그리드 환경에서의 사이트 자율성 보장을 위한 접근 제어 시스템에 관한 연구

김 법 균[†] · 정 성 종[††] · 안 동 언[†††] · 장 행 진[††††] · 박 형 우[†††††]

요 약

지리적으로 분산된 이 기종의 유휴 자원들을 서로 연결하여 가상의 고성능 컴퓨팅 자원으로 사용하는 그리드에서 자원에 대한 접근 제어 시스템의 구축은 필수적이다. 본 논문에서는 그리드 환경 구축 시 전 세계적으로 가장 많이 사용되는 Globus Toolkit을 기반으로 하는 그리드 접근 제어 시스템을 설계 및 구현한다. 특히, 각 자원을 제공하는 사이트의 자율성을 보장하기 위해 각종 환경 설정 파일들을 이용하고 이를 이용한 부가 서비스 개발이 용이 하도록 다양하고 자세한 정보를 만들어 내도록 설계 및 구현하였다.[1]

A Study on Access Control System for Site Autonomy in Grid Environment

Beob-Kyun Kim[†] · Seung-Jong Chung[††] · Dong-Un An[†††] · Haeng-Jin Jang[††††] · Hyung-Woo Park[†††††]

ABSTRACT

Grid makes a virtual high-performance computing resource by connecting geographically distributed heterogeneous resources. Building access control system is an important factor in the grid environment. In this paper, we design and implement a grid access control system based on Globus Toolkit, which is one of the popular grid middleware. Especially, to guarantee the site autonomy for resource provider, we use several environment configuration files. Moreover, we design and implement PGAM to produce more detail and diverse information to ease the development of value added services.

키워드 : 그리드(Grid), 글로버스(Globus), 접근 제어(Access Control)

1. 서 론

인터넷이 보편화되고 컴퓨터 및 네트워크 성능이 향상됨에 따라 분산 자원 기반의 고성능 어플리케이션들은 더 큰 컴퓨팅 파워를 요구하고 있다. 그리드는 지리적으로 분산된 고성능, 대용량의 자원들과 첨단 장비들을 원격에서 동시적으로 사용하여 단일 시스템처럼 사용하는 환경이다[1][2][9][10][11]. 이러한 그리드 환경에서 어플리케이션을 수행하기 위해서는 각 자원을 이용하기 위한 권한을 획득하는 메커니즘이 필수적으로 구축되어야 할 것이다. 각 자원의 소유자의 입장에서는 기존의 어카운트 발급 및 권한 부여에 사용되던 방식이 그리드 환경에 그대로 적용되기를 바라고 자원을 사용하고자 하는 사용자 입장에서는 원격의 고성능 자원을 마치 자신의 로컬 머신처럼 사용하고자 할 것이므로 양쪽 모두 다 충족시킬 수 있는 별도의 접근 제어 시스템이 반드시 구현되어야 한다.

그리드 접근 제어 시스템은 그리드의 특성상 각 자원 제공자가 기존에 사용하던 자원 관리 정책을 그대로 반영할 수 있는 기능을 제공해야 한다[1][2]. 즉, 각 사이트의 자율성을 보장해 주어야 하며, 이를 보장하기 위한 세련된 메커니즘을 제공해야 한다. 이를 통해, 각 자원 소유자의 그리드 환경에 대한 자원 제공 의욕을 고취시킬 수 있다. 또한, 그리드 접근 시스템 자체가 그리드 환경 내에서 서비스를 이용할 때 가장 먼저 이용하게 되는 서비스 중에 하나이므로 다른 서비스에서 이용할 수 있도록 사용자의 신상 정보나 시간, 로컬 계정에 관한 정보 등의 정보를 최대한 상세하게 생산해 줄 필요가 있다[3][4][5][6][7][8][10].

그리드는 여러 도메인에 걸쳐 분산되어 있는 자원을 공유, 교환, 검색, 통합할 수 있도록 한다. 현재 진행 또는 구축된 대부분의 그리드 프로젝트[12][13][14][15]들에서는 상당히 제한적인 범위 내에서의 접근 제어만을 허용하고 있으

† 준 회 원 : 전북대학교 컴퓨터공학과 박사과정
†† 정 회 원 : 전북대학교 컴퓨터공학과 교수
††† 종신회원 : 전북대학교 컴퓨터공학과 부교수
†††† 정 회 원 : 한국과학기술정보연구원 그리드연구실 선임연구원
††††† 정 회 원 : 한국과학기술정보연구원 그리드연구실 실장
논문접수 : 2004년 11월 17일, 심사완료 : 2005년 2월 14일

며, 각 시스템에 특화된 형태로 설계 운용되고 있다. 또한, 다른 그리드 관련 연구들에서는 PSE (Problem Solving Environment)환경을 구축하는 경우가 많아서 특정 문제 해결을 위해 최적화된 형태로 구축되는 경우가 많다. 따라서 현재 구축된 그리드 프로젝트들에서의 접근 제어 시스템들은 본 논문에서 구현한 시스템과는 직접적인 비교가 힘들다.

본 논문에서 구현한 시스템은 각 자원의 계정을 관리한다는 측면에서 크게 보면 어카운팅 관련 연구의 일부분으로 볼 수도 있으며, 이와 관련된 시스템들로 GridX(Grid Exchange)[4], DGAS(DataGrid Accounting System)[18], GridBank [19] 등이 있다. 그러나 이들 연구들에서는 어카운팅 정보의 추출, 어카운팅 정보의 보관과 서비스에 대해서만 언급할 뿐이며, 그리드 환경에 적합한 접근 제어 기능은 각 시스템에 적합한 형태로 특화시켜 사용하고 있다. 그리고 현재 드레프트 수준으로 GSAX (Grid Service Accounting Extensions)[5]가 나와 있는데 이는 동적으로 그리드 서비스가 생성 및 서비스 될 때 어카운팅 정보의 처리 방법을 모색한 것이다. 또한, 그리드 유저가 연결된 로컬 계정의 활용도가 낮을 경우 관리자의 로컬 계정 관리 부담이 급격하게 증가한다는 점에 착안하여, 그리드 유저들을 위한 어카운트 집합을 미리 생성해 놓고 사용 빈도가 높은 그리드 유저의 경우 지속적으로 하나의 계정을 이용할 수 있도록 하고 자원 사용에 대한 추적과 어카운팅이 용이하도록 하는 계정 관리 메커니즘이 제안되었다[8].

