DOI: 10.3745/KIPSTA.2008.15-A.3.151

# 그리드 환경의 적응형 오류 극복 관리 시스템 설계 및 구현

김 은 경[†] · 김 지 영[††] · 김 윤 희[†††]


## 요 약

서비스 이동과 자원 상태 변화 등 실행 환경 변화가 빈번히 발생하는 그리드 컴퓨팅 환경은 다양한 응용 프로그램 작업 환경을 지원하고 사용자에게 끊임없는 작업 환경을 보장하기 위하여 고가용성을 지원하는 미들웨어가 필수적으로 필요하다. 기존의 분산 환경 미들웨어 역시 고가용성 지원 서비스가 일부 연구자에 의해 진행되고 있으나 공개표준은 아니며 다양한 그리드 서비스에 대한 고려가 없다. 본 논문에서는 환경에 따라 적응하는 서비스 미들웨어 런타임 서비스 관리 시스템을 통해 자율적으로 작업 환경을 재구성하도록 하여 미들웨어의 가용성을 증대시키고 안정적으로 서비스의 계속성과 데이터 및 자료의 일관성을 보장하는 방법을 제시하고 프로토타입 Wapee(Web-Service based Application Execution Environment)를 통해 실제 환경에서 적용 가능성을 확인한다.


**키워드 : 자율성 미들웨어, 온톨로지 기반, 오류 감내, 그리드**

# Design and Implementation of Adaptive Fault-Tolerant Management System over Grid

Eun Kyung Kim[†] · Jeu Young Kim[††] · Yoonhee Kim[†††]


## ABSTRACT

A middleware in grid computing environment is required to support seamless on-demand services over diverse resource situations in order to meet various user requirements [1]. Since grid computing applications need situation-aware middleware services in this environment. In this paper, we propose a semantic middleware architecture to support dynamic software component reconfiguration based fault and service ontology to provide fault-tolerance in a grid computing environment. Our middleware includes autonomic management to detect faults, analyze causes of them, and plan semantically meaningful strategies to recover from the failure using pre-defined fault and service ontology trees. We implemented a referenced prototype, Web-service based Application Execution Environment(Wapee), as a proof-of-concept, and showed the efficiency in runtime recovery.


**Keyword : Autonomic Middleware, Ontology, Fault-tolerance, Grid**

## 1. 서 론

그리드 컴퓨팅은 인터넷으로 구성된 가상 조직(Virtual Organization: VO) 사이에서 컴퓨팅 자원 및 데이터, 응용 프로그램 등의 컴퓨팅 리소스의 모든 요소를 동적으로 공유하여 어느 곳에서라도 접근이 가능하게 해준다. 다양하고 동적인 자원들이 산재되어 있는 그리드 환경에서는 응용 프로그램 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 요구사항 및 컨텍스트(Context), 다양한 환경적 요인들의 변화에 따라 그 기능이 달라져야 한다. 이렇게 컨텍스트 기반의 다중 서비스가 결합되는 그리드 컴퓨팅 환경에서는 어떠한 서비스의 장애 시에도 응용 프로그램이 안정적으로 계속 수행되도록 서비스 이동성 지원과 적응형 동적 자원 지원, 서비스 오류 복구를 지원하는 런타임 환경에 적응력 있는 미들웨어가 필수적으로 필요하다. 기존의 가용성 증대 방법론들은 자원 부족으로 인한 작업 내 동일 서비스들에 대한 자원 재 할당


† 정 회 원 : SK 커뮤니케이션즈 연구원
†† 준 회 원 : 숙명여자대학교 컴퓨터과학 석사과정
††† 종신회원 : 숙명여자대학교 컴퓨터과학과 부교수(교신저자)


논문접수 : 2007년 2월 27일
수 정 일 : 1차 2007년 10월 16일, 2차 2008년 1월 10일, 3차 2008년 3월 3일
4차 2008년 3월 12일, 5차 2008년 3월 18일
심사완료 : 2008년 3월 18일

이 불가능할 경우, 서비스 자체의 오류로 인하여 동일 서비스를 계속적으로 제공하는 것이 불가능할 경우, 네트워크 성능 저하와 같은 다른 환경적 요인에 의하여 서비스 성능 및 결과의 질이 떨어질 경우에 있어서 그 문제를 해결하는데 한계가 있다.

