technische universität
dortmund

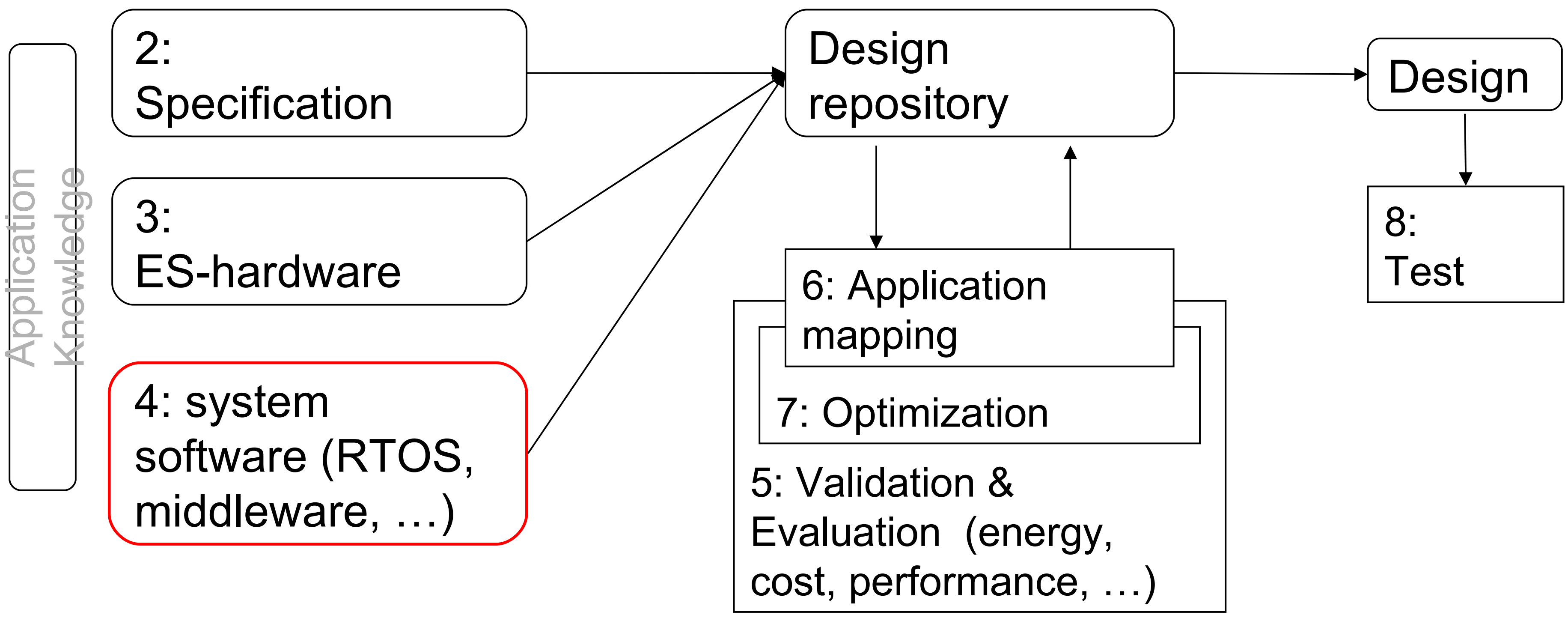
fakultät für informatik
informatik 12

Middleware

Peter Marwedel
TU Dortmund,
Informatik 12
Germany

2009/11/27

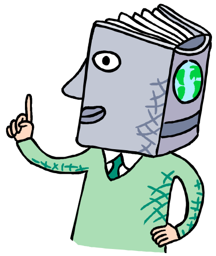
Structure of this course



Numbers denote sequence of chapters

Reuse of standard software components

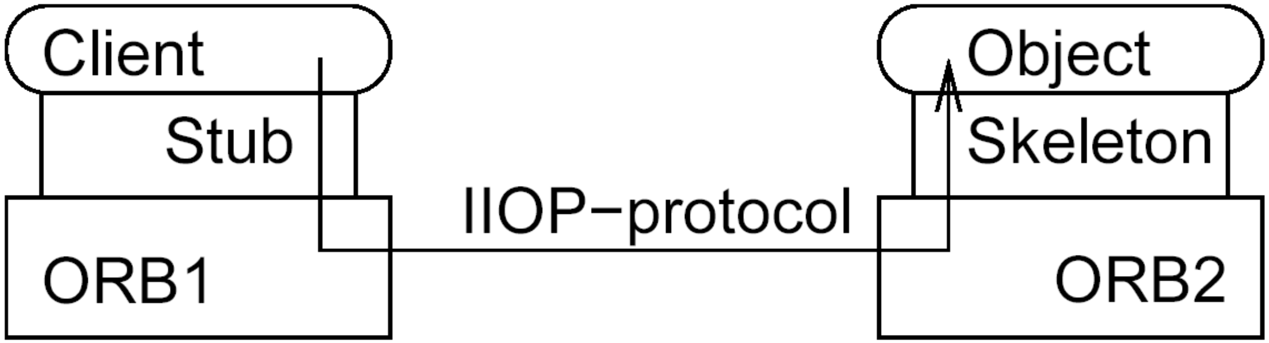
Knowledge from previous designs to be made available in the form of **intellectual property** (IP, for SW & HW).



- Operating systems
- Middleware
- ….

Impact of Priority Inversion on Access Methods for Remote Objects

Software packages for access to remote objects;
Example:
CORBA (Common Object Request Broker Architecture).
Information sent to Object Request Broker (ORB) via local stub.
ORB determines location to be accessed and sends information via the IIOP I/O protocol.



Access times not predictable.

Real-time (RT-) CORBA

A very essential feature of RT-CORBA is to provide

- *end-to-end predictability of timeliness in a fixed priority system*.
- This involves *respecting thread priorities between client and server for resolving resource contention*,
- and bounding the latencies of operation invocations.
- Thread priorities might not be respected when threads obtain mutually exclusive access to resources (priority inversion).
- RT-CORBA includes provisions for bounding the time during which such priority inversion can happen.

Real-time CORBA
- Thread priority management -

- RT-CORBA includes facilities for thread priority management.
- Priority independent of the priorities of the underlying OS, even though it is compatible with the RT-extensions of the POSIX standard for OSs [Harbour, 1993].
- The thread priority of clients can be propagated to the server side.
- Priority management for primitives for mutually exclusive access to resources. Priority inheritance protocol must be available in implementations of RT-CORBA.
