An MPSoC middleware: Network transparency for on-chip/off-chip modules

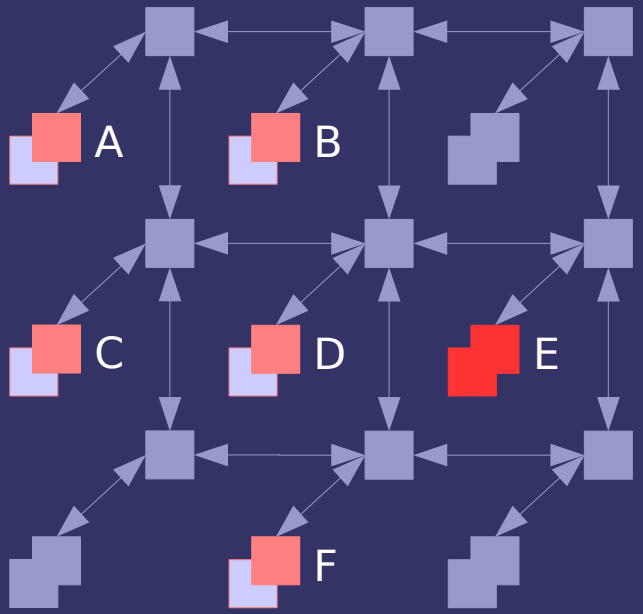
Computer Architecture & Networks Group

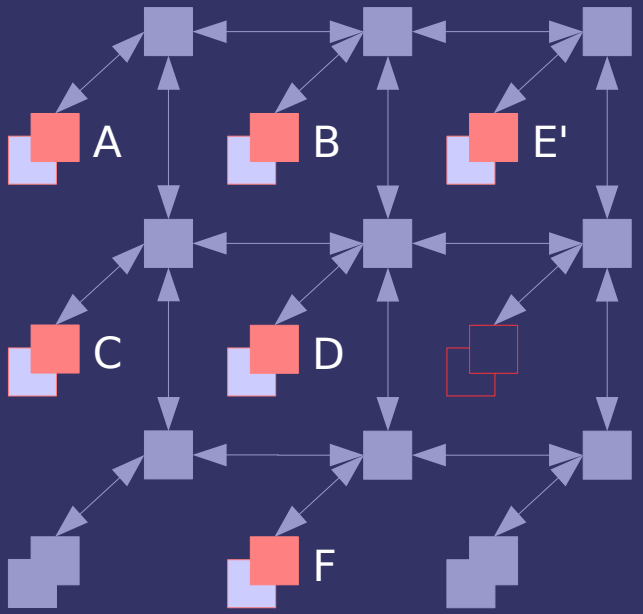
University of Castilla-La Mancha

Francisco Moya <francisco.moya@uclm.es>

Goals (1)

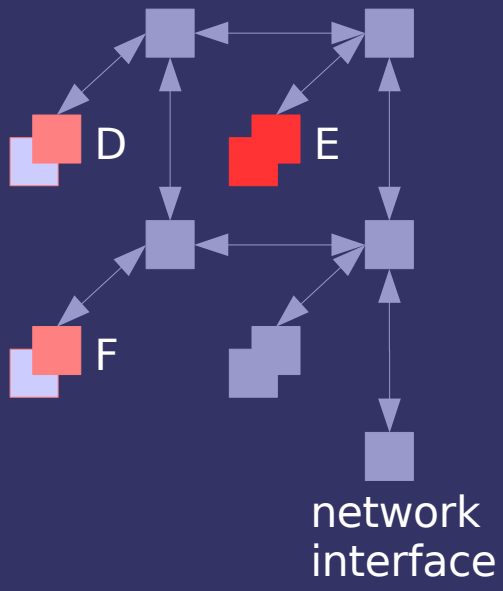
- Ability to replace faulty or buggy components

