## Dakota State University Beadle Scholar

Masters Theses

Spring 4-1-2011

# Integration of MS Excel with SAP Database

Shikha Jain Dakota State University

Follow this and additional works at: https://scholar.dsu.edu/theses

#### **Recommended** Citation

Jain, Shikha, "Integration of MS Excel with SAP Database" (2011). *Masters Theses*. 195. https://scholar.dsu.edu/theses/195

This Thesis is brought to you for free and open access by Beadle Scholar. It has been accepted for inclusion in Masters Theses by an authorized administrator of Beadle Scholar. For more information, please contact repository@dsu.edu.

# **Integration of MS Excel with SAP Database**

A graduate project submitted to Dakota State University in partial fulfillment of the requirements for the degree of

Master of Science in Information Systems 04,2011.

> By Shikha Jain

Project Committee:

Dr Shan Ronghua Dr. Stephen Krebsbach Dr. Mark Moran



# **PROJECT APPROVAL FORM**

We certify that we have read this project and that, in our opinion, it is satisfactory in scope and quality as a project for the degree of Master of Science in Information Systems.

Student Name: Shikha Jain.

Master's Project Title: Integration of MS Excel with SAP Database

on the Shem Date: 4/27/11 Faculty supervisor:

Committee member: Standy Date: 4/27/11

Committee member: Mark Moran	Date: 4 /27/11
------------------------------	----------------

## ABSTRACT

The aim of this project is to empower individuals with the ability to simply launch an excel with updated and consolidated data ( "Resource Time Management and Approval Process" in our case ), view and update it and upload the excel back with updated data at individual's convenience and hence updating the SAP integrated Database Directly, therefore connecting different platforms and accessing Data from a heterogeneous combination of Systems. (The MS Excel with all its popular features can also be stored and processed at user's convenience.)

Also the user can update the SAP database by simply loading back the updated excel sheet on the SAP database. The user just needs to load back the rows which are to be changed or just need to add the rows to be added in the SAP database (not the whole data).

Integration of MS Excel with sap is advantageous because sap has many complex business oriented logic enabled data mining capabilities, more and more business's are going towards it. Sap reduces the redundant errors in the system as it provides real time information. It is easy to implement globally and updates need only to be done once for implementing it company wide.

# DECLARATION

I hereby certify that this project constitutes my own product, that where the language of others is set forth, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions or writings of another.

I declare that the project describes original work that has not previously been presented for the award of any other degree of any institution.

Signed,

Shikha Jain. University ID – 1875943.

# TABLE OF CONTENTS

PROJECT APPROVAL FORM	ERROR! BOOKMARK NOT DEFINED.
ABSTRACT	
DECLARATION	
TABLE OF CONTENTS	V
LIST OF FIGURES	VII
CHAPTER 1	
INTRODUCTION	
BACKGROUND OF THE PROBLEM	
STATEMENT OF THE PROBLEM	
OBJECTIVES OF THE PROJECT	
CHAPTER 2	
LITERATURE REVIEW	
2.1 SAP	
2.2 REMOTE FUNCTION CALL (RFC)	
2.3 ABAP WEB DYNPRO	
2.4 Web services	
2.5 MS EXCEL SHEET	
CHAPTER 3	
SYSTEM DESIGN AND CREATION	
System design for creating dynamic excel she	EET 8
FUNCTION GROUP	
DATABASE TABLE	
TABLE TYPE	
REMOTE FUNCTION MODULE	
ABAP WEB DYNPRO OBJECT	
DOWNLOAD TO EXCEL SHEET	
UPLOAD EXCEL SHEET INTO SAP DATABASE	
WEB SERVICE	
CHAPTER 4	

RESULTS	57
Download Excel	57
UPLOAD EXCEL	59
CHAPTER 5	63
CONCLUSIONS	63
REFERENCES	64
APPENDICES	65
APPENDIX A: TECHNICAL ARCHITECTURE DIAGRAM	66
APPENDIX B: SAP CODE	67
Function Y_RFC_WS_CORP_PROJ_MEM_HOURS	67
Download Excel Methods	
ONACTIONEXPORT_TO_EXCEL	
ZSUBMIT	
UPLOAD EXCEL SHEET METHODS	71
ONACTIONUPLOAD_EXCEL	
SAVE_DB	

# LIST OF FIGURES

Figure 1. Web Dynpro Programming Model
Figure 2 . Web Services Architecture
Figure 3. Project Flow
Figure 4. Technical Architecture
Figure 5 .Function Group-Initial Screen
Figure 6.Function Group-Create Object
Figure 7.Create Function Group
Figure 8.Function Group-Create Object Entry
Figure 9.Function Group Created14
Figure 10. ABAP Dictionary-Initial Screen
Figure 11.Database Table Creation
Figure 12. Maintain Table 16
Figure 13. Maintain Table-Delivery Tab16
Figure 14. Defining Database Table 17
Figure 15. Database Table-Technical Settings 19
Figure 16. Selecting Table Maintenance Generator
Figure 17. Table Maintenance Dialog21
Figure 18. Saving Settings in Table Maintenace
Figure 19. Maintain Table Views:Initial Screen
Figure 20. Data Browser-Table YTABPROJ
Figure 21.Selction to create Table Type
Figure 22.Creating Table Type
Figure 23.Selection to create Function module
Figure 24.Create Function Module Screen
Figure 25. Create Function Module-Filling up information
Figure 26.Function Module properties
Figure 27.Selecting Function Module as Remote
Figure 28.Default source code of a function module

Figure 29.Specifying Export Parameters of function module	30
Figure 30. Source code of the function module	32
Figure 31.Source code of function module Y_FM_PROJ_MEMBERS	33
Figure 32.Exceuting function Y_CORP_PROJ_MEMBERS_HOURS	33
Figure 33.Result screen-Y_CORP_PROJ_MEMBERS_HOURS	34
Figure 34.Display data from PROJTAB from results screen	34
Figure 35. Creating ABAP Web Dynpro Object	35
Figure 36. Create Service call from Web Dynpro Object	35
Figure 37. Web Dynpro wizard	36
Figure 38.Web Dynpro Wizard: Select Service	37
Figure 39. Mapping controller context with view context	38
Figure 40. Designing end user view	38
Figure 41. Context mapping for Download Excel	39
Figure 42. Method Export_To_Excel	40
Figure 43. Method Z_SUBMIT	43
Figure 44. UPLOAD EXCEL UI Element	45
Figure 45. Buffer string to hold uploaded excel file	46
Figure 46. File upload element to browse the file on the users system	47
Figure 47. UI Element UPLOAD EXCEL linked to method UPLOAD_EXCEL	47
Figure 48. Method UPLOAD_EXCEL	48
Figure 49. Upload Excel context in web dynpro object	. 51
Figure 50. Method SAVE_DB	52
Figure 51.Creating Web dynpro application	. 53
Figure 52. Activating web dynpro application	. 54
Figure 53.Executing web dynpro application	. 55
Figure 54. User View	. 56
Figure 55. User entered PROJECT ID	. 57
Figure 56. User hits DOWNLOAD EXCEL BUTTON	. 58
Figure 57. Generated Excel Sheet	. 59
Figure 58. Browsing SAP database table	. 60
Figure 59. Table YTABPROJ entries	. 60

Figure 60. Changing APPROVED column in the excel sheet	. 61
Figure 61. Browsing the excel file to be uploaded	. 61
Figure 62. Users clicks UPLOAD EXCEL button	. 62
Figure 63. Updated database table YTABPROJ	. 62

# **CHAPTER 1**

# INTRODUCTION

#### **Background of the Problem**

Before this project was implemented, the HR department of Infobahn had to manually verify and process the billing of the number of project hours billed per person in a particular department. They need to manually take out the entries if they wanted to calculate the numbers of man hours per project or under a particular manager and store the data in the excel sheet for reference to the higher authorities. Thus majority of the time and resources were deployed in the billing hours calculation process as it is time consuming and the employees range thousands in number. Also it was difficult for the HR group to make changes to the data entries or add new data for the said purpose. They had to send email for the same to the data group who take care of the billing data and the data storage via manual data entries for each and every week for any resource allocated to any number of projects.

#### Statement of the problem

The work load from the HR Department can be significantly reduced by automating the whole process of employee hours and billing management.

HR can use a web link which will show them an optional filter to get employee hours records in a consolidated and convenient MS Excel sheet, using the user friendly and popular features of MS Excel they can sort, calculate, formulate, manipulate record information and approve or Reject the time sheets of the employees in the Excel itself. All they have to do is to simply save and load the excel sheet via the same web link. All the data entries for multiple records, employees and projects will be entered in the respective data records in the SAP Database (DB2), thus eliminating the huge resource allocation for manual data entries, approval emails and unconsolidated view for managers.

Using Excel sheet it's easy for the HR person to create the charts depending upon the data extracted from the SAP database. Charts are easy form of taking a broad picture in mind of for e.g. total number of resource hours under a particular manager or the total billing under projects.

Also it is easy for the HR people to make changes to the billing data or to add new data by simply adding it in the excel sheet and loading that excel sheet in the SAP database. It also saves a lot of time as they don't need to exchange emails and they get the confirmation right away if the data they have provided does not match to the type of columns in the database.

Moreover the end user doesn't need to have SAP installed in their system. They can perform data manipulation and see the data by just opening up a web link.

#### **Objectives of the project**

The Project Objective is accomplished in three major activities. (With activity one, being the major part of this project.)

- 1. The first step is to create the function modules in SAP ERP which the company is using ) that can access the data from the SAP database by specifying the type of information like project id, etc. The SAP database I will be using is DB2. The tables have been created in the database which contains the information related to the project hours billed per person.
- 2. The second step is to create a ABAP Web Dynpro Object. This object specifies the end user layout ( which will be rendered by the web service to the end user ) and specifies how to connect the excel sheet to the SAP database from the end user point of view (both uploading MS Excel sheet and downloading excel sheet)
- 3. The last step is to create the web services from the ABAP Web Dynpro object created in step 2. Web services are standalone and executable entities which can be published, searched and called across a network. Creating web service gives a web page url to the end user which they can use to access the application independent of the system.

# **CHAPTER 2**

## LITERATURE REVIEW

### 2.1 SAP

SAP is an Enterprise Resource Planning package which helps to automate a companies business and enhance its information system. SAP stands for "Systems, Applications And Products in Data Processing".

SAP ERP solutions offer products that cover the key areas of the business organizations. Some of them are: -

Customer Relationship Management.

Material Management.

Financials.

Human Resources.

**SAP** uses the concept of function modules which can be purchased, installed and run separately but all of them extract data from a common database. Each function module handles specific business task of its own but they can be linked when required. This helps in global integration of the data; updates can be done only once companywide and also provides real-time information. The modules include the utilities for marketing, sales, customer relationship management, inventory, human resources etc.

Deployment and maintenance costs of SAP systems vary depending on the size of the organization and whether the business needs full support of all the SAP modules or just few depending on the organization requirements. No doubt when SAP gets fully implemented the businesses efficiency greatly increases in terms of the enterprise resource planning. But

deploying SAP and customizing the SAP function modules according to business requirements takes lot of time and resources.

SAP client systems can communicate with each other using SAP provided security protocols. However to communicate with other systems to SAP vendor must use third party protocols.

Some of the advantages of implementing SAP are:-

- Updates on the system database needs to be done only once
- Allows easier integration of the global data like currencies, languages etc
- Reduces the redundant errors in the system as it provides real time information

SAP also has some disadvantages as mentioned below:-

- Implementing SAP has a risk of failure associated with it.
- Return on investment sometimes takes more time.
- The ERP packages of the vendor and the customer might be different and might be difficult to customize to suit each other. Also it might take lot of time and resources.

### 2.2 Remote function Call (RFC)

This project typically requires the use of Remote Function Call to integrate SAP with the external systems which in our case is MS Excel Sheet.

Remote function call is the SAP interface for communication between the client and server over TCP/IP protocol. Remote function calls can be inbuilt software that comes with SAP package or can be written using ABAP language using SAP ABAP workbench. RFC's provide in which external programs written in other languages or external software can use the data returned from the SAP server as well as can insert the data back into the SAP server.

Following interfaces made up the RFC interface:-

- An interface designed for calling ABAP programs
- To call a remote function module each ABAP program uses the command CALL
   FUNCTION FUNCTION\_NAME....DESTINATION. The parameters specified in

the DESTINATION specify that the called function module is to be run in a different system than to the calling system. CALL FUNCTION command helps SAP in the RFC communication with the external system.

- RFC function modules are to be registered in the SAP system as a remote function module or else they will not be called by the external system unless specified as RFC.
- If both calling and called programs are in the SAP system one of the function modules must be registered as remote.
- If one of the programs is the non-ABAP program, it needs to be programmed in such a way that it can communicate with the other ABAP RFC program.
- To call function modules in SAP system, external programs can use RFC supported interfaces and execute them in SAP systems. Similarly ABAP programs can also use these interfaces to execute external programs

To make a function module Remote :-

- Go to transaction SE37 (Function Builder).
- Go to the Attributes tab in the ABAP Workbench
- Set the "Remote-Enabled module" flag.

