

# WSDL2RPG – FAQ

## FAQ How to Troubleshoot the “Not all data was written” Error Message

### Status of this Document

Date: 02.02.2012  
Version: 1.2  
Credits: John Fox

### Question

The web service stub crashes with an “Not all data was written” error message?

### Answer

The “16: rcvchunk: saveproc: Not all data was written!” error message is sent by the HTTP API when a user written save procedure failed to save the data that was just received from the server.

The WSDL2RPG web service stubs use their own special save procedure to analyze the data received from the server. Depending on the format of that data, the procedure either forwards the data to the MIME decoder, or directly to the eXpat parser. This is done by procedure `HTTP_receiveResponse()`.

Therefore the “Not all data was written!” error message is just the final message that you see. But it does not explain what went wrong. For example the core problem might be one of these possible errors, or something completely different:

- Range of subscript value or character string error.
- Division by zero
- Value too large
- Invalid numeric number format
- Invalid xml document received from server
- Unexpected data received from server
- Unexpected Soap element received from the web service
- etc.

### General Troubleshooting

In order to determine the actual problem you should review the job log. If there is nothing that points to the problem, you should enable the HTTP API debug log, recompile your program and call the web service again:

```
WebServicePortName_Port_setHttpDebug(  
    *ON: 'PathToYourDebugLogFile.log')
```

Hopefully, the log will reveal the cause of the problem.

### Sample 1: MCH0603 - Range of subscript value or character string error

The first thing you should do, after having noticed the “Not all data was written” error message, is to have a close look at the job log:

DSPJOBLOG

Then press F10 for “Display detailed messages” and F18 to get to the bottom of the list of messages. Now you should see something similar to this:

```
Display All Messages
System:
Job . . . : W502115003  User . . . : RADDATZ  Number . . . : 628203

Press F1 for additional message information.

>> call INEWS003TG
Range of subscript value or character string error.
16: recvdoc: saveproc: Not all data was written
-1009: Range of subscript value or character string error.
16: recvdoc: saveproc: Not all data was written
-1009: Range of subscript value or character string error.

Press Enter to continue.

F3=Exit  F5=Refresh  F12=Cancel  F17=Top  F18=Bottom

3          a          5/1
```

Now that you have identified the actual error, move your cursor to the error message and press F1 to get additional information for the message:

```
Additional Message Information

Message ID . . . . . : MCH0603      Severity . . . . . : 40
Message type . . . . . : Escape
Date sent . . . . . : 01.02.12      Time sent . . . . . : 11:48:28

Message . . . . . : Range of subscript value or character string error.
Cause . . . . . : Either the subscript value is less than the lower bound of
the array, or greater than the upper bound of the array, or the compound
operand defined a character string outside the bounds of the base character
string. The instruction ended.

Press Enter to continue.

F3=Exit  F6=Print  F9=Display message details  F12=Cancel
F21=Select assistance level

3 >> a 1/1
```

Since there is nothing interesting here you can go on with F9 to get the message details:

```
Display Message Details

Message ID . . . . . : MCH0603      Severity . . . . . : 40
Date sent . . . . . : 01.02.12      Time sent . . . . . : 11:48:28
Message type . . . . . : Escape
From . . . . . : RADDATZ      CCSID . . . . . : 65535

From program . . . . . : INEWS003
From library . . . . . : RADDATZ
From module . . . . . : INEWS00301
From procedure . . . . . : s0_anyTypeRefStart_of_s0_CityStateToZipCo
deResult
From statement . . . . . : 1053

Press Enter to continue.

F1=Help  F3=Exit  F12=Cancel

3 >> a 1/1
```

Here you get the next pieces of the puzzle:

- the name of the module the message came from
- the RPG statement number the message was sent from

Having this information, please open the source member and go to the specified line number:

```

Columns . . . . : 6 100
SEU==>
FHT * *. 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
0010.42 000000
0010.43 // Get access to current array item 000000
0010.44 if (MultiRef_doCaptureData() and MultiRef_hasItemIndex()); 000000
0010.45     x = MultiRef_getItemIndex(); 000000
0010.46 else; 000000
0010.47     if (depth = 1); 000000
0010.48         s0_anyType.x = s0_anyType.x + 1; 000000
0010.49     endif; 000000
0010.50     x = s0_anyType.x; 000000
0010.51 endif; 000000
0010.52 000000
0010.53 pCurrentItem = %addr(s0_anyType.item(x)); 000000
0010.54 000000
0010.55 select; 000000
0010.56 when (depth = 1 000000
0010.57     and 000000
0010.58     name = 'anyType' 000000
0010.59     and 000000
0010.60     namespace = 'http://www.ripedev.com/'); 000000
0010.61     if (MultiRef_isReference(attrs)); 000000

F3=Exit  F5=Refresh  F9=Retrieve  F10=Cursor  F11=Toggle  F12=Cancel
F16=Repeat find      F24=More keys

a 7/1