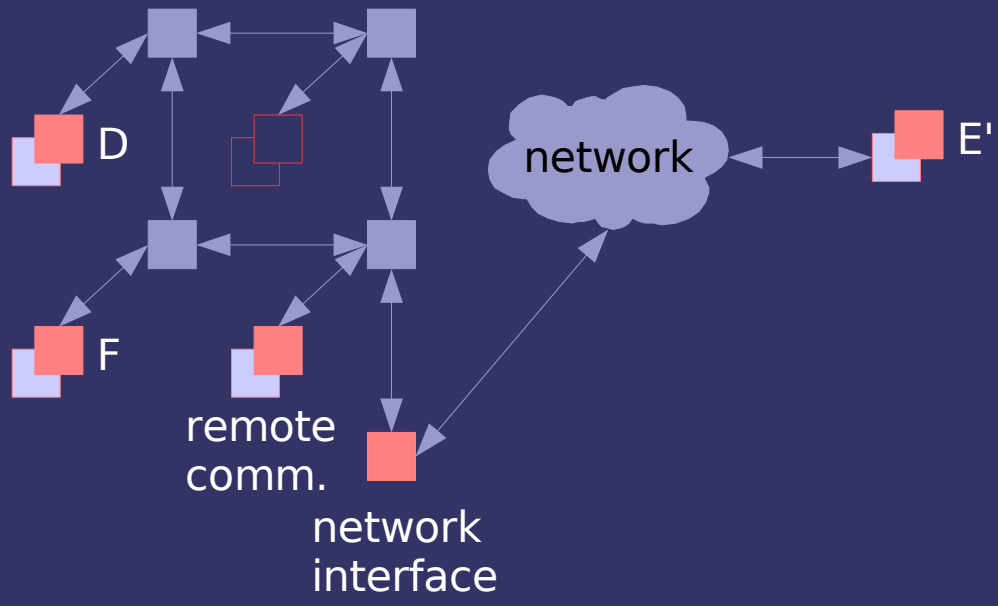




Goals (2)

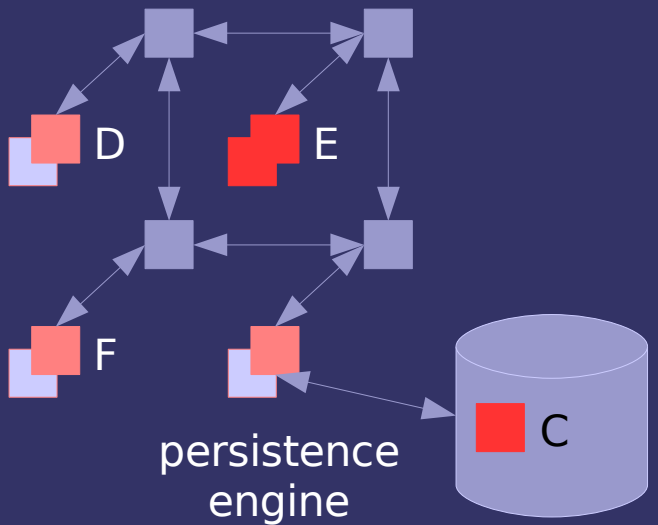
- Ability to replace on-chip components using off-chip components

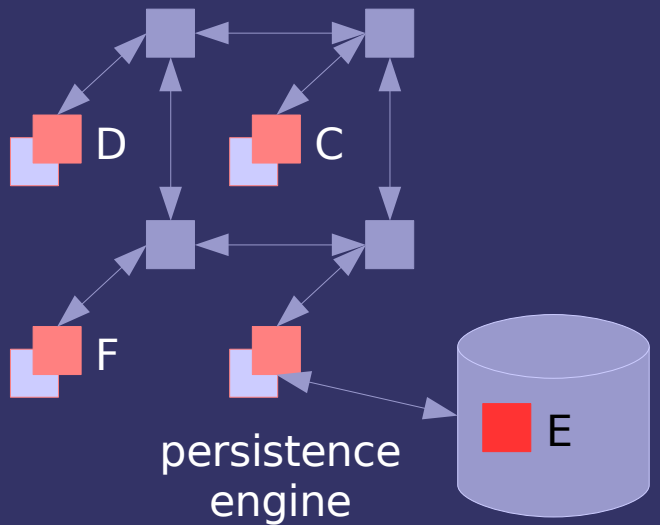




Goals (3)

