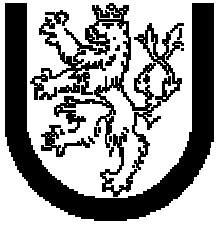
Inside Active Network Node

Tomáš Koutný



University of West Bohemia
Faculty of Applied Sciences
Department of Computer Science and Engineering
Plzeň, Czech Republic

Inside Active Network Node

- Slight Introduction into Active Networking
- Active Node in General
- Application & Capsule State Transition Diagrams
- Bees Application Architecture
- How about Distributed Computing
- The Grade Project
- Live Show
- Next Work
- References

Slight Introduction into Active Networking
Classic IP tracert

- Uses standardized ICMP protocol
  =>any change is virtually impossible
- Application subsequently sends packets with increasing TTL
- Route is discovered upon received ICMP-error packets
- Let us have three nodes making triangle: A, B & C
- Possible to discover routes such as A-B from A
- Impossible to discover routes such as A-C from B

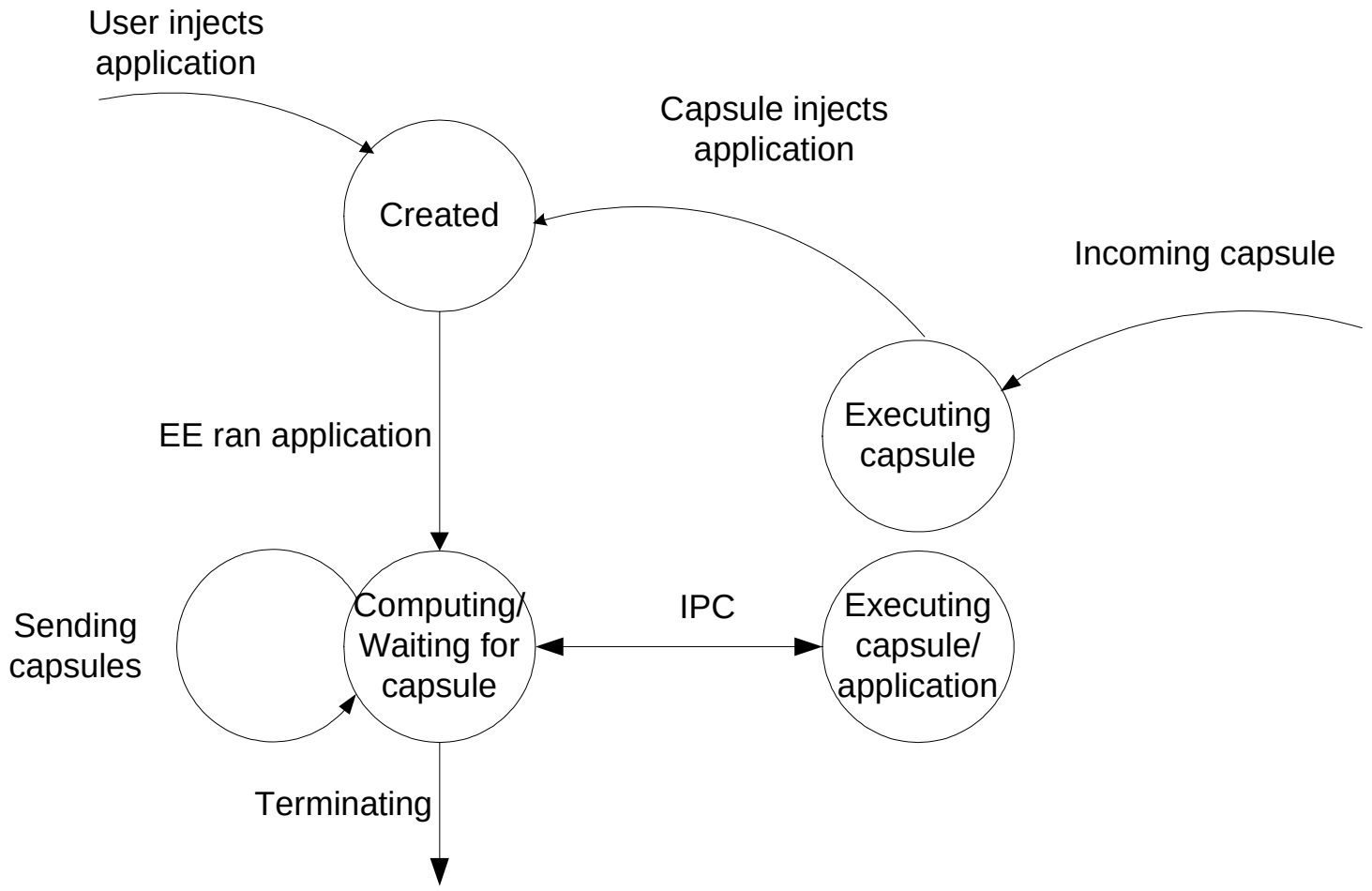
Slight Introduction into Active Networking
Active tracert

