



CICS[®] TS Support for the WebSphere[®] Service Registry and Repository

CICS SupportPac CA1N - Installation and User's Guide

Version 2.1 – April 2008

Authors:

Martin Cocks, Mark Cocker, Daniel Millwood, Ian J Mitchell, Paul Kettley, Kate Shepherd

IBM United Kingdom Ltd.
Hursley Park, Winchester, SO21 2JN. UK.

Note! Before using this information and the product it supports, be sure to read the general information under “Notices” on page 25.

Second edition (April 2008)

This edition applies to Version 2.1 of SupportPac CA1N - CICS TS Support for the WebSphere Service Registry and Repository and to all subsequent versions, releases, and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

© Copyright IBM Corporation 2006, 2008. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Contents	3
Introduction	5
License	5
Feedback	6
Installation	7
How to download and install the SupportPac	7
Software requirements	8
Publishing WSDL files from z/OS batch to WSRR	9
Running DFHWS2SR	10
Example JCL	10
Job control statements for DFHWS2SR	11
Symbolic parameters	11
The temporary work space	11
Input parameters for DFHWS2SR	12
Parameter use	12
Parameter description	13
Completion code	14
Retrieving WSDL documents from WSRR into z/OS	15
Running DFHSR2WS	16
Example JCL	16
Job control statements for DFHSR2WS	17
Symbolic parameters	17
The temporary work space	17
Input parameters for DFHSR2WS	18
Parameter use	18
Parameter description	19
Completion code	20
Configuring SSL between the z/OS batch utilities and WSRR	21
Problem Determination	22
References	24
Notices	25
Trademarks	25

Introduction

This SupportPac™ provides the tools to streamline the job of registering, discovering, and governing the Web services resources that are generated from CICS applications. Together, these tools provide additional support for organizations intending to move to a Service-Oriented Architecture (SOA). The tools also provide additional evidence of IBM's ongoing commitment to SOA as the environment for conducting today's business operations.

The new tools enable you to automatically publish the Web Service Description Language (WSDL) documents that are derived from CICS application interfaces, from the CICS z/OS environment to the IBM WebSphere Service Registry and Repository (WSRR), where other Web applications can access them. The tools also enable you to retrieve WSDL documents from WSRR into the z/OS environment where they can be modified and 'reverse engineered' into new CICS application high-level language structures.

The SupportPac includes:

- DFHWS2SR - a z/OS batch utility to publish a WSDL document to WSRR.
- DFHSR2WS - a z/OS batch utility to retrieve a WSDL document from WSRR.

When enabling your CICS application as a Web service provider, you can use the CICS Web services assistant (DFHLS2WS) that is provided with CICS TS V3.1 and above to generate the WSDL interface from your programs high-level language structures. Use this SupportPac to automate the publication of this WSDL directly from z/OS to WSRR.

When developing a CICS application that invokes a Web service, use this SupportPac to automate the retrieval of the WSDL document from WSRR directly to z/OS. You can then use the CICS Web services assistant (DFHWS2LS) to generate the high-level language structures required to pass data to CICS when invoking the Web service.

WSRR provides a Web user interface, plug-ins for development environments, and tools that have the following capabilities:

- Store all Web services and XML documents in a central location; for example, keeping your CICS and WebSphere Web services together.
- Add metadata to Web services and relationships between Web services; for example, relating several Web services together as an application set.
- Refine WSDL documents; for example, adding and changing endpoints, as well as adding comments.
- Search for and extract WSDL documents; for example, to develop client applications and mediation components.
- Manage both the life cycle and governance of WSDL documents; for example, publicizing changes to interested parties.

License

The international license agreement for this SupportPac is available as a set of language-specific text files in the `licenses` directory within `caln.zip`. In addition the `notices.txt` file in the same directory contains important notices and information for excluded components.

Feedback

What do you think of this SupportPac? We welcome your comments by e-mail to suppacs@uk.ibm.com or via a public newsgroup.

To access the newsgroup from an application capable of using the Network News Transfer Protocol (NNTP), link to:

`news://news.software.ibm.com/ibm.software.cics.z-os`

To access the newsgroup from a Web browser, link to:

`http://www.ibm.com/developerworks/forums/dw_forum.jsp?forum=543&cat=9`

Installation

This section describes the tasks that you must perform to download and install the SupportPac and the software required to run it.

The SupportPac is aimed at experienced CICS system programmers who are responsible for installing and customizing CICS. Knowledge of downloading files from the Internet, uploading files to z/FS, and UNIX commands is assumed. Options for the **cp**, **pax**, and **chmod** commands are detailed in the “z/OS UNIX System Services Command Reference” manual.

How to download and install the SupportPac

The following steps guide you through downloading and installing the SupportPac.

