

CA IDMS™ 19.0 Web Services for Modernization

John Mallinson & Nakesha Newbury
CA Technologies

IUA/CA IDMS™ Technical Conference May 7-11, 2018



Abstract

The vision of CA IDMS 19.0 is to improve CA IDMS modernization capabilities through features that enable customers to expand investments in core CA IDMS applications and improve developer productivity using modern skills and industry-standard technology. This session discusses a CA IDMS 19.0 project to simplify integration of CA IDMS applications and Web services.



Copyright © 2018 CA. All rights reserved.



Agenda

- 1 LEVERAGING YOUR INVESTMENT WITH WEB SERVICES
- 2 WEB SERVICES WITHIN CA IDMS
- 3 XML GENERATION AND PARSING
- 4 WEB SERVICES DEMO
- 5 INSTALLATION/CONFIGURATION, TROUBLESHOOTING & DOC



Copyright © 2018 CA. All rights reserved.



3

Leveraging and Extending CA IDMS

Leverage
CA IDMS
databases

- Keep your database in place
- Access from web services
- Use standard interfaces

Extend
CA IDMS
applications

- Reuse your application business logic
- Invoke web services
- Provide web services



Copyright © 2018 CA. All rights reserved.



Web Services participants

- The calling program
 - Consumer
 - Requester
 - Sometimes called outbound Web services
- The responding (or 'called') program
 - Provider
 - Producer
 - Sometimes called inbound Web services
 - Sometimes called the 'service implementation'



Copyright © 2018 CA. All rights reserved.



CA IDMS™ Web Services

CA IDMS Web Services

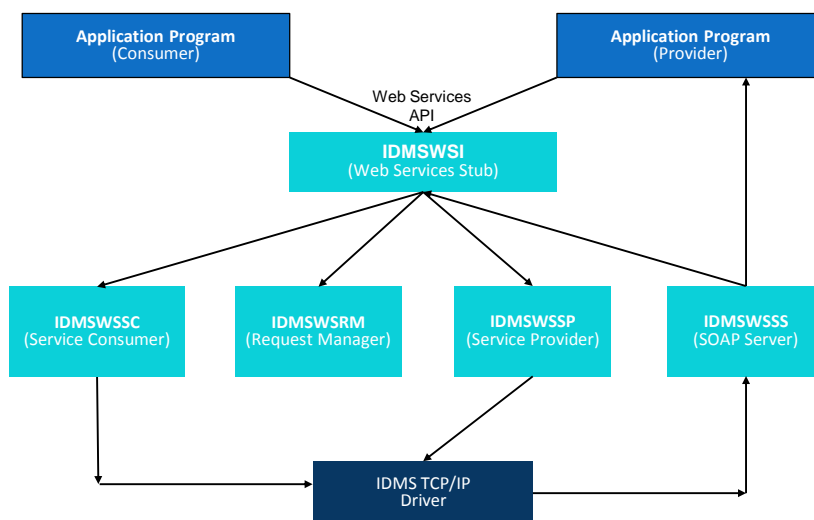
- Web Service Consumer
- Web Service Provider
- Web Services API
- Leverage and extend CA IDMS applications
 - COBOL
 - ADS
 - PL/1
 - Assembler



Copyright © 2018 CA. All rights reserved.



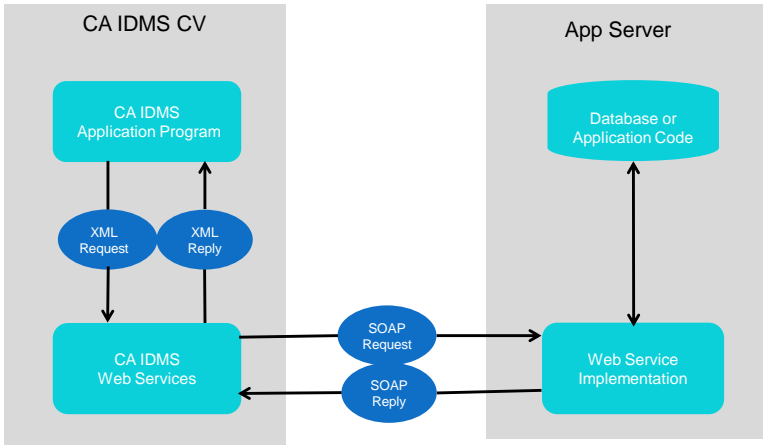
CA IDMS Web Services Internal Architecture



Copyright © 2018 CA. All rights reserved.



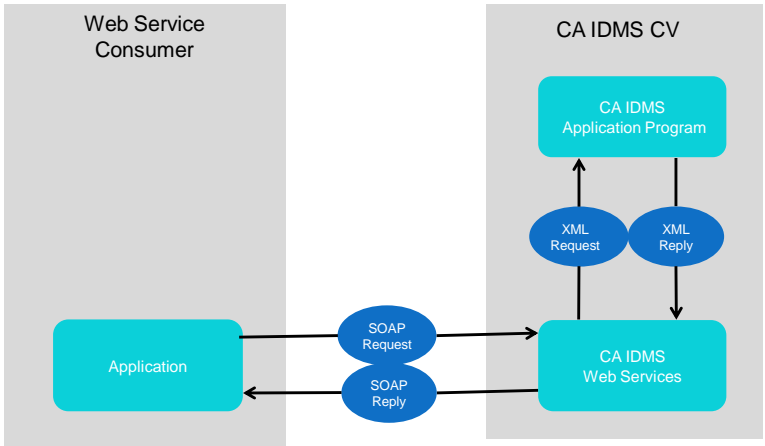
CA IDMS as a Web Service Consumer



Copyright © 2018 CA. All rights reserved.



