메타컴퓨팅환경의 어플리케이션관리서비스

김 윤 희
숙명여자대학교 정보과학부 컴퓨터과학전공
yulan@cs.sookmyung.ac.kr

Evaluation of Application Management Services in Metacomputing Environment

Yoonhee Kim
Dept. of Computer Science, Sookmyung Women' s University

요 약

이종 네트웍 환경아래 서로 다른 서비스 요구사항을 가진 대형 분산 어플리케이션을 효율적으로 관리하는 것은 매우 어려운 일이다. 또한 기존 네트웍 관리 시스템들은 서비스관점보다는 시스템중심으로 관리하여 어플리케이션에 맞는 종합적인 관리를 수행하는 것은 복잡하고 이해하기 어렵다. 이 논문에서 어플리케이션 수행을 보다 적극적으로 관리하고 어플리케이션과 네트웍의 요구사항을 분석, 실행하는 Proactive Application Management System (PAMS)을 제안하고 구현하였다. 각 관리 서비스는 모바일 에이전트로 이루어져 변화에 따라 능동적인 관리가 가능토록 하였다. 실험결과는 에이전트 중심으로 하는 PAMS가 성능과 실패관리에 능동적으로 대처하여 긍정적인 효과가 있음을 보여준다.

1. 서 론

초고속 네트웍과 눈부신 컴퓨팅 기술의 발전은 네트웍 자체와 그 위에서 수행되는 네트웍 어플리케이션의 크기와 복잡도를 증가 시키고 있다. 그런 네트웍과 분산 어플리케이션들을 관리하는 것은 더욱 복잡해지고 때로는 효과적인 관리가 거의 불가능하다. 불행히도 기존의 네트웍 관리 시스템들은 네트웍과 플랫폼을 중심으로 관리정보를 모으고 이에 따라 반 자동적인 관리를 해왔다. 아직 이종 네트웍 환경에서 대형 어플리케이션을 총괄적, 지능적, 효율적, 능동적 관리하는 데에 필요한 연구는 미흡하다.

대형 네트웍 관리에 대한 중요성은 날로 중요해짐에 따라 네트웍관리시스템에 대한 연구는 관리 복잡성을 줄이고 변화에 따라 능동적으로 대처 가능한 방향으로 진행되고 있다. 하나의 중앙 집중형 관리자보다 여러 관리자들이 통신 프로토콜을 정하여 상호 협동하는 구조로 진행되고있는 Management by Delegation (MbD)[3], Code Mobility [4]있다. 관리체-에이전트의 구조 대신 Common Object Request Broker Architecture (CORBA) 의 동등계층 관계를 이용한 연구도 Telecommunications Information Networking Architecture (TINA) 프레임웍을 중심으로 이루어지고 있다[1]. 웹을 기반으로 하는 네트웍 관리도 최근 들어 활발히 진행되고 있다 (JMAPI, WEBEM) [2].

그러나, 이종 네트웍상위의 어플리케이션을 관리하는 연구는 아직 미비한 편이나 현재까지 IETF가 공통 어플리케이션 관리 정보를 모아 저장하는 Application Management MIB [6] 와 MIB for Application [5]이 제안했고, DMTF가 WBEM에서 사용되는 프로세스 정보 정의의 Common Information Model (CIM)을 제안하였다 [2]. 그러나 아직도 프로그램 적용형 어플리케이션 관리 방법에 대한 연구는 초기 단계이다.

이 논문에서 어플리케이션을 중심으로 그 요구사항에 맞게 관리정보를 수집하고 제어를 수행하는 Proactive Application Management System (PAMS)을 제안했다. PAMS는 분산 어플리케이션이 불안정한 이종네트웍에서 수행되었을 때 그 성능의 최적화를 지향하고 수행실패 발생 시 능동적으로 어플리케이션을 관리하도록 관리 서비스를 제공한다. PAMS는 모바일 에이전트를 이용하여 분산 어플리케이션의 Quality of Service (QoS) 요구사항을 준수토록 구현이 용이하다. 이 논문에서 성능과 실패관리를 위해 세가지 능동 기술을 적용하고 그 결과를 평가한다. 즉 성능저하 및 실행 실패 발생시 능동적 또는 수동적 중복 수행이나 이동 수행(Migration)을 통하여 그 어플리케이션이 성공적으로 수행을 마치도록 제어하였다. 그 결과, PAMS는 에이전트 중심의 이 기술들을 통해 성능향상과 낮은 오버헤드를 가짐을 보여주었다.

