Distributed Autonomic Management: An Approach and Experiment towards Managing Service-Centric Networks

Pradeep Ray[1], N. Parameswaran[1], Lundy Lewis[2], and Gabriel Jakobson[3]

[1]University of New South Wales, Sydney, Australia
[2]Southern New Hampshire University, Manchester, New Hampshire, USA
[3]Altusys Corp., Princeton, New, Jersey, USA

Abstract

This paper describes a novel approach for managing service-centric communications networks called distributed autonomic management (DAM). Current approaches to network management employ the client/server model, cooperative stationary agents, and/or non-intelligent mobile agents. The DAM model consists of communities of mobile and stationary intelligent agents in collaboration. We discuss an experiment with DAM and proceed to discuss outstanding research issues. The DAM approach uses the properties and characteristics of autonomic systems in support of managing service-oriented communications networks and protecting e-commerce and business enterprises against cyber terrorism.

1. Introduction

Integrated service management is the discipline of monitoring and controlling large networks that include multiple network technologies, diverse computer systems attached to the network, and services offered by the network [1]. Centralized approaches using the classic client/server paradigm have demonstrated an inadequacy for effective management of such networks. Research has been conducted on decentralized approaches [2], but the solutions thus far have suffered from: increased bandwidth consumption as the network grows, inflexibility against evolving networking technologies, a lack of self-management with decreased manual intervention, and a lack of dealing with security and cyber attacks. Further, we are entering an era of service-centric networking [3], and the traditional client/server paradigm seems to be incongruous with this new style of networking.

It is against this background that a new approach and a new paradigm are needed for managing and protecting such large, service-oriented networks. In this paper we present a new management paradigm called Distributed Autonomic Management (DAM) where several communities of stationary and mobile intelligent agents, distributed hierarchically over the network, collectively monitor and control the network components and services with minimal human intervention. Our goal is to provide a flexible balance of autonomic control with the decentralization of management over a network - a goal that has so far been elusive in the integrated network management field.

Our DAM approach is inspired by (i) the human body's immunization system, (ii) recent work on cognition, and (iii) recent work on autonomic computing [4]. The biological metaphor of the human body's immune system serves as the guiding principle for the approach. For example, the purpose of a smallpox vaccination is to train the body's immunization agents to attack and destroy artificial, non-threatening smallpox antibodies. Subsequently, when an authentic smallpox agent enters the body, then the body's immunization agents recognize the foreign agent, migrate towards it, surround it, and destroy it. Such immunization agents are wired to do so as a result of evolution. Figure 1 shows the difference in concept between the client/server model and the DAM model. The figure is for illustration purposes only; the number of management clients and servers often run into the hundreds in real-world applications. On the DAM model, a community of management agents resides at a home base and venture from the home base upon demand to nodes in domains to perform their duties and report back to home base. Alternatively, agents may destroy themselves once their tasks are completed, or may reside temporarily or permanently at nodes if necessary, or they may migrate from node to node if duty requires. Particular tasks will dictate the appropriate dispersion and behavior of agents.

978-1-4244-1672-1/08/$25.00 ©2008 IEEE.

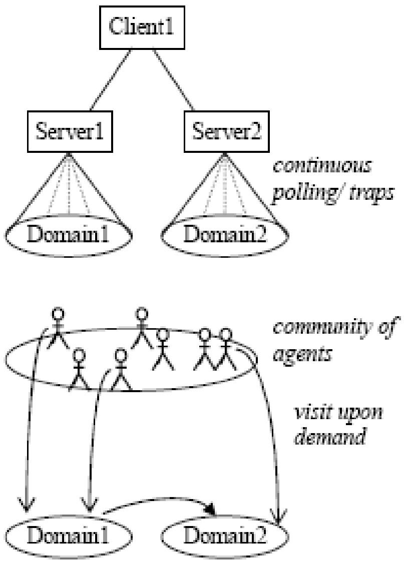

Figure 1. Client/server (top) and DAM (bottom)

In the remainder of the paper, Section 2 describes related work that uses agents for integrated management. Section 3 describes a lab experiment that uncovered special problems in our approach and offered insights into further research issues. Section 4 discusses community structure, task decomposition, and agent cognition in the DAM approach. Section 5 offers a summary and outlook.

