ALV REPORT GENERATION

IN ABAP

(ADVANCED BUSINESS APPLICATION PROGRAMMING)

UNDER THE ABLE GUIDANCE OF:

Mr. Amit Moza
Senior Programming Officer
Project ICE, ONGC

SUBMITTED BY:

NAME: Diksha
COLLEGE: Indira Gandhi Delhi Technical University for Women, Kashemere Gate, Delhi-110006
ROLL NO. : 04601012013
COURSE: Bachelor of Technology in Computer Science
BATCH: 2013-17

Contents

DECLARATION.....5

CERTIFICATE.....6

ABSTRACT.....7

ACKNOWLEDGEMENT.....8

ONGC Profile 8

INTRODUCTION TO THE INDUSTRY 8

HISTORY OF THE COMPANY 8

GLOBAL RANKING 9

ONGC VISION 10

ONGC MISSION 10

OBJECTIVES OF THE COMPANY 10

Project ICE 11

The Objectives of the Project 11

Major steps in the implementation of the Project 11

Modules implemented in ONGC: 12

Replicability 14

ABAP Language 16

SAP transaction code 17

ABAP Data Types and Constants 18

ABAP Variables 19

ABAP System Variables 21

Control Statements 21

ABAP Report Programming 23

ABAP Dialog Programming 24

Difference between Report and Dialog Programs 24

Screen Painter SE51 27

Screen Attributes 27

Project Profile 36

Constraints 37

Hardware Requirements 38

HARDWARE COMPONENTS 38

MINIMUM REQUIREMENTS 38

PROJECT STRUCTURE 42
SOURCE CODE 44
CONCLUSION 49
Biblography 50

DECLARATION

I, **DIKSHA**, student of 3rd year pursuing **Bachelor of Technology in Computer Science** from **Indira Gandhi Delhi Technical University for Women, Kashemere Gate, Delhi-110006** hereby declare that the work presented in this dissertation is the outcome of my own work, is bonafide and correct to the best of my knowledge and this work has been carried out taking care of the Engineering Ethics. The work presented does not infringe any patented work and has not been submitted to any other university or organization for the award of any degree or any professional diploma.

(Signature of student)
DIKSHA

Certified that the above statement made by the student is correct to the best of my knowledge and belief.

Mr. Amit Moza
Senior Programming Officer
Project ICE, ONGC

CERTIFICATE

OIL AND NATURAL GAS CORPORATION LIMITED

Project ICE, Scope Minar, Laxmi Nagar, New Delhi- 110092

Date - 29 August 2016

This is to certify that **DIKSHA** student of **Bachelor of Technology in Computer Science** from **Indira Gandhi Delhi Technical University for Women, Kashmere Gate, Delhi- 110006** has successfully completed her project with ONGC Ltd., Delhi during the period of 01/06/2016 to 30/06/2016.

She has done her project titled '**ALV Report Generation'** under our guidance. We have observed that, her work has been excellent and appreciate her sincere learning. She has performed the project with energy and enthusiasm. We wish her all the best for her future endeavours.

RAJIV KHERA
CE (E&T)
PDC
Project ICE

AMIT MOZA
Project Mentor
Senior Programming Officer
PDC, Project ICE

ABSTRACT

The name of the Project is ‘ALV Report Generation’. The purpose of the project is to generate an ALV report for the given database(s).

The ALV report generation system designed as a part of this project is a robust application and can be used by any university for generating its set of reports.

The project uses SAP as the ERP software. The projects objective is to maintain all the university information related to leave sanctions.

- The user can view the list of all the departments in the university.
- The project helps the university in understanding the details of the courses it offers.
- This project helps organizations keep track of their faculty and the subjects they are teaching as well.
- Working with SAP allows for easy access and availability to the program anywhere and everywhere.

ACKNOWLEDGEMENT

I, **DIKSHA,** express my deep sense of gratitude and reverence to the esteemed organization, ONGC, for giving me opportunity to undertake training and providing me with great infrastructure and facilities.

I express my profound gratitude to my mentor **Mr. Amit Moza, Senior Programming Officer, Project ICE, ONGC** for providing guidance and expert supervision for this project, which helped in completing the project on time. Without his friendly help, guidance and helpful suggestions it was difficult to complete the assigned task.

I am truly grateful to **Mr. RAJIV KHERA, CE (E&T)**, **ICE** for his professional guidance and significant inputs throughout the duration of the project. He has regularly checked my project and has always helped me whenever I needed him.

I am truly thankful to the entire ICE PDC team in Project ICE ONGC, Delhi for their support and timely help in solving problems related to EHP6 FOR SAP ERP 6.0 from time to time.

ONGC Profile

INTRODUCTION TO THE INDUSTRY

Oil and Natural Gas Corporation Limited (ONGC) is engaged in the business of exploration and drilling of crude oil and natural gas and is the world's second biggest exploration and production company ONGC owns and operates more than 11000 kilometers of pipelines in India, including nearly 3200 kilometers of sub-sea pipelines. The company contributes more than 78% of India's oil and gas production.

Today, ONGC is the flagship company of India; and making this possible is a dedicated team of nearly 40,000 professionals who toil round the clock. It is this toil which amply reflects in the performance figures and aspirations of ONGC. The company has adapted progressive policies in scientific planning, acquisition, utilization, training and motivation of the team.

Over 18,000 experienced and technically competent executives mostly scientists and engineers from distinguished Universities/Institutions of India and abroad form the core of its manpower. They include geologists, geophysicists, geochemists, drilling engineers, reservoir engineers, petroleum engineers, production engineers, engineering & technical service providers, financial and human resource experts, IT professionals and so on.

HISTORY OF THE COMPANY

The India Petroleum Industry is a case in point for exhibiting the giant leaps India has taken after its independence towards its march to attain a self-reliant economy. During the Independence era of 1947, the India Petroleum Industry was controlled by foreign companies and India's own expertise in this sector was limited. Now, after 60 years, the India Petroleum Industry has become an important public sector undertaking with numerous skilled personnel and updated technology that is comparable to the best in the world. The vim and the achievement during these years is the growth of productivity in petroleum and petroleum-based products. Even the consumption has multiplied itself nearly 30 times in the post-independence era.

The ONGC originally set up as a Directorate in 1955, was transformed into a Commission in 1956. In 1958, the Indian Refineries Ltd., a government undertaking, came into existence. The Indian Oil Company (IOC), also a government undertaking, was set up in 1959 with the purpose of marketing petroleum-related products. Indian Oil Corporation Ltd. was formed in 1964 with the merger of the Indian Refineries Ltd. and the Indian Oil Company Ltd. Presently, 17 refineries operate under the India Petroleum Industry.

GLOBAL RANKING

• ONGC ranks as the Numero Uno Oil & Gas Exploration & Production (E&P) Company in Asia, as per Platt's 250 Global Energy Companies List for the year 2007.

• ONGC ranks 23rd Leading Global Energy Major amongst the —Top 250 Energy Majors of the World in the Platt's List‖ based on outstanding performance in respect of Assets, Revenues, Profits and Return on Invested Capital (RIOC) for the year 2007.

• ONGC is the only Company from India in the Fortune Magazine's list of the World's Most Admired Companies 2007. ONGC is 9th position in the Industry of Mining, crude oil production.

• ONGC ranks 3rd Oil & Gas Exploration & Production (E&P) Company in the world and 23rd among leading global energy majors as per Platt's 250 Global Energy Companies List for the year 2009.

