Social System Inspired Anticipatory Models for Software Infrastructures

Zenon Chaczko[1] Marcel Caroly[1] and Ryszard Klempous[2]
[1] University of Technology, Sydney, Australia,
[2] Wroclaw University of Technology, Wroclaw, Poland,
e-mails:zenon.chaczko@uts.edu.au, marcel.caroly@uts.edu.au,
ryszard.klempous@pwr.wroc.pl.

Abstract
This paper describes innovative, charitable (anticipatory) model of software infrastructure, technical challenges and the rationale of its design that drove the concept. Social models such as the benefactor/beneficiary model of MUlti-agent Distributed Storage Middleware can be successfully applied to many areas as a means of automating a wide range of resource distribution challenges both in traditional and future distributed systems. An environment populated with redundant components can benefit from a charitable entity, allowing possibly wasted resources to be efficiently distributed to the network on a voluntary basis.
Keywords: Social Anticipatory Systems, Disaster and Crisis Management, Social Model Software Architecture, Real-Time Information Systems, Middleware.

1 Introduction

The age of ubiquitous or pervasive computing presents a vision of the near future where intelligence is embedded within the environment (Tanenbaum & Steen, 2002). This vision has perhaps been heralded by the recent arrival of mobile communication and computation technologies and the drive to integrate these technologies with the global infrastructure of the Internet. The combination of sensors, processors and wireless networks allow development of context aware services that are capable of acting autonomously on behalf of users in the environment (Szymanski et al., 2005). However, the inclusion of many small, but highly constrained devices in our computing infrastructure has created a range of new challenges for the developers of modern software systems infrastructures. Middleware is a software infrastructure layer that is said to lie between the operating system and the application on each side of a distributed system as indicated by Serain (1999) and in the works of authors at the ObjectWeb Middleware Community (2006). This infrastructure abstracts the underlying resources, mechanisms and protocols from the application developer providing a higher level capability that makes it easier for programmers to develop and deploy their applications (Taylor, 2005). Middleware is not just an API, a bus or simply an additional layer in an application. Middleware is a software infrastructure which is includes a number of independent applications and services such as an ORB in Corba (Murphey, 1998) or a

Lookup facility in Jini[1] middleware (Jini, 2007). There are many challenges of middleware for future distributed systems. Ad-hoc dynamic behaviour, scalability, pervasiveness and openness of infrastructure are being introduced into systems through wireless technologies. Applications are becoming much larger, data stores become enormous and distributed; the software systems composed of many components become more complex and interdependent. Meanwhile, the introduction of distributed sensor networks means that systems will be deployed across broader geographical areas, on greater numbers of nodes and on smaller resource constrained devices (Mikic-Rakic, 2005), (Tubaishat M. et al., 2003). These challenges and the nature of embedded technologies restrict the choices software developers make during system development.

Social anticipatory systems are complex self-organizing and self regulating systems whose emergent properties make them particularly suitable for highly decentralised systems (Weiser, 2007). Socio-inspired design models are based around the concept of using community behaviour to tackle issues such as dynamic re-configurability and adaptation as well as service availability and resource limitations.

Presented research work examines merits an innovative anticipatory model of Multi-Agent Distributed Storage Middleware (MUDS-MW) based on concepts of social benevolence. The interconnectedness and interdependence of complex network of infrastructure systems provided a reasonable rationale and were the real driving factors that led to the inception of the model. Our examination of resource constraints led us to investigate the ongoing work in developing middleware for mobile computing, distributed embedded systems and sensor networks. This investigation culminated in the development of a socio-inspired model of MUDS middleware that could be adapted in a range of applications starting from wireless sensor networks (WSN) based systems, remote teaching labs that we develop at UTS for sometime (Moulton at al., 2004), collaborative simulations and ending on collaborative models of education such as 24-7 Virtual Student Exchange (Chaczko et al., 2006). The outcome of this work is a model of middleware that operates as a charity. A model that is capable of accepting contributions from resource wealthy nodes and distributing this wealth to agents that interact with needy beneficiaries and contributors around the network. The model takes advantage of the massive scale of future distributed systems by introducing the concept of a charity within the middleware.

2 Multi-Agent Distributed Storage Middleware

The primary goal of any middleware solution is to connect users with remote resources in such a way that the underlying complexity, heterogeneity and anticipatory characteristics (i.e. autonomics and adaptation) of the computing and communication environment is abstracted from users (Tanenbaum & Steen, 2002).

[1] Lookup services is Jini middleware component; Jini and the Jini Technology Development Community are trademarks of Sun Microsystems, Inc.