1. From a workstation, download the ca1n.zip file from the IBM CICS SupportPacs Web site to your workstation:
 - 1.1. Browse to the Web site www.ibm.com/software/htp/cics/tserver/support/ or to the CICS TS V3.2 Information Center Welcome page.
 - 1.2. Click on the link "SupportPacs"
 - 1.3. Click on the link "View all: By Category"
 - 1.4. Click on the link "CA1N"
 - 1.5. Click on the link "FTP"
 - 1.6. Providing you agree to the terms and conditions shown, click on the link "I agree"
 - 1.7. Save the ca1n.zip file onto your workstation.

2. Extract the contents of ca1n.zip to a local directory. The directory should contain the following files:

\ca1n\ca1n.pdf	– this document
\ca1n\ca1n.pax.Z	– the SupportPac code in the z/OS pax archive format

3. Using **ftp**, transfer the file \ca1n\ca1n.pax.Z in binary mode to your z/OS system. For example, from a workstation command shell, enter the following commands:

```
ftp mvs.mysite.com
(login with your hostname, user ID and password)
ftp> mkdir /install-directory
ftp> cd /install-directory
ftp> bin
ftp> put caln\caln.pax.Z
ftp> quit
```

Where *install-directory* is the directory to be used for the SupportPac files on z/FS, such as "/usr/lpp/cicsts/ca1n".

4. Expand the ca1n.pax.Z file; for example, from a z/OS UNIX shell enter the following commands:

```
cd /install-directory
pax -rvf caln.pax.Z
```

5. Ensure the appropriate permissions and attributes are set for the files in the *install-directory* and its subdirectories. In particular, the files in *install-directory/bin* must have the correct execution permission set. For example, from a z/OS UNIX shell enter the following command:

```
chmod 755 /install-directory/bin/*
```

6. Copy the sample JCL files to data sets. For example, from a z/OS UNIX shell enter the following command:

```
cp -T /install-directory/sampleJCL/* "//'hlq.CA1N.JCL'"
```

Substitute *hlq.CA1N.JCL* for the data set where you want to store the SupportPac JCL files.

Verify that the following directories and data set libraries now exist:

Directory	Description
/install-directory/	User created high-level installation directory
/install-directory/bin	Contains z/OS UNIX shell scripts
/install-directory/classes	Contains the Java class libraries
/install-directory/sampleJCL	Contains sample JCL files

Software requirements

To use the SupportPac you must have the following software products installed at the correct version:

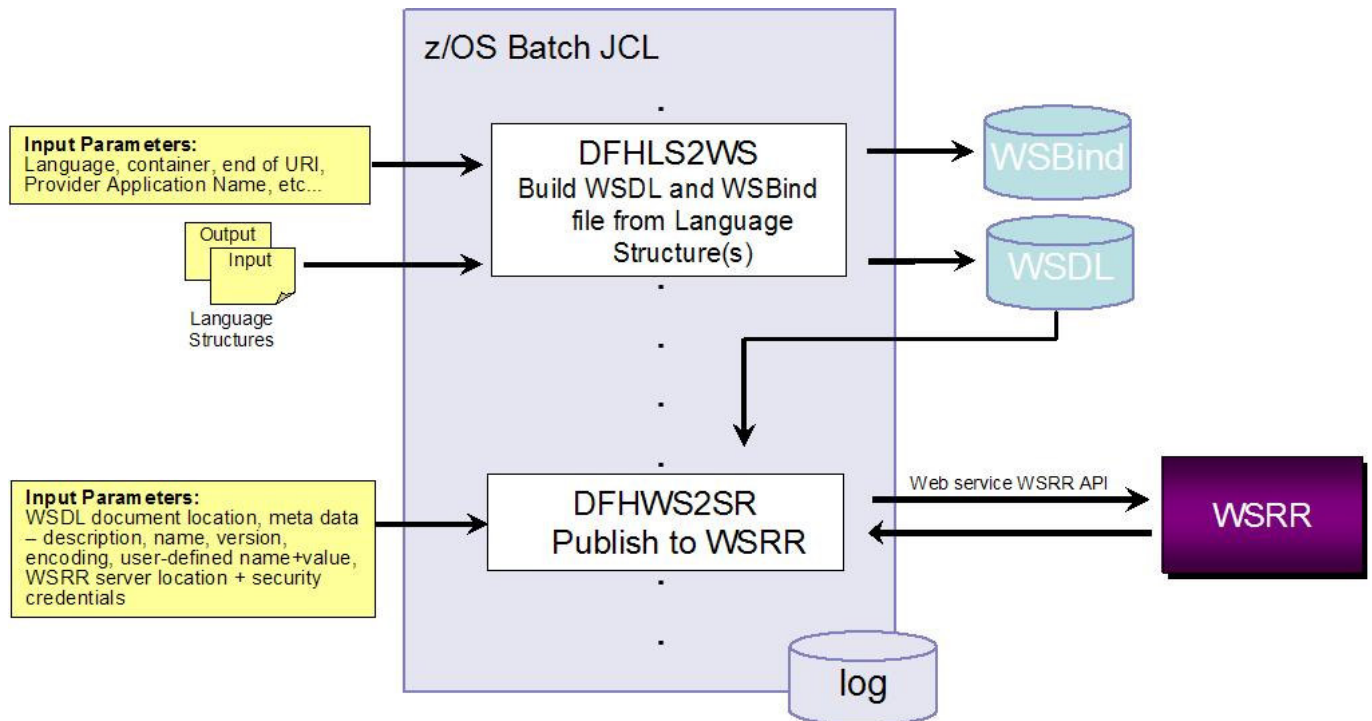
- IBM SDK for z/OS Java 2 Technology Edition V1.4
- IBM WebSphere Service Registry and Repository V1

