Co-funded by the
Erasmus+ Programme
of the European Union
MASTIS

Distributed Database Systems and Data Warehouses

Dr. Volodymyr Sokol
(vlad.sokol@gmail.com)

LECTION 3

Date’s Twelve Rules for a DDBMS

- **Fundamental principle**
  - To the user, a distributed system should look exactly like a non-distributed system
- **(1) Local autonomy**. The sites in a distributed system should be autonomous. In this context, autonomy means that:
  - local data is locally owned and managed
  - local operations remain purely local
  - all operations at a given site are controlled by that site

Date's Twelve Rules for a DDBMS

- **(2) No reliance on a central site.** There should be no one site without which the system cannot operate. This implies that there should be no central servers for services such as transaction management, deadlock detection, query optimization, and management of the global system catalog
- **(3) Continuous operation.** Ideally, there should never be a need for a planned system shutdown, for operations such as:
  - adding or removing a site from the system
  - the dynamic creation and deletion of fragments at one or more sites
- **(4) Location independence.** Location independence is equivalent to location transparency. The user should be able to access the database from any site. Furthermore, the user should be able to access all data as if it were stored at the user's site, no matter where it is physically stored

Date's Twelve Rules for a DDBMS

- **(5) Fragmentation independence.** The user should be able to access the data, no matter how it is fragmented
- **(6) Replication independence.** The user should be unaware that data has been replicated. Thus, the user should not be able to access a particular copy of a data item directly, nor should the user have to specifically update all copies of a data item
- **(7) Distributed query processing.** The system should be capable of processing queries that reference data at more than one site

Date’s Twelve Rules for a DDBMS

- **(8) Distributed transaction processing.** The system should support the transaction as the unit of recovery. The system should ensure that both global and local transactions conform to the ACID rules for transactions, namely: atomicity, consistency, isolation, and durability
- **(9) Hardware independence.** It should be possible to run the DDBMS on a variety of hardware platforms
- **(10) Operating system independence.** As a corollary to the previous rule, it should be possible to run the DDBMS on a variety of operating systems

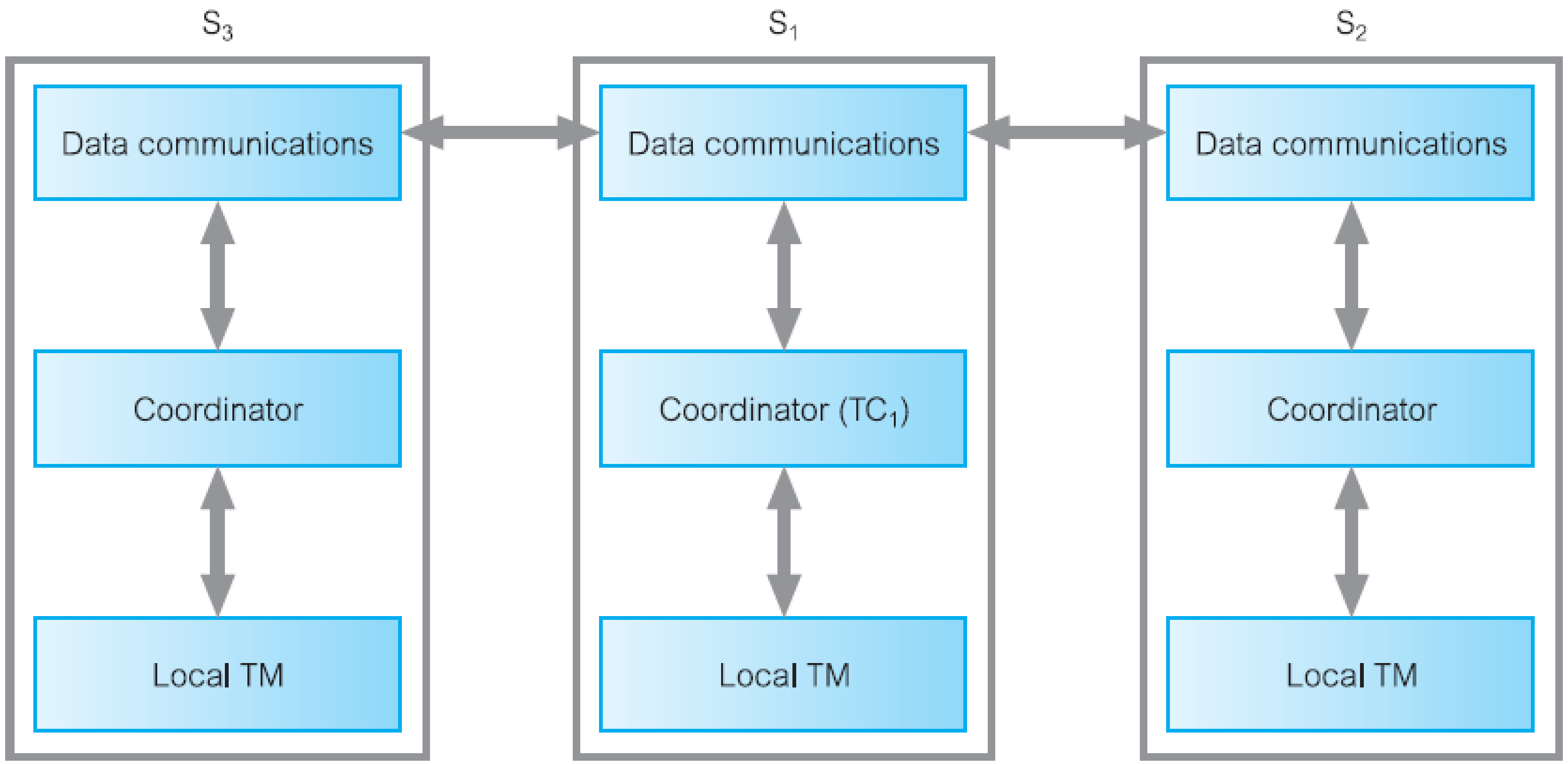
Date's Twelve Rules for a DDBMS

- **(11) Network independence.** Again, it should be possible to run the DDBMS on a variety of disparate communication networks
- **(12) Database independence.** It should be possible to have a DDBMS made up of different local DBMSs, perhaps supporting different underlying data models. In other words, the system should support heterogeneity

Distributed Transaction Management

- The **transaction manager** coordinates transactions on behalf of application programs, communicating with the **scheduler**
- In the event of a failure occurring during the transaction, the **recovery manager** ensures that the database is restored to the state it was in before the start of the transaction
- The **buffer manager** is responsible for the efficient transfer of data between disk storage and main memory
- In addition, there is also a **global transaction manager** or **transaction coordinator** at each site to coordinate the execution of both the global and local transactions initiated at that site
- Inter-site communication is still through the **data communications** component (transaction managers at different sites do not communicate directly with each other)

Distributed Transaction Management



Co-funded by the
Erasmus+ Programme
of the European Union
MASTIS

Distributed Concurrency Control - Objectives

Good concurrency control mechanism for distributed DBMSs should:

- be resilient to site and communication failure
- permit parallelism to satisfy performance requirements
- incur modest computational and storage overhead
- perform satisfactorily in a network environment that has significant communication delay
- place few constraints on the structure of atomic actions