분산 시스템의 성능 모니터링과 레포팅 툴의 아키텍처 모델링*

김 기**, 최은미**

Distributed System Architecture Modeling of a Performance Monitoring and Reporting Tool

Ki Kim, Eunmi Choi

Abstract

To manage a cluster of distributed server systems, a number of management aspects should be considered in terms of configuration management, fault management, performance management, and user management. System performance monitoring and reporting take an important role for performance and fault management. In this paper, we present distributed system architecture modeling of a performance monitoring and reporting tool. Modeling architecture of four subsystems are introduced: node agent, data collection, performance management & report, and DB schema. The performance-related information collected from distributed servers are categorized into performance counters, event data for system status changes, service quality, and system configuration data. In order to analyze those performance information, we use a number of ways to evaluate data corelation. By using some results from a real site of a company and from simulation of artificial workload, we show the example of performance collection and analysis. Since our report tool detects system fault or node component failure and analyzes performances through resource usage and service quality, we are able to provide information for server load balancing, in short term view, and the cause of system faults and decision for system scale-out and scale-up, in long term view.

Key Words: Distributed System, Performance Monitoring and Reporting, System Architecture Modeling, Performance Evaluation, Workload Simulation

* 본 논문은 한국 시뮬레이션 학회 2003년 춘계학술대회에서 발표한 내용을 수정, 보완한 것임.

** 한동대학교 전산전자공학부

1. 서론

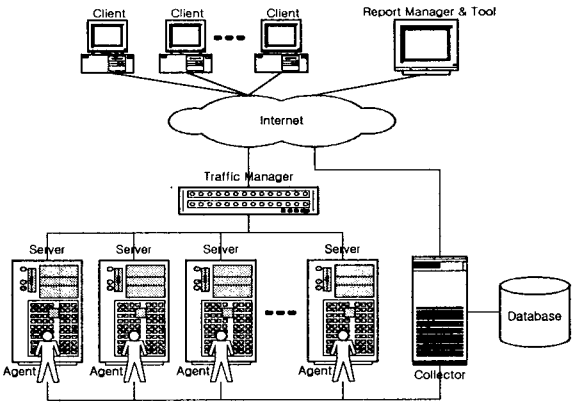
확장되는 인터넷의 사용자 층과 다양해지는 서비스 범위들로 인하여 서비스를 제공하는 서버 단의 클러스터화는 필수 불가결한 사항이 되어가고 있다. 분산 상에서의 클러스터를 사용함으로써, 인터넷 서비스를 제공하는 서버들을 하나의 형태 (SSI: Single System Image)로 이루어주어, 기하급수적으로 증가하는 인터넷 사용자들의 서비스 요청에 부합하게 처리하도록 서버 단의 성능 향상을 기대하게 되었다. 이 외에도 확장성 (Scalability), 고가용성 (High Availability), 경제성 (Cost-Effectiveness), 장애대처 능력 (Fault Tolerance)을 가진 고성능 시스템 (High Performance System)으로 클러스터 시스템의 기대 효과는 높아지고 있다[1,2]. 이러한 분산 시스템을 구성하고 있는 여러 서버들을 관리할 때, 시스템 설정(Configuration) 관리, 장애(Fault) 관리, 성능(Performance) 관리, 보안(Security) 관리, 사용자(User) 관리 등의 관리 측면이 고려되어야 한다[6]. 시스템 성능을 모니터링하고 레포팅을 하는 것은 서버관리의 기본적인 부분이며 장애 발생의 원인과 추세 파악, 성능 최적화, 서버추가 혹은 자원 확장의 판단근거를 제공하게 된다[4].

분산 시스템 상의 성능 모니터링을 하는 목적은 이를 통하여 시스템 성능을 측정 및 평가하고, 시스템 튜닝에 효과적이며, 성능의 병목 현상을 찾아내며, 시스템과 서비스에 따른 부하 현상을 찾아낼 수 있게 된다. 또한 시스템의 Capacity Planning 과 예상되는 부하에 따른 성능의 기대치를 유추해 낼 수 있다[7]. 이를 위하여 가상적인 부하를 시뮬레이션하여 서버 시스템에서 영향을 미치는 현상들과 전체 클러스터에게 파급되는 성능 결과를 연구하는 것이 중요하다.

본 논문에서는 우선적으로 모니터와 보고서 툴의 분산 상에서의 시스템 아키텍처를 Subsystem별로 나누어 설계 구조를 소개한다. 이어서 주요하게 고려되는 성능 정보들과 이벤트 정보, 서비스 정보와 시스템 정보를 설명하며, 시스템에서의 효과적인 성능 모니터링과 분석 방법을 제시한다. 또한 다양한 부하 상황 속에서의 시스템 성능 평가 및 유추를 하기 위하여 실사이트 운영 결과와 시뮬레이션 결과를 보인다.

2장에서는 성능 모니터링과 레포팅 툴의 구조도를 각 Subsystem별로 나누어 시스템 모델링을 설명한다. 3장에서는 분산 클러스터 시스템 성능 모니터링을 위하여 수집하는 정보를 설명하며, 4장에서 성능 정보 분석과 평가 방법론을 소개한다. 5장은 실제 레포팅 툴을 이용한 결과와 6장은 시스템 평가의 고려 사항들을 살펴본다. 7장에서는 관련연구와 비교를 설명하고 8장에서는 본 논문의 결론을 내린다.

