제 19 회 한국정보처리학회 춘계학술발표대회 논문집 제 10 권 제 1 호 (2003. 5)

클러스터 시스템의 효과적인 성능 모니터링과 레포팅

김기, 최은미
한동대학교 전산전자공학부
e-mail : emchoi@handong.edu

Effective Performance Monitoring and Reporting Management for Cluster Systems

Ki Kim, Eunmi Choi
School of Computer Science & Electronic Engineering, Handong Global University

요 약

인터넷 서비스 서버들의 가용성과 확장성, 부하분산의 특성들을 가지는 클러스터 시스템에서 성능 모니터링과 레포팅을 통한 관리 방법이 중요하게 대두되고 있다. 이 논문에서는 클러스터 시스템으로부터 성능 정보를 모으는 방법과 관리에 필요한 기능 및 디자인 설계를 설명하고, 구현한 시스템을 이용하여 실 사이트에서 수집한 정보의 보고서를 보여준다. 이의 결과에 따라서, 대상 클러스터 시스템의 정보와 성능의 최적성을 분석하며, 장기간의 정책을 수립하는 판단 근거를 기술한다.

1. 서론

인터넷 사용의 확장으로 사용자와 서비스 요청이 급격하게 늘어나게 되면서 인터넷 서비스를 제공하는 서버들은 같은 서비스를 제공하는 서버들을 하나의 형태 (SSI: Single System Image)로 묶어 클러스터 시스템을 이루어서 확장성(Scalability), 고가용성(High Availability), 경제성(Cost-Effectiveness), 이상상황으로부터 신뢰성(Fault Tolerance)을 가진 고성능 시스템(High Performance System)으로 서비스를 제공하게 되었다.[1,2]

서비스 수요증가로 인해 클러스터 시스템이 확장됨에 따라 관리자는 원거리에 위치하여 대다수의 서버들을 관리하는 경우가 발생하였다. 또한 이상상황이 발생하는 경우, 다수의 서버 상에서 복잡한 어플리케이션들이 맞물려서 서비스를 제공하므로 클러스터 시스템의 관리자가 어떠한 문제로 인하여 이상상황이 발생된 것인지 원인을 찾기가 어렵다. 성능 모니터링은 서버 관리의 근본적인 부분이며 이상상황 발생의 추세파악, 서버 추가 혹은 자원확장의 판단 근거, 성능 최적화의 판단 근거를 제공한다. 또한 작업 할당의 불균형으로 인한 전체 시스템의 성능 저하 현상을 파악할 수 있으며 장기간의 서버관리 정책을 수립하는 판단 근거를 제공할 수 있다. 이와 같이 클러스터 시스템의 관리와 성능에 대한 평가의 중요성 [2,6]이 대두되고 있다.

본 논문에서는 클러스터 시스템에서의 효과적인 성능 모니터링과 관리 방법을 제시하며, 이를 제공하기 위한 클러스터 시스템 상에서의 구성 아키텍처와 컴포넌트들을 소개한다. 실 시스템에서의 결과 보고서를 통하여 성능 최적화와 서버 관리 정책에 대한 자료를 보이도록 하겠다.

다음 절에서는 모니터링과 관리에 필요한 아키텍처 구성을 설명하며, 3 절에서는 성능 모니터링을 위해 수집해야 하는 정보와 관리에 필요한 기능을 설명한다. 4 절에서는 실 사이트의 모니터링 결과 보고서를 설명한다. 5 절에서는 클러스터 시스템의 성능 분석에 대해서 설명하며, 6 절에서 본 논문의 결론을 내린다.

2. 클러스터 시스템의 모니터링과 관리 아키텍처

2.1 구성 컴포넌트 개요

클러스터 시스템의 모니터링과 관리를 위한 구성 요소는 크게 모니터링, 수집, 저장, 그리고 관리의 4 부분으로 나눈다.

1) **모니터링 Agent 들**: 서버들에 위치하여 상태변화 이벤트와 주어진 주기마다 성능 수치를 얻어내며 정보관리 콘솔로부터 명령을 받아 서버에 적용하는 일을 담당한다.

2) **수집자 (Collector)**: 각 서버의 Agent 로부터 성능수치와 상태변화 이벤트를 모아 Database 에 저장하는 일을 담당한다.

3) **저장 Database**: 각 서버의 정보, 성능수치, 상태변화의 저장소 역할을 담당한다.

4) 보고서와 관리 Console: 각 서버의 Agent 와 연결되어 클러스터 시스템에 서버를 추가, 삭제 명령과 현재 서버의 상태를 보여주며, Database 에 저장된 자료로부터 자동으로 관리에 필요한 보고서를 만드는 일을 담당한다. 생성된 보고서는 XML 와 SVG 형태로 저장하여 만들어진 자료가 다시 쓰일 수 있도록 하였다.