Publishing WSDL files from z/OS batch to WSRR

Use the z/OS cataloged procedure DFHWS2SR provided by the SupportPac to publish a WSDL document stored on z/FS to WSRR, together with optional metadata.

If required, first create a WSDL document from your CICS application's high-level language structures (copybooks) using the DFHLS2WS procedure provided with CICS TS V3.1 and above.

It is possible to call DFHLS2WS followed by DFHWS2SR within the same z/OS batch JCL to keep the Web services binding file (.wsbind), WSDL, and the WSDL stored in WSRR in step with the copybooks. The following diagram illustrates the components involved in this scenario:



The user ID under which the DFHWS2SR procedure runs must have authority to:

- Read files from the SupportPac bin directory (**WORKDIR** symbolic parameter)
- Read files from the Java product directories (**JAVADIR** symbolic parameter)
- Read the WSDL file to be published (**LOCATION** parameter)
- Optionally, read the trust store and key store files (**TRUSTSTORE** and **KEYSTORE** parameters)
- Write access to the temporary directory (**TMPDIR** symbolic parameter)
- Write access the log file (**LOGFILE** parameter)
- Access to TCP/IP services to send Web services requests to the WSRR server (**HOSTPORT** parameter)

If you want to connect to WSRR using HTTPS, please read the "Configuring SSL between the z/OS batch utilities and WSRR," topic on page 21.

Running DFHWS2SR

To run the DFHWS2SR procedure you must write JCL that includes symbolic parameters and input parameters.

1. Customize the template JCL, WSDLPUB, provided in the sample JCL directory using information in the “Job control statements for DFHWS2SR” topic on page 11.
2. Submit the JCL.
3. Check the JCL output contains the message “Program 'DFHWS2SR' completed SUCCESSFULLY.” and it completed with a return code of 0.
4. If the job did not complete successfully review the messages in the JCL output to resolve the problem.

Example JCL

The following example is a customized version of the sample JCL file WSDLPUB. The two user-defined properties in this example are “Host” and “Publisher”.

```
//WSDLPUB JOB (MYSYS,AUSER),MSGCLASS=H,
// CLASS=A,NOTIFY=&SYSUID,REGION=0M
//WSDLPUB JCLLIB ORDER=(CICS.CICSTS.CA1N.JCL)
// EXEC DFHWS2SR,
// JAVADIR='java/IBM/J1.4',
// WORKDIR='/usr/lpp/cicsts/ca1n',
// TMPDIR='/u/username/tmp',
// TMPFILE='WS2SR'
//INPUT.SYSUT1 DD *
LOGFILE=/u/username/tmp/WS2SR.log
LOCATION=/usr/lpp/cicsts/cicsts31/samples/webservices/wsdl/*
placeOrder.wsdl
NAME=placeOrder
DESC=CICS TS V3.2 Catalog sample - place order service
VERSION=3
ENCODING=UTF-8
HOSTPORT=http://wsrr.my-example.server:443
USERNAME=wasadmin
PASSWORD=wasadmin
KEYSTORE=/u/username/DummyClientKeyFile.jks
KEYPWD=wasadmin
TRUSTSTORE=/u/username/DummyClientTrustFile.jks
TRUSTPWD=wasadmin
Host=IBM CICS Transaction Server
Publisher=CICS SupportPac CA1N
*/
```

Job control statements for DFHWS2SR

JOB

Initiates the job.

EXEC

Specifies the procedure name (DFHWS2SR).

DFHWS2SR requires sufficient storage to run a Java™ virtual machine (JVM). You are advised to specify `REGION=0M` on the EXEC statement.

INPUT.SYSUT1 DD

Specifies the input. The input parameters are usually specified in the input stream. However, they can be defined in a data set, or in a member of a partitioned data set.

