韓國 컴퓨터情報學會 論文誌
第10卷 第4號, 2005. 9.

2005-10-4-1-18

계산 그리드에서 워크플로우 기반의 사용자 환경 설계 및 구현

황 선 태*, 심 규 호**

Design and Implementation of Workflow-based User Environment on Computational Grid

Sun-Tae Hwang *, Gyu-Ho Sim**

요 약

고속의 컴퓨터, 대용량 저장장치, 초고속 네트워크는 현재 우리가 쉽게 접근할 수 있는 컴퓨팅 인프라이다. 하지만 분자 시뮬레이션과 같은 자연과학 및 응용과학 분야에의 시뮬레이션에서는 여전히 더 많은 컴퓨팅 파워, 더 커다란 저장장치를 필요로 한다. 이러한 요구는 그리드 컴퓨팅[1]이라는 차세대 분산 컴퓨팅 환경을 우리에게 제시하였다. 하지만 현재까지 제안된 그리드 컴퓨팅 기술은 통신 인터페이스와 프로토콜 등의 글로버스 툴킷[2,3]과 같은 미들웨어 수준에 대한 연구만이 중심이 되고 있다. 이러한 환경은 응용 플랫폼에 대한 연구의 부족과 어플리케이션의 부족을 가져왔으며, 그 결과 사용자는 그리드 컴퓨팅 기술에 대한 이용을 미비하게 만들었다. 따라서 본 연구에서는 분자 시뮬레이션 그리드 (MGrid: Molecular Simulation Grid System) 에서 적용을 목적으로 고효율(High Throughput)의 시뮬레이션 실험을 위한 사용자 환경(User Environment)을 정의하고, 사용자에게 친근한 추상화된 작업 모델을 제안함으로써 보다 효율적이고 안정적인 그리드 자원 이용을 가능하게 한다.[4,5]

Abstract

High speed computer, large scale storage device and high speed computer network are computing infrastructure which we can easily access to in these days. However, many computer simulations in natural or applied science such as molecular simulation require more computing power as well as larger scale of storage. Grid computing which is a next generation of distributed computing environment, is one of solution for the new requirements. Even though many researches have been going on Grid computing, those are oriented to communication interface and protocols, and middleware like globus tool kits[2,3]. Therefore research on application level platform or application itself is yet premature and it makes real users be difficult to utilize Grid system for their research. In this paper, we suggest a new user environment and an abstract job model for simulation experiments on MGrid(Molecular Simulation Grid). It will make users be able to utilize Grid resources efficiently and reliably.

▸ Keyword : Workflow, Computational Grid, Job Model, Molecular Simulation

---

• 제1저자 : 황선태
• 접수일 : 2005.07.16, 심사완료일 : 2005.09.05
* 국민대학교 컴퓨터학부 부교수, ** 한국전자통신연구원 연구원

Ⅰ. 서 론

최근BT, NT, ET등 다양한 분야에서 컴퓨터를 이용한 연구를 하고 있다. 이러한 분야의 연구는 직접 실험의 어려움과 많은 시간을 필요로 하는 특징을 가진다. 실험자는 직접 실험의 어려움과 실험 수행 시간을 단축하기 위해 컴퓨터의 빠른 계산 속도를 이용하여 시뮬레이션을 통해 실험을 수행하게 된다. 하지만 현재 컴퓨터의 빠른 계산 속도에 불구하고 시뮬레이션 실험은 더 빠른 계산 장치를 요구하고 있다.

그리드 컴퓨팅 기술(Grid Computing Technology)은 많은 계산 자원과 커다란 저장 장치를 요구하는 자연 과학 및 응용 과학 분야의 시뮬레이션에서 커다란 혁명으로 다가왔다. 초고속 인터넷으로 연결된 고성능의 컴퓨터들은 가상의 자원 풀(Resource Pool)을 구성하고, 사용자는 하나의 커다란 슈퍼 컴퓨터를 사용하듯 가상화된 계산 및 저장 자원을 투명하고 신뢰성 있게 접근 할 수 있게 되었다. 이는 지리적으로 분산되어 있는 휴지상태의 자원을 많은 계산 자원을 필요로 하는 연구자에게 제공함으로써 고효율 실험을 실험 자에 제공할 수 있다. 하지만 초고속 네트워크, 클러스터(Cluster) 기술과 가상화된 자원에 접근하기 위한 저 수준의 프로토콜 및 인터페이스 등은 거의 완성 단계에 머무르는 반면, 시뮬레이션 실험을 위한 사용자 환경과 사용자의 작업 흐름을 정의할 수 있는 환경에 대한 연구는 미비한 것이 현실이다. 따라서 연구에서는 사용자가 자신의 작업 흐름을 정의하기 쉬운 환경과 고효율(High Throughput) 실험을 지원할 수 있는 작업 모델을 제안하고 있다.

