An Overview of the Bio-Networking Platform

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

The Bio-Networking Platform

• The Bio-Networking Architecture
  – a network application architecture that
    • models autonomous adaptive agents after several biological concepts and mechanisms.

• The Bio-Networking Platform (bionet platform)
  – a middleware platform that
    • hosts the Bio-Networking Architecture on real networks
    • aids developing, deploying and executing cyber-entities by providing a rich set of reusable software components.
Our Observation and Goal

• A lot of research efforts have
  – successfully clarified autonomous adaptive agents
  – showed they work well in many applications.
• # of large-scale agent systems is currently very limited
  – Even in agent simulation systems, the scale of agents involved is often kept small, except several exceptions.
  – The scale of agent systems running on actual networks is usually much smaller.
    • e.g. The claim that Auctionbot is scalable is supported by an experiment with only 90 agents.
• Our goal
  – design, implement and measure an efficient and scalable infrastructure for large-scale network applications (agents, or cyber-entities).
  – use cyber-entities, beyond simulations, for Internet-based distributed computing.

Key Features of Cyber-entities

• Decentralized
  – No centralized entity (e.g. directory) on networks
• Autonomous
  – No intervention from other cyber-entities
• Adaptive
  – continuously sense surrounding environment condition and behave according to the sensed condition
  – change their behavioral policies through evolution
• Self-describing
  – keep a set of descriptive information and exchanges it with each other
• Accountable
  – responsible for and explain their activities (e.g. what they did, when they did it, and whom they did it with) through their energy expense.
Design Strategy of the Bionet Platform

- Identify a set of functional requirements derived from the features of cyber-entities
  - e.g. energy mgt., relationship maintenance, discovery, environment sensing, migration, lifecycle mgt., etc.
- Identify a set of common networking and operating functionalities required to deploy and execute CEs.
  - e.g. I/O, concurrency, messaging, network connection management, reference management, etc.
- Design and implement those platform functionalities as a set of reusable objects.
- Each bionet platform per network node

Architecture

- A Cyber-entity (CE) is an autonomous mobile object. CEs communicate with each other using FIPA ACL.
- A platform rep keeps references to bionet services and container.
- 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.
Cyber-entity

- CEs communicate with each other through:
  - interface CyberEntity {
      oneway void send(in string message);
      string metadata();
  };
- send()
  - used to send a message in an asynchronous (non-blocking) manner.
  - Messages are formatted with a subset of FIPA ACL with some extensions.
- metadata()
  - used to obtain cyber-entity’s attributes (self-descriptive information).
  - The mandatory attributes
    - cyber-entity’s GUID (globally unique ID)
    - cyber-entity’s reference
    - type of service that the cyber-entity provides, and
    - the energy units that the cyber-entity requires to provide its service.
  - CEs can specify any additional info as their optional attributes.

- A GUID is a 32-digits string data made from hexadecimal representations of
  - IP address
  - JVM identity hash code
    - through calling System.identityHashCode()
  - the current time in milliseconds, and
  - a random number.
- A cyber-entity’s reference is formatted as a stringfied CORBA IOR.
- Attributes are represented as name-value pairs.
  - GUID='sti3sdr98rd56fn...
  - ref='IOR:daforimk1cmd...
  - serviceType='HTTP/1.1'
  - serviceCost=100.0
• When a CE receive a message through `send()`, it put the message to its message queue.
• Each CE uses 2 threads to:
  – (1) fetch and process incoming messages
  – (2) invoke behaviors
    • sense the nearby environment,
    • identify behaviors suitable for the current environment condition, and
    • invoke the most suitable behavior

![Cyber-entity diagram]

Bionet Services

• Each bionet platform provides 8 bionet services.
• CEs use bionet services to invoke their behaviors.
  – e.g. bionet lifecycle service when a CE replicates

Bionet Lifecycle Service
  – used to initialize and install a CE.
    • maintains a thread pool that contains threads assigned to CEs
    • maintains a GUID pool that contains GUIDs assigned to CEs
  – allows a CE to replicate itself.
  – allows a CE to reproduce a child CE with a partner

Bionet Relationship Management Service
  – allows a CE to establish, examine, update and eliminate their relationships with other CEs.
• **Bionet Energy Management Service**
  – keeps track of energy level of the CEs running on a local platform.
  – allows a CE to pay energy amounts for
    • invoking a service provided by another CE,
    • using resources
    • performing behaviors (i.e. invoking a bionet service).