2. Related Work

Related progress on agent-based management has included two primary approaches: the stationary intelligent agent approach [5, 6] and the mobile agent approach [7, 8]. These two approaches are related to our DAM paradigm. The agent approach has induced some interesting ideas towards (i) endowing traditional simple network management protocol (SNMP) agents that were essential in the client/server paradigm with some form of intelligence and (ii) collaboration of SNMP agents with mobile agents [9, 10]. These ideas represent a natural reluctance to move away from the traditional client/server approach.

The bandwidth problem has been the primary focus of agent-based research in the field [11, 12]. There has been very little work on managing forwarding looking network technologies, e-business management, or detecting/preventing cyber terrorism [13, 14]. Thus far, none of the problems have found satisfactory solutions. Further, current related work examines tasks that are fairly well-understood in the community, for example the detection of faults and performance degradations of distributed networks [6, 11]. Hard tasks such as the management of forward-looking networking services, e-business management, and detecting and preventing denial-of-service attacks have received less attention because the implicit paradigm does not allow clear thinking about such problems. Nonetheless, these are the sorts of problems that are of utmost importance in the present day world, and new approaches that allow thinking about them are crucial.

There is research on intrusion detection systems (IDSs) that aims to detect and prevent denial-of-service attacks [13, 15]. The research conceives of a network of distributed, communicative, collaborative IDSs and sensors layered over the Internet, called an Internet Firewall. However, the approach depends on stationary IDSs, and thus the decision of how to disperse IDSs over the Internet to get maximal coverage and protection is problematic. Our approach offers a potential solution to this problem in that IDSs would be designed as mobile cognitive agents who can disperse themselves dynamically over the Internet as denial-of-service attacks unfold.

3. An Experiment with the DAM Concept

A prototype version of a network management system called NMbee was implemented at the University of New South Wales [16]. It is based on the DAM concept and was implemented in the Beegent Agent Framework developed at Toshiba Corporation [17, 18]. Figure 2 shows the Beegent system architecture. The central component of the system is the Agent Router (AR) who creates, instructs, and destroys agents. The AR can receive messages from two sources: the user and an agent. Agents must consult the AR for storing or retrieving data from the management and ontology databases. Further, the system requires that an agent wrapper reside on each managed node. Communication is achieved via XML messages over the HTTP protocol.

For the NMbee prototype system, three agent types were designed and implemented in the Beegent framework:

(1) A Monitoring Agent (MonBee) was allowed to migrate to a single node and monitor an SNMP parameter. This type of agent is good for monitoring parameters on a node for a long period of time, as it takes no network overhead and moves processing away from the main server.

(2) A Segment Agent (SegBee) was assigned a segment composed of one or more nodes to insure that the segment satisfies a pre-defined state in the ontology database. The agent migrates to nodes in the segment and collects data to insure the state is satisfied. If the goal is not met on any node, the agent informs the Agent Router of the node where the failure occurred.

(3) An Service Level Agreement (SLA) Agent (SLAbee) works on top of SegBees. It is given a series of nodes and an SLA defined in the ontology database that must hold between users and network services. SLAbees can migrate to any node in the SLA path and request a parameter value from a SegBee.

