Project Report

On

DEPARTMENT COMMUNICATION SYSTEM USING SAP

Submitted for

partial fulfillment of the degree in

Bachelor of Engineering

(Computer Science & Engineering)

Sixth Semester

By

Pritam R.Sah

Saurabh R.Nampalliwar

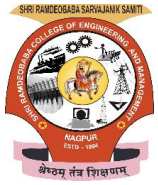
Sambhav S.Chopda

Abhishek R.Parmar

Anurag R.Tiwari

Under the Guidance of

Prof. Ruchika Sinhal



Department of Computer Science & Engineering

Shri Ramdeobaba College Of Engineering & Management

(An Autonomous College of Rashtrasant Tukadoji Maharaj Nagpur University)

2013-2014

CERTIFICATE

This is to certify that the Project Report on

"Department Communication System using SAP "

is a bonafide work and it is submitted

for

For partial fulfillment of the degree in

Bachelor of Engineering in Computer Science & Engineering

Sixth Semester

By

Pritam R.Sah

Saurabh R.Nampalliwar

Sambhav S.Chopda

Abhishek R.Parmar

Anurag R.Tiwari

during the academic year 2013-2014

under the guidance of

Prof. Ruchika Sinhal
Guide

Dr. M.B.Chandak
Head, Computer Science & Engg

Dr. V. S. Deshpande
Principal

Department of Computer Science & Engineering

Shri Ramdeobaba College Of Engineering & Management

(An Autonomous College of Rashtrasant Tukadoji Maharaj Nagpur University)

2013-2014

ACKNOWLEDGEMENT

We sincerely express our gratitude to our guide **Prof.Ruchika Sinhal** for her valuable guidance and kind suggestion. We are indebted her for her untiring devotion and willingness to go with us to any length .She has shown a stimulating interest in this project and has been encouraging us every time.

We sincerely express our gratitude to **Dr.V.S.Deshpande,** Principal, SRCOEM, Nagpur and **Dr.M.Chandak**, Head Dept. of CSE, SRCOEM for providing us this opportunity and support.

We are greatly thankful to **Rolling Arrays, Singapore** for giving us the access to the SAP server. We are also thankful to **Mr.Ravindra Sahu** of Rolling Arrays, Singapore without whose help and guidance we would not be able complete the project.

We are sincerely thankful to our faculty teachers and the Department of Computer Science and Engineering for their valuable suggestion, helping hand and guidance extended to us.

The acknowledgement would be incomplete without giving thanks to our friends for their help throughout the course of this project and all those related people who directly or indirectly helped us for the completion of project successfully.

Name of the Projectees

Pritam R.Sah
Saurabh R.Nampalliwar
Sambhav S.Chopda
Abhishek R.Parmar
Anurag R.Tiwari

ABSTRACT

Communication is necessary for exchange of information. Similarly, interaction between the student and Head of the Department is important. The students has to contact the Department head for several purposes.

The main purpose of our project is to provide online interaction between the student and the Head of the department .The students can query to the Head regarding various purpose. The Head of Department, in return can attend to the query and depending upon the reason of the Query. The student can view the status of the Query whether approved by Head or not.

The Module can reduce most of the manual work and thus which can be done online. For the creation of module we are used SAP ABAP WebDynPro Programming.

LIST OF FIGURES

CONTENTS

INTRODUCTION

From a technological point of view, SAP.s Web Dynpro and ABAP is a revolutionary step in the development of Web-based user interfaces. It is completely unlike any design paradigm previously used by SAP and represents a quantum leap in the development of Web-based enterprise resource planning (ERP) applications. Web Dynpro applications are built using declarative programming techniques based on the Model View Controller (MVC) paradigm. That is, you specify which user interface elements you wish to have on the client, and where those elements will get their data from. You also define the possible navigation paths declaratively in your application. All the code to create the user interface is then generated automatically within a standard runtime framework. This relieves you from the repetitive coding tasks involved in writing HTML and making it interactive with JavaScript. ABAP Web Dynpro has been available since SAP Net Weaver 2004s (SAP Net Weaver Application Server 7.0). For developing the entities of a Web Dynpro application, the Object Navigator (transaction code SE80) has been enhanced. Web Dynpro is designed to support structured development. The software modularization units are Web Dynpro components, which can be combined to build up complex applications.

