BELGIUM CAMPUS
iTversity
It's the way we're wired

DATABASE DEVELOPMENT

ARE YOU READY?

DBD371/381

COURSE OUTLINE

- Distributed database systems overview
- Overview of computer networking
- Distributed database design concepts
- Transparencies in distributed database design
- Distributed Database Management System (DDBMS) concepts
- Distributed DBMS architectures
- Data Access and Control
- Distributed Query Processing
- Distributed Transaction Management
- Introduction to NoSQL Databases

Updated: May 2024

DATABASE DEVELOPMENT 371/381

IMPORTANT INFORMATION

DURATION

2

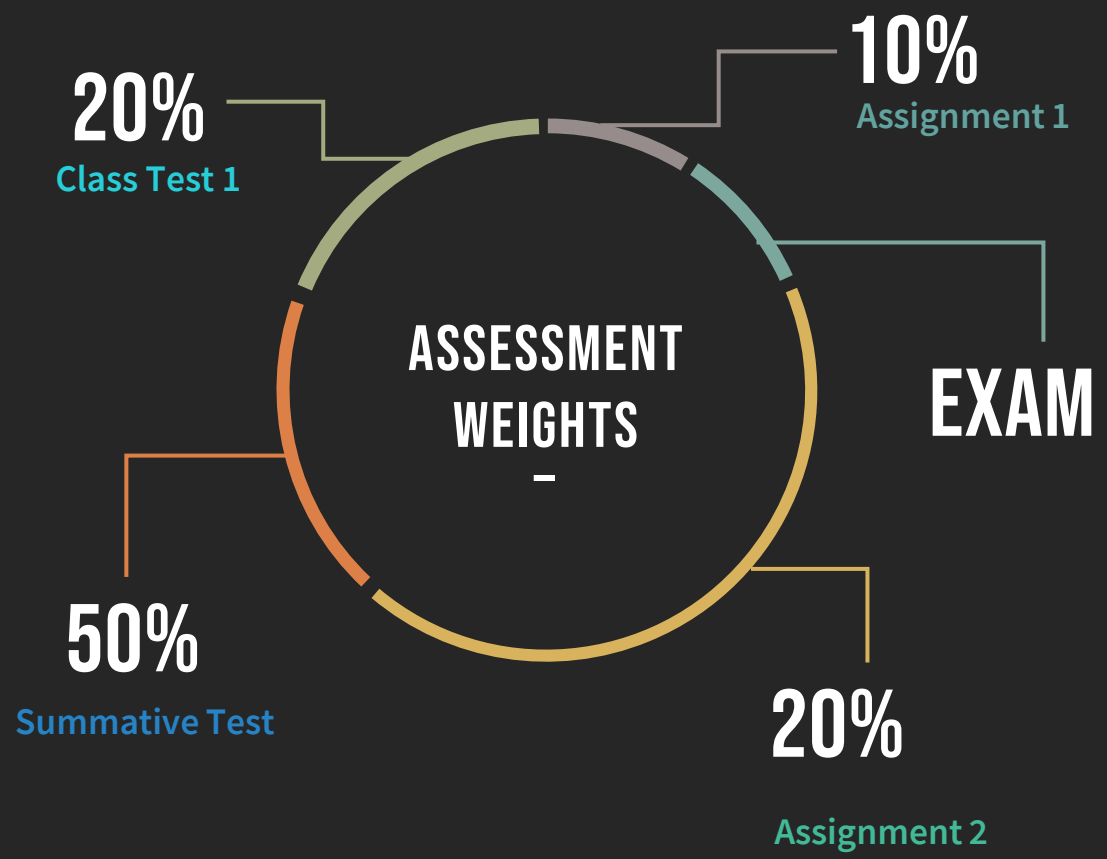
weeks

ASSESSMENTS

2 assignments/ projects

2 tests

1 Examination



RECOMMENDED BOOKS

- Özsu, M.T., Valduriez, P. (2011), *Principles of Distributed Database Systems*, Third Edition, Springer-Verlag New York. [ISBN 978-1-4419-8833-1].
- Jocelyn O. Padallan (2020) *Distributed Database Architecture*. Ashland: Arcler Press. Available at: https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=2725224&site=ehost-live
- Amit Phaltankar et al. (2020) MongoDB Fundamentals : A Hands-on Guide to Using MongoDB and Atlas in the Real World. Birmingham: Packt Publishing. Available at: https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=2713655&site=ehost-live

DISTRIBUTED DATABASES

Introduction

LEARNING OBJECTIVES

BELGIUM CAMPUS iTversity
It's the way we're wired

- Distributed Database System characteristics
- Data Delivery Alternatives
- Distributed Database System vs Centralized Client-Server System
- Promises of DDBSs
- Types of transparencies
- Potential problems with distributed DBMSs
- Distributed Database Design issues

What is a Distributed Database System?

