SAP
Records Management

Visualization of Records Management in BSP

Developer Documentation

May 11, 2004

Contents

1 Introduction

To make the best use of this document, you require a sound working knowledge of the following areas:

- Technical terminology and architecture of the Records Management Framework. For more information, see the Records Management reference documentation for developers.
- Business Server Pages and the Portal

This document is split into four sections, which are aimed partly at different target groups:

- **Information about developing service providers**
  This information is aimed mainly at developers who want to implement a BSP visualization for their own service providers.
- **Information about using the BSP Framework**
  This information is aimed m ainly at consultants and developers who want to use the BSP service providers in their own scenarios (BSP applications or the Portal, for example).

- **Information about the delivered service providers**
  This information is aimed mainly at consultants and deve lopers who want to use the BSP service providers in their own scenarios (BSP applications or the Portal, for example).

- **Information about using the BSP service provider in the Portal**
  This information is aimed at consultants and developers who want to use Records Management in a Portal scenario.

This document does not discuss the fundamentals of Business Server pages, the Portal, or Records Management; for more information about these topics, see the appropriate Online Documentation.

2 Developing Service Providers for BSP

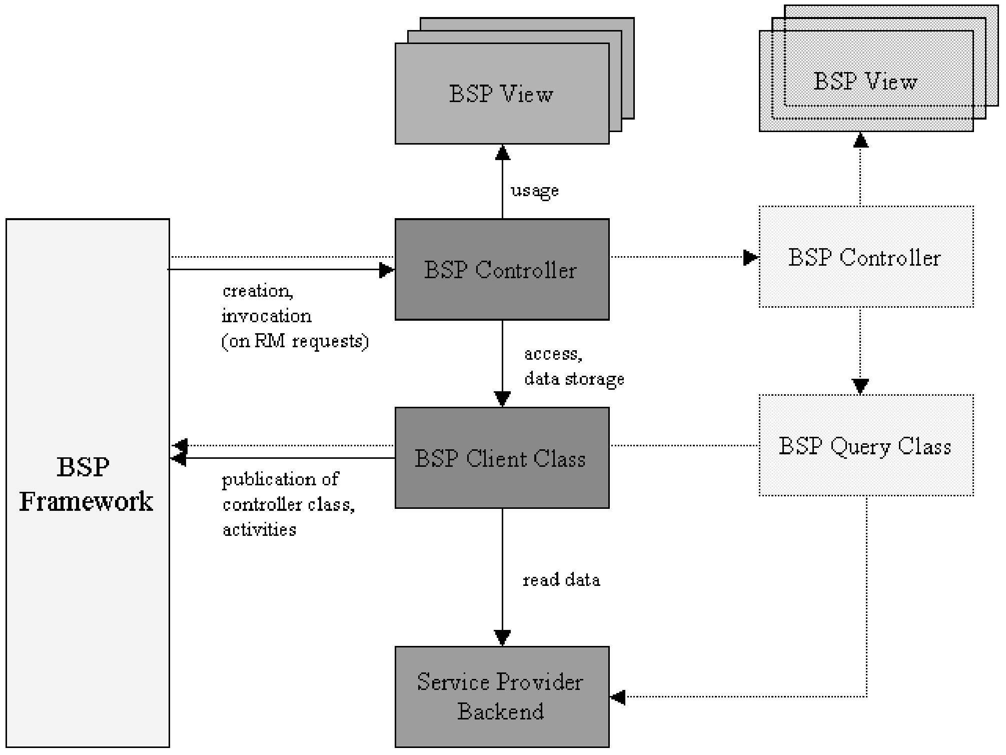
2.1 Programming Model

The programming model for BSP service providers follows the model -view controller (MVC) pattern; the prerequisite for developing BSP service providers is a fundamental understanding of this programming model and the other properties of Business Server Pages (for more information, see the BSP tutorial in the Online Documentation).

A BSP service provider consists of four modules:

- BSP client class
- BSP query class
- BSP controller class (with BSP views)
- Service provider back end

The interaction between the modules is depicted in the following figure:



2.1.1 BSP Client Class

The BSP client class is registered in the framework. It is used to format data from the back end, and to save values from a session lo   cally.