2. 모니터링과 성능 레포팅 툴의 구조도



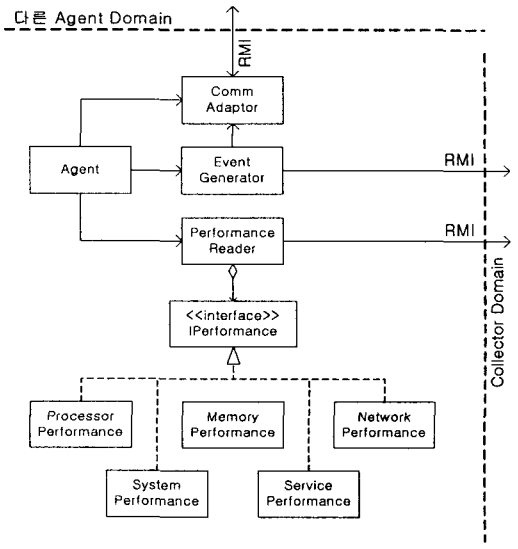
<그림 1> 클러스터 스템의 성능 레포트를 위한 구조도

클러스터 시스템은 외부에서 들어오는 서비스 요청을 앞단의 Traffic Manager를 통하여 각 서버들에게 분산시킴으로 서비스를 제공한다. 클러스터 시스템의 성능 모니터링 시스템은 서비스를 제공하는 서버노드 단의 Agent Subsystem, 각 서버 노드에 설치된 Agent로

부터 데이터를 수집하는 Collector Subsystem, 데이터 수집을 관리하고 보고서를 생성하는 Subsystem, 수집된 정보를 저장하는 데이터베이스 Subsystem으로 나누어진다[11,12].

2.1 서버노드 단의 Agent Subsystem

성능 모니터링은 각 서버 노드에 설치된 Agent를 통하여 이루어지며 다음의 클래스들로 구성되어서 일을 처리하게 된다(그림2).

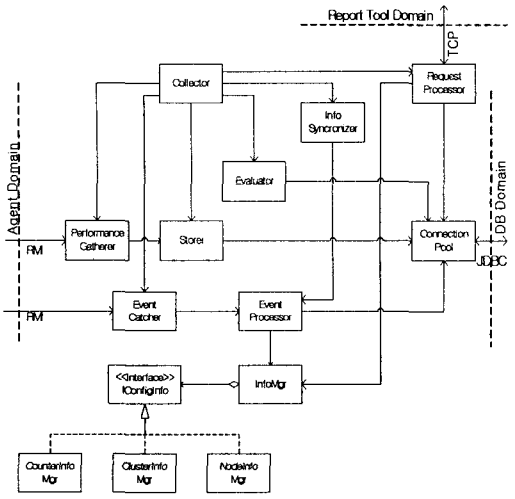


<그림 2> 서버노드 단에서의 모니터링 관련 클래스 구조

1) **Agent**: 클러스터 시스템의 각 노드에 설치되어 동작하며, 능동적으로 운영 중에 발생하는 이벤트와 장애 상황을 Event Generator에게 전달을 한다. 또한 Agent는 주기적으로 성능수치를 노드로부터 읽어서 PerformanceReader에게 전달한다.
2) **EventGenerator**: 상태변경, 장애등 운영에 필요한 사건에 대해 이벤트를 발생하여 RMI를 통하여 Collector에게 전달한다.
3) **CommunicationAdaptor**: 노드간의 정보와 이벤트를 주고 받는다.
4) **PerformanceReader**: 각 성능수치를 담당하는 성능 객체를 IPerformance를 통하여 호출하여 부분별 성능수치를 수집하여 RMI를 통하여 Collector에게 전달한다.

2.2 데이터 수집 Subsystem

Collection부분에서 하는 일은 각 서버노드의 Agent로부터 이벤트와 성능수치를 수집하고, 수집 내용을 데이터베이스에 저장하여 보고서 작성에 쓰이도록 한다(그림3).



<그림 3> 데이터 수집관련 클래스 구조

1) **Collector**: 데이터를 수집하기 위한 Collection 부분에서 최초로 초기화되어 모듈들을 생성하고 관리한다.
2) **Request Processor**: 사용자에게 보고서 생성 요청과 수집관리에 관한 요청을 받아 처리할 수 있는 TCP통신을 제공하여 해당하는 정보 제공과 명령을 처리한다.
3) **PerformanceGatherer**: RMI를 통하여 각 Agent들로부터 성능수치를 수집한다.
4) **EventCather**: 각 서버 Agent에서 발생된 이벤트를 수집하여 Event Processor에게 넘겨준다.
5) **EventProcessor**: 수집 내용을 데이터베이스에 저장하고 클러스터와 노드에 관련된 변경된 정보를 InfoMgr에게 보낸다.
6) **Storer**: 가공되지 않은 수집된 그대로의 성

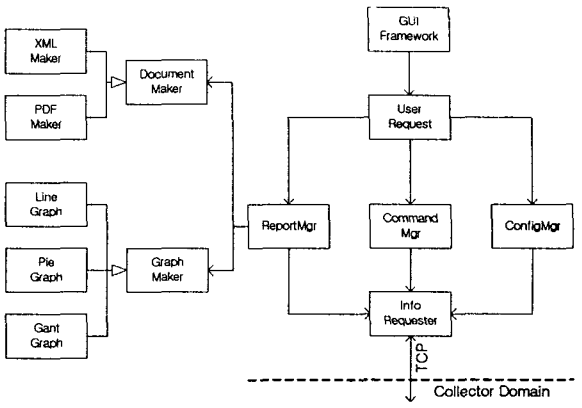
능수치를 ConnectionPool을 통하여 데이터베이스에 저장한다.