- Pools of preexisting threads avoid the overhead of thread creation and thread-construction.

Message passing interface (MPI)

- Library designed for high-performance computing (hpc)
- Based on asynchronous/synchronous message passing
- Comprehensive, popular library
- Available on a variety of platforms
- Considered also for multiple processor system-on-a-chip (MPSoC) programming for embedded systems;
- MPI includes many copy operations to memory ☹ (memory speed ~ communication speed for MPSoCs); Appropriate MPSoC programming tools missing.
- Mostly for homogeneous multiprocessing

http://www.mhpcc.edu/training/workshop/mpi/MAIN.html#Getting_Started

MPI (1)

Sample blocking library call (for C)**:**

- MPI_Send(*buffer,count,type,dest,tag,comm*) where
  - *buffer*: Address of data to be sent
  - *count*: number of data elements to be sent
  - *type*: data type of data to be sent (e.g. MPI_CHAR, MPI_SHORT, MPI_INT, …)
  - *dest*: process id of target process
  - *tag*: message id (for sorting incoming messages)
  - *comm*: communication context = set of processes for which destination field is valid
  - function result indicates success

http://www.mhpcc.edu/training/workshop/mpi/MAIN.html#Getting_Started

MPI (2)

Sample non-blocking library call (for C):

- MPI_Isend(*buffer,count,type,dest,tag,comm,request*) where
  - *buffer … comm:* same as above
  - *request*: the system issues a unique "request number". The programmer uses this system assigned "handle" later (in a WAIT type routine) to determine completion of the non-blocking operation.

http://www.mhpcc.edu/training/workshop/mpi/MAIN.html#Getting_Started

RT-issues for MPI

- MPI/RT: a real-time version of MPI [MPI/RT forum, 2001].
- MPI-RT does not cover issues such as thread creation and termination.
- MPI/RT is conceived as a potential layer between the operating system and standard (non real-time) MPI.