The class must implement the IS_SP_VISUALIZATION_BSP_CLASS class role; this role is registered in the RM Registry.

The name of the implemented BSP controller is also published in the client class, as well as the available activities and the authorization check.

2.1.2 BSP Query Class

Just like the BSP client class, the BSP query class implements the IS_SP_VISUAL_QUERY_BSP_CLASS class role; this role is required for the search activity. Its other properties match the BSP client class.

2.1.3 BSP Controller Class

The BSP controller class is created in the BSP development environment. It gets its data in accordance with the BSP specifications from CL_BSP_??? and implements the IF_SRM_SP_BSP_CONTROLLER interface. Its task is to handle RM requests (activities) and pr ocess user events. The controller class uses the regular BSP methods to call one or more views for the display.

The BSP controller class is not registered in the RM Registry. This class is generated again each time an http request is sent, which means tha t it is not suited to saving data between two http requests.

2.1.4 BSP View

The BSP views are used to display a BSP service provider. Any HTML elements can be used, as well as elements of the HTMLB and XHTMLB BSP extension.

Important note: Since the `<srm:element>` *tag is used to integrate the BSP views of the service providers in a page, you are* ***not*** *allowed to use the following HTMLB tags or HTML elements in BSP views for service providers:*

- `<htmlb:content>`
- `<htmlb:page>`
- `<htmlb:document>`
- `<htmlb:documentBody>`
- `<htmlb:documentHead>`
- `<HTML>`; `</HTML>`
- `<HEAD>`; `</HEAD>`
- `<BODY>`; `</BODY>`

2.1.5 Service Provider Back End

For persistent data storage, the back end is used in the IS_SP_CONTENT_CONNECTION_CLASS class role – as in a service provider for SAPGUI.

2.2 Important Interfaces

2.2.1 BSP Client Class

2.2.1.1IF_SRM_SP_CLIENT_BSP

In the GET_BSP_APPLICATION method, you publish the name of the BSP application and the controller.

Example:

```
method IF_SRM_SP_CLIENT_BSP~GET_BSP_APPLICATION .
  bsp_application-appname = 'SRM_NOTE'.
  bsp_application-controllername = 'NOTE.DO'.
endmethod.
```

2.2.1.2IF_SRM_SP_ACTIVITIES

See the documentation about developing service providers.

2.2.1.3IF_SRM_SP_AUTHORIZATION

See the documentation about developing service providers.

2.2.2 BSP Query Class

2.2.2.1IF_SRM_SP_QUERY_BSP

See the information about the IF_SRM_SP_CLIENT_BSP interface.

2.2.3 BSP Controller Class

2.2.3.1 IF_SRM_SP_BSP_CONTROLLER

2.2.3.1.1 Method ON_DESTROY

This method is called when the service provider is unloaded from the framework. Here, any locks must be released. This avoids “dead” lock entries.

Example:

```
method IF_SRM_SP_BSP_CONTROLLER~ON_DESTROY.
  data: my_client type ref to zcl_my_client_class.

  my_client ?= me->if_srm_sp_bsp_controller~sp_client.

  my_client->release_all_locks( ).

endmethod.
```

2.2.3.1.2 Method ON_NEW_REQUEST

This method is called when a new RM request is waiting to be processed. Here, the client must be instructed to load the data from the back end and format it, if necessary. The ACTIVITY_STATE of the RM request object must be set as follows:

- IF_SRM_REQUEST=>ACTIVITY_FINISHED_WITH_OK
  Set this value if the activity has been completed and you no longer want to display the service provider.
- IF_SRM_REQUEST=>ACTIVITY_ONGOING
  Set this value if the activity has been started and you then want to display the service provider (this is the regular case).
- IF_SRM_REQUEST=>ACTIVITY_ERROR or NO_AUTHORIZATION
  Set this value when errors occur.

