웹 서비스 기반 그리드 환경에서 자율적인 서비스 재구성 관리 기법

김은경° 김윤희
숙명여자대학교 정보과학부 컴퓨터과학과
{kimek°, yulan}@sookmyung.ac.kr

A Web-Service Based Autonomic Service Reconfiguration on Grid Environments

Eun-kyung Kim°, Yoonhee Kim
Dept. of Computer Science, Sookmyung Women's University

요 약

웹 서비스 기반 그리드 미들웨어 기술은 동적인 자원의 활용성, 지능적인 자원 분배는 효과적으로 지원하지 못하고 있다. 그리드 환경에서의 가용성을 증대시키기 위하여 자원 부족으로 인한 작업 내 동일 서비스들에 대한 자원 재할당이 불가능 할 경우, 서비스 자체의 오류로 인하여 동일 서비스를 계속적으로 제공하는 것이 불가능한 경우, 네트워크 성능 저하와 같은 다른 환경적 요인에 의하여 서비스 성능 및 결과의 질이 떨어질 경우에 있어서 그 문제를 해결하는데 한계가 있다. 이 논문에서는 웹 서비스 기반 그리드 환경에 따라 적응하는 서비스 미들웨어에서 자율적으로 오류 관리를 지원하는 방법을 제시하고 프로토타입 Wapee(Web-Service based Application Execution Environment)를 통해 실제 환경에서 적용가능성을 확인한다.

1. 서 론

그리드 컴퓨팅은 인터넷으로 구성된 가상 조직(Virtual Organization: VO)사이에서 컴퓨팅 자원 및 데이터, 어플리케이션 등의 컴퓨팅 리소스의 모든 요소를 동적으로 공유하여 어느 곳에서라도 접근이 가능하게 해준다[1]. 최근의 그리드 서비스는 공개 표준 기술인 기존의 웹 서비스 인프라에 사용할 수 있게 보완된 OGSI(Open Grid Service Infrastructure)를 보완하고 WSRF(Web Service Resource Framework)를 정의하여 WS-Resource 개념이 도입되었다. WSRF는 기존 웹 서비스 기술과 그리드 기술과의 융합을 위해 새롭게 제안된 표준으로 웹 서비스를 통해 상태의 정보를 가지고 있는 자원들을 모델링하고 접근하는 프레임워크를 정의하고 있다. 현재 공개되어있는 미들웨어인 Globus ToolKit 4.0[2]은 기존의 웹 서비스 표준안을 기반으로 효율적인 자원 관리를 위해 확장되어 공개 표준 기술의 사용으로 이질적인 자원들을 일관성 있게 사용하고 관리할 수 있다. 공개 표준 기술의 사용이라는 관점에서는 유틸리티 컴퓨팅보다 한 발 앞서있으나 서비스의 계속성과 유연한 이동성 및 오류에 대한 복구지원, 상호 보안, 지능적인 자원 분배, 응용을 위한 QoS(Quality of Services) 지원 등의 서비스 고가용성은 효과적으로 지원하지 못하고 있다. 그러므로 자율 컴퓨팅이 그리드 컴퓨팅의 핵심 요소가 될 수 있다.

1) 자율 컴퓨팅은 생명체의 자기 보호, 자기 치유, 자기 구성 등의 특성을 가지는 자율 신경계를 모델링한 기술로 컴퓨터에 위임한 융통성 있는 정책에 의해 판단되고 행동하는 컴퓨팅을 말한다. <표 1>은 이를 가능하게 해주는 자율 컴퓨팅의 자기 관리 기술을 설명한다[3]. 다양하고 동적인 유비쿼터스 서비스(U-Services) 들이 연동되는 어플리케이션 실행 환경에서 요구되는 서비스들의 종류가 다양화되고 그 기능이 복잡해짐에 따라, 하나의 서비스로 사용자의 다양한 요구사항을 만족시켜 주기 힘들게 되었다. 또한 같은 종류의 서비스라도 사용자가 필요로 하는 컨텍스트(context)와 다양한 환경적 요인들의 변화에 따라 그 기능이 달라져야 한다. 따라서 최근에는 사용자의 컨텍스트에 따라 적합한 서비스를 자율적으로 선택하여 이용하기 위한 방법들이 활발히 연구되고 있다. 끊임없는 U-서비스 이용환경을 제공하기 위하여 서비스 이동성 지원과 동적인 자원의 할당, 서비스 이용 시 발생할 수 있는 오류 복구 등을 통한 자율적인 고가용성을 지원하는 미들웨어가 필수적으로 필요하다.