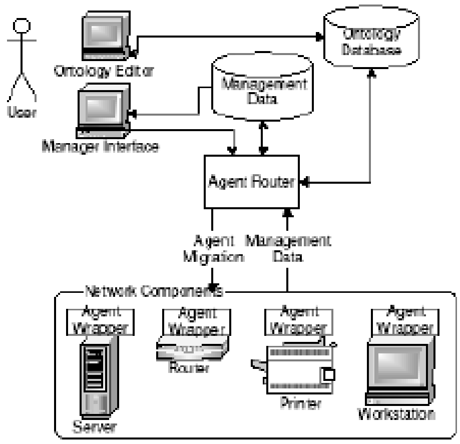

Figure 2. Beegent system architecture

In order to evaluate the DAM concept, a series of experiments were conducted over seven network types, including a simple Ethernet network and a complicated wireless/WAN connected network. Each experiment was conducted twice over a 24 hour period, first with a faultless network and then with a faulty network, and each experiment compared a traditional client/server management system with the NMbee system. The parameters that were measured were (i) resource, CPU, and RAM usage in all managed nodes and the server, (ii) bandwidth usage of all nodes in the network, (iii) bandwidth distribution, and (iv) speed of fault detection. Detailed statistics are provided in [16]; here we summarize the lessons gleaned from the experiments. On smaller networks, the large footprints of Beegent agents (15KB) and requisite agent wrappers were comparatively resource intensive because (i) the management traffic required by traditional network management platforms is quite small and (ii) the Beegent system wasn't designed specifically for network management. However, as the size of the network grew, the traditional architecture tended to induce a corresponding increase in network traffic and deteriorate, whereas the NMbee system remained stable. Finally, fault detection was faster with NMbee because agents reside on nodes they are monitoring. Thus, we are encouraged that a community of collaborating agents based on the DAM paradigm represents a viable approach and suggests research problems whose solutions will contribute to the management of service-centric networks.

4. Specifics of the DAM Approach

The DAM approach is akin to the way in which business enterprises evolve and manage themselves [14, 19, 20]. The basic building block is a community. A community *C* is composed of one stationary agent SA and set of *k* mobile agents MA_j, $0 \le j \le k$. Both SA and MA_j are cognitive agents. For example in Figure 3 we show a management structure comprising six communities, $C_1,\ldots,C_6$, managing three domains. Each domain has been assigned a community C_i, $1 \le i \le 3$, to look after the management tasks. C_4 is the manager of C_1 and C_2, which in turn have C_6 as their manager.

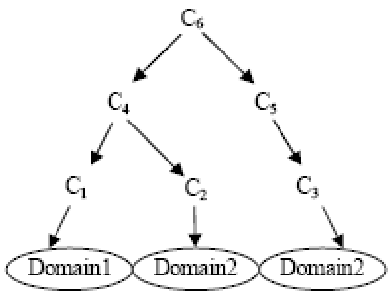

Figure 3. Hierarchy of communities in DAM

Information in the management structure flows from bottom to top and from top to bottom. Information from the bottom typically involves data collection, inferences thereof, and results of actions taken for the tasks that were assigned previously from the top. Similarly, information from the top involves tasks that need to be carried out in response to the information that was received from the bottom. For both SAs and MAs, service management knowledge would be represented explicitly as beliefs, goals, plans, and meta-level reasoning rules stored in their internal data structures [21]. An SA in any community achieves management tasks from its superior and decomposes it into several subtasks, and accordingly assigns them to its sub-SAs. Further, the SA may send the MAs from its community to network nodes to collect and process data and return the results. A larger network is thus managed by several communities which are distributed over the network and organized hierarchically.

Fundamental to this management structure is a task representation as shown in Figure 4. It shows how a task G delegated to a stationary agent SA is decomposed and assigned to its children nodes and its mobile agents. Bold lines denote tasks stored in an SA's memory, thin lines denote tasks stored in an MA's memory. This structure helps monitor the achievement of the task as the subtasks are carried out by the lower level nodes and the MAs.

