REPORT DOCUMENTATION PAGE

AFRL-SR-AR-TR-04-

0127

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
	Final Report	May 15, 00 - Nov 14, 03

4. TITLE AND SUBTITLE
Self-Evaluating Space and Robotic Agents

5a. CONTRACT NUMBER

5b. GRANT NUMBER
F49620-00-1-0302

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Dr. Ronald Marsh

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of North Dakota
P. O. Box 9015
Grand Forks, ND 58202

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Department of the Air Force
Air Force Office of Scientific Research
4015 Wilson Blvd.
Arlington, VA 22203-1954

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Distribution Statement A: Approved for public release. Distribution unlimited

20040304 012

13. SUPPLEMENTARY NOTES
DODAAD CODE: 4B858
AFOSR Program Manager: Dr. Robert Herklotz

14. ABSTRACT

In the past three years, we focused on self-evaluative methods for agents that interact with other agents and dynamic environments. We started with the observation that it would be beneficial for agents to sense prevailing qualitics that stem from their interactions such as situation awareness, sociability, coordination, autonomy, failure tolerance, timeliness, and purposefulness. Agents usually have access to constraining requirements over these qualities. Additionally, many of these qualities are conflicting but a balance is desirable for a given domain and agents can discover that in operation. Wc have shown that an agent could attempt adjustments in its interactions to bring about favorable global changes. Such abilities require agents to have capabilities at the architectural level. Theoretical results include various models of relationships among social notions, and several models of autonomy, trust, and Power. Simulation results include two implemented multiagent systems as testheds. Over 60 published reports listed at the end of this report present our theoretical developments and reports of experiments from simulations. In the next two sections we briefly outline theoretical developments and implemented simulations.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Ronald Marsh
					19b. TELEPHONE NUMBER *(Include area code)*

Project Final Report

Self-Evaluating Space and Robotic Agents

AFSOR # F49620-00-1-0302

May 2000 – December 2003

Henry Hexmoor, Ph.D.
College of Engineering
Department of Computer Science & Computer Engineering
University of Arkansas
Fayetteville, AR 72701
hexmoor@uark.edu

Ronald Marsh, Ph.D.
Computer Science Department
University of North Dakota
Grand Forks, ND 58201
rmarsh@cs.und.edu

Report date: February 12, 2004

Summary

In the past three years, we focused on self-evaluative methods for agents that interact with other agents and dynamic environments. We started with the observation that it would be beneficial for agents to sense prevailing qualities that stem from their interactions such as situation awareness, sociability, coordination, autonomy, failure tolerance, timeliness, and purposefulness. Agents usually have access to constraining requirements over these qualities. Additionally, many of these qualities are conflicting but a balance is desirable for a given domain and agents can discover that in operation. We have shown that an agent could attempt adjustments in its interactions to bring about favorable global changes. Such abilities require agents to have capabilities at the architectural level. Theoretical results include various models of relationships among social notions, and several models of autonomy, trust, and Power. Simulation results include two implemented multiagent systems as testbeds. Over 60 published reports listed at the end of this report present our theoretical developments and reports of experiments from simulations. In the next two sections we briefly outline theoretical developments and implemented simulations.

Theoretical Work

We have developed two agent architectures that empower the agent with self-evaluating methods. The first architecture is VONBDI that extends the BDI paradigm with values, norms, and obligations. The second architecture explores how agents can use coherence to organize their memory and to evaluate their interactions with other agents. We looked at issues of group dynamics. Specifically, we developed a model of agent teams that gives prominence to autonomy, cooperation, and responsibility. Relative and absolute senses of autonomy are explored and shown useful in various implementations.

Simulations

Two applications domains are implemented in simulation. One is teaming among low-orbit satellites, which started in 2000 and continued at UND. This software package we implemented allowed the simulation of intelligent autonomous agents. The software uses the paradigm of satellites in orbit communicating with one or more ground stations or each other. Each satellite and each ground station is considered to be an individual intelligent autonomous agent. The software can restrict communication to agents that are in site of each other or can allow communication regardless of relative positions. All agent communication is done via network sockets. The simulation was successfully used to show that an agent's autonomy considerations affect its interaction in a team. We showed that teaming is superior to scheduling task for low-orbit satellites. Work at UND started in summer of 2000 with the low-orbit satellite simulator. We modified the simulation software to use a metric to predict when a ground station is approaching saturation and to spawn an additional ground stations. The most realistic approach was to spawn another third agent (a new type of agent for our simulation) to act as a facilitator to "load balance" the ground stations. The modifications allowed the removal of a ground

station(s) and possible the facilitator should the overall workload drop to a level below what our metric indicates as a saturation point. We modified our simulation to behave in a non-deterministic manor. An issue left to be determined is when we "spawn" additional agents to act as ground stations or facilitators do we create additional agents or do we convert existing agents (satellites) into ground stations and facilitators. In our particular paradigm, converting existing agents was not realistic. However, in the more general paradigm (i.e. a paradigm of autonomous aircraft flying towards a target point) converting existing agents would be very realistic. Finally, since the latest work performed as UND has not yet been published we have included the manuscript for the work (appendix 1) that will be submitted for publication in the near future.

The second simulation entirely designed and implemented at UA was to mediate interactions among a number of UCAVs flying over a terrain with SAM sites. In our implemented testbed three or more fighter aircraft agents have the mission to deliver a bomb over a remote designated site. There is a one to one relationship between agents and planes. Artificial agents control all the planes except one, which is controlled by a human operator. The human operator controls its plane in the field along with the other planes, and will have similar visual and auditory sensing as well as similar flight maneuvering capabilities. The system is implemented in Java. We used this simulator for empirical investigation of roles and social influences. Our preliminary results demonstrate that responding to team members in need of help improves the team's overall effectiveness. This extends our earlier treatment of teams and measurement of team effectiveness along the dimensions of cohesion. We examined methods of capturing the relationships between social actions and resulting influences. We simulated autonomous versus human controlled UCAV. The results are the following:

- By adjusting trust levels among agents in similar situations, the human supervisor is relieved.
- Adjusting autonomy among agents, changes their performance.
- Agents performance drops when it has to wait longer for human decision.
- Agents use past experience of waiting for human decisions and compose a *reliance* factor. Using this notion, they adjust their waiting cycle for human decisions.

Publications

Books and Journal special issues—

1. Henry Hexmoor, 2003. Proceedings of International Conference on Integration of Knowledge Intensive MultiAgent Systems (KIMAS), IEEE.
2. Henry Hexmoor, 2003. Agent Autonomy in a group, A special issue of Journal of Connection Science, Volume 14, No. 4, Taylor and Francis.
3. Henry Hexmoor, Cristiano Castelfranchi, and Rino Falcone, 2003. Agent Autonomy, In Kluwer series: Multi-Agent Systems, Artificial Societies, and Simulated Organizations, Gerhard Weiss (editor).

