Middleware and Distributed Systems

Introduction

Dr. Martin v. Löwis

What is Middleware ?

- Bauer et al. Software Engineering, Report on a conference sponsored by the NATO SCIENCE COMMITTEE Garmisch, Germany, 7th to 11th October 1968

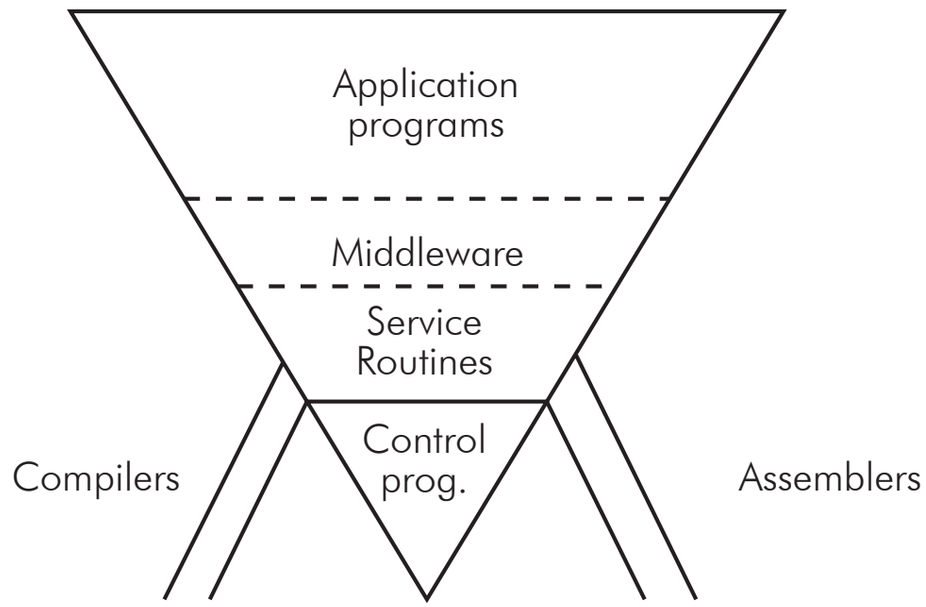


Figure 3. d'Agapeyeff's Inverted Pyramid

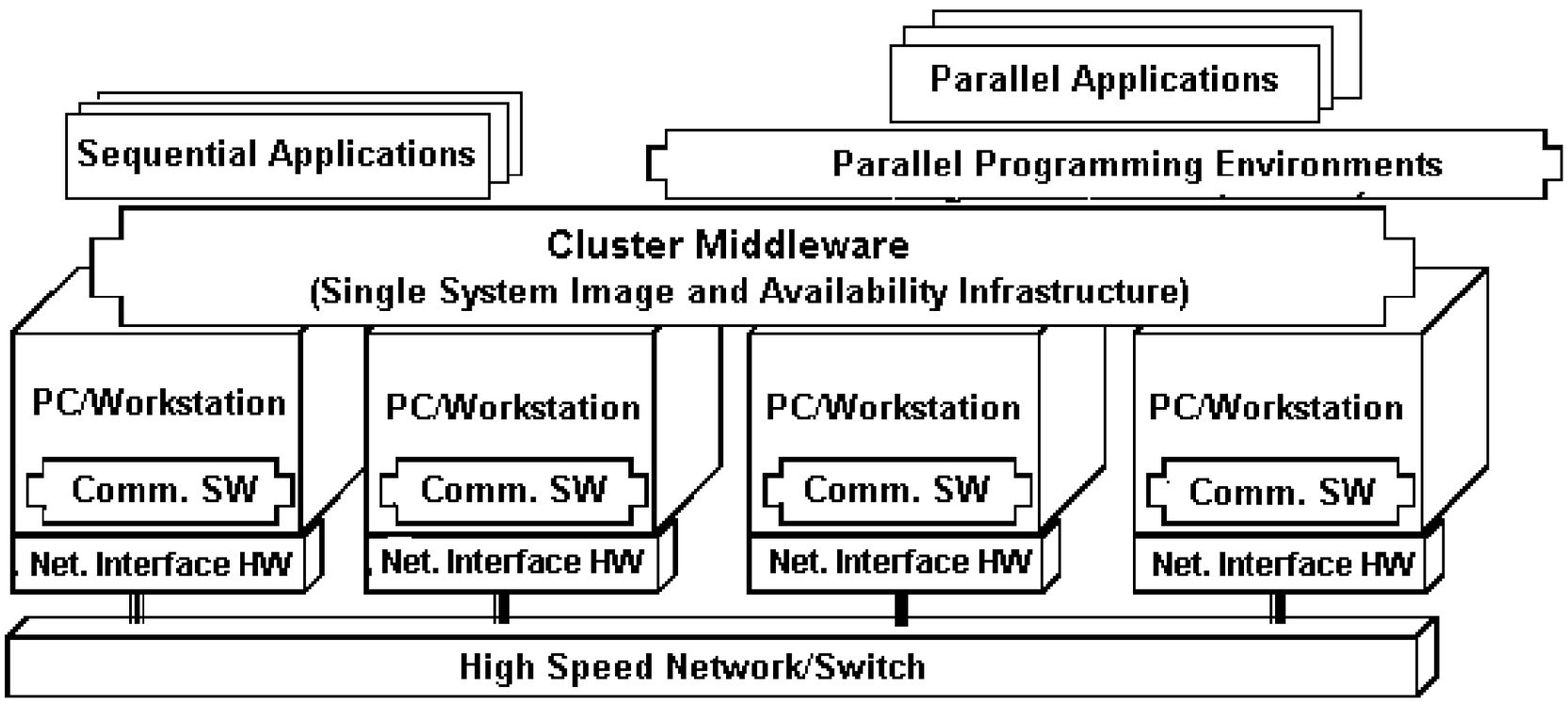
- *d'Agapeyeff*: "The point about this pyramid is that it is terribly sensitive to change in the underlying software such that the new version does not contain the old as a subset. It becomes very expensive to maintain these systems and to extend them while keeping them live."

