Towards Self-Adaptive Networking with Symbiotic Behaviors of Multi-Agents

Paskorn Champrasert and Junichi Suzuki
Department of Computer Science
University of Massachusetts, Boston
{paskorn and jxs}@cs.umb.edu

Abstract—*This paper describes a biologically-inspired architecture, called SymbioticSphere, which allows network systems to autonomously adapt to dynamic environmental changes. SymbioticSphere consists of two major system components: application services and middleware platforms. Each service and platform is designed as a biological entity and implements biological behaviors such as migration, replication, death and energy exchange. This paper describes how agents and platform behave and interact with each other. Simulation results show that services and platforms autonomously adapt to dynamic network conditions (e.g., user location, network traffic and resource availability) by invoking their behaviors suitable for the conditions. Simulation results also show that services and platforms autonomously behave in a symbiotic manner to pursue their mutual benefits (adaptability).*

1. INTRODUCTION

Network systems have become integral components to operate large-scale network applications. Since they are rapidly increasing in complexity and scale, they face several challenges, particularly *autonomy* and *adaptability*. Network systems are expected to autonomously adapt to dynamic conditions in the network (e.g., network traffic and resource availability) in order to improve user experience, expand operational longevity and reduce maintenance cost [1, 2]. In order to meet these challenges (i.e., autonomy and adaptability), we propose to apply key biological principles and mechanisms to design network systems. This is motivated by the observation that various biological systems have developed the mechanisms to achieve autonomy and adaptability.

SymbioticSphere is an architecture that applies biological principles and mechanisms to design network systems. In SymbioticSphere, each network system consists of two major components: *application services* and *middleware platforms*. SymbioticSphere models these two different types of components as different biological species. Individual services and platforms are modeled as biological entities, analogous to bees in a bee colony. A service is designed as a software agent. Each agent implements a functional service (e.g., web service) and biological behaviors such as energy exchange, replication, death and migration. A platform runs on a network host and operates agents. Each platform provides runtime services that agents use to perform their services and behaviors, and implements biological behaviors such as replication, death and energy exchange.

Agents and platforms are decentralized. There are no central entities to control and coordinate agents/platforms (i.e., no directories and no resource managers). Each of agents and platforms periodically senses its surrounding environment conditions such as network traffic and resource availability, and adaptively performs its behaviors suitable for the conditions. For example, agents may invoke the migration behavior for moving towards the network hosts that accept a large number of user requests for their services. This leads to the adaptation of agent locations, and agents can collectively reduce response time for users. Platforms may invoke the replication behavior for placing additional (child) platforms on neighboring network hosts. This leads to the adaptation of platform availability, and platforms can collectively make more resources available for agents.

Agents and platforms are designed to adapt to dynamic network environments by performing these (regular) behaviors. However, regular behaviors of one species (e.g., agents) can degrade the adaptation of other species (e.g., platforms) in some circumstances. For example, if too many agents migrate toward a user, the platforms running close from the user have a risk to crash due to overloading or resource extinction. In order to address this issue, each agent/platform implements a special type of behaviors, called *symbiotic behaviors*. Each symbiotic behavior is a sequence of regular behaviors that an agent and platform perform in order. Symbiotic behaviors are designed to augment the adaptability of agents and platforms by allowing the two species to cooperate with each other.

This paper describes the regular and symbiotic behaviors in SymbioticSphere and evaluates their impacts on the adaptability of network systems (i.e., agents and platforms). Simulation results show that network systems adapt to dynamic environment conditions (e.g., user location, network traffic and resource availability) through collective behavior invocations and interactions of individual agents and platforms. Simulation results also show that symbiotic behaviors complement regular behaviors and allow agents and platforms to survive longer and pursue their mutual benefits (adaptability).

2. SYMBIOTICSPHERE

This section describes the architecture of SymbioticSphere.