Book Chapters –

1. H. Hexmoor, Introduction to the Special issue on Agent Autonomy in groups, In Journal of Connection Science, pages 245-248, H. Hexmoor (Ed.), Taylor and Francis.
2. H. Hexmoor, 2002. Adaptivity in Agent-based Systems via Interplay between Action Selection and Norm Selection, In Self-Adaptive Adaptive Softwae: Applications, pages 216-226, Bob Laddaga, Paul Robertson, and Howie Shrobe, (Ed), Springer LNCS2614.
3. H. Hexmoor 2002. Position: Evolution of Agent Architectures, In Radical Agents, 2 pages, Walt Truszkowski, (editor) Springer.
4. H. Hexmoor, and G. Beavers, 2003. Self-Adaptivity via Introspection and Monitoring of Norms and Values, In Self-Adaptive Adaptive Software III, Bob Laddaga (Ed), Springer.
5. H. Hexmoor and G. Beavers, 2002. In Search of Simple and Responsible Agents, In Radical Agents, (Walt Truszkowski, editor), 14 pages, Springer.
6. S. Brainov and H. Hexmoor, 2003. Quantifying Autonomy, In Agent Autonomy, Chapter 4, page 43-56, Kluwer.

Journal Articles–

1. S. Yang and H. Hexmoor, 2003. Optimal connections between high-tech companies: An algorithm and its applications, Submitted to the Journal of Mathematics Sociology.
2. H. Hexmoor and S. Battula, 2003. Human-Agent Interaction: Case Studies in Human Supervised UAV, Accepted in Informatica: An International Journal of Computing and Informatics.
3. H. Hexmoor, 2001. Stages of Autonomy Determination, IEEE Transactions on Man, Machine, and cybernetics- Part C (SMC-C), Vol. 31, No. 4, Pages 509-517, November 2001
4. H. Hexmoor, 2003. A Model of Absolute Autonomy and Power: Toward Group Effects, In Journal of Connection Science, Volume 14, No. 4. page 323- 334, Taylor & Francis Ltd.
5. H. Hexmoor, C. Castelfranchi, R. Falcone, 2003. A Prospectus on Agent Autonomy, In Agent Autonomy, Chapter 1, pages 1-8, H. Hexmoor, C. Castelfranchi, and R. Falcone (Eds) ,Kluwer.

Refereed Conferences—

1. H. Hexmoor and K. M. Krishna, (In press 2004). Social Reasoning and Collaboration among a Large Group of Robots, In Proceedings of the Fifth International Symposium on Collaborative Technologies and Systems (CTS 2004), Waleed Smari and William McQuay (Eds), Society for Modeling and Simulation International.
2. D. Ward and H. Hexmoor, 2003. Deception as a Means for Power Among Collaborative Agents, In online Proceedings of workshop on Collaboration Agents: Autonomous Agents for Collaborative Environments (COLA'03) 2003 IEEE/WIC International Conference on Web Intelligence / Intelligent Agent Technology, See http://www.cs.unb.ca/~ghorbani/cola/program/
3. K. M. Krishna and H. Hexmoor, 2003. Social control of a group of collaborating Multi-Robot Multi-target tracking agents, In Proceedings of 22nd Digital Avionics Systems Conference (DASC-22), Indianapolis, Indiana.
4. K. M. Krishna and H. Hexmoor, 2003. Towards Quantification of the need to Cooperate between Robots, In Proceedings of Performance Metrics for Intelligent Systems Workshop (PerMIS-03), Elena Messina (Ed), NIST, Washington DC.
5. K. M. Krishna, H. Hexmoor, S. Pasapuletti, 2003. Avoiding Collision Logjams Through Cooperation and Conflict Propagation, In proceedings of 1st International Conference on Integration of Knowledge Intensive Multi-Agent Systems (KIMAS-03), page 58-64, H. Hexmoor (Ed), IEEE press, Boston, MA.
6. D. Ward and H. Hexmoor, 2003. Coping with Deception, In proceedings of 1st International

Conference on Integration of Knowledge Intensive Multi-Agent Systems (KIMAS-03), page 94-99, H. Hexmoor (Ed), IEEE press, Boston, MA.
7. P. Poli, and H. Hexmoor, 2003. Effects of Power on Trust and Autonomy, In AAMAS-03 workshop titled: Autonomy, Delegation, and Control, H. Hexmoor (Ed), Melbourne, Australia.
8. D. Ward and H. Hexmoor, 2003. Towards Deception in Agents, In 2nd International Joint Conference on and Autonomous Agents and Multiagent systems, (AAMAS-03), pages 1154-1155, Melbourne, Australia.
9. H. Hexmoor and P. Poli, 2003. Benevolence, Trust and Autonomy, In Proceedings of The International Conference on Artificial Intelligence (IC-AI 2003), pages 808- 814, H. Arabnia (Ed), Las Vegas, Nevada.
10. H. Hexmoor and P. Poli, 2003. Effects of Reciprocal Social Exchanges on Trust and Autonomy, In The Third International Conference on Intelligent Systems Design and Application, Ajith Abraham, Katrin Franke, Mario Koppen (Eds), pages 159-170, Tulsa, Springer.
11. G. Beavers and H. Hexmoor, 2003. Understanding Agent Trust, In Proceedings of The International Conference on Artificial Intelligence (IC-AI 2003), pages 769-775, H. Arabnia (Ed), Las Vegas, Nevada.
12. G. Beavers, and H. Hexmoor, 2003. Types and limits of Agent Autonomy, In AAMAS workshop titled: Computational Autonomy, Potential, Risks, Solutions, Melbourne, Australia.
13. X. Zhang and H. Hexmoor, 2003. Balancing Individual Capabilities and Social Peer Pressure for Role Adoption, In CEEMAS-03, V. Marik, J. Muller, M. Pechoucek (Eds), pages 100-110, Prague, Springer LNAI2691.
14. H. Hexmoor and S. Pasupuletti, 2003. Institutional versus Interpersonal Influences on Role Adoption, In AAMAS workshop titled: Representation and approaches for time-critical resource/role/task allocation, J. Modi and T. Wagner (Eds), Melbourne, Australia.
15. H. Hexmoor and S. Battula, 2003. Adjusting Autonomy and Reliance on Agents in Human Supervised UAV, In Agent-Based Computing (ABC 2003), in conjunction with the 7th World Multiconference on Systemics, Cybernetics and Informatics, Florida.
16. N. Lacey and H. Hexmoor, 2003. Norm Adaptation and Revision in a Multi-Agent System, In Proceedings of the 16th International FLAIRS Conference (FLAIRS-2003), Melbourne, Florida
17. H. Hexmoor and S. Battula, 2003. Towards Collaboration between Human and Social Agents that mind Human Social Personality, In Proceedings of the Fourth International Symposium on Collaborative Technologies and Systems (CTS 2003), pages 205-210, Waleed Smari and William McQuay (Eds), Society for Modeling and Simulation International Volume 35 Number 1, ISBN: 1-56555-258-X, San Diego, Ca.
18. H. Hexmoor and S. Battula, 2003. Discovering Human Desired Norms of Interaction Improves Social Adeptness in Agents, In AAAI Spring Symposium on Human Interactions with Autonomous Systems in Complex Environments, Stanford, CA, AAAI press.
19. H. Hexmoor, S. Battula, 2003. Agent Human Interaction, In Agent Based Computing, Melbourne, Fl.
20. H. Hexmoor, 2002. Social Actions and Influences: An extended abstract, In Proceedings of Computer Human Interaction (CHI-2002) workshop on Socially Adept Agents, S. Marsh (Ed), Minneapolis.
21. H. Hexmoor, 2002. A Model of Absolute Autonomy and Power: Toward Group Effects, In the AAAI-02 workshop on Autonomy, Delegation, and Control: Interacting with Autonomous Agents, Edmonton, Canada.
22. H. Hexmoor, and X. Zhang, 2002. Socially Intelligent Air Combat Simulator, In Proceedings of The Seventh Pacific Rim International Conference on Artificial Intelligence, (PRICAI-