본 논문의 구성은 다음과 같이 구성된다. 2장에서 글로벌 그리드 포럼(Global Grid Forum)[6]에서 제안한 작업 모델에 관한 관련연구를 담고 있으며, 3장에서는 시뮬레이션 실험 분석을 통해 정의한 워크플로우 모델과 고효율 실험을 위한 작업 모델을 제안한다. 4장에서는 이러한 작업 모델을 기반으로 구현한 작업 정의환경과 작업 제어환경 구현에 대해 설명하며, 마지막으로 5장에서는 결론 및 향후 과제를 제시한다.

Ⅱ. 관련 연구

2.1 Grid Job

글로벌 그리드 포럼의 그리드 컴퓨팅 환경 리서치 그룹에서는 GGF3, GGF4, GGF5에서 라이프 사이클(life-cycle)을 가지며, Job Descriptor, Application Descriptor, Application Metadata등의 메타데이터로 구성된 Grid Job을 소개하였다.[7]

Grid Job에서는 작업을 제출할 때 필요로 하는 자원 스케줄러(Scheduler)에 관련된 모든 정보를 제공한다. 이러한 정보들은 RSL 스크립트나 PBS[8] 배치 스크립트로 변경이 가능하며, 사용자에게 지역 투명성(location transparency)을 제공하기도 한다. 또한 Grid Job을 발행(publish)을 통하여 다른 사용자와의 공유가 가능하며, 새로운 코드를 생성하는데 비용을 줄일 수 있는 가능성을 가지고 있다. 마지막으로 Job에 대한 요구에 대한 QoS (Quality of Service)를 명시하여 자원 중계자(Resource Broker)를 통해 적절한 자원으로 분산을 가능하게 한다.

Grid Job의 개념은 다양한 그리드 서비스에 대해 단순화하고 있으며, 여러 단계의 애플리케이션에 대한 워크플로우(Workflow)를 명시 할 수 있다. 워크플로우에 대한 명시를 통해 사용자는 각 작업 노드간의 기능보다는 작업 노드간의 관계에 대해 보다 관심을 가지며 작업을 정의할 수 있으며, 지역 투명성을 제공함으로써 효과적인 자원 사용을 가능하게 한다.

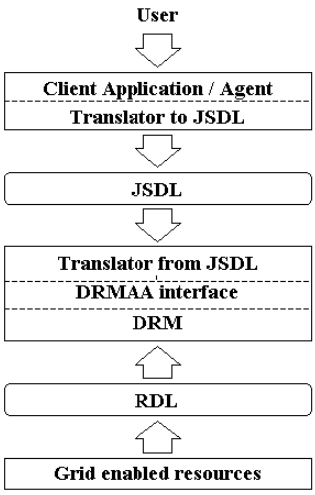

그림 1. JSDL 번역 흐름
Fig. 1 JSDL Interpret flow diagram

2.2 Job Submission Description Language

GGF의 JSDL 워킹 그룹에서 정의하고 있는 JSDL(Job Submission Description Language)은 프로그래밍 언어에 상관없이 독립적인 작업 제출을 위한 메타 언어의 표준 스펙을 정의하고 있다[9]. JSDL은 XML 스키마를 통해 데이터 타입을 정의하여 특정 언어 및 플랫폼에 독립적인 표준안을 제시하고 있다. JSDL에서는 계산 자원의 요구사항 이외에 계산시 필요로 하는 입/출력 파일에 대한 정보 및 작업간의 종속 관계 등의 부 과적인 정보를 포함하고 있다.