2.1 MUDS-MW Model

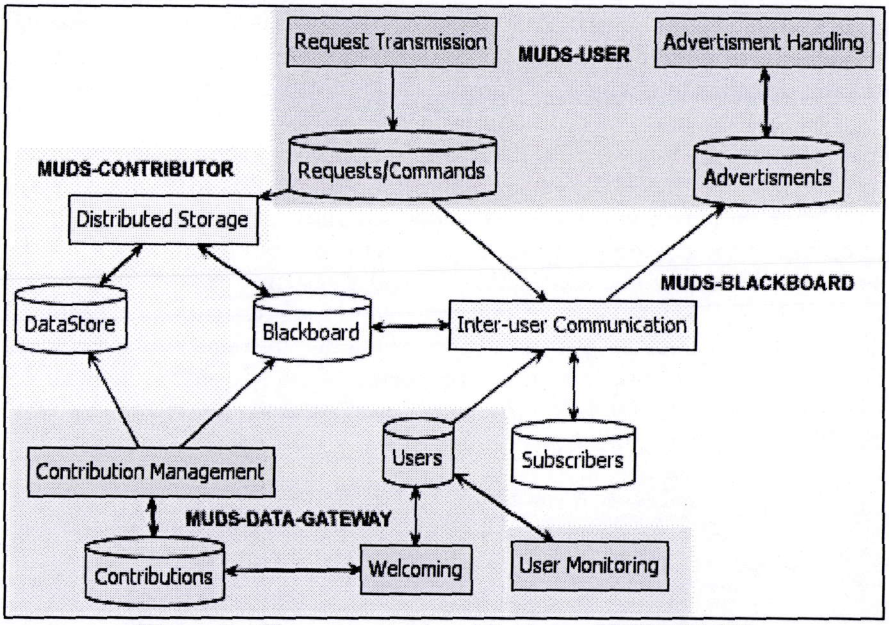

Figure 1: The conceptual architecture of MUDS-MW

MUDS-Middleware (MUDS-MW) is aimed at highly decentralised environments with very large numbers of devices that have differing resource constraints. The model is based on the concept of distributed storage, inter-process communication via blackboards and rapid resource distribution in a dynamic environment. The aim of this model is to act as an effective mechanism for gathering and distributing resources in the dynamic ad-hoc and often hostile environment that characterizes future distributed systems. A particular characteristic of the decentralized environment is that of scale, redundancy and idle resources. The MUDS-MW takes advantage of these factors by introducing the concept of a charity that is capable of accepting contributions from resource wealthy nodes and distributing this wealth to agents that interact with needy beneficiaries and contributors around the network. We based this model around the concept of agents. The concept of an agent is that of an entity that does not have complete control of its environment. Rather an agent has the ability to sense some aspect of its environment and act on its senses in line with its objectives. An agent can be said to "partially control" or "influence" its environment and in some cases an agent may possibly fail in achieving its goals due to factors in the environment that it cannot control or manage (Woolbridge, 2002). Agents are said to have social, autonomous, and pro-active properties making them suitable for highly decentralized systems. Some have gone as far as to say that agents are In fact Wooldridge goes as far as saying that agents are a "natural next step for distributed/concurrent systems research" (Woolbridge, 2002).

2.2 Agent-based Design and Simulation of MUDS-MW

The general architectural model of the proposed software system was constructed using the Prometheus methodology (Phadgam, 2004) creating a high level solution that acts like a charity organization within a resource constrained network. We than modelled the operation and interaction of contributors with the MUDS-MW charity using tools such as Repast simulation (Repast, 2005) and the game theory. The model is composed of three types of agents: the MUDS benefactor, the beneficiary and a charity. Inter-user communication in this environment is supported through a virtual blackboard that resides on top of the contributions made by system benefactors as depicted in Figure1 Middleware is designed to be used within a broader system context. The users are both the benefactors and beneficiaries of resources within the system. As a benefactor the user generously contributes a portion of its own storage resources to a charity. But as beneficiaries, the users require additional resources from the environment in order to serve its users. The charity will respond to user requests for resources by finding contributions to allocate to the user.