- A distributed database is a collection of multiple, logically interrelated databases distributed over a computer network.
- A distributed database management system (distributed DBMS) is then defined as the software system that permits the management of the distributed database and makes the distribution transparent to the users
- **Two important terms**

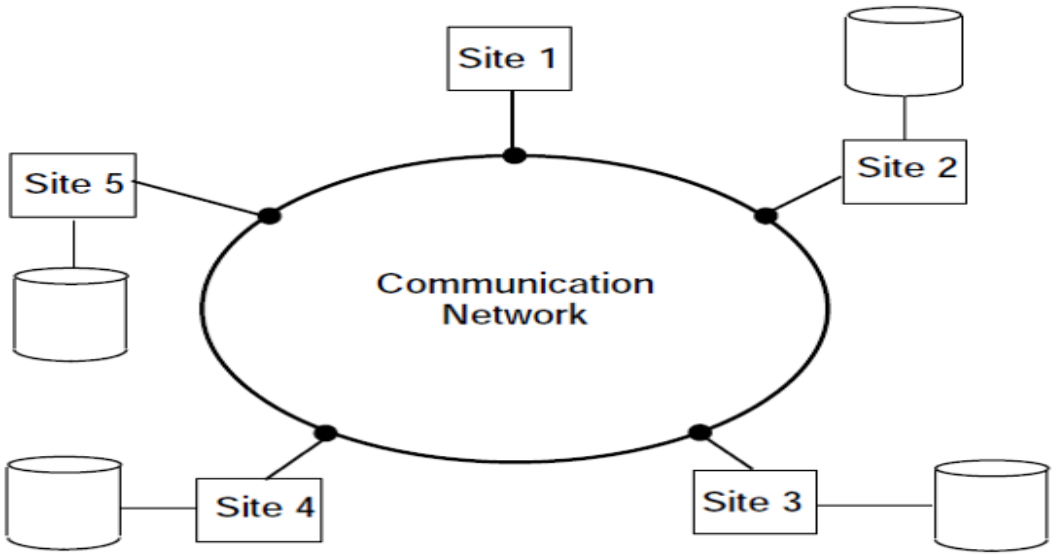
1. "**Logically interrelated**" →not only be logically related, but there should be some structure among the files, and access should be via a common interface
2. "**Distributed over a computer network**."→implies that the communication between the Databases is done over a network instead of through shared memory or shared disk

MAIN CHARACTERISTICS:

- The DDBS, in its general meaning, is a solution that has been adopted for a single organization
- **Major characteristics of a DDBS**
  - **Data management at multiple sites**: data in a DDBS is stored and managed at geographically multiple sites where tables are created, data entered by different users and multiple applications running.
  - **Local requirements:** Each of the sites storing data in a DDBS is called a l**ocal site**. Every site is mainly concerned with the requirements and data management for the users directly associated with that site, called **local users**.
  - **Global perspective:** Apart from catering the local requirements, the DDBS also fulfils the global requirements. The DDBS fulfils the global requirements in a transparent way,

What is a Distributed Database System?

- A distributed DB is an environment where data are distributed among a number of sites



ACCESS TO A DDBS

- In a DDBS environment, three types of accesses are involved:
  1. **Local access:** the access by the users connected to a site and accessing the data from the same site.
  2. **Remote access:** a user connected to a site, lets say site 1, and accessing the data from site 2.
  3. **Global access:** no matter from where ever the access is made, data will be displayed after being collected from all locations.

APPLICATION OF DISTRIBUTED DATABASES

- Candidate applications for a DDBS have following two main characteristics:
  - 1- Large number of users
  - 2- Users are physically spread across large geographical area

Candidates for a DDBS

1. **Banking Applications:**A bank has large number of customers and its branches are spread across the whole country and sometimes many of them have branches around the world
2. **Air ticketing:** facility to book a seat in any airline from any location to any destination.
3. **Business at multiple locations:** A company having offices at multiple locations, or different units at different locations, like production, warehouses, sales operating from different locations, each site storing data locally: these units need to access each other's data and data from all the sites is required for the global access.

DATA DELIVERY ALTERNATIVES

- In distributed databases, data is “delivered” from the sites where they are stored to where the query is posed.
- **Three orthogonal dimensions/Alternatives of DDBS exist:**

1. Delivery modes
2. Frequency
3. Communication methods

Delivery modes

- Delivery modes are:

1. **Pull-only**→the transfer of data from servers to clients is initiated by a client request
2. **Push-only** →the transfer of data from servers to clients is initiated by a server push in the absence of any specific request from clients.
3. **Hybrid**→combines the **client-pull** and **server-pus**h mechanisms Transfer of information from servers to clients is first initiated by a client pull (by posing the query), and the subsequent transfer of updated information to clients is initiated by a **server push.**

Frequency

- Three typical frequency measurements that can be used to classify the regularity of data delivery.

1. **Periodic:**
   - **D**ata are sent from the server to clients at regular intervals defined by system default or by clients using their profiles. Both pull and push can be performed in periodic fashion.
   - **Periodic delivery** is carried out on a **regular and pre-specified repeating schedule** eg A client request for IBM's stock price every week
2. **Conditional,**
   - Data are sent from servers **whenever certain conditions installed by clients in their profiles are satisfied**. Such conditions can be as simple as a given time span or as complicated as event-condition-action rules is
   - mostly used in the hybrid or push-only delivery systems
3. **Ad-hoc Or Irregular:**
   - is irregular and is performed mostly in a pure pull-based system.
   - Data are pulled from servers to clients in an ad-hoc fashion whenever clients request

