Resilient VO services on grid nodes

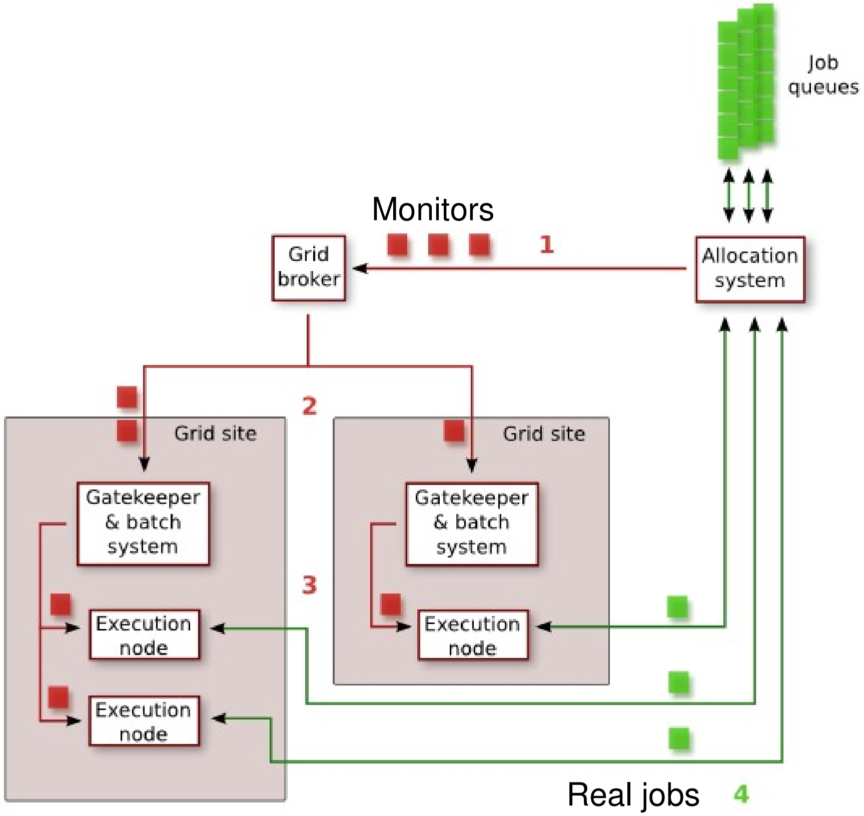
CERN openlab II monthly meeting

13 November 2007

Xavier Gréhant

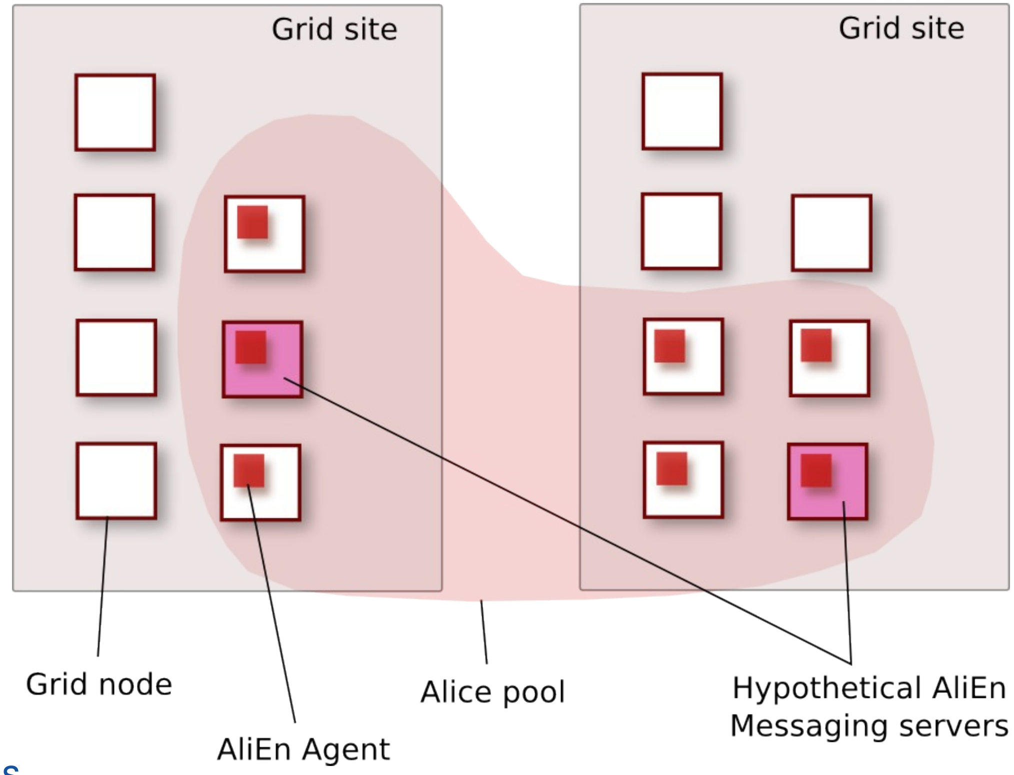
Motivation

- Scientific collaborations (Virtual Organizations - VO)
  - Submit monitors to the grid
  - Connect directly with these monitors
  - Run VO services on grid nodes
- Constraint:
  - Grid nodes come and go
  - VO services need resiliency
- Replacement for *VO-Box*
  - Dedicated machine on every grid site
  - Direct access to VO admins
  - “Dirty unsafe hack”



Election mechanism

- Example: AliEn
