Natural PDA Name	Default Level 1 Name	BCUST2N Example	Description
Row PDA	Internal PDA in object browse-select subprogram	B*5char*NE1	BCUST2NE1 (in BCUST2N)	*N* rows of data from the file being browsed (for example, 20 rows containing Customer Number, Business Name, Phone Number, etc.). **Note:** A user exit is available to add additional and derived data to the row PDA.
Key PDA	A*5char*K	A*5char*K	ACUST2K	All elementary components that make up the specified keys for the object browse subprogram (for example, Customer Number, Business Name).
Restricted PDA	A*5char*P	A*5char*P	ACUST2P	State information for a stateless environment. This allows the server to "remember" the position of the last call and retrieve the next *n* rows of data in the object browse subprogram.
Business service message parameters	CDBUINFO	BUSINESS-INFO	CDBUINFO	Messages from the server that pertain to all rows (for example, 20 rows retrieved successfully).
Common service transaction parameters	CDBUINFO	CDBUINFO	CDBUINFO	Transaction logic flags indicating when a transaction has been processed or whether a back out transaction statement has been issued.
Browse-specific service state parameters	CDBUPDA	CDBUPDA	CDBUPDA	Parameters specific to the object browse subprogram.

Note:
The Business Service wizard assigns up to five characters to identify the service (*5char*).

Note:
An object browse-select subprogram requires an object browse subprogram. To access the object browse subprogram, the key PDA and restricted PDA for the object browse-select subprogram are identical to those used by the object browse subprogram.

The advantage of using Natural Construct-generated code is that the generator can recognize and write the appropriate code for a Natural Construct object. The CDBUPDA and CDBUINFO parameter data areas provide options to control the functionality of the object browse-select subprogram.

An object browse-select subprogram can optionally use an object maintenance subprogram (for example, BCUST2N uses the MCUST2N object maintenance subprogram). The parameters to call the object maintenance subprogram do not need to be exposed because all required data is contained in the row PDA. With this philosophy, you should be aware of the following considerations:

- If the object maintenance subprogram uses the hash method of record locking (as opposed to the timestamp method), the hashed value is required. To facilitate this, the object browse-select subprogram adds a field called ROW-HASH to the EXTRA-ROW-DATA parameter in the internal row PDA.

- If the object maintenance subprogram contains intra-object relationships, which translate into two or three-dimensional arrays, the row PDA becomes large and potentially insufficient since Natural can only handle three dimensions and the row PDA already has a dimension (which would then require a 4th dimension). Because of this, the object browse-select subprogram will not automatically process object maintenance subprograms containing intra-object relationships.

   Note:
   If this functionality is required, it can be handled within user exits.

Extensions of the Object Browse Row PDA

The internal row PDA (B*5char*NE1) contains the variables in the object browse subprogram's row PDA, as well as the following variables:

Variable	Format	Purpose
03 EXTRA-ROW-DATA		
04 ROW-STATE	A2	Contains variable indicating the row state. This variable passes internal actions and messages between the client and the server. Existing states can be found in CDSTATE. For a list of valid values, see ROW-STATE Values.
04 ROW-HASH	B20	Contains the hashed value of the row when it was populated by the object browse subprogram. This variable is for internal use only; do not change the value. The hashed value is compared with the hashed value of the object maintenance subprogram data when the object is retrieved and locked. If the values are the same, no data maintenance has taken place between the time the object was displayed in the row and the time it was locked for data maintenance. If this variable is not in the extra row data, it is not required either because the object browse-select subprogram is not using an object maintenance subprogram to do data maintenance or because the object maintenance subprogram is using the timestamp or log counter method of optimistic record locking. With this method, the required data is contained in the object browse PDA and no extra variables are required.
04 ROW-ID	N5	Maintains state with the .NET client dataset. This variable is for internal use only; do not change the value.
04 ROW-ERROR-DATA		Contains error information at the row level. If an object maintenance subprogram is used with an object browse-select subprogram, this information would typically be contained in the CDPDA-M variables after a call to the object maintenance subprogram.
05 ##ERROR-FIELD	A32	Contains the name of the field that is causing the error.
05 ##MSG-NR	N4	Contains the message number used to retrieve the error from SYSERR.
05 ##MSG	A79	Contains the row message (typically from CDPDA-M) after maintenance has been performed on the row.

