



FEU INSTITUTE OF TECHNOLOGY
COLLEGE OF ENGINEERING • COLLEGE OF COMPUTER STUDIES

ITSE333A

ABAP

Lesson 8: Business Server Pages

Anthony D. Aquino



Agenda

1. Maintenance and execution of BSPs
2. Usage of BAPIs in BSPs
3. HTMLB
4. Model View Controller





Business Server Pages (BSP)

- Web extension for ABAP
- Enables the use of ABAP and server-side JavaScript within HTML pages created and hosted on a SAP System
- Can be used for the realization of extensive portal solutions
- Availability of BSPs starting with SAP Web Application Server 6.20



Maintenance and execution of BSP's

- Maintenance by transaction SICF
- Execution: `http://<host>:<ABAP Port>/sap/bc/bsp/sap/<Program name>\<Page name>.htm`

E.g.: `http://g51as1.informatik.tu-muenchen.de:`

`8051/sap/bc/bsp/sap/<Programname>\<Page name>.htm`

Maintain service

Create Host/Service External Aliases System Monitor Inactive

Filter Details

Virtual Host Service Path

Service

Description

Lang. Ref.Service:

Filter Reset Detail

Virtuelle Hosts / Services	Documentation	Referenz Service
default_host	VIRTUAL DEFAULT HOST	
sap	SAP NAMESPACE; SAP IS OBLIGED NOT T...	
option	RESERVED SERVICES AVAILABLE GLOBA...	
public	PUBLIC SERVICES	
ap	Application Platform	
bc	BASIS TREE (BASIS FUNCTIONS)	
bsp	BUSINESS SERVER PAGES (BSP) RUNTI...	
sap	NAMESPACE SAP	