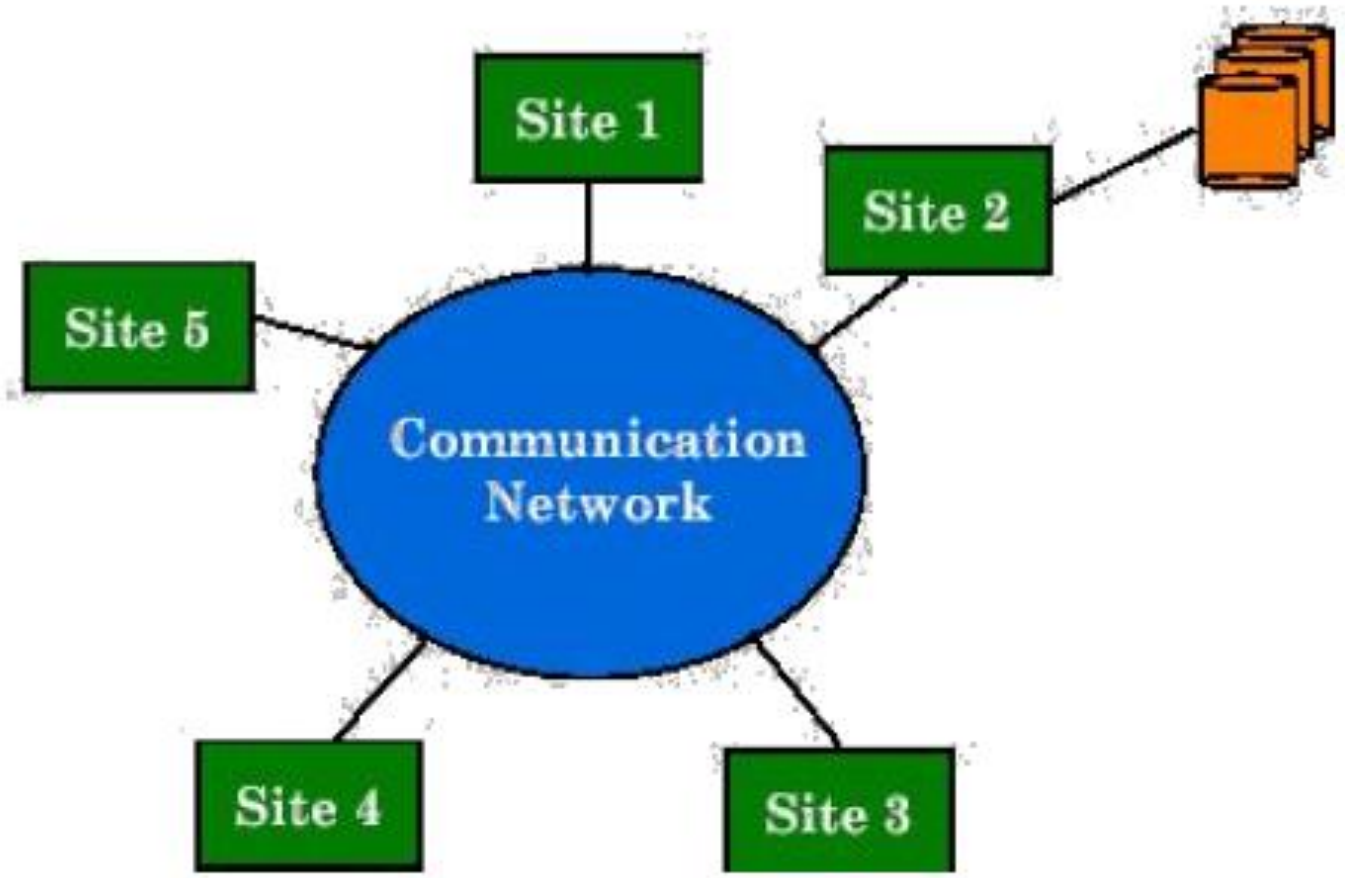
Communication methods

- Communication methods determine the various ways in which servers and clients communicate for delivering information to clients and consists of mainly two alternatives:
- **unicast :→**
  - The communication from a server to a client is one-to-one:
  - the server sends data to one client using a particular delivery mode with some frequency.
- **one-to-many**
  - the server sends data to a number of clients.
  - may use a multicast or broadcast protocol.

RESEMBLING SETUPS:

- **Centralized C/S System**
  - Data management is carried on a single centralized system
  - data is accessed from different machine (clients)
  - All machines are connected with each other through a communication link (network).
  - Data storage and management is mainly done on the server.
  - The data is associated with a single site(server)
  - Rest of the machines are accessing data from the server.