2.1 The Architecture of SymbioticSphere

Figure 1 shows the architecture of SymbioticSphere. Similar to biological entities (e.g., bees), which strive to seek and consume food for living, the agents and platforms in SymbioticSphere store and expend *energy* for living. Each agent gains energy in exchange for performing its service to other agents or human users, and expends energy to use network and computing resources. Each platform gains energy in exchange for providing resources to agents, and periodically evaporates energy. As a living entity, the ultimate goal of each agent and platform is to survive for a long time by balancing its energy gain and expenditure.

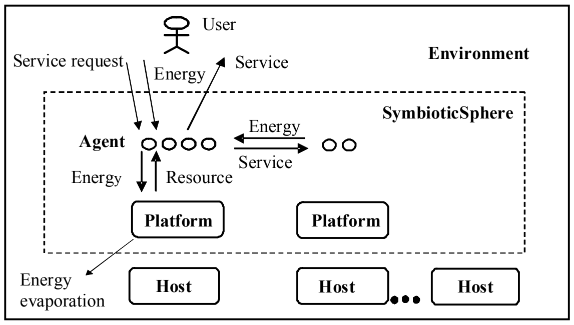

Figure 1. Energy Exchange in SymbioticSphere

Agents and platforms are modeled as different biological species. SymbioticSphere follows ecological principles to design energy exchange among agents, platforms and environment. It models a user as the Sun, agents as producers, and platforms as consumers[1]. Similar to the Sun, users have unlimited amount of energy. Each agent gains energy from users[2] and transfers 10% of its energy level to an underlying platform for consuming resources provided by the platform. Each platform gains energy from agents and evaporates 10 % of its energy level to the environment. This energy exchange rule follows an ecological fact that about 10% of the energy maintained by producer species goes to consumer species [3]. Due to space limitation, see [4] for more details on energy exchange in SymbioticSphere.

2.2. Agents

The Each agent consists of three parts: *attributes*, *body* and *behaviors*. *Attributes* carry descriptive information regarding the agent, such as agent ID, energy level and description of a service it provides. *Body* implements a service that the agent provides. For example, an agent may implement a web service, while another agent may implement a physical model for scientific simulations. *Behaviors* implement actions that are inherent to all agents. Although SymbioticSphere defines nine standard agent behaviors [5], this paper focuses on three of them.

- *Replication*: Agents may make a copy of themselves as a result of abundance of energy. A replicated (child) agent is placed on the platform that its parent agent resides on, and it receives the half amount of the parent's energy level.
- *Death*: Agents die due to energy starvation. When an agent dies, an underlying platform removes the agent and releases all resources allocated to the agent.
- *Migration:* Agents may move from one platform to another.
- *Death*: Agents die due to energy starvation. When an agent dies, an underlying platform removes the agent and releases all resources allocated to the agent.
- *Migration:* Agents may move from one platform to another.

2.3 Platforms

Each platform runs on a network host and operates agents. It consists of *attributes*, *behaviors* and *runtime services*. *Attributes* carry descriptive information regarding the platform, such as platform ID, energy level and health level. Health level indicates how healthy an underlying host is. It is defined as a function of three properties: resource availability on, age of and freshness of a host. Resource availability indicates how much resources are available for agents and platforms on a host. Age indicates how long a host has been alive (i.e., how much stable a host is). Freshness indicates how recently a host joined the network. After a new host joined the network, its freshness gradually decreases from the maximum value. When an unstable host resumes from a failure, its freshness starts with the value that the host has when it went down.

Behaviors are the actions inherent to all platforms.

- *Replication.* Platforms may make a copy of themselves as a result of abundance of energy (i.e., higher demand for resources available on the platforms). The child platform inherits the half of the parent's energy level.
- *Death.* Platforms die due to lack of energy. A dying platform uninstalls itself and releases all resources the platform uses. Despite the death of a platform, an underlying host remains active so that another platform can run on it in the future.

Runtime services are middleware services that agents and platforms use to perform their behaviors.