Symbolic parameters

The following symbolic parameters are defined in cataloged procedure DFHWS2SR:

JAVADIR=path

Specifies the name of the Java directory that is used by DFHWS2SR. The value of this parameter is appended to `/usr/lpp/` giving a complete path name of `/usr/lpp/path`.

TMPDIR=tmpdir

Specifies the location of a directory in z/FS that DFHWS2SR uses as a temporary workspace. The user ID under which the job runs must have read and write permission to this directory.

The default value is `/tmp`

TMPFILE=tmpprefix

Specifies a prefix that DFHWS2SR uses to construct the names of the temporary workspace files.

The default value is `WS2SR`

WORKDIR=path

Specifies the location of the directory in z/FS in which the SupportPac is installed. The user ID under which the job runs must have read permission to this directory.

The temporary work space

DFHWS2SR creates the following three temporary files during execution:

```
tmpdir/tmpprefix.in
tmpdir/tmpprefix.out
tmpdir/tmpprefix.err
```

where

`tmpdir` is the value specified in the **TMPDIR** parameter
`tmpprefix` is the value specified in the **TMPFILE** parameter.

The default names for the files when **TMPDIR** and **TMPFILE** are not specified are:

```

/tmp/WS2SR.in
/tmp/WS2SR.out
/tmp/WS2SR.err

```

Important: DFHWS2SR does not lock access to the generated z/FS file names. Therefore, if two or more instances of DFHWS2SR run concurrently and use the same temporary workspace files, there is nothing to prevent one job overwriting the workspace files while another job is using them. This can lead to unpredictable failures.

Therefore, you are advised to devise a naming convention and operating procedures that will avoid this situation. For example, you can use the system symbolic parameter **SYSUID** to generate workspace file names that are unique to an individual user.

These temporary files are deleted before the end of the job.

Input parameters for DFHWS2SR

```

.-CCSID=Cp037-.
>>+-----+-----+-----+-----+-----+----->
   '-CCSID=value-'   '-DESC=value-'   '-ENCODING=value-'

>---HOSTPORT=scheme://+-domain name+-:port number----->
                        '-IP address--'

>--+-----+-----+-----+-----+-----+----->
   '-KEYSTORE=value-+-----+
                        '-KEYPWD=value-'

>--+-----+-----+-----+-----+-----+----->
   '-NAME=value-'    +-PropertyName=value--+

>--+-----+-----+-----+-----+-----+----->
   '-TRUSTSTORE=value-+-----+
                        '-TRUSTPWD=value-'

                        .-VERSION=1-----.
>--+-----+-----+-----+-----+-----+-----><
   '-USERNAME=value-+-----+   '-VERSION=value-'
                        '-PASSWORD=value-'

```

Parameter use

- You can specify the input parameters in any order.
- Each parameter must start at the beginning of a new line.
- A parameter (and its continuation character, if you use one) must not use characters beyond column 72.
- If a parameter is too long to fit on a single line, use an asterisk (*) character at the end of the line to indicate that the parameter continues on the next line. Everything (including spaces) before the asterisk is considered part of the parameter. For example:

```

LOCATION=/dir/*
placeOrder.wsdl

```

is equivalent to

LOCATION=/dir/placeOrder.wsdl

- A # character in the first character position of the line is a comment character. The line is ignored.
- All keywords and values are case sensitive unless otherwise specified.

Parameter description

CCSID=Cp037|value

The character encoding used to read the WSDL file from z/FS. A list of character encodings that can be specified is available in the “Supported Encodings” topic in the Java manuals. For Java 1.4, the character encodings are available from:

<http://java.sun.com/j2se/1.4.2/docs/guide/intl/encoding.doc.html>

DESC=value

Sets the description in WSRR of the WSDL document that that you want to publish.

ENCODING=value

The character set encoding of the WSDL document used by WSRR to set the “Encoding” property. If **ENCODING** is not specified, WSRR obtains this value from the `<?xml encoding="encoding_value">` declaration typically found at the beginning of the WSDL document.

HOSTPORT=scheme://{domain name|IP address}:port number

The URI of the WSRR server. The scheme can be HTTP or HTTPS. The port number defaults to 80 for the scheme HTTP and 443 for HTTPS.

KEYSTORE=value

The fully qualified name of the key store file.

KEYPWD=value

The password to decrypt the key store file.

LOCATION=value

The fully qualified name of the WSDL document that you want to publish. If a document of a different type is specified here that document will not be published.

LOGFILE=value

The fully qualified name of the file into which DFHWS2SR writes its activity log and trace information. DFHWS2SR creates the file (but not the directory structure) if it does not already exist.

NAME=value

The name of the WSDL document used by WSRR to set the “Name” property. If you do not set the **NAME** parameter, WSRR uses the unqualified name of the WSDL document.

PASSWORD=value

The password to access WSRR.