VUS (Virtual User System)[17]은 폴란드 국가 그리드 환경을 구축하면서 설계되었는데, 각 센터의 모든 그리드 계정을 소수의 가상 유저 서버 (Virtual Users' Server)를 통해 일률적으로 관리하는 방식이다. 소수의 가상 유저 서버는 Kerberos 시스템과 유사한 방식의 계정 관리를 통해, 그리드 환경에서 활동하는 모든 그리드 유저의 모든 활동을 감시하고 데이터베이스화하는 역할을 한다. 결국 각 사이트의 그리드 참여 및 탈퇴가 자유롭지 못하고 각 사용자도 그리드 환경을 이용하기 위해서는 각 사이트에 대한 접근 권한을 사이트에게 의뢰하는 것이 아니라 소수의 가상 유저 서버에게 허가를 받는 형식이다.

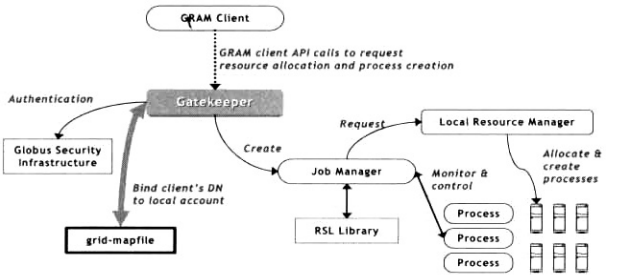
본 논문에서는 특정 시스템에 국한되지 않고 일반적인 그리드 환경에 적용할 수 있고, 각 사이트의 자율성을 보장하며, 어카운팅과 같은 부가 서비스에도 이용할 수 있도록 PGAM (Policy based Grid Account Manager)를 설계하였다. 특히, 그리드 환경 구축 시 가장 많이 사용되는 Globus Toolkit [10]을 기반으로 설계하였으며, 위에서 언급한 그리드 접근 제어 시스템의 요구 사항을 반영하도록 설계 및 구현하였다. 논문의 구성은 2장에서는 Globus Toolkit과 Globus Toolkit에서의 접근 제어 방식에 대해 살펴보고, 3장에서는 본 논문에서 구현한 그리드 접근 제어 시스템의 구조와 기능 및 동작을 기술하고 4장에서는 구현 결과를 설명하며 5장에서 결론을 맺는다.

2. Globus Toolkit

2.1 Globus Toolkit

Globus Toolkit [10]은 그리드 환경 구축 시 가장 많이 사용되고 있는 미들웨어 중의 하나이다. Globus Toolkit이 가장 많이 사용되는 이유는 Globus Toolkit이 분리될 수 없는 단일 시스템이 아니라, 그리드에서 필요로 하는 다양한 서비스들을 독립적인 요소로써 제안하고 있기 때문이다. 또한, 기존의 각 시스템 및 네트워크의 관리 정책이나 운영 도구들을 무시하지 않고 각 요소들과 협력하여 그리드를 구축해 나간다는 점이다. Globus Toolkit에서 제공되는 핵심 서비스는 크게 Resource Management, Information Service, Data Management, 그리고 Grid Security 등으로 나눌 수 있다.

Globus Toolkit에서 Resource Management를 담당하는 부분을 GRAM (Grid Resource Allocation Management)라 부르며 가장 중심이 되는 요소로서 원격지의 자원들을 사용할 수 있게 하고, 분산 자원들을 동시에 사용하게 하며, 자원들의 관리상의 상이함을 처리한다. 특히, GRAM의 주요 구성 요소 중의 하나인 Gatekeeper는 클라이언트의 접근을 가장 먼저 처리하는 부분으로 클라이언트의 프락시(proxy)가 grid-mapfile에 등록되어 있는 그리드 사용자의 것인지를 구분하고 그에 상응하는 처리를 해준다[20]. 다음 그림은 이 과정을 설명해주고 있다.



(그림 1) Gatekeeper와 grid-mapfile

2.2 Globus Toolkit에서의 접근 제어

위 그림과 같이, Globus Toolkit에서는 외부 사용자가 로컬 자원을 이용하기 위해서 각 자원의 실제 로컬 계정을 얻는 대신 grid-mapfile이라는 파일을 이용한다. 그리드 사용자의 DN(Distinguished Name)을 로컬 자원의 계정그리드 사용자는 로컬 자원의 계정을 얻기 위해, 사용자의 DN (Distinguished Name)과 로컬 계정을 함께 기입한다. 다음 그림은 실제 가능한 grid-mapfile의 내용이다. "/O=Grid/OU=hi.ac.kr/CN=hdg"라는 DN을 가진 사용자는 로컬 자원에서 gw1이라는 계정으로 작업을 수행하게 되며 수행 가능한 권한은 로컬 계정 gw1의 권한으로 제한된다.

```
"/O=Grid/OU=hi.ac.kr/CN=hdg" gw1
"/O=Grid/OU=hi.ac.kr/CN=how" gw2
"/O=Grid/OU=bye.com/CN=say" gw3
```

(그림 2) grid-mapfile의 예

기본적으로, 다수의 DN이 하나의 로컬 계정에 대해 결합 가능하며, 각 DN은 신뢰있는 CA(Certificate Authority)로부터 발급 받은 인증서(Certificates)로 인증된다. (그림 1)의 경우 "/O=Grid/OU=hi.ac.kr/CN=hdg"라는 DN의 그리드 사용자는 로컬 머신에서 gw1이라는 사용자 계정을 통해 서비스를 이용하게 된다. 다수의 DN이 하나의 로컬 계정에 대해 결합 가능하므로, 다수의 그리드 사용자가 하나의 로컬 계정을 공유하게 된다. 이로 인해, 특정 프로세스가 발생했을 때 이 프로세스의 실제 주인이 어느 그리드 사용자인지를 구분하기 위해서는 시스템 커널을 통한 복잡한 메커니즘이 필요하거나 처리 불가능한 경우도 발생한다. 또한, 동일한 계정을 다수의 사용자가 공유함으로써 각 사용자의 데이터 보안과 같은 부분에서 문제를 야기할 수도 있다. 그리고 각 사용자의 DN과 로컬 계정의 바인딩 과정을 담당하는 부분이 없으므로 시스템 관리자의 수작업에 의해 이루어지고 있다.

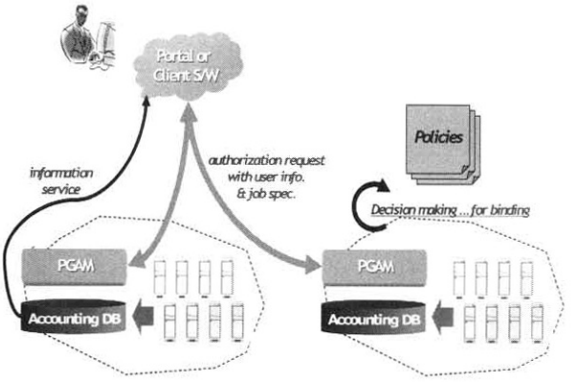
3. 그리드 접근 제어 시스템

3.1 그리드 접근 제어 시스템을 이용한 그리드 응용

본 논문에서 구현한 그리드 접근 제어 시스템인 PGAM은 Globus Toolkit을 기반으로 구현되었으며, 이를 이용하여 그리드 어플리케이션을 수행하고자 할 때는 그리드 사용자는 다음과 같은 과정을 거치게 된다.