What is Middleware? (2)

- Things that have been called "middleware" in the past:
  - Implementations of RPC protocols (DCOM, IONA Orbix)
  - Messaging Systems (MPI, MQSeries, Tibco Rendezvous)
  - Database systems (Oracle, PostgreSQL)
  - Run-time systems for programming languages (JVM, .NET Framework)
  - Application Servers (IBM WebSphere, IIS)
  - Transaction Processing Monitors (BEA Tuxedo, Microsoft MTS)
  - Specifications describing these systems
- Focus of this lecture: middleware for communications

Distributed Systems

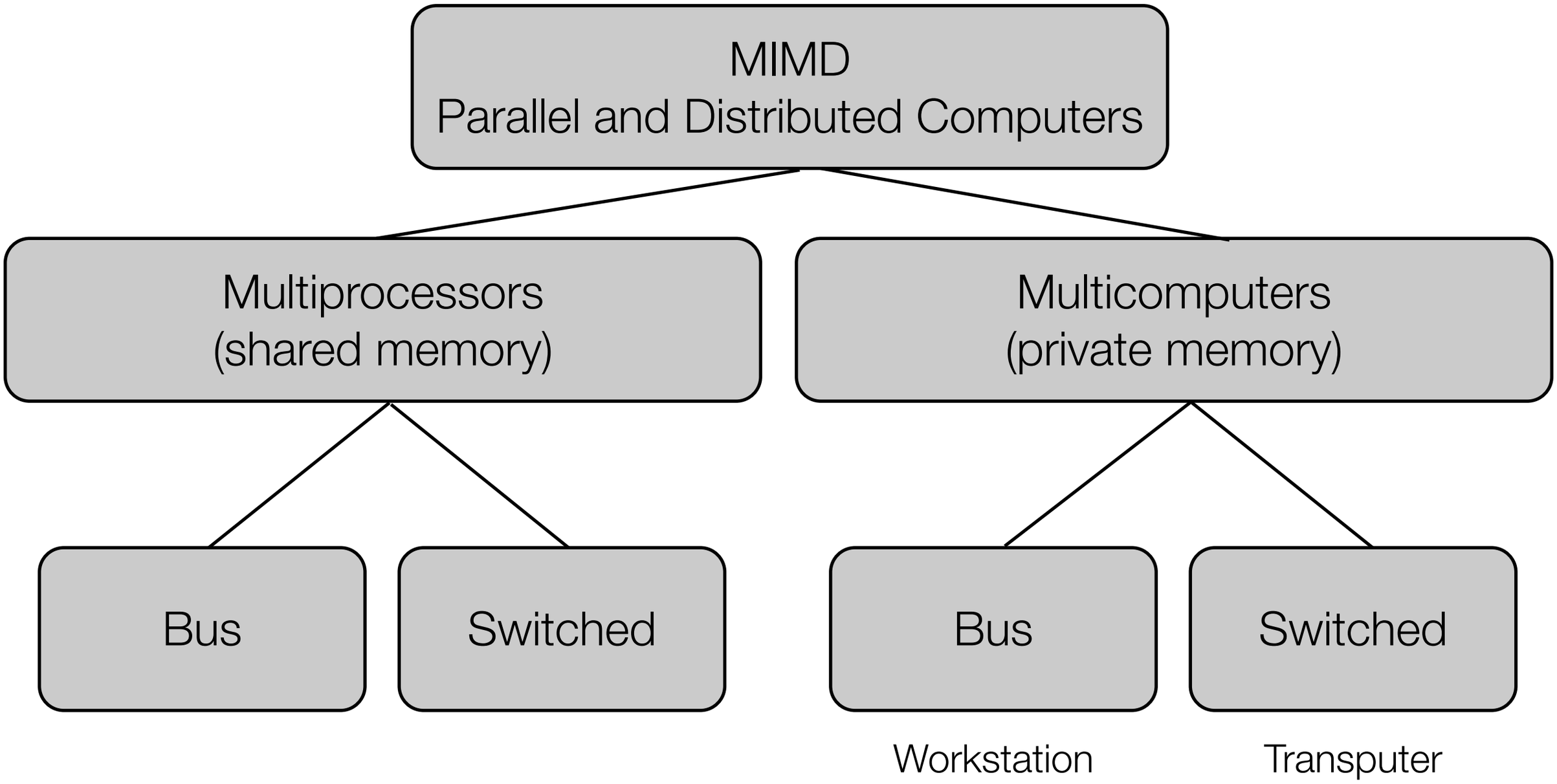
"A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable."



Parallel vs. Distributed Systems

- Flynn 1966: SISD, MISD, SIMD, MIMD
  - MIMD is not distributed computing
  - Network speed
  - Multicomputer concept
- Data exchange for SIMD and MIMD
  - Shared memory, message passing
  - Interconnection network design

Taxonomy (Tanenbaum)