PropertyName=value

WSRR allows you to publish metadata with a WSDL document through the use of user-defined properties. Each property is a name and value pair. You can define up to 250 properties using the format *PropertyName=value*, where *PropertyName*

is the user-defined name and *value* is the string to set it to; for example:

Host=IBM CICS Transaction Server

Avoid specifying duplicate property names.

TRUSTPWD=*value*

The password to decrypt the trust store file.

TRUSTSTORE=*value*

The fully qualified name of the trust store file.

USERNAME=*value*

The user name to access WSRR that is used by WSRR to set the "Owner" property.

VERSION=1|*value*

The version number of the WSDL document that is used by WSRR to set the "Version" property.

Completion code

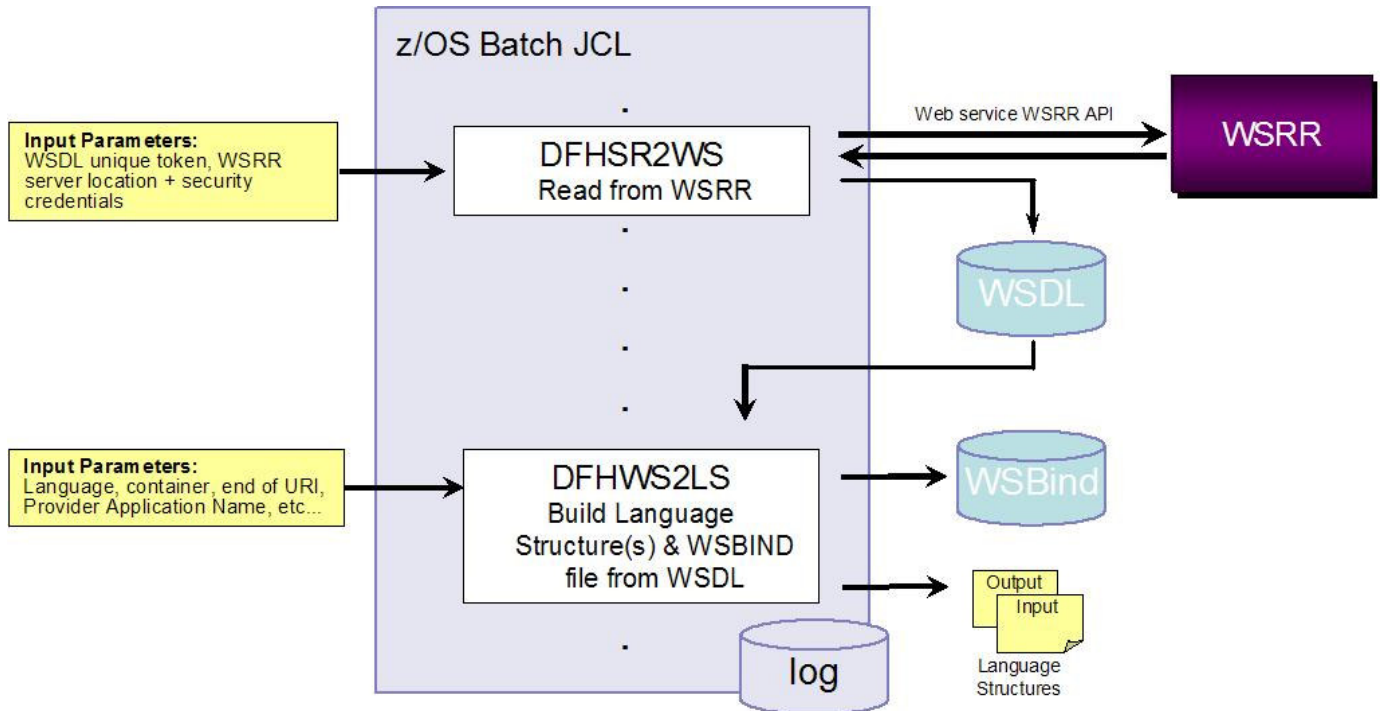
Completion code	Meaning
0	OK. The job completed successfully and only Informational messages have been issued.
4	Input Error. The job did not complete successfully. One or more error messages were issued to SYSOUT whilst validating the input parameters.
8	Input Error. The job did not complete successfully. One or more error messages were issued to SYSOUT whilst validating the input parameters.
12	Error. The job did not complete successfully. One or more error messages were issued to SYSOUT during execution.

Retrieving WSDL documents from WSRR into z/OS

You can use the z/OS cataloged procedure DFHSR2WS, provided by the SupportPac, to retrieve a WSDL document stored in WSRR and place a copy of it on z/FS.

If required, once a WSDL document has been retrieved, create high-level language structures (copybooks) from it using the DFHWS2LS procedure provided with CICS TS V3.1 and above.