본 논문에서는 개방형 표준을 고려하고 그리드 환경하의 적응형 서비스 오류 극복 관리를 해주는 서비스 미들웨어의 기술을 연구하였다. 미들웨어를 통하여 실행 중에 오류가 탐지되면 동적으로 실행환경을 재구성하도록 하여 오류 발생 처리를 투명하게 처리하는 적응형 런타임 서비스 관리를 개발하고 기존에 개발되어있는 온톨로지를 이용한 시맨틱스 오류 관리를 이용하여 오류 발생 시 시스템에서 자율적으로 런타임 작업 실행환경을 재구성하도록 하여 미들웨어의 가용성을 증대시키고 안정적으로 서비스의 계속성과 데이터 및 자료 관리의 일관성을 보장하기 위한 기술을 제시한다.

## 2. 관련연구

본 장에서는 오류 관리를 지원하는 관련 연구들을 소개한다.

GEMS(Grid Enactor and Management Service)[2]는 그리드 작업을 수행하고 모니터링, 작업 실패를 감지하고 재실행을 지원하는 시스템이다. 로컬리소스 관리자 서버와 모니터 클라이언트라는 두 개의 주요를 가지고 모니터 클라이언트는 현재 자신의 시스템에서 작업이 수행되고 있는 상태를 전달한다. 서버는 서비스를 마쳤거나 어떤 오류로 인해 중지 되었다면 해당 내용이 담긴 메시지를 받게 된다. 시스템의 결함으로 인해 주기적인 메시지 전달이 이루어 지지 않는다면 서버는 노드에 대한 오류를 감지하고 새로 실행되는 작업에 대해 새로운 리소스를 할당하여 재실행하게 한다.

Condor[4]는 컴퓨팅 작업이 대부분인 workload에 대해 특화된 관리시스템이다. 작업 queuing 메커니즘, 스케줄링 정책, 우선순위 설계(Priority scheme), 자원 모니터링(Resource monitoring), 자원 관리 등의 기능을 제공한다. 오류 관리는 체크포인트 기법을 통해 제공된다.

DAGMan[4]은 Directed Acyclic Graph Manager의 약자로 Condor를 위한 메타 스케줄러이다. 이는 Condor 스케줄러보다 상위 레벨에서 작업 사이의 의존성을 관리한다. 의존성 있는 작업의 실행 중 일부 작업에 대해 오류가 발생하였을 경우 작업 제출 파일(Rescue DAG)을 재 정의함으로써 오류를 극복한다. Rescue DAG는 DAGMan의 입력 파일로 사용되고, 이미 수행이 성공적으로 끝난 작업에 대해서 완료되었음을 표시하여 재실행되지 않도록 한다.

## 3. 적응형 서비스 오류 극복 관리 시스템 설계 및 구조

고가용성을 지원해주는 서비스 관리 미들웨어 구조는 적응형 런타임 서비스 관리, 오류 관리, 자원 관리로 크게 세 가지 기능으로 구성된다. 이 논문을 통하여 설계되고 구현된 것은 적응형 런타임 서비스 관리 부분과 자원 관리 부분이며 오류 관리 부분의 내용은 참고문헌 [7]를 참고한다.

### 3.1 적응형 런타임 서비스 관리

적응형 런타임 서비스 관리는 3가지 계층으로 분류가 가능하며 각 계층의 기능은 아래와 같다.

- 서비스 컨테이너 계층: 응용 프로그램 서비스 등록 및 해지 기능, 구동된 서비스들을 관리해주는 Life-cycle 기능, 서비스간의 커뮤니케이션 기능, 서비스의 상태와 필요한 데이터를 이동시켜주는 마이그레이션 기능, 재구성 서비스 등
- 시스템 서비스 계층: 서비스 모니터링, 디렉터리 서비스, 워크플로우 서비스, 오류 극복 서비스 등
- 서비스 풀 계층: 상태변화를 감지하는 모니터링, 적응형 서비스 및 서비스 중심 자원 위에서 동작제어를 위한 서비스 등

응용 프로그램이 특정한 상황에 따라 전달되는 데이터 및 서비스를 사용하기 위해서는 미들웨어에서 상황을 인식하고 이를 전달하는 메커니즘이 필요하다.