• ONGC ranks 24th among the Global publicly-listed Energy companies as per PFC Energy 50‖ (Jan 2008)

• Finance Asia 100 list ranks ONGC no 1 among Indian Blue Chips.

• Occupies 155th rank in —Forbes Global 2000‖ list 2010 of the world's biggest companies for 2010 based on sales, profits, assets and market capitalization.

• ONGC ranked 402nd position as per Fortune Global 500 - 2009 list; based on revenues, profits, assets and shareholder's equity.

ONGC VISION

To be a world class Oil and Gas Company integrated in energy business with dominant Indian leadership and Global presence.

ONGC MISSION

• Dedicated to excellence by leveraging competitive advantages in R&D and technology with involved people.

• Imbibe high standards of business ethics and organizational values.

• Abiding commitment to safety, health and environment to enrich quality of community life.

• Foster a culture of Trust, openness and mutual concern to make a stimulating and challenging experience for our people.

• Strive for customer delight through quality products and services.

OBJECTIVES OF THE COMPANY

To maximize production of hydrocarbon, self-reliance in technology, promoting indigenous efforts to achieve self-reliance in technology, promoting indigenous efforts to achieve in all related equipment, material and services.

• Assist in conservation of oil, more efficient use energy and development of alternate source of energy.

• Environmental protection

• Observe 100% safety in work.

Project ICE

The objective of the ICE project is to optimize and standard the business processes to enable availability of information on real time basis; and eliminate duplication of activities to increase efficiency and transparency.

The Oil and Natural Gas Corporation (ONGC) realigned its business processes to bring all the legacy systems under a common Enterprise Resource Planning (ERP) also known as Information Consolidation for Efficiency (ICE).

Earlier, the ONGC monitored its activities on a daily, monthly, quarterly and annual basis at both work center and corporate levels. This involved a comprehensive and an exhaustive process exercised in a manual manner. In absence of any integrated online data system, there was lack of co-ordination and no time management between the activity period and data availability. This resulted constraints in logical decision making as well.

Post implementation of ICE yielded better results including optimization and standardization of business process, higher productivity, reduction of cost, strengthening efficiency thereby increasing customer service and satisfaction.

The Objectives of the Project

a) Optimization and standardization of business processes.

b) Moving up the Value chain

c) Higher Productivity

d) Cost Reduction

e) Strengthening Efficiencies

f) Lowering of Inventories

g) Increasing Customer service and satisfaction

Major steps in the implementation of the Project

1. Project Preparation (Design for all Business modules for all Stages and phases)

2. Business Blueprint (Design for all Business modules for all Stages and phases)

3. Realization (phase wise)

4. Final Preparation (phase wise)

5. Go-live and Support (phase wise)

Modules implemented in ONGC:

(1) Production and Planning (PP): The primary objective of the PP module is to track planned and actual costs of production processing of Crude Oil, Natural Gas and VAP. It facilitates real time updating of data, helps in calculating actual & standard costs at any stage in the product cycle, monitors real time production environment with online availability of Information related to Materials & Products, as well as customized report generation for faster decision making.

(2) Plant Maintenance (PM): The PM module provides a system for the management and maintenance of technical systems including the cost incurred in the planned and breakdown maintenance. By being integrated with other modules it gives the cost of each maintenance activity. It will also track various audit activities and their 9 follow up actions in ONGC. New feature, like online availability of equipment manuals was invoked through LDM functionality.

(3) Financial Accounting: This module Integrates General Ledger, Accounts Payable, Accounts Receivable with all the sub ledgers synchronized with the G/L in an on-line, real-time manner. The existing UFSO (KUBER) is a standalone module with only FI functions. In ICE FI function is integrated with all the adopted R/3 modules starting from supply to the sales. FI function have been suitably updated and up linked to this integrated system to seamlessly interact with all other modules for comprehensive transaction tracking and reporting facilities in all the areas of Financial Management System.

(4) Controlling (CO): Controlling (CO) covers the functionalities of Cost Centre Accounting, Profit Centre Accounting and Product Costing for wide range of Management reporting. Controlling features are integrated to the operational modules such as Sales & Distribution, Material Management, Production Planning, Plant Maintenance, OLM, Project System and Financial Accounting.

(5) Joint Venture Accounting (JVA): This module is to cover the Joint Venture activities, starting from Joint operating agreements, Work Programs, Equity equations, Expenditure, Cash Calls, Recovery, Billing and Accounting (as operator and non-operator).

(6) Sales & Distribution (SD): SD module comprises of entire Sales & Distribution activities starting from sales agreements to delivery and generation & printing of invoice in integrated sales process for all products of ONGC including scrap and services. It is integrated with financial accounting for account receivable management; material management and production planning for real time stock updating and quality management for 10 quality

analysis and reporting. Fully compliant with Indian taxation requirement including VAT, it will generate statutory documents, eg. Excise invoice and sales registers and maintain audit trail of transactions through document flow.

(7) Project System (PS): This module encompasses all phases of a project from Project Conceptualization, Budgeting, planning of costs and resources and approval of Estimates to Execution, payment and Completion of the project in an integrated scenario. Many customized developments have been made in PS module for Engg. Services, Drilling, Work over, Survey, NELP, Dry docking and consultancy/ R&D operations. It enables the treatment of a project as an Enterprise with links to other functional modules and the project can be analyzed in its entirety.

(8) Material Management (MM): This module integrates all transactions and functions necessary for material requirement planning, procurement, inventory management, invoice verification, and material valuation. In addition to handling special stock types for Crude oil and other product materials transported by pipeline, this will monitor stocks and automatically generate purchase order proposals for the purchasing department. Existing IMMS system have been seamlessly updated into this system. Additional feature of mapping Service Contracts and Works has been done in this module.

(9) Quality Management (QM): QM module covers inspection of procured material, inspection of in-house products, and generation of Quality certificate for issuing finished products to the Customers. Among many features, Vendor/Material complaints processing, quality clearance certificate for incoming material and for the products, failure analysis etc. shall be available through this system.

(10) ABAP: The ABAP (Advanced Business Application Programming) development team provides support to functional module team pertaining to any new developments, enhancements, feasibility, data migration etc. in the standard SAP R/3 system so as to configure as per ONGC's business process pertaining to MIS reports, strategic decision making reports.

(11) Business Information Warehouse (BW): This module shall generate analytical and strategic reports for Business Analysis and performance tracking including the Corporate Key Performance Indicators. The inputs will come from all finance and logistics SAP modules as well as from non-SAP systems like Excel files also. This would become the single, integrated, MIS System for ONGC. These reports would be available online and on the web.

Replicability

Project Ice is replicated both within India and outside the country as well. Public Sectors like Indian Oil Corporation (IOC), Gas Authority of India Limited (GAIL) and Steel Authority of India Limited are or have replicated the ICE model. ONGC operates in 10 different countries across the globe where project ICE was replicated. For instance in Sudan ONGC has implemented this project successfully. 12 “Change Management” is what which has been the USP (Unique Selling Point) of the project ICE to be replicated. Due care has been taken for the Data Centre system to be replicated if need be as per the requirement of the organization. A fully redundant setup with three tier security zones, with no single point of failure has been established. Backup as well as a geographically separated disaster recovery center is also part of the scheme.

SAP

SAP, started in 1972 by five former IBM employees in Weinheim, Germany, states that it is the world's largest inter-enterprise software company and the world's third-largest independent software supplier, overall.

The original name for SAP was German: Systeme, Anwendungen, Produkte, German for "**Systems Applications and Products.**"

The original SAP idea was *to provide customers with the ability to interact with a common corporate database for a comprehensive range of applications*.