- ABAP stands for Advanced Business Application Programming
- ABAP is one of the many application-specific fourth-generation languages (4GLs) first developed in the 1980s.
- ABAP was one of the first languages to include the concept of Logical Databases (LDBs), which provides a high level of abstraction from the basic database level(s).
- All ABAP programs reside inside the SAP database. They are not stored in separate external files like Java or C++ programs.
- ABAP programs execute under the control of the runtime system, which is part of the SAP kernel.

TYPES OF ABAP PROGRAMS

ABAP distinguishes two types of executable programs:-

1. Reports

    - Reports follow a relatively simple programming model whereby a user optionally enters a set of parameters and the program then uses the input parameters to produce a report in the form of an interactive list.

2. Module pools

    - Module pools define more complex patterns of user interaction using a collection of screens.

WEB DYNPRO ARCHITECTURE

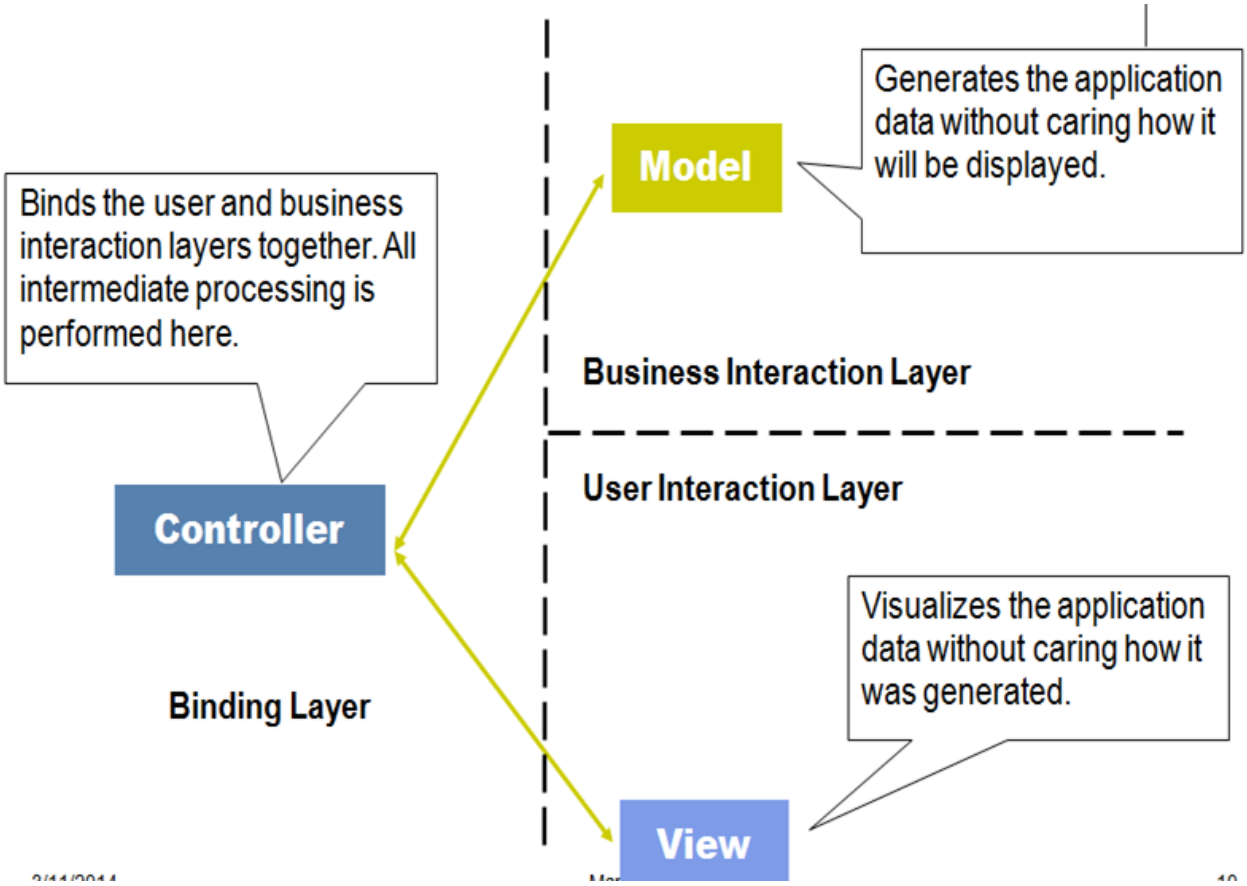


Fig.1 Architecture of Web DynPro

It is based on Model View Controller (MVC) Architecture and supports:

1. Clear separation of business and display logic.
2. Execution on number of client platforms.
3. Extensive Platform Independence of interfaces.

Model:-

- The model is used as an application object of the application data administration.

- In this way, only the model is used to process data internally, without making reference to the application and its user interface.

View:-

- The view handles the graphical and textual output at the interface and therefore represents the input and output data in each interface element, such as pushbuttons, menus, dialog boxes and so on.
- The view takes of visualization.

Controller:-

- The controller interprets and monitors the data that is input by the user
- Input data is forwarded and changes to the model data are initiated.
- The controller uses the model methods to change the internal status and then informs the view about this.

THE OBJECT-ORIENTED PROGRAMMING MODEL OF ABAP OBJECTS

The object-oriented concepts of ABAP Objects are essentially the same as those of other modern object-oriented languages like C++ or Java. A small number of concepts that did not prove to be successful in these other languages were not included in ABAP Objects. On the other hand, ABAP Objects also has helpful language elements that C++ and Java do not offer. Some specific features of ABAP Objects only exist because of the guaranteed upward compatibility of older ABAP language elements. Major differences in comparison to other object-oriented languages are in the development environment. You can use the entire range of functions of the ABAP Workbench with ABAP Objects.

