Components for the Grid with ProActive and Fractal

Matthieu Morel

(with Denis Caromel and Françoise Baude)

OASIS Team - INRIA

CoreGrid meeting
june 16-17th 2005

Objectives

- Observation : complexity and heterogeneity of the Grid
  - ➡ complex design, deployment and reusability
  - ➡ performance issues
- Answer : framework for programming and deploying components for the Grid
  - ➡ implementation of the Fractal model for ProActive
  - ➡ extensions for the Grid

Outline

- Context
- Functionalities
- Architecture
- Optimisations
- Perspectives

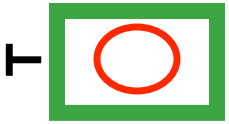
Context : ProActive

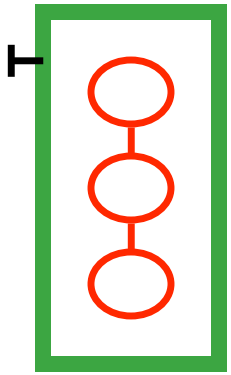
- A library for parallel, distributed and concurrent computing
- Written in Java
- Active object model
- Meta Object Protocol
- Deployment framework

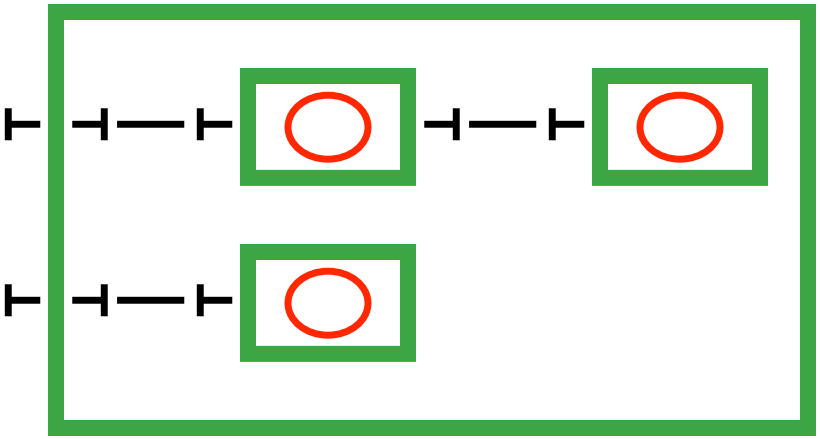
A new implementation of Fractal

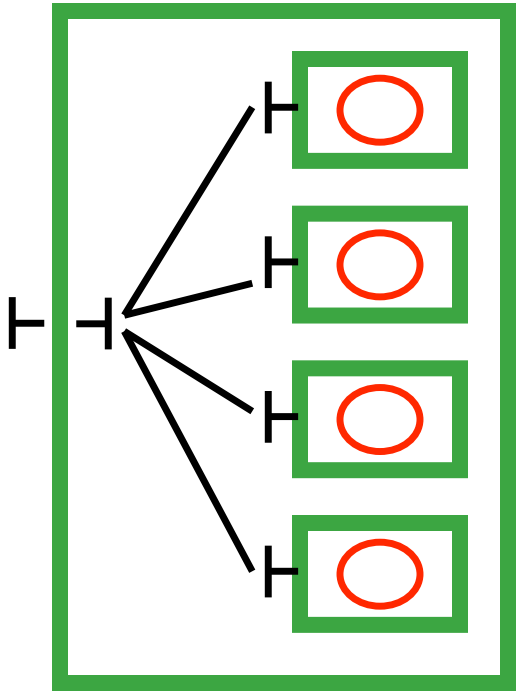
- Why not use Julia?
  - Could not reuse features offered by ProActive!
  - ➡ had to go for our own implementation
- Specific goals
- Specific architecture
- Conformance to the Fractal specification :
  - reflective (controllers : Life Cycle, Content, Binding, Attributes)
  - custom controllers
  - Component and Interface
  - typed components
  - bootstrap component
  - no templates, no sharing
    - ➡ conformance level 3.2 (max is 3.3)
  - Standard FractalADL

ProActive components : 4 flavors



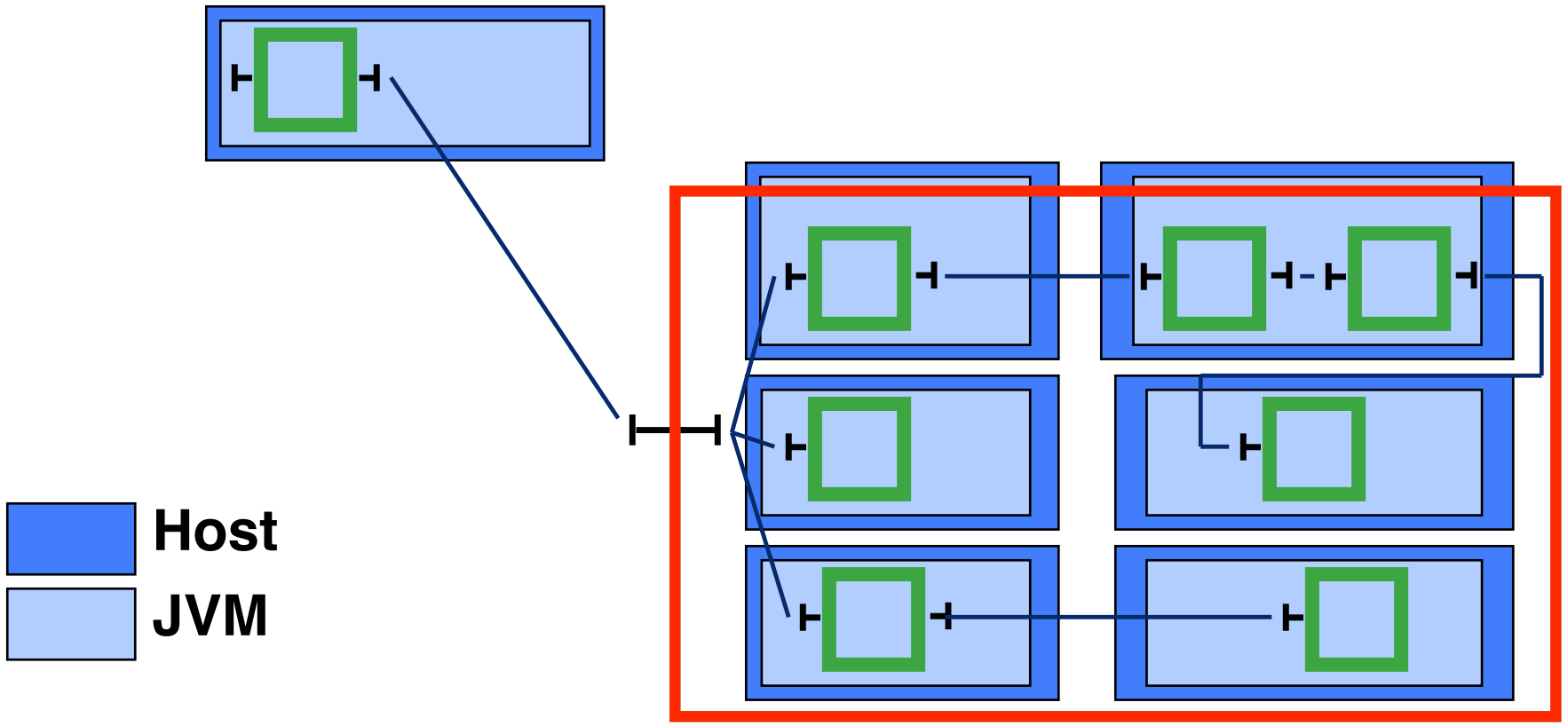






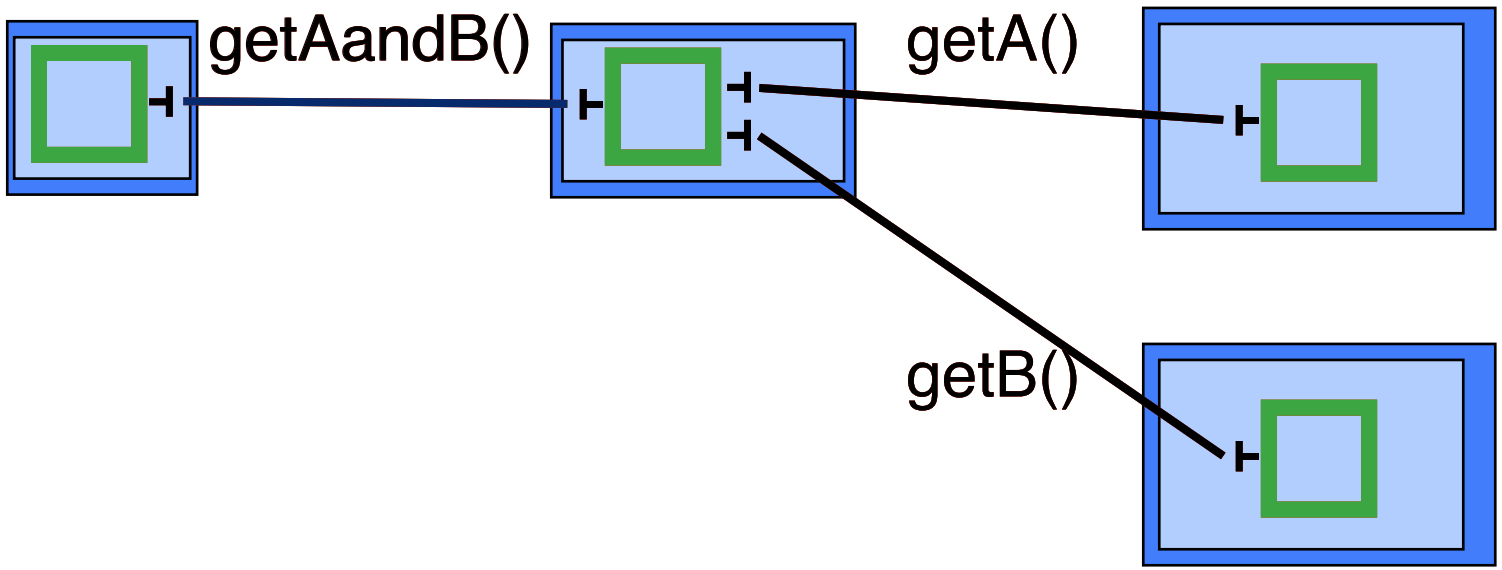
Functionalities : distribution

- 1 component can be distributed over several hosts
- Distribution is transparent



