Toward Self-Organizing, Self-Repairing and Resilient Large-Scale Distributed Systems

Alberto Montresor **Hein Meling** **Özalp Babaoğlu**

Technical Report UBLCS-2002-10

September 2002

Department of Computer Science
University of Bologna
Mura Anteo Zamboni 7
40127 Bologna (Italy)

The University of Bologna Department of Computer Science Research Technical Reports are available in gzipped PostScript format via anonymous FTP from the area `ftp.cs.unibo.it:/pub/TR/UBLCS` or via WWW at URL `http://www.cs.unibo.it/`. Plain-text abstracts organized by year are available in the directory `ABSTRACTS`. All local authors can be reached via e-mail at the address *last-name*`@cs.unibo.it`. Questions and comments should be addressed to `tr-admin@cs.unibo.it`.

Recent Titles from the UBLCS Technical Report Series

2001-2 *Formalization, Analysis and Prototyping of Mobile Code Systems* (Ph.D. Thesis), Mascolo, C., Janaury 2001.

2001-3 *Nature-Inspired Search Techniques for Combinatorial Optimization Problems* (Ph.D. Thesis), Rossi, C., Janaury 2001.

2001-4 *Desktop 3d Interfaces for Internet Users: Efficiency and Usability Issues* (Ph.D. Thesis), Pittarello, F., January 2001.

2001-5 *An Expert System for the Evaluation of EDSS in Multiple Sclerosis*, Gaspari, M., Roveda, G., Scandellari, C., Stecchi, S., February 2001.

2001-6 *Probabilistic Information Flow in a Process Algebra*, Aldini, A., April 2001 (Revised September 2001).

2001-7 *Architecting Software Systems with Process Algebras*, Bernardo, M., Ciancarini, P., Donatiello, L., July 2001.

2001-8 *Non-determinism in Probabilistic Timed Systems with General Distributions*, Aldini, A., Bravetti, M., July 2001.

2001-9 *Anthill: A Framework for the Development of Agent-Based Peer-to-Peer Systems*, Babaoglu, O., Meling, H., Montresor, A., November 2001 (Revised September 2002).

2002-1 *A Timed Join Calculus*, Bünzli, D. C., Laneve, C., February 2002.

2002-2 *A Process Algebraic Approach for the Analysis of Probabilistic Non-interference*, Aldini, A., Bravetti, M., Gorrieri, R., March 2002.

2002-3 *Quality of Service and Resources' Optimization in Wireless Networks with Mobile Hosts* (Ph.D Thesis), Bononi, L., March 2002.

2002-4 *Specification and Analysis of Stochastic Real-Time Systems* (Ph.D. Thesis), Bravetti, M., March 2002.

2002-5 *QoS-Adaptive Middleware Services* (Ph.D. Thesis), Ghini, V., March 2002.

2002-6 *Towards a Semantic Web for Formal Mathematics* (Ph.D. Thesis), Schena, I., March 2002.

2002-7 *Revisiting Interactive Markov Chains*, Bravetti, M., June 2002.

2002-8 *User Untraceability in the Next-Generation Internet: a Proposal*, Tortonesi, M., Davoli, R., August 2002.

2002-9 *Towards Adaptive, Resilient and Self-Organizing Peer-to-Peer Systems*, Montresor, A., Meling, H., Babaoglu, O., September 2002.

2002-10 *Towards Self-Organizing, Self-Repairing and Resilient Large-Scale Distributed Systems*, Montresor, A., Babaoglu, O., Meling, H., September 2002.

2002-11 *Messor: Load-Balancing through a Swarm of Autonomous Agents*, Montresor, A., Meling, H., Babaoglu, O., September 2002.

Toward Self-Organizing, Self-Repairing and Resilient Large-Scale Distributed Systems

Alberto Montresor [1] **Hein Meling** [2] **Özalp Babaoğlu** [1]

Technical Report UBLCS-2002-10

September 2002

1 Introduction

Modern distributed systems are gaining an increasing importance in our every day's lives. As access to networked applications become omnipresent through PC's, hand-held and wireless devices, more and more economical, social and cultural transactions are becoming dependent on the reliability, availability and security of distributed applications. As a consequence of the increasing demands placed by users upon networked environments, the scale and complexity of current distributed systems are also following an increasing trend. Furthermore, modern systems show an extremely high dynamism, resulting in extremely complex and unpredictable interactions among their distributed components, making it impossible to formally reason about their behavior.