• **Bionet Resource Sensing Service**
  – allows CEs to sense the type, amount and unit cost of available resources.
    • CPU cycles and memory space

• **Bionet CE Sensing Service**
  – allows a CE to discover other CEs running on the same platform.
    • used to establish relationships with neighboring CEs when a CE is created or completes migration

• **Bionet Migration Service**
  – allows a CE to migrate to another bionet platform.
    • *weak migration*: transmits CE’s data state, CE’s class definition, and CE’s class name

• **Bionet Pheromone Emission/Sensing Service**
  – allows a CE to leave its pheromone (or trace) on a local platform when it migrates to another platform
    • so that other CEs can find the CE at a destination platform

• **Bionet Social Networking Service**
  – allows a CE to search other CEs through their relationships.
    • 4 phases involved: *query initialization, query matching, query forwarding, and query hit backtracking*
    • Examples of Search criteria
      – GUID='sti3sdr98rd56fn...
      – serviceType=='HTTP/1.1' and serviceCost<100.0

---

[Diagram showing the process of networking and migration]
**Bionet Message Transport**

- Bionet message transport
  - abstracts low-level networking details such as I/O, concurrency, messaging, network connection management.
  - Marshaling/unmarshaling messages issued by a CE
    - IIOP version 1.1 used currently
  - TCP connection setup and management
  - Message delivery on a TCP connection
    - One-to-one messaging, currently
    - One-to-many broadcasting/multicasting (future work)
  - Threading (thread pooling) to accept incoming messages

**Bionet Container**

- Bionet container
  - contains a table listing all the CEs running on the same platform.
  - uses the interfaces of the CORBA POA to
    - demultiplex incoming messages,
    - dispatch incoming messages to target CEs,
    - create CE references.
  - keeps track of the current traffic load by counting
    - the size of incoming IIOP messages
    - the number of method dispatches.
External Tools

• An example: graphical performance monitoring tool

---

Initial Measurement Results

• Goal
  – to make sure the bionet platform components are efficient and scalable

• Experiment configuration
  – 2 Windows 2000 PCs connected with a 100Mbps Ethernet switching hub.
    • Each PC runs a Java 1.4.2 VM on 1.8 GHz Pentium 3 CPU and 512 MB RAM.
**Bootstrap overhead and bootstrap footprint**

<table>
<thead>
<tr>
<th>Platform component</th>
<th>Overhead</th>
<th>Footprint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bionet message transport</td>
<td>22 msec</td>
<td>6.65 KB</td>
</tr>
<tr>
<td>Bionet container</td>
<td>102 msec</td>
<td>8.88 KB</td>
</tr>
<tr>
<td>Bionet class loader</td>
<td>12 msec</td>
<td>3.97 KB</td>
</tr>
<tr>
<td>Platform representative</td>
<td>72 msec</td>
<td>5.23 KB</td>
</tr>
<tr>
<td>Relationship map service</td>
<td>23 msec</td>
<td>3.48 KB</td>
</tr>
<tr>
<td>Social networking service</td>
<td>79 msec</td>
<td>12.03 KB</td>
</tr>
<tr>
<td>CE sensing service</td>
<td>46 msec</td>
<td>7.82 KB</td>
</tr>
<tr>
<td>Migration service</td>
<td>51 msec</td>
<td>4.88 KB</td>
</tr>
<tr>
<td>Phenomone emission service</td>
<td>37 msec</td>
<td>3.39 KB</td>
</tr>
<tr>
<td>Lifecycle service</td>
<td>19.217 sec</td>
<td>43.07 KB</td>
</tr>
<tr>
<td>Resource sensing service</td>
<td>84 msec</td>
<td>42.12 KB</td>
</tr>
<tr>
<td>Energy management service</td>
<td>40 msec</td>
<td>8.17 KB</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>19.794 sec</td>
<td>149.64 KB</td>
</tr>
</tbody>
</table>

*Bootstrap overhead:* how long does it take to initialize each platform component?

*Bootstrap footprint:* how much memory space is consumed for each component?

Platform components can be initialized efficiently and they are small-footprint.

