그리드 작업 실행 시스템의 설계 및 구현

김상완°, 함재균, 권오경, 이종숙
한국과학기술정보연구원 슈퍼컴퓨팅센터
{sangwan, jaehahm, okkwon, jsruthlee}@kisti.re.kr

A Design and Implementation of Grid Job Execution System

Sangwan Kim°, Jaegyoon Hahm, Ohkyoung Kwon, Jongsuk Lee
KISTI Supercomputing Center

요 약

인터넷과 네트워크 속도의 발전으로 인하여 그리드 컴퓨팅(grid computing)의 필요성 및 가능성이 점차 증대되고 있다. 사용자는 다양한 종류의 수많은 그리드 자원을 그리드 미들웨어를 이용하여 일관된 방법으로 이용할 수 있으며, 자신의 작업을 원격에서 실행 및 제어할 수 있다. 그리드 미들웨어 중에서도 그리드 자원에서 작업을 실행시킬 수 있도록 그리드 자원을 통제하는 그리드 작업 실행 시스템은 미들웨어의 가장 핵심적인 부분이라고 할 수 있다. 본 연구에서 개발된 그리드 작업 실행 시스템에서 사용자의 작업은 작업 기술 언어로 표현되어, 자원 관리 노드에서 작업의 상태 변화에 따라 절절한 단계를 거쳐 실행된다. 또 자원의 동시 할당을 위한 자원 예약 기능을 제공하며, 서비스 형태로 외부에 공개 되므로 클라이언트는 플랫폼에 관계없이 제공된 서비스를 이용할 수 있는 특징을 지닌다.

1. 서 론

그리드 컴퓨팅(grid computing)[1]은 네트웍으로 연결되어 있는 다양한 종류의 컴퓨팅 자원(computing resource)을 통합하여 보다 효율적으로 사용하기 위한 컴퓨팅 방식이다. 인터넷의 발전과 네트웍 속도의 증가로 인하여 원격에 있는 컴퓨터를 로컬에 있는 컴퓨터처럼 쓸 수 있게 됨에 따라 그리드 컴퓨팅 기술에 대한 연구가 전 세계적으로 더욱 활발히 이루어지고 있다. 그리드에 연결되는 컴퓨팅 자원은 서로 다른 관리 도메인에 속해 있거나, 지역적으로 분산되어 있거나, 혹은 이기종의 시스템으로 구성되어 있는 것을 전제로 하고 있다.

기존의 고전적인 컴퓨팅 방식에서는 사용자가 직접 자신의 작업을 실행하고 실행중인 작업을 제어하였지만, 그리드 컴퓨팅 환경에서는 사용자가 그리드 미들웨어(grid middleware)를 통하여 자신의 작업을 실행하고, 제어함으로써 일관된 방법으로 다양한 그리드 자원을 이용할 수 있게 한다. 그리드 미들웨어는 사용자의 작업을 성공적으로 실행할 수 있는 그리드 자원을 선택하여 작업을 실행하고, 모니터링 하며 실행된 결과를 사용자가 원하는 위치로 옮겨주는 역할을 담당한다. 이 중에서 작업을 그리드 자원에서 실행하고, 그리드 자원을 관리하는 작업 실행 시스템은 미들웨어의 가장 기본적인 기능이라고 할 수 있다.

본 연구에서는 그리드 작업 실행 시스템을 설계하고 구현하였다. 본 논문의 구성은 다음과 같다. 제2절에서는 그리드 작업을 기술하기 위해 필요한 사항과 그것을 표현할 수 있는 작업 기술 언어에 대해 설명한다. 제3절에서는 본 연구에서 설계 및 구현된 작업 실행 시스템의 구조와 동작원리에 대해서 설명한다. 제4절에서는 그리드 자원의 동시 할당을 위해 필요한 자원 예약 기능에 대해서 설명하고 제5절에서 결론을 맺는다.

2. 작업의 정의 및 표현

배치 작업(batch job)이란 사용자의 개입 없이 실행될 수 있는 컴퓨터 프로그램을 말하며, 간단한 유닉스 쉘 스크립트에서부터 여러 개의 계산 노드를 필요로 하는 병렬 프로그램을 모두 포함한다. 작업의 실행도중 사용자가 직접 개입하여야 하는 작업을 대화식 작업(interactive job)이라고 하며, 이 논문에서 작업은 배치 작업만을 의미하는 것으로 한다.

