提供一個高效率 Web 服務環境的代理者合作群

Cooperative Agent Group for a Highly Efficient Web Service Environment

Yuh-Jong Hu（胡毓忠）
Email: jong@cs.nccu.edu.tw
URL: http://www.cs.nccu.edu.tw/\~jong
Emerging Network Technology Lab.
Department of Computer Science ,
National Chengchi University, Taipei, Taiwan

NSC 88-2213-E-004-005

摘要

本研究的原始目的在於利用軟體代理者（包含固定式及行動式代理者）科技來偵測，重設定，甚至於再造 Web 服務環境以提升 Web 服務品質。執行的環境是建立在我們的新趨勢網路科技實驗室中。 用 PCs 來扮演 Web 伺服器，網路中間節點，及 Web 主端電腦。軟體代理者是利用 Java 物件導向程式語言來繕寫。我們利用物件導向程式語言的特性來產生代理者原件並顯示代理者群如何被動的來觀測及主動的來重設定 Web 網路服務的環境。

Abstract

The primary objective of this research is to use software agent (both stationary agent and mobile agent) technology to monitor, reconfigure, and even reengineer Web services environment for promoting quality of Web services. The simulated Web services environment was set up in our ENT(Emerging Network Technology) lab with several PCs to play Web servers, network interior nodes, and Web clients. The software agents were coded in Java object-oriented programming language to take the OO language characteristics to generate our agent entity. We shown how our agent entity group passively monitoring and actively reconfiguring this Web service environment.

1. Introduction

Internet-based WWW is increasing very rapidly for the past few years. This research project was based on software agent technology to monitor the WWW access workloads, which including network transmission traffic, Web servers access rates, and other Web-related operations. The software agents passively monitor the high volume of WWW access, exchange the associated network and server traffic status. Furthermore, software agents can actively react, tune, and block these traffics to do the autonomous workload balance and optimization. Original software agents in this research were covered both stationary agent and mobile agent. Due to lack of suitable mobile agent system engine for security access mechanisms operations, we only consider stationary agent in our cooperative agent group for Web performance monitoring and tuning.

2. Background

This project was designed and implemented via three undergraduate students under the information project course. One student was a project leader, who is responsible for the overall workload balancing so he artificially generate workload traffic from several client machines to simulate the real world end-user web access requests.
The other two students were doing Web servers log-file analysis and network traffic monitoring. The objective of this project is to balance the WWW network and servers traffic via software agents. There already lots of agent-enabled applications developed in the research community but very little agent-enabled workload balancing applications available.

In this research, the software agents might be classified as stationary agent or mobile agent. For stationary agents, they are located at each Web server to observe the Web servers' traffic regularly or irregularly to balance the servers' workload. These workload balancing mechanisms were autonomously execute via software agents and it is transparent to the end users. On the mobile agent side, they were doing pure network web traffic balancing. Each mobile agent was launched from client machine with time-marked flag in each mobile agent entity. As long as the mobile agent arrives at the web server destination, the mobile agent's travelling time were collected to evaluate the network congestion status.

In general, the slow down of Web access time might be dichotomized as two factors: network communication delay or Web server performance degraded. Our research is to show the end-user what are the possible factors that downgrade Web server access performance. If possible, through agent's cooperation to react and balance traffic jam workload autonomously. In our study, we also found out that the mobile agent security is one of the important issues must be solved before mobile agent techniques can be really applied to the real world load balancing operations. Without proper mobile agent authentication and authorization mechanisms, the network router and gateway can not provide the services for foreign incoming mobile agent to do the monitoring and configurable operations. So in our simulated workload balancing environment, we did not use a full power of mobile agents to do the network traffic monitoring and reconfiguration. Instead, we use stationary agent in each interior routing node to play the mobile agent's simulated operations. Besides, we did not have a reliable mobile agent system engine to do the mobile agent security so only a sequence of simple request messages were sent from client machines to pass through the interior routing nodes. These simple request messages were configured as mobile agents to simulate the mobile agents' operations. Finally, the Web server compute each request message travelling time based on the time difference between message sending and receiving clock time.