2.4 Behavior Policies of Agents and Platforms

Each agent/platform has policies for its behaviors. A behavior policy defines when to and how to invoke a particular behavior. Each behavior policy consists of *factors* (F_i), which evaluate environment conditions (e.g. network traffic) or agent/platform/host status (e.g. energy level and health

[1] In the ecological system, producers (e.g., shrubs) convert the Sun light energy to chemical energy. The chemical energy is transferred to consumers (e.g., hares) as consumers consume producers [3].

[2] Each agent specifies the price (in energy units) of its service.

level). Each factor is given a *weight* (W_i) relative to its importance. Behaviors are invoked if the weighted sum of factor values ($\Sigma F_i * W_i$) exceeds a threshold.

The factors in agent migration behavior policy include:

- *Energy Level:* Agent energy level, which encourages agents to move in response to higher energy level.
- *Health Level Ratio*: The ratio of health level on a remote host to the local host, which encourages agents to move to platforms running on healthier hosts. This ratio is calculated with three health level properties (i.e., resource availability, freshness or age) as follows:

$$Health\ Level\ Ratio = \sum_{i}^{3} \left(\frac{HealthLevelProperty_i\ on\ remote\ host - HealthLevelProperty_i\ on\ local\ host}{HealthLevelProperty_i\ on\ local\ host} \right) \quad (1)$$

- *Service Request Ratio*: The ratio of # of incoming service requests on a remote platform to the local platform, which encourages agents to move towards users.
- *Migration Interval*: Time interval to perform migration, which discourages agents to migrate too often.

If there are multiple neighboring platforms that an agent can migrate to, the agent calculates a weighted sum of the above factors for each of the platforms, and moves to a platform that generates the highest weighted sum.

The factors in agent replication behavior policy include:

- *Energy Level:* Agent energy level, which encourages agents to replicate themselves in response to higher energy level.
- *Request Queue Length:* The length of service request queue, which the local platform maintains to queue incoming service requests. This factor encourages agents to replicate themselves in response to higher demands.

The factors in agent death behavior policy include:

- *Energy Level:* Agent energy level. Agents die when they run out of their energy.
- *Energy Loss Rate*: The rate of energy loss in between the current and previous simulation cycles. This factor is calculated with the following equation, where E_t and E_{t-1} are the energy levels in the current and previous simulation cycles. Agents have higher risk to die in response to sharp drop in demands for their services.

$$Energy\ Loss\ Rate = \frac{E_{t-1} - E_t}{E_{t-1}} \quad \ldots\ldots (2)$$

The factors in platform replication behavior include:

- *Energy Level:* Platform energy level, which encourages platforms to replicate themselves in response to higher energy level.
- *Health Level Ratio*: The ratio of health level on a remote host to the local host, which encourages platforms to replicate themselves on healthier neighboring hosts. This ratio is calculated with Equation (1).
- *The Number of Agents:* The number of agents on each platform. This factor encourages platforms to replicate themselves in response to higher agent population on them.

If there are multiple neighboring hosts that a platform can replicate itself on, the platform places a child platform on a host whose health ratio is highest among others.

The factors in platform death behavior include:

- *The Number of Agents*: The number of agents running on each platform. This factor discourages platforms to die when agents run on them.
- *Energy Loss Rate:* The rate of energy loss in platforms. This factor is calculated with Equation 2. Platforms have higher risk to die in response to sharp drop in demands for their resources.

Each agent/platform expends energy to invoke behaviors (i.e., behavior cost) except death behavior. When the energy level of an agent/platform goes over the cost of a behavior, the agent/platform decides whether it performs the behavior by calculating a weighted sum of factors.

2.5 Symbiotic Behaviors

SymbioticSphere currently provides six symbiotic behaviors. Each symbiotic behavior is defined as a sequence of regular behaviors that an agent and platform perform in a cooperative manner to pursue their mutual benefits (e.g., gaining more energy to survive longer) and improve their adaptability. There are two types of symbiotic behaviors: agent-initiated symbiotic behaviors (A1, A2 and A3) and platform-initiated symbiotic behaviors (P1, P2 and P3).