7) **Evaluator**: 가공되지 않은 성능수치를 분, 시간, 일 단위의 시간에 따라 분석데이터를 만들어 별도의 데이터베이스 스키마에 저장한다.
8) **InfoSynchronizer**: 실제로 운용되고 있는 클러스터 환경과 Collector에 저장되어 있는 환경을 비교하고, 수집하고 있는 시스템의 정지, 네트워크 문제로 인하여 발생될 수 있는 불일치를 주기적으로 확인하여 정보를 동기화 한다.
9) **InfoMgr**: EventProcessor에게 받은 정보를 바탕으로, 가지고 있는 클러스터, 노드 등의 환경정보를 갱신하고 관리한다.
10) **CounterInfoMgr**: 성능수치의 목록과 단위 등의 정보를 관리한다.
11) **ClusterInfoMgr**: 클러스터의 목록과 서비스 종류, 클러스터에 속한 노드들의 목록 등의 정보를 관리한다.
12) **NodeInfoMgr**: 노드의 목록과 시스템 자원, 운영체제 등의 정보를 관리한다.
13) **ConnectionPool** : 자료처리 속도향상을 위하여 데이터베이스에 연결하는 커넥션들을 pool로 만들어 제공한다.

2.3 수집관리 및 보고서 생성 Subsystem

사용자에게 노드별로 수집할 성능수치와 수집주기를 설정하는 기능을 제공하며 분석을 위한 보고서를 만든다(그림4).

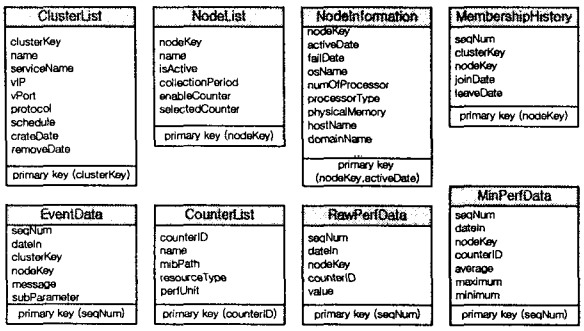
1) **GUI Framework**: 사용자에게 보고서 생성과 수집관리를 할 수 있도록 인터페이스를 제공한다.
2) **UserRequest**: 사용자의 request와 옵션을 분석하여 command와 parameter를 만들고 유지하여 사용자의 request에 따라서 ReportMgr, CommandMgr, ConfigMgr에게 일을 전달하여 준다.
3) **CommandMgr**: 사용자에게 수집되기 원하는 성능수치 선택과 수집주기를 설정할 수 있는 command를 만들어서 Info Requester에게 전달한다.
4) **ConfigMgr**: 사용자에게 클러스터, 노드목록등 클러스터 시스템의 기본적인 정보와 설정을 제공하며 현재의 클러스터 시스템의 전체 configuration을 저장한다.
5) **ReportMgr**: 사용자에게 받은 request와 옵션을 토대로 데이터를 요구하며 보고서를 생성하도록 DocumentMaker와 Graph Maker에게 정보를 전달한다.
6) **DocumentMaker**: 보고서를 문서로 생성 제공하며, **XmlMaker**는 XML 형식으로, **PdfMaker**는 PDF 형식으로 생성한다.
7) **GraphMaker**: 보고서에 필요한 그래프를 생성한다. 세부 클래스로는 **LineGraph, PieGraph, GantGraph**가 있다.



<그림 4> 수집관리 및 보고서 생성 관련 클래스 구조

2.4 데이터베이스 Subsystem

1) **ClusterList, NodeList, CounterList** : 클러스터, 노드, 성능수치의 목록과 정보를 저장한다.
2) **RawPerfData** : 실시간 성능수치를 저장한다.
3) **MinPerfData, HourPerfData, DayPerfData** : 성능수치를 분, 시간, 일 간격으로 평균값, 최대값, 최소값을 계산하여 저장한다.
4) **EventData** : 서버에서 발생된 장애상황, 상태변화의 이벤트를 저장한다.



<그림 5> 데이터베이스 스키마

5) MembershipHistory : 클러스터와 노드의 소속 정보를 저장한다.

3. 성능 모니터링을 위해 수집하는 정보

성능 모니터링을 위하여 수집하는 정보는 성능 정보, 이벤트 데이터 정보, 서비스의 질, 시스템의 정적인 정보로 구성이 된다[11].

3.1 성능 데이터

• 프로세서 성능 데이터

Processor Load	프로세서가 사용된 시간의 백분율
User Time	사용자 모드에서 소비한 프로세스 시간의 백분율
Privileged Time	특권 모드에서 명령 실행하면서 경과된 시간의 백분율
Interrupt / sec	프로세스가 받아 처리한 하드웨어 인터럽트의 초당 발생 횟수

• 메모리 성능 데이터

Available Memory	실행되는 프로세스에 사용할 수 있는 실제 메모리의 양
Page Fault / sec	초당 페이지 부재의 평균 수
Page Read / sec	하드 페이지 부재를 해결하기 위해 디스크를 읽은 비율
Page Write / sec	실제 메모리의 공간을 비우기 위해 페이지를 디스크에 쓴 비율

• 네트워크 성능 데이터

TCP Connection	TCP 연결 개수
TCP Connection Failure	TCP 연결에 실패한 개수
Datagrams Received / sec	UDP 데이터그램을 받은 비율
Datagrams Sent / sec	UDP 데이터그램을 보낸 비율
Bytes Received / sec	네트워크 어댑터에서 받는 바이트의 비율
Bytes Sent / sec	네트워크 어댑터에서 보내는 바이트의 비율

• 시스템 성능 데이터

Processor Queue Length	프로세서 대기열에 있는 쓰레드 수
Context Switches / sec	모든 프로세서가 한 쓰레드에서 다른 쓰레드로 전환한 전체 횟수
Processes	프로세스의 수
Threads	쓰레드의 수

• 서비스 성능 데이터

Application에 따라 수집하는 성능수치는 달라질 수 있으며 Window 2000 Server 의 IIS Web Server와 관련된 성능수치를 예로 든다.

