Review of Data Warehouse: Need for data warehouse, Big data, Data Pre-Processing, Three tier architecture; MDDM and its schemas, Introduction to Spatial Data warehouse, Architecture of Spatial Systems, Spatial: Objects, data types, reference systems; Topological Relationships, Conceptual Models for Spatial Data, Implementation Models for Spatial Data, Spatial Levels, Hierarchies and Measures Spatial Fact Relationships.

Introduction to Data warehouse

A data warehouse is a collection of data marts representing historical data from different operations in the company. This data is stored in a structure optimized for querying and data analysis as a data warehouse. Table design, dimensions and organization should be consistent throughout a data warehouse so that reports or queries across the data warehouse are consistent. A data warehouse can also be viewed as a database for historical data from different functions within a company.

The term Data Warehouse was coined by Bill Inmon in 1990, which he defined in the following way: "A warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision making process". He defined the terms in the sentence as follows:

Subject Oriented: Data that gives information about a particular subject instead of about a company's ongoing operations.

Integrated: Data that is gathered into the data warehouse from a variety of sources and merged into a coherent whole.

Time-variant*:* All data in the data warehouse is identified with a particular time period.

Non-volatile*:* Data is stable in a data warehouse. More data is added but data is never removed. This enables management to gain a consistent picture of the business. It is a single, complete and consistent store of data obtained from a variety of different sources made available to end users in what they can understand and use in a business context. It can be

Used for decision Support

Used to manage and control business

Used by managers and end-users to understand the business and make judgments

Data Warehousing is an architectural construct of information systems that provides users with current and historical decision support information that is hard to access or present in traditional operational data stores

Other important terminology

Enterprise Data warehouse*:* It collects all information about subjects (*customers, products, sales, assets, personnel*) that span the entire organization

Data Mart: Departmental subsets that focus on selected subjects. A data mart is a segment of a data warehouse that can provide data for reporting and analysis on a section, unit, department or operation in the company, e.g. sales, payroll, production. Data marts are sometimes complete individual data warehouses which are usually smaller than the corporate data warehouse.

Decision Support System (DSS): Information technology to help the knowledge worker (executive, manager, and analyst) makes faster & better decisions

Drill-down: Traversing the summarization levels from highly summarized data to the underlying current or old detail

Metadata: Data about data. Containing location and description of warehouse system components: names, definition, structure…

Benefits of data warehousing

Data warehouses are designed to perform well with aggregate queries running on large amounts of data.

The structure of data warehouses is easier for end users to navigate, understand and query against unlike the relational databases primarily designed to handle lots of transactions. Data warehouses enable queries that cut across different segments of a company's operation. E.g. production data could be compared against inventory data even if they were originally stored in different databases with different structures.

Queries that would be complex in very normalized databases could be easier to build and maintain in data warehouses, decreasing the workload on transaction systems.
Data warehousing is an efficient way to manage and report on data that is from a variety of sources, non uniform and scattered throughout a company.
Data warehousing is an efficient way to manage demand for lots of information from lots of users.
Data warehousing provides the capability to analyze large amounts of historical data for nuggets of wisdom that can provide an organization with competitive advantage.

Operational and informational Data

• Operational Data:

Focusing on transactional function such as bank card withdrawals and deposits
Detailed
Updateable
Reflects current data

• Informational Data:

Focusing on providing answers to problems posed by decision makers
Summarized
Non updateable

DATA WAREHOUSE ARCHITECTURE AND ITS SEVEN COMPONENTS

1. Data sourcing, cleanup, transformation, and migration tools
2. Metadata repository
3. Warehouse/database technology
4. Data marts
5. Data query, reporting, analysis, and mining tools
6. Data warehouse administration and management
7. Information delivery system

1 Data warehouse database

This is the central part of the data warehousing environment. This is the item number 2 in the above arch. diagram. This is implemented based on RDBMS technology.

2 Sourcing, Acquisition, Clean up, and Transformation Tools

This is item number 1 in the above arch diagram. They perform conversions, summarization, key changes, structural changes and condensation. The data transformation is required so that the information can by used by decision support tools. The transformation produces programs, control statements, JCL code, COBOL code, UNIX scripts, and SQL DDL code etc., to move the data into data warehouse from multiple operational systems.