Functionalities : concurrency

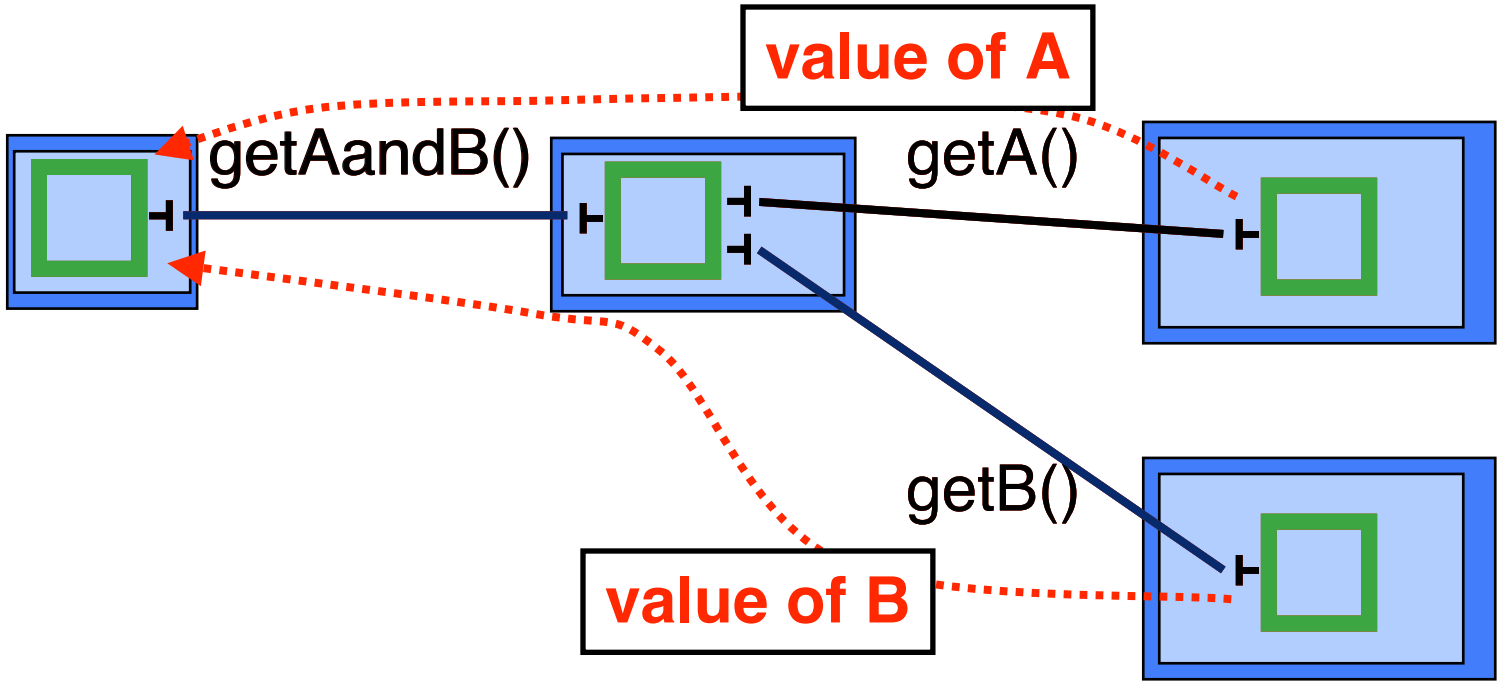
- Example 1 : synchronous method calls



Functionalities : concurrency

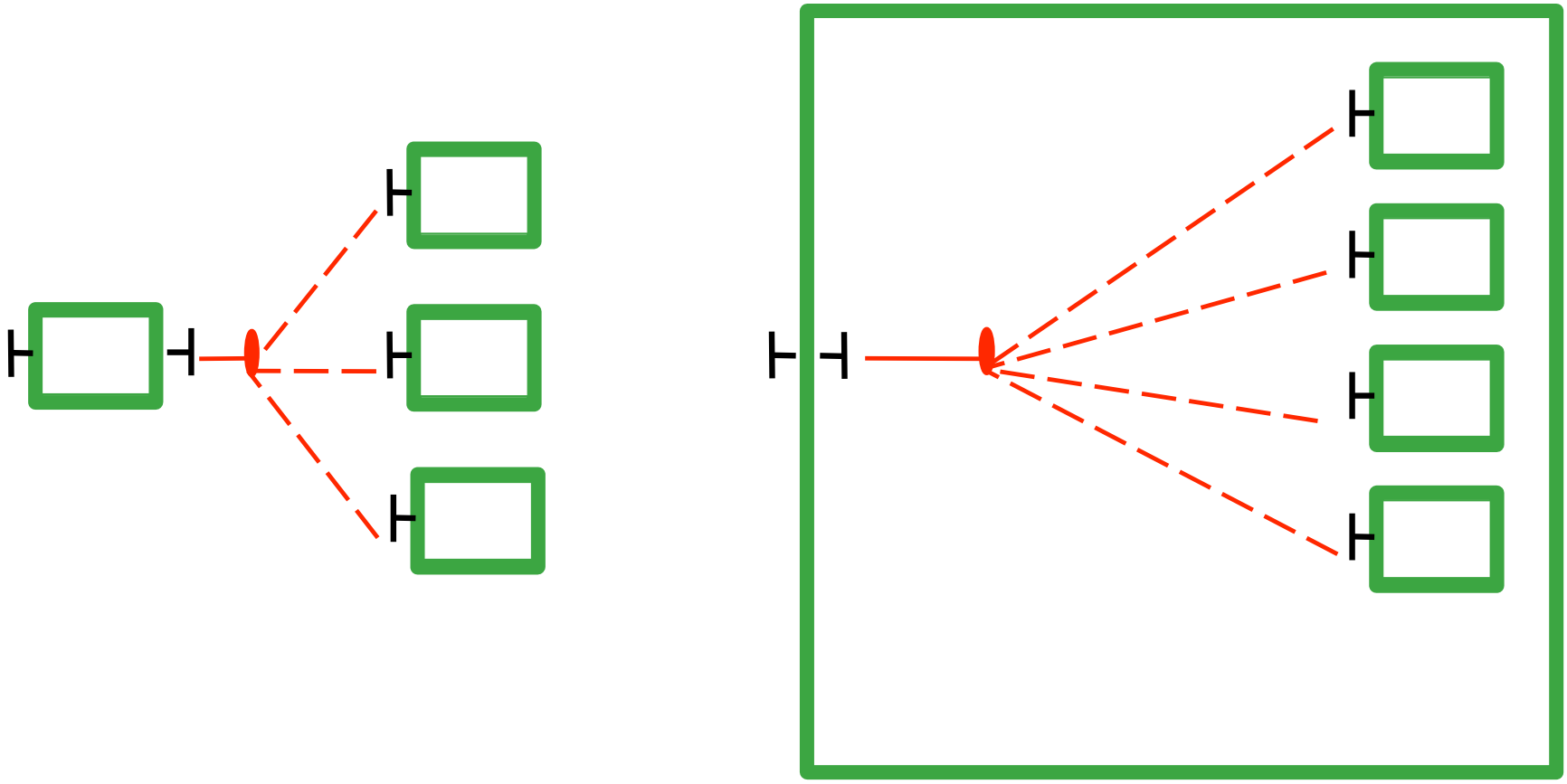
- Example 2 : asynchronous method calls with futures and automatic continuations

Non blocking method calls



Functionalities : groups

- Typed group communications
- 2 modes : broadcast or scatter

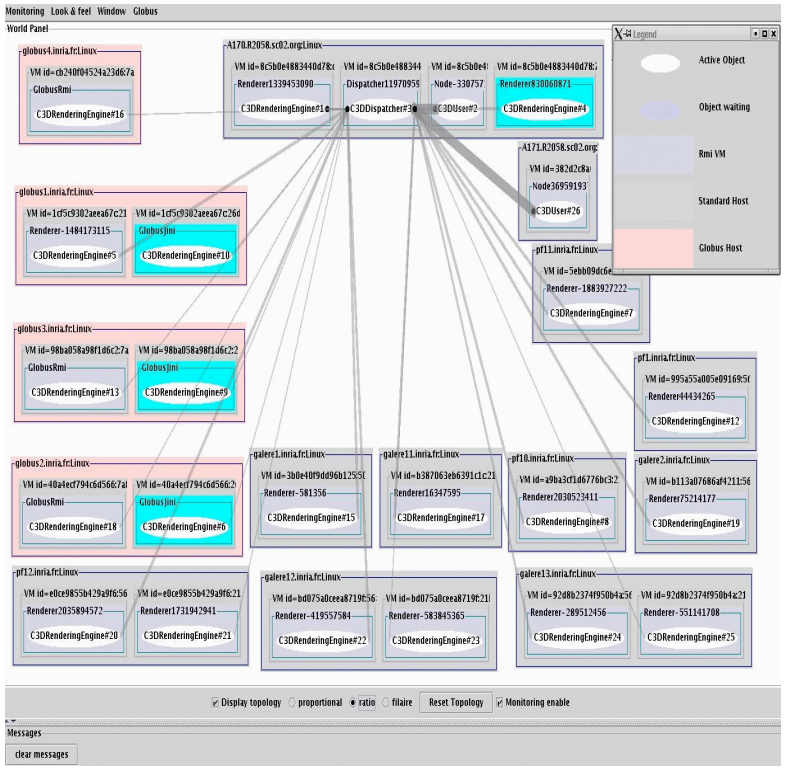


Functionalities : tools

- Deployment framework
  - virtual nodes
  - connection to hosts
  - creation of remote JVMs
  - instantiation / assembly / binding of components
- common ADL = common tools with the Fractal community
  - composition of virtual nodes
  - FractalGUI
    - run-time capabilities

