Gregor Wolf

September 7, 2015 6 minute read

ABAP Push / Messaging Channel and SAPUI5 Demo Application

Based on the Blog series published by Masoud Aghadavoodi Jolfaei:

ABAP Channels Part 1: WebSocket Communication Using ABAP Push Channels

ABAP Channels Part 2: Publish/Subscribe Messaging Using ABAP Messaging Channels

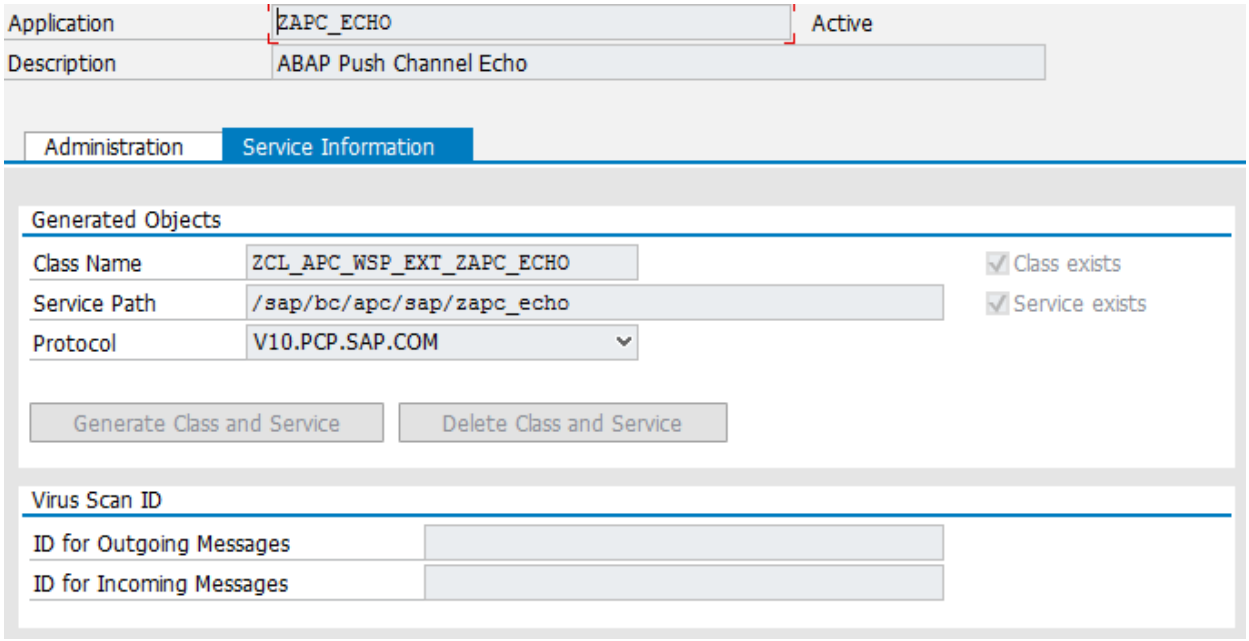
ABAP Channels Part 3: Collaboration Scenario Using ABAP Messaging and ABAP Push Channels

Specification of the Push Channel Protocol (PCP)

I've created an SAPUI5 Demo application using the ABAP Push Channel (APC, WebSockets) and ABAP Messaging Channel (AMC). To benefit from the WebSocket technology in your browser please check the support matrix at https://en.wikipedia.org/wiki/WebSocket#Browser_implementation. At the ABAP Backend I've used a NetWeaver ABAP Application Server 7.40 SP8.

ABAP Push Channel

Let's define the ABAP Push Channel first. Check out the above mentioned Blog's for details:



Here's the source code of the class Implementation:

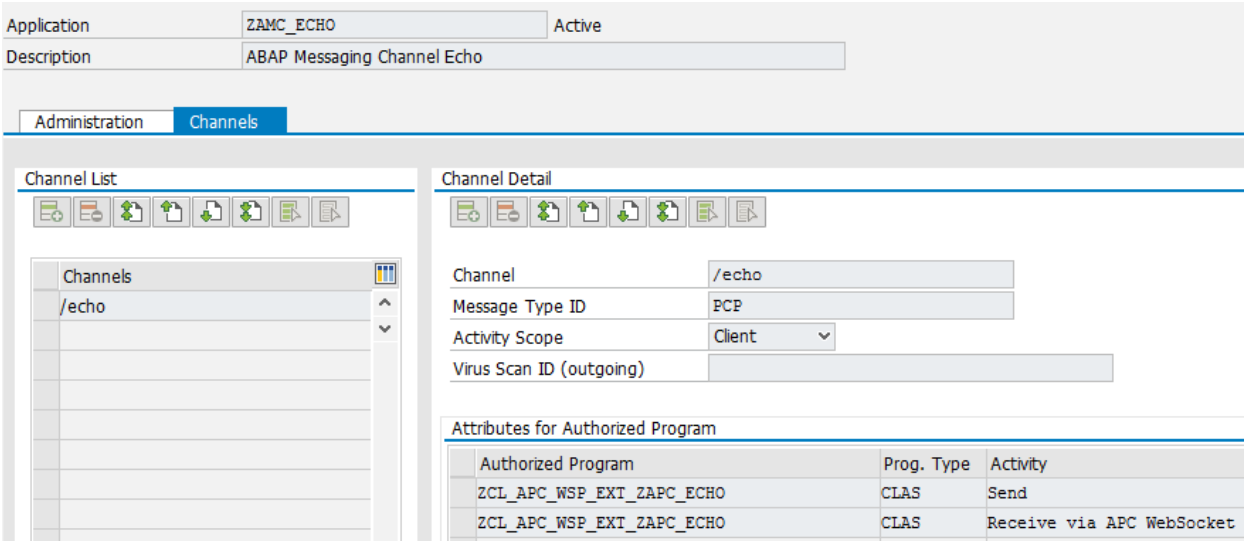
```
CLASS zcl_apc_wsp_ext_zapc_echo DEFINITION
  PUBLIC
  INHERITING FROM cl_apc_wsp_ext_stateless_pcp_b
  FINAL
  CREATE PUBLIC .
  PUBLIC SECTION.
    METHODS if_apc_wsp_ext_pcp~on_message
         REDEFINITION .
    METHODS if_apc_wsp_ext_pcp~on_start
         REDEFINITION .
  PROTECTED SECTION.
  PRIVATE SECTION.
    CLASS-METHODS prepare_message_for_ui
      IMPORTING
        !iv_text       TYPE string
      RETURNING
        VALUE(rv_text) TYPE string .
ENDCLASS.
CLASS zcl_apc_wsp_ext_zapc_echo IMPLEMENTATION.
* <SIGNATURE>---------------------------------------------------------------------------
* | Instance Public Method ZCL_APC_WSP_EXT_ZAPC_ECHO->IF_APC_WSP_EXT_PCP~ON_MESSAGE
* +--------------------------------------------------------------------------------------
* | [--->] I_MESSAGE                      TYPE REF TO IF_AC_MESSAGE_TYPE_PCP
* | [--->] I_MESSAGE_MANAGER              TYPE REF TO IF_APC_WSP_MESSAGE_MANAGER_PCP
* | [--->] I_CONTEXT                      TYPE REF TO IF_APC_WSP_SERVER_CONTEXT
* +--------------------------------------------------------------------------------------
  METHOD if_apc_wsp_ext_pcp~on_message.
```

```
    DATA: lo_producer TYPE REF TO cl_amc_message_type_pcp.
    TRY.
* retrieve the text message
        DATA(lv_text) = i_message->get_text( ).
        lo_producer ?= cl_amc_channel_manager=>create_message_producer(
          i_application_id = 'ZAMC_ECHO'
          i_channel_id     = '/echo' ).
        lv_text = prepare_message_for_ui( lv_text ).
        DATA(lo_msg) = cl_ac_message_type_pcp=>create( ).
        lo_msg->set_text( i_message = lv_text ).
        lo_producer->send( i_message = lo_msg ).
      CATCH cx_ac_message_type_pcp_error INTO DATA(lx_pcp_error).
        MESSAGE lx_pcp_error->get_text( ) TYPE 'E'.
      CATCH cx_amc_error INTO DATA(lx_amc_error).
        MESSAGE lx_amc_error->get_text( ) TYPE 'E'.
      CATCH cx_apc_error INTO DATA(lx_apc_error).
        MESSAGE lx_apc_error->get_text( ) TYPE 'E'.
    ENDTRY.
  ENDMETHOD.
* <SIGNATURE>---------------------------------------------------------------------------------
* | Instance Public Method ZCL_APC_WSP_EXT_ZAPC_ECHO->IF_APC_WSP_EXT_PCP~ON_START
* +--------------------------------------------------------------------------------------------
* | [--->] I_CONTEXT                      TYPE REF TO IF_APC_WSP_SERVER_CONTEXT
* | [--->] I_MESSAGE_MANAGER              TYPE REF TO IF_APC_WSP_MESSAGE_MANAGER_PCP
* +--------------------------------------------------------------------------------------------
  METHOD if_apc_wsp_ext_pcp~on_start.
    TRY.
* bind the WebSocket connection to the AMC channel
        DATA(lo_binding) = i_context->get_binding_manager( ).
        lo_binding->bind_amc_message_consumer(
          i_application_id = 'ZAMC_ECHO'
          i_channel_id     = '/echo' ).
      CATCH cx_apc_error INTO DATA(lx_apc_error).
        DATA(lv_message) = lx_apc_error->get_text( ).
        MESSAGE lx_apc_error->get_text( ) TYPE 'E'.
    ENDTRY.
  ENDMETHOD.
* <SIGNATURE>---------------------------------------------------------------------------------
* | Static Private Method ZCL_APC_WSP_EXT_ZAPC_ECHO=>PREPARE_MESSAGE_FOR_UI
* +--------------------------------------------------------------------------------------------
* | [--->] IV_TEXT                        TYPE        STRING
* | [<-()] RV_TEXT                        TYPE        STRING
* +--------------------------------------------------------------------------------------------
  METHOD prepare_message_for_ui.
    TYPES: BEGIN OF t_message,
             text TYPE string,
             user TYPE uname,
             date TYPE timestamp,
           END OF t_message.
```

```
    DATA: ls_message TYPE t_message.
    ls_message-text = iv_text.
    ls_message-user = sy-uname.
    GET TIME STAMP FIELD ls_message-date.
    DATA(lo_json_witer) = cl_sxml_string_writer=>create(
                            type = if_sxml=>co_xt_json
                          ).
    CALL TRANSFORMATION id SOURCE ls_message = ls_message
                        RESULT XML lo_json_witer.
    DATA(lv_xstr) = lo_json_witer->get_output( ).
    CALL FUNCTION 'ECATT_CONV_XSTRING_TO_STRING'
      EXPORTING
        im_xstring = lv_xstr
*       im_encoding = 'UTF-8'
      IMPORTING
        ex_string  = rv_text.
  ENDMETHOD.
ENDCLASS.
```