1. 클라이언트 시스템에 로그인한다.
   : Single-Sign-On을 위한 프락시(Proxy)를 띄운다. GPDK와 같은 웹 포탈의 경우 MyProxy와 같은 프락시 서비스를 받을 수도 있다. 이하 각 단계에서 사용자 인증시 사용된다.
2. MDS (Metacomputing Directory Service)를 통해 그리드 환경에서 이용할 수 있는 자원들의 목록을 얻는다.
3. 자신의 어플리케이션을 실행하기 위해 필요한 자원을 선정한다.
   : 어플리케이션을 위한 스케줄링을 통해 적절한 자원 후보를 선정한다.
4. 그리드 접근 제어 시스템을 통해 각 자원에 대한 사용 권한을 획득한다.
   :사용자의 신상 정보, 필요한 자원의 양(용량, 시간 포함)을 제공하여 각 자원으로부터 사용 권한을 획득한다. 일부 자원에서 사용 권한 부여를 거부할 경우 다른 후보 자원들에게 사용 권한을 요청함으로써 어플리케이션을 수행하기 위해 필요한 자원을 확보한다.
5. RSL (Resource Specification Language)과 같은 스크립트를 이용하여 그리드 어플리케이션을 실행한다.

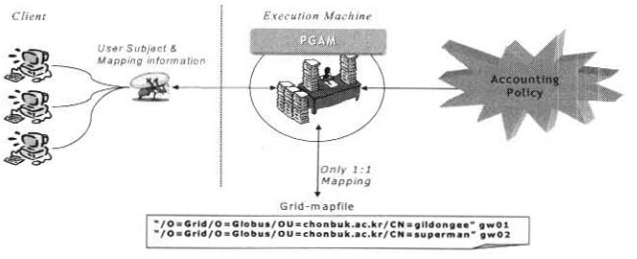
아래 그림은 어카운팅 서비스까지 고려했을 경우 전체 작업 흐름을 보여주고 있다. 그림에서 PGAM은 사용자가 입력한 정보(신상 정보, 필요한 자원의 양 등)를 바탕으로 접근 권한을 부여할 것인지를 판단한다. 각 사이트마다 접근 권한을 부여하기 위한 정책이 수립되어 있으며, 이 정책에 따라 각 사용자에 대한 접근 권한을 부여할 것인지가 결정된다. 이때, 사용자의 신상정보 및 작업 수행을 위해 필요한 자원 정보 등에 대한 정보를 얻게 되며 이러한 정보들은 작업 수행 시 발생되는 로컬 정보로부터 그리드 환경에서의 정보로의 변경을 위한 정보로 사용될 수 있다.



(그림 3) 그리드 접근 제어 시스템과 어카운팅 정보 서비스

3.2 그리드 접근 제어 시스템의 기능

본 논문에서는 Globus Toolkit에서의 문제점을 보완할 수 있는 기능을 제공하며, 이 기능은 다음 그림을 통해 설명될 수 있다.



(그림 4) 그리드 접근 제어 시스템의 기능

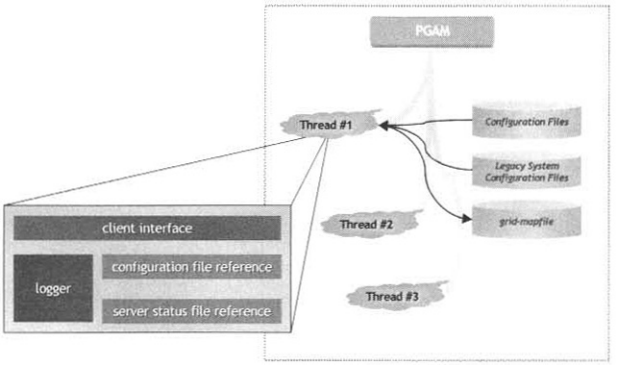
그리드 사용자가 사용 권한을 얻기 위해서 서버가 요구하는 사용자와 관련된 정보를 제공하면 서버 쪽에서는 이 정보들을 바탕으로 관리 정책을 참조하여 바인딩 여부를 판단하게 된다. 관리 정책은 시스템 관리자가 기존의 관리 정책을 그리드 환경에 적용할 수 있도록 하는 기능을 제공하며, 그리드 환경에서 새로 등장하는 요소에 대한 시스템의 정책을 반영하는 기능도 제공한다.

그리고 Globus Toolkit이 기본적으로 다수의 DN과 하나의 로컬 계정의 동시 결합을 허용하는데 따른 문제점을 해

결하기 위해, 자원 제공자가 원할 경우, 1:1 바인딩만 허용할 수도 있도록 설계되었다.

3.3 그리드 접근 제어 시스템의 구조

클라이언트의 요청에 대해서 효율적으로 대응하기 위해, 클라이언트의 요청이 발생하게 되면, 쓰레드를 발생시키게 되는데, 각 쓰레드는 아래 그림과 같은 모듈들로 구성된다.



(그림 5) 그리드 접근 제어 시스템의 구조

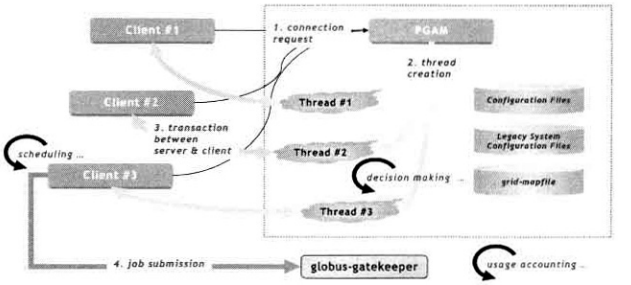
각 쓰레드는 기존의 시스템에서 사용하던 시스템 환경 설정 파일과 접근 제어 시스템에서 새로 도입된 환경 설정 파일을 참조하여 기동되고 grid-mapfile의 내용을 관리하기 위해 참조된 내용을 이용한다. 각 쓰레드는 client interface, logger, configuration file reference, server status file reference 등의 모듈로 구성되는데 각 모듈들의 세부 기능은 다음과 같다.