2.1. 작업의 기술

그리드에서 사용자가 작업을 수행하려고 할 때 가장 먼저 해야 할 일은 자신이 수행할 작업을 미들웨어가 이해할 수 있는 적절한 방법으로 기술하는 것이다.[2] 작업을 기술할 때는 다음 사항들이 포함되어야 한다.

- 필요한 파일 및 입력 데이터가 있는 곳의 위치 : 실행될 스크립트 또는 바이너리와 입력 파일들이 있는 곳의 URL을 명시하여야 한다. 작업 실행 시스템은 작업을 시작하기 전에 이 URL에서 파일들을 가져온다. 사용자가 작업의 실행에 필요한 입력 데이터를 파일의 형태로 표현하는 것이 어려운 경우에는 적절한 방법으로 그것을 기술해 주어야 한다. 예를 들어, 작업의 표준 출력으로 사용할 수 있는 네트웍 상의 특정 호스트의 주소와 포트 번호는 파일의 형태로 표현이 어려운 입력 데이터라고 할 수 있다.
- 작업을 실행하는 방법 : 어떤 파일을 어떻게 실행하며, 입력과 출력을 어떤 형태로 사용할 것인지를 기술하여야 한다.
- 작업의 종료 후 실행결과를 옮길 위치 : 작업의 실행결과로 여러 개의 출력 파일이 생성될 수 있으며, 생성된 파일을 전송하기 위한 FTP 주소와 같은 URL을 명시하여야 한다.

2.2. 작업 기술 언어 JSML

사용자가 위와 같은 것을 명시하기 위해서는 작업을

기술할 수 있는 언어가 필요한데, 본 연구에서는 JSML (Job Specification Markup Language)이라는 XML기반의 작업 기술 언어를 개발하였다. JSML은 확장성을 고려하여 설계되었다. 그림1은 JSML의 예를 보여준다. 크게 나누어 <stagein>, <job>, <stageout>의 세 부분으로 구성되는데, <stagein>은 입력 파일을 가져올 수 있는 URL (HTTP 또는 FTP주소)을 명시하고 있고, <job>은 실행할 실행 파일과 아규먼트(argument), 환경변수, 표준 입출력 등을 명시하고 있다. <stageout>은 생성된 파일을 옮겨 놓을 수 있는 URL (FTP주소)를 명시하고 있다.

```
<?xml version="1.0" encoding="UTF-8"?>
<jsml version="1.0">
<submit>
  <stagein>
    <file type="remote" name="exec.sh">
      <url>http://host.example.com/path/to/exec.sh</url>
    </file>
    <file type="remote" name="input">
      <url>http://host.example.com/path/to/input</url>
    </file>
  </stagein>
  <job name="my_job" type="single">
    <executable><file name="exec.sh"/></executable>
    <arguments>arg1 arg2 arg3<arguments>
    <environment/>
    <stdin><file name="input"/></stdin>
    <stdout><file type="local" name="stdout.out"/></stdout>
    <stderr><file type="local" name="stderr.out"/></stderr>
  </job>
  <stageout>
    <ftp>
      <server>host.example.com</server>
      <username>user</username>
      <password>pass</password>
      <path>/path/to/dir/</path>
      <file name="stdout.out">stdout.out</file>
      <file name="stderr.out">stderr.out</file>
    </ftp>
  </stageout>
</submit>
</jsml>
```

그림1. JSML의 예

2.3. 작업기술 방법의 확장

그림1은 아주 간단한 형태의 작업 기술 방식의 예를 보여주고 있다. 상황이 조금 더 복잡해짐에 따라 사용자는 다음과 같은 기능을 요구할 것이며, JSML은 다음과 같은 요구 사항을 만족시키도록 쉽게 확장될 수 있다.

- 여러 유사한 작업을 한꺼번에 기술하기를 원하는 경우: 예를 들어 파라미터 스윕(parameter sweep)의 경우와 같이, 같은 실행 파일에 입력 아규먼트만 다르게 하여 여러 개의 작업을 실행하기를 원할 경우 서로 다른 입력 아규먼트에 의해 각각 독립된 작업이 실행되어야 한다.
- 여러 작업들 간의 상관관계를 명시하기를 원하는 경우: 작업A의 실행결과가 작업B의 입력으로 이용될 경우, 작업A가 종료된 다음 작업B가 시작되어야 하는 상관관계가 생기게 되며, 작업 기술언어는 이러한 상관관계를 표현 할 수 있어야 한다.
- 그 외에 작업과 관련된 부가적인 정보를 표현할 수 있어야 한다. 작업의 종류에 따라서 작업 실행 시스템이 처리해야 할 절차가 달라질 수 있는데, 이러한 경우 작업의 종류에 따라 추가적으로 필요한 정보를 표현할 수 있어야 한다.