02), pages 365-374, Tokyo, Japan.
23. H. Hexmoor, 2002. Architectural Principles for Social Influence among Benevolent Agents, In proceedings of the AISB 2002 symposium on Adaptive Agents and MultiAgent Systems, p. 120-124, London, UK.
24. N. Lacey and H. Hexmoor, 2002. A Framework for Coherent-Based MultAgent Adaptivity, In proceedings of the AISB 2002 symposium on Adaptive Agents and MultiAgent Systems, AAMAS-02, p. 95- 104, London, UK.
25. N. Lacey, H. Hexmoor, and G. Beavers, 2002. Planning at the Intention Level, In Proceedings of the 15th International FLAIRS Conference (FLAIRS-2002), pages 8-12, Pensacola, Florida.
26. N. Lacey and H. Hexmoor, 2002. Representing and Revising Norms and Values Within an Adaptive Agent Architecture, In Proceedings of The International Conference on Artificial Intelligence (IC-AI 2002), pages 354-360, Las Vegas, Nevada.
27. G. Beavers and H. Hexmoor, 2002. Obligations in a BDI Agent Architecture, In Proceedings of The International Conference on Artificial Intelligence (IC-AI 2002), pages 1334-1340, Las Vegas, Nevada.
28. N. Lacey and H. Hexmoor, 2002. A Constraint-Based Approach to MultiAgent Planning, In 13th Midwest AI and Cognitive Science Conference, MAICS-2002, p. 1- 6, Chicago, Illinois.
29. G. Beavers and H. Hexmoor, 2002. Building Effective Teams, In 13th Midwest AI and Cognitive Science Conference, MAICS-2002, p. 15-20, Chicago, Illinois.
30. H. Hexmoor, and G. Beavers, 2002. Measuring Team Effectiveness, In proceedings of 20th IASTED International Multi-conference: Applied Informatics (AI 2002), p. 338-343, Innsbruck, Austria, ACTA Press.
31. H. Hexmoor, and X. Zhang, 2002. Algorithms for Utility-based Role Exchange, In Proceedings of Intelligent Autonomous Systems, IAS-7, pages 396-403, Marina Del Ray, CA.
32. H. Hexmoor and J. Vaughn, 2002. Computational adjustable autonomy for NASA Personal Satellite Assistant, In proceedings of SAC-2002, Madid.
33. H. Hexmoor, 2001. From Inter-Agent Interaction to Multiagent Social Networks, In Proceedings of ISAI-01, pages 157-164, Rajendra Akerkar (Ed), Allied Publishers, Kolhapur, India.
34. H. Hexmoor, 2000. Situated Autonomy, In Proceedings of ATAL-00, Boston.
35. H. Hexmoor and G. Beavers, 2002. In Search of Simple and Responsible Agents, In the Proceedings of GSFC Workshop on Radical Agents, MD.
36. H. Hexmoor, and X. Zhang, 2001. Utility-based Role Exchange, In Proceedings of CEEMAS-01, Poland.
37. G. Beavers and H. Hexmoor, 2001. Teams of Agents, In Proceedings of the 2001 IEEE Systems, Man, and Cybernetics Conference, Pages 574-582, IEEE press.
38. H. Hexmoor and G. Beavers, 2001. Towards Teams of Agents, In Proceedings of the International Conference in Artificial Intelligence, H. R. Arabnia, (ed), (IC-AI'2001), pages 8-13, Las Vegas, CSREA Press.
39. H. Hexmoor and X. Zhang, 2001. Norms, Roles, and Simulated RoboCup, In 2nd workshop on norms and institutions in multiagent systems, (Agents 2001), Montreal, CA, ACM press.
40. S. Brainov and H. Hexmoor, 2001. Quantifying Relative Autonomy in Multiagent Interaction, In the IJCAI-01 workshop on Autonomy, Delegation, and Control: Interacting with Autonomous Agents, Seattle, WA.
41. H. Hexmoor, 2001. Weakly Dependent Agents May Rely on Their Luck or Collaboration: A Case for Adaptation, In AISB-01, York, UK.
42. H. Hexmoor, and H. Duchscherer, 2001. Efficiency as Motivation for Teaming, In Proceedings of FLAIRS 2001, AAAI press.

43. H. Hexmoor, H. Holmback and L. Duncan, 2001. Detecting, Expressing, and Harmonizing Autonomy in Communication Between Social Agents, AAAI spring symposium on Robust Autonomy, Stanford, AAAI press.
44. H. Hexmoor, 2000. Position: Situated Autonomy, In Proceedings of ATAL-00 Panel on Autonomy, 1 page, Boston.
45. H. Hexmoor, 2000. Position: Empirical versus Formal Approaches to Agent-Based Systems, In Proceedings of Formal Approaches to Agent-based Systems, 1 page, NASA GSFC, MD.
46. H. Hexmoor and Harry Duchscherer, 2000. Shared Autonomy and Teaming: A preliminary report. In Proceedings of Workshop on Performance Metrics for Intelligent Systems, NIST, Washington, DC.
47. H. Hexmoor, 2000. A Cognitive Model of Situated Autonomy, Springer LNAI2112 -pages 325-334, Kowalczk, Wai Loke, Reed, and William (eds).(This is a revised and expanded version of our PRICAI-01 workshop paper held in Australia, http://www.ida.liu.se/~nanre/pricai-team-workshop2000.html)
48. H. Hexmoor, 2000. Towards Empirical Evaluation of Tradeoffs between Agent Qualities, In PRIMA-2000, (C. Zhang and V. Woo, eds), LNAI Volume 1881, Australia.
49. H. Hexmoor, 2000. Conversational Policy: A case study in air traffic control, In Proceedings of International Conference in AI, IC-AI, 2000.
50. H. Hexmoor 2000. Air Traffic Control Agents: Landing and Collision Avoidance, In Proceedings of International Conference in AI, IC-AI, 2000.
51. H. Hexmoor and J. Yang 2000. Pointing: A component of a multimodal robotic interface. In Proceedings of the Workshop in Interactive and Entertainment Robots (Wire-2000), p. 103-107, CMU, PA. http://www.cs.cmu.edu/~trb/wire/
52. H. Hexmoor, 2000. Air Traffic Control and Alert Agent, In Proceedings of Agents 2000, Barcelona, Spain.
53. H. Hexmoor, 2000. Resolving Conflicts Among Actions in Concurrent Behaviors: Learning to Coordinate, In Proceedings of FLAIRS 2000, p. 133-136, AAAI Press.
54. H. Hexmoor, 2000. Case Studies of Autonomy, In Proceedings of FLAIRS 2000, p. 246-249, AAAI Press.

Appendix 1

Abstract

This report describes a multi-agent system where independent software agents interact with each other through a broker to achieve a common goal. Our system consists of satellites (considered to be agent) which send information (military, geographical etc) to ground station(s) (considered to be another type of agent) wherein the received information will be processed. However if we have 'n' number of satellites which send information simultaneously it is difficult for the ground station to receive all the information at the same time, thereby leading to overloading of the system. An intelligent system that would create additional ground stations when required thus maintaining optimal performance of the system is desired. The research work presented here addresses the above-mentioned issue.

