Message Class 2003 - PL/I Wrapper

The messages have the format:

2003*nnnn*

where	2003	is the message class, and
	nnnn	is the message number in the range 0000 - 9999

Overview of Messages

`20030001` | `20030002` | `20020003` | `20030004` | `20030005` | `20030006` |
`20030007` | `20030008` | `20030009` | `20030010` | `20030011` | `20030012` |
`20030101` | `20030102` | `20030103` | `20030104` | `20030105` | `20030106` |
`20030107` | `20030108` | `20030109` | `20030110` | `20030111` | `20030112` |
`20030113` | `20030114` | `20030115` | `20030116` | `20030117`

20030001 **Missing option TARGET - (BATCH_ZOS/IMS_ZOS/CICS_ZOS)**

Explanation No option `TARGET` was specified during generation. `TARGET` is a required option.

Action Specify the option `TARGET`.

20030002 **Wrong option *`target_option`* for TARGET - (BATCH_ZOS/IMS_ZOS/CICS_ZOS)**

Explanation A wrong option `TARGET` was specified during generation. Valid targets are `BATCH_ZOS`, `IMS_ZOS` and `CICS_ZOS`.

Action Specify a correct option `TARGET`.

20020003 **Unbounded arrays not supported: *`library_name`*/*`program_name`*/*`parameter_name`***

Explanation Unbounded arrays are not supported. The appendix of the message gives you the library, program and parameter name of the IDL parameter causing the error.

Action Adapt your IDL accordingly and re-generate your clients, server, stubs, etc.

20030004 ***idl_data_type*** **without maxlength not supported:** ***library_name*****/*****program_name*****/*****parameter_name***

Explanation The following IDL data types are not supported if no maximum length is given:

IDL Data Type	Description
AV	Alphanumeric variable length
KV	Kanji variable length

The appendix of the message gives you the library, program and parameter name of the IDL parameter causing the error.

Action Adapt your IDL accordingly and re-generate your clients, server, stubs, etc.

20030005 **Length for** ***idl_data_type*** **fields must be even:** ***library_name*****/*****program_name*****/*****parameter_name***

Explanation The length of the IDL data types below must be even:

IDL Data Type	Description
KV	Kanji variable length
K	Kanji

This is because the resulting PL/I data type graphic is measured in graphics (2 bytes each) and the IDL length for K, KV in bytes.The appendix of the message gives you the library, program and parameter name of the IDL parameter causing the error.

Action Adapt your IDL accordingly and re-generate your clients, server, stubs, etc.

20030006 **IDL data type** ***idl_data_type*** **not supported:** ***library_name*****/*****program_name*****/*****parameter_name***

Explanation The following IDL data types are not supported:

IDL Data Type	Description
BV	Binary variable length

Action Adapt your IDL accordingly and re-generate your clients, server, stubs, etc.

20030007 **Maximum length for *`idl_data_type`* usually *`idl_max_len`*: *`library_name`*/*`program_name`*/*`parameter_name`***

Explanation For the following IDL data types the maximum length of the resulting PL/I data types is usually restricted by the PL/I programming language.

IDL Data Type	IDL Maximum Length	Resulting Length in PL/I
K	32766 bytes	16383 graphics
KV	32766 bytes	16383 graphics
A	32767 bytes	32767 characters
AV	32767 bytes	32767 characters
B	4095 bytes	32760 bits

In this case a warning message is produced, generation of sources continues. However, it is likely that the compilation of the generated sources is not possible.The appendix of the message gives you the library, program and parameter name of the IDL parameter causing the warning message.

Action If the compilation process of the generated sources fails, adapt your IDL accordingly and re-generate your clients, server, stubs, etc.

20030008 **Invalid compressionLevel value**

Explanation The following IDL data types below are not supported if a maximum length is given:

IDL Data Type	Description
BV	Binary variable length

The Appendix of the message gives you the library, program and parameter name of the IDL parameter causing the error.

Action Adapt your IDL accordingly and re-generate your clients, server, stubs, etc.

20030009 **Precision for *`idl_data_type`* exceeds 15: *`library_name`*/*`program_name`*/*`parameter_name`***

Explanation For the following IDL data types the maximum precision of the resulting PL/I data types is usually restricted by the PL/I programming language.

IDL Data Type	PL/I Restriction
N	Depending on your compiler, 15 or 31 numeric picture characters in the picture clause.
NU	
P	Depending on your compiler, 15 or 31 digits
PU	

If the precision exceeds 15 a warning message is produced, generation of sources continues. However, depending on your compiler, compilation of the generated sources may not possible.The appendix of the message gives you the library, program and parameter name of the IDL parameter causing the warning message.

Action If the compilation process of the generated sources fails, adapt your IDL accordingly and re-generate your clients, server, stubs, etc.

20030010 **IDL data type *`type`* must be ALIGNED: *`library_name`*/*`program_name`*/*`parameter_name`***

Explanation The IDL data types below have to be aligned. This applies only for RPC servers in the environment IMS:

IDL Data Type	Description
L	Logical
B	Binary

The appendix of the message gives you the library, program and parameter name of the IDL parameter causing the warning message.

Action Adapt your IDL and re-generate your clients, server, stubs, etc.