  - Alice Environment (alien.cern.ch)
  - Infiltrates *Agents* on grid nodes
  - Connected in a VO-wide allocation system
- Example of a future service:
  - A messaging server
  - To connect grid nodes to Alice central job queue
  - Always one per grid site
- Predrag Buncic's idea:
  - Election mechanism by agents
  - Electing agents to carry on services



Proof of concept: *SmartCitizens*

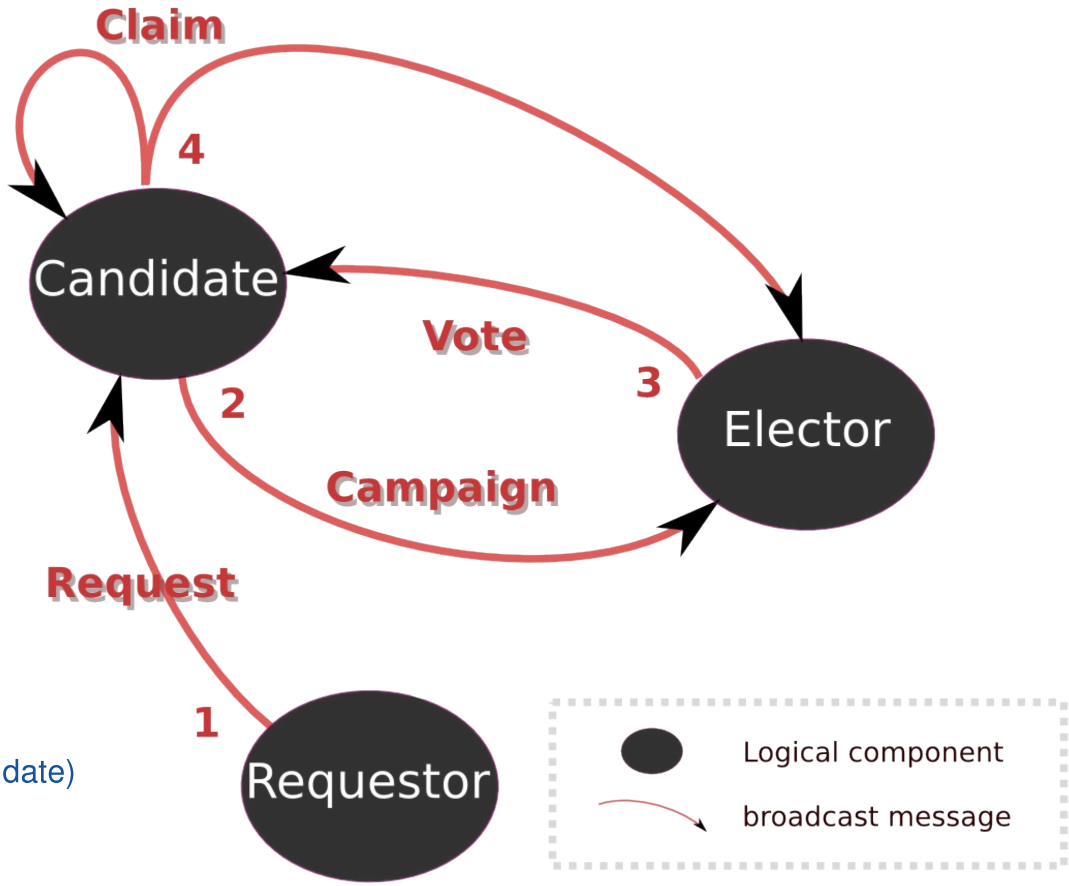
- We developed a proof of concept: *SmartCitizens*
  - Based on message passing
  - Between peers (logical components)
  - Deployable (*SmartFrog* components - smartfrog.org)