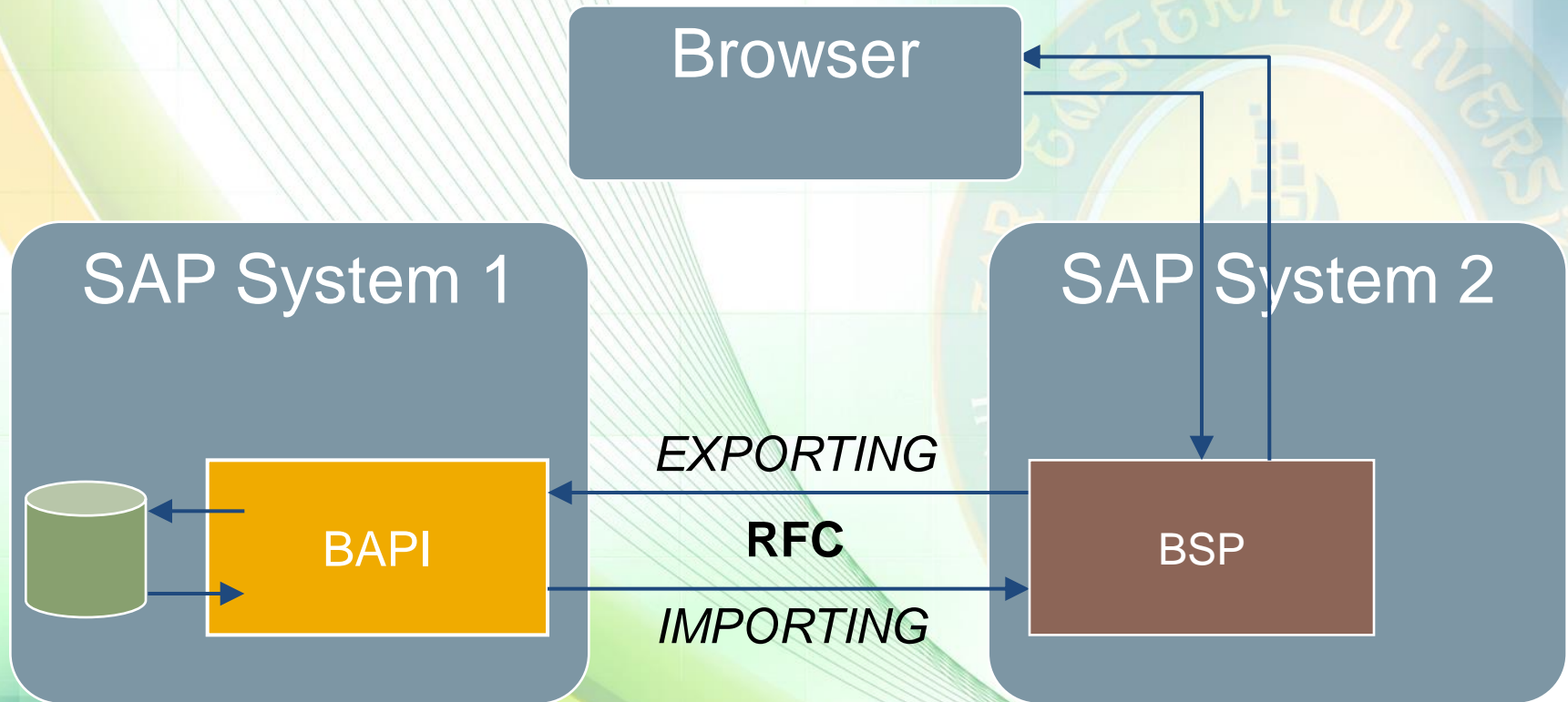


Usage of BAPIs in BSPs I

- Business Application Programming Interface = BAPI
 - BAPI represent an interface to the outside world
 - SAP systems offer a wide range of different BAPIs
 - BAPI calls are executed using RFC
 - Transaction „BAPI“ shows all available BAPIs of the current SAP system
 - Search for BAPIs using BAPI Browser, cross-system access possible
 - Use of BAPIs with help of information available in the BAPI Browser



Usage of BAPIs in BSPs II

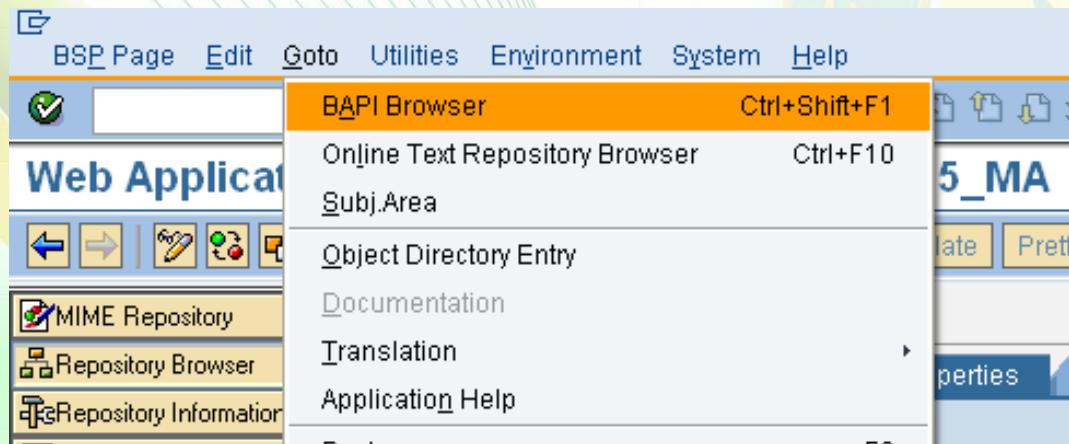


- This example: Call of an external BAPI
- Other possible examples: Call of an internal BAPI



Using BAPI Browser I

- Call BAPI Browser in transaction SE80
 - Menu item ,Goto' → ,BAPI Browser'





Using BAPI Browser II

- BAPI Browser components:

The screenshot displays the SAP BAPI Browser interface. On the left, a tree view titled 'Systems and BAPIs' lists various systems and BAPIs. The BAPI 'BAPI_ACCASSIGNMENT_LOADDATA' is selected, and its details are shown in the main pane. The details include the BAPI's interface definition and a call example.

