International Journal of Computer Science and Engineering (IJCSE)
Vol.1, Issue 1 Aug 2012 38-44
© IASET

International Academy of Science, Engineering and Technology
Connecting Researchers; Nurturing Innovations

SAP ABAP –DEVELOPING PROGRAMS IN A SIMPLE WAY

S. KISHOR KUMAR, M. SURESH KUMAR & V. V. S. N. SASTRY
Dept of Computer Science, GITAM Institute of Science, GITAM University, Visakhapatnam, India

ABSTRACT

In the fourth generation languages one of the languages is SAP-ABAP. ABAP is very easy to learn. However to start programming with ABAP, programmers are expected to have some basic knowledge of data bases, other programming knowledge and preferably also some basic principles of Object oriented concepts. In SAP many modules are available especially for technical people. SAP-ABAP is one of the modules.

KEYWORDS: ABAP, SAP, Education, Industry, Language.

INTRODUCTION

ABAP stands for Advanced Business Application Programming and it is a high level programming language used in SAP for the development and other customization processes. ABAP is a business programming language. ABAP is a high level programming language created by SAP. ABAP is considered one of many fourth generation languages and it was developed in early 1980s. ABAP is an acronym for 'Advanced Business Application Programming'.

ABAP is used by SAP to develop SAP R/3 applications. ABAP programs are stored in a database unlike with other popular programming languages like C++ or JAVA where the programs are stored as files on the computer. The ABAP environment (syntax checking, compiling and executing) is a part of SAP Basis component. All SAP data and programs run on an SAP system. A SAP system consists of one or more application servers, data base and client terminals. A typical SAP landscape will have three systems for Development, Testing and Production. A data dictionary is a centralized storage location for information about the data that is stored in a database. This information is often called “metadata” (data about data). SAP’s data dictionary is called the ABAP Dictionary.

ROLE OF ABAPER

Customizing: Modifications and Enhancements.

Reusability: Reuse the existing components.

Develop new application: New applications can be built using the components of existing applications.

THE ABAP DATA DICTIONARY

A data dictionary provides answers to questions such as:

1. What data is contained in the database?

2. What are the attributes of this data: name, length, format, etc.?
3. What relationships exist among different data objects?
4. Enforces data integrity.
5. Manages data definitions without redundancy.
6. Is tightly integrated with the rest of the ABAP Workbench.

When data integrity rules are defined in the ABAP Dictionary, the system automatically prevents the entry of invalid data. Defining the data integrity rules at the dictionary level means they only have to be defined once, rather than in each program that accesses that data. The following are examples of data lacking integrity:

1. A date field with a month value of 13.
2. An order assigned to a customer number that does not exist.

Additionally, the system provides easy navigation between development objects and dictionary definitions. For example, if you can double-click on the name of a dictionary object in your program code, the system will take you directly to the definition of that object in the ABAP Dictionary. When a dictionary object is changed, a program that references the changed object will automatically reference the new version the next time the program runs. Since ABAP is interpreted, it is not necessary to recompile programs that reference changed dictionary objects.

ABAP REPORTING

ABAP is a programming language created and used by SAP for the development of application programs, including: Reports, Module Pool Programming, Interfaces, Data conversions, User Exits & BADI. All of R/3s applications and even parts of its basis system were developed in ABAP. ABAP is an event-driven programming language. User actions and system events control the execution of an application. ABAP used to be called ABAP/4. The “4” stands for “fourth generation language”.

ABAP WORKBENCH

The ABAP Workbench is used by SAP for the development of standard and custom application software. The ABAP Workbench is also used to create dictionary objects. It consists of the following components:
ABAP Editor is used to maintain programs.

1. ABAP Dictionary is used to maintain Dictionary objects.
2. Repository Browser is used to display a hierarchical structure of the components in a package.
3. Menu Painter is used to develop graphical user interfaces, including menu bars and toolbars.

4. Screen Painter is used to maintain screen components for an online program.
5. Repository Information System contains information about development and runtime objects, such as data models, dictionary types and table structures, programs, and functions.
6. Test and Analysis Tools, such as the Syntax Check and the Debugger.
7. Function Builder, which allows you to create and maintain function groups and function modules.
8. Data Modeler, a tool which supports graphical modeling.
9. Workbench Organizer, which maintains multiple development projects and manages their distribution.

REPORTING

Report programs produce lists and can be divided into conventional reports and interactive reports. Conventional reports do not allow interaction by the user; therefore, the basic list contains extensive information that the user must often sort through to find the relevant data. Interactive reports allow interaction by the user; therefore, the user can produce secondary, detailed lists of the basic list by choosing the relevant data and requesting more information.