<표 1> 자율 컴퓨팅 기술

자율 컴퓨팅 기술	기능 설명
자가 구성 기술 (Self-Configuration)	소프트웨어가 없음을 감지했을 때, 소프트웨어를 자동 설치하는 등 새로운 구성요소를 사용할 수 있도록 하거나 기존의 구성 요소를 제거하는 등의 유연성 확보를 돕는다.
자가 치유 기술 (Self-Healing)	시스템의 기능 장애를 탐지하고 정책에 기반을 두어 실패한 요소를 다시 시작하는 등의 보정 작업을 한다.
자가 최적화 기술 (Self-Optimization)	QoS를 제공하기 위하여 시스템의 용량이 증가한 것을 발견하면 현재 업무 부하를 조정하는 등의 전반적인 효율을 개선한다.
자가 보호 기술 (Self-Protecting)	잘못된 요청을 파악하고 외부 침입 감지 시 리소스를 오프라인으로 끄집어내는 등의 안전을 지속적으로 강화한다.

본 논문에서는 다양한 서비스들이 연동되는 어플리케이션 실행 환경에서 서비스 및 어플리케이션의 오류 복구를 통한 웹 서비스 기반 미들웨어의 고가용성 지원방법을 제시한다. 컨텍스트 및 사용자 요구사항의 변화, 그리고 예상하지 못했던 서비스 오류 발생 시 런타임 서비스 관리(Runtime Service Manager) 시스템을 통해 자율적으로 런타임 작업 실행환경을 재구성하도록 하여 미들웨어의 가용성을 증대시키고 안정적으로 서비스의 계속성과 데이터 및 자료의 일관성을 보장한다. 서비스의 계속성 및 일관성을 유지하기 위하여 서비스를 저장,

* 본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 육성·지원사업의 연구결과로 수행되었음

유지, 관리하는 기법과 사용자의 상황을 유지하여 어플리케이션 실행 시에 서비스를 실시간 동적으로 재구성하기 위한 기술을 제시한다.

본 논문의 2장에서는 필요한 관련 연구를 소개하고, 3장에서는 본 논문에서 제안하는 자율성을 지원하는 시스템인 Wapee(Web-Service based Application Execution Environment)에 대한 설계 및 구조를 소개하며 4장에서는 프로토타입을 통한 성능평가를, 마지막으로 5장에서 결론 및 향후 연구를 정리한다.

2. 관련 연구

기존 웹 서비스 오류 표준 기술로는 SOAP 메시지 처리 중 발생하는 오류에 대한 자세한 내용을 기술할 수 있도록 여러 요소를 지원하는 SOAP(Simple Object Access Protocol)에서의 Fault[4]부분, 문제 확인과 오류 관리를 지원하는 웹 서비스 오류 메시지를 명세화 한 WSRF의 WS-BaseFault와 분산 컴퓨팅 기술로 CORBA(Common Object Request Broker Architecture)표준에 포함된 오류 감내 기법으로 감지, 통보, 분석 메커니즘을 제공하는 FT(Fault Tolerance)-CORBA, ping 메커니즘과 referral component를 통해 네트워크 등 접속 오류를 감지하고 컴포넌트의 상태를 파악하여 자율적으로 새로운 인스턴스를 사용자에게 제공함으로서 오류를 극복하는 DCOM(Distributed Component Object Model)의 Fault Tolerance[5] 등이 있다.

