AGENT BASED CONTROL FOR EMBEDDED APPLICATIONS

S.Sri Eswari Devi, V.Ramachandran
Centre For Professional Development Education
Anna University, Chennai, 600 025.
eswaris_99@yahoo.com,rama@annauniv.edu

Abstract

In real time Embedded Applications such as Missile Guidance, Satellite monitoring systems and Boiler temperature control, the system has to be monitored and controlled continuously without any interrupt. But, because of various factors such as over loading, system failure, resource unavailability, network failure the system may be interrupted and the whole control will collapse. A distributed Mobile Agent model for embedded application has been proposed to overcome these problems. Mobile Agent based control model identifies the critical situation, dynamically search for the best host availability in the clustered network, and allocates the process to the host clustered in the network based on the Best Host Selection Algorithm (BHS). This model comprises multi-agent system where agents negotiate the performance of the processors in the clustered network and mobile agent system to migrate the process and socket to a new host based on the information collected by the multi agent system. To manage the critical situation, the multi-agents and mobile agents render the services such as Yellow page service (YPS), Discovery and location service (DLS), Performance monitoring service (PMS), Fault sensing service (FSS) and Process migration service (PMiS).

Introduction

The Objective of this poster is to construct a distributed software agent model for real time embedded system applications. A network of multi-agents and mobile agents can be used to achieve this. Mobile agent is an autonomous software agent that can migrate from host to host in a heterogeneous network environment [1,4,2]. Here the state of the running process is saved and transported to a new host, and restored, allowing the process to resume from where it is suspended. The mobile agent systems are not merely the process migration systems. The process migration systems [3] do not allow the process to choose when and where they migrate and they move transparently to balance the load. But Mobile agents can move to any host at any time dynamically according to the algorithm given. By reducing the network traffic, mobile agent provides an effective means of overcoming network latency [5,6], and helps to construct more robust and fault-tolerant systems [7]. The Multi-agents in this proposed model are CPU Performance analyzer agent, Memory maintainer agent, Disk Performance Analyzer agent, Network sensor agent, Decision Maker agent, and Yellow Pager agent.

Architecture of Agent Model for Embedded Application:

The proposed model comprises four component levels. The components in this model are

- Distributed Client Application.
- Cluster comprising agent servers and agent clients.
- Controller Area Network (CAN)
- Real- time embedded Devices to be monitored and controlled.

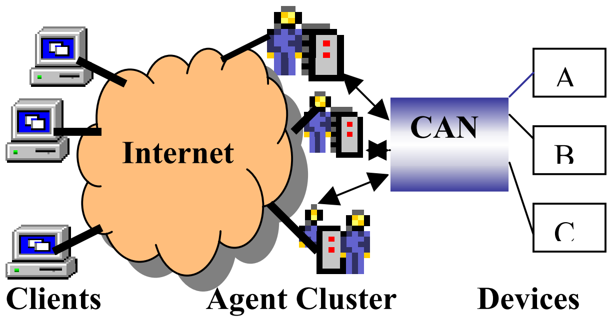


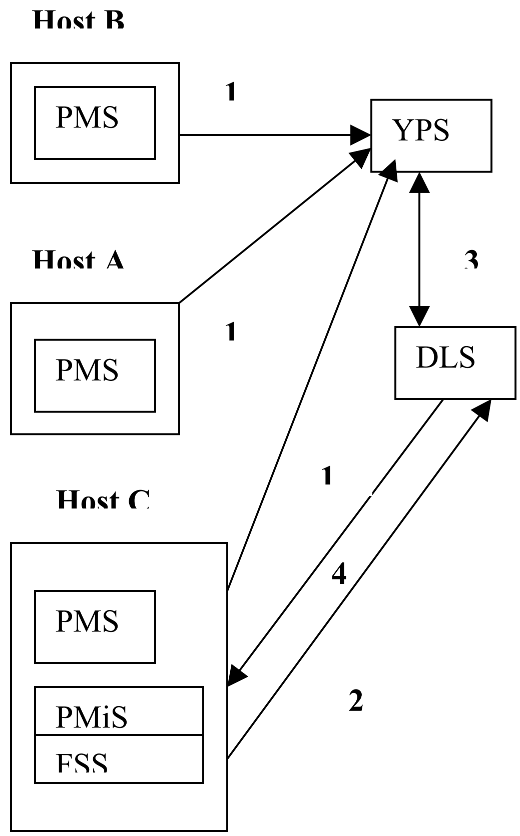
Fig I. Agent Model for Embedded Application

The heart of this architecture is the agent cluster, which encompasses agent servers and agent clients. These agent servers and clients have both static and dynamic agents. The static agents such as CPU Performance analyzer, Memory maintainer, Disk Performance Analyzer and Network sensor provide information about the resource availability, Network availability, and Bandwidth and Network latency. In perilous situations, the proposed Decision Maker Agent roams across the network and discovers the resources and informs the mobile controller agent to migrate to the new discovered host. The proposed Mobile Controller agent controls and monitors the real time devices through Controller Area Network. This agent handles the real time critical devices, which should not be interrupted in any situation.