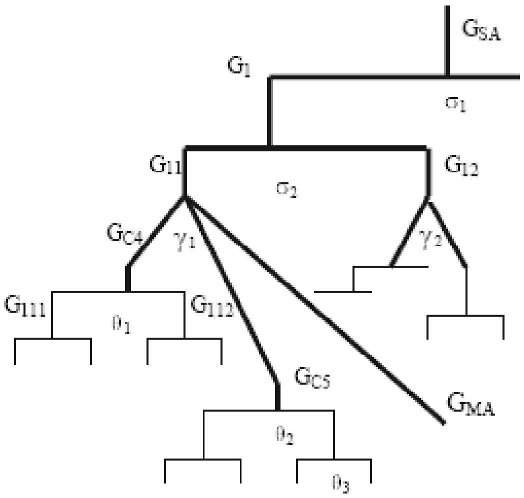

Figure 4. Sample task structure in DAM

In this example, the SA decomposes G into subtasks G_1 and G_2 (G_2 isn't shown in the Figure) by applying the group decomposition operator σ_1 where the subtasks are examined once again by the group. The SA further decomposes G_1 into G_{11} and G_{12} applying σ_2. It then decomposes G_{11}, using γ_1, into G_{C4}, G_{C5}, and G_{MA}, which are subtasks for the children agents C_4, C_5 and the MA. The SA then delegates the subtasks G_{C4}, G_{C5}, and G_{MA} to the children C_4, C_5, and MA, respectively. Each one of these agents further decompose these tasks and ultimately derive executable actions by invoking procedures stored in its memory. Figure 5 shows the hierarchy of decomposition operators used by the agents by partially ordering them according to their level of abstraction. For example, σ_1 is more abstract than σ_2.

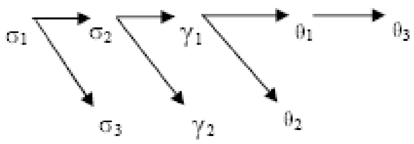

Figure 5. Operator hierarchy, partially ordered

An agent's memory and cognitive architecture is based on the traditional belief-desire-intention (BDI) architecture [22], augmented with learning algorithms found in case-based reasoning. Figure 6 shows the basic structure of an agent's cognitive functions, where a case library is synonymous with a memory [23]. The cognition of an agent works as follows: The input to the agent is a particular task and a library of former episodes or methods of performing the task. When the agent is triggered or assigned a task, similar cases are retrieved in order to find the best way to perform the task. Importantly, similar cases that aren't perfectly on target may suggest a general procedure for performing the task, albeit with some tweaking. Such tweaking is the function of the adaptation module. In the execute phase, the agent attempts to achieve the tasks in the order specified in the task structure, applies further decomposition if necessary, and determines the results. It additionally records the results (good or bad) in the original case. The modified input case, then, is organized into the case library for future reference and future problem solving. Thus, the cognitive agent's problem solving ability is expected to become increasingly fine-tuned over time and exhibit some degree of adaptability in unforeseen situations. Importantly, in our approach, there are interesting alternatives regarding the way the MAs are assigned tasks by an SA agent:

1. An agent may be fitted with a general plan at home base and sent to a remote node to perform a task.
2. An agent residing at a node, already fitted with a general plan, may be sent plan parameters from a superior at home base.
3. An agent residing at a node may be sent a case to expand its knowledge.
4. Agents may share knowledge by sharing cases or by sharing adaptation procedures.

5. Summary and Outlook

The classic client/server approach to integrated network management has become problematic for several reasons: (i) the managed nodes underlying an information system (e.g. transmission devices, computer systems, software applications, and communications media) are becoming increasingly complex, and thus the added volume of data resulting from client/server communication cuts into the payload of the network significantly, (ii) deploying multiple managers-of-managers (MOMs) is a logical patch, overly complex, and does not scale, and (iii) the client/server model doesn't lend itself to concepts by which to study hard tasks such as service management, detection and prevention of denial-of-service attacks and self-healing. Research done so far to address these problems by using either intelligent agents or mobile agents has suffered from three drawbacks:

1. The intelligent agents are presented in isolation as a substitute for traditional network management software, and do not address the problems relating to dynamic network configurations, task delegation, self-management, and proactive cooperation.
2. The mobile agents invariably consist of mobile code that will migrate to remote nodes and perform limited remote computations. However, it is not possible to reprogram them remotely, nor do they have the expertise to cooperate and do not exhibit cognitive abilities.

3. The problem solving procedures and methods are hard coded in the agents (for example, as rules or as methods in the object oriented paradigm). Consequently, they exhibit inflexible behavior and lack the capabilities of learning and awareness of their situations, particularly when the environment changes unpredictably, and they do not address issues encountered in new and forward-looking service technologies such as wireless networks, mobile networks, ad hoc networks, and active/programmable networks.