#### 2.3 ABAP Web Dynpro

ABAP Web Dynpro is the SAP standard to develop web based applications. It consists of a GUI environment and runtime environment. The transaction to create ABAP Web Dynpro Object is SE80.

The main advantages of ABAP Web Dynpro are:-

- 1. It is a GUI based environment which significantly reduces the implementation effort and time.
- 2. There is a strict separation between layout and data.
- 3. Components are reusable.
- 4. It is useful to develop user interface applications.

The Web Dynpro programming model is based on the Model View Controller model. **Model** forms the interface with the back end system to access the back-end data **View** makes the data available for view in the browser **Controller** makes the data available to the user using View, and also processes the data on user input and presents it back to the user through the View Following figure shows the Web Dynpro programming model

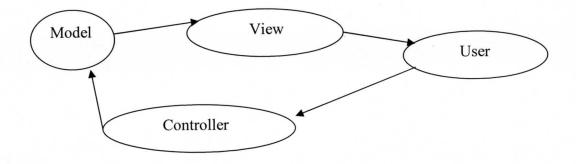


Figure 1. Web Dynpro Programming Model

#### 2.4 Web services

An easy way to integrate functions implemented on widely differing software components is by using Web Services. Web services are based on open and accepted standards. They can combine functions into a single process even if the functions are implemented on widely different software components. Web services are standalone and executable entities which can be published, searched and called across a network.

Web services architecture involves interactions between three primary entities- service requestor, service registry service provider. These entities interact with each other through bind, publish and find requests (Ref: 1). The service provider provides access to the web service. It publishes the description of the service in the service registry. The service requestor looks up the service description in the service registry and binds itself to the The logical view of the web services architecture is shown below

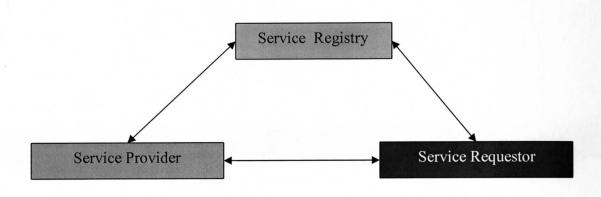


Figure 2. Web Services Architecture

#### 2.5 MS Excel Sheet

Microsoft Excel Sheet is a spreadsheet application which is distributed by Microsoft. It contains a grid of cells arranged in columns and rows which makes it easy to view and manipulate the data. It contains a library of inbuilt functions for arithmetical, statistical and engineering and financial needs. Moreover the user can create macros using VBA (Visual Basic for Applications) for user-defined calculations or operations.

The Excel sheet offers different options like to create charts, histograms, pivot tables etc from the data in the sheet to the user. It is useful to the user to get the broad view of the organizational data or performance etc.

Moreover Microsoft offers the flexibility to integrate MS Excel to other applications database. It helps to extract real-time data from other applications and present it to the user or update the database by loading Excel sheet into the system.

# **CHAPTER 3**

## SYSTEM DESIGN AND CREATION

### System design for creating dynamic excel sheet

The systems design is to gather the latest and updated information of the number of project hours billed per person in a particular project from SAP database and show it in an Excel Sheet to the end user. The user can specify criteria such as project hours billed for a person for a particular week ending date or multiple weeks or depending upon the project how many total man hours have been billed as a whole. Additional functionality can be added as per the requirements. Also the user can update or modify the data on the excel sheet and load it back into the SAP database by just loading the updated excel sheet into SAP system.

Following technologies are used in the project:-

Application Server (SAP) - Programming Language  $\rightarrow$  ABAP, ABAP Web dynpro

Web Service (SAP) - ABAP Web Service wizard

MS excel Sheet

Versions: -

MS Excel -- MS Office Suite 3.0

Sap -- ECC 6.0

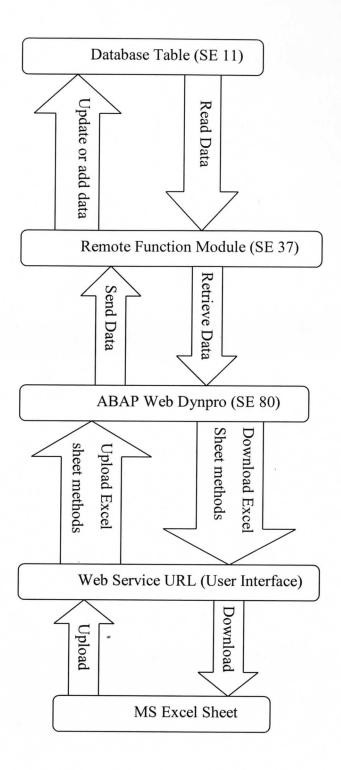
SAP based its architecture on 3-tier client-server model (presentation server, application server, database server).



### Figure 3. Project Flow

For this project we link different technologies in our multilayer architecture as shown in figure below:-

....Continued on next page



**Figure 4. Technical Architecture** 

We use web services to integrate SAP and MS Excel. Web services are application programming interfaces (API's) that can be accessed over internet and can be executed on remote system. Since the system landscapes are heterogeneous in nature all the functions of a process cannot be implemented using the same technology and on the same component. Web services can simplify this integration. Since our priority in programming this project is the selection of data, the integration of ABAP web service is appropriate.

We create functions (in SAP System) that can access the data in SAP from DB2 database based upon requesting parameters. After which we will create a ABAP Web Dynpro Object that can specify the end user layout and also define methods which can relate the user input with the RFC's we created. Then we create a web service in SAP from Web Dynpro object using ABAP web service wizard which gives us a url which the end user can access regardless of the system.

When the user access MS Excel and tries to access the data in DB2 database, the web service will invoke the function modules in SAP system to retrieve the data in DB2 database and show it in MS Excel in the form of columns and rows.

If the user wants to update the data in the database he/she needs to upload the excel sheet lying in his/her system, which then invokes the function modules in the RFC to update the database.

The sequence of activities which needs to be performed as per the business requirements are mentioned below:-

#### **Function Group**

We need to develop function modules which can extract data from SAP Database depending upon the requirements and show it in the Excel Sheet to the end user.

To develop the function module first we need to create the Function Group. It will be a group of function modules which can extract data from the database and present it to the Excel sheet. For the function module to extract the data from the database we also need to create the database table which contains all the relevant information according to business requirements. The database table is also part of the function group

The **function group** name which can be created using SE11 transaction of SAP is Y\_CORP\_PROJ\_MEMBERS\_HOURS

	em <u>H</u> elp
Object Navigator	
(← → ) 昌, 🖬 🖷 Edit Object	
MIME Repository	
器Repository Browser	
Repository Information System	
Tag Browser	
🖶 Transport Organizer	
Test Repository	
Function Group	

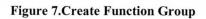
**Figure 5**. Function Group-Initial Screen

Since the function group does not exist it asks to whether we want to create the object

🔄 Create Obje	rct			1
Function Group Create Object?		ROJ_M	EMBERS	_HOURS does not exist.
Yes	No	x	Cancel	

Figure 6.Function Group-Create Object

Function group	Y_CORP_PROJ_MEMBERS_HOURS	
Short text	CORPORTATE PROJECT MEMBERS AND	THERE E
Person Responsible	JAIN (2)	

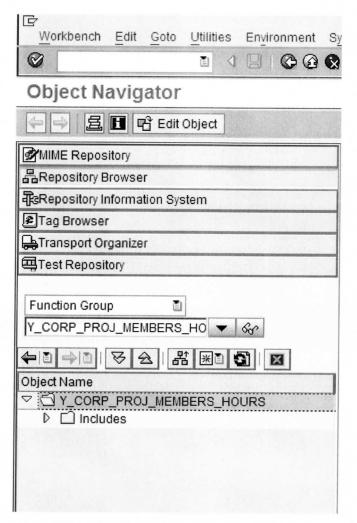


If we click on save window pops up

	Object Directory		
Object	KSIK FUOK I	_CORF_PROJ_HEHBERS_HOURS	
Attributes			
Package		\$TMP	
Person R	esponsible	JAIN 🕝	
Original S	System	DE2	
Original	anguage	EN English	

Figure 8.Function Group-Create Object Entry\*

We can specify \$TMP as a package to which the function group belongs . \$TMP is a package local to the system. We can specify any other package depending upon the requirements. Since we are creating a local object, choose local object in the options shows in the Fig above which then takes us to the following screen



**Figure 9. Function Group Created** 

It shows that function group Y\_CORP\_PROJ\_MEMBERS\_HOURS is created

## **Database table**

Then we need to create the Database table which contains the information such as PROJ\_ID, EMP\_ID, EMP\_NAME, HOURS\_BILLED, WEEK\_END\_DATE etc

For that we go to SE11 transaction of SAP which takes us to ABAP Dictionary initial screen

Di	ctionary Object	Edit	Goto	Utiliti	es	Environ	ment	Syste
	SE11					60		
AB	AP Dictio	nary	y: Ini	tial	Sci	reen		
品	I & ₩ H							

Figure 10. ABAP Dictionary-Initial Screen

ABAP dictionary is a collection of database objects where we can create objects like table, views, data type etc. For our case we create database table named YTABPROJ.

Database table	YTABPROJ 🗃
OView	
O Data type	
O Type Group	
ODomain	
O Search help	
O Lock object	
හි Display	Change Create

## Figure 11.Database Table Creation

When we click on Create it takes us to the following screen

Table Edit Goto Utilities Extr	
Dictionary: Maintain T	able
+ → ୭2°° 6 6 . ¢	🔠 🛃 🔲 📴 👔 Technical Settings Indexes Append Structure
ransp. Table YTABPROJ	New(Revised)
hort Description	
Attributes Delivery and Mainte	mance Fields Entry help/check Currency/Quantity Fields
Attributes Delivery and Mainte	A
Delivery Class	A Display/Maintenance Allowed with Restrictions N Display/Maintenance Not Allowed
Delivery Class	A Display/Maintenance Allowed with Restrictions

Figure 12. Maintain Table

Since we need to add columns and define the data types of those columns we choose the option Display Maintenance Allowed.

Dictionary: Maintain T	able
4 - 1 1 2 3 <b>C</b> 6 1 4	• 品 / Indexes Append Structure
Fransp. Table YTABPROJ	New(Revised)
Short Description Y CUSTOM T	ABLE FOR PROJECT ASSET NAMES & BILLING
Attributes Delivery and Mainte	

Figure 13. Maintain Table-Delivery Tab

# This takes to the following screen

					al Settings Indexes Append Structure	
		4 <b>.</b> 5		Technic	al Settings Indexes Append Structure	
	PROJ	masa	ve(Revised)			
YCU	STON	I TABLE FOR PR	OJECT ASSET N	IAMES & BI	LLING	
very a	nd Ma	aintenance F	ields Entry	help/check	Currency/Quantity Fields	
	81	474	P Srob Hole			
Key		and the second sec				1/8
notice and a second	Annese the					
	CALIFORNIA COMPANY	and the second s				
		CHARS				
		CHAR20	CHAR			
		CHAR20	CHAR			
		NUM	NUMC			
		CHAR1	CHAR	1		
ineres						
- Andrew Ann						
	J	I				
	Key	Key Initi.         Key Initi.         M	Very and Maintenance       F         Key Initi       Data element         Ø       MANDT         Ø       CHAR10         Ø       CHAR20         O       CHAR1         O       CHAR2	Very and Maintenance       Fields       Entry	Very and Maintenance       Fields       Entry help/check	Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a construction         Image: Second process of a construction           Image: Second process of a c

# Figure 14. Defining Database Table

Here we have added the columns for the table and defined the data types of those columns respectively. Following columns and their associated data types have been added

Table Name - YTABPROJ

Column name	Data Type
CLNT	MANDT
WEEK END DATE	DATUM
EMP ID	CHAR8
LAST NAME	CHAR20
FIRST NAME	CHAR20
HOURS BILLED	NUM
APPROVED	CHAR1
PROJ ID	CHAR10

The composite primary key of the table consists of columns – CLNT, WEEK END DATE, EMP ID, PROJ ID

Every table in SAP needs to be client specific, so it is mandatory to include column name CLNT for ever table we create.

Then we go to Technical Settings tab

Technical Settings

Which takes us to the following screen

	THE STATES IN THE STATES INTO STATES IN THE STATES INTO STATESTATES INTO STATE	echnical Settings	
2 6 Revised	>Active		
Name	YTABPROJ	Transparent Table	
Short text	Y CUSTOM TA	ABLE FOR PROJECT ASSET NAMES & BILLING	
Last Change	AJAIN	12/09/2010	
Status	New	Not saved	
Logical storage par			
Data class	APPLO		
Size category	0		
Buffering			
Buffering not all	owed		
and the second sec	ed but switched off		
O Buffering switch			
Buffering type			
Single records	buff.		
Generic Area B	uffered	No. of key fields	
***************************************			1

# Figure 15. Database Table-Technical Settings

Here we can specify the technical settings of the table.