- Typical leader election problem in mobile networks
  - We present our algorithm in the following, simple and easy
  - For more advanced algorithms:
    - Learn about *Consensus* on Wikipedia
    - Read about the *Paxos Algorithm* (http://groups.csail.mit.edu/tds/paxos.html)

(Thx O. Pernet for these links.)

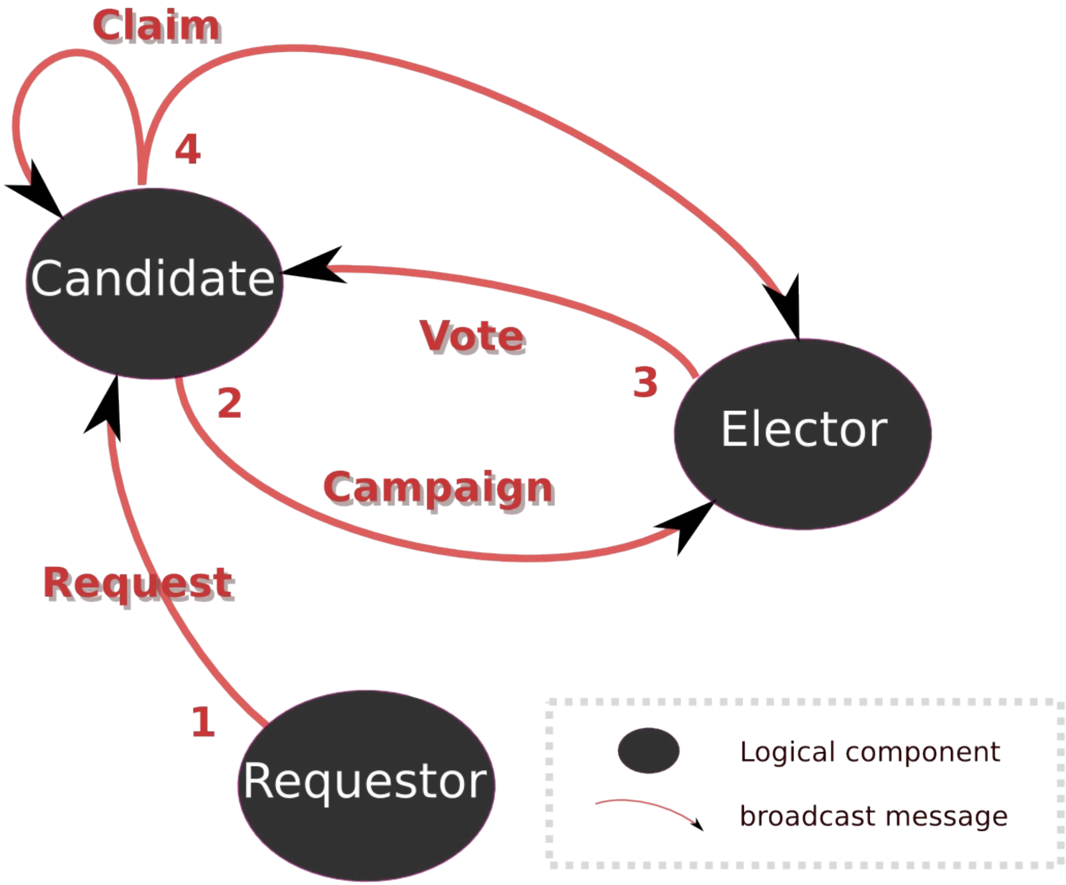
SmartCitizens: Logical components

- Request:
  - Request new applications
- Campaign:
  - Send application
  - With true self-appreciation
- Vote:
  - Send personal preference
  - Takes into account:
    - Candidates' appreciations
