Dynamic Reconfiguration of Network Applications and Middleware in the Bio-Networking Architecture

Jun Suzuki, Ph.D.
jsuzuki@ics.uci.edu
www.ics.uci.edu/~jsuzuki/
netresearch.ics.uci.edu/bionet/
Dept. of Information and Computer Science
University of California, Irvine

Overview

- Our focus:
  - Dynamic reconfigurability of network systems
    - to adapt to changes in network conditions
      - Adaptation through reconfiguration
- Two approaches
  - Network-aware reconfigurable applications
    - autonomously reconfigure their behaviors to adapt to dynamic network conditions (e.g. network load)
  - Reconfigurable middleware systems
    - reconfigures their internal components to adapt to resource availability (e.g. available memory space, available transport protocols).

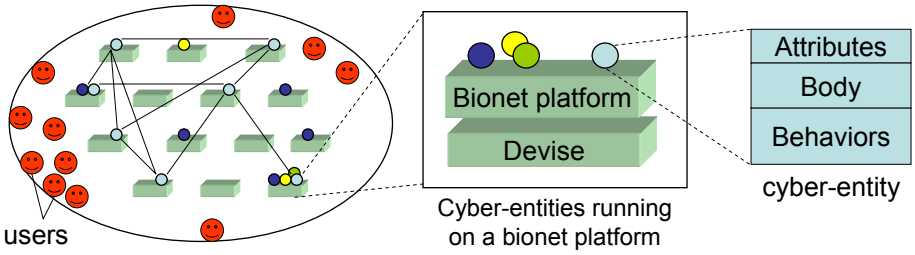
Bio-Networking Architecture

- Observation
  - Desirable properties of network applications (e.g. adaptability) have been already realized in various biological systems (e.g. bee colony, bird flock, etc.).
- The Bio-Networking Architecture
  - applies key biological principles and mechanisms for designing network applications.
  - a framework for developing large-scale, highly distributed, heterogeneous, and dynamic network applications.

Biological Concepts Applied

- Decentralized system organization
  - biological entities = cyber-entities (CEs)
    - the smallest component in an application
- Lifecycle
  - Each CE stores and expends *energy*
    - in exchange for performing service.
    - for using resources.
  - Each CE replicates itself and reproduce a child with a partner.
- Evolution
  - Dynamic reconfiguration of network applications *through evolution*

Structure of Network Apps



- Attributes
  - ID
  - Relationship list
  - Age
  - …etc.
- Body
  - Executable code
  - Non-executable data

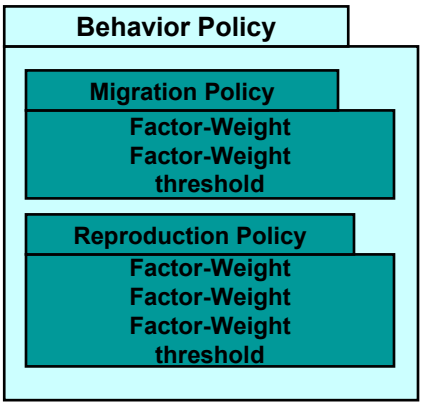
- Behaviors
  - Communication
  - Migration
  - Replication and reproduction
  - Death
  - Resource sensing
  - State change
  - Energy exchange and storage
  - Relationship establishment
  - Social networking (discovery)

Cyber-Entity's Behavior Policy

Each CE has its own policy for each behavior.

A behavior policy consists of *factors* (F), *weights* (W), and a *threshold*.

– If $\sum F_i \cdot W_i$ > threshold, then migrate.



Example migration factors:

– *Migration Cost*
  - A higher migration cost (energy consumption) may discourage migration.

– *Distance to Energy Sources*
  - encourages CEs to migrate toward energy sources (e.g. users).

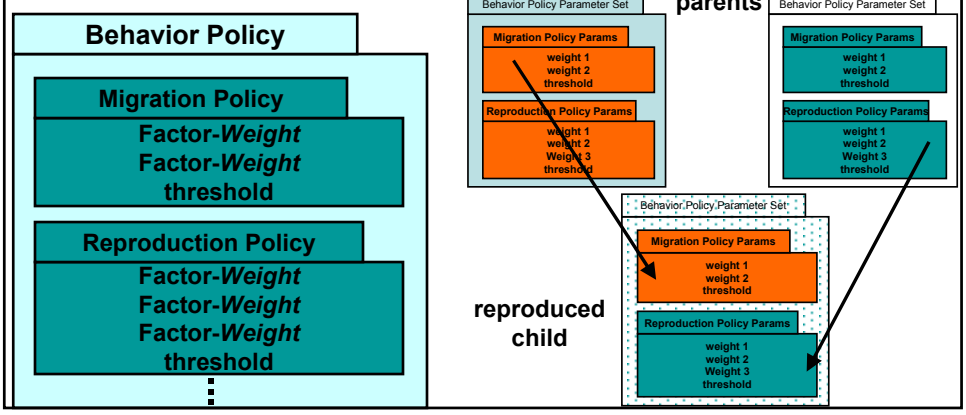
– *Resource Cost*
  - encourages CEs to migrate to a network node whose resource cost is cheaper.