JSDL의 목적은 그리드 환경에서의 배치 시스템간의 상호 호환성을 제공하기 위함이며, XML로 정의된 JSDL은 (그림 1)과 같이 이질적인 계산 그리드 환경에서의 배치시스템으로 제출되며 각 배치 시스템은 자신이 해석할 수 있는 작업 구조로 JSDL을 번역하여 작업을 수행하게 된다. JSDL로 정의된 작업들은 각 배치 시스템에서 필요로 하는 정보를 추상화하고 있으며, 클라이언트 혹은 그리드 포탈 등을 통해 사용자는 각 작업들에 대해 접근할 수 있다. JDSL은 그리드 환경뿐만 아니라 다른 기타 배치 시스템에 제출 가능하며, 계산 작업에 대한 요구사항에 대한 언어로써 필요한 기능을 표현한 어휘들을 포함하고 있다.

2.3 WfMC 레퍼런스 모델

표준 워크플로우 구조를 기반으로 일반적인 워크플로우 시스템의 구조를 정의해 보면 워크플로우 시스템은 프로세스 정의구조, 워크플로우 엔진, 워크리스트 핸들러 및 UI, 애플리케이션의 4가지 핵심 구성 요소와 이들의 기능을 지원하는 데이터 영역으로 정의된다.[10][11] WfMC 워크플로우 서비스를 5개의 기능적 인터페이스로 구분하여 보면 다음과 같다.

첫째, 정의된 프로세스의 상호 교환을 위한 인터페이스

둘째, 워크플로우 클라이언트 애플리케이션 인터페이스

셋째, 워크플로우 Invoked 애플리케이션 인터페이스

넷째, 워크플로우 엔진간의 상호 연동을 위한 인터페이스

다섯째, 수행 결과 내역의 감시 및 통계 처리를 위한 인터페이스

WfMC에서는 이들 5가지 인터페이스에 대해 WfMC 레퍼런스 모델[12]을 정의하고 있으며 각각의 인터페이스를 구성하는 API들도 정의하고 있다.

III. 워크플로우(Workflow) 모델

워크플로우는 문서, 정보 혹은 작업 프로세스에 대한 전체 혹은 일부를 미리 정의된 규칙을 통한 자동화로 정의한다.[13] 일반적으로 워크플로우 정의에서는 문서나 데이터보다는 작업 프로세스의 자동화를 중심으로 표현한다. 하지만 시뮬레이션과 같이 데이터 중심으로 대용량의 데이터를 생산하는 작업에서는 작업의 흐름보다는 얻고자 하는 데이터에 초점을 맞출 필요가 있다.

3.1 데이터 의존성

시뮬레이션 실험에서는 파일 혹은 데이터를 중심으로 실험 프로세스가 진행된다. 다시 말하면, 앞선 시뮬레이션 실험의 결과 데이터는 뒤에 발생할 시뮬레이션 실험의 입력 데이터로 사용되며, 최종 결과 데이터는 시뮬레이션 실험의 최종 결과물로써 실험자는 실험 결과 데이터 관리와 분석을 필요로 하게 된다.

이러한 데이터간의 흐름으로부터 각 시뮬레이션 실험의 각각의 입력 파일은 앞선 시뮬레이션 실험의 결과 데이터

또는 이미 존재하는 파일과의 관련 정보를 표현함으로, 데이터 중심으로 시뮬레이션 실험을 정의할 수 있다. 이러한 데이터간의 의존성 정보를 통해 각 시뮬레이션 프로세스는 자신의 수행해야 할 시기를 자신의 입력 파일과 의존성이 있는 파일의 존재 여부로써 알 수 있으며, 이러한 정보를 통해 각 시뮬레이션 프로세스의 실행순서의 정의가 가능하다.

3.2 실행 프로세스 실행 순서

앞 절에서 설명한 것과 같이 데이터 의존성 정보를 통해 실행 프로세스의 실행 순서를 결정할 수 있다. 하지만 시뮬레이션 실험 도중 실험자의 컨트롤에 의해 비동기적으로 실행할 시뮬레이션 프로세스가 존재할 경우, 이는 데이터 의존성 정보만으로 실행 순서를 결정되는 것은 실험 자에 의해 의도된 상황이 아니다. 이를 위해 본 연구에서는 데이터 의존성 정의와 더불어 실험 프로세스의 실행 순서를 DAG(Directed Acyclic Graph) 형태로 정의한다. DAG 형태로 표현된 실험 프로세스는 자동화 프로세스를 정의하며, DAG에 포함되지 않는 실험 프로세스는 사용자에 의해 제어되는 비동기적 실행 흐름으로 정의하게 된다.