3. 작업 실행 시스템

사용자가 기술한 작업은 작업 기술 언어로 표현되어 작업 실행 시스템으로 전달되고, 작업 실행 시스템은 그것을 해석하여 정해진 절차에 따라 작업을 실행 시킨다. 작업 실행 시스템은 입력 파일 준비, 작업의 실행, 출력 파일 전송의 크게 세 단계를 거쳐 작업을 실행한다.

3.1. 작업 실행 시스템 구성

그림2는 본 연구에서 설계한 작업 실행 시스템의 구성도이다.

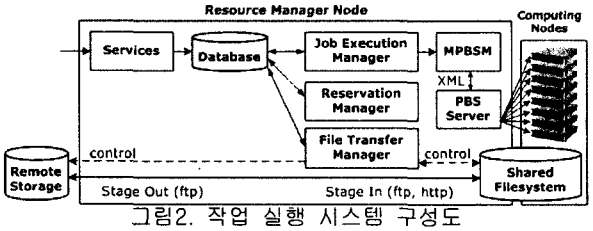

그림2. 작업 실행 시스템 구성도

그리드 자원은 그리드 작업을 실행할 수 있는 컴퓨터이며, 슈퍼컴퓨터나 클러스터와 같은 장비가 그리드 자원에 해당된다. 그림2에서 자원 관리 노드(resource manager node)는 그리드 자원을 통제하고, 작업을 실행하는 기능을 담당한다. 자원 관리 노드는 서비스, 데이터베이스, 작업 실행 관리자, 파일 전송 관리자, 자원 예약 관리자로 나뉘며, 각각이 담당하는 기능은 다음과 같다.

- 서비스 : 사용자의 작업 실행 요청을 받아들이는 기능을 하며, 웹 서비스(Web Service)와 같은 형태로 외부에 공개 되어 있다. 서비스는 사용자가 작업을 제출하고, 작업의 상태를 모니터하며, 작업을 제어할 수 있는 유일한 접촉점이 된다. 사용자가 제공한 작업 내용은 2절에서 설명한 작업 기술 언어로 표현되어 서비스에 전달되며, 서비스는 그것을 해석하여, 새로운 작업을 데이터베이스에 등록한다.
- 작업 실행 관리자 : 작업에 대한 처리 절차를 전반적으로 맡아 관리한다. 데이터베이스에 등록된 모든 작업은 상태를 가지며, 작업 실행 관리자는 주기적으로 데이터베이스를 검사하여, 작업의 상태를 변화시켜 나간다.
- 파일 전송 관리자 : 파일 전송 관리자는 입출력 파일을 전송하는 역할을 담당하며, 작업이 시작되기 전에 입력 파일을 작업의 실행이 끝난 다음에 출력파일을 전송한다.
- 데이터베이스 : 제출된 작업의 상태 및 전송 중인 파일 및 자원 예약 상태가 데이터베이스에 저장되며, 관리자는 데이터베이스를 주기적으로 읽어 자신이 처리해야 할 항목이 있는지를 검사하게 된다.
- 자원 예약 관리자 : 자원의 동시 할당의 경우와 같이 특정 시간에 자원의 할당을 보장받기 위해서는 자원에 대한 예약이 가능해야 하며, 자원 예약 관리자는 자원의 예약상황을 관리하고, 상태를 갱신시킨다.

3.2. 작업 실행 상태

서비스에 의해 등록된 작업은 작업 실행 관리자와 파

일 전송 관리자에 의해서 상태가 갱신되며, 최종 상태에 이르면 종료된다. 그림3은 작업 상태의 변화하는 과정을 나타내고 있다.

그림3. 작업의 실행 상태와 변화

3.3. 작업 실행 관리자

작업 실행 관리자는 데이터베이스에 등록된 작업을 처리 순서에 따라 통제해 주는 기능을 하지만, 작업을 스스로 실행시키지는 않는다. 실제 작업은 그리드 자원에 연결되어 있는 계산 노드에서 실행되며, 계산 노드에서 작업을 실행하기 위해서는 PBS(Portable Batch System)[3]과 같은 로컬 배치작업 시스템의 도움을 받아야 한다. 본 연구에서는 작업 실행 관리자가 PBS를 통하여 작업을 실행할 수 있게 하기 위해 MPBSM (MoreDream PBS Manager)라는 인터페이스 모듈을 설계하였다. MPBSM은 PBS 서버 데몬과 같이 네트웍상의 특정 포트에서 동작하며, PBS 클라이언트 명령어(qsub, qstat, qdel, etc)와 같은 기능을 담당한다. 그림4는 MPBSM의 역할을 보여준다.