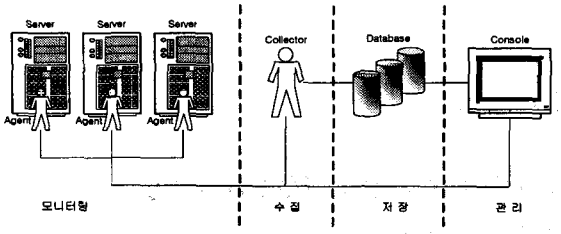

그림 1 클러스터 시스템의 모니터링과 보고서 관리 구성도

2.2 컴포넌트 별 설계

각 구성 요소들의 세부 설계를 통하여 Class 간의 관계성과 기능을 살펴보기로 하겠다.

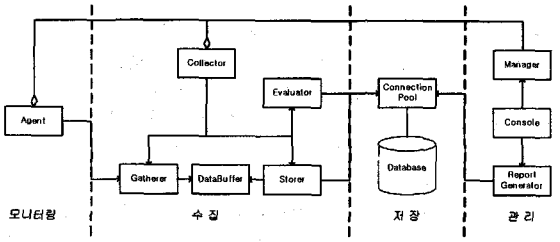

그림 2 클래스 다이어그램

클래스 이름	설명 및 주요 메소드
Agent	서버를 모니터링 하기 위해 서버에 위치하며 상태변화 이벤트와 성능 정보를 수집하여 Collector 클래스에 전달한다 또한 ManagementConsole 로부터 명령을 받아 적용시킨다.
	start(), stop(), pause(), resume(), joinCluster(), leaveCluster() probe(), readPerfData(), sendPerfData(), sendEventData()
Collector	Agent 로부터 전달되는 정보를 모아 저장하기 위해 수집 컴포넌트에 속하는 Class 들을 초기화 시킨다. 또한 Management Console 로부터 명령을 받아 적용시킨다.
	init(), start(), stop(), pause(), resume()
Gatherer	Agent 로부터 직접적으로 접속해 성능 정보와 이벤트를 받는다.
	receivePerfData(), receiveEventData()
DataBuffer	Database 에 일괄저장하기 전에 성능정보들을 잠시 저장한다.
	putData(), getData()
Storer	주기적으로 DataBuffer 에 있는 data 들을 Database 에 저장한다.
	isUpToTime() storePerfData(), storeEventData()
Evaluator	Database 에 저장된 성능정보의 단위시간(1 분, 1 시간, 1 일) 동안의 Avg, Max, Min 값을 계산한다.
	isUpToTime(), EvaluateData()
ConnectionPool	Database 와의 Connection 을 pool 로 만들어 관리한다.
	getConnection(), returnConnection()
ReportGenerator	사용자에게 보고서를 작성하여 보여준다.
	makeReport()
Manager	클러스터의 생성, 제거와 클러스터에 노드를 추가, 삭제 한다.
	createCluster(), removeCluster(), joinNode(), leaveNode()
Console	ReportGenerator 와 Manager 를 초기화 하며 UI 를 제공한다.

표 1 클래스 별 설명과 주요 메소드

3. 성능 모니터링과 관리

클러스터 시스템의 성능을 모니터링을 하고 그 성능의 상태에 따라서 시스템을 보다 효과적으로 관리하기 위하여 다음과 같은 고려 사항들이 있다.

1) 서버 노드의 자원 관리 차원
   - 각 서버 노드들의 자원사용량 측정 (메모리 사용량, CPU 사용량, 네트워크 점유율)
   - 각 서버들의 Scale-up 결정 자료 제공 (시스템에 요구되는 작업 요청량, 서버의 부하량)
2) 서비스 제공 관리 차원
   - 사용자의 request 의 분포시간, 처리량, request 의 workload type 파악
   - 사용자들의 Active Connection 성향
   - 어플리케이션 서비스의 안정성과 가용성 자료 제공
3) 클러스터 시스템의 관리 차원
   - 클러스터 관리 (생성, 제거, 노드 추가/삭제, 서비스 제어: 시작/중지/일시중지, 부하분산 방식 변경)
   - 부하 분산 정도 측정 – 불균형으로 파생되는 성능 저하 및 이상 상황 분석, 부하 분산 스케줄링 알고리즘에 대한 분석
   - 클러스터 시스템의 Scale-out 결정 자료 제공 (서버 추가/삭제 결정)

4. 실 사이트 성능 보고서 예제

웹 검색서비스를 제공하는 E 사의 성능 모니터링 일간 보고서를 통하여, 클러스터 시스템의 정보, 가용성, 부하분산, 자원사용, 서비스 요청의 시간분포 등을 측정하고 보여주는 예제를 소개한다.