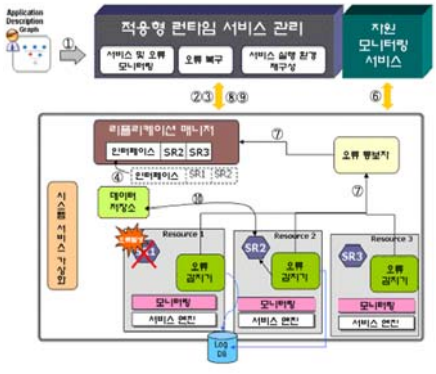
본 논문의 런타임 서비스 관리는 다음과 같은 구조를 제공한다. 첫째, 폴링(polling)기법을 사용하여 주기적으로 실행 작업과 VO상의 자원들의 상태를 파악하는 모니터링 서비스를 제공한다. 이로써 작업의 실행 중 시스템 및 자원, 서비스 등의 결함으로 오류가 발생되면 이를 신속히 감지할 수 있으며 유용한 자원에 대해 신속하게 맵핑하여 작업 시간의 효용성을 높인다.

둘째, 모니터링 서비스를 통하여 발생한 오류를 감지하고 적절한 상황 판단을 통하여 자율적으로 서비스를 재구성하도록 제공한다. 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.

### 3.2 런타임 서비스 관리 시스템을 통한 서비스 재구성

(그림 1)은 실행시간 수준에서 오류가 발생했을 경우 서비스를 재구성하는 과정을 보여준다.

① 사용자의 작업 요청을 받아 생성된 ADG(Application Deployment Graph)를 정의하고 기존에 정의되어 있는 오류 이외에 새롭게 발견될 수 있는 오류를 예측하여 새로운 속성으로 추가한다.




(그림 1) 런타임 서비스 재구성

②, ③, ④ 런타임 서비스 관리는 서비스 인스턴스를 생성하며 구동된 서비스를 감시하고 상황 변화를 인식한다. 서비스 인스턴스 생성시에 리플리케이션 매니저는 서비스 리플리카의 목록이 정의된 리플리케이션 인터페이스를 정의한다[6]. 이 인터페이스를 통해 서비스 리플리카를 생성한다.

⑤ 작업 실행에 필요한 입력파일 및 실행 결과에 따른 출력파일을 자원으로 이동한다.

⑥, ⑦ 모니터링 서비스를 통해 실행중인 작업의 상태를 파악한다. 오류 감지기에 의해 오류가 감지되면 오류 통보자에게 통보된다. 감지된 오류 정보는 리플리케이션 매니저에게 전달된다.

⑧ 오류 상황에 대비하여 미리 생성해 둔 인터페이스를 바탕으로 리플리카에 대한 정보와 감지된 결함 정보를 런타임 서비스 관리에 통보한다. 런타임 서비스 관리는 감지된 오류의 상태를 파악하고 미리 정의되어 있는 오류 속성을 비교 판단한다.

⑨ 오류 속성과 오류 보고를 바탕으로 해당 오류를 분류하고 그에 의한 복구방법을 선택한다. 이때 변경되는 부분을 체크하여 새롭게 런타임 서비스를 재구성한다. 재구성하기 위해 ADG가 변경되면 그 내용을 자동으로 업데이트하여 진행한다(사용자에게 ADG를 생성하기 위한 또 다른 요구사항은 받지 않고, 자동으로 재구성한다).

⑩ 작업의 결과는 데이터 저장소에 저장된다.

⑪ 변경된 내용을 바탕으로 작업을 재실행하여 사용자에게 결과를 제공한다. 이때 사용자는 자동으로 재구성 된 내용을 보고 받을 수 있다.

런타임 서비스 구성 단계에서 처리할 수 있는 오류로는 인증 관련 오류, 파일 전송 중 발생되는 오류, 작업 실행 중 발생되는 일부 오류 등이 있다.

## 4. 성능 비교

### 4.1 실험 및 결과 분석

이 장에서는 앞에서 설계한 시스템의 프로토타입인 Wapee를 통해 오류 복구 메커니즘을 실험하고 그 결과로 시스템의 성능을 평가한다.

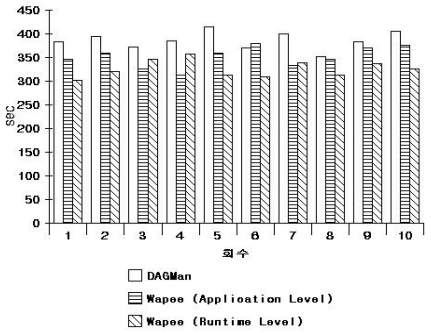
실험은 GT4환경의 리눅스 노드 5개에서 실시하였다. 각 노드는 512M이상의 메모리와 3.0Ghz이상의 CPU를 제공한다. 이때 실험 시간은 시스템의 성능이나 실험 시의 네트워크 상황을 감안한다.