Gradually, the applications have been assembled and today many corporations, including IBM and Microsoft, are using SAP products to run their own businesses.

SAP applications, built around their latest R/3 system, provide the capability to *manage financial, asset, and cost accounting, production operations and materials, personnel, plants, and archived documents*.

The R/3 system runs on a number of platforms including Windows 2000 and uses the client/server model. The latest version of R/3 includes a comprehensive Internet-enabled package.

ABAP Language

ABAP stands for **Advanced Business Application Programming**. It is a programming language developed by SAP.

ABAP language syntax

- **ABAP is not case sensitive.**
- **Every statement begins with a keyword and ends with a period.**(WRITE is the keyword to print on screen)

```
WRITE 'Hello World!'.
```

- **Chained statements.**If consecutive statements have identical part at the beginning, then ABAP allows you to chain these statements into a single statement. First write the identical part once and then place a colon (:). Then write the remaining parts of the individual statements separated by commas.Normal Statements:

```
WRITE 'Hello'.
```

```
WRITE 'ABAP'.
```

Chained Statement:

```
WRITE: 'Hello', 'ABAP'.
```

- **Comments.**If you want to make the entire line as comment, then enter asterisk (*) at the beginning of the line.

```
* This is a comment line
```

If you want to make a part of the line as comment, then enter double quote (“) before the comment.

```
WRITE 'COMMENT'.   "Start of comment
```

SAP transaction code

SAP Transaction code is a short cut key attached to a screen. Instead of using SAP easy access menu we can also navigate to a particular screen in SAP by entering the transaction code (T-code for short) in the command field of the standard toolbar.

Some of the useful transaction codes for ABAP developers.

T CODE	DESCRIPTION
SE11	ABAP Data Dictionary
SE16	Data Browser
SE37	Function Builder
SE38	ABAP Editor
SE41	Menu Painter
SE51	Screen Painter
SE71	SAP Script Layout
SE80	ABAP Workbench
SE91	Message Maintenance
SE93	Maintain Transaction

ABAP Data Types and Constants

Data Type describes the technical characteristics of a Variable of that type. Data type is just the blue print of a variable.

Predefined ABAP Types

DATA TYPE	DESCRIPTION	DEFAULT LENGTH	DEFAULT VALUE
C	Character	1	‘ ‘
N	Numeric	1	0
D	Date	8	00000000
T	Time	6	000000
X	Hexa Decimal	1	X’0’
I	Integer	4	0
P	Packed	8	0
F	Float	8	0

User Defined Data Types

Use **TYPES** keyword to define the data types.

```
TYPES: name(10) TYPE c,
       length   TYPE p DECIMALS 2,
       counter  TYPE i,
       id(5)    TYPE n.
```

Structured data types

Structured data type is grouping of several simple data types under one name.

Use the keywords **BEGIN OF** and **END OF** to create a structured data type.

```
TYPES: BEGIN OF student,
        id(5)     TYPE n,
        name(10)  TYPE c,
```

```
    dob      TYPE d,
    place(10) TYPE c,
    END OF student.
```

Constants

Constants are used to store a value under a name. We must specify the value when we declare a constant and the value cannot be changed later in the program.

Use **CONSTANTS** keyword to declare a constant.

```
CONSTANTS: pi  TYPE p DECIMALS 2 VALUE '3.14',
        yes TYPE c VALUE 'X'.
```

ABAP Variables

ABAP Variables are instances of data types. Variables are created during program execution and destroyed after program execution.

Use keyword **DATA** to declare a variable.

```
DATA: firstname(10) TYPE c,
    index       TYPE i,
    student_id(5) TYPE n.
```

While declaring a variable we can also refer to an existing variable instead of data type. For that use **LIKE** instead of**TYPE** keyword while declaring a variable.

```
DATA: firstname(10) TYPE c,
    lastname(10)  LIKE firstname. " Observe LIKE keyword
```

Structured Variable

Similar to structured data type, structured variable can be declared using **BEGIN OF**and **END OF** keywords.

```
DATA: BEGIN OF student,
    id(5)     TYPE n,
    name(10)  TYPE c,
    dob      TYPE d,
```

```
    place(10) TYPE c,
    END OF student.
```

We can also declare a structured variable by referring to an existing structured data type.

```
TYPES: BEGIN OF address,
      name(10)   TYPE c,
      street(10) TYPE c,
      place(10)  TYPE c,
      pincode(6) type n,
      phone(10)  type n,
      END OF address.

Data: house_address  type address,
     office_address like house_address.
```

Each individual field of the structured variable can be accessed using hyphen (-). For example, name field of the house_address structure can be accessed using housing_address-name.

Character is the default data type.

```
DATA: true.  " By default it will take C as data type
```

ABAP System Variables

ABAP system variables is accessible from all ABAP programs. **These fields are filled by the runtime environment**.
The values in these fields indicate the state of the system at any given point of time.The complete list of ABAP system variables is found in the **SYST** table in SAP. Individual fields of the SYST structure can be accessed either using **"SYST-"** or **"SY-".**

```
WRITE:/ 'ABAP System Variables'.

WRITE:/ 'Client : ', sy-mandt.

WRITE:/ 'User : ', sy-uname.

WRITE:/ 'Date : ', sy-datum.

WRITE:/ 'Time : ', sy-uzeit.
```

Control Statements

To control the flow of the ABAP program use the following statements.

IF Branching Conditionally

IF statement – The code between **IF** and **ENDIF** is executed only if the condition is true.

```
DATA: a TYPE i VALUE 10.  " We can assign a value in the declaration



IF a > 5.

 WRITE:/ 'Condition True'.

ENDIF.
```

IF-ELSE statement – The code between **IF** and **ELSE** is executed if the condition is true, the code between **ELSE** and**ENDIF** is executed if the condition is False.

```
DATA: a TYPE i VALUE 1.



IF a > 5.

 WRITE:/ 'Condition True'.

ELSE.

 WRITE:/ 'Condition False'.
```

```
ENDIF.
```

IF-ELSEIF statement – Used to check multiple conditions.

```
DATA: a TYPE i VALUE 2.


IF a > 5.
  WRITE:/ a, 'Greater Than', 5.
ELSEIF a > 4.
  WRITE:/ a, 'Greater Than', 4.
ELSEIF a > 3.
  WRITE:/ a, 'Greater Than', 3.
ELSE.
  WRITE:/ a, 'Less Than', 3.
ENDIF.
```

CASE-ENDCASE – Branching based on the content of the variable.

```
DATA: a TYPE i VALUE 4.


CASE a.
  WHEN 3.
    WRITE:/ a, 'Equals', 3.
  WHEN 4.
    WRITE:/ a, 'Equals', 4.
  WHEN OTHERS.
    WRITE:/ 'Not Found'.
ENDCASE.
```

ABAP Report Programming

SAP-ABAP supports two types of Programs - Report Programs & Dialog Programs. Report Programs are used when large amounts of data needs to be displayed

Purpose/Use of Report Programs:

- They are used when data from a number of tables have to be selected and processed before presenting
- Used when reports demand a special format
- Used when the report has to be downloaded from SAP to an Excel sheet to be distributed across.
- Used when the report has to be mailed to a particular person.