20030011 **Total number of digits supported is 29: *`library_name`*/*`program_name`*/*`parameter_name`***

Explanation For PL/I the total number of digits (*`number1`*+*`number2`*) supported by EntireX is 29, see *Mapping IDL Data Types to PL/I Data Types*.

Action Adapt your IDL and re-generate your clients, server, stubs, etc.

20030012	**Maximum digits after decimal point supported is 7: *`library_name`*/*`program_name`*/*`parameter_name`***
Explanation	For PL/I the digits after decimal point (*`number2`*) supported by EntireX is 7, see *Mapping IDL Data Types to PL/I Data Types*.
Action	Adapt your IDL and re-generate your clients, server, stubs, etc.

20030101	**Out of memory**
Explanation	The operating system could not satisfy a memory request.
Action	Increase your memory resources and retry.

20030102	**Error calling broker stub**
Explanation	Calling the broker stub by the Generic RPC Services program (`xxxSRVI`) failed.
Action	Check if the broker stub is correctly linked and/or installed in your environment.

20030103	**ERXCOM version invalid**
Explanation	The field `COM_VERSION` in the RPC communication area is not correctly assigned. This is checked in the Generic RPC Services program (`xxxSRVI`) and Specific RPC Functions module (`xxxSRVS`).
Action	Correctly assign the `COM_VERSION` field. For information on how to declare and initialize the RPC communication area, see *Using the RPC Communication Area* under *Writing Applications with the PL/I Wrapper*.

20030104	**ERXCOM size invalid**
Explanation	The field `COM_SIZE` in the RPC communication area is not correctly assigned.
Action	Correctly assign the `COM_SIZE` field. For information on how to declare and initialize the RPC communication area, see *Using the RPC Communication Area* under *Writing Applications with the PL/I Wrapper*.

20030105	**Value of IDL PU data type must be positive**
Explanation	With a packed-decimal unsigned IDL data type, negative values cannot be sent.
Action	Either use positive values (including zero) or change the IDL data type to P packed decimal.

20030106	**Invalid function**
Explanation	The Generic RPC Services program (`xxxSRVI`) was invoked with an invalid function assigned to `COM_FUNCTION` in the RPC communication area.
Action	Correct your program. For a list of valid functions, refer to `COM_FUNCTION`.

20030107	**Internal error**
Explanation	An internal error occurred in a specific RPC functions module (`xxxSRVS`).
Action	Contact Software AG support.

20030108	**Calling Generic RPC Services (`xxxSRVI`) failed**
Explanation	Calling the Generic RPC Services (`xxxSRVI`) by the RPC stub from the Specific RPC Functions module (`xxxSRVS`) failed.
Action	Contact Software AG support.

20030109	**CICS error RESP1/RESP2 *`resp1`*/*`resp2`* [- *`additional_error_text`*]**
Explanation	CICS returned an error during an `EXEC CICS LINK` call. `RESP1` and `RESP2` are CICS return codes. The additional error text gives more information on the program called. Possible programs are: calling Generic RPC Services (`xxxSRVI`)
Action	Contact Software AG support.

20030110	**CICS PGMIDERR [- *`additional_error_text`*]**
Explanation	CICS did not find the program specified in an `EXEC CICS LINK` call. The additional error text gives more information on the program called, possible programs are: calling Generic RPC Services (`xxxSRVI`)
Action	Contact Software AG support.

20030111	**Reserved**
Explanation	None.
Action	None.

20030112 **User ID missing**

Explanation The required field `COM_CLIENT_USERID` in the RPC communication area is not given.

Action Specify a user ID. For information on the required settings in the RPC communication area, see *Using the RPC Communication Area* under *Writing Applications with the PL/I Wrapper*.

20030113 **Password missing**

Explanation The password is required in the following situations:

- If EntireX Security is installed, a password is required in the field `COM_CLIENT_PASSWORD`. See *Step 5: Optional Settings in the RPC Communication Area* under *Writing Applications with the PL/I Wrapper*.
- If the flag `COM_CLIENT_NATSECURITY` is switched on, a password is required in the field `COM_CLIENT_PASSWORD` or `COM_CLIENT_RPCPASSWORD`.

Action Depending on the situation, specify a password or see *Using Natural Security* under *Writing Applications with the PL/I Wrapper*.

20030114 **RPC Protocol reply faulty**

Explanation The reply from the RPC Server is invalid and does not follow the rules of the RPC protocol.

Action Contact Software AG support.

20030115 **Library missing**

Explanation The field `COM_SERVER_LIBRARY` in the RPC communication area is not given. The field is required for RPC conversations.

Action For information on how to work with RPC conversations, see *Conversational RPC* under *Writing Applications with the PL/I Wrapper*.

20030116 **Last conversation not ended**

Explanation The RPC communication area holds an ongoing RPC conversation which has not been ended. It is not possible to ship in parallel simple non-conversational RPC requests or open another RPC conversation using the same RPC communication area.

Action For information on how to work with RPC conversations, see *Conversational RPC* under *Writing Applications with the PL/I Wrapper*.

20030117 **No ongoing conversation**

Explanation The RPC communication area holds an ongoing RPC conversation which has not been ended. It is not possible to ship in parallel simple non-conversational RPC requests or open another RPC conversation using the same RPC communication area.

Action For information on how to work with RPC conversations, see *Conversational RPC* under *Writing Applications with the PL/I Wrapper*.