A1: When an agent wants to move toward a user but there is no platform running on a neighboring host closer to the user, the agent can propose the local platform to replicate itself on the neighboring host. If the local platform's health level is low, the platform accepts the agent's proposal. The agent gives the platform the energy units of platform replication cost, and the platform replicates itself on the host that the agent wants to migrate to. As a result, the agent can migrate to the replicated (child) platform and improve response time. The platform improves its health level because resource availability becomes higher.

A2: When an agent is dying due to energy starvation, the agent can ask the local platform to shoulder agent migration cost so that the agent can migrate to a platform on a healthier host (i.e., a platform less crowded with agents). If the local platform's health level is low, the platform agrees with the agent. As a result, the agent can have a higher chance to receive more service requests (i.e., energy) from users and survive longer. The platform improves its health level because resource availability becomes higher.

A3: When an agent is dying due to energy starvation, the agent can ask the local platform to shoulder agent migration cost so that the agent can migrate to a neighboring platform closer to a user. If the local platform's health level is low, the platform agrees with the agent. As a result, the agent can improve response time. The platform improves its health level because resource availability becomes higher.

P1: When a platform replicates on a healthier host, the platform can propose an agent working on it to migrate to the replicated (child) platform. If the agent's energy level is low, it accepts the platform's proposal. The platform gives the agent the energy units of agent migration cost, and the agent migrates. As a result, the parent platform increases its health level because resource availability becomes higher. The child platform can survive longer because it gains energy from the migrating agent. On its destination platform (i.e., platform less crowded with agents), the agent can have a higher chance to receive more service requests (i.e., energy) from users and survive longer.

P2: When a platform has very low resource availability the local host has a risk to crash due to overloading, the platform can propose a local agent to migrate to a platform on a healthier host. If the agent's energy level is low, it accepts the platform's proposal. The platform gives the agent the energy units of agent migration cost, and the agent migrates. As a result, the platform increases its health level. The local agent can migrate to a platform on healthier host. As a result, the agent and platform can reduce the risk to be wiped out due to the local host crash.

P3: When a platform is dying due to energy starvation, the platform can propose the local agents to shoulder platform replication cost so that the platform can replicate itself on a host closer to a user. If the platform dies, the agents die off on the platform too. Thus, the agents accept the platform's proposal, and some of them migrate to a replicated (child) platform. As a result, the migrating agents gain more energy from a user (i.e., survive longer) and improve response time. The child platform can gain more energy from the agents and survive longer.

3. Simulation Results

This section shows a set of simulation results to evaluate how agents and platforms improve their adaptability using their regular behaviors and symbiotic behaviors. Figure 2 shows a simulated network system (a data center). The network system consists of hosts connected in a 7 x 7 grid topology, and users send service requests to agents via user access point. This paper assumes that a single (virtual) user runs on the access point, and it emulates multiple users to send service requests. At the beginning of a simulation, one agent and one platform are deployed on a host that is furtherest from a user.

Each host has 256 MB memory space[3]. Out of the space, an operating system and Java VM consume 128 and 64 MB, respectively. The remaining space is available for a platform and agents on each host. Each agent and platform consumes 5 and 20 MB, respectively. This assumption is obtained from a prior empirical experiment [5]. Figure 3 shows how a user changes service request rate over time. This service request rate is taken from a workload trace of the 1998 Olympic official website [6]. The peak is 9,600 requests/min.

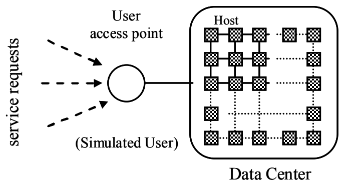

Figure 2. Simulated Network

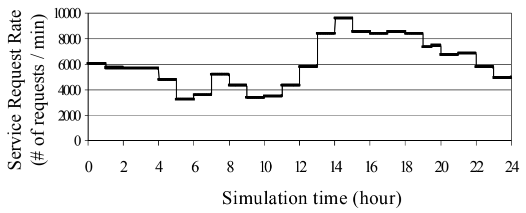

Figure 3. Service Request Rate

3.1 Evaluation of Regular Behaviors

This section evaluates how agents and platforms autonomously adapt to dynamic environmental conditions by using regular behaviors.

