http://dx.doi.org/10.5392/JKCA.2014.14.05.008

슈퍼컴퓨팅환경에서의 대규모 계산 작업 처리 기술 연구

HTCaaS(High Throughput Computing as a Service) in Supercomputing Environment

김석규*, 김직수**, 김상완**, 노승우**, 김서영**, 황순욱**
상명대학교 게임학과*, 한국과학기술정보연구원 국가슈퍼컴퓨팅연구소**

Seok-kyoo Kim(skkim@smu.ac.kr)*, Jik-Soo Kim(jiksoo.kim@kisti.re.kr)**,
Sangwan Kim(sangwan@kisti.re.kr)**, Seungwoo Rho(seungwoo0926@kisti.re.kr)**,
Seoyoung Kim(sssyyy77@kisti.re.kr)**, Soonwook Hwang(hwang@kisti.re.kr)**

요약

슈퍼컴퓨팅 자원들은 주로 MPI와 같은 메시지 교환 인터페이스에 기반한 통신 집적도가 높은 고성능 컴퓨팅(HPC: High Performance Computing) 응용 분야를 지원하는데 활용되어 왔다. 반면에, 대규모 계산처리 컴퓨팅(HTC: High Throughput Computing) 방식의 패러다임은 주로 계산 집적도가 높고(상대적으로 적은 I/O 연산), 독립적인(작업들 간의 통신이 적음) 많은 수의 작업을 처리하는 것을 요구하고 있다. 국내에서도 고에너지 물리, 신약개발, 핵물리와 같은 연구 분야를 중심으로 대규모 컴퓨팅 자원을 요구하는 계산처리에 대한 수요가 증가하고 있다. 본 논문에서는 이러한 HTC 과학 응용들에 대한 효율적인 지원을 국가차원의 슈퍼컴퓨팅 분산 환경에서 제공하기 위해 연구/개발되어진 대규모 계산처리 서비스(HTCaaS: High Throughput Computing as a Service)의 전체 구조 및 구성 요소, 실행 시나리오 및 실제 응용 적용 사례 등에 대해 서술한다.

■ 중심어 : | 대규모 계산처리 컴퓨팅 | 메타작업 | 멀티레벨 스케줄링 | 대규모 계산처리 서비스 |

Abstract

Petascale systems(so called supercomputers) have been mainly used for supporting communication-intensive and tightly-coupled parallel computations based on message passing interfaces such as MPI(HPC: High-Performance Computing). On the other hand, computing paradigms such as High-Throughput Computing(HTC) mainly target compute-intensive (relatively low I/O requirements) applications consisting of many loosely-coupled tasks(there is no communication needed between them). In Korea, recently emerging applications from various scientific fields such as pharmaceutical domain, high-energy physics, and nuclear physics require a very large amount of computing power that cannot be supported by a single type of computing resources. In this paper, we present our HTCaaS(High-Throughput Computing as a Service) which can leverage national distributed computing resources in Korea to support these challenging HTC applications and describe the details of our system architecture, job execution scenario and case studies of various scientific applications.

■ keyword : | High-Throughput Computing | Meta-Job | Multi-level Scheduling | HTCaaS |

접수일자 : 2014년 02월 04일
수정일자 : 2014년 03월 11일
심사완료일 : 2014년 03월 12일
교신저자 : 김직수, e-mail : jiksoo.kim@kisti.re.kr

I. 서 론

대규모 계산이나 많은 양의 데이터 접근이 요구되는 계산과학 분야에서는 복잡한 문제 풀이를 위해 다수의 고성능 계산 자원을 동시에 활용하는 대규모 계산처리 컴퓨팅(High-Throughput Computing) 방식을 채택하여 사용하고 있다. 기존의 HPC 방식과는 달리, HTC 방식의 컴퓨팅 패러다임은 주로 계산 집적도가 높고(상대적으로 적은 I/O 연산), 독립적인(작업들 간의 통신이 적음) 많은 수의 작업을 처리하는 것을 요구하고 있다. 하지만 계산 과학 연구자들이 직접 계산 자원을 구축하고 관리하는 데에는 추가적인 지식 습득이 요구되며, 동시에 많은 비용이 소비된다. 또한 이같이 직접 계산 자원을 구축하는 경우, 실제로 만족할 만한 성능을 제공하지 못하는 경우가 대다수이다. 이러한 한계점을 해결하고자 수년간 서비스 그리드(EGI[1]) 또는 데스크톱 그리드(BOINC[2])와 같은 계산 자원이 제공되어 왔으며, 최근에는 IT 및 하드웨어 기술이 급속도로 발전하여 저비용의 최첨단 자원의 활용이 가능함에도 불구하고 실제 응용 과학자 또는 연구자들의 접근 및 활용에는 여전히 한계가 존재한다.

국내에서도 일부 고에너지 물리, 바이오와 같은 연구분야를 중심으로 대용량 계산 처리에 대한 수요가 증가하고 있다. 하지만 대부분의 고에너지 물리 실험의 경우, 외국의 연구소에서 실제 실험이 수행되며 기반 인프라를 제공하기 때문에 국내 연구자들은 외국에서 제공하는 환경을 이용하고 있는 실정이고, 대용량 계산처리를 위한 기술 개발은 거의 이루어지고 있지 않다. 이에 따라서, 한국과학기술정보연구원에서는 2008년도부터 신약개발 분야를 시작으로 고에너지 물리, 핵물리 등의 분야를 지원하기 위한 대규모 계산처리 환경을 개발해왔다.