Important Points to Note about Report Program

- Report Programs are always Executable Programs. Program Type is always 1.
- Every Report program corresponds to a particular Application Type i.e. either with Sales & Distribution, FI - CO etc. It can also be Cross Application i.e. type '*'.
- Report Programming is an Event-driven programming.
- The first line of a report program is always Report <report-name>.
- In order to suppress the list heading or the name of the program the addition No Standard Page Heading is used.
- The line size for a particular report can be set by using the addition line-size <size>.
- The line count for a particular page can be set by using the addition line-count n(n1). N is the number of lines for the page and N1 is the number of lines reserved for the page footer.
- To display any information or error message we add a message class to the program using the addition: Message-id <message class name>. Message classes are maintained in SE91.

```
Report <report name> no standard page heading
line-size <size>
line-count <n(n1)>
message-id <message class>.
```

Therefore an ideal report program should start with:

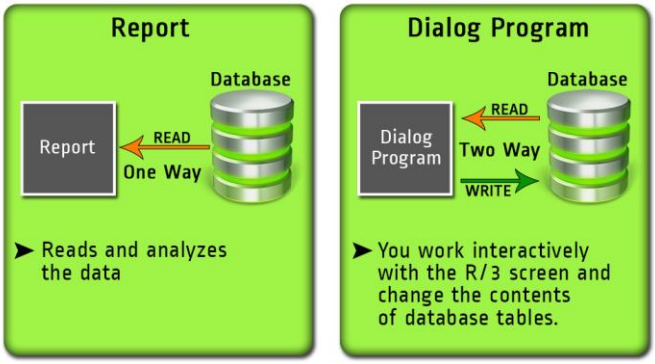
ABAP Dialog Programming

SAP-ABAP supports two types of programs - Report Program and Dialog Program.If your ABAP program demands user input, Dialog programming is used.A user dialog is any form of interaction between the user and the program and could be any of the following:

- Entering data
- Choosing a menu item
- Clicking a button
- Clicking or double clicking an entry
- Dialog program is also used when we need to navigate back and forth between screens

Dialog programs are created with type as 'M' - Module Pool. They cannot be executed independently and must be attached to at least one transaction code in which you specify an initial screen.

Difference between Report and Dialog Programs



Report Program:

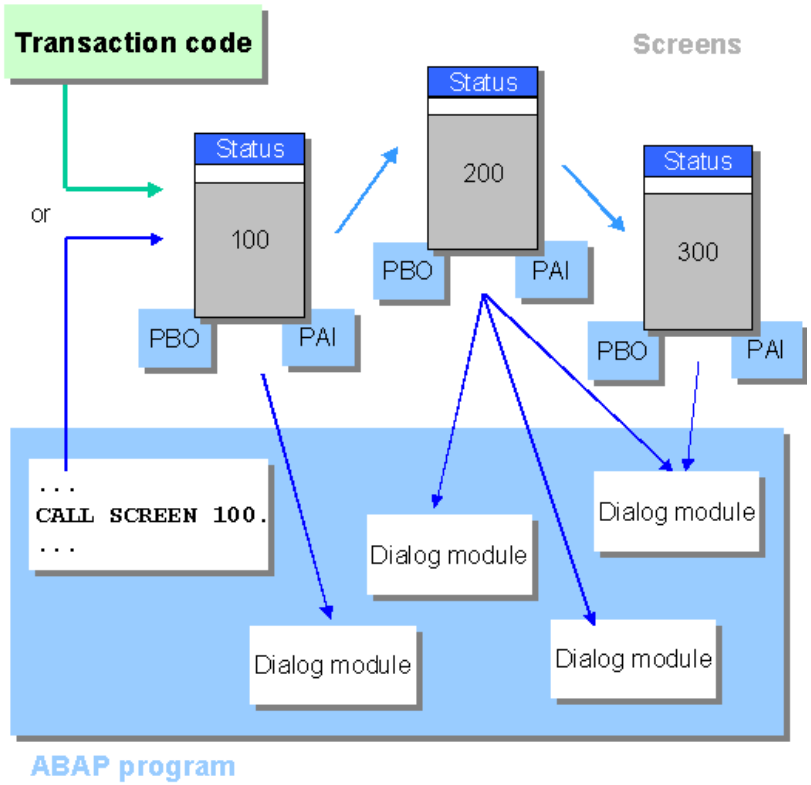
A report is a program that typically reads and analyzes data in database tables without changing the database.

Dialog Program:

A dialog program allows you to work interactively with the system and to change the contents of the database tables. Each dialog program has a certain sequence of screens that are processed by the system one after the other.

Components of a Dialog Program

A dialog-driven program consists of the following basic components:



Transaction code

The transaction code starts a screen sequence. You create transaction codes in the Repository Browser in the ABAP Workbench or using Transaction SE93. A transaction code is linked to an ABAP program and an initial screen. As well as using a transaction code, you can start a screen sequence from any ABAP program using the CALL SCREEN statement.

ABAP Program

Each screen and GUI status in the R/3 System belongs to one ABAP program. The ABAP program contains the dialog modules that are called by the screen flow logic, and also process the user input from the GUI status. ABAP programs that use screens are also known as dialog programs. In a module pool (type M program); the first processing block to be called is always a dialog module. However, you can also use screens in other ABAP programs, such as executable programs or function modules. The first processing block is then called differently; for example, by the runtime environment or a procedure call. The screen sequence is then started using the CALL SCREEN statement.

Dialog modules are split into PBO modules and PAI modules. Dialog modules called in the PBO event are used to prepare the screen, for example by setting context-specific field contents or by suppressing fields from the display that are not needed. Dialog modules called in the PAI event are used to check the user input and to trigger appropriate dialog steps, such as the update task.

Screens

Each dialog in an SAP system is controlled by one or more screens. These screens consist of a screen mask and its flow logic. Since the flow logic influences the program flow, screens are sometimes referred to as "dynamic programs". You create screens using the Screen Painter in the ABAP Workbench (SE51). Each screen belongs to an ABAP program.

The screen has a layout that determines the positions of input/output fields and other graphical elements such as checkboxes and radio buttons. The flow logic consists of two parts:

Process Before Output (PBO) : This defines the processing that takes place before the screen is displayed.

Process After Input (PAI) : This defines the processing that takes place after the user has chosen a function on the screen.

All of the screens that you call within an ABAP program must belong to that program. The screens belonging to a program are numbered. For each screen, the system stores the number of the screen which is normally displayed next. This screen sequence can be either linear or cyclic. From within a screen chain, you can even call another screen chain and, after processing it, return to the original chain. You can also override the statically-defined next screen from within the dialog modules of the ABAP program.

GUI status

Each screen has a GUI status. This controls the menu bars, standard toolbar, and application toolbar, with which the user can choose functions in the application. Like screens, GUI statuses are independent components of an ABAP program. You create them in the ABAP Workbench using the Menu Painter.

Screen Painter SE51

You can create a screen from the Screen Painter initial screen or from the object list in the Object Navigator.