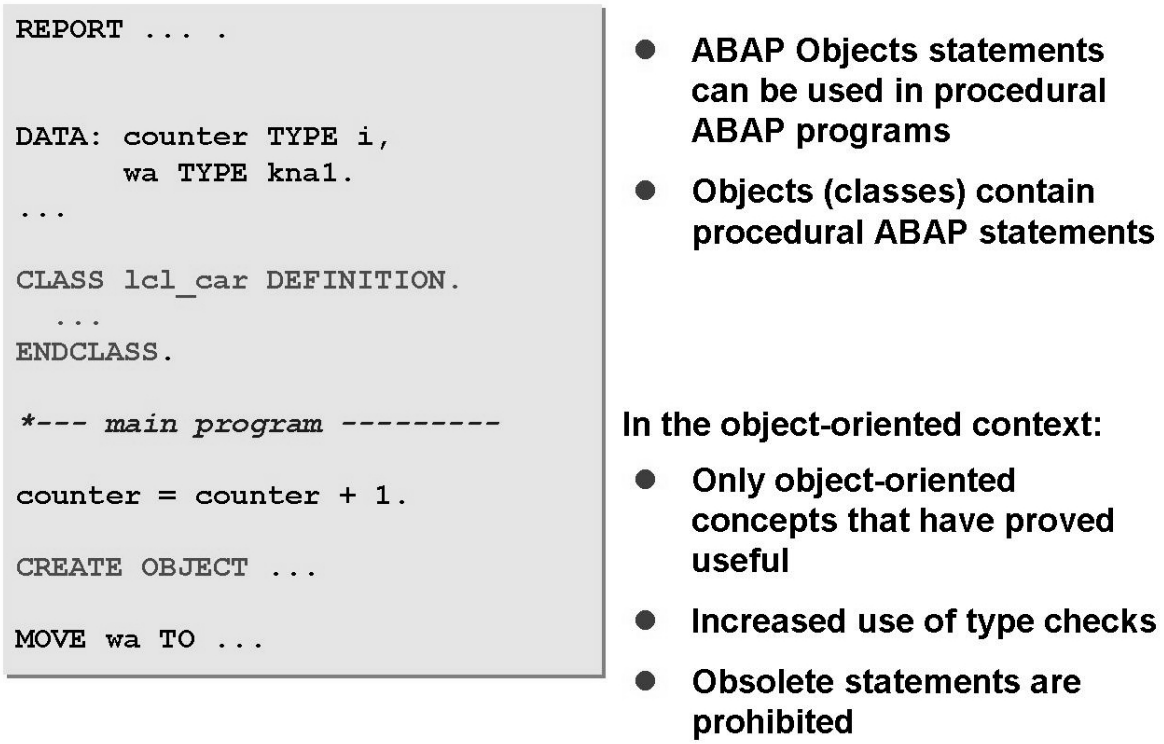


Fig.2 ABAP Objects as a Compatible Extension of ABAP

ABAP Objects is not a new language, but has been designed as a systematic extension of ABAP. All of the extensions, including the old procedural parts, are upwardly compatible. Type checks in the object-oriented contexts of ABAP Objects are stricter than those in the procedural contexts. In developing ABAP Objects, the ABAP language was cleaned up, in particular in the object-oriented contexts. This means that obsolete statements lead to syntax errors. However, it is also advisable to avoid obsolete statements in the purely procedural environment, as this creates source texts that are safer and more flexible. Nevertheless, as the language is upwardly compatible, it is not possible to prevent the

