

Ku Band Link Utilization Analysis System

A PROJECT REPORT

**Summer Project Work for Second Year
B.Tech CSE**

Under Guidance:

**Mr. A S Dobre
Manager(Programming)
Project ICE(Basis)
ONGC, New Delhi**

Submitted by:

**Asankhaya Sharma
Roll No. 03712
2nd Year
B Tech CSE**

**National Institute of Technology
Warangal, AP**



**OIL AND NATURAL GAS CORPORATION
LIMITED**

**PROJECT ICE, 14th floor, Core-4, South Tower, Scope
Minar,
Laxmi Nagar, Delhi-92. Tel : 22446171, Fax: 22446426.**

CERTIFICATE

This is to certify that **Mr. Asankhaya Sharma (Roll No. 03712)** a student of 4th semester of Bachelor of Computer Science & Engineering (B.Tech) at **National Institute of Technology, Warangal**, has been undergoing project training at Project ICE ONGC Delhi from May 2005 to June 2005.

He has successfully completed the project on “**KU Band Link Utilization Analysis System**” under the supervision and guidance of Mr. A S Dobre, Manager (Programming).

In the project he has worked in ABAP/4 (Advanced Business Application Programming) on SAP 4.6c ERP platform and covered various topics related to SAP R/3 Systems.

Countersigned
(Dr. S.S. Arora)
Chief Mathematician
ABAP Team Lead

(Mr. A.S. Dobre)
Manager (Programming)
ICE Basis

Acknowledgement

I take this opportunity to express my profound sense of gratitude and respect to all those who helped me throughout the duration of this project.

First and foremost I would like to express my thanks to **Mr. A.S. Dobre**, Manager (Programming) ONGC, Delhi for giving me an opportunity to undertake this project and providing crucial feedback that influenced the development of this project, which were critical in the deployment of the project without which I would not have been able to complete the project.

I would especially like to thank **Dr. S.S. Arora**, Chief Mathematician, ABAP Team Lead, for giving me an opportunity to undertake the project work in this esteemed concern.

Finally I would like to extend my profound thanks to all my esteemed colleagues who extremely helped me in specific areas of this project.

(Asankhaya Sharma)

Abstract

The well planned, properly executed industrial training helps a lot in inducting good work culture. It provides linkage between student and industry in order to develop awareness of industrial approach to problem solving based on broad understanding of process & operation industrial organization.

This project is based on the requirements of ONGC related to the maintenance of KU Band and to analyze its utilization by generating different types of critical and specific reports as per the needs of the management.

The system being diversified in nature demanded a three to four weeks period for understanding the ABAP/4 concepts, business processes and familiarizing with the platform of development for the purpose of completion of the project.

The project has been developed on SAP platform using ABAP/4 language providing a good user interface making its use simple. The project helped me in understanding the key development concepts related to programming in ABAP/4 for customer requirement. The study was exploratory in nature and hopefully laid the foundation for future work in this area.

The training undertaken in such an indigenous company gave me an opportunity to gain practical experience increasing my horizon of knowledge. I have tried to share some of my knowledge by way of this project report.

Preface

Title	Page no.
1. Company Profile	7
2. Enterprise Resource Planning	9-15
2.1 What is ERP?	9
2.2 ERP implementation in ONGC	10
2.3 Need for implementing SAP	13
3. System Application Product (SAP)	14-34
3.1 About SAP	14
3.2 Distributed Processing and Integration with R/3	16
3.3 R/3 Basics	18
3.4 R/3 Instance Provider Services	20
3.5 Architecture of SAP R/3 Systems	21
3.6 SAP Application Modules	24
3.7 Advantages of SAP Approach	26
3.8 R/3 Lock Concept	28
3.9 Hardware and Software Specifications	33
4. Basis Details	35
5. Advanced Business Application Programming	36-60
5.1 Introduction to ABAP Programming Language	36
5.2 Salient Features of ABAP	37
5.3 Types of ABAP Program	42
5.4 ABAP Development Workbench	33
5.5 Tools for Software Development	46
6. Ku Band Analysis System	61-80
6.1 Existing System	61
6.2 Objectives of Proposed System	62
6.3 Feasibility Study	63

	6.4 System Design	65
	6.5 Database Design	66
	6.6 Context Analysis	71
	6.7 Data Flow	72
	6.8 Testing Plan	79
7.	Conclusion	81
8.	Bibliography	82
9.	Annexure	83-116
	9.1 Sample Output	83
	9.2 Source Code	86

Company Profile

Born as a modest corporate entity within serene Himalayan settings on 14th August, 1956 as Commission, Oil and Natural Gas Corporation Limited (ONGC), has grown in to a full-fledged horizontally integrated upstream petroleum company.

Today, ONGC is a flagship public sector enterprise and India's highest profit making corporate, which has achieved the landmark of registering a net profit of Rs. 12983.05 crore in the year 2004-05. Since its inception ONGC has produced more than 600 million metric tones of crude oil and supplied more than 200 billion cubic meters of gas, thus Fuelling India's economy.

To achieve this sustained growth, ONGC decided to double the oil and gas reserves. In 45 years of operation, ONGC accreted six billion tones oil and oil equivalent reserves, and ONGC has drawn a plan to double these reserves in the next 20 years. Secondly, the global recovery factor of ONGC is of the order of 28 percent; the target is to raise this to the level of 40 percent, over the same 20 years.

Out of the six billion tones of oil and gas reserve accretion, four billion tones are expected to come from Offshore and Deep Waters. To improve recovery factor from the existing fields, ONGC is investing Rs. 2,000 crore in 15 Redevelopment Schemes.

ONGC has recognized the need to expand its business through profitable ventures related to petroleum and energy sectors by entering into joint ventures with other Indian and foreign companies.

ONGC-Joint venture group (ONGC-JVG) has been formed to give impetus to joint venture activities in areas other than E&P. ONGC-JVG is responsible for identification and developing new business opportunities with Indian and foreign companies in following areas:

- Participation in downstream projects like refining/gas processing /LNG/ power projects etc.
- Participation in construction projects, pipelines, process plants etc.

The Corporation is now venturing out to new areas i.e. deepwater exploration and drilling, exploration in frontier basins, marginal field development, optimization of field development plan field recovery and other allied areas of service sector. Engagements in these areas will require best-in-class technology, processes and practices and savvy use of the R&D assets to their fullest advantage.

ONGC is looking towards companies / service providers established in the industry for technology transfer and absorption, and technological collaboration and support. We intend to achieve this objective through alliances and sustained relationship.

Information Technology and Communication Systems are being integrated and upgraded. A number of new projects – project PROMISE (Professional Review of Major Infocom Systems & Equipments), project ICE (Information Consolidation for Efficiency), project IMPETUS (Implementing Maintenance & Procurement Efforts Through Upgraded Systems) are launched. All these projects are aimed at integrating InfoTech resources, integrate company-wide ERP management system and facilitate better maintenance and repair of equipment and facilities.

What is ERP?

India's GNP is now the fifth largest in the world. The transition to an open economy has thrust India squarely into the information age, an age that is forcing the country to cope with a deluge of new information relationships. Integrated ERP solutions optimize source utilization by providing up-to-the-minute information on demand for quick decision making. It is based on Resource-based theory combined with the strategy process perspective & with existing literature on information technology & ERP.

Enterprise Resource Planning system offers a software-based system that handle's an enterprise's total information system needs in an integrated fashion.

The demand for so-called Enterprise Resource Planning (ERP) systems has soared. Triggered by Y2K- compliance problems & the popularity of systems such as SAP R/3, corporate investments in ERP have been significant over the last years where the global markets is expected to reach \$180 billions in the year 2003. It is a software architecture that facilitates the flow of information among all functions within an enterprise. It sits on a common database and is supported by a single development environment. ERP systems are customized to support an organization's business processes.

Enterprise Resource Planning (ERP) systems are modular application software that helps businesses increase the productivity of such mission-critical components as human resources, finance, parts purchasing, inventory control, supply chain and customer relationship management. ERP systems are enterprise-wide and claim to incorporate best business practices that replace legacy systems and current business processes.

ERP implementation in ONGC

SAP AG (NYSE: SAP) signed the largest ever deal in India with the **Oil and Natural Gas Corporation Limited (ONGC)** for implementation of the **ICE** (Information Consolidation for Efficiency) Project. The deal worth Rs.950 million (\$19 million) is aimed at consolidating the IT efforts at ONGC through the implementation of **Enterprise Resource Planning (ERP)** package on SAP.

“The main objective of the ICE Project is to optimize and standardize business processes for integrated information availability. The ERP package will enable availability of information on real time basis and elimination of duplication of activities across business processes by capturing data at source point in turn facilitating decision support, better operation control and efficient cost management”.

ONGC's decision to adopt my SAP Oil & Gas Upstream solutions reconfirms SAP deep industry knowledge and global vision of bringing industry specific business solutions to its customers.” ONGC will benefit immensely by adopting the Best Business Practices provided in the my SAP solution evolved from SAP close association with TOP 10 Oil & Gas Companies like Exxon-Mobil, Shell, Chevron-Texaco, Saudi Armco etc. using SAP solutions.

ERP implementation in ONGC not only automates and integrates its business processes, but also strengthens Management Information System and Data Warehousing for informed and fast decision-making. Like any giant enterprise, ONGC is made up of various resources that fit into building a vibrant enterprise - human, capital, material, etc.

A cohesive synergetic effort from all these resources is vital to take an organization on the path to progress - Towards a brighter future. Successful alignment of business and technology goals has long been considered a business necessity. However, forward- looking, technology-enabled, enterprises

must move beyond alignment and towards integration, if they have to face the market competition. The most relevant form of IT in an enterprise is in the form of Enterprise Resource Planning (ERP). ERP is a tool that facilitates management of all the resources that are vital for the efficient functioning of the enterprise in an integrated manner. A number of software packages are available for implementing ERP, viz., BAAN, SAP, People Soft etc.

ONGC has taken a strategic decision to implement ERP in various business process areas of the organization. In ONGC, SAP, an industry leader in ERP solutions, is already being used in **UFSO (financial accounting)** and **SHRAMIK (HRD)** projects.

Existing Scenario in ONGC

In the later half of the nineties, ONGC realized the importance of having organization-Wide Integrated Information systems. **Integrated Material Management System (IMMS)** was conceived as a first step towards BPR. This was a major strategic initiative with a view to reduce inventory and realize the concept of just-in-time (JIT) procurement. IMMS demonstrated ONGC's capability in adopting IT-based systems for making its business processes more efficient.

ONGC's workforce had no inhibitions to switch over to IMMS, as it was based on the existing MM processes. Thus, IMMS proved to be a harbinger for subsequent ERP initiatives on even wider canvas.

In addition to this support business process, company's top management took another strategic decision to implement **Exploration & Production Information Network (EPINET)**. EPINET is an integrated information database related to core E&P operations of the organization, which would assist in informed decision-making in the core business arena.

After EPINET, it was the turn of financial accounting system. **Up gradation of Financial Services of ONGC (UFSO)** was conceived and rolled out using SAP, a very powerful ERP implementation tool. A lot of groundwork was done before the actual implementation, as a result of which UFSO has been a grand success.

After the successes of these earlier projects, ONGC has undertaken **SHRAMIK** project for managing HR processes. This project is currently being rolled out and will be working with UFSO in an integrated manner. The user base of SHRAMIK is envisaged to be far bigger than the earlier projects.

All these projects are shaping up well and are catering very well to the purpose for which they were implemented. Integrating them into seamless and cohesive systems is the next challenge and, an enterprise- wide ERP is the answer. Thus, company's reservoir of knowledge would always be one - coherent and relevant.

Project ICE (Information Consolidation for Efficiency) has been initiated to implement ERP in the following major areas:

- A) Enterprise Management** - Strategic Enterprise Management, Business Intelligence, Managerial Accounting, Financial Accounting, Joint Venture Management.
- B) Exploration and Production** - Exploration and Appraisal, Development and Production.
- C) Supply-** Supply Chain Optimization.
- D) Manufacturing-** Gas and Fuel Manufacturing.
- E) Marketing** - Contract and Pricing Formulation, Commercial Sales.
- F) Business Support** - HR Core functions, HR - Analytics, Procurement, Treasury / Corporate Finance Management, Fixed Asset Management, E&C and Maintenance.

Need for implementing ERP

- i) ERP creates a single version of the truth because everyone is using the same system and database.
- ii) To standardize business processes. Standardizing the processes and using a single, integrated computer system can save time, increase productivity and reduce headcount. Implementation of ERP is seen as an enabler of Business Process Re- Engineering (BPR).
- iii) To standardize HR information - especially, in companies with multiple business units, HR may not have a unified, simple method for tracking employee time and, communicating with them about benefits and services. ERP can fix that.
- iv) To streamline internal processes and create a platform for e-commerce.

Fully integrated ERP implementation in the organization is envisaged to facilitate the following:

- Leverage E&P operations of the organization through integrated database. And knowledge bank thus enabling Improved Oil Recovery (IOR) and Enhanced Oil Recovery (EOR).
- Help cut down costs, wherever possible, to maintain our competitive edge.
- Deploy scarce resources in the most optimal fashion.
- Avoid duplicity of efforts and costs by knowledge sharing through Decision Support Systems (DSS) and Management Information Systems (MIS).
- Synergizes initiatives and provide them a strategic direction in tune with the Organization's strategy and plan.

About SAP

SAP (Systems, Applications, and Products in Data Processing) the company was founded in Germany in 1972 by five ex-IBM engineers located in Walldorf, Germany. SAP has subsidiaries in over 50 countries around the world from Argentina to Venezuela. The original five founders have been so successful that they have multiplied many times over such that SAP AG is now the third largest software maker in the world with over 17,500 customers (including more than half of the world's 500 top companies). SAP are Maintaining and increasing their dominance over their competitors through a combination Of embracing the internet with mySAP.com extending their solutions with CRM to head Off Siebel & adding functionality to their industry solutions.

SAP employs over 27,000 people worldwide today, and had revenues of \$7.34 billion and Net Income of \$581 million in FY01. SAP is listed in Germany (where it is one of the 30 stocks which make up the DAX) and on the NYSE. There are now 44,500 installations of SAP, in 120 countries, with more than 10 million users.

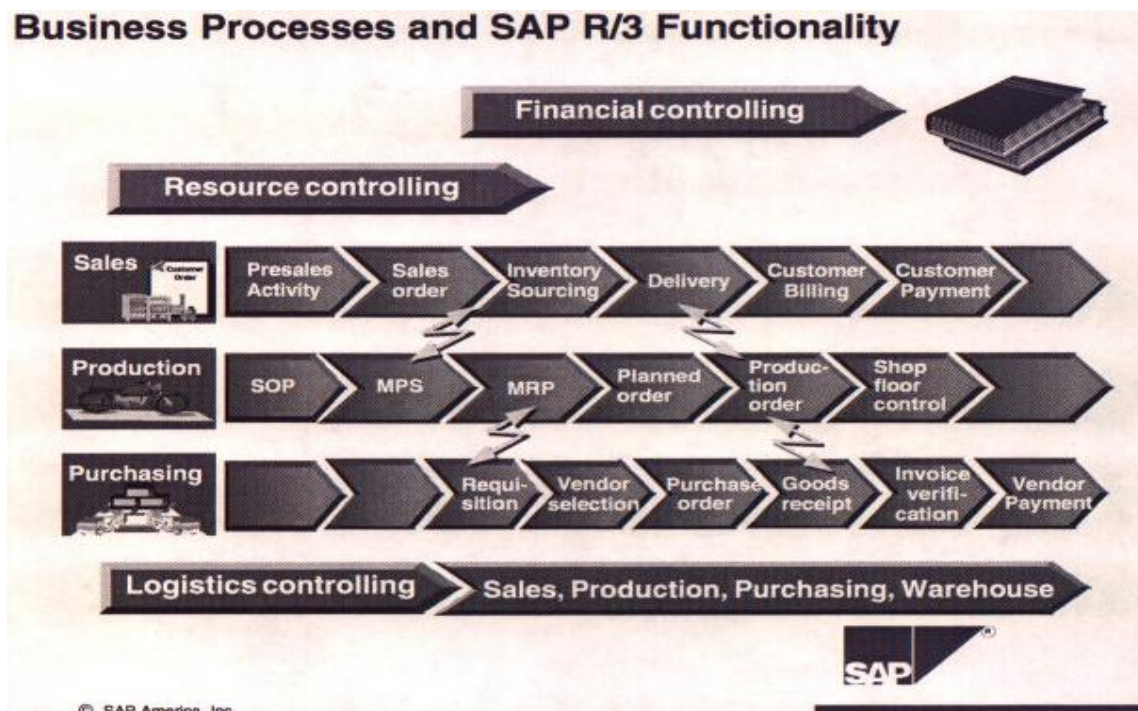
SAP R/3 is delivered to a customer with selected standard process turned on, and many other optional processes and features turned off.

At the heart of SAP R/3 are about 10,000 tables which control the way the processes are executed. Configuration is the process of adjusting the settings of these tables to get SAP to run according to the user's requirements. Since SAP contains sensitive and valuable company information it is vital that it be well secured. SAP security is robust and detailed and provides a mechanism to decrease the risk of someone causing havoc in your SAP system.

SAP structure embodies what are considered the "best business practices". A company Implementing SAP adapts its operations to it to achieve its efficiencies and power. The Process of adapting procedures to the SAP model involves "Business Process Re-Engineering" which is a logical analysis of the events and relationships that exist in an Enterprise's operations. SAP structure embodies what are considered the "**best business Practices**". A company implementing SAP adapts its operations to it to achieve its efficiencies and power.

The process of adapting procedures to the SAP model involves "Business Process Re-engineering" which is a logical analysis of the events and relationships that exist in an enterprise's operations. In order to understand a system like SAP a thorough understanding of the events and relationships that take place in a business is required.

The exact details of each action, the timing of that action and its interrelationships with every other process must be understood. Before an operation can be automated or computerized a thorough study of the business must be undertaken. This task is called **Business Process Engineering**.



Distributed processing and Integration with R/3:

Distributed Processing:

SAP customers make different demands on R/3 application systems. R/3 technology meets these requirements with flexible concepts:

- The cost-efficient implementation of R/3 on a centralized or two-tier client/server system is the ideal solution for small and mid-sized installations.
- Mid-sized and large installations require high throughput performance with an acceptable management effort demanding three-tier client/server implementation.
- Very large installations do not simply demand powerful systems. They also require a very high level of system availability. The use of parallel database servers in three-tier client/server architecture is considered to be the optimum solution.
- Group wide and worldwide installations have the greatest need for performance and availability. Many other requirements also have to be considered, such as the integration of existing structures and the necessity of geographical distribution.
- The ideal solution is provided by cooperative client/server computing with distributed applications.

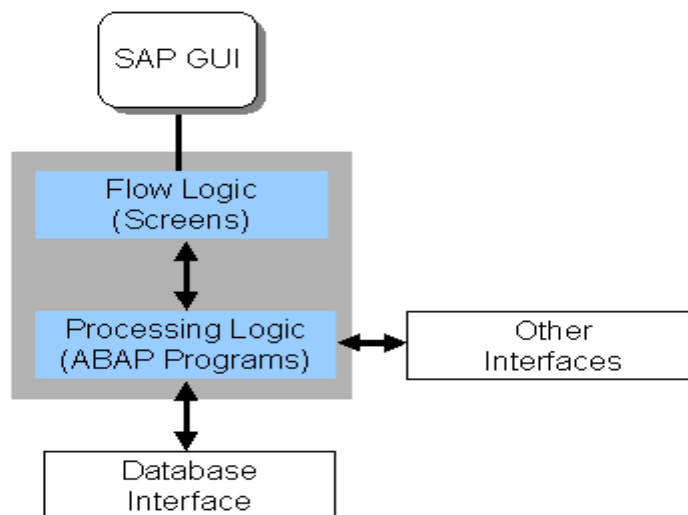
Integration and openness:

Business processes do not end at the borders of computers or applications. The R/3 infrastructure technology provides a software bus to guarantee the complete implementation of business processes. It handles message-based integration for almost any number of application components. This ensures that different applications work together instead of against each other.

- Messages are exchanged between applications via the Application Link Enabling concept (ALE). This technology controls the communication between R/3 subsystems, R/2 and R/3 applications, and, of course, between non-SAP systems and R/3.
- SAP uses OLE technology (Object Linking and Embedding) from Microsoft and the ODBC standard (Open Database Connectivity) to link PC components with R/3.
- SAP Business Workflow controls the individual processing steps executed in the application components.

R/3 Basics

R/3 application programs run within the R/3 Basis system on the work processes of application servers. This makes them independent of the hardware and operating system that are used & unable to run outside the R/3 System. All application programs, along with parts of the R/3 Basis system, are written in the ABAP Workbench using ABAP, SAP programming language. The individual components of application programs are stored in a special section of the database called the R/3 Repository. The R/3 Repository serves as a central store for all of the development objects in the R/3 System. A work process contains a screen processor for processing user input, an ABAP processor for processing the program logic, and a database interface for communicating with the database. These components of a work process determine the following structure of an application program:



An application program consists of two components, each of which has a different task:

1. Flow Logic: Interaction between application programs and the user is implemented using screens. Screens are processed by the screen processor of a work process. As well as the input mask, they consist of flow logic. This is coding written using a special set of keywords called the screen language. The input mask is displayed by the SAP GUI which also transfers the user action on the screen back to the flow logic. In the program flow, screens react to the user actions and call program modules. These program modules form the processing logic.