MODULE POOL PROGRAMMING

Module pool programming (or online programming) involves the creation of a module pool (a collection of ABAP modules) and one or more screens. The modules are called by the screen processor during program execution. Batch input processing is used to ensure the safe transfer of data into the SAP system. This process is an automatic, protected data transfer to the SAP system which uses SAP transactions to validate data as it populates the SAP database. ABAP contains statements which conform to CPI-C standards (Common Program Interface-Communications). These are used for programming communications programs. ABAP can read and write sequential data sets.

ONLINE PROGRAMMING

In ABAP, there are two different types of programs: Report programs and online programs. Online programs (also called module pool programs, dialog programs, or transactions) do not produce lists. These programs are a collection of screens, their Flow Logic, and the code within the main ABAP program. The term “screen” refers to the actual, physical image that the users sees. The term “flow logic” refers to the code behind the screens (i.e., the logic that initializes screens and responds to a user’s requests on the screens). Each screen has its own Flow Logic. The term “dynpro” (dynamic program) refers to the combination of the screen and it’s Flow Logic. Online Programs can only be started using a transaction code, in which an initial screen is defined.

When creating an online program, many tools will be used within the ABAP Development Workbench: Screen Painter, ABAP Editor, Menu Painter, ABAP Dictionary, and Object Navigator. The

Screen Painter is used to maintain all components of the screens. The ABAP Editor is used to maintain the main ABAP program. This program consists of data declarations and modules. The ABAP modules contain the main processing logic of the online program and are “called” from within the Flow Logic of the screens. The Menu Painter is used to maintain the graphical user interface (GUI) which consists of the function key assignments, standard toolbar, application toolbar, menu bar, and title bar. The ABAP Dictionary is used to link dictionary field definitions to screen fields as well as to create program work areas (with the TABLES statement). One should always use the Object Navigator to create online programs because the system will automatically maintain an online program’s sub-objects and user will be able to see the hierarchy list of these sub-objects. From this hierarchy list, one will be able to branch to the Screen Painter, ABAP Editor, Menu Painter, and ABAP Dictionary.

SCREEN COMPONENTS

The Screen Painter is used to maintain all components of a screen: Screen attributes, screen layout, field attributes, and Flow Logic. The screen attributes refer to the basic characteristics of a particular screen. For example, the type of screen and the size of the screen. The screen layout refers to the physical appearance of the screen. Some of the possible elements a screen can contain include: Text fields, input/output templates, radio buttons, check boxes, frames, and pushbuttons. The screen layout is maintained in the Full screen Editor of the Screen Painter. The field attributes refer to the characteristics of each field (element) for a particular screen. For example, the type of field and the length of the field. Field attributes are maintained in the Field List of the ScreenPainter.

The Flow Logic refers to the code behind the screens. It is important to remember that Flow Logic commands are different from ABAP commands (e.g., IF … ENDIF is not valid in Flow Logic). Each screen has its own Flow Logic which is divided into two main events:

- PROCESS BEFORE OUTPUT (PBO) - event processed before the screen is displayed.
- PROCESS AFTER INPUT (PAI) - event processed after the user has invoked a function code (e.g., clicked on a push button) or pressed the ‘Enter’ key.

Each screen must be activated. If a change is made to any component of a screen, the screen must be re-activated before execution.

SCREEN LAYOUT

Some of the elements that can be “painted” on a screen in the Full screen Editor are Text fields, Input/output template, Radio buttons, Check boxes, Frame, Pushbutton, Tab strip Controls.

Icons can also be used on a screen. Icons can be used on a pushbutton or on a text element. The Full screen Editor supports two modes: Graphical and alphanumeric. The difference between these two modes is the interface: The graphical mode (available only on Windows 95, Windows NT, and UNIX/Motif platforms) is the more user-friendly mode with a drag-and-drop “painting” interface. With the alphanumeric mode, one can “paint” screens using the keyboard and menu paths. To maintain the

Full screen Editor mode, use the "Utilities >Settings and then click on the Screen Painter tab. Here one can check or uncheck the Graphical layout editor checkbox.

Object Oriented Programming (OOP) is a programming paradigm that uses the concept of objects that can encapsulate data and its related functions on data.

Some of the features of OOP are: Encapsulation, Abstraction, Inheritance and Polymorphism.

The basic building blocks of any OOP language are Classes. The entire principles of object oriented programming can be understood if one can clearly understand the concept of Classes, Instances and Methods in detail.

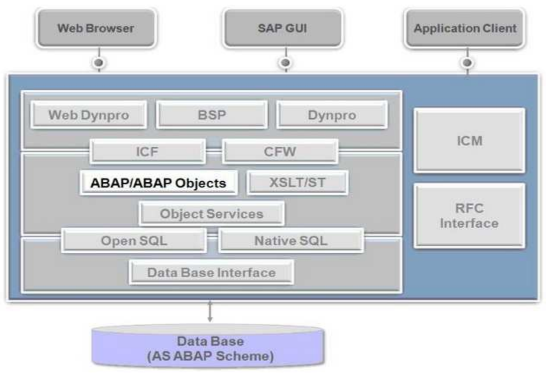

Fig. 1 ABAP Objects

ABAP: A HYBRID PROGRAMMING LANGUAGE

ABAP has evolved as a hybrid programming language. ABAP runtime supports both procedural programming as well as ABAP Objects. One might know that ABAP is an event driven programming language and will now know that ABAP is an object oriented programming language. Although applications with pure Object Oriented ABAP can be designed, most applications still use a mix of procedural and object oriented approach of ABAP. Thus one will have the advantage to take the best practices of both paradigms.