현재 연구/개발 중인 “다중 응용 및 사용자 지원 대규모 계산 작업 처리 기술(HTCaaS: High-Throughput Computing as a Service)”은 분산되어 있는 서로 다른 종류의 계산 자원을 하나의 통합된 형태로 연동해 연구자들에게 제공한다. 다양한 종류의 계산 자원에 대한 기술적인 세부사항을 추상화하여 연구자들이 하부의 인프라에 대해 신경 쓸 필요 없이 자신들이 실행해야 할 작업 프로세스에만 집중할 수 있게 함으로써 연구 개발 시간의 단축 및 연구 생산성 향상에 도움을 줄 수 있다. 또한, 국가 슈퍼컴퓨팅 공동 활용체제인 PLSI (Partnership & Leadership for the nationwide Supercomputing Infrastructure)[3-5]에 적용을 하여 국가 슈퍼컴퓨팅 인프라의 활용도를 높이는 데에도 크게 기여할 수 있다.

본 연구는 다양한 계산 과학 분야(신약개발, 고에너지 물리 등)의 연구자들이 그들의 연구에 필수적인 대규모 데이터 처리 또는 반복적인 연산 작업을 위해 분산되어 있는 이기종 컴퓨팅 자원들을 하나의 통합된 형태로 연동하여 용이하게 연구할 수 있는 서비스를 구축 및 제공하고자 한다. ([그림 1]은 이러한 HTCaaS의 개념도를 보여주고 있음)

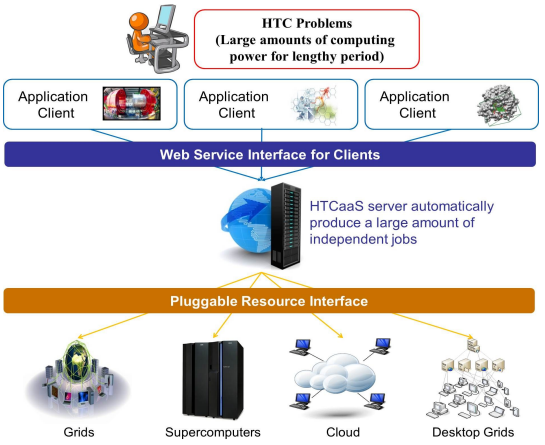

그림 1. HTCaaS 개념도

II. 관련 연구

1. HTCondor

HTCondor[6]는 Wisconsin 대학의 Condor 프로젝트를 통해 개발된 (분산) 작업 스케줄러로서 계산 중심 작업(Compute-intensive jobs)에 특화되어 개발된 워크로드 관리 시스템으로 시작하여 현재는 다양한 형태의 분산 작업 실행 관리를 지원하는 시스템으로 발전하였다. HTCondor에서는 작업 큐잉 메커니즘, 스케줄링 정책,

우선순위 구조, 자원 모니터링 및 자원 관리 등의 기능을 지원하며 이러한 HTCondor는 다수의 데몬(daemon)들로 구성되어 실행된다[7].

HTCondor는 가장 오래된 HTC 관련 프로젝트로서 많은 연구자들이 사용하고 있으나, 그래픽 사용자 도구와 같은 사용자의 편의를 도울 수 있는 기능이 미비해서 컴퓨터에 미숙한 연구자들이 사용하기에는 어려운 환경이다. 또한, HTCondor는 LoadLeveler, PBS, Sun Grid Engine과 같이 로컬 클러스터 자원 관리가 주목적인 로컬 배치 스케줄러(Local Batch Scheduler)에 가깝기 때문에 HTCaaS가 다른 배치 스케줄러와 같이 HTCondor를 하부 구조로서 통합할 수 있는 형태이기도 하다.

2. DIRAC (Distributed Infrastructure with Remote Agent Control)

DIRAC[8] 프로젝트는 분산되어 존재하는 고성능 컴퓨팅 자원을 필요로 하는 사용자 커뮤니티(예를 들어 LHCb Collaboration[9] 등)를 위한 그리드 솔루션으로서 특정 사용자 커뮤니티와 다양한 컴퓨팅 자원들 사이에 하나의 미들웨어 계층을 형성하여 다양한 자원에 대해 최적화되고 안정적인 활용을 가능하게 해준다.

특히 DIRAC에서는 HTCaaS와 같이 '파일럿 작업(Pilot Job)'이라는 개념을 사용하는 프레임워크 접근 방식을 중심으로 설계되었다. 파일럿 작업은 가용 컴퓨팅 자원에 보내져 중앙 WMS(Workload Management System) 큐 내의 적절한 작업을 실행시켜주는 역할을 하며 작업 할당을 제공받기 이전에 먼저 수행 환경을 검사하는데, 문제가 발생하는 경우 DIRAC WMS 프레임워크로 문제 사항을 알리게 된다. 이러한 구조는 사용자의 작업을 배치하기 이전에 수행되어 전체 부하에 영향을 미치지 않아 효율적으로 실행된다.

DIRAC은 연구자들이 변수를 바꿔가며 대규모 응용계산을 실행해볼 수 있게 하는 파라미터 스윕(Parameter Sweep)기능을 제공하지 않는 단점이 있다.

