Actors

Paul Preißner

Fakultät für Informatik
Technische Universität München

May 18th 2017

Actors - “A Model of Concurrent Computation in Distributed Systems“

- paradigm: ”everything” is an actor (thread, process, core, socket, node, system, ...) $\rightarrow$ one actor encapsulates one computation unit
- an actor may send messages to actors it knows by name
- an (idling) actor receiving a message will accept it and execute the computation defined within, resulting in the possible actions:
  - sending new messages
  - creating new actors
  - updating its local state
- an actor can only influence its own local state

Actors - example

Actors

- Actor semantics have three main properties
  - Encapsulation & atomicity (actors don't share state, process one message at a time)
  - Fairness (every actor makes progress, every message delivered eventually)
  - Location transparency (physical location not bound to identifier, hidden migration)
- in reality, some aspects aren't implemented faithfully (for efficiency, complexity)
- concerns about scalability & performance

Actors

- Controversy: unbounded nondeterminism (unbounded delay yet guarantee of service)
- Many (actively) supported languages
- History: C. Hewitt et al. '73 $\rightarrow$ W. Clinger '81 $\rightarrow$ G. Agha '85, MIT Message Passing Semantics Group, Caltech, etc.
- Little use around millennium, recent resurgence due to strong relevance to distributed/cloud computing (e.g. Twitter systems scalability)

Focus - Expansion

- Focus: current usage in large scale computing
- Expansion: comparison to other models/paradigms of concurrent computation

Q&A