그러나 WS-BaseFault와 SOAP Fault는 웹 서비스 오류 메시지에 대한 정의와 오류 메커니즘을 통해 오류를 처리하나 시스템 오류나 어플리케이션 상의 오류 복구 환경은 존재하지 않는다. FT-CORBA와 DCOM과 같은 오류 복구 기술을 적용한 관련연구로는 GEMS(Grid Enactor and Management Service)[6], Armor(Adaptive Reconfigurable Mobile Objects of Reliability)[7], MEAD (Middleware for Embedded Adaptive Dependability)[8]등이 있다. GEMS는 지속적인 메시지 폴링으로 결함을 감지하나 본 논문에서는 Armor, MEAD처럼 발생된 결함에 대한 상황을 실시간으로 분석하여 자율적으로 처리가 가능하도록 하였다. 또한 고가용성을 위해 위에 제시된 관련 연구처럼 실패된 작업을 처음부터 다시 시작하지 않고 오류가 발생된 부분부터 재시작 하도록 하여 서비스의 연속성을 높였고 오류에 대처할 동일 서비스가 없는 경우, 이를 대체할 수 있는 유사 서비스를 결정하여 제공하였다.

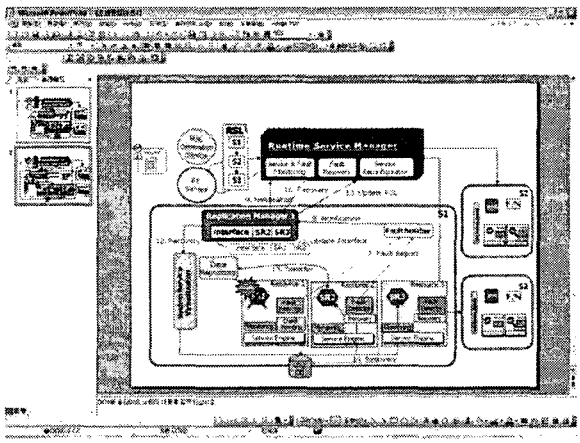
3. 자율성 지원 시스템 설계 및 구조

자율적인 런타임 서비스 관리 시스템은 다양하고 동적인 U-서비스를 지원하기 위한 자율 미들웨어로 지리적으로 분산된 그리드 상의 분산된 자원을 이용하여 어플리케이션 실행에 필요한 요구사항을 분석하고 실행에 필요한 작업명세서 및 파일들을 자동적으로 생성하고 오류 관리를 해주는 인터페이스를 제공한다. 또한 사용자에게 유연한 환경을 제공하기 위한 조건으로 오류 관리를 통한 서비스 실행의 일관성을 보장한다.

자율적인 오류 관리를 위하여 다음과 같은 구조를 제공한다.

첫째, 폴링(polling)기법을 사용하여 주기적으로 실행 중인 작업과 VO상의 자원들의 상태를 파악하는 모니터링 서비스를 제공한다. 다양한 응용 어플리케이션을 지원함에 있어 실행중인 작업 및 계산 자원에 대한 모니터링은 매우 유용하다. 작업의 상태를 파악함으로 오류를 감지하고 작업을 제어함으로써 계산 자원을 보다 효율적으로 이용할 수 있다. 이로써 작업의 실행 중 시스템 및 자원, 서비스 등의 결함으로 오류가 발생되면 이를 신속히 감지 할 수 있으며 유용한 자원에 대해 신속하게 맵핑하여 작업 시간의 효용성을 높인다.

둘째, 모니터링 서비스를 통하여 발생한 오류를 감지하고 적절한 상황 판단을 통하여 자율적으로 서비스를 재구성하도록 제공한다. 서비스 재구성 환경을 위하여 리플리케이션(Replication)기법을 사용해 복구함으로써 지속적인 서비스 이용 환경을 제공한다. 오류에 대한 능동적인 서비스를 제공하기 위해 서비스 변화를 수용할 수 있는 서비스 업데이트와 작업에 필요한 데이터 및 파일의 신속하고 정확한 마이그레이션(Migration)이 동반된다. 오류 감지기(Fault Detector)에 의해 오류가 감지되면 오류 통보자(Fault Notifier)에게 통보된다. 감지된 오류 정보는 리플리케이션 매니저(Replication Manager)에게 전달되고 오류 상황에 대비하여 미리 생성해 둔 인터페이스를 바탕으로 리플리카에 대한 정보와 감지된 결함 정보를 런타임 서비스 매니저에게 통보한다(<그림 1> 참고). 런타임 서비스 매니저는 보고된 오류의 속성을 바탕으로 미리 정의되어 있는 후보자원에 해당 데이터 및 파일을 이동하고 서비스를 재구성하고 자율적으로 업데이트한다. 이때 사용자에 의해 입력받았던 요구사항 등은 기존의 정보를 그대로 유지하며 데이터 저장소를 통해 이전에 진행 중이던 작업 정보를 제공받아 오류가 발생된 시점부터 재실행되고 사용자는 자율적으로 재구성된 내용을 보고 받을 수 있다. 이를 위해 서비스 변화를 수용할 수 있는 서비스 재구성 서비스와 작업에 필요한 데이터 및 파일의 신속하고 정확한 마이그레이션 및 리플리카 관리가 필요하다.