Recent examples of these trends may be found in the *peer-to-peer* (P2P) and *ad-hoc networks* (AHN) application areas. P2P systems are distributed systems based on the concept of resource sharing by direct exchange between *peer* nodes, in the sense that all nodes in the system have equal role and responsibility [7]. Exchanged resources include content, as in popular P2P document sharing applications, and CPU cycles or storage capacity, as in computational and storage grid systems. P2P systems exclude any form of centralized structure, requiring control to be completely decentralized. The P2P architecture enables true distributed computing, creating networks of resources that can potentially exhibit very high availability and fault-tolerance. In AHN, heterogeneous populations of mobile, wireless devices cooperate on specific tasks, exchanging information or simply interacting informally to relay information between themselves and the fixed network [10]. Communication in AHN is based on multihop routing among mobile nodes. Multihop routing offers numerous benefits: it extends the range of a base station; it allows power saving; and it allows wireless communication, without the use of base stations, between users located within a limited distance of one another.

Both P2P and AHN may be seen as instances of *dynamic networks*, that are being influenced by a large number of input sources (users/nodes) and that exhibit extreme variability in structure and load. The topology of the system typically changes rapidly due to nodes voluntarily joining or leaving the network, due to involuntary events such as crashes and network partitions, or due to frequently changing interconnection patterns. The load in the system may also shift rapidly from one region to another; for example, when certain documents may become "hot" in a document sharing system.

1. Department of Computer Science, University of Bologna, Mura Anteo Zamboni 7, 40127 Bologna (Italy), Email: {montresor,babaoglu}@CS.UniBO.IT
2. Department of Telematics, Norwegian University of Science and Technology, O.S. Bragstadsplass 2A, N-7491 Trondheim (Norway), Email: meling@item.ntnu.no

Unfortunately, traditional techniques for building distributed applications, are showing their inadequacy in dealing with the aforementioned complexity. The main problem is that distributed applications are often designed in an inflexible and centralized manner, being based on client-server architectures. This centralized and top-down design is often motivated by the need for ease of system management, but may result in technical problems due to failure to adapt to changing environmental conditions. For example, minor perturbations (e.g., a software update or a failure) in some remote corner of the system may have unforeseen, and at times catastrophic, global repercussions. In addition to being fragile, many situations (e.g., adding/removing components, topology changes) arising from the highly dynamism of modern distributed systems require manual intervention to keep distributed applications functioning.

In our opinion, we are quickly reaching a threshold moment in the evolution of distributed computing: the complexity of distributed applications has reached a level that puts them beyond our ability to deploy, manage and keep functioning correctly through traditional techniques. What is required is a paradigm shift, capable to confront this complexity explosion and enable the construction of robust, scalable, self-organizing and self-repairing distributed systems.

2 Future Directions

The centralized, top-down paradigms on which current distributed systems are based strongly contrast with the situation in the natural world. The behavior of natural systems may appear unpredictable and imprecise, but at the same time living organisms and the ecosystems in which they live show a substantial degree of resilience. Examples of such resilient systems include social insect colonies, evolutionary systems, mammalian nervous systems, and immune networks.

These systems can be seen as instances of *complex adaptive systems* (CAS), which have been used successfully to explain certain biological, social and economical phenomena, can also be the basis of a programming paradigm for distributed applications. CAS are characterized by total lack of centralized coordination. In the CAS framework, a system consists of a large number of *autonomous entities* (agents), that individually have very simple behavior and that interact with each other in simple ways. Despite the simplicity of a single agent, a system composed of a large number of such agents typically exhibits what is called *emergent behavior* [3] that is surprisingly complex and hard to predict. Furthermore, the emergent behavior of CAS is highly adaptive to changing environmental conditions and unforeseen scenarios, is resilient to deviant behavior (failures) and is self-organizing toward desirable global configurations.