3. Falkon(A FAst and Light-weight tasK executiON framework)

Falkon(A FAst and Light-weight tasK executiON framework)[10]은 MTC(Many-Task Computing) 특성을 갖는 응용을 지원하기 위한 미들웨어이다. 보통 MTC[11]란, 서로 독립적으로 수행되는 작업들 간 파일을 통해 데이터를 주고받는 느슨하게 연결된 계산(loosely coupled computing) 모델로, 주로 백만 개 또는 그 이상의 대규모 작업들을 수행하거나, 상대적으로 작업 수행 시간이 짧은 작업(초, 분 단위의 작업) 또는 데이터 중심 작업(Data intensive tasks)이 이에 해당된다. 즉, 짧은 시간 내에 많은 양의 자원을 사용하는 작업들을 의미하는데, Falkon에서는 이러한 특성을 갖는 작업들의 성능을 극대화하기 위한 실행 모델을 반영한 구조를 갖는다. Falkon 역시 HTCaaS와 유사한 아키텍처를 가지고 있으나, HTCaaS는 Pull 모델에 기반한 작업 프로세싱(Falkon은 Push기반)과 사용자 편의성을 위한 다양한 클라이언트를 제공한다는 점에서 좀 더 실제 사용자를 위한 서비스에 가깝다고 할 수 있다.

4. AliEn (Alice Environment)

AliEn[13]은 다중 사용자를 지원하는 작업 실행 환경의 대표적인 예로, 유럽입자물리연구소(CERN)의 강입자 충돌 실험의 하나인 ALICE(A Large Ion Collider Experiment)[12]를 지원하기 위해 개발된 그리드 프레임워크이다. ALICE는 입자충돌을 통해 생성되는 신호처리 및 데이터 생성, 데이터 가공, 처리, 분석 등의 일련의 과정을 거치게 되는데 이 때 생성되는 방대한 양의 데이터를 전 세계에 흩어져 있는 연구자들에게 손실없이 효율적으로 처리하고 분석하게 하기 위해 설계되었다. AliEn은 2001년부터 현재까지 다수의 데이터챌린지를 통해 그 성능 및 안정성이 검증되었다. 2009년 11월, 최초로 가속기로부터 실제 데이터를 얻는데 성공했고, 2010년부터 양산되는 많은 양의 데이터를 처리하고 분석하고 있다.

그러나 ALICE 실험만을 위해 만들어진 HTC 환경이기 때문에 다른 응용을 지원하지 못하고, 클라우드 자원을 이용할 수 없는 단점을 가지고 있다. 2011년 이후로 개발이 중단된 상태이다.

5. DIANE (DIstributed ANalysis Environment)

스위스 CERN에서 개발한 DIANE[14]은 기존 그리드 미들웨어 위에서 동작하는 상위 수준 미들웨어로서, 하위 수준의 컴퓨팅 자원들과 최상위의 응용 프로그램을 연결하는 경량 프레임워크이다. DIANE은 분산된 마스터/워커 모델에 기반을 두어 응용 프로그램의 워크플로우를 관리한다. 관련된 기본적인 가정은 한 응용프로그램이 여러 개의 작은 독립적인 태스크로 쪼개질 수 있다는 것이다. 이 가정은 모든 응용프로그램에 적용될 수는 없지만 고에너지물리(High Energy Physics)나 바이오 인포매틱스(Bio-Informatics) 응용 등 많은 분야들에 있어 적합하다.

DIANE 역시 그리드 환경에서 만들어진 미들웨어라서 클라우드 자원이나 슈퍼컴퓨팅 자원으로의 확장성은 떨어지는 단점을 가지고 있다.

6. HPCaaS (HPC as a Service)

본 논문에서 설명하는 HTCaaS와 유사한 이름을 가지고 있는 HPCaaS[20]도 기본적으로는 슈퍼컴퓨팅 자원을 다중 사용자 및 다양한 과학 응용들의 지원에 사용하고자 하였다. 단일 응용의 성능을 최대한 끌어내기 위한 전용 HPC 시스템 구축보다는 이러한 시스템을 보다 다양한 응용에 동시 적용하면서 적절한 스케줄링을 통한 자원배분을 통해 전체적인 생산성 및 유연성을 증가시킬 수 있음을 보이고 있다.

그러나 3장에서 설명하듯이 HTCaaS는 주로 지원하는 응용들이 독립적이고 많은 수의 작업들로 구성이 되어 있으며, 이러한 응용들의 효과적인 지원을 위해서는 대규모 자원의 연동이 필수적이다. 이를 위해 파일럿 작업에 기반한 멀티레벨 스케줄링 (Multi-level Scheduling) 기법을 사용함으로써 단순히 주어진 클러스터 자원을 배분하기만 하는 HPCaaS에 비해 높은 유연성, 확장성 그리고 다양한 스케줄링 기법의 개발을 가능하게 하는 장점이 있다.

III. HTCaaS(High Throughput Computing as a Service)

HTCaaS는 다양한 자원의 효율적인 연동 및 스케줄링을 위해 파일럿 작업의 형태인 에이전트(Agent)에 기반한 멀티레벨 스케줄링(Multi-level Scheduling) 기법을 사용하고 있다 (로컬 배치스케줄러를 통한 에이전트 실행으로써 자원을 확보하고, 작업들은 별도의 큐를 통해 배포 및 실행함). 본 논문에서는 PLSI의 다중 클러스터 자원을 사용하기 위한 기술과 사용자의 편의성을 돕기 위해 개발된 그래픽 사용자 인터페이스(GUI : Graphic User Interface)와 커맨드 사용자 인터페이스(CLI : Command Line Interface) 등에 대해 소개한다.