Figure 4 shows how service availability (i.e., the number of agents) and resource availability (i.e., the number of platforms) change dynamically. Starting with an agent and a platform at 0:00, they change their populations through replication in order to handle the demand placed on them (6,000 requests/min). When service request rate increases from 12:00 to 2:00, agents gain more energy form users and replicate themselves more often. In response to higher energy intake, they also transfer more energy to platforms. As a result, platforms also increase their population through replications. When service request rate decreases from 15:00, some of agents and platforms die because they cannot balance energy gain and expenditure due to less energy transfer from users. Figure 4 shows that biological mechanisms in SymbioticSphere contribute for agents and platforms to autonomously adapt their availability to dynamic demand changes.

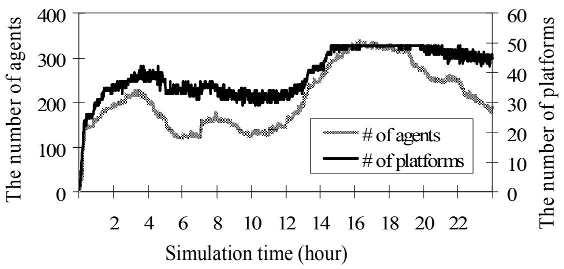

Figure 4. The Number of Agents and Platforms

[3] Currently, memory availability represents resource availability.

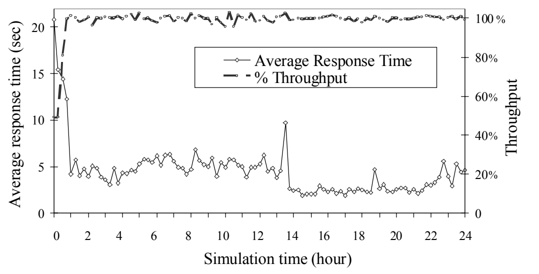

Figure 5. Response Time and Throughput

Figure 5 shows the average response time and the throughput achieved by agents. In the first hour, response time is high (25 sec) because there is only one agent and one platform needs to process 6,000 requests a minute at the beginning of a simulation. As a result, throughput does not reach 100%. However, as agents and platforms replicate themselves and agents migrate towards users, the response time drops to 1 second at 2:00. (throughput reaches 100%.) After 2:00, the response time is constantly 1 second and the throughput is constantly 100%, although service request rate increases from 12:00 to 2:00.This means agents and platforms responsively change their populations and locations against demand changes. Figure 5 shows that the biological mechanisms in SymbioticSphere contribute for agents and platforms to collectively retain response time and throughput performance by adjusting their populations and locations.

3.2 Evaluation of Symbiotic Behaviors

This section evaluates how symbiotic behaviors contribute for agents and platforms to improve their adaptability.

Figures 6, 7, and 8 show how symbiotic behaviors complement regular behaviors to improve the adaptability of agents and platforms. Figure 6 shows that agent-initiated symbiotic behaviors (A1, A2 and A3) contribute to improve response time performance. Figure 7 shows that platform-initiated symbiotic behaviors (P1, P2 and P3) contribute to improve the degree of load balancing. The load balancing index (LBI) indicates how workload (i.e., the number of service requests) is distributed over available platforms. (It is calculated as a standard deviation of workload; the smaller, the higher degree of load balancing.) Load Balancing Index (LBI) is measured with Equation 3 (LBI is a standard deviation of x_i).

$$Load\ Balancing\ Index = \sqrt{\frac{\sum_{i}^{N}(X_i - \mu)^2}{N}} \qquad (3)$$

x_i indicates (the number of messages processed by agents running on platform i) / (the amount of resources utilized by platform i and agents running on platform i). μ represents the expected average of x, which means (the total number of messages processed by all agents) / (the total amount of resources utilized by all platforms and all agents). N is the number of platforms.