The functionalities of these tools are listed below:

To remove unwanted data from operational db
Converting to common data names and attributes
Calculating summaries and derived data
Establishing defaults for missing data
Accommodating source data definition changes

3 Meta data

It is data about data. It is used for maintaining, managing and using the data warehouse. It is

classified into two:
***Technical Meta data*:** It contains information about data warehouse data used by warehouse designer, administrator to carry out development and management tasks. It includes,

Info about data stores
Transformation descriptions. That is mapping methods from operational db to warehouse db
Warehouse Object and data structure definitions for target data
The rules used to perform clean up, and data enhancement
Data mapping operations
Access authorization, backup history, archive history, info delivery history, data acquisition history, data access etc.,
Business Meta data: It contains info that gives info stored in data warehouse to users. It includes,
Subject areas, and info object type including queries, reports, images, video, audio clips etc.
Internet home pages
Info related to info delivery system
Data warehouse operational info such as ownerships, audit trails etc.,

4 Access tools

Its purpose is to provide info to business users for decision making. There are five categories:
Data query and reporting tools
Application development tools
Executive info system tools (EIS)
OLAP tools
Data mining tools

5 Data marts

Departmental subsets that focus on selected subjects. They are independent used by dedicated user group. They are used for rapid delivery of enhanced decision support functionality to end users. **Data mart is used in the following situation:**
Extremely urgent user requirement
The absence of a budget for a full scale data warehouse strategy
The decentralization of business needs
The attraction of easy to use tools and mind sized project

Data mart presents two problems:

1. Scalability: A small data mart can grow quickly in multi dimensions. So that while designing it, the organization has to pay more attention on system scalability, consistency and manageability issues
2. Data integration

6 Data warehouse admin and management

The management of data warehouse includes,
Security and priority management
Monitoring updates from multiple sources
Data quality checks
Managing and updating meta data
Auditing and reporting data warehouse usage and status
Purging data
Replicating, sub setting and distributing data
Backup and recovery

Data warehouse storage management which includes capacity planning, hierarchical storage management and purging of aged data etc.,

7 Information delivery system

• It is used to enable the process of subscribing for data warehouse info.
• Delivery to one or more destinations according to specified scheduling algorithm

Building a Data warehouse:

There are two reasons why organizations consider data warehousing a critical need. In other words, there are two factors that drive you to build and use data warehouse. They are:

Business factors:

Business users want to make decision quickly and correctly using all available data.

Technological factors:

To address the incompatibility of operational data stores
IT infrastructure is changing rapidly. Its capacity is increasing and cost is decreasing so that building a data warehouse is easy
There are several things to be considered while building a successful data warehouse

Business considerations:

Organizations interested in development of a data warehouse can choose one of the following two approaches:
Top - Down Approach (Suggested by Bill Inmon)
Bottom - Up Approach (Suggested by Ralph Kimball)

Top - Down Approach

In the top down approach suggested by Bill Inmon, we build a centralized repository to house corporate wide business data. This repository is called Enterprise Data Warehouse (EDW). The data in the EDW is stored in a normalized form in order to avoid redundancy.
The central repository for corporate wide data helps us maintain one version of truth of the data. The data in the EDW is stored at the most detail level. The reason to build the EDW on the most detail level is to leverage
1. Flexibility to be used by multiple departments.
2. Flexibility to cater for future requirements.

The disadvantages of storing data at the detail level are

1. The complexity of design increases with increasing level of detail.
2. It takes large amount of space to store data at detail level, hence increased cost.

Once the EDW is implemented we start building subject area specific data marts which contain data in a de normalized form also called star schema. The data in the marts are usually summarized based on the end users analytical requirements.
The reason to de normalize the data in the mart is to provide faster access to the data for the end users analytics. If we were to have queried a normalized schema for the same analytics, we would end up in a complex multiple level joins that would be much slower as compared to the one on the de normalized schema.

We should implement the top-down approach when

1. The business has complete clarity on all or multiple subject areas data warehosue requirements.
2. The business is ready to invest considerable time and money.