2. Processing logic: The components of application programs that are responsible for data processing in the R/3 System are ABAP programs. ABAP programs run on the ABAP processor of a work process. They receive screen input from the screen processor and send it to the screen processor. ABAP contains a special set of commands called OPEN SQL allowing to read from and write to the database regardless of the database being used. The database interface converts the OPEN SQL commands into commands of the relevant database. Interfaces provide other means of sending data to and from ABAP programs. When working together with screens, ABAP programs play a more passive role, acting as a container for a set of modules that can be called from the flow logic.

R/3 Instance Provider Services

An R/3 Instance provides one or more services. A central R/3 system consists of one instance, which provides all of the services. In a distributed R/3 system, the enqueue service and the message service is always grouped in one instance: the central instance. Normally, each instance runs on its own computer, which is the application server. From an operating-system perspective, one instance consists of combined processes and the main memory they use. These processes include one dispatcher and multiple work processes for each instance.

The SAP infrastructure technology is a powerful basis for the R/3 applications and the ABAP/4 Development Workbench. It enables a virtually unrestricted distribution of applications and databases, as well as unprecedented integration of in-house and external software components.

Architecture of SAP R/3 systems

SAP R/3 is a highly configurable software package built on best business practices in the Enterprise Resource Planning (ERP) areas and developed over more than 25 years of enterprise systems experience. It is designed to streamline data collection and processing based on the informational needs of the industry segment. Better information leads to better business decisions, which in turn leads to improved business performance. Enterprise Resource Planning systems represent the most complicated business systems that a company will implement. This complexity is measured in terms of the number of system users, database size, transaction volumes, and other system metrics.

Layer by Layer-R/3 System Structure:

The R/3 system's structure follows a layer model with independent function layers connected by interfaces. The software between the business applications, operating system, and the hardware is called middleware, and is transparent to the user. The quality of the middleware strongly influences the performance of client/server applications, as well as their openness and portability. The basis layer contains the R/3 system's middleware.

Middleware enables applications to be independent of the system interfaces of the operating system, database system, and communications system; it also ensures optimum handling of business transactions. The application layer, which implements the business functions and processes of the R/3 system, sits on the basis layer. This basis layer is written in the C/C++ programming language, while the application layer is written in the SAP-developed 4GL language ABAP/4. The ABAP/4 Development Workbench is SAP programming environment for the development of enterprise-wide client/server business solutions.

Logical Services in the R/3 System:

An essential characteristic of the R/3 system architecture is the software-oriented client/server approach, through which various program modules can be distributed flexibly across different computers (also known as partitioning of applications). This gives users great flexibility in planning and operating an R/3 system, and provides scalability to match their business growth. The individual systems in an R/3 environment provide computing capacity and storage for the logical layers of the client/server architecture.

Management of Working Data:

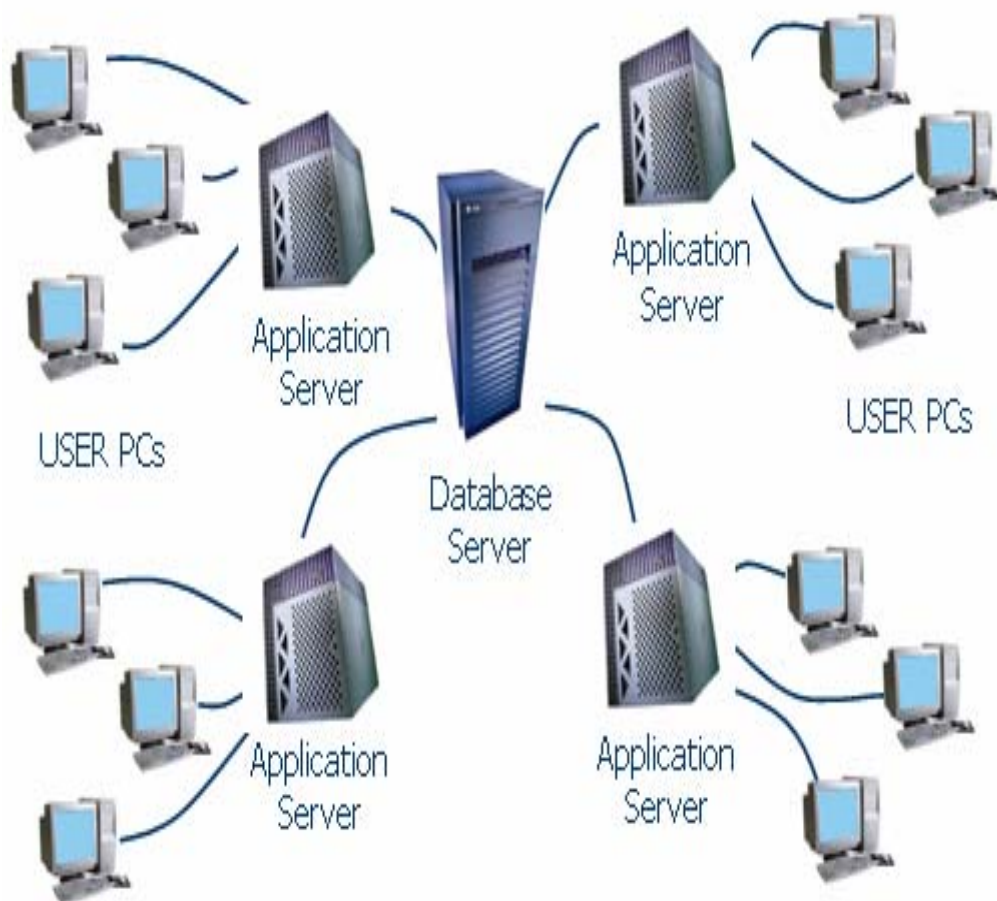
The database layer is used for the management and administration of the working data, which includes the master data, transaction data, and the metadata maintained in the repository that describes the database structure. The industry-standard SQL (Structured Query Language) defines and manipulates all data and runs on industry-standard relational database management systems (RDBMS).

The Application Layer where Business Logic Resides:

The entire R/3 business logic runs on the application layer; the application layer in the middle builds on the underlying database layer. The applications access, retrieves, and inserts data into the database. Because Internet technology does not handle business transactions, the Internet-enabling layer has been created above the application layer. The SAP R/3 system has seven services under the application layer. The R/3 system's configuration determines the specific processes available on the server and the extent of the processes.

Starting at the Top The Presentation Layer:

The top layer, which is closest to the users, is the presentation layer. It manages the interface that users operate to access data, run applications, and view results. Users work with R/3 applications through the SAP graphical user interface (SAPGUI), which provides a common look-and-feel across all platforms.



SAP Application Modules

SAP has several layers where the application modules are the epicenter of the whole system. These modules may not all be implemented in a typical company but they are all related and are listed below:

- **FI Financial Accounting**—It is designed for automated management and external reporting of general ledger, accounts receivable, accounts payable and other sub-ledger accounts with a user defined chart of accounts. As entries are made relating to sales production and payments journal entries are automatically posted. This connection means that the "books" are designed to reflect the real situation.
- **CO Controlling**— It represents the company's flow of cost and revenue. It is a management instrument for organizational decisions. It too is automatically updated as events occur.
- **IS Asset Management**— It is designed to manage and supervise individual aspects of fixed assets including purchase and sale of assets, depreciation and investment management.
- **PS Project System**— It is designed to support the planning, control and monitoring of long-term, highly complex projects with defined goals.
- **WF Workflow**— It links the integrated SAP application modules with cross- application technologies, tools and services.
- **IS Industry Solutions**— It combines the SAP application modules and additional industry-specific functionality. Special techniques have been developed for industries such as banking, oil and gas, pharmaceuticals, etc.
- **HR Human Resources**— It is a complete integrated system for supporting the planning and control of personnel activities.

- **PM Plant Maintenance**— In complex manufacturing process maintenance means more than sweeping the floors. Equipment must be services and rebuilt. These tasks affect the production plans.
- **MM Materials Management**— It supports the procurement and inventory functions occurring in day-to-day business operations such as purchasing, inventory management, reorder point processing, etc.
- **QM Quality Management**— It is a quality control and information system supporting quality planning, inspection, and control for manufacturing and procurement.
- **PP Production Planning**— It is used to plan and control the manufacturing activities of a company. This module includes; bills of material, routings, work centers, sales and operations planning, master production scheduling, material requirements planning, shop floor control, production orders, product costing, etc.
- **SD Sales and Distribution**— It helps to optimize all the tasks and activities carried out in sales, delivery and billing. Key elements are: pre-sales support, inquiry processing, quotation processing, sales order processing, and delivery processing, billing and sales information system.

Advantages of SAP Approach

SAP solutions have major advantages over other available solutions namely:

Architecture: SAP SEM is built on state-of-the-art computing technologies based on the business framework architecture-capable of processing huge volumes of information & supporting rapid decision-making. Unlike other solutions on the market, SAP implementation offers high levels of data integration by ensuring synchronization of information between SAP SEM & the underlying business execution systems. This open connectivity allows business components to interact with multiple business systems.

Functionality: Building on existing SAP R/3 functionality, SAP SEM includes functionality for advanced business consolidation & business information collection which links internal information with automated collection of relevant external information through the internet. SAP SEM also supports advanced business planning and simulation based on both internal & external information-allowing modeling risks & rewards & more effectively manage the uncertainties of the future.

Ease of implementation: SAP SEM is based on business framework architecture using SAP Business Information Warehouse technology with easy access for end users through the Business Explorer & its advanced data warehouse & OLAP capabilities.

Ready-to-use ABAP enables easy integration with the business execution systems & with those of external partners. Web-enabled end-user access in combination with SAP Business Framework architecture & other advanced SAP R/3 & Business Information Warehouse technologies, allows flexible & rapid deployment of SAP SEM enterprise-wide.

Componentization: As with other SAP solutions, the Business Framework architecture lets you deploy SAP SEM components incrementally because SAP uses state-of-the-art, object-oriented techniques, one can start deploying SAP SEM components today & integrate future components step-by-step.

Lowest Total Cost of Ownership: Companies implementing or planning to implement non-SAP SEM-type components are all too familiar with the expense involved. Using SAP R/3 as the foundation for SAP SEM, SAP customers will achieve lowest total cost of ownership, by virtually eliminating the additional costs & concerns over version compatibility that result from attempting to integrate multiple vendor products. SAP is focusing its development efforts on significantly enhancing the already impressive SAP SEM solution – bringing more cost benefits in the future.

R/3 Lock Concept

Lock Mechanisms in the Database System:

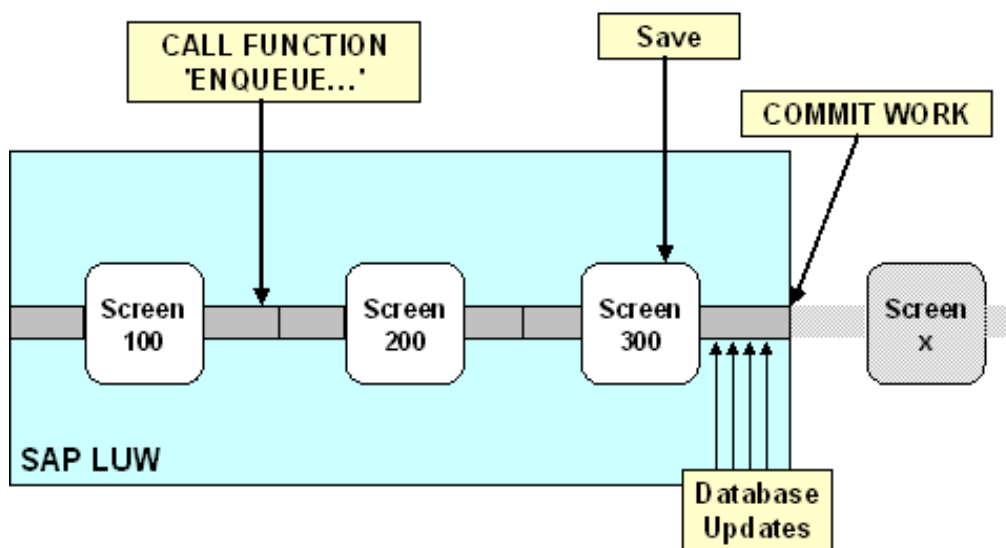
The database system automatically sets database locks when it receives change statements (INSERT, UPDATE, MODIFY, DELETE) from a program. Database locks are physical locks on the database entries affected by these statements. We can only set a lock for an existing database entry, since the lock mechanism uses a lock flag in the entry. These flags are automatically deleted in each database commit. This means that database locks can never be set for longer than a single database LUW; in other words, a single dialog step in an R/3 application program.

Physical locks in the database system are therefore insufficient for the requirements of an R/3 transaction. Locks in the R/3 System must remain set for the duration of a whole SAP LUW, that is, over several dialog steps. They must also be capable of being handled by different work processes and even different application servers. Consequently, each lock must apply on all servers in that R/3 System.

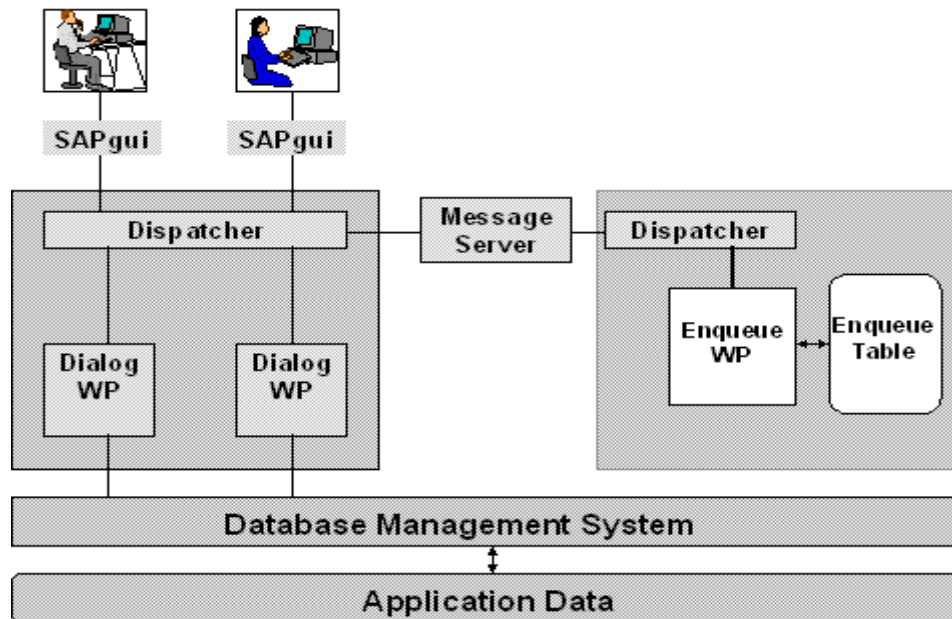
SAP Locks: To complement the SAP LUW concept, in which bundled database changes are made in a single database LUW, the R/3 System also contains a lock mechanism, fully independent of database locks,

That allows you to set a lock that spans several dialog steps. These locks are known as SAP locks. The SAP lock concept is based on lock objects. Lock objects allow you to set an SAP lock for an entire application object. An application object consists of one or more entries in a database table, or entries from more than one database table that are linked using foreign key relationships.

A lock object definition contains the database tables and their key fields on the basis of which locking is to be set. When a lock object is created, the system automatically generates two function modules with the names ENQUEUE_<lock object name> and DEQUEUE_<lock object name> .We can then set and release SAP locks in your ABAP program by calling these function modules in a CALL FUNCTION statement.



These function modules are executed in a special enqueue work process. Within an R/3 System, enqueue work processes run on a single application server. This server maintains a central lock table for the entire R/3 System in its shared memory.



The enqueue function module sets an SAP lock by writing entries in the central lock table. If the lock cannot be set because the application object (or a part of it) is already locked, this is reflected in the return code sy-subrc. The following diagram shows the components of the R/3 System that are involved in setting a lock.

Unlike the database, which sets physical locks, the SAP lock mechanism sets logical locks. This means that

- **A locked database entry is not physically locked in the database table:** The lock entry is merely entered as a lock argument in the central R/3 lock table. The lock argument is made up of the primary key field values for the tables in the lock object. These are import parameters of the enqueue function module. The lock is independent of database Locks. It is released either implicitly when the database update or the SAP transaction ends, or explicitly, using the corresponding desuetude

function module. You can use a special parameter in the update function module to set the exact point at which the lock is released during the database update.

- **A locked entry does not necessarily have to exist in a database table:**
To set a lock as a precaution for a database entry that is not written to the database until the update at the end of the SAP LUW.
- **The effectiveness of the locks depends on cooperative application programming:** Since there are no physical locks in the database tables themselves, all programs that use the same application objects must look in the central table themselves for any locks. There is no mechanism that automatically prevents a program from ignoring the locks in the lock table.

Lock Types:

There are two types of lock in the R/3 System:

- **Shared lock:** Shared locks (or read locks) allow you to prevent data from being changed while you are reading it. They prevent other programs from setting an exclusive lock (write lock) to change the object. It does not, however, prevent other programs from setting further read locks.
- **Exclusive lock:** Exclusive locks (or write locks) allow you to prevent data from being changed while you are changing it yourself. An exclusive lock, as its name suggests, locks an application object for exclusive use by the program that sets it.

No other program can then set either a shared lock or an exclusive lock for the same application object.

Lock Duration: If several programs running concurrently all set a shared lock for the same application object in the system, it can make it almost impossible to set an exclusive lock, since the program that needs to set that lock will be unable to find any time when there are no locks at all set for that object. Conversely, a single exclusive lock prevents all other programs from reading the locked object.

At the end of an SAP LUW, you should release all locks. This either happens automatically during the database update, or explicitly, when you call the corresponding desuetude function module. Locks that are not linked to a database update are released at the end of the SAP transaction.

Hardware & Software Specifications

When setting up a SAP System, we need to install the main components that enable the system to operate. These are the:

- Central instance.
- Database instance.
- Dialog instances, if required.

The configuration of the system is generally planned well in advance of the installation together with the hardware vendor. Configuration planning involves deciding whether a central system or standalone database system is to be installed, and how many dialog instances are required. The configuration is worked out with the hardware partner on the basis of sizing information that reflects the system workload. Details such as the set of applications that are to be deployed, how intensively these are to be used, and the number of users enables the hardware vendor to recommend a configuration that performs well.

An SAP System consists of a database server, (optional) additional application servers, and a number of front end computers. The graphic below shows a typical distribution of the instances of an SAP System over several computers.

The database server is the computer on which the database is installed. This server can also accommodate the central instance (the SAP instance that includes the message server and enquire server processes). If the central instance is installed on a separate application server, the database server is called a standalone database server.

Software Requirement

SAP R/3 4.6c with Oracle 9.2.0.4.0 as backend database.

Hardware Requirement

Production Server	: HP Superdome 9 CPU 875 MHz PA870 Main Memory: 10GB
Development Server	: HP RP 8410 8 CPU 875 MHz Main Memory: 10 GB
Quality Server	: HP RP 8710 9 CPU 875 MHz Main Memory: 16 GB
Application Server (7 Nos.)	: HP RP 7410 PA8700 8 CPU Main Memory: 16GB
Hard Disk Array	: Storage Area Network 22.5 TB
Backup Unit	: Robotic Tape Library ESL 9000, 594 tapes of 400 GB each.
Clients	: Intel 2.4 GHz/256 MB/40 GB/Windows XP/ SAP GUI 6.2

Basis Details

BASIS Software:

- The R/3 Basis software also called middleware provides the runtime environment for the R/3 applications.
- It defines a stable architectural framework for future system enhancements.
- Enables the distribution of resources and components.
- Offers interfaces to distribute system components and non-SAP products.
- Contains the administration tools for the entire system.

Day to Day Work (24x7 Basis):

- ❖ System Administration
- ❖ User Maintenance – Help Desk Manning
- ❖ Job Monitoring
- ❖ Background Job Overview
- ❖ Log Monitoring
- ❖ Co-ordination with project teams and outside project
- ❖ Backup of the Systems
- ❖ Monitoring and Ensuring Suitable Environmental Conditions for Data Center
- ❖ Archive Management
- ❖ Copy Analysis Log
- ❖ Lock Transactions
- ❖ Monitor Current Workload
- ❖ Setups/Tune Buffers
- ❖ System Trace Operating System Monitor
- ❖ Spool Administration
- ❖ User Master Data Reconciliation

Introduction to ABAP/4 Programming

Language

SAP R/3 incorporates its own unique programming language called ABAP. ABAP is an event-driven fourth-generation language. It is a language that is constantly evolving with recent releases incorporating object-oriented capabilities (ABAP objects). The robustness of the language is evident in the wide range of functionality and high performance capabilities within the R/3 system, allowing applications to process huge amounts of customer data. SAP R/3 provides an environment rich in tools for developing business applications using the ABAP programming language.

