Action Concertée Incitative
[ACI]
Globalisation des Ressources
Informatiques et des Données
[GRID]

Resource Management for Parallel Adaptive Components

Luc Courtrai, Frédéric Guidec,
Nicolas Le Sommer, Yves Mahéo

Valoria, Université de Bretagne-Sud

Context and objectives

- Research domain : Grid Computing
  - Applications exploiting one or several distributed platforms
- Target platforms (~ clusters)
  - more and more heterogeneous (hardware and network)
  - often non dedicated (*e.g.* workstations in a fast LAN $\rightarrow$ multiple applications and users)
- Approach
  - Complexity and reusabilty : components
  - Clusters : parallelism
  - Heterogeneity : adaptation

Parallel adaptive components

- Parallel component
  - Component ≡ deployment unit
    - a component is deployed on a distributed platform
    - a component involves several activities running in parallel
- Adaptive component
  - Adaptation…
    - at deployment time
    - at run time
  - … according to the environment
    - Environment ≡ set of resources whose state can be observed at run time

Research directions

- Objectives
  - Definition of a basic model of parallel component
  - Modelling of the environment in terms of resources used (or usable) by a component
    - Definition of resource observation schemes
- Prototype
  - Java
  - Deployment platform for parallel adaptive components: Concerto

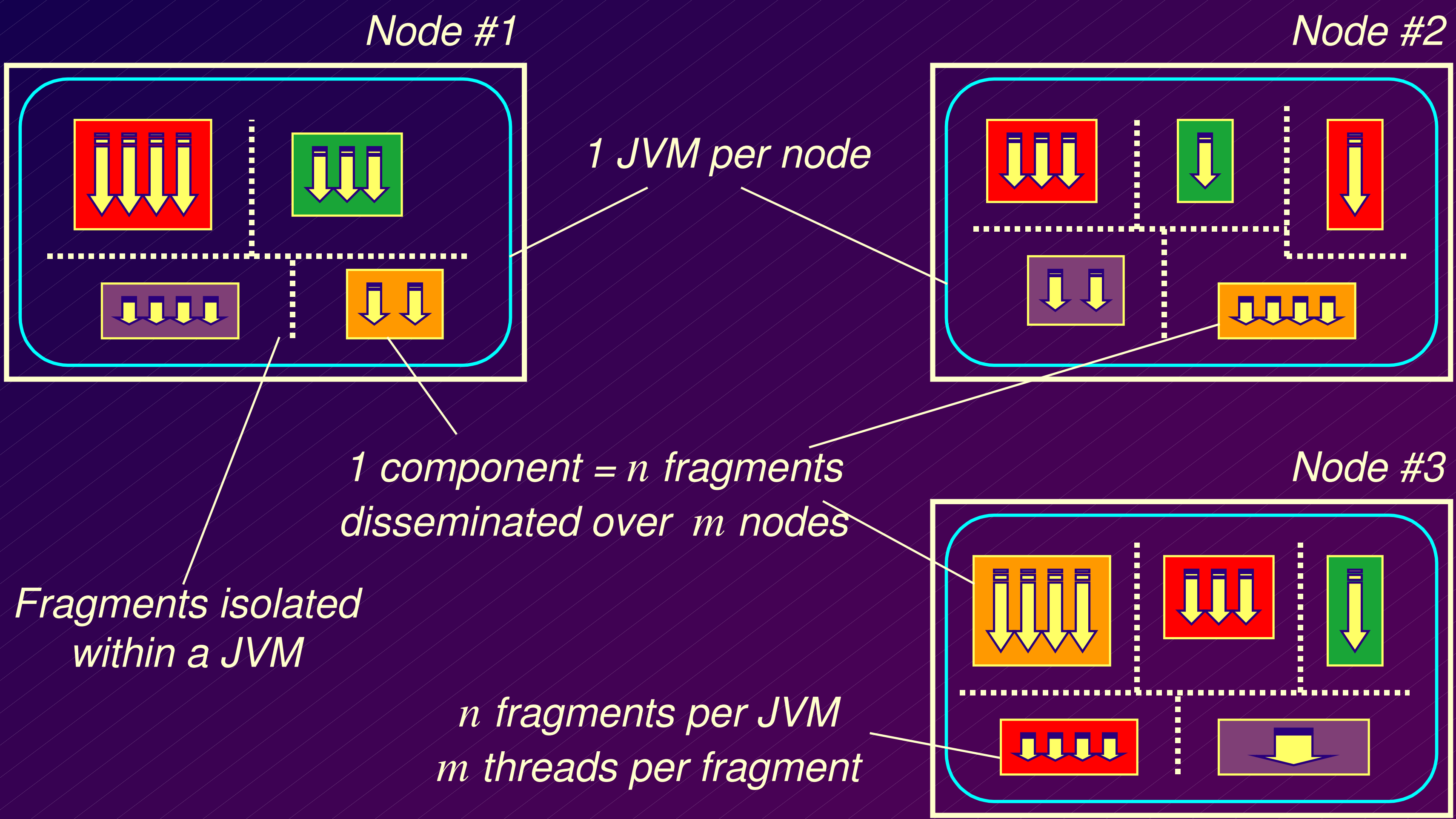
Parallel component

- Structure
  - Component ≡ set of threads
  - Grouping of threads in fragments
    - Fragment ≡ unit of placement
    - threads within the same fragment can share objects
    - threads of distinct fragments communicate in the usual ways (sockets, RMI, JMS,…)
- Naming
  - Identification as a resource
  - Name provided by the functional part (RMI, etc.)

Interfaces

- *Functional* interface
  - Under the programmer's responsability
    - RMI interface, client-server with TCP/UDP, etc.
- *Life-cycle* interface
  - Deployment, launching, termination…
- *Resource* interface
  - Component ≡ (observable) resource

Deployment of components on the Concerto platform



Deployment

- XML Descriptor
- Structure
  - Lists of fragments and threads
- Directives for the placement of fragments
  - Replication
  - Placement on a specific node
- Deployment constraints
  - Needs regarding the platform (*e.g. RMI registry*)