CA IDMS as a Web Service Provider



Copyright © 2018 CA. All rights reserved.



The CA IDMS™ Web Service API

Web Services API

- Well defined, extendable interface
 - Simplifies application development
 - Isolate user code from product changes
 - Provide consistent base for product enhancement
- CA IDMS Callable Service
 - COBOL, ADS, PL/1, Assembler
- WS API Functions
 - Data transfer
 - Session management
 - Option management

Web Services API Functions

Function Code	Description	Used By
4	INITIALIZE	Consumer/Provider
8	SETOPTION	Consumer/Provider
12	GETOPTION	Consumer/Provider
16	REQUEST	Consumer
20	SEND	Provider
24	RECEIVE	Provider
28	RELEASE	Consumer/Provider



Copyright © 2018 CA. All rights reserved.



Using the Web Services API

Consumer		Provider	
Operation	API Function	Operation	API Function
Initialize Environment	WSINITIALIZE	Initialize Environment	WSINITIALIZE
Manage Options	WSGETOPTION WSSETOPTION	Manage Options	WSGETOPTION WSSETOPTION
Send Request	WSREQUEST	Receive Request	WSRECEIVE
and Receive Response		Send XML Response	WSSEND
Free Resources	WSRELEASE	Free Resources	WSRELEASE



Copyright © 2018 CA. All rights reserved.



Invoking the Web Services API

```

COBOL          CALL 'IDMSWSI' USING
                function,
                return-code,
                error-info,
                function-dependent-parameter1,
                . . .

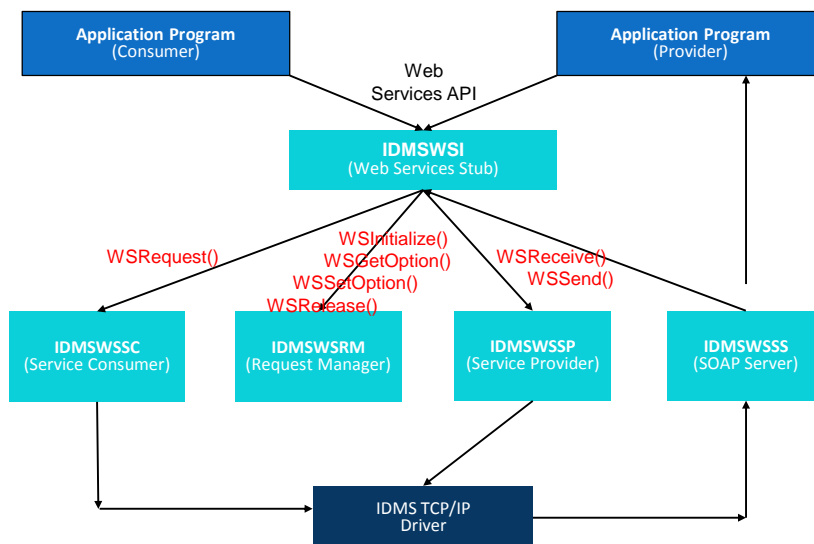
ADS            LINK TO PROGRAM 'IDMSWSI' USING
                function,
                return-code,
                error-info,
                function-dependent-parameter1,
                . . .
    
```



Copyright © 2018 CA. All rights reserved.



CA IDMS Web Services Internal Architecture



Copyright © 2018 CA. All rights reserved.



Web Services API – API Records

- Every API call will have the following 4 records at the beginning of each call and a variation of the additional records following.

•WS-FUNCTION-CODE-RECORD	API Function Code
•WS-RETURN-CODE-RECORD	API Return Code
•WS-ERROR-INFO	API Error Information
•WS-INTERFACE-VERSION-NUMBER	API Interface version
•WS-REQUEST-INFO	Request SOAP information
•WS-REQUEST-HANDLE-RECORD	Request Handle record
•WS-OPTION-NUMBER-RECORD	Get/Set Option number
•WS-OPTION-VALUE-RECORD	Get/Set Option value
•WS-REQUEST-MSG-DATA (Module)	Request Message
•WS-REQUEST-MSG-PTR-RECORD	Request Pointer
•WS-RESPONSE-MSG-DATA (Module)	Response Message
•WS-RESPONSE-MSG-PTR-RECORD	Response Pointer
•WS-REQUEST-MSG-DESCRIPTOR	Request Length
•WS-RESPONSE-MSG-DESCRIPTOR	Response Length



Copyright © 2018 CA. All rights reserved.