The advantage of using the Top Down approach is that we build a centralized repository to cater for one version of truth for business data.

The disadvantage of using the Top Down approach is that it requires more time and initial investment. The business has to wait for the EDW to be implemented followed by building the

data marts before which they can access their reports.

Bottom Up Approach

The bottom up approach suggested by Ralph Kimball is an incremental approach to build a data warehouse. Here we build the data marts separately at different points of time as and when the specific subject area requirements are clear. The data marts are integrated or combined together to form a data warehouse. Separate data marts are combined through the use of conformed dimensions and conformed facts. A conformed dimension and a conformed fact is one that can be shared across data marts.

A Conformed dimension has consistent dimension keys, consistent attribute names and consistent values across separate data marts. The conformed dimension means exact same thing with every fact table it is joined.

A Conformed fact has the same definition of measures, same dimensions joined to it and at the same granularity across data marts.

The bottom up approach helps us incrementally build the warehouse by developing and integrating data marts as and when the requirements are clear. We don't have to wait for knowing the overall requirements of the warehouse. We should implement the bottom up approach when

1. We have initial cost and time constraints.
2. The complete warehouse requirements are not clear. We have clarity to only one data mart.

The advantage of using the Bottom Up approach is that they do not require high initial costs and have a faster implementation time; hence the business can start using the marts much earlier as compared to the top-down approach.

The disadvantages of using the Bottom Up approach is that it stores data in the de normalized format, hence there would be high space usage for detailed data. We have a tendency of not keeping detailed data in this approach hence losing out on advantage of having detail data .i.e. flexibility to easily cater to future requirements.

Bottom up approach is more realistic but the complexity of the integration may become a serious obstacle.

Design considerations

To be a successful data warehouse designer must adopt a holistic approach that is considering all data warehouse components as parts of a single complex system, and take into account all possible data sources and all known usage requirements.

Most successful data warehouses that meet these requirements have these common characteristics:

Are based on a dimensional model

Contain historical and current data

Include both detailed and summarized data

Consolidate disparate data from multiple sources while retaining consistency

Data warehouse is difficult to build due to the following reason:

Heterogeneity of data sources

Use of historical data

Growing nature of data base

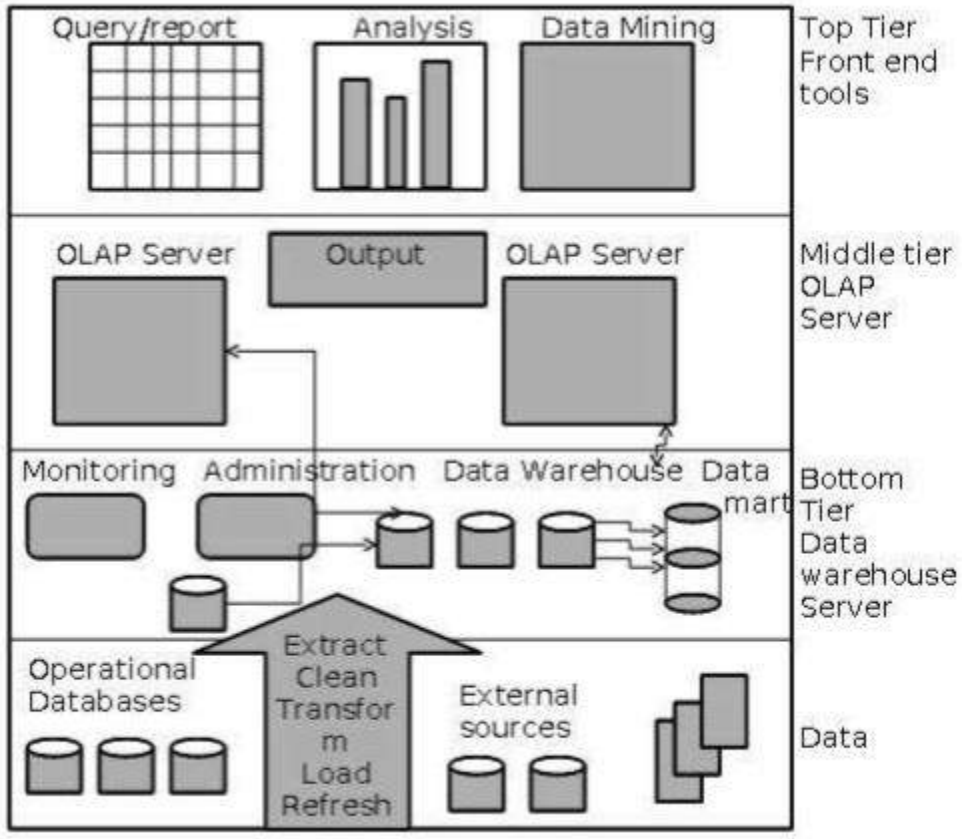
Data warehouse design approach muse be business driven, continuous and iterative engineering approach. In addition to the general considerations there are following specific points relevant to the data warehouse design:

Three-Tier Data Warehouse Architecture

Generally a data warehouses adopts a three-tier architecture. Following are the three tiers of the data warehouse architecture.

- **Bottom Tier** − The bottom tier of the architecture is the data warehouse database server. It is the relational database system. We use the back end tools and utilities to feed data into the bottom tier. These back end tools and utilities perform the Extract, Clean, Load, and refresh functions.
- **Middle Tier** − In the middle tier, we have the OLAP Server that can be implemented in either of the following ways.
    - By Relational OLAP (ROLAP), which is an extended relational database management system. The ROLAP maps the operations on multidimensional data to standard relational operations.
    - By Multidimensional OLAP (MOLAP) model, which directly implements the multidimensional data and operations.
- **Top-Tier** − This tier is the front-end client layer. This layer holds the query tools and reporting tools, analysis tools and data mining tools.

The following diagram depicts the three-tier architecture of data warehouse −



Data Warehouse Models

From the perspective of data warehouse architecture, we have the following data warehouse models −

- Virtual Warehouse
- Data mart
- Enterprise Warehouse

Virtual Warehouse

The view over an operational data warehouse is known as a virtual warehouse. It is easy to build a virtual warehouse. Building a virtual warehouse requires excess capacity on operational database servers.

Data Mart

Data mart contains a subset of organization-wide data. This subset of data is valuable to specific groups of an organization.

In other words, we can claim that data marts contain data specific to a particular group. For example, the marketing data mart may contain data related to items, customers, and sales. Data marts are confined to subjects.

Points to remember about data marts −

- Window-based or Unix/Linux-based servers are used to implement data marts. They are implemented on low-cost servers.
- The implementation data mart cycles is measured in short periods of time, i.e., in weeks rather than months or years.
- The life cycle of a data mart may be complex in long run, if its planning and design are not organization-wide.
- Data marts are small in size.
- Data marts are customized by department.
- The source of a data mart is departmentally structured data warehouse.
- Data mart are flexible.

Enterprise Warehouse

- An enterprise warehouse collects all the information and the subjects spanning an entire organization
- It provides us enterprise-wide data integration.
- The data is integrated from operational systems and external information providers.
- This information can vary from a few gigabytes to hundreds of gigabytes, terabytes or beyond.

Load Manager

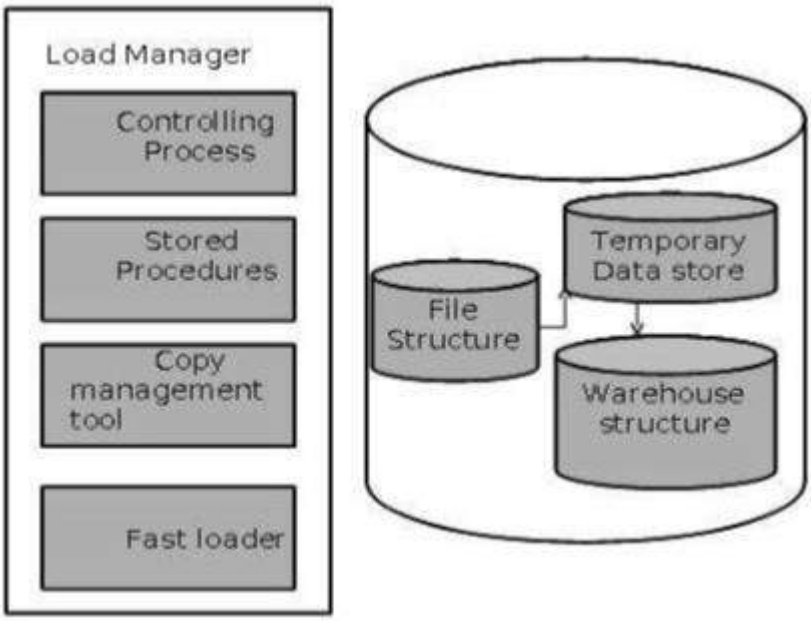
This component performs the operations required to extract and load process.