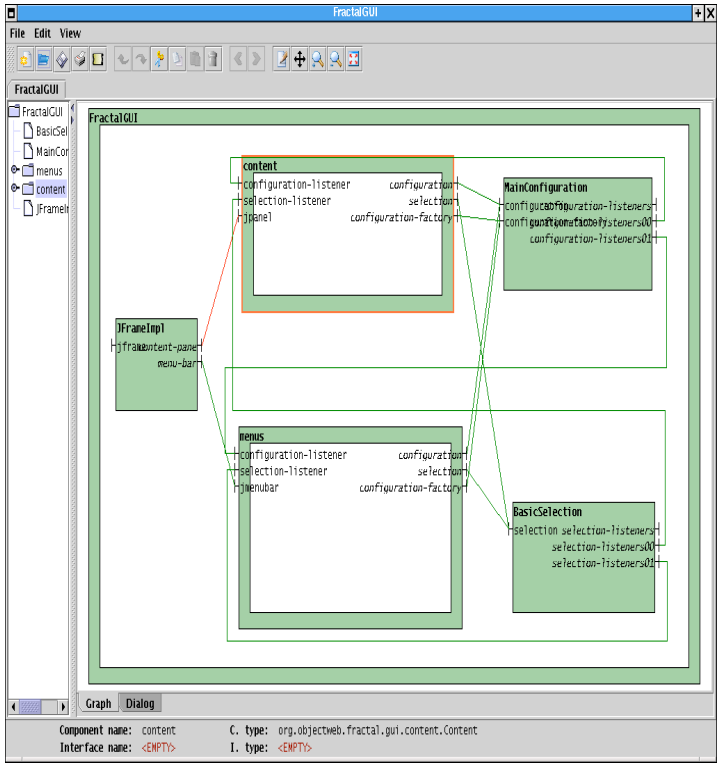
Design and monitoring tools

IC2D

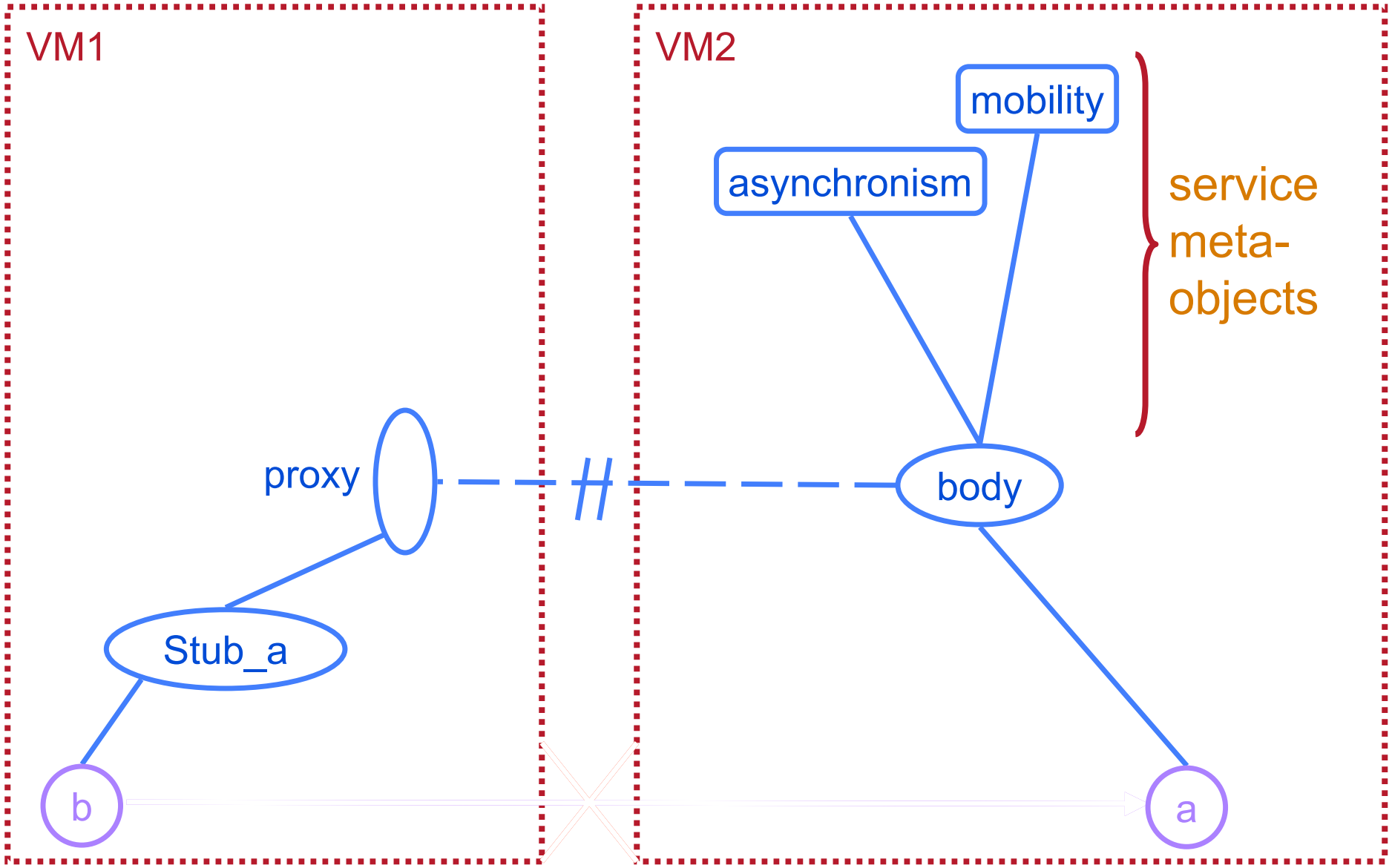


U

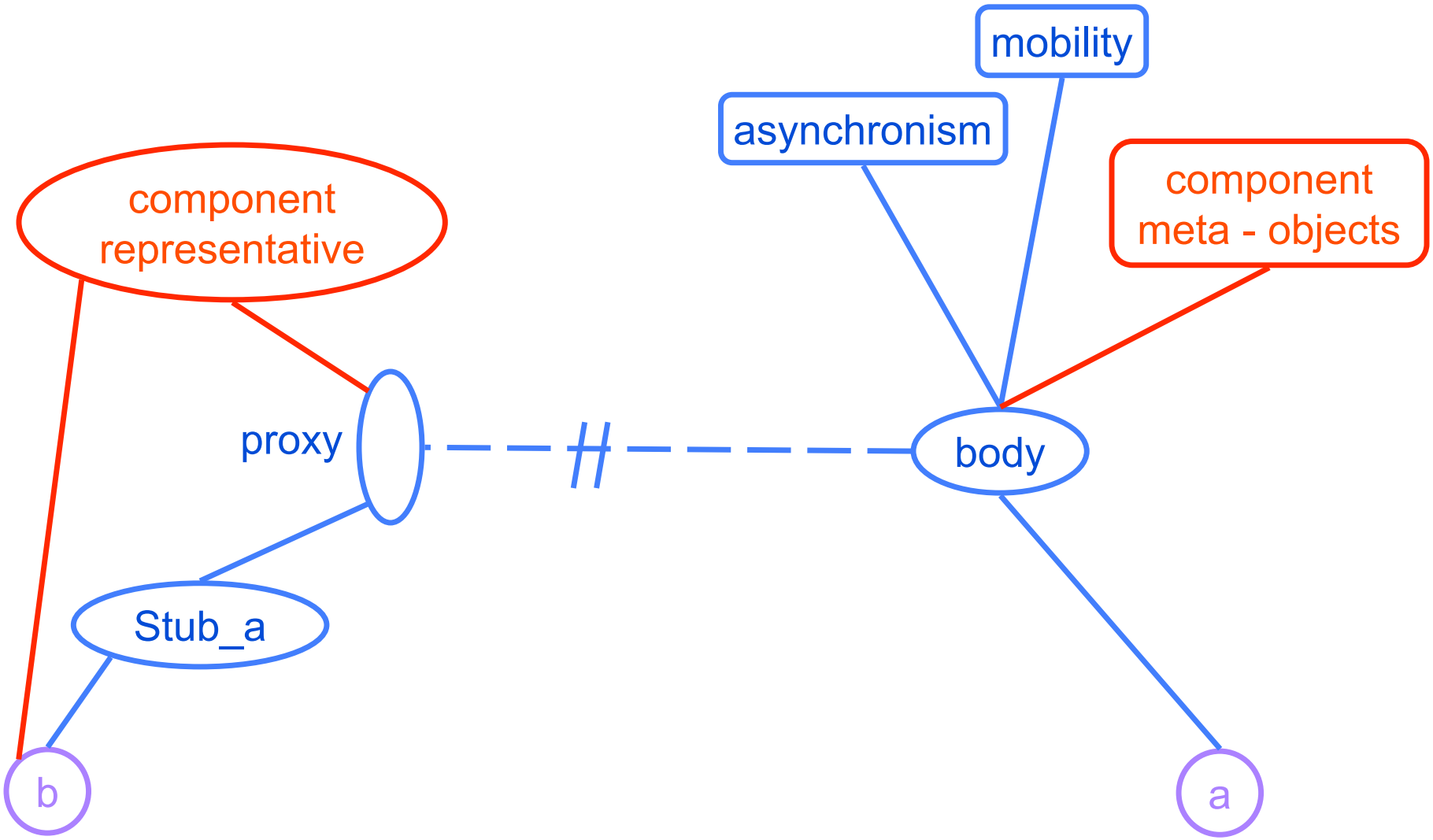
FractalGUI



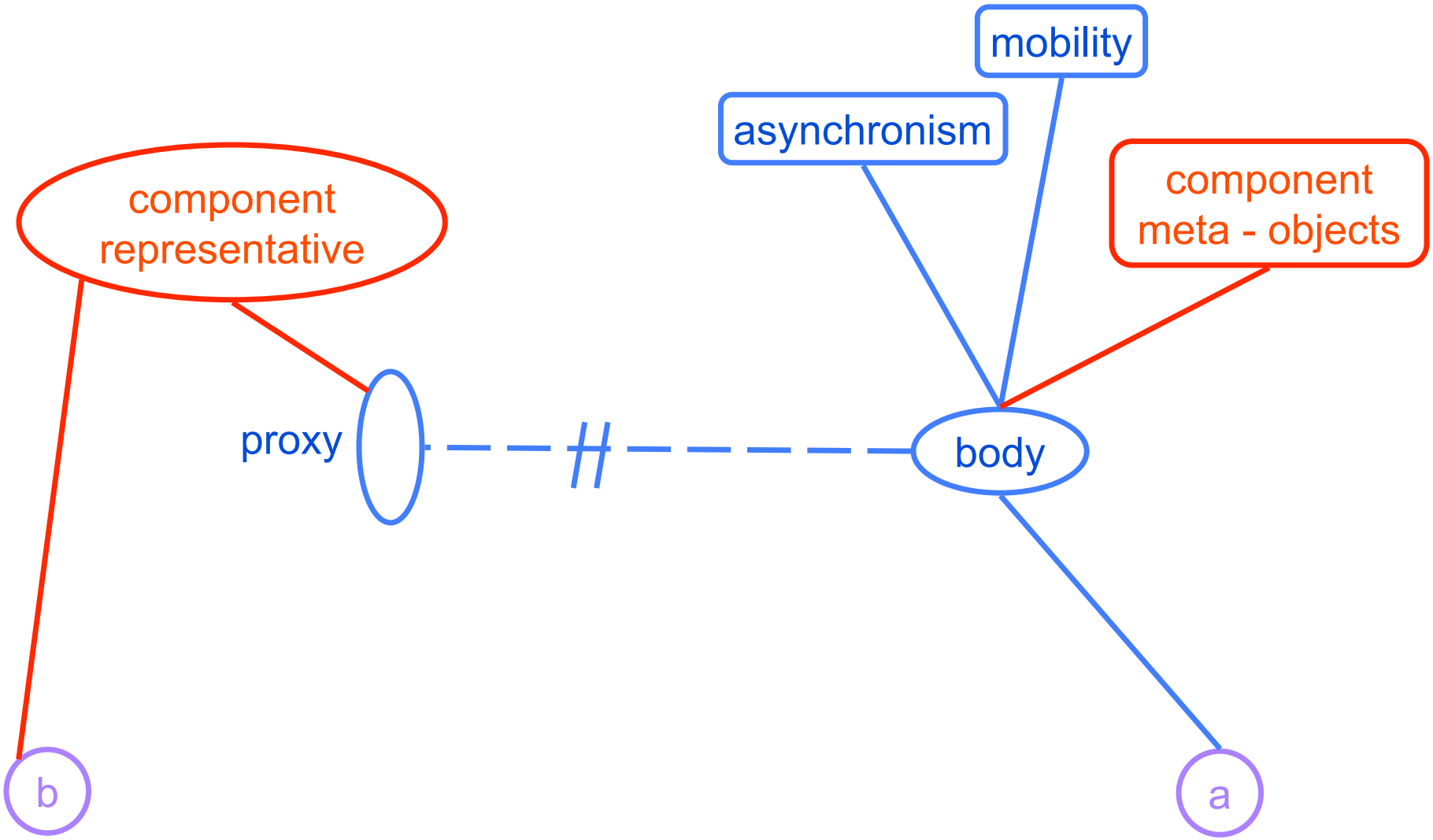
Architecture : MOP



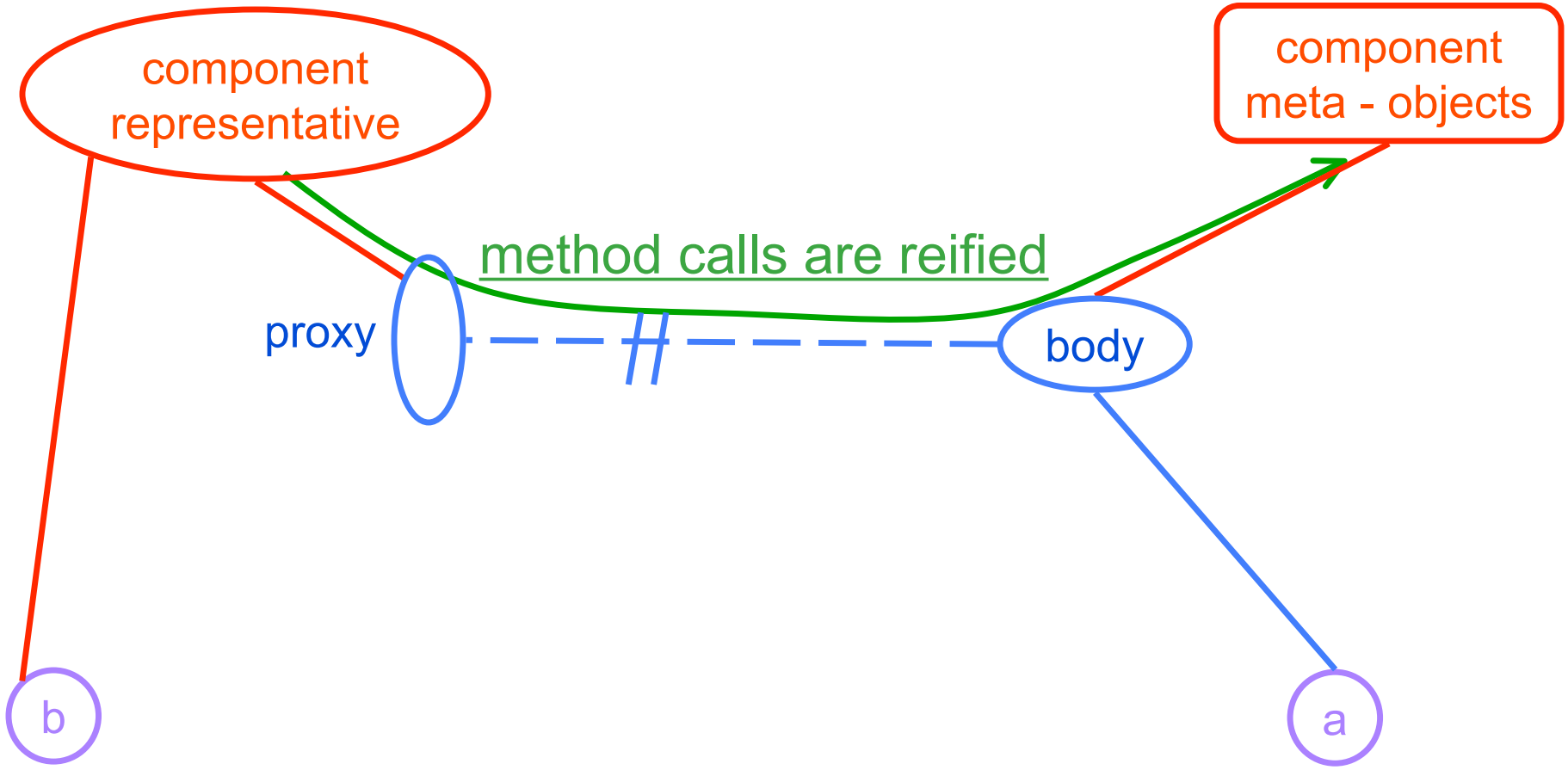
Architecture : component stubs



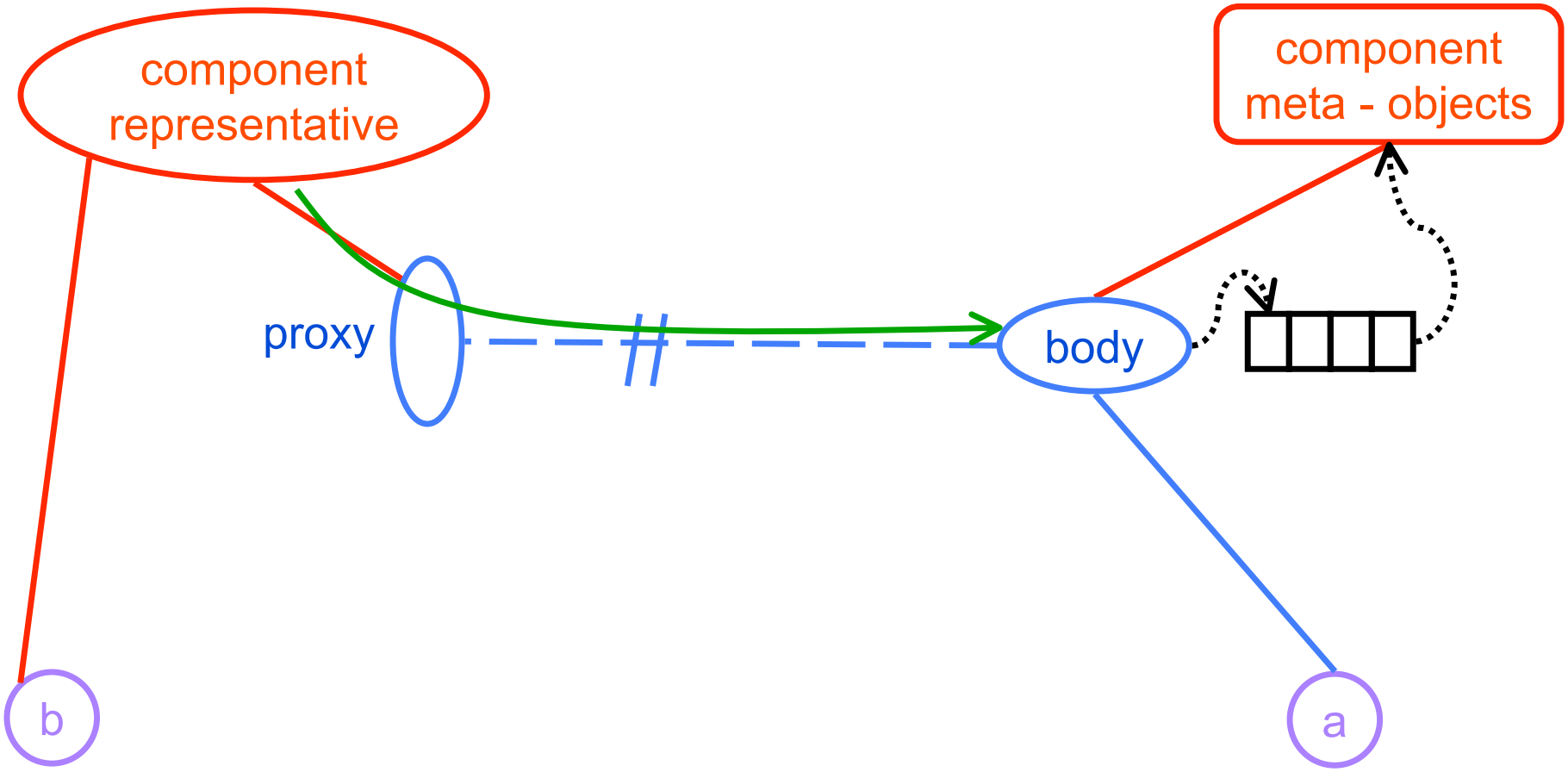