Web Services API – INITIALIZE (4)

- Allocate and initialize Web Services data structures

Example for COBOL, ADS and PL/I

```
WS-INITIALIZE,  
return-code,  
error-info,  
request-handle,  
interface-version.
```



Copyright © 2018 CA. All rights reserved.



Web Services API – SETOPTION (8)

- Dynamically override default settings of the CA IDMS Web services system-level options

Option Name	Number	Description
LOG-SERVICES	1	Turn Web Services Logging on or off
LOG-PROGRAM	2	Log Specific program
REQUIRE-SIGNON	3	Require CV logon
CHECK-AUTH	4	Requires that User is part of Services security Group
CONNECT-TIMEOUT	5	Specify wait time for external services
READ-WRITE-TIMEOUT	6	Specify wait time for TCP/IP calls
XML-CODE-PAGE	7	Set codepage value for XML Processing

Example: COBOL, ADS and PL/I

```
WS-SETOPTION,
return-code,
error-info,
request-handle,
option-number,
option-value.
```



Copyright © 2018 CA. All rights reserved.



Web Services API – GETOPTION (12)

- GETOPTION retrieves the values for the Web Services system-level options.

Example for COBOL, ADS and PL/I

```
WS-GETOPTION,
return-code,
error-info,
request-handle,
option-number,
option-value.
```



Copyright © 2018 CA. All rights reserved.



Web Services API – REQUEST (16)

- The REQUEST function builds and transmits a SOAP service request.

Example: COBOL, ADS and PL/I

WS-REQUEST,
return-code,
error-info,
request-handle,
request-info,
request-message-data,
request-message-descriptor,
response-message-data,
response-message-descriptor.



Copyright © 2018 CA. All rights reserved.



Web Services API – SEND (20)

- The SEND function is used to transmit a Response message to a service Consumer

Example: COBOL, ADS and PL/I

WS-SEND,
return-code,
error-info,
request-handle,
response-message-data,
response-message-descriptor.



Copyright © 2018 CA. All rights reserved.



Web Services API – RECEIVE (24)

- The RECEIVE function is used to return the address and length of an incoming Web service Request buffer

Example: COBOL, ADS and PL/I

WS-RECEIVE,
return-code,
error-info,
request-handle,
request-message-data,
request-message-descriptor.



Copyright © 2018 CA. All rights reserved.



Web Services API – RELEASE (28)

- The RELEASE function is used to terminate a Web Services request. It frees all structures allocated on behalf of the Web Services request

Example: COBOL, ADS and PL/I

WS-RELEASE,
return-code,
error-info,
request-handle.



Copyright © 2018 CA. All rights reserved.



XML Generation and Parsing

What is XML Generation and Parsing?

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  <soap:Body>
    <IDMSWSPIOperation
      xmlns="http://www.IDMSWSPI.Request.com">
        <InputFields>
          <EmplID>0472</EmplID>
          <dbname>EMPDEMO</dbname>
        </InputFields>
      </IDMSWSPIOperation>
    </soap:Body>
  </soap:Envelope>
```

What is XML Generation and Parsing?

- XML Generation
 - Uses input variables to create an XML Message
 - Final message includes your data as the payload

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body><IDMSWSPIOperation xmlns="http://www.IDMSWSPI.Request.com">
    <InputFields>
      <EmplID>0472</EmplID>
      <dbname>EMPDEMO</dbname>
    </InputFields>
  </IDMSWSPIOperation></soap:Body>
</soap:Envelope>
```

Input
EmplID: 0472
dbname: EMPDEMO

Output
EmplID: 0472
dbname: EMPDEMO

- XML Parsing
 - Extracts the payload from a given XML message
 - Payload is interpreted as individual variables for use Web services
- Two main approaches: use of IBM's COBOL functions or use of built-in SQL/XML functions



Copyright © 2018 CA. All rights reserved.



Use of SQL/XML Functions for XML Generation

```
0000-MAINLINE-BEG. } COBOL paragraph name
EXEC SQL
  set :DOC= } 01 DOC PIC X(1000) VALUE SPACES.
  xmlserialize(content
  xmlelement(name "soapenv:Envelope"
  , xmlnamespaces(
    'http://schemas.xmlsoap.org/soap/envelope/'
    as "soapenv"
  , default 'HTTP://CA.COM/HR/GLOBALXML'),
    , xmlelement(name "soapenv:Header")
    , xmlelement(name "soapenv:Body"
    , xmlelement(name "IDMSWSPIOperation"
      , xmlnamespaces('http://www.IDMSWSPI.Request.com'
      , xmlconcat(xmlelement(name "InputFields"
        , xmlconcat(xmlelement(name "EmplID", :EMPLID)
          xmlelement(name "dbname", :DBNAME)
        )))) as char(1000))
  ))))
END-EXEC.
```

Example of generation of an XML message

01 EMPLID PIC X(4).
01 DBNAME PIC X(8).



Copyright © 2018 CA. All rights reserved.



COBOL XML Generate

Enterprise COBOL for z/OS V3R3 introduced the XML GENERATE statement that can accept almost any COBOL data structure and generate XML documents.