3.3 사용자 워크플로우 정의

본 연구에서는 데이터의 흐름과 실행 프로세스의 흐름을 중심으로 하여 워크플로우 모델을 정의한다. 여기서 제시한 워크플로우 모델은 데이터의 의존성 정보와 실행흐름을 서로 다른 공간으로 분리한다. 먼저 데이터 의존성에 대한 정의는 데이터 흐름공간(data flow space)로 정의한다. 각 Task의 입력 파일은 Repository내의 파일 혹은 다른 Task의 결과 파일과 관련되어 있다. 이에 각 Task의 입력파일을 Repository 혹은 다른 Task의 결과 파일과의 의존성 정보를 정의함으로써 데이터의 흐름을 정의할 수 있다. 이러한 어플리케이션 수준의 모니터링과 같은 비 동기 작업을 제외하면 데이터 의존성 정보를 통해 Task의 실행 순서 정의가 가능하다. 다른 하나의 Task의 실행 순서 정의로써 제어 공간(Control space)로 정의한다. 각 Task의 실행 순서를 DAG(Directed Acyclic Graph) 형태로 표현한다. 어플리케이션 수준의 모니터링과 같은 비 동기 Task에 대해서는 실행 순서를 정의하지 않고, 단지 데이터의 의존성 정보만이 존재한다.(그림 2)

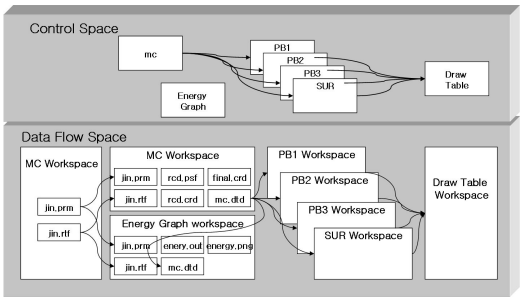

그림 2. 분리된 데이터 및 제어 흐름 정의
Fig. 2 Define separated data and control flow

3.4 액티브 파일 시스템(Active File System)

Ⅲ-3 절에서 설명한 데이터 의존성 정보와 각 실험 프로세스의 실행 순서 정의에 대해 본 논문에서는 (그림 3)와 같이 액티브 파일 시스템(Active File System)이라는 개념 모델로써 정립한다. 액티브 파일 시스템에서는 사용자가 관심을 가지는 데이터의 표현을 액티브 파일(Active File)로 정의하고 있다. 액티브 파일은 일반 파일 시스템에서의 파일 속성과 파일이 존재하지 않음, 파일이 생성중임, 파일 생성완료와 같은 파일의 동적인 상태를 표현하고 있다. 두 번째 액티브 파일 시스템에서는 이러한 액티브 파일들의 논리적 그룹으로서 프로덕트 리스트 (Product List) 를 정의한다. 프로덕트 리스트는 동일한 생성자 (Creator) 에 의해 만들어진 액티브 파일들의 집합으로써 일반 파일 시스템의 디렉토리와 유사한 개념이나 트리 형태를 취하지 아니하며, 각 프로덕트 리스트간에는 수평적인 관계를 가지고 있다. 마지막 구성요소로써 생성자 (Creator) 를 정의한다. 생성자는 액티브 파일을 생성하는 주체로써 기본 골격으로 입출력 파일의 정보와 실행 스크립트 그리고 실행순서에 관련한 정보를 포함하고 있다. 입출력 파일은 생성자에 포함된 실행 스크립트내의 입출력 파일과 관련되어 있다. 입력 파일에서는 액티브 파일과의 데이터 의존성 정보를 표현하고 있으며, 출력파일은 하나의 액티브 파일과의 관계 정보를 표현하고 있다.[14][15]

3.5 고효율(High Throughput) 실험

시뮬레이션 실험을 살펴보면, 일정한 패턴으로 특정 값의 변경만으로 실험을 반복적으로 수행하는 특징을 가진다. 이러한 실험에서의 특정 값은 입력 데이터이거나, 시뮬레이션을 수행하기 위한 스크립트의 일부 값이 가능하다.

