Some Mobile Middleware Projects

Gian Pietro Picco

Dipartimento di Elettronica e Informazione
Politecnico di Milano, Italy

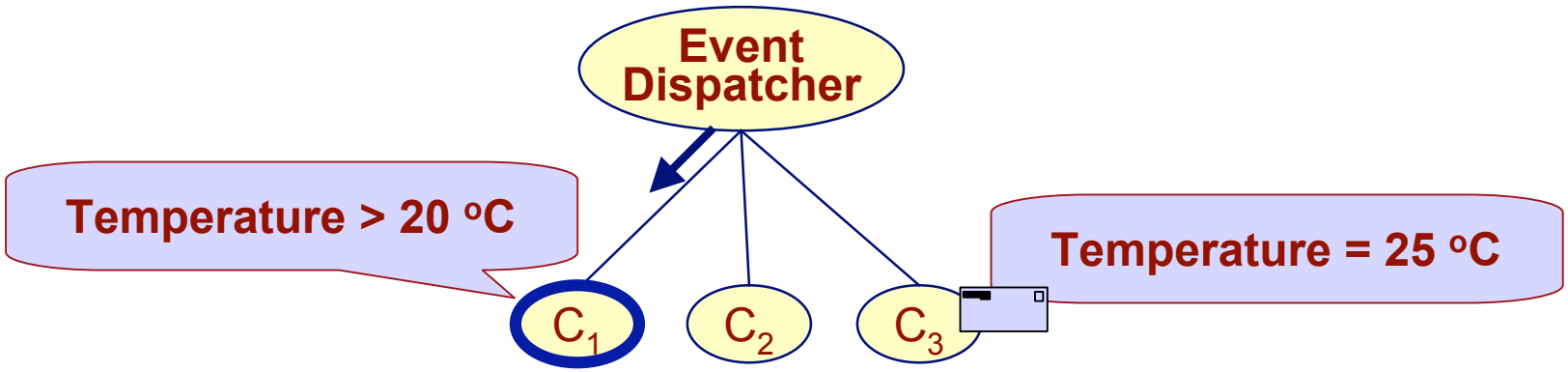
picco@elet.polimi.it

http://www.elet.polimi.it/~picco

Politecnico di Milano

Publish-Subscribe

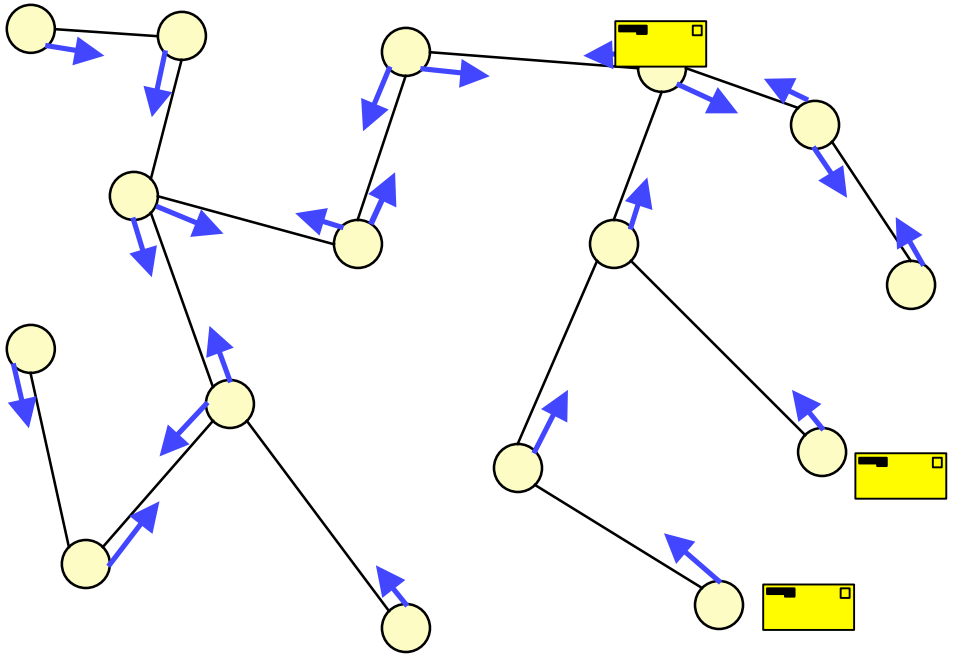
- Application components can publish asynchronous ***event notifications***, and/or declare their interest in event classes by issuing a ***subscription***
  - Extremely simple API: only two primitives (**`publish`**, **`subscribe`**)
- Subscriptions are collected by an ***event dispatcher*** component, responsible for routing events to all matching subscribers
  - Can be centralized or distributed
- Communication is transiently asynchronous, implicit, multipoint
- High degree of decoupling among components
  ➔ easy to add and remove components
  ➔ appropriate for dynamic environments