- Can be leveraged by CA IDMS/DC COBOL programs that extract CA IDMS database data that needs to be returned in XML format
- CA Web Services COBOL programs compiled with the XMLSS compile option so that the z/OS XML System Services parser is used
- Ease of use in that extracted CA IDMS database data can be reformatted into any XML output format required
- IBM also supports XML Generate for PL/1 programs



Copyright © 2018 CA. All rights reserved.



COBOL XML Generate

Using an example from the CA Web Services Demo Provider program ...

1. A COBOL data structure is defined containing the fields extracted from the Employee Demo Data base required for the Provider Service. CA Web Services provides routines that will execute the XML Generate for the COBOL data structure 'OutputFields'. Just define the output data under this 01 level.

```
01 OutputFields.
05 EmpID          PIC X(04) VALUE SPACES.
05 EmpFirstName   PIC X(10) VALUE SPACES.
05 EmpLastName    PIC X(15) VALUE SPACES.
05 EmpStreet      PIC X(20) VALUE SPACES.
05 EmpCity        PIC X(15) VALUE SPACES.
05 EmpState       PIC X(02) VALUE SPACES.
05 EmpZip         PIC X(05) VALUE SPACES.
```

2. The data structure fields are case sensitive, the above fields will appear with upper/lower case in the XML data tag.
<EmpFirstName> </EmpFirstName>



Copyright © 2018 CA. All rights reserved.



COBOL XML Generate

Using an example from the CA Web Services Demo Provider program ...

1. The COBOL XML Generate Statement creates an XML Response from the COBOL data structure.

```
XML GENERATE WS-RESPONSE-MESSAGE
  (1:WS-RSP-MSG-BUFF-LEN)
  FROM OutputFields          Defined COBOL data structure
  COUNT IN WSPI-XML-OUT-LENGTH
  WITH ENCODING WS-CODEPAGE-VALUE
  ON EXCEPTION
    MOVE 'NO ' TO WSPI-WAS-GENERATE-SUCCESS
```

2. The resulting XML structure is stored in WS-RESPONSE-MESSAGE where it can be wrapped by a SOAP Envelope as a Service Response

```
<OutputFields><EmpID>0472</EmpID>
<EmpFirstName>ROBBY</EmpFirstName><EmpLastName>WILDER
</EmpLastName><EmpStreet>4567 E. GROWTH ST</EmpStreet>
<EmpCity>SOUTHBORO</EmpCity><EmpState>MA</EmpState><Emp
pZip>03145</EmpZip></OutputFields>
```



Copyright © 2018 CA. All rights reserved.



COBOL XML Parse

The COBOL XML Parse statement to transform XML String into COBOL data items.

XML string

```
<EmpID>0472</EmpID>
```

COBOL Parse XML string

```
XML PARSE CWA1-REPLY-BUFFER
  (1:CWA1-REPLY-BUFFER-LENGTH)
  WITH ENCODING CWA1-CODEPAGE-VALUE
  PROCESSING PROCEDURE CPA1-PARSE-XML
  ON EXCEPTION
    MOVE 'NO ' TO CWA1-WAS-PARSE-SUCCESSFUL
```

Evaluate the data tags and populate data into COBOL data structure

```
EVALUATE FUNCTION UPPER-CASE(CWA1-EDITED-ELEMENT)
  WHEN 'EMPLID'
    MOVE XML-TEXT TO WS-EMP-ID
```

Results in:

WS-EMP-ID = 0472

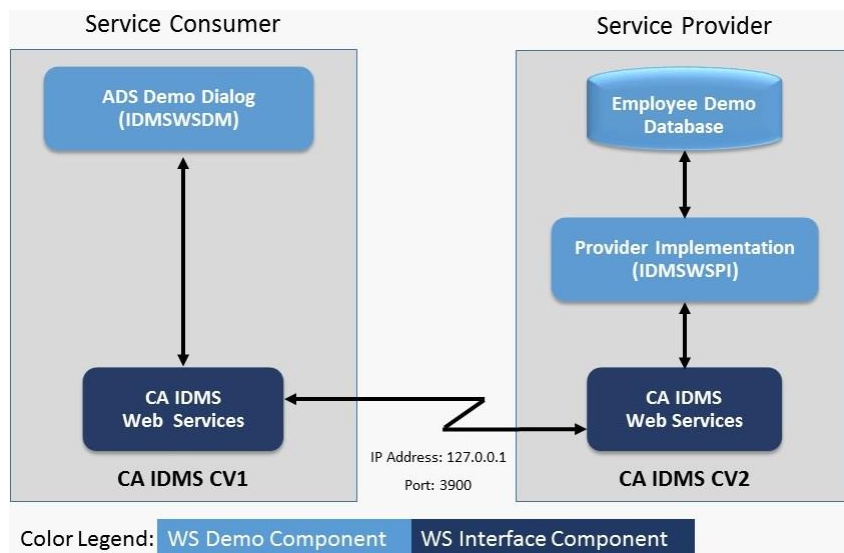


Copyright © 2018 CA. All rights reserved.