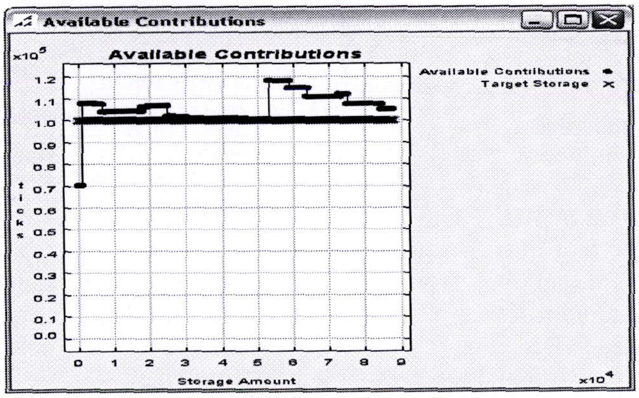

Figure 2: Available Contributions

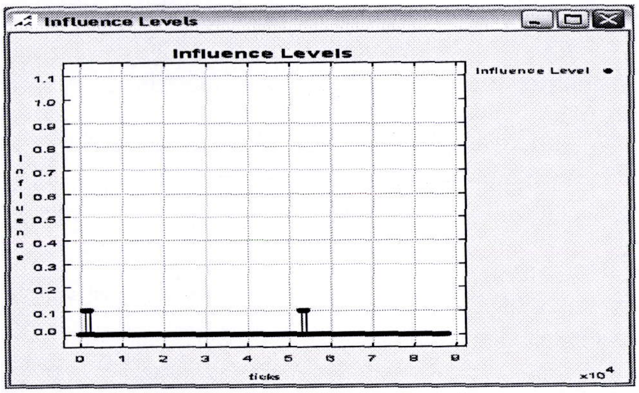

Figure 3: Influence Levels

The charity ultimately acts as a broker between the needy beneficiaries and generous contributors within the system. Once allocated these contributions is at the full disposal its beneficiary. However, these contributions are still officially owned by the benefactors of these resources. This means that the contributor can deny beneficiaries of its resources at any point in time. The proposed model of a benefactor / beneficiary model originates from the relationship between wealthy benefactors and emergency services (i.e. fire brigades, ambulance services, police service and other community services). Whilst emergency services and charities respond to both disasters and emergencies, a charity will provide regular and continuing support to a community (including the emergency services within the community). This means that charities can provide supporting services that distribute resources from benefactors to needy members of the community. The model allows us to consider the challenge of ad-hoc network behaviour as benefactors can. The presented model focuses on the concept of contribution, and the affect of that influence has on the generosity of wealthy donors.

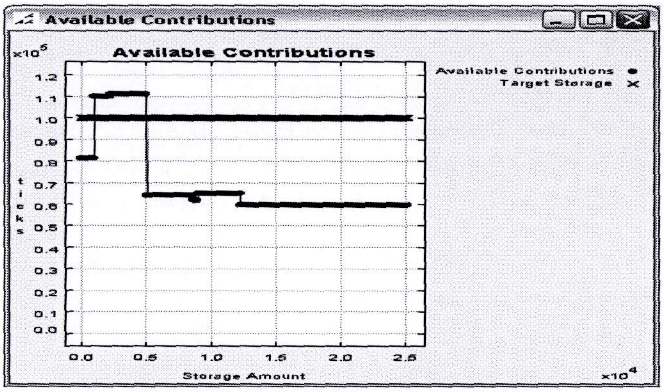

Figure 4: Available contributions

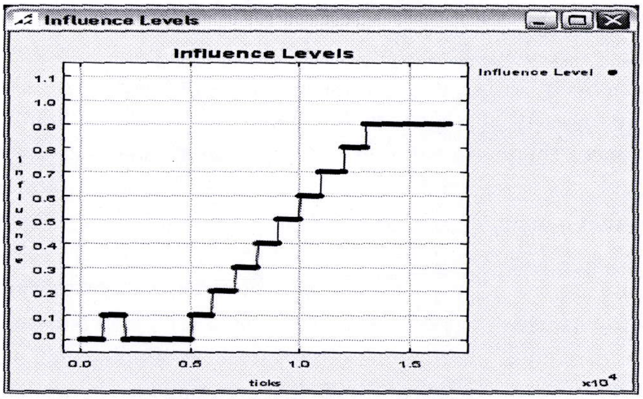

Figure 5: Influence levels

As a solution MUDS-MW gave us many useful insights into the concept of socially inspired middleware, allowing exploring such important issues of anticipatory infrastructure systems as:

- *Resource collection:* The ability of the charity to meet mission targets through campaigns that influence the generosity of benefactors.

- *Resource preservation/conservation:* Self-interested behaviour of the benefactors causes them to become less generous as their own resources become more constrained. This allowed us to model selfish behaviour and greed in the environment.

- *Resource pooling and Crisis Management:* Charities gather resources and respond to pressing needs in the community. The model allowed us to explore the ability of the system to cope with ad-hoc changes such as the loss of resources. We took advantage of the concept of missions as a means of setting goals for the charity such as maintaining a certain level of free resources for emergency scenarios.