use of such statements entirely. For a list of obsolete language elements, refer to the ABAP keyword documentation. Each of the obsolete statements is also specifically noted as forbidden in the object-oriented context.

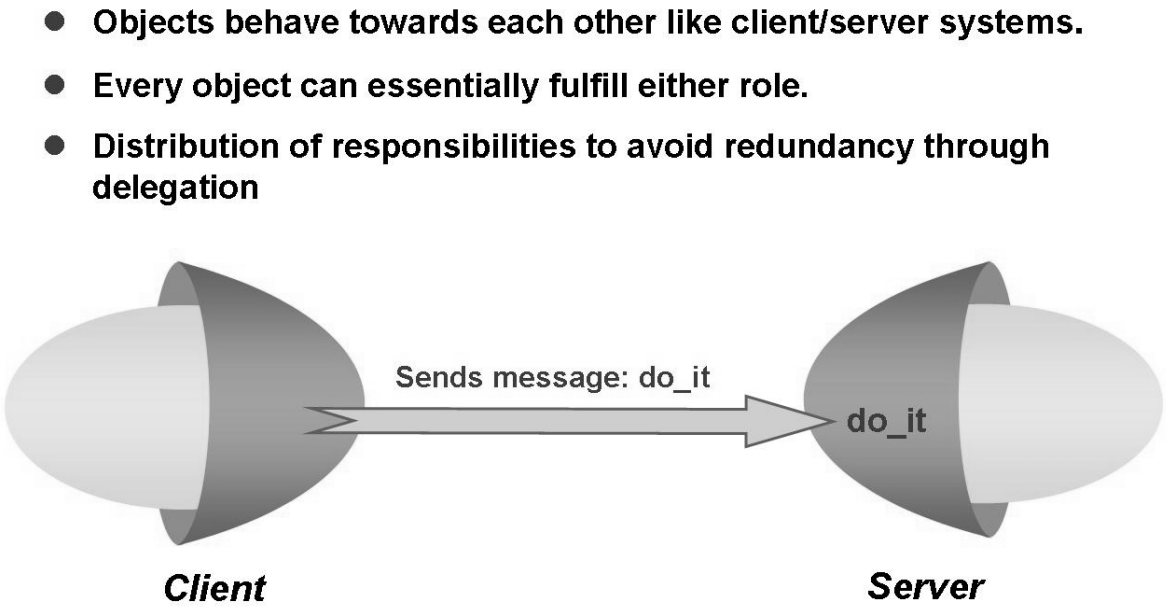


Fig.3 Client/Server Relationships between Objects

Objects behave like client/server systems: When one object sends a message to another object, telling it to behave in a certain way, the first object can be seen as a client and the other as a server. To be able to separate requests and deliveries of services, the following must be true:

- The client object must adhere to the server object's protocol.
- The protocol must be clearly described so that a potential client can follow it Without any problems.

. In principle, objects can perform both roles simultaneously: They can offer services to other objects while requesting services at the same time.In object-oriented programming, the services are distributed amongst the objects in such a way as to avoid redundancies and so that each object offers exactly those services that are within its area of responsibility. If an object needs any other services, it requests these from other objects. This is known as the **principle of delegation** of tasks.

Characteristics of the Object-Oriented Programming Model

- Objects are a direct abstraction of the real world
- Objects are units made up of data and the functions belonging to that data
- Processes can be implemented realistically.

Advantages of the Object-Oriented Programming Model over the Procedural Programming Model

- Improved software structure and consistency in the development process
- Reduced maintenance effort and less susceptibility to errors
- Better integration of the customer/user into the analysis, design, and Maintenance process.
- The options for extending the software are simpler and more secure.

The software development process

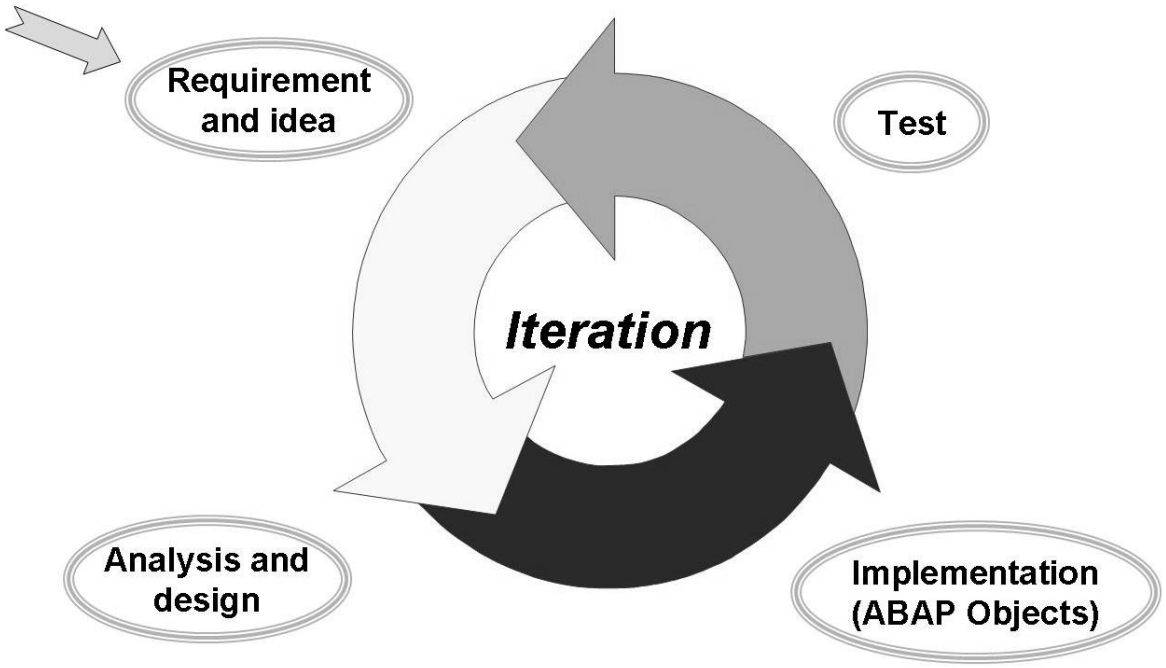


Fig.4 Software Development Process

THE ABAP DICTIONARY

In the ABAP Dictionary, you can create user-defined types (data elements, structures, and table types for use in ABAP programs or in interfaces of function modules, object methods, etc. Database objects such as tables, indexes and views can also be defined in the ABAP Dictionary and created with this definition in the database. The ABAP Dictionary also provides a number of services that support program development. For example, setting and releasing locks, defining an input and attaching a field help to a screen field are supported.