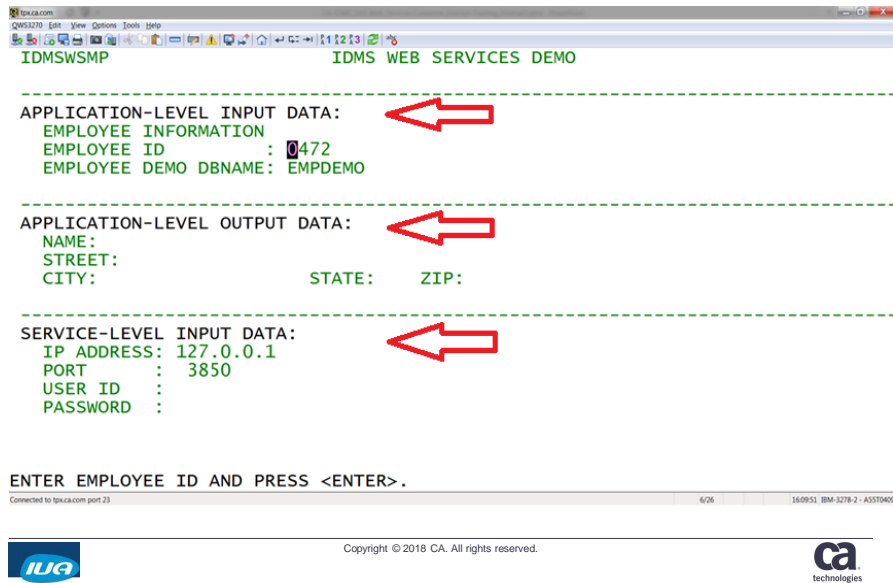
Web Services Demo Programs

CA IDMS Web Services Demo



CA IDMS Web Services Demo

- Enter task code 'IDMSWSDM'



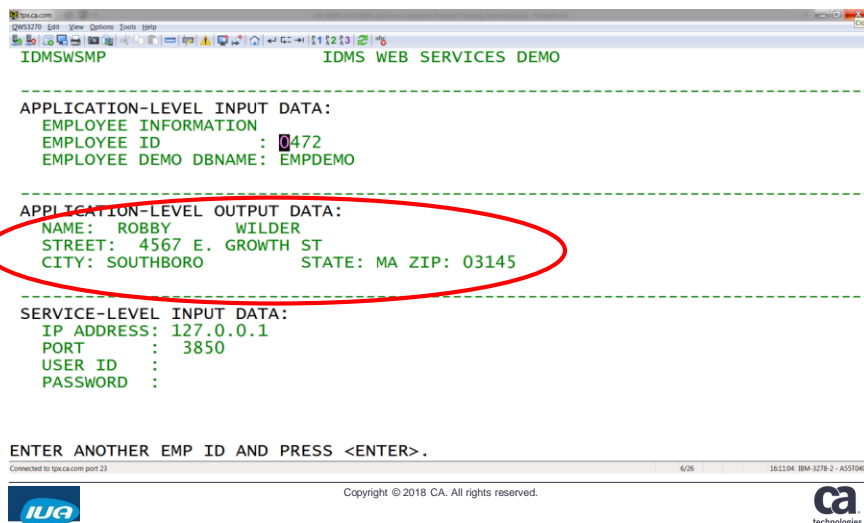
The screenshot shows a web browser window with the title 'IDMS WEB SERVICES DEMO'. The page displays the following sections:

- APPLICATION-LEVEL INPUT DATA:** (indicated by a red arrow)
 - EMPLOYEE INFORMATION
 - EMPLOYEE ID : 0472
 - EMPLOYEE DEMO DBNAME: EMPDEMO
- APPLICATION-LEVEL OUTPUT DATA:** (indicated by a red arrow)
 - NAME:
 - STREET:
 - CITY:
 - STATE:
 - ZIP:
- SERVICE-LEVEL INPUT DATA:** (indicated by a red arrow)
 - IP ADDRESS: 127.0.0.1
 - PORT : 3850
 - USER ID :
 - PASSWORD :

At the bottom, there is a prompt: 'ENTER EMPLOYEE ID AND PRESS <ENTER>.' and a status bar showing 'Connected to tpx.ca.com port 23'.

CA IDMS Web Services Demo

- CA IDMS Web Services Consumer receives reply from CA Web Services Provider Service.



The screenshot shows the same web browser window as the previous one, but with the output data populated. The 'APPLICATION-LEVEL OUTPUT DATA' section is circled in red:

- APPLICATION-LEVEL OUTPUT DATA:**
 - NAME: ROBBY WILDER
 - STREET: 4567 E. GROWTH ST
 - CITY: SOUTHBORO STATE: MA ZIP: 03145

The 'APPLICATION-LEVEL INPUT DATA' and 'SERVICE-LEVEL INPUT DATA' sections remain the same. The status bar at the bottom now shows '161104 IBM-3278-2-A3570409'.