Systems and BAPIs

- F30_400
- F31
- F12_400
- FINBASIS_800
- FINBTR@D3CLNT000
- FINBTR@D3CLNT800
- EMPCLNT100
- G11clnt101**

BAPI ACCASSIGNMENT_LOADDATA

- BAPI_ADDRESSCONTPART_GETDETAIL
- BAPI_ADDRCONTPART_SAVEREPLICA
- BAPI_ADDRESSCONTPART_CHANGE
- BAPI_ADDRESSORG_GETDETAIL
- BAPI_ADDRESSORG_SAVEREPLICA
- BAPI_ADDRESSORG_CHANGE
- BAPI_ADDRESSPERS_GETDETAIL
- BAPI_ADDRESSPERS_SAVEREPLICA
- BAPI_ADDRESSPERS_CHANGE
- BAPI_APPRAISAL_CHANGE
- BAPI_APPRAISAL_CREATE
- BAPI_APPRAISAL_DELETE
- BAPI_APPRAISAL_GETDETAIL
- BAPI_APPRAISAL_GETLIST
- BAPI_APPRAISAL_STATUS_CHANGE
- BAPI_APPRAISAL_SCALE_GETDETAIL
- BAPI_APPRAISAL_MODEL_GETLIST
- BAPI_APPRAISAL_MODEL_GETDETAIL
- BAPI_APPRAISER_GETLIST
- BAPI_APPRAISEE_GETLIST
- BAPI_BUPR_SEMPL00_CHANGE
- BAPI_BUPR_SEMPL00_GETDETAIL
- BAPI_BUPA_FRG0070_ADD
- BAPI_BUPA_FRG0070_REMOVE

Interface definition

```
types: begin of BBP_STD_ACC
CLIENT(000003)          type C
LOG_SYS(000010)         type C
ACC_CAT(000005)         type C
VALUE(000064)          type C
VALID_FROM              type D
VALID_TO                type D
end of BBP_STD_ACC

types: begin of BAPIRET2
TYPE(000001)            type C
ID(000020)              type C
NUMBER(000003)          type N
MESSAGE(000220)         type C
LOG_NO(000020)          type C
LOG_MSG_NO(000006)      type N
MESSAGE_V1(000050)      type C
MESSAGE_V2(000050)      type C
MESSAGE_V3(000050)      type C
MESSAGE_V4(000050)      type C
PARAMETER(000032)       type C
ROW                     type I
FIELD(000030)           type C
SYSTEM(000010)          type C
end of BAPIRET2

types: RETURN            type table of
                        BAPIRET2

types: ACCOUNTASSIGNMENT type table of
BBP_STD_ACC
```

Call example

```
*Required Data Definition
data ACCOUNTASSIGNMENT type
ACCOUNTASSIGNMENT
data RETURN            type
RETURN

* Call Example
CALL FUNCTION 'BAPI_ACCASSIGNMENT_LOADDATA'
DESTINATION 'G11clnt101'

TABLES
ACCOUNTASSIGNMENT = ACCOUNTASSIGNMENT
RETURN            = RETURN
```




Usage of BAPIs in BSPs III

Example scenario:

- Call BAPI „BAPI_USER_GET_DETAIL“
 - Show details of a certain user
 - BAPI offers broad functionality for query of user data
 - Use of BAPI in order to show the „last changed on“ date



Example: Call BAPI in BSP – Step 1

Step 1: Definition of variables

- Temporary variables
data username type c.
data tmp_islocked type BAPISLOCKD.
data tmp_moddat type BAPIMODDAT.
- Interface variables
data ISLOCKED type BAPISLOCKD.
data LASTMODIFIED type BAPIMODDAT.