Architecture : component stubs



Architecture : communications

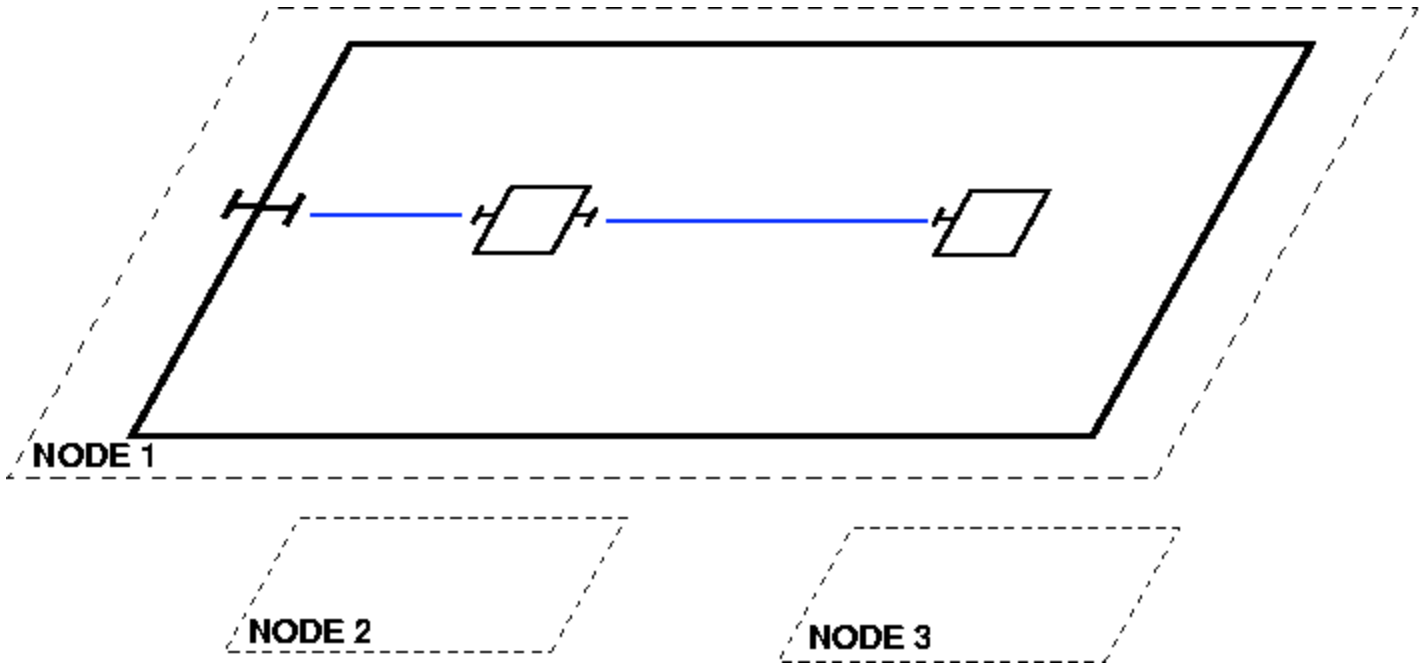


Architecture : request queue



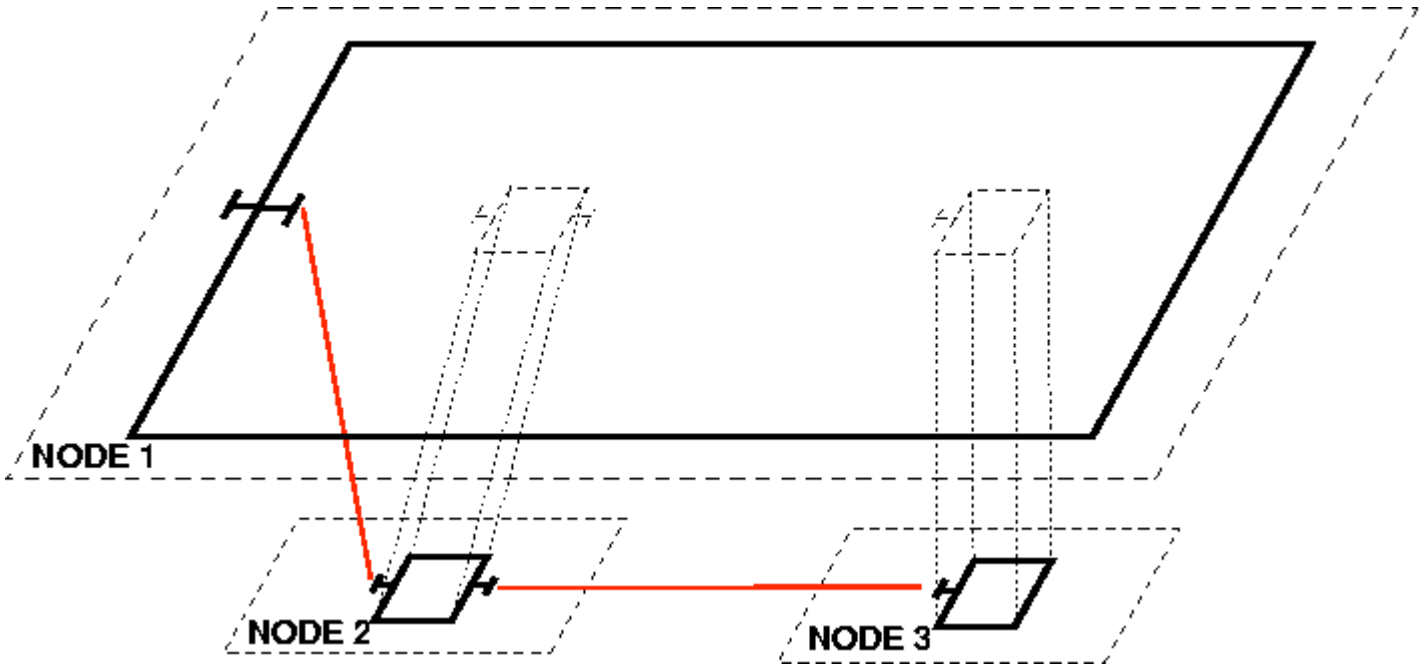
Optimisations

- Optimisations
  - shortcuts for distributed communications
    - local components :



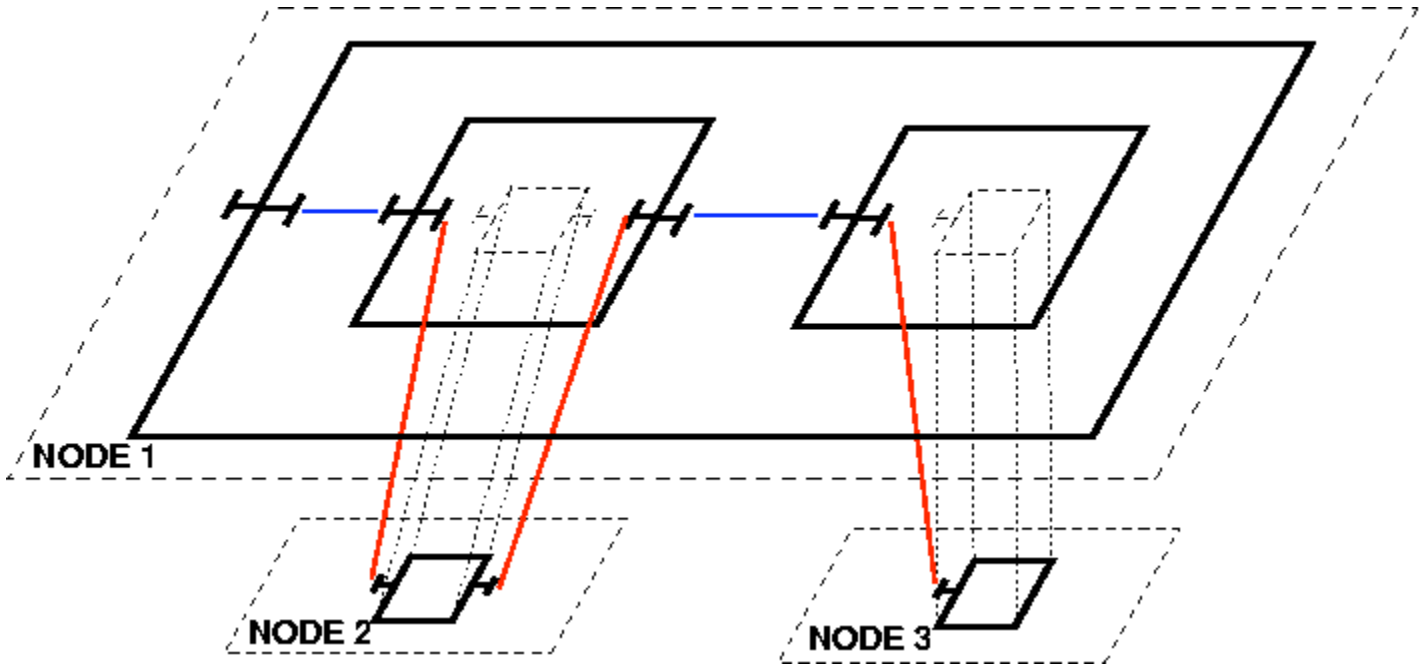
Optimisations

- Optimizations
  - shortcuts for distributed communications