ABAP stands for Advanced Business Application Programming 4GL. ABAP/4 is SAP fourth-generation language & it is the backbone of R/3 system boasting a combination of features that give it power and flexibility. Although other languages may offer similar capabilities, ABAP/4 is designed specifically for SAP functionality—making it a more efficient tool for SAP development in many cases. The language's flexibility & ease of use combined with its specially designed commands enable to quickly design & implement both small & large scale business solutions for R/3 systems.

Originally, SAP developed the programming language ABAP/4 (Advanced Business Application Programming) for internal use, to offer the application developers better working conditions. ABAP/4 is continually improved and adapted to the growing requirements of the business applications. Today, ABAP/4 is the only tool for developing applications at SAP. SAP customers use ABAP/4 for their own enhancements or for modifying R/3 standard solutions according to their special requirements.

The ABAP/4 Development Workbench contains all tools you need to create and maintain ABAP/4 programs.

Salient Features of ABAP:

ABAP/4 is event-driven

The language is structured to accommodate applications that don't simply start and stop or respond to user dialog; ABAP/4 applications are executed by user interaction and by events occurring elsewhere in the system, and they're set up to initiate other down line events and applications in response to user decisions. Whereas other languages require complex programming to achieve this sort of event-oriented execution, ABAP/4 is loaded with conveniences to enable it. User actions and system events control the application's execution.

Thus developers can develop complex interactive programs that support "drill-down" events.

ABAP/4 is designed to enable distributed applications

SAP systems are distributed systems, unifying diverse databases within a company and beyond, integrating applications in your client's company, and creating distributed systems between your client and your client's partner companies. This distribution is enabled by Remote Function Calls (RFCs), function modules that can be called on between systems. RFCs can enable applications between SAP systems or between a SAP system and a non-SAP system. They enable SAP-external applications, establishing a path for extraction of SAP data to non-SAP applications. Making good use of RFCs ABAP/4 applications can be given tremendous flexibility whatever may be its purpose.

Defining subroutines

ABAP/4 provides two techniques for defining subroutines: forms in ABAP/4 are local subroutines of a single program, & functions are global components that are called by many different programs. The language provides two techniques for defining subroutines: local and global. Global subroutines, called Function

Modules, are stored in a central library and available to all programs. They use a standard interface for passing parameters and exemption handling. The Function Module library has efficient searching tools allowing developers to find appropriate modules to incorporate into their programs.

Both kinds of subroutines support encapsulation of local data as well as different methods to transfer data via interface parameters.

ABAP/4 facilitates logical databases

In most SAP data-retrieval applications—from report generation to user dialog to down line data flow—the business objects are essentially being used data records of mixed type. While working with the application we work with the aggregation of databases. SAP allows easy definition and manipulation of logical databases, providing a convenient way to create data structures that can accommodate these complex requirements. You can simplify database access considerably by using ABAP/4 to create application-internal tables mapped from more static SAP database tables, in essence creating easily manipulated virtual objects that do the work of pulling out data from many locations in SAP. Moreover, you can store your logical database configurations in a repository for later use in other applications.

ABAP/4 comes with all the extras

Like most popular 4GLs, ABAP/4 comes packed in a filled-with-frills workbench. The ABAP/4 development environment offers a data modeler and a repository for data models, data and table structures, functions that are reused, and user interface screens.

ABAP/4 applications have cross-platform portability. And to accommodate SAP large international installation base, ABAP/4 supports language-dependent text, allowing users to create programs that will display in their language.

Portability

The nature of SAP Basis software guarantees that ABAP programs can be supported by many different systems. There are only a few cases where transfer of programs to other platforms could cause problems.

Support for structured programming techniques

Unlike many other programming language, ABAP/4 supports combining of a sequence of statements that have identical beginnings into a single statement using a colon & commas with the identical part appearing before the colon.

ABAP/4 features a set of elementary types (including character, integer and date) & two construction concepts (records & internal tables) helps in building complex types & data objects. References tell the program to use the definition of the existing data object for a new data object so that both objects will have the same information.

Language supported by ABAP/4

ABAP/4 contains a subset of SQL (Open Structured Query Language) that is an integral part of the language. Open SQL and the database interface of the R/3 system form layers between the Database Management System and the application program. This layered architecture enables the developer to concentrate on conceptual aspects rather than worry about things like memory management, pointer arithmetic or network issues. ABAP/4 programs using Open SQL can access data from all database systems that are supported by the R/3 system.

Supporting Internal Tables

The language uses the concept of internal tables, which provide a consistent mapping of persistent database tables into runtime objects. For example, the contents of a database table can be mapped into an internal table at runtime,

so that the internal table is a snapshot of the database table itself. Internal tables are a means of storing data with a fixed structure within the memory of an ABAP program. In ABAP, internal tables provide the functions of dynamic arrays, while saving the programmer the task of program-controlled dynamic memory management. One particularly important use is to store and format the contents of database tables within a program. They are also the best way to implement very complex data structures in an ABAP program.

Designing Reports & Supporting Graphical User Interface

An ABAP/4 report is a program that retrieves data from the database, groups it together according to different criteria & presents it on the screen or as a printed list. Logical databases provides procedures for retrieving complex data from databases making it easy to develop well-structured reports & re-usability that distinguishes ABAP/4 from many other programming languages. It also provides many tools & language constructs for designing applications with dialog boxes & user-input screens with an easy-to-use Graphical User Interface. Dictionary-based validation rules & consistency checks provide data integrity even at the user-input level that is the program to make sure data is valid at the time the user inputs it so that if the data is incorrect, the user has to correct it right then.

Multi-language support

ABAP/4 makes it easy to develop international application programs for worldwide use. It provides the users, language-dependent texts, whenever text is displayed on a screen or printed list. These texts can be translated into different languages without requiring to change or generate a program. ABAP/4 offers language-dependent texts in the situations say for example program documentation & program title, interface descriptions of functions, selection texts on selection screens, menu functions & push buttons etc.

Re-usability

ABAP/4 supports the system-wide re-use of different component namely Structure definitions & types from Dictionary, Functions with a flexible interface & exception handling, Logical databases for data retrieval in report. The ABAP/4 Development Workbench contains a large number of re-usable functions where each function is uniquely determined by its name & belong to a function group.

Flexibility

ABAP/4 offers a high degree of flexibility firstly if a program is executed after dictionary objects have been changed & activated, the program is only re-compiled if it refers to one of the changed objects avoiding the rigidity of using header files, secondly the interface of a function can be extended without affecting existing callers & finally a report using a Logical Database can freely choose the tables that are to be read from the database.

Open Interfaces

R/3 system users generally work with many different tools & products in a client / server environment so applications written in ABAP/4 should be able to communicate with these components. It offers various open interfaces that allows this communication. Remote Function Call supports direct program-to-program communication between R/3 systems, between an R/3 system & a main-frame-based SAP R/2 system, or between an R/3 system & external programs such as C/C++ or Visual Basic programs.

SAP R/3 provides the “ABAP Workbench” containing a repository of development tools. These tools provide the developer with a range of functionality that covers the entire software development cycle.

Types of ABAP/4 programs

There are two types of ABAP/4 programs: dialog programs and reports.

Report Programs:

Report programs are used to analyze data from database tables. Usually, the result of such a report program is in the form of a list which is output to the screen or sent to a printer. This list is a report in the usual sense of the word. Therefore, a report program is a program which creates a report based on the evaluation of one or more database tables. In reporting ABAP/4 Open SQL is used to read data from the R/3 database. A report consists of a selection screen, on which you define the data set you wish to display, and a list, which displays the data itself. Typically, reports merely read data from the database.

However, it is also possible to make changes in the database using a report. Report programs are based on logical databases which are special ABAP/4 programs which provide access to all databases. List in SAP is called a report.

Dialog Programs:

Dialog programs are organized as module pools which contain dialog modules. Each “dynpro “(a Dynamic program which consists of a screen and its flow logic) is based on exactly one ABAP/4 dialog program.

In dialog programming you use the **Screen Painter** to create screens and program the sequence in which they appear.

You write an ABAP program (ABAP/4 **module pool**) for your screens. Dialog program is a SAP transaction. This is a collection of dialog modules which are called by the flow logic of your screens. You use dialog programs for both reading and changing database tables.

ABAP Development Workbench

The ABAP Workbench is SAP graphical programming environment. It allows you to create new ABAP client/server applications, working as a team, and to change existing SAP applications. The ABAP/4 programming language is the central tool in the Development Workbench that is integral to the R/3 Basis and available with all the R/3 application modules. In addition, the Development Workbench consists of a set of tools including the ABAP/4 programming language that support the efficient development of large-scale enterprise applications across a distributed client/server environment.

The Workbench can be used to program & prepare coding for ABAP requirements.

- Designing dialogs with a graphical editor.
- Creating menus with a menu editor.
- Debugging an application for pinpointing debugs in the application.
- Testing an application for efficiency.
- Provides predefined functions for easy use.
- Controlling access to objects under development.
- Creating a new or accessing predefined database information.

ABAP Development Workbench Architecture

The ABAP/4 Development Workbench provides access to SAP development tools which cover the entire software development cycle.

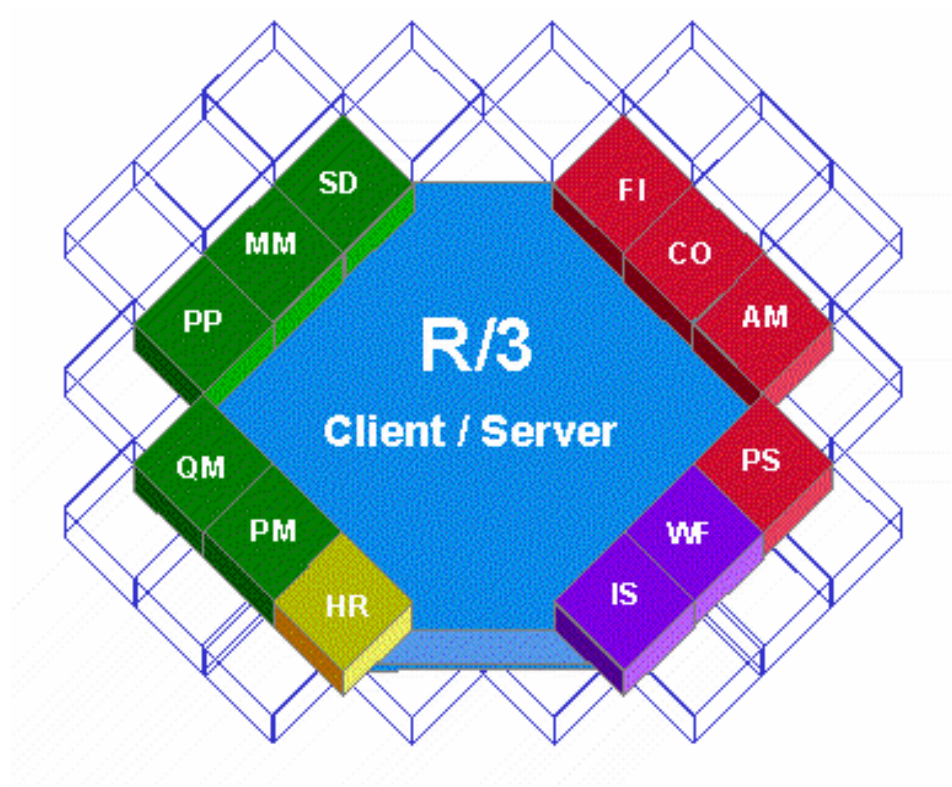
- The Development Workbench is the environment used to create and edit ABAP/4 programs.
- These tools can be used both for customer-specific developments and enhancements to R/3 applications supplied by SAP.
- All applications created with the ABAP/4 Development Workbench can run without further modifications on any platforms, database systems and graphical user interfaces supported by SAP.

ABAP/4 in the SAP R/3 System is a **shared** development environment. This means that all programs reside in the R/3 Repository. Therefore, other programmers are working with programs that reside in the common storage of the R/3 Repository. You must exercise care to make sure that your program follows established naming conventions and are unique within the R/3 System.

The ABAP Workbench provides the following tools for navigating among development objects: The Repository Browser, the R/3 Repository Information System, and the application hierarchy.

All three navigation tools use a "file manager" type interface for displaying development objects. The Workbench also contains the Data Browser for displaying the contents of database tables. Thus ABAP/4 language is the central tool in the excellent development package called the ABAP/4 development workbench, and it is the synergy of these two components that guarantees an efficient development process for large-scale enterprise application across a distributed client server environment. Using the tools available in the development package, we can concentrate on the conceptual aspects of new application, without getting mired down in memory management, pointer arithmetic, and network optimization. The ABAP/4 development workbench

proves its strength in the R/3 system itself, where more than a thousand developers used the Workbench tools to build an integrated package of business application. SAP continues to extend the functionality of the workbench with new improvement such as integrated desktop applications or business process modeling.



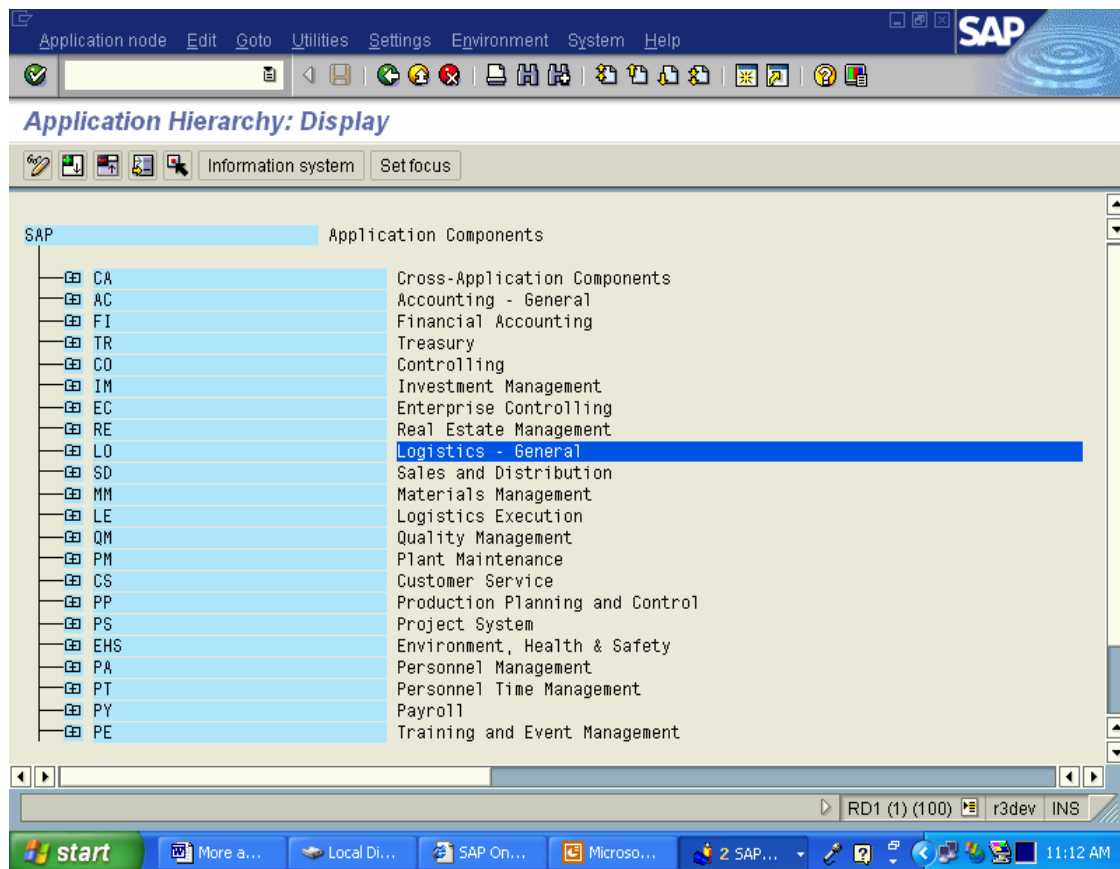
Tools for software development:

ABAP/4 Repository:

The R/3 Repository contains all of the development objects in the system. From the Repository Information System you can search for Dictionary objects, program objects, function groups, and so on. To access the R/3 Repository Information System from the ABAP Workbench, choose *Overview-> Repository Infosys*. The initial screen of the Repository Information System displays a hierarchical list of all the different types of objects in your R/3 System. The Repository can be used to: generate lists of programs, tables, fields, data elements, and domains, find where tables and fields are used in screens and ABAP programs, display foreign key relationships and so on. The object categories in the R/3 System are modeling objects, ABAP Dictionary objects, programming objects, and environment objects. The Repository Browser is the central tool for organizing and managing your personal development objects, the most commonly-used tool in day-to-day development. Its user interface resembles that of a file manager. These object lists contain every element belonging to a particular development class, program, or function group. Working with object lists helps in keeping track of development objects without losing sight of the overall context. The root node of every object list is always a development class. The lowest level of objects can be traverse from using the development class.

Application Hierarchy

The Application Hierarchy depicts the organization of all the applications in your R/3 system. The application hierarchy is an organizational tool. Each company defines its own hierarchy explicitly. To start the application hierarchy from the Workbench initial screen, choose *Overview Application hierarchy SAP* or *Customer*. The nodes in the application hierarchy are either title nodes or development nodes. Development nodes have an accompanying development class indicating there are actual objects associated with the node. Title nodes have no development class and are used to help visually organize the hierarchy. Whenever you assign a development class to a title node, a new development node is created. Direct branching from the application hierarchy to the Repository Information System is possible. This function can be used whenever you want to restrict your search to the objects in one or more specific applications.

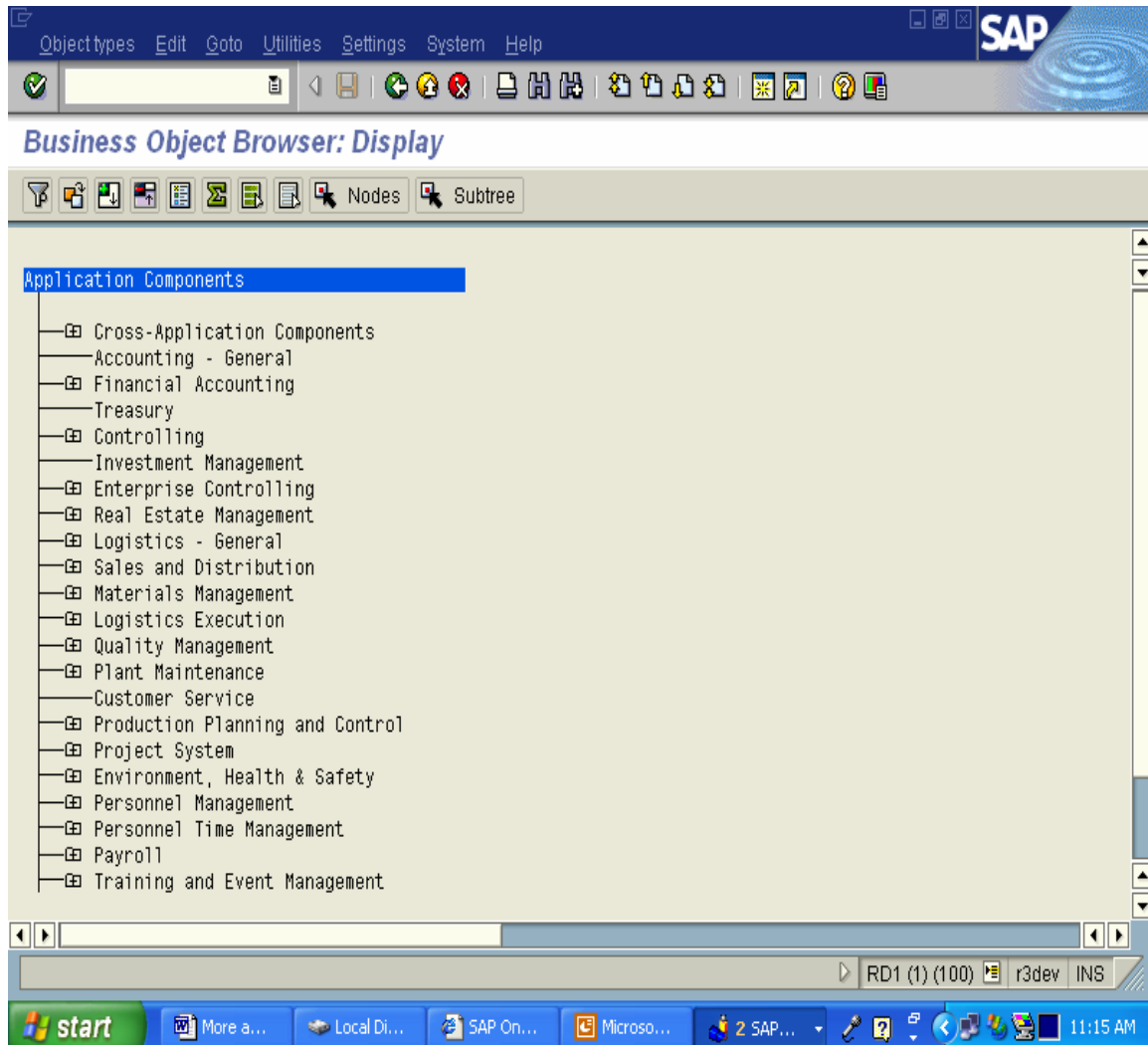


Object Browser

The ABAP Development Workbench contains a special tool, the Object Browser. The Object Browser is a navigation tool for accessing R/3 Repository objects. This browser assists in drilling down in the object hierarchy. Development classes are categories of SAP objects that can be accessed. A common transport route is defined for all objects within a development class. Customer-specific development classes begin with Y or Z.