Example: Call BAPI in BSP – Step 2

Step 2: Definition CALL FUNCTION

```
CALL FUNCTION 'BAPI_USER_GET_DETAIL'  
  DESTINATION 'G11clnt101',  
  EXPORTING ....  
  IMPORTING .....
```

- CALL FUNCTION refers to the BAPI name
- DESTINATION refers to the RFC-connection
- EXPORTING refers to input parameters of a BAPI
- IMPORTING refers to output parameters of a BAPI



Example: Call BAPI in BSP – Step 3

Step 3: Complete call within a BSP:

```
CALL FUNCTION 'BAPI_USER_GET_DETAIL'  
  DESTINATION 'G51'  
  EXPORTING  
    username      = 'master-adm'  
  IMPORTING  
    lastmodified  = tmp_moddat.
```

- Query of the „Last changed on“ date (→lastmodified) of user „master-adm“
- Output is assigned to variable tmp_moddat



Example: Call BAPI in BSP – Result

- Result:

User details	
Master-Adm last modified?	20060710130916
Master-Adm locked?	UUUU

Fertig Internet 100%



Example: Call BAPI in BSP

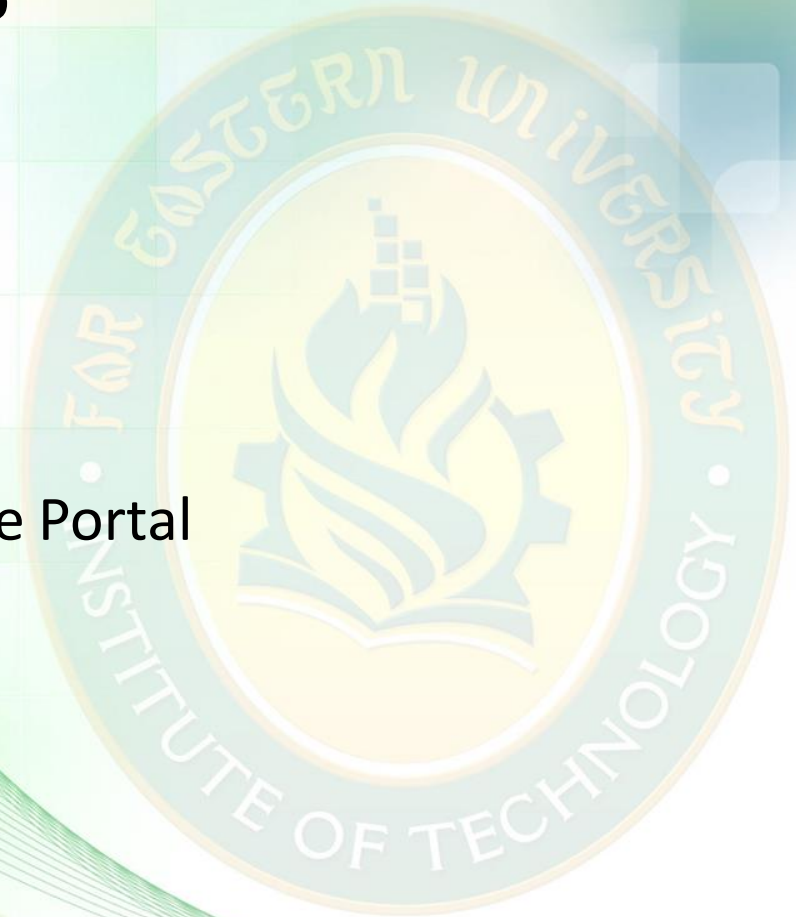
- Complete source code:

```
<%@page language="abap" %>
<%@extension name="htmlb" prefix="htmlb" %>
<htmlb:content>
  <htmlb:page title="Flugresultate" >
    <table width="100%" height="100%" cellpadding="2" border="0">
      <tr class="sapTbvCellStd"> <td bgcolor="Whitesmoke" valign="top">
        <h2> User details </h2>
        <%
          data: username type c.
          data ISLOCKED type BAPISLOCKD.
          data lastmodified type bapimoddat.
          data tmp_islocked type bapislockd.
          data tmp_moddat type bapimoddat.
          CALL FUNCTION 'BAPI_USER_GET_DETAIL'
            DESTINATION ,G51'
            EXPORTING
              USERNAME                = 'master-adm'
              IMPORTING
                islocked              = tmp_islocked
                lastmodified          = tmp_moddat.
            %>
        <tr>
          <td bgcolor="Whitesmoke" valign="top"> Master-Adm last modified? </td>
          <td bgcolor="Whitesmoke" valign="top"><%= tmp_moddat %></td>
        </tr>
        <tr>
          <td bgcolor="Whitesmoke" valign="top"> Master-Adm locked? </td>
          <td bgcolor="Whitesmoke" valign="top"><%= tmp_islocked %></td>
        </tr>
      </table>
    </htmlb:page>
  </htmlb:content>
```