**Overhead to initialize a CE on a bionet platform**

<table>
<thead>
<tr>
<th>Activity</th>
<th>Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class loading</td>
<td>0 msec</td>
</tr>
<tr>
<td>Instantiation</td>
<td>0 msec</td>
</tr>
<tr>
<td>Created by a developer</td>
<td>0 msec</td>
</tr>
<tr>
<td>Replicated by a parent cyber-entity</td>
<td>60 msec</td>
</tr>
<tr>
<td>Initialized through the lifecycle service</td>
<td>60 msec</td>
</tr>
<tr>
<td>Discovers 10 cyber-entities running on the same</td>
<td>10 msec</td>
</tr>
<tr>
<td>platform using the CE sensing service</td>
<td></td>
</tr>
<tr>
<td>Establishes (initial) relationships with the</td>
<td>10 msec</td>
</tr>
<tr>
<td>discovered 10 cyber-entities using the relationship</td>
<td></td>
</tr>
<tr>
<td>management service</td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>128 msec</td>
</tr>
</tbody>
</table>

CEs can be initialized and installed efficiently.

**Latency for message transmission between two CEs**

- How long it takes for a CE to send a message to another CE?

A single sender CE and a range of receiver CEs (1, 100, ...1000) were deployed.

Latency is small enough for CEs and comparable with existing distributed object platforms.

**Throughput of a CE on a bionet platform**

- How many interactions 2 CEs can perform per a second?

Throughput is large enough for CEs and competitive with existing distributed object platforms.

Throughput is almost constant as the number of CEs grows, indicating the bionet platform scales.
• Overhead in each of a discovery process

<table>
<thead>
<tr>
<th>Phase in a discovery process</th>
<th>overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relationship establishment between 2 cyber-entities</td>
<td>2 msec</td>
</tr>
<tr>
<td>Query initialization</td>
<td>7 msec</td>
</tr>
<tr>
<td>Query forwarding</td>
<td>20 msec</td>
</tr>
<tr>
<td>Query matching (on a discovery responder)</td>
<td>GUID matching 0 msec</td>
</tr>
<tr>
<td></td>
<td>Complex matching 10 msec</td>
</tr>
<tr>
<td>Query hit backtracking</td>
<td>24 msec</td>
</tr>
</tbody>
</table>

GUID matching:
GUID=="sti3sdr98rd56fn..."
Complex matching:
serviceType=="HTTP/1.1" and serviceCost<150.0

Each phase is efficient enough.

• Overhead to transmit a CE between 2 platforms

Measured the transmission time over the network and the processing time at both origin and destination platforms.

As the size of a CE (mobile code) grows, the overhead increases linearly, instead of exponentially, indicating

---

**Standardization Effort at OMG**

• The goals of the OMG Super Distributed Objects (SDOs) working group
  – provide a standard computing infrastructure that incorporates massive numbers of objects (SDOs) including hardware devices and software components
  – deploy them in a highly-distributed and ubiquitous environment, and
  – allow them to seamlessly interwork with each other.

• Status
  – The SDO RFI issued (’00), and responses gathered (’01)
  – The SDO white paper published (’01)
  – The first RFP published (Jan. 02).
  – The initial proposals submitted (Sept. 02)
  – The revised joint proposal was submitted and adopted (March 2003).
    • by Hitachi, GMD Fokus and UCI
    • “PIM and PSM of SDOs”
      – www.omg.org/cgi-bin/doc?dtc/03-04-02
      – defines resource data model for SDOs and a set of interfaces to access and manipulate the SDO resource data.
On-going and Future Work

• On-going work
  – Extended set of measurements on larger-size networks
  – Adaptation experiments on actual networks
    • using an artificial evolutionary process
  – Efficient algorithms for decentralized discovery of CEs
  – Finalization of the SDO specification

• Future work
  – Reconfigurability of bionet platform components
    • All the platform components might not be used by every user, and might not be deployed on every network node.
  – Deployment of the bionet platform on PDAs, cell phones and sensor devices.
  – Model-driven development of CEs
    • to automatically (or semi-automatically) generate source code skeleton for a CE from its model (e.g. UML).

Thank you

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

• Sponsors
  – NSF (National Science Foundation)
  – DARPA (Defense Advanced Research Program Agency)
  – AFOSR (Air Force Office of Science Research)
  – State of California (MICRO program)
  – Hitachi
  – Hitachi America
  – Novell
  – NTT (Nippon Telegraph and Telephone Corporation)
  – NTT Docomo
  – Fujitsu
  – NS Solutions Corporation