Then to let the rows need to be added to the table we need to go to Table Maintenance Generator

ictionary: M	Settings Display Object List Ø Worklist	Ctrl+	Shift+F5	\$2 42 1 2 2 2 2 4 4 4 4 4 4 4 4 4 4 4 4 4
<b>1</b> 23 F	-		I Settings Indexes Append Structur	
ansp. Table ort Description	Activation Log Database Object			
Attributes Del	iv Runtime Object Assign Authorization Gre	oup		
(CRB-	Table Maintenance Gen	erator	efined Type	
Field	Table Contents		cim_Short Description	
CLNT	Where-Used List	Ctrl+	Shift+F3	0 Client
WEEK ENDDATE	Versions		O Date	
EMPID	CHARS	CHAR	8	0 Character field, 8 characters long
LASTNAME	CHAR20	CHAR	20	0 Char 20
FIRSTNAME	CHAR20	CHAR	20	0 Char 20
HOURSBILLED	NUM	NUMC	2	0 Sequence number
APPROVED	CHAR1	CHAR	1	0 Single-Character Indicator

Figure 16. Selecting Table Maintenance Generator

This takes to the following screen

Generated Objects	Edit Goto Environment	t <u>U</u> tilities System <u>H</u> elp
0	1 I I 🖉	④ Q □ B B B C C C C B B B C Q G
Generate Tabl	e Maintenance	Dialog: Generation Environment
🖉 🛅 Find Scr. Nun	nber(s)	
Table/View	YTABPROJ	
Technical Dialog Details	3	
Authorization Group	&NC& 🗇 w/o auth. grou	up
Authorization object	S_TABU_DIS	
Function group	Y CORP PROJ MEMB	Fn.Gr.Text
Package	\$TMP	Temporary Objects (never transported!)
Maintenance Screens	/	
Maintenance type	O one step	
	two step	
Maint. Screen No.	Overview screen	100
	Single screen	200
Dialog Data Transport D	Details	
Recording routine	O Standard recording	g routine
	no, or user, record	ling routine
Compare Flag	Automatically Adj	justable 🚺 Note

Figure 17. Table Maintenance Dialog

		COCIDHBIOD	
Generate 1	able Maintenanc	e Dialog: Generation	Environment
Find Scr. Nu	mber(s)		
Table/View	YTABPROJ	]	
Technical Dialog I	Change Object Directory	Entry	
Authorization Gro	Object R3TR FUGR Y	CORP PROJ MEMB	
Authorization obje			
Function group	Attributes		and the second s
Package	Package	STMP	0
Maintenance Scr	Person Responsible	AJAIN	
Maintenance typ		and the second	
<b>Constant</b>	Original System	DE2	
Maint. Screen N	Original language	EN English	
	B 2 & Lock Overview		
Dialog Data Transp	Completion of Co		
Recording routine	Ö Standard record:	na rouline	
	(1) DO, OF USET, FECO		
Compare Flag	Automatically A		

## Figure 18. Saving Settings in Table Maintenace

The table is also specified in the package \$TMP and local to the system.

Then to add data to the table we need to go to the SM30 transaction which takes us to the Maintain Tables Screen

☑ SM30	1		a 🛛 I 🗎	的間日	80.06	31 🕱 🛛	2
Maintain Tabl	e Views: II	nitial S	creen				
🛗 Find Maintenance	Dialog						
Table/View	YTABPROJ						
	YTABPROJ						
Table/View Restrict Data Range No Restrictions	YTABPROJ						
Restrict Data Range	YTABPROJ						
Restrict Data Range No Restrictions	YTABPROJ						
Restrict Data Range No Restrictions O Enter conditions	YTABPROJ						
Restrict Data Range No Restrictions O Enter conditions	YTABPROJ			品の			

### Figure 19. Maintain Table Views: Initial Screen

To add rows to the table we need to select Maintain option .

Then we can add the rows in the table as shown below

Γ	Menu	4		Back Ex	it Cancel System	Display Choose	Sort Ascending So	rt Descendin
	ble: splay		ABPROJ 8 of 8	Fixed	d Columns:	4 L1	ist Width 0250	
	CLNT	PROJID	WEEK_ENDDATE	EMPID	LASTNAME	FIRSTNAME	HOURSBILLED	APPROVED
5	600 600		11/06/2010 11/13/2010	01875943 10012760		SHIKHA MARIA	40 38	Y Y

Figure 20. Data Browser-Table YTABPROJ

## Table type

Since we will be using the tables YTABPROJ in our function module as a data type we need to define a data type of the table YTABPROJ before we can add it in our function module. To do this we go to SE11 transaction, choose the function group Y PROJ\_MEMBERS\_HOURS

It will display all types of objects which can be created under a particular function group. Since we need to create the table type , we choose create table type option

Object Name	
<ul> <li>✓ ☐ \$TMP AJAIN</li> <li>✓ ☐ Dictionary Objects</li> <li>✓ ☐ Database Tables</li> <li>YTABPROJ</li> </ul>	
D Class Library	<u>T</u> able Type
<ul> <li>D Function Groups</li> <li>D Web Dynpro</li> <li>D Transactions</li> <li>D Form Objects</li> </ul>	<u>S</u> tructure D <u>a</u> ta Element D <u>o</u> main
	S <u>e</u> arch Help Lock Object
	Type Group

Figure 21.Selction to create Table Type

The screen asks about the options we need to give to the table type we want to create

Table Type	YTABTYP_	PROJ	New(Revised)	
Short text	TABLE TY	PE FOR YTABPROJ	nonconsensed	
Attributes Lin	е Туре 🧹	Initialization and Access	Key	
● Line Type	YTA	BPROJ	Ø	
O Predefined Type Data Type				
No. of Character	s 0	Decimal Place	es 0	
O Reference type				
O Name of Ref. T	ype			
O Deference to D	a defined To			
O Reference to P	redenned Ty	Data Type		
		Length 0	Decimal Places	0

Figure 22. Creating Table Type

## **Remote Function Module**

After the table type is created we can create the **Remote Function Module** which takes the data from the SAP database and displays it in Excel. We call it remote because it is connected to system other than SAP (to which it belongs) through web services. It will display the data in Excel using Excel web services.

To create the function module go to the SE11 transaction of SAP and create the function named Y\_RFC\_W\_CORP\_PROJ\_MEM\_HOURS as shown below

Object Navigator		
🔶 🚽 🚊 🖬 📽 Edit Object		
MIME Repository		
器Repository Browser		
Repository Information System		
PTag Browser		
ጔTransport Organizer		
马Test Repository		
Function Group		
Dbject Name		
Diject Name	Create	Function <u>G</u> roup
Dbject Name          Dbject Name         Image: Corp_PROJ_MEMBERS_HOURS         Includes         LY_CORP_PROJ_MEMBERS_H(	<u>C</u> reate C <u>h</u> ange	▶ Function Group Function Module
Dbject Name           Dbject Name           Includes	<u>C</u> reate	Eunction Module     Subroutine
Dbject Name          Dbject Name         Image: Corp_PROJ_MEMBERS_HOURS         Includes         LY_CORP_PROJ_MEMBERS_H(	<u>C</u> reate Change Display Check	Eunction Module     Subroutine     PBO Module
Dbject Name          Dbject Name         Image: Corp_PROJ_MEMBERS_HOURS         Includes         LY_CORP_PROJ_MEMBERS_H(	<u>C</u> reate Change Display Check Act <u>i</u> vate	Function Module     Subroutine     PBO Module     PAI Module
Dbject Name          Dbject Name         Image: Corp_PROJ_MEMBERS_HOURS         Includes         LY_CORP_PROJ_MEMBERS_H(	<u>Create</u> Change Display Check Activate Test Sequence	Eunction Module     Subroutine     PBO Module     PAI Module     Screen
Dbject Name          Dbject Name         Image: Corp_PROJ_MEMBERS_HOURS         Includes         LY_CORP_PROJ_MEMBERS_H(	<u>C</u> reate Change Display Check Act <u>i</u> vate	Eunction Module     Subroutine     PBO Module     PAI Module     Screen     GUI Status
Dbject Name          Image: Corp_PROJ_MEMBERS_HOURS         Image: Corp_PROJ_MEMBERS_HOURS         Includes         LY_CORP_PROJ_MEMBERS_HOURS	<u>Create</u> Change Display Check Activate Test Sequence Unit Test Copy	
Dbject Name          Dbject Name         Image: Corp_PROJ_MEMBERS_HOURS         Includes         LY_CORP_PROJ_MEMBERS_H(	<u>Create</u> Change Display Check Activate Test Sequence Unit Test	
Dbject Name          Image: Corp_PROJ_MEMBERS_HOURS         Image: Corp_PROJ_MEMBERS_HOURS         Includes         LY_CORP_PROJ_MEMBERS_HOURS	<u>C</u> reate Change Display Check Activate Test Sequence Unit Test Copy Delete Eind	
Dbject Name           Dbject Name           Image: Corp_PROJ_MEMBERS_HOURS           Includes           LY_CORP_PROJ_MEMBERS_H(	<u>Create</u> Change Display Check Activate Test Sequence Unit Test Copy Delete	Function Module         Subroutine         PBO Module         PAI Module         Screen         GUI Status         GUI Titles         Include         Transaction         Class (Definition)
Dbject Name           Dbject Name           Y_CORP_PROJ_MEMBERS_HOURS           T           Includes           LY_CORP_PROJ_MEMBERS_H(	<u>C</u> reate Change Display Check Activate Test Sequence Unit Test Copy Delete Eind	

Figure 23.Selection to create Function module

Here we need to specify the function name and the function group name

26

_	ORP_PROJ_MEMBER		
	Create Function M	odule	×
14	Function Module	ð	
	Function group	Y_CORP_PROJ_MEMBERS_HOURS	
4	Short text		

Figure 24.Create Function Module Screen

Create Function M	odule	X
Function Module	Y_RFC_WS_CORP_PROJ_MEM_HOURS	
Function group	Y_CORP_PROJ_MEMBERS_HOURS	
Short text	RFC (FM) FOR PROJECT ASSET DETAILS	

Figure 25. Create Function Module-Filling up information

When we Save the function module name it gives us the following screen

MIME Repository	Function module	_RFC_WS_CO	RP_PROJ_MEM_H	OURS Inactive		
Repository Browser	Altribules Import	Export	Changing	Tables Excepti	ons Source cod	
子。 保Repository Information System		La provincia de			000105 000	
₽ Tag Browser	Classification					
异 Transport Organizer	Function Group	Y_CORP_PR	ROJ_MEMBERS_HO	URS CORPORTATI	E PROJECT MEMBE	RS AND T
Test Repository	anneal 1		FOR PROJECT A	OR PROJECT ASSET DETAILS		
Function Group	Processing Type		Gei	neral Data		
Y_CORP_PROJ_MEMBERS_HO V	Normal Function Modu	le	Pe	rson Responsible	AJAIN	
	O Remote-Enabled Modu	La	st Changed By	AJAIN		
	O Update Module		Ch	anged on	12/09/2010	
Object Name	Start immed.	Ра	ckage	STMP		
CORP_PROJ_MEMBERS_HOURS     Function Modules	O Immediate Start, No	Pro	igram Name	SAPLY_CORP_PR	OJ_MEMBI	
✓  ☐ Includes	O Start Delayed		INC	CLUDE Name	LY_CORP_PROJ_	MEMBERS
LY_CORP_PROJ_MEMBERS_HOURSTOP	O Coll.run		Ori	ginal Language	EN	
LY_CORP_PROJ_MEMBERS_HOURSUXX			No	t released		
				Edit Lock		
				Global		

**Figure 26. Function Module properties** 

Since the function module we need should be remote function module we check the

box

'Remote - Enabled Module'

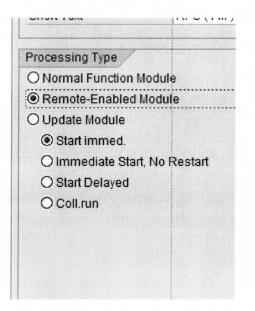
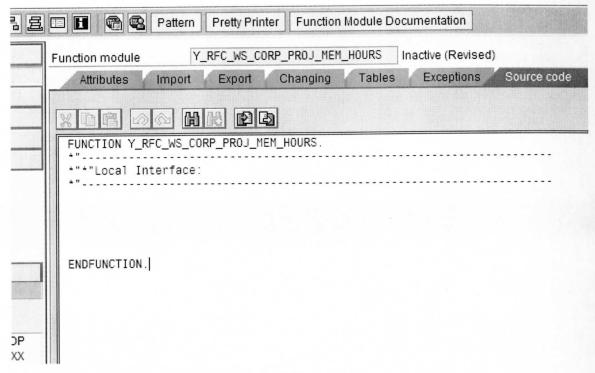


Figure 27.Selecting Function Module as Remote

The default source code of any function module created within SAP looks like below if we check the tab 'Source Code'



#### Figure 28.Default source code of a function module

Since we need to export the data into the Excel sheet and the data is of the form of the table as in the database, the export parameter PROJTAB is of the type YTABTYP\_PROJ.