- client interface : 클라이언트와의 연결을 담당한다.
- logger : 동작 중 발생하는 모든 과정에 대한 기록을 담당한다.
- configuration file reference : 환경 설정 파일에 대한 레퍼런스로, 각 내용에 접근할 때는 semaphore가 적용된다.
- server status file reference : 계정의 상태와 같은 서버의 상태와 관련된 내용에 대한 레퍼런스로, 각 내용에 접근할 때는 semaphore가 적용된다.

클라이언트의 요청을 처리할 전담 쓰레드들이 생성되고 이들은 logger와 configuration file reference, server status file reference 등의 정보를 공유함으로써 다수의 클라이언트에게 동시에 일관성 있는 응답을 할 수 있도록 설계되었다. 특히, logger의 경우 다수의 쓰레드가 발생시키는 기록을 한꺼번에 관리하여야 하므로, 쓰레드 별로 따로 관리되다가 쓰레드 종료시 통합 로그에 병합되며 이 과정은 3.5절에서 설명한다.

3.4 그리드 접근 제어 시스템의 동작

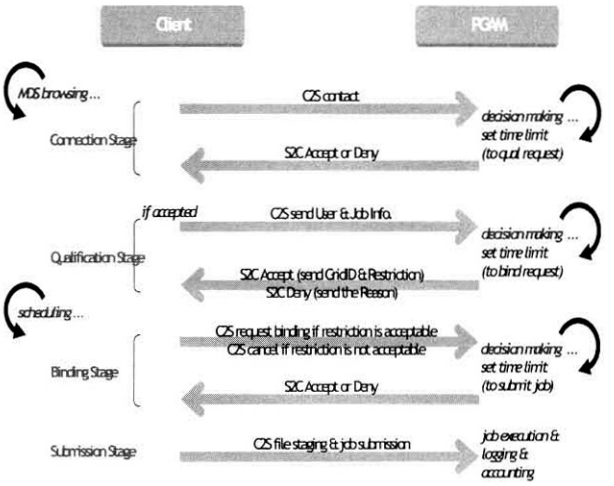
PGAM은 앞에서 설명한 바와 같이 각 클라이언트의 요청마다 전담 쓰레드가 발생시킨다. 각 쓰레드는 다음 그림과 같은 과정을 거쳐 클라이언트의 요청을 처리하며 환경 설정 파일들의 내용을 참조하여 grid-mapfile의 내용을 관리한다.



(그림 6) 그리드 접근 제어 시스템의 동작

클라이언트의 요청을 받으면, 전담 쓰레드가 발생되고 클라이언트가 제공한 사용자 정보와 사용자가 요구하는 권한 정보를 받아 환경 설정 파일을 참조하여 제공가능한 로컬 계정을 결정하게 된다. 클라이언트는 결합 가능한 로컬 계정의 리스트와 각 계정의 사용 권한을 서버로부터 받아 전체적인 스케줄링을 통해 결합할 계정을 선정하여 요청하게 된다. 서버에서 grid-mapfile의 내용을 수정하면 클라이언트는 비로소 자신의 작업을 실행할 수 있게 된다.

이러한 과정을 진행하는 동안 서버 쪽에서는 환경 설정 파일의 내용에 따라 연결 자체를 거부하거나 제한할 수도 있으며, 사용자가 요구하는 권한이 과도할 경우 사용자의 DN과 로컬 계정과의 결합을 거부할 수도 있다. 이 과정을 순서대로 그림으로 표현하면 아래와 같다.

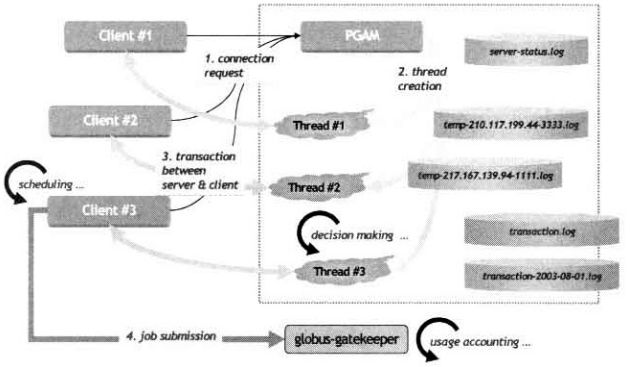


(그림 7) 그리드 접근 제어 시스템의 동작 과정

각 단계마다 다음 단계까지의 제한 시간을 결정하여 통보하고, 제한 시간 내에 다음 단계의 요청이 없는 경우 연결 자체를 취소하는 기능을 갖도록 함으로써, 서버의 부하를 줄이도록 하였다. 클라이언트는 결합 가능한 서버 쪽의 로컬 계정을 확보한 후, 제한 시간 내에 실제 결합시켜 사용할 계정을 선택하여 통보하여야 한다.

3.5 그리드 접근 제어 시스템의 로그 관리

어카운팅과 같은 기타 부가 서비스의 개발이 용이하도록 다양한 정보를 기록하도록 하였으며, 각 쓰레드별로 별도의 로그가 남는다. 쓰레드의 동작이 종료되기 직전에 쓰레드 내에서 발생한 모든 기록이 전체 로그로 옮겨지게 되며, 옮겨진 후 쓰레드 내의 로그가 기록되어 있던 로그는 삭제된다. 효율적인 관리를 위해서, 각 로그는 일정 시간 후에 일별, 주별, 또는 월별 로그로 분할되어 관리된다. 이 과정은 별도의 커멘드를 통해 수행되며, 일반적으로 cron job 형식으로 수행된다.