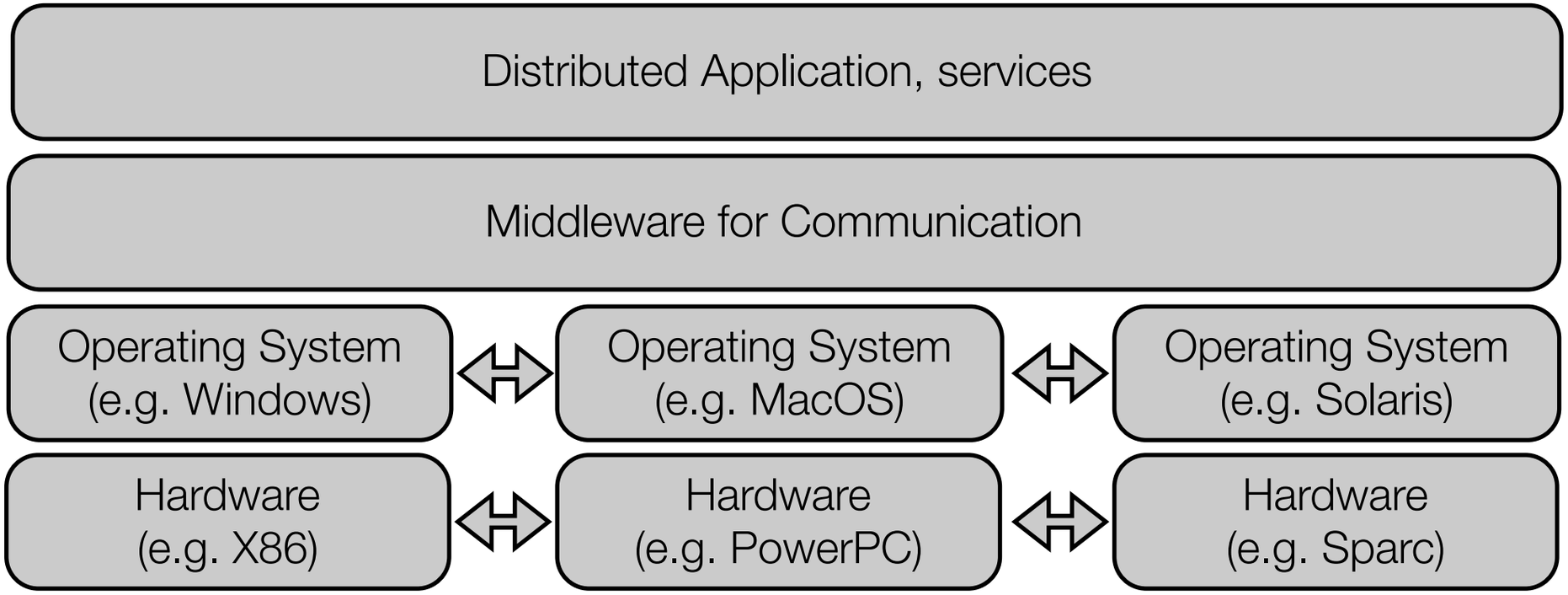


Distributed System: Definitions

- Tanenbaum (Distributed Operating Systems): A distributed system is a collection of independent computers that appear to the users of the system as a single computer.
  - example: network of workstations where a command started on one machine may arbitrarily run on any machine
- Coulouris et al.: [system] in which hardware or software components located at networked computers communicate and coordinate their actions only by passing messages.
  - Consequences: concurrency, no global clock, independent failures
  - Challenges: heterogeneity, openness, security, scalability, failure handling, concurrency, need for transparency

Our View of Distributed System

- Distributed system: Utilize ‘services’ on other nodes
  - Number of independent computing nodes, connected through network
  - Computing node
    - One or multiple processors with one or multiple layers of memory
    - Number of external devices
    - Distributed system can contain heterogeneous resources
- Parallel systems: Compute one parallel application
  - Typically communication through shared memory


Distributed Application, services
Middleware for Communication
Operating System (e.g. Windows)
Operating System (e.g. MacOS)
Operating System (e.g. Solaris)
Hardware (e.g. X86)
Hardware (e.g. PowerPC)
Hardware (e.g. Sparc)

Distributed application

- Number of programs or program parts, which are executed concurrently on one or more nodes
  - Different names: Tasks (Ada), Processes (Hermes, NIL), Objects (Corba)
  - Historically names as ‘processes’
- Additional problems in contrast to sequential applications
  - Synchronization between processes
  - Communication between processes
  - Consideration of partial failures
- Middleware supports distributed application development