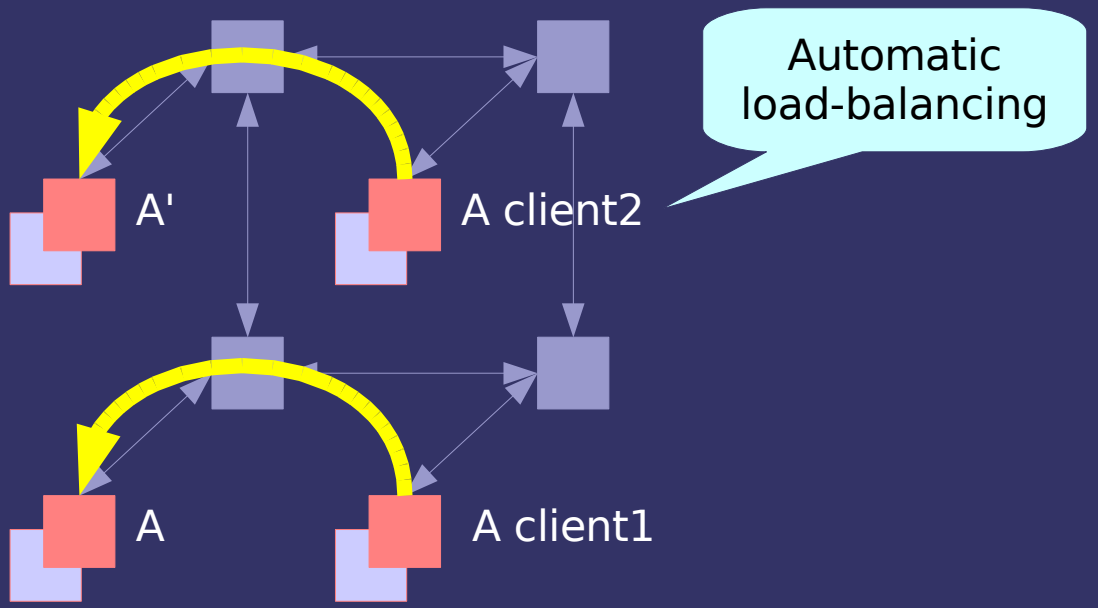
- Object persistence
  - Implicit activation & object migration





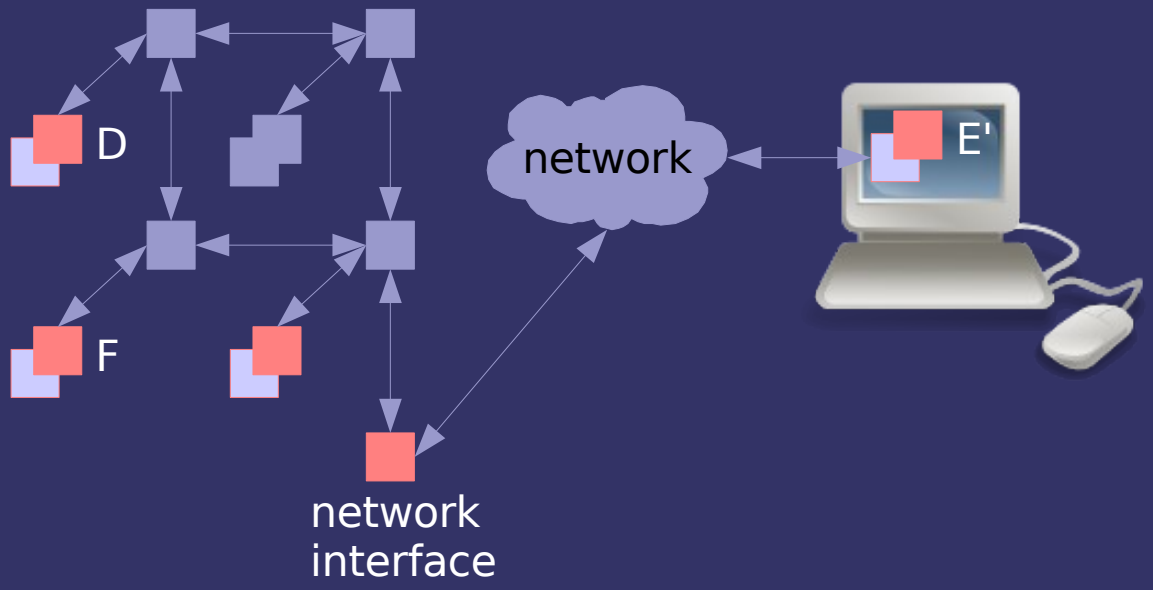
Goals (4)