It is possible to call DFHSR2WS followed by DFHWS2LS within the same z/OS batch JCL; doing so will keep the copybooks, Web service binding file, and WSDL document on z/FS in step with the WSDL document stored in WSRR. To keep a CICS application which makes use of the generated copybooks in step takes additional work. The following diagram illustrates the components involved in this scenario:



The user ID under which the DFHSR2WS procedure runs must have authority to:

- Read files from the SupportPac bin directory (**WORKDIR** symbolic parameter)
- Read files from the Java product directories (**JAVADIR** symbolic parameter)
- Optionally, read the trust store and key store files (**TRUSTSTORE** and **KEYSTORE** parameters)
- Write the WSDL document to z/FS (**LOCATION** parameter)
- Write access to the temporary directory (**TMPDIR** symbolic parameter)
- Write access the log file (**LOGFILE** parameter)
- Access TCP/IP services to send Web services requests to the WSRR server (**HOSTPORT** parameter)

If you want to connect to WSRR using HTTPS, please read the "Configuring SSL between the z/OS batch utilities and WSRR," topic on page 21.

Running DFHSR2WS

To run the DFHSR2WS procedure you must write JCL that includes symbolic parameters and input parameters.

1. Customize the template JCL, WSDLREAD, provided in the sample JCL directory using information in the “Job control statements for DFHSR2WS” topic on page 17.
2. Submit the JCL.
3. Check the JCL output contains the message “Program 'DFHSR2WS' completed SUCCESSFULLY.” and it completed with a return code of 0.
4. If the job did not complete successfully review the messages in the JCL output to resolve the problem.

Example JCL

The following example is a customized version of `hlq.CA1N.JCL(WSDLREAD)`.

```
//WSDLREAD JOB (MYSYS,AUSER),MSGCLASS=H,
// CLASS=A,NOTIFY=&SYSUID,REGION=0M
//WSDLREAD JCLLIB ORDER=(CICS.CICSTS.CA1N.JCL)
// EXEC DFHSR2WS,
//   JAVADIR='java/IBM/J1.4',
//   WORKDIR='/usr/lpp/cicsts/ca1n',
//   TMPDIR='/u/username/tmp',
//   TMPFILE='SR2WS'
//INPUT.SYSUT1 DD *
LOGFILE=/u/username/tmp/SR2WS.log
LOCATION=/u/username/tmp/
HOSTPORT=https://wsrr.my-example.server:9443
USERNAME=wasadmin
PASSWORD=password
KEYSTORE=/u/username/DummyClientKeyFile.jks
KEYPWD=wasadmin
TRUSTSTORE=/u/username/DummyClientTrustFile.jks
TRUSTPWD=wasadmin
NAME=aWSDLdocument.wsdl
NAMESPACE=http://optional/namespace/for/wsdl
VERSION=3
*/
```

Job control statements for DFHSR2WS

JOB

Initiates the job.

EXEC

Specifies the procedure name (DFHSR2WS).

DFHSR2WS requires sufficient storage to run Java virtual machine (JVM). You are advised to specify `REGION=0M` on the EXEC statement.

INPUT.SYSUT1 DD

Specifies the input. The input parameters are usually specified in the input stream. However, they can be defined in a data set, or in a member of a partitioned data set.

Symbolic parameters

The following symbolic parameters are defined in cataloged procedure DFHSR2WS:

JAVADIR=path

Specifies the name of the Java directory that is used by DFHSR2WS. The value of this parameter is appended to `/usr/lpp/` giving a complete path name of `/usr/lpp/path`

TMPDIR=tmpdir

Specifies the location of a directory in z/FS that DFHSR2WS uses as a temporary work space. The user ID under which the job runs must have read and write permission to this directory.

The default value is `/tmp`

TMPFILE=tmpprefix

Specifies a prefix that DFHSR2WS uses to construct the names of the temporary workspace files.

The default value is `SR2WS`

WORKDIR=path

Specifies the location of the directory in z/FS in which the SupportPac was installed. The user ID under which the job runs must have read permission to this directory.

The temporary work space

DFHWS2SR creates the following three temporary files during execution:

```
tmpdir/tmpprefix.in
tmpdir/tmpprefix.out
tmpdir/tmpprefix.err
```

where

`tmpdir` is the value specified in the **TMPDIR** parameter
`tmpprefix` is the value specified in the **TMPFILE** parameter.

The default names for the files when **TMPDIR** and **TMPFILE** are not specified are:

```
/tmp/SR2WS.in  
/tmp/SR2WS.out  
/tmp/SR2WS.err
```

Important: DFHSR2WS does not lock access to the generated z/FS file names. Therefore, if two or more instances of DFHSR2WS run concurrently, and use the same temporary workspace files, there is nothing to prevent one job overwriting the workspace files while another job is using them. This can lead to unpredictable failures.

Therefore, you are advised to devise a naming convention and operating procedures, to avoid this situation. For example, you can use the system symbolic parameter **SYSUID** to generate workspace file names that are unique to an individual user.