CA IDMS Web Services Demo – Request API

- To send a Service Request for the Consumer, the ADS dialog uses the Web Service Request API

```

!* WSREQUEST()      - PERFORM A REQUEST TO CONSUME A WEB SERVICE
!*****
MOVE 16 TO WS-FUNCTION-CODE.      ! WSREQUEST()
LINK TO PROGRAM 'IDMSWSI' USING
  (WS-FUNCTION-CODE-RECORD,
   WS-RETURN-CODE-RECORD,
   WS-ERROR-INFO,
   WS-REQUEST-HANDLE-RECORD,
   WS-REQUEST-INFO,
   WSDemo-REQUEST-MSG-DATA,
   WS-REQUEST-MSG-DESCRIPTOR,
   WSDemo-RESPONSE-MSG-DATA,
   WS-RESPONSE-MSG-DESCRIPTOR) .

```

Defines Service Request SOAP information
 Defines Service Request message
 Defines Service Request Length
 Defines Service Response message
 Defines Service Response Length



Copyright © 2018 CA. All rights reserved.



Configuring Web Services

How to Install Web Services?

- Easy installation:
 1. Apply PTF
 2. Execute HOLDDATA instructions



Copyright © 2018 CA. All rights reserved.



Web Services Installation

HOLDDATA includes instructions for:

1. Adding Web Services DC messages to the dictionary.
2. Updating SYSTEM dictionary to define Web Services tasks, programs, LTERM, and PTERM.
3. Setting MULTIPLE ENCLAVE IS ON
4. Adding Web Services API records and modules to dictionaries.
5. Adding demo dialog processes, records and modules to a dictionary.
6. Adding the demo dialog map to the dictionary.
7. Generating the demo dialog, IDMSWSDM.
8. Re-start the CV



Copyright © 2018 CA. All rights reserved.



Installation, Configuration & Troubleshooting Hold Data Instructions

Changing the Listener Port prior to executing HOLDDATA step 2:

Step 2: Updating SYSTEM dictionary to define Web Services tasks, programs, LTERM, and PTERM.

- Edit IDMSAPI and/or IDMSNAPI (source in CAGJSRC library)
PORT IS <your-port-number>

```
ADD PTERM TCPWSRV
  ENABLED
  IN LINE TCPIP
  MAXIMUM ERRORS IS 3
  PRINTER CLASS IS 1
  READBUFFER
  TYPE IS LISTENER
  TASK IS RHDCNP3W MODE IS USER
  PORT IS 3850
.
```

- Edit Premap Process IDMSWSDM-PM in IDMSAPI (source in CAGJSAMP)

```
MOVE 3850 TO MAP-CV-PORT.
```



Copyright © 2018 CA. All rights reserved.



Installation, Configuration & Troubleshooting Hold Data Instructions

3) Setting MULTIPLE ENCLAVE IS ON

```
ADD PROGRAM IDMSWSI
  CONCURRENT
  DYNAMIC
  DUMP THRESHOLD IS 0
  ENABLED
  ERROR THRESHOLD IS 5
  ISA SIZE IS 0
  LANGUAGE IS COBOL
  MPMODE IS SYSTEM
  NOMAINLINE
  MULTIPLE ENCLAVE IS ON
  NEW COPY IS ENABLED
  OVERLAYABLE
  PROGRAM
  PROTECT
  REENTRANT
  NONRESIDENT
  REUSABLE
  NOSAVEAREA
.
```

```
MOD SYS xxx
MULTIPLE ENCLAVE IS ON .
GENERATE .
```

```
SYSGEN 19.0
*+ MAXIMUM ERUS IS 25
*+ MAXIMUM TASKS IS 10
*+ MESSAGE RETENTION IS 7
*+ MULTIPLE ENCLAVE IS ON
```



Copyright © 2018 CA. All rights reserved.



Web Services in CA IDMS Service Configuration

- Invoke Task Code “WSQP”

```
LOG WEB SERVICES = NO
- TO MODIFY, ENTER "WSQP LOG WEB SERVICES=XXX",
  WHERE "XXX" IS "YES" OR "NO".
LOG PROGRAM =
- TO MODIFY, ENTER "WSQP LOG PROGRAM=XXXXXXXX",
  WHERE "XXXXXXXX" IS A PROGRAM NAME OR SPACES.
REQUIRE SIGNON = NO
- TO MODIFY, ENTER "WSQP REQUIRE SIGNON=XXX",
  WHERE "XXX" IS "YES" OR "NO".
CHECK AUTHORIZATION = NO
- TO MODIFY, ENTER "WSQP CHECK AUTHORIZATION=XXX",
  WHERE "XXX" IS "YES" OR "NO".
```

- Configuration Options:
 - LOG WEB SERVICES
 - LOG PROGRAM
 - REQUIRE SIGNON
 - CHECK AUTHORIZATION



Copyright © 2018 CA. All rights reserved.



Web Services Installation

- Restart your CV after completing the HOLDDATA
- Run the Web Services Demo application to test your installation:
 - “ADS IDMSWSDM”



Copyright © 2018 CA. All rights reserved.