Screen Attributes

Screen attributes enable the system to assign and process a screen. You can set the following screen attributes:

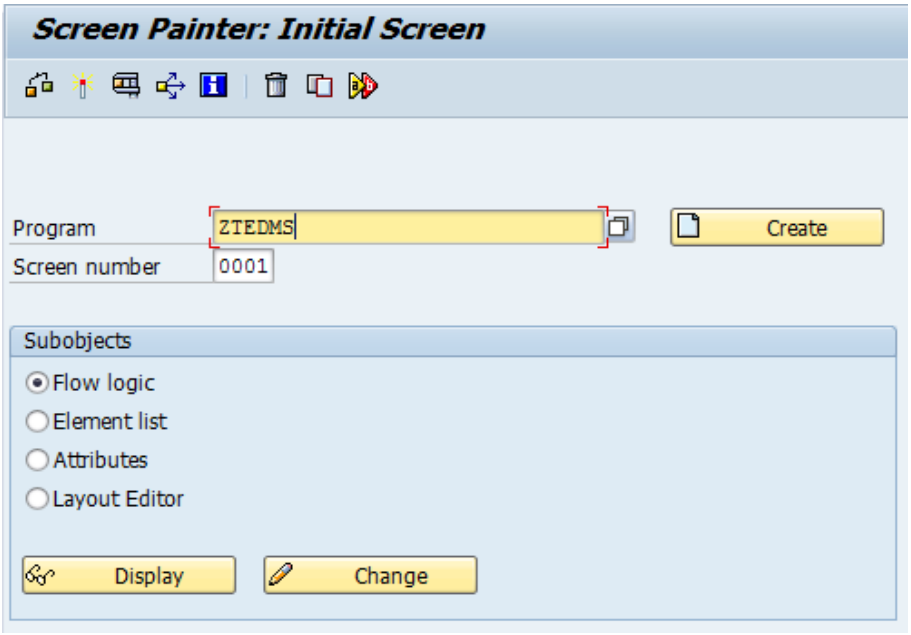
Attribute	Description and ergonomic guidelines
Program	Name of the module pool to which the screen belongs.
Screen number	Identifies a unique name up to 4 numbers long.
Original Language	Identifies a screen's maintenance language. When you create a screen, the system sets this value to the module pool's maintenance language.
Short description	Describes a screen's purpose.
Original language	Maintenance language for the screen. The system sets this attribute to the same original language as the module pool when you create the screen.
Development class	Identifies the development class the screen belongs to.
Last changed/ Last generated	Date and time that the screen was last changed or generated
Screen type	
Normal	If you set this option, the screen is flagged as a normal screen. This is the default setting.
Sub-screen	Identifies the screen as a sub-screen.
Modal dialog box	Identifies a specialized interface for display of lists in a dialog box.
Selection screen	Automatically-created screen. Selection screens are used to get values from the user at the beginning of a program. This data is

	used to restrict database selections. The system sets this attribute automatically.
Settings	
Hold data	The system only supports the *Hold data*, *Set data*, and *Delete data* functions (under System ® User profile) on the screen if this option is selected. The system automatically redisplays the user's last entries from the screen when the user displays the screen a second or subsequent time.
Switch off runtime compression	If you set this option, the screen is not compressed at runtime. Ergonomic guideline: You should not use this option, since empty lines may appear on the screen if you hide fields dynamically at runtime. When gaps occur, users typically need longer to process the screen.
Hold scroll position	Use this option to specify whether the vertical and horizontal scroll positions should be retained for a screen. If you set the attribute, the scroll position is retained when the user returns to the screen after processing another screen. This also applies if the length or width of the screen changes, if other sub-screens are used, or if the cursor is placed outside the visible area. This setting is intended for large screens on which the scroll position has previously been lost as a result of certain actions.
Other attributes	
Next screens	Number of the next screen to be displayed, assuming that the screen sequence is processed statically.
Cursor position	Element on which the cursor is positioned when the screen is first displayed. If you leave this field blank, the system positions the cursor on the first input field.
Screen group	Four-character ID for a group of logically-related screens
Lines/columns occupied	Size of the area occupied by screen elements.
Maintenance lines/columns	Screen size in lines and columns. The size of a screen is measured relative to the position of its top left-hand corner.

Screen Painter: Initial Screen

To start the Screen Painter, choose the corresponding pushbutton on the initial screen of the ABAP Workbench or enter Transaction **SE51**. From here, you can:

- Create new screens.
- Test an existing screen.
- Create new components for an existing screen.



The *Object components* text box lists the different screen component views. Each view lets you edit a different aspect of a screen.

If you choose...	**You can...**
Layout Editor	Maintain a screen's layout
Element list	Maintain the ABAP Dictionary or program fields for a screen and assign a program field to the OK_CODE field in the Screen Painter.
Screen attributes	Maintain a screen's attributes
Flow logic	Edit a screen's flow logic

Once you are within a particular view, you can use the Screen Painter's *Goto* menu to enter the other views.

Screen Painter Concepts

The Screen Painter is a ABAP Workbench tool that allows you to create screens for your transactions. You use it both to create the screen itself, with fields and other graphical elements, and to write the flow logic behind the screen.

Screen Painter Architecture

You use the Screen Painter to create and maintain all elements of a screen. These are:

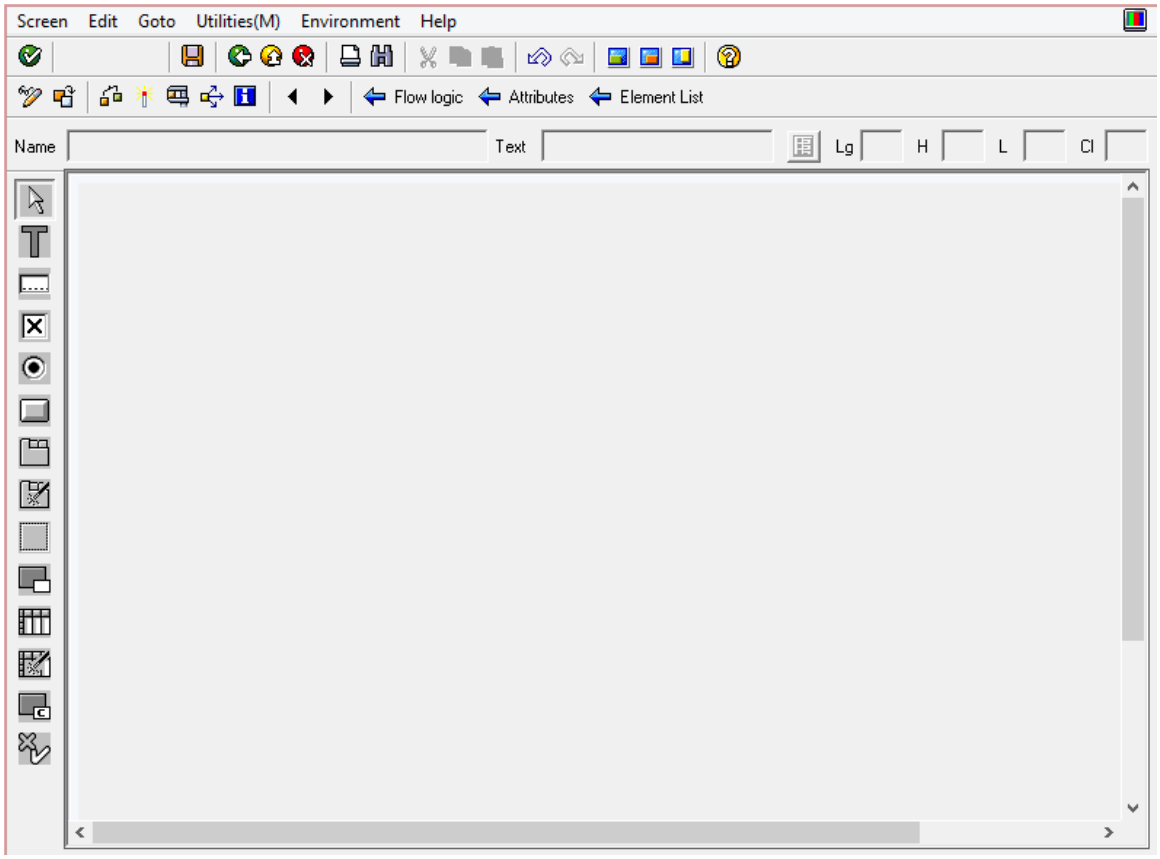
Screen Attributes	Describe a screen object in the SAP system. Screen attributes include the program the screen belongs to and the screen type.
Screen layout	Screen elements are the parts of a screen with which the user interacts. Screen elements include checkboxes and radio buttons.
Fields	Correspond to screen elements. Fields are defined in the ABAP Dictionary or in your program.
Flow logic	Controls the flow of your program.