이 논문의 구성은 다음과 같다. 2장에서 PAMS 구조의 간단한 소개를 하고, 3장에서 PAMS통해 제공되는 관리 서비스들을 소개하고 성능과 실패관리 과정을 통해 얻은 결과를 논하고 마지막 4장에서 결론을 맺는다.

2. Proactive Application Management System (PAMS) 의 구조

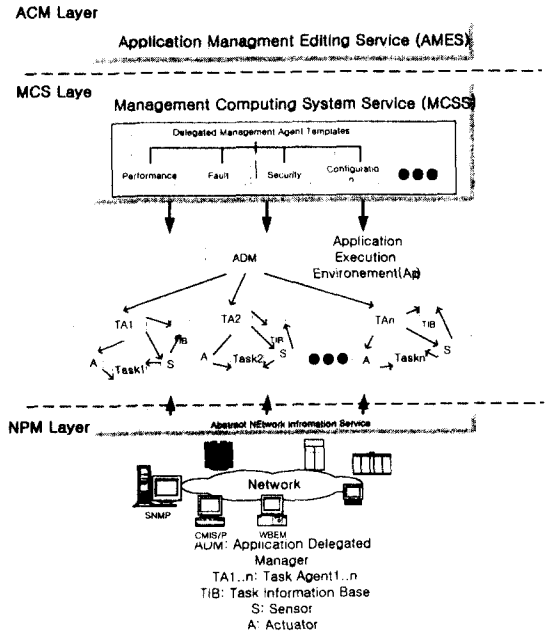

그림 1 PAMS의 실시간 구조

PAMS는 크게 그림 1에서 보는 것처럼 크게 두개 층 - Application Centric Management (ACM) 층과 Management Computing System (MCS) 층으로 나뉜다. ACM 층에서는 어플리케이션 개발자로 하여금 어플리케이션을 개발하고 그에 맞는 어플리케이션 QoS 요구사항 또는 성능 또는 실패 등의 관리 요구사항을 명시하도록 하는 개발툴을 제공한다. MCS 층에서는 다양한 네트웍 어플리케이션을 위한 관리 서비스들이 에이전트로 존재하여 요구사항에 맞게 적절한 관리 에이전트를 선택하여 수행되도록 제어한다. 일단 어플리케이션의 요구사항이 ACM툴에 의해 정해지면 MCS는 적절한 어플리케이션 실행환경 (Application Execution Environment: AEE)을 생성하여 그 어플리케이션의 실행동안 어플리케이션 요구사항을 충족하도록 능동적으로 자원을 할당 제어한다. 이때 MCS는 하나의 어플리케이션 대리관리자 (Application Delegated Manager: ADM)를 만들어 관리 서비스들을 관리토록 한다. 분산 어플리케이션의 여러개의 태스크들에 따라 태스크 에이전트 (Task Agent: TA)를 만들어 태스크를 감시토록 한다. TA들은 적절한 센서를 통해 태스크 수행을 감시하고 실행 시 태스크가 어플리케이션 요구사항을 지키지 못하게 되었을 경우 엑추에이터를 통해 수행을 일시정지하고 태스크 실행 상태를 저장하거나 다른 자원이 지원될 경우 그 자원으로 태스크를 이동시켜 실행을 계속하도록 한다.

TA의 주요 관리 활동으로는 크게 세가지 기능으로 나눌 수 있다. 첫째는 변화 감지 (Change_Detection)으로 태스크가 허용된 범위 안에서의 동작하고 있는 지의 여부를 감시하고, 둘째는 분석 평가 (Analysis_Verification) 단계로 한계 변화에 대한 정확한 원인분석을 수행하며 마지막으로 세번째 단계인 적응 계획 (Adaptation_Plan)을 통하여 적절한 적응방안을 수행한다. 다음 장에서는 위의 3가지 단계를 통하여 성능과 실패제어를 능동적으로 수행함을 보여주고 그 결과를 분석한다.