실험에 사용된 응용 시나리오는 대량의 데이터를 동시에 접근하는 웹 기반 응용 프로그램이 웹 상에 존재하는 다수의 신문 사설들을 모아오는 서비스와 모아온 데이터를 기반으로 사용자가 상세 검색을 통해 특정 사설을 검색하는 서비스, 입력한 단어들의 빈도수를 랭킹으로 나타내는 서비스를 순서에 따라 워크플로우 형태로 구성하여 실행하도록 하였다. 이때 상세 검색 서비스 이용 시 메모리 자원 부족 오류를 발생하도록 유도하였다.

본 실험에서는 가정된 오류 사항인 메모리 자원 부족 오류를 극복하는 두 가지 메커니즘을 제시하였다. 첫 번째는 실행시간 수준의 오류 복구 메커니즘으로 동일한 서비스가 실행될 수 있는 가용자원이 존재할 경우이고, 두 번째는 동일 서비스 제공이 불가능하여 온톨로지를 이용하여 유사서비스로 대체하는 응용 프로그램 수준의 오류 복구 메커니즘이다.

DAGMan은 자원 오류에 대해 새로운 자원을 할당하여 오류를 복구하는 메커니즘을 제공한다. 본 시나리오와 같이 특정 자원(서비스)을 사용해야 하는 작업에 있어 해당 자원이 존재하지 않는다면 Rescue 파일을 수동 생성하여 사용자가 재실행해야 한다. 이에 반해 Wapee의 오류 관리는 오류의 종류를 추상화 시켜 이를 복구할 수 있는 전략을 제공하여 대체 서비스를 검색하고 오류 상황을 자동으로 극복한다. DAGMan은 오류가 존재하지 않을 경우 모니터링을 제공하는 Wapee 시스템보다 작업 시간이 짧았으나 오류가 발생하였을 경우에는 오류를 즉시 파악하지 못하고 수동으로 재 시작해야 하는 등의 문제점이 존재했다.

(그림 2)는 DAGMan과 Wapee를 사용하여 오류가 발생했을 경우 오류를 복구하여 작업이 완료되기까지의 전체 실행시간을 도식화한 그래프이다. DAGMan을 이용하면 평균 386.37초가 소요되며, Wapee를 이용하면 실행시간 수준 오류 복구 기능만 사용하는 경우 평균 362.29초, 응용 프로그램 수준 오류 복구 기능까지 사용하는 경우 350.90초가 소요되어 빠른 시간에 서비스 재구성 및 재실행이 가능하다. Wapee를 사용하면 오류가 발생한 작업에 대해서만 작업을 재실행해줌으로써 시스템의 부담을 줄일 수 있고, 적은 오버헤드로 자동적인 서비스 재구성 및 재실행이 가능하여 서비스의 계속성을 보장할 수 있다.




(그림 2) 전체 실행 시간 비교(오류복구포함)

### 4.2 타 오류 관리 지원 미들웨어와의 비교

<표 1>은 기존에 개발되어있는 오류 관리 기법을 제공하는 미들웨어와 본 논문의 프로토타입인 Wapee를 비교한 것이다.

본 논문에서는 효율적인 자원 할당을 위해 메타스케줄러로 Condor-G를 사용하였고, 워크플로우 기반의 작업 환경을 제공하기 위해 DAGMan을 이용하였다. 이러한 환경을 기반으로 오류가 발생되면 오류 관리 서비스 및 작업, 자원 모니터링 서비스를 통하여 자동으로 작업을 재실행함으로써 기존 연구들보다 사용자에게 유연하고 효율적인 런타임 작업 환경을 제공한다.

〈표 1〉 오류 관리 지원 미들웨어 비교

| | GEMS | SPHINX | Condor /DAGMan | Wapee |
|---|---|---|---|---|
| 워크플로우 지원 | x | o | o | o |
| 작업스케줄링 지원 | o | o | o | o |
| 오류감지 모니터링 | o | o | o | o |
| 자동적인 작업 재실행 기능 | x | x | x | o |
| 그리드 환경 지원 | o | o | o | o |

## 5. 결 론

본 논문에서는 다양한 서비스 지원을 위해서 응용 프로그램 작업 실행 환경을 보장하고 서비스 이동성 지원과 적응형 동적 자원 지원, 서비스 오류 복구를 지원하는 그리드 환경의 서비스 미들웨어를 구축하고 고가용성을 지원하는 방법을 제시하였다. 서비스의 고가용성을 증대시키기 위한 다양한 방법 중 오류 복구를 통한 적응형 서비스 재구성 기법을 통해 사용자에게 끊임없는 작업 환경을 보장한다. 신속한 오류 복구를 위하여 서비스 및 인스턴스 복제를 관리하고 복제된 서비스 및 인스턴스를 바탕으로 실행 환경을 재구성하여 오류 발생에 대하여 유연한 시스템을 제공한다.

