Chapter 1: Introduction

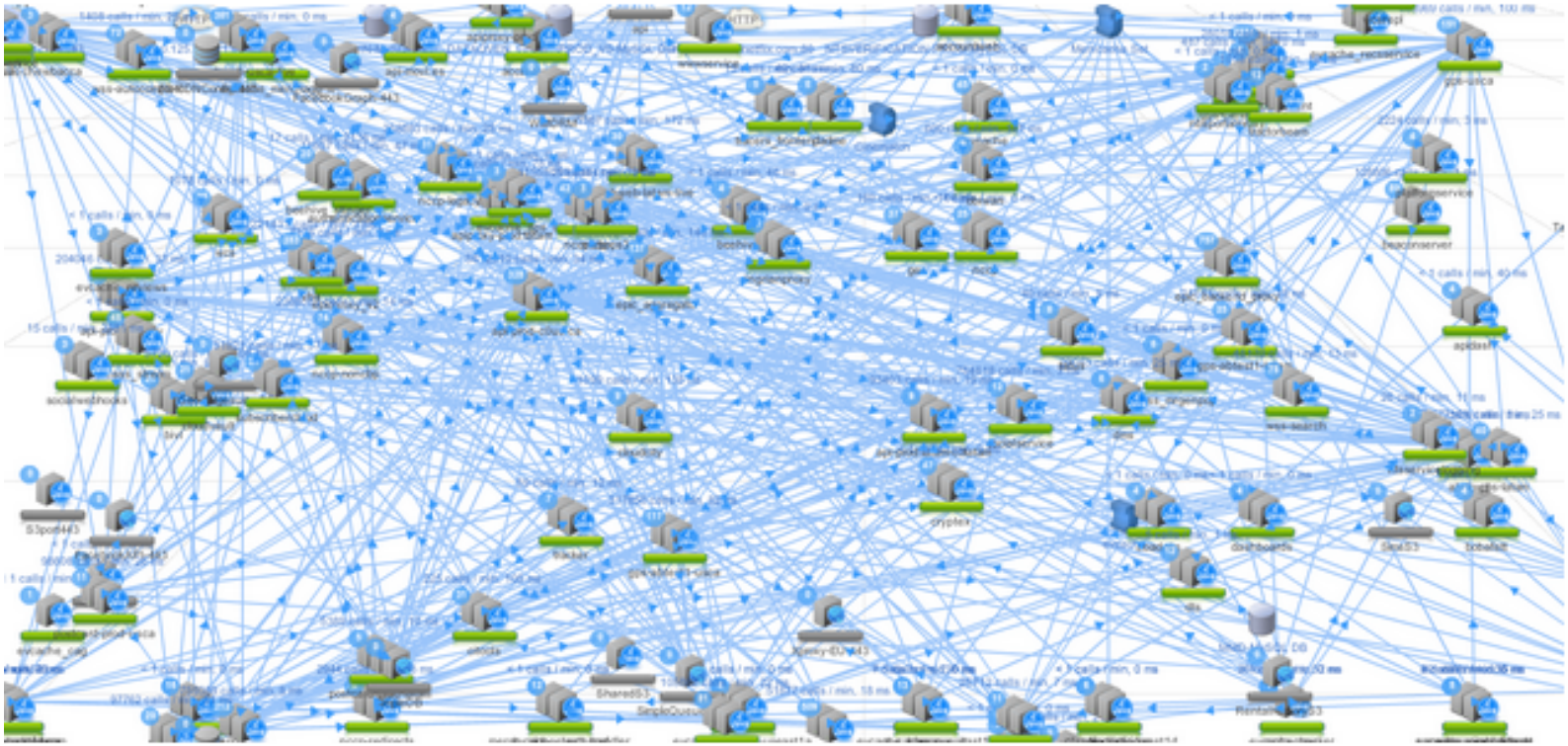
What is a Distributed System?

What is a Distributed System?

- *A distributed system is a collection of independent computers that appears to its users as a single coherent system.*

Examples (1)

Examples (1)



