A Layered Model for Communicating Hierarchical Components

University of Westminster

- ✍ A component Model for the Grid:
  Hierarchy and Asynchrony
- ✍ A layered organisation of components
- ✓✍ Conclusion and Perspectives

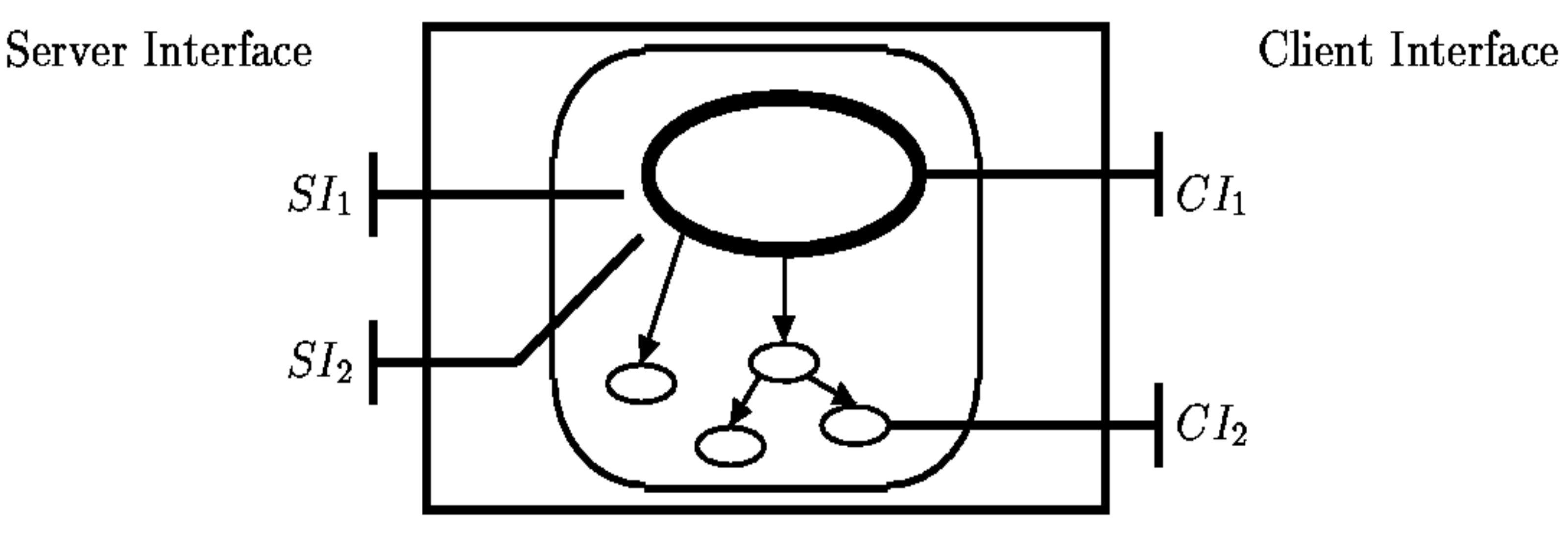
Ludovic Henrio

Context (previous works)