- Active application injects just one capsule
- Capsule is a smart packet = code + data
- There's a code associated with capsule, which is run on every node visited by capsule
- Code adds address of visited node into data load while heading for destination
- On return it passes data load to application as a report
- No need for standardization
  => Several versions can coexist
  to enable e.g. tracing of A-C from B

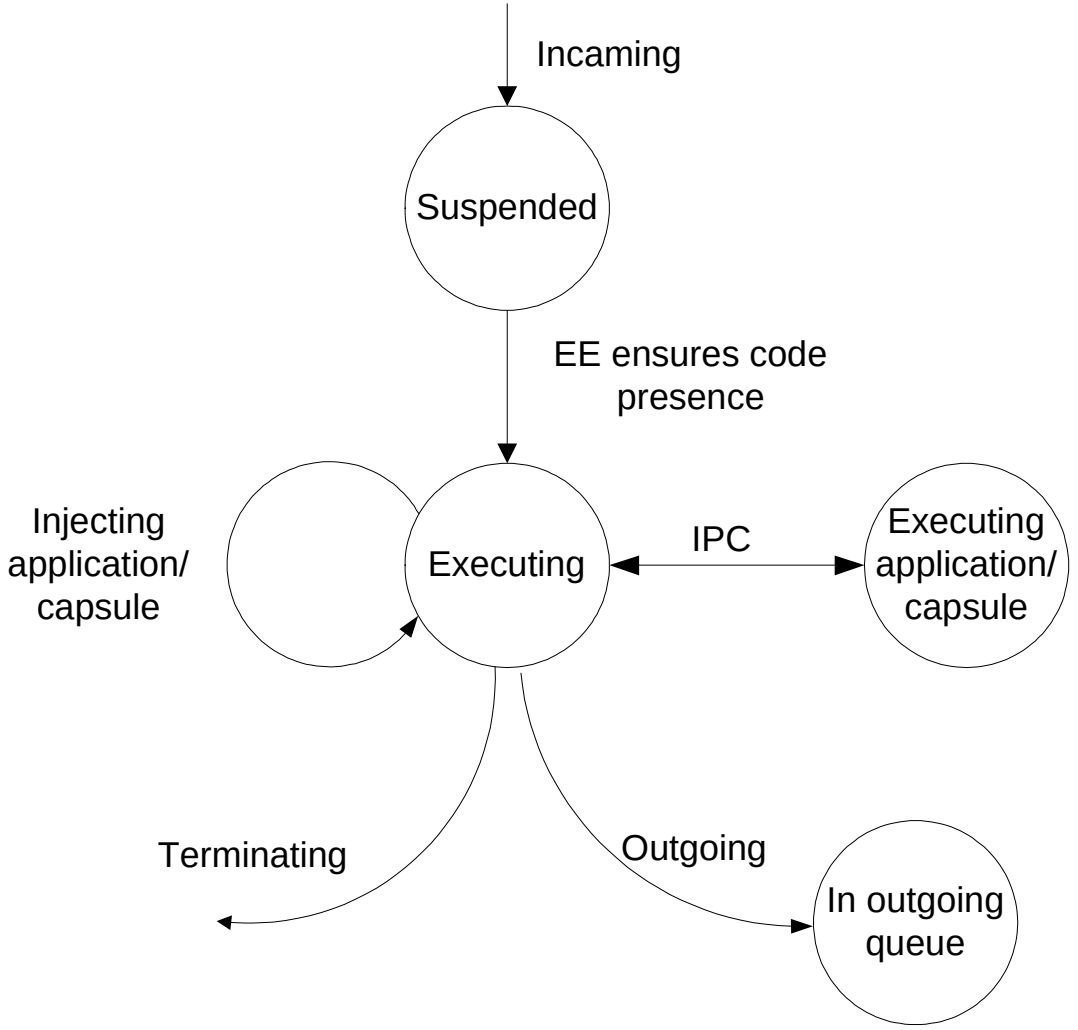
Active Node in General

- Underlying OS providing HW abstraction
- On top of OS:
  - Incoming queue
  - Outgoing queue
  - Router for nodes with multiple connections
  - Built-in code distributor – the only one standardized app
  - One or more execution environments (EE)
  - Kernel managing above items and shared memory
- Each EE can support different type of capsule code
- General-programming EEs: ANTS, Bees, Magician
- Specific-programming EEs: PLAN, NetScript, CANEs

Application State Transition Diagram



Capsule State Transition Diagram



Bees Application Architecture

- Bees originates from ANTS; Java
- ANTS application may use several protocols
- Each protocol is composed from m classes forming n capsules
- Application injects capsules, capsules returns to application on their own initiative
- Bees provides new enhancements, e.g. to compose protocol with a number of “helper” protocols
- Bees provides better security
- Bees is the next evolution of ANTS

How about Distributed Computing

- Active networks were not designed to let capsules execute computation-intensive tasks
- Capsules data load has limited size; 16kB for ANTS
- It would be necessary to differently manage resource consumption restrictions
- What about accessing file system of the node?
- Tested EEs allowing general programming did not prove themselves to be ready for heavy load at this time.
- Support of e.g. semaphores and barriers must be added as protocol implementations – they do not exist yet

The Grade Project

- Designed to study & enable execution of computation-intensive tasks in active networking style
- Capsule's data load is not limited
- Would bring synchronization primitives, load-balancing engine & decentralized communication for self-migrating processes
- Capsule still remains in the role of carryall for application
- Active application has the role of process of classic distributed application
- Applications are free to implement whatever they want
- It follows concept known from ANTS/Bees

The Grade Project Synchronization Primitives

- Provided by external libraries by default
- Application's own ones with specific optimizations
- Blocking routine
  - Works with capsules of synchronization protocol
  - Routine exits under given conditions
  - Implementation of well-known procedures such as distributed leader election
  - Data are delivered to application asynchronously – event driven
    =>application can wait for an object
    =>supported by OS kernel
    =>no CPU utilization
  - tracert application synchronizes with its capsule via such wait