1. HTCaaS 구조 및 작동 방식

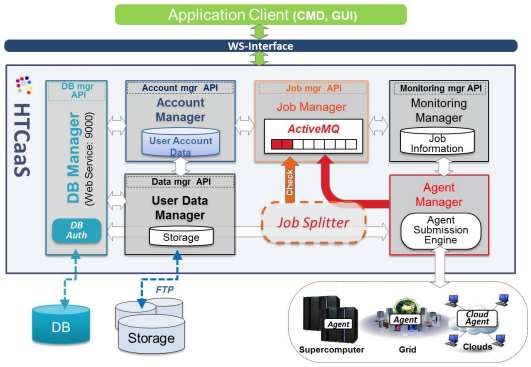

그림 2. HTCaaS 전체 시스템 구조도

HTCaaS의 구조 및 동작 방식은 다음과 같다. 사용자는 GUI 또는 CLI형태로 제공되는 HTCaaS 클라이언트를 통해 HTCaaS 서버에 접속하여 로그인(Account Manager를 통한 인증 작업 수행)을 한 후, 작업 제출을 위한 작업 생성 과정을 수행한다. 작업 생성 후, 작업 전송을 요청하면 먼저 사용자 코드와 데이터가 서버 측 저장소에 저장되며, 그 후에 메타 작업 메시지를 생성해 작업 큐에 전달 후, 관련 정보를 모니터링을 통해 데이터베이스에 저장한다. 메타 작업은 Job Manager에 의해 서브 작업으로 나눠져 작업 큐에 들어간다.

한편, Agent Manager는 주기적으로 등록되어 있는 하부 계산 자원들의 상태를 모니터링하고, 이 정보를 데이터베이스에 유지한다. 동시에 작업 큐에 사용자의 작업이 들어온 것을 주기적으로 확인하며, 그 때마다 적절

한 하부 계산 자원을 선택하여 에이전트를 실행시킨다 (하부 자원을 담당하는 로컬 배치 스케줄러 작업으로서 에이전트를 실행 - 각 배치 스케줄러에 특화된 작업 스크립트로 표현됨). 에이전트가 실행되면 먼저 작업 큐에서 서브작업 메시지를 가져오고, 이 정보를 바탕으로 저장소에서 사용자 코드와 데이터를 가져와 이를 설치하고 실행한다. 작업 실행이 완료되면 그 결과를 저장소에 저장하고, 관련 정보를 모니터링 서비스를 통해 데이터베이스에 반영한다. 에이전트는 작업을 처리하면서 주기적으로 자신의 상태와 작업 상태를 기록하게 되며, 이 정보는 Agent Manager 또는 Job Manager가 에이전트의 수행 에러 혹은 작업 실패의 경우 장애 복구를 하기 위한 메커니즘으로 활용된다.

HTCaaS에서는 작업과 에이전트에 대한 정보를 저장하기 위해 MySQL 데이터베이스를 사용하고 있으며, DB Manager라는 내부 모듈(다른 Manager들과 달리 Public API를 제공하지 않음)을 통해서 Web Service Interface로 제공하고 있다. 사용자는 전체 작업의 진행 상황을 클라이언트를 통해 확인할 수 있고, 완료된 작업에 대해서는 저장소로부터 그 결과를 로컬 컴퓨터로 가져올 수 있다.

[그림 2]는 HTCaaS의 전체 구조를 도식화한 것이며, [그림 3]은 앞서 설명한 과정을 포함한 전체 실행 시나리오를 나타낸다. (① Login ② User Input Data ③ MetaJob Submit ④ Agent Submit ⑤ Job Request & Dispatch ⑥ User Data request/distribute ⑦ Execution & Monitoring ⑧ User Output Data ⑨ Results)

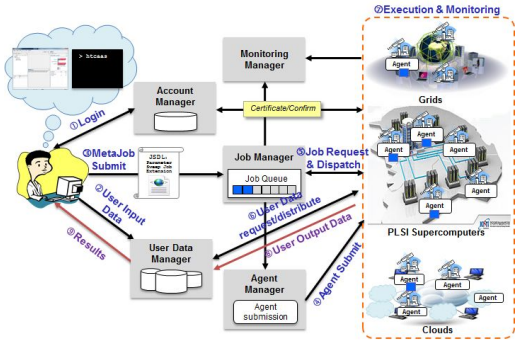

그림 3. HTCaaS 실행 시나리오

작업 제출을 위해서는 사용자의 등록 및 로그인 과정이 요구되며 이는 Account Manager를 통해 수행된다. 등록이 된 사용자에 한하여 인증서와 계정 유효성을 검사하게 되며, 사용자의 인증서가 유효함이 증명되면 작업 제출이 가능하다. 작업 제출 또는 실행 과정 중에는 HTCaaS의 여러 모듈에서 Account Manager로의 접근이 다수 요청되는데, 가령 User Data Manager 모듈을 통해 사용자의 데이터를 저장소로 옮기려 하는 경우에 Account Manager를 통해 사용자 계정 정보를 얻어 데이터를 업로드 또는 다운로드가 가능하며, 작업 제출 시에는 사용자에게 허용되는 컴퓨팅 자원 종류를 얻을 때에 Account Manager를 거쳐 그 정보를 전달 받을 수 있다.

이기종 자원을 지원하기 위해 HTCaaS는 자체 사용자에 대한 인증 및 사용하고자 하는 자원에 대한 인증이 필요하다. 본 논문에서 사용된 PLSI 환경에서는 1단계는 HTCaaS 사용자인지 검사하고, 2단계는 PLSI 사용자인지 검사를 하게 된다. [그림 4]는 HTCaaS에서 지원하는 인증 과정을 도식화한 것이다. 사용자 (HTCaaS Client 부분)는 사용자의 아이디와 패스워드, 그리고 (PLSI 사용자인 경우) PLSI 인증서를 입력하고 HTCaaS 서비스 내의 Account Manager를 통해 HTCaaS 사용자 리스트와 PLSI 인증이 모두 유효한지를 검사받는다. 이러한 다단계 인증 체계는 사용자에게는 보이지 않고 단일한 인터페이스로 표현되고, 결과적으로 인증의 유효성 검사 결과만이 전달된다.