(그림 8) 그리드 접근 제어 시스템의 로그 관리

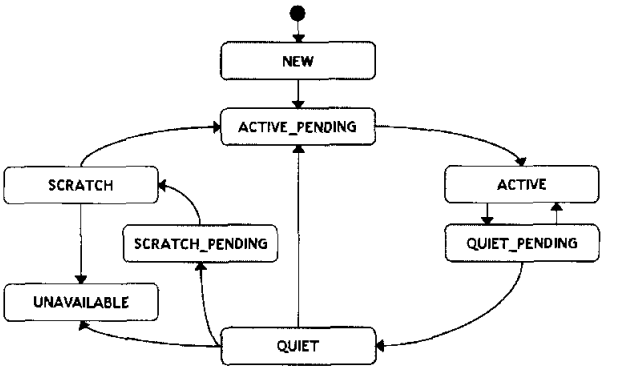
동일한 클라이언트에서 2명 이상의 그리드 사용자가 접근할 수도 있으므로 각 쓰레드의 개별 로그 파일의 경우, 파일의 이름을 클라이언트의 주소와 포트 번호를 동시에 사용하였으며 이 로그 파일들의 경우 각 쓰레드의 실행이 종료되면서 transaction.log로 옮겨진 후 삭제된다. 다음 그림은 실제 발생되는 로그의 일부분이다.

```
SL: Init>> (N) Handler Started ...
================================================================================
[ Connection Management Thread Started ]
...

[ Bind Stage ]
Date   : 2003/10/16 09:57:39
Client : 210.117.187.244-57026

Client sent following Binding Information.
      +--- DN       :: /O=Grid/OU=hi.ac.kr/CN=hdg
      +--- Status   :: student
      +--- RealName :: Hong, Gildong
      +--- GWID     :: gw02
SL: Bind>> (S) [Accepted] (ATL:60) DN /usr/local/bin is binded to Local User Name gw02
SC: Bind>> (S) [Accepted] (ATL:60) DN /usr/local/bin is binded to Local User Name gw02
--------------------------------------------------------------------------------
[ Connection Management Thread Ended ]
Date   : 2003/10/16 09:57:39
Client : 210.117.187.244-57026
================================================================================
```

(그림 9) 그리드 접근 제어 시스템의 로그 예(사용자의 요구에 의한 사용자의 DN과 로컬 계정의 결합)

3.6 계정의 상태 관리

Hacker[8]는 그리드 환경 내에서 다수의 그리드 사용자가 한정된 로컬 계정을 공유하는 과정에서 발생하는 충돌 문제를 해결하고 로컬 계정의 활용률을 극대화하기 위한 방법론을 제시하였는데, 이는 지정된 시간 내에 동일한 사용자가 재결합을 요구할 경우 이전에 결합되었던 계정을 재결합시켜주고, 각 계정의 상태를 Active, Quite, Scratch, Unavailable 등의 단계로 나누어 효율적으로 관리할 수 있도록 하는 특징이 있다. 본 논문에서는 이 과정을 단순화시켜 적용시켰으며, 시스템 동작 중에 실시간으로 이러한 정보를 유지하며, 사용자의 요청이 있을 때마다 환경 설정 파일과 함께 이러한 정보를 참조하여 적용시키고 있다. (그림 10)은 이러한 상태 전이도를 보여 주고 있으며, 그리드 사용자와 로컬 계정과의 온라인 연결의 유무에 따라 ACTIVE와 QUIET로 나뉘고 새로운 연결은 SCRATCH 상태에서 일정한 시간내에 다시 연결을 요청한 경우에는 QUIET 상태에서 곧바로 ACTIVE 상태로 연결되어 재사용할 수 있도록 하였다. 또한, 각 결합에 대해 grid-mapfile에서는 단순히 그리드 사용자의 DN과 로컬 계정의 결합만을 기록하고 있는데, 본 시스템에서는 그 외에 사용자의 실제 이름, 계좌 정보 등 다른 정보들도 해쉬테이블 형태로 기록하여 다른 부가 서비스에서 필요로 하는 정보가 있을 경우 추후 융통성 있게 필드를 추가할 수 있도록 하였다.



(그림 10) 계정의 상태 관리

3.7 환경 설정

본 논문에서 설계한 시스템은 각 자원의 고유한 운영 정책을 그대로 반영시키고 각 사이트의 자율성을 보장할 수 있도록 하였다. 예를 들면, limits.conf와 같은 파일을 이용하여 사용자의 시스템 사용 시 적용되어야 할 각종 제한 사항을 그리드 사용자에게도 그대로 적용시킬 수 있도록 하였다. 또한, 그리드 환경에 참여함에 따라, 새롭게 설정이 필요한 내용을 기존에 사용하던 설정 파일들과 비슷한 형식의 환경 설정 파일을 도입하여 시스템 관리자에게 그 관리를 맡김으로써 사이트의 자율성을 보장할 수 있도록 하였다.

다음 그림은 그리드 접근 제어 시스템에서 사용하는 새로 도입된 환경 설정 파일들의 리스트를 보여주고 있다.

```
IDMG> ls -l ./etc
합계 24
-rw-r--r--  1 app  staff   447 9월  3 11:47 grid-id.conf
-rw-r--r--  1 app  staff   486 9월 21 05:13 policy-host.conf
-rw-r--r--  1 app  staff   992 10월 23 10:31 policy-lifetime.conf
-rw-r--r--  1 app  staff   493 9월 23 09:33 policy-user.conf
```

(그림 11) 그리드 접근 제어 시스템의 환경 설정 파일들

```
GRID_ID
{
    max-binding = 5
}
gw1
{
    bindable = true
}
gw2
{
    bindable = true
}
gw3
{
    bindable = false
}
gw4
{
    bindable = true
}
```

```
acceptable-host-ips
{
        210.117.187.133,            #
        210.117.131.70,             #
        210.117.131.71              #
}
acceptable-host-groups
{
        210.117.187.230.249         #
}
defaults
{
        max-connection = 5
        max-connection-per-host = 1
        max-connection-per-group = 3
}
max-connection-per-host
{
        210.117.187.133 = 2
}
max-connection-per-group
{
        210.117.187.230.249 = 4
}
```

(그림 12) grid-id.conf, policy-host.conf

(그림 12)은 grid-id.conf와 policy-host.conf 파일의 내용 중 일부이다. 시스템 관리자는 grid-id.conf 파일을 이용하여 그리드 환경에 제공할 로컬 계정을 결정하고 이를 통해, 그리드 환경에 제공하는 자원의 양을 조절할 수 있다. policy-host.conf나 policy-user.conf 와 같은 파일 등을 통해 시스템에서 자원을 제공하거나 하지 않고자 하는 호스트, 사용자 등의 설정을 할 수 있으며, policy-lifetime.conf 파일을 통해 재 연결 제한 시간을 조절할 수 있다.

3.7.1 환경 설정을 통한 사이트 자율성 보장

그리드 접근 제어 시스템은 각 사이트의 고유한 운영 정책을 그리드 환경에서도 그대로 적용시킬 수 있는 방법을 제공하여야 한다. 본 연구에서 설계한 PGAM은 사이트의 자율성을 보장하기 위해 기존의 시스템에서 사용자의 권한을 제어하기 위해 사용하던 방법을 이용할 수 있도록 하고 이를 바탕으로 그리드 환경에 참여하였을 경우 필요한 기능을 새롭게 도입하였다.