Starting the Layout Editor

To start the graphical layout editor from the initial screen of the Screen Painter:

- Enter a program name and a screen number.
- Select Graphical Screen Painter and choose Continue.
- Choose Layout Editor from the list of Components on the initial screen.
- Choose Change. The system opens your screen in the full screen editor.

Components of the Layout Editor



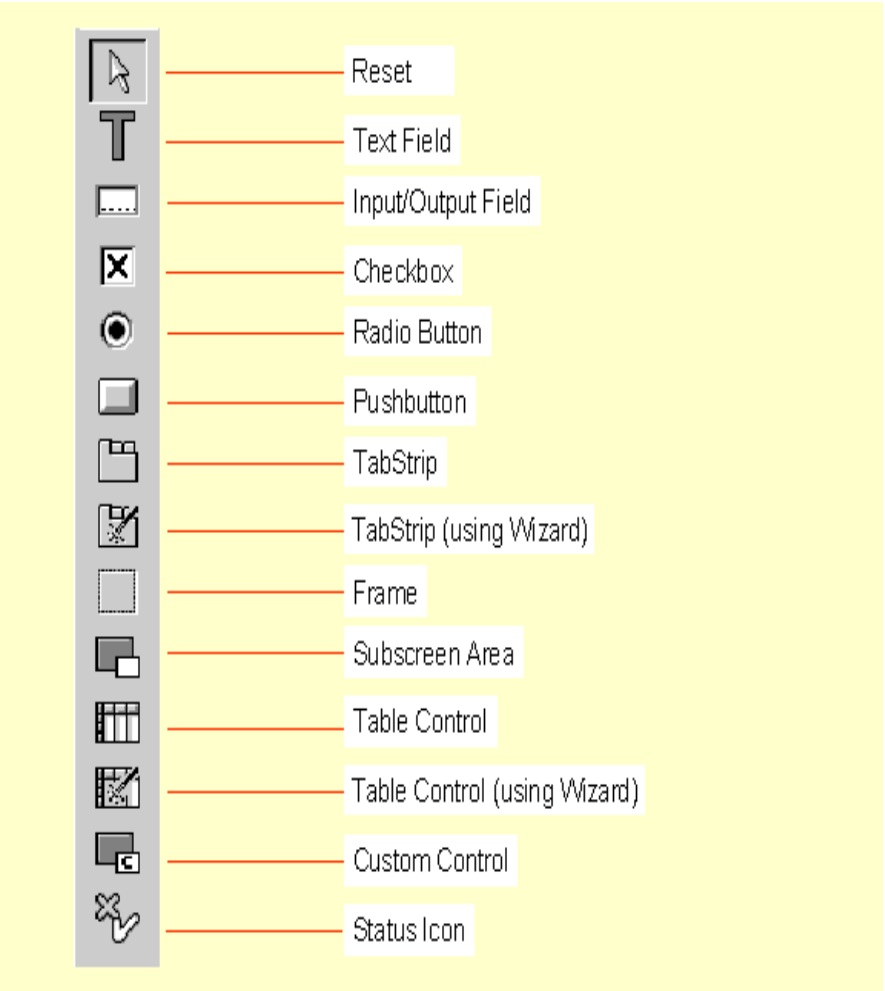
- **Element palette**, for creating screen elements. You can **drag and drop** these onto the screen. For further details about the individual elements

- **Work area**. This is the main part of the full screen editor, in which you design the screen itself.

- **Element bar.**

When you select a screen element, the major attributes associated with the element appear in this line. You can also change these attributes in the corresponding field.

Element Pallet (Graphical Screen Painter)



Element Types: Overview

You can use the following screen elements in the Screen Painter (both graphical and alphanumeric):

Text Fields

Text fields provide labels for other elements. Text labels (sometimes called keywords) are display-only elements: neither the user nor the ABAP program can modify them at runtime. Text elements appear in a fixed position on the screen.

Text elements can also include literals, lines, icons, and other static elements. They can include all alphanumeric characters. However, you cannot begin a text with an '**_**' (underscore) or a '?' (question mark). (question mark). If you use a text to label a radio button or checkbox, the text must have the same element name as the element it labels.

If the text consists of several words, join the words together with underscores. The underscores allow the system to recognize the words as a unit. They are replaced by spaces at runtime.

Input/Output Fields

Input/output fields are sometimes called templates. You use them for entering and displaying data. To define the size of an entry element, enter underscore characters in the *Text* field as follows:

You can also use any other characters to format your template. For numeric values, you can define a comma (,) as the separator and a period (.) as the decimal point. As the last character of the template, you can set a **V** as place holder for signs.

Input/Output fields have no text labels. To assign a label to one, place a text field next to it.

Input/output fields can have a maximum defined length of 255 characters. The visualized length is also a maximum of 255 characters.

Dropdown List Box 

This is a special type of input/output field. Dropdown list boxes contain a list of entries from which the user can choose one. You cannot type an entry into the text field of a list box.

Checkbox Elements

Use checkbox elements to allow a user to select one or more options in a group. Program control is not immediately passed back to a work process on the application server. Further selections are possible until the user pushes a button or chooses a menu function.

Radio Button Elements

Radio buttons are exclusive selection buttons that belong to a logical group. If a user selects one, the other buttons in the group are automatically deselected. You must both add the buttons and define them as a radio-button group in order to make their selection mutually exclusive.

When a user selects a radio button, control is not passed back to a work process on the application server immediately. As with checkboxes, further selections are possible until the user either presses a pushbutton or selects a menu function.

Pushbutton Elements

You use pushbuttons to trigger a particular function. When a user chooses one, the system sends the associated function code to the underlying ABAP program. At that point, control

automatically returns to a work process on the application server that processes the PAI (Process After Input) module.

There is currently no link to the interface defined in the Menu Painter. The system does not check whether the selected function codes correspond to a valid status.

A pushbutton label can be simple text or it can be dynamic text that changes at runtime. You must define fields in your program for dynamic text.

Boxes

A box groups a set of elements that belong together - for example, a radio button group. Boxes are only for display. Boxes provide visual emphasis but have no other function.

The top edge of a box normally contains a left-justified header. This header can be either a text field or an output field. If the header is a field element that is empty at runtime, the lines of the box are closed.

Tab-strip Controls

Tab-strip controls are complex graphical elements.

Tab-strip Control Wizard

The Tab-strip Control Wizard takes you step by step through the procedure for creating a working tab-strip control.

Sub-screen Areas

Sub-screen elements are rectangular areas of a screen reserved for displaying other screens at runtime. A sub-screen area cannot include any other screen elements. You use sub-screens to include other screens within your main program.

To use a sub-screen, define a second screen that appears in a sub-screen area of the first screen.

Table Controls

Table controls are also complex graphical elements.

Table Control Wizard

The Table Control Wizard takes you step by step through the procedure for creating a working table control.

Custom Container

You can use the custom container to embed one or more controls within a screen area. The custom container, like other screen elements, supports resizing and compression. In the Screen Painter, a rectangular area appears as a placeholder for one or more controls. The control itself does not appear on the screen until runtime.

You can define in the element attributes whether you want the control to be resizable.