CENTRALIZED C/S SYSTEM



EXAMPLE

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

ASG

ENO	PNO	RESP	DUR
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E8	P3	Manager	40

PROJ

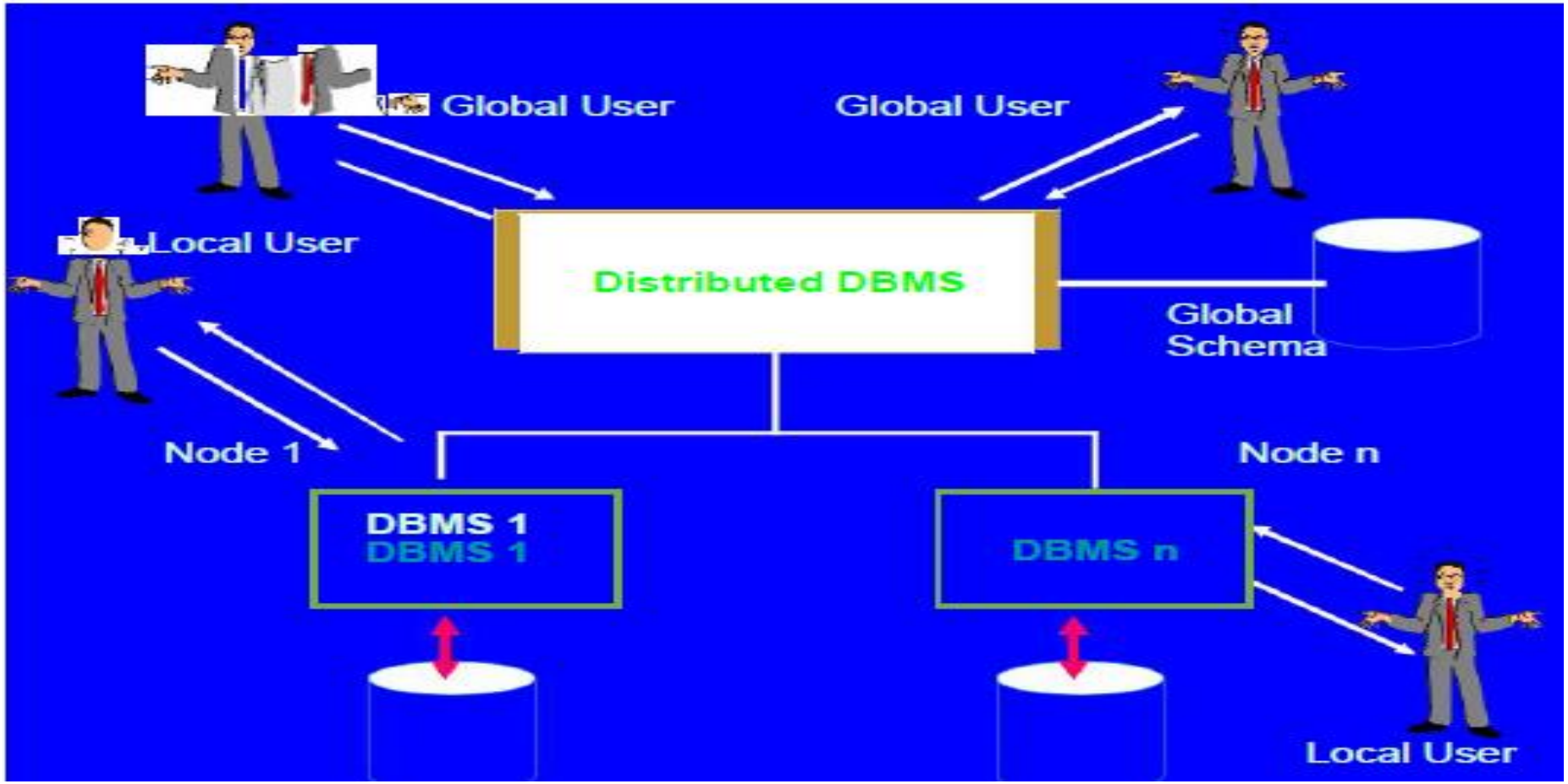
PNO	PNAME	BUDGET
P1	Instrumentation	150000
P2	Database Develop.	135000
P3	CAD/CAM	250000
P4	Maintenance	310000

PAY

TITLE	SAL
Elect. Eng.	40000
Syst. Anal.	34000
Mech. Eng.	27000
Programmer	24000

- An engineering firm that has offices in Boston, Waterloo, Paris and San Francisco.
- They run projects at each of these sites and would like to maintain a database of their employees, the projects and other related data.
- This sample database is used throughout this course

Distributed Database System



There are a number of local dbmss called local nodes

DISTRIBUTED DATABASE SYSTEM

- The DDBMS contains the global schema, that is basically the merger of all local schemas.
- There is no data underlying the global schema rather the data is contained with the local DBMSs.
- The user connected to the DDBMS layer are called global users, as their queries are replied by collecting data from all the local nodes, and this activity of distribution is transparent from the global users.

REASONS FOR DDBS:

- **Local units want control over data**. Data is generated locally and accessed locally but there are situations where you require data from all sites
- **Reduce telecom cost:** data is closer and local access thus reduces the overall telecommunication cost.
- **Reduce the risk of telecom failures:**failure at a single site will disturb only the users of that site

Promises of DDBSs

- **Four fundamental Advantages:**
  1. Transparent Management of Distributed and Replicated Data
  2. Reliable access to data through distributed transactions
  3. Improved performance
  4. Easier system expansion

Transparent Management of Distributed and Replicated Data

- Transparency refers to separation of the higher-level semantics of a system from lower-level implementation issues i.e **Transparent system "hides" the implementation details from users.**
  - ❑Provides for the development of complex applications
  - ❑Users can pose a query without paying any attention to the fragmentation, location, or replication of data, and let the system worry about resolving these issues.
  - ❑For a system to adequately deal with this type of query over a distributed, fragmented and replicated database, it needs to be able to deal with a number of different types of transparencies.