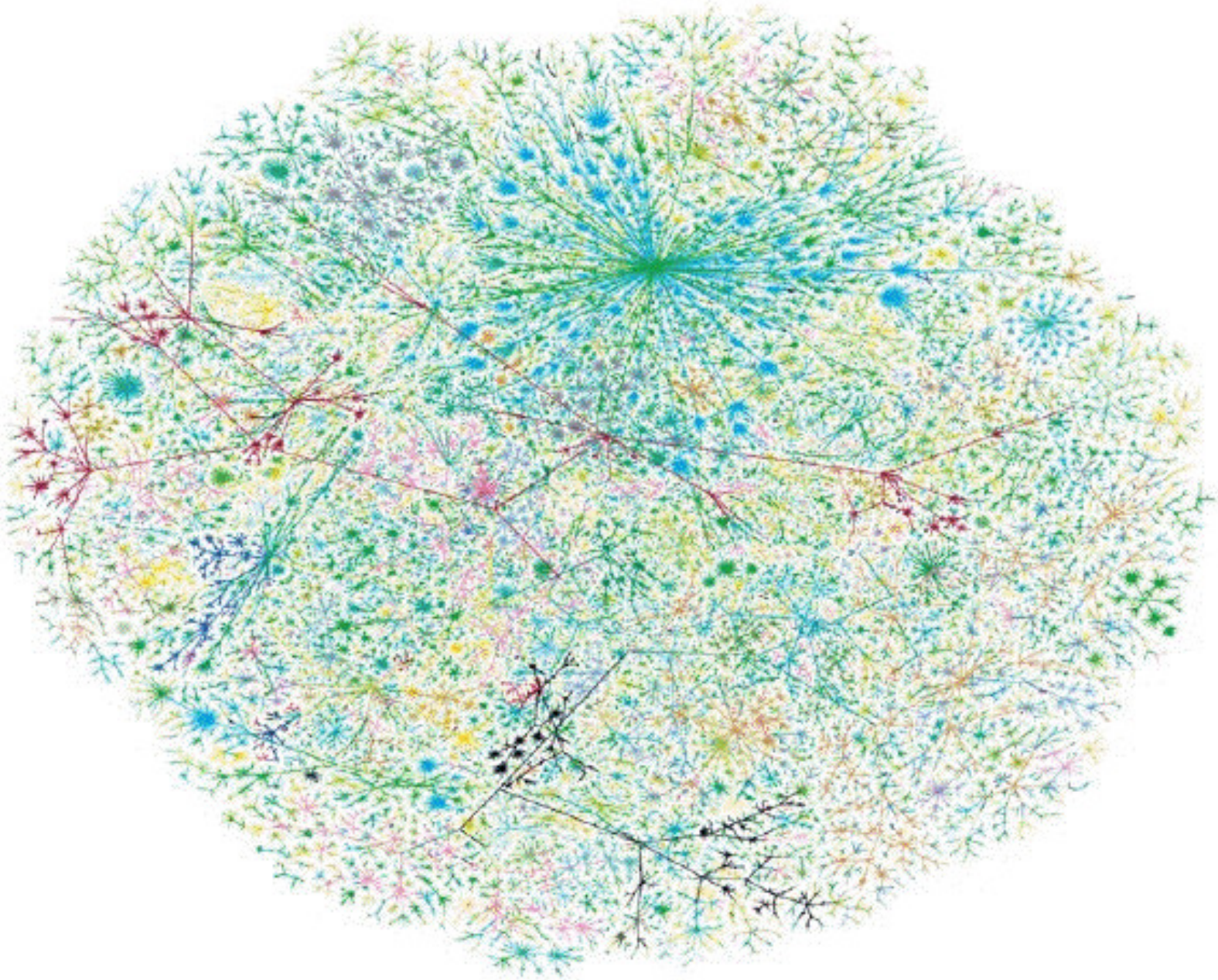
http://techblog.netflix.com/2012/06/netflix-operations-part-i-going.html

The Internet is just a world passing around notes in a classroom. –Jon Stewart

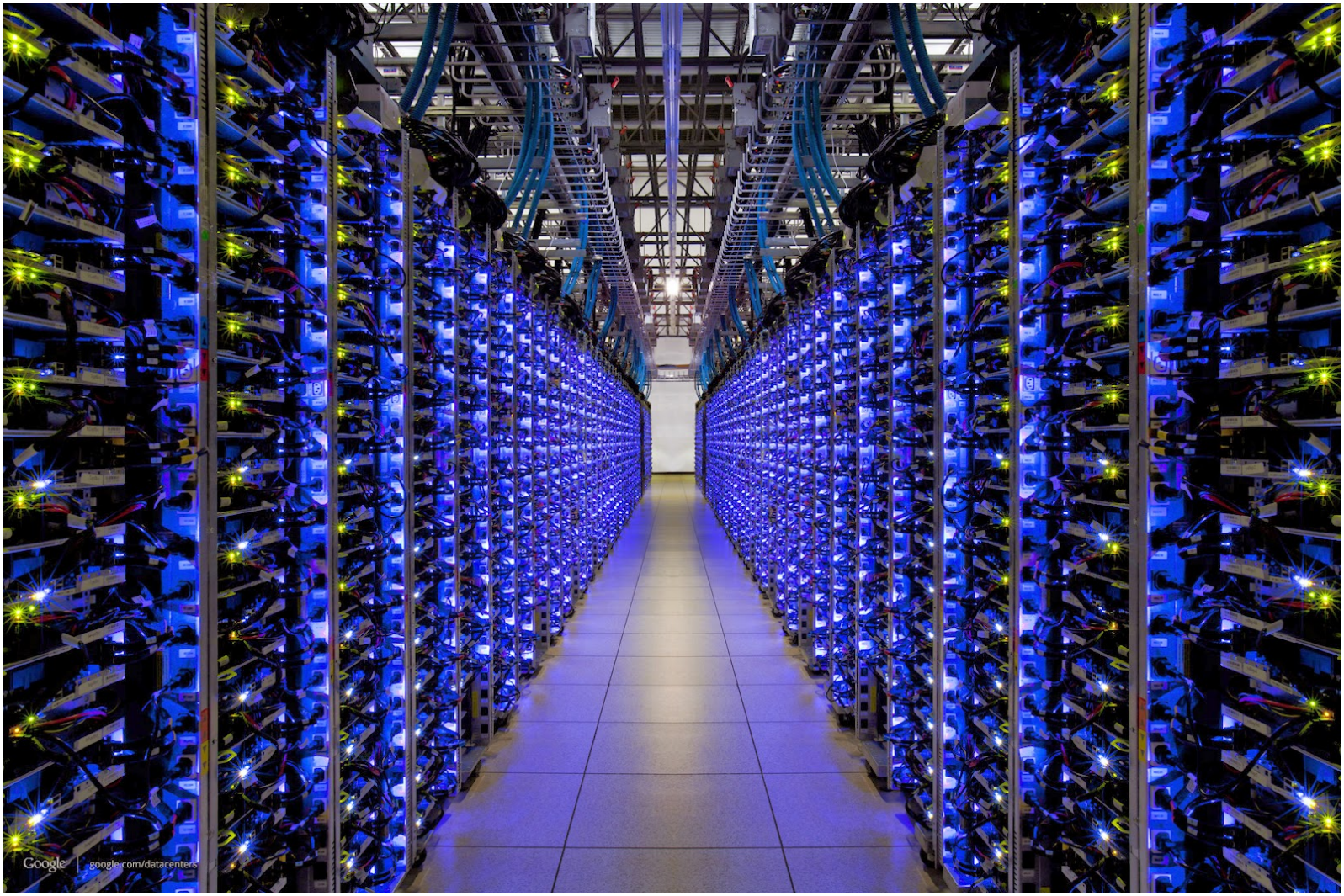
Examples (2)