- Replication transparency
  - Client-side transparency



Goals (5)

- Standards compliance
  - Ability to interact with standard distributed objects (CORBA, Ice, ...)



Goals (6)

- The above goals also imply
  - Incremental development from all-software always fully functional executable models
  - Transparent monitorization of object interaction
    - Distributed debugging support (Marzullo-Neiger)
  - Leverages standard distributed platform services
    - Event propagation, life-cycle management, naming, object trading, encryption, ...
  - Distributed applications may easily be ported to an MPSoC

Why object-based?

- Objects make persistence easy to implement
- An object is a natural hardware abstraction
  - Equivalent to a component
- Most standard distributed heterogeneous platforms are already object-based

We can already do that... (?)

- Fault tolerance...
  - ... either ad-hoc or on-chip module redundancy, and degraded operation modes (must be planned in advance)
- Replication...
  - ... either ad-hoc or explicit load balancing policies (replica-aware clients)
- Incremental HW/SW partitioning
  - ... interfaces must be re-synthesized (prevents run-time binding)
- Abstract channel modeling (SystemC)
  - ... just for modeling/simulation

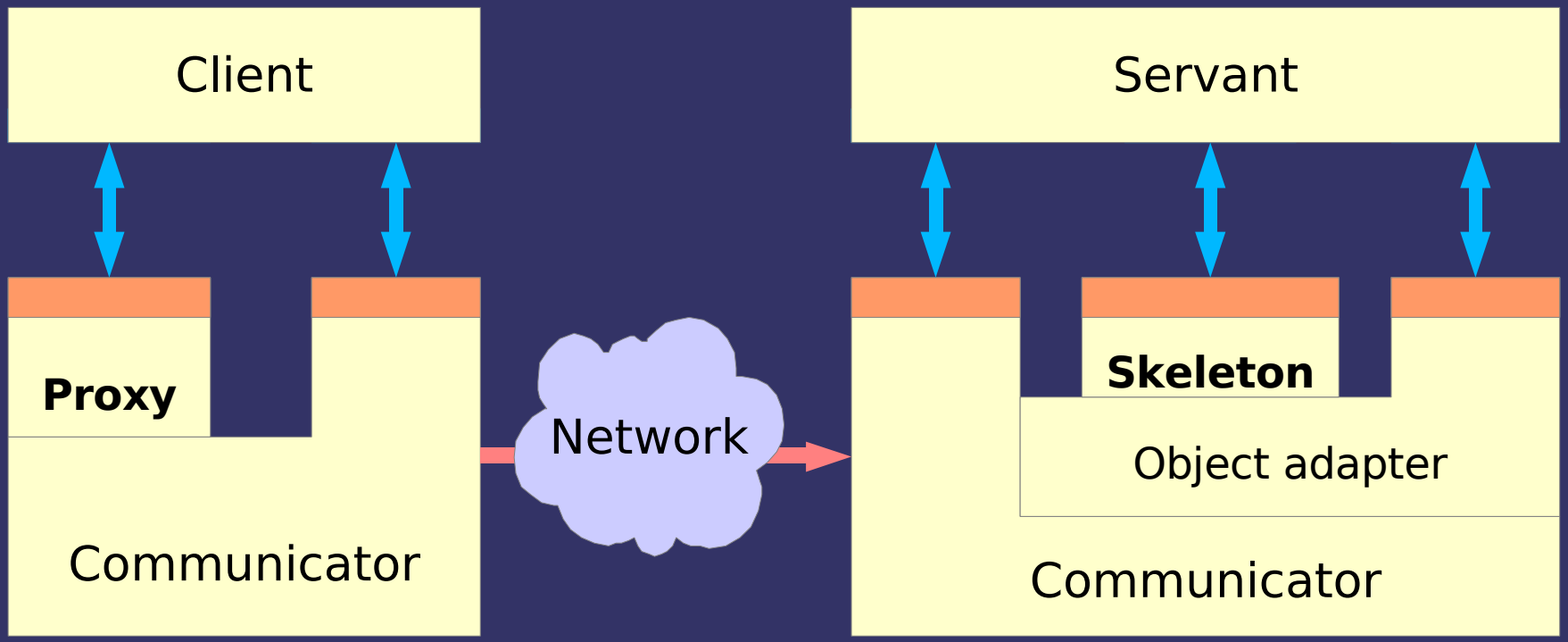
MPSoC as a multi-computing device

- Homogeneous multi-processor
  - Explicit parallelism: MPI, PVM, OpenMP ...
- A single complex computing device
  - Implicit parallelism
  - Thread-level task partitioning
  - Need for some system level OS services
    - Distributed context switch, ...
- Heterogeneous distributed platform
  - Object-level task partitioning
  - Full location transparency
  - Minimal need for support services

Is an MPSoC heterogeneous?

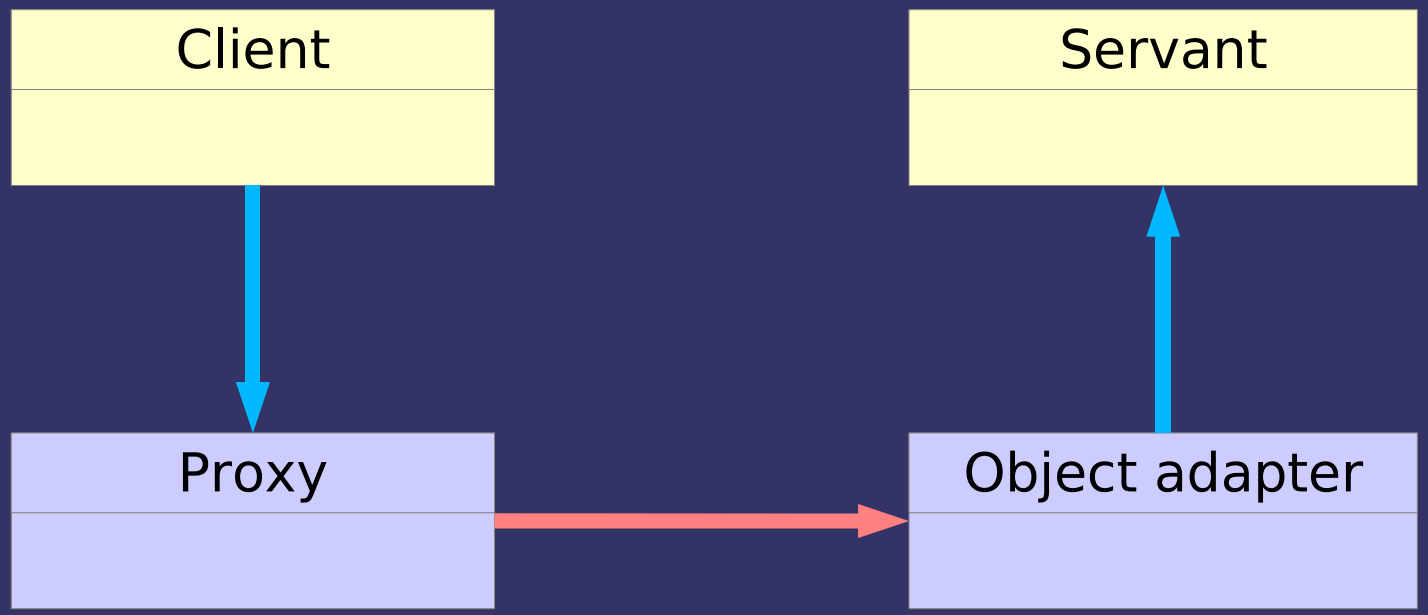
- Complex on-chip network topologies
- Multi-vendor processing cores
  - ARM + PPC + DSP + ...
- Application specific computing cores
- Heterogeneous node OS
  - RTOS for hard real-time functionality
  - uCLinux for easier development
  - Raw nodes for reducing memory footprint
- A cluster of MPSoCs is also an MPSoC!

