

Release 1873

**Service Request 82355
Web EDB Inquiry Rewrite**

Installation Instructions

July 21, 2009
Prepared by Maxine Gerber
Revised: August 13, 2009

Information Resources & Communications
Office of the President
University of California

This document provides installation instructions for release 1873. Please review these instructions carefully before proceeding with the installation.

Note: This release is dependent on the prior installation of release 1862.

1. DB2 Installation

This portion of the installation creates DB2 objects.

1.1 Install and execute the following **new** DDL members to create new stored procedures.

Replace *WLMENV* and *COLLID* as necessary with appropriate campus values.

DDL Members	Database	Installed?	Run
SPFCTSPR			
SPHMESPR			
SPMSGSPR			
SPSPBSPR			
SPUSE00C			

1.2 Start stored procedure

Once a stored procedure has been created/modified, the procedure must be started using the DB2 START command (from SPUFI or in batch) as shown below:

- -STOP PROCEDURE (*AUTHID*.*)
- -START PROCEDURE (*AUTHID*.*)

Replace *AUTHID* in the above command with the appropriate value.

1.3 Grant access

Execute the one-time DDL member PPOT1873, replacing *WEBID* with the surrogate ID used for Web EDB Inquiry. The CURRENT SQLID must be set to the value of *AUTHID* of the authorization ID that will own the Web accessed objects. Once modified, execute the DDL member (PPOT1873).

DDL Member	Executed?
PPOT1873	

2. Copylib Member Installation

2.1 Install the following **new** copy members:

Copylib Members	Installed?
PS001	
PS001I01	
PS001O01	
PS003	
PS003I01	

PS003O01	
PS004	
PS005	
PS010	
PS011	
PS011I01	
PS011O01	
PS014	
PS016	
PS020	
PS020I01	
PS020O01	

2.2 Install **modified** copy members:

Copylib Members	Installed?
CPWSTOKN	
UCCOMMON	
UCPIAUTH	

3. *Bind Member Installation*

3.1 Install the following **new** bind members:

Bind Members	Installed?
PPFCTSPR	
PPHMESPR	
PPMSGSPR	
PPSPBSPR	
PPWWUUSE	
PS001	
PS003	
PS005	
PS011	
PS014	
UCSSIMNT	

4. *COBOL Program Preparation*

At UCOP, all COBOL programs pass through the DB2 pre-compiler, whether or not the program contains embedded SQL, to resolve INCLUDE references. Your site may have different requirements.

The “SPAS” designation indicates that a module needs to be compiled and linked into the appropriate SPAS loadlib. If only “SPAS” is indicated in the Compile column, then the module is either a called procedure, or only called by a stored procedure. The SPAS package bind differs from that used for CICS dual-use binds. If the SPAS application is running, a STOP and START PROCEDURE is required to install the new version.

"DUAL" programs must be compiled twice and linked into batch and online libraries ("LOADLIB" and "OLOADLIB" respectively). "CICS" programs must be CICS pre-compiled and compiled once and linked into OLOADLIB. "BATCH" programs must be compiled once and linked into the batch LOADLIB only.

Note: Web service programs PS* must be compiled using the integrated CICS translator feature of the Cobol compiler. They cannot be compiled with a separate CICS translator step. Sample JCL for compiling CICS programs with the options identified in the notes above can be found in the release JCL library member COMPILER.

4.1 Install, compile, and link the following **new** programs:

PROGRAM	DB2?	Compile Type	Package Bind?	Done?
PPDELTK	No	CICS	No	
PPFCTSPR	Yes	SPAS	Yes	
PPHMESPR	Yes	SPAS	Yes	
PPMSGSPR	Yes	SPAS	Yes	
PPSPBSPR	Yes	SPAS	Yes	
PPWPLT	No	CICS	No	
PPWUUSE	Yes	SPAS	Yes	
PS001	Yes	CICS (See note above)	Yes	
PS003	Yes	CICS (See note above)	Yes	
PS004	No	CICS (See note above)	No	
PS004D	No	CICS (See note above)	No	
PS005	Yes	CICS (See note above)	Yes	
PS005D	No	CICS (See note above)	No	
PS010	No	CICS (See note above)	No	
PS010D	No	CICS (See note above)	No	
PS011	Yes	CICS (See note above)	Yes	
PS014	Yes	CICS (See note above)	Yes	
PS014D	No	CICS (See note above)	No	
PS016	No	CICS (See note above)	No	
PS016D	No	CICS (See note above)	No	
PS020	No	CICS (See note above)	No	
UCSSIMNT	Yes	Batch	No	

4.2 Install, compile, and link the following **modified** programs:

PROGRAM	DB2?	Compile Type	Package Bind?	Done?
UCFNAUTH	Yes	CICS	No	
UCROUTER	Yes	CICS	Yes	
UCWMMNU	Yes	CICS	Yes	

5. Plan Binds

5.1 Bind the following plans:

Bind Member	Done?
UCSSIMNT	

6. *Obsolete Objects*

6.1 Obsolete net.data objects

All PPS net.data objects are now obsolete. Follow local procedures for removing all objects in the net.data directories listed below.