예를 들어, 그리드 환경에 자원을 제공할 때 각 사용자의 권한을 제어하고 싶을 경우 각 시스템에서는 그리드 환경에 제공할 전용 계정을 만들고 각 계정에 적절한 권한(최대 디스크 용량, 큐, 최대 프로세스 수 등)을 기존에 사용하던 방식대로 부여한 후 grid-id.conf 파일에 전용 계정을 등록하면 된다. 시스템 운영 중에 변화가 있어 관리자가 제공하는 자원의 사용 권한을 줄이고 싶은 경우 기존의 시스템 설정 파일을 수정하여 각 계정의 권한을 축소하거나 grid-id.conf 파일을 수정하여 권한이 큰 계정을 비활성화 하면 된다.

4. 구현 및 실행 결과

4.1 구현 환경

본 논문에서 구현한 시스템은 다음과 같은 환경에서 구축 및 테스트 되었다. 다양한 플랫폼에서 동작할 수 있도록 Java와 Python을 이용하여 구현하였다.

〈표 1〉 실험환경

구분	환경
운영체제	Redhat Linux 7.2, 7.3, 8.0, 9.0 / IBM AIX 5.1L
언어	Python 2.2 or Later / wxPython / mod_python-4.1.1

4.2 실행 결과

다음은 텍스트 환경에서 테스트한 화면으로 좌측이 클라이언트, 우측이 서버이다.

(그림 13) 텍스트 모드 실행 결과

그리드 사용자가 자신의 정보를 보내면(Qual Stage), 서버는 그 정보를 보고 접근 권한 부여를 결정한다. (그림 12)과 같이 새로 추가된 환경 설정 파일에 사이트 관리자가 정한 정책대로 사용자에 대한 접근 권한 부여 여부를 가리게 된다. 그리드 사용자는 다시 그 정보를 받고 원하는 계정을 선택하면, 서버는 grid-mapfile에 사용자의 DN과 로컬 계정을 결합시킨 후 결합된 정보를 통보해 준다. 그리드 사용자는 결합 정보를 바탕으로 자신의 작업을 수행시키기 위한 스케줄링 작업을 마치게 되며 지정된 시간 내에 작업을 실행시키면 된다.

(그림 14)와 (그림 15)는 사용자의 신상 정보를 보냈을 때, 각각 사용자의 신상 정보가 거부 또는 허락된 경우 서버로부터 수신 받은 내용이 화면에 출력된 것이다. 허락된 경우 사용자가 결합 가능한 계정의 목록을 받게 되며 사용자는 이 목록에서 원하는 계정을 선택하여 다시 요청하면 서버에서 사용자의 DN과 로컬 계정을 결합시켜준다. 이후 사용자는 서버에서 제공한 계정을 통해 자신의 작업을 수행할 수 있다.

```
CL: Critical Error!!>> Qualification of User Info. Failed.
    [Denied] Not Allowable User Information.
```

(그림 13) 사용자의 신상 정보 거부

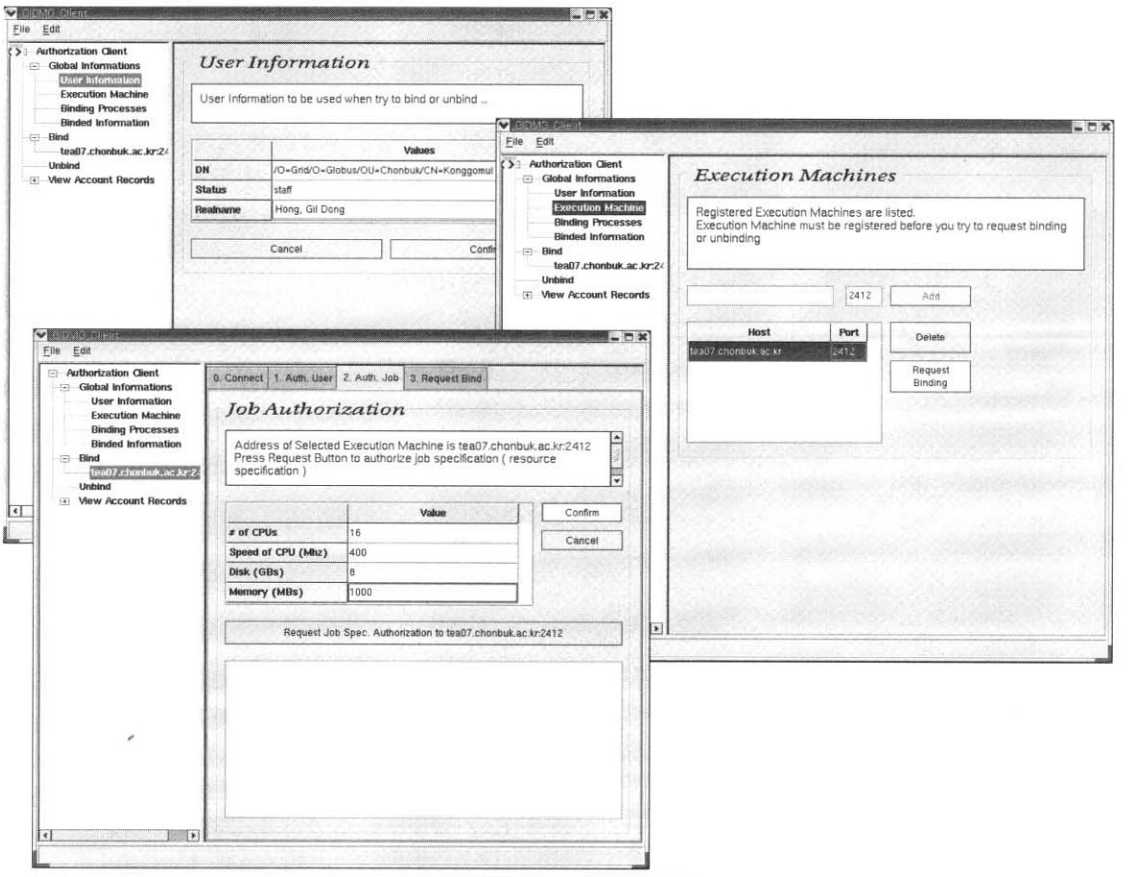
```
CL: Qual>> (S) [Accepted] (ATL:60)
{gw02 : { DN : , STAT : 0, PSTAT : 1, LIFETIME : 0, HOST : , PORT : }}
{gw03 : { DN : , STAT : 0, PSTAT : 0, LIFETIME : 0, HOST : , PORT : }}
```

(그림 14) 결합 가능한 계정 목록

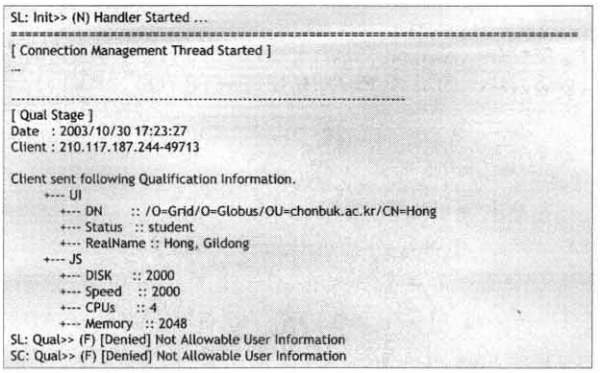
사용자가 서버에게 보내는 사용자 정보 및 자원의 명세 등은 python 언어의 사전(dictionary) 자료형을 이용하여 전송하고 있으므로 추후 새로운 필드의 사용자 정보 및 자원 명세가 추가될 필요가 있을 경우 간단하게 추가할 수 있다.