3. 능동적인 성능및 실패제어 관리

3.1 성능관리

분산 어플리케이션의 성능관리는 감시해야 하는 관리 대상의 다양성으로 인해 쉽지 않다. 성능관리는 크게 감시와 제어로 볼 수 있는데 감시는 자원 및 어플리케이션의 성능 활동을 쫓아 가는 기능으로 볼 수 있다. 제어 기능은 성능관리로 하여금 성능을 향상시킬 수 있도록 조절을 가하는 일이다. 이때 자원의 성능상태 정도를 나타내는 성능 메트릭스를 추출하는 일이 중요하고 [7], 그 메트릭스에 따라 한계치 조절 방법[8], 측정주기[9]등 결정하는 하는 것이 성능관리의 주요 이슈들이다.

본 실험에서는 컴퓨팅 중심 어플리케이션을 선택하여 어플리케이션의 수행시간을 성능 메트릭스로 결정하였다. 또한 성능 제어를 위해서는 중복 및 이동 제어를 통해 능동적인 관리가 가능토록 했다. 중복의 경우 능동 중복 (Active Redundancy)와 수동 중복 (Passive Redundancy)으로 나눈다. 각 기능은 그림 2와 같의 에이전트 템플릿 (Agent Template)으로 구현되었다.

능동 중복 기능은 그림 2 (a)처럼 서로다른 자원위에 두세트의 실행환경을 생성, 수행하여 최적의 결과를 취하는 방법이며 수동 중복 기능 (그림 2(b))은 서로다른 자원위에 두세트의 실행환경을 생성하되 한세트를 대기상태 (stand-by)로 유지하여 이상 발생시 즉각적인 전환이 가능토록 한다. 그림 2 (c)는 이상 발생시에 대체 환경을 생성하고 이동을 수행하는 방법이다. 이 방법의 오버헤드는 상대적으로 크므로 이 방법은 어플리케이션 단위사이즈가 커서 그 오버헤드가 전체 수행시간에 상대적으로 적을 경우에만 사용하는 것이 바람직하다.

PAMS 성능 관리서비스에 사용된 어플리케이션은 두가지로, 평균 30초가량 수행시간의 작은 사이즈와 450초 가량의 수행시간을 가진 큰 사이즈의 컴퓨팅 어플리케이션들이다. 어플리케이션 수행시에 CPU 사용량을 99%로 하였을 때 수행시간의 차이를 조사하여 성능관리가 발생하였을 때와 하지않았을 때를 비교하였다. 이동 기능을 사용하였을 때 작은 어플리케이션의 경우 평균 25%의 성능 향상이 있었고 큰 어플리케이션의 경우, 75%였다. 유사하게 능동 중복 기능으로는 각각 31%, 174%의 향상과 수동 중복기능으로는 22%와 114% 향상이 있었다.