```

Now it is clear that “**s0\_anyType.item**” is an array with too less elements. In other words: The web service sent more elements that the stub module could place into that array. Hence you have to slightly increase the number of elements to a more reasonable value. But first you have to find the reference field. Therefore go to the top of the procedure watching for the statement where “**s0\_anyType**” is defined:

```

Columns . . . . : 6 100
SEU==>
FHT D DName+++++ETDsFrom+++To/L+++IDc.Keywords+++++Comments+++++
0010.22 D userdata * value 000000
0010.23 D depth 10I 0 value 000000
0010.24 D namespace 1024A varying const 000000
0010.25 D name 1024A varying const 000000
0010.26 D path 24576A varying const 000000
0010.27 D attrs * dim(32767) 000000
0010.28 D * const options(*varsize) 000000
0010.29 * 000000
0010.30 D x S 10I 0 inz 000000
0010.31 * 000000
0010.32 D s0_anyType DS likes(s0_RpgArrayOfAnyType_t) 000000
0010.33 D based(userdata) 000000
0010.34 * 000000
0010.35 D currentItem S like(s0_anyTypeRef_t) 000000
0010.36 D based(pCurrentItem) 000000
0010.37 * 000000
0010.38 D emptyItem S like(s0_anyTypeRef_t) 000000
0010.39 D inz 000000
0010.40 * ----- 000000
0010.41 /free 000000

F3=Exit  F5=Refresh  F9=Retrieve  F10=Cursor  F11=Toggle  F12=Cancel
F16=Repeat find      F24=More keys

a 2/9

```

Since all reference fields are defined in the stub module, you have to open it to get to the reference field:

```

Columns . . . : 6 100 Browse RADDATZ/QMSDL2RPG
SEU==> INEWS0301
FMT * *. 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
0010.29 * 000000
0010.30 D x S 10I 0 inz 000000
0010.31 * 000000
0010.32 D s0_anyType DS liked(s0_RpgArrayOfAnyType_t) 000000
0010.33 D based(userdata) 000000
0010.34 * 000000
0010.35 D currentItem S like(s0_anyTypeRef_t) 000000
0010.36 D based(pCurrentItem) 000000
0010.37 * 000000

Columns . . . : 6 100 Browse RADDATZ/QMSDL2RPG
SEU==> INEWS003
0083.00 * 000000
0084.00 D s0_RpgArrayOfAnyType_t... 000000
0085.00 D DS template 000000
0086.00 D qualified 000000
0087.00 D x 10I 0 000000
0088.00 D item like(s0_anyTypeRef_t) 000000
0089.00 D dim(DIM_A1) 000000
0090.00 * 000000

F3=Exit F5=Refresh F9=Retrieve F10=Cursor F11=Toggle F12=Cancel
F16=Repeat find F24=More keys
String s0_RpgA... found.

a 1/65

```

Voila, you got it! Constant DIM\_A1 specifies the number of items of the “s0\_anyType.item”. The last step to do is to change that constant to match your actual needs:

```

Columns . . . : 6 100 Browse RADDATZ/QMSDL2RPG
SEU==> INEWS003
FMT * *. 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
0082.00 * 000000
0083.00 * 000000
0084.00 D s0_RpgArrayOfAnyType_t... 000000
0085.00 D DS template 000000
0086.00 D qualified 000000
0087.00 D x 10I 0 000000
0088.00 D item like(s0_anyTypeRef_t) 000000
0089.00 D dim(DIM_A1) 000000
0090.00 * 000000

Columns . . . : 6 100 Browse RADDATZ/QMSDL2RPG
SEU==> INEWS003
0054.00 /COPY QMSDL2RPG,PHSDL2R90 WSDL2RPG: AppHfsg 000000
0055.00 * 000000
0056.00 * Array dimensions: 000000
0057.00 D DIM_A1 C 12 620201
0058.00 * 000000
0059.00 * Web Service specific types: 000000
0060.00 D s0_anyTypeRef_t... 000000
0061.00 D S 120A varying 000000

F3=Exit F5=Refresh F9=Retrieve F10=Cursor F11=Toggle F12=Cancel
F16=Repeat find F24=More keys

a 19/1

```

## Sample 2: RNX0105 – A character representation of a numeric value is in error

Beside the job log you should also look at the HTTPAPI debug log. Starting with v1.15 WSDL2RPG places additional debug information into that log for some errors.

For a RNX0105 error message the job log may look like this:

```
Display All Messages

Job . . . : W502115001   User . . . : RADDATZ   System:
Number . . . : 649981

>> CALL PGM(PMR007BT)
A character representation of a numeric value is in error.
Failed unmarshalling value 'a1' of array item 'index(1)'.
16: rcvchunk: saveproc: Not all data was written
-1009: Failed unmarshalling value 'a1' of array item 'index(1)'.
16: rcvchunk: saveproc: Not all data was written
-1009: Failed unmarshalling value 'a1' of array item 'index(1)'.

Press Enter to continue.

F3=Exit  F5=Refresh  F12=Cancel  F17=Top  F18=Bottom

1      >>      a      5/1
```

The messages enclosed with a **brown frame** are what you get back to the command line, if you used the generated test program to call the web service.