Reconfiguration of Network Applications

- Evolution as a means to reconfigure behaviors of network applications.
  - Biological entities adjust themselves for environmental changes through species diversity and natural selection
  - CEs evolve by
    - generating behavioral diversity among them, and
      - CEs with a variety of behavioral policies are created
        - » by human developers manually, or
        - » through *mutation* and *crossover* (automatically).
    - executing natural selection.
      - death from energy starvation
      - tendency to replicate/reproduce from energy abundance

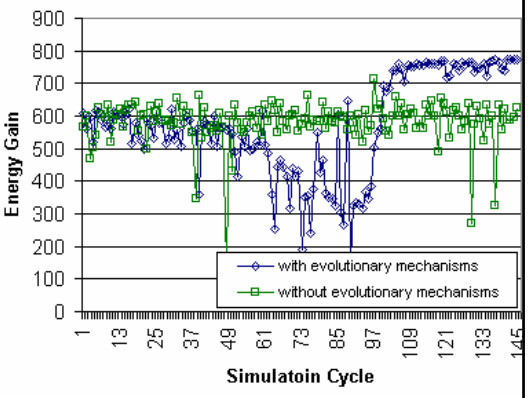
Mutation and Crossover

- Weight values in each behavior policy change dynamically through mutation.
- Mutation occurs during replication and reproduction.
- Crossover occurs during reproduction.
- A child CE inherits different behaviors from different parents through crossover.



A Simulation Result

- Users (energy sources move around network randomly.
- Evolutionary CEs gain more energy than non-evolutionary ones;
- Evolutionary CEs adap better to dynamic network conditions.
  - by moving closer to users and avoiding network nodes whose resource cost is expensive.

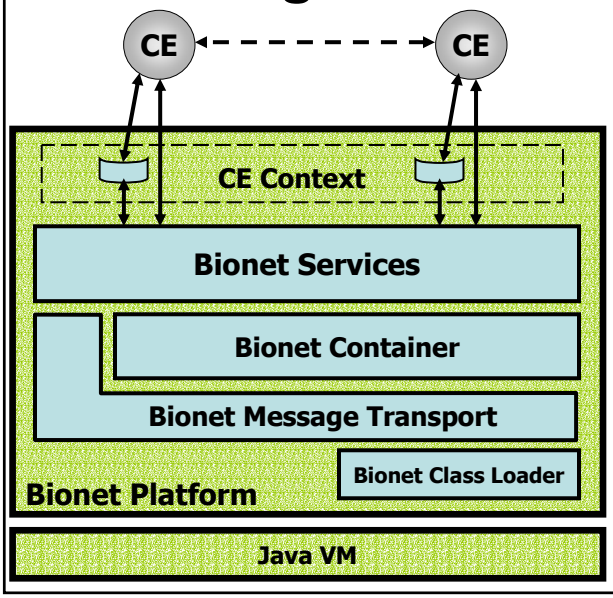


  - by increasing weight values of *distance-to-user* and *resource cost* factors.

Status and Issue

- Through simulations, we have already confirmed
  - Effectiveness of energy concept
  - Effectiveness of mutation and crossover
  - Adaptability of CEs through evolutionary reconfiguration mechanisms in dynamic networks
- Issue
  - Acceleration of evolutionary process
    - by reducing energy loss and time delay.

Empirical Implementation of Reconfigurable Network Apps



A *Cyber-entity (CE)* is an autonomous mobile object. CEs communicate with each other using FIPA ACL.

A *CE context* provides references to available bionet services.

Bionet services are runtime services that CEs use frequently.

Bionet container dispatches incoming messages to target CEs.

Bionet message transport takes care of I/O, low-level messaging and concurrency.

Bionet class loader loads byte code of CEs to Java VM.

Bionet Services

- CEs use bionet services to invoke their behaviors.
  - e.g. bionet lifecycle service when a CE replicates
- Each bionet platform provides 9 bionet services
  - Bionet Lifecycle Service
  - Bionet Relationship Management Service
  - Bionet Energy Management Service
  - Bionet Resource Sensing Service
  - Bionet CE Sensing Service
  - Bionet Pheromone Emission/Sensing Service
  - Bionet Topology Sensing Service
  - Bionet Social Networking Service
  - Bionet Migration Service

Status

- Design phase done. Implementation underway.
  - Has already implemented bionet class loader, bionet message transport, bionet container, and 5 bionet services
  - Now implementing the other 4 bionet services
- Measurements started.
  - Has confirmed bionet platform performs competitively compared with existing ORBs and mobile agent platforms.
- Several design constructs have been reflected to the OMG Super Distributed Objects specification.
- Just started implementing evolution mechanisms that have been used and evaluated in simulation study.
- Will evaluate the characteristics of evolutionary reconfiguration on actual network environment.

Applications

- Content distribution
- Web service
- Peer-to-Peer networks
- Disaster response networks

Reconfiguration of Middleware

- Making not only network applications but also underlying middleware systems to be reconfigurable.
- Approach to reconfigure middleware
  - Compose middleware as a set of components.
  - Middleware
    - sense its context such as available resources and systems current configuration.
    - determine a strategy to reconfigure middleware according to the obtained context.
    - execute the determined reconfiguration strategy.

Status

- In early design stage
  - Investigating middleware reconfiguration mechanisms using the components implemented in bionet platform.
- Designing a metaobject protocol to inspect/modify configuration of middleware components.
- MDA-like approach to reconfigure middleware.
- Biologically-inspired way to reconfigure middleware?

Thank you

- All the papers/documents related to the Bio-Networking Architecture are available at:
- **netresearch.ics.uci.edu/bionet/**