	9 B M M 8 2 2					
Function Builder: Change Y_RFC	WS_CORP_PF	ROJ_MEN	LHOURS			
		attern Pretty P	nnter Function Module	Documentatio	n	
MIME Repository	Function module	Y_RFC_WS_	CORP_PROJ_MEM_HOURS	Inactive (R	evised)	
ARepository Browser	Attributes Im	port Econt	Changing Tab	les Except	tions Source code	
arepository Information System						
E Tag Browser	xore					
Transport Organizer	Parameter Name	Typing	Associated Type	Pass Val	Short text	Long Text
Test Repository	PROJTAB	TYPE	YTABTYP_PROJ	0	Y CUSTOM TABLE FOR PROJECT ASS	Crea
	•••••					
Function Group						
Y_CORP_PROJ_MEMBERS_HO 🔻 🗞						
Object Name						
V CORP_PROJ_MEMBERS_HOURS						
C Function Modules						
Y_RFC_WS_CORP_PROJ_MEM_HOURS						
Includes						
			9			
	-					
	3					

Figure 29. Specifying Export Parameters of function module

The import parameter of the function module is PROJ\_ID, and WEEK\_ENDDATE.

The function module first takes all the rows in the database table YTABPROJ as input for a particular project by calling another function module Y\_FM\_PROJ\_MEMBERS.

The function module takes the input as week end date and returns all the rows from the database table that end with that particular week end date. This way it displays the list of the total number of billed hours submitted by the employees for a particular week in a particular project.

The Souce Code of the function module Y\_RFC\_WS\_CORP\_PROJ\_MEM\_HOURS is shown below.

FUNCTION Y\_RFC\_WS\_CORP\_PROJ\_MEM\_HOURS.

\*"\_\_\_\_\_

- \*"\*"Local Interface:
- \*" IMPORTING
- \*" VALUE(PROJECT\_ID) TYPE CHAR10
- \*" VALUE(WEEK\_ENDDATE) TYPE DATUM OPTIONAL
- \*" EXPORTING
- \*" VALUE(PROJTAB) TYPE YTABTYP\_PROJ

\*"\_\_\_\_\_

DATA: IT\_PROJTAB TYPE STANDARD TABLE OF YTABPROJ.

DATA: IT\_PROJTAB\_WEEK TYPE STANDARD TABLE OF YTABPROJ.

DATA: WA\_PROJTAB TYPE YTABPROJ.

CALL FUNCTION 'Y\_FM\_PROJ\_MEMBERS' EXPORTING PROJECT\_ID = PROJECT\_ID IMPORTING PROJECT\_TAB = IT\_PROJTAB.

\*\*SELECT \* FROM YTABPROJ APPENDING TABLE PROJTAB WHERE PROJID = PROJECT\_ID AND WEEK\_ENDDATE = WEEK\_ENDDATE.

IF WEEK\_ENDDATE IS NOT INITIAL.

LOOP AT IT\_PROJTAB INTO WA\_PROJTAB WHERE WEEK\_ENDDATE = WEEK\_ENDDATE.

\*\*READ \* FROM IT\_PROJTAB INTO WA\_PROJTAB WITH KEY WEEK\_ENDDATE = WEEK\_ENDDATE.

APPEND WA\_PROJTAB TO IT\_PROJTAB\_WEEK.

ENDLOOP.

ELSE.

IT\_PROJTAB\_WEEK = IT\_PROJTAB. ENDIF.

PROJTAB = IT\_PROJTAB\_WEEK.

#### ENDFUNCTION.

E Function Module Edit Goto Utilities Environmer	it System Help
Function Builder: Change Y_RFC	WS CORP_PROJ_MEM_HOURS
	E E R R Pattern Pretty Printer Function Module Documentation
MIME Repository	Function module Y_RFC_WS_CORP_PR0J_MEM_HOURS Active
Repository Browser	Attributes Import Export Changing Tables Exceptions Source code
Repository Information System	
ETag Browser	
🖵 Transport Organizer	FUNCTION Y_RFC_WS_CORP_PROJ_MEM_HOURS.
Test Repository	+"-"Local Interface:
Function Group Y_CORP_PROJ_MEMBERS_H0 ▼ & COBJEC Name © LY_CORP_PROJ_MEMBERS_HOURS © LY_CORP_PROJ_MEMBERS_HOURS © Sunction Modules Y_FM_PROJ_MEMBERS Y_RFC_WS_CORP_PROJ_MEM_HOURS D _ Includes	<ul> <li>TIPEORTING</li> <li>YALUE(PROJECT_ID) TYPE CHAR10</li> <li>VALUE(WEEK_ENDDATE) TYPE DATUM OPTIONAL</li> <li>EXPORTING</li> <li>VALUE(PROJTAB) TYPE YTABTYP_PROJ</li> <li>VALUE(PROJTAB) TYPE STANDARD TABLE OF YTABPROJ.</li> <li>DATA: IT_PROJTAB_WEEK TYPE STANDARD TABLE OF YTABPROJ.</li> <li>DATA: IT_PROJTAB_WEEK TYPE STANDARD TABLE OF YTABPROJ.</li> <li>DATA: WA_PROJTAB TYPE YTABPROJ.</li> </ul>
	CALL FUNCTION 'Y_FM_PROJ_MEMBERS' EXPORTING PROJECT_ID = PROJECT_ID IMPORTING PROJECT_TAB = IT_PROJTAB. **SELECT * FROM YTABPROJ APPENDING TABLE PROJTAB WHERE PROJID = PROJECT_ID AND WEEK_ENDDATE = WEEK_ENDDATE. IF WEEK_ENDDATE IS NOT INITIAL.
	LOOP AT IT_PROJTAB INTO WA_PROJTAB WHERE WEEK_ENDDATE = WEEK_ENDDATE.

Figure 30. Source code of the function module

The remote function module Y\_RFC\_WS\_CORP\_PROJ\_MEM\_HOURS calls another function module Y\_FM\_PROJ\_MEMBERS. The source code Y\_FM\_PROJ\_MEMBERS is shown below

Function Builder: Change Y_FM	
	S C Pattern Pretty Printer Function Module Documentation
MilliE Repository Repository Browser ReRepository Information System	Function module Y_FM_PR0J_MEMBERS Active Attributes Import Export Changing Tables Exceptions Source code
E Tag Browser	FUNCTION Y_FM_PROJ_MEMBERS.
Function Group	
Image: State	SELECT * FROM YTABPROJ APPENDING TABLE PROJECT_TAB WHERE PROJID = PROJECT_ID.

Figure 31.Source code of function module Y\_FM\_PROJ\_MEMBERS

Function module Y\_FM\_PROJ\_MEMBERS has a importing parameter PROJ\_ID and exporting parameter PROJECT\_TAB.

It takes project id as an input and returns all the rows from the database table associated with that project id.

# Executing function module Y\_CORP\_PROJ\_MRMBERS\_HOURS

Menu		Save data record	Back E	xit Cance	I System	Execute	Debugging
Test for function group Function module Uppercase/Lowercase RFC target sys:		ORP_PROJ_MEMBER FC_WS_CORP_PROJ		RS			
Import parameters		Value					
PROJECT ID	11/6/2010						

Result is displayed for the week end date of 11/6/2010

Menu	Save data record Back Exit Cancel System Ch
est for function group unction module ppercase/Lowercase	Y_CORP_PROJ_MEMBERS_HOURS Y_RFC_WS_CORP_PROJ_MEM_HOURS
ntime: 1,853 Micro	oseconds
C target sys:	
Import parameters	Value
PROJECT_ID WEEK_ENDDATE	11/06/2010
WEEK_ENDDATE	11/06/2010
-	11/06/2010 Value

Figure 33.Result screen-Y\_CORP\_PROJ\_MEMBERS\_HOURS

Clicking on PROJTAB 1 Entry shows the list of rows returned from the database table

### YTABPROJ

Me	nu 🕴 🗌		Ba	ck Exit Cancel	System Single Entry	Left Margin	n Scroll Left	Scroll Right	Right Mare
	1 Entr	гy							
CLN	PROJID	WEEK_ENDDA	EMPID	LASTNAME	FIRSTNAME	HO	A		
100		11/06/2010	01875943	JAIN	SHIKHA	40	_		
		11/06/2010	01875943	JAIN	SHIKHA	40	Y		

Figure 34.Display data from PROJTAB from results screen

### **ABAP Web Dynpro Object**

Now the next step is to create an ABAP Web Dynpro object which will be a web service to export the data from SAP database to the excel sheet or to update SAP database from a excel sheet.

We can create ABAP Web Dynpro object in se80 transaction on ABAP workbench The ABAP Web Dynpro object we are creating is Y\_WDA\_PROJ\_MEMB\_HOURS.

Name	Y_WDA_PROJ_MEMB_HOURS	
Description	Y_WDA_PROJ_MEMB_HOURS	
Туре	Web Dynpro Component	
	O Web Dynpro Component Interface	)
Window Name	Y_WDA_PROJ_MEMB_HOUR	
View Name	MAIN	

Figure 35. Creating ABAP Web Dynpro Object

To call the remote function module we created earlier from ABAP Web Dynpro object we need to create a service call from Web Dynpro Object as shown below

Web Dynpro Comp. / Intf.         Y_WDA_PROJ_MEMB_HOURS         ↓		Used Components Implemented interfaces Used Web Dynpro Components				
Object Name ▽ 4월 Y_WDA_PROJ_MEMB_HOURS		Component Use	Component	Description of Component		
COMPONENTCONTROLLER	Create	· · · · · · · · · · · · · · · · · · ·	Web Dynpro Component (Interface)			
<ul> <li>▷ component Interface</li> <li>♥ I Views</li> <li>I MAIN</li> <li>♥ Windows</li> <li>■ Y_WDA_PROJ_MEMB_F</li> </ul>	Change Display Create/Change	► Configuration	V <u>i</u> ew <u>W</u> indow <u>C</u> ustom Controller			
	Check → Activate		<u>Service Call</u> <u>Mime Object</u>			
	C <u>o</u> py Re <u>n</u> ame <u>D</u> elete		Web Dynpro Application			
	Wh <u>e</u> re-Used Lis Other Functions					

Figure 36. Create Service call from Web Dynpro Object

This opens a wizard which will automatically create a code to call RFC from Web Dynpro Object.

년 Web Dynpro Wizard: Select Con	troller		×
<ul> <li>Start</li> <li>Select Controller</li> <li>Select Service Type</li> <li>Select Service</li> <li>Adapt Context</li> <li>Specify Method Name</li> <li>Generate Controller</li> </ul>	want to enha component. Enter the nar • Enter contro compo	event of an already existing controller, use the input help. / Controller	•
	Component	Y_WDA_PROJ_MEMB_HOURS	
	Controller	COMPONENTCONTROLLER	
	[	🔄 Back 🛃 Continue 🗶 Cancel	

Figure 37. Web Dynpro wizard

In the wizard we select the RFC name to be used in the Web Dynpro Service Call

C Web Dynpro Wizard: Select Service

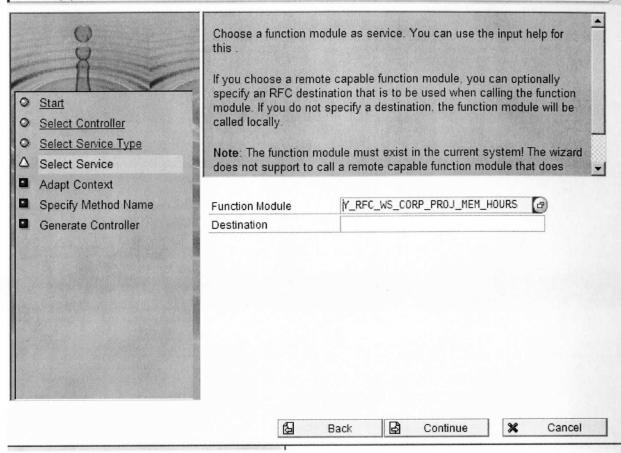


Figure 38.Web Dynpro Wizard: Select Service

Then we map controller's context with view's context as shown below.