The size and complexity of the load manager varies between specific solutions from one data warehouse to other.

Load Manager Architecture

The load manager performs the following functions −

- Extract the data from source system.
- Fast Load the extracted data into temporary data store.
- Perform simple transformations into structure similar to the one in the data warehouse.



Extract Data from Source

The data is extracted from the operational databases or the external information providers. Gateways is the application programs that are used to extract data. It is supported by underlying DBMS and allows client program to generate SQL to be executed at a server. Open Database Connection(ODBC), Java Database Connection (JDBC), are examples of gateway.

Fast Load

- In order to minimize the total load window the data need to be loaded into the warehouse in the fastest possible time.
- The transformations affects the speed of data processing.

- It is more effective to load the data into relational database prior to applying transformations and checks.
- Gateway technology proves to be not suitable, since they tend not be performant when large data volumes are involved.

Simple Transformations

While loading it may be required to perform simple transformations. After this has been completed we are in position to do the complex checks. Suppose we are loading the EPOS sales transaction we need to perform the following checks:

- Strip out all the columns that are not required within the warehouse.
- Convert all the values to required data types.

Warehouse Manager

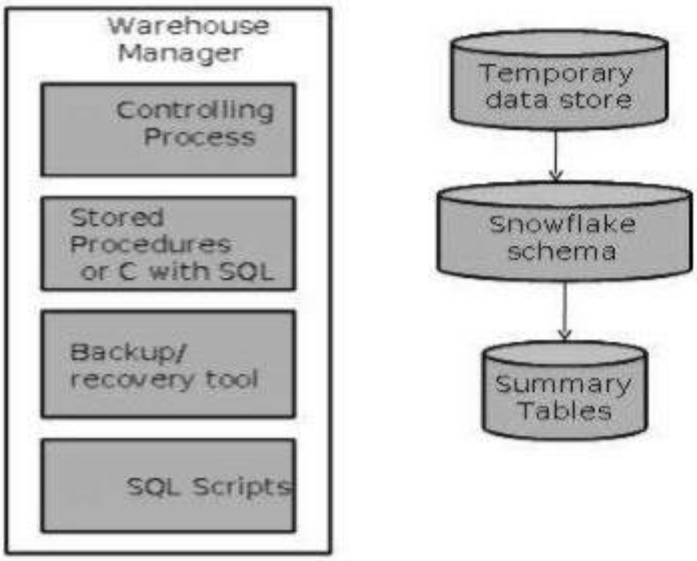
A warehouse manager is responsible for the warehouse management process. It consists of third-party system software, C programs, and shell scripts.

The size and complexity of warehouse managers varies between specific solutions.

Warehouse Manager Architecture

A warehouse manager includes the following −

- The controlling process
- Stored procedures or C with SQL
- Backup/Recovery tool
- SQL Scripts



Operations Performed by Warehouse Manager

- A warehouse manager analyzes the data to perform consistency and referential integrity checks.
- Creates indexes, business views, partition views against the base data.
- Generates new aggregations and updates existing aggregations. Generates normalizations.
- Transforms and merges the source data into the published data warehouse.
- Backup the data in the data warehouse.
- Archives the data that has reached the end of its captured life.

Note − A warehouse Manager also analyzes query profiles to determine index and aggregations are appropriate.