Types of transparencies

1. **Data Independence**
2. **Network Transparency Or distribution transparency**
   - **Distribution transparency**
   - **Naming transparency**
3. **Replication Transparency**
4. **Fragmentation Transparency**

Types of transparencies

- **Data Independence:**

❑The immunity of user applications to changes in the definition and organization of data, and vice versa.

❑Data definition occurs at two levels :

1. Logical structure (schema definition) of the data
2. physical structure(physical data description)

- **Therefore two types of Data Independence:**
  1. **Logical data independence: refers to the immunity of user applications to changes in the logical structure**
  2. **Physical data independence: deals with hiding the details of the storage structure from user applications**

  If we change the physical or lower level then there is little or no effect on the conceptual level.

  ➢ When a user application is written, it should not be concerned with the details of physical data organization. Therefore, the user application should not need to be modified when data organization changes occur due to performance considerations.

NETWORK TRANSPARENCY OR DISTRIBUTION TRANSPARENCY

- User should be protected from the operational details of the network; possibly even hiding the existence of the network.
- The user is unaware of even the existence of the network, that frees him from the problems and complexities of network.
- **Distribution transparency** requires that users do not have to specify where data are located.(Location transparency)
- The command used to perform a task is independent of both the location of the data and the system on which an operation is carried out.
- **Naming transparency** means that a unique name is provided for each object in the database

Replication Transparency

- While data are replicated the transparency issue is that the system should handle the management of copies and the user should act as if there is a single copy of the data
- The DDBS hides the replication from the end user, with the advantage that, the user gets the benefits of the system and does not need to know the details or understand the technical details.

Fragmentation Transparency

- Fragmentation refers to the division of each database relation into smaller fragments and treat each fragment as a separate database object for reasons of performance, availability, and reliability.
- Each replica is not the full relation but only a subset of it; thus less space is required and fewer data items need be managed.
- Two types of fragmentation alternatives exist:
  1. **Horizontal fragmentation:** relation is partitioned into a set of sub-relations each of which have a subset of the tuples (rows) of the original relation
  2. **Vertical fragmentation:** each sub-relation is defined on a subset of the attributes (columns) of the original relation.

FRAGMENTATION ALTERNATIVES – HORIZONTAL

$PROJ_1$: projects with budgets less than $200,000

$PROJ_2$: projects with budgets greater than or equal to $200,000

PROJ

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York
P3	CAD/CAM	250000	New York
P4	Maintenance	310000	Paris
P5	CAD/CAM	500000	Boston

$PROJ_1$

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York

Stored in London

$PROJ_2$

PNO	PNAME	BUDGET	LOC
P3	CAD/CAM	250000	New York
P4	Maintenance	310000	Paris
P5	CAD/CAM	500000	Boston

Stored in Boston

FRAGMENTATION ALTERNATIVES – VERTICAL

$PROJ_1$: information about project budgets

$PROJ_2$: information about project names and locations

PROJ

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York
P3	CAD/CAM	250000	New York
P4	Maintenance	310000	Paris
P5	CAD/CAM	500000	Boston

Horizontal partitioning is more common

$PROJ_1$

PNO	BUDGET
P1	150000
P2	135000
P3	250000
P4	310000
P5	500000

Stored in London

$PROJ_2$

PNO	PNAME	LOC
P1	Instrumentation	Montreal
P2	Database Develop.	New York
P3	CAD/CAM	New York
P4	Maintenance	Paris
P5	CAD/CAM	Boston

Stored in Boston

15

RESPONSIBILITY OF TRANSPARENCY

- Transparency is desirable since it hides all the technical details from the users, and make the use/access very easy
- Three major players are responsible of providing transparency
  1. Language/Compiler,
  2. Operating System
  3. DDBMS.