HTMLB

- Extension of HTML by SAP
- HTML-Business
 - HTML-Business for Java
 - HTML-Business for ABAP
- HTMLB is used e.g. for Enterprise Portal





Comparison HTML / HTMLB

HTML



HTMLB

```
<%@page language="abap" %>

<html>
<head>
  <title>My BSP</title>
<% data: vari type i. %>
</head>

<body>
<!--This is a comment --%>
<% vari = 5 %>
In this coding following value
  is assigend to variable
  „vari“:
<%= vari %>
<br>
That's it.

</body>
</html>
```

```
<%@page language="abap" %>
<%@extension name="htmlb"
  prefix="htmlb" %>

<htmlb:content design="design2003" >
  <htmlb:page title=„My BSP" >
    <% data: vari type i. %>

    <!--This is a comment --%>
    <% vari = 5. %>
    In this coding following value is
      assigend to variable „vari“:
    <%= vari. %>
    <br>
    That's it using HTMLB.

  </htmlb:page>
</htmlb:content>
```




HTMLB Statements

`<%@extension name="htmlb" prefix="htmlb" %>`

- Naming of extension; in this case „htmlb“

`<htmlb:content design="design2003" >`

- Begin of content-tags with explicit design definition
- Standard designs: design2003, design2002 and classic
- Design2003 can be used only for MS IE Version 5.5 or higher

`<htmlb:page title="Meine BSP" >`

- Begin of page-tags with definition of title for BSP
- Explicit `<head>` tag is not required any more
- Explicit `<body>` tag is not required any more

`</htmlb:page>`

`</htmlb:content>`

- Closing tags for page and content



HTMLB – Text fields

- Text fields:

```
<htmlb:textView text = „Hi this is a textView“ design =„Emphasized“ />
```

Hi, this is a textView

- Text fields include different attributes:

- Text: The displayed text
- Design: Different design types
 - Emphasized
 - Header 1-3 - Headlines
 - Reference – *italic Reference*
 - Standard
 - etc.



HTMLB – Forms, Input fields

Forms

- `<htmlb:form id = "myFormId" method = "post" encodingType = "multipart/form-data" >`
Equivalent of *form* in HTML

Input Fields

- `<htmlb:inputField id="IP1" />`
- Input field:

What about an input field:



HTMLB – Buttons, Events

Buttons

- `<htmlb:button id = "SaveChanges" onClick = "SubmitChange" text = "Submit Button" onClientClick = "onInputProcessing(htmlbevent);" />`
- Button for sending forms
- Important: `onClientClick` raises event, in this case: `onInputProcessing(htmlbevent)`
- Event is called `onInputProcessing`
- Event_id is called: `htmlbevent`



HTMLB - Eventhandler

- Events are defined in the Eventhandler!