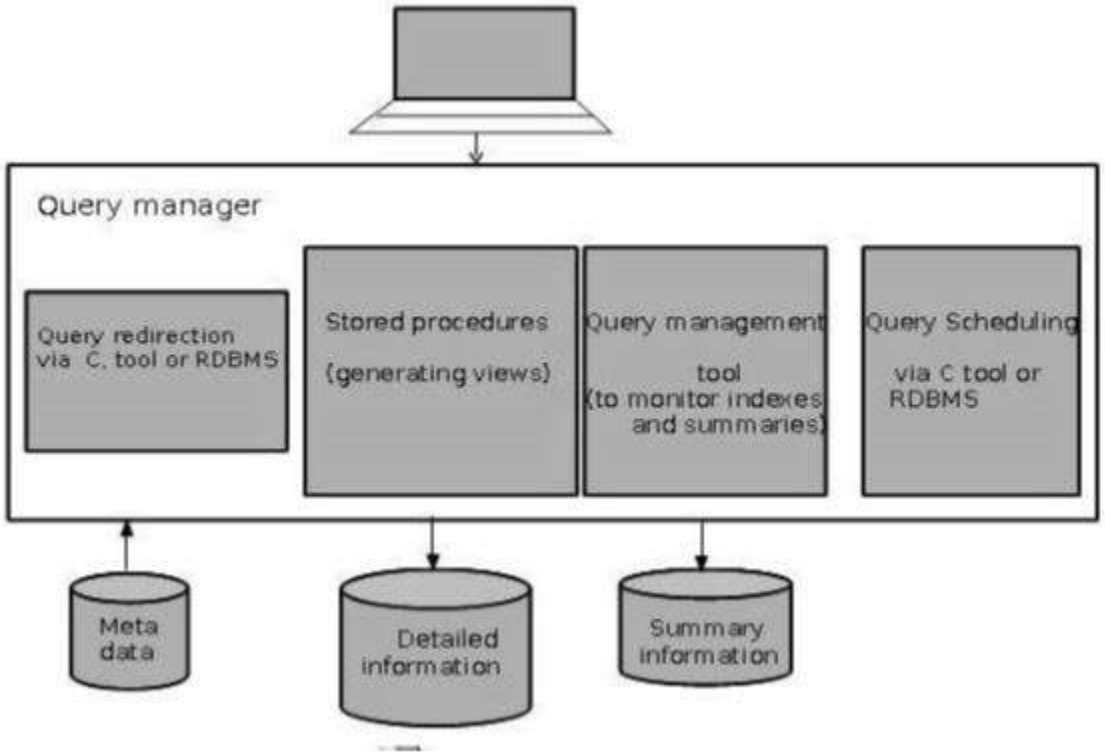
Query Manager

- Query manager is responsible for directing the queries to the suitable tables.
- By directing the queries to appropriate tables, the speed of querying and response generation can be increased.
- Query manager is responsible for scheduling the execution of the queries posed by the user.

Query Manager Architecture

The following screenshot shows the architecture of a query manager. It includes the following:

- Query redirection via C tool or RDBMS
- Stored procedures
- Query management tool
- Query scheduling via C tool or RDBMS
- Query scheduling via third-party software



Spatial data is associated with geographic locations such as cities,towns etc. A spatial database is optimized to store and query data representing objects. These are the objects which are defined in a geometric space.

Characteristics of Spatial Database

A spatial database system has the following characteristics

1. It is a database system
2. It offers spatial data types (SDTs) in its data model and query language.
3. It supports spatial data types in its implementation, providing at least spatial indexing and efficient algorithms for spatial join.

Example

A road map is a visualization of geographic information. A road map is a 2-dimensional object which contains points, lines, and polygons that can represent cities, roads, and political boundaries such as states or provinces.

In general, spatial data can be of two types:

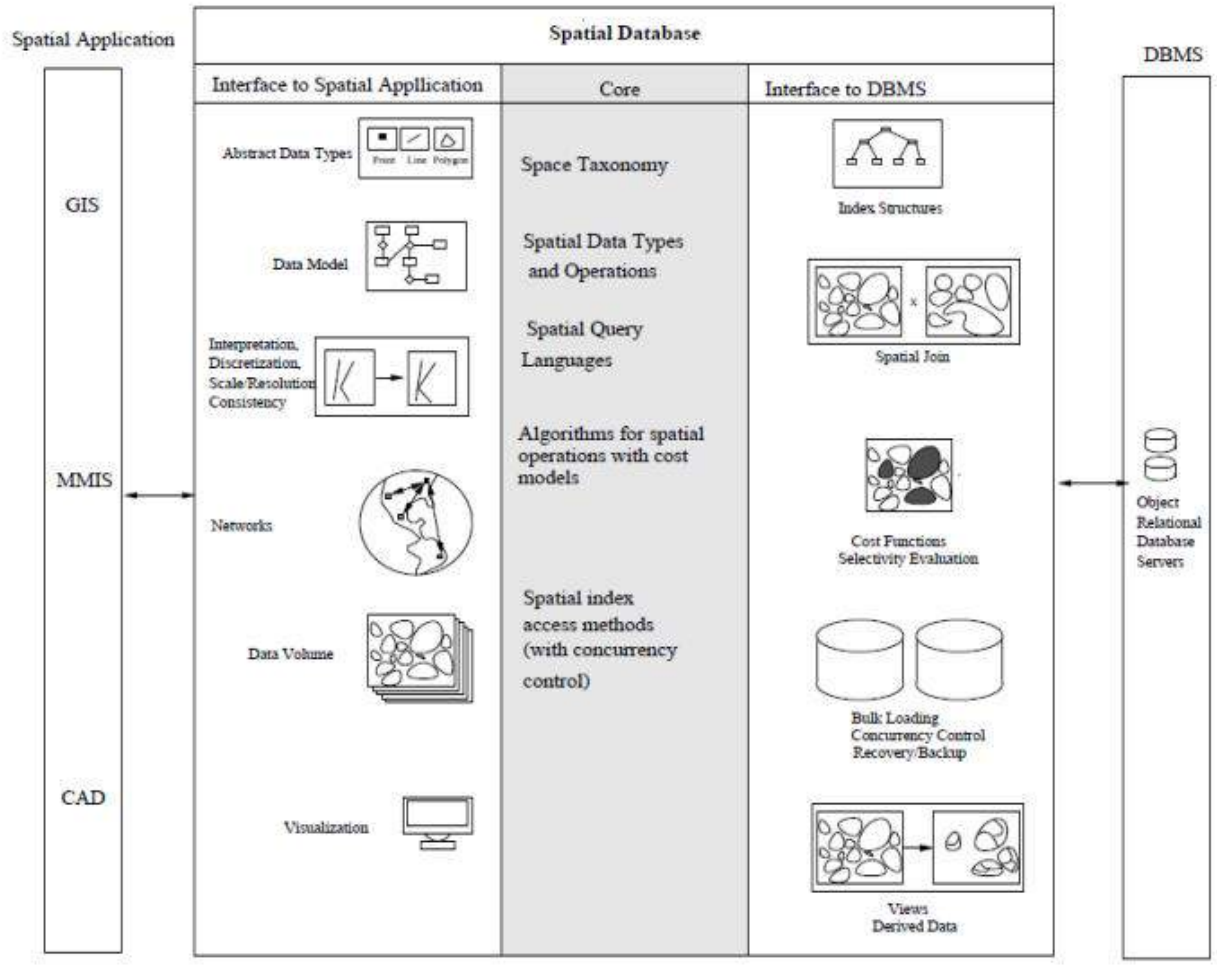
1. Vector data: This data is represented as discrete points, lines and polygons
2. Rastor data: This data is represented as a matrix of square cells.

.ARCHITECTURE OF SPATIAL DATABASE The architecture of Spatial Database Management System is three layer architecture.

1) The top layer: The top layer comprises of the Spatial Data Application through which user communicate directly with the database. These Application includes GIS(Geographic Information

System), MMIS(Multimedia Information System) or CAD(Computer Aided Design). These can be used for capturing, storing, manipulating, analysing, managing and presenting all types of spatial or geographical data.