Troubleshooting

CA IDMS Web Services Troubleshooting

- Enable system level logging of Web Services programs using task WSQP
- CA Web Services API offers additional troubleshooting techniques
- The first 4 fields in every API call are WS-FUNCTION-CODE, WS-RETURN-CODE, WS-ERROR-INFO and WS-REQUEST-HANDLE

CA IDMS Web Services Troubleshooting

- At a high level, WS-RETURN-CODE gives a quick indication of the status of the API call

Return Code	Severity	Description
0	Successful	Successful return
4	Warning	Request processed, warning msg issued
8	Error	Request fails, error message returned
12	Critical	Request fails, Service terminated
16	Systemic	Request fails, impact to all Services



Copyright © 2018 CA. All rights reserved.



CA IDMS Web Services Troubleshooting

- WS-ERROR-INFO provides additional fields that define the result of the call

Error Type:

INTERNAL (I)	Generated from failures in CA IDMS/DC operations
API (A)	Failure to adhere to Web Services API protocol
XML (X)	Generated if XML Parsing or Generation fails
HTTP (H)	API receives an unexpected HTTP status code
TCPIP (T)	An unexpected TCPIP code received
SOAP (S)	An unexpected SOAP fault code received
OTHER (O)	An unclassified error occurred

Error Text: Text that describes additional content to the error



Copyright © 2018 CA. All rights reserved.



CA IDMS Web Services Troubleshooting

- Enable Logging at System Level using Task Code WSQP

In your CV to monitor, enter:

WSQP LOG WEB SERVICES=YES (Turns logging on for all Web Services Programs)

or

WSQP LOG WEB SERVICES=NO (Turns logging off)

WSQP LOG WEB SERVICES=USERPRGM (Turns logging on for a single program)

or

WSQP LOG WEB SERVICES= (Clears single program logging off)

Sample Web Services Log:

```
15:42 IDMS DC504900 V130 T242 IDMSWSSS --- DATABASE ERROR STATUS 3020, DURING:
15:42 IDMS DC504902 V130 T242 IDMSWSSS SOAP LOGIC WSEND
15:42 IDMS DC504600 V130 T242 IDMSWSI CALLING SEND
15:42 IDMS DC504800 V130 T242 IDMSWSSP START PROGRAM
15:42 IDMS DC504800 V130 T242 IDMSWSSP MESSAGE DATA SENT:
15:42 IDMS DC504800 V130 T242 IDMSWSSP Service requested invalid or unavailable
```



Copyright © 2018 CA. All rights reserved.



CA IDMS Web Services Troubleshooting

- Enable Logging Dynamically using API function SETOPTION

MOVE 8 TO WS-FUNCTION-CODE. ! WSSETOPTION()

MOVE 1 TO WS-OPTION-NUMBER.

MOVE 'YES' TO WS-OPTION-VALUE.

LINK TO PROGRAM 'IDMSWSI' USING

(WSDemo-FUNCTION,

WSDemo-RETURN,

WS-ERROR-INFO,

WSDemo-SESSION,

WS-OPTION-NUMBER-RECORD,

WS-OPTION-VALUE-RECORD).

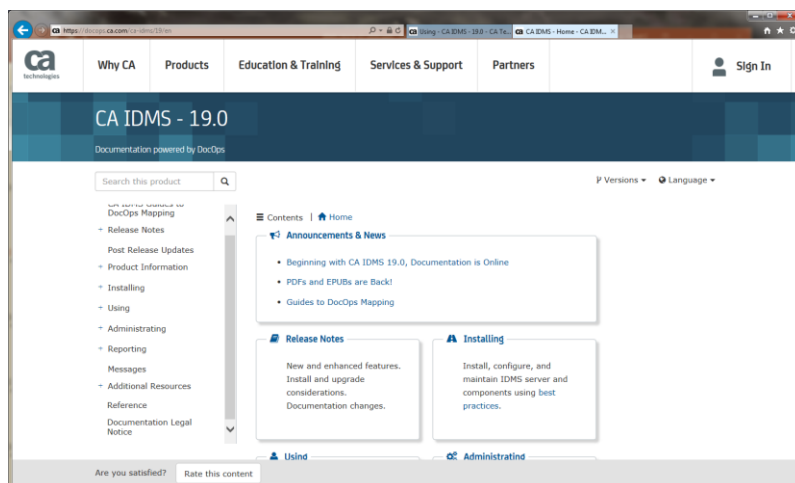


Copyright © 2018 CA. All rights reserved.