Parallels between CAS and modern distributed systems are immediate. In this paper, we suggest the possibility of using ideas and techniques derived from CAS to enable the construction of robust, scalable, self-organizing and self-repairing distributed systems as ensembles of autonomous agents that mimic the behavior of some natural or biological process. In our opinion, the application of CAS will enable developers to meet the challenges arising in dynamic network settings and to obtain global properties like resilience, scalability and adaptability, without explicitly embedding them into the individual agents.

As an instance of CAS drawn from nature, consider an ant colony. Several species of ants are known to group objects (e.g., dead corpses) in their environment into piles so as to clean up their nests. Observing this behavior, one could be mislead into thinking that the cleanup operation is being coordinated by some "leader" ant. Resnick [8] describes an artificial ant colony exhibiting this very same behavior in a simulated environment. Resnick's artificial ant follows three simple rules: (i) wanders around randomly, until it encounters an object; (ii) if it was carrying an object, it drops the object and continues to wander randomly; (iii) if it was not carrying an object, it picks up the object and continues to wander. Despite their simplicity, a colony of these "unintelligent" ants is able to group objects into large clusters, independent of their initial distribution in the environment. This simple algorithm could be used to design distributed data analysis and search algorithms, by enabling artificial ants to travel through a network and cluster "similar" information items. It is also possible to consider a simple variant (the inverse) of the above artificial ant that drops an object that it may be carrying only after having wandered about randomly "for a while" without encountering other objects. Colonies of such ants try to disperse objects uniformly over their environment rather than clustering them into piles. As such, they could form the basis for a distributed load balancing algorithm.

What renders CAS particularly attractive from a distributed systems perspective is the fact that global properties such as adaptation, self-organization and resilience are exactly those that are desirable to distributed applications, and we may obtain these properties without explicitly embedding them into the individual agents. In the above example, there are no rules for ant behavior specific to initial conditions, unforeseen scenarios, variations in the environment or presence of deviant ants (those that do not follow the rules). Yet, given large enough colonies, the global behavior is surprisingly adaptive and resilient. This adaptiveness and resilience may be traced back to several sources. First, complex systems are composed of large number of entities, each of them interchangeable for another. Moreover, interconnections between entities are flexible, allowing transfer of tasks between entities. Finally, the differences between entities enables a diversity of responses in a changing environment.

Instances of CAS drawn from nature have already been applied to numerous problems with notable success [2, 5]. Among them, particularly interesting are ant colonies [2]. Artificial ant colonies have been used for solving complex optimization problems, including those arising in communication networks. For instance, numerous simulation studies have shown that packets in networks can be routed by artificial ants that mimic real ants that are able to locate the shortest path to a food source using only trails of chemical substances called *pheromones* deposited by other ants [2].

3 Current and Future Work

In order to pursue these ideas further, we have initiated the *Anthill* project [1], whose aim is to design a novel framework for the development of peer-to-peer applications based on ideas borrowed from CAS such as multi-agent systems. The goals of Anthill are to provide an environment that simplifies the design and the deployment of novel P2P applications based on swarms of agents, and provide a "testbed" for studying and experimenting with CAS-based P2P systems in order to understand their properties and evaluate their performance.

Anthill uses terminology derived from the ant colony metaphor. An Anthill distributed system is composed of a self-organizing overlay network of interconnected *nests*. Each nest is a peer entity sharing its computational and storage resources. The network is characterized by the absence of a fixed structure, as nests come and go and discover each other on top of a communication substrate. Nests handle requests originated by local users, by generating one or more *ants* – autonomous agents that travel across the nest network trying to satisfy the request. Ants communicate indirectly by reading or modifying their environment, through information stored in the visited nodes. For example, an ant-based distributed lookup service could leave routing information to guide subsequent ants toward a region of the network where the searched key is more likely to be found.