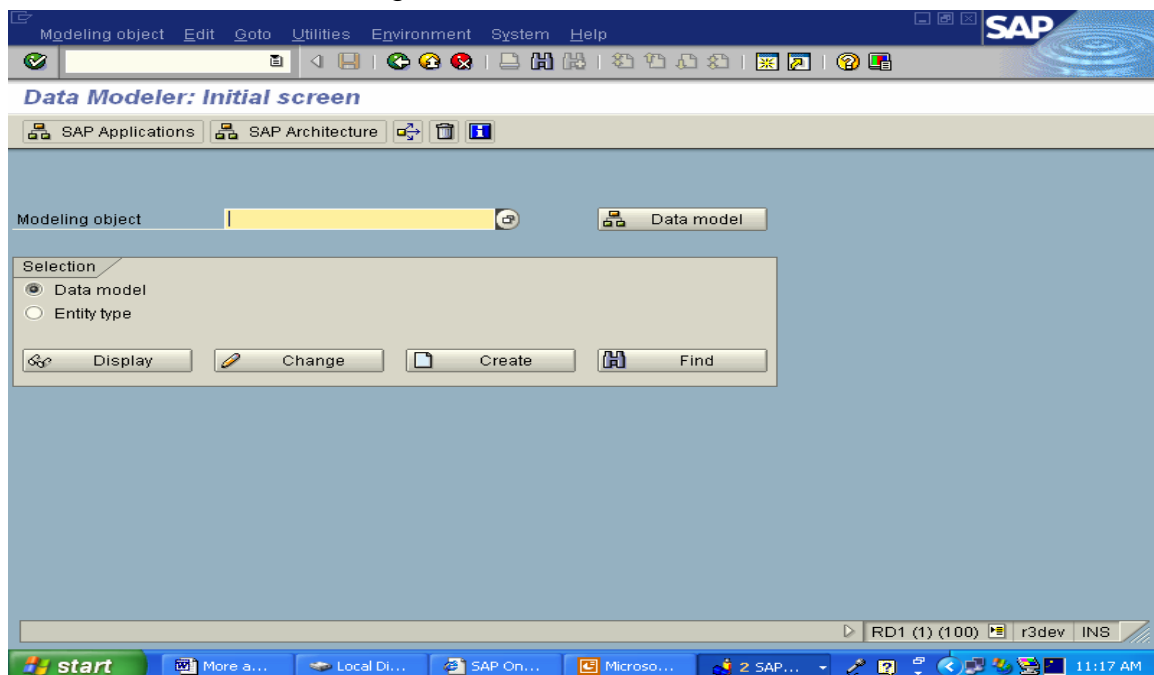
The Object Browser provides a programming context. Because a program consists of relationships between data, it is hard for a programmer to see the relationships among separate data components. The Object Browser corrects this problem by supplying the context for viewing programming relationships. Object Browser can be used to navigate through a list of development objects. Development objects are the components which are used to build an application. When you use the Object Browser, it automatically calls other tools when your actions require them. For example, you create a new data definition from the Object Browser screen. The Browser calls the Data Dictionary and, after you create your definition, returns you to the Browser screen. An entire application can be created using the Object Browser without directly calling any of the other tools.

In fact, the recommended method for creating an application is from the Object Browser because the program is easy to be viewed in the phase of development.



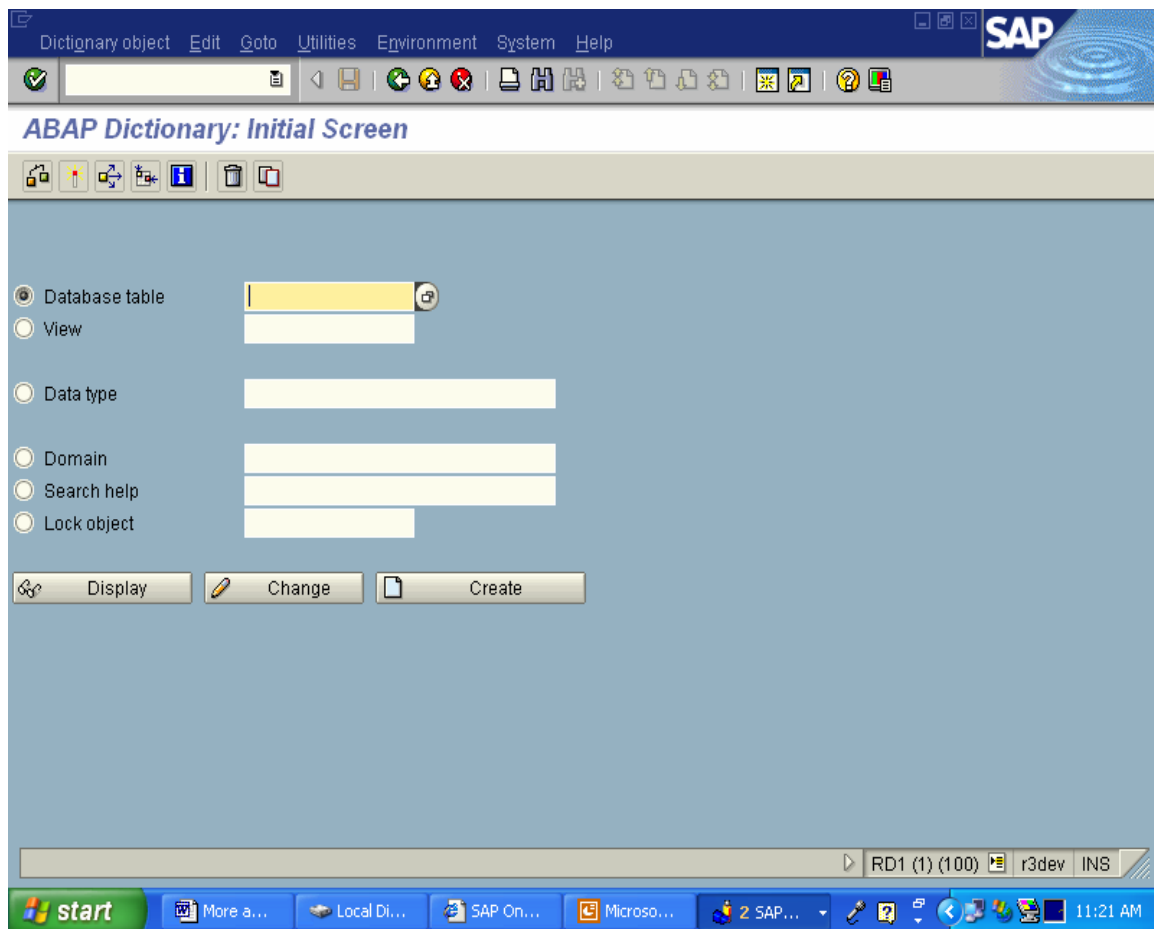
Data Modeler

The Data Modeler is a development tool from the ABAP/4 Development Workbench, which can be used to create data models according to the SAP SERM method (SERM = Structured Entity Relationship Model). In addition to supporting modeling functions, the Data Modeler allows to map the models created by the developer to the ABAP/4 Dictionary. Owing to the close integration of the Data Modeler with the ABAP/4 Dictionary, both top-down and bottom-up modeling approaches are possible. The basic objects of the Data Modeler are data models and entity types, and the relationships and specializations existing between the entity types. The problem to be modeled - generally a small section of the real or abstract world - is mapped to a data model. The individual (physical or abstract) objects of the section to be modeled are represented by entity types that are connected by relationships. In the Data Modeler, data models can be created with any number of hierarchy levels and in the process reuse parts of data models. The data model components can be expanded or compressed in a hierarchy list and in the graphics display, thus facilitating work with large data models. Modeling involves use of a graphical editor, which allows to design data models.



ABAP/4 Dictionary

The ABAP/4 dictionary describes the logical structure of the objects used in the application development & shows how they are mapped to the underlying relational database in tables or views completely integrated with ABAP/4 Development Workbench. Tables, Structures and Views are the three most important objects of ABAP/ 4 dictionary. The basic objects for defining data in the ABAP/4 dictionary are tables, domains & data elements. Once these objects are defined & activated they are available to all system components thus ensuring data consistency, integrity & security.



ABAP Programming language

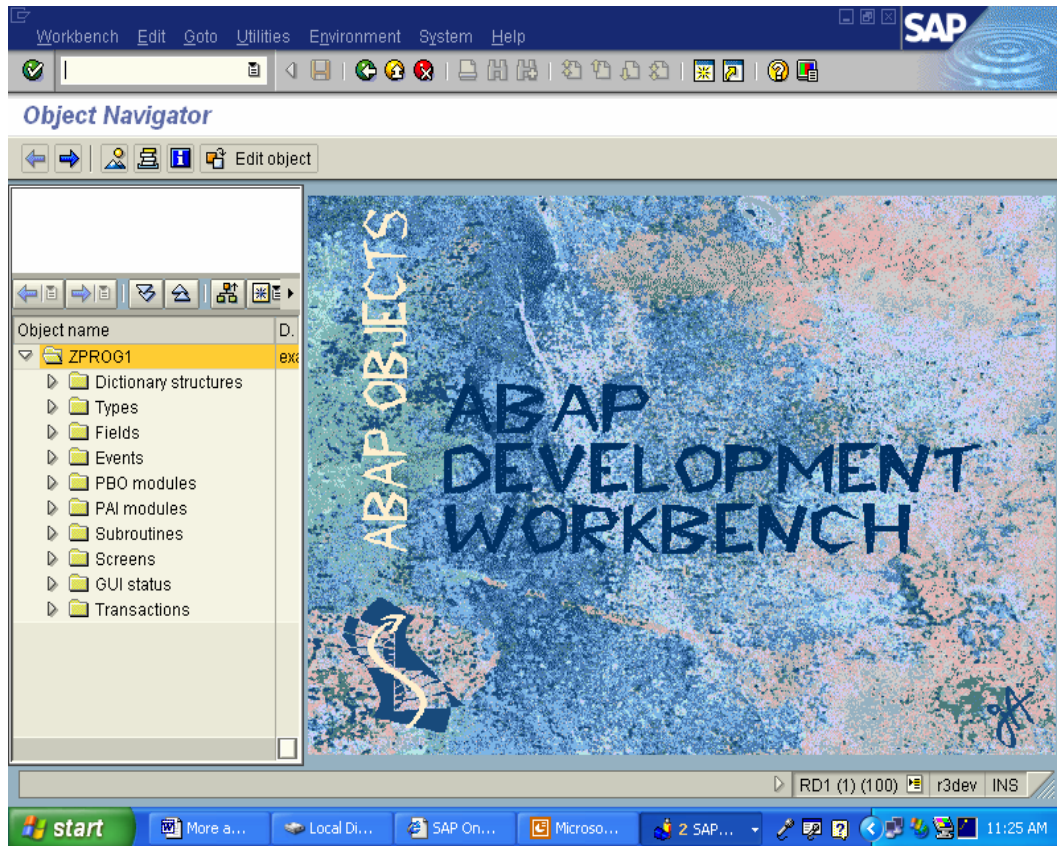
ABAP/4 is a programming language developed by SAP for the interactive development of application programs.

- ABAP/4 is a fourth-generation programming language that is the backbone of the R/3 System.
- The language is event-driven with user actions and system events controlling the execution of the applications.
- You can use the ABAP/4 programming language to create entirely new Client / server applications, as well as extend existing R/3 modules, which is the most common use of the ABAP/4 language.

The language links to an integrated Data Dictionary containing type definitions and data structure that the developer can use in all programs. The Dictionary contains over 8000 table definitions, most related to the business modules of the SAP system. SAP provides development models to assist programmers understand concepts and develop skills.

These models are realistic business models with the relevant database tables populated with huge amounts of realistic data. The language is event-driven; that is, user actions and system events control the application's execution. Thus developers can develop complex interactive programs that support "drill-down" events.

The Function Module library has efficient searching tools allowing developers to find appropriate modules to incorporate into their programs. Embedded in the language is a subset of Structured Query Language (SQL) called Open SQL. Open SQL and the database interface of the R/3 system form layers between the Database Management System and the application program. This layered architecture enables the developer to concentrate on conceptual aspects rather than worry about things like memory management, pointer arithmetic or network issues.



ABAP Editor

The ABAP/4 Editor can be used in any one of three available modes: \

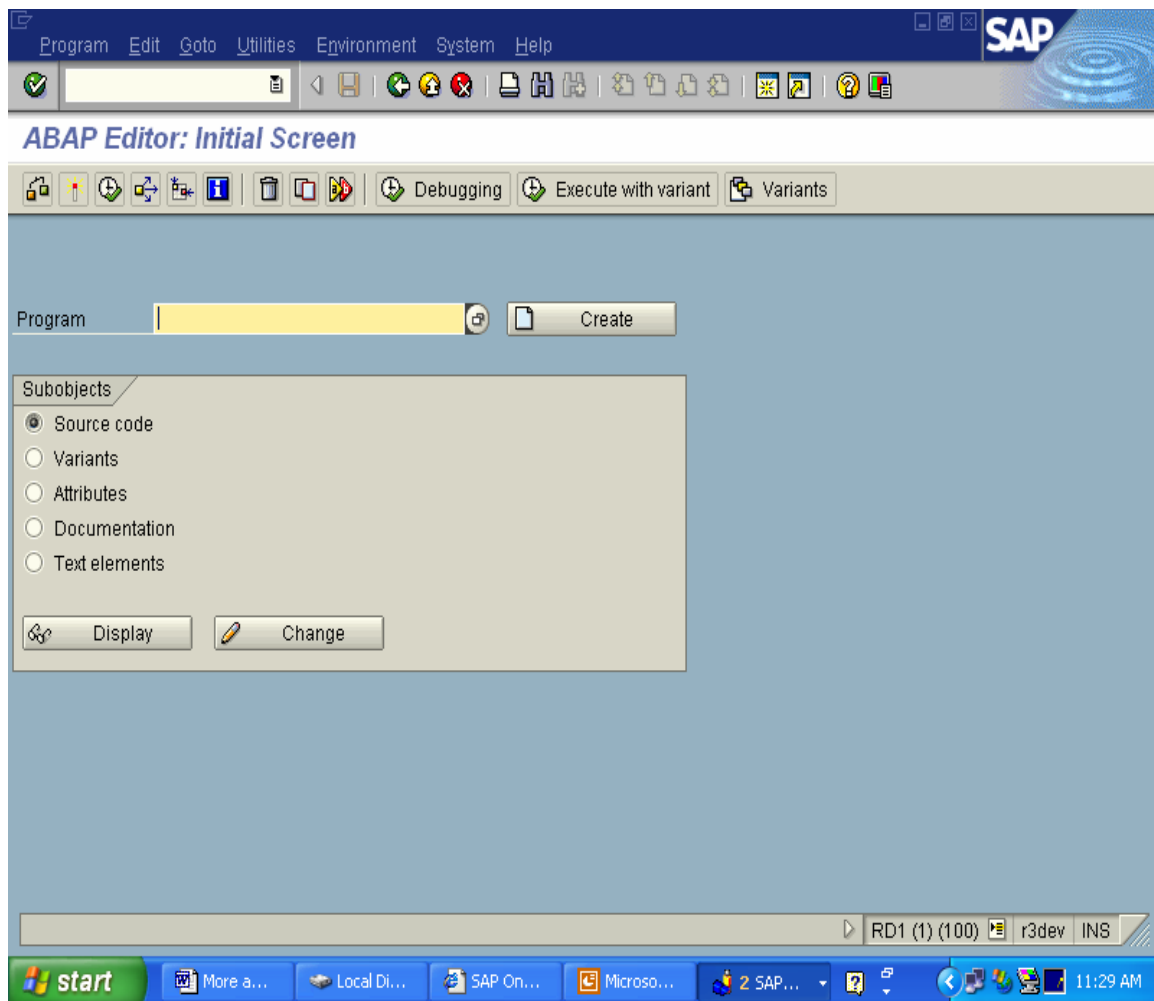
PC Mode: It provides word-processing style copy, cut, and paste commands.

PC Mode with Line Numbers: It provides word-processing style copy, cut, and paste commands but with the addition of line numbers. This is the default mode.

Command Mode: It supplies the same functionality associated with the R/2 version of the SAP enterprise software.

The various sample programs illustrated in these lessons assume that you are in the Editor in PC Mode with Line Numbering. In the PC Mode, you can execute all functions with the mouse.

Once ABAP/4 Editor has been accessed, switching between editor modes can be one by choosing Settings Editor mode.

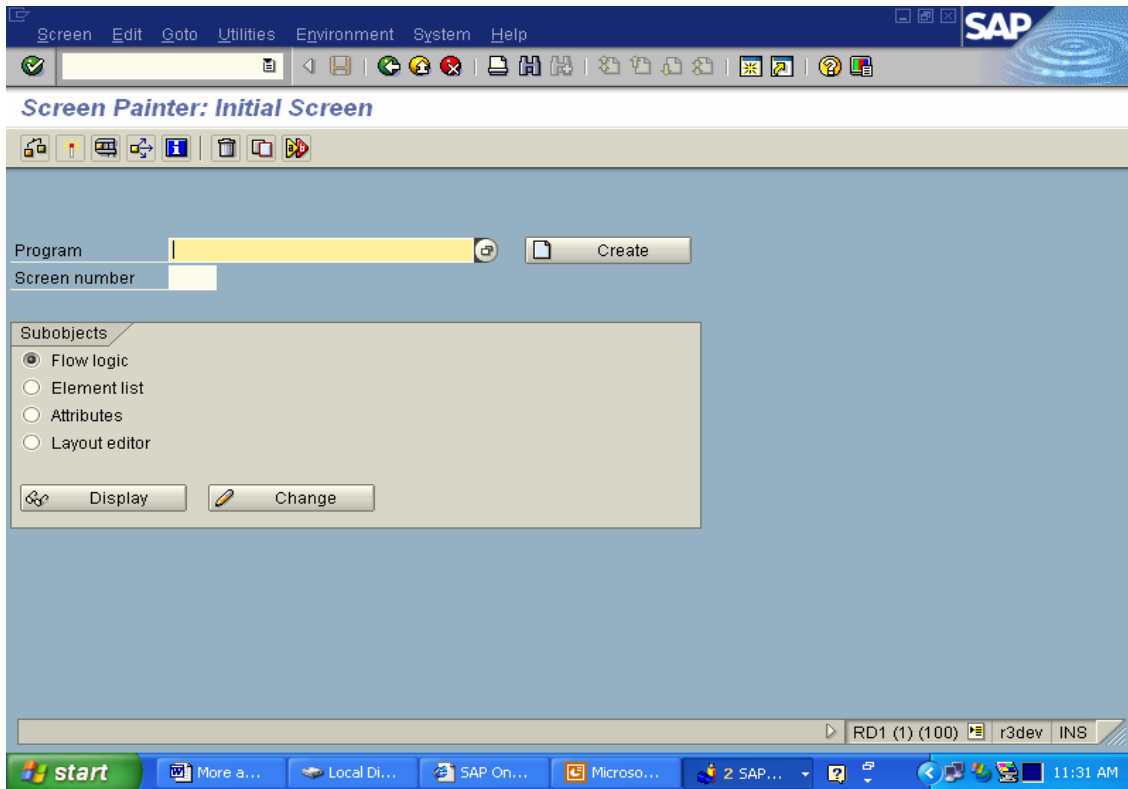


Screen Painter

The Screen Painter is an ABAP Workbench tool that allows to create screens for transactions. It can be used both to create the screen itself, with fields and other graphical elements, and to write the flow logic behind the screen. In German, SAP screens are sometimes referred to as “dynopros” means combination of the screen & its accompanying flow logic. The Screen Painter is used to create and maintain all elements of a screen.

Screen Attributes	Describe a screen object in the R/3 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.
Elements	Correspond to screen elements. Fields are defined in the ABAP Dictionary or in your program.
Flow logic	Controls the flow of your program.

The Screen Painter has a *layout editor* that you use to design your screen layout. It works in two modes namely Graphical mode & Alphanumeric mode. Both modes offer the same functions but use different interfaces. In graphical mode, you use a drag and drop interface similar to a drawing tool. In alphanumeric mode, you use your keyboard and menus. Graphical mode is available only on MS Windows 95, MS Windows NT, and Unix / Motif platforms.