Request / sec	초당 실행한 요청의 수
Web Bytes Recived / sec	서비스가 받은 데이터 바이트의 비율
Web Bytes Sent / sec	서비스가 보낸 데이터 바이트의 비율

3.2 이벤트 데이터

• 상태변화 이벤트

Node Active	새로운 노드가 클러스터 시스템에 추가됨을 나타낸다.
Node Fail	노드가 시스템이나 전원, 네트워크 등의 문제로 클러스터 시스템으로부터 이탈을 나타낸다.
Node Overload	노드의 상태가 과도한 서비스 요청으로 overload 상태에 빠지게 됨을 나타낸다.
Node Underload	노드의 상태가 overload 상태에서 정상의 상태로 돌아옴을 나타낸다.

• 운영 이벤트

Cluster Created	새로운 클러스터가 생성됨을 나타낸다.
Cluster Removed	클러스터가 제거됨을 나타낸다.
Node Join to Cluster	노드가 클러스터에 Join하여 서비스하게 됨을 나타낸다.
Node Leave from Cluster	노드가 클러스터에서 빠지게 됨을 나타낸다.

• 서비스 이벤트

Service Start	서비스가 시작되어 클라이언트로부터 요청을 받아 들이게 됨을 나타낸다.
Service Stop	서비스가 중지되어 클라이언트로부터 요청을 받아 들일 수 없음을 나타낸다.
Service Pause	기존 세션의 서비스는 계속 수행하지만 새로운 세션은 받아들이지 않음을 나타낸다.
Service Resume	서비스가 Pause 상태에서 Start 상태로의 복귀를 나타낸다.

3.3 서비스의 질

Service Response Time	클라이언트가 클러스터 시스템에게 서비스를 요청하여 응답을 받는 시간을 표현하기 위해 각 서버 노드에게 주기적으로 서비스 요청을 보내어 응답시간

3.4 서버 노드 정보

OS Name	서버 노드의 운영체제
Number of Processor	서버 노드가 가지고 있는 CPU의 개수
Processor Type	CPU의 종류
Processor Clock Speed	CPU의 Clock Speed
Physical Memory	서버 노드가 가지고 있는 Memory의 크기
File System Type	파일 system의 종류
Host Name	서버 노드의 Host Name
Domain Name	서버 노드가 속한 Domain

4. 성능 정보 분석과 평가 방법

시스템 성능 정보를 분석하기 위하여 일정 시간의 시스템의 정보를 샘플로 채택하여 분석하여 시스템의 성능 경향을 평가할 수 있다. 정보 분석에는 성능 수치와 시간과의 관계성, 수치 간의 관계성, 이벤트와 다른 요소들과의 관계성들로 평가를 한다[11].

4.1 성능수치 vs. 시간

가장 기본적인 정보 수치로써, 시간의 변화에 따른 성능수치의 변화를 분석한다. 고려되는 성능수치는 자원사용, 서비스 요청, 서비스의 질이 될 수 있다. 이 분석을 통하여 특정 자원의 시간에 따른 사용 변화, 부족한 자원의 서비스에 미치는 영향, 서비스의 요청의 변화, 서비스 요청이 집중되는 시간대, 서비스 응답 시간의 변화, 부하분산을 측정하기 위한 여러 서버 노드의 성능수치 비교 등을 할 수 있다.

4.2 이벤트 vs. 시간

시스템에서 일어나는 상태변경, 장애 발생의 이벤트를 발생 시간 순서와 종류별로 분석한다. 이를 통하여 시스템이 장애 상황까지의 상태 변화 과정을 Error Trace이나 Status Trace를 함으로써 문제의 원인을 찾을 수 있으며, 장애 상황 발생의 추세를 파악한다.

4.3 성능수치 vs. 성능수치

성능수치간의 상관관계를 분석함으로써, 자원과 자원의 관계, 서비스 요청과 자원의 관계, 서비스 요청과 서비스 질과 관계, 서비스의 질과 자원의 관계를 분석하여 파악 할 수 있다.

4.4 성능수치 vs. 이벤트

성능수치와 이벤트간의 상관관계를 분석할 수 있으며, 연관성을 측정하여, 이벤트가 시스템에 미치는 성능 영향을 분석할 수 있다.

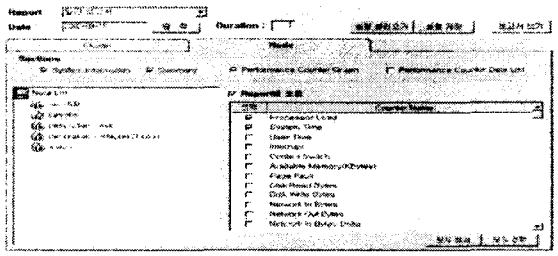
4.5 이벤트 vs. 이벤트

이벤트간의 상관관계를 분석한다. 이 분석을 통하여 특정한 이벤트 발생 전후 일정 시간 내에 발생한 다른 이벤트의 빈도조사, 특정 두 이벤트가 일정 시간 내에 발생한 빈도 조사,

특정 시간 내에 발생한 이벤트의 빈도 조사, 특정 두 이벤트가 발생한 시간 간격 조사, 여러 이벤트가 순차적으로 발생한 추이 패턴의 빈도를 조사하여 분석 할 수 있다.