Figure 8 shows the average health level. The result shows that agents and platforms with symbiotic behavior (i.e., A1, A2, A3, P1, P2 and P3) contribute to higher health level than the agent and platforms with regular behaviors. This happens because platforms cooperate agents to move to platforms working on healthier host and agents cooperate platforms to replicate to the healthier host. The average health level is increased.

However, Figures 6, 7 do not clearly demonstrate whether symbiotic behaviors significantly improve response time and LBI results. The average response time is not significantly different when using regular behaviors only and using symbiotic behaviors as well. The LBI results contain high variances.

Therefore, this simulation study carried out an ANOVA (analysis of variance) method to evaluate how the response time and LBI results become better in the case of using symbiotic behaviors as well as regular behaviors. The ANOVA results indicate that the response time and LBI results become better with the confidence of 99.99% by using symbiotic behaviors.

Figures 6-8 show that the symbiotic behaviors in SymbioticSphere contribute for agents and platforms to improve their degree of adaptability to dynamic demand changes.

Figure 9 shows how the combinations of symbiotic behaviors impact the performance of agents and platforms. The performance of agents and platforms are measured with seven performance metrics of response time, throughput, LBI, resource efficiency, average platform health level, average agent energy level and average platform energy level.

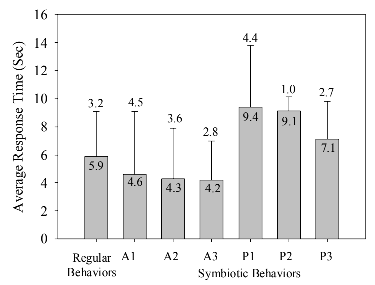

Figure 6. Average Response Time

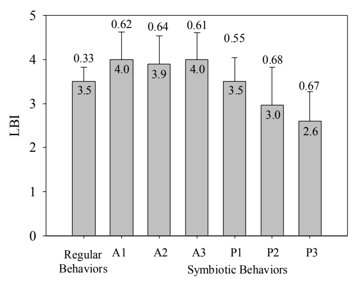

Figure 7. LBI

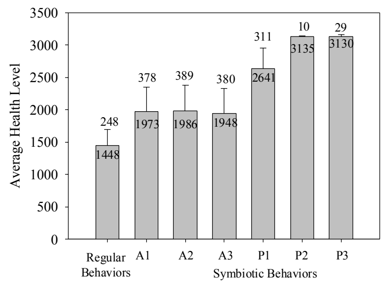

Figure 8. Average Health Level

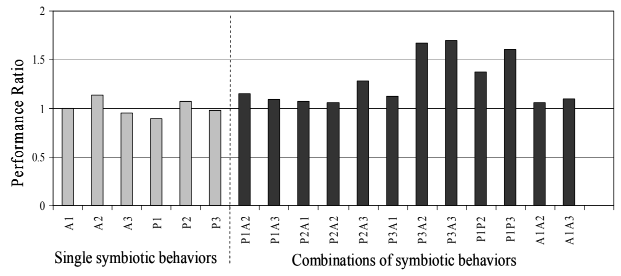

Figure 9. Performance Ratio

Resource efficiency indicates how many service requests can be processed per resource unit. It is measured as (the total number of user requests processed by agents) / (the total amount resources consumed by agents and platforms). The performance ratio is measured as:

$$Performance \quad Ratio = \sum_{i}^{7}\left(\frac{PSi \ - \ PRi}{PRi}\right) \quad (4)$$

PS_i indicates performance metric i with symbiotic behaviors and PR_i indicates performance metric i without them.