×

Y_WDA_PROJ_MEMB_HOURS.COMPONENTCONTF
Context COMPONENTCONTROLLER
CONTEXT
▶ D Y_RFC_WS_CORP_PROJ_M
22

Figure 39. Mapping controller context with view context

Now we are designing the end user view where user can input the parameters PROJECT\_ID and WEEK\_ENDDATE and get the information from the SAP database.

iew	MAIN	Active	· · · · · · · · · · · · · · · · · · ·		
Properties	Layoul Inbound Plug	gs Outbound Plugs	Context Attributes Actions 1	Methods	
Favorites text action selection complex layout graphic integration	Week Ending at: MAIN.Y	_RFC_WS_CORP_	el , maintain & Upload	P ☐ TRANSF	ARENTCONTAINER PARENT_CONTAINER RESULTTABLE
			•		
				Property	Value
				Property enabled	
				Company and a second	Value
				enabled	Value
				enabled explanation	Value
				enabled explanation hotkey	Value
				enabled explanation hotkey imageFirst	Value
				enabled explanation hotkey imageFirst imageSource	Value
				enabled explanation hotkey imageFirst imageSource text	Value
				enabled explanation hotkey imageFirst imageSource text textDirection	Value
				enabled explanation hotkey imageFirst imageSource text textDirection tooltip	Value
				enabled explanation hotkey imageFirst imageSource text textDirection tooltip visible	Value

### **Download to Excel Sheet**

To download the results to excel sheet we need to create context variables in the web dynpro object which can map to the user defined input and the variables which can map to the excel sheet columns which needs to be generated. Figure below shows the IMPORTING variables as PROJ\_ID and WEEK\_ENDDATE which map to the user input. The CHANGING variables which are defined in the figure below map to the columns in the excel sheet to be generated

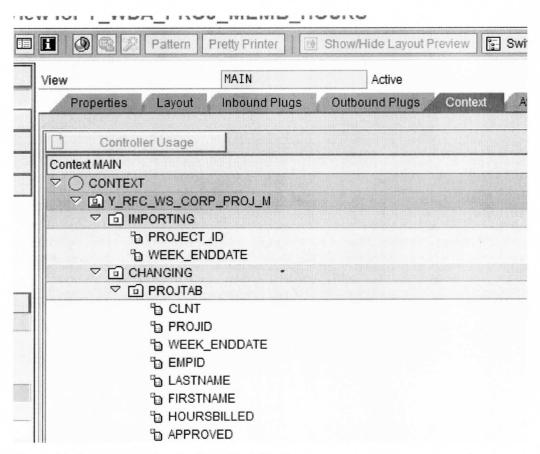


Figure 41. Context mapping for Download Excel

To download the results into excel sheet after the end user enters the input parameters PROJ\_ID and WEEK\_END\_DATE we have created the button **Download Excel** at the end user view layout. This button is linked to action method Export\_To\_Excel of the Web Dynpro object Y\_WDA\_PROJ\_MEMB\_HOURS as shown in Figure 39

The method Export\_To\_Excel is shown below:-

∕iew ♪	IAIN		Active	9			
Properties Layout Ir	nbound Plugs	Outbo	und Pl	ugs Conte	ext Attributes	Actions	Methods
🖕 Method List 🔏	Method						
Landress and the second s	PORT_TO_EXCE		<u> </u>	1			
Parameter	Туре	RefTo	Opt	Associated T	уре		Sh
WDEVENT	Importing			CL_WD_CUS	STOM_EVENT		
						*****	and the second second second second
		ALCONTROL .		1000110301007		*****	
method ONACTIONEXPORT_TO	Linning						
Lastronamolicano annucleano annuel lastronamolicano de lastronamonia de lastronamonias	Linning						
hannonedencommentations and homeneedence and homeneedence	Linning						
method ONACTIONEXPORT_TO	Linning						
method ONACTIONEXPORT_TO wd_this->zsubmit(	Linning						
<pre>wd_this-&gt;zsubmit(     ).</pre>	_EXCEL .	wd_con1	text_	node.			
<pre>method ONACTIONEXPORT_TO wd_this-&gt;zsubmit( ). DATA lo_nd_projtab TY</pre>	_EXCEL . PE REF TO if						
<pre>method ONACTIONEXPORT_TO wd_this-&gt;zsubmit( ). DATA lo_nd_projtab TY DATA lt_projtab TYPE</pre>	_EXCEL . PE REF TO if. wd_this->Ele	ments_p	rojtal	b.			
<pre>method ONACTIONEXPORT_TO wd_this-&gt;zsubmit( ). DATA lo_nd_projtab TY</pre>	PE REF TO if, wd_this->Ele	ments_p AB> via	rojtal lead	b. selection	WS_CORP_PROJ_		0JTAB'
<pre>method ONACTIONEXPORT_TO wd_this-&gt;zsubmit( ). DATA lo_nd_projtab TY DATA lt_projtab TYPE * pavingte from <contex< pre=""></contex<></pre>	_EXCEL . PE REF TO if, wd_this->Ele T> to <projt. ntext-&gt;path_ tant child</projt. 	ments_p AB> via	rojtal lead	b. selection	_ws_corp_proj_		OJTAB'

Figure 42. Method Export\_To\_Excel

Below is the code of method Export\_To\_Excel. This method basically first calls the method Z\_SUBMIT to retrieve the values from the database depending upon user input by

calling RFC Y\_RFC\_PROJ\_MEM\_HOURS. Then it converts the data into the excel format to show it to the end user.

```
method ONACTIONEXPORT_TO_EXCEL .
```

\* Calling the execute method of the rfc ( Code of this custom execute method showed in screen below ) -

```
wd_this->zsubmit().
```

st Below code to convert the context ( output of the rfc ) to excel -

DATA lo nd projtab TYPE REF TO if\_wd\_context\_node.

DATA lt projtab TYPE wd\_this->Elements\_projtab.

\* navigate from <CONTEXT> to <PROJTAB> via lead selection lo\_nd\_projtab = wd\_context->path\_get\_node( path =

Y RFC WS CORP PROJ\_M.CHANGING.PROJTAB` ).

- @TODO handle non existant child
- \* IF lo nd\_projtab IS INITIAL.
- \* ENDIF.

lo\_nd\_projtab->get\_static\_attributes\_table( importing table = lt\_projtab ).

DATA LW\_PROJTAB TYPE ytabproj. data str type string. data xstr type xstring.

concatenate str \*
 'PROJECT ID'
 'WEEK ENDING DATE'
 'EMPLOYEE ID'
 'LAST NAME'
 'FIRST NAME'
 'HOURS'
 'APPROVED'
 cl\_abap\_char\_utilities=>newline into str
 separated by cl\_abap\_char\_utilities=>horizontal\_tab.

Loop at lt\_projtab into lw\_projtab.

concatenate str LW\_PROJTAB-PROJID LW\_PROJTAB-WEEK\_ENDDATE LW\_PROJTAB-EMPID LW\_PROJTAB-LASTNAME

```
LW_PROJTAB-FIRSTNAME
LW_PROJTAB-HOURSBILLED
LW_PROJTAB-APPROVED
cl_abap_char_utilities=>newline into str
separated by cl_abap_char_utilities=>horizontal_tab.
```

endloop.

CALL FUNCTION 'SCMS_STRING_TO_XSTRING'
EXPORTING
text = str
* MIMETYPE = ' '
* ENCODING =
IMPORTING
BUFFER = xstr
EXCEPTIONS
FAILED = 1.
and we have a set of the second set of the secon
CALL METHOD cl_wd_runtime_services=>attach_file_to_response
EXPORTING
i_filename = 'Hours.xls'
i_content = xstr
i_mime_type = 'EXCEL'
i in new_window = ABAP_FALSE

i\_inplace = ABAP\_FALSE.

endmethod.

The method Z\_SUBMIT is called by the method above to execute the RFC Y\_RFC\_PROJ\_MEM\_HOURS to get the values depending upon the input values from the user

View MAIN Active Properties Layout Inbound Plugs Outbound Plugs Context Attributes Actions Method List Attributes Actions Method ZSUBMIT Parameter Type RefTo Opt Associated Type	
Method List   Image: Antipolation range     Method     Image: Antipolation range     Method     Image: Antipolation range     Method     Image: Antipolation range     Image: A	
Method ZSUBMIT  Parameter  Type RefTo Opt Associated Type  Image: Second	Methods
Parameter Type RefTo Opt Associated Type	
Parameter     Type     RefTo     Opt     Associated Type	
	Short
	and in these
*** CLEAR THE TABLE FIRST	
DATA lo_nd_projtab TYPE REF TO if_wd_context_node.	
DATA lt_projtab TYPE wd_this->Elements_projtab.	
DATA LT_CLEAR_PROJTAB TYPE wd_this->Elements_projtab.	
<pre>* navigate from <context> to <projtab> via lead selection lo_nd_projtab = wd_context-&gt;path_get_node( path = `Y_RFC_WS_CORP_PROJ_M.CHANGING.PRO</projtab></context></pre>	JTAB').
<ul> <li>@TODO handle non existant child</li> <li>^ IF lo_nd_projtab IS INITIAL.</li> <li>* ENDIF.</li> </ul>	

Below is the code shown for Z\_SUBMIT:-

method ZSUBMIT .

\*\*\* CLEAR THE TABLE FIRST

DATA lo\_nd\_projtab TYPE REF TO if\_wd\_context\_node.

DATA lt\_projtab TYPE wd\_this->Elements\_projtab. DATA LT\_CLEAR\_PROJTAB TYPE wd\_this->Elements\_projtab.

\* navigate from <CONTEXT> to <PROJTAB> via lead selection

43

Figure 43. Method Z\_SUBMIT

```
lo_nd_projtab = wd_context->path_get_node( path =
Y RFC WS CORP PROJ M.CHANGING.PROJTAB ).
       * @TODO handle non existant child
       * IF lo_nd_projtab IS INITIAL.
       * ENDIF.
       ** @TODO compute values
       ** e.g. call a model function
       *
       ** lo_nd_projtab->bind_table( new_items = lt_projtab set_initial_elements =
abap_true ).
       lo_nd_projtab->bind_table( LT_CLEAR_PROJTAB ).
       *** EXECUTE THE RFC
       DATA 10_COMPONENTCONTROLLER TYPE REF TO IG_COMPONENTCONTROLLER .
       lo_COMPONENTCONTROLLER = wd_this->get_componentcontroller_ctr( ).
         lo_componentcontroller->execute_y_rfc_ws_corp_proj_mem(
         ).
```

endmethod.

#### Upload Excel Sheet into SAP database

Figure below shows the UI Element we have created to upload the excel file and update the data into SAP

#### database.

1.120 <sup>-</sup> ШЕШЕ <sup>-</sup> ПООКО		
ttern   Pretty Printer   🕑 Show/Hide Layout Preview 🕞 Switch Context Editor View		
MAIN Active		
rout Inbound Plugs Outbound Plugs Context Attributes Actions Me	thods	
EMPLOYEE HOURS APPROVAL REPORT Project ID MAIN.Y_RFC_WS_CORP_ Week Ending at: MAIN.Y_RFC_WS_CORP_ Resource Management - Please Download Excel , maintain & Upload DOWNLOAD EXCEL BrowseUPLOAD EXCEL		TCONTAINER NT_CONTAINER JLTTABLE 1 2 ON_3 [Header] DN_ROW
	Property	Value
	Properties (FileUpload)	
	ID	FILE_UPLOAD
	activateAccessKey	
	contextMenuBehaviour	Inherit

Figure 44. UPLOAD EXCEL UI Element

To integrate this UI element with the web dynpro object we need to define a context (variable) named DATASOURCE which can hold the excel sheet as a buffer string.

.

/iew MAIN	Active			
Properties Layout Inbound Plugs Outb	ound Plugs	Context	Attributes	Actions
Controller Usage				
Context MAIN				≪GY_WDA_
				Context CO
▷ 🖾 Y_RFC_WS_CORP_PROJ_M				
D DATASOURCE				
				0 5 8 9
				8
Property		Va	lue	
Property Attribute		Va	lue	
			lue TASOURCE	
Attribute			TASOURCE	
Attribute Attribute Name		DA Tyj	TASOURCE	
<u>Attribute</u> Attribute Name Type assignment		DA Tyj	TASOURCE	
<u>Attribute</u> Attribute Name Type assignment Type		DA Tyj	TASOURCE	
Attribute Attribute Name Type assignment Type Read-only		DA Tyj	TASOURCE	
Attribute Attribute Name Type assignment Type Read-only Default Value		DA Tyj ST	TASOURCE	
Attribute Attribute Name Type assignment Type Read-only Default Value Null Value		DA Tyj ST	TASOURCE be RING	
Attribute Attribute Name Type assignment Type Read-only Default Value Null Value Input Help Mode		DA Tyj ST	TASOURCE be RING	
Attribute Attribute Name Type assignment Type Read-only Default Value Null Value Input Help Mode Determined Input Help		DA Tyj ST	TASOURCE be RING	

# Figure 45. Buffer string to hold uploaded excel file

To browse the excel file on the users system we create a File Upload element as shown below

♦ FILE_U ■ UPLO/	JPLOAD AD_EXCEL	
Property	Value	Binding
Properties (FileUpload)		
ID	FILE_UPLOAD	
activateAccessKey		
contextMenuBehaviour	Inherit 🔳	
contextMenuId		
data	MAIN.DATASOURCE	Ø
enabled	V	
explanation		

Figure 46. File upload element to browse the file on the users system

The UI Element UPLOAD EXCEL is linked to the method UPLOAD\_EXCEL on web dynpro object Y\_WDA\_PROJ\_MEMB\_HOURS

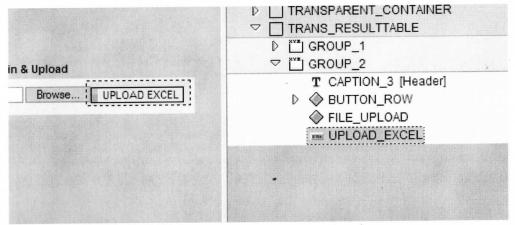


Figure 47. UI Element UPLOAD EXCEL linked to method UPLOAD\_EXCEL

The method UPLOAD EXCEL is shown below

View MA	IN		Active		
Properties Layout Int	ound Plugs	Outbo	und Pl	ugs Context Attributes Actio	ons Methods
Method List	Method 0AD_EXCEL			•	
Parameter	Туре	RefTo	Opt	Associated Type	Short Description
WDEVENT	Importing			CL_WD_CUSTOM_EVENT	
method ONACTIONUPLOAD_EXC	•• 2 3 3 1				
TYPES : BEGIN OF str_itab, CLNT(3) TYPE C, PROJID(10) TYPE C, WEEK_ENDDATE(8) TYI EMPID(8) TYPE c,					
LASTNÄMĖ(10) TYPE FIRSTNAME(10) TYPE HOURSBILLĖD(2) TYPI APPROVED(1) TYPE C END OF str_itab. DATA : t_table1 TYPE STAN i data TYPE STAN	C, E C, , ANDARD TABL			5,	

48

Figure 48. Method UPLOAD\_EXCEL

The code of UPLOAD EXCEL is below

method ONACTIONUPLOAD\_EXCEL .