We found MUDS-MW to be most suitable for environments in which contributions do not regularly deprive the system of their resources. The Repast agent simulation toolkit allowed us to create a model of contribution in the charitable domain of MUDS-MW. Figure 2 and Figure 3 illustrate the behaviour of the system in a relatively stable, but changing environment. The charity works towards achieving a target goal (represented by the horizontal line in Figure 2). Changes in the availability of resources cause the charity to exude a level of influence on the system benefactors (Figure 3) pushing them to provide more resources to the system. In a highly unstable and dynamic environment (as depicted in Figure 4 and Figure 5) the charity becomes strained. The charity becomes desperate in environments where benefactors are highly unreliable, rare or simply not wealthy enough to contribute adequate quantities of resources. Though the charity is effective at providing relief in emergencies, it becomes strained when disasters strike faster than the charity can collect and pool resources. We proposed to model this scenario by either limiting the rate at which benefactors enter the system or by lowering the quantity of resources they can provide. The sudden loss of resources in this scenario causes the charity to make ever stronger appeals for resources (see Figure 5). However, in this case, the charity is unable to get a good response from its benefactors as they simply cannot offer any further to the cause. Such a situation would be highly undesirable in applications that require high dependability from their resources. Charities can only provide a best-effort approach to distributing resources due to the ad-hoc nature of the environment. The charity will continue working towards its goals despite even if there are no resources available such that it can provide instantaneous access to any new resources that are added to the system.

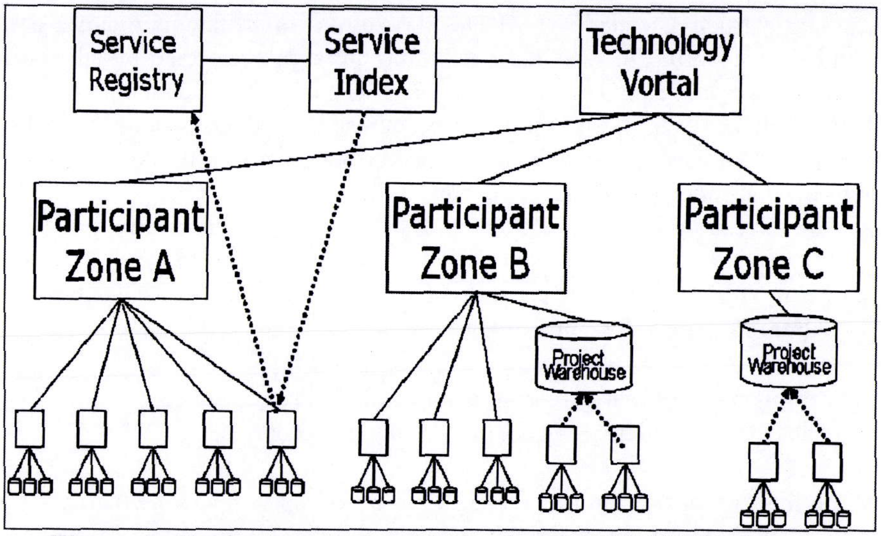

Figure 6: Sharing resources and technology in 24/7 VSX Scheme

3 Applications of MUDS-MW

MUDS-MW model is most suitable for application environments in which resource donors periodically do not use all their resources. This is a frequent case for educational institutions which may not use all their resource all the time. Sharing resources or lab facilities among institutions through remote access is gaining a momentum as cases of the Remote Labs initiative at UTS (Moulton et al., 2004) or iLabs at MIT (Harward et al., 2006) indicate. An interesting model of sharing lab resources, project development environments and tools (project warehouse) using the technology Vortal (as depicted see the Figure 6) among educational institutions residing in three different time zones while cooperating in their teaching programs was introduced by Chaczko, Klempous, Nikodem and Rozenblit, (2006). The model exploits benefits of working in separated by 8 hour 3 time zones around the globe for better utilisation of resources and enhanced student experience. Independent studies show that computer equipment (servers) is being used on average in about 15-30 percent of their maximum processing ability (Rutrell, 2008), having so much capital sitting idle could be a real motivation for the research in implementing better solutions for utilisation of computation and network resources and even charitable actions. Wasting computing power and energy doesn't make much business sense nor is socially responsible. In current climate offering some tax incentives for charitable actions of donors would provide the model with real merits.