Politecnico
di Milano

A Distributed Event Dispatcher

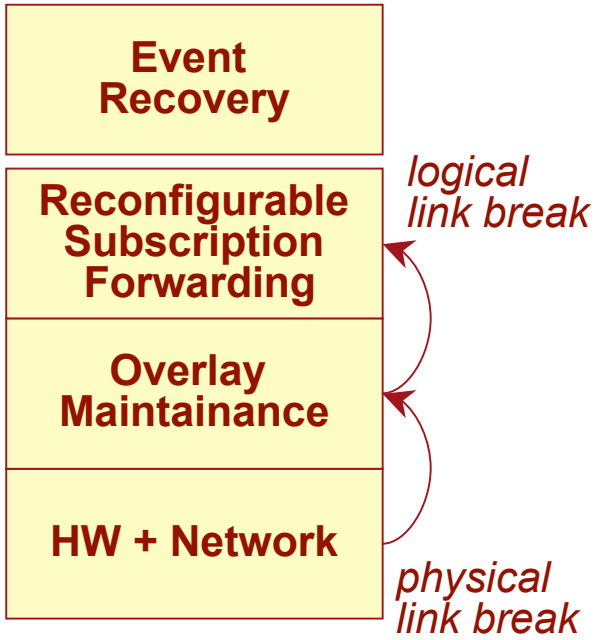
- Each dispatcher forwards subscriptions to neighbors
  - Subscriptions are never sent twice over the same link
- Events follow the routes laid by subscriptions
- Optimizations may exploit coverage relationships
  - E.g., **"Distributed *"** > **"Distributed systems"**
  - Fusion, subsumption, summarization



Politecnico di Milano

Dealing with Topological Reconfiguration

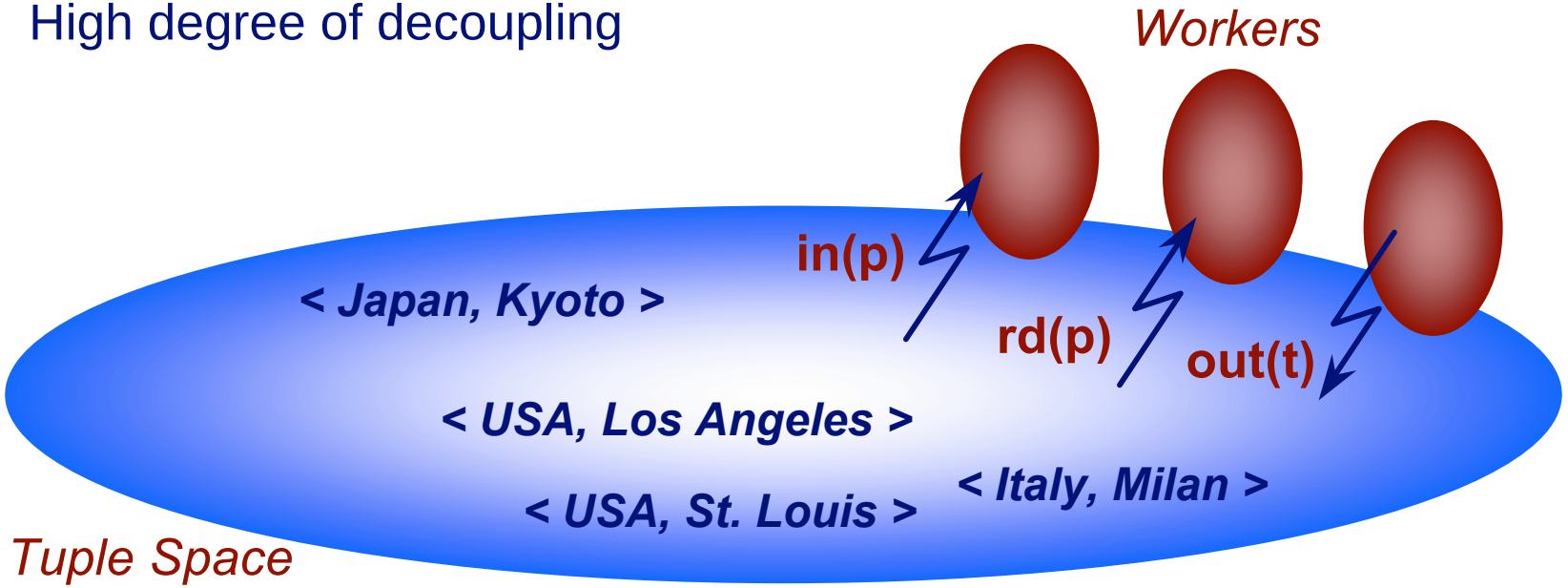
- Topological reconfiguration:
  - Physical (e.g., wireless links)
  - Logical (e.g., P2P networks, admin)
- Involves solving three problems:
  - ***Reconstruct the overlay network***
    - In some environments, may be under control of some user (e.g., sysadm)
    - Depends on the deployment scenario (e.g., fixed vs. mobile)
  - ***Rearrange the subscription tables***
    - Content-based routing defines a new problem, for which the existing subject-based or multicast solutions are no longer applicable
  - ***Minimize event loss***
    - E.g., using epidemic algorithms