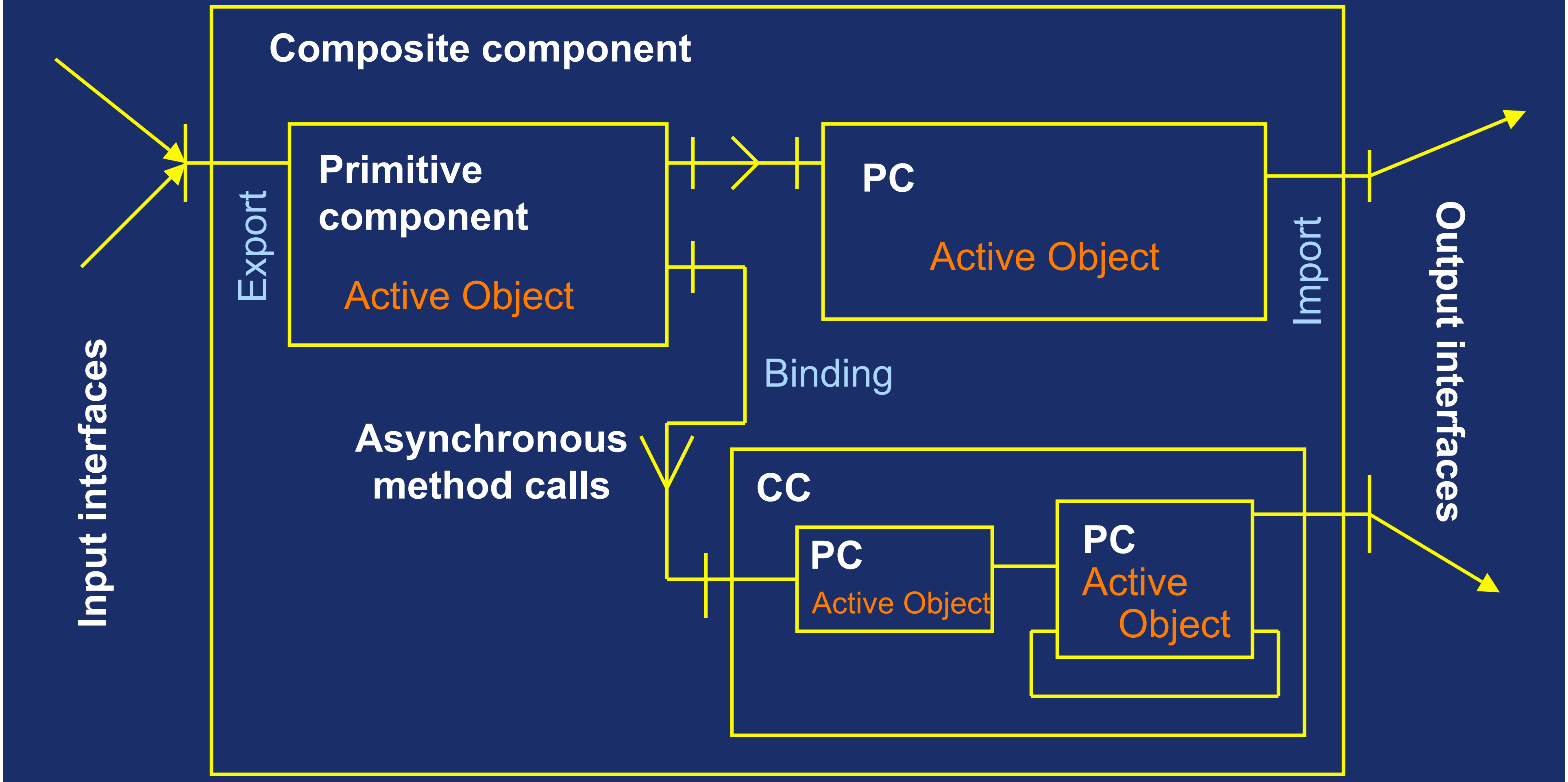
- Fractal: a component model specification
- An implementation in ProActive
  - Hierarchical composition
  - Asynchronous, distributed components
  - Non-functional aspects and lifecycle
- Formal aspects
  - Kell calculus → component control (passivation)
  - ASP components → Hierarchical aspects and deterministic components

Components from ASP Terms: Primitive Components

- Server Interface = potential service
- Client Interface = reference to an active object