The goal of the research was to design an environment that would allow the system to make intelligent decisions to improve system performance. Using the concept of network queuing and allowing the system to add or redirect existing agents (with the help of a broker), we have designed an environment that can optimize system performance and resource allocation. Two software simulations of the multi-agent system were used to observe the effectiveness of the solution. The results of simulation are presented in this report.

1. Introduction

Earth observing satellites also known as remote sensing satellites carry instruments that take remote measurements from space that show what is happening on the earth. One of the main objectives of Earth observing satellites is to obtain images of specified areas on the Earth's surface and then send it back to the ground stations. The images can be used in several environmental, military and geographical applications. While the satellite continuously sends messages, the ground station only receives them when the satellite and ground station are in a direct line of sight of each other. An assumption has been made that the environment is very dynamic and that the orbit and number of satellites are not fixed, neither is the location nor the number of ground stations. Any ground station is capable of receiving data from any satellite as long as they are in line of sight with each other. There are times when no satellite is in line of sight and the ground station is idle and times when several satellites are simultaneously in line of sight and the ground station is saturated with messages and is not capable of processing them in a timely manner. This is when one needs a broker in the system which will optimize the message processing by creating additional ground stations which eases the load on the system and removing the added ground stations when no longer needed by conserving the available hardware resources.

The efficiency of the system has been a primary concern in the design of the systems, which deal with efficient allocation of resources. M. Lemaitre et. al. [1] worked on the problem of finding equitable and efficient allocations of resources resulting from the co-

exploitation of an Earth Observation Satellite by several agents. Taking a centralized perspective, in which decisions are made by an impartial arbitrator, a simple and general modelization of the problem has been set, based on two levels of utility functions: the individual utilities of agents, and the collective utility. Four different procedures for selecting the best allocations have been proposed. The first procedure, allocating satellite revolutions to each agent in turn, is quite perfectly equitable but lacks efficiency. The second one amount to a classical utilitarist perspective: the collective utility function is a linear combination of normalized individual utilities, but the coefficients are chosen in a way to favor equity. The third proposed approach is a genuine bi-criteria approach, allowing comparing allocation over two criteria: efficiency and equity. Finally, an egalitarist approach is used, in which a unique collective utility function is used to characterize equitable and efficient allocations. Above four mentioned procedures as been used by M.Lemaitre et. al. [1], to show the allocation of resources among the different agents must be equitable. But it must also be efficient, that is, the available resources must not be under-exploited. Since our research deals with the real time simulation of Earth Observing Satellites and moreover we are dealing with the efficient allocation of resources, we are making our system efficient by introducing a broker, which takes decisions with the variation of load to maintain the system performance.

Load balancing is done in our system based on slope, which is calculated from queue size and the time; the broker varies the number of active ground stations receiving messages. This load balancing activity has been studied intensively by many researchers in past. Yun Sik Kim et. al. [2] introduced one of the unique traffic features of LEO satellite networks, non-uniform traffic load distribution, and proposed the traffic load balancing scheme to resolve this problem, they uses a near-neighbor residual bandwidth information to apportion excess load from heavily loaded satellites to their under loaded neighbors in the network. Sasa Desic et. al. [3] studied a simulation of a distributed system with capability of load balancing by using the artificial load. They also compared the system's performance using several stationary load balancing methods and the system performance without load balancing methods; he showed that several agent based load balancing solutions have shown significant improvement in system performance in comparison with the system without load balancing method. Adrian Vasilache et. al. [4] developed a simulator for a multiple home agent's architecture in a Mobile IP network. They used a discrete event based simulator it subsequently relies on event analysis and interpretation. The events are processed within the Event Dispatcher, and infinite loop that fetches events from a dedicated queue and triggers the appropriate routine. Each event has a time stamp that allows the placement in the queue according to its occurrence. They also comparatively studied the behavior of several load balancing policies and introduced a more realistic customized double threshold load balancing policy.

2. Background

An Agent is a computational system that inhabits dynamic unpredictable environments. It has sensors to gather data about the environment and can interpret this data to reflect events in the environment. Furthermore, it can execute commands that produce effects in the environment. Multi agent systems are computational systems in which several

autonomous agents interact and work together to perform tasks or satisfy goals. Many researchers are building agents that can work in complex dynamic multi agent domains [5, 6]. Such domains include virtual theater [7], realistic virtual training environments [8, 9, 10, and 6] RoboCup robotic and virtual soccer [11 and 12], and semantic web [13], among others.

Coordinating the actions of the agents is very important because an agent that considers the activities of other agents when forming its own plan is usually better able to choose actions that lead to outcomes that it favors. On the other hand, it is obviously not a good strategy for the agents of a cooperative multiagent team simply to ignore each other, because the intended effects of one agent's action may already have been achieved by the actions of other agents, it is also not a good strategy for each agent to keep track of all the activities of other agents, because the effort required might prevent the agent from doing useful work itself.

Sen et al. [14] studied the effect of limited local knowledge on group behavior for the resource utilization problem where a number of agents are distributed between several identical resources. They concluded that an agent may benefit more from limited knowledge of the environment rather than complete global knowledge. Hogg et al. [15 and 16] analyzed a similar problem and studied the effects of local decisions on group behavior.

Agent request broker is the agent communication mechanism for exchanging knowledge and permitting inter-agent negotiation. Cooperation is the fundamental characteristic of multi-agent system where the overall system exhibits significantly greater functionality than the individual components [17]. In other words cooperation underlines the structure of multi-agent systems [18]. In most literatures cooperation is regarded as the common sense behavior. Few definitions of co-operation have been presented below:

- Definition 1[19]: one agent adopts the goal of another agent. Its hypothesis is that two agents have been designed in advance, and there is no conflict goal between them, furthermore one agent only adopts another agent's aim passively.

- Definition 2 [20]: one autonomous agent accepts another autonomous agent's goal. Its hypothesis is that only cooperation only occurs between the agents, which have the ability of rejecting or accepting the cooperation.

3. Problem Description

As stated, a satellite transfers messages to the ground station only if both are in line of sight and since the orbit and number of satellites are not fixed, there is a lot of variation in the load on the system. There are times when there are many satellites in sight of the ground stations and it is overloaded. There are times when no satellites in sight of the ground stations and are idle. Therefore, at times one needs more ground stations and at times very few depending on the load. Yet ground stations are not capable of making any decisions regarding the load variations. By incorporating a broker in the system we can

provide the system with the necessary decision making capability. We will make broker intelligent enough so that it can change number of ground stations by adding a new ground station to take overload or by killing a ground station when it is no more required to take the load. This results in increase of performance of the system by optimal use of the available resources.

4. Real Time Simulation

4.1 System Overview

In this section we will discuss the satellite simulation (SATSIM) environment that we designed for evaluating the effectiveness of message transmission and receipt by agents. SATSIM was designed to operate as a real-time standalone simulation model and allows the user to configure the environment to generate different amounts of data allowing the analysis of different possible situations. SATSIM was written in the C++ language and uses the SOLID (Interference Detection Library Copyright © 1997, 1998 Gino van den Bergen) library for determining when objects are within line of sight. All agent interactions are through sockets using the TCP/IP protocol. Once the simulation is started, satellites continuously send messages to ground stations; however, ground stations can only receive the messages if they are in line-of-sight. Figure 1 shows the environment with one ground station and two satellites. The red line indicates which agents are insights of each other. As shown in Figure 2 SATSIM has two primary components, an agent and a broker. The broker creates sockets and waits for connections from the agents to establish a communication pattern. Agents are either satellites or ground stations. Ground station parameters include the name of the machine hosting the broker to connect to, the latitude and longitude of the ground station, and a unique ID. Satellite parameters include the name of the machine hosting the broker to connect to, the initial latitude and longitude, the altitude, and a unique ID.