Google Search I'm Feeling Lucky

Examples (2)



Examples (2)



Examples (3)

- Google search, GMail, Google Docs, et al

Examples (3)

- Google search, GMail, Google Docs, et al
- Amazon

Examples (3)

- Google search, GMail, Google Docs, et al
- Amazon
- Facebook, Twitter, EBay

Examples (3)

- Google search, GMail, Google Docs, et al
- Amazon
- Facebook, Twitter, EBay
- LinkedIn, WhatsApp, Instagram, Snapchat, Uber

Examples (3)

- Google search, GMail, Google Docs, et al
- Amazon
- Facebook, Twitter, EBay
- LinkedIn, WhatsApp, Instagram, Snapchat, Uber
- Hadoop, Hive, Zookeeper, et al

Examples (3)

- Google search, GMail, Google Docs, et al
- Amazon
- Facebook, Twitter, EBay
- LinkedIn, WhatsApp, Instagram, Snapchat, Uber
- Hadoop, Hive, Zookeeper, et al
- No-SQL Databases like HBase, Cassandra, MongoDB, et al

Examples (3)

- Google search, GMail, Google Docs, et al
- Amazon
- Facebook, Twitter, EBay
- LinkedIn, WhatsApp, Instagram, Snapchat, Uber
- Hadoop, Hive, Zookeeper, et al
- No-SQL Databases like HBase, Cassandra, MongoDB, et al
- Distributed file systems: NFS (Network File System), Google File System (GFS), Hadoop Distributed File System (HDFS), Amazon S3, Lustre, GlusterFS, Parallel Virtual File System (PVFS)

Examples (3)

- Google search, GMail, Google Docs, et al
- Amazon
- Facebook, Twitter, EBay
- LinkedIn, WhatsApp, Instagram, Snapchat, Uber
- Hadoop, Hive, Zookeeper, et al
- No-SQL Databases like HBase, Cassandra, MongoDB, et al
- Distributed file systems: NFS (Network File System), Google File System (GFS), Hadoop Distributed File System (HDFS), Amazon S3, Lustre, GlusterFS, Parallel Virtual File System (PVFS)
- P2P: Peer to Peer: Bit Torrent, Gnutella

Examples (3)

- Google search, GMail, Google Docs, et al
- Amazon
- Facebook, Twitter, EBay
- LinkedIn, WhatsApp, Instagram, Snapchat, Uber
- Hadoop, Hive, Zookeeper, et al
- No-SQL Databases like HBase, Cassandra, MongoDB, et al
- Distributed file systems: NFS (Network File System), Google File System (GFS), Hadoop Distributed File System (HDFS), Amazon S3, Lustre, GlusterFS, Parallel Virtual File System (PVFS)
- P2P: Peer to Peer: Bit Torrent, Gnutella
- Domain Name System (DNS)

Examples (3)

- Google search, GMail, Google Docs, et al
- Amazon
- Facebook, Twitter, EBay
- LinkedIn, WhatsApp, Instagram, Snapchat, Uber
- Hadoop, Hive, Zookeeper, et al
- No-SQL Databases like HBase, Cassandra, MongoDB, et al
- Distributed file systems: NFS (Network File System), Google File System (GFS), Hadoop Distributed File System (HDFS), Amazon S3, Lustre, GlusterFS, Parallel Virtual File System (PVFS)
- P2P: Peer to Peer: Bit Torrent, Gnutella
- Domain Name System (DNS)
- Git: distributed version control system.

Examples (3)

- Google search, GMail, Google Docs, et al
- Amazon
- Facebook, Twitter, EBay
- LinkedIn, WhatsApp, Instagram, Snapchat, Uber
- Hadoop, Hive, Zookeeper, et al
- No-SQL Databases like HBase, Cassandra, MongoDB, et al
- Distributed file systems: NFS (Network File System), Google File System (GFS), Hadoop Distributed File System (HDFS), Amazon S3, Lustre, GlusterFS, Parallel Virtual File System (PVFS)
- P2P: Peer to Peer: Bit Torrent, Gnutella
- Domain Name System (DNS)
- Git: distributed version control system.
- SETI: a distributed computing project in which volunteers donate idle computer power to analyze radio signals for signs of extraterrestrial intelligence.

Examples (3)

- Google search, GMail, Google Docs, et al
- Amazon
- Facebook, Twitter, EBay
- LinkedIn, WhatsApp, Instagram, Snapchat, Uber
- Hadoop, Hive, Zookeeper, et al
- No-SQL Databases like HBase, Cassandra, MongoDB, et al
- Distributed file systems: NFS (Network File System), Google File System (GFS), Hadoop Distributed File System (HDFS), Amazon S3, Lustre, GlusterFS, Parallel Virtual File System (PVFS)
- P2P: Peer to Peer: Bit Torrent, Gnutella
- Domain Name System (DNS)
- Git: distributed version control system.
- SETI: a distributed computing project in which volunteers donate idle computer power to analyze radio signals for signs of extraterrestrial intelligence.
- Folding@home (FAH or F@h): a distributed computing project for disease research that simulates protein folding, computational drug design, and other types of molecular dynamics.