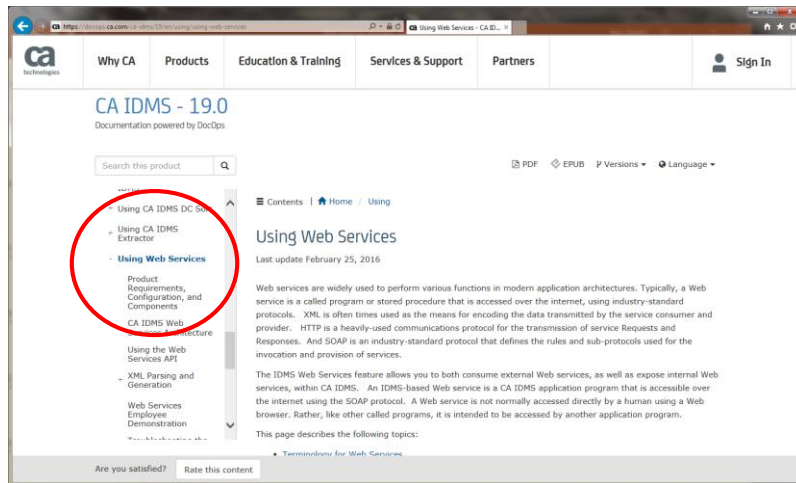


Web Services Documentation

<https://docops.ca.com/idms>



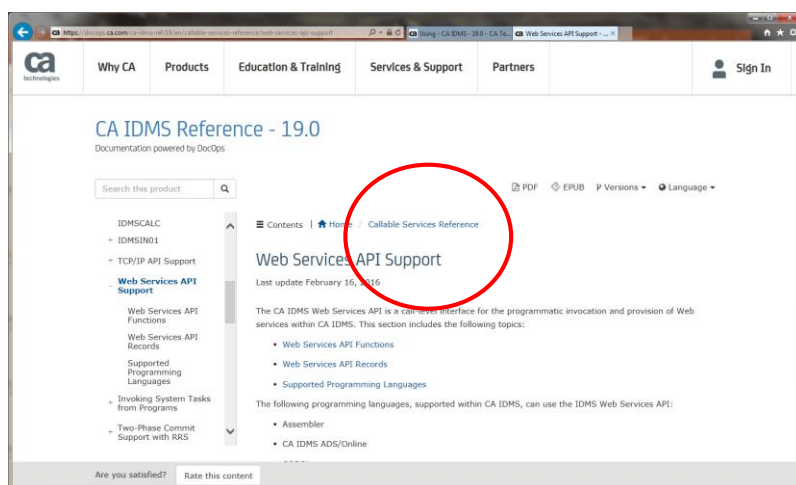
Using Web Services



Copyright © 2018 CA. All rights reserved.



Web Services API Reference



Copyright © 2018 CA. All rights reserved.



Summary

- The Web Services feature provides another means by which clients can modernize their CA IDMS-based applications, as well as provide modern-day applications easy access to CA IDMS-based data using industry-standard technology.



Copyright © 2018 CA. All rights reserved.



FOR INFORMATION PURPOSES ONLY Terms of this Presentation

This presentation was based on current information and resource allocations as of May 2018 and is subject to change or withdrawal by CA at any time without notice. Notwithstanding anything in this presentation to the contrary, this presentation shall not serve to (i) affect the rights and/or obligations of CA or its licensees under any existing or future written license agreement or services agreement relating to any CA software product; or (ii) amend any product documentation or specifications for any CA software product. The development, release and timing of any features or functionality described

in this presentation remain at CA's sole discretion. Notwithstanding anything in this presentation to the contrary, upon the general availability of any future CA product release referenced in this presentation, CA will make such release available (i)

for sale to new licensees of such product; and (ii) to existing licensees of such product on a when and if-available basis as part of CA maintenance and support, and in the form of a regularly scheduled major product release. Such releases may be made available to current licensees of such product who are current subscribers to CA maintenance and support on a when and

if-available basis. In the event of a conflict between the terms of this paragraph and any other information contained in this presentation, the terms of this paragraph shall govern.

Certain information in this presentation may outline CA's general product direction. All information in this presentation is for your informational purposes only and may not be incorporated into any contract. CA assumes no responsibility for the accuracy or completeness of the information. To the extent permitted by applicable law, CA provides this presentation "as is" without warranty of any kind, including without limitation, any implied warranties or merchantability, fitness for a particular purpose, or non-infringement. In no event will CA be liable for any loss or damage, direct or indirect, from the use of this document, including, without limitation, lost profits, lost investment, business interruption, goodwill, or lost data, even if CA is expressly advised in advance of the possibility of such damages. CA confidential and proprietary. No unauthorized copying or distribution permitted.



Copyright © 2018 CA. All rights reserved.



Questions & Answers



Copyright © 2018 CA. All rights reserved.



Please Complete a Session Evaluation Form

- The number for this session is **A05**
- After completing your session evaluation form, place it in the envelope at the front of the room

The image shows a 'Session Evaluation Form' template. At the top, it features the IUA and CA logos, and the title 'IUA / CA EMS Technical Conference Session Evaluation Form'. Below the title, there are fields for 'Session Number:' and 'Name (Optional):'. A section for 'Rate this overall session' includes a table with columns for 'Poor', 'Good', and 'Excellent', each with a rating scale from 1 to 5. The main body of the form consists of several rows, each with a statement followed by a table of rating options: 'Strongly Dislike', 'Dislike', 'Neutral', 'Like', and 'Strongly Like'. The statements are: 'The speaker was prepared and knowledgeable of the subject covered', 'The session met my expectations', 'The material is suitable for my company', 'I would recommend this session to a colleague', 'The session length was appropriate for the content', and 'This session would be useful as a reference'. Each row has a 'Comments' field. At the bottom, there is a 'General Comments' section with a large text area for additional feedback.



Copyright © 2018 CA. All rights reserved.