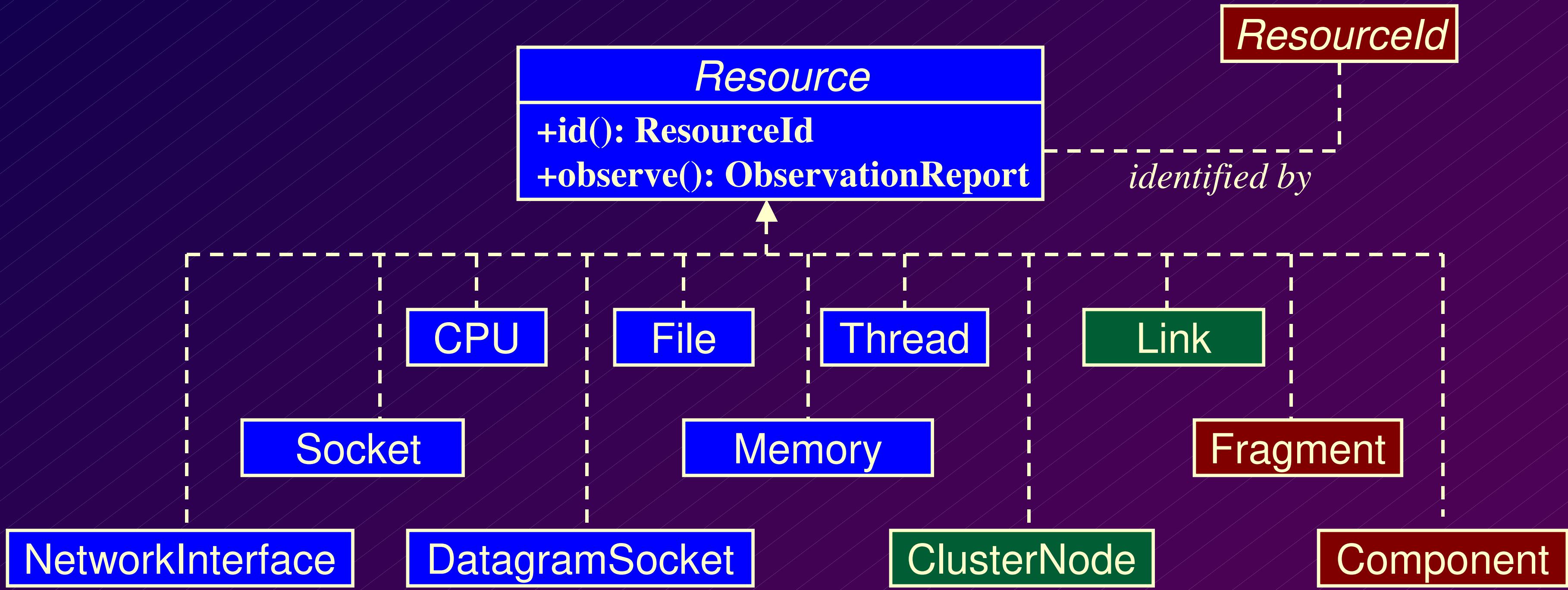
Resource management

- Services offered to components
  - Resource discovery
  - Observation of a resource's state
  - Notification on state changes
- Approach
  - Model and observe resources locally on each node
  - Take into account the distribution of resources
  - Model and manage "global" resources

Resource modelling

- "System" resources
  - Characterize the hardware platform
    - e.g. CPU, memory, swap, network interfaces, hard disks
- "Conceptual" resources
  - Characterize applicative resources
    - e.g. threads, sockets, directories, files
  - … or provide application-level services
    - e.g. RMI registry, communication library
  - … or take part into the deployment model
    - e.g. component, fragment

Resource modelling



- To each type of resource corresponds a Java class in Concerto
- New types of resources can be added in the system when needed

Distributed resources

- Every resource modelled as an object in Concerto …
  - ... has a unique identifier
    - One can designate a resource without ambiguity wherever it is located in the cluster
  - … is *observable*
    - One can consult the state of a resource, expressed as an observation report
    - A specific type of observation report corresponds to each type of resource

Resource manager

- An instance created on each node of the cluster
  - Knows all the local resources and can observe their state.
    - The resource manager present on a node is informed of every resource creation or destruction on this node.
- The different resource manager instances cooperate to provide a uniform service over the whole cluster
  - Identification, localization and consultation of the state of resources (local or distant)

Tracking resources

- Two kinds of patterns for tracking and observing resources selectively
  - Resource patterns
    - Selection on the type of resources or their attributes
  - Search patterns
    - Specification of the scope of the search
    - *e.g.* local search, limited to a given node, limited to neighbors, global