    - distributed components



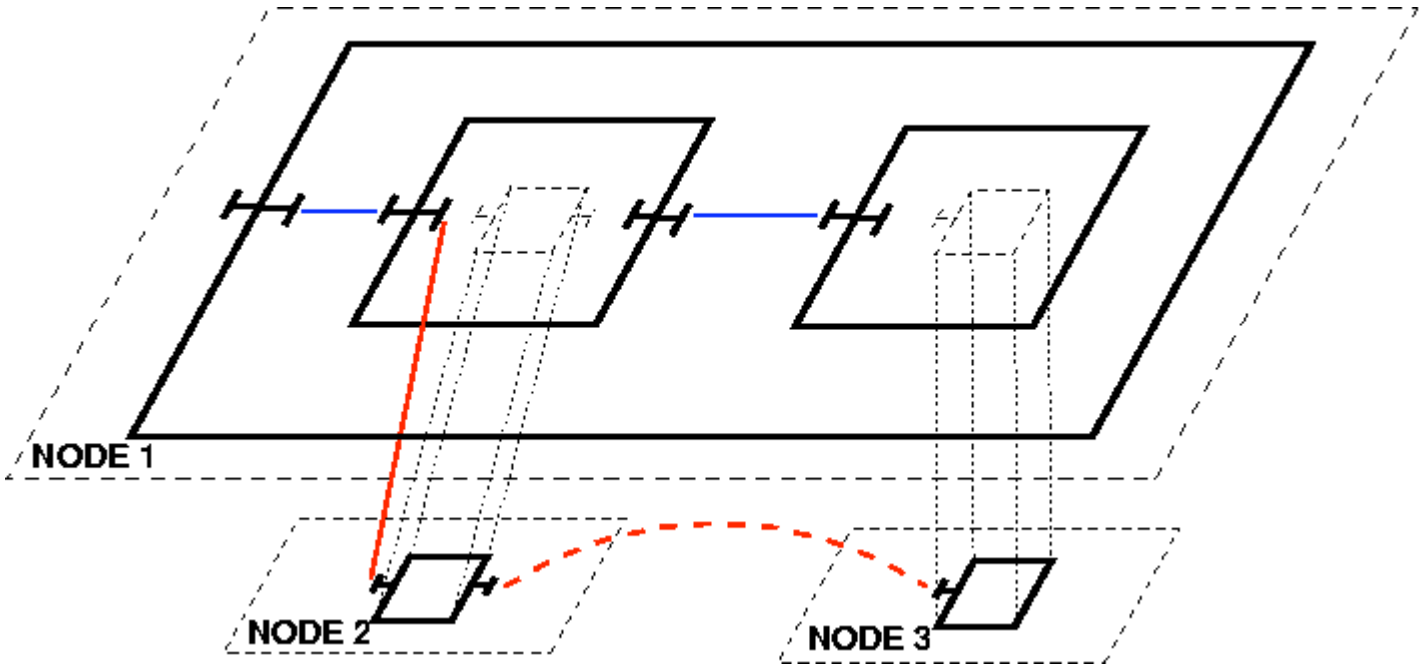
Optimisations

- Optimizations
  - shortcuts for distributed communications
    - distributed components



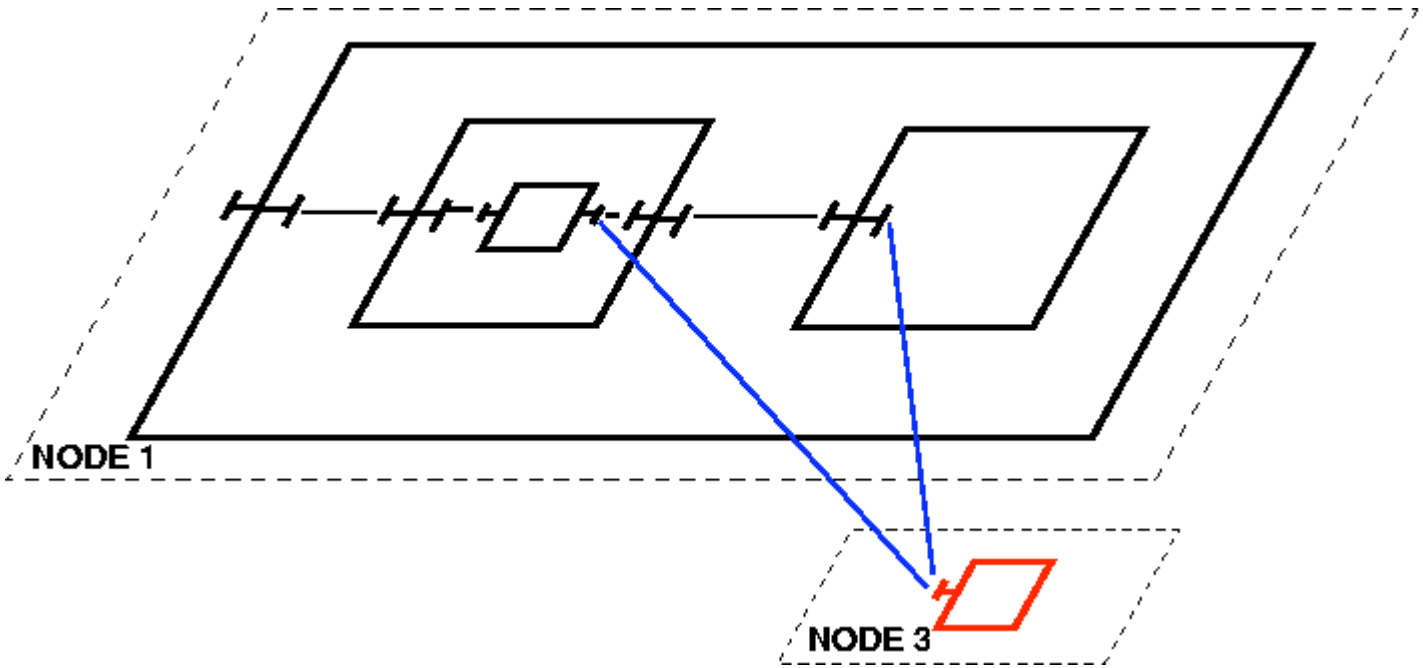
Optimisations

- Optimizations
  - shortcuts for distributed communications
    - distributed components : **tensioning**



Sharing ?