Directory	Done?
WHTML	
WIMAGES	
WINCLUDE	
WMACRO	
WSOURCE	

7. *JCL Changes*

7.1 Create JCL for the **on-request** program UCSSIMNT. Sample JCL can be found in the JCL member (RUNSSIMN).

Program	Sample JCL	Installed?
UCSSIMNT	RUNSSIMN	

8. *CICS Resource Definition Operations*

8.1 A CICS resource entry is required for the following new objects. The CICS Resource Definition Operations (RDO) is provided in CARDLIB (RDOPROD). Replace *GROUPID* with the campus group. Update the CSD. JCL member LOADRDO, which contains sample JCL for batch loading the RDO definitions. Perform a CEDA install for the group if the region is already executing.

RDO Entry	Type	Done?
UCBT	TRANSACTION	
UCBP	TRANSACTION	
T010	TRANSACTION	
UCBT	DB2ENTRY	
UCBP	DB2ENTRY	
UCBT	DB2TRAN	
UCBP	DB2TRAN	
PPDELTK	PROGRAM	
PPWPLT	PROGRAM	
PS001	PROGRAM	
PS004	PROGRAM	
PS004D	PROGRAM	
PS003	PROGRAM	
PS005	PROGRAM	
PS005D	PROGRAM	
PS010	PROGRAM	
PS010D	PROGRAM	

PS011	PROGRAM	
PS014	PROGRAM	
PS014D	PROGRAM	
PS016	PROGRAM	
PS016D	PROGRAM	
PS020	PROGRAM	

- 8.2 The CICS PLT (Program List Table) contains a list of programs which execute when the region starts. PPWPLT is a new program which must be added to the PLT. This is done by modifying the source code for the region's PLT and then assembling it, typically done by mainframe systems. A sample of PLT source code can be found in PAYDIST.R1873.TABLES(DFHPLTD4).

9. *Web Server Directories*

Three directories must be created on the web server to be used to store properties files and wsdl on the server, but external to the application. Each one will then be associated with a custom property in the Web Application Server jvm. This should be done prior to installing the web services or web applications.

Custom Property Name	Description	Directory Created?	Custom Property Defined?
APP_LOG_ROOT	directory for PPS web application logs		
EDBINQUIRY_PROPERTY_HOME	directory for the EDBInquiry properties file		
PPSWEB_PROPERTY_HOME	directory for the PPSWeb properties file		
PPSWS_WSDL_HOME	directory for the web service wsdl		

- 9.1 Create directories for each of the items in the table above.
- 9.2 Define the jvm custom properties for each of the items in the table above as follows:

- Go to the Integrated Solutions Consol (Admin consol for your web server).
- Expand “Servers” and click on “Application servers”.
- Find and click on the target server for the PPSWeb and EDBInquiry applications.
- Expand “Java and Process Management” under “Server Infrastructure”.
- Click on “Process Definition”.
- Click on “Java Virtual Machine”.
- Click on “Custom Properties”.
- Define a new custom property by clicking on “New” button. Enter the name from the table above and the directory created for this purpose.
- Save.

10. Web Service Installation

Web Service	Channel or Commarea Based	Commarea Length	WSDL Modified?	WSDL Installed?	wsbind file generated?	Done?
PS001	Channel	n/a				
PS003	Channel	n/a				
PS004	Commarea	14				
PS005	Commarea	58				
PS010	Commarea	38				
PS011	Channel	n/a				
PS014	Commarea	173				
PS016	Commarea	22				
PS020	Channel	n/a				

10.1 Modify the wsdl in the list above

- Use binary FTP to copy the files from PAYDIST.R1862.WSDL to a PC and then rename them with a .wsdl extension.
- Using any text editor, modify this file to indicate the correct url endpoint for the web services. Replace *endpoint* and *port* with the appropriate local values.
- Using binary FTP, copy the wsdl to the wsdl directory on OMVS. These will be used by a batch process to generate the wsbind files.
- Using binary FTP, copy the wsdl to the wsdl directory on the web server defined in the step titled "Web Server Directories". These will be used by the web application to determine the endpoints of the services.