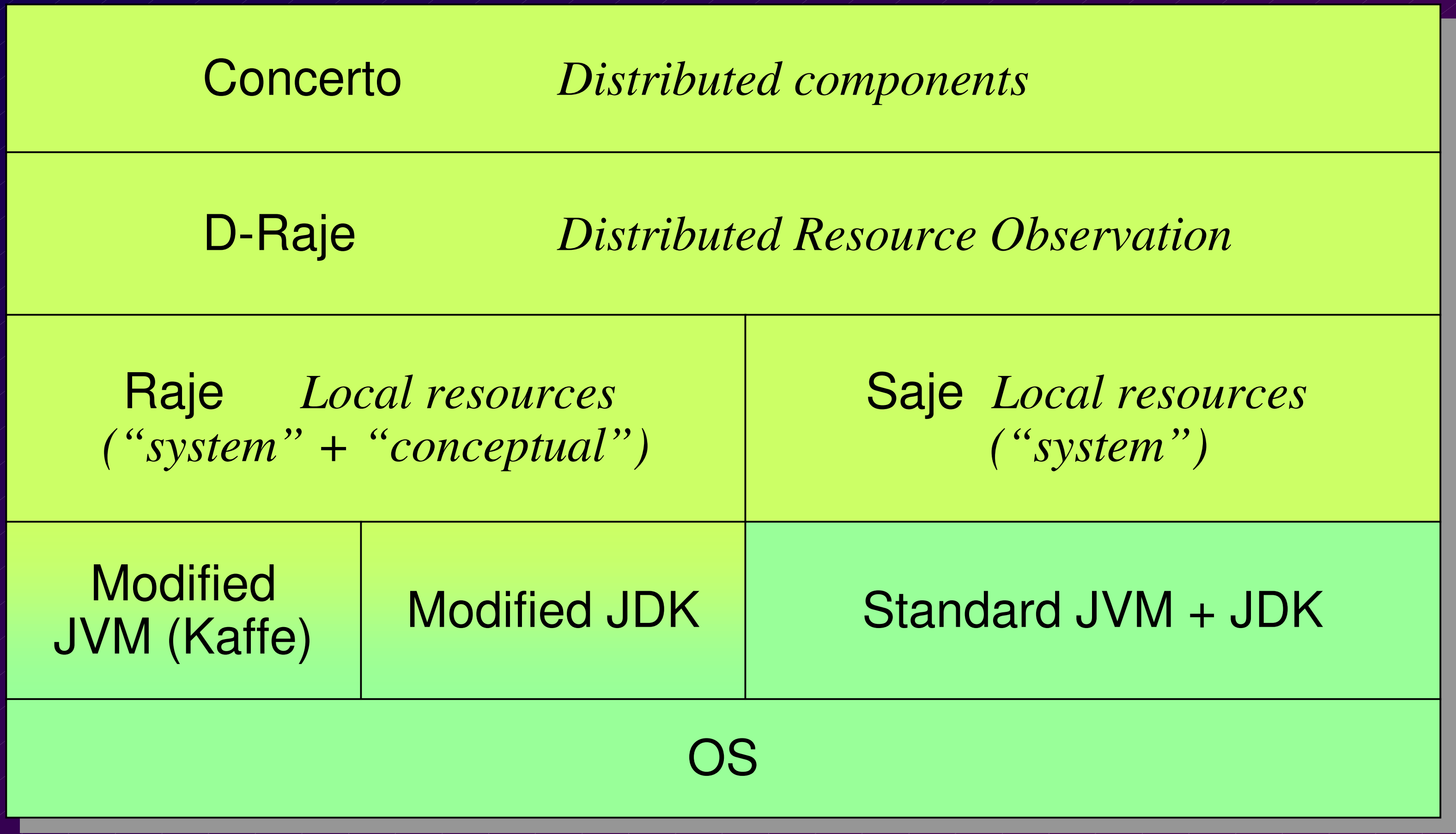
Example

```
// Acces to the  Resource Manager
ResourceManager m = ResourceManager.getManager();

// Pattern definitions
ResourcePattern SockPat =
    new SocketPattern(InetAddress.AnyAddress, "195.83.160/24"
                      PortRange.AnyPort, new PortRange(0,1023));

SearchPattern RamaPat = new LocalSearch("rama"));


// Search for a set of resources
Set RamaSockIds = m.getResourcesIds(RamaPat, SockPat);

// Observation of one of the resources
ResourceId resId = an element of the RamaSockIds set

ObservationReport report = m.getObservationReport(resId);
```