Examples (3)

- Google search, GMail, Google Docs, et al
- Amazon
- Facebook, Twitter, EBay
- LinkedIn, WhatsApp, Instagram, Snapchat, Uber
- Hadoop, Hive, Zookeeper, et al
- No-SQL Databases like HBase, Cassandra, MongoDB, et al
- Distributed file systems: NFS (Network File System), Google File System (GFS), Hadoop Distributed File System (HDFS), Amazon S3, Lustre, GlusterFS, Parallel Virtual File System (PVFS)
- P2P: Peer to Peer: Bit Torrent, Gnutella
- Domain Name System (DNS)
- Git: distributed version control system.
- SETI: a distributed computing project in which volunteers donate idle computer power to analyze radio signals for signs of extraterrestrial intelligence.
- Folding@home (FAH or F@h): a distributed computing project for disease research that simulates protein folding, computational drug design, and other types of molecular dynamics.
- Bitcoin: decentralized digital currency

Examples (3)

- Google search, GMail, Google Docs, et al
- Amazon
- Facebook, Twitter, EBay
- LinkedIn, WhatsApp, Instagram, Snapchat, Uber
- Hadoop, Hive, Zookeeper, et al
- No-SQL Databases like HBase, Cassandra, MongoDB, et al
- Distributed file systems: NFS (Network File System), Google File System (GFS), Hadoop Distributed File System (HDFS), Amazon S3, Lustre, GlusterFS, Parallel Virtual File System (PVFS)
- P2P: Peer to Peer: Bit Torrent, Gnutella
- Domain Name System (DNS)
- Git: distributed version control system.
- SETI: a distributed computing project in which volunteers donate idle computer power to analyze radio signals for signs of extraterrestrial intelligence.
- Folding@home (FAH or F@h): a distributed computing project for disease research that simulates protein folding, computational drug design, and other types of molecular dynamics.
- Bitcoin: decentralized digital currency
- Virtually every substantial website!

In-class Exercise

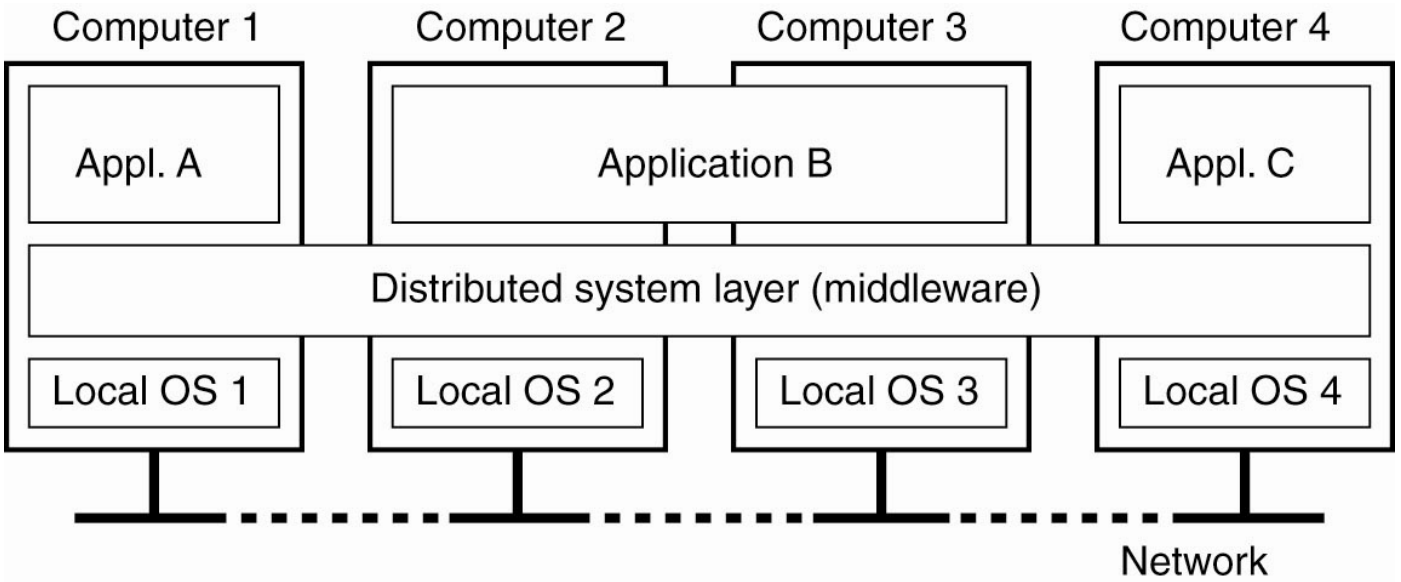
- Walk through architecture of various distributed systems ranging from: single server/client, multiple server/clients to point to point.

How to Implement a Distributed System?

- In order to support a single system view on multiple computers and networks, we need a layer of abstraction implemented in software that is logically placed in the middle of higher layer of users and applications and the lower layer of operating systems and networks. We call this layer the middleware.

How to Implement a Distributed System?

- In order to support a single system view on multiple computers and networks, we need a layer of abstraction implemented in software that is logically placed in the middle of higher layer of users and applications and the lower layer of operating systems and networks. We call this layer the middleware.



Goals of a Distributed System?

- Makes resources accessible

Goals of a Distributed System?

- Makes resources accessible
- Reasonably hides the fact that resources are distributed across a network (*Transparency*)

Goals of a Distributed System?

- Makes resources accessible
- Reasonably hides the fact that resources are distributed across a network (*Transparency*)
- Open

Goals of a Distributed System?

- Makes resources accessible
- Reasonably hides the fact that resources are distributed across a network (*Transparency*)
- Open
- Scalable

Making Resources Accessible

- Benefits

Making Resources Accessible

- Benefits
  - Better economics by sharing expensive resources
  - Easier to collaborate and exchange information
  - Create virtual organizations where geographically dispersed people can work together using groupware
  - Enables electronic commerce
- Problems

Making Resources Accessible

- Benefits
  - Better economics by sharing expensive resources
  - Easier to collaborate and exchange information
  - Create virtual organizations where geographically dispersed people can work together using groupware
  - Enables electronic commerce
- Problems
  - Eavesdropping or intrusion on communication
  - Tracking of communication to build a profile

Transparency

Type	Description
Access	Hide differences in data representation and how a resource is accessed

Transparency

Type	Description
Access	Hide differences in data representation and how a resource is accessed
Location	Hide where a resource is located

Transparency

Type	Description
Access	Hide differences in data representation and how a resource is accessed
Location	Hide where a resource is located
Migration	Hide that a resource may move to another location

Transparency

Type	Description
Access	Hide differences in data representation and how a resource is accessed
Location	Hide where a resource is located
Migration	Hide that a resource may move to another location
Relocation	Hide that a resource may move to another location while in use

