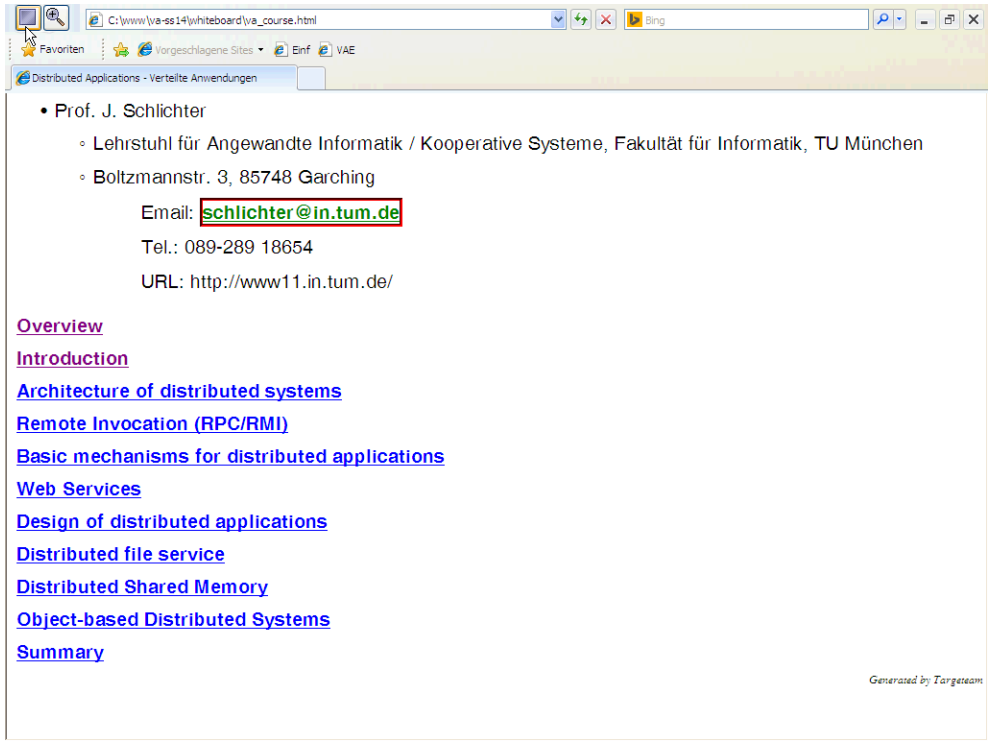
Script **generated by TTT**

Title: Distributed_Applications (15.04.2014)

Date: Tue Apr 15 14:40:04 CEST 2014

Duration: 79:10 min

Pages: 14



Architecture of distributed systems

Issues

This section focuses on the following issues

Discussion of basic aspects of distributed systems.

Transparency as a key concept of distributed systems.

How do distributed components cooperate? Thus, we discuss models of cooperation among components of distributed applications.

What is the client-server model?

System Models

Transparency

Paradigms for distributed applications

Client-server model

Software layers

applications , services
middleware
operating system
computer and network devices

Middleware

is defined as a layer of software whose purpose is to mask heterogeneity and to provide a convenient programming model to application programmers.

hides the complexity of the communication between two or more systems or services.

major categories of middleware: distributed objects, distributed components, publish-subscribe, web service, peer-to-peer.

Examples are Corba, Java RMI, DCOM (Microsoft's Distributed Component Object Model).

Middleware services are e.g.: communication facilities, naming of remote entities (objects), persistence (distributed file system), distributed transactions, facilities for security.

A distributed system can be described in form of descriptive models.

Architectural model

defines the interaction between components and the mapping onto the underlying network.

Software layers

System architectures

Interaction model

Failure model

Security model

The following sections of the course will discuss in more detail various aspects of these system models.

System Models

A distributed system can be described in form of descriptive models.

Architectural model

defines the interaction between components and the mapping onto the underlying network.

Software layers

System architectures

Interaction model

Failure model

Security model

The following sections of the course will discuss in more detail various aspects of these system models.

Generated by Targeteam

Failure model

The failure model defines the ways in which failures may occur and how they are handled.

distinction between failures of processes and communication channels.

different types of failures:

crash faults: the process simply stops due to Hardware failures or Software errors.

message loss: messages may be lost due to buffer overflow of routers or network congestion.

fail stop failures: the process fails by crashing; system notifies relevant partners.

timing failures: a local clock exceeds the bounds on its rate of drift from real time or transmission takes longer than the specified bound.

arbitrary failures (non-malicious Byzantine failure): a process arbitrarily omits intended processing steps, takes unintended processing steps or sends corrupted messages.

malicious Byzantine failure: an attacker who has studied the system attempts to break it. Examples are the corruption or replay of messages, or the modification of the program (install hacked version).

System Models

A distributed system can be described in form of descriptive models.

Architectural model

defines the interaction between components and the mapping onto the underlying network.

Software layers

System architectures

Interaction model

Failure model

Security model

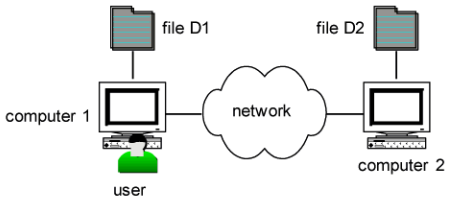
The following sections of the course will discuss in more detail various aspects of these system models.

Generated by Targeteam

Access transparency

Problem : How to access objects in a distributed system.

⇒ Access transparency provides access to local and remote objects in exactly the same way.



Migration transparency

Problem : Object relocation in distributed systems.

⇒ Objects may migrate from one computer to another without influencing the correct behavior of running applications.

Host migration transparency

Problem : Computer migrates from one subnetwork to another subnetwork, e.g. if a user connects his laptop computer to different subnetworks. This requires a dynamic assignment of the IP address (e.g. DHCP), a name server, etc.

⇒ The computer supports the same environment, the same applications, and the same look-and-feel, no matter where the mobile workers are currently connected to the network.

Types of migration

off-line migration.

on-line migration.

Transparency

key concept for better exploitation of resources within a distributed, heterogeneous system.

Location transparency
Access transparency
Replication transparency
Migration transparency
Language transparency
Other transparencies
Goal for distributed applications

Other transparencies

There are a number of other transparencies relevant for distributed systems.

Failure transparency

Problem : Partial failure in distributed systems, for example computer crashes or network failures.

⇒ Up to a certain degree, failures are masked by the system.

Concurrency transparency

concurrent access to shared resources by distributed users or application components.

Problem : shared access to objects in distributed systems.

⇒ Several users or application programs can access objects simultaneously (for example shared data) without mutual influence.

Execution transparency

Execution transparency implies that processes may be processed on different runtime systems.

Performance transparency

allows for dynamic reconfiguration of the system to improve the overall system performance when changes in load characteristics are detected.

Scalability transparency

supports extensions and enhancements of the system or the applications without the need of modifications to the system structure or changes to the application algorithms.

key concept for better exploitation of resources within a distributed, heterogeneous system.

Location transparency

Access transparency

Replication transparency

Migration transparency

Language transparency

Other transparencies

Goal for distributed applications

Transparency

key concept for better exploitation of resources within a distributed, heterogeneous system.

Location transparency

Access transparency

Replication transparency

Migration transparency

Language transparency

Other transparencies

Goal for distributed applications

Generated by Targeteam

Paradigms for distributed applications

Information Sharing

Message exchange

Naming entities

Bidirectional communication

Producer-consumer interaction

Client-server model

Peer-to-peer model

Group model

Publish-Subscribe model

Taxonomy of communication

- Message serialization

Levels of Abstraction