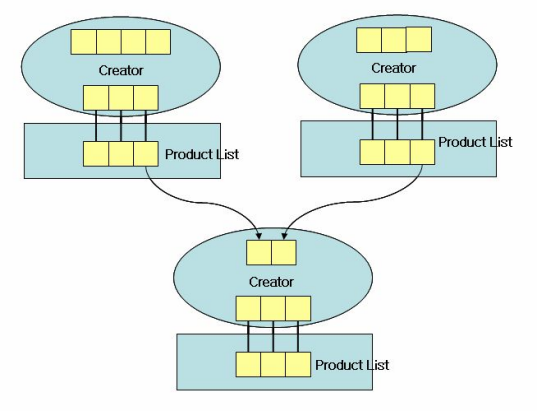

그림 3. 액티브 파일 시스템 개념도
Fig. 3 The concept diagram of Active File System

본 연구에서는 이러한 특정 값의 변경을 통한 새로운 실험을 단순화 하는 목적으로 매개 변수화 (Parameterize) 기법을 제안하고 있다. 즉 시뮬레이션 실험 정의 시 매개 변수화 될 수 있는 값을 유연하게 매개 변수화 함으로써, 실험자의 단순한 반복 작업을 최소화하고 있다. 이러한 기법은 사용자에 의해 정의된 작업들에 대해 템플릿 형태로 표현하고, 그 인스턴스(Instance)로서 매개 변수 값의 변화를 통해 대량의 실험 작업을 생성할 수 있으며, 그리드 자원으로 제출하여 고효율의 실험을 가능하게 할 수 있다.

3.6 작업모델

앞장에서 설명했듯이 여러 단계를 가지는 시뮬레이션 실험에서 자동화된 실험 프로세스와 비 동기적인 실험 프로세스들이 동시에 존재할 경우 기존의 워크플로우 모델에서는 정의하기 어려운 단점을 가지고 있다. 이에 본 연구에서는 DAG (Directed Acyclic Graph) 로써 자동화된 작업 흐름을 정의하고, DAG에 포함되지 않은 작업을 비 동기적인 작업으로 표현함으로써 유연한 워크플로우 표현을 가능하게 한다.

이를 위해 본 연구에서는 3 종류의 작업모델을 제안하고 있다. 첫 번째 모델은 태스크(Task)로써 각 실험 프로세스를 표현하며, 두 번째 모델은 DAG로써 각 실행 순서를 가지는 다수의 태스크의 집합을 정의한다. 마지막으로 Job으로써 시뮬레이션 실험 전체를 표현하는 작업 모델로써 각 프로세스의 구조를 (그림 4)와 같이 표현할 수 있다

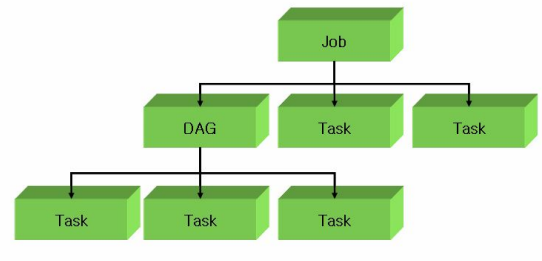

그림 4. 3 수준의 작업 모델
Fig. 4 The three level job model

본 연구에서는 데이터의 흐름과 실행 프로세스의 흐름을 중심으로 하여 워크플로우 모델을 정의한다. 여기서 제시한 워크플로우 모델은 데이터의 의존성 정보와 실행흐름을 서로 다른 공간으로 분리한다. 먼저 데이터 의존성에 대한 정의는 데이터 흐름 공간(data flow space)으로 정의한다. 각 Task의 입력 파일은 Repository내의 파일 혹은 다른 Task의 결과 파일과 관련되어 있다. 이에 각 Task의 입력 파일을 Repository 혹은 다른 Task의 결과 파일과의 의존성 정보를 정의함으로써 데이터의 흐름을 정의할 수 있다. 이러한 어플리케이션 수준의 모니터링과 같은 비 동기 작업을 제외하면 데이터 의존성 정보를 통해 Task의 실행 순서 정의가 가능하다.

Ⅳ. 구현

본 연구에서의 워크플로우는 각 프로세스간이 실행순서 정의와 각 프로세스에서의 입력파일에 대한 의존성 관계를 정의함으로써 정의한다. 사용자에 의한 워크플로우의 정의는 사용자의 실험 순서를 본 연구에서 제시한 작업 언어로 표현하는 것이다. 이를 위해 구현된 작업정의 환경 저작도구는 사용자가 직접 XML로 정의된 작업 정의 언어로써 자신의 작업을 작성하는 것이 아니라 그래픽 환경으로써 직관적으로 작업을 정의할 수 있는 환경을 제공하게 된다. (그림 5)는 각각의 작업 프로세스를 정의한 후 프로세스간의 실행 순서를 정의하는 화면이다. 사용자는 각 실험 프로세스에 대해 화살표로써 각 작업의 실행 프로세스를 정의할 수 있다. 다음 (그림 6)는 각 파일들 간의 의존성 정보를

정의하는 환경이다. 각 실행 프로세스에서의 입력파일은 그 파일이 이전에 생성되는 실행 프로세스의 결과 파일과의 연관성을 정의한다. 사용자는 각 입력파일과 다른 프로세스의 결과 파일과 바인드(bind) 함으로써 각 데이터간의 의존성 정보를 정의할 수 있다.