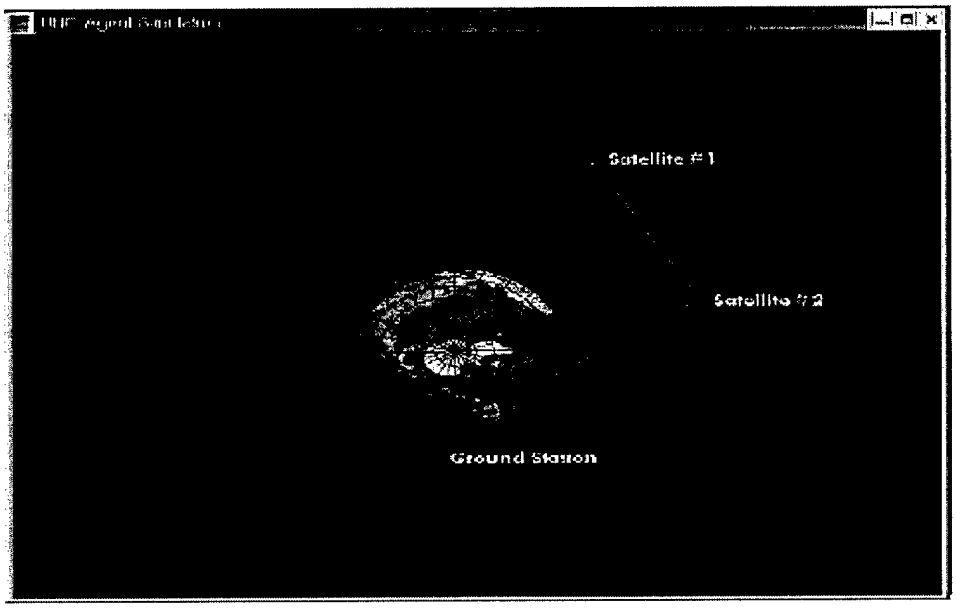

Figure 1: Real Time Simulation Environment.

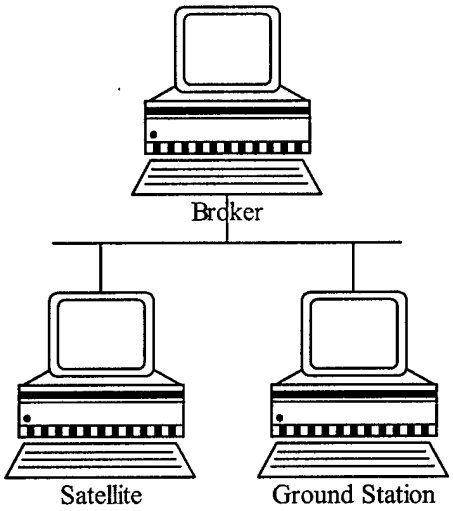

Figure 2: Physical Model Of system

The major problem with a real-time simulation, such as this one, is the non-independence of the agents. Agent independence was obtained by placing the different components on different machines. This, however, resulted in the loss of messages not only due to collisions on the network, but more importantly due to the simultaneous arrival of messages at the broker. We drastically reduce this symptom by applying simple queuing theory of storing all satellite messages received by the broker in a FIFO queue. The messages are then passed on to a ground station for processing, when this message is processed an acknowledgement is sent back to the broker, with the receipt of acknowledgement broker removes a message from the queue and sends it to the ground agent for further processing.

4.2 Real Time Simulation results

Our first investigation was to study the load variation by varying the number of satellites and allowing only one ground station in the system. The purpose of this simulation is to study the system capacity that our current system can handle efficiently without the message loss. As the number of satellites increases, the rate at which the messages are sent from these satellites will also increase and since we have only one ground station, the number of unprocessed messages in queue is also expected to increase results in increased load.

Load balancing was implemented in the following manner; broker receives messages from the satellite(s) and stores in a queue. For every seven messages received the broker calculates the slope and if the slope is greater than 1.22 a new ground station is created for processing messages other wise a ground station is deleted. Then broker sends a message to ground station(s) for processing from the queue. The ground station(s) then

processes the message and sends the acknowledgement to the broker which in turn sends the next message from the queue, this process continues until the queue is empty.
The results shown below demonstrate that as the number of satellites increases from 1 to 6, number of unprocessed messages in queue also increases. But as the number of satellite increases over 6 (as in the case with 8 satellites), number of unprocessed messages decreases instead of increasing. This suggests that there has been loss of messages because of hardware limitations. Hence in our simulation we considered only up to six satellites. The following Figures 4.1, 4.2, 4.3, 4.4 and 4.5 show the variation of queue size over time for different number of satellites.

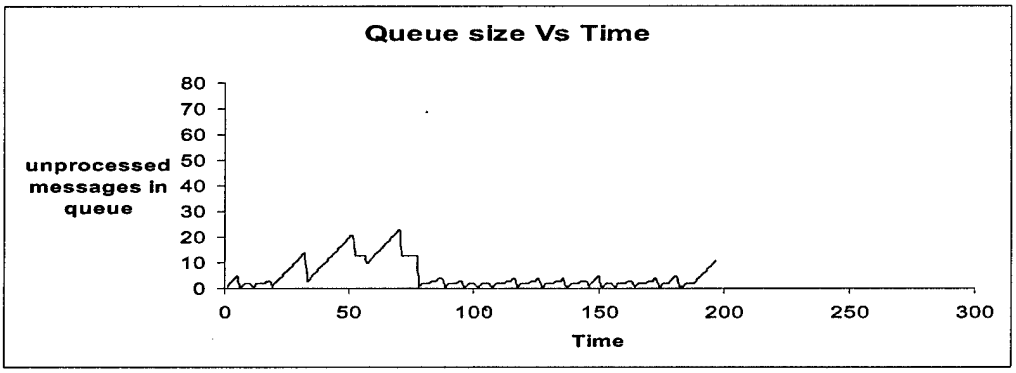

Figure 4.1: One satellite and one ground station.

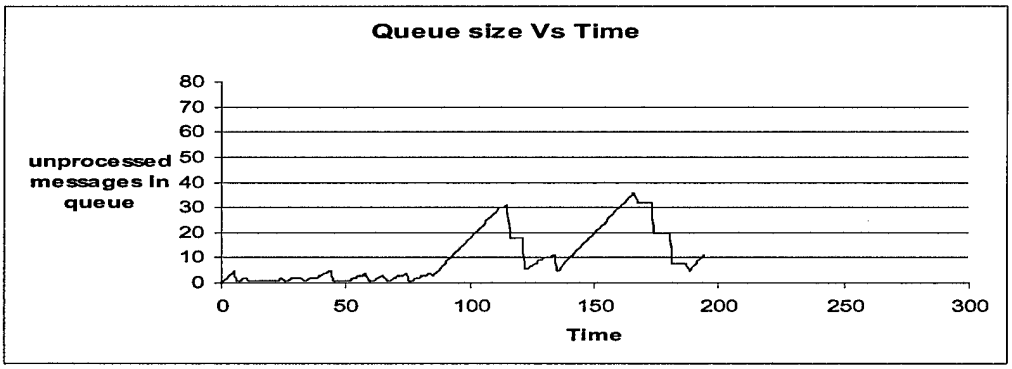

Figure 4.2: Two satellite and one ground station.