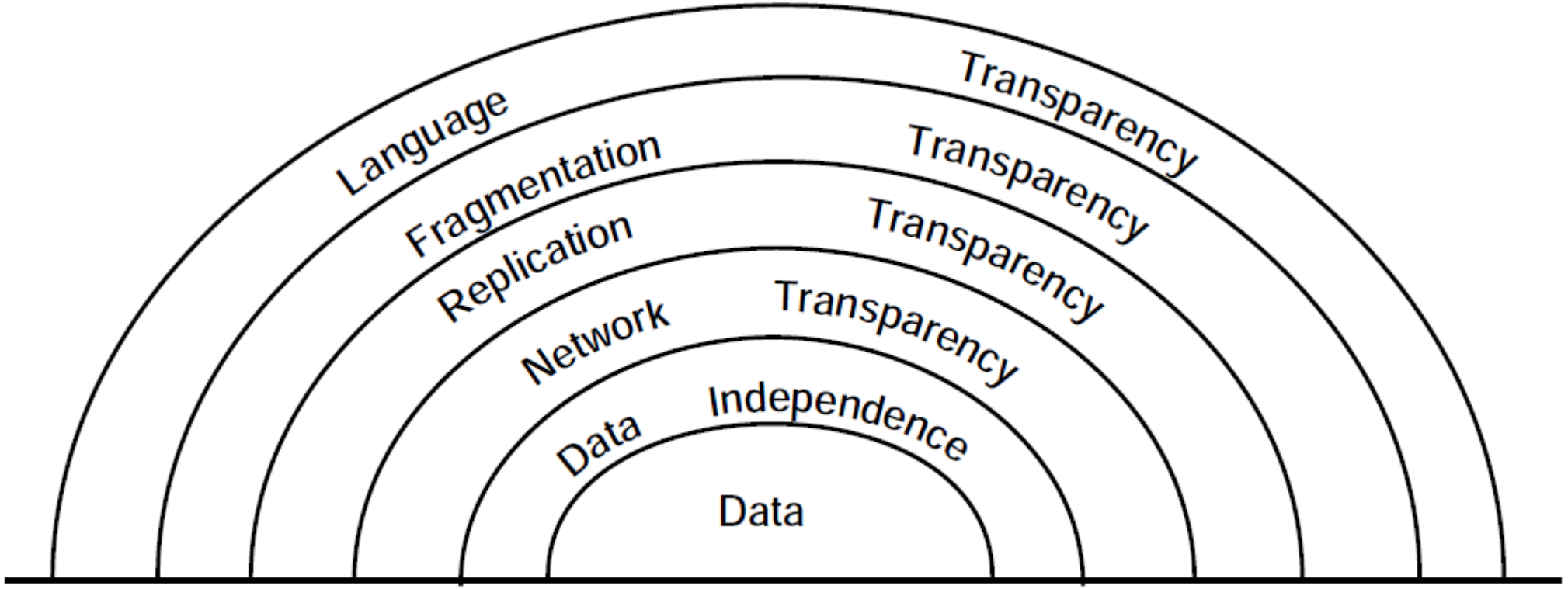
Who Should Provide Transparency?

- Three distinct layers at which the transparency services can be provided
  1. **Access layer:** responsibility of providing transparent access to data resources transparency features can be built into the **user language**, which then translates the requested services into required operations i.e. language or compiler used to develop the front end application programs in a database system (VB, VB.Net, JSP, Perl)
  2. **Operating system level. O**perating systems provide some level of transparency to system users. This layer is not so visible to the end-users most of the time, since it resides below the DDBMS or the language used to develop the application. Both DDBMS and the language heavily depend on the features supported by the operating system (OS).
  3. **DBMS Level** : It is the responsibility of the DBMS to make all the necessary translations from the operating system to the higher-level user interface.

PRACTICAL SITUATION:

- The features of all three components are used together to provide a very cohesive and easy to use environment for the users.
- Eg
- Distributed OS provides the network transparency,
- DDBMS provides the fragmentation or replication transparency,
- The front-end tool provides the transparency of linking and manipulation of data.
- All these features combined, make the DDBMS a viable and user friendly option to use.

Layers of Transparency



CONCEPT CHECK

- Discuss the various forms of transparency within a distributed database environment.
- Find out what "language transparency" entails. Discuss your findings with the rest of the class.

Reliability Through Distributed Transactions

- DDBS environment reduces the chances of single point of failure and that increases the reliability of the entire system.
- The failure of a single site, or the failure of a communication link which makes one or more sites unreachable, is not sufficient to bring down the entire system.
- In the case of a distributed database,this means that some of the data may be unreachable, but with proper care, users may be permitted to access other parts of the distributed database.
- The "proper care" comes in the form of support for distributed transactions and application protocols
- **Concurrency issues**: the concurrent access means the access of data by multiple users at the same time. however these issues become more critical in case of a DDBS: Consistency of data across multiple sites is a serious issue especially when same data is duplicated at multiple site
- Distributed transactions should be able to be executed at a number of sites at which they access the local database without fear of failure

Reliability Through Distributed Transactions

- User applications do not need to be concerned with coordinating their accesses to individual local databases nor do they need to worry about the possibility of site or communication link failures during the execution of their transactions
- Supporting replicas requires the implementation of replica control protocols that enforce a specified semantics of accessing them

Improved Performance

- The DDBS provides improved performance through :

1. **Data Localization :**data is stored in close proximity to its points of use
   - Advantages
     - 1. Since each site handles only a portion of the database, contention for CPU and I/O services is not as severe as for centralized databases.
     - 2. Localization reduces remote access delays that are usually involved in wide area networks (for example, the minimum round-trip message propagation delay in satellite-based systems is about 1 second).
2. **Query Parallelism:** a query in certain situations can be executed in parallel, that improves performance.
   1. Two types of query parallelism,
      1. inter-query :multiple queries executed at the same time
      2. intra-query: query is split across multiple sites and this split components are executed in parallel

Easier System Expansion

- In a distributed environment, it is much easier to accommodate increasing database sizes.
- Expansion can usually be handled by adding processing and storage power to the network
- For many applications, it is more economical to put together a distributed computer system (whether composed of mainframes or workstations) with sufficient power than it is to establish a single, centralized system to run these tasks.