Evaluation

Explicit

- Computation partitioning
- Communication
- Data distribution

Implicit

- Synchronization (implied by communic., explicit possible)
- Expression of parallelism (implied)
- Communication mapping

Properties

- Most things are explicit
- Lots of work for the user ("*assembly lang. for parallel prog*.")
- doesn't scale well when # of processors is changed heavily

Based on Wilfried Verachtert (IMEC): *Introduction to Parallelism,* tutorial, DATE 2008

Ptheads

- Shared memory model
  - Completely explicit synchronization
  - Originally used for single processor
  - Exact semantics depends on the memory consistency model
  - Synchronization is very hard to program correctly
- Consists of standard API
  - Locks (mutex, read-write locks)
  - Condition variables
  - Typically supported by a mixture of hardware (shared memory) and software (thread management)
- Support for efficient producer/consumer parallelism relies on murky parts of the model
- Pthreads can be used as back-end for other programming models (e.g. OpenMP)

Based on Wilfried Verachtert (IMEC): *Introduction to Parallelism,* tutorial, DATE 2008

PThreads Example

```
threads = (pthread_t *) malloc(n*sizeof(pthread_t));
pthread_attr_init(&pthread_custom_attr);

for (i=0;i<n; i++)
  pthread_create(&threads[i], &pthread_custom_attr, task, …)

for (i=0;i<n; i++) {
  pthread_mutex_lock(&mutex);
  <receive message>
  pthread_mutex_unlock(&mutex);
}

for (i=0;i<n; i++)
  pthread_join(threads[i], NULL)
```

```
void* task(void *arg) {
  …
  pthread_mutex_lock(&mutex);
  <send message>
  pthread_mutex_unlock(&mutex);
  return NULL
}
```


Based on Wilfried Verachtert (IMEC): *Introduction to Parallelism,* tutorial, DATE 2008

OpenMP

Explicit

- Expression of parallelism (mostly explicit)

Implicit

- Computation partitioning
- Communication
- Synchronization
- Data distribution

Parallelism expressed using pragmas

- Parallel loops (essentially data parallelism)
- Parallel sections
- Reductions

Implementations target shared memory hardware

Lack of control over partitioning can cause problems

Based on Wilfried Verachtert (IMEC): *Introduction to Parallelism,* tutorial, DATE 2008

Universal Plug-and-Play (UPnP)

- Extension of the plug-and-play concept
- Goal: *Enable the emergence of easily connected devices and to simplify the implementation of networks in the home and corporate environments*
- Examples: Discover printers, storage space easily, control switches in homes and offices
- Exchanging data, no code (reduces security hazards)
- Agreement on data formats and protocols
- Classes of predefined devices (printer, mediaserver etc.)
- http://upnp.org

Devices Profile for Web Services (DPWS)

- More general than UPnP
- *The **Devices Profile for Web Services** (DPWS) defines a minimal set of implementation constraints to enable secure Web Service messaging, discovery, description, and eventing on resource-constrained devices. …*
- *DPWS specifies a set of built-in services:*
  - *Discovery services: used by a device connected to a network to advertise itself and to discover other devices.*
  - *Metadata exchange services: provide dynamic access to a device's hosted services and to their metadata.*
  - *Publish/subscribe eventing services: allowing other devices to subscribe to asynchronous event messages*
- *Lightweight protocol, supporting dynamic discovery, … its application to automation environments is clear.*

Network Communication Protocols - e.g. JXTA -

- *Open source peer-to-peer protocol specification.*
- *Defined as a set of XML messages that allow any device connected to a network to exchange messages and collaborate independently of the network topology.*
- *Designed to allow a range of devices to communicate. Can be implemented in any modern computer language.*
- *JXTA peers create a virtual overlay network, allowing a peer to interact with other peers even when some of the peers and resources are behind firewalls and NATs or use different network transports. Each resource is identified by a unique ID, so that a peer can change its localization address while keeping a constant identification number.*

http://en.wikipedia.org/wiki/JXTA

Real-time data bases (1)

Goal: store and retrieve persistent information
Transaction= sequence of read and write operations
Changes not final until they are committed
Requested ("ACID") properties of transactions

1. **Atomic:** state information as if transaction is either completed or had no effect at all.
2. **Consistent:** Set of values retrieved from several accesses to the data base must be possible in the world modeled.
3. **Isolation:** No user should see intermediate states of transactions
4. **Durability:** results of transactions should be persistent.

Real-time data bases (2)

Problems with implementing real-time data bases:

1. transactions may be aborted various times before they are finally committed.
2. For hard discs, the access times to discs are hardly predictable.

Possible solutions:

1. Main memory data bases
2. Relax ACID requirements

Summary

- Communication middleware
  - CORBA
  - MPI
  - Pthreads
  - OpenMP
  - JXTA
  - DPWS