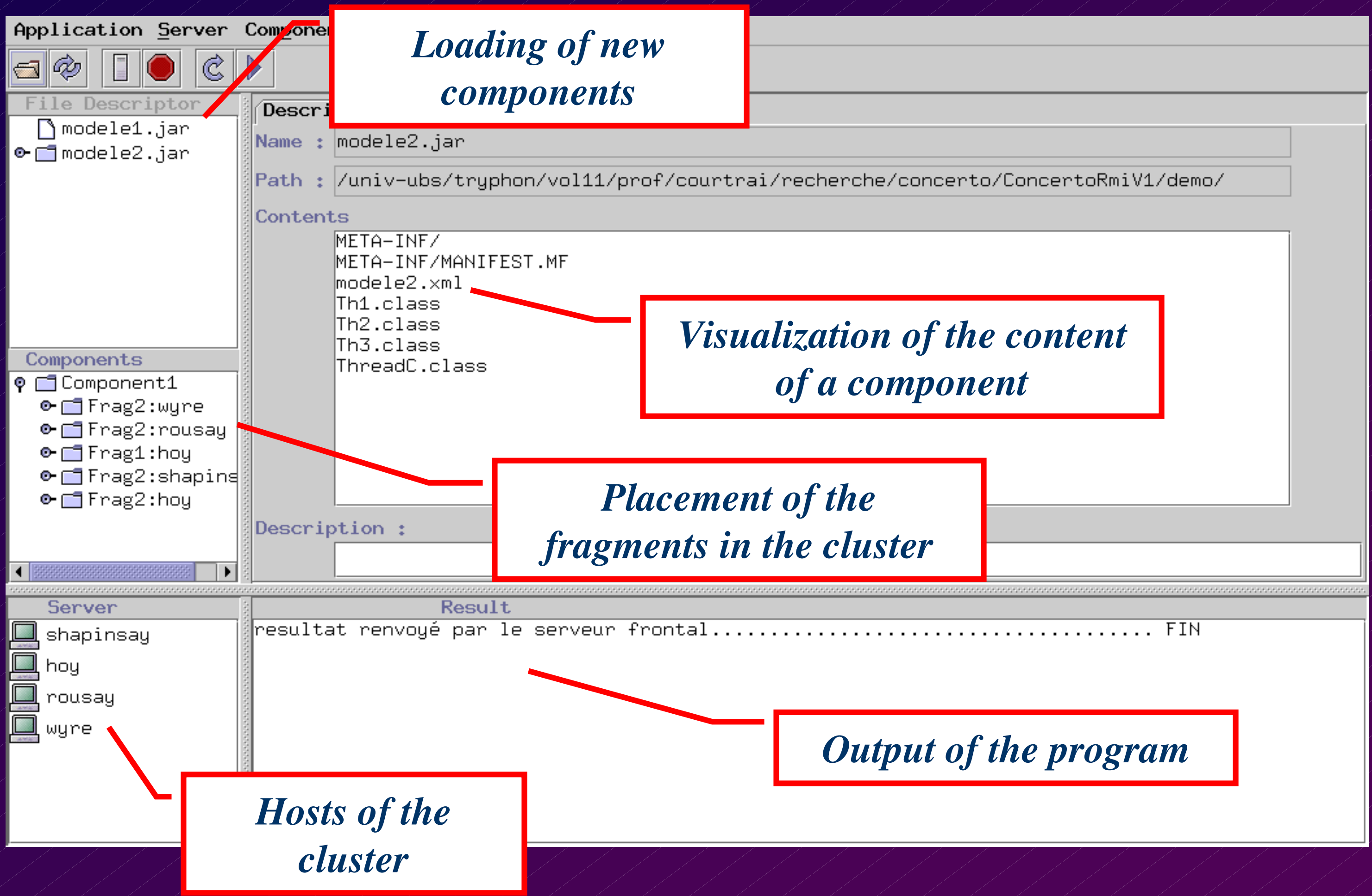
Other features

- Sporadic resources
  - Examples : network interface, USB device,...
  - For each type of resource:
    - Monitor polling the system and maintaing a list of resource objects
- Notification of events
  - specification of events pertaining to resources, based on patterns
  - atomic and composite events
  - local or distant monitoring

Architecture of the platform



Concerto Control Panel



Conclusion

- Concerto: a Java platform for the deployment of parallel components that can take advantage of resource information
- Basic component model
  - basis for more structured parallel components
- Management of resource information
  - basis for adaptation
  - external adaptation strategies to be integrated in the component model