- Several papers about each layer
- A running system, REDS (Reconfigurable Dispatching System), available as open source at **`zeus.elet.polimi.it/reds`**

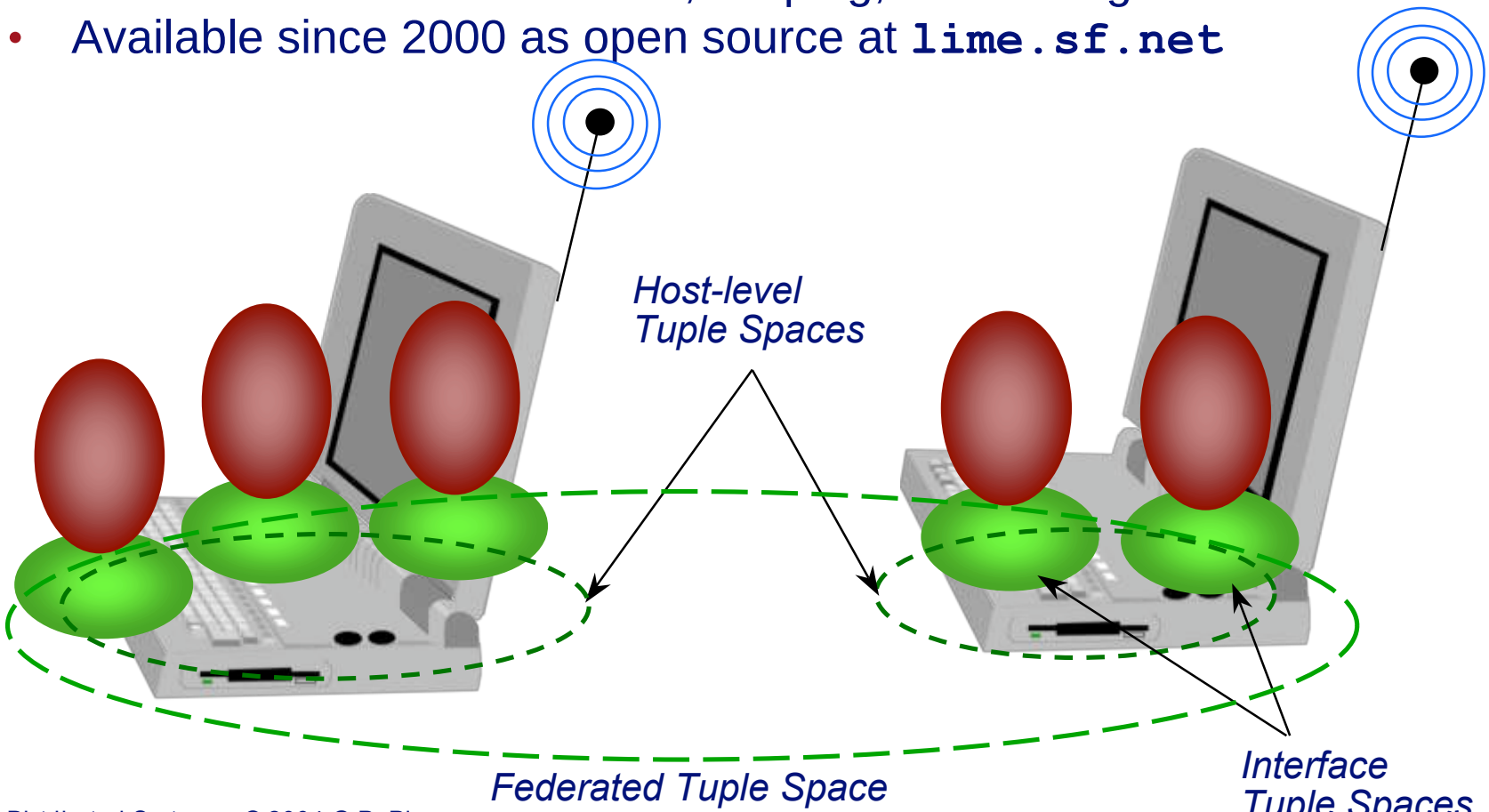
Linda and Tuple Spaces

- Data sharing model proposed in the 80s by Carriero and Gelernter, mostly used for parallel computation
  - Data is contained in ordered sequences of typed fields (***tuples***)
  - Tuples are stored in a persistent, global shared space (***tuple space***)
  - Synchronous and asynchronous (probe) primitives
- Recently revitalized in the context of distributed computing
  - E.g., IBM TSpaces, Sun JavaSpaces, GigaSpaces
- Communication is persistent, implicit, content-based, generative
- High degree of decoupling



Lime: Linda in a Mobile Environment

- Goal: support for (physical and logical) mobility
- Key concept: transiently shared tuple spaces
  - Connectivity determines the accessible portion of the tuple space
  - Engagement and disengagement reconcile the data
- Additional features: reactions, scoping, addressing
- Available since 2000 as open source at **`lime.sf.net`**



TinyLime