다음은 GUI 환경에서 사용하기 위한 클라이언트 툴이다. GUI 툴은 python 언어로 개발되었다. Python 언어는 Java처럼 가상 머신(Virtual Machine)을 이용하는 방식을 취하고 있으므로 python interpreter만 설치되어 있다면 운영체제나 하드웨어와 관계없이 어느 플랫폼에서나 거의 동일한 동작을 보여주는 언어이다.

다음 그림은 사용자가 자신의 정보와 작업에 필요한 자원의 명세를 보냈을 때 서버쪽에서 거부한 결과가 남은 기록이다. 서버 쪽에서는 허용 또는 거부하고 싶은 사용자 또는 서버에 관한 설정을 policy-user.conf, policy-host.conf 등을 통해 반영할 수 있다.

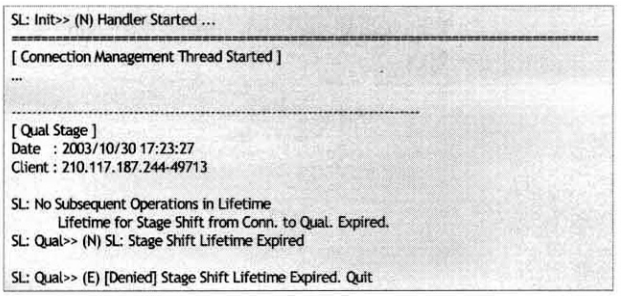


(그림 16) GUI 모드 실행 결과



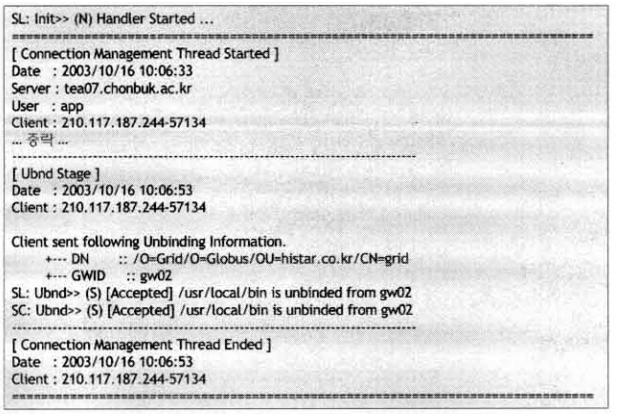
(그림 17) 그리드 접근 제어 시스템의 로그 예(특정 사용자 정보에 대한 접근 거부)

아래 그림은 사용자가 일정 시간 내에 속행되어야 할 작업을 진행하지 않은 경우 서버에서 강제로 연결을 종료시키면서 발생시키는 기록이다. 서버에서는 policy-lifetime.conf 파일을 이용하여 제한 시간을 조절할 수 있다.



(그림 18) 그리드 접근 제어 시스템의 로그 예(제한 시간 만료에 따른 강제 연결 종료)

다음 그림은 사용자의 요청에 의해 사용자의 DN과 로컬 계정의 결합을 해제하였을 때 남는 기록이다. 결합 시 발생하는 기록은 (그림 9)에서 보여진 바 있다.



(그림 19) 그리드 접근 제어 시스템의 로그 예(사용자의 요구에 의한 사용자의 DN과 로컬 계정의 결합 해제)

4.3 관련 연구와의 비교

이 장에서는, PERMIS (Privilege and Role Management Infrastructure Standards Validation), Akenti, CAS (Community Authorization Service)와 같이 유사한 기능의 시스템들과 간단한 비교를 해본다.

4.3.1 PGAM vs. PERMIS

PERMIS[21]는 각 사용자에게 role을 부여하고 role에 따라 그 권한이 제한된다. 특히 PGAM과 다른 점은 각 사용자의 Certificate가 하나의 저장소에 집중된다는 것이다. 이와는 반대로, PGAM에서는 각 사용자의 Certificate가 각 사용자들에게 분산되어 있어 훨씬 더 유연한 구조를 가지고 있다. 사용자의 요구에 대해 PA (Privilege Allocator)에 의해 그 role이 부여되며, 이는 PGAM에 비해 상대적으로 사이트의 자율성에서 제한을 받는 형태이다.

4.3.2 PGAM vs. Akenti

Akenti[22]는 웹 자원에 대한 접근 제어를 목표로 설계되어 있다. 특히, 이는 VO와 같은 그리드 환경에서 요구하는 특성에 적용되어 사용하기가 어려운 측면이 있다는 것을 의미한다. 접근 제어를 자원쪽에 위치시킨다는 점은 본 시스템과 유사하다.

4.3.3 PGAM vs. CAS

CAS[23]는 Globus 팀에서 개발한 것으로 OGSA 기반으로 개발되었다. 사용자가 자원에 대한 접근 권한을 얻기 위해서는 CAS 서버를 통해 새로운 Certificate을 발급받아야 하며, 여기에는 접근 권한에 대한 명세가 포함되어 있다. 각 자원은 사용자가 제시한 이 Cerficate에 들어있는 접근 권한에 따라 자원을 제공한다. 이는 각 사이트가 자율적으로 사용자의 접근 권한 요청 시 판단을 하는 게 아니라 CAS 서버에 사전에 접근 권한을 부여하는 규칙을 제공해야 한다는 것을 의미한다. PGAM은 각 사이트에서 접근 권한을 직접 판단한다.

5. 결 론

본 논문에서는 그리드 환경에 적용하기 위한 접근 제어 시스템으로 PGAM(Policy based Grid Account Manager)을 설계 및 구현하였다. 본 시스템은 그리드 환경 구축 시 전 세계적으로 가장 많이 사용되는 미들웨어 중의 하나인 Globus Toolkit을 기반으로 하고 있다.