3. Workload Balancing Simulated Environment

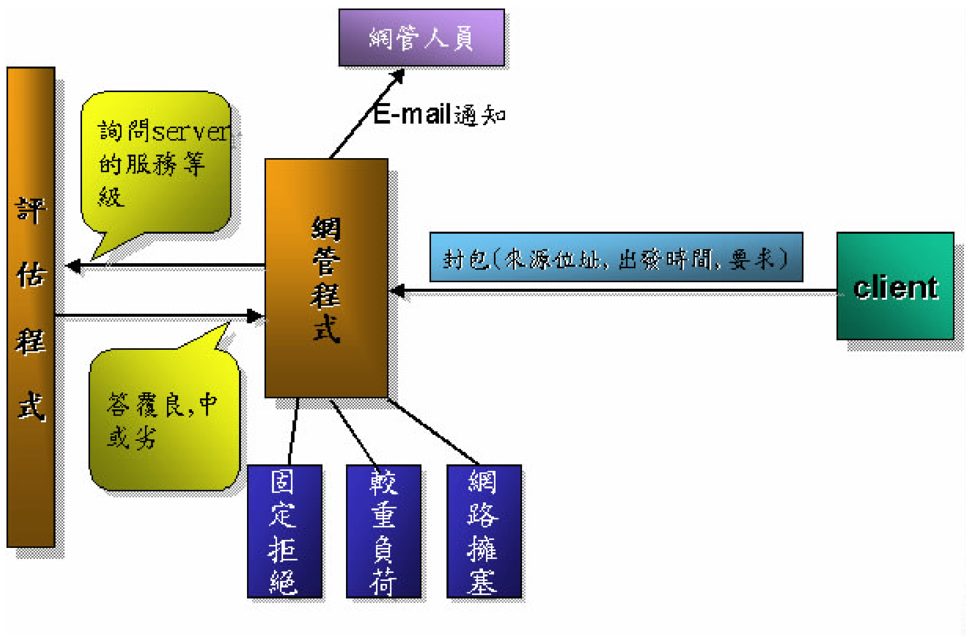
This project's workload simulated environment was implemented in our ENT (Emerging Network Technology) lab. There are total 12 Intel-based PCs in this simulated environment. Three client machines were constantly generating Web server access requests via their local stationary agents to remote side 3 Web servers. For a period of time, these agents in the client machines show the performance of the Web server access for end-user with understandable figure charts. The retrievable contents in the Web servers are Taiwan film database with several different information data format. The remaining PCs are configured as network interior routing nodes to route the Web server request messages from client machines in an optimal manner. All the workload optimization were done via software agents but we did not exclude the manual-based optimization for system administrator.

In this Web service environment, both Web server agents and network routing agents work together to pinpoint the possible performance bottleneck and show the result on client machines in understandable figures for end-user. The possible improvement of traffic jam via the following several mechanisms:

- Block client IPs with high and intensive workload requests.
- Tune Web server acceptable workload level based on its own capacity.
- Balance Web server workload with multiple mirror Web servers .
- Route client machine's requests via less congested routing path to alleviate network traffic jam.

4. Filter agents to enable/disable Web services

One of the special filter agents was proposed in this study to reconfigure the Web services flexibly. This filter agent is similar to firewall access control program, which allows Web server to enable and disable Web services based on its own service capacity and request sending agent's request workload level. More detailed see the following figure:



5. Conclusion

Agent-based workload balancing optimization is our research objective. The Web services bottleneck can be pinpointed via our agent entity group's cooperation. Some of agent entities are network performance observers, while the other agent entities are Web servers' status observers. These agents cooperate with each other to find out what are the real performance degraded factors and show the end-user in their client machines. This study is very similar to some of the Web browser's Web server access status report. The only difference is our agent entities can proactive to disable/enable (or reconfigure) some of the Web services to reduce/increase the Web server workload. We did not consider mobile agent operations in this study due to the lack of secure mobile agent system engine with associated authentication and authorization mechanisms to reconfigure the interior routing nodes. Instead, stationary agents were located at each interior nodes to reconfigure the routing path. So to use similar active network packet's mobile agent to reconfigure might need further study.

6. References

(1)Arnold, Ken and Gosling, James *The Java Programming Language,* Addison-Wesley, 1996.

(2)Meer, de Hermann, et al, Tunnel Agents for Enhanced Internet QoS, *IEEE Concurrency,* pp. 30-39, April-June 1998.

(3)Wu, K.-L., P.S. Yu, and A. Ballman, SpeedTracer: A Web usage mining and analysis tool," *IBM System Journal*, Vol. 37, No. 1, 1998.

(4)Heidemann, John, et al., Modeling the Performance of HTTP over Several Transport Protocols, ACM/IEEE Transactions on Networking, 5 5, 616-630, Oct., 1997.

(5) Joe, Touch, et al., Analysis of HTTP Performance, http://www.isi.edu/lsam/publications/http-perf/

(6)An Expalnation of SPECWeb96 Benchmark, http://www.specbench.org/osg/web96/webpaper.html

(7)World Wide Web Server Benchmarking, http://www.sgi.com/products/WebForce/WebStone/

(8)Mobile Object Systems Toward the Programmable Internet, Edited by Jan Vitek and Christian Tschudin, Springer-Verlag, 1997.