Complications Introduced by Distribution

1. There are certain aspects that complicate a DDBS environment.

- **Selection of the Copy**:
  - In case of replication, the distributed database system is responsible for choosing the right copy based on a number constraining variables ☹ (distance or load,site availability etc)
- **Failure recovery:**
  - in case of replication the synchronization of copies after failure has to be dealt with carefully. is considerably harder than for a centralized system
  - IF some sites fail (e.g., by either hardware or software malfunction), or if some communication links fail while an update is being executed, the system must make sure that the effects will be reflected on the data residing at the failing or unreachable sites as soon as the system can recover from the failure.
  - The synchronization of transactions on multiple sites is considerably harder than for a centralized system.
- **Complexity:**
  - Since the data is stored at multiple sites and has to be managed the overall system is more complex as compared to a centralized database system.
- **Cost:**
  - A DDBS involves more cost, as the hardware and the trained manpower has to be deployed at multiple sites.
- **Distribution of Control**:
  - The access to the data should be allowed carefully. Rights to access data should be well defined for local sites.

Potential problems with distributed DBMSs

1. Database design:
2. Query processing:
3. Complexity of building distributed applications
4. Increased cost of replicating resources
5. Managing distribution,
6. the devolution of control to many centers and the difficulty of reaching agreements,
7. security

Design Issues

1. **Distributed Database Design:** how the database and the applications that run against it should be placed across the sites.
2. **Distributed Directory Management**: global to the entire DDBS or local to each site
3. **Distributed Query Processing:** how to decide on a strategy for executing each query over the network in the most cost-effective way
4. **Distributed Concurrency Control**: how to maintain the consistency of multiple copies of the database
5. **Distributed Deadlock Management**: alternatives of prevention, avoidance, and detection/recovery deadlocks
6. **Reliability of Distributed DBMS:** how to ensure the consistency of the database as well as to detect failures and recover from them
7. **Replication:** should the distributed database be (partially or fully) replicated
8. **Relationship among Problems:** relationship among the components
9. **Additional Issues:** Since the environment has changed significantly are there other issues

Distributed Database Design

- The question that is being addressed is how the database and the applications that run against it should be placed across the sites
- Two basic alternatives:
  1. **Partitioned (or non-replicated)** : database is divided into a number of disjoint partitions each of which is placed at a different site.
  2. **Replicated.** Database can be either fully replicated (also called fully duplicated) where the entire database is stored at each site, or partially replicated (or partially duplicated) where each partition of the database is stored at more than one site, but not at all the sites.
- Two fundamental design issues are
  1. **fragmentation,** the separation of the database into partitions called fragments,
  2. **distribution,** the optimum distribution of fragments.

Distributed Directory Management

- A directory contains information (such as descriptions and locations) about data items in the database.
- A directory may be global to the entire DDBS or local to each site; it can be centralized at one site or distributed over several sites; there can be a single copy or multiple copies

Distributed Query Processing

- Query processing deals with designing algorithms that analyze queries and convert them into a series of data manipulation operations.
- The problem is how to decide on a strategy for executing each query over the network in the most cost-effective way,
- **Factors to be considered:**
  - ❑distribution of data
  - ❑ communication costs
  - ❑lack of sufficient locally-available information
- The objective is to optimize where the inherent parallelism is used to improve the performance of executing the transaction, subject to the above-mentioned constraints.

Distributed Concurrency Control

- Concurrency control involves the synchronization of accesses to the distributed database, such that the integrity of the database is maintained.
- One not only has to worry about the integrity of a single database, but also about the consistency of multiple copies of the databases
- The condition that requires all the values of multiple copies of every data item to converge to the same value is called mutual consistency.
- Two general classes
    1. **pessimistic** , synchronizing the execution of user requests before the execution starts,
    2. **optimistic,** executing the requests and then checking if the execution has compromised the consistency of the database.
- Two fundamental primitives that can be used with both approaches
    1. **Locking:** which is based on the mutual exclusion of accesses to data items,
    2. **Timestamping**: where the transaction executions are ordered based on timestamps.

Distributed Deadlock Management

- The competition among users for access to a set of resources (data, in this case) can result in a deadlock if the synchronization mechanism is based on locking.
- Alternatives :
  1. prevention:
  2. avoidance,
  3. detection/recovery

Reliability of Distributed DBMS

- It is important that mechanisms be provided to ensure the consistency of the database as well as to detect failures and recover from them
- The implication for DDBSs is that when a failure occurs and various sites become either inoperable or inaccessible, the databases at the operational sites remain consistent and up to date.
- when the computer system or network recovers from the failure, the DDBSs should be able to recover and bring the databases at the failed sites up-to-date