TYPES :

BEGIN OF str\_itab, CLNT(3) TYPE C, PROJID(10) TYPE C, WEEK\_ENDDATE(8) TYPE C, EMPID(8) TYPE c, LASTNAME(10) TYPE c, FIRSTNAME(10) TYPE C, HOURSBILLED(2) TYPE C, APPROVED(1) TYPE C, END OF str\_itab.

DATA : t\_table1 TYPE STANDARD TABLE OF str\_itab, i\_data TYPE STANDARD TABLE OF string, lo\_nd\_data TYPE REF TO if\_wd\_context\_node, lo\_el\_data TYPE REF TO if\_wd\_context\_element, l\_string TYPE string, fs\_table TYPE str\_itab, l\_xstring TYPE xstring,

fields TYPE string\_table, lv field TYPE string. \*\* DATA : t table TYPE if\_main=>elements\_PROJTAB, \*\* data table TYPE if\_main=>elements\_PROJTAB. DATA lo\_nd\_projtab TYPE REF TO if\_wd\_context\_node. DATA lo\_el\_projtab TYPE REF TO if\_wd\_context\_node. DATA T\_TABLE TYPE wd\_this->Elements\_projtab. DATA DATA TABLE TYPE wd this->Elements\_projtab. DATA lt\_projtabprev TYPE wd\_this->Elements\_projtab. \*\*\*\*\* navigate from <CONTEXT> to <PROJTAB> via lead selection lo nd projtab = wd\_context->path\_get\_node( path = Y\_RFC\_WS\_CORP\_PROJ\_M.CHANGING.PROJTAB` ). \*\*\*\* \*\*\*\*\* @TODO handle non existant child \*\*\*\*\* IF lo\_nd\_projtab IS INITIAL. \*\*\*\*\* ENDIF. \*\*\*\* \*\*\*\*\*\* CLEAR THE CURRENT TABLE. \*\*\*\* \*\*\*\* lo\_nd\_projtab->get\_static\_attributes\_table( importing table = lt\_projtabprev ). \*\*\*\* \*\*\*\*DATA: WA\_TEMP TYPE YTABPROJ. \*\*\*\* \*\*\*\* LOOP AT 1t projtabprev INTO wa\_temp. lo\_nd\_projtab->remove\_element( EXPORTING element = lo\_nd\_projtab ). \*\*\* \*\*\*\* ENDLOOP. \* get single attribute wd\_context->get\_attribute( EXPORTING name = `DATASOURCE` IMPORTING value = 1\_xstring ). CALL FUNCTION 'HR KR\_XSTRING\_TO\_STRING' EXPORTING in\_xstring = l\_xstring IMPORTING out\_string = l\_string. SPLIT l\_string AT cl\_abap\_char\_utilities=>newline INTO TABLE i\_data. \* Bind With table Element. LOOP AT i\_data INTO l\_string. SPLIT l\_string AT cl\_abap\_char\_utilities=>horizontal\_tab INTO TABLE fields.

```
READ TABLE fields INTO lv_field INDEX 1.
      *
           fs table-CLNT = lv_field.
            READ TABLE fields INTO lv_field INDEX 1.
         IF LV_FIELD NE 'PROJECT ID'.
          fs table-PROJID = lv_field.
            READ TABLE fields INTO lv_field INDEX 2.
          fs_table-WEEK_ENDDATE = lv_field.
            READ TABLE fields INTO lv_field INDEX 3.
          fs table-EMPID = lv_field.
            READ TABLE fields INTO lv_field INDEX 4.
          fs table-LASTNAME = lv_field.
            READ TABLE fields INTO lv_field INDEX 5.
           fs table-FIRSTNAME = lv_field.
            READ TABLE fields INTO lv_field INDEX 6.
           fs table-HOURSBILLED = lv_field.
            READ TABLE fields INTO lv_field INDEX 7.
          fs_table-APPROVED = lv_field.
          APPEND fs_table TO t_table1.
          ENDIF.
         ENDLOOP.
       **lo nd data = wd_context->get_child_node( 'PROJTAB' ).
       **lo nd data->bind_table( T_TABLE1 ).
       CLEAR DATA_TABLE.
       lo_nd_projtab->bind_table( DATA_TABLE ).
        lo nd_projtab->bind_table( new_items = T_TABLE1 set_initial_elements = abap_false
).
       ***
            DATA lo_nd_projtab TYPE REF TO if_wd_context_node.
       ***
       ***
            DATA lt_projtab TYPE wd_this->Elements_projtab.
       ***
       **** navigate from <CONTEXT> to <PROJTAB> via lead selection
       ***
            lo_nd_projtab = wd_context->path_get_node( path =
Y_RFC_WS_CORP_PROJ_M.CHANGING.PROJTAB`).
       ***
       **** @TODO handle non existant child
       **** IF lo_nd_projtab IS INITIAL.
       **** ENDIF.
       ***
            lo_nd_projtab->get_static_attributes_table( importing table = lt_projtab ).
       ***
       ***
       ***
       ***
```

```
***
           DATA lo_nd_projtab TYPE REF TO if_wd_context_node.
       ***
       ***
           DATA lt_projtab TYPE wd_this->Elements_projtab.
       ***
       **** navigate from <CONTEXT> to <PROJTAB> via lead selection
       *** lo_nd_projtab = wd_context->path_get_node( path =
Y_RFC_WS_CORP_PROJ_M.CHANGING.PROJTAB` ).
       ***
       **** @TODO handle non existant child
       **** IF lo_nd_projtab IS INITIAL.
       **** ENDIF.
       ***
       ***** @TODO compute values
       ***** e.g. call a model function
       ****
       *** lo_nd_projtab->bind_table( new_items = lt_projtab set_initial_elements =
abap_true ).
       ***
```

```
wd_this->save_db( ).
```

endmethod.

The web dynpro context PROJTAB where the excel sheet is uploaded is shown below

Propetties Layout Inbound Plug	Active	
Properties / Lavour Inbodisc Prog	s Outbound Plags Context Attribu	utes Actions Methods
EN ANTIN L		
Controller Usage		
Context MAIN		SY_WDA_PROJ_MEMB_HOURS.COMPONENTCONTROLLE
C CONTEXT		Context COMPONENTCONTROLLER
Y_RFC_WS_CORP_PROJ_M		
TI IMPORTING		D D Y_RFC_WS_CORP_PROJ_M
"D PROJECT_ID		
B WEEK_ENDDATE		
C CHANGING		
PROJTAB		
D CLNT		
D WEEK_ENDDATE		
B EMPID		
D LASTNAME		
D FIRSTNAME		
B HOURSBILLED		
D APPROVED		
D DATASOURCE		
Property	Value	ndx
Nodes		
Node Name	PROJTAB	
Dictionary structure	YTABPROJ	
Cardinality	0.n	
Selection	0.1	
Initialization Lead Selection		R
Singleton		
Supply Function	the second s	1

Figure 49. Upload Excel context in web dynpro object

îew	MAIN	A	ctive			
Properties Layout	Inbound Plugs	Outboun	d Plugs Context	Attributes	Actions	Methods
List Method List	66 Method	<b></b>	1			
Method SAVE_	L <del></del>					
8 7 M 10						
Parameter	Type	RefTo (	Opt Associated Type	¥		Short Desc
	••					
Method SAVE_DB .						
and a second second second second		wd_context				
method SAVE_DB .	b TYPE REF TO If_					
method SAVE_DB . DATA lo_nd_projtal	b TYPE REF TO if_ YPE wd_this->Elem	ents_projt R> via les	tab.	CORP_PR0J_M.		PROJTAB" ).

The context table PROJTAB is saved to the database table by the method SAVE\_DB.

Figure 50. Method SAVE\_DB

The code for the method SAVE\_DB is as below:-

```
method SAVE_DB .
DATA lo_nd_projtab TYPE REF TO if_wd_context_node.
DATA lt_projtab TYPE wd_this->Elements_projtab.
* navigate from <CONTEXT> to <PROJTAB> via lead selection
lo_nd_projtab = wd_context->path_get_node( path = `Y_RFC_WS_CORP_PROJ_M.CHANGING.PROJTAB`
).
* @TODO handle non existant child
* IF lo_nd_projtab IS INITIAL.
* ENDIF.
lo_nd_projtab->get_static_attributes_table( importing table = lt_projtab ).
DATA: WT_TABLE1 TYPE YTABPROJ.
LOOP AT lt_projtab INTO WT_TABLE1.
MODIFY YTABPROJ FROM WT_TABLE1.
```

ENDLOOP.

Endmethod

### Web Service

After we have created the methods to import data into excel and update data from excel into SAP database we need to create a web service from the ABAP web dynpro object which acts an interface to the end user and the SAP application. For that first we need to create web dynpro application from abap web dynpro object/component.

Web Duppro Explor					
web Dyribio Explore	er: Change View	for Y_W	DA_PR	OJ_MEMB_HO	URS
	1 4 2 <b>2</b>		Pattern	Pretty Printer	Show/Hide Layout Preview
MIME Repository	V	iew		MAIN	Active
Repository Browser		Properties	Layout	Inbound Plugs	Outbound Plugs Context
Repository Information System		. repaired			e un come rage comos
₽Tag Browser					
Transport Organizer		Method	not an	Method Type Descrip	tion
Test Repository		ONACTIONS	UBMIT	1 Event F SUBMIT	
		WDDOAFTER	ACTION	0 Method 🗈 Method 1	for non-action specific operatio
Web Dynpro Comp. / Intf. 📳		WDDOBEFOF			
Y_WDA_PROJ_MEMB_HOURS	▼ &r	WDDOEXIT		0 Method 🛅 Controll	er Clean-Up Method
·		WDDOINIT		0 Method 🛅 Controll	er Initialization Method
		WDDOMODIF			for Modifying the View Before F
Object Name		WDDOONCON	TEXTMENU		for Modifying the Context Menu
▽ 点 Y_WDA_PROJ_MEMB_HO	Create	1	Web Dypp	Rethod B ro Component (Interface	<u>,</u>
ൺ COMPONENTCONTR ഉ എ Component Interface	Change			to component (intenace	
✓ ☐ Views			View		
MAIN	Display	*	Window		
🗢 🛅 Windows	Create/Change Configur	ation	Custom C	ontroller	
Y_WDA_PROJ_ME	Check	•	Service Ca	ll	
	Activate Copy		Mime Object		•
			Web Dynp	ro Application	
	Rename	Rename		0 Method 🗓	
	Delete	Delete		0 Method 🚺	
	Where-Used List			0 Method 🗈	
	Other Functions	•		0 Method 🛅	
	-			0 Method 🗃	

Figure 51.Creating Web dynpro application

Following figure shows that web dynpro application y\_wda\_proj\_mem\_hours is created. Then we need to activate the application to execute it.