ROW-STATE Values

The available values for the ROW-STATE variable are:

Value	Row State
blank	#BLANK-STATE
A	#ADD-ROW
D	#DELETE-ROW
E	#ROW-ERROR
G	#GET-ROW
U	#UPDATE-ROW
AB	#ADD-BACKED-OUT
AP	#ROW-ADDED-PENDING
AS	#ROW-ADDED-SUCCESSFULLY
DB	#DELETE-BACKED-OUT
DP	#ROW-DELETED-PENDING
DS	#ROW-DELETED-SUCCESSFULLY
GS	#RETRIEVED-SUCCESSFULLY
IS	#INVALID-STATE
IU	#INTERVENING-UPDATE
RM	#RESTRICTED-METHOD
UB	#UPDATE-BACKED-OUT
UP	#ROW-UPDATED-PENDING
US	#ROW-UPDATED-SUCCESSFULLY

Subsets of the Object Browse Standard PDAs

An object browse-select subprogram includes several subsets of the standard PDAs for an object browse subprogram.

The CDBUINFO PDA (in the BUSINESS-INFO level 1 structure) contains the following subset of variables in the CDPDA-M PDA:

- ##MSG

- ##MSG-NR

- ##RETURN-CODE

For more information, see Parameters Common to Object-Browse and Object-Maint Models.

The CDBUPDA PDA contains the following subset of variables in the CDBRPDA PDA:

- 2 INPUTS

- 3 RANGE-OPTION

- 2 INPUT-OUTPUTS

- 3 RESTART

- 3 ACTUAL-ROWS-RETURNED

- 2 OUTPUTS

- 3 END-OF-DATA

- 3 RESTARTED

For more information, see Parameters Specific to the Object-Browse Model.

Additional Standard PDA Variables

The CDBRPDA PDA contains the following additional variables for an object browse-select subprogram:

Variable	Value	Format	Purpose
2 ##TRANSACTION	Blank, 1, 2, 3, 4, or 5	A1	Determines where a transaction is completed. The following values are available: • Blank (default) Typically, the default is the aggressive row object, but this can be overridden in the object browse-select subprogram (determined by the service coder). • 1 (aggressive row object) Indicates that an END TRANSACTION statement is issued after each row has been updated; if an error occurs while maintaining a row, processing will continue. • 2 (passive row object) Indicates that an END TRANSACTION statement is issued after each row has been updated; if an error occurs, it is noted and all processing will end immediately. • 3 (business service object) Indicates that an END TRANSACTION statement is issued at the end of the business service processing; if an error occurs, it is noted and any changes to previous rows are backed out. • 4 (client controlled object) Indicates that the server does not issue an END TRANSACTION statement. This is helpful if the transaction spans more than one database and/or environment. The END TRANSACTION statement can be issued as an independent client call. • 5 (unique object) Indicates that there is unique transaction logic which requires custom coding.
2 #BACKOUT	True/False	L	Indicates whether a back out transaction statement has been issued. If a back out has been issued, this variable is set to True.

Parameters Specific to the Object-Generic-Subp Model

Every object generic subprogram contains a minimum of one PDA, including four level 1 structures that are available when the Categorize parameters option is selected (which is recommended for advanced business service developers):

- #BIZ-INPUTS

- #BIZ-OUTPUTS

- #BIZ-STATE

- #BIZ-INPUT-OUTPUTS

When the Categorize parameters option is not selected, the PDA is similar to those used for the subprograms that are being wrapped. The only difference is that all level 1 structures in the wrapper subprogram include an "E1-" prefix. For example, the parameters for the CALC subprogram are:

```
DEFINE DATA PARAMETER
1 INPUT-DATA
  2 #FUNCTION (A30)
  2 #FIRST-NUM (N5.2)
  2 #SECOND-NUM (N5.2)
  2 #SUCCESS-CRITERIA (N5)
1 OUTPUT-DATA
  2 #RESULT (N11.2)
  2 #TIME (T)
  2 #SUCCESS (L)
END-DEFINE
```

If this subprogram is wrapped, the wrapper subprogram will contain the following parameters:

```
DEFINE DATA
  PARAMETER
  01 E1-INPUT-DATA
    02 #FUNCTION (A30)
    02 #FIRST-NUM (N5.2)
    02 #SECOND-NUM (N5.2)
    02 #SUCCESS-CRITERIA (N5)
  01 E1-OUTPUT-DATA
    02 #RESULT (N11.2)
    02 #TIME (T)
    02 #SUCCESS (L)
```

The "E1-" prefix helps distinguish between the external parameters in the wrapper subprogram and the internal local data area variables used to call the wrapped subprogram.

Notes:

1. For an example of an object generic subprogram, refer to the BSTRINGN subprogram in the SYSBIZDE library. This StringManipulation business service is located in the DEMO domain, version 010101.
2. For more information on using object generic subprograms, see Wrap Multiple Subprograms into a Business Service.