5. 성능 모니터링 레포팅 툴의 결과

성능 모니터링 레포팅 툴을 구현하여 실사이트에 적용하여 나온 결과를 소개한다. 웹 검색 서비스를 제공하는 E사의 성능 모니터링 일간 보고서를 통하여, 클러스터 시스템의 정보, 가용성, 부하분산, 자원사용, 서비스 요청의 시간 분포 등을 측정하고 보여준다[10].



<그림 6> 보고서 생성을 위한 GUI

5.1 시스템의 정보

클러스터의 서비스 종류, 생성 일시와 서버노드들의 이름, OS, 프로세서에 대한 정보를 알 수 있다. <그림 7>에서는 웹 서비스를 하기 위하여 5개의 서버 노드가 포함된 것을 나타낸다.

Information

Service Application	Web Server
Create	2003-02-07
Destroy	Continue
Up Time	2일 0시간 0분 0초
Virtual IP	10.1.1.201
Port	0
Protocol	TCP

Member Nodes

Node Name	Operating System	Processor
Web Searcher1	Linux v2.4.18	Pentium 4 1794MHz (1개)
Web Searcher2	Linux v2.4.18	Pentium 4 1794MHz (1개)
Web Searcher3	Linux v2.4.18	Pentium 4 1794MHz (1개)
Web Searcher4	Linux v2.4.18	Pentium 4 1794MHz (1개)
Web Searcher5	Linux v2.4.18	Pentium 4 1794MHz (1개)

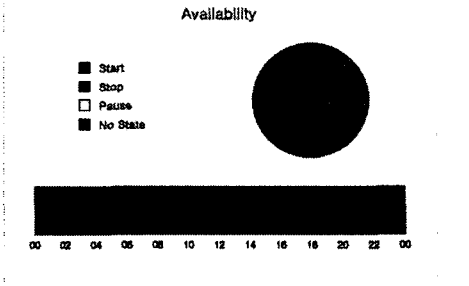
<그림 7> 클러스터 시스템의 정보

5.2 클러스터 시스템의 가용성

전체 클러스터 시스템의 가용성은 (서비스 가능시간/총시간)으로 가용률을 계산하여 나타낸다. <그림 8>은 가용률 100%의 예이다. 가용성의 상태를 Start, Stop, Pause, No State의 4가지 종류의 이벤트를 이용하여 표현하며 파이 그래프에서는 각 상태의 비율을, 막대 히스토그램 그래프에서는 각 상태의 발생, 지속 시간을 볼 수 있다. <그림 8>은 클러스터 시스템이 24시간 동안 Start 상태로 지속됨을 보여준다.

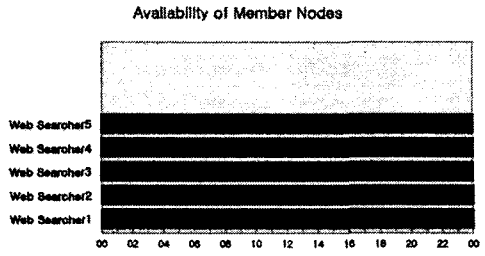
Cluster Availability

Total Time	Run Time	Ratio
86400 (sec)	86400 (sec)	100 (%)



<그림 8> 클러스터 시스템의 가용성(E사)

클러스터 시스템의 속한 서버 노드들의 가용성은 속해 있는 각 노드의 막대 히스토그램 그래프로 보여준다. <그림 9>는 모든 노드들이 24시간 동안 Start상태로 지속되었음을 보여준다.



<그림 9> 각 노드별 가용성(E사)

5.3 부하분산 측정

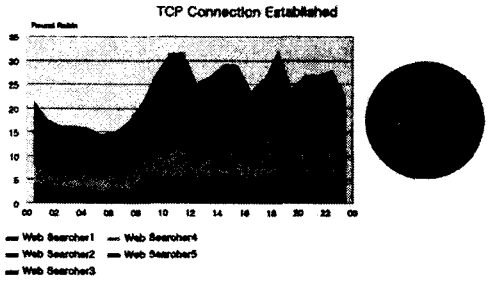
부하 분산에 사용된 스케줄링 방식을 사용한 기간과 함께 목록으로 보여준다. <그림 10>에서는 Round Robin방식을 2003-02-07부터 사용했음을 볼 수 있다.

Scheduling Methods

Method Name	From	To
Round Robin	2003-02-07 05:06:16	Continued

<그림 10> 클러스터 시스템의 부하분산 방식(E사)

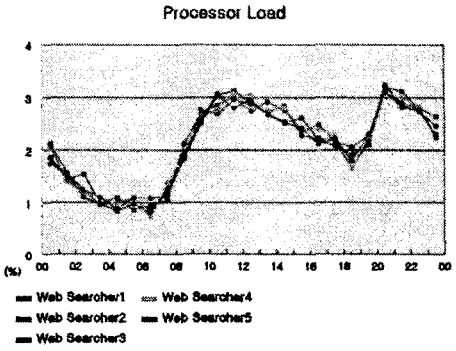
모든 TCP 연결을 통한 서비스 요청을 각 노드들이 시간별로 얼마만큼 분담했는지 그래프의 면적을 보고 알 수 있다. <그림 11>에서는 각 시간별로 면적이 균등하고 파이 그래프의 면적이 각각 20%대로 일정하므로 부하분산이 잘 이루어지고 있음을 알 수 있다.