A screenshot of the HTMLB Event Handler editor. The interface shows a tabbed view with "Page" (example.htm), "Properties", "Layout", "Event Handler" (selected), "Page Attributes", and "Type Definitions". The "Event Handler" tab displays a list of events, with "OnInputProcessing" selected. The code editor shows the following code:

```
1  * event handler for checking and processing user input and
2  * for defining navigation
3
4  case event_id.
5  when 'htmlbevent'.
6      navigation->next_page( 'BACKHOME' ).
7
8  endcase.
```



HTMLB – RadioButtons I

Radiobuttons

```
<htmlb:radioButtonGroup id = "myID" columnCount = "2" >  
    <htmlb:radioButton id = "RBGenderFemale" text =  
        "female" tooltip = "I am female" disabled = "false" />  
    <htmlb:radioButton id = "RBGenderMale" text = "male"  
        tooltip = "I am male (and this button is disabled)"  
        disabled = "true" />  
</htmlb:radioButtonGroup>
```

- RadioButtons always belong to a RadioButtonGroup
- RadioButtonGroup with attributes
 - Id: Unique ID
 - columnCount: Number of columns
 - currentItem: Defines the active item



HTMLB – RadioButtons II

```
<htmlb:radioButtonGroup id = "myID" columnCount = "2" >  
    <htmlb:radioButton id = "RBGenderFemale" text =  
    "female" tooltip = "I am female" disabled = "false" />  