ABAP Messaging Channel

For the ABAP Messaging Channel it is important to define the Authorized Programs:



HCP Git to GitHub

This is just to document how to bring an app developed in SAP Web IDE from the HCP Trial Git repository to GitHub. I've used the following steps adopted from the documentation:

Adding an existing project to GitHub using the command line – User Documentation

First of all clone the HCP Git repository to the local system and switch to the new folder:

```
git clone https://<HCP Username>@git.hanatrial.ondemand.com/<HCP Accountname>/<HCP GIT Re
cd apcecho/
```

Now create a new repository on GitHub and add this repository. I've named it "github":

```
git remote add github https://github.com/gregorwolf/apcecho.git
```

As I've created some files (Readme, License) in the GitHub Repository this files have to be pulled from the repository:

```
git pull github
```

And merged with the master:

```
git merge github/master
```

Then push the changes to both repositories:

```
git push github master
git push origin master
```

SAPUI5 Frontend

I've used the SAPUI5 Application Template in SAP Web IDE to bootstrap my app. You can get the source at:

gregorwolf/apcecho

You can use the Project SAPUI5-Deployer by Graham Robinson to import this Git repository directly into your ABAP stack.

That's for the moment. Looking forward for your feedback.

Like Alert Moderator

Assigned tags

ABAP Connectivity | abap | abap connectivity | apc | openui5 |

View more...