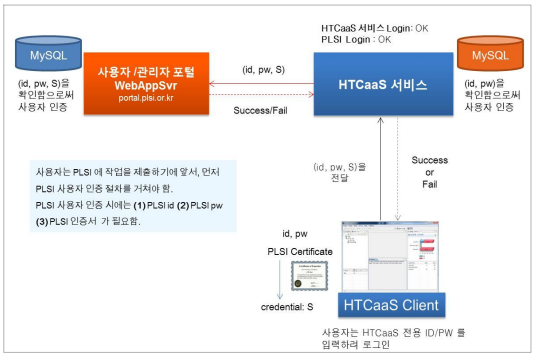

그림 4. HTCaaS 인증 구조

2. HTCaaS 구성요소

HTCaaS는 [그림 2]의 전체시스템 구조도에서 보는 바와 같이 Account Manager, User Data Manager, Job Manager, Monitoring Manager, Agent Manager, Agent 등으로 이루어져 있다.

2.1 Account Manager

Account Manager는 기본적으로 4가지 기능(로그인, 인증 관리, 사용자 관리, 사용자 정보 관리)을 한다. 로그인은 사용자의 아이디와 패스워드를 입력받아 인증 관리 기능을 통해 그 유효성을 검사 후, 검사 결과를 돌려준다. 인증 관리에서는 로그인 모듈에서의 사용자 식별 요청을 처리하고 클라이언트로부터 인증서 검사를 위해 전송되는 암호화된 메시지를 통해 PLSI 데이터베이스, 웹 서비스와의 통신을 수행하고 인증서 유효성 검사 결과를 돌려준다. HTCaaS 사용자 인증 검사를 위해 DB Manager에 사용자 ID 존재 여부와 암호 일치 여부를 검사 후 그 결과를 반환한다.

2.2 User Data Manager

User Data Manager는 사용자가 제출한 메타 작업에 요구되는 입력 데이터를 업로드하거나, 에이전트에 의해 처리가 끝난 결과 데이터들을 전송하는 역할을 담당한다. 이때 필요한 데이터들은 별도의 저장소에 저장되어 있으며 에이전트나 사용자가 데이터에 대해 요청하면 파일 전송 프로토콜을 통해 필요한 곳으로 데이터를 보낸다. 또한 파일 및 디렉터리 관리는 별도의 사용자 인증 과정을 거쳐 각 사용자 별로 독립적으로 이뤄진다.

2.3 Job Manager

Job Manager는 크게 작업 제출, 상태 검사, 작업 취소, 작업 결과 확인 등의 4가지 기능을 수행한다. 작업 제출은 사용자가 메타 작업을 제출 시, 사용자 아이디와 작업 스크립트 파일이 전달되고 작업이 서브 작업으로 나뉘어 작업 큐에 들어가면 그 작업 정보들이 DB Manager를 통해 HTCaaS 서버의 데이터베이스에 저장된다. 각각의 메타 작업들은 메타 작업 ID에 의해 구분되는데, 이를 통해 작업 상태 검사 모듈에서 작업 상태를 확인한다. 작업 상태 검사 모듈에서는 작업 상태 검사 요청을 받으면 작업 ID(Meta & Sub Job ID) 정보를 Monitoring 모듈에 요청하여 실제 작업 수행 상태에 대한 정보를 제공받는다. 사용자(Client)가 작업을 취소하려는 경우에는 취소 모듈에 작업 정보를 전달하여 취소 요청을 하며 모니터링 모듈에 작업 플래그를 변경함으로써 작업 취소를 유도한다. 최종적으로 완료된 메타 작업의 결과는 작업 결과 모듈을 통해 작업 결과 정보를 제공받고, 결과 데이터를 User Data Manager에 요청하여 결과 파일을 사용자에게 전달한다.

2.4 Monitoring Manager

Monitoring Manager는 기본적으로 에이전트 모니터링과 작업 모니터링으로 구성된다. 에이전트 모니터링의 경우, 현재 동적으로 실행 중인 에이전트들을 모니터링 하여 자원 확보에 대한 장애 복구 또는 동적 자원 할당에 필요한 정보로 활용된다. 작업 모니터링은 현재 에이전트들이 실행 중인 작업들에 대한 정보를 메타 작업과 이를 구성하는 단위 작업들로 관리하게 되며, 이는 작업 실패 복구 및 사용자에게 작업 수행 상태, 진행 경과 및 결과를 알려주는데 활용된다.

2.5 Agent Manager

Agent Manager는 백엔드 자원 초기화, 입력 큐 검사, 에이전트 워커 실행, 에이전트와 CE 정책, 자원 선택, 에이전트 워커 제출, 에이전트 상태 검사 관리 도구로 구성된다. 백엔드 자원 초기화에서는 관리자에 의해 요청된 자원 초기화(Init) 요청에 따라 수행되며, 자원의 정보를 제공 받은 후 DB Manager에게 그 정보를 전달한다.

2.6 Agent

Agent 모듈은 작업 요청, 입출력 파일 요청, 응용설치, 응용 실행, 출력 파일 검증, 에이전트 시작/정지로 총 6단계로 크게 나누어진다. 먼저 해당하는 사용자 큐로부터 작업을 확인하여, 큐에 작업이 존재하면 작업에 관련된 명세가 서술되어 있는 서브작업메시지를 가져온다. 이 작업 메시지에는 실행파일 및 입력 파일의 위

치, 출력 파일의 위치, 파라미터 스윕의 방법 등이 명시되어 있다. 작업 메시지에 명시되어 있는 입력파일들을 User Data Manager로부터 가져온 후, 해당하는 응용 프로그램의 설치 및 응용 실행을 시작한다. 작업이 끝나면 최종 출력 결과 파일이 유효한지 검사한 후 유효하다면 검증을 마친 후 다음 작업의 실행을 준비한다. 이때 중간 작업 및 에이전트의 상태가 변할 때 마다 DB Manager를 통해 작업 및 에이전트정보를 기록하며, 일정한 주기마다 에이전트 신호를 보냄으로써 에이전트 및 작업의 실행 유무를 판단한다.