2) The middle layer: The middle layer is the spatial database where all of the domain knowledge are encapsulated. The Spatial Data Application Communicates directly with this layer. This layer interacts directly with both Application layer and database server. This layer has three parts:



FME for Spatial Data Integration

While there are many tools and software that can help you make use of spatial data, FME is the software of choice for those that need to integrate their spatial data. Safe Software and FME came into existence because of this exact problem. Spatial data varies widely and is often stuck in formats that cannot be easily used by all applications, making it extremely difficult for GIS experts to make use of all the information they have. While it was possible to transform proprietary formats in the past, much of the data would be lost in the conversion. Thus, FME was born.

Spatial models

The term "spatial modelling" refers to a particular form of disaggregation, in which an area is divided into a number (often a large number) of similar units: typically grid squares or polygons. The model may be linked to a GIS for data input and display. The transition from non-spatial to spatial modelling is often considered to be pretty significant, and there are a number of modelling packages that advertise their spatial modelling capabilities: indeed, many are labelled as landscape or landuse modelling tools.

Modular models

The term "modular modelling" usually refers to the use of interchangeable components (or modules) in a model. The component may be a single equation, but typically it is a large component: for example, a plant submodel or a soil water submodel. There have been calls for the development of modular modelling approaches for some two decades, and some working systems, motivated by the advantages that this would confer on the modelling process in terms of model construction, testing and reuseability of components. In addition, a major motive for the adoption of object-oriented software engineering approaches has been its support for modularity in modelling.

The purest form is "plug-and-play" modularity, in which the interfacing between a module and the main model is pre-defined (like the pins on an integrated circuit chip). All the modeller needs to do is to load the module, and it is automatically part of the model. Simile enables you to do this, as a two-step operation. First, you load the module (a Simile model, loaded into a submodel window). Then you select an Interface Specification File which defines the links between the submodel and the rest of the model. This approach has considerable merits: it means that the same Simile model can be inserted as a module in a range of other models, with different interfacing for each one.

At the other extreme, Simile supports "free-form" modularity, in which it is entirely up to the modeller to decide how the inserted submodel links to the rest of the model. This means that the modeller has access to a much greater range of models to use as submodels — ones that were developed with no intention that they be used as a submodel in someone else's model. This removes the need for careful defining of interfaces which plagues current modular (and indeed object-oriented) systems.

Object-oriented models

The term "object-oriented" has a formal meaning in software engineering: it is not just "modelling with objects" in the sense of individual-based modelling. Rather, it reflects a commitment to a number of principles which together characterise the object-oriented approach, including message-passing, encapsulation (hiding internal detail), inheritance (from class to subclass), and polymorphism (the same procedure can operate on different data types). There is a strong movement towards the adoption of object-oriented software engineering approaches in ecological modelling.

It may come as some surprise, therefore, that not only does Simile not incorporate most of the characteristic features of the object-oriented approach, but that we have

deliberately decided not to incorporate them. This is a controversial area, which we won't develop here. But briefly:

- the message-passing paradigm is inappropriate for systems based on differential/difference equations;
- encapsulation is just what modellers does not want: they should have access to any attributes of any object;
- inheritance, and a class hierarchy, can rapidly become extremely messy when the class is a designed model, and the subclass is some modification of this design;

Rejecting encapsulation does not mean that we are against modularity or re-use of components: quite the opposite. We are just against the principle that you really should not know what is inside a modelled component.

Simile does, however, support certain object-oriented design concepts. A Simile diagram with submodels corresponds closely to a UML class diagram. The Simile notation of placing one submodel inside another to indicate containership corresponds closely to a composition association. And Simile provides explicit notation for an association between classes.

Applications

Simile is used in a wide range of applications in the earth, environmental and life sciences, but in this page we would like to highlight some particularly interesting areas to help explain what makes Simile especially well suited to these research projects.

- Agriculture
- Ecosystem Services
- Forestry
- Water Management/Hydrology

In Data Warehouses and On-Line Analytical Processing systems hierarchies are used to analyze high volumes of historical data. On the other hand, the advantage of using spatial data in the analysis process is widely recognized. Therefore, in order to satisfy the growing requirements of decision-making users it is necessary to extend hierarchies for representing spatial data. Based on an analysis of real-world spatial applications, this paper defines different kinds of spatial hierarchies and gives a conceptual representation of them. Further, we study the summarizability problem and classify the topological relationships between hierarchy levels according to the procedures required for ensuring correct measure aggregation.