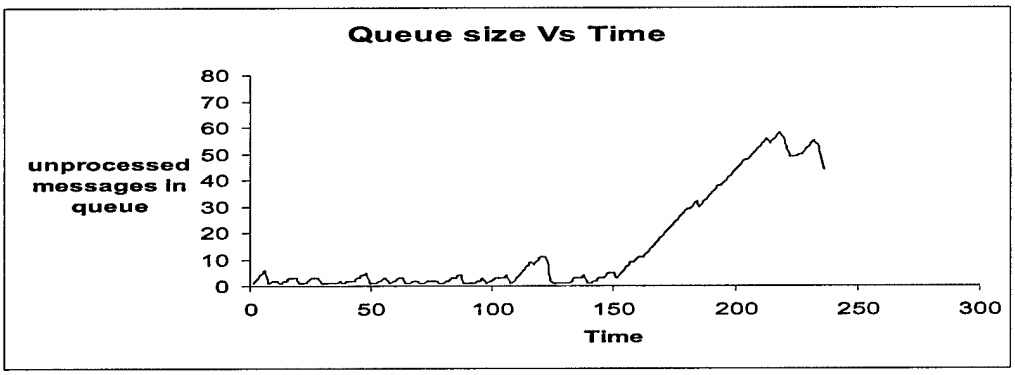

Figure 4.3: Four satellites and one ground station.

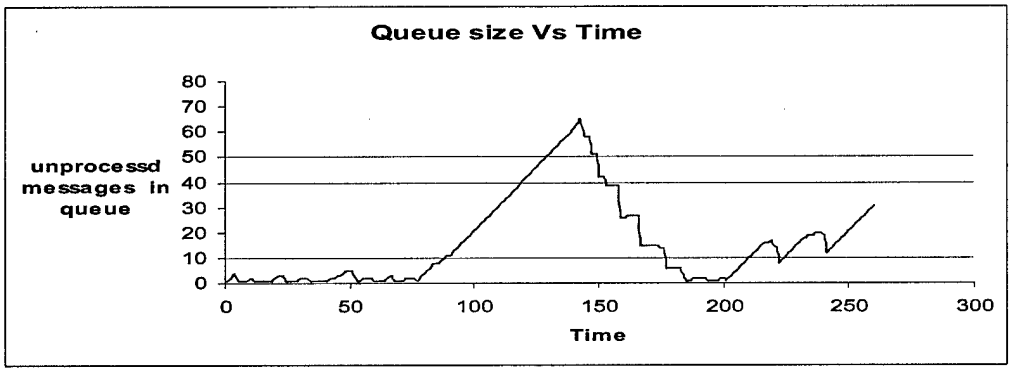

Figure 4.4: Six satellites and one ground station.

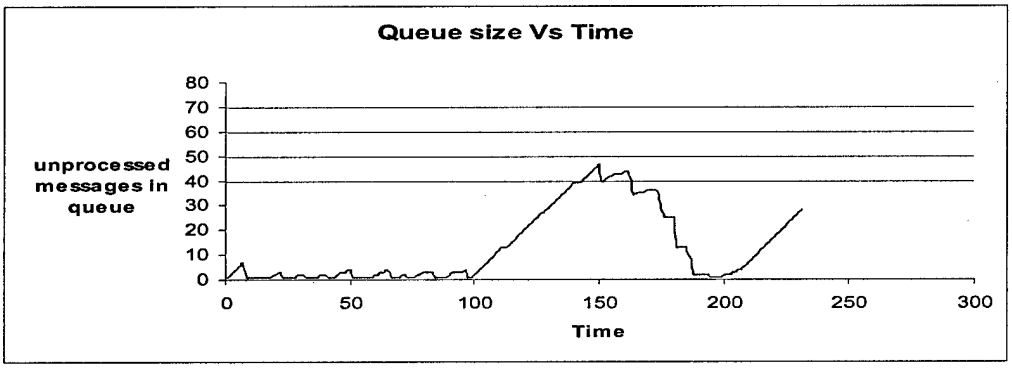

Figure 4.5: Eight satellites and one ground station.

Our second investigation was to make the broker intelligent and allow it to make decisions regarding load balancing. As the number of satellites increases, the size of the unprocessed messages in the queue also increases, since we have only one ground station running at any time, the increase in the queue size cannot be satisfied by one ground station, hence we have to add new ground stations whenever it is necessary to satisfy the increased message traffic, and kill the added stations whenever it is no longer need, to conserve the system resources. We have arbitrarily set the maximum queue size to 28

messages. For any instance of simulation if queue size exceeds 28 messages our simulation is assumed to have failed.

This adding and killing the ground stations is determined by the slope calculated from the queue size and the time. Based on the simulation results we found that the critical slope is 1.22. Using slope value less than 1.22 the system doesn't fail but doesn't work at the optimal performance level either. However, using a slope value greater than 1.22 results in the queue size exceeding 28 messages which results in system failure. The slope is calculated after the arrival of seventh message; this number was determined experimentally. If we calculate slope after the arrival of more than seven messages the system fails (queue size exceeds 28), this is because broker couldn't add the required number of ground stations in a timely manner to avoid the failure. Broker adds the ground station if the slope is above 1.22 and kill the added ground station if the slope is below 1.22.

The results shown in Figures 4.6, 4.7, 4.8 and 4.9 demonstrate the affect of adding ground stations when necessary and killing them when they are no longer needed.

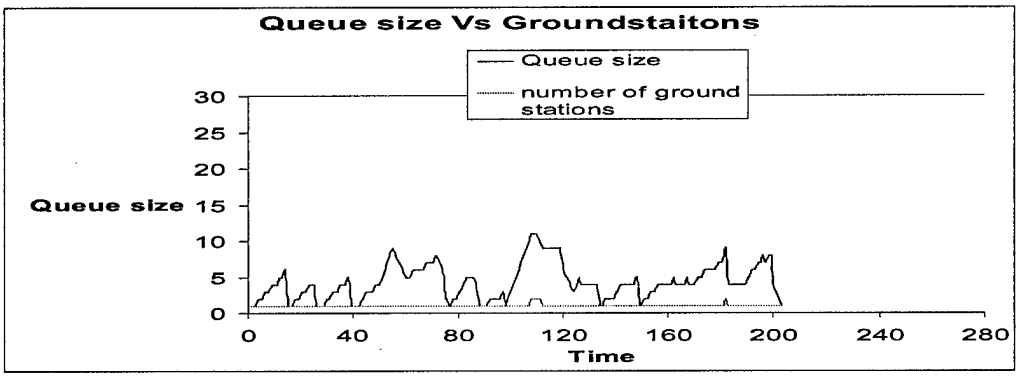

Figure 4.6: For one satellite and multiple ground stations.