Object-based distributed heterogeneous computing



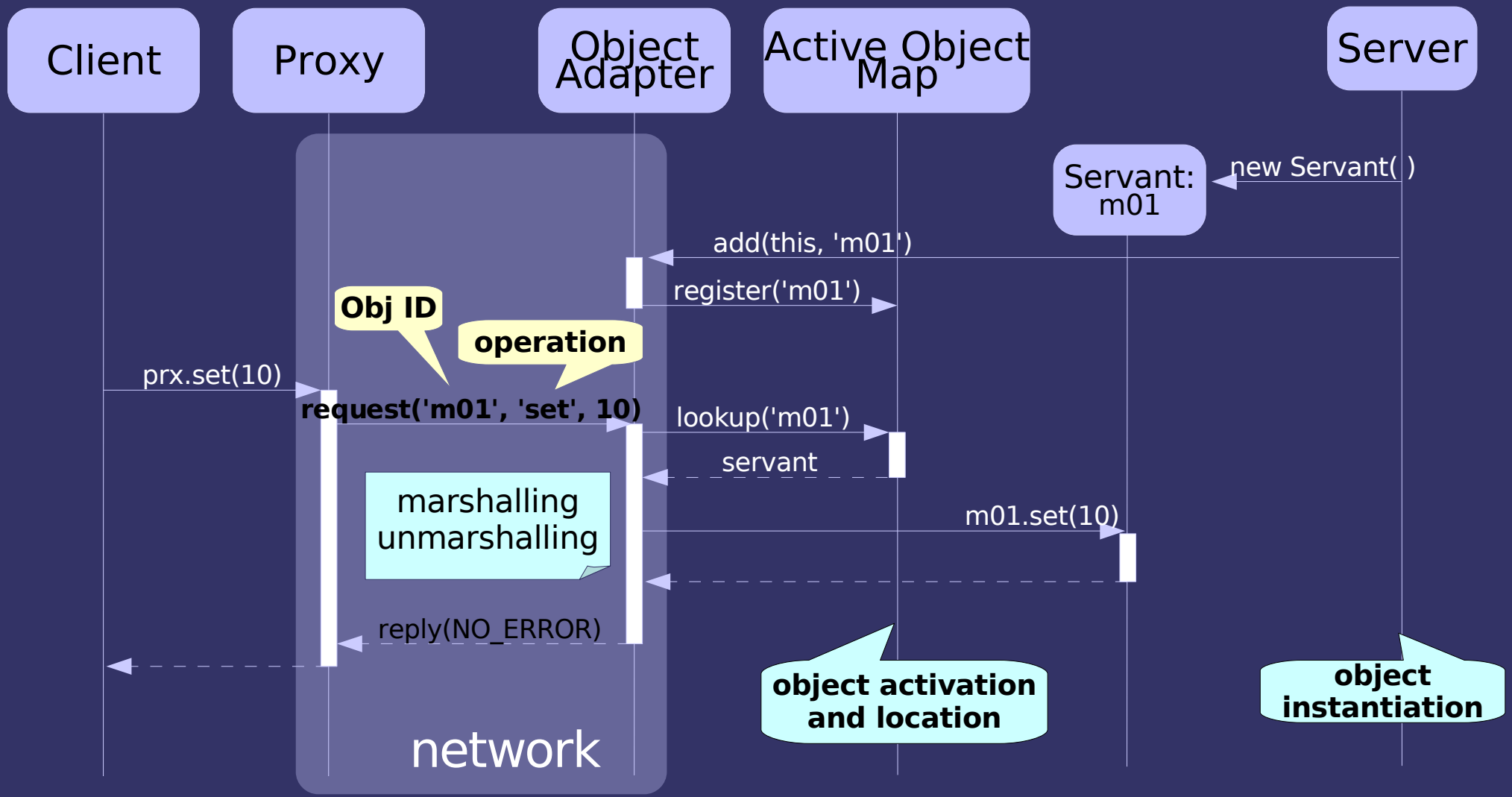
Object-based distributed heterogeneous computing

Object-based distributed heterogeneous computing



Similar to SystemC abstract channels!