3. HTCaaS 클라이언트

HTCaaS는 다수의 작업을 제출하고 또 관리하는데 용이하도록 간단하고 편리한 사용자 인터페이스를 제공한다. 이를 위해 HTCaaS에서는 두 가지 방식의 클라이언트를 제공한다. 하나는 텍스트 방식의 사용자 인터페이스인 CLI(Command Line Interface)이고, 다른 하나는 eclipse의 RCP(Rich Client Platform)를 기반으로 한 그래픽 사용자 인터페이스(GUI)이다.

3.1 HTCaaS CLI 클라이언트

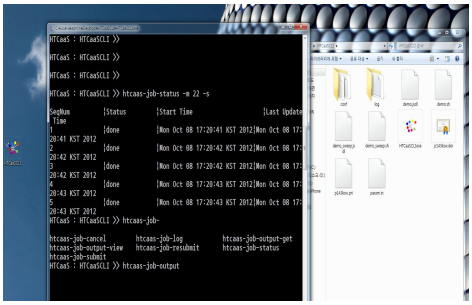

그림 5. CLI 실행 화면

HTCaaS CLI(Command Line Interface)[그림 5]는 명령어 기반 인터페이스에 익숙한 과학 전문가들을 위한 클라이언트로, 작업 준비 및 제출 등의 인터페이스를 HTCaaS에서 정의된 간단한 명령어를 통해 이용할 수 있다. 다수의 작업을 제출하고 관리하기 위해 미리 정의된 명령어를 통해 용이하게 수행 가능하며 또한 리눅스의 기본 명령어들을 사용할 수 있어 명령어 기반 인터페이스에 친숙한 사용자의 경우 더욱 편리하게 활용 가능하다. HTCaaS CLI는 Java로 개발되어서 윈도우즈, 리눅스, 맥 OS 모두에서 사용 가능하도록 지원되며, 다양한 CLI 명령어[표 1]를 제공하고 있다.

표 1. HTCaaS CLI 명령어

명령	설명
htcaas-job-submit	작업 제출
htcaas-job-status	작업 상태 보기
htcaas-job-output-view	작업의 결과 파일 리스트를 보기
htcaas-job-output-get	작업의 결과 파일들을 다운로드
htcaas-job-log	작업의 로그 보기
htcaas-file-list	원격 서버(ftp서버)에 (특정 위치의) 파일 리스트 보기
htcaas-file-put	로컬에 존재하는 특정 파일을 원격 서버(ftp)로 업로드
htcaas-file-get	원격 서버(ftp)에 위치한 특정 파일을 로컬로 다운로드

3.2 HTCaaS GUI 클라이언트

HTCaaS는 다수의 작업을 제출하고 또 관리하는데 용이하도록 간단하고 편리한 사용자 인터페이스를 제공한다. 이를 위해 그래픽 사용자 인터페이스를 제공한다.

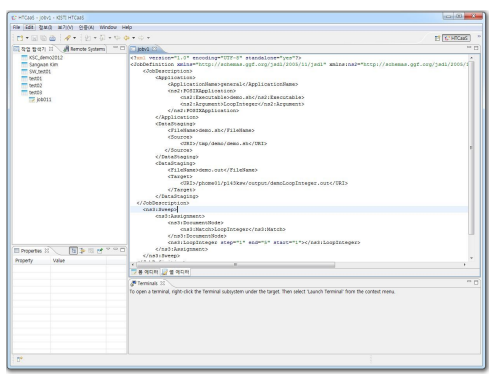

그림 6. HTCaaS GUI 메인 화면

[그림 6]은 로그인에 성공하면 제공되는 기본 사용자 인터페이스 (초기) 화면으로 메인 화면에 해당한다. 메인 화면은 크게 3개의 부분으로 기능에 따라 나눌 수 있다. 왼쪽 부분은 프로젝트 목록으로, 사용자가 생성한

모든 프로젝트를 보여준다. 오른쪽 윗부분은 사용자가 제출할 작업을 기술하고 제출하는 부분이고, 나머지 한 부분(오른쪽 아래 또는 가장 오른쪽에 위치하게 됨)은 제출한 작업 상태를 모니터링 하거나 클러스터 상태 또는 명령어에 대한 정보 리스트를 제공한다.

4. 응용적용사례

대규모 계산을 요구하는 용용 가운데 신약개발 분야의 Autodock3, 고에너지 물리 가속기 시뮬레이션 분야의 Madgraph5+PYTHIA를 이용하여 실험을 해보았다.

4.1 신약개발 분야

가상 스크리닝은 신약개발을 위한 첫 단계로 주어진 질병의 타겟 단백질에 대해서 화합물 도킹을 이용한 계산을 통해 신약후보물질을 찾아낸다. 도킹 알고리즘 중에 Autodock3[15]를 활용한 계산을 이용하면 하나의 도킹을 계산하는데 평균적으로 20분 정도가 소요되고, 30만개의 화합물을 대상으로 하면 39년 정도의 시간의 소요된다.