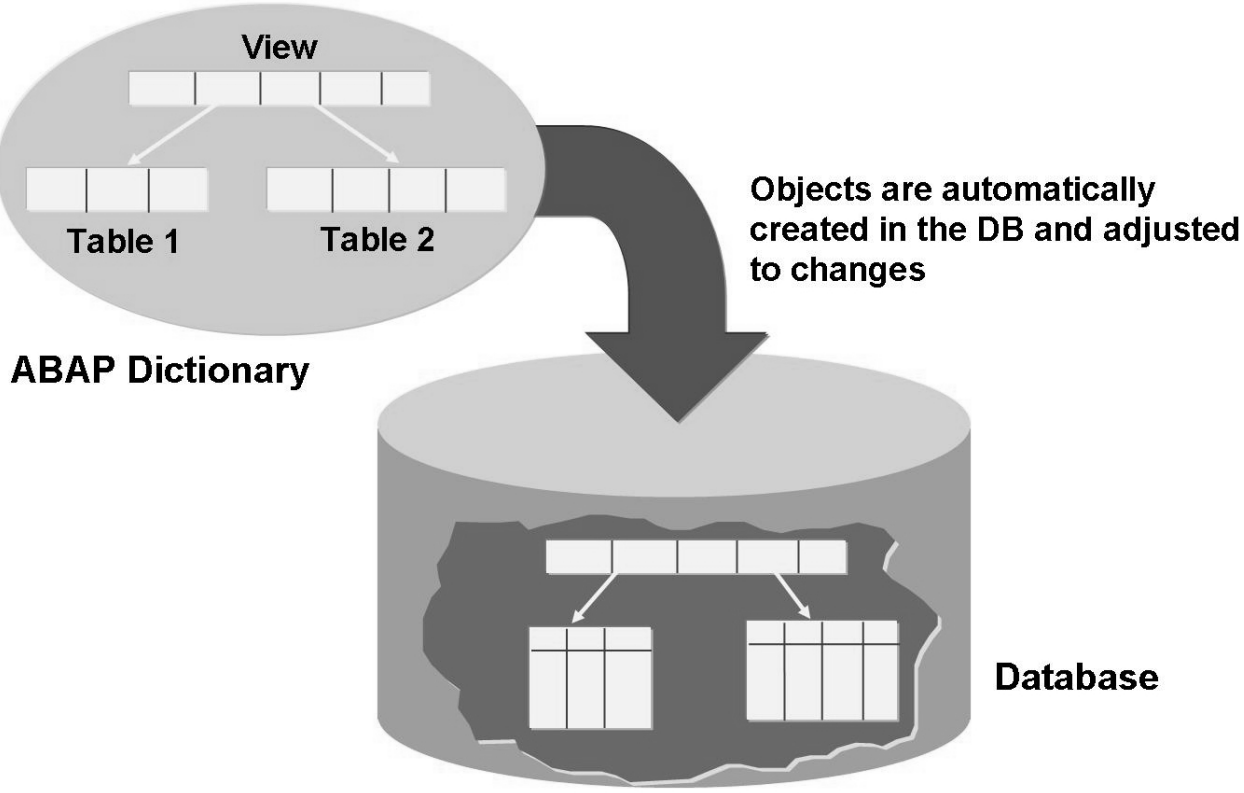


Fig.5 Database objects in the ABAP Dictionary

Tables and database views can be defined in the ABAP Dictionary. These objects are created in the underlying database with this definition. Changes in the definition of a table or database view are also automatically made in the database. Indexes can be defined in the ABAP Dictionary to speed up access to data in a table. These indexes are also created in the database.