Simple message-oriented protocol



What is already done?

- Simplified distributed objects with extremely small footprint (picoObjects)
- Hardware version of some middleware components
  - IP blocks behave as distributed objects
  - Resources and latency are comparable to Xilinx IPIF
- Standard middleware on top of a reliable message transport for MPARM using scratchpad memories (quelib)
  - Still missing meassurents

The overhead

- Hardware version

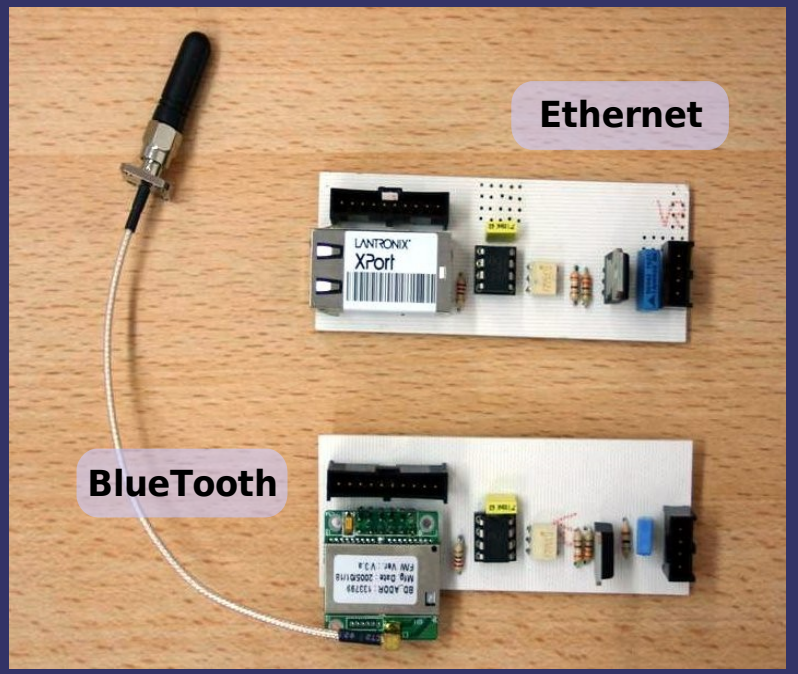
	OPB IPIF client		*OPB client*		*OPB IPIF server*		*OPB server*		*OCP client*	
	wrapper + func.	*func.*	*proxy + obj.*	*obj.*	*wrapper + func.*	*func.*	*adapter + obj.*	*obj.*	*proxy + obj.*	*obj.*
Slices	235	27	32	27	80	21	26	21	28	27
FF	326	37	50	37	129	32	36	36	41	37
Luts	395	49	61	53	55	37	45	32	55	53
Critical path (ns)	5.5	5.110	5.417	5.417	5.707	2.419	5.076	2.384	5.417	5.417

- Software version
  - Data on MPARM is not yet available

The overhead (2)

- Minimal latency version (picoObjects)
  - Porting to MPARM still in development

Middleware	size (server)
TAO	**1.738,0 KB**
nORB	**567,0 KB**
UIC/CORBA	**35,0 KB**
JacORB (Java)	**243,0 KB**
ZEN (Java)	**53,0 KB**
MicroQoSCORBA (TINI)	**21,0 KB**
picoCORBA (C)	**5,0 KB**
picoCORBA (Java)	**4,9 KB**
picoCORBA (TINI)	**3,8 KB**
picoCORBA (PIC12C509)	**415 words**
picoICE (PIC12C509)	**478 words**



What are the next steps?

- Profiling and statistics on MPARM middleware implementation
- Porting picoObjects on top of MPARM for minimum latency
- External network interface for MPARM
  - Clusters of MPARMs
  - Host-target transparent interaction
- Persistence implementation on top of an abstract storage API

Questions?