Transparency

Type	Description
Access	Hide differences in data representation and how a resource is accessed
Location	Hide where a resource is located
Migration	Hide that a resource may move to another location
Relocation	Hide that a resource may move to another location while in use
Replication	Hide that a resource has multiple copies

Transparency

Type	Description
Access	Hide differences in data representation and how a resource is accessed
Location	Hide where a resource is located
Migration	Hide that a resource may move to another location
Relocation	Hide that a resource may move to another location while in use
Replication	Hide that a resource has multiple copies
Concurrency	Hide that a resource may be shared by several competitive users

Transparency

Type	Description
Access	Hide differences in data representation and how a resource is accessed
Location	Hide where a resource is located
Migration	Hide that a resource may move to another location
Relocation	Hide that a resource may move to another location while in use
Replication	Hide that a resource has multiple copies
Concurrency	Hide that a resource may be shared by several competitive users
Failure	Hide the failure and recovery of a resource

Degree of Transparency

Completely hiding the distribution aspects from users is not always a good idea in a distributed system.

Degree of Transparency

Completely hiding the distribution aspects from users is not always a good idea in a distributed system.

- Attempting to mask a server failure before trying another one may slow down the system

Degree of Transparency

Completely hiding the distribution aspects from users is not always a good idea in a distributed system.

- Attempting to mask a server failure before trying another one may slow down the system
- Expecting several replicas to be always consistent could degrade performance unacceptably

Degree of Transparency

Completely hiding the distribution aspects from users is not always a good idea in a distributed system.

- Attempting to mask a server failure before trying another one may slow down the system
- Expecting several replicas to be always consistent could degrade performance unacceptably
- For mobile and embedded devices, it may be better to expose distribution rather than trying to hide it

Degree of Transparency

Completely hiding the distribution aspects from users is not always a good idea in a distributed system.

- Attempting to mask a server failure before trying another one may slow down the system
- Expecting several replicas to be always consistent could degrade performance unacceptably
- For mobile and embedded devices, it may be better to expose distribution rather than trying to hide it
- Signal transmission is limited by the speed of light as well as the speed of intermediate switches

Openness

An open distributed system offers services according to standard rules that describe the syntax and semantics of those services.

Openness

An open distributed system offers services according to standard rules that describe the syntax and semantics of those services.

- Use of standard protocols.

Openness

An open distributed system offers services according to standard rules that describe the syntax and semantics of those services.

- Use of standard protocols.
- Services are described via interfaces, which are often describe via an Interface Definition Language (IDL). Interfaces only specify syntax so semantics is left to the ambiguities of natural language.

Openness

An open distributed system offers services according to standard rules that describe the syntax and semantics of those services.

- Use of standard protocols.
- Services are described via interfaces, which are often describe via an Interface Definition Language (IDL). Interfaces only specify syntax so semantics is left to the ambiguities of natural language.
- Interoperability, Portability, Extensibility.
- Separating policy from mechanism. For example: *caching* in a web browser.

Scalability

Scalability can be measured against three dimensions.

Scalability

Scalability can be measured against three dimensions.

- Size: be able to easily add more users and resources to a system

Scalability

Scalability can be measured against three dimensions.

- Size: be able to easily add more users and resources to a system
- Geographical: be able to handle users and resources that are far apart

Scalability

Scalability can be measured against three dimensions.

- Size: be able to easily add more users and resources to a system
- Geographical: be able to handle users and resources that are far apart
- Administrative: be able to manage even if it spans independent administrative organizations

Centralized versus distributed implementations.

Centralized Solutions with Scalability Problems

Centralized Solutions with Scalability Problems

- Centralized services.

Centralized Solutions with Scalability Problems

- Centralized services.
- Centralized data.

Centralized Solutions with Scalability Problems

- Centralized services.
- Centralized data.
- Centralized algorithms.

Distributed Approach

Characteristics of decentralized algorithms:

Distributed Approach

Characteristics of decentralized algorithms:

- No machine has complete information about the system state.

Distributed Approach

Characteristics of decentralized algorithms:

- No machine has complete information about the system state.
- Machines make decisions based only on local information.

Distributed Approach

Characteristics of decentralized algorithms:

- No machine has complete information about the system state.
- Machines make decisions based only on local information.
- Failure of one machine does not ruin the algorithm.

Distributed Approach

Characteristics of decentralized algorithms:

- No machine has complete information about the system state.
- Machines make decisions based only on local information.
- Failure of one machine does not ruin the algorithm.
- There is no implicit assumption that a global clock exists.

In-class exercise. Simulate a centralized and a distributed algorithm for the same problem in class!

Scaling Techniques

Scaling Techniques

- Hiding communication latencies: Examples would be asynchronous communication as well as pushing code down to clients (E.g. Javascript)

Scaling Techniques

- Hiding communication latencies: Examples would be asynchronous communication as well as pushing code down to clients (E.g. Javascript)
- Distribution: Taking a component, splitting into smaller parts, and subsequently spreading them across the system. (E.g. Domain Name System)

Scaling Techniques

- Hiding communication latencies: Examples would be asynchronous communication as well as pushing code down to clients (E.g. Javascript)
- Distribution: Taking a component, splitting into smaller parts, and subsequently spreading them across the system. (E.g. Domain Name System)
- Replication: Replicating components increases availability, helps balance the load leading to better performance, helps hide latencies for geographically distributed systems. Caching is a special form of replication.

Distributed System Development Pitfalls

False assumptions made by first time developer *(formulated by Peter Deustch)*.

Distributed System Development Pitfalls

False assumptions made by first time developer *(formulated by Peter Deustch)*.

- The network is reliable

Distributed System Development Pitfalls

False assumptions made by first time developer *(formulated by Peter Deustch)*.

- The network is reliable
- The network is secure

Distributed System Development Pitfalls

False assumptions made by first time developer *(formulated by Peter Deustch)*.