4.1 클러스터 시스템의 정보

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 3 클러스터 시스템의 정보

클러스터의 서비스 종류, 생성 일시와 클러스터에 속해있는 노드들의 이름, OS, 프로세서에 대한 정보를 알 수 있다. 위의 예에서는 웹 서비스를 하기 위한 클러스터 시스템에 5 개의 서버 노드가 포함된 것을 나타낸다.

4.2 클러스터 시스템의 가용성

1) 전체 클러스터 시스템의 가용성

Cluster Availability

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

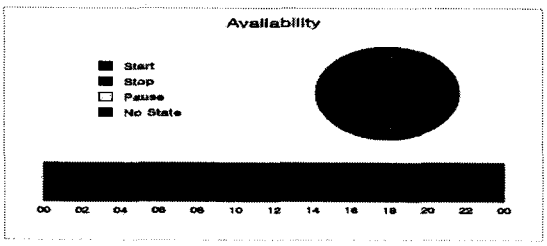

그림 4 클러스터 시스템의 가용성

가용률을 서비스 가능시간 / 총시간 으로 계산하여 나타낸다. 그림 4 는 가용률 100%의 예이다. 클러스터 시스템의 가용성을 Start, Stop, Pause, No State 의 4 가지 상태로 표현하며 파이 그래프에서는 각 상태의 비율을, 막대 히스토그램 그래프에서는 각 상태의 발생, 지속 시간을 볼 수 있다. 위의 예에서는 클러스터 시스템이 24 시간 동안 Start 상태로 지속되었음을 볼 수 있다.

2) 클러스터 시스템에 속한 노드들의 가용성

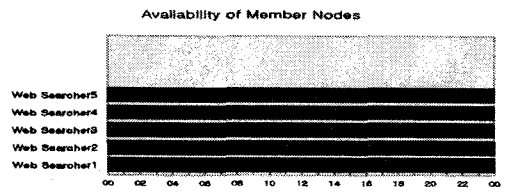

그림 5 클러스터 시스템의 각 노드별 가용성

클러스터 시스템에 속해 있는 각 노드의 가용성을 막대 히스토그램 그래프로 보여준다. 모든 노드가 같은 시간에 Stop 상태라면 서비스를 제공할 수 없으므로 클러스터 시스템의 상태 역시 Stop 일 수 밖에 없다. 그림 5 에서는 모든 노드들이 24 시간 동안 Start 상태로 지속되었음을 볼 수 있다.

4.3 부하분산

Scheduling Methods

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

그림 6 클러스터 시스템의 부하분산 방식

부하 분산에 사용된 스케쥴링 방식을 사용한 기간과 함께 목록으로 보여준다. 그림 6 에서는 Round Robin 방식을 2003-02-07 부터 사용했음을 볼 수 있다.

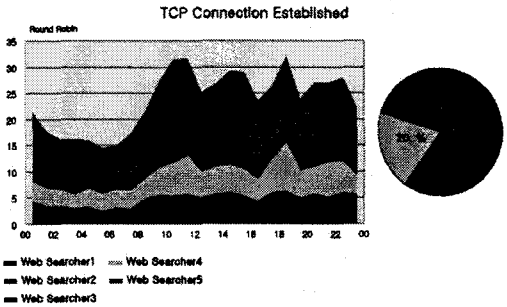

그림 7 클러스터 시스템의 부하분산

모든 TCP 연결을 통한 서비스 요청을 각 노드들이 시간별로 얼만큼 분담했는지 그래프의 면적을 보고 알 수 있다. 그림 7 에서는 각 시간별로 면적이 균등하고 파이 그래프의 면적이 각각 20%대로 일정하므로 부하분산이 잘 이루어지고 있음을 알 수 있다.