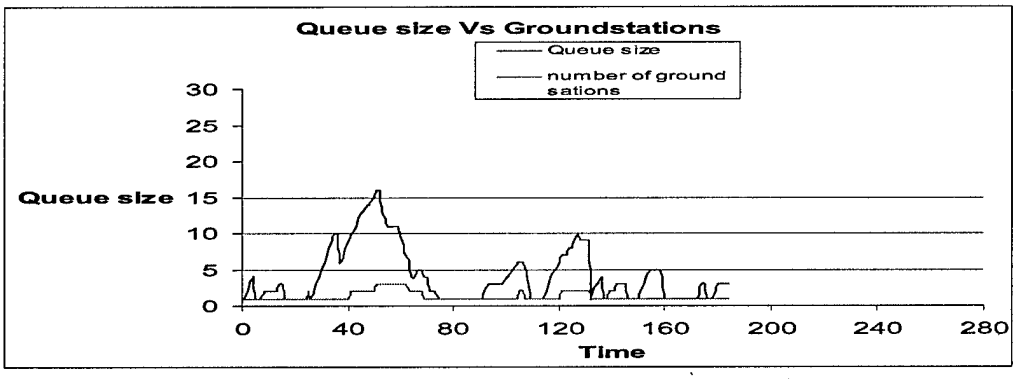

Figure 4.7: For two satellites and multiple ground stations.

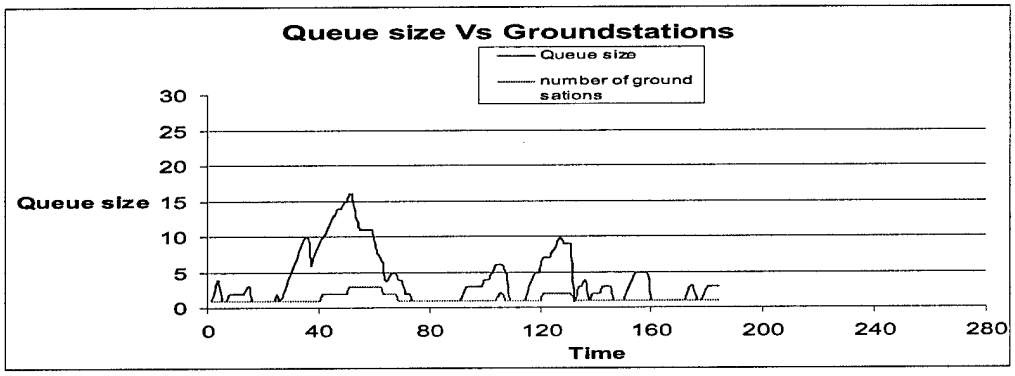

Figure 4.8: For four satellites and multiple ground stations.

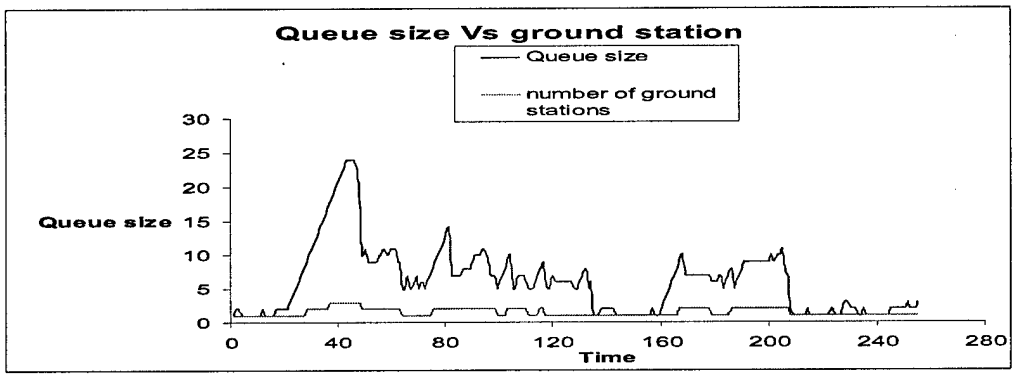

Figure 4.9 For six satellites and multiple ground stations.

The hardware support for this simulation environment was provided by three systems with Red Hat Linux version 7.2 were connected with an 8 port 10/100 fast ethernet hub on a 10 mbps network. Two of the three systems, representing satellite and broker, had the same configuration of Celeron i686 366MHz and the third one was Pentium i586 166MHz executing a ground station.

5. Software Simulation

5.1 System Overview

This simulation is run on a single processor machine, and is concentrated as a single process running at any instance. In this scenario, we simulate the agent side behavioral pattern as to when another agent should be created and when deleted. The messages from the broker are sent to the agent and we simulate the same as the agent receiving the messages from the broker and each message received is added to its queue that collects all the messages to be processed and the messages in queue are deleted once there are processed. Initially the messages coming from the broker are greater in number than the outgoing messages from the queue, and this results in a drastic increase in queue size. This observation helps us to decide the necessity of creating another agent to balance the load using the slope as heuristic. When another agent is created, the messages are processed at higher rate thus decreasing the queue size. Finally there comes a stage when

the queue size is such that one agent can handle the traffic, this the time when decision is taken that other agents should be deleted. A multiple test runs have been made to define the heuristic for agent creation and deletion. The results of the simulation are graphed for better study and understanding. The results are taken by account of one agent spawning just one agent all the through the process and the same agent spawning multiple number of agents depending on the necessity in another process.

The results are formulated based on the implementation that when the messages are created, they will be stored in a queue and for every seven messages created, a slope is calculated and if the slope is greater than 1.22 the delay time for processing messages is reduced other wise increased. The messages are deleted from the queue based on the delay time. This process continues until the queue is empty.

The message generation in this simulation environment is a poisson distribution with varying lambda values between 1 and 6, the same is represented in Figure 5.1, to resemble the message receiving pattern at the broker side of the real time simulation environment. The time for calculating the slope is also duplicated to resemble the real time simulation environment. This simulation was run on an Athlon AMD dual processor system for approximately 2 minutes.

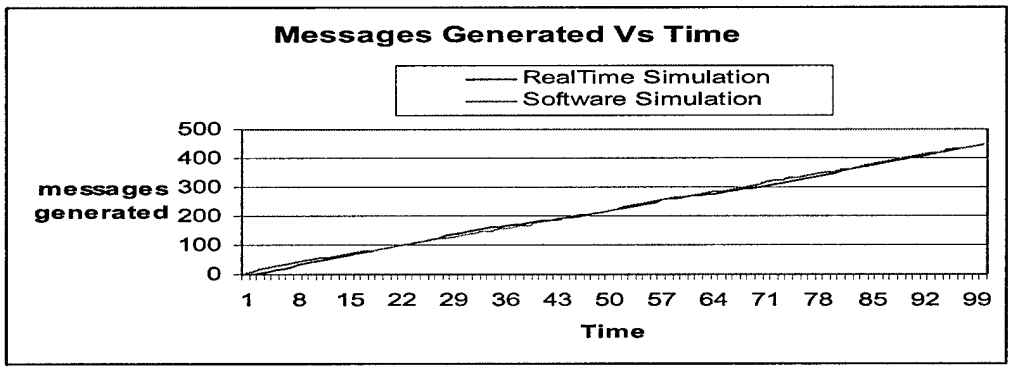

Figure 5.1: Comparison of message generations.

5.2 Software Simulation Results

The results for the software simulation for single ground station are shown in Figure 5.2. Figure 5.3 demonstrates the same when multiple ground stations are operational.