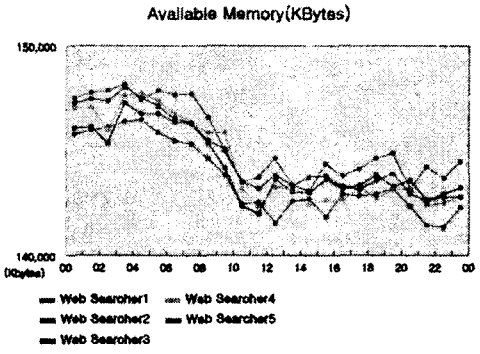
<그림 11> 클러스터 시스템의 부하분산 (E사)

5.4 자원 사용

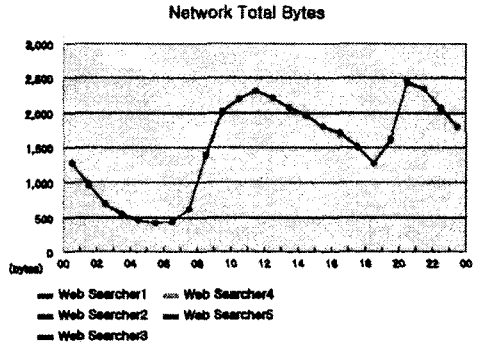
<그림 12>에서 <그림 14>까지의 결과 그래프에서 나타난 프로세서의 로드가 최대 4%, 사용 가능한 메모리의 양이 최소 140MByte, 네트워크 사용량이 최대 2.5KByte를 넘지 않으므로 주요한 자원이 서비스를 제공하는데 부족함이 없는 것으로 판단 할 수 있다[3].



<그림 12> 자원사용 - Processor Load (E사)



<그림 13> 자원사용 - Available Memory (E사)

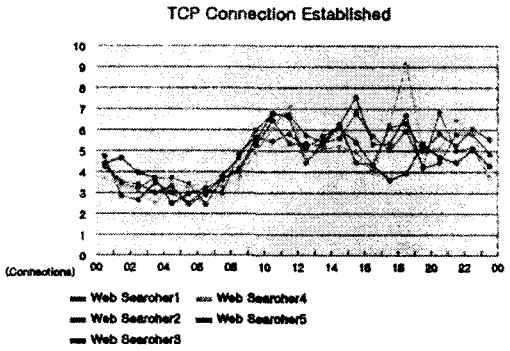


<그림 14> 자원사용 - Network Total Bytes (E사)

5.5 서비스 요청

E사의 서비스 특징은 TCP/IP를 통해 서비스를 제공하므로 TCP 연결의 분포가 곧 서비

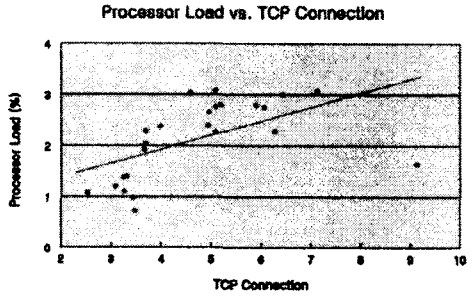
스 요청의 분포와 동일하다고 볼 수 있다. <그림 15>에서, 06시부터 서비스 요청이 증가하여 12시에 정점에 이르며 00시까지 꾸준한 서비스 요청을 처리하고 있음을 확인 할 수 있다.



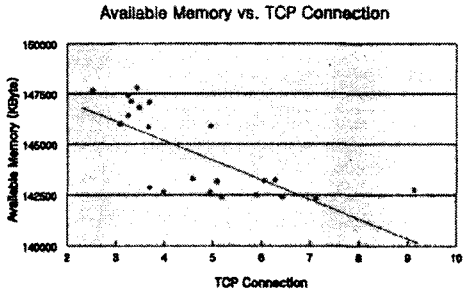
<그림 15> 자원사용 - TCP Connection (E사)

5.6 성능의 상관관계 분석

E사에서 측정한 시스템 성능들을 분석하기 위하여 성능간의 상관관계를 볼 수가 있다. <그림 16>과 <그림 17>에서 Linear Regression을 통하여 TCP Connection수가 증가함에 따라서 다른 성능 요소가 영향을 받음을 평가한 것이다. 상관관계를 통하여 알 수 있는 것은 사용자 수가 증가함에 따라서 서버단의 Processor 사용량이 증가하며, 잔류 Memory양이 감소함을 알 수가 있다.



<그림 16> TCP 연결 수와 Processor Load 간의 상관관계



<그림 17> TCP 연결 수와 Available Memory와의 상관관계

6. 성능 분석을 통한 시스템 평가

6.1 자원사용

서버 노드의 자원사용의 내역은 3.1절에서 소개한 여러 가지 성능 데이터들을 이용하여 평가할 수 있다. 성능 분석을 통하여 자원사용의 추세와 최대 사용량을 파악하고, 자원의 부족으로 인한 서비스의 제약의 평가와 scale-up (각 서버 노드의 자원의 한계를 확장하는 것) 혹은 scale-out(클러스터에 서버 노드를 추가함으로 성능을 확장하는 것)의 판단 기준으로 삼을 수 있다.