10.2 Generate the wsbind files for Channel Based web services.

The wsbind files are used by CICS to correctly route a web service request to the appropriate program. They are generated using an IBM-supplied batch java utility. Sample JCL can be found in JCL member DFHWS2L1.

For each web service in the list above identified as "Channel" based, modify the following parameters in the INPUT DD statement with appropriate local values.

- **PDSLIB=//*COPYLIB*** - The process generates two copylib members, but the PPS service will not use them. However a copylib PDS must be specified here to which the user has write access, e.g. PDSLIB=//PAYMLG.TEST.COPYLIB.
- **REQMEM=DUMMY1 & RESPMEM=DUMMY2** - These are the member names which will be created in the copylib referenced above. The names may be changed or left as is, since they will never get used.
- **LOGFILE=*LOGDIR*** - The process will generate a log in an HFS directory. Change this to an appropriate directory to which the user has write access, e.g. /u/cicsdppp/wsbind/provider/PS001.log
- **WSBIND=*PICKUPDIR*/PS0xx.wsbind** - This references the "pickup" directory described above, along with the name of the wsbind file. Change *PICKUPDIR* to the local value, e.g. WSBIND=/u/cicsdppp/wsbind/provider/PS001.wsbind
- **WSDL=*WSDLDIR*/PS0xx.wsdl** - Replace *WSDLDIR* with the local value, e.g. WSDL=/u/cicsdppp/webservices/wsdl/PS001.wsdl
- **URI=*ENDPOINT*:*PORT*/PS0xx** - Replace *ENDPOINT* and *PORT* with local values, e.g. URI=http://cicsdppp.ucop.edu/PS001

10.3 Generate the wsbind files for Commarea Based web services.

The wsbind files are used by CICS to correctly route a web service request to the appropriate program. They are generated using an IBM-supplied batch java utility. Sample JCL can be found in JCL member DFHWS2L2.

For each web service in the list above identified as "Commarea" based, modify the following parameters in the INPUT DD statement with appropriate local values.

- **LOGFILE=*LOGDIR*** - The process will generate a log in an HFS directory. Change this to an appropriate directory to which the user has write access, e.g. /u/cicsdppp/wsbind/provider/PS001.log
- **WSBIND=*PICKUPDIR*/PS0xx.wsbind** - This references the "pickup" directory described above, along with the name of the wsbind file. Change *PICKUPDIR* to the local value, e.g. WSBIND=/u/cicsdppp/wsbind/provider/PS001.wsbind
- **WSDL=*WSDLDIR*/PS0xx.wsdl** - Replace *WSDLDIR* with the local value, e.g. WSDL=/u/cicsdppp/webservices/wsdl/PS001.wsdl
- **URI=*ENDPOINT*:*PORT*/PS0xx** - Replace *ENDPOINT* and *PORT* with local values, e.g. URI=http://cicsdppp.ucop.edu/PS001
- **VENDOR-APP-INTERFACE-LENGTH=*COMMAREALEN*** - Replace *COMMAREALEN* with the Commarea Length value indicated in the table above.

10.4 Install the Web services in CICS

To have CICS scan the pipeline and install URIMAPs and WEBSERVICE definitions for the new web services, enter the following command:

```
CEMT PERFORM PIPELINE(PPSPROV) SCAN
```

11. Web Application Installation

- 11.1 If customizations to the application are required, download the source archive file from the release dataset PAYDIST.R1873.PPSZIP using binary FTP. Rename the file as PPSWebApps.zip. Import this file into RAD, code and test customizations, and generate the new .ear file for deployment. See the Detail Design document for details.
- 11.2 If no customizations are required, download the .ear file from the release dataset PAYDIST.R1873.PPSEAR using binary FTP. Rename the file as PPSWebAppsEAR.ear.
- 11.3 Deploy the .ear file to the appropriate web application server. This will cause two applications to be installed: PPSWeb and EDBInquiry.