Services rendered by Agent Clusters:

The agent cluster renders some basic services to monitor and control the devices without interruption. The major services rendered are

- Yellow Page Service (YPS).
- Discovery and Location Service (DLS).
- Performance Monitoring Service (PMS).
- Fault Sensing Service (FSS).
- Process Migration Service (PMiS).



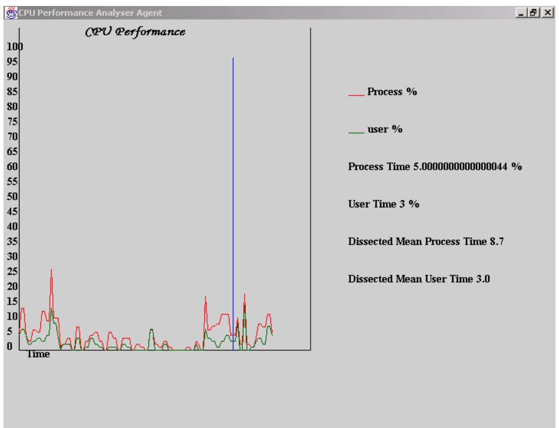
FigII. Communication Flow of agent Cluster

1. The YPS samples the performance of processors in the clustered network, which are collected by the PMS. In any critical situation PMS will report the FSS, the reasons and factors behind the situation.
2. The FSS will notify the fault to the DLS.
3. DLS refers the Yellow pages and makes the decision to shift the process to a new host without any interruption.
4. After making the decision DLS informs the decision to the PmiS .

Factors Considered by the Performance Agents:

Because processor and memory resource have sufficient influence on the operation of the process, it is necessary to monitor the processor and memory and give higher weight to the processor and memory availability. The CPU Performance analyzer agent monitors

1. Processor - %processor Time for processor usage
2. Processor - %User Time
3. Processor - Processor queue length for bottleneck detection

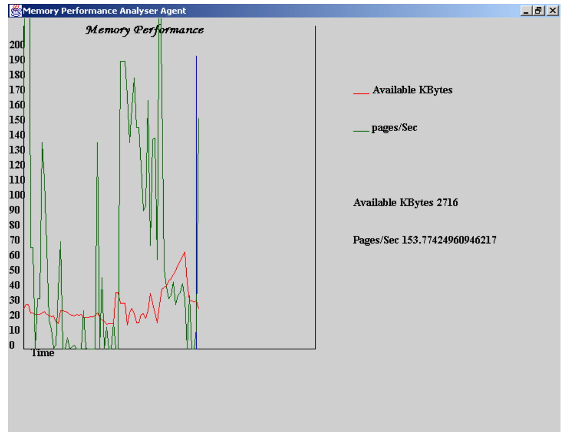


FigIII. CPU Performance Analyzer Agent

The % Processor time shows the percentage of elapsed time that a processor is busy, executing the non-idle thread. To determine whether a processor bottleneck exists due to higher level of demand for processor time, the value of Processor queue length is monitored. When any bottleneck occurs it will be reported to the Fault Servicing Agent. To monitor the low-memory condition the Memory maintainer agent monitors

1. Memory - Available K Bytes
2. Memory - Pages/Sec

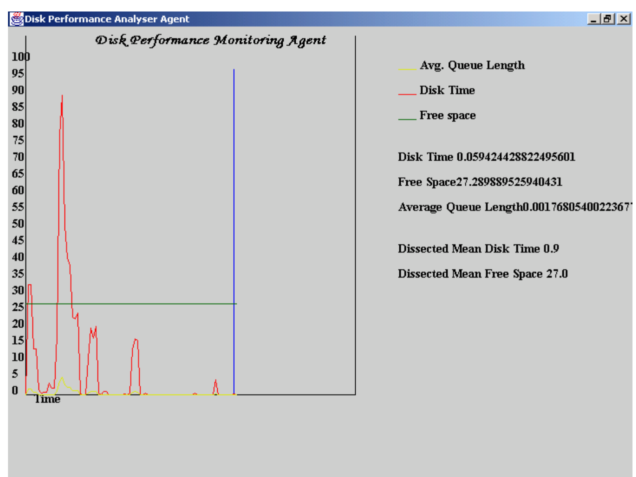
The agent monitors the available K Bytes to find how many bytes of memory are currently available for use by processes. The agent keeps note of the number of pages that were either retrieved from disk due to hard page faults or written to disk to free the space in working set due to page faults by monitoring the Pages/Sec .