사용되어지는 데이터는 단백질, 화합물 등이 있는데, 단백질은 PDB 파일로 평균적으로 400KB 크기를 가지고, 화합물 역시 PDB 파일로 평균적으로 2~3KB 크기이며, 30만개를 기준으로 1GB 정도의 저장 공간이 필요하다. 결과 파일은 DLG 파일로 평균적으로 400KB 크기이며 실험 시나리오는 다음과 같다.

1. 단백질 PDB에서 단백질을 구성하는 아미노산 및 금속 기둥의 단백질 구성 성분을 제외한 다른 분자들을 제거
2. 단백질 PDB, 화합물 PDB를 PDBQT로 변환
3. AutoGrid를 이용해, 에너지 값 계산을 위한 그리드와 그리드 파라미터 값을 계산하기 위한 GPF 파일 생성
4. AutoGrid를 수행하여 화합물 주변에 그리드를 생성
5. Autodock3를 계산하기 위한 DPF 파일을 생성
6. DPF 파일을 이용하여 Autodock3 계산을 수행
7. 결과 파일인 DLG 파일 획득

이러한 과정을 거쳐 SARS (중증 급성 호흡기 증후군)에 대한 대규모 신약후보물질탐색을 수행하였으며, 그리드 자원을 활용하여 110만개의 화합물을 대상으로 평균 2500개의 CPU를 동시 활용하여 실험을 하였고, 총 42년의 계산을 11일 만에 완료하였다[16][17].

PLSI 자원을 활용한 경우는 50만개의 화합물을 대상으로 평균 1000개의 CPU를 활용하여 약 2주 만에 작업을 끝낼 수 있었다.

4.2 고에너지물리 (High Energy Physics)

고에너지 물리 분야에서는 고에너지 가속기 실험을 통해 얻어지는 데이터의 예측 및 검증을 위한 대용량 Monte Carlo (MC) 시뮬레이션 데이터를 분산 처리하여 HTCaaS에 적용하였다. 고에너지 입자 충돌 프로세스가 가속기 내 검출기에서 관측되는 현상을 시뮬레이션 하는 프로그램인 MadGraph5[18], PYTHIA[19]를 조합하여 실행 후 결과를 얻을 수 있었다.

응용의 실험에 사용된 데이터는 출력파일이 파라미터 스윕 범위에 따라 분할 개수가 결정되며, 분할 개수가 500인 경우 결과 파일 하나의 크기는 약 72.4MB정도가 되었다. 이러한 연구는 고에너지 물리 분야뿐만이 아니라 MadGraph5, PYTHIA와 같은 프로그램을 이용하는 천체물리 등 유사 분야에 적용 가능할 수 있어서 향후 새로운 응용 적용에 가이드라인을 제시할 수 있었다.

PLSI 자원을 활용하여 약 1천만 개의 이벤트 시뮬레이션을 평균 500개의 CPU를 활용하여 2시간 만에 끝낼 수 있었다 (3.5GHz 쿼드코어 머신에서 약 8일 소요).

4.3 핵물리 (Nuclear Physics)

핵물리 분야는 가속기 물리의 다체계산 (N-body Phase Space Calculation)에 적용하여 실험을 하고 있는 상황이다. 현재 PLSI에서 HTCaaS를 이용해 몇 차원 위상공간 계산까지 가능한지 테스트함으로써 실제 핵입자 물리 이론 연구에 도움을 줄 수 있도록 하고자 한다.

IV. 결론 및 향후 과제

기존의 그리드 기반의 HTCaaS 시스템을 확장한

PLSI기반의 HTCaaS 시범 서비스를 제공하여 신약개발, 인공위성 구조 최적설계, 고에너지 물리, 의학물리 등의 국내외 커뮤니티 활용을 지원하였다. 향후 더 많은 수의 국내 대학 개발 연구팀에게 국가 슈퍼컴퓨팅 인프라를 포함하는 대용량의 계산 자원을 활용할 수 있도록 개발할 예정이다.

HTCaaS는 분산되어 서로 다른 특징을 갖는 인프라 구조로 구성된 계산 자원들을 하나의 통합된 형태로 연구자들에게 제공한다. 즉, 본 연구를 통해 다양한 계산과학 분야의 응용 연구자들은 분산되어 존재하는 고성능 컴퓨팅 자원으로 쉽게 접근이 가능하며, 컴퓨팅 자원으로의 계산 작업 구성 및 제출의 자동화를 통해 용이하게 연구 수행을 할 수 있다. 이로써 국내 계산과학 분야의 연구 생산성을 높이고, 국가 슈퍼컴퓨팅 인프라의 활용도를 높이는 데에도 크게 기여할 수 있다.

슈퍼컴퓨터 인프라 기반의 HTCaaS 시범 서비스를 구축하고, 연구자가 쉽게 사용할 수 있게 HTCaaS GUI 및 CLI 클라이언트를 제공하여 서비스에 대한 접근성을 높임으로써 국가 슈퍼컴퓨팅 자원을 효율적으로 사용하고, 다양한 응용 분야에 활용될 것으로 기대된다.

참 고 문 헌

[1] http://www.egi.eu/

[2] http://boinc.berkeley.edu/

[3] http://www.plsi.or.kr

[4] 김성준, 성진우, 장지훈, 이상동, “국가 슈퍼컴퓨팅 공동활용 환경을 위한 통합 모니터링 환경 구축”, 한국콘텐츠학회 추계 종합학술대회 논문집, 제5권, 제2호(하), pp.517-521, 2007.

[5] 우준, 박석중, 이상동, 김형식, “국가 슈퍼컴퓨팅 공동활용체제 구축을 위한 글로벌공유파일시스템 성능 분석”, 한국콘텐츠학회 추계 종합학술대회 논문집, 제5권, 제2호(하), pp509-512, 2007.