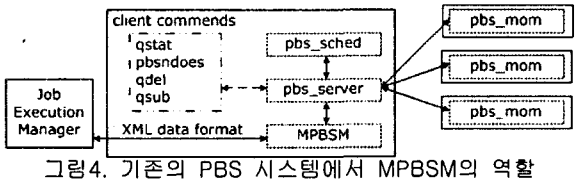

그림4. 기존의 PBS 시스템에서 MPBSM의 역할

MPBSM은 작업 실행 관리자와 XML형식의 데이터를 질의-응답 방식으로 교환하며 작업 실행 관리자의 통제를 받는다.

4. 자원의 동시할당 및 자원 예약

제3절에서 설명한 작업 실행 시스템은 서비스 형태로 외부에 공개 되므로, 다양한 클라이언트(또는 사용자)로부터 자원 할당 요청을 받게 된다. 이때 클라이언트는 그리드 자원에서 이용할 수 있는 자원의 양을 서비스에 질의를 통해 미리 조사한 후 작업을 제출하게 되는데, 질의 시점과 작업 제출 시점의 시간 차이로 인하여, 질의 결과 얻은 자원의 양을 할당 받을 수 없는 상황이 일어날 수 있다. 예를 들어 그림5에서와 같이 클라이언트 A와 B가 같은 그리드 자원을 할당하려고 할 경우 클라이언트 A의 개입으로 인하여 클라이언트 B는 요청한 개수의 CPU를 할당받지 못하게 될 수도 있다.

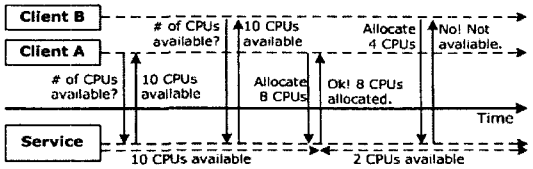

그림5. 자원 할당이 실패하는 경우

이 문제를 해결할 수 있는 방법 중 하나는 클라이언트가 그리드 자원을 미리 예약할 수 있게 함으로써 할당시점에서 예약된 범위 내에서 자원할당을 보장하는 것이다. 예약된 자원은 다른 클라이언트가 할당할 수 없도록 자원 예약 매니저에 의해서 관리되며, 정해진 시간 내에 할당되어야한다. 할당할 수 있는 시간을 초과한 예약은 무효화 되어 다시 다른 클라이언트가 이용할 수 있게 된다. 그림5는 이러한 예약상황의 상태 변화를 나타내고 있다.

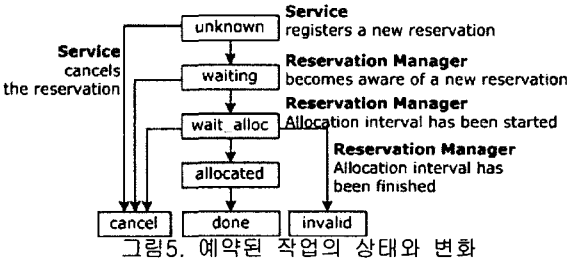

그림5. 예약된 작업의 상태와 변화

자원 예약 기능은 서비스를 통해 외부에 제공되며, 예약된 자원은 데이터베이스의 자원 할당 테이블에 등록되어 자원 예약 매니저에 의해 관리된다.

5. 결론

그리드 미들웨어는 분산된 그리드 자원을 통합하여 사용자에게 단일한 접근방법을 제공해 준다. 작업 실행 시스템은 그리드 자원을 제어 하여 사용자의 작업을 실행하는 역할을 담당한다. 본 연구에서 개발된 작업 실행 시스템은 XML기반의 확장성 있는 작업 기술 언어를 지원하며, 동시 할당을 위한 자원 예약 기능 및 기존의 PBS와 같은 배치 작업 시스템과도 연동된다.

참고문헌

[1] I. Foster and C. Kesselman, "The Grid: Blueprint for a New Computing Infrastructure", Morgan Kaufmann, 1998

[2] Global Grid Forum Job Submission Description Language Working Group http://www.epcc.ed.ac.uk/~ali/WORK/GGF/JSDL-WG/

[3] Portable Batch System, http://www.openpbs.org/