A Distributed Objects Framework using Virtual Machines

Stefano Gobbo

gobbo@elet.polimi.it

POLITECNICO DI MILANO

Dipartimento di Elettronica e Informazione

Milano Italy

State of Art

- A proliferation of software architectures designed to support parallel programming in heterogeneous systems

- Each of these solutions poses its attention on the approach to a particular class of problems

State of Art

- The scalability of these environments is bounded by self-contained designs
  - different semantics
  - different paradigms

- The number of such environments makes prohibitive the definition of a communication standard

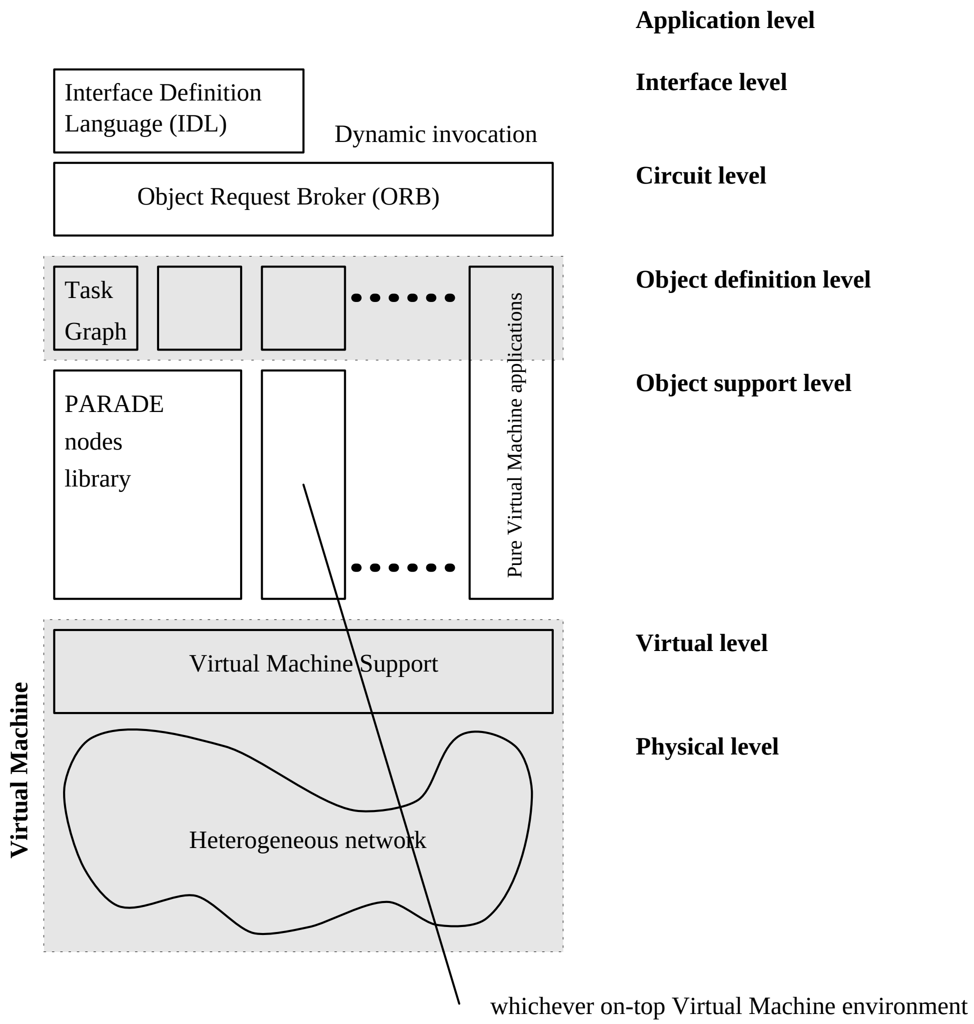
The problem

A babel of languages and environments for designing parallel applications, each one speaking its own language and well suited for the solution of a limited (even though wide) class of problems.

Proposal

The development of a framework in which standalone parallel applications designed using different tools or languages and following different paradigms can communicate.

Framework levels



Framework features

- Object Oriented characterization
- Definition of an agreement policy
- Unique run-time support

Object Oriented characterization

- Management of parallel application as *Distributed Objects* that export *methods*

- Applications are allowed to reference methods, *using* existing Objects or *exploiting* them to build new ones