    - Local info (e.g. distance to candidate)
- Claim:
  - Say if elected



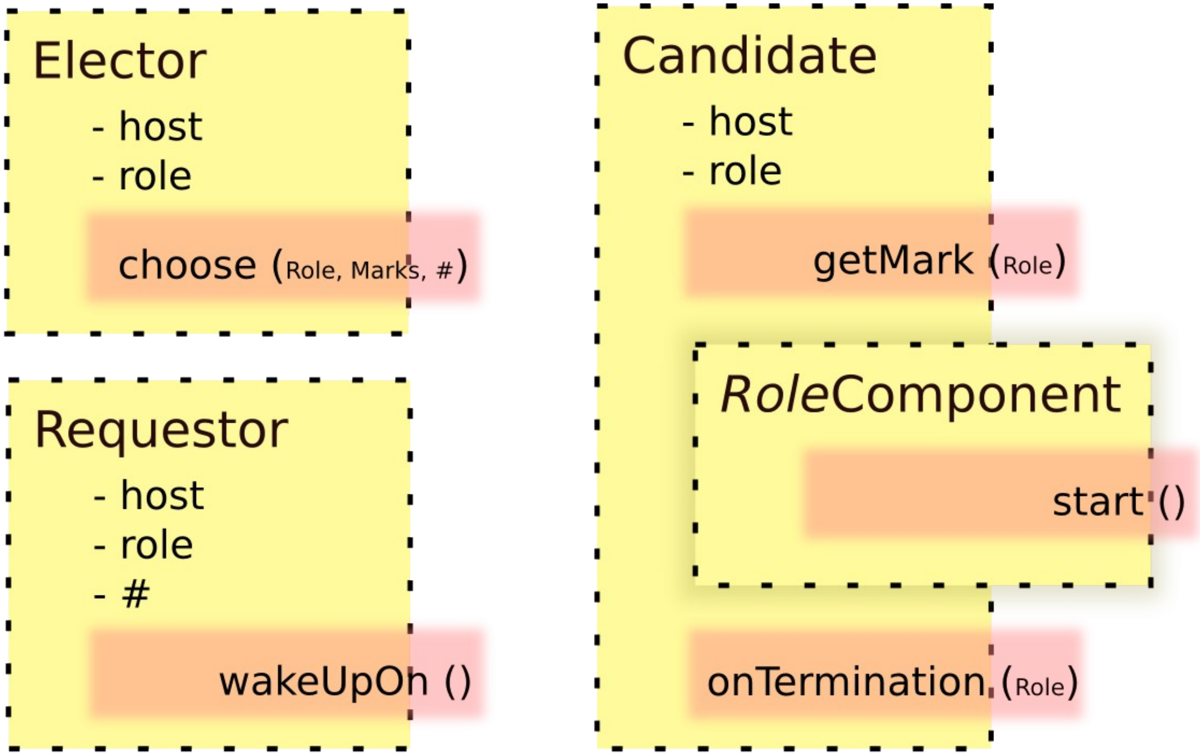
SmartCitizens: Logical components

- Request:
  - *Role*
  - # of wanted representants: *NR*
- Campaign:
  - If *Role* matches: *Mark*
  - # of wanted representants: *NR*
- Vote:
  - *NR* preferred candidates
- Claim:
  - If elected



SmartCitizens: Deployable components (SmartFrog)

- Candidate:
  - Able to launch real component
  - Starts component only if elected
  - May launch a requestor on termination
- All components:
  - Placed anywhere
  - Role-specific
  - Own decision methods