Related Blogs

Real-time notifications and workflow using ABAP Push Channels (websockets) Part 1: Creating the APC and AMC
By **Brad Pokroy** , Oct 16, 2014

Introduction to ABAP Channels
By **Olga Dolinskaja** , Nov 27, 2014

ABAP Channels Part 2: Publish/Subscribe Messaging Using ABAP Messaging Channels
By **Masoud Aghadavoodi Jolfaei** , Mar 26, 2014

Related Questions

ABAP Push Channel is not supported in the selected project
By **Former Member** , Nov 03, 2016

ABAP Push Channel ...How to deal with data?
By **Amol Samte** , Oct 21, 2016

ABAP Push Channel... Send data to UI5 application once data changes..
By **Former Member** , Nov 10, 2016

10 Comments

 Graham Robinson

September 21, 2015 at 6:23 am

Great example Gregor – now I just wish all my customers were on NW ABAP 7.4 SP8. 🙁

Like (0)

 Gregor Wolf | Post author

September 21, 2015 at 7:01 am

Perhaps they can use their Gateway / Fiori system 😉

Like (0)

 Graham Robinson

September 21, 2015 at 7:05 am

Or even their SolMan system 😈

Like (0)

 Hemendra Sabharwal

November 16, 2015 at 12:24 pm

Thanks Greg, It's a really good example to follow,

I will try to replicate the same here in our sandbox to get the better end-to-end understanding on the ABAP Push.

Thanks,

Warm Regards

Hemendra

Like (0)

 Meinrad Funke

April 18, 2016 at 4:43 am

Hi Gregor,

first of all I have to thank you for that blog. We are working in a project where we have appr. 10 apps for workers in a yard. And we want to be able to send them messages while they are working on their tasks.

Your blog helped me to build a similar demo application. And immediately after finishing it and considering to use it in the project – two questions arise

– We want to send messages from our backend to Fiori apps. And we do not want to create APC to the backend. How to integrate the (Fiori) Gateway? I.e. how exchange messages across several systems?

– What about the lifecycle of the web socket? Will it be closed automatically if the user exits the app and returns to the Launchpad? What happens in case of navigating to another app?
I did not find information on that. Maybe you have a link or something?

Best regards

Meinrad

Like (0)

Gregor Wolf | Post author

May 18, 2016 at 5:35 am

Dear Meinrad,

sorry for the late reply. Regarding your question:

> how exchange messages across several systems?

If you don't have a NetWeaver ABAP 7.50 infrastructure you have to fall back to use RFC calls between the systems. In 7.50 ABAP Push Channels also support acting as a client.

> What about the lifecycle of the web socket? Will it be closed automatically if the user exits the app and returns to the Launchpad? What happens in case of navigating to another app?

Without having done any real world testing I would think that the websocket is closed. Perhaps Masoud Aghadavoodi Jolfaei has some experience on that.

Best regards

Gregor

Like (0)

Masoud Aghadavoodi Jolfaei

May 18, 2016 at 7:34 am

Hi Meinrad,

for a "system2system" communication (a kind of ABAP Channel bridge between systems) based on APC I recommend to check the "APC Detached Client" pattern. With this you are able to spawn WebSockets between systems their life cycle does not bound to the client report. A sample code is provided in "Developing an APC Detached Client" section. I think the question regarding the lifecycle of WebSockets

used in apps running inside Fiori Lunchpad (FLP) has to be answered by FLP colleagues.I have forwarded this question to the colleagues.

Cheers,

Masoud

Like (0)

 Meinrad Funke

May 26, 2016 at 5:46 am

Hi Masoud,

thanks for this answer. Unfortunately we are still on NW7.40. But it looks interesting and I think I'll have to keep it in mind for next project.

Regards

Meinrad

Like (0)

 Michael Dohse

June 15, 2017 at 4:22 pm

Hi Gregor,

is this scenario based on a embedded gateway hub?

If I am right, this will actually not work in a central gw-hub scenario, correct?

Thanks for short clarification...

michael

Like (0)

 Gregor Wolf | Post author

June 15, 2017 at 5:35 pm

Hi Michael,

you could use the SAP Gateway Push to bring the message to the central hub and register an ICF handler where you implement the translation to the ABAP Push Channel.

Best regards
Gregor

Like (0)