The concept in our DAM approach is that of a society of cognitive agents, some stationary and some mobile, who have specialized expertise and collaborate to achieve overall management of the network. The properties we wish to explore are adaptable knowledge representation with flexible cooperation, light weight mobile agents, scalability, and situational awareness. The following are outstanding questions in the DAM approach to service management to be explored in our further trials:

- The Service Model: How can SAs and MAs update the model of the network and services relevant to their problems at hand?
- Task Assignment: When creating a new agent, how do we identify the tasks for the agent?
- Community Management: When communities are added, deleted, or reorganized, how is the task representation tree re-structured?
- Community Cooperation Strategies: How will multiple communities interact? How will community-to-community cooperation occur?
- Agent Cooperation Strategies: How will the SAs and MAs interact? How will peer-to-peer cooperation occur?
- Awareness: What are the meta-level reasoning issues while agents carry out assigned tasks (e.g. an agent's awareness of his sibling agents)?
- E-business and cyber terrorism: How can agents collaborate to protect networks and information systems from denial-of-service attacks?
- Agent Formalization: How do we formalize the concept of agent cognition**,** including the structure of a case and the roles of beliefs, attitudes, and intentions?
- Community Formalization: How do we formalize the concept of an agent community, including the categories and roles of agents, the skills of agents, collaboration and communication requirements, and mobility requirements?
- Implementation medium: Do COTS systems exist for further exploration, e.g. British Telecom's Zeus Agent Building Tooldit is a candidate [24,25].
- Evaluation: How do we conduct performance evaluations, including measurements of community dynamics, cooperation, scalability, load balancing, adaptability, and learning?

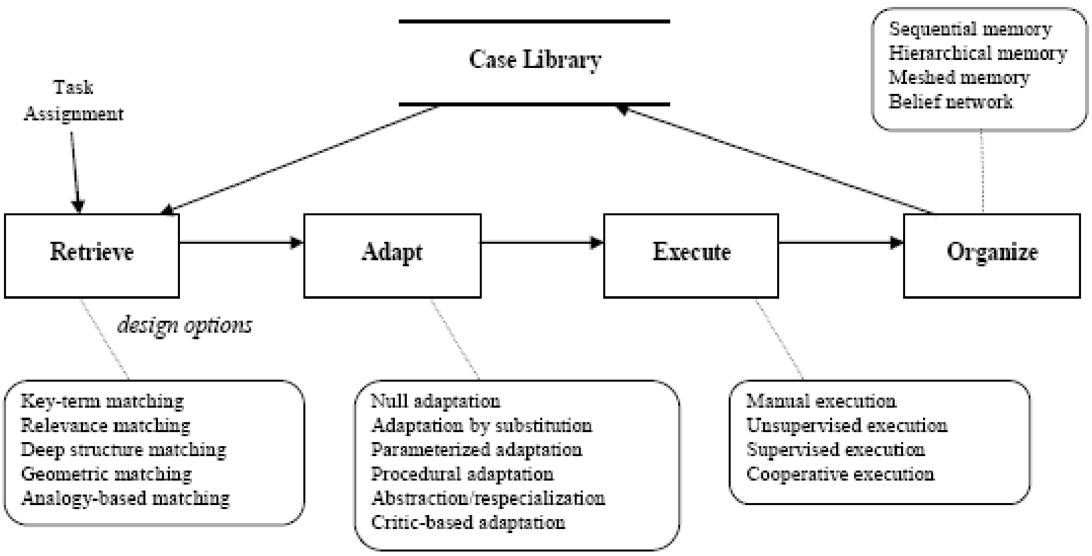

Figure 6. An agent's cognitive functions

References

[1] Lewis, L. *Managing Business and Service Networks*. Kluwer Academic/Plenum Publishers. 2002.

[2] Glitho, R and T Magedanz (editors). *IEEE Network Magazine* Special Issue on Applicability of Mobile Agents to Telecommunications. May/June 2002, Vol. 16, No. 3.