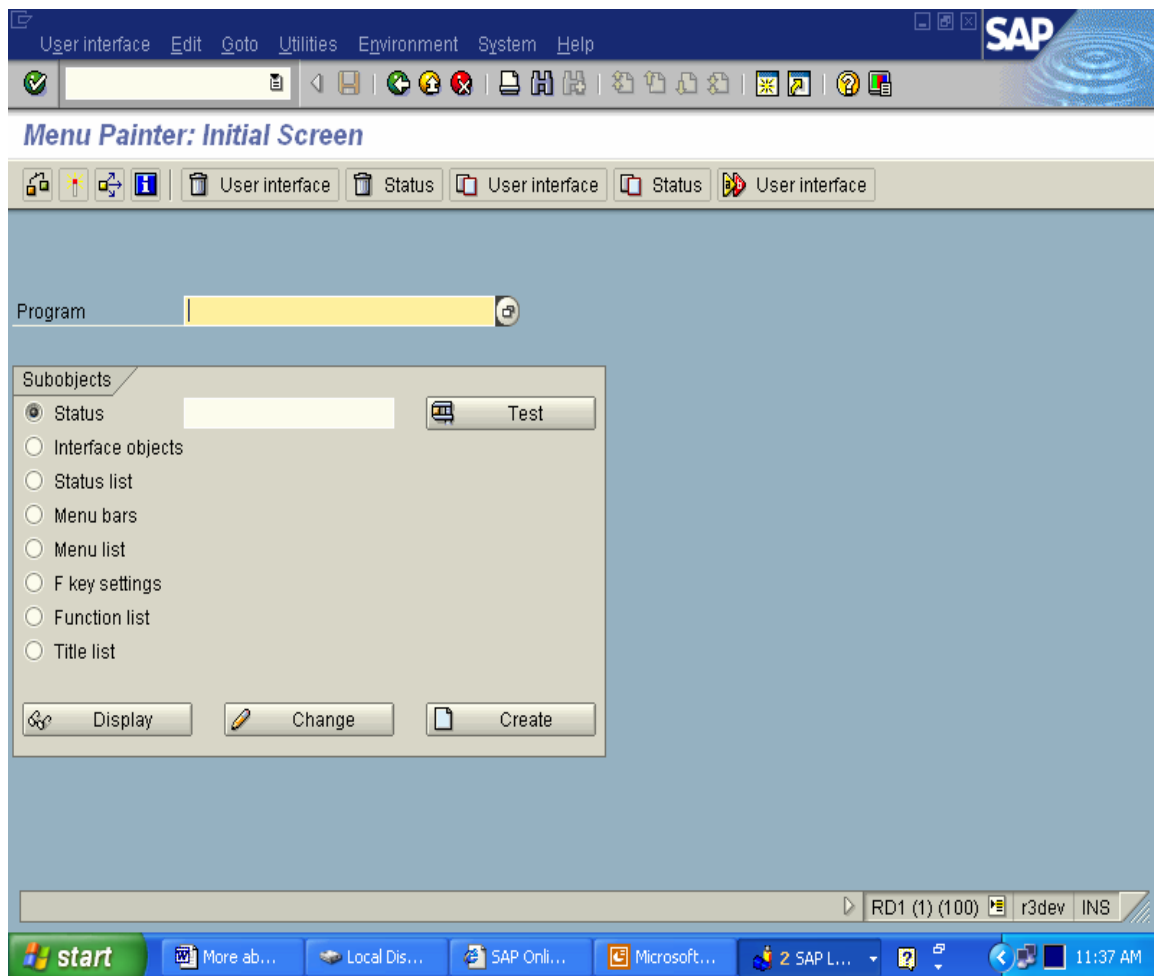


Menu Painter

ABAP programs contain a wide variety of functions, which fall into different categories within the user interface. It is important for users to be able to differentiate between these categories, and to choose the right function easily. In the R/3 System, you arrange functions using the Menu Painter.

An instance of the user interface, consisting of a menu bar, a standard toolbar, an application toolbar, and a function key setting, is called a **GUI status**. The **GUI status** and **GUI title** defines how the user interface will look and behave in an ABAP program.

The principal object of a user interface is the GUI status. This can be assigned to any screen (screen, selection screen, or list). Technically, it consists of a **reference** to a menu bar, a standard toolbar, and a function key setting.



Data Browser

The Data Browser is a tool for retrieving information about a table without using an ABAP program. You can browse the contents of a table and branch from a specific entry to its related check table entries. If the table attributes allow it, you can also create or update table records with the Data Browser. The Data Browser is used to access table entries without using an ABAP program. The Data Browser can be used for displaying table records, all table field values & related text field values & branch from table entries to their related check-table entries. If a table has *Table maintenance allowed* set creation or updation of table records becomes possible with the Data Browser. To start the Data Browser, choose *Overview-> Data Browser* from the Workbench tools initial screen. We can reach the Data Browser from the *Environment* menu in the Repository Browser or the *Utilities* menu in the ABAP Editor.

The screenshot shows the SAP Data Browser interface for Table ZICE_DIARY_D. The title bar reads "Data Browser: Table ZICE_DIARY_D: Selection Screen". The interface includes a menu bar (Program, Edit, Goto, Settings, System, Help) and a toolbar with various icons. Below the title bar, there is a section for "Number of entries" with a search icon. The main area contains a list of fields with input boxes and "to" labels, indicating a range selection. The fields are: DIARY_NO, LINE_NO, RECIVED_FROM, RECIVED_DEPT, RECIVED_DATE, RECIVED_COURIER, RECIVED_COUR_NO, SEND_DEPT, SEND_DATE, SEND_COURIER, and STATUS. At the bottom, there are input boxes for "Width of output list" (set to 250) and "Maximum no. of hits" (set to 200). The status bar at the bottom shows "RD1 (4) (100)" and "r3dev INS".

Field	Value	to	Value
DIARY_NO			
LINE_NO			
RECIVED_FROM			
RECIVED_DEPT			
RECIVED_DATE			
RECIVED_COURIER			
RECIVED_COUR_NO			
SEND_DEPT			
SEND_DATE			
SEND_COURIER			
STATUS			

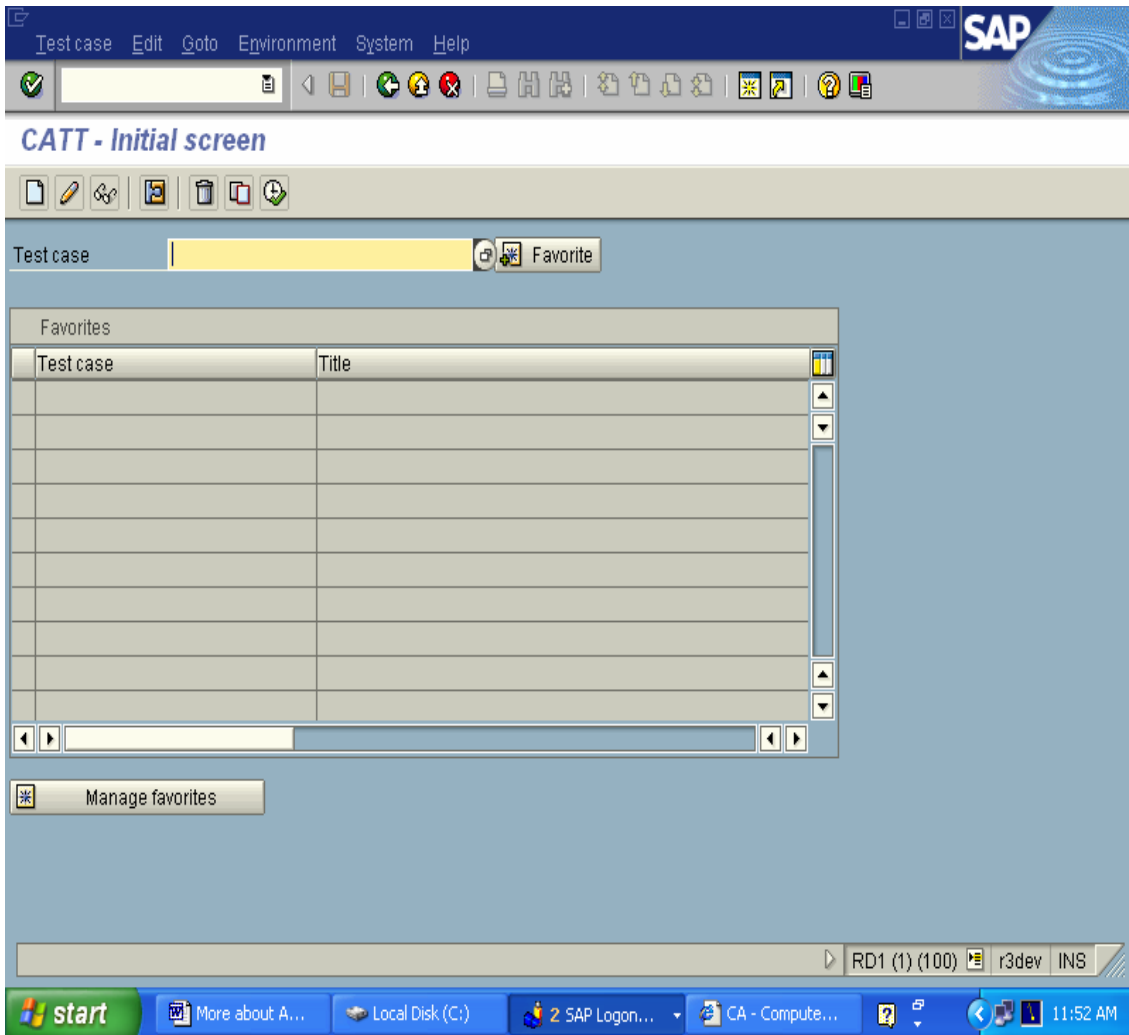
Width of output list: 250
Maximum no. of hits: 200

Computer Aided Test Tool

CATT is delivered as an integral component of the ABAP/4 Workbench since R/3 Release 3.0. It contains all functions required for creating, running, maintaining, and logging test procedures. ABAP/4 Workbench infrastructure provides functions such as the correction or transport systems, the Repository information system, or the multi-lingual applications. A correct test run is guaranteed by synchronous update and explicit table buffer refresh. This is especially important for transaction sequences in which transactions use the results of previous transactions. The test procedure runs in the central

R/3 system and controls the transactions via Remote Function Call (RFC) in the satellite systems. It can be used to test transactions, check system messages, check values & database updates, setup customizing tables. The CATT logs contain all information relevant for the test run and are stored centrally in the database of the executing R/3 system.

- CATT is integrated into R/3 infrastructure.
- CATT provides a user-friendly recording facility.
- CATT has an expert in which complex test scenarios can be created.
- The modular CATT concept minimizes test creation resource requirements.
- CATT performs authorization checks.
- CATT logs test results and stores the logs centrally in the database.



Existing System

ONGC is a very large organization spread over a wide geographic area in India and as well as abroad. The means of communication between the rigs (where drilling, testing and entire production of oil is been done) is of prime importance. Considering this ONGC has installed Ku Band Links from all remote locations. After the implementation of SAP these remote locations are now connected to the system. There is no existing system in ONGC to carry out a complete analysis of the Ku Band with respect to its utilization. The only information available for these links is through HP Open View software which can tell if a particular link is down or not.

Since the number of remote locations is very large there is an urgent need for creation of a system to monitor the usage of Ku Band. The problems been faced today are

- ❖ Querying such a bulky data manually is tedious job and may be error prone.
- ❖ The essential information/exceptional reporting required by the management are not available readily.
- ❖ There is a delay in compiling available data into meaningful reports.
- ❖ Security, integrity and privacy of data is not maintained.
- ❖ Data redundancy.
- ❖ Information is not available at proper time and it delays the overall schedule.

Objectives of Proposed System

To create a system that can monitor the utilization of all the Ku Band Users across various remote locations. ONGC has approximately 225 remote locations spread across India from where the users log on the SAP system.

To maintain the record of users using Ku Band and to generate different reports as per the need of the management.

The complete analysis of Ku Band usage is necessary as it costs a lot for ONGC to lease new lines and also it helps to monitor the bandwidth of current usage by the employees.

This system should give the details of the user utilizing the Ku Band such as the logon time, company code, business area etc.

The efficiency of the system lies in the database management so as all the records for the system are properly managed and within a fraction of second all the necessary reports are generated.

Feasibility Study

We do the feasibility study of our software from each and every aspect so that there should be no complications during the working of the system. The whole design of the software is based on the results of the initial investigation.

Feasibility study is a test of a system proposal according to its workability impact on the organization, ability to meet user needs and effective use of resources.

The objective of the feasibility study is not to solve the problem but to acquire a sense of its scope. During the study the problem definition is crystallized and aspects of the problem to be included in the system are determined.

The result of a feasibility study is a formal proposal. This is simply a report – a formal document detailing the nature and scope of the proposed solution. The proposal summarizes what is known and what is going to be done which is developed by is a synopsis for the application software.

Technical Feasibility

The technical feasibility of a system determines whether the technology can be integrated into organization. Technical evolution must also asses whether the existing system can be upgraded to use the new technology and whether the organization has the expertise to use it.

The ONGC is already equipped with all the hardware and software. And the non existence of any current system for this particular problem necessitates the need for a new system.

Operational Feasibility

There are two aspects of the operational feasibility of a system. One aspect is a technical performance and the other is acceptance. Technical performance determines whether the system can provide correct and timely information as required by the organization personnel. Creating a new system to manage the utilization of Ku Band will ensure that accurate, up-to-date and timely information is provided to users. Acceptance of a new system revolves around the current system and the personnel using it. This new system will automate report generation and make it user friendly. The system also cuts down paper work and timely delay of the existing manual system.

Economical Feasibility

The economical feasibility of the system looks upon the financial aspect of the project. It determines whether the investment that goes into the project is recoverable. The cost-benefit analysis is a commonly used method in evaluating the effectiveness of the system. As hardware and software are already available and so investment is to be made in that direction. This system only generates desired reports fetching the data from the database and by manipulating it, hence no extra amount has to be spent.

Project Risk

This software is tailored software which can be altered according to the desire and need of the users interacting with it but as every system has some of its drawbacks lying in its risk of failure; since it's a completely new system the risk in its implementation is minimal. The only risk that could lead to a failure of this system could be that of the connectivity with the database.

System Design

General Design Issues

The most creative and challenging phase of the system development life cycle is system design. It refers to the technical specification that will apply in implementing the candidate system that fulfills the requirements of the user, which in turn are identified during the analysis phase. On the basis of system analysis the system is designed following a top down approach. Functional requirements, modular design, data dictionary and input/output format are carefully analyzed and designed so that it is easy to understand the whole system by going through the design of the system. Modular approach is taken into consideration so that there is minimum coupling and maximum cohesion.

System design was used for the creation of the proposed system in which the logical and detailed structure of the “Ku Band Link Utilization Analysis System”, which was useful in coding the system in the later stage. System design is the most creative and challenging phase of the software development cycle. It provides the understanding and procedural details necessary for its implementation. It includes the designing of database, inputs, outputs and security.

The design forms the blueprint of the system and how the components relate to each other. The flow has been meticulously designed in order to maximize the efficiency. The system design transforms a logical representation of what a given system is required to do into physical specifications. The specifications are converted to a physical reality during development. Design and specification of the design are in accordance with the prescribed rules and practices of an organization. A structural design is followed in the system design. It's a data flow based methodology. This approach begins with a system specification that identifies inputs and outputs and describes the functional aspects of the system.

Database Design

The general theme behind database is to handle information as an integrated whole. A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and efficiently. The general objective is to make information access easy, quick and efficient. The main objectives of designing a database are:

Controlled Redundancy:

Redundant data occupies space and therefore is wasteful, if versions of the same data are in different phases of updating; the system often gives conflicting information. A unique aspect of database design is storing data only once or with minimum redundancy.

Ease of Learning and Use:

A major feature of user friendly database package is how easy it is to learn and use. Related to this point is ways in which the database can be modified without interfering with established ways of using data.

Data Independence:

An important database objective is changing hardware and storage procedures or adding new data without having to re-write application program.

More Information at Low Cost:

Using, storing and modifying data at low cost are important. Although hardware prices are falling, software and programming costs are on the rise. This means programming and software enhancements should be kept simple and easy to update.

Accuracy and Integrity:

The accuracy of a database ensures that data quality and content remain constant. Integrity controls data inaccuracies where they occur.

Performance:

This objective emphasizes response time to inquiries suitable to the use of data. How satisfactory the response time is depends on the nature of the user-database dialogue. For example, inquiries regarding updating the Ku Band Users Master Table should be handled in a few seconds.

Security Design:

There are a lot of security features available in ABAP. Only those users who have valid user-id and user-password can navigate and update the database that too if the particular transaction is maintained in their role.

Tables Used:

The database design for this project comprises of the following three tables:

S.No.	Name of table	Description
1	Usr41	User Additional Data table
2	Zbc_kub_locmst	Location master table
3	Zbc_kub_login	Login data table

Of these usr41 is present in sap to monitor the logon details and location of the user as well as the application server to which logged on. The other two were designed keeping this project in mind. The design of the tables along with their attributes, data types and their respective size is shown on the following pages.

❖ **Usr41:-** User master table showing details.

Dictionary: Display Table

Transparent table: **USR41** Active

Short description: User master: Additional data

Attributes Fields Currency/quant. fields

New rows Data element/Direct type

Fields	Key	Init.	Field type	Data...	Lgth.	Dec.p...	Check table	Short text
MANDT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	MANDT	CLNT	3	0		Client
BNAME	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	XUBNAME	CHAR	12	0		User name in user m
TERMID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	XUTERMID	INT4	10	0		Terminal ID
SERVER	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	XUSERVER	CHAR	20	0		Server
TERMINAL	<input type="checkbox"/>	<input checked="" type="checkbox"/>	XUTERMINAL	CHAR	36	0		Terminal
SPRACHE	<input type="checkbox"/>	<input type="checkbox"/>	XULANGU	LANG	1	0		Language
LOGON DATE	<input type="checkbox"/>	<input type="checkbox"/>	XULDATE	DATS	8	0		Last logon date
LOGON TIME	<input type="checkbox"/>	<input type="checkbox"/>	XULTIME	TIMS	6	0		Last logon time

RD1 (1) (220) r3dev INS

❖ **Zbc_kub_locmst:-** Location master table showing details

SAP

Table Edit Goto Utilities Extras Environment System Help

Dictionary: Display Table

Transparent table: ZBC_KUB_LOCMST Active
Short description BC : Store Locations and IP Details of KUBand Links

Attributes Fields Currency/quant. fields

Fields	Key	Init.	Field type	Data...	Lgth.	Dec.p...	Check table	Short text
MANDT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	MANDT	CLNT	3	0		Client
IPADDRESS	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	IPADDRESS	CHAR	15	0		IP address connector
REMOTE_LOC	<input type="checkbox"/>	<input checked="" type="checkbox"/>	DESC30	CHAR	30	0		Description
LOCATION	<input type="checkbox"/>	<input checked="" type="checkbox"/>	BUKRS	CHAR	4	0		Company Code
REGION	<input type="checkbox"/>	<input checked="" type="checkbox"/>	ZREGIONK	CHAR	3	0		Region Code for KU t

RD1 (1) (220) r3dev INS

❖ Zbc_kub_login:- Data table login information

SAP

Table Edit Goto Utilities Extras Environment System Help

Dictionary: Display Table

Transparent table: ZBC_KUB_LOGIN Active

Short description: BC : Store Login Stats of users logon thru Ku-Band

Attributes Fields Currency/quant. fields

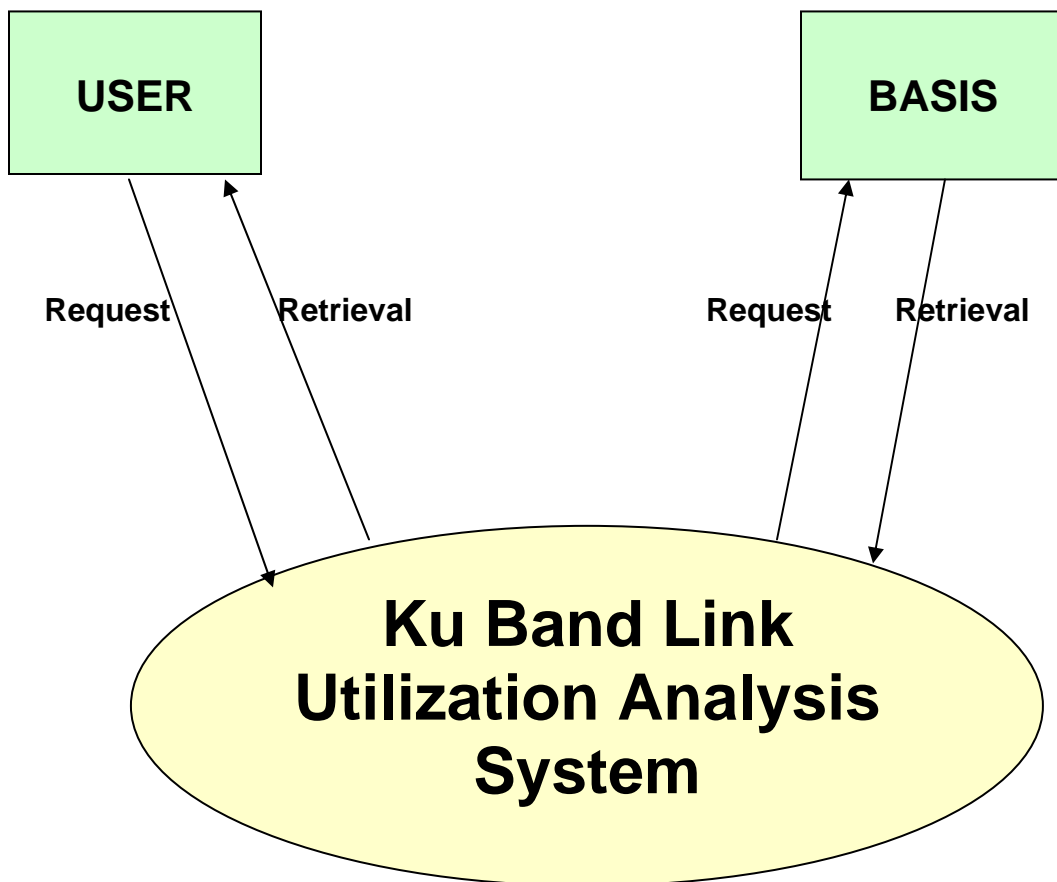
New rows Data element/Direct type

Fields	Key	Init	Field type	Data...	Lgth.	Dec.p...	Check table	Short text
MANDT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	MANDT	CLNT	3	0		Client
BNAME	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	XUBNAME	CHAR	12	0		User name in user m
IPADDRESS	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	IPADDRESS	CHAR	15	0		IP address connecte
TERMINALNAME	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	SADATERMID	CHAR	18	0		Terminal ID
LOGON DATE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	XULDATE	DATS	8	0		Last logon date
LOGON TIME	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	XULTIME	TIMS	6	0		Last logon time
TIME	<input type="checkbox"/>	<input type="checkbox"/>	ADDI TIME2	TIMS	6	0		Time at which data re
TIME INTERVAL	<input type="checkbox"/>	<input type="checkbox"/>	SECONDS	INT4	10	0		Number in seconds