The message with the **green frame** is the message which was captured by WSDL2RPG and appended to the HTTPAPI debug log.

The message with the **red frame** is the original error message that was sent by the RPG runtime.

Hopefully you had enabled the debug log before you called the web service. If you did not do that, please enable the debug log and call the web service again:

```
MonthList_Port_setHttpDebug(
    *ON: '/home/raddatz/wsdl2rpg/PMR007B.log');
```

With the debug log enabled you can take advantage from the additional debug information that has been appended to the HTTPAPI debug log as shown below:

```

93 chunk size = 1365
94 get_chunk_size returned 1365
95 calling comm_blockread
96 <?xml version="1.0" encoding="UTF-8"?><soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
97 comm_blockread returned 1365
98 ** 2012-02-02-13.50.06.705000: Entering HTTP_receiveResponse()
99
100 WSDL2RPC: call stack frames:
101 Program      Library      Act.Group    Statement    Procedure
102 QUOCPP       QPDA        *DFTACTGRP   /079D
103 QUOMAIN     QPDA        *DFTACTGRP   /0603
104 QUOCMD      QSYS        *DFTACTGRP   /01D2
105 PMR007BT    RADDAT2    *NEW          9800         _QRNP_PEP_PMR007BT
106 PMR007BT    RADDAT2    *NEW          588          MonthList_getMonthsSimple
107 HTTPAPIR4   LIBHTTP    *NEW          5948         HTTP_URL_POST_RAW2
108 HTTPAPIR4   LIBHTTP    *NEW          6554         HTTP_PERSIST_POST
109 HTTPAPIR4   LIBHTTP    *NEW          4142         DO_OPER
110 HTTPAPIR4   LIBHTTP    *NEW          4468         RECVCHUNK
111 PMR007B     RADDAT2    *NEW          1185         HTTP_receiveResponse
112 WSDL2RPCRT  RADDAT2    *NEW          100100       WSDL2R97_XMLSAXParser_parse
113 EXPAT       LIBHTTP    *NEW          28           XML_Parse
114 EXPAT       LIBHTTP    *NEW          14           XML_ParseBuffer
115 EXPAT       LIBHTTP    *NEW          5           prologInitProcessor
116 EXPAT       LIBHTTP    *NEW          3           prologProcessor
117 EXPAT       LIBHTTP    *NEW          130         doProlog
118 EXPAT       LIBHTTP    *NEW          1           contentProcessor
119 EXPAT       LIBHTTP    *NEW          199         doContent
120 WSDL2RPCRT  RADDAT2    *NEW          235900       endElement
121 PMR007B     RADDAT2    *NEW          1016         impl_MonthEnd_of_impl_getMonthsSimpleResponse
122 WSDL2RPCRT  RADDAT2    *NEW          136300       WSDL2R98_UnMarshaller_v0Int
123 WSDL2RPCRT  RADDAT2    *NEW          35200        WSDL2R90_AppMsg_sendEscMsgAndDump
124 WSDL2RPCRT  RADDAT2    *NEW          33200        WSDL2R71_http_DebugLog_dumpCallStack
125 WSDL2RPCRT  RADDAT2    *NEW          20200        WSDL2R80_CallStack_dump
126
127
128
129 WSDL2RPC: USR5024: Failed unmarshalling value 'al' of array item 'index(1)'.
130 Cause . . . . . : The following error message prevented the value of field 'index' from being retrieved from XPath '/getMonthsSimpleReturn/index'.
131 Message text . . . : A character representation of a numeric value is in error.
132
133 WSDL2RPC (expat error): (-1009) Failed unmarshalling value 'al' of array item 'index(1)'.
134
135 received at line: 1
136 at column: 345
137 from procedure: WSDL2R97_XMLSAXParser_parse
138 and statement: 101800
139 ** 2012-02-02-13.50.09.526000: Failed parsing XML message
140 ** 2012-02-02-13.50.09.526000: Leaving HTTP_receiveResponse()
141 SetError() #16: rcvchunk: saveproc: Not all data was written
142 http_close(): entered
  
```

Do you miss the original error message that was sent by the RPG runtime? Well, it is there. Just scroll to the right to see it:

```

127
128
129
130 rom being retrieved from XPath '/getMonthsSimpleReturn/index'. Message text . . . : A character representation of a numeric value is in error.
131
132
  
```

Your comments are important to me! Please send me your comments about this FAQ. I will greatly appreciate it.

[thomas.raddatz@tools400.de](mailto:thomas.raddatz@tools400.de)