- The network is reliable
- The network is secure
- The network is homogeneous

Distributed System Development Pitfalls

False assumptions made by first time developer *(formulated by Peter Deustch)*.

- The network is reliable
- The network is secure
- The network is homogeneous
- The topology does not change

Distributed System Development Pitfalls

False assumptions made by first time developer *(formulated by Peter Deustch)*.

- The network is reliable
- The network is secure
- The network is homogeneous
- The topology does not change
- Latency is zero

Distributed System Development Pitfalls

False assumptions made by first time developer *(formulated by Peter Deustch)*.

- The network is reliable
- The network is secure
- The network is homogeneous
- The topology does not change
- Latency is zero
- Bandwidth is infinite

Distributed System Development Pitfalls

False assumptions made by first time developer *(formulated by Peter Deustch)*.

- The network is reliable
- The network is secure
- The network is homogeneous
- The topology does not change
- Latency is zero
- Bandwidth is infinite
- Transport cost is zero

Distributed System Development Pitfalls

False assumptions made by first time developer *(formulated by Peter Deustch)*.

- The network is reliable
- The network is secure
- The network is homogeneous
- The topology does not change
- Latency is zero
- Bandwidth is infinite
- Transport cost is zero
- There is one administrator

Distributed System Development Pitfalls

False assumptions made by first time developer *(formulated by Peter Deustch)*.

- The network is reliable
- The network is secure
- The network is homogeneous
- The topology does not change
- Latency is zero
- Bandwidth is infinite
- Transport cost is zero
- There is one administrator

"A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable."
–Leslie Lamport

Types of Distributed Systems

Types of Distributed Systems

- *Distributed Computing Systems*

Types of Distributed Systems

- *Distributed Computing Systems*
  - Cluster Computing Systems
  - Grid Computing Systems

Types of Distributed Systems

- *Distributed Computing Systems*
  - Cluster Computing Systems
  - Grid Computing Systems
- *Distributed Information Systems*

Types of Distributed Systems

- *Distributed Computing Systems*
  - Cluster Computing Systems
  - Grid Computing Systems
- *Distributed Information Systems*
  - Transaction Processing Systems
  - Enterprise Application Integration

Types of Distributed Systems

- *Distributed Computing Systems*
  - Cluster Computing Systems
  - Grid Computing Systems
- *Distributed Information Systems*
  - Transaction Processing Systems
  - Enterprise Application Integration
- *Distributed Pervasive Systems*

Types of Distributed Systems

- *Distributed Computing Systems*
  - Cluster Computing Systems
  - Grid Computing Systems
- *Distributed Information Systems*
  - Transaction Processing Systems
  - Enterprise Application Integration
- *Distributed Pervasive Systems*
  - Home Systems
  - Electronic Health Care Systems
  - Sensor Networks

Types of Distributed Systems

- *Distributed Computing Systems*
  - Cluster Computing Systems
  - Grid Computing Systems
- *Distributed Information Systems*
  - Transaction Processing Systems
  - Enterprise Application Integration
- *Distributed Pervasive Systems*
  - Home Systems
  - Electronic Health Care Systems
  - Sensor Networks

In-class Exercise: Classify the examples we have seen so far into the three categories above.

Cluster Computing

Cluster Computing

- A computer cluster is primarily used to run a single program in parallel on multiple machines. The program relies on parallel libraries and frameworks such as MPI, MapReduce and others.

Cluster Computing

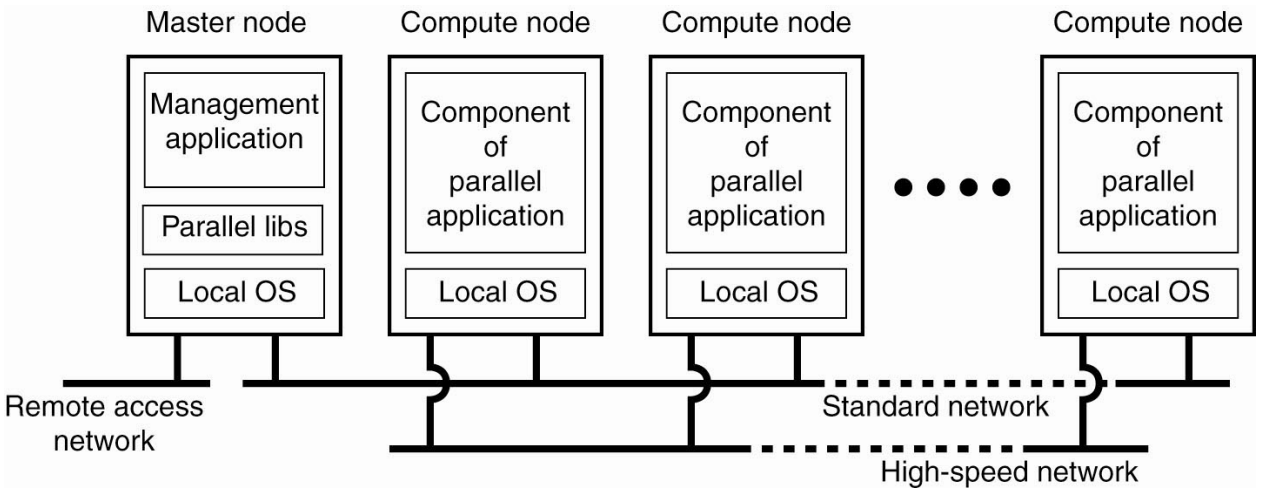
- A computer cluster is primarily used to run a single program in parallel on multiple machines. The program relies on parallel libraries and frameworks such as MPI, MapReduce and others.
- A *computer cluster* consists of a collection of compute and I/O nodes that are controlled and accessed from a single master node. The master allocates nodes to a parallel program, maintains a queue of submitted jobs and provides an interface to manage the cluster.