[6] http://research.cs.wisc.edu/htcondor/

[7] D. Thain, T. Tannenbaum, and M. Livny, "Distributed Computing in Practice: The Condor Experience," Concurrency and Computation: Practice and Experience, Vol.17, Issue 2-4, pp.323-356, 2005.

[8] A. Casajus et al, "DIRAC Pilot Framework and the DIRAC Workload Management System," Journal of Physics: Conference Series, Vol.219, No.6, p.062049, 2010.

[9] http://lhcb.web.cern.ch/lhcb/

[10] I. Raicu, Y. Zhao, C. Dumitrescu, I. Foster, and Mike Wilde, "Falkon: a Fast and Light-weight tasK executiON framework," ACM/IEEE conference on Supercomputing (SC'07), 2007.

[11] I. Raicu, I. Foster, and Y. Zhao, "Many-Task Computing for Grids and Supercomputers," ACM Workshop on Many-Task Computing on Grids and Supercomputers (MTAGS'08), 2008.

[12] www.cern.ch/alice/

[13] http://alien2.cern.ch/

[14] http://cern.ch/DIANE

[15] http://autodock.scripps.edu

[16] T. T. Nguyen, H. J. Ryu, S. H. Lee, S. Hwang, V. Breton, J. H. Rhee, and D. Kim, "Virtual screening identification of novel severe acute respiratory syndrome 3C-like protease inhibitors and in vitro confirmation," Bioorganic & Medicinal Chemistry Letters, Vol.21, No.10, pp.3088-3091, 2011.

[17] T. T. Nguyen, H. J. Ryu, S. H. Lee, S. Hwang, J. Cha, V. Breton, and D. Kim, "Discovery of novel inhibitors for human intestinal maltase: virtual screening in a WISDOM environment and in vitro evaluation," Biotechnology Letters, Vol.33, No.11, pp.2185-2191, 2001.

[18] http://madgraph.hep.uiuc.edu/

[19] http://pythia6.hepforge.org/

[20] G. Shainer, T. Liu, J. Layton, and J. Mora, “Scheduling Strategies for HPC as a Service (HPCaaS),” IEEE Cluster Computing and Workshops, 2009.

저 자 소 개

김 석 규(Seok-Kyoo Kim) 정회원

- 1992년 2월 : 서울대학교 계산통계학과(이학사)
- 1994년 2월 : 서울대학교 계산통계학과(이학석사)
- 1994년 ~ 1999년 : 현대전자, 현대정보기술 근무
- 1999년 ~ 2003년 : 엔씨소프트 팀장
- 2010년 2월 : 서울대학교 컴퓨터공학부(공학박사)
- 2011년 1월 ~ 2014년 2월 : 한국과학기술정보연구원 선임연구원
- 2014년 3월 ~ 현재 : 상명대학교 게임학과 조교수

<관심분야> : 그리드 컴퓨팅, 분산 컴퓨팅, 병렬처리, 디지털 스토리텔링, 컴퓨터 게임, HCI, 증강현실

김 직 수(Jik-Soo Kim) 정회원

- 1997년 2월 : 서울대학교 계산통계학과(이학사)
- 1999년 2월 : 서울대학교 계산통계학과(이학석사)
- 1999년 7월 ~ 2002년 6월 : 육군사관학교 전산학과 교수사관
- 2009년 5월 : 미국 메릴랜드 주립대학교 전산학과(이학박사)
- 2009년 9월 ~ 2012년 5월 : 삼성 SDS CSP 연구소 수석연구원
- 2012년 6월 ~ 현재 : 한국과학기술정보연구원 선임연구원

<관심분야> : 분산 컴퓨팅, 그리드/클라우드 컴퓨팅

김 상 완(Sangwan Kim) 정회원

- 1999년 2월 : 포항공과대학교 전자전기공학과(공학사)
- 2001년 2월 : 포항공과대학교 컴퓨터공학과(공학석사)
- 2001년 2월 ~ 현재 : 한국과학기술정보연구원 슈퍼컴퓨팅센터 (선임연구원)

<관심분야> : 분산 컴퓨팅, 클라우드 컴퓨팅

노 승 우(Seungwoo Rho) 정회원

- 2009년 2월 : 서울시립대학교 전자전기컴퓨터공학부(공학사)
- 2011년 2월 : 서울시립대학교 전자전기컴퓨터공학부(공학석사)
- 2011년 5월 ~ 현재 : 한국과학기술정보연구원 연구원

<관심분야> : 클라우드 컴퓨팅, 분산 컴퓨팅, 그리드 컴퓨팅, 유시티 등

김 서 영(Seoyoung Kim) 정회원

- 2010년 2월 : 숙명여자대학교 컴퓨터과학과(이학사)
- 2012년 2월 : 숙명여자대학교 대학원 컴퓨터과학부(이학석사)
- 2012년 ~ 현재 : 한국과학기술정보연구원 연구원

<관심분야> : 분산 컴퓨팅, 클라우드 컴퓨팅, 메타 스케줄링 등

황 순 욱(Soonwook Hwang) 정회원

- 1990년 : 서울대학교 수학과(이학사)
- 1995년 : 서울대학교 계산통계학과(이학석사)
- 2003년 : 미국 남가주대학교 전산과학과(이학박사)
- 2003년 ~ 2006년 : 일본 국립정보학연구소 연구원
- 2006년 ~ 현재 : 한국과학기술정보연구원 책임연구원

<관심분야> : 그리드 컴퓨팅, 클라우드 컴퓨팅, 분산 컴퓨팅 등