These temporary files are deleted before the end of the job.

Input parameters for DFHSR2WS

```
.-CCSID=Cp037-.  
>>+-----+--HOSTPORT=scheme://--domain name--:port number-->  
  '-CCSID=value-'                               '-IP address--'  
  
>--+-----+--LOCATION=value--LOGFILE=value-->  
  '-KEYSTORE=value-+-----+  
                        '-KEYPWD=value-'  
  
>--+-----+----->  
  '-TRUSTSTORE=value-+-----+  
                        '-TRUSTPWD=value-'  
  
>--+-----+--NAME=value-----<  
  '-USERNAME=value-+-----+  
                        '-PASSWORD=value-'  
  
>--+-----+-----><  
  '-NAMESPACE=value-'   '-VERSION=value-'
```

Parameter use

- You can specify the input parameters in any order.
- Each parameter must start at the beginning of a new line.
- A parameter (and its continuation character, if you use one) must not use characters beyond column 72.
- If a parameter is too long to fit on a single line, use an asterisk (*) character at the end of the line to indicate that the parameter continues on the next line. Everything (including spaces) before the asterisk is considered part of the parameter. For example:

```
LOCATION=/dir/*  
placeOrder.wsdl
```

is equivalent to

```
LOCATION=/dir/placeOrder.wsdl
```

- A # character in the first character position of the line is a comment character. The line is ignored.

- All keywords and values are case sensitive unless otherwise specified.

Parameter description

CCSID=Cp037|*value*

The character encoding used to write the WSDL document to z/FS. A list of character encodings that can be specified is available in the “Supported Encodings” topic in the Java manuals. For Java 1.4, the character encodings are available from:

<http://java.sun.com/j2se/1.4.2/docs/guide/intl/encoding.doc.html>

HOSTPORT=scheme://{domain name|IP address}:port number

URI of the WSRR server. The scheme must be HTTP or HTTPS. The port number defaults to 80.

KEYSTORE=*value*

Only required if the scheme specified in HOSTPORT is HTTPS. Fully qualified filename of the key store file.

KEYPWD=*value*

The password to decrypt the key store file.

LOCATION=*value*

Fully qualified name of the WSDL document read from WSRR. DFHSRWS creates the file (but not the directory structure) if it does not already exist. Any imported WSDL documents are created in this folder; no relative directory structure is created.

LOGFILE=*value*

The fully qualified name into which DFHSR2WS writes its activity log and trace information. DFHSR2WS creates the file (but not the directory structure) if it does not already exist.

NAME=*value*

The name of the WSDL document stored in WSRR.

NAMESPACE=*value*

Optionally specify the namespace qualification for the WSDL document.

PASSWORD=*value*

The password to access WSRR.

TRUSTPWD=*value*

The password to decrypt the trust store file.

TRUSTSTORE=*value*

The fully qualified name of the trust store file.

USERNAME=*value*

The user name to access WSRR.

VERSION=*value*

Optionally specify the version of the WSDL document to retrieve from WSRR.

Completion code

Completion code	Meaning
0	OK. The job completed successfully and only Informational messages have been issued.
4	Input Error. The job did not complete successfully. One or more error messages were issued to SYSOUT whilst validating the input parameters.
8	Input Error. The job did not complete successfully. One or more error messages were issued to SYSOUT whilst validating the input parameters.
12	Error. The job did not complete successfully. One or more error messages were issued to SYSOUT during execution.

Configuring SSL between the z/OS batch utilities and WSRR

The z/OS batch utilities require a key store and a trust store to be provided. These can be created using a key configuration program such as ikeyman.

WebSphere Application Server (WAS), which hosts WSRR, provides some sample key stores called DummyClientKeyFile.jks, DummyServerKeyFile.jks, and trust stores called DummyClientTrustFile.jks and DummyServerTrustFile.jks. You can use these samples for testing the z/OS batch utilities with WSRR by specifying the client key and trust files in the z/OS batch utilities and the server key and trust files in WAS. As the keys in these stores are shipped with WAS, you must replace them when used in a production environment.

An example of using self-signed certificates would require you to complete the following steps:

1. Change the keys by opening the client key file using ikeyman.
2. Delete the "websphere dummy client" certificate.
3. Create a new self-signed certificate.
4. Extract the certificate and add it into the server's trust file.
5. Open the server's key file.
6. Delete the "websphere dummy server" certificate.
7. Create a new self-signed certificate.
8. Extract the certificate and add it into the client's trust file.

When you complete these steps, the client and server are able to use SSL with the new certificates.

Problem Determination

The utilities, DFHWS2SR and DFHSR2WS, write messages to their JCL output to report any problems that are encountered whilst they are being run. If these messages do not provide enough information to resolve the error, this section provides additional information on the potential problems that could be encountered and provides a solution to resolve them.