<그림 1> 런타임 서비스 재구성

런타임 서비스 구성 단계에서 처리할 수 있는 오류로는 인증 관련 오류, 파일 전송 중 발생되는 오류, 작업 실행 중 발생되는 일부 오류 등을 들 수 있다. 오류 정보는 자율 관리 매니저에게 전달되고 오류에 대한 분석을 시도한다. 오류 속성의 분석을 통해 결정된 원인을 파악한 후 오류 상황 극복을 위한 여러 가지 전략을 검토한 후 상황에 맞게 극복 플랜을 실행한다. 오류 상황에 대비하여 미리 생성해 둔 리플리카에 대한 정보를 전달받으면 오류 정보를 바탕으로 해당 오류를 분류하고 그에 대한 복구방법을 선택 한 후 작업 실행 명세서를 자동 변경하여 런타임 서비스를 진행한다. 또한 리플리카에 의한 복구가 불가능한 경우 자동으로 서비스 프로파일의 정보를 이용한 서비스 재구성을 시도하여 서비스의 계속성을 보장하도록 한다.

4. 실험 결과 및 분석

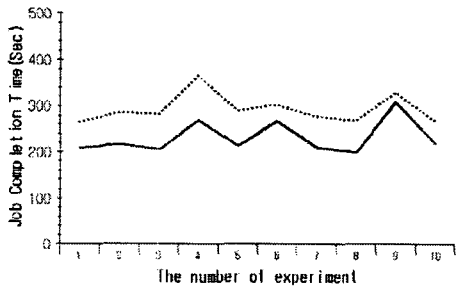
이 장에서는 앞에서 설계한 프로토타입인 Wapee를 통해 오류 복구 매커니즘을 실험하고 그 결과로 시스템의 성능을 평가한다. 본 실험은 웹 기반에서의 대량의 데이터를 분석하는 어플리케이션을 오류 복구 메커니즘을 제공하는 Wapee 시스템에서 실행하여 오류 복구에 있어 성능을 평가한다. 사용되는 실험 시나리오 어플리케이션은 대량의 데이터를 동시에 접근하는 웹 기반 어플리케이션으로 웹상에 존재하는 다수의 신문 사설들을 모아 사용자가 선택한 단어가 들어 있는 특정 사설을 검색하고, 선택한 단어들의 빈도수를 랭킹으로 표현하여 나타낸다.

실험은 GT4환경의 리눅스 노드 5개에서 실시하였다. 각 노드는 512M이상의 메모리와 3.0Ghz이상의 CPU를 제공한다. 이때 실험 시간은 시스템의 성능이나 실험 시의 네트워크 상황을 감안한다. <그림 2>는 작업에 오류가 발생하지 않고 작업을 완료했을 경우의 작업 완료 시간과 본 연구에서 개발한 Wapee 시스템의 모니터링 서비스를 통해 작업을 수행한 완료 시간을 도표화한 것이다. Wapee는 실행되고 있는 작업을 모니터하여 오류를 감지하고 발생된 오류에 대해 적절한 복구 메커니즘을 적용함으로써 오류를 극복한다. 따라서 기존 응용 실행 시간보다 약 26.3%의 오버헤드가 존재하였다. 하지만 사용자에게 작업의 현 상태를 파악할 수 있게 하며 오류가 발생할 경우 이를 시스템 상황 등을 고려하여 능동적으로 대처할 수 있어 작업 시간에 대한 효율성이 증대된다. <그림 3>은 작업 실행 중 오류가 발생하였을 경우 이를 DAGMan과 Wapee의 오류 관리를 이용하여 복구 하였을 경우의 작업 완료 시간을 도표화한 것이다. DAGMan은 자원 오류에 대해 새로운 자원을 할당하여 오류를 복구하는 매커니즘을 제공하나 특정 자원을 사용해야 하는 작업에 있어 해당 자원이 존재하지 않는다면 Rescue 파일을 자동 생성하여 사용자가 재 실행해야한다. 이에 반해 Wapee의 오류 관리는 오류의 종류를 추상화 시켜 이를 복구할 수 있는 전략을 제공하여 자율적으로 대체 서비스를 검색하여 오류 상황을 자율적으로 극복한다. 결과적으로 기존의 응용프로그램을 이용한 실험에서는 모니터링을 제공하는 Wapee 시스템보다 오류가 존재하지 않을 경우에는 작업 시간의 단축을 보였으나 오류가 발생하였을 경우 오류를 즉시 파악하지 못하고 수동으로 재 시작해야 하는 등의 문제점이 존재한다. Wapee를 사용하면 적은 오버헤드로 서비스 재구성 및 재실행이 가능하며 이때 오류가 발생한 작업에 대해서만 작업을 재실행해줌으로써 시스템에 부담을 줄일 수 있다. 또한 자동적 오류 복구를 지원하여 서비스의 계속성을 보장할 수 있다.

