

The Bio-Networking Architecture

Bi-weekly report #11 (October 28, 2002): Bio-Networking Platform

PI: Tatsuya Suda (suda@ics.uci.edu)
University of California, Irvine
<http://netresearch.ics.uci.edu/bionet/>

Introduction

The Bio-Networking platform is a middleware that provides reusable software components for deploying and executing cyber-entities (CEs), service components in the Bio-Networking Architecture. As reported in the biweekly report #3 submitted on July 8, 2002, the architecture of the Bio-Networking platform is organized as shown in Figure 1. Cyber-entities run on a Bio-Networking platform and use services provided by the Bio-Networking platform for performing its services and invoking its behaviors (e.g. discovery, replication and migration). The CE context is an entry point for a cyber-entity to access Bionet services. It examines if a Bionet service requested by a cyber-entity is available on the Bio-Networking platform, and if it is, it obtains a reference to the requested Bionet service. The Bionet services provide a set of runtime services that cyber-entities frequently use. The Bionet container dispatches incoming messages to cyber-entities running on a local Bio-Networking platform. The Bionet message transport abstracts low-level networking and operating details such as network I/O, concurrency, messaging, and network connection management. The Bionet class loader dynamically loads class definitions of CEs into a Java virtual machine when they migrate from a Bio-Networking platform to another.

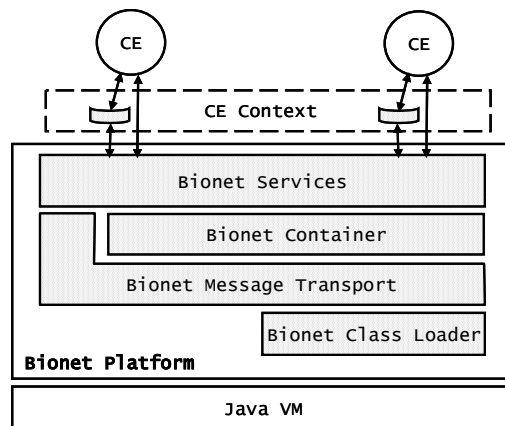


Figure 1. Architectural Components in the Bio-Networking Platform

To date, the PI has provided two biweekly reports describing the achievements of the project in the area of Bio-Networking Platform design and implementations. In the first biweekly report (report #3, submitted on July 8, 2002), the PI described an overview of each of the architectural components and reported some preliminary performance measurement of the Bionet message transport and Bionet container. In the second biweekly report (report #6, submitted on August 19, 2002), the PI described the design and implementation of cyber-entities and the CE context and reported additional results of the preliminary performance measurement of the Bionet message transport and Bionet container.

New Accomplishments

Since the submission of the bi-weekly report #6 (submitted on August 19, 2002), the PI has made further progress in the implementation and measurement of the Bio-Networking platform. The PI has recently submitted and published works in this research area

- In a standard proposal entitled “The Initial Submission to the Platform Independent Model (PIM) and Platform Specific Model (PSM) for Super Distributed Objects,” presented at the Object Management Group technical meeting [SSS02].

The PI’s recent accomplishments include (1) further implementation of the Bio-Networking platform, and (2) additional measurements of the Bio-Networking platform to examine its efficiency. The PI also submitted an official standard technology proposal to the Object Management Group (OMG), the largest standard making body for object-oriented software technologies, in order to reflect the key designs of the Bio-Networking platform to the OMG Super Distributed Objects specifications [SSS02, SS02a, SS02b].

Implementation of the Bio-Networking Platform

Since the bi-weekly report #6 submitted on August 19, 2002, regarding the Bio-Networking platform, the PI has implemented Bionet migration service and measured the performance of Bionet lifecycle service and Bionet migration service. These services are part of the Bionet services (see Figure 1). The PI has implemented five Bionet services (lifecycle, relationship, energy, resource sensing, and migration) out of nine key Bionet services. Nine key Bionet services are briefly described below.