Symptom	You are unable to start the tooling.
Cause	There are 3 setup problems that could cause this problem to occur: <ol style="list-style-type: none">1. The files are missing2. The file permissions have been set incorrectly3. The JCL is not pointing at the correct locations.
Solution	Validate that the files have been installed correctly and that the file permissions have been set as documented in the installation instructions. Validate the JCL used to run the job references the installation locations of the SupportPac.
Symptom	JCL job completes with RC = 12 and an ERROR message in the JCL output reports that parameter <i>parmname</i> is invalid or missing.
Cause	The parameter <i>parmname</i> is either invalid or missing.
Solution	Validate that the parameter is present in the JCL and that the value it is set to is correct. If it is a long string, check that it follows the rules for continuing on a new line.
Symptom	JCL job completes with RC = 12 and the second to last message in the JCL output is "Invoking the Web service".
Cause	The WSRR server specified by the HOSTPORT parameter is not accessible.
Solution	Validate that the server name and port number are correctly specified in the HOSTPORT parameter; if the values are correct check that your WSRR is running.
Symptom	JCL job completes with RC = 12 and an ERROR message in the JCL output reports that the SSL certificate is unknown.
Cause	The SSL handshake failed whilst attempting to connect to WSRR.
Solution	Make sure that SSL certificates that you are using are known to both WSRR and are installed in the key store and trust store that have been specified in the JCL parameters.
Symptom	When attempting to retrieve a WSDL document from WSRR, the job completes with RC = 4 and no WSDL document has been written to the specified location.
Cause	There was no WSDL document in WSRR that matched the NAME ,

VERSION and **NAMESPACE** values that were specified in the JCL parameters.

Solution Modify the **NAME**, **VERSION** and **NAMESPACE** parameters to match those of the WSDL document you want to retrieve from WSRR.

Symptom When attempting to retrieve a WSDL document from WSRR, the job completes with RC = 4 and a WARNING message in the JCL output reports that multiple matches were found.

Cause More than one document in WSRR matched the **NAME**, **NAMESPACE** and **VERSION** parameters values specified on the JCL input.

Solution If the first document returned by the query was not the WSDL document required, refine the **NAME**, **NAMESPACE** and **VERSION** parameters so that only the required WSDL document is returned.

Symptom When attempting to publish a WSDL document to WSRR, the job completes with RC = 12 and an ERROR message in the JCL output reports that the file was not found.

Cause The WSDL document specified by the **LOCATION** parameter could not be found. This is either because there is no WSDL document at that location or the permissions set for the WSDL document at that location mean that it is not visible to the program.

Solution Check that the **LOCATION** parameter references the WSDL document that is to be published. If the parameter is correct, change the files permissions of the WSDL document; for example, use the **chmod** command in a z/OS UNIX shell.

References

- IBM CICS Transaction Server
ibm.com/software/http/cics/tserver/v32/
 - IBM WebSphere Service Registry and Repository
ibm.com/software/integration/wsrr/
 - IBM information on Service-Oriented Architecture
ibm.com/software/solutions/soa/
 - IBM developerWorks
ibm.com/developerworks
- Article: Java Batch Jobs on z/OS and OS/390
ibm.com/developerworks/eserver/library/es-java-batchz.html
- Web site for Java on z/OS
ibm.com/servers/eserver/zseries/software/java

Notices

The provisions set out in the following two paragraphs do not apply in the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

Information contained and techniques described in this publication have not been submitted to any formal IBM test and are distributed on an "AS IS" basis.

The use or implementation of any information contained and/or of any technique described in this document is the user's responsibility and depends on the user's ability to evaluate and integrate the information and/or technique into the user's operational environment. While IBM has reviewed each item for accuracy in a specific situation, IBM offers no guarantee or warranty that the same or similar results will be obtained elsewhere. Users attempting to adapt any technique described in this document to their own environments do so at their own risk.

The information contained in this publication could include technical inaccuracies or typographical errors.

Changes are periodically made to the information contained herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any reference in this publication to an IBM licensed program or another IBM product is not intended to state or imply that only IBM's program or other product may be used. Any functionally equivalent program that does not infringe applicable intellectual property rights may be used instead of the referenced IBM licensed program or other IBM product.

The user is responsible for evaluating and verifying the operation of the material supplied in conjunction with this publication in conjunction with other products, except those expressly designated by IBM.

International Business Machines Corporation may have patents or pending patent applications covering subject-matter described in this document. The furnishing of this document does not give you any license to any such patent. You can send license inquiries, in writing, to:

The IBM Director of Licensing
International Business Machines Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

CICS	MVS	WebSphere
IBM	SupportPac	z/OS
Java	OS/390	

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the

United States, other countries, or both.

UNIX is a trademark of The Open Group in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.