4 Conclusion and Future Work

Adaptable middleware is needed to cope with the scale, ad-hoc nature and resource constraints of future distributed systems. Our examination indicates that social software systems infrastructures such as MUDS-MW prove to be an innovative and useful mechanism of adapting to resource constraints in large scale distributed systems. We found that the social behaviour of a charity can be used to take advantage of the massive level of redundancy in an environment to address storage constraints in a distributed environment. Our current work focus is on contribution within the MUDS-MW and the domain of distributed storage. A variety of work is still needed to complete the proposed model of middleware. This includes additions of such functions as:

- Allowing the charity to have a negative influence on contributions when contributions are growing at two fast a rate.
- Identifying the optimal size and number of contributors needed to meet mission targets such that a charity can adapt to changing scenarios.
- Modelling the distribution of resources to needy users and the identifying how the system can balance the difference between the distribution rate of resource to uses and the rate of contribution growth within the environment.
- Implementing the model using technologies such agent based technologies such as COUGAAR, JAS, JADE, HTP (Ng et al. 2006).

Any future work in this area will also consider further practical applications such as the use of this model in high performance grid computing, data warehousing, global caching and power distribution. We also believe that it would be useful to study other social models and see how particular characteristics may be applied to add useful services to current middleware solutions.

References

Abowd D. (1998) Software Design issues for Ubiquitous Computing. IEEE CS Annual Workshop on VLSI: System Level Design, Orlando, Florida, USA.

Britton C (2004) IT Architectures and Middleware, Strategies for Building Large, Integrated Systems, Second Edition, Unisys, Addison-Wesley, USA.

Chaczko, Z.C., Klempous, R., Nikodem, J. & Rozenblit, J. (2006) 24/7 Software Development in Virtual Student Exchange Groups: redefining the work and study week', Proceedings of 7th International Conference on Information Technology Based Higher Education and Training ITHET'06, Sydney, Australia, pp. 30-38.

Harward J., Mao T.T, Jabbour I. (2006) iLab Interactive Services–Overview, version published on 05-18-2006, http://icampus.mit.edu/iLabs/Architecture/Downloads/default.aspx, visited 27/03/2007.

Jini (2007) Technology Development Community site [online], last visited 10/05/2007 http://java.sun.com/developer/technicalArticles/jini/jinicommunity/

Mikic-Rakic M et al (2005) A Style-Aware Architectural Middleware for Resource-Constrained, Distributed Systems in IEEE Transactions on Software Engineering Vol. 31, No 3 March, pp 256-272.

Moulton Bruce D., Lasky Vladimir L. & Murray Stephen J. (2004) The development of a Remote Laboratory: Educational Issues, World Transactions on Engineering and Technology Education, UICEE 2004, Vol.3, No.1.

Murphey N. (1998) Introduction to CORBA for Embedded Systems [online] http://www.embedded.com/98/9810fe2.htm [14/10/2005].

Ng, C., Alibhai, Z., Sabaz, D., Uncu O., Gruver W.A. (2006) Framework for Developing Distributed Systems in a Peer-to-Peer Environment Systems, International Conference on Systems, Man and Cybernetics, SMC apos., Conference on Volume 1, 8-11 Oct., pp.735-739.

ObjectWeb Middleware (2006) Community http://middleware.objectweb.org/ last visited 27/09/2006.

Phadgam L (2004) Developing Intelligent Agent Systems, a Practical Approach, Wiley Series in Agent Technology, Wiley.

Repast Source forge Project [online] http://repast.sourceforge.net/ [24/10/2005].

Serain D. (1999) Middleware, trans. by Iain Craig, Springer-Verlag London Ltd, UK.

Szymanski B. et al. (2005) Advances in Pervasive Computing and Networking. Spring Science+Business Media Inc., USA.

Tanenbaum A. Steen M. (2002) Distributed Systems Principles and Paradigms, Prentice Hall, USA.

Taylor I. (2005) From P2P to Web Services and Grids, Peers in a Client/Server World. Published by Springer-Verlag London Ltd 2005, USA.

Tubaishat M. et al. (2003) Sensor Networks: An Overview. Potentials, IEEE. 22: 20-33.

Weiser Mark (2007) UbiComp http://www.ubiq.com/hypertext/weiser/UbiHome.html Page, last visited 27/03/2007.

Woolbridge M. (2002) An Introduction to Multi-agent Systems, John Wiley & Sons Ltd, UK.

Wooldbridge M. Jennings R (1995) Intelligent Agents, theory and Practice, The Knowledge Engineering Review 10 (2) pp.115-152.

Yasin Rutrell, (2008) How to eke out energy savings in servers, processors, networks and storage systems, Cover story in Government Computer News, Federal Computer Week, Washington Technology, http://www.gcn.com/print/26_29/45439-1.html, last viewed 05/11/08