Control Frameworks is like a technology framework that controls the desktop applications. In a SAP R/3 system one can use ABAP to control one's desktop applications. This means that one can build up an application with custom controls like a Tree control, a docking control, a split screen etc. SAP is not all about having form like screens, press enter and navigate. One can have desktop applications built that allow features like drag drop and many more and have them connected to the SAP R/3 back end. The application logic runs at the back end that controls the front end. The SAP GUI acts as a custom container for the controls on the front end. These controls could be a Active-X or a JAVA Beans control.

ABAP OBJECTS & CONTROL FRAME WORKS

While using a control in ABAP program, there are mainly 2 objects involved. One instance of this control is created in the ABAP program and another one is created on the presentation server (your desktop PC). The ABAP program communicates to the presentation server using the CFW and executes the operation on the presentation server where the screen and the controls can be seen. When an action is performed on the presentation server (like a double click) this information is sent via CFW to the application server where the program handles the logic to handle 'Double Click'.

Now, these controls are not communicated in every step between backend application and presentation server. This is to improve the performance. Otherwise it will take few seconds for each operation which is definitely not desired. This communication is done via 'Automation Queue'. These method calls are buffered in this automation queue and are sent to front end once you reach a synchronization point. At this point the queue is flushed. The control on the presentation server can have events (like double click). All the events are not sent to the application server from the presentation server. This is because most of them might not be relevant for a particular program. If events are to be handled, a filter must be constructed and then this event will be sent to the application server where it can be handled.

DYNAMIC PROGRAMMING

With the increasing complexity of the applications, sometimes it becomes necessary to make certain decisions only at runtime. Some examples could be:

The name of the DB table (from which the data is to be selected) is known at runtime only

1. For any kind of DB operation (select, insert, update or delete) the name of the table is known at runtime only.
2. If the DB table can change, so has to the structures / internal tables.
3. Function module / Method name to be called is known at runtime only.
4. Class to be instantiated is known at run time only.

In all above (and many more such cases), dynamic coding is required so that the code is neat, clean, flexible and robust.

REMOTE FUNCTION CALL

An RFC is used to access (read / write) data from remote systems. Within SAP landscapes, RFC continues to serve as basis of nearly all major integration points like: ALE, IDocs, Portal, etc. In case of Asynchronous (ARFC) call, the system passes the control back to the calling program immediately. The RFC call is executes in a different work process or in a new internal session.

RFC calls are of two types:

1. Asynchronous call without response
2. Asynchronous call with response

In case of a RFC, a program can be made to wait until it receives some response from the calling function module and then process the result of the RFC call.

In case of a RFC without response, any error message or system exceptions from the executing function module cannot be handled in the calling program. Synchronous RFCs (SRFC) are most commonly used RFC. SRFC is ideal choice when the results of the function module are needed immediately after the call. The SRFC call is passed immediately to the remote system and the caller program is halted until the response from the function module is not received. The function group of the function module is loaded into the internal session of the target system and the context is retained until the RFC session is closed.

Thus, if repeated calls of the different function modules of the same function group and destination are made, then the global data of the function group can be accessed collectively. This makes a SRFC a great choice when data is repeatedly required from a target system.

An RFC is used when you want to access (read / write) data from remote systems. Within SAP landscapes, RFC continues to serve as basis of nearly all major integration points like: ALE, IDocs, Portal, etc. The main task of the RFC run time is to facilitate the execution of function modules in remote systems. This remote system could be SAP system or an external program linked with one of the SAP-supplied RFC connector libraries. One of the first things needed to understand to work effectively is how RFC calls are actually communicated and executed.

CONCLUSIONS

SAP-ABAP is very useful language to computer Science engineering graduates. Now the industry needs the Computer graduates from this domain. In this domain many opportunities are available for Computer Science engineering graduates. Career and job opportunities in SAP ABAP are huge and also the monetary benefit one gets in this career is good. The entry would be a bit challenging as the demand for this job is high so is the number of candidates for this job. Apart from having very good opportunities for SAP ABAP people the technology would long last for a great period of time which gives security of job in the market.

REFERENCES

1. http://www.sapbrainsonline.com
2. http://www.abaplearning.com
3. http://www.help.sap.com
4. http://www.abapprogramming.net
5. http://www.saptechnical.com
6. http://www.passionateaboutsap.com
7. http://www.sapabaptutorial.com