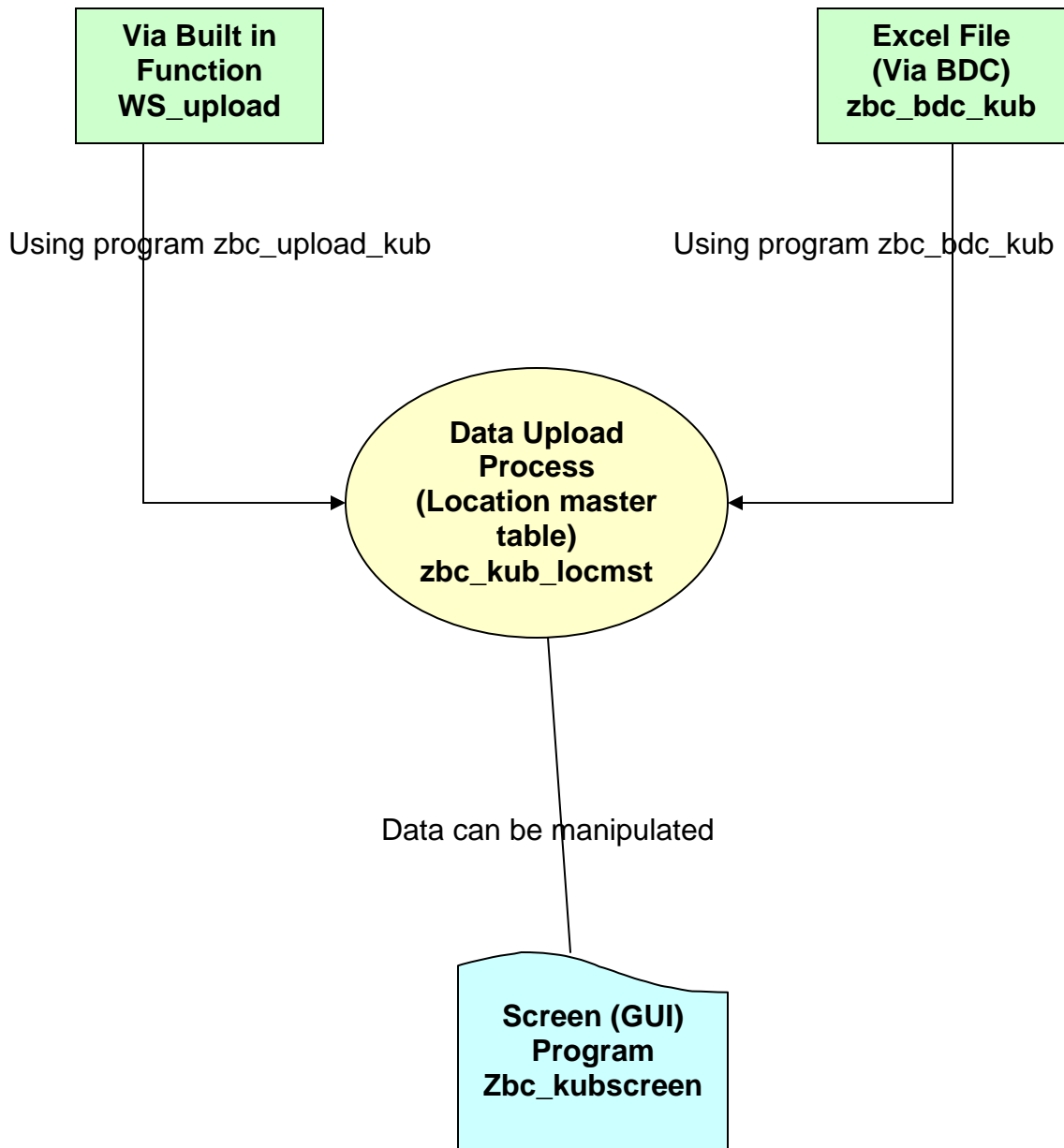
RD1 (1) (220) r3dev INS

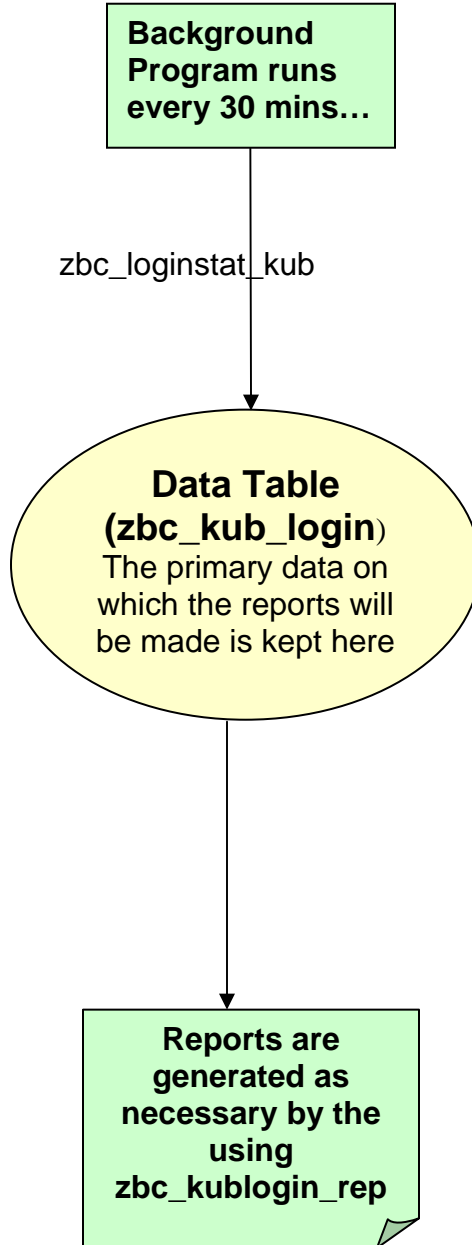
start report.doc - Microsof... Dictionary: Display T... HTML Builder - Mozilla... 11:36 AM

Context Analysis Diagram



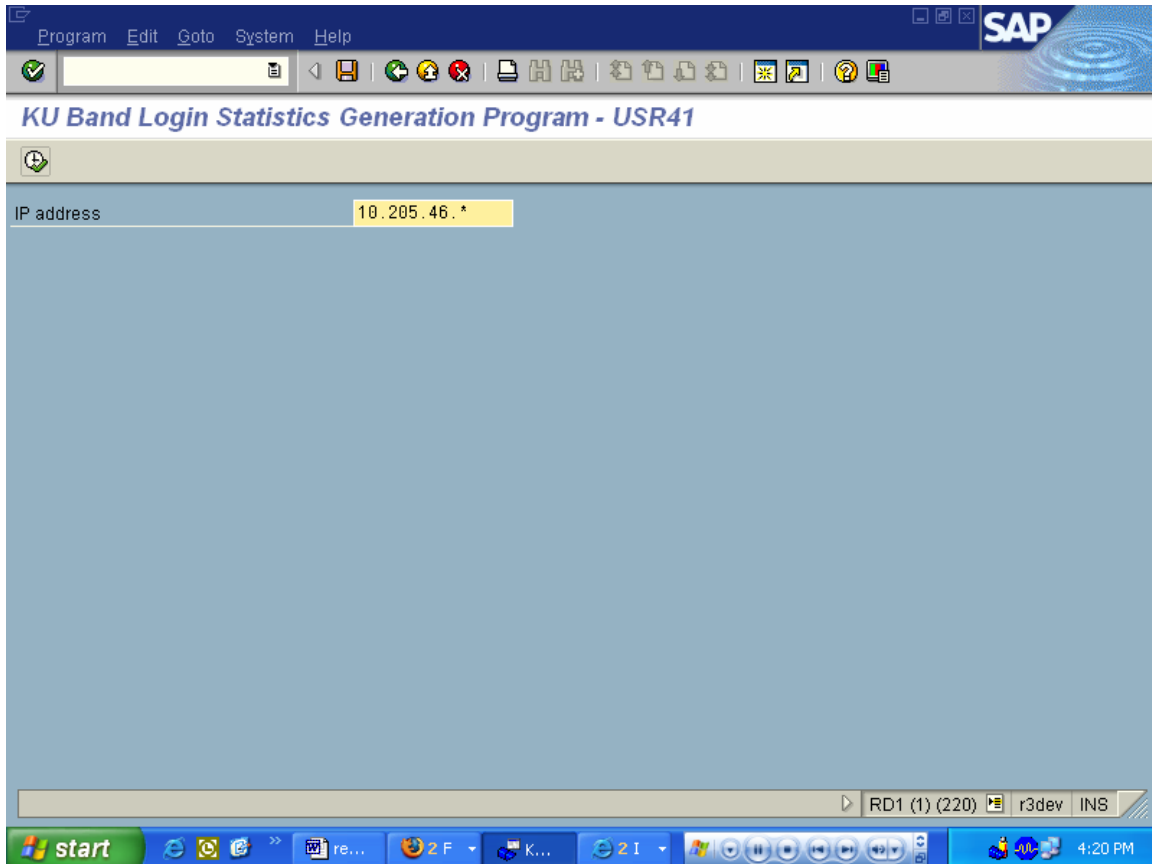
Data Flow Diagram





Zbc_loginstat_kub:-

Enter the ipaddress to generate the statistics. Proposed use is to save a variant and then schedule the program to run after every 30 minutes.



Zbc_kubscreen:-

This program presents a screen which can be used for maintaining the location master table. Different commands include creating, changing, displaying and deleting locations.

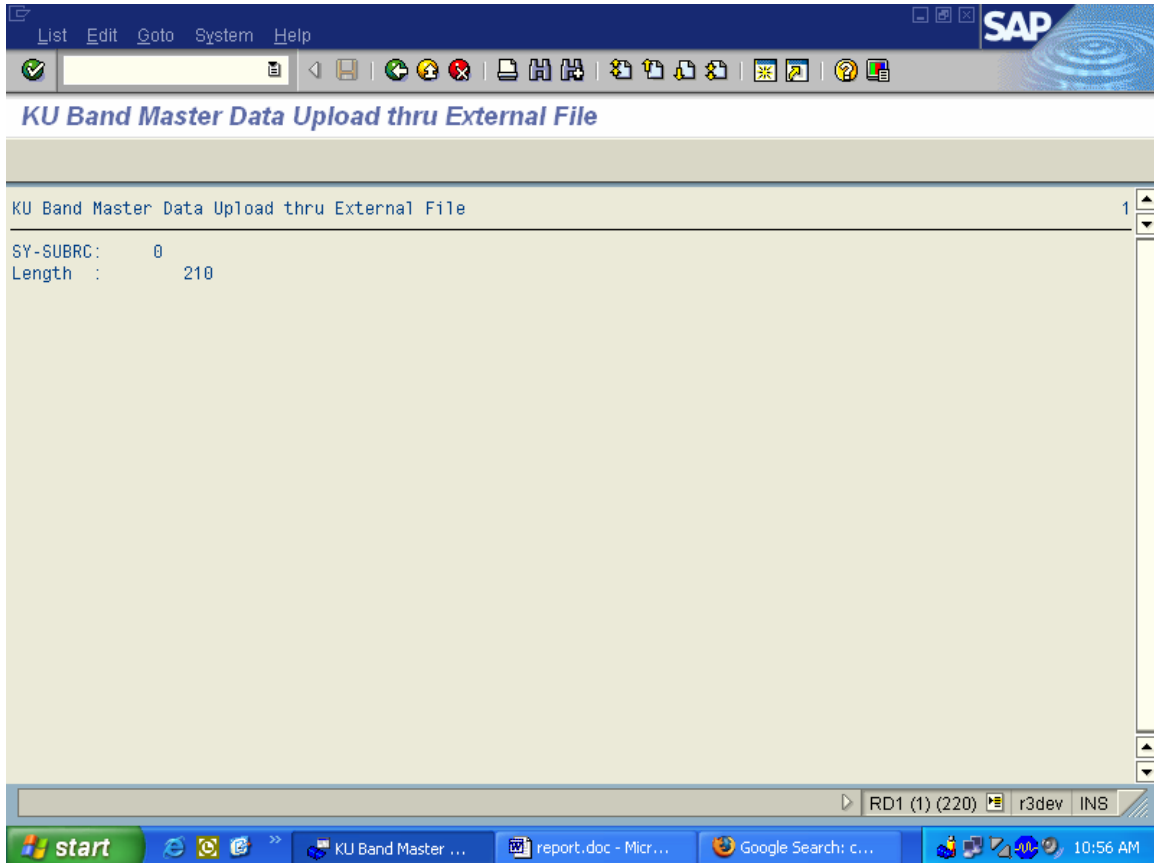
The screenshot displays the SAP R/3 interface for the Zbc_kubscreen program. The top bar includes the SAP logo and a menu with 'System' and 'Help'. Below this is a toolbar with various icons for file operations and navigation. The main header area shows 'SAP R/3' and a set of action buttons: 'Create', 'Change', 'Display', 'Delete', and 'Refresh'. The central area is titled 'KU Band Master Data' and contains a form with the following fields:

KU Band Master Data	
IP address	<input type="text"/>
Description	<input type="text"/>
Company Code	<input type="text"/>
Region	<input type="text"/>

At the bottom of the window, a status bar shows 'RD1 (1) (220)' and 'r3dev INS'. The Windows taskbar at the very bottom includes the 'start' button, several application icons, and the system clock showing '5:20 PM'.

Zbc_kub_upload:-

This program uploads the data form the presentation pc to the database using function pool ws_upload.



Zbc_bdc_kub:-

This program uploads the data from an external file using BDC.

The screenshot shows the SAP BDC program 'BDC for zbc_kub_mst' running in a window. The interface includes a menu bar (Program, Edit, Goto, System, Help) and a toolbar. The main area is titled 'BDC program for uploading Ipaddress and Location'. It features a 'File name' input field with a browse button. Below this, there are two radio buttons: 'Generate session' (selected) and 'Call transaction'. Under 'Generate session', there are fields for 'Session name' (CAB_CPSHARMA), 'User' (CAB_CPSHARMA), 'Keep session' (checkbox), and 'Lock date'. Under 'Call transaction', there are fields for 'Run mode' (N), 'Update session' (L), 'Error sessn' (input field), 'User' (CAB_CPSHARMA), 'Keep session' (checkbox), and 'Lock date'. At the bottom, there are fields for 'Nodata indicator' (/), 'SMALLLOG' (checkbox), and 'DATASET' (input field). The status bar at the bottom shows 'RD1 (1) (220)', 'r3dev', and 'INS'. The Windows taskbar at the very bottom shows the start button and several open applications, including 'BDC for zbc_...', 'report.doc - ...', 'Get Links - Li...', and 'Asankhaya', with the time '11:23 AM'.

Program Edit Goto System Help

SAP

BDC for zbc_kub_mst

BDC program for uploading Ipaddress and Location

File name

☒ Generate session

Session name CAB_CPSHARMA

User CAB_CPSHARMA

Keep session ☐

Lock date

☐ Call transaction

Run mode N

Update session L

Error sessn

User CAB_CPSHARMA

Keep session ☐

Lock date

Nodata indicator /

SMALLLOG ☐

DATASET

RD1 (1) (220) r3dev INS

start BDC for zbc_... report.doc - ... Get Links - Li... Asankhaya 11:23 AM

Zbc_kublogin_rep:-

This program generates the reports as per the requirements. The different radio buttons provide options for users to generate reports as per the requirement.

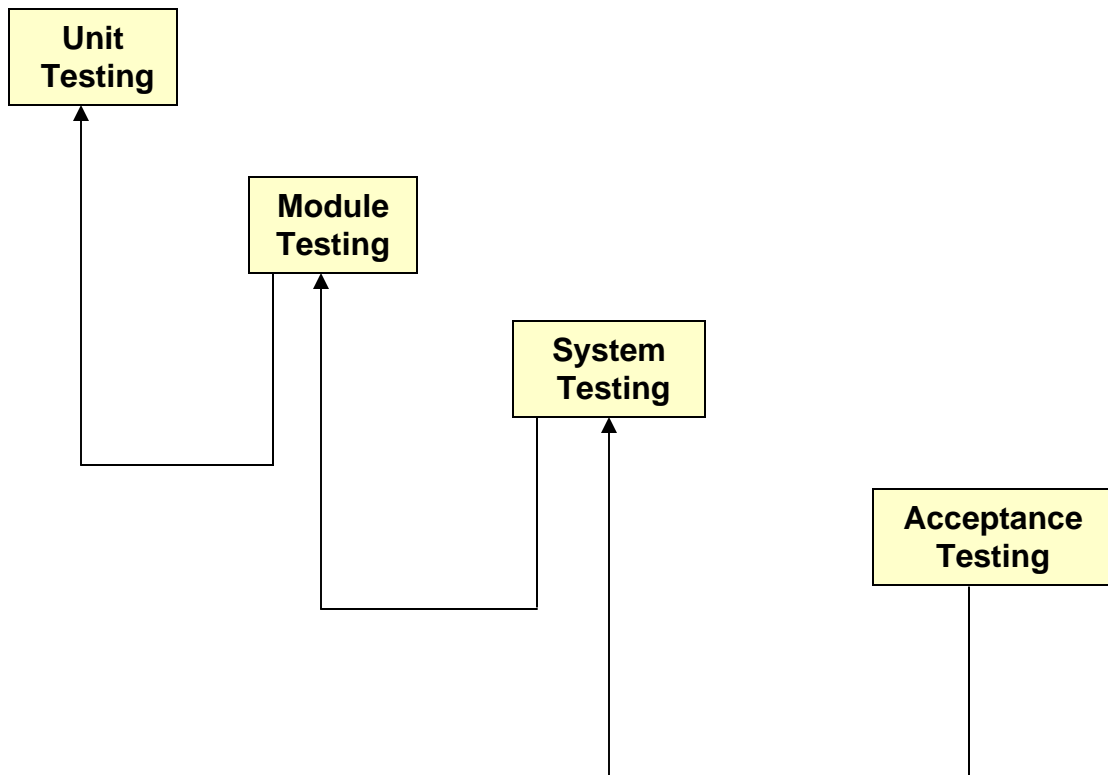
The following reports can be generated:-

- Current Ku Band Users
- Usage of Ku Band according to IP Address
- Usage of Ku Band according to IP Address and Name
- Company Code Wise summary (you can get detailed reports by clicking the hotspots in the summary list down to the IP address)
- Region Wise Summary (you can get detailed reports by clicking the hotspots in the summary list down to the IP address)

The screenshot shows the SAP 'KU Band Link Utilisation Report' window. The title bar includes 'Program Edit Goto System Help' and the SAP logo. Below the title bar is a menu bar with icons for file operations. The main content area is divided into two sections. The first section, 'Select Options : IPAddress List', contains three rows of input fields: 'IP address' (with a yellow highlight), 'Last logon date', and 'Region'. Each row has a 'to' field and a right arrow button. The second section, 'Choose Radio Button : Report Option', contains five radio buttons: 'Current KU Band Users.' (selected), 'IP Address Wise KU Band Usage', 'IP Add/User Wise KU Band Usage', 'CmpCd Wise KU Band Usage', and 'Region Wise KU Band Usage'. At the bottom of the window, there is a status bar showing 'RD1 (1) (220)' and 'r3dev INS'. The Windows taskbar at the very bottom shows the start button, several open applications (including 'KU Band Link...', 'report.doc - ...', 'Get Links - A...', and 'https://mem...'), and the system clock showing '11:55 AM'.

Testing

Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct, the goal will be successfully achieved.



Standard ABAP Extended Syntax check:-

ABAP has a built in extended syntax checking tool which checks the program for several errors and warnings such as

- PERFORM/FORM Interfaces
- CALL FUCNTION Interfaces
- Screen Consistency
- Portability
- Parameter IDs.