Important: The view for the visualization is not actually called until the – redefined – `do_request` *method of the controller is performed.*

Example:

```
method IF_SRM_SP_BSP_CONTROLLER~ON_NEW_REQUEST.
  data: my_client type ref to zcl_my_client_class.

  my_client ?= me->if_srm_sp_bsp_controller~sp_client.

  case im_request->get_activity( ).
       when IF_SRM_ACTIVITY_LIST=>DISPLAY.
              my_client->open( im_for_update = ‘ ‘ ).
       when IF_SRM_ACTIVITY_LIST=>MODIFY.
              my_client->open( im_for_update = ‘X‘ ).
  endcase.

  im_request->set_activity_state( if_srm_request=>activity_ongoing ).

endmethod.
```

2.2.3.1.3 Method ON_USER_EVENT

This method is called when a user interaction (such as a HTMLB element event or button click) has taken place. Here, the user interaction must be evaluated and processed. If you no longer want to display the service provider (*Exit* or *Close* function), then the ACTIVITY_STATE for the RM request object must be set to IF_SRM_REQUEST=>ACTIVITY_FINISHED_WITH_OK. The asynchronous response must then be sent using IF_SRM_BSP_CLIENT_EVENT~ SEND_ASYNC_ANSWER.

Example:
```
method IF_SRM_SP_BSP_CONTROLLER~ON_NEW_REQUEST.
  data: my_client type ref to zcl_my_client_class.

  my_client ?= me->if_srm_sp_bsp_controller~sp_client.

  event = cl_htmlb_manager=>get_event( runtime->server->request ).

  CASE event->server_event.

       WHEN 'saveClick'.
              my_client->save( ).

       WHEN 'exitClick'.
              my_client->discard( ).
              my_client->release_all_locks( ).

  ENDCASE.

  im_request->set_activity_state( if_srm_request=>activity_ongoing ).

  me->if_srm_sp_bsp_controller~sp_client_event->send_async_answer(
                                                     im_request ).

endmethod.
```

2.2.3.1.4 Method ON_REQUEST_ANSWER

This method is called to provide responses to an RM request sent by the SP itself (to display a subobject, for example).

2.2.3.1.5 Method ON_REFRESH

This method is not currently used by the framework; only an empty message body needs to be created.

2.2.3.2IF_SRM_BSP_CLIENT_EVENT

A reference to IF_SRM_BSP_CLIENT_EVENT exists as the SP_CLIENT_EVENT attribute for IF_SRM_SP_BSP_CONTROLLER; this attribute is filled by the framework and can be read and used by the SP.

2.2.3.2.1 Method SEND_REQUEST

This method is called by the SP to send its own RM requests.

2.2.3.2.2 Method SEND_ASYNC_ANSWER

The SP calls this method to send an asynchronous response (to end the request, for example).

3 Using the BSP Framework

3.1 The <srm:element> Tag

You can use the `<srm:element>` tag to integrate the visualizations of SPs in separate BSP applications
Note: To get more information about how to use the `<srm:element>` *tag, also read the F1 help for the tag.*

3.2 Example Programs

You can find an example of how to use the `<srm:element>` tag in the SRM_DEMO_RECORD BSP application.

3.3 Attributes

3.3.1 ID

To enable the central state to be assigned correctly, you must give the element a unique ID. This ID must be unique within the internal mode, which means that an ID must be generated for each day and user login.

3.3.2 EMPTY_CONTENT_TEXT

Here, you can specify a text that is displayed when no element is loaded in the tag.

3.3.3 EMBEDDED

If the EMBEDDED switch is set to IF_SRM=>TRUE, then the embedding BSP application is responsible for rendering the HTMLB tags `<htmlb:content>` and `<htmlb:page>`. This is useful if the tag is being implemented on a BSP page on which other forms or elements are located, as well as Records Management elements.

3.3.4 EVENT_CALLBACK

Here, you can specify an instance of a class whose methods are called by the RM Framework when the displayed element itself triggers an RM request (to select a record in a document, for example). This class must implement the IF_SRM_BSP_CALLBACK interface.

4 Delivered Standard Service Providers

4.1 Organizer

The delivered BSP service provider *Organizer* enables you to display a view created with transaction SRMVIEWGEN for BSP. Depending on the configuration, the user can use this view to access search functions or to access certain objects directly.
The BSP Organizer determines which roles are assigned to the user and finds the assigned Organizer views within these roles.