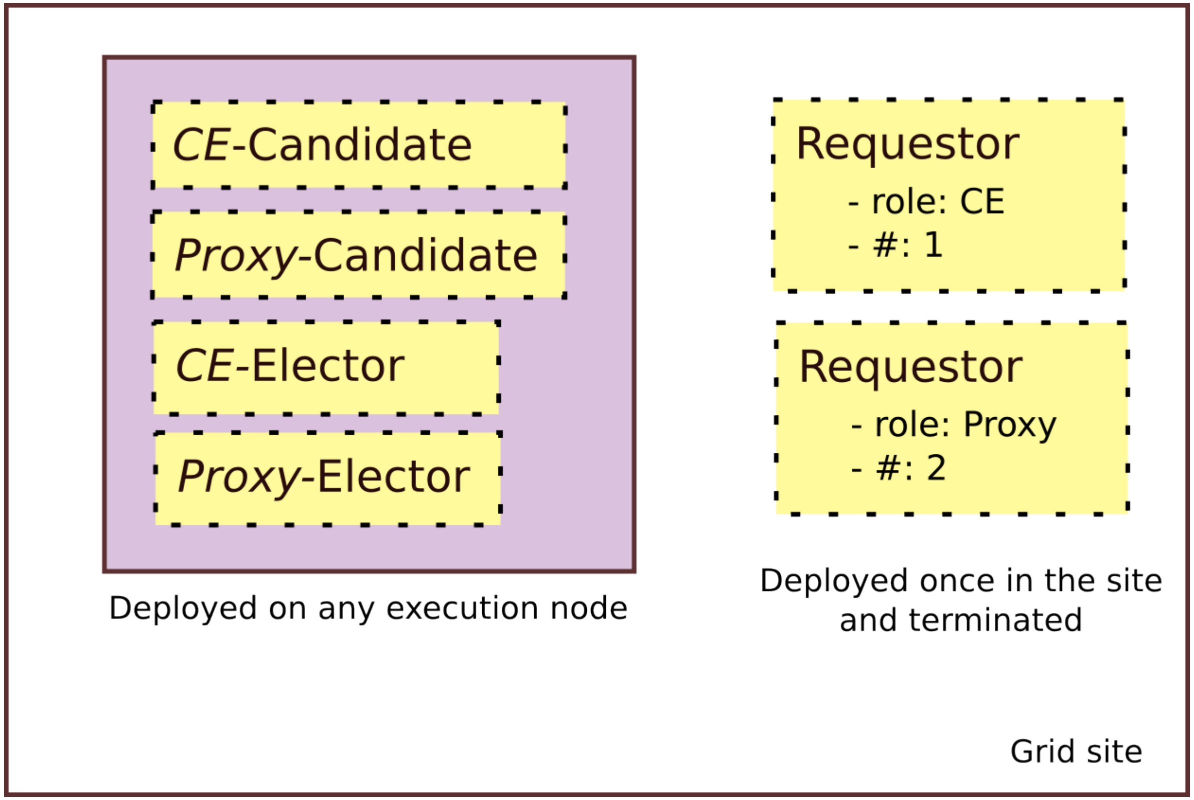


“Place anyway, deploy if elected”

Example deployment

- Candidates and Electors
  - On each node
  - For each role
- Requestors
  - One per role once
  - Then launched by terminating candidates

- 1 CE and 2 proxys always present on the site.



Status

- Proof of concept
  - Working
  - Basic: designed to handle unexpected but 'nice' events
    - What if almost all nodes die together?
    - What if nodes get isolated?
    - ...
- Distributed with SmartDomains (Xen domains mgt)
  - Useful for live-migration of virtual machines
  - However *SmartCitizens* is very general
    - Placement is static with SmartFrog
    - New strategy: “Place anyway; deploy if elected.”
    - New components in SmartFrog?

Summary: main ideas

- Grid execution nodes may host anything (not only jobs)
- *VO pools* dynamically partition *grid nodes*
- “Local services” for VO $\in$ (*VO pool* $\cap$ *grid site*)
  <=> mobile networks
- *Place anyway* (everywhere)*; deploy if elected*
- Implemented: *SmartCitizens*