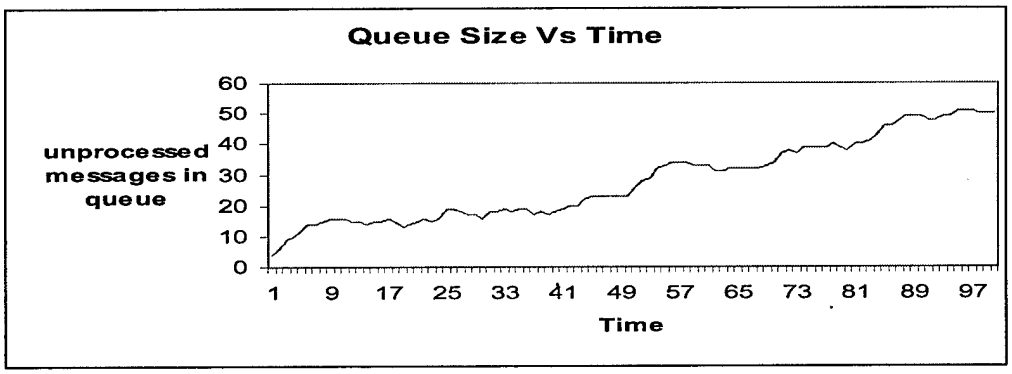

Figure 5.2: For single ground station.

6. Conclusions

Introducing an intelligent broker in the system made the system efficient enough by making decisions in timely manner by better utilizing the available resources. The simulations presented in this paper show the best-performance of the system; based on the real time simulation results we found that the critical slope of the queue size versus time is 1.22 and this was supported by the demonstration of software simulation. Broker adds a ground station if this slope is more than 1.22 to satisfy the increased load on the system and kills non-required ground station if the slope is less than 1.22 to make the optimum use of the available resources. These experiments justify the implementation of decision making broker in our system for increase in performance.

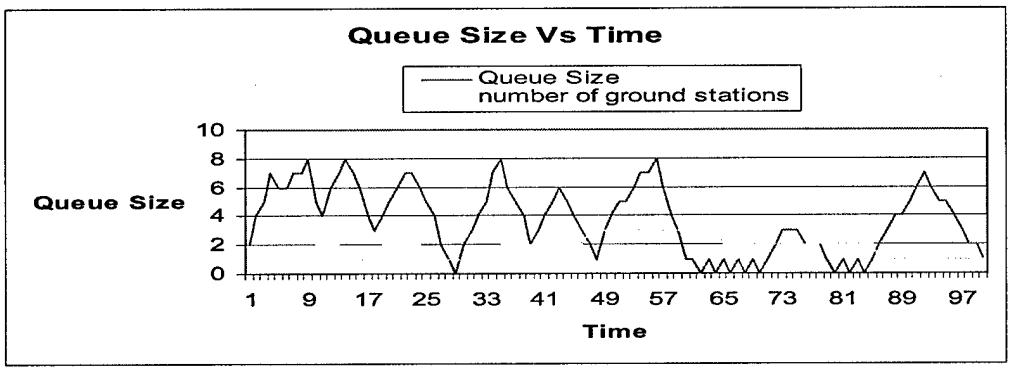

Figure 5.3: For multiple ground stations.

References

1) M.Lemaitre, G.Verfaillie, F.Jouhaud, J.-M. Lachiver and N.Bataille. Selecting and Scheduling Observations of Agile Satellites. Aerospace Science and Technology, (6):367-381, 2002.

2) Yun Sik Kim, Young-Ho Bae, Youngjae Kim, and Chul Hye Park. Traffic Load Balancing in Low Earth Orbit Satellite Networks: Proceedings of IEEE , pages 191-195, 1998.

3) Sasa Desic and Darko Huljenic, Agents Based Load Balancing with Component Distribution Capability: Proceedings of the 2nd IEEE/ACM international Symposium on Cluster Computing and the Grid(CCGRID'02), 2002.

4) Adrian Vasilache and Jie Li and Hisao Kameda . Load Balancing Policies for Multiple Home Agents Mobile IP Networks: Proceeding of IEEE, pages 178-185, Japan, 2002.

5) M.d'Inverno, M.Luck. Making and Breaking Engagements: An Operational Analysis of Agent Relations-hips In Multi Agent Systems: Methodologies and Applications. Lecture Notes in Artificial Intelligence, Xha-ng and Lukose (eds.), pages 48-62, Springer-verlag, Berlin, Germany, 1997.

6) M. Tambe and P.Rosembloom, "Resc: an approach for real time, dynamic agent tracking, "in Proc. Int Joint Cong. Artificial Intelligence, 1995, pp. 95-103.

7) B.Hayes-Roth, L.Brownston, and R Gen, "Multiagent collaboration in directed improvisation, "in proc.Int. Cong.Multi-Agent Systems, SanFrancisco, CA, 1995, pp. 148-153.

8) D. Franklin and K. Hammond, "The intelligent classroom: providing competent assistance, "in proc. 5th Int Conf. Autonomous Agent Systems, Montreal, ON, Canada, 2001. pp. 161-168.

9) T.Hogg and B.Huberman, "Controlling chaos distributed systems," IEEE Trans. Syst, Man, Cybern, Vol. 21.pp. 1325-13332, Nov/Dec. 1991.

10) A.Rao, A.Lucas, and D.Morley, "Agent-Oriented Architecture for Air-Combat Simulation, "The Australian Artificial Intellegence Institute, Tech. Rep. Tech Note 42, 1993.

11) J.Kephart, T.Hogg, and B.Huberman, "Dynamics of computational ecosystems: implications for DAI, "Distrib. Artif.Intell., vol.2,1989.

12) P. Stone and D. McAllestor, architecture for action selection in robotic soccer, "in Proc. Int. Joint Conf. Artificial Intelligence. 1995, pp.95-103.

13) T.Payne, R.Singh, and K.Sycara, "Calendar agents on the semantic web, "IEEE Intel Syst., Vol. 17,pp. 84-86, May 2002.

14) S.Sen, S.Roychowdhury, and N.Arora, "Effects of local information on group behavior, "in Proc. 2nd Int cong. Multi-Agent Systems, Kyoto, Japan, 1996, pp.315-321.

15) K.Pimentel and K.Teixeira, Virtual Reallity:Through the New Looking Glass. New York:McGraw-Hill, 1994.

16) J. Kephart, T.Hogg, and B. Huberman, "Dynamics of computational ecosystems: implications for DAI,"Distrib. Artif. Intell, vol.2,1989.

17) OMG, "The Common Object Request Broker: Architecture and Specification, Revision 1.1", OMG TC Document,(1991).

18) R.G.Smith, R.Davis. Frameworks for cooperation in distributed problem solving. IEEE Transactions on systems, Man and Cybernetics, 11(1): 61-70 (1994).

19) M.Luck, M.d'Inverno, Engagement and Cooperation in Motivated Agent Modeling, Distributed Artificial Intelligence Architecture and Modelling: Proceedings of the First Australian Workshop on Distributed Artificial Intelligence, Zhang and Lukose (eds.), pages 70-84, Springer-Verlag, Berlin, Germany, 1996.

20) Nelson Minar, Kwindla Hultman Kramer, Pattie Maes. Cooperating Mobile Agents for Mapping networks,http://www.media.mit.edu/~nelson/research/routescoopagents/