The screenshot shows the SAP SLIN overview window. The title bar includes 'Checks', 'Edit', 'Goto', 'System', and 'Help'. The main area displays a table with the following data:

Check for program ZBC_KUBLOGIN_REP	Error	Warnings	Messages
Fatal errors	0	0	0
PERFORM/FORM interfaces	0	3	0
CALL FUNCTION interfaces	0	0	0
External program interfaces	0	0	0
Screen consistency	0	0	0
Authorizations	0	0	0
PF-STATUS and TITLEBAR	0	0	0
SET/GET parameter IDs	0	0	0
MESSAGE	0	1	0
Output CURR/QUAN fields	0	0	0
Field attributes	0	10	0
Breakpoints	0	0	0
Syntax check warnings	0	0	0
Portability ASCII/EBCDIC	0	0	0
Check load sizes	0	0	0
Multilingual capability	0	0	0
Other	0	0	1
Hidden errors and warnings	0	0	0

The bottom of the window shows the SAP taskbar with the 'start' button and several open applications: 'SLIN overview', 'report.doc - Microsof...', and 'Google Search: date i...'. The system clock indicates 12:50 PM.

Conclusion

“Ku Band Link utilization analysis system” has successfully met the requirement of the organization. The new system provides up-to-date analysis tool for the users. The features incorporated in the system are up to their requirements and will add to their efficiency and productively.

There are many advantages of the new system. During product development a special emphasis has been given to the user friendly feature of the system.

Throughout the development all the industry standards have been met. Coding had been done as per the client specified requirements.

Bibliography

Books and References:

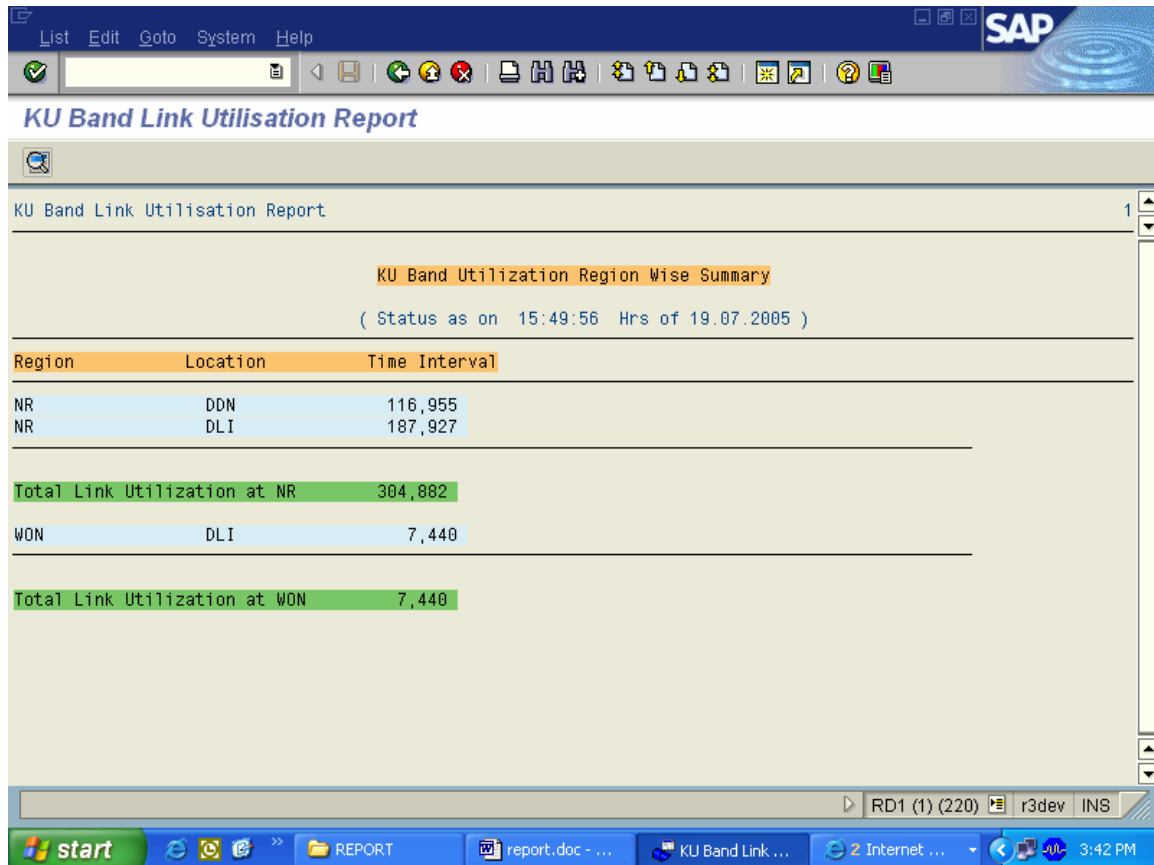
- Title: ABAP objects
Publisher: Pearson Education
Authors: Dr. Horst Keller and Sascha Kruger
- Title: BC-ABAP
Publisher: SAP AG Press

Web References:

- www.sap.com
- www.sapimg.com
- www.abapmirrorz.com
- www.it-minds.com

Sample Output

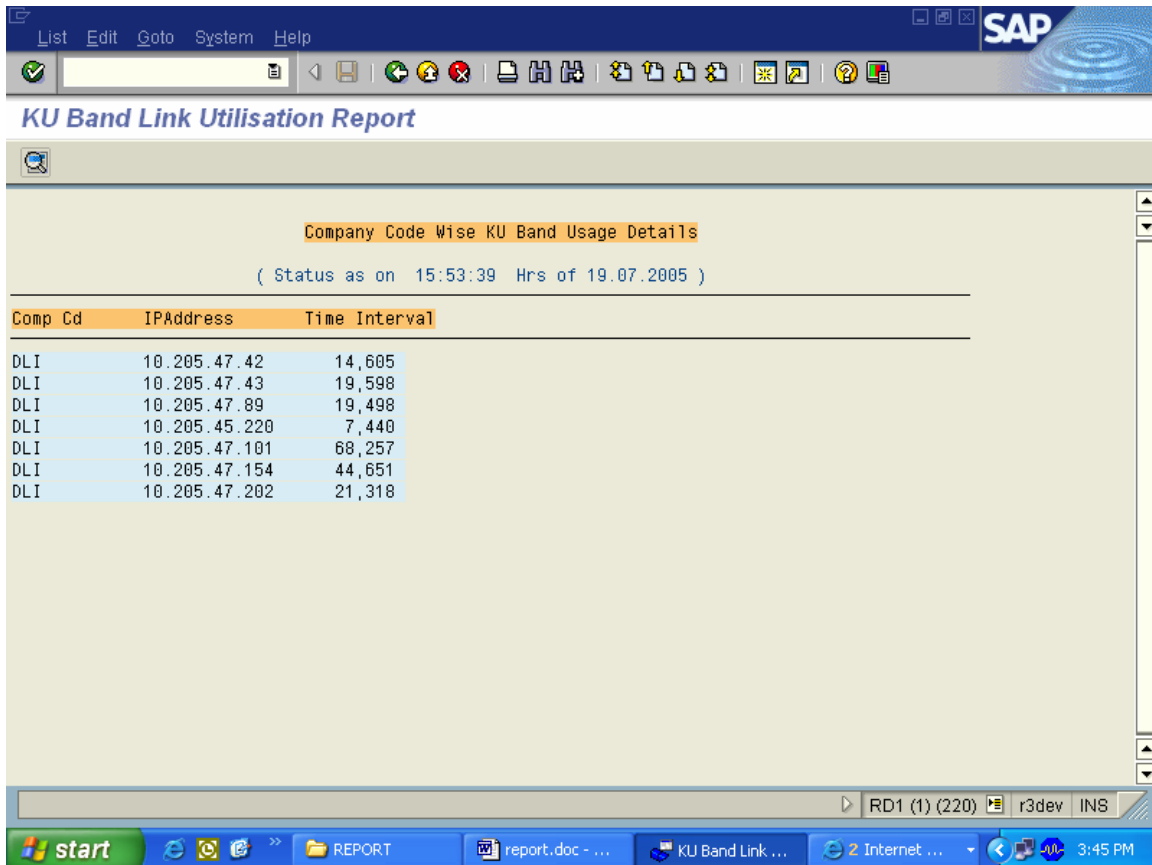
All data here is presented as per the test client.



The screenshot shows the SAP 'KU Band Link Utilisation Report' window. The title bar includes 'List', 'Edit', 'Goto', 'System', and 'Help' menus, along with a toolbar and the SAP logo. The report content is titled 'KU Band Link Utilisation Report' and includes a sub-header 'KU Band Utilization Region Wise Summary' with a status note '(Status as on 15:49:56 Hrs of 19.07.2005)'. The main data is presented in a table with three columns: 'Region', 'Location', and 'Time Interval'. The table lists utilization for regions NR and WON at locations DDN and DLI. Summary rows for 'Total Link Utilization at NR' and 'Total Link Utilization at WON' are highlighted in green. The bottom of the window shows a taskbar with the Windows start button, several open applications (REPORT, report.doc, KU Band Link, Internet Explorer), and a system clock showing 3:42 PM.

Region	Location	Time Interval
NR	DDN	116,955
NR	DLI	187,927
Total Link Utilization at NR		304,882
WON	DLI	7,440
Total Link Utilization at WON		7,440

This report is created when the user clicks the radio button for region wise summary. This report displays the Ku Band Usage region wise. for the next detailed report when the user clicks a hotspot on region a new report is generated.



The screenshot shows the SAP 'KU Band Link Utilisation Report' interface. At the top, there is a menu bar with 'List', 'Edit', 'Goto', 'System', and 'Help'. Below the menu is a toolbar with various icons. The main title 'KU Band Link Utilisation Report' is displayed in a blue header. The report content area has a title 'Company Code Wise KU Band Usage Details' and a status line '(Status as on 15:53:39 Hrs of 19.07.2005)'. A table with three columns: 'Comp Cd', 'IPAddress', and 'Time Interval' is shown. The table contains seven rows of data. The bottom of the window shows a taskbar with the Windows start button, several open applications, and a system clock showing 3:45 PM.

Comp Cd	IPAddress	Time Interval
DLI	10.205.47.42	14,605
DLI	10.205.47.43	19,598
DLI	10.205.47.89	19,498
DLI	10.205.45.220	7,440
DLI	10.205.47.101	68,257
DLI	10.205.47.154	44,651
DLI	10.205.47.202	21,318

This report is the detailed report for a particular company code. Now again if the user clicks the hotspot for a ip address a new report comes as follows.



The screenshot shows the SAP 'KU Band Link Utilisation Report' window. The title bar includes 'List', 'Edit', 'Goto', 'System', and 'Help' menus, along with a toolbar and the SAP logo. The report title is 'KU Band Link Utilisation Report'. Below it, a subtitle reads 'KU Band IP Address/User Wise Detail Report' followed by '(Status as on 15:56:10 Hrs of 19.07.2005)'. The main content area displays a table with four columns: 'SrNo', 'IPAddress', 'User Name', and 'Time Interval'. The table contains six rows of data, all for the IP address '10.205.47.101' and user 'SAB_NITISH'. The 'Time Interval' values are 8,200, 6,716, 2,794, 9,509, 24,612, and 16,426. The bottom status bar shows 'RD1 (1) (220)', 'r3dev', and 'INS'. The Windows taskbar at the bottom includes the 'start' button, several open applications like 'REPORT', 'report.doc - ...', 'KU Band Link ...', and 'Internet ...', and a system clock showing '3:48 PM'.

SrNo	IPAddress	User Name	Time Interval
1	10.205.47.101	SAB_NITISH	8,200
2	10.205.47.101	SAB_NITISH	6,716
3	10.205.47.101	SAB_NITISH	2,794
4	10.205.47.101	SAB_NITISH	9,509
5	10.205.47.101	SAB_NITISH	24,612
6	10.205.47.101	SAB_NITISH	16,426

This is the most detailed report it shows the usage according to ipaddress and username for that particular company code.

Source Code

Program 1:- zbc_kublogin_rep

REPORT zbc_kublogin_rep .

TABLES : zbc_kub_login,
 zbc_kub_locmst.

**

TYPE-POOLS: slis.

TYPES : BEGIN OF ty_ipadd ,
 ipaddress TYPE zbc_kub_login-ipaddress,
 time_interval TYPE zbc_kub_login-time_interval,
 END OF ty_ipadd.

TYPES : BEGIN OF ty_ipadd1 ,
 ipaddress TYPE zbc_kub_login-ipaddress,
 bname TYPE zbc_kub_login-bname,
 time_interval TYPE zbc_kub_login-time_interval,
 END OF ty_ipadd1.

TYPES : BEGIN OF ty_ipadd2 ,
 location TYPE t001-bukrs,
 time_interval TYPE zbc_kub_login-time_interval,
 END OF ty_ipadd2.

TYPES : BEGIN OF ty_ipadd2_det,

```
location TYPE t001-bukrs,  
ipaddress type zbc_kub_login-ipaddress,  
time_interval TYPE zbc_kub_login-time_interval,  
END OF ty_ipadd2_det.
```

```
TYPES : BEGIN OF ty_ipadd3 ,  
    region TYPE zbc_kub_locmst-region,  
    location TYPE t001-bukrs,  
    time_interval TYPE zbc_kub_login-time_interval,  
END OF ty_ipadd3.
```

```
TYPES : BEGIN OF ty_curr_users,  
    bname TYPE usr41-bname,  
    terminal TYPE usr41-terminal,  
    last_name TYPE zusrmst-last_name,  
    first_name TYPE zusrmst-first_name,  
    designation TYPE zusrmst-designation,  
END OF ty_curr_users.
```

DATA :

```
ist_kublogin TYPE STANDARD TABLE OF zbc_kub_login WITH HEADER  
LINE,  
ist_ipadd TYPE STANDARD TABLE OF ty_ipadd WITH HEADER LINE,  
ist_ipadd1 TYPE STANDARD TABLE OF ty_ipadd1 WITH HEADER LINE,  
ist_ipadd2 TYPE STANDARD TABLE OF ty_ipadd2 WITH HEADER LINE,  
ist_ipadd2_det TYPE STANDARD TABLE OF ty_ipadd2_det WITH  
HEADER  
LINE,
```

ist_ipadd3 TYPE STANDARD TABLE OF ty_ipadd3 WITH HEADER LINE,
ist_curr_users TYPE STANDARD TABLE OF ty_curr_users WITH HEADER
LINE.

** ALV Tables/structure declaration.

DATA: ist_kub_fldcat TYPE slis_t_fieldcat_alv WITH HEADER LINE,
ist_layout TYPE slis_layout_alv,
ls_events TYPE slis_alv_event OCCURS 0,
it_events TYPE slis_alv_event,
gt_list_top_of_page TYPE slis_t_listheader.

DATA : g_no_lines TYPE sy-tabix,
g_repid LIKE sy-repid.

DATA : wa_cnt1 LIKE sy-tabix.

**-----*

** Constant Declaration

**-----*

CONSTANTS c_mark VALUE 'X'.

*

*

* Initialization Event - processing prior to selection screen *

INITIALIZATION.

* Select-Options and Parameters & Radio Button Declarations *

SELECTION-SCREEN BEGIN OF BLOCK kub_sel WITH FRAME TITLE text-001.

SELECT-OPTIONS: s_ipadd FOR zbc_kub_login-ipaddress,
s_logdt FOR zbc_kub_login-logon_date,
s_region FOR zbc_kub_locmst-region.

SELECTION-SCREEN END OF BLOCK kub_sel.

SELECTION-SCREEN BEGIN OF BLOCK kub_rep_sel WITH FRAME TITLE text-003.

PARAMETERS: p_sel1 RADIOBUTTON GROUP lim ,
p_sel2 RADIOBUTTON GROUP lim,
p_sel3 RADIOBUTTON GROUP lim,
p_sel4 RADIOBUTTON GROUP lim,
p_sel5 RADIOBUTTON GROUP lim.

SELECTION-SCREEN END OF BLOCK kub_rep_sel.

* Validation for Select-Options :Purchase Requisition Number *
* and Processing Status & Deletion Indicators *
* *

*AT SELECTION-SCREEN ON BLOCK kub_sel.

* PERFORM validate_input.

START-OF-SELECTION.

g_repid = sy-repid.

PERFORM generate_itab.

```

    PERFORM generate_reports.
*   PERFORM init_layout.
*   PERFORM generate_fcat.
*   PERFORM generate_report.

```

```

END-OF-SELECTION.

```

```

AT LINE-SELECTION.

```

```

    CASE sy-lsind.

```

```

        WHEN '1' .

```

```

            PERFORM detaillist1.

```

```

        WHEN '2' .

```

```

            PERFORM detaillist2.

```

```

        ENDCASE.

```

```

*&-----*
*&   Form generate_itab
*&-----*
*   text
*-----*
* --> p1   text
* <-- p2   text
*-----*

```

```

FORM generate_itab.

```

```

    CASE c_mark.

```

```

        WHEN p_sel1.

```

DATA : strval(10) TYPE c.

strval = '10.206%'.

```
SELECT a~bname a~terminal b~last_name b~first_name b~designation
INTO CORRESPONDING FIELDS OF TABLE ist_curr_users
FROM usr41 AS a INNER JOIN zusrmst AS b
ON a~bname = b~cpfno
WHERE a~terminal LIKE strval.
```

```
* SELECT bname terminal
* INTO CORRESPONDING FIELDS OF TABLE ist_curr_users
* FROM usr41 WHERE terminal LIKE strval.
*
```

WHEN p_sel2.

```
SELECT ipaddress SUM( time_interval ) AS time_interval
FROM zbc_kub_login INTO CORRESPONDING FIELDS OF TABLE
ist_ipadd
WHERE ipaddress IN s_ipadd AND logon_date IN s_logdt
GROUP BY IPADDRESS .
```

WHEN p_sel3.

```
SELECT ipaddress bname SUM( time_interval ) AS time_interval
FROM zbc_kub_login INTO CORRESPONDING FIELDS OF TABLE
ist_ipadd1
WHERE ipaddress IN s_ipadd AND logon_date IN s_logdt
GROUP BY IPADDRESS bname .
```

WHEN p_sel4.

```

SELECT b~location SUM( a~time_interval ) AS
time_interval
    INTO TABLE ist_ipadd2
    FROM zbc_kub_login AS a INNER JOIN zbc_kub_locmst AS b
    ON a~ipaddress = b~ipaddress
    WHERE a~ipaddress IN s_ipadd AND a~logon_date IN s_logdt
    GROUP BY B~LOCATION .

```

WHEN p_sel5.

```

SELECT b~region b~location SUM( a~time_interval ) AS
time_interval
    INTO TABLE ist_ipadd3
    FROM zbc_kub_login AS a INNER JOIN zbc_kub_locmst AS b
    ON a~ipaddress = b~ipaddress
    WHERE a~ipaddress IN s_ipadd AND a~logon_date IN s_logdt
    GROUP BY B~REGION b~location.

```

ENDCASE.

```

SELECT b~location b~ipaddress SUM( a~time_interval ) AS
time_interval
    INTO TABLE ist_ipadd2_det
    FROM zbc_kub_login AS a INNER JOIN zbc_kub_locmst AS b
    ON a~ipaddress = b~ipaddress
    WHERE a~ipaddress IN s_ipadd AND a~logon_date IN s_logdt
    GROUP BY B~LOCATION b~ipaddress .

```

```

SELECT * FROM zbc_kub_login
    INTO CORRESPONDING FIELDS OF TABLE ist_kublogin

```

```

        WHERE ipaddress IN s_ipadd AND logon_date IN s_logdt.
ENDFORM.          " generate_itab
*&-----*
*&   Form generate_fcat
*&-----*
*      text
*-----*
* --> p1      text
* <-- p2      text
*-----*
FORM generate_fcat.
  DATA : l_pos LIKE sy-tabix.
  CLEAR: l_pos.

```

```

** ist_fcatpr
  l_pos = l_pos + 1.
  ist_kub_fldcat-fieldname = 'IPADDRESS'.
  ist_kub_fldcat-ref_fieldname = 'IPADDRESS'.
  ist_kub_fldcat-ref_tabname  = 'ZBC_KUB_LOGIN'.
  APPEND ist_kub_fldcat.
  CLEAR ist_kub_fldcat.

  l_pos = l_pos + 1.
  ist_kub_fldcat-fieldname = 'LOGON_DATE'.
  ist_kub_fldcat-ref_fieldname = 'LOGON_DATE'.
  ist_kub_fldcat-ref_tabname  = 'ZBC_KUB_LOGIN'.
  ist_kub_fldcat-tabname     = 'IST_KUBLOGIN'.
  ist_kub_fldcat-col_pos    = l_pos.
  ist_kub_fldcat-seltext_m = 'Logon_Date'.
  APPEND ist_kub_fldcat.
  CLEAR ist_kub_fldcat.

```

```

l_pos = l_pos + 1.
ist_kub_fldcat-fieldname = 'LOGON_TIME'.
ist_kub_fldcat-ref_tabname = 'ZBC_KUB_LOGIN'.
ist_kub_fldcat-seltext_m = 'Logon_Time'.
ist_kub_fldcat-col_pos = l_pos.
APPEND ist_kub_fldcat.
CLEAR ist_kub_fldcat.