Replication

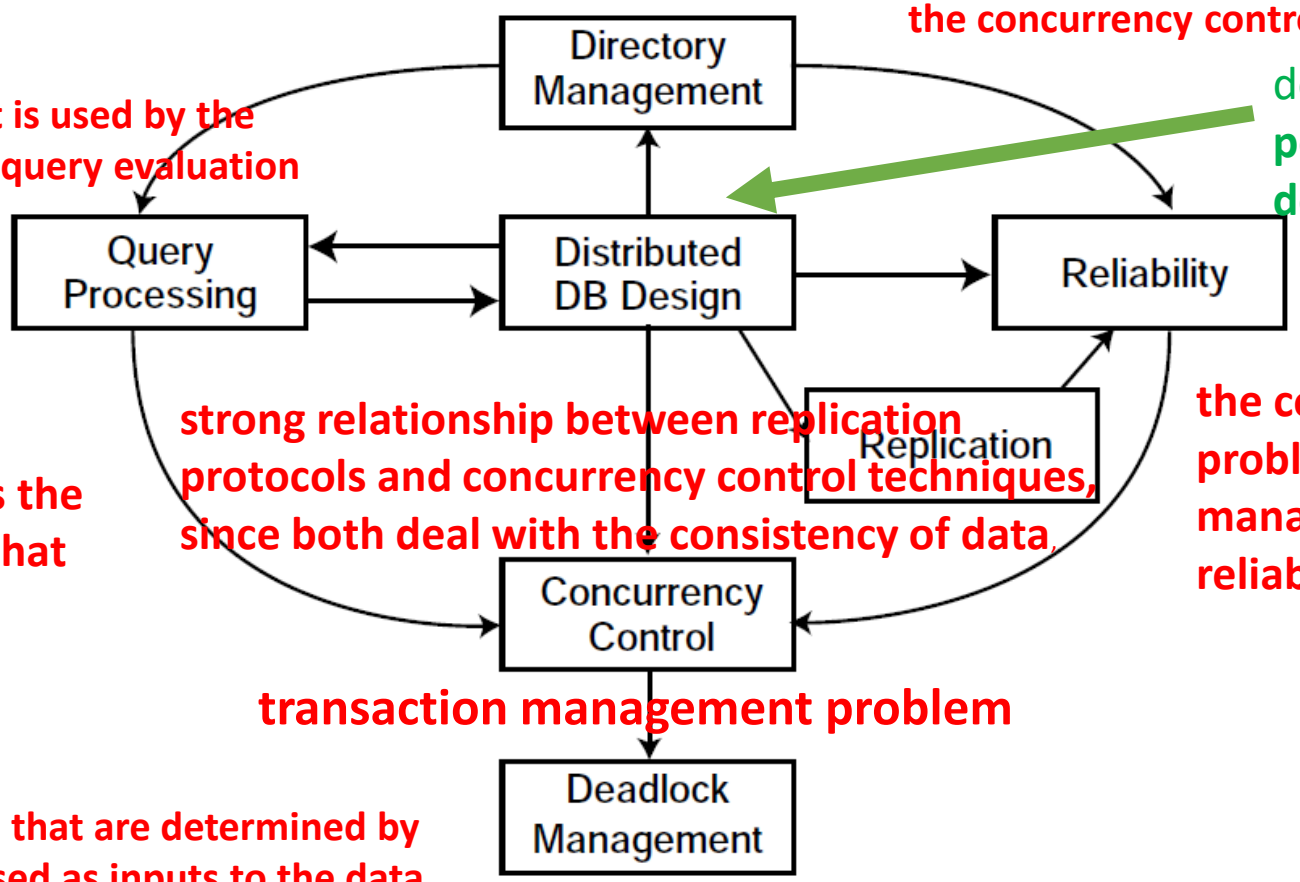
- It is necessary to implement protocols that ensure the consistency of the replicas,i.e., copies of the same data item have the same value.
- Protocols can be:
  1. Eager in that they force the updates to be applied to all the replicas before the transaction completes, or
  2. Lazy in that the transaction updates one copy (called the master) from which updates are propagated to the others after the transaction completes

Relationship among Problems

- The design of distributed databases affects many areas
- It affects directory management, because the definition of fragments and their placement determine the contents of the directory (or directories) as well as the strategies that may be employed to manage them.
- Each problem is affected by the solutions found for the others, and in turn affects the set of feasible solutions for them.

Definition of fragments and their placement determine the contents of the directory as well as the strategies that may be employed to manage them

Local recovery techniques and distributed reliability protocols. influence the choice of the concurrency control techniques



design lies at the heart of all issues
performance of a DDBS mainly depends on the database design

fragment structure and placement is used by the query processor to determine the query evaluation strategy.

strong relationship between replication protocols and concurrency control techniques, since both deal with the consistency of data

the concurrency control problem, the deadlock management problem, and reliability issues closely linked

Replication of fragments affects the concurrency control strategies that might be employed.

transaction management problem

Access and usage patterns that are determined by the query processor are used as inputs to the data distribution and fragmentation algorithms.

Additional Issues

- environment has changed significantly
- One of the important developments has been the move towards “looser” federation among data sources, which may also be heterogeneous
- This has given rise to the development of multi-database systems
  - ❑ federated databases
  - ❑ data integration systems
- The growth of the Internet as a fundamental networking platform has raised important questions about the assumptions underlying distributed database systems.
- re-emergence of peer-to-peer computing,
- growth of the World Wide Web

Both aim to improve data sharing

- there is also a strong relationship between distributed databases and parallel databases.

DISCUSSION

- Discuss some of the factors that influenced the evolution from centralized DBMSs to distributed DBMSs.
- Problems encountered in distributed database systems are not isolated from one another. Each problem is affected by the solutions found for the others, and in turn affects the set of feasible solutions for them.
- Discuss how these problems are related to one another.

QUESTIONS ??

REVISION QUESTIONS

- Explain the difference between a distributed database and distributed processing.
- What are the four implicit assumptions of distributed database environment?
- In distributed databases, we characterize data delivery alternatives along three orthogonal dimensions. What are those three dimensions? List the options available in each of the dimensions.
- List and explain the transparency features of a DDBMS.
- Describe the three data fragmentation strategies. Give some examples of each.
- What is data replication, and what are the three replication strategies?