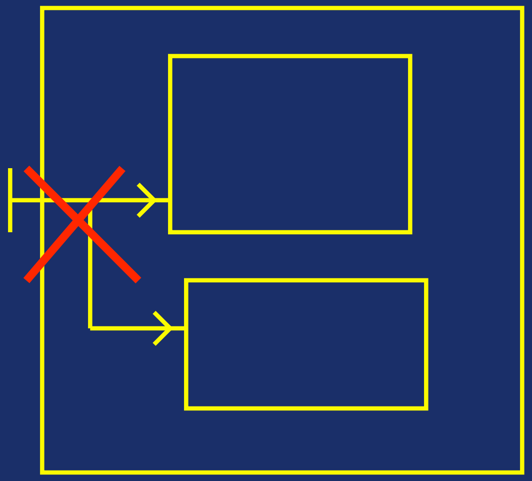


Hierarchical Composition

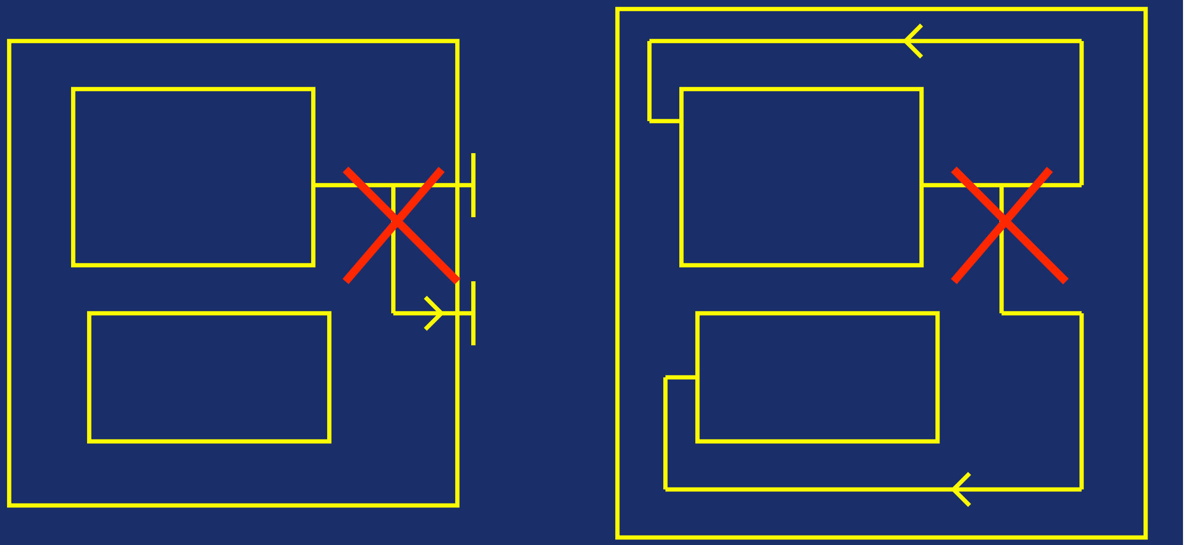


Invalid composition

Interface exported twice

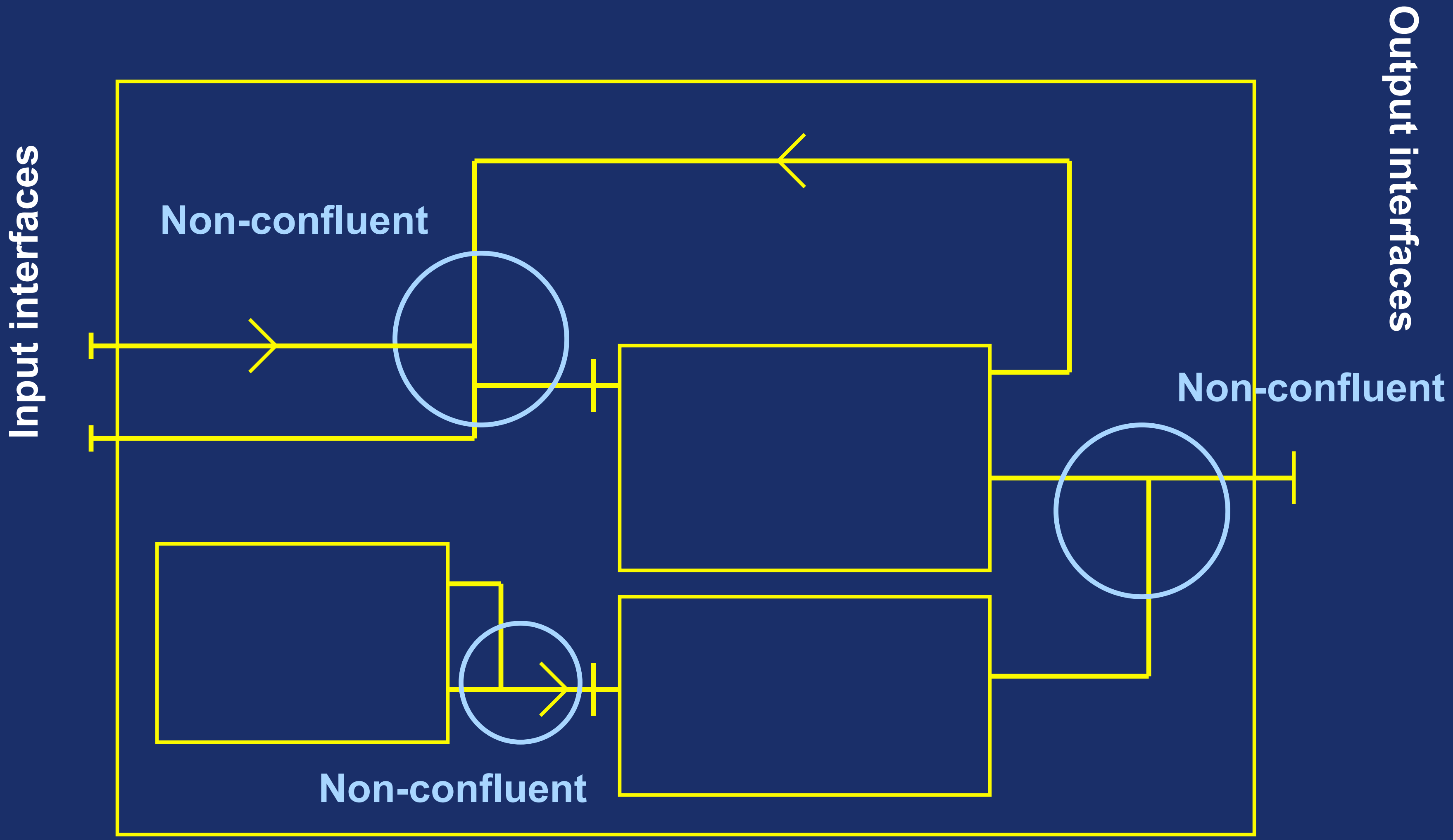


Output plugged twice



Except with group communication …

Valid Compositions



Deterministic Components

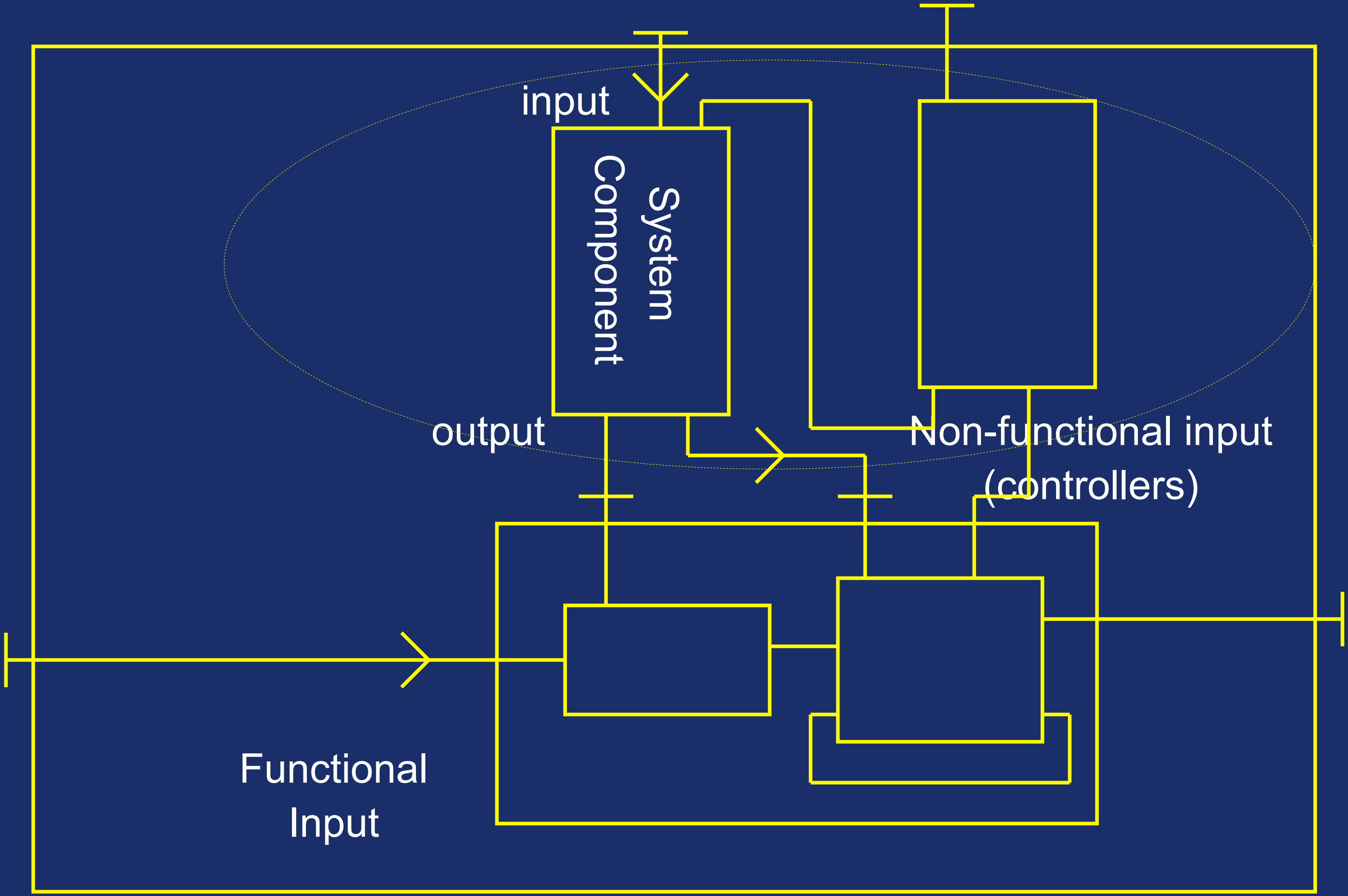
- Specification of deterministic components:
  - Deterministic primitive components
  - Deterministic composition of components:

    Based on a one-to-one mapping from Server to client interface

- Semantics as a translation to ASP

**Components provide
a convenient abstraction
for statically ensuring determinism**

A Layered View



Composition Rules: Components

- System Components provide / trigger non-functional services

- System components are part of the component platform

- System components can themselves be managed by another layer of components …

Composition Rules: Behaviour

- Effects of system component actions must not modify the functional semantics

- Composition rules between non-functional and functional aspects: Transparency and independence
  - (functional) programmer point of view
  - Better adaptativity

Conclusion

- Organize components and interfaces by layers
  - Functional aspects are addressed by the layer
  - Non-functional aspects are managed by the higher layer
- A layered model:

  Same composition rules between the above layers
- Define composition rules between layers for better transparency and adaptativity

➡ Provide design methodology and link between component platform and applications

Perspectives (ongoing work)

- Show application to existing (or envisioned) features / platforms
  - ProActive Components
  - Grid platforms and Grid management aspect

    Reconfiguration, adaptation, fault tolerance, migration, …

- Loosen strict independence between layers and propose new composition rules
  - Compromise between transparency and expressivity
  - Which semantic properties can still be ensured ?