```

```

l_pos = l_pos + 1.
ist_kub_fldcat-fieldname = 'TIME'.
ist_kub_fldcat-ref_tabname = 'ZBC_KUB_LOGIN'.
ist_kub_fldcat-seltext_m = 'Time'.
ist_kub_fldcat-just = 'C'.
ist_kub_fldcat-col_pos = l_pos.
APPEND ist_kub_fldcat.
CLEAR ist_kub_fldcat.

```

```

l_pos = l_pos + 1.
ist_kub_fldcat-fieldname = 'TIME_INTERVAL'.
ist_kub_fldcat-ref_tabname = 'ZBC_KUB_LOGIN'.
ist_kub_fldcat-seltext_m = 'Duration'.
ist_kub_fldcat-col_pos = l_pos.
APPEND ist_kub_fldcat.
CLEAR ist_kub_fldcat.

```

```

ENDFORM.          " generate_fcat
*&-----*
*&   Form generate_report
*&-----*

```

```

*      text
*-----*
* --> p1      text
* <-- p2      text
*-----*

```

FORM generate_report.

IF NOT ist_kublogin IS INITIAL.

CALL FUNCTION 'REUSE_ALV_LIST_DISPLAY'

EXPORTING

i_callback_program = g_repid

is_layout = ist_layout

it_fldcat = ist_kub_fldcat[]

* i_default = 'X'

* i_save = 'A'

* it_events = ls_events

TABLES

t_outtab = ist_kublogin

EXCEPTIONS

program_error = 1

OTHERS = 2.

IF sy-subrc <> 0.

MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno

WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.

ENDIF.

ELSE .

MESSAGE i042(zbc).

ENDIF.

```

ENDFORM.                " generate_report

*&-----*
*&   Form init_layout
*&-----*
*   text
*-----*
* --> p1      text
* <-- p2      text
*-----*

FORM init_layout.
    ist_layout-zebra      = 'X'.
    ist_layout-no_vline   = 'X'.
ENDFORM.                " init_layout

*&-----*
*&   Form generate_reports
*&-----*
*   text
*-----*
* --> p1      text
* <-- p2      text
*-----*

FORM generate_reports.

DATA:l_cnt TYPE sy-tabix.
DATA :c_cnt(4) TYPE c.

CASE c_mark.

    WHEN p_sel1.

```


NEW-LINE NO-SCROLLING.

SKIP.

WRITE:/37 ' KU Band Current Users Detail Report' COLOR 7.

SKIP 1.

NEW-LINE NO-SCROLLING.

WRITE:/35 '(Status as on ',sy-uzeit,' Hrs of',sy-datum,')'.

WRITE:/(230) sy-uline.

FORMAT COLOR 7.

WRITE :/1 'SrNo', 7 'UserID'.

WRITE : 21 'Last Name',

36 'First Name', 50 'Designation'.

WRITE:/(230) sy-uline.

LOOP AT ist_curr_users .

FORMAT COLOR 2.

l_cnt = l_cnt + 1 .

c_cnt = l_cnt.

WRITE:/1 c_cnt, 7 ist_curr_users-bname,

21 ist_curr_users-last_name,

36 ist_curr_users-first_name,

50 ist_curr_users-designation.

ENDLOOP.

WHEN p_sel2.

NEW-LINE NO-SCROLLING.

SKIP.

WRITE:/:37 ' KU Band IP Address Wise Detail Report' COLOR 7.

SKIP 1.

NEW-LINE NO-SCROLLING.

WRITE:/:35 '(Status as on ',sy-uzeit,' Hrs of',sy-datum,')'.

WRITE:/(230) sy-uline.

FORMAT COLOR 7.

WRITE :/1 'SrNo', 7 'IPAddress'.

WRITE : 22 'Time Interval'.

WRITE:/(230) sy-uline.

LOOP AT ist_ipadd.

FORMAT COLOR 2.

l_cnt = l_cnt + 1.

c_cnt = l_cnt.

WRITE:/ c_cnt,7 ist_ipadd-ipaddress,

23 ist_ipadd-time_interval.

ENDLOOP.

WHEN p_sel3.

NEW-LINE NO-SCROLLING.

SKIP.

WRITE:/:37 'KU Band IP Address/User Wise Detail Report ' COLOR 7.

SKIP 1.

NEW-LINE NO-SCROLLING.

WRITE:/:35 '(Status as on ',sy-uzeit,' Hrs of',sy-datum,')'.

WRITE:/(230) sy-uline.

FORMAT COLOR 7.

```
WRITE :/1 'SrNo', 7 'IPAddress'.  
WRITE : 22 'User Name',36 'Time Interval'.  
WRITE:/(230) sy-uline.
```

```
LOOP AT ist_ipadd1.  
  FORMAT COLOR 2.  
  l_cnt = l_cnt + 1.  
  c_cnt = l_cnt.  
  WRITE:/ c_cnt,7 ist_ipadd1-ipaddress,  
    23 ist_ipadd1-bname ,36 ist_ipadd1-time_interval.  
ENDLOOP.
```

```
WHEN p_sel4.
```

```
NEW-LINE NO-SCROLLING.  
SKIP.  
WRITE:/37 'KU Band Utilization Company Code Wise Summary' COLOR 7.  
SKIP 1.  
NEW-LINE NO-SCROLLING.  
WRITE:/35 '( Status as on ',sy-uzeit,' Hrs of',sy-datum,')'.  
WRITE:/(230) sy-uline.  
FORMAT COLOR 7.
```

```
WRITE :/1 'SrNo', 7 'Company Code'.  
WRITE : 22 'Time Interval'.  
WRITE:/(230) sy-uline.
```

```
LOOP AT ist_ipadd2.
```

```

FORMAT COLOR 2.
l_cnt = l_cnt + 1.
c_cnt = l_cnt.
WRITE:/ c_cnt,12 ist_ipadd2-location HOTSPOT ON ,
23 ist_ipadd2-time_interval HOTSPOT ON .
HIDE: ist_ipadd2-location.
ENDLOOP.

```

WHEN p_sel5.

```

NEW-LINE NO-SCROLLING.
SKIP.
WRITE:/37 'KU Band Utilization Region Wise Summary' COLOR 7.
SKIP 1.
NEW-LINE NO-SCROLLING.
WRITE:/35 '( Status as on ',sy-zeit,' Hrs of',sy-datum,')'.
WRITE:/(230) sy-uline.
FORMAT COLOR 7.

```

```

WRITE :/1 'Region'.
WRITE :18 'Location' , 36 'Time Interval'.
WRITE:/(230) sy-uline.

```

```

LOOP AT ist_ipadd3.
FORMAT COLOR 2.
.
WRITE:/1 ist_ipadd3-region HOTSPOT ON ,
20 ist_ipadd3-location HOTSPOT ON,
36 ist_ipadd3-time_interval HOTSPOT ON.
HIDE: ist_ipadd3-location.

```

```

AT END OF region.
SUM.
FORMAT COLOR 5.
WRITE:/(95) sy-uline.
SKIP.
WRITE:/1 'Total Link Utilization at',
        ist_ipadd3-region,
        35 ist_ipadd3-time_interval.
SKIP.
ENDAT.
ENDLOOP.

```

```

ENDCASE.

```

```

ENDFORM.          " generate_reports
*&-----*
*&   Form detaillist1
*&-----*
*   text
*-----*
* --> p1      text
* <-- p2      text
*-----*

```

```

FORM detaillist1.
DATA:l_cnt TYPE sy-tabix.
DATA :c_cnt(4) TYPE c.

```

```

IF NOT ist_ipadd2 IS INITIAL.
  IF NOT p_sel4 IS INITIAL.

```

SKIP.

WRITE:/30 'Company Code Wise KU Band Usage Details' COLOR 7.

SKIP 1.

NEW-LINE NO-SCROLLING.

WRITE:/25 '(Status as on ',sy-uzeit,' Hrs of',sy-datum,')'.

WRITE:/(95) sy-uline.

FORMAT COLOR 7.

WRITE :/1 'Comp.Cd', 14 'IPAddress' , 30 'Time Interval'.

WRITE:/(95) sy-uline.

FORMAT COLOR 2.

loop at ist_ipadd2_det where location = ist_ipadd2-location.

write:/1 ist_ipadd2_det-location, 14 ist_ipadd2_det-ipaddress

hotspot on,

30 ist_ipadd2_det-time_interval hotspot on.

hide:

ist_ipadd2_det-ipaddress.

endloop.

ENDIF.

ENDIF.

IF NOT ist_ipadd3 IS INITIAL.

IF NOT p_sel5 IS INITIAL.

SKIP.

WRITE:/30 'Company Code Wise KU Band Usage Details' COLOR 7.

SKIP 1.

NEW-LINE NO-SCROLLING.

WRITE:/25 '(Status as on ',sy-uzeit,' Hrs of',sy-datum,')'.

WRITE:/(95) sy-uline.

FORMAT COLOR 7.

WRITE :/1 'Comp Cd', 14 'IPAddress' , 30 'Time Interval'.

WRITE:/(95) sy-uline.

FORMAT COLOR 2.

loop at ist_ipadd2_det where location = ist_ipadd3-location.

write:/1 ist_ipadd2_det-location, 14 ist_ipadd2_det-ipaddress

hotspot on,

30 ist_ipadd2_det-time_interval hotspot on.

hide: ist_ipadd2_det-ipaddress.

endloop.

ENDIF.

ENDIF.

ENDFORM. " detaillist

```
*&-----*
*&   Form detaillist2
*&-----*
*   text
*-----*
* --> p1   text
* <-- p2   text
```

FORM detaillist2.

DATA:l_cnt TYPE sy-tabix.

DATA :c_cnt(4) TYPE c.

NEW-LINE NO-SCROLLING.

SKIP.

WRITE:/37 'KU Band IP Address/User Wise Detail Report ' COLOR 7.

SKIP 1.

NEW-LINE NO-SCROLLING.

WRITE:/35 '(Status as on ',sy-uzeit,' Hrs of',sy-datum,')'.

WRITE:/(230) sy-uline.

FORMAT COLOR 7.

WRITE :/1 'SrNo', 7 'IPAddress'.

WRITE : 22 'User Name',36 'Time Interval'.

WRITE:/(230) sy-uline.

LOOP AT ist_kublogin where ipaddress = ist_ipadd2_det-ipaddress.

FORMAT COLOR 2.

l_cnt = l_cnt + 1.

c_cnt = l_cnt.

WRITE:/ c_cnt,7 ist_kublogin-ipaddress,

23 ist_kublogin-bname ,36 ist_kublogin-time_interval.

ENDLOOP.

ENDFORM. " detaillist2

Program 2:- Zbc_kubscreen

REPORT zbc_kubscreen .

DATA : ok_code LIKE sy-ucomm.

TABLES: zbc_kub_locmst.

START-OF-SELECTION.

CALL SCREEN 100.

END-OF-SELECTION.

&-----

*& Module STATUS_0100 OUTPUT

&-----

* text

MODULE status_0100 OUTPUT.

SET PF-STATUS 'SCR100'.

SET TITLEBAR 'Ku Band Master Data '.

ENDMODULE. " STATUS_0100 OUTPUT

&-----

*& Module USER_COMMAND_0100 INPUT

&-----

* text

MODULE user_command_0100 INPUT.

CASE ok_code.

WHEN 'ADD' .

INSERT INTO zbc_kub_locmst VALUES zbc_kub_locmst.

if sy-subrc <> 0.

MESSAGE i030(zbc) WITH zbc_kub_locmst-ipaddress.

ELSE.

MESSAGE i051(zbc).

ENDIF.

WHEN 'DELETE' .

DELETE FROM zbc_kub_locmst

WHERE ipaddress = zbc_kub_locmst-ipaddress.

IF sy-subrc = 0 .

MESSAGE i050(zbc).

ELSE.

MESSAGE i076(zbc).

ENDIF.

WHEN 'CHANGE' .

UPDATE zbc_kub_locmst FROM zbc_kub_locmst.

WHEN 'DISPLAY' .

SELECT SINGLE * FROM zbc_kub_locmst WHERE

ipaddress = zbc_kub_locmst-ipaddress.

```

IF sy-subrc <> 0.
    MESSAGE i049(zbc) .
ENDIF.

WHEN 'REFRESH' .

    CLEAR zbc_kub_locmst.

WHEN 'BACK' .
    LEAVE TO SCREEN 0.

WHEN 'EXIT' OR 'CANCEL' .
    LEAVE TO SCREEN 0.

ENDCASE.
ENDMODULE.          " USER_COMMAND_0100 INPUT

```

Program 3:- Zbc_loginstat_kub

```

REPORT zbc_loginstat_kub .
TABLES : zbc_kub_login.
TYPES:
    BEGIN OF ty_kub_login,
        bname TYPE usr41-bname,
        ipaddress TYPE zbc_kub_login-ipaddress,
        terminalname TYPE zbc_kub_login-terminalname,
        logon_date TYPE usr41-logon_date,
        logon_time TYPE usr41-logon_time,
        terminal TYPE usr41-terminal,
        time TYPE zbc_kub_login-time,
        time_interval TYPE zbc_kub_login-time_interval,

```

END OF ty_kub_login.

DATA : wa_kub_login TYPE ty_kub_login,
ist_kub_login TYPE TABLE OF ty_kub_login,
wa_zbc_kub_login TYPE zbc_kub_login.

DATA: strval LIKE zbc_kub_login-ipaddress,
time_int LIKE sy-uzeit,
len TYPE i.

DATA: l_hrs TYPE i,
l_mnt TYPE i,
l_sec TYPE i.

PARAMETERS:

p_ipadd TYPE zbc_kub_login-ipaddress . "obligatory.

START-OF-SELECTION.

IF p_ipadd IS INITIAL.

p_ipadd = '*' .

ENDIF.

strval = p_ipadd .

len = strlen(strval).

DO len TIMES.

REPLACE '*' WITH '%' INTO strval.

ENDDO.

```
SELECT bname terminal logon_date logon_time FROM usr41 INTO  
CORRESPONDING FIELDS OF TABLE ist_kub_login  
WHERE terminal LIKE strval.
```

```
LOOP AT ist_kub_login INTO wa_kub_login.  
    wa_kub_login-time = sy-uzeit.
```

```
SPLIT wa_kub_login-terminal AT '-' INTO  
wa_kub_login-ipaddress wa_kub_login-terminalname.
```

```
MOVE-CORRESPONDING wa_kub_login TO wa_zbc_kub_login.  
MODIFY zbc_kub_login FROM wa_zbc_kub_login.  
MODIFY ist_kub_login FROM wa_kub_login TRANSPORTING time ipaddress  
terminalname.
```

```
ENDLOOP.
```

```
break-point.
```

```
LOOP AT ist_kub_login INTO wa_kub_login.  
    IF wa_kub_login-time < wa_kub_login-logon_time .  
        time_int = 86400 - ( wa_kub_login-logon_time - wa_kub_login-time ) .  
    ELSE.  
        time_int = wa_kub_login-time - wa_kub_login-logon_time .  
    ENDIF.
```

```
l_hrs  = time_int+0(2).  
l_mnt  = time_int+2(2).  
l_sec  = time_int+4(2).
```

```
wa_kub_login-time_interval = ( l_hrs * 3600 ) + ( l_mnt * 60 )  
                             + l_sec .
```

```
MOVE-CORRESPONDING wa_kub_login TO wa_zbc_kub_login.  
MODIFY zbc_kub_login FROM wa_zbc_kub_login.
```

```
ENDLOOP.
```

```
END-OF-SELECTION.
```

Program 4:- Zbc_kub_upload

```
REPORT ZBC_KUB_UPLOAD .
```

```
tables : zbc_kub_locmst.
```

```
DATA:FLENGTH TYPE I.
```

```
DATA:it_kub type standard table of zbc_kub_locmst with header line.
```

```
data: wa_kub like line of it_kub.
```

```
CALL FUNCTION 'WS_UPLOAD'
```

```
EXPORTING
```

```
    CODEPAGE      = 'IBM'
```

```
    FILENAME      = 'C:\test.txt'
```

```
    FILETYPE      = 'DAT'
```

```
IMPORTING
```

```
    FILELENGTH    = FLENGTH
```

```
TABLES
```

```
    DATA_TAB     = it_kub
```

EXCEPTIONS

CONVERSION_ERROR = 1

FILE_OPEN_ERROR = 2

FILE_READ_ERROR = 3

INVALID_TABLE_WIDTH = 4

INVALID_TYPE = 5.

if sy-subrc = 0 .

loop at it_kub into wa_kub.

insert zbc_kub_locmst from wa_kub.

endloop.

WRITE: 'SY-SUBRC:', SY-SUBRC,

 / 'Length :', FLENGTH.

endif.

Program 5:- Zbc_bdc_kub

report ZBC_BDC_KUB

 no standard page heading line-size 255.

DATA: g_string TYPE string. " For GUI_UPLOAD Function

SELECTION-SCREEN BEGIN OF BLOCK b1 WITH FRAME TITLE text-001.

PARAMETERS: p_file LIKE rlgrap-filename OBLIGATORY.

SELECTION-SCREEN END OF BLOCK b1.

include bdcrecx1.

INITIALIZATION.

group = sy-uname.

parameters: dataset(132) lower case.

*** DO NOT CHANGE - the generated data section - DO NOT CHANGE ***

*

* If it is necessary to change the data section use the rules:

* 1.) Each definition of a field exists of two lines

* 2.) The first line shows exactly the comment

* '* data element: ' followed with the data element

* which describes the field.

* If you don't have a data element use the

* comment without a data element name

* 3.) The second line shows the fieldname of the

* structure, the fieldname must consist of

* a fieldname and optional the character '_' and

* three numbers and the field length in brackets

* 4.) Each field must be type C.

*

*** Generated data section with specific formatting - DO NOT CHANGE ***

types: begin of record,

* data element: IPADDRESS

IPADDRESS_001(015),

* data element: DESC30

REMOTE_LOC_002(031),

* data element: BUKRS

LOCATION_003(004),

* data element: ZREGIONK

REGION_004(003),

end of record.

*** End generated data section ***

DATA : ist_record TYPE STANDARD TABLE OF record.

DATA : wa_record type record. " work area for record

AT SELECTION-SCREEN ON VALUE-REQUEST FOR p_file.

PERFORM get_filename USING p_file.

PERFORM upload_file USING p_file.

start-of-selection.

perform open_group.

LOOP AT ist_record INTO wa_record.

perform bdc_dynpro using 'ZBC_KUBSCREEN' '0100'.

perform bdc_field using 'BDC_CURSOR'
'ZBC_KUB_LOCMST-IPADDRESS'.

perform bdc_field using 'BDC_OKCODE'
'=DELETE'.

perform bdc_field using 'ZBC_KUB_LOCMST-IPADDRESS'
wa_record-IPADDRESS_001.

perform bdc_field using 'ZBC_KUB_LOCMST-REMOTE_LOC'
wa_record-REMOTE_LOC_002.

```

perform bdc_field      using 'ZBC_KUB_LOCMST-LOCATION'
                        wa_record-LOCATION_003.
perform bdc_field      using 'ZBC_KUB_LOCMST-REGION'
                        wa_record-REGION_004.
perform bdc_transaction using 'ZBC_KUB_MST'.
endloop.
perform close_group.

```

```

*&-----*
*&   Form get_filename
*&-----*
*   text
*-----*
*   -->P_FILE text
*-----*

```

```

FORM get_filename USING  P_FILE.

```

```

CALL FUNCTION 'WS_FILENAME_GET'

```

```

    EXPORTING

```

```

        def_path      = 'C:\'

```

```

        mask          = ',*.txt.'

```

```

    IMPORTING

```

```

        filename      = p_file

```

```

    EXCEPTIONS

```

```

        inv_winsys     = 1

```

```

        no_batch       = 2

```

```

        selection_cancel = 3

```

```

        selection_error = 4

```

```

        OTHERS         = 5.

```

```

IF sy-subrc <> 0.

```

```

    PERFORM callerr USING text-003.

```

ENDIF.

```
ENDFORM.                " get_filename

*&-----*
*&   Form callerr
*&-----*
*   text
*-----*
*   -->P_TEXT_003 text
*-----*

FORM callerr USING   l_text.
  CALL FUNCTION 'FC_POPUP_ERR_WARN_MESSAGE'
    EXPORTING
      popup_title = text-001
      is_error    = 'X'
      message_text = l_text
      start_column = 30
      start_row   = 8.

  STOP.

ENDFORM.                " callerr

*&-----*
*&   Form upload_file
*&-----*
*   text
*-----*
*   -->P_P_FILE text
*-----*

FORM upload_file USING l_filename.
  g_string = l_filename.
```

CALL FUNCTION 'GUI_UPLOAD'

EXPORTING

filename = g_string

filetype = 'ASC'

has_field_separator = 'X'

TABLES

data_tab = ist_record

EXCEPTIONS

file_open_error = 1

file_read_error = 2

no_batch = 3

gui_refuse_filetransfer = 4

invalid_type = 5

no_authority = 6

unknown_error = 7

bad_data_format = 8

header_not_allowed = 9

separator_not_allowed = 10

header_too_long = 11

unknown_dp_error = 12

access_denied = 13

dp_out_of_memory = 14

disk_full = 15

dp_timeout = 16

OTHERS = 17.

IF sy-subrc <> 0.

PERFORM callerr USING text-003.

ENDIF.

ENDFORM. " upload_file