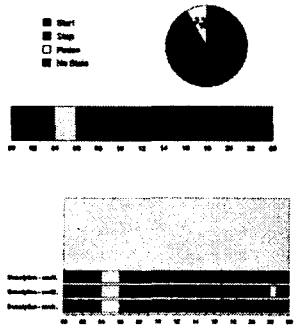
6.2 확장성

서비스와 밀접한 관계를 가지는 자원의 사용량이 최대 임계값을 넘는 경우는 서버성능을 scale-up하거나, 혹은 서버의 일부 자원의 확장만으로는 서비스를 증가 할 수 없을 경우와 클러스터 시스템의 가용성을 높이고자 하는 경우엔 서버 개수를 scale-out의 결정 기준으로 삼을 수 있다[9].

6.3 가용성

가용성에 관한 결정은 이벤트 vs. 시간 분석을 통한 클러스터 시스템의 서비스의 가용성 자료와 노드들의 가용성 자료를 통한 분석이 가능하다. 전체 클러스터 시스템의 가용률(A_C)

은 각 노드의 가용률을 이용하여 추정할 수 있다. 클러스터 시스템의 가용률이 낮으면 노드를 새로 추가하여 잠재적 서비스 실패 가능성을 낮출 필요성이 있다[8].



$$A_N = \frac{T_{NS}}{T_{NS} + T_{NF}}$$

A_N : 노드의 가용성
T_{NS} : 노드의 서비스 가능 기간
T_{NF} : 노드의 장애 기간

$$A_C = 1-(1-A_N)^m$$

AC : 클러스터의 가용률
m : 클러스터의 노드개수

<그림 18> 시스템 가용성에 대한 예와 수식

6.4 부하분산

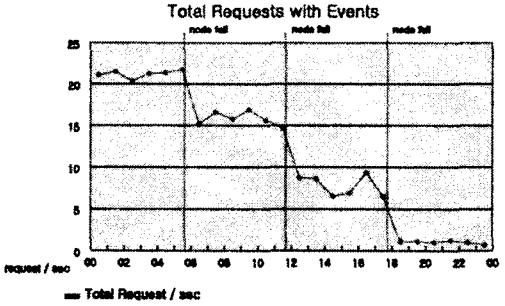
클러스터 시스템에서는 Round Robin, Least Connection, Weighted RR, Weighted LC 등의 부하분산 방식을 사용하여 클라이언트로부터의 요청을 서버 노드들에게 분산시키게 된다. 부하분산 방식의 결정과 부하분산이 어떻게 이루어졌는지 분석하기 보통 서비스 요청수의 성능수치를 이용하며, Web Service를 하는 클러스터 시스템의 예에서는 TCP Connection 수치 등을 이용하여 분석이 가능하다.

6.5 서비스의 질

클러스터 시스템의 서비스의 질을 나타내는 한 방법으로써 클라이언트 입장에서의 서비스 요청에 대한 서버의 응답시간을 고려할 수 있다. 서비스의 질을 떨어뜨리는 요인이 일정 시스템 구성 요소의 병목현상에 의하는 경우가 대부분이므로 각 요소들의 성능을 측정하고 상태를 평가하는 정책을 수립할 수 있게 된다.

6.6 장애 추적 및 시스템 관리

클러스터 시스템이 장애에 빠지는 이유를 파악하기 위해서는 각 컴포넌트들이 어떤 동작, 상태나 설정의 변경, 어떤 문제 봉착 시에 이벤트를 발생시키고 그것들을 수집함으로써, 각 단계마다 자원사용의 상황을 같이 고려하여 시스템의 상태와 원인을 파악해야 한다. 앞에서 언급한 이벤트 vs. 시간 분석을 통해 이벤트들의 history와, 이벤트 vs. 이벤트 분석을 통해 이벤트간의 연관성(Event Correlation)과 근본 원인(Root Cause)을 파악할 수 있으며, 이벤트 vs. 성능수치 분석으로 이벤트가 발생한 시점에서의 시스템의 상황을 파악 할 수 있다. <그림 18>은 노드의 Fail로 인하여 클러스터 시스템 전체의 성능이 저하되는 상황을 이벤트와 성능 수치의 관련성을 이용하여 추적하여 가는 예를 보여주고 있다.



<그림 19> 노드 fail로 인한 성능저하 상황 추적

7. 관련 연구

WatchTower[13]는 Windows NT/2000 OS가 관리하는 성능수치에 쉽게 접근을 제공하며 모니터링 소프트웨어에 쉽게 삽입되게 하였다. perfmon 정도의 overhead와 필요한 정보로 정리 수집된 데이터의 크기를 축소시킨다. 유사한 연구로써 Windows cluster를 대상으로 하는 HPVM monitor[14]가 있으며 통신

과 clock을 포함한 모니터링 하부구조를 가지고 있다. SHRIMP PerformanceMonitor[15]는 클러스터 상의 어플리케이션을 대상으로 성능 정보를 수집하며 adaptive clock 동기화 알고리즘을 구현 하고 있다. Paradyn[16]은 장기간 동안 큰 규모의 병렬 프로그램과 분산 프로그램을 대상으로 하며, 병목현상을 일으키는 가능성 있는 성능 수치를 자동으로 찾아준다.