- Object Broking mechanism

Definition of an agreement policy

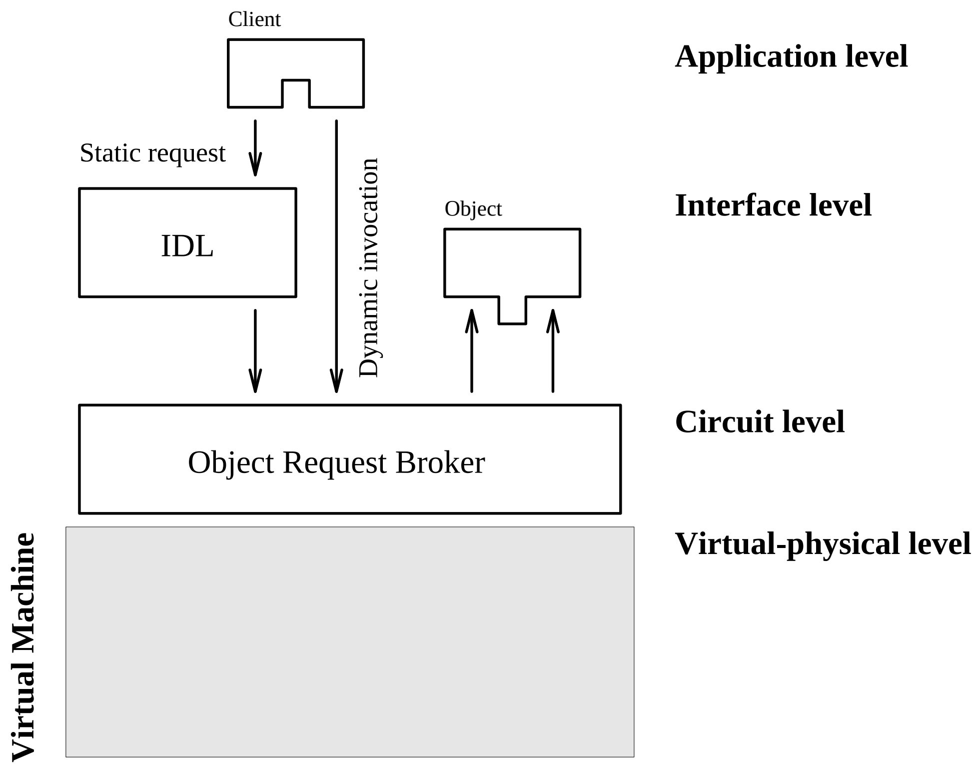
Client applications refer to Distributed Objects (servers) through interfaces:

- Written in C-like fashion using an Interface Definition Language (IDL)
- Independent from both the language and the environment used to develop the server application
- Describing methods exported by the Object

Object Request Broker

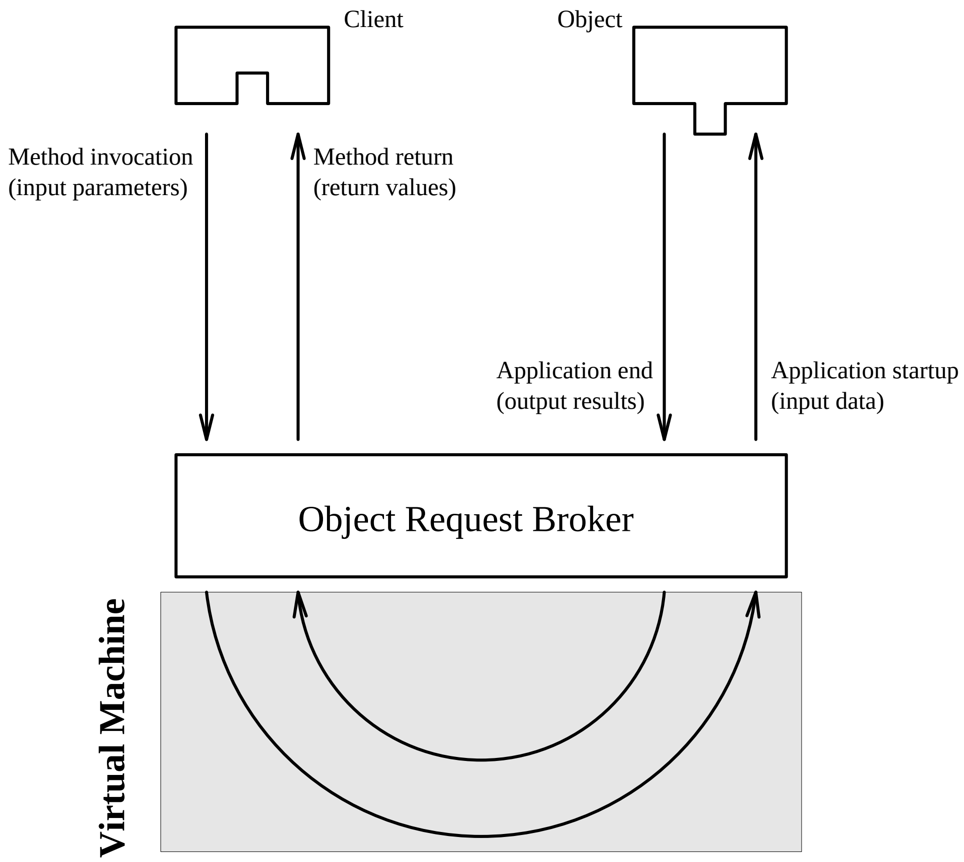
Manages operations involving Distributed Objects:

- Objects registrations
- Resolution of both static (well-known) and dynamic (look-up) requests
- Objects invocation



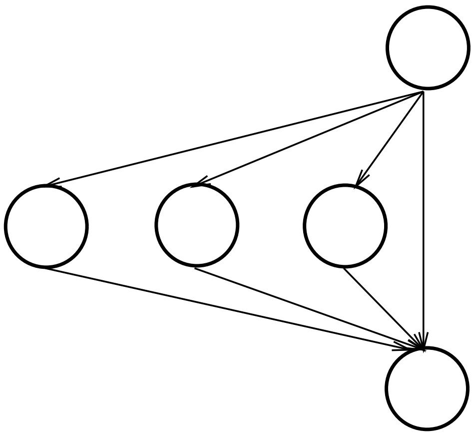
Unique run-time support

- Distributed kernel
- Object Request Broking facilities
- Applications interfacing

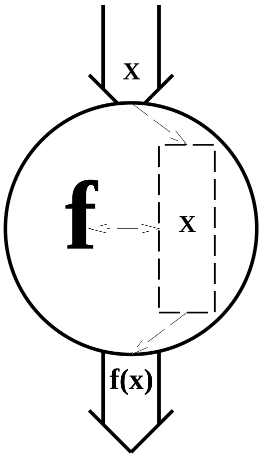


The PARADE Environment

- User-friendly environment
- Data Flow paradigm (a)
- Each node executes a sequental function *f(x)* (b)
- Nodes execution is triggered by the arrival of data x (b)



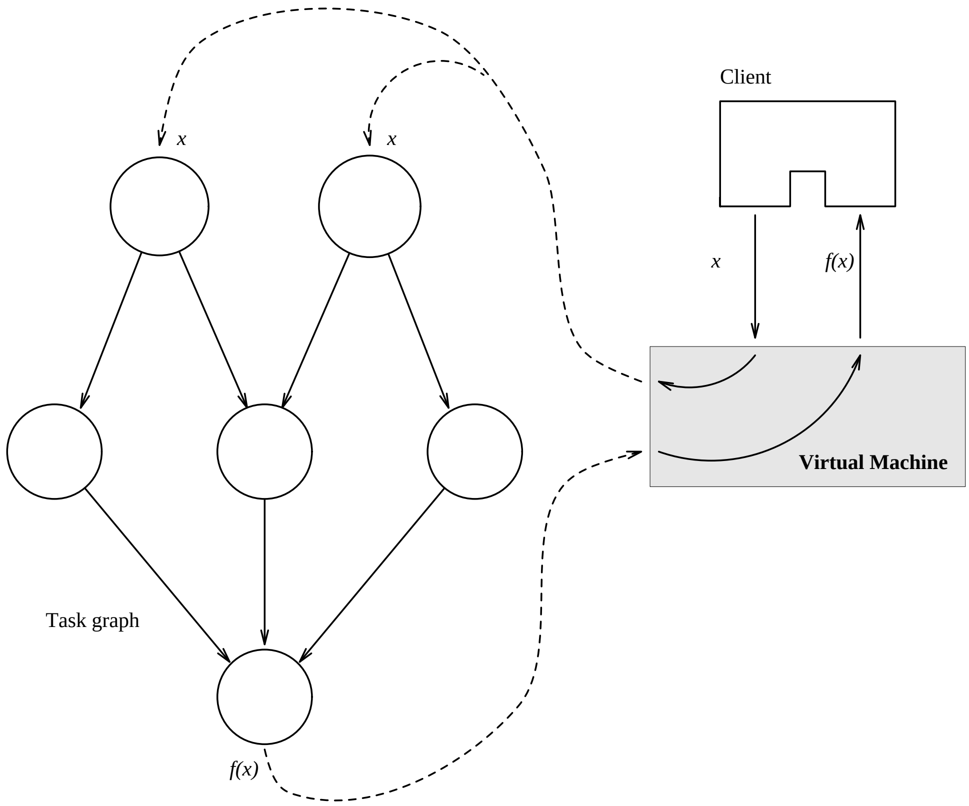
a)



b)

Exporting PARADE Applications

- Porting PARADE over PVM
- Leaving PVM the control of execution of PARADE applications
- Attaching IDL interfaces to PARADE applications



Conclusions

- *Reuse* of existing projects:
  - Virtual machine support implemented using **PVM**
  - ORB and IDL specifications follow the **OMG CORBA** definition
  - Pre-existent development environments
- Scalable system
- Exploit the best of each instrument
- Flexible low level control over standalone parallel applications
- Library of parallel applications