Cluster Computing

- A computer cluster is primarily used to run a single program in parallel on multiple machines. The program relies on parallel libraries and frameworks such as MPI, MapReduce and others.
- A *computer cluster* consists of a collection of compute and I/O nodes that are controlled and accessed from a single master node. The master allocates nodes to a parallel program, maintains a queue of submitted jobs and provides an interface to manage the cluster.
- A cluster usually has a dedicated high-speed interconnection network and an optional administrative network.

Cluster Computing

- A computer cluster is primarily used to run a single program in parallel on multiple machines. The program relies on parallel libraries and frameworks such as MPI, MapReduce and others.
- A *computer cluster* consists of a collection of compute and I/O nodes that are controlled and accessed from a single master node. The master allocates nodes to a parallel program, maintains a queue of submitted jobs and provides an interface to manage the cluster.
- A cluster usually has a dedicated high-speed interconnection network and an optional administrative network.



Transaction Processing Systems (1)

Transaction primitives:

Transaction Processing Systems (1)

Transaction primitives:

- BEGIN_TRANSACTION

Transaction Processing Systems (1)

Transaction primitives:

- BEGIN_TRANSACTION
- END_TRANSACTION

Transaction Processing Systems (1)

Transaction primitives:

- BEGIN_TRANSACTION
- END_TRANSACTION
- ABORT_TRANSACTION

Transaction Processing Systems (1)

Transaction primitives:

- BEGIN_TRANSACTION
- END_TRANSACTION
- ABORT_TRANSACTION
- READ, WRITE

Transaction Processing Systems (1)

Transaction primitives:

- BEGIN_TRANSACTION
- END_TRANSACTION
- ABORT_TRANSACTION
- READ, WRITE

ACID characteristic properties of transactions:

Transaction Processing Systems (1)

Transaction primitives:

- BEGIN_TRANSACTION
- END_TRANSACTION
- ABORT_TRANSACTION
- READ, WRITE

ACID characteristic properties of transactions:

- Atomic: To the outside world, the transaction happens indivisibly

Transaction Processing Systems (1)

Transaction primitives:

- BEGIN_TRANSACTION
- END_TRANSACTION
- ABORT_TRANSACTION
- READ, WRITE

ACID characteristic properties of transactions:

- Atomic: To the outside world, the transaction happens indivisibly
- Consistent: The transaction does not violate system invariants

Transaction Processing Systems (1)

Transaction primitives:

- BEGIN_TRANSACTION
- END_TRANSACTION
- ABORT_TRANSACTION
- READ, WRITE

ACID characteristic properties of transactions:

- Atomic: To the outside world, the transaction happens indivisibly
- Consistent: The transaction does not violate system invariants
- Isolated: Concurrent transactions do not interfere with each other

Transaction Processing Systems (1)

Transaction primitives:

- BEGIN_TRANSACTION
- END_TRANSACTION
- ABORT_TRANSACTION
- READ, WRITE

ACID characteristic properties of transactions:

- Atomic: To the outside world, the transaction happens indivisibly
- Consistent: The transaction does not violate system invariants
- Isolated: Concurrent transactions do not interfere with each other
- Durable: Once a transaction commits, the changes are permanent

Transaction Processing Systems (1)

Transaction primitives:

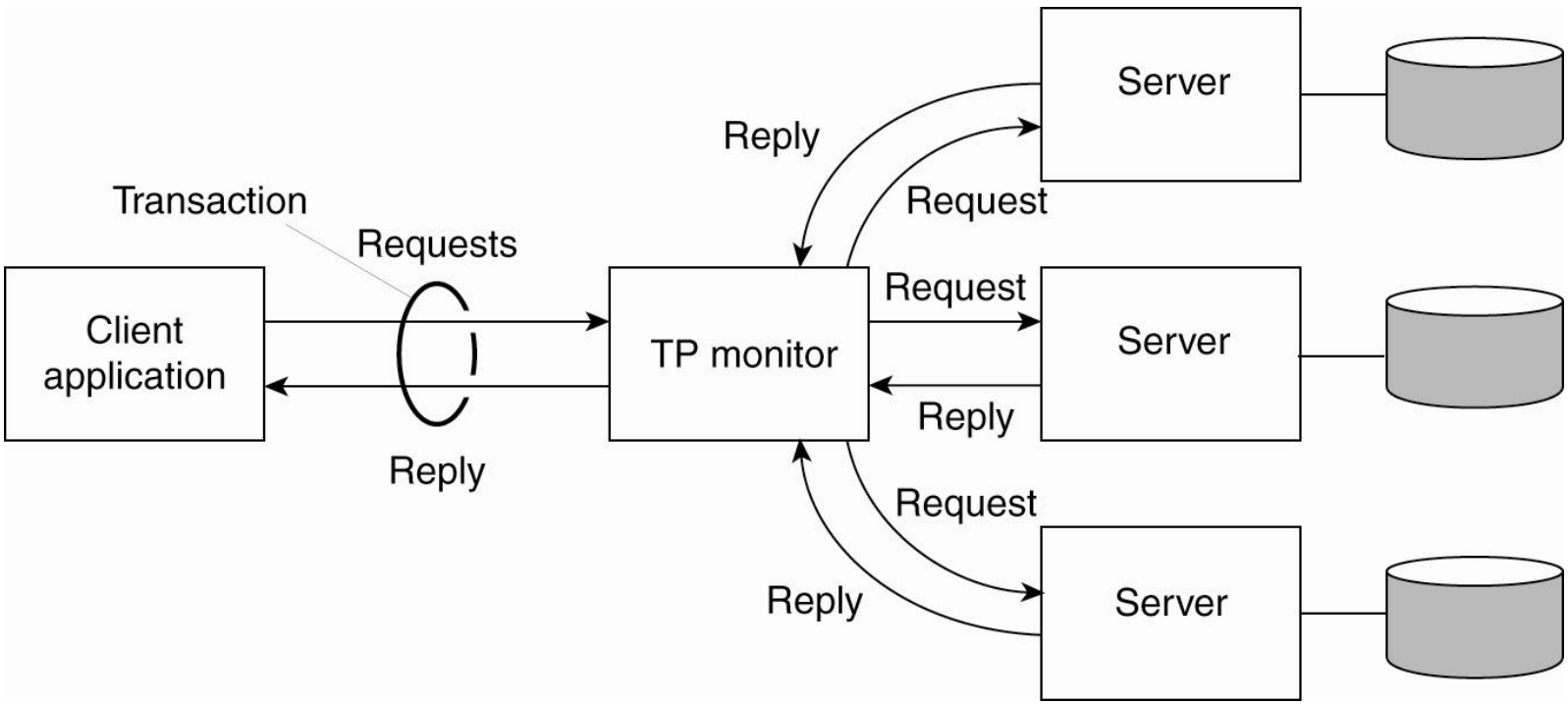
- BEGIN_TRANSACTION
- END_TRANSACTION
- ABORT_TRANSACTION
- READ, WRITE

ACID characteristic properties of transactions:

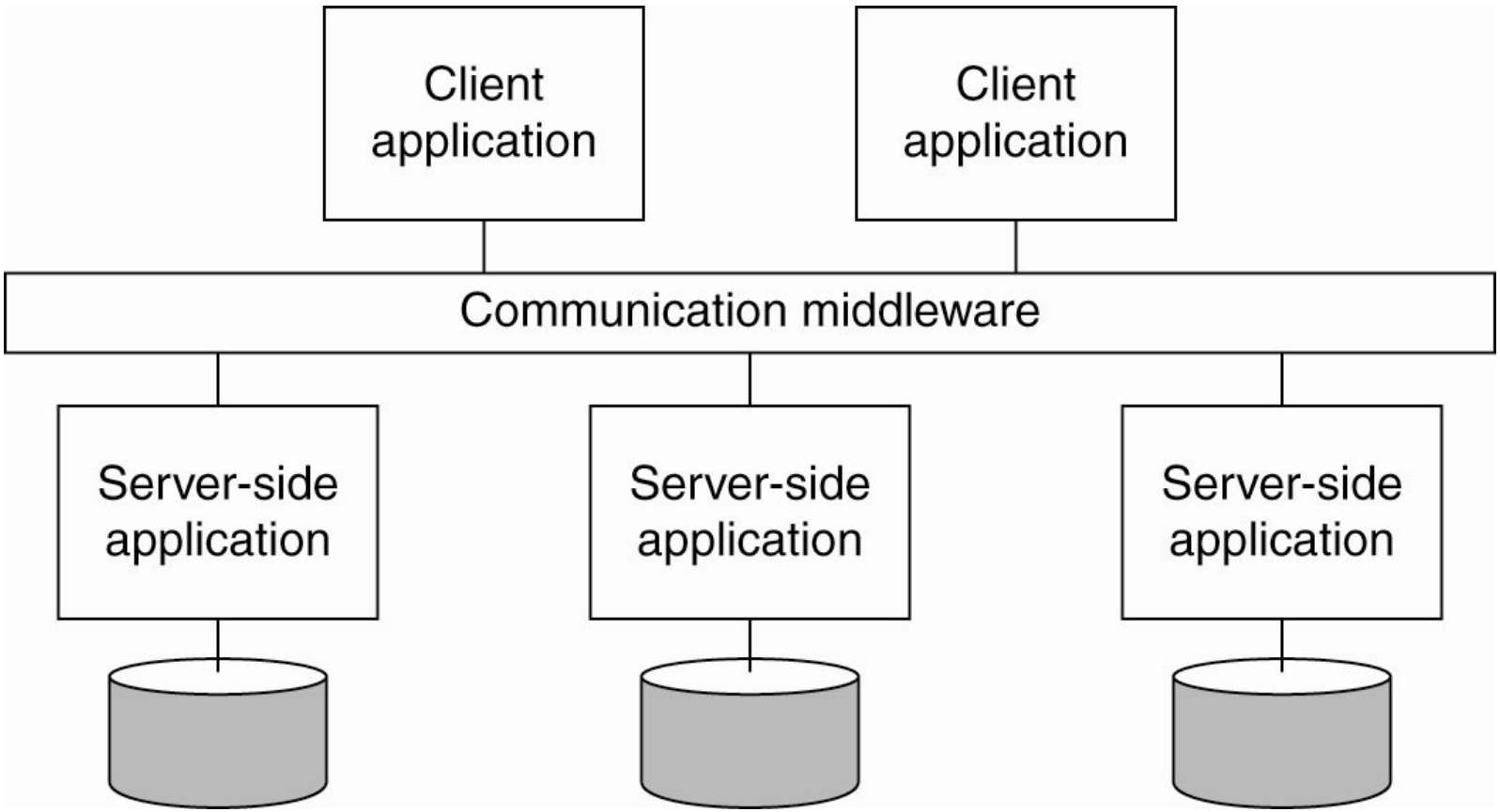
- Atomic: To the outside world, the transaction happens indivisibly
- Consistent: The transaction does not violate system invariants
- Isolated: Concurrent transactions do not interfere with each other
- Durable: Once a transaction commits, the changes are permanent

Transactions can be nested. Durability applies to top-level transactions only. For example: an airline and a hotel database.

Transaction Processing Systems (2)



Enterprise Application Integration



Middleware as a communication facilitator for enterprise application integration.

Distributed Pervasive Systems

Requirements for pervasive systems:

- Embrace contextual changes.
- Encourage ad hoc composition.
- Recognize sharing as the default.

Examples: Home systems, Body Area Networks, Sensor Networks.

Chapter 1: Recommended Exercises

- **Problem 1**. What is the role of middleware in a distributed system?
- **Problem 2**. Explain what is meant by transparency, and give examples of different types of transparency.
- **Problem 3**. Scalability can be achieved by applying different techniques. What are these techniques?
- **Problem 4**. Give further examples of distributed pervasive systems.