    <htmlb:radioButton id = "RBGenderMale" text = "male"  
    tooltip = "I am male (and this button is disabled)"  
    disabled = "true" />  
</htmlb:radioButtonGroup>
```

- **RadioButton with attributes**
 - Id: Unique ID
 - Text: Text of RadioButtons
 - Tooltip: Text that is displayed as tooltip
 - Disabled: RadioButton cannot be selected



HTMLB - Layout

- RadioButtons

Let's try some radio buttons:

☐ female ☐ male

- Tree with treeNodes

Some cool stuff: treeView

e-environment

▼ Desk

▼ Network

- LAN
- WAN
- Infrared

- TabStrips

Some other very cool stuff: tabstrips

Tab 1

Tab 2

Tab 4

Body of Tab 4

- TableView

TableView example 1

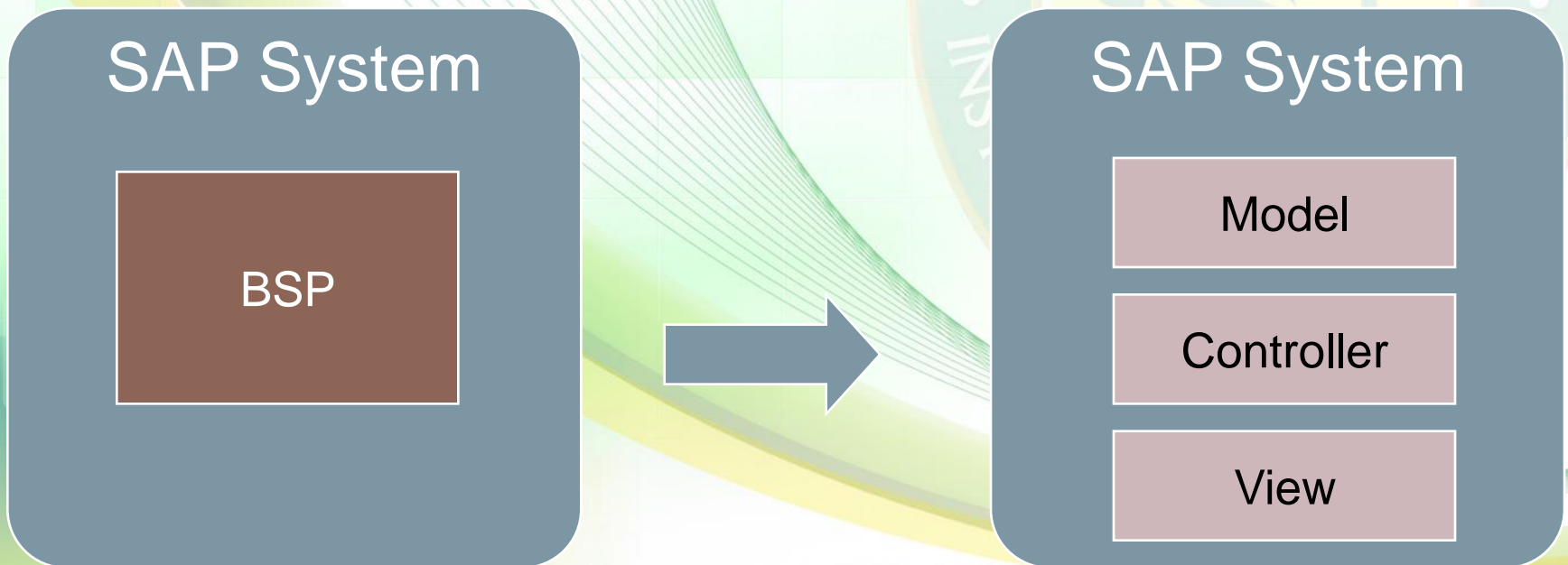
	MDT	ID	Nr	Lnd	Abflugstadt	FLH	Lnd	Ankunftstadt	FLH	Flugdauer	Abflug	Ankunft	Entf	In	Charter	Ankunft n Tag(e) später
<input type="checkbox"/>	902	AA	0026	DE	FRANKFURT	FRA	US	NEW YORK	JFK	7:20	08:30	09:50	3.851,0000	MI		0
<input type="checkbox"/>	902	AA	0064	US	SAN FRANCISCO	SFO	US	NEW YORK	JFK	5:21	09:00	17:21	2.572,0000	MI		0
<input type="checkbox"/>	902	AZ	0555	IT	ROME	FCO	DE	FRANKFURT	FRA	2:05	19:00	21:05	845,0000	MI		0
<input type="checkbox"/>	902	AZ	0788	IT	ROME	FCO	JP	TOKYO	TYO	12:55	12:00	08:55	6.130,0000	MI		1
<input type="checkbox"/>	902	AZ	0789	JP	TOKYO	TYO	IT	ROME	FCO	15:40	11:45	19:25	6.130,0000	MI		0