8		8   21 L 21   🗷 🛛 🖓 📭
Maintain se	rvice	
Create Host/Service	ce 🎾 🗊 🖬 🔿 External Aliases 🔃 🕽	System Monitor Inactive
Filter Details		
Virtual Host	Service Path	
Service		
Description	EN English Ref.Service:	
Lang.		
Filter	Reset Detail	
8 <u>2</u> <u>1</u> <u>1</u> <u>1</u>		
irtuelle Hosts / Sen	vices	Documentation Referenz Serv
	ytest1234	test appl.
	ytest_01	Test Application
	@ ytest_3303	testing
	ytest_adobe	Test
	ytest_dyn_atr	Web Dynpro Application ytest_dyn_atr / Co
	ytest_excel_upd	Test Application
	ytest_sm30	Test Application
	ytest_sm30_kb	dasdasda
	ytest_sm30_utility	SM30 Utility Application
	ytest_std_screen	Test Application
	ytest_tablock	Web Dynpro Application ytest_tablock / Co
		Get details of flight
		Web Dynpro Application y_bapi_get_sales I
	😡 y_bapi_xml_download	Using XML Download to XLS
	y_wda_proj_memb_hours	y_wda_proj_memb_hours
		view container example
	zmwdgt3355_pay_inq2	PAYMENT INQUIRY
	zmwdgt3356_vendor_inquiry	Vendor Tree with Bank and Address Details
	zmwdgt3356_vend_bank_addr	View of Vendor bank and address details
	zmwdgt3357_email_notify	Email notification
	zvwdgt3355_pay_inq	Payment_inquiry
	ZVWDGT3355_Pay_Inq_app	Payment Inquiry App
	(🖗) zvwdgt3356_email	Send emails

Figure 52. Activating web dynpro application

After activating the application we can execute the application as shown below

Web Dynpro Application Edit Goto	Utilities Envir	onment System H	Help		
	the second s		Financial strength in the strength in the strength of the stre	x 2 0 G	
Web Dynpro Explorer: Ch	ange We	b Dynpro Ap	plication		
← → ♡ 6 6 4 4 5 5 5 6 4 5 5 5 5 5 5 5 5 5 5 5					
MIME Repository		Application	y_wda_proj_m	emb hours	Saved
Repository Browser		Properties	Parameters		Guica
बिद्धRepository Information System					
₽Tag Browser		Description	y_wda_proj_i	memb_hours	
🖶 Transport Organizer		Component	Y_WDA_PROJ	_MEMB_HOURS	
Test Repository		Interface View	Y_WDA_PROJ	_MEMB_HOUR	
		Plug Name	DEFAULT		
Web Dynpro Comp. / Intf. 📓		Help Menu Text			
Y_WDA_PROJ_MEMB_HOURS 🔻 🗞		Help Link			
		Handling of Messa	ages	and a street of the state	
Inconstant for the second seco		Show Message	e Component o	n Demand	
Object Name ▽ 슯 Y_WDA_PROJ_MEMB_HOURS		O Always Display	Message Com	ponent	
COMPONENTCONTROLLER		Laise			
D ch Component Interface		Administration Dat	ta /		
Views		Created By	AJAIN	Created on	04/09/2011
MAIN		Last changed by		Changed on	
Windows		Package	\$TMP		
Y_WDA_PROJ_MEMB_HOUF	<	Language	EN		
Veb Dynpio Applications D 23 y_wda_proj_memb_hours		URL	http://p2e	edeq.am.elcompanio	es.net:8041/sap/bc/web
	<u>C</u> reate C <u>h</u> ange Display		•		
	Create/Ch	ange Configuration			
	Check				
	Test				
	Delete				
	Display Ob	ject Directory Entry			
	Change Pa	ackage <u>A</u> ssignment sport Entry			

55

Figure 53. Executing web dynpro application

Executing web dynpro application takes us to the following screen

) Back - 🕗 - 💌 💈 🏠 🔎 Search 😪 Revortes 🥝 🖉		
EMPLOYEE HOURS APPROVAL REPORT Project ID		
Week Ending at		
Resource Management - Please Download Excel , maintain & Upload		CACING MALERIA
DOWNE OAD EXCEL	Browse	UPLOAD EXCEL

Figure 54. User View

To the users we can give the URL which is provided by the Web Dynpro application.

The users can use the above URL as long as they are in the companies secure vpn connection. But they don't need to have SAP installed on their systems. This URL acts as a web service which is independent of the SAP application but can be used from anywhere.

# **CHAPTER 4**

## RESULTS

This section contains the result table generated in the MS Excel after user specified input and also the SAP database which is updated upon upload of MS Excel sheet.

### **Download Excel**

First we will show how the excel sheet is generated on user specified input criteria. The users enters the PROJECT ID as PROJECTDSU.

Stack + Q - K 2 () Diserch Armanies () () + S () () () 4 3 Norther (Scholar an-economics net 304/ heads (webdynet heady , unds and , nemo , how share - impurger () 1 EMPLOYEE BOURS APPROVAL REPORT Project ID PROJECTOSU Resource Management - Please Download Facel, maintain & Upload DOWNELOAD ENCEL Browse. UPLOAD EXCEL

Figure 55. User entered PROJECT ID

The users then clicks DOWNLOAD EXCEL button which pops up the following screen

EMPLOYEE HOURS APPROVAL REPORT	
Project ID PROJECTOSU	
esource Management - Please Download Excet , maintain & Upload	
DOWNLOAD EXCEL	Contraction of the Contraction o
	File Download Do you want to open or save this file?
	Name: Hours.xls Type: Microsoft Office Excel 97-2003 Worksheet, 126 bytes From: p2edeq.am.ekcompanies.net
	Save Cancel

#### Figure 56. User hits DOWNLOAD EXCEL BUTTON

The user has the option of saving or opening up the excel sheet. The excel sheet generated is shown below.

.

	- · · · · · · · · · ·					HOUPS(3),305 - BAICH	SOTT EXCEL	
Ho	me Insert	Page Layout	Formulas Da	ata Review	View			
	Cut Copy	Calibri *	11 · [A' A']	= = = >	📑 Wrap Text	General *		Normal
Pacta	Copy Format Painter	<b>B</b> <i>I</i> <u>U</u> ∗ <u></u>				\$ - % , .00	Conditional Format Formatting * as Table *	Good
Clipb	oard 🖗		G	Ali	gnment <sup>13</sup>	Number <sup>1</sup> ×		Styles
A	2 🗸	( fe						
A		В	C	DE	F G	H	JK	L N
1	PROJECT	D	The second s		the second se	APPROVED		
2	PROJECTO	SU	20110424	1875943 SHIKHA	JAIN 75	Y		
3								
4 5								
5								
6 7								
7								
8 9								
10								
L0 L1	6	(41 x 391)						
2	U	MIX 321						
.2								
4								
.4 .5 .6 .7								
6								
.7								
.8 .9								
0								

Figure 57. Generated Excel Sheet

# **Upload Excel**

Now we will show how the user can update the SAP database table using excel sheet.

.

For that we first check what all values we have in SAP database table.

	<u>Settings Utilities Sy</u> stem <u>H</u> elp
SE16	
Data Browse	r: Initial Screen
Table Name	YTABPROJ

Figure 58. Browsing SAP database table

Figure below shows the values we have in SAP database table

0		1			80008 🗷		
Data	Browser	: Table Y	TABPR	OJ Select Er	ntries 7		
sr Q	AVB	5 5					
able: isplay	YT/ red Fields:	ABPROJ 8 of 8	Fixer	d Columns.	4 List	Uid+6 0350	
			1 1200		4 List	. WIUCH 0200	
CLNT	1	WEEK_ENDDATE			FIRSTNAME	HOURSBILLED	APPROVED

Now if the end user wants to update using excel sheet the APPROVED column of the YTABPROJ to N for PROJECT ID PROJECTDDSU

0-		) - (* - ) =							Hou	urs[3].xls - 1	Micr
	Hom	e Insert	Page Layout	Formulas	Data R	Review	View				
(A)	n y cu	rt	Calibri	- 11 - A A		= >-	- Br Wra	p Text	General		*
Past	te Co		<b>B</b> <i>I</i> <u>U</u> -	- A - A			📰 🔤 Mer	ge & Center	- \$ - %	· · · · · · · · · · · · · · · · · · ·	.00
Ŧ	Clipboa	rmat Painter	\	int (	3	Ali	ignment		Nui	mber	r <u>s</u>
	H2	-	(* fx	N							
7	А		8	С	D	E	F	G	Н	1	
1		PROJECT II	D	WEEK END	EMPLOYE	LAST NA	M FIRST NAM	HOURS	APPROVED	)	
2		PROJECTD	SU	20110424	1875943	SHIKHA	JAIN	75	N		
3											
4											
2 3 4 5 6											
6											

Figure 60. Changing APPROVED column in the excel sheet

Then the user can locate the excel sheet from his system using Browse button of the

web service

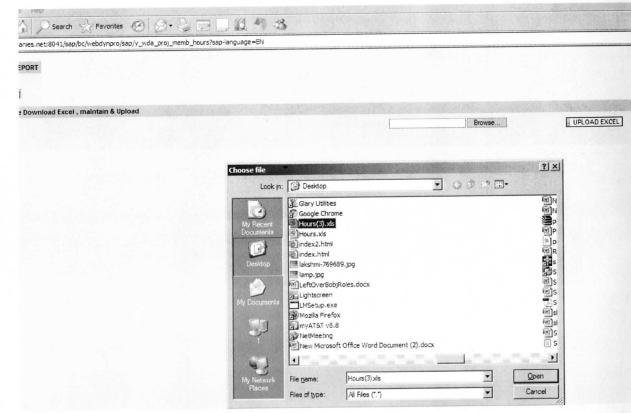


Figure 61. Browsing the excel file to be uploaded.

The user then clicks on UPLOAD EXCEL button to upload the excel file on the SAP database to update table YTABPROJ.

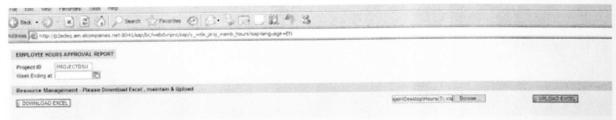


Figure 62. Users clicks UPL OAD EXCEL button

#### Figure 62. Users clicks UPLOAD EXCEL button

Now if we check the database table YTABPRO, it shows the updated entry for PROJID PROJECTDSU as APPROVED column to be N (was Y previously)

C	7		1			80008 🗷		
D	ata	Browser	: Table Y	TABPRO	OJ Select E	ntries 7		
50	r Q	AVB	B 8 B					
ak	ole:	YTA	BPROJ					
115	splay	ed Fields:		Fixed	i Columns:	4 L1	st Width 0250	
)15	-		8 of 8		LASTNAME	4 Li FIRSTNAME	st Width 0250 HOURSBILLED	APPROVED
	CLNT	PROJID	8 of 8 WEEK_ENDDATE		LASTNAME		HOURSBILLED	APPROVED
	-	PROJID	8 of 8 WEEK_ENDDATE 04/12/2011	EMPID	LASTNAME BOBBY	FIRSTNAME	HOURSBILLED 68 68	N N
	CLNT 600 600	PROJID	8 of 8 WEEK_ENDDATE 04/12/2011 04/12/2011	EMPID 10012999 10012999	LASTNAME BOBBY	FIRSTNAME MARIA MARIA JAIN	68 68 68 75	N N
	CLNT 600 600	PROJID PROJBCV736	8 of 8 WEEK_ENDDATE 04/12/2011 04/12/2011 04/24/2011	EMPID 10012999 10012999 1875943 10012999	LASTNAME BOBBY BOBBY SHIKHA KATHY	FIRSTNAME MARIA MARIA JAIN DRAPA	HOURSBILLED 68 68 75 36	N N N
	CLNT 600 600 600	PROJID PROJBCV736 PROJECTDSU	8 of 8 WEEK_ENDDATE 04/12/2011 04/12/2011 04/24/2011 04/12/2011	EMPID 10012999 10012999 1875943 10012999 88881234	LASTNAME BOBBY BOBBY SHIKHA KATHY JOHN	FIRSTNAME MARIA JAIN DRAPA KNIGHT	68 68 68 75 36 40	N N N N Y
	CLNT 600 600 600 600	PROJID PROJBCV736 PROJECTDSU PROJECTQC1	8 of 8 WEEK_ENDDATE 04/12/2011 04/12/2011 04/22/2011 04/12/2011 04/10/2011	EMPID 10012999 10012999 1875943 10012999	LASTNAME BOBBY SHIKHA KATHY JOHN SHIKHA	FIRSTNAME MARIA MARIA JAIN DRAPA	HOURSBILLED 68 68 75 36	N N N

Figure 63. Updated database table YTABPROJ

## **CHAPTER 5**

## CONCLUSIONS

- 1. SAP ERP helps to automate a company's business and also helps to enhance the information systems.
- 2. The HR department can save a lot of time and resources by integrating the SAP database with the Excel Sheet.
- 3. Excel Sheet provides a simple and easy way to make calculation on the data, make comparisons among different set of values etc. instead of doing it manually.
- 4. It is easy to send the information in excel sheet to the user who does not have access to the company's database.

There can be many future enhancements to this application like:-

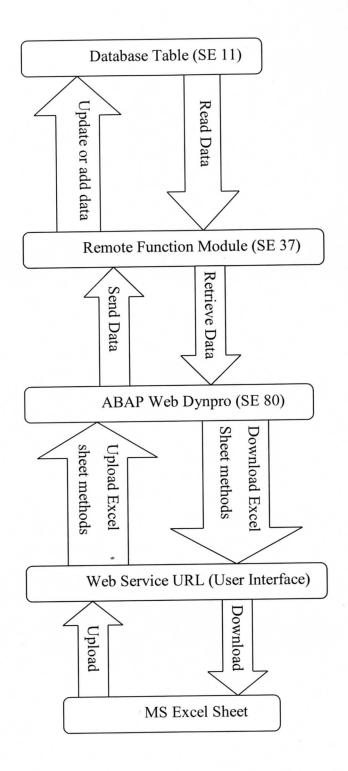
- The user can have the option of directly emailing the excel file generated just by specifying the email addresses instead of manually emailing it to each address
- Same architecture could be used in other set of business requirements like viewing the leave history of an employee, viewing and updating the inventory in excel format etc.