PGAM은 그리드 환경에서 각 사이트의 자율성을 보장하고 사용자의 스케줄링 작업 시 도움을 줄 수 있도록 설계 및 구현되었다. 그리드 환경에서의 모든 서비스는 각 자원을 제공하는 사이트의 고유한 운영 정책에 영향을 주지 않고 각 사이트의 운영 정책에 그리드 환경에서도 그대로 반영될 수 있도록 설계되어야 한다. 따라서 본 시스템은 사이트의 자율성을 보장하기 위해서, 기존에 로컬 사용자들의 권한을 제어하던 방법(시스템 설정 파일의 수정, 디스크 쿼터 제한 등)을 사용할 수 있도록 하고 새로운 환경 설정 파

일들을 추가하여 각 계정의 그리드 환경 제공 여부 등을 제어할 수 있도록 하였다. 그리드 환경의 사용자들의 사이트 접근도 각 사이트의 자율에 맡기도록 함으로써, 각 사이트의 그리드 환경에 대한 참여 의욕을 고취하였다. 그리고 그리드 사용자의 로컬 자원에 대한 접근 권한을 부여하면서 다른 부가 서비스에 필요한 정보를 얻을 수 있도록 설계하였고 다양한 플랫폼에서도 동작할 수 있도록 Python 언어를 사용하였다. 또한, 그리드 사용자의 요구가 있을 때에만 접근 권한을 부여함으로써 시스템의 보안에 유리하게 하는 효과가 있다. 그리드 사용자의 입장에서는 자신의 어플리케이션을 수행하기 위해 필요한 자원을 스케줄링 할 때, 반드시 거쳐야 할 접근 권한 획득 과정을 일정 자격을 갖춘 사용자의 경우 쉽게 해결할 수 있는 도구를 제공하였다.

앞으로, 실제 시스템 운용 및 부가 서비스에 필요한 그리드 사용자를 포함하는 클라이언트의 정보에 대한 세부적인 연구와 서버의 더욱 더 세련된 계정 관리 메커니즘의 반영 방법에 대한 연구가 필요하다. 접근 권한 부여를 위한 정보로 전송되는 사용자 정보 및 자원 명세 등을 XML로 다루기 위한 연구도 필요하다. 또한, 사용자의 신분 인증 기능에 대한 보강 연구와 자원 접근 권한 획득 시 사용자에게 공지할 내용에 대한 연구가 필요하다.

참 고 문 헌

[1] Foster, C. Kesselman(eds), "The Grid : Blueprint for a New Computing Infrastructure" Mogan Kaufmann Publishers, 1998.

[2] Foster, C. Kesselman(eds), S. Tuecke "The Anatomy of the Grid: Enabling Scable Virtual Organizations", Intl. J. Supercomputer Applications, 2001.

[3] S. Mullen et al, "Grid Authentication, Authorization and Accounting Requirements Research Document", (draft), GGF8, 2003

[4] Sebastian Ho, "GridX System Design Documentation", (draft), Bioinformatics Institute, 2002

[5] A. Beardsmore et al, "GSAX (Grid Service Accounting Extensions)", (draft), GGF6, 2002

[6] R. Baker et al, "Conceptual Grid Authorization Framework and Classification", (draft), GGF8, 2003

[7] K. Czajkowski, I. Foster, et al, "A Resource Managment Architecture for Metacomputing Systems", Proc. of the 4th Workchop on Job Scheduling Strategies for Parallel Processing, 1998

[8] ThomasJ. Hacker, Brian D. Athey, "Account Allocations on the Grid", Center for Parallel Computing University of Michigan. 2000.

[9] http://www.girdforum.org

[10] http://www.globus.org

[11] http://www.gridforumkorea.org

[12] http://www.eurogrid.org

[13] http://www.apgrid.org

[14] http://www.ipg.nasa.gov

[15] http://www.ngrid.or.kr

[16] http://pacct.sourceforge.net/

[17] M. Kupczyka, N. Meyer, P. Wolniewicz, "Symplifying Administration and Management Processes in Polish National Center", 2002

[18] http://www.to.infn.it/grid/accounting

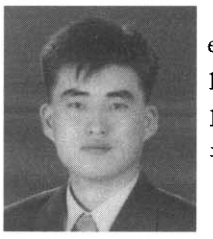
[19] A. Barmouta, R. Buyya, "GridBank: A Grid Accounting Service Architecture (GASA) for Distributed Systems Sharing and Integration", 26th Australasian Computer Science Conference (ACSC2003) Adelaide, Australia Feb 2003

[20] K. Czajkowski, et al, "A Resource Management Architecture for Metacomputing Systems", Proc. IPPS/SPDP '98 Workshop on Job Scheduling Stategies for parallel Processing, pg. 62-82, 1998

[21] http://www.permis.org

[22] http://www-itg.lbl.gov/Akenti/

[23] L. Pearlman, et al, "A Community Authorization Service for Group Collaboration", IEEE Workship on Policies for Distributed Systems and Networks (2002)



김 법 균

e-mail: kyun@chonbuk.ac.kr

1994년 전북대학교 컴퓨터공학과(공학사)

1997년 전북대학교 컴퓨터공학과(공학석사)

관심분야 : 분산 및 병렬처리, Grid

안 동 언

e-mail: duan@moak.chonbuk.ac.kr

1981년 한양대학교 전자공학과(공학사)

1987년 KAIST 전산학과(공학석사)

1995년 KAIST 전산학과(공학박사)

1995년~현재 전북대학교 전자정보공학부 부교수

2001년~2002년 전북대학교 정보검색시스템연구센터 센터장

관심분야 : 정보검색, 한국어정보처리, 문서분류, 문서요약, Grid

정 성 종

e-mail : sjchung@moak.chonbuk.ac.kr
1975년 한양대학교 전기공학과(공학사)
1981년 Houston대학교 전자공학과(공학석사)
1988년 충남대학교 전산공학과(공학박사)
1985년~현재 전북대학교 전자정보공학부 교수
1996년~1998년 전북대학교 전자계산소 소장
2001년~현재 전북대학교 BK21 전자정보사업단 단장
관심분야 : 정보검색, Grid

장 행 진

e-mail : hjjang@kisti.re.kr
1987년 서울산업대학교 전자계산학과 학사
1994년 호서대학교 전자계산학과 석사
현재 한국과학기술정보연구원 선임연구원
관심분야 : 네트워크, 분산 및 병렬처리, 그리드

박 형 우

e-mail: hwpark@kisti.re.kr
1985년 서울시립대학교 전자공학과 공학사
1996년 성균관대학교 정보공학과 공학석사
2001년 성균관대학교 전기전자컴퓨터공학과 공학박사
현재 한국과학기술정보연구원 그리드연구실 실장
관심분야 : 그리드, 차세대 TCP, 인터넷 관리(QoS, 보안)