Status Icons

Status icons are output fields that contain an icon. The icon is specified at runtime. You use icons to indicate statuses in your application. Icons are predefined in the system and are each two to four characters long.

Project Profile

The name of the Project is ‘ALV Report Generation’. The purpose of the project is to generate an ALV report for the given database(s).

The ALV report generation system designed as a part of this project is a robust application and can be used by any university for generating its set of reports.

The project uses SAP as the ERP software.

Availability, Efficiency, Accuracy, Time bound Updates and Confidentiality are some of the reasons that define the importance of a SAP based Employee leave management system.

The project uses SAP as the ERP software. The projects objective is to generate ALV report as per the user’s demand for the university database.

The user can view the list of all the departments in the university.

The project helps the university in understanding the details of the courses it offers.

This project helps organizations keep track of their faculty and the subjects they are teaching as well.

Working with SAP allows for easy access and availability to the program anywhere and everywhere.

Constraints

General Constraints

- Recovery of data after a system crash will be possible only if backups are taken at regular intervals.
- Manual interfaces cannot be fully avoided. Documented proofs like dates etc. will have to be verified by the concerned staff before entering it into the computerized system
- Entry of the record has to done manually.

Hardware Constraints

The performance of the system will be dependent on the network conditions like network congestion, bandwidth etc. The primary memory (RAM) and the secondary memory (Hard Disk Space) requirement of the system at the client end will be the same as that required by the web browser and the operating system. At the server end memory requirements will be that of the server software (Operating system, Database Software, etc.) and the space required to store the data. The space required to store the data would increase as more and more records are added to the system.

Hardware – Software Interface

An Internet Web Server, running SAP logon. The application software is developed in SAP R/3 using ABAP/4 language. The backend database is ORACLE. The Client systems with internet facility equipped with web browser will be able to access the system.

Memory Constraints

No memory constraints are applicable. A normal memory configuration is more than sufficient.

Hardware Requirements

SR No.	HARDWARE COMPONENTS	MINIMUM REQUIREMENTS
1.	MOTHERBOARD	965 –GV CHIPSET
2.	HARD DIDSK	2 GB
3.	RAM	512 MB

Software Requirements

SR No.	SOFTWARE COMPONENTS	MINIMUM REQUIREMENTS
1.	OPERATING SYSYEM	WINDOWS 8/7/XP
2.	APPLICATION	SAP ECC 6.0
3.	ENVIRONMENT	SAP Empowered Wifi System

NOTE: For running any SAP application there should be collaboration of the company with SAP AG.

Software Requirement Specification

Non-Functional Requirements:

1. Usability
The system ability to provide search help is user friendly.

2. Performance
The client and server interaction and server resource uses is very efficient.

3 Modifiability or Extensibility
The system ability to add (unspecified) future functionality.

4 Reusability
The system has the ability to be reused in future system.

Assumptions and Dependencies
• It is assumed that the user is familiar with the basic computer fundamentals.
• Timely backup of data should be taken to avoid data loss in case of system crash.
• Floppies and other removable media should be scanned for viruses before use.
• Proper configuration of the client, database server and network is necessary for the system to function as intended.
• It is assumed that the maintenance of the database will be assigned to the authorized person only.

Feasibility Study

The main objective of the feasibility study is to treat the technical, Operational, logical and economic feasibility of developing the computerized system. All systems are feasible, given unlimited resources and infinite time. It is both necessary and prudent to evaluate the feasibility of the project at system study phase itself. The feasibility study to be conducted for this project involves.

- Technical Feasibility
- Operational Feasibility
- Economic Feasibility
- Legal Feasibility

Technical Feasibility

Technical feasibility includes Risk Resources availability and technologies. The management provides latest hardware and software facilities for the successful completion of the projects. With these latest hardware and software support the system will perform extremely well. The system is available through Internet.

Operational Feasibility

In the existing system it is very difficult to maintain and update huge amount of information. The development of the system was started to help Functional and Technical Team. This system will handle the request in a better way and make the process easier thus, it is sure that the system developed is operationally feasible.

Economic Feasibility

In the economic feasibility the development cost of the system is evaluated weighing it against the ultimate benefit derived from the new system. It is found that the benefit, from the new system would be more than the cost and time involved in its development.

Legal Feasibility

In the legal feasibility it is necessary to check that the software we are going to develop is legally correct which means that the ideas which we have taken for the proposed system will be legally implemented or not. So, it is also an important step in feasibility study.

THE SOFTWARE PROCESS MODEL USED

THE INCREMENTAL MODEL

The incremental model is an intuitive approach to the waterfall model. Multiple development cycles take place here, making the life cycle a —multi-waterfall‖ cycle. Cycles are divided up into smaller, more easily managed iterations. Each iteration passes through the requirements, design, implementation and testing phases. A working version of software is produced during the first iteration, so that we have working software early on during the software life cycle.

Advantages

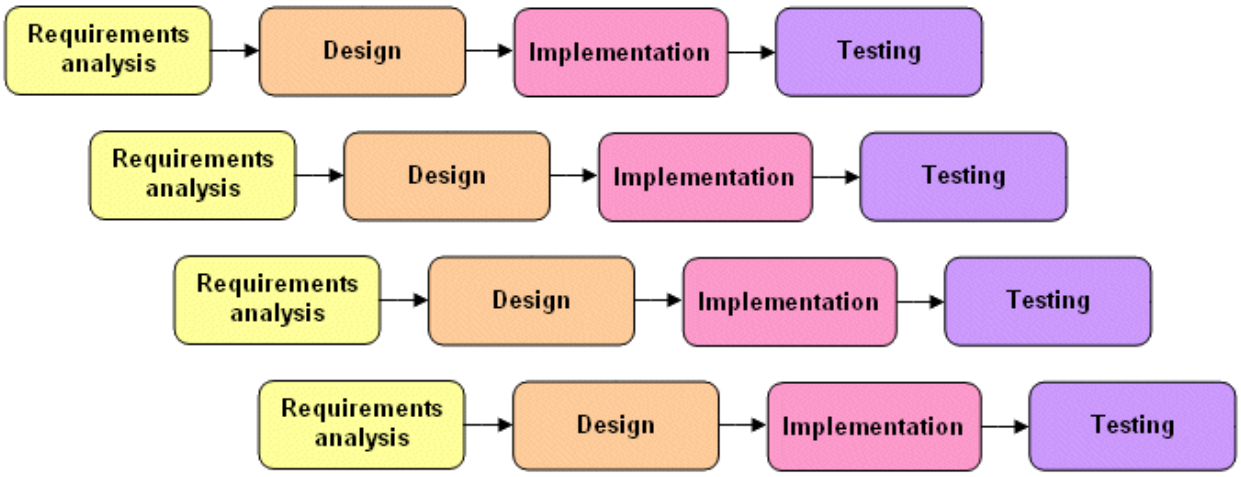
(1) It is useful when staffing is unavailable for the complete implementation.

(2) Can be implemented with fewer staff people.

(3) If the core product is well received then the additional staff can be added.

(4) Customers can be involved at an early stage.

(5) Each iteration delivers a functionally operational product and thus customers can get to see the working version of the product at each stage.