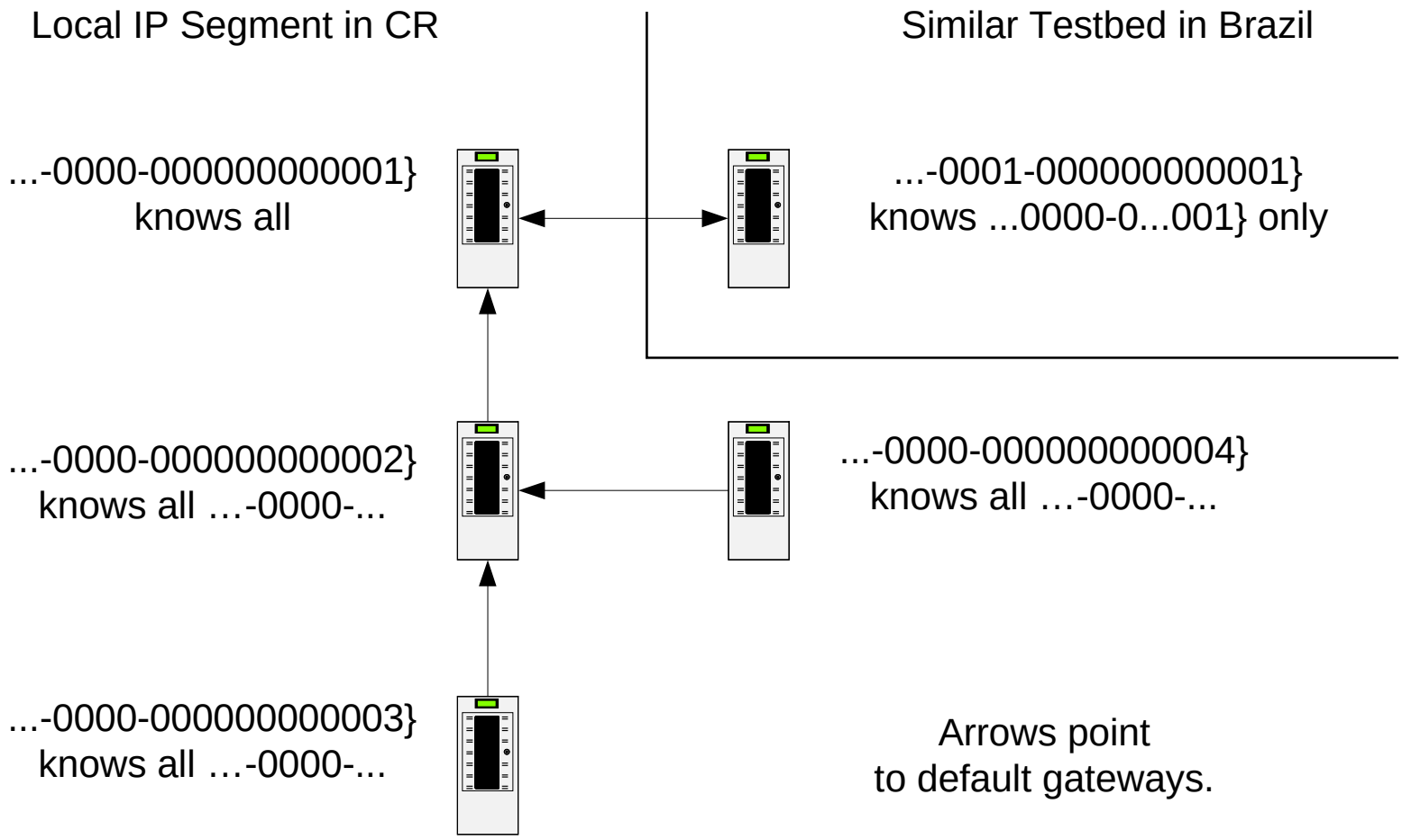
The Grade Project
Communication Over Nodes

- Applications can move themselves to another nodes
- Communication must be maintained
  - All applications keeps logical ring to assure entire addressability
  - Each application manages a list of other applications and nodes it exchanges data with
  - List's organization is based on data transmission rate
  - Node is treated as a special application, as files can be fetched from/stored to
  - Applications frequently communicating with each others forms so called nearly-ideal cluster; relationship discovery => nearly
  - Applications in such cluster notify themselves about their moves

The Grade Project
Pilot Application

- Parallel computation of prefixes
  (input: 1, 2, 3, 4, 5; output: 1, 3, 6, 10, 15)
- Will use gradient load-balancing method with topology discovery designed for environment such as AN/Grade
- Computation uses barriers for synchronization, pilot application would use modified barriers fitting exactly to the problem
- No central process/active application coordinating others
- No static assignment of processes/active applications to given nodes

Live Show


Local IP Segment in CR
Similar Testbed in Brazil
...-0000-000000000001}
knows all
...-0001-000000000001}
knows ...0000-0...001} only
...-0000-000000000002}
knows all …-0000-...
...-0000-000000000004}
knows all …-0000-...
...-0000-000000000003}
knows all …-0000-...
Arrows point
to default gateways.

Next Work

- Finish pilot application
- Make it AN compliant
- Introduce security measures
  => interpreted code is necessary
- Make some routines available in a form of libraries
- Address next problems such as e.g. routing, code caching, casting, etc.
- Enable use of Grade server for applications running outside

References

- M.I.T., University of Utah and the Flux Group
  ANTS, Bees, JNodeOS and JanosVM,
  http://www.cs.utah.edu/flux/janos/
- T. Koutný and J. Šafařík,
  ”Gradient method with topology discovery for load-balancing in Active Networks”,
  Proceedings of 11th Annual IEEE International Conference on the Engineering of Computer Based Systems, May 2004
- T. Koutný and J. Šafařík,
  “Maintaining Communication Channels for Migrating Processes in the Environment of Active Networks”,
  Submitted to PDCN 2005
- R. Chow and T. Johnson,
  "Distributed Operating Systems & Algorithms",
  Addison-Wesley Longman, Inc., 1997