Some requirements for a Mikado programming model

Jean-Bernard Stefani - INRIA

Remainder

- One objective of Mikado (WP1) : to define a distributed/mobile programming model
- Deliverable planned for 9/02 on requirements for the Mikado programming model

Outline

- Target
- General
- Domains
- Mobility & Reconfiguration
- Communications
- Failures & Atomicity
- Resources

Target

- Programming model vs descriptive model
  - descriptive model -> semantical constructs
  - a protean model for including descriptive features ?
- System-level programming
  - routers, firewalls, packet filters, protocols, event handlers, resource managers, etc.
- Application-level programming
  - transparency, abstractions (failures, communications, concurrency, ...)

General

- Implementability
  - no hidden distributed consensus (no difficult distributed problem hidden in basic constructs)
  - direct mapping on today's WAN technology
- Low-level
  - no hidden costs (communication, routing, execution, etc)
  - available trade-offs (consistency vs performance, etc)
  - no implied policy (resource management, security, etc)
  - explicit locations

Domains

- Spatial partitioning of computations
  - administrative & physical boundaries (e.g. naming, security, failures, communication)
  - mobility & migration (hardware & software objects)
  - groups & clusters in computation, communication
  - hierarchies
  - programming & descriptive aspects

Mobility & Reconfiguration

- Moving between locations / domains
- Late binding
- Adding, removing, replacing computational components
- Instantiating, activating, suspending, passivating components
- Connector & communication components

Communications

- Capturing low-level communication primitives
  - asynchronous point-to-point
  - asynchronous broadcast
  - asynchronous multicast
- Protean structure for accommodating different communication semantics

Failures & atomicity

- Failure detectors
  - time and clocks
  - descriptive & programming
- Failure modes
  - descriptive & programming
  - confinement zones
  - errors & exceptions
- Atomicity
  - assumptions & observations
  - transactions: abstractions, concurrency control, recovery, refinement, persistence

Resources

- Capturing resource dependencies
  - processing, communication, memory, domains
  - sharing & multiplexing (spatial, temporal)
  - descriptive & programming
- Resource accounting
  - principals & contexts
  - run-time dependencies