Figure 9 shows that the combinations of symbiotic behaviors improve performance ratios than single symbiotic behaviors do. This is because the combinations of symbiotic behaviors can improve several performance metrics simultaneously. For example, P3A1 improve the average response time and the LBI because agents can reduce response time with A1 and platforms can distribute workload with P2. Hence, the performance ratio of P3A1 is higher than that of P3 and A1.

4. RELATED WORK

This work is an extension of previous research work [4, 7, 8], which report agents and platforms improve their adaptability, scalability and survivability with their regular behaviors. The previous work did not investigate symbiotic behaviors. This paper shows that symbiotic behaviors complement regular behaviors to improve the adaptability of agents and platforms. This work is the first attempt to improve the adaptability of network systems through cooperation (or symbiosis) between application components (agents) and middleware platforms.

[5] proposes biologically-inspired architecture to allows network applications (agents) to adapt to dynamic network conditions. However, platforms are not designed as biological entities. As a result, they do not adapt to dynamic network conditions. In SymbioticSphere, both agents and platforms are biologically-inspired adaptive entities, and they improve their adaptability in a symbiotic manner. There is no notion of symbiosis between agents and platforms in [5].

[9] and [10] implement the concept of symbiosis between different groups of peers (hosts) in peer-to-peer networks. Peer groups symbiotically connect or disconnect with each other to improve the quality of query results. A special type of peers, cooperative peers, implements the symbiotic behaviors for peer group connection/disconnection. Rather than a symbiosis between groups of hosts, SymbioticSphere focuses on a symbiosis between agents and platforms

5. CONCLUSION

This paper presents two different (regular and symbiotic) behaviors that agents and platforms implement in SymbioticSphere, and describes how agents and platforms act and interact with each other. Simulation results show that agents and platforms autonomously adapt to dynamic environmental conditions (e.g., user location, network traffic and resource availability) by using their regular behaviors. Simulation results also show that symbiotic behaviors improve the adaptability of agents and platforms.

REFERENCES

[1] P. Dini, W. Gentzsch, M. Potts, A. Clemm, M. Yousif and A. Polze, "Internet, Grid, Self-adaptability and Beyond: Are We Ready?," *Proc. of the IEEE Int'l Workshop on Self-Adaptable and Autonomic Computing Systems*, August 2004.

[2] R. Sterritt and D. Bustard, "Towards an Autonomic Computing Environment," In *Proc. of 14th IEEE Int'l Workshop on Database and Expert Systems Applications*, September 2003.

[3] R. M. Alexander, "Energy for Animal Life," Oxford University Press, May 1999.

[4] P. Champrasert and J. Suzuki, "SymbioticSphere: A Biologically-inspired Network Architecture for Autonomic Grid Systems," *Proc. of IASTED CIIT*, October 2005.

[5] J. Suzuki and T. Suda, "A Middleware Platform for a Biologically-inspired Network Architecture Supporting Autonomous and Adaptive Applications" *IEEE J. on Selected Areas in Comm.* Feb. 2005.

[6] C. Lefurgy, K. Rajamani, F. Rawson, W. Felter, M. Kistler, and T. W. Keller. "Energy management for commercial servers," *IEEE Computer*, 36(12):39-48, December 2003.

[7] P. Champrasert and J. Suzuki, "SymbioticSphere: A Biologically-Inspired Autonomic Architecture for Self-Adaptive and Self-Healing Server Farms," *Proc. of IEEE Int'l Workshop on Autonomic Communications and Computing*, June 2006.

[8] P. Champrasert and J. Suzuki, "A Biologically-Inspired Autonomic Architecture for Self-Healing Data Centers," *Proc. of IEEE International Conference on Computer Software and Applications Conference*, September 2006.

[9] N. Wakamiya and M. Murata, "Toward Overlay Network Symbiosis," In *Proc. of P2P 2005*, September 2005.

[10] Junjiro Konishi, Naoki Wakamiya and Masayuki Murata, "Proposal and evaluation of a cooperative mechanism for pure P2P file sharing networks," *Proc. of International Workshop on Biologically Inspired Approaches to Advanced In-formation Technology*, January 2006.