Three different type categories exist in the ABAP Dictionary:

- **Data elements:** Describe an elementary type by defining the data type, length, and possibly decimal places.
- **Structures:** Consist of components that can have any type.
- **Table types:** Describe the structure of an internal table.

COMPONENTS OF SAP NETWEAVER

- SAP Business Information Warehouse (BIW)
- SAP Enterprise Portal
- SAP Business Intelligence
- SAP XI Exchanging Infrastructure
- SAP Knowledge Warehouse (KW)
- SAP Master Data Management (MDM)

SAP Business Information Warehouse: As the term suggests, SAP BIW provides data warehouse functionality for businesses. Several layers such as extraction layer, staging layer, transformation layer, loading layer and reporting and analysis layer integrate together in this component of SAP NetWeaver.

SAP Enterprise Portal: This component unifies and aligns the details, information and processes of different departments in a single space.

SAP Business Intelligence: This analytical, reporting and data warehousing solution enables better decisions in enterprises. SAP BI simplifies data manipulation and allows users to analyze and share information with stakeholders.

SAP XI Exchanging Infrastructure: SAP XI enables enterprises to connect systems from different vendors. These vendors might and might not use SAP warehouses or applications. This component extensively supports at cross-company scenarios.

SAP Knowledge Warehouse: SAP KW helps an organization to set up and manage both industry-specific and enterprise-specific knowledge in a unified system. It is the main element that manages the knowledge processes of an enterprise.

SAP Master Data Management: SAP MDM provides a single; integrate software solution that helps in creating, distribution and managing the entire master data at an enterprise level. This also provides the accurate data of an enterprise giving real-life status.

ABAP CONTEXTS

Within the large quantities of data in the SAP System database, there are always smaller sets of basic data that you can use to derive further information. In a relational database model, for example, these are the key fields of database tables.

When an application program requires further information in order to continue, this is often found in these small amounts of basic data. The relational links in the database are often used to read further data on the basis of this basic data, or further values are calculated from it using ABAP statements.

It is often the case that certain relationships between data are always used in the same form to get further data, either within a single program or in a whole range of programs. This means that a particular set of database accesses or calculations is repeatedly executed, despite the fact that the result already exists in the system. This causes unnecessary system load, which can be alleviated by using contexts.

Contexts are objects within the ABAP Workbench, consisting of key input fields, the relationships between these fields, and other fields which can be derived from them. Contexts can link these derived fields by foreign key relationships between tables, by function modules, or by other contexts.

You define contexts abstractly in the ABAP Workbench (Transaction SE33), and use instances of them as runtime objects in your ABAP programs. You define contexts in the Context Builder, which is part of the ABAP Workbench. They consist of key input fields, the relationships between the fields, and the other fields and values that you can derive from them. Contexts can link these derived fields by foreign key relationships between tables, by function modules, or by other contexts.

In the ABAP Workbench, contexts consist of fields and modules. Their fields are divided into key fields and derived fields. The modules describe the relationship between the fields.

Technically, contexts are special ABAP programs. You can save them:

- In a non-executable program, with the name CONTEXT_S_<context name>. For the example context DEMO_TRAVEL, this would be CONTEXT_S_DEMO_TRAVEL).
- In a generated executable program, with the name CONTEXT_X_<context name>.For the example context DEMO_TRAVEL, this would be CONTEXT_X_DEMO_TRAVEL).

CONTROLLERS OF A WEBDYNPRO COMPONENT

The component controller provides data and processing logic that it should be possible to display or change for all views of a component. It has three interfaces:

- **Interface IF_<controller_name>** for coding within the controller
- **Interface IG_<controller_name>** for cross-controller coding within the current component.
- **Interface IWCI_<component_name>** for cross-component coding. On the ABAP language level it represents the interface controller.