본 논문에서 제안하는 시스템은 SHRIMP와 Paradyn과 같이 분산 클러스터 시스템을 대상으로 하여 여러 서버 노드의 모니터링을 원격에서 제어와 분석이 가능하다. 또한, Java로 작성되어 다양한 Platform에서도 agent를 설치하여 모니터링을 수행하는 이식성을 제공한다. 이외에도, 성능수치를 주기적으로 가공하여 데이터의 크기를 축소시키고 여러 가지 데이터를 표현하는 분석 도구를 제공하여 관리자가 다양한 방법으로 분석할 수 있도록 하였다.

8. 결론

본 논문에서는 분산 서버 클러스터 시스템의 성능 관리와 장애 관리를 위한 성능 모니터링과 레포팅 툴을 소개하였다. 이 시스템 아키텍처의 구성도를 각 Subsystem 별로 모델링 하였다. 노드단의 Agent, 데이터 Collection, 성능 관리와 레포트, DB 스키마 부분으로 나누어, 성능을 측정하고 평가하기 위한 시스템 아키텍처 구조를 설명하였다. 또한, 시스템의 성능을 분석하기 위해 수집해야 하는 여러 가지 정보 요소를 보였으며, 수집된 데이터를 분석하기 위한 방법으로 성능수치와 시간, 시스템 이벤트들의 상관관계를 이용을 하였다. 실사이트의 자료를 이용하여 모니터링과 레포팅 툴의 결과를 보았으며, 일정한 장애 상황을 추정하였다. 이러한 성능 분석 툴 관리를 통해 자원사용, 확장성, 가용성, 부하분산, 서비스의 질 등을 평가 할 수 있으며 이상 상황 추적과 성능 병목 현상을 파악하여 시스템 튜닝 결정에 효과적인 처리를 할 수 있다.

참고문헌

[1] Andrew S. Tanenbarum, Maarten van Steen, Distributed Systems principles and Paradigms , Prentice Hall, 2002

[2] Rajkumar Buyya, High Performance Cluster Computing vol. 1 , Prentice Hall, 1999

[3] "Monitoring and Tuning Your Server", Microsoft Technews, http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/iis/reskit/iis50rg/iischp5.asp

[4] Zeisler, Varma, Wallace, Kalich, Xion, "TMN CORBA Matures: A Service Provider Gateway for Measuring Service Availability", IEEE Network Operations and Management Symposium, Vol.2 1998, pp374-380

[5] Mark Grand, "Patterns in Java, Volume1, A Catalog of Reusable Design Patterns Illustrated with UML" , John Wiley & Sons, INC., 1998.

[6] Henz-Gerd, Sebastian Abeck, Bernhard Neumair, "Integrated Management of Network Systems", Morgan Kaufmann, 1998.

[7] Raj Jain, "The Art of Computer Systems Performance Analysis", Wiley, 1991

[8] Wensong Zhang, Shiyao Jin, Quanyuan Wu, "Creating Linux Virtual Server", LinuxExpo 1999 Conference, http://www.linuxvirtualserver.org/ols/lvs. ps.gz

[9] Wensong Zhang, "Linux Virtual Server for Scalable Network Services", Ottawa Linux Symposium 2000, http://www.linuxvirtualserver.org/ols/clvs.ps.gz

[10] 김기, 최은미, "클러스터 시스템의 효과적

인 성능 모니터링과 레포팅", 제19회 한국정보처리학회 춘계학술발표대회 논문집 제10권 제1호, pp133-136, 2003.

[11] 김기, 최은미, "클러스터 시스템의 성능 레포트 툴의 아키텍처 모델링", 2003 시뮬레이션학회 춘계 학술대회 논문집, pp67-71, 2003.

[12] 김기, 최은미, "분산 시스템의 효과적인 웹 클러스터 성능 모니터링과 분석", 제20회 한국정보처리학회 추계학술발표대회 논문집, 2003.

[13] M. W. Knop, P. K. Paritosh, P. A. Dinda, and J. M. Schopf, "Windows performance monitoring and data reduction using watchtower", Technical Report NWU-CS-01 -6, Department of Computer Science, Northwestern University, June 2001.

[14] G. Sampemane, S. Palkin, A. Chien, "Performance monitoring on an hpvm cluster", In Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA '00), June 2000.

[15] C. Liao, M. Martonosi, and D. W. Clark. "Performance monitoring in a Myrinet- connected Shrimp cluster.", In Proceedings of the SIGMETRICS Symposium on Parallel and Distributed Tools (SPDT), August 1998.

[16] B. P. Miller, M. D. Callaghan, J. M. Cargille, J. K. Hollingsworth, R. B. Irwin, K. L. Karavanic, K. Kunchithapadam, and T. Newhall, "The paradyn parallel performance measurement tools.", In IEEE Computer, pages 37-46, Nov 1995.

주 작 성 자 : 김 기
논문투고일 : 2003. 9. 20
논문심사일 : 2003. 10. 22
심사판정일 : 2003. 10. 22

저자소개

김 기 (e-mail : bart@seed.handong.edu)
2002 한동대학교 전산전자공학부 학사
2001 (주)투웨이커뮤니케이션 연구원
2002 ~ 현재 한동대학교 전산전자공학부 대학원 석사 과정
관심분야: 분산 시스템

최은미 (e-mail : emchoi@handong.edu)
1988 고려대학교 이과대학 전산과학과 학사
1991 미국 미시간 주립대학 Computer Science 공학 석사
1997 미국 미시간 주립대학 Computer Science 공학 박사
1998 ~ 현재 한동대학교 전산전자공학부 교수
관심분야: 분산 시스템, 클러스터 시스템, 분산 병렬 처리, 분산객체 모델링, 보안 프로토콜