- *Bionet lifecycle service*; allows cyber-entities to change their internal state, replicate, and reproduce.
- *Bionet relationship management service*; allows cyber-entities to establish, examine, update and eliminate their relationships with other cyber-entities.
- *Bionet resource sensing service*; allows cyber-entities to inquire the type, amount and cost of available resources (CPU cycle and memory space).
- *Bionet energy management service*; keeps track of the energy levels of cyber-entities running on a local platform, and allows the cyber-entities to pay energy units for receiving services from another cyber-entity, for utilizing resources (CPU cycle and memory space), and for performing their behaviors.
- *Bionet discovery service*; allows cyber-entities to search for other cyber-entities on a remote node through their relationships. This discovery is called cyber-entity level discovery because the discovery is performed with the knowledge of cyber-entities (i.e. relationships).
- *Bionet cyber-entity sensing service*; allows cyber-entities to search for other cyber-entities running on a local and neighboring platforms. This discovery is called platform level discovery because the discovery is performed with the knowledge of platforms (i.e. platform connectivity).
- *Bionet pheromone emission service*; allows cyber-entities to emit their pheromones (traces) and to sense pheromones emitted by other cyber-entities.
- *Bionet topology sensing service*; allows cyber-entities to sense the existence of remote platforms within N hops.
- *Bionet migration service*; allows cyber-entities to migrate to another platform.

The bionet migration service, which the PI recently implemented, employs a weak migration mechanism. With our weak migration mechanism, a CE migrates only with its data

state allocated in heap memory space (e.g. values of instance variables). In our weak migration mechanism, a CE does not migrate with its execution state. When a CE migrates, a bionet migration service running on a source platform stops the CE's execution, serializes its data state, and transfers a Java mobile code that includes the CE's class definition, serialized data state and class name. A bionet migration service running on a destination platform receives the mobile code, desterializes the received CE, and initializes it. When a bionet migration service at a destination platform desterializes the received CE, it uses a Bionet class loader (see Figure 1) to dynamically load the CE's class definition into a local Java virtual machine.

Measurement of the Bio-Networking Platform

Table 1 and 2 show preliminary measurements of the Bio-Networking platform that the PI implemented. In both measurements, two Bio-Networking platforms run on two different network nodes whose configurations are identical; a Java 2 virtual machine (version 1.4.1) atop a Windows 2000 with an Intel Pentium 3 processor (1.8 GHz), 512 MB RAM, and 100 Mbps Ethernet connection.

Table 1 shows the overhead for a Bionet lifecycle service to initialize a CE and to change a CE's state. The CE initialization overhead measures the time for a Bionet lifecycle service to make a CE ready for executing and accepting requests for its service. The CE state change overhead measures the time for a Bionet lifecycle service to change a CE's runtime state. (A CE has three possible states; autonomous, active or inactive.) The measurement results show that the CE initialization and CE state change overhead is fairly small. The overhead to change a CE's state from active to autonomous is larger than other state changes, because the CE needs to acquire a new thread when it becomes autonomous.

Measure		Measurement result
CE initialization overhead		38.2 msec
CE state change overhead	Autonomous to active	3.6 msec
	Active to Autonomous	34.3 msec
	Active to inactive	3.2 msec
	Inactive to active	2.4 msec

Table 1. Overhead to Initialize a CE and to Change CE State

Table 2 shows the overhead for a Bionet migration service to move a CE from one Bio-Networking platform to another platform. The preliminary measurement data shown in Table 2 reveal that the migration overhead is significantly larger than the two-way communication overhead for two CEs to exchange empty string messages. This is because a migration process incurs significantly more number of functions than that required for regular message exchange between two CEs. Migration process, for instance, includes serialization of CE's data state, mobile code generation and transmission, dynamic class loading, deserialization of the CE's data state, instantiation of a CE, and initialization of a CE, as described in the previous section.

Measure	Measurement result
Migration Overhead	46.3 msec
Two way communication	0.52 msec

Table 2. Overhead of CE Migration

Reference

[SSS02] S. Sameshima, J. Suzuki and T. Suda, "The Initial Submission to the Platform Independent Model (PIM) and Platform Specific Model (PSM) for Super Distributed Objects," Super Distributed Objects Domain SIG, Object Management Group, OMG document number: sdo/2002-09-02, September 2002.

[SS02a] J. Suzuki and T. Suda, "Extended Components in the Hitachi-UCI SDO PIM/PSM," Telecom Domain Task Force, Object Management Group TC meeting at Helsinki, OMG document number: sdo/02-10-08, Helsinki, Finland, October 2002.

[SS02b] J. Suzuki and T. Suda, "An Overview of the Bio-Networking Architecture," The Super Distributed Objects Forum, Object Management Group TC meeting at Helsinki, OMG document number: sdo/02-10-05, Helsinki, Finland, October 2002.