  - Inter-process communication and synchronization mechanisms

Lecture Objective

- Focus on middleware in distributed systems
- Ability to develop own middleware components
- Deep understanding of principles underlying distributed systems
- Lecture metadata
  - 4 SWS, 6 credit points, registration until 13.11.2009
  - Oral exam, 4 successful exercise presentations as precondition

Lab Exercise

- Development of an own middleware stack
- Given wire format and interface definition language
- New exercises every three weeks on Mondays
  - First exercise in the next lecture
  - To be presented by all groups in the lecture (12.11., 10.12., 14.1., 4.2.)
  - 1-2 students per group

Topics

- Inter-process communication
  - Interface description, remote procedure call, message-oriented middleware, stream communication
  - OMG IDL, ASN.1, WSDL, ..., IIOP, XDR, ...
- Design patterns and algorithms in distributed systems
  - Broker, Proxy, Adapter, ..., Life Cycle, Identification and State, ...
- Naming services and registries
- Distributed systems architecture models
- Adaptive and real-time middleware

Topics (contd.)

- Security in distributed systems
  - Kerberos, PKI
- Clock synchronization and coordination of distributed activities
- P2P systems
- Transactions and distributed shared memory
- Cluster and grid computing

Topics not discussed

- Language Mappings
- Distributed File Systems
- Distributed Operating Systems
- Mobile Computing

TWP - The Wire Protocol

- Simple messaging protocol
- Support for few data types: integers, strings, binary, structs, arrays
- Extensible
- Protocols defined in TDL
- Beispiel:

```
protocol Echo = id 2 {
    message Request = 0 {
        string text;
    }

    message Reply = 1 {
        string text;
        int number_of_letters;
    }
}
```