현재 서비스 표준안을 기반으로 효율적인 자원 관리를 위해 확장하여 이질적인 자원들을 일관성 있게 사용하고 관리할 수 있으나 서비스의 계속성과 유연한 이동성 및 오류에 대한 복구지원, 상호 보안, 지능적인 자원 분배, 응용을 위한 QoS지원 등은 효과적으로 지원하지 못하고 있다. 그러므로 융통성 있는 정책에 의해 판단되고 행동하는 자율 컴퓨팅을 그리드 컴퓨팅에 접목하여 자율적으로 서비스를 재구성하는 방법을 제시하여야 할 것이다.

## 참 고 문 헌


[1] M. Weiser, “The computer for the 21st Century,” Scientific American, vol.265, no.3, pp.94-104, September, 1991.

[2] Satish Tadepalli, Calvin Ribbens, Srinid Varadarahan, “GEMS: A Job Management System for Fault Tolerant Grid Computing”, High Peformance Computing Symposium, 2004.

[3] Jang-uk In, Paul Avery, Richard Cavanaugh, Laukik Chitnis, Mandar Kulkarni, Sanjay Ranka, “SPHINX: A Fault-Tolerant System for Scheduling in Dynamic Grid Environments”, 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS’05), p.12b, 2005.

[4] Condor/DAGMan http://www.cs.wisc.edu/condor/ dagman

[5] Zbigniew Kalbarczyk, Ravishankar K Iyer, Long Wang, “Application Fault Tolerance with Armor Middleware”, Internet Computing, Vol.9, No.2, pp.28-37, March/April, 2005.

[6] 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.

[7] Eduardo Ostertag, James Hendler, Ruben Prieto Diaz, Christine Braun., “Computing similarity in a reuse library system: an AI-based approach”, ACM Trans. Softw. Eng. Methodol., vol.1, no.3, pp.205-228, 1992.

[8] R. Prieto-Diaz, P. Freeman, “Classifying Software for Reuse”, IEEE Software, 4(1), pp.6-16, 1987.

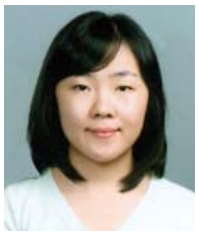
[9] Hwayoun Lee, Ho-Jin Choi, In-Young Ko., “A Semantically-Based Software Component Selection Mechanism for Intelligent Service Robots”, Proceedings of 4th Mexican International Conference on Artificial Intelligence (MICAI2005), Monterrey, Mexico, November, 2005.

[10] Yoonhee Kim, Eun-kyung Kim, Beom-Jun Jeon, In-Young Ko, and Sung-Yong Park, “Wapee: A Fault-Tolerant Semantic Middleware in Ubiquitous Computing Environments”, EUC Workshops, IFIP International Federation for Information Processing, LNCS 4097, Seoul, August, 2006.



### 김 은 경

e-mail : kekeeo@skcomms.co.kr
1999년~2003년 숙명여자대학교 컴퓨터과학(학사)
2003년~2007년 숙명여자대학교 컴퓨터과학(석사)
2007년~현 재 SK 커뮤니케이션즈 연구원
관심분야 : 그리드 컴퓨팅, 자율 컴퓨팅, 온톨로지 및 지식 기반 시스템, 텍스트 마이닝, 자연어 처리


### 김 지 영

e-mail : wldud5@sm.ac.kr
2001년~2005년 숙명여자대학교 컴퓨터과학(학사)
2006년~현 재 숙명여자대학교 컴퓨터과학 석사과정
관심분야 : 그리드 컴퓨팅, 온톨로지, 지능형 시스템



### 김 윤 희

e-mail : yulan@sm.ac.kr
1987년~1991년 숙명여자대학교 전산학과(학사)
1994년~1996년 Syracuse University 전산학과(석사)
1996년~2000년 Syracuse University 전산학과(박사)
1991년~1994년 한국전자통신연구소 연구원
2000년~2001년 Rochester Institute of Technology 컴퓨터공학과 조교수
2001년~2004년 숙명여자대학교 컴퓨터과학과 조교수
2004년~현 재 숙명여자대학교 컴퓨터과학과 부교수
관심분야 : 그리드 컴퓨팅 환경(PSE), 워크플로우 제어, 분산 어플리케이션/서비스 관리