FigIV. Memory Performance Analyzer Agent

The agent will become aware of the memory shortage if the system is paging frequently. To determine the disk usage, the Disk Analyzer agent monitors

1. Physical Disk - %Disk Time
2. Physical Disk - Average disk queue length
3. Logical Disk - %Free Space.



FigV. Disk Performance Analyzer Agent

If network traffic exceeds local Area network (LAN) capacity, performance typically suffers across the network. To prevent this situation, the network sensor agent monitors

1. Network Segment - Broadcast frames received/sec
2. Network Segment - %network Utilization
3. Network Segment - Total frames received/second.

The network sensor agent creates a baseline by monitoring the Broadcast frames/second over time. A large variation in broadcast levels leads to lower performance. The agent also monitors the percentage of network utilization to identify the problems such as congestion, flooding that will be caused due to the higher value of network utilization.

Best Host Selection Algorithm:

Whenever any critical situation occurs the process has to be migrated to a new host depending upon this algorithm.

Every Processor analyzer agent analyzes the processor and sends the report to the yellow pages. Consider a clustered network with N processors. Let n be the total number of samples taken by the Processor analyzer agent. The first step is to find the dissected mean for latest m processor values, which may be any integer less than n. A_{PP}, A_{PU} are the dissected arithmetic mean of the Processor - %Process Time and Processor - %user Time respectively.

$$A_{PP}= \sum_{i=n-m}^{n} X_{Ppi} / m$$

$$A_{PU}= \sum_{i=n-m}^{n} X_{PUi} / m$$

Performance Mean measure (PMM) $=W_{PP}A_{PP}+W_{PU}A_{PU}$

Where WPP and WPU are the weight to Processor - %process Time and Processor- %user Time whose value range from 0 to 1. This will be reported to the yellow pages and also variation from Best processor is also conveyed to the yellow pages.

Variation from Best Processor V_P

$$V_P= \sum_{i=n-m}^{n} (PMM_i - A_{lowP})^2 / m$$

Where A_{lowP} is the lowest processor mean measure in the clustered network. Similarly Memory mean measure (MMM) and Variation from Best memory V_M have to be computed and reported to the yellow pages.

$$V_M= \sum_{i=n-m}^{n} (MMM_i - A_{lowM})^2 / m$$

To find the Disk mean measure (DMM), find the dissected mean for Disk Time (A_{DD}) and Logical Free space (A_{DF}). The Disk mean measure is calculated as follows.

$$DMM==W_{DD}A_{DD} / W_{DF}A_{DF}$$

W_{DD} and W_{DF} are the weight to the Disk time and Free Space respectively. Variation from Best Disk V_D has to be calculated and reported to the yellow pages.

$$V_D= \sum_{i=n-m}^{n} (DMM_i - A_{lowD})^2 / m$$

The overall variance can be calculated as

$$V=W_PV_P+W_MV_M+W_DV_D$$

Where $W_P+W_M+W_D=1$

W_P, W_M, W_D are the weight to the processors, memory and disk respectively. The best host is selected based on the low variance value. After the host selection, the network sensor agent tests the network conditions of the host. If the host does not satisfy the minimum threshold network condition, the host is rejected and the network sensor agent acts on the next best host. After the host selection and network sensing, the process is shifted to the new selected host.

Conclusion:

Because of the use of mobile agents, the Bandwidth and latency problem is solved automatically. Hence this model can be used effectively to solve the problems in embedded applications. This can be further extended to the mobile computing applications.

References:

[1]. Danny B. Lange
Mobile Objects and Mobile Agents: The Future of Distributed Computing?
ECOOP Proceedings 1987-2000,
danny@acm.org, http://www.acm.org/~danny

[2]. H. Peine, U. Kaiserslautern
Assessing the Performance Advantage of Mobile Agents in Distributed Searching
In the fourth Plenary Workshops for CaberNet Members, Pisa, Italy, 9-12 October 2001
peine@informatik.uni-kl.d

[3]. David Reilly
Mobile Agents - Process migration and its implications
http://www.davidreilly.com/topics/software_agents/mobile_agents/

[4]. Harrison et al
Mobile Agents: Are they a good idea? March 28 1995 [online] at
http://www.research.ibm.com/massive/mobag.ps

[5]. Aridor, Y. and Lange, D.B.: Agent Design Patterns: Elements of Agent Application Design, In Proceedings of the Second International Conference on Autonomous Agents (Agents '98), ACM Press, 1998, pp. 108-115.

[6]. B. Brewington, R. Gray, K. Moizumi, D. Kotz, G. Cybenko, and D. Rus.
Mobile agents in distributed information retrieval.
In Intelligent Information Agents, pages 355– 395, 1999.

[7]. Nelson Minar, Matthew Gray, Oliver Roup, Raffi Krikorian, and Pattie Maes
Hive: Distributed Agents for Networking Things
August 3, 1999 Appearing in ASA/MA ’99
<nelson@media.mit.edu> http://hive.media.mit.edu/