[3] IBM Services Sciences, Management and Engineering, http://www.research.ibm.com/ssme.

[4] Horn, P. Autonomic Computing: IBM's Perspective on Information Technology. IBM white paper at ibm.com/research/autonomic.

[5] Magedanz, T., "Intelligent Agents for Telecommunications", Special Issue: Intelligent Agents for Telecommunications Management, Journal of Network and Systems Management, Vol 8 No.3, 2000.

[6] Cheikhrouhou, M., P. Conti, J. Labetoulle, and K. Marcus. Intelligent Agents for Network Management: a Fault Detection Experiment. In *6th International Symposium on Integrated Network Management*, Boston, 1999.

[7] Magedanz, T., and R. Glitho, "Mobile Agent-Based Network and Service Management", Special Issue: Mobile Agent Based Network Management, *Journal of Network and Systems Management*, Vol 7 No. 3, 1999.

[8] Satoh, I. A Framework for Building Reusable Mobile Agents for Network Management. *IEEE/IFIP Network Operations and Management Symposium*. Florence, Itally, 2002.

[9] Zaph, M., K. Herrmann, and K. Geihs. Decentralized SNMP Management with Mobile Agents. In *6th International Symposium on Integrated Network Management*, Boston, 1999.

[10] Pagurek, B., Y. Wang, and T. White. Integration of Mobile agents with SNMP: Why and How. *IEEE/IEFT Network Operations and Management Symposium*. Hawaii, 2000.

[11] Bohoris, C., G Pavlou, H. Cruickshank. Using Mobile Agents for Network Performance Management. *IEEE/IEFT Network Operations and Management Symposium*. Hawaii, 2000.

[12] Tripathi, A., T. Ahmed, S. Pathak, M. Carney, and P. Dokas. Paradigms for Mobile Agent Based Active Monitoring of Network Systems. *IEEE/IFIP Network Operations and Management Symposium*. Florence, Italy, 2002.

[13] Chank, R. Defending Against Flooding-Based Distributed Denial-of-Service Attacks: A Tutorial. *IEEE Communications Magazine*. October 2002.

[14] Ray, P. *Integrated Management from an E-Business Perspective: Concepts, Architectures, and Methodologies*. Kluwer Academic/Plenum Publishers, January 2003.

[15] Gibson, S. Distributed Reflection Denial of Service: Description and Analysis of a Potent, Increasingly Prevalent, and Worrisome Internet Attack." February 2002. grc.com/dos/drdos.htm

[16] Stephan, R. and Hoo, A., An Agent-Based Network Management System, UNSW Thesis directed by Dr. Pradeep Ray, School of Computer Science and Engineering, UNSW, Sydney, 2002.

[17] Toshiba Corporation, Beegent Multi-Agent Framework, www2.toshiba.co.jp/beegent/index.htm

[18] T. Kawamura, T. Hasegawa, A. Ohsuga and S. Honiden, "Bee-gent: Bonding and Encapsulation Enhancement Agent Framework for Development of Distributed Systems", *Proceedings of the Sixth Asia Pacific Software Engineering Conference*, 1999.

[19] Flatin, J. P. M., S. Znaty, and J. P. Hubaux "A Survey of Distributed Network and Systems Management Paradigms", Swiss Federal Institute of Technology, Lausanne, Switzerland, 1998.

[20] Ray, P. *Cooperative Management of Enterprise Networks*. Kluwer Academic/Plenum Publishers. 2000.

[21] Wooldridge, M. An Introduction to MultiAgent Systems. John Wiley and Sons, 2002.

[22] Wooldridge, M., Reasoning about Rational Agents: MIT Press, 2000.

[23] Lewis, L. *Managing Computer Networks: A Case-Based Reasoning Approach*. Artech House. 1995.

[24] Nwana, H., D. Ndumu, L. Lee, and J. Collis. "ZEUS: A Tool-Kit for Building Distributed Multi-Agent Systems" in Applied Artificial Intelligence Journal, Vol 13 (1), 1999.

[25] British Telecom Intelligent Agenat Research. http://more.btexact.com/projects/agents/zeus/