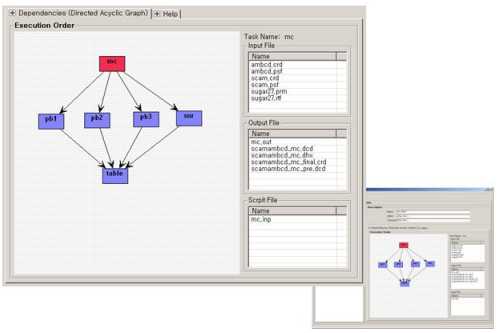

그림 5. 작업 흐름 정의
Fig. 5 Define the Workflow

마지막으로 매개 변수화 된 프로세스들은 각 프로세스의 복제를 통해 다수의 프로세스를 정의할 수 있으며 복제된 각 프로세스는 (그림 7)과 같이 사용자 정의에 따라 스크립트와 파일에 대한 매개 변수화 된 일부 값을 변경을 통하여 다중의 시뮬레이션 프로세스로 정의할 수 있다. 이러한 모델을 통해 작업정의 환경에서는 Job 상위에 프로젝트(Project) 개념을 정의한다. 프로젝트는 유사한 Job들간의 집합으로 정의하고 프로젝트 내에 포함된 Job들은 같은 매개변수에 대해 서로 다른 값을 가지게 된다. 하나의 프로젝트 내의 Job들은 사용자에 의해 매개변수의 값에 대한 변경이 가능하며, 이러한 형태의 프로젝트는 하나의 프로젝트 시트(Project Sheet)로 표현하여 매개변수 값에 대한 각 Job들을 분석할 수 있는 환경을 제공한다. (그림 8)과 같이 프로젝트 시트의 에트리뷰트 값은 매개변수 화 된 값들과 결과물로 사용자의 정의에 의해 구성 가능하다.

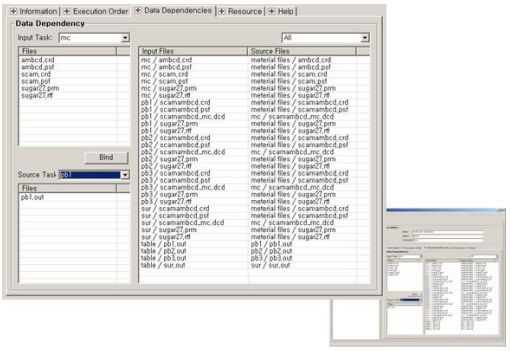

그림 6. 데이터 의존성 정의
Fig. 6 Define Data Dependencies

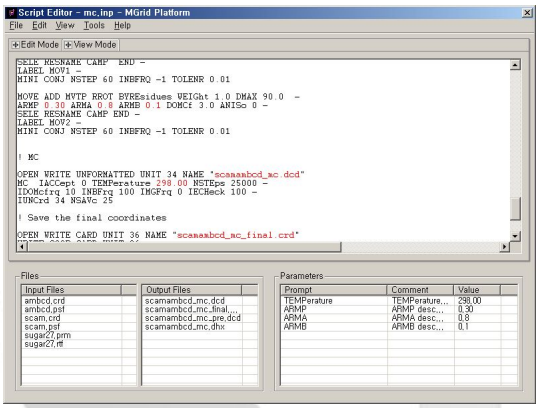

그림 7. 스크립트 분석 및 매개 변수화
Fig. 7 Script analysis and parameterize

V. 결론 및 향후 과제

본 연구에서는 응용 시뮬레이션 연구자의 워크플로우를 분석하고, 단순하고 반복적인 작업을 최소화하기 위한 계산 그리드 상에서의 워크플로우 모델과 데이터 지향적인 매개변수화 기법을 정의, 구현하였다. 본 연구에서 제시한 워크플로우 모델에서는 프로세스의 실행 순서와 데이터간의 연관성을 표현한 데이터 흐름을 서로 다른 공간에서 분리하여 정의함으로써 직관적이고 유연한 작업 정의를 가능하게 하였다. 또한 응용 시뮬레이션 실험을 분석하여, 단순하고 반복적인 작업을 최소화 하도록 매개 변수화 개념을 도입하였다.