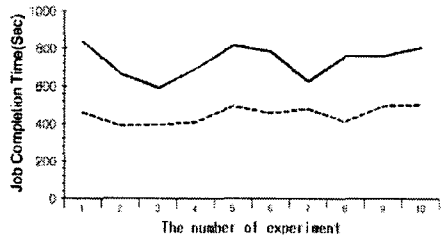
5. 결 론

다양한 U-서비스 지원을 위해서는 끊임없는 서비스를 보장하고 서비스 이동성 지원과 적응형 동적 자원 지원, 서비스 오류 복구를 지원하는 자율 미들웨어가 필수적으로 필요하다. 이에 따라 본 논문에서는 개방형 표준을 고려하고 고가용성을 지원하여 U-서비스 환경에서 고가용성을 지원하는 다양한 방법 중 오류 복구를 통한 자율적인 서비스 재구성 기법을 제시하고 프로토타입 구현을 통해 검증했다. 웹 서비스 기반의 미들웨어는 끊임없는 서비스가 제공되는 환경을 보장하기 위하여 서비스 이동성 지원과 적응형 동적 자원 지원, 실행 환경 재구성기법을 통하여 고가용성을 지원해야한다. 끊임없는 서비스 지원을 위한 고가용성 서비스를 제공하기 위하여 오류 탐지, 오류 분석, 오류 복구 관리자를 통하여 작업 실행 시 실행환경, 어플리케이션 및 서비스 재구성을 통하여 웹 서비스 기반의 시스템에 자율성을 부여한다.

차후에는 다양한 오류 복구 방법을 추가하여 고가용성을 지원하기 위한 폭넓은 방안을 제시해, 다양한 서비스를 사용하는데 있어 사용자에게 유연하고 투명한 실행 환경을 제공해야 할 것이다.



< 그림 2 > 모니터링 제공에 따른 오버헤드 비교



< 그림 3 > 오류 관리를 이용한 작업 수행시간 비교

6. 참고문헌

[1] Globus Project, http://www.globus.org/
[2] I. Foster, "A Globus Toolkit Primer" http://www-unix.globus.org/toolkit/docs/4.0/key/GT4_Primer_0.6.pdf
[3] http://www.ibm.com/autonomic
[4] OASIS http://www.xml.org/xml/resources_focus_beginnerguide.shtml
[5] DCOM Fault http://msdn.microsoft.com/
[6] Satish Tadepalli, Calvin Ribbens, Srinid Varadarahan "GEMS: A Job Management System for Fault Tolerant Grid Computing", High Peformance Computing Symposium, 2004
[7] Zbigniew Kalbarczyk,Ravishankar K Iyer,Long Wang," Application Fault Tolerance with Armor Middleware" Internet Computing, March/April 2005 (Vol 9, No 2) pp 28-37
[8] P Narasimhan, C F Reverte, S Ratanotayanon and G S Hartman,"Middleware for Embedded Adaptive Dependability" IEEE Workshop on Large Scale Real-Time and Embedded Systems, Austin, TX, December 2002