4.4 자원 사용

자원의 사용을 나타내기 위해 각 성능 수치의 한시간 동안의 평균값을 라인 그래프 형태로 보여준다. 그림 7 에서 서비스 요청의 증감과 그림 8 - 그림 11 의 자원 사용 그래프들의 증감 형태가 비슷한 형태를 가지고 있다. 따라서 이들 자원과 서비스는 밀접한 관계를 가지고 있으며, 어떤 자원의 부족으로 인한 서비스의 제약이 있는지 확인하고 scale-up 혹은 scale-out 의 판단 기준으로 삼을 수 있다. 그래프에서 나타난 프로세서의 로드가 최대 4%, 사용 가능한 메모리의 양이 최소 140MByte, 네트워크 사용량이 최대 2.6KByte 를 넘지 않으므로 주요한 자원이 서비스를 제공하는데 부족함이 없는 것으로 판단 할 수 있다.[5]

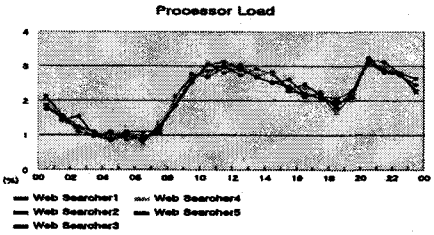

그림 8 자원사용 □ Processor Load

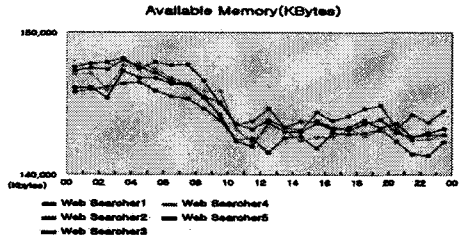

그림 9 자원사용 □ Available Memory

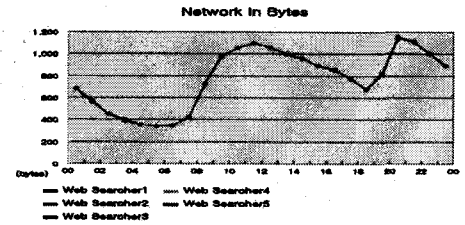

그림 10 자원사용 □ Network In Bytes

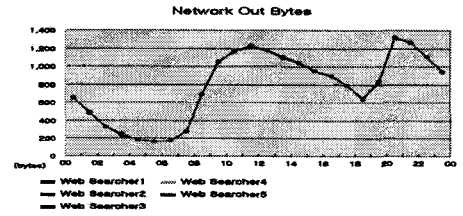

그림 11 자원사용 □ Network Out Bytes

4.5 서비스 요청

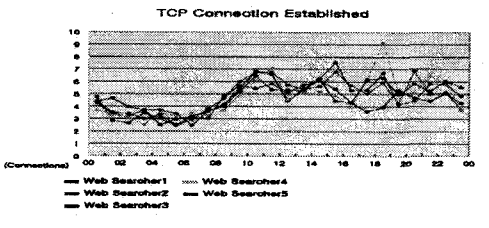

그림 12 서비스 요청 □ TCP Connection Established

E 사의 서비스는 TCP/IP 를 통해 서비스를 제공하므로 TCP 연결의 분포가 곧 서비스 요청의 분포와 동일하다고 볼 수 있다. 그림 12 에서, 06 시부터 서비스 요청이 증가하여 12 시에 정점에 이르며 00 시까지 꾸준한 서비스 요청을 처리하고 있음을 확인 할 수 있다.

5. 클러스터 시스템의 성능 분석

1) 가용성 (Availability)

장기적인 가용성 정책 결정 시 고려 사항은, 전체 클러스터 시스템의 가용률이 높아 서비스가 정지하지 않았더라도 노드들의 가용률의 평균값이 낮다면 클러스터 시스템에 노드를 더 추가하여 클러스터 시스템의 잠재적 서비스 실패 가능성을 낮출 필요성이 있다.[3] 서비스의 가용성 자료와 노드들의 가용성 자료를 통하여 분석이 가능하다.

2) 확장성 (Scalability)

서비스와 밀접한 관계를 가지고 있는 자원의 양이 임계값을 넘는지 확인하고, 현재 자원으로 Request 의 서비스를 충당할 수 있는 지의 여부를 판단하여 서버 성능의 Scale-up 혹은 서버 갯수의 Scale-out 을 결정한다.[4]

6. 결론

클러스터 시스템을 모니터링하여 성능 수치를 수집, 저장하고 보고서를 생성함으로써 장기간 서버 관리 정책 수립의 자료가 될 수 있으며 생성된 보고서는 XML 와 SVG 형태로 저장하여 만들어진 자료가 다시 쓰일 수 있도록 하였다. 관리적인 측면에서는 원격에서 클러스터 시스템의 현재 상태를 쉽게 파악하고 제어할 수 있도록 구현하였다.

향후 연구로는, 보고서에서 자원확장과 같은 관리가 요구되는 것을 판단하여 명시해 줄 수 있는 기능과 생성된 보고서를 PDF 등 다른 문서형태로 변환하는 기능을 포함시키고자 한다.

참고 문헌

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

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

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

[6] 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