Journal of Network and Computer Applications 33 (2010) 653–660

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Distributed autonomic management: An approach and experiment towards managing service-centric networks

Pradeep Ray [a], N. Parameswaran [b], Lundy Lewis [c,*]

[a] School of Information Systems Technology and Management, University of New South Wales, Sydney 2052, Australia
[b] School of Computer Science and Engineering, University of New South Wales, Sydney 2052, Australia
[c] Department of Information Technology, Southern New Hampshire University, Manchester, NH 03106-1045, USA

ARTICLE INFO

Article history:
Received 10 August 2009
Received in revised form
5 February 2010
Accepted 18 March 2010

Keywords:
Network and system management
Service-centric communications
Autonomic computing
Intelligent mobile agents

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 (Lewis, 2001). 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, e.g. Glitho and Magedanz (2002), 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 (e.g. see IBM Services, 2008), 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 (see Autonomic Computing, 2008). 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. Fig. 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 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.

In the remainder of the paper, Section 2 describes related work that uses agents for integrated management. Section 3

* Corresponding author.
E-mail addresses: p.ray@unsw.edu.au (P. Ray), paramesh@cse.unsw.edu.au (N. Parameswaran), l.lewis@snhu.edu (L. Lewis).

1084-8045/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jnca.2010.03.016

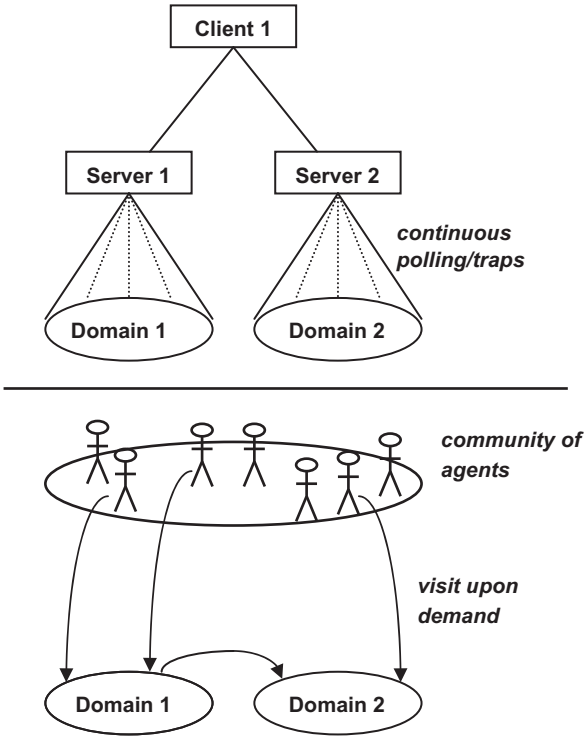

Fig. 1. Client/server (top) and DAM (bottom).

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, and 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 (e.g. see Magedanz, 2000; Cheikhrouhou et al., 1999) and the mobile agent approach (e.g. see Eid et al., 2005; Magedanz and Glitho, 1999; Satoh, 2002). 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, as shown in Zaph et al. (1999) and Pagurek et al. (2000). 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 and has been discussed in Bohoris et al. (2000) and Tripathi et al. (2002). There has been very little work on managing forward looking network technologies, e-business management, or detecting/preventing cyber terrorism such as discussed in Chank (2002) and Ray (2003). 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 as discussed in Cheikhrouhou et al. (1999) and Bohoris et al. (2000). 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, e.g. Chank (2002) 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 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, discussed fully in Stephan and Hoo (2002) and Stephan et al. (2004). It is based on the DAM concept and was implemented in the Beegent Agent Framework developed at Toshiba Corporation. For more information about Beegent, see Beegent (2008) and Kawamura et al. (1999). Fig. 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) A 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.

In order to evaluate the DAM concept, seven experiments were conducted over seven network types, where the first four experiments involved relatively small networks and the remaining three involved larger networks. The networks were: (1) a star topology network, (2) a token ring topology network, (3) a bus topology network, (4) a broadcast Ethernet network, (5) a wireless network, (6) multiple connected Ethernet networks, and (7) two Ethernet networks connected by a WAN link. Each network was emulated using different ports on an Ethernet network, where each port was used to represent a single node.

Each experiment was conducted twice over a 24 h period, first with a faultless network and then with a faulty network. Network overhead is reported in 1 min intervals in bytes. Network resource