11.4 Extract PPSWeb\WEB-INF\classes\PPSWeb.properties from the .ear file and update the values below with appropriate local values:

Property	Description	Done ?
database.context.name=jdbc/@ @PPSDataSource@ @	Data source name on the server. The default is PPSDataSource, eg. jdbc/PPSDataSource	
database.authid=@ @authid@ @	DB2 Authid, eg. database.authid=PAYDP4	
edbinquiryserver=@ @EDB inquiry@ @	Server which will host EDBInquiry, eg. https://ssedev3.ucop.edu	
edbinquirycontextroot=@ @EDBInquiry@ @	Context root of an instance of EDBInquiry, eg. edbinquirycontextroot=EDBInquiry	
webpanurl=@ @web pan server@ @/PANWeb	The url of the web PAN application, eg. https://ssedev3.ucop.edu/PANWeb	
webmeriturl=@ @web merit server@ @/webmerit	The url of the web Merit application, eg. https://ssedev3.ucop.edu/webmerit	
session_timeout=@ @timeout@ @	Number of minutes for session timeout, eg. session_timeout=10	
securityprefix=@ @security prefix@ @	RACF prefix for \$ppsfunc, eg. securityprefix=PPSW	
maxnumversions=@ @max screen versions@ @	Maximum number of screen versions, eg. maxnumversions=2	

11.5 Extract EDBInquiry\WEB-INF\classes\EDBInquiry.properties from the .ear file and update the values below with appropriate local values:

Property	Description	Done ?
database.context.name=jdbc/@ @PPSDataSource@ @	Data source name on the server. The default is PPSDataSource, eg. jdbc/PPSDataSource	
database.authid=@ @authid@ @	DB2 Authid, eg. database.authid=PAYDP4	
webppserver=@ @web pps server@ @	Server which will host PPSWeb, eg. https://ssedev3.ucop.edu	
webppscontextroot=@ @PPSWeb@ @	Context root for an instance of PPSWeb, eg. webppscontextroot=PPSWeb	
header_logo=@ @header logo@ @	Name of logo file for the EDB Inquiry screen header, eg. header_logo=PayPersSys3A.jpg	
session_timeout=@ @timeout@ @	Number of minutes for session timeout, eg. session_timeout=10	
securityprefix=@ @security prefix@ @	RACF prefix for \$ppsfunc, eg. securityprefix=PPSW	
maxnumversions=@ @max screen versions@ @	Maximum number of screen versions, eg. maxnumversions=2	

11.6 Install the two modified properties files in the appropriate server directories defined in the section labeled Web Server Directories above.

11.7 If the data source referenced above in the database.context.name properties does not exist in the server, it should be added at this time.

11.8 Restart the web application server.

12. Modifications to Web PAN and Web Merit Properties File

In order for the web PAN and Merit applications to correctly return to the new applications, the properties files for these two applications must be modified.

12.1 Modify the Web Merit login.properties file

Modify the two properties below as follows, replacing `***servername***` with the name of the server which will host the PPSWeb application. Then restart the web application server on which Web Merit is running.

- `appl.menuurl=http(s)://***servername***/PPSWeb/mainMenu.do`
- `appl.logouturl=http(s)://***servername***/PPSWeb/logoff.do`

12.2 Modify the WebPAN ApplicationResources.properties file

Modify the four properties below as follows, replacing `***servername***` with the name of the server which will host the PPSWeb application. Then restart the web application server on which Web PAN is running.

- `net.data.base = http(s)://***servername:port***/PPSWeb`
- `net.data.edb.link =accessEDB.do?headeraction=locateid&browseLink=true`
- `net.data.mainmenu.link=mainMenu.do`
- `net.data.logout.link = logoff.do`

13. Security Considerations

13.1 Refer to release 1862 for information on CICS transaction security protecting transaction CPIH.

13.2 Certificate Installation

Each web server on which the new applications are installed must have a client certificate authorizing it to access the web services in the CICS region. For each web server hosting the web applications:

- A Certificate Signing Request (CSR) must generated by the server hosting the web application and sent (via email) to the Mainframe systems staff.
- Mainframe systems will use the CSR to generate a certificate and send it back to the web server administrator.
- The certificate must then be installed on the server hosting the web application and the server configured to use the certificate when invoking the web services.

- In RACF, permit READ access to CICSxxxx.CPIH to the userid associated with the certificate being used by the web application server that is making Web Service requests to the region.

14. Testing

- 14.1 Perform installation verification testing as described in the Test Plan. In addition, perform any further local testing.

15. Forms

- 15.1 Update UPAY form

Follow local procedures to implement usage of the following new **UPAY** form

- UPAY930

16. Install in Production

- 16.1 Place modified objects in production.

17. Control Table Updates

- 17.1 Execute **PPP004** to update the following DB2 CTL tables with release transactions.
- **System Messages Table (08)** – Use the transactions in CARDLIB (MSGPROD).