Note: To use the BSP Organizer, the user must be assigned ***precisely*** *one BSP view with the assigned role (PFCG); this means that multiple BSP views are not allowed and the standard SAPGUI view does not exist.*

4.2 Record

The delivered BSP service provider for records enables you to display and change records and to insert and delete elements in the record.

4.3 Documents

The delivered BSP service provider for documents enables you to display and upload documents.

4.4 Archive Documents

The delivered BSP service provider for documents enables you to display and upload ArchiveLink documents.

4.5 Notes

The delivered BSP service provider for notes enables you to display notes.

4.6 URL

The delivered BSP service provider for URLs enables you to display URLs.

4.7 Search Dialog for Records, Documents, and Notes

The standard search dialog enables you to find records, documents, and notes. The standard dialog can be modified to a customer's requirements by using an extension.

4.8 Implementing an Extension Class

An extension class must implement the IF_SRM_GENSP_QUERY_EXT interface.

4.8.1 Method GET_ICON

This method gets the icon that is displayed in the search list. The standard implementation gets the icon that is registered for the appropriate service.

4.8.2 Method GET_PARA_BEE

You can use this method to modify the way the search screen is displayed A reference to a BSP extension expression (BEE) must be returned (reference to IF_BSP_BEE).

4.8.3 Method GET_PARA_TAB

If a user -defined search screen i s implemented, then this method must be used to convert the values from the input fields of this screen. You can decode the values of the input fields from the HTTP request object (IM_REQUEST); the search parameter tables EX_PROP_QUERY and EX_CONTENT_QUERY need to be given values here. You can use EX_PROP_LIST to control which attributes are specified for the entries that are found.

Note: For more detailed information about the structure of the parameter tables, see the developer documentation about the Generic SP.

4.8.4 Method GET_RESULT_FIELDCAT

This method gets the field catalog that is needed to display the search results in the HTMLB table control. The RENDER_CELL method fills the fields of the field catalog.

4.8.5 Method GET_TITLE

This method gets the title of the search dialog.

4.8.6 Method RENDER_CELL

This method displays the individual fields of the search results. For each field, it calls each row of the set of search results once. You can use the importing parameter IM_COLUMN_ID to determine the displayed column; y ou can use the importing parameters IM_DOCUMENT_DATA and IM_DOCUMENT_PROP to determine the displayed row or its attributes. You must use the returning parameter RE_REPLACEMENT_BEE to return a reference to a BSP extension expression ().

Note: For more information about BSP extension expressions and how to use them in the HTMLB table control, see the Online Documentation; use the key word “Tableview Iterator”.

4.9 Using an Extension Class

You assign the extension class to an element type (not a service provider) in transaction SRMREGEDIT. To do this, open the element type in change mode and specify the name of the extension class as the connection parameter *GENSP_QUERY_EXT*.
If no query extension is defined for an element type, the standard class CL_SRM_DEFAULT_QUERY_EXT is used.

5 Using the BSP Service Provider in the Portal

To use the BSP service provider in the Portal, you must implement a BSP application that integrates the `<srm:element>` tag. You can use the administration function of the Portal to integrate this BSP application as an iView. If required, you can use the `<srm:element>` tag with Javascript code to send and receive Portal events; in this case, you must integrate the Javascript code on the BSP page that also includes the `<srm:element>` tag. Using the appropriate BSP functions, you can process incoming Portal events and forward them to the tag; if you want to convert events from the tag or from Records Management into Portal events, you can use the EVENT_CALLBACK function of the `<srm:element>` tag.

For an example implementation of a Portal service, see the page `iview.htm` of the SRM_DEMO_RECORD BSP application. This page enables you to search for records or documents and then display them; you can use this page directly, or as a template for your own implem entations. If you use the page directly, then you can use the parameters RMS_ID and SPS_ID to select the element type and RMS used by the search function.

Note for Portal users: If you want to use more than one instance of the `<srm:element>` *tag on a Porta l page, you must make sure that the parameter ID of each tag is unique, by using a GUID, for example. Otherwise, the RM BSP Framework cannot make a unique assignment between the tag and the service providers.*

For more information about using BSP pages in the Portal, see the Portal documentation and the documentation about Business Server Pages.