# REFERENCES

- 1. SAP Library. http://help.sap.com/
- 2. SAP Developers Network https://www.sdn.sap.com
- 3. http://www.ibm.com/developerworks/webservices/library/ws-wsilover/
- 4. http://en.wikipedia.org/wiki/SAP\_ERP
- 5. http://en.wikipedia.org/wiki/Remote Function Call
- 6. http://en.wikipedia.org/wiki/Visual\_Basic
- 7. <u>http://help.sap.com/saphelp\_nw04s/helpdata/en/77/3545415ea6f523e10000000a15510</u> 6/content.htm

# APPENDICES

## Appendix A: Technical architecture diagram



#### **Appendix B: SAP Code**

## Function Y\_RFC\_WS\_CORP\_PROJ\_MEM\_HOURS

FUNCTION Y\_RFC\_WS\_CORP\_PROJ\_MEM\_HOURS.

\*"\*"Local Interface:

\*" IMPORTING

\*" VALUE(PROJECT\_ID) TYPE CHAR10

- \*" VALUE(WEEK\_ENDDATE) TYPE DATUM OPTIONAL
- \*" EXPORTING
- \*" VALUE(PROJTAB) TYPE YTABTYP\_PROJ
- \*"\_\_\_\_\_

\*"\_\_\_\_\_

DATA: IT\_PROJTAB TYPE STANDARD TABLE OF YTABPROJ.

DATA: IT\_PROJTAB\_WEEK TYPE STANDARD TABLE OF YTABPROJ.

DATA: WA PROJTAB TYPE YTABPROJ.

CALL FUNCTION 'Y\_FM\_PROJ\_MEMBERS' EXPORTING PROJECT\_ID = PROJECT\_ID IMPORTING PROJECT\_TAB = IT\_PROJTAB.

\*\*SELECT \* FROM YTABPROJ APPENDING TABLE PROJTAB WHERE PROJID = PROJECT\_ID AND WEEK\_ENDDATE = WEEK\_ENDDATE.

IF WEEK\_ENDDATE IS NOT INITIAL.

LOOP AT IT\_PROJTAB INTO WA\_PROJTAB WHERE WEEK\_ENDDATE = WEEK\_ENDDATE.

\*\*READ \* FROM IT\_PROJTAB INTO WA\_PROJTAB WITH KEY WEEK\_ENDDATE = WEEK\_ENDDATE.

APPEND WA\_PROJTAB TO IT\_PROJTAB\_WEEK.

ENDLOOP.

ELSE. IT\_PROJTAB\_WEEK = IT\_PROJTAB. ENDIF.

PROJTAB = IT\_PROJTAB\_WEEK.

ENDFUNCTION.

#### **Download Excel Methods**

## ONACTIONEXPORT\_TO\_EXCEL

method ONACTIONEXPORT\_TO\_EXCEL .

\* Calling the execute method of the rfc ( Code of this custom execute method showed in screen below ) -

```
wd_this->zsubmit(
).
```

st Below code to convert the context ( output of the rfc ) to excel -

DATA lo\_nd\_projtab TYPE REF TO if\_wd\_context\_node.

DATA lt\_projtab TYPE wd\_this->Elements\_projtab.

\* navigate from <CONTEXT> to <PROJTAB> via lead selection lo\_nd\_projtab = wd\_context->path\_get\_node( path = uc comp\_proj\_m\_crance\_proj\_tab; )

Y\_RFC\_WS\_CORP\_PROJ\_M.CHANGING.PROJTAB` ).

- @TODO handle non existant child
- \* IF lo\_nd\_projtab IS INITIAL.
- \* ENDIF.

DATA LW\_PROJTAB TYPE ytabproj. data str type string. data xstr type xstring.

concatenate str 'PROJECT ID' 'WEEK ENDING DATE' 'EMPLOYEE ID' 'LAST NAME' 'FIRST NAME' 'HOURS' 'APPROVED' cl\_abap\_char\_utilities=>newline into str separated by cl\_abap\_char\_utilities=>horizontal\_tab.

Loop at lt\_projtab into lw\_projtab.

concatenate str LW\_PROJTAB-PROJID LW\_PROJTAB-WEEK\_ENDDATE LW\_PROJTAB-EMPID LW\_PROJTAB-LASTNAME LW\_PROJTAB-FIRSTNAME LW\_PROJTAB-HOURSBILLED LW\_PROJTAB-APPROVED cl\_abap\_char\_utilities=>newline into str separated by cl\_abap\_char\_utilities=>horizontal\_tab.

endloop.

CALL FUNCTION 'SCMS\_STRING\_TO\_XSTRING' EXPORTING = str text = ' ' MIMETYPE ENCODING = IMPORTING BUFFER = xstr EXCEPTIONS = 1. FAILED CALL METHOD cl\_wd\_runtime\_services=>attach\_file\_to\_response EXPORTING = 'Hours.xls' i filename = xstr i\_content i\_mime\_type = 'EXCEL' i\_in\_new\_window = ABAP\_FALSE i\_inplace = ABAP\_FALSE.

endmethod.

#### ZSUBMIT

method ZSUBMIT .

\*\*\* CLEAR THE TABLE FIRST

```
DATA lo_nd_projtab TYPE REF TO if_wd_context_node.
         DATA lt projtab TYPE wd_this->Elements_projtab.
          DATA LT CLEAR_PROJTAB TYPE wd_this->Elements_projtab.
       * navigate from <CONTEXT> to <PROJTAB> via lead selection
         lo nd projtab = wd_context->path_get_node( path =
Y_RFC_WS_CORP_PROJ_M.CHANGING.PROJTAB` ).
       * @TODO handle non existant child
       * IF lo nd projtab IS INITIAL.
       * ENDIF.
       ** @TODO compute values
       ** e.g. call a model function
       ** lo_nd_projtab->bind_table( new_items = lt_projtab set_initial_elements =
abap_true ).
       lo_nd_projtab->bind_table( LT_CLEAR_PROJTAB ).
       *** EXECUTE THE RFC
       DATA 10_COMPONENTCONTROLLER TYPE REF TO IG_COMPONENTCONTROLLER .
       lo_COMPONENTCONTROLLER = wd_this->get_componentcontroller_ctr( ).
         lo_componentcontroller->execute_y_rfc_ws_corp_proj_mem(
```

).

endmethod.

#### Upload excel sheet methods

#### ONACTIONUPLOAD\_EXCEL

method ONACTIONUPLOAD\_EXCEL .

```
TYPES :
              BEGIN OF str_itab,
              CLNT(3) TYPE C,
              PROJID(10) TYPE C,
              WEEK ENDDATE(8) TYPE C,
              EMPID(8) TYPE c,
              LASTNAME(10) TYPE c,
              FIRSTNAME(10) TYPE C,
              HOURSBILLED(2) TYPE C,
              APPROVED(1) TYPE C,
              END OF str_itab.
         DATA : t_table1 TYPE STANDARD TABLE OF str_itab,
                i_data TYPE STANDARD TABLE OF string,
                lo_nd_data TYPE REF TO if_wd_context_node,
                lo_el_data TYPE REF TO if_wd_context_element,
                l_string TYPE string,
                fs_table TYPE str_itab,
                l_xstring TYPE xstring,
                fields TYPE string table,
                lv_field TYPE string.
       **
           DATA : t table TYPE if_main=>elements_PROJTAB,
       **
                  data table TYPE if_main=>elements_PROJTAB.
         DATA lo nd projtab TYPE REF TO if_wd_context_node.
         DATA lo el projtab TYPE REF TO if_wd_context_node.
         DATA T_TABLE TYPE wd_this->Elements_projtab.
         DATA DATA TABLE TYPE wd_this->Elements_projtab.
         DATA lt_projtabprev TYPE wd_this->Elements_projtab.
       ***** navigate from <CONTEXT> to <PROJTAB> via lead selection
          lo nd projtab = wd_context->path_get_node( path =
Y_RFC_WS_CORP_PROJ_M.CHANGING.PROJTAB` ).
       ****
       ***** @TODO handle non existant child
       ***** IF lo_nd_projtab IS INITIAL.
       ***** ENDIF.
       ****
       ****** CLEAR THE CURRENT TABLE.
       ****
       **** lo_nd_projtab->get_static_attributes_table( importing table = lt_projtabprev ).
       ****
       ****DATA: WA_TEMP TYPE YTABPROJ.
       ****
```

```
LOOP AT lt_projtabprev INTO wa_temp.
****
       lo nd projtab->remove_element( EXPORTING element = lo_nd_projtab ).
***
**** ENDLOOP.
* get single attribute
  wd context->get attribute(
    EXPORTING
      name = `DATASOURCE`
    IMPORTING
      value = 1 xstring ).
  CALL FUNCTION 'HR_KR_XSTRING_TO_STRING'
    EXPORTING
      in_xstring = l_xstring
    IMPORTING
      out string = 1_string.
  SPLIT l_string AT cl_abap_char_utilities=>newline INTO TABLE i_data.
* Bind With table Element.
  LOOP AT i_data INTO l_string.
    SPLIT l_string AT cl_abap_char_utilities=>horizontal_tab INTO TABLE fields.
     READ TABLE fields INTO lv_field INDEX 1.
*
*
     fs table-CLNT = lv_field.
      READ TABLE fields INTO lv_field INDEX 1.
   IF LV_FIELD NE 'PROJECT ID'.
    fs_table-PROJID = lv_field.
      READ TABLE fields INTO lv_field INDEX 2.
    fs_table-WEEK_ENDDATE = lv_field.
      READ TABLE fields INTO lv_field INDEX 3.
    fs_table-EMPID = lv_field.
      READ TABLE fields INTO lv_field INDEX 4.
    fs_table-LASTNAME = lv_field.
      READ TABLE fields INTO lv_field INDEX 5.
    fs_table-FIRSTNAME = lv_field.
      READ TABLE fields INTO lv_field INDEX 6.
    fs_table-HOURSBILLED = lv_field.
      READ TABLE fields INTO lv_field INDEX 7.
    fs table-APPROVED = lv_field.
    APPEND fs_table TO t_table1.
   ENDIF:
  ENDLOOP.
**lo_nd_data = wd_context->get_child_node( 'PROJTAB' ).
**lo_nd_data->bind_table( T_TABLE1 ).
```

CLEAR DATA\_TABLE.

lo\_nd\_projtab->bind\_table( DATA\_TABLE ).

```
lo_nd_projtab->bind_table( new_items = T_TABLE1 set_initial_elements = abap_false
```

).

```
DATA lo_nd_projtab TYPE REF TO if_wd_context_node.
      ***
      ***
           DATA lt_projtab TYPE wd_this->Elements_projtab.
      ***
      ***
      **** navigate from <CONTEXT> to <PROJTAB> via lead selection
      *** lo_nd_projtab = wd_context->path_get_node( path =
Y_RFC_WS_CORP_PROJ_M.CHANGING.PROJTAB` ).
      ***
      **** @TODO handle non existant child
      **** IF lo_nd_projtab IS INITIAL.
       **** ENDIF.
       ***
           lo_nd_projtab->get_static_attributes_table( importing table = lt_projtab ).
       ***
       ***
       ***
       ***
       **_____
           DATA lo_nd_projtab TYPE REF TO if_wd_context_node.
       ***
       ***
           DATA lt_projtab TYPE wd_this->Elements_projtab.
       ***
       ***
       **** navigate from <CONTEXT> to <PROJTAB> via lead selection
       *** lo_nd_projtab = wd_context->path_get_node( path =
Y_RFC_WS_CORP_PROJ_M.CHANGING.PROJTAB` ).
       ***
       **** @TODO handle non existant child
       **** IF lo_nd_projtab IS INITIAL.
       **** ENDIF.
       ***
       ***** @TODO compute values
       ***** e.g. call a model function
       ****
       *** lo_nd_projtab->bind_table( new_items = lt_projtab set_initial_elements =
abap_true ).
       ***
         wd_this->save_db( ).
```

endmethod.

#### SAVE\_DB

method SAVE\_DB .

DATA lo\_nd\_projtab TYPE REF TO if\_wd\_context\_node.

DATA lt projtab TYPE wd\_this->Elements\_projtab.

\* navigate from <CONTEXT> to <PROJTAB> via lead selection lo\_nd\_projtab = wd\_context->path\_get\_node( path = `Y\_RFC\_WS\_CORP\_PROJ\_M.CHANGING.PROJTAB` ).

\* @TODO handle non existant child

\* IF lo\_nd\_projtab IS INITIAL.

\* ENDIF.

lo\_nd\_projtab->get\_static\_attributes\_table( importing table = lt\_projtab ).

DATA: WT\_TABLE1 TYPE YTABPROJ.

LOOP AT lt\_projtab INTO WT\_TABLE1.

MODIFY YTABPROJ FROM WT\_TABLE1.

ENDLOOP.

Endmethod