- A feature of the Fractal model
- Currently not in our implementation
- Used for representing resources (database, sensors etc…)



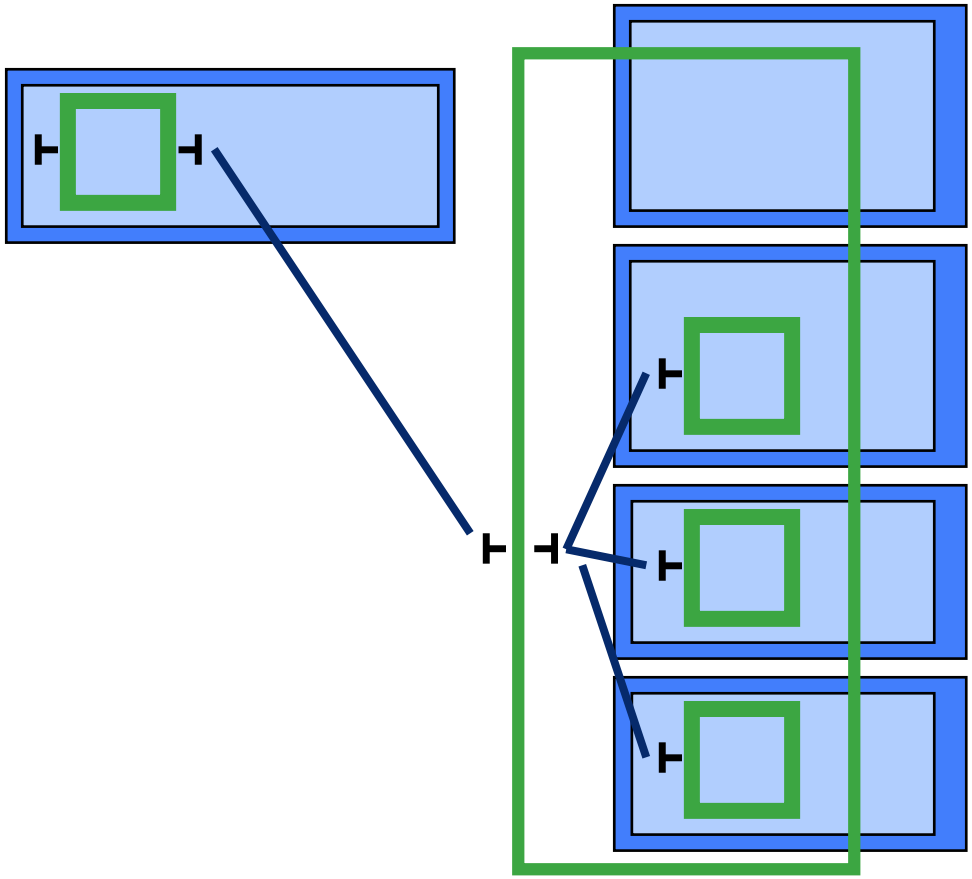
Dynamic reconfiguration?

- Specified in the model but :
- Shortcuts ?
- Sharing ?

➔ complex operations !

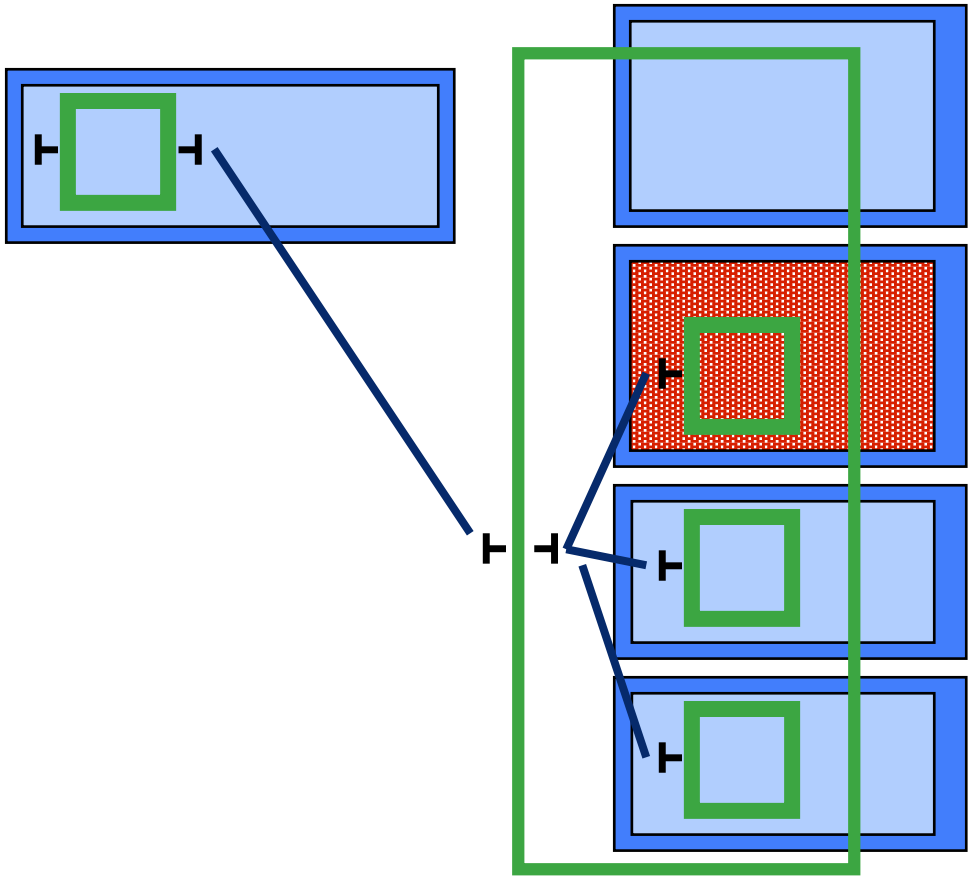
Perspectives : load-balancing

- Adaptability to stressed environment
- Connections maintained :
  - bindings preserved, **NO lost** communications



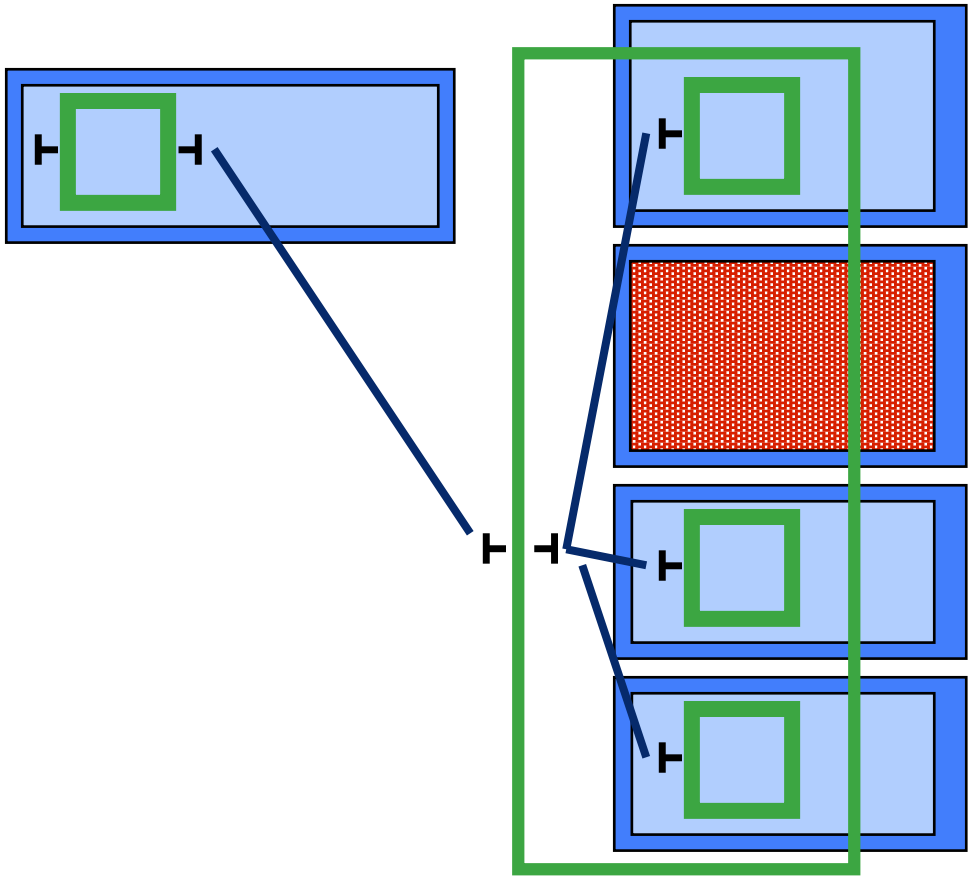
Perspectives : load-balancing

- Adaptability to stressed environment
- Connections maintained :
  - bindings preserved, **NO lost** communications