Within the component, mapping is possible to any context element of the component controller. The attributes of a component controller are known to all methods that are called within a component and can be used by them, provided they have the *Public* property. Otherwise, their visibility is restricted to the component controller. Events and methods assigned to the component controller are visible throughout the component. So, for example, any action of a view of the component can call such a method of the component controller.

The visibility of a specially marked number of methods, events and context nodes of a component controller can be extended beyond the borders of one's own component. They then form the interface controller of the component.

Interface Controller

The interface controller is used for **cross-component** communication. It defines the controller part of the interface of a Web Dynpro component. The interface controller itself does not contain any implementation.

- For the interface controller of a Web Dynpro controller, the methods are implemented in the related component controller.
- For the interface controller of a component interface definition, the implementation is performed in the component controller of the embedding component.

Custom Controller

The properties and use of the custom controller that can be added optionally correspond exactly to those of the component controller. This means that it is visible to all elements of the component and the lifetime of the elements is the lifetime of the component. The custom controller gives you the option of structuring functions and data within a component. It makes sense to create and maintain a custom controller if a certain subset of views of a component should be equipped with a special function or with a special set of data.

View Controller

Just like the component controller, the view controller contains data and functions. Furthermore, the view controller also has An **IF_<Controllername> interface**, which means that these data and functions are visible within the same view only.

The lifetime of the controller element of a view is restricted to the lifetime of the view. To create simple Web Dynpro applications, it is sufficient if you maintain the component controller and the view controller of a view. The two controllers described below are optional and help you to structure complicated components and applications, thereby increasing their reusability.

Window Controller

For every window created in a WebDynpro component, a window controller is generated. Such a window controller is visible throughout the component – it behaves like a component or custom controller.

WORKING

The working of our project depends on different views and component controller. Component controller consist of the context in which contains node and attribute. The text field in the views need to bind with the attribute in context. Our project consist of two modules i.e. HOD and Student module. When any student want to send any request to the HOD then he/she can directly send queries to HOD by just logging in into our software and then he/she can type the subject and send it to HOD. The status of his/her request can be check after the student log-in. whereas in HOD module, the number of request send by the student can be seen by HOD after he/she log-in. HOD module consist of 2 button i.e. Accept or Reject. When HOD click on Accept then the status of that particular student will be change to "Request Accepted" otherwise "Request Rejected" when HOD click on Reject button. All the data is saved in data dictionary table. The queries are saved in database i.e. data dictionary table in SAP ABAP. The id and password is also match when student or HOD log-in into our software.

In this way Student and HOD can communicate with each other and there is no need of written application. This reduces the manual work of the student as well as HOD.

APPLICATIONS

- Business Application.
- Data Warehousing.
- Database &Technology.
- Analytics.
- Cloud.
- Mobile.

CONCLUSION

Department Communication System will be a very revolutionary step which will take the Communication between the Head of Department and the Student online.

The Module will decrease most of the manual work done through the papers, and provide the easier way to the students which are called the social generation to have a system which suits their environment of online communication.

Thus, we conclude that this project will:

- Reduce the manual work of communication to much extend ,
- Change the way of interaction between staff and students and,
- Reduce the Communication Gap.

FUTURE SCOPE

The project which currently provides interaction between the Head and the student can be extended for communication between the whole department.

The Extension may include the system which communicates between all the departments of the college and the Administrative Body of the College.

Thus, Communication becoming online, easy and eco-friendly as reducing the use of papers.

REFERENCES

Book:

[1] Introduction to SAP ABAP WebDynPro: SAP Partners

[2] ABAP Objects, SAP Net Weaver: Instructor Handbook by SAP.

[3] ABAP Dictionary: Instructor Handbook by SAP.

[4] ABAP objects and Dictionary: Rolling Arrays, Singapore.

Websites:

[1] http://www.youtube.com/watch?v=_yLPNYnCQaY

[2] http://s.ytimg.com/yts/swfbin/player-vflRtOPPh/watch_as3.swf