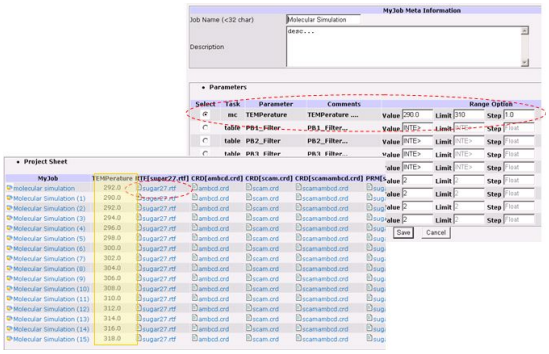

그림 8. 프로젝트 시트
Fig. 8 Project Sheet

향후 과제로써 데이터를 중심의 실험환경을 위해 사용자는 결과데이터에 대한 요구를 통해, 생성하고자 하는 데이터간의 의존성 정보를 통해 각 실험 프로세스들이 실행할 수 있는 효율적인 알고리즘의 개발이 이루어져야 할 것이며, 이를 통해 연구자들에서 데이터 중심으로 연구할 수 있는 실험 환경의 제공이 가능해 질 것이다.

참고문헌

[1] Ian Foster, Carl Kesselman, Steven Tuecke, "The Anatomy of the Grid", International J. Supercomputer Applications. 2001

[2] Ian Foster, Carl Kesselman, "Globus: A Metacomputing Infrastructure Toolkit", http://www.globus.org/

[3] Ian Foster, Carl Kesselman, "The Globus Project: A Status Report"

[4] 정갑주, 황선태, 김동욱, 홍상현, 이종현, 이건영, 구기범, "MGrid: A Molecular Simulation Grid System on N*Grid Testbed", 컴퓨터시스템 연구회 추계 학술발표회 논문집, pp139-145, 8 Nov 2003

[5] 홍상현, 황선태, 심규호, 정갑주, 김동욱, 조금원, "MGrid를 위한 PSE 설계 및 구현" 컴퓨터시스템 연구회 추계 학술발표회 논문집, pp134-138, 8 Nov 2003

[6] Global Grid Forum, http://www.globus.org

[7] Tomasz Haupt, "Grid Job and Distributed Simulation Systems", Cooperative Computing Laboratory, CAVS, ERC, Mississippi State University

[8] Albeaus Bayucan, "Grid-enabled PBS: the PBS -Globus Inerface", Veridian System, PBS Products

[9] Job Submission Description Language: http://www.epcc.ed.ac.uk/~ali/WORK. GGF/JSDL-WG

[10] 오종태, DB에이전트를 이용한 전자상거래 워크플로우 모델링 도구 설계, 한국컴퓨터정보학회논문지, 제8권 제3호, 2003

[11] 이용주, 시멘틱 e-워크플로우 프로세스를 이용한 동적 웹 서비스 조합. 한국컴퓨터정보학회논문지, 제 10권 제 1호, 2005

[12] David Hollingsworth "Workflow Management Coalition The Workflow Reference Model" 19-Jan-95, http://www.wfmc.org

[13] David Hollingsworth, "Workflow Management Coalition The Workflow Reference", 1995

[14] 홍상현, "컴퓨팅 그리드에서의 HPC 사용자를 위한 PSE 설계 및 구현, 석사학위논문, 국민대학교, 전산과학과, 2003

[15] 심규호, "계산 그리드에서 분자 시뮬레이션 실험을 위한 Job 모델에 관한 연구", 석사학위논문, 국민대학교, 전산과학과, 2004

저 자 소 개

황 선 태

1985년 서울대학교 컴퓨터공학과 졸업
1987년 서울대학교 대학원 컴퓨터공학과(공학석사)
1996년 Manchester University, PhD
1997년~현재 국민대학교 컴퓨터학부 부교수

심 규 호

2002년 국민대학교 컴퓨터학부 졸업
2005년 국민대학교 일반대학원 전산과학과(이학석사)
2005년~현재 한국전자통신연구원 연구원