Zeile 2 von 46



Model View Controller

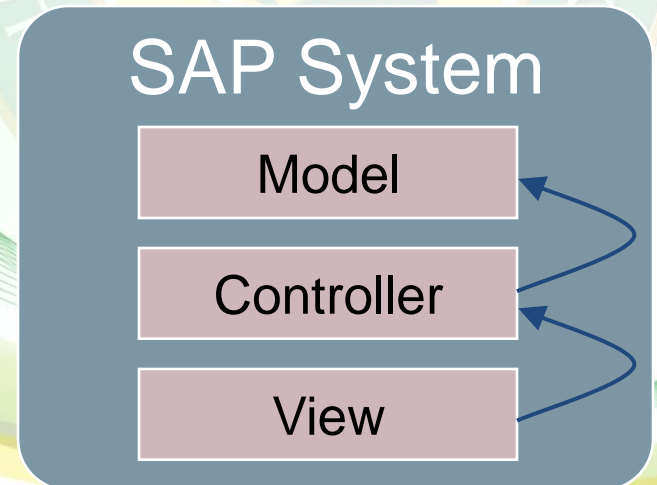
- Up-to-now: Presentation logic and application logic both are included in a BSP
- Model View Controller: Separation between logical layers





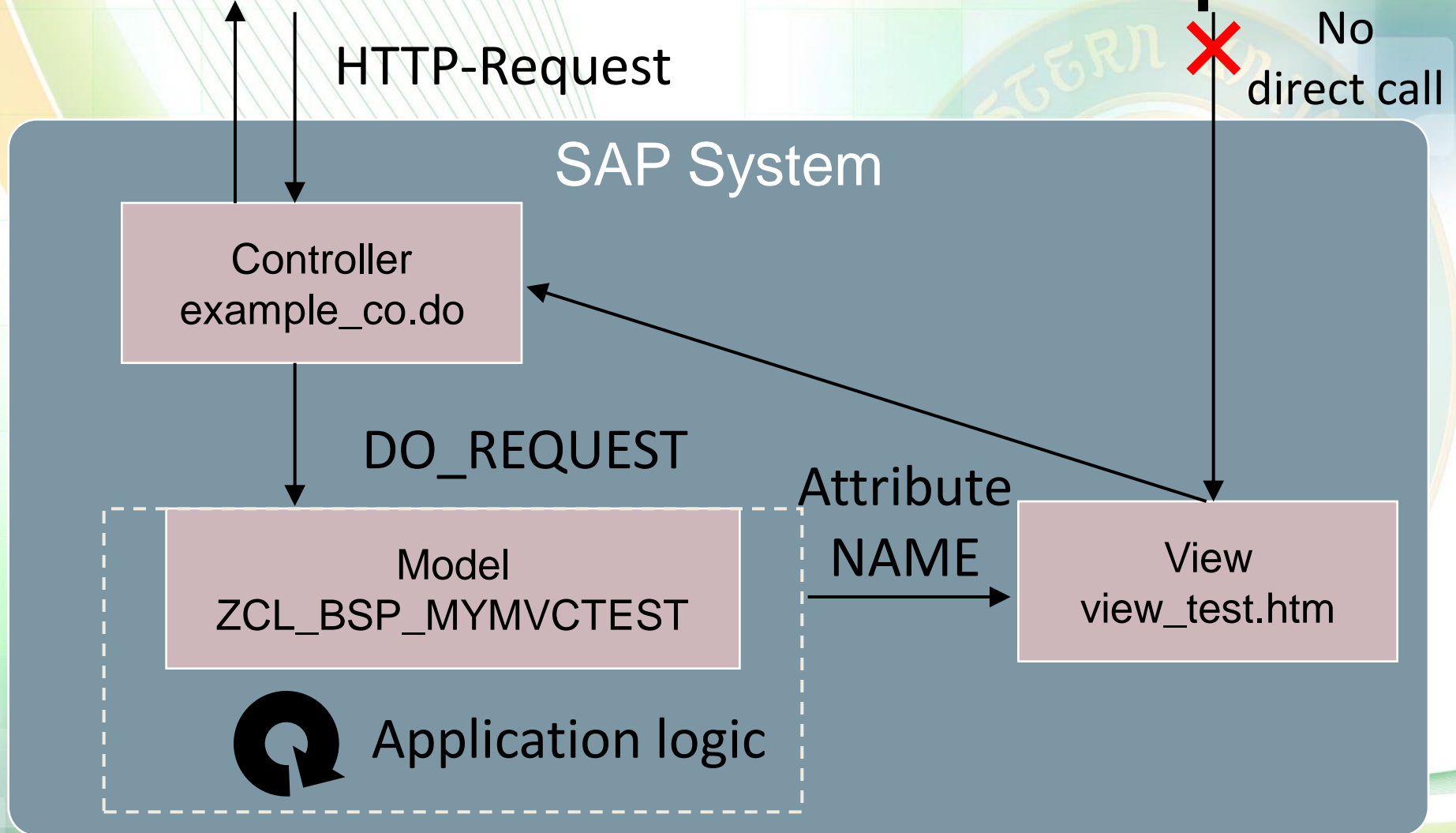
Model View Controller

- Model
 - Consists of a class derived from CL_BSP_MODEL
 - Processing takes place in background
- Controller
 - Handles Requests and forwards to Model
- View
 - Calls Controller
 - Only responsible for the visualization of data





Model View Controller - Example





Model View Controller - Example

1. Create Controller example_co.do
2. Derive own class ZCL_BSP_MYMVCTEST from CL_BSP_CONTROLLER2
3. Redefine method DO_REQUEST
4. Define view view_test.htm
5. Define attribute NAME
6. Call Controller





FEU INSTITUTE OF TECHNOLOGY
COLLEGE OF ENGINEERING • COLLEGE OF COMPUTER STUDIES

Model View Controller - Example