The aim of Anthill is to simplify P2P application development and deployment by freeing the programmer of all low-level details including communication, security and ant scheduling. Developers wishing to experiment with new protocols need to focus on designing appropriate ant algorithms using the Anthill API and defining the structure of the P2P system. When writing their protocols, developers may exploit a set of library components and services provided by nests. Examples of such services include failure detection, document downloading and ant scheduling for distributed computing applications.

A Java prototype of the Anthill runtime environment has been developed. The runtime environment is based on JXTA [4], an open-source P2P project promoted by Sun Microsystems. The benefits of basing our implementation on JXTA are several, including the reuse of various transport layers for communication, and JXTA also deals with issues related to firewalls and NAT. In addition to the runtime environment, Anthill includes a simulation environment to help developers analyze and evaluate the behavior of P2P systems. All simulation parameters, such as the structure of the network, the ant algorithms to be deployed, characteristics of the workload presented to the system, and properties to be measured, are easily specified using XML.

After having developed a prototype of Anthill, we are now in the process of testing the viability of our ideas by developing CAS-based P2P applications, including a load-balancing algorithm for grid computing based on the "dispersing" ant described in the previous section [6] and a document-sharing application based on keyword pheromone trails left by exploring ants [1]. Both our intuition and preliminary simulation results indicate that applying CAS-based techniques for solving various problems in distributed systems is a promising approach. Some of our results may be compared with those obtained by existing techniques. However, the approach followed to obtain our algorithms is completely different from previous approaches, as we mimick the behavior of natural systems, thus inheriting their strong resilience and self-management properties.

The use of CAS techniques in the context of information systems is not new. Current efforts in CAS design can be characterized as *harvesting* — combing through nature, looking for a biological process having some interesting properties, and applying it to a technological problem by adapting it through an enlightened trial-and-error process. The result is a CAS that has been empirically obtained and that appears to solve a technological problem, but without any scientific explanation of why. In the future, we seek to develop a rigorous understanding of why a given CAS does or does not perform well for a given problem. A systematic study of the rules governing fitness of CAS offers a bottom-up opportunity to build more general understanding of the rules for CAS behavior. This study should not be limited to biological systems; other areas such as economics (e.g., *game theory* [9]) can be rich sources of inspiration. The ultimate goal is the ability to synthesize a CAS that will perform well in solving a given task based on the accumulated understanding of its regularities when applied to different tasks. The achievement of this goal would enable the systematic exploitation of the potential of CAS, freeing technologists from having to comb through nature to find the desired behavior.

We conclude with a brief discussion of the inherent limitations of applying a CAS-based approach to building a distributed system. Natural systems show a great degree of resilience, however their behavior can sometimes be unpredictable and imprecise. Implementing a distributed system based on CAS will clearly provide only weak and probabilistic guarantees on the system behavior, and thus it may not be directly applicable to systems requiring strong properties like consistency, synchronization, and timely response.

References

[1] Ö. Babaoğlu, H. Meling, and A. Montresor. Anthill: A Framework for the Development of Agent-Based Peer-to-Peer Systems. In *Proc. of the 22th Int. Conf. on Distributed Computing Systems*, Vienna, Austria, July 2002.

[2] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, 1999.

[3] J. Holland. *Emergence: from Chaos to Order*. Oxford University Press, 1998.

[4] Project JXTA. http://www.jxta.org.

[5] E. Klarreich. Inspired by Immunity. *Nature*, 415:468–470, Jan. 2002.

[6] A. Montresor, H. Meling, and O. Babaoğlu. Messor: Load-Balancing through a Swarm of Autonomous Agents. In *Proc. of the 1st Workshop on Agent and Peer-to-Peer Systems*, Bologna, Italy, July 2002.

[7] A. Oram, editor. *Peer-to-Peer: Harnessing the Benefits of a Disruptive Technology*. O'Reilly, Mar. 2001.

[8] M. Resnick. *Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds*. MIT Press, 1994.

[9] K. Ritzberger. *Foundations of Non-Cooperative Game Theory*. Oxford University Press, Jan. 2002.

[10] C. Toh. *Ad Hoc Mobile Wireless Networks*. Prentice Hall, 2002.