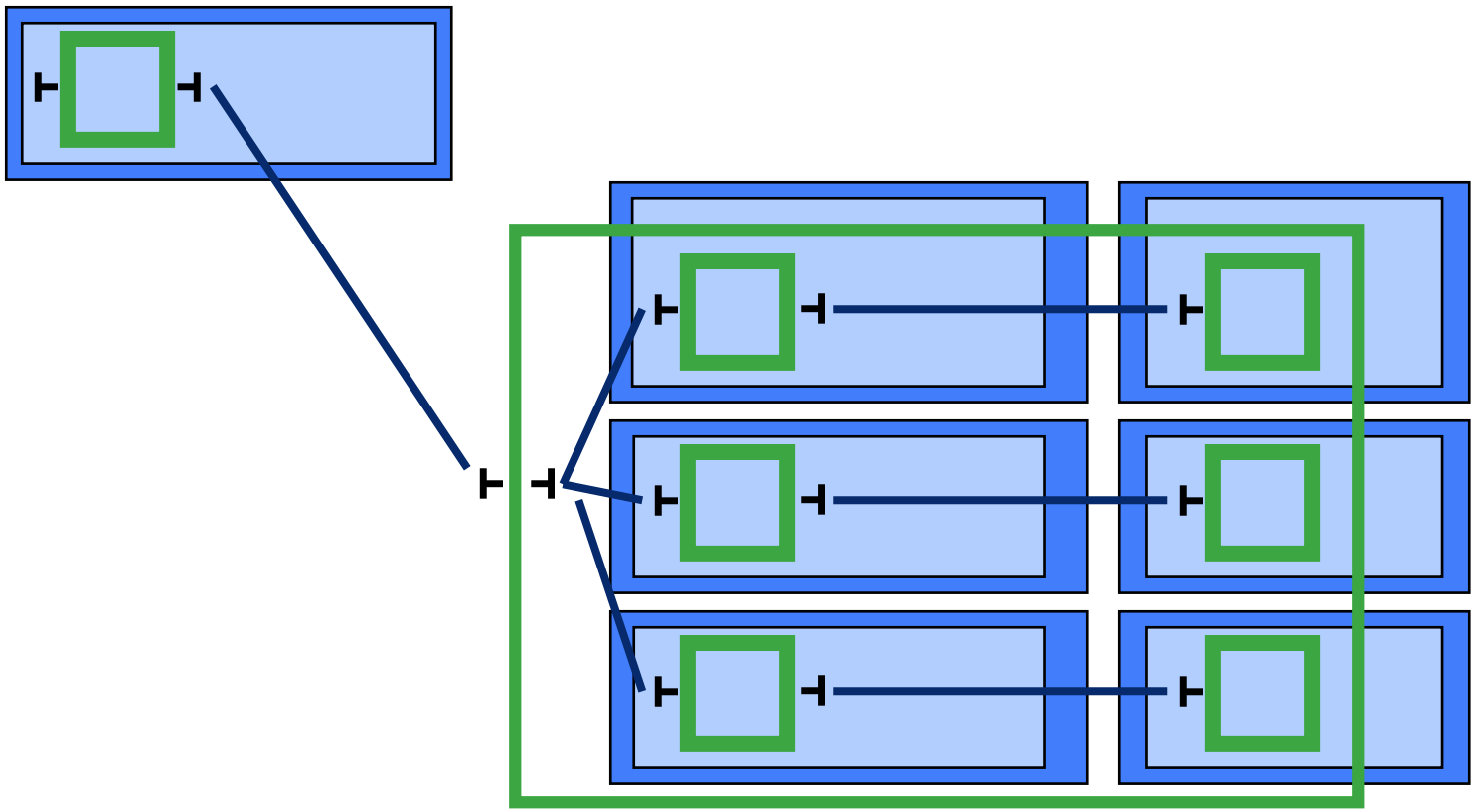
Perspectives : load-balancing

- Adaptability to stressed environment
- Connections maintained :
  - bindings preserved, **NO lost** communications