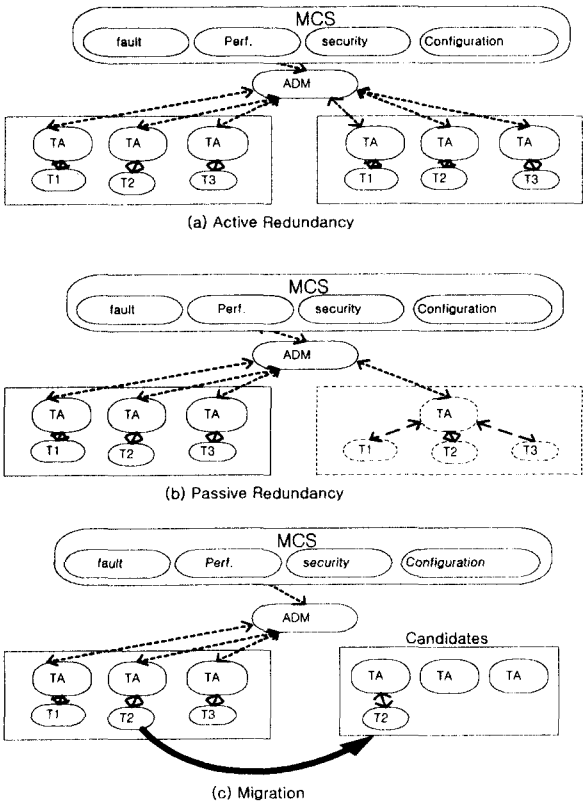

그림 2 관리서비스를 위한 제어 기능들

3.2 실패 관리

어플리케이션 실패관리의 주요 목적은 자원 또는 어플리케이션의 잠재적 혹은 실제 실패에서 효과적으로 회복하는 것이다. 자원 또는 시간의 중복사용이 가장 널리 알려진 방법이다. SCOP[10]는 자원의 중복사용의 비용을 최소화 하는 능동적 중복 구조를 제안 했다.. 본 실험에서 에이전트 중심으로 중복기능과 이동을 통해 자동 실패 회복을 시도했으며 그를 위한 오버헤드를 측정했다.

능동 중복 기능을 통한 실패제어는 실패한 세트의 결과 대신 성공한 실행을 취하는 것이고, 수동 중복의 경우 오류 발생한 태스크에 대해 대기상태의 태스크에게 연계하여 수행하도록 하였다. 어플리케이션 사이즈를 60초, 6000초로 하였을 경우, 능동 중복 기능의 경우, 각각 0.10%, 0.02%의 오버헤드가 측정되었고, 수동 중복의 경우, 각각 0.42%, 0.02%이하가 측정되었다.

실험결과를 통하여 PAMS가 대형 분산어플리케이션을 관리하는 데 효율적임을 확인하였다. 나아가 모바일 에이전트와 에이전트 템플릿의 사용은 가용한 자원을 효율적으로 실시간 사용하기에 적합하다.

4. 결론

이 논문에서 에이전트 중심의 어플리케이션 관리 시스템인 PAMS를 설계하고 그 프로토타입을 구현하였다. PAMS는 어플리케이션 요구사항에 맞추어 총괄적으로 어플리케이션과 자원을 관리하도록 하였다. 서로 크기의 어플리케이션들로 실험한 결과 PAMS의 관리 서비스들은 성능향상과 실패관리에 효과적임을 보여주었다. 다양한 어플리케이션을 지원할 수 있도록 관리 서비스 템플릿을 개발하여 PAMS의 기능 확장이 필요하며 어플리케이션과 관리 서비스 템플릿간의 매핑 기능에 대한 연구를 진행 중 이다.

참고문헌

[1] J. Pavon and J. Tomas, CORBA for Network and Service Management in the TINA Framework, IEEE Communication Magazine, March 1998.

[2] J. P. Thompson, Web-Based Enterprise Management Architecture, IEEE Communication Magazine, March 1998

[3] G. Goldszmidt and Y. Yemini, Distributed Management by Delegation, in 15th international Conference on Distbuted Computing, June 1995.

[4] M. Baldi, S. Gai and G. Picco, Exploiting Code Mobility in Decentalized and Flexible Network Management, In First International Workshop, MA97, Berlin, Germany, April 97.

[5] C. Krupczak and J. Saperia, Definition of System-Level Managed Objects for Applications, RFC2287, Feb 1998.

[6] C. Kalbfleisch, C. Krupczak, R. Presuhn, and J. Saperia, Application Management MIB, Internet-draft, Nov. 98.

[7] Michael Katchabaw, Stephen Howard, Andrew Marshall, Michael Bauer, "Evaluating the Cost of Management: A Distributed Application Management Testbed," Proceeding of the 1996 CAS conference(CASCON'96) Toronto, Canada, Nov.12-14 pp29-41.

[8] Marina Thottan, Chuanyi Ji, "Adaptive Thresholding for Proactive Network Problem Detection, Procedding of the 1998 international workshop for Systems Management, Newport, April, 1998.

[9] P Dini, G. Bochmann,T. Koch, B. Kramer, "Agent based Management of Distributed Systems with Variable Polling Frequency Policies,"

[10] J. Xu, A. Bondavalli, F. D. Giandomenico, "Dynamic Adjustment of Dependability and Efficiency in Fault-Tolerant Software", in "Predictably Dependable Computing Systems", B. Randell, J. C. Laprie, H. Kopetz and B. Littlewood Ed., Springer-Verlag, 1995, pp.155-172.