PROJECT STRUCTURE

ZTEST_ DEPARTMENT

- **DEPT_NAME**
- BUILDING
- BUDGET

ZTEST_COURSE

- **COURSE_ID**
- TITLE
- *DEPT_NAME*
- CREDITS

ZTEST_ INSTRUCTOR

- **ID**
- NAME
- *DEPT_NAME*
- SALARY
- COURSE_ID

SOURCE CODE

```
*&---------------------------------------------------------------------*
*& Report  ZTEST_DIKSHA_PROJECT
*&
*&---------------------------------------------------------------------*
*& Created on: 24-06-2016
*& Author: Diksha
*&---------------------------------------------------------------------*

REPORT  ZTEST_DIKSHA_PROJECT.

TABLES: ZTEST_DEPARTMENT, ZTEST_COURSE, ZTEST_INSTRUCTOR.

TYPE-POOLS: slis.  " SLIS contains the ALV data types

TYPES: BEGIN OF ty_dept,
       dept_name TYPE ZTEST_DEPARTMENT-DEPT_NAME,
       budget TYPE ZTEST_DEPARTMENT-BUDGET,
      END OF ty_dept,

      BEGIN OF ty_course,
       course_id TYPE ZTEST_COURSE-COURSE_ID,
       title TYPE ZTEST_COURSE-TITLE,
       dept_name TYPE ZTEST_COURSE-DEPT_NAME,
      END OF ty_course,

      BEGIN OF ty_instructor,
       name TYPE ZTEST_INSTRUCTOR-NAME,
       dept_name TYPE ZTEST_INSTRUCTOR-DEPT_NAME,
       salary TYPE ZTEST_INSTRUCTOR-SALARY,
       course_id TYPE ZTEST_INSTRUCTOR-COURSE_ID,
      END OF ty_instructor,

      BEGIN OF ty_out,
       sel,
       dept_name TYPE ZTEST_DEPARTMENT-DEPT_NAME,
       budget TYPE ZTEST_DEPARTMENT-BUDGET,
       course_id TYPE ZTEST_COURSE-COURSE_ID,
       title TYPE ZTEST_COURSE-TITLE,
       name TYPE ZTEST_INSTRUCTOR-NAME,
       salary TYPE ZTEST_INSTRUCTOR-SALARY,
      END OF ty_out.
```

```
DATA: wa_dept TYPE ty_dept,
    wa_course TYPE ty_course,
    wa_instructor TYPE ty_instructor,
    wa_out  TYPE ty_out,
    it_dept TYPE STANDARD TABLE OF ty_dept,
    it_course TYPE STANDARD TABLE OF ty_course,
    it_instructor TYPE STANDARD TABLE OF ty_instructor,
    it_out  TYPE STANDARD TABLE OF ty_out.

*DATA: wa_fcat_out TYPE slis_fieldcat_alv,
*     it_fcat_out TYPE slis_t_fieldcat_alv,
*     wa_layout   TYPE slis_layout_alv,
*     wa_top      TYPE slis_listheader,
*     it_top      TYPE slis_t_listheader.

*DATA: IT_DEPARTMENT LIKE ZTEST_DEPARTMENT OCCURS 0 WITH HEADER LINE,
*     IT_COURSE LIKE ZTEST_COURSE OCCURS 0 WITH HEADER LINE,
*     IT_INSTRUCTOR LIKE ZTEST_INSTRUCTOR OCCURS 0 WITH HEADER LINE,
*

DATA: it_fieldcat  TYPE slis_t_fieldcat_alv, "used for internal table
    wa_fieldcat  TYPE slis_fieldcat_alv. "used for work area

PARAMETERS: WA_DNAME LIKE ZTEST_DEPARTMENT-DEPT_NAME MATCHCODE OBJECT
ZTEST_DNAME.

PERFORM get_budget .
PERFORM get_course .
PERFORM get_instructor .
PERFORM prepare_output .
PERFORM display_alv .

FORM get_budget .

SELECT dept_name budget
FROM ZTEST_DEPARTMENT INTO TABLE it_dept
WHERE DEPT_NAME = WA_DNAME .

ENDFORM .

FORM get_course .

SELECT course_id title dept_name
FROM ZTEST_COURSE INTO TABLE it_course
WHERE DEPT_NAME = WA_DNAME .
```

```
ENDFORM .

FORM get_instructor .

SELECT name dept_name salary course_id
FROM ZTEST_INSTRUCTOR INTO TABLE it_instructor
WHERE DEPT_NAME = WA_DNAME .

ENDFORM .

FORM prepare_output .

 LOOP AT it_dept INTO wa_dept.
   wa_out-dept_name = wa_dept-dept_name.
   wa_out-budget = wa_dept-budget.

   LOOP AT it_course INTO wa_course
   WHERE dept_name = wa_dept-dept_name.
     wa_out-course_id = wa_course-course_id.
     wa_out-title = wa_course-title.

     LOOP AT it_instructor INTO wa_instructor
     WHERE dept_name = wa_dept-dept_name
       AND course_id = wa_course-course_id.
         wa_out-name = wa_instructor-name.
         wa_out-salary = wa_instructor-salary.

         APPEND wa_out TO it_out.
         CLEAR: wa_out, wa_instructor.

     ENDLOOP.
     CLEAR wa_course.
   ENDLOOP.
    CLEAR wa_dept .
 ENDLOOP.

ENDFORM .

*SELECT * FROM ZTEST_DEPARTMENT INTO CORRESPONDING FIELDS OF TABLE IT_DEPARTMENT
WHERE DEPT_NAME = WA_DNAME.

* Build field catalog *

FORM display_alv .
```

```
  wa_fieldcat-fieldname  = 'DEPT_NAME'.    " Fieldname in the data table
  wa_fieldcat-seltext_m  = 'Department Name'.   " Column description in the output
  APPEND wa_fieldcat TO it_fieldcat.

  wa_fieldcat-fieldname  = 'BUDGET'.
  wa_fieldcat-seltext_m  = 'Department Budget'.
  APPEND wa_fieldcat TO it_fieldcat.

  wa_fieldcat-fieldname  = 'COURSE_ID'.
  wa_fieldcat-seltext_m  = 'Course ID of Courses'.
  APPEND wa_fieldcat TO it_fieldcat.

  wa_fieldcat-fieldname  =  'TITLE'.
  wa_fieldcat-seltext_m  = 'Title of the Course'.
  APPEND wa_fieldcat TO it_fieldcat.

  wa_fieldcat-fieldname  = 'NAME'.
  wa_fieldcat-seltext_m  = 'Instructor(s) Name'.
  APPEND wa_fieldcat TO it_fieldcat.

  wa_fieldcat-fieldname  = 'SALARY'.
  wa_fieldcat-seltext_m  = 'Instructor(s) Salary'.
  APPEND wa_fieldcat TO it_fieldcat.

* Pass data and field catalog to ALV function module to display ALV list *

  CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
    EXPORTING
     it_fieldcat   = it_fieldcat
    TABLES
     t_outtab      = it_out "internal table with the data to be output
    EXCEPTIONS
     program_error = 1
     OTHERS        = 2.

ENDFORM .
```

CONCLUSION

The ALV report generation management system offers a quick and easy way to generate ALV reports as per the project's need. We conclude that it is:

- Robust
- User Friendly
- Quick
- Secure and Safe.

FUTURE SCOPE

The developed system is flexible and changes can be made easily. The system is developed with an insight into the necessary modification that may be required in the future. Hence the system can be maintained successfully without much rework.

One of the main future enhancements of our system is to add auto generated prints of different letters issued during the process

BIBLIOGRAPHY

BOOKS

- BC – ABAP Programming Release 4.6C
- Sams Teach Yourself ABAP in 21 Days

WEBSITE REFERENCES

- http://help.sap.com
- http://www.abapprogramming.net
- http://www.tutorialspoint.com/sap/sap_programming_language.htm
- http://www.saphub.com/abap-tutorial/