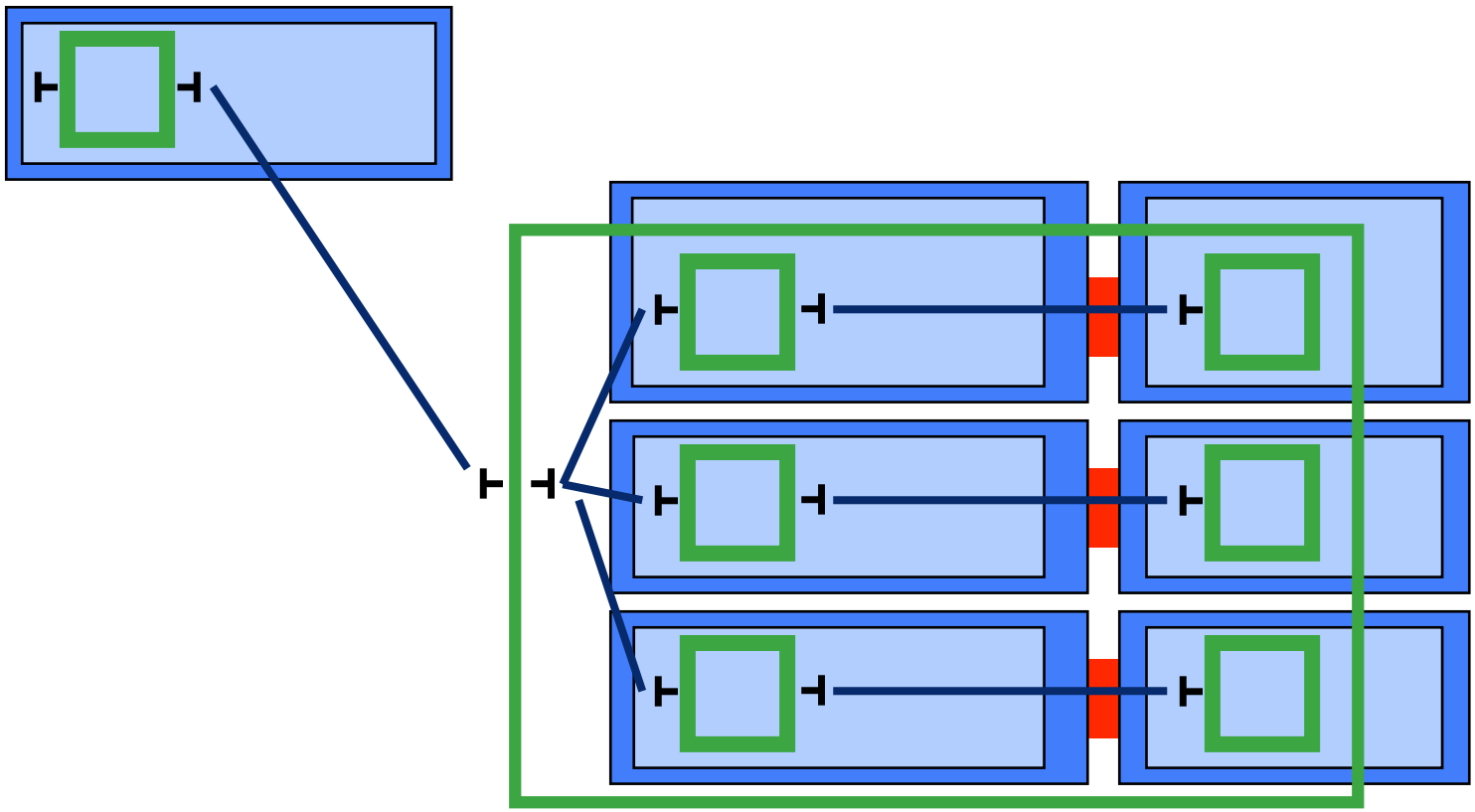
Perspectives : co-allocation

- When lots of communications between components



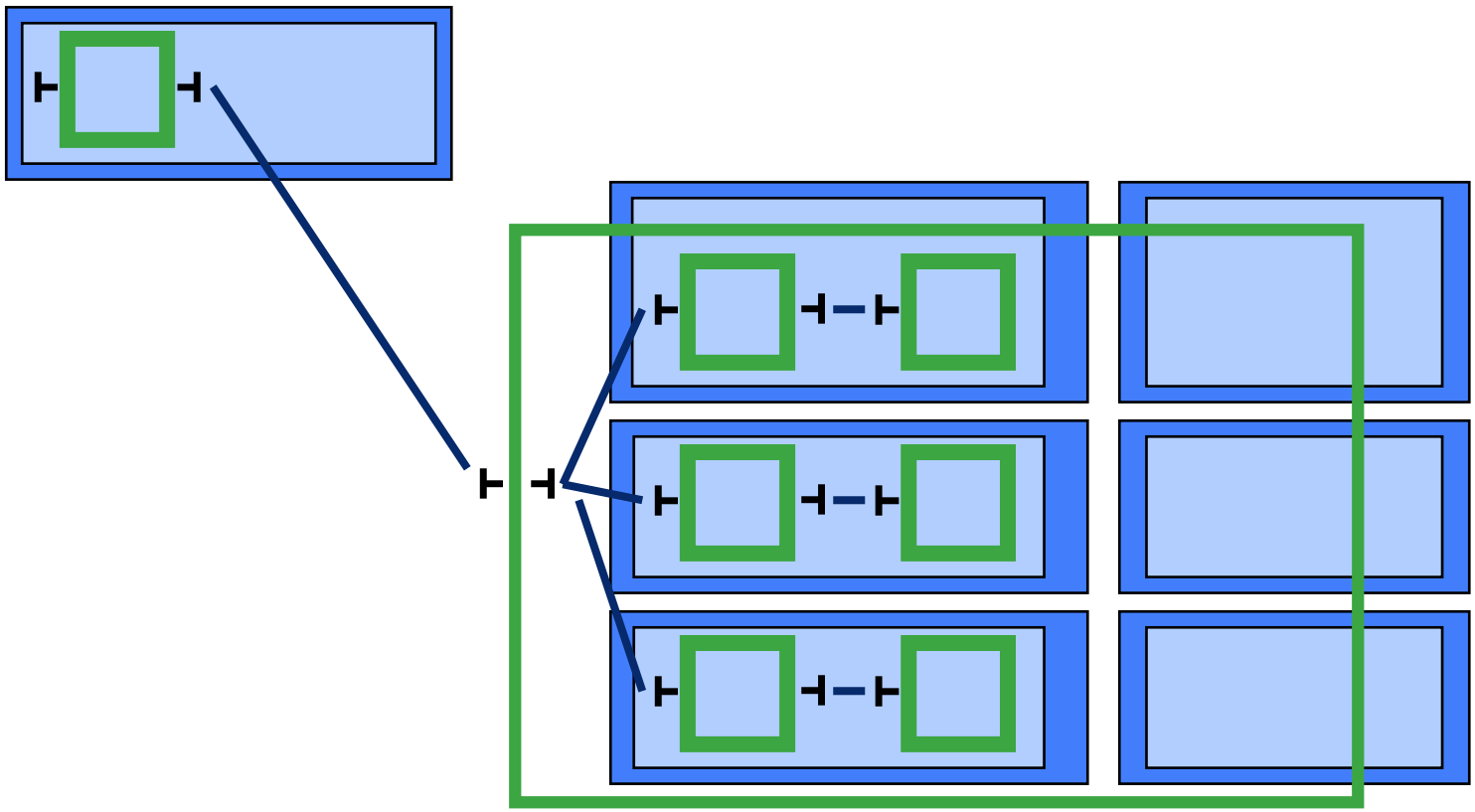
Perspectives : co-allocation

- When lots of communications between components



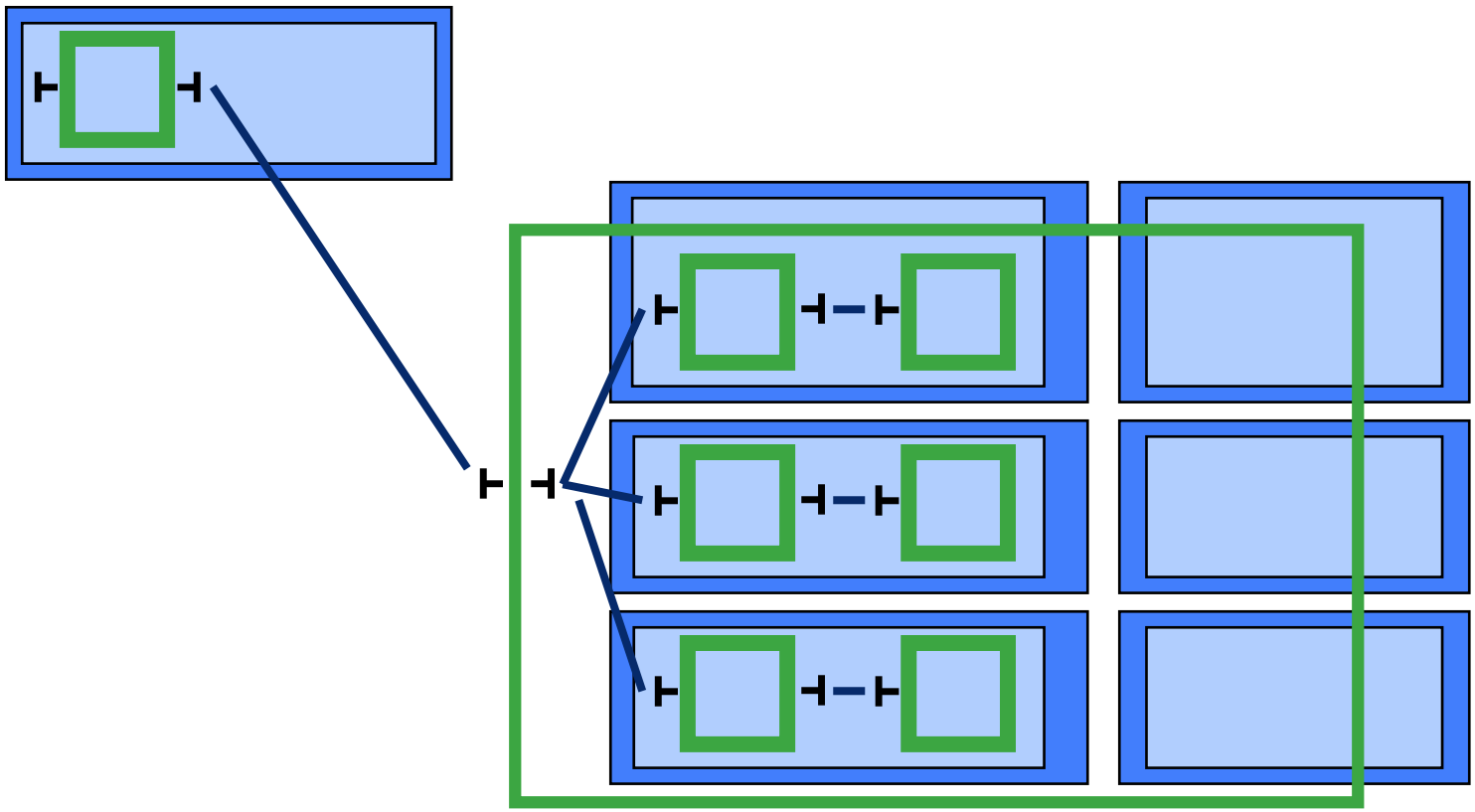
Perspectives : co-allocation

- When lots of communications between components



Perspectives : co-allocation

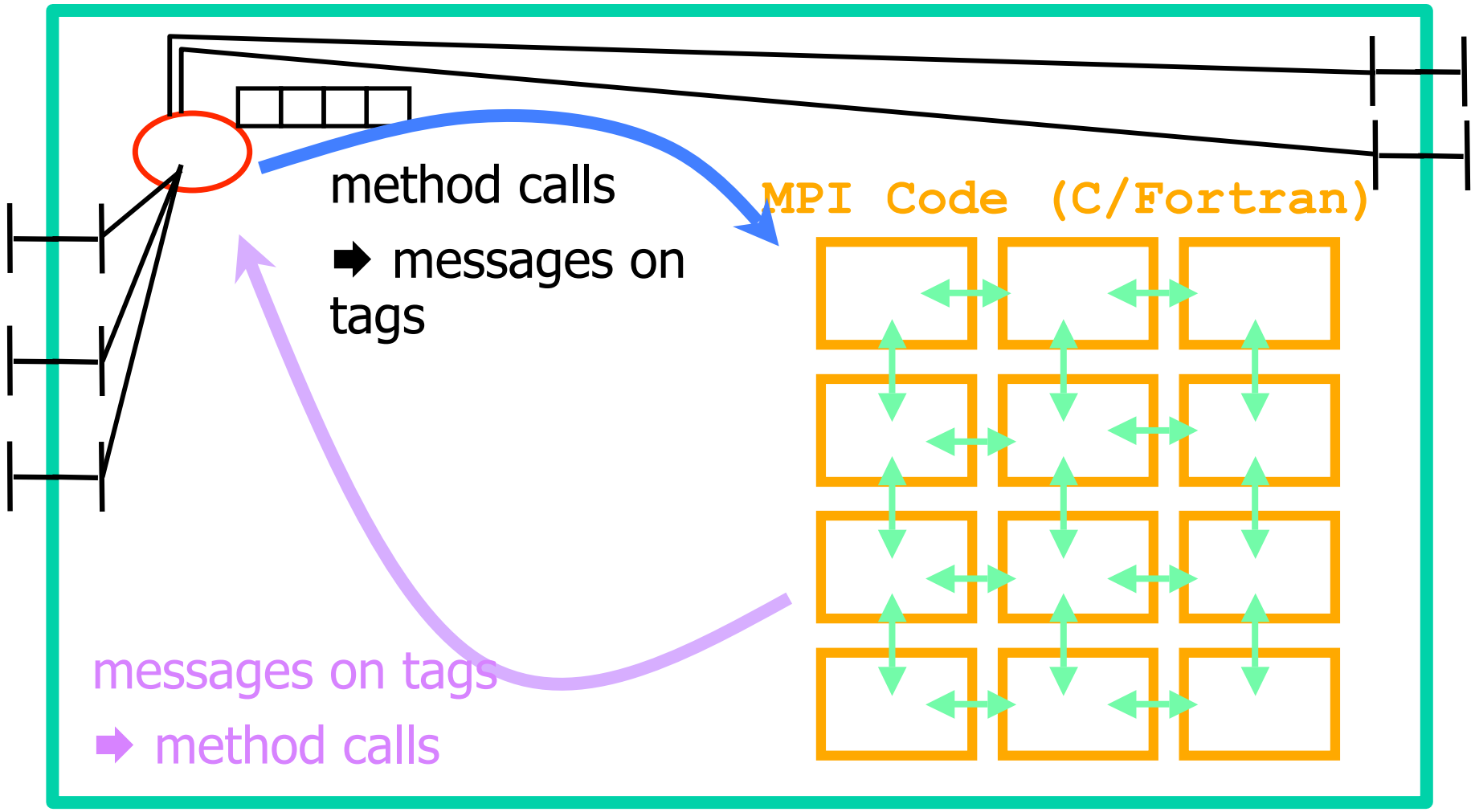
- When lots of communications between components
- Dynamic behavior or specified during design (virtual nodes)



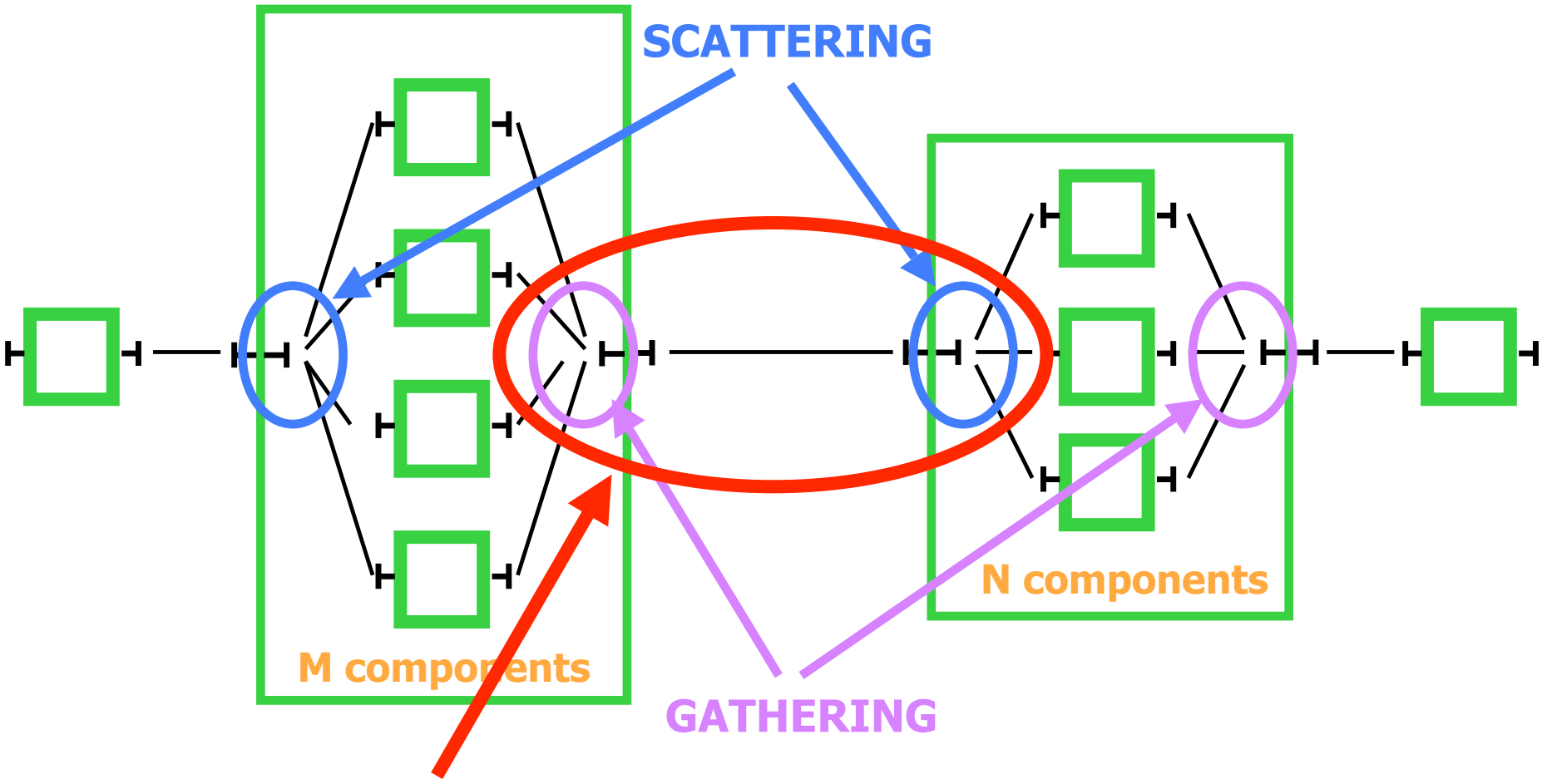
Perspectives : packaging

- Archives of components (sub-systems)
  - ADL
  - Classes
  - Native codes
- Composition of :
  - components
  - virtual nodes

Perspective : legacy code wrappers



Perspective : MxN communications



Conclusion

- Fractal model a viable candidate for a Grid component model
  - Simple
  - Extensible
  - Powerful
- A part of a framework for Grid / distributed components

(http://proactive.objectweb.org)

Thank you !