# GRID를 이용한 HLA 기반 객체 지향 분산 시뮬레이션

김창훈[0] 이태동 유양선 정창성 박형우*
고려대학교 전자공학과, 한국과학기술정보연구원 슈퍼컴퓨팅센터
(manipp[0], leetd, thezoot)@snoopy.korea.ac.kr, csjeong@charlie.korea.ac.kr, hwpark@hpc.ne.kr

# HLA-Based Distributed Object-Oriented War Game Simulation on GRID

Chang-Hoon Kim[0] Tae-Dong Lee Yang-Sun Ryu Chang-Sung Jeong Hyoung-Woo Park
Dept. of Electronics Engineering Graduate School, Korea University
Supercomputing Center, Korea Institute of Science and Technology Information

요 약

본 논문은 GRID 상에서 HLA(High Level Architecture)를 기반으로 한 분산 객체 지향 wargame simulation의 디자인과 구현에 관해 기술한다. HLA는 DIS[1]의 뒤를 이어 제안된 아키텍처로서 simulation에 원활한 data 교류와 동기화를 제공한다. 또한, GRID는 전세계에 펼쳐져 있는 자원들에 대한 관리와 접근, 사용을 위한 다양한 기능과 안전하고 편리한 security를 보장한다. 본 논문에서는 HLA를 사용해서 simulation에 뛰어난 상호 연동 능력과 재사용성을 부여하고, GRID를 통해 대규모의 프로젝트를 위한 광범위한 자원을 보다 안전하고 효율적으로 사용할 수 있도록 하는 환경을 구현하였다. 우리는 이 simulation을 HDOWS-G(HLA-based Distributed Object-oriented War game Simulation on Grid)라 부르기로 한다.

## 1. Introduction

HLA의 궁극적인 목표는 넓은 분산 환경 하의 다양한 simulation들이 상호간에 강력한 호환성을 가지고 긴밀한 협조 하에 동작하고, 한번 운용된 simulation들을 미래에 보다 발전된 기술과 함께 보다 편리하게 재사용하고자 하는데 있다. HLA는 이를 위한 기본 framework이다.

이러한 대규모의 simulation들을 운용하는 데 있어서 또 하나의 중요한 문제는 자원(resource) 의 확보이다. 이들 simulation은 종류에 따라서, 다량의 데이터와 계산량을 필요로 한다. 그러나, 종종 단일한 영역에 한정된 자원들 만으로는 요구되는 자원의 범위를 감당하기 힘들다. 본 논문에서는 이러한 자원 문제를, GRID 환경을 도입함으로써 해결해 보고자 한다. GRID 환경이란, 사용자가 어떤 큰 문제를 해결하고자 할 때, 마치 단일 시스템을 쓰듯이 전세계에 펼쳐져 있는 자원들을 쓸 수 있도록 제공된 사용자 환경이라 할 수 있다[2]. 이러한 환경은 필수적으로 자원의 이질성이나 보안 문제, 자원 상태를 제어하기 위한 미들웨어를 필요로 한다. Globus는 가장 광범위하게 사용되는 GRID 미들웨어로서 기존의 각 시스템이나 네트워크 정책과 강력한 호환성을 가진다.

결국, HDOWS-G에서는 HLA를 이용해서 상호 연동 능력과 재사용성을 충분히 살리면서도 Globus를 이용해서 충분한 자원을 활용하고 확장성을 가질 수 있도록 하고 있다.

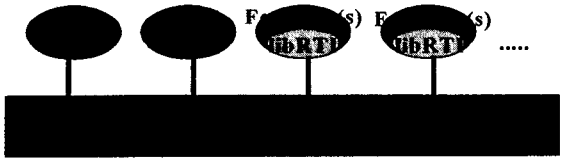
2장에서는 HLA와 Globus에 대한 전체적인 소개를 하고, 3장과 4장에서는 HDOWS-G의 구조와 구현에 대해 다룬다.5장에서는 결론과 앞으로 할 일을 언급한다.

## 2. Overview of HLA and Globus

### 2.1 HLA

HLA의 첫 번째 구성 요소는 Federation rule이다.federation rule은 federation이 되기 위한 조건을 기술한 것이다. HLA에서 federate는 하나의 simulation에 참여한 applicaton들을 말한다.federation은 RTI를 지원하는 federate들의 집합이며, federate들은 RTI라는 interface를 통해서 상호 작용한다. 이 interface를 정의한 것이 두 번째 구성 요소인 Interface Specification이다. 마지막 구성 요소인 OMT는 federaton에 사용되는 객체들에 대한 정보 양식을 말한다. feration 전체에 대한 객체 정보를 모델링 한 것을 FOM으로 부른다[3].

Federate들은 RTI의 Interface를 불러서 다른 federate에게 영향을 주고, RTI의 Callback을 받아서 자신의 federate에게 반영한다. [그림 1]은 RTI의 구성 요소들을 나타낸다. RtiExec는 FedExec의 생성과 소멸을 담당한다. FedExec는 Federate들의 참가와 탈퇴를 담당한다. Fedetate들은 libRTI를 통해서 RTI의 여섯 가지 서비스를 이용한다. 이들 서비스들은 federation의 생성과 federate의 참가, class 선언과 객체 생성, data filtering, 시간 전진 등의 기능을 담당한다.



[그림 1] Components of RTI

* 본 연구는 2002년 한국과학기술정보연구원의 그리드 미들웨어 연구 과제(어플리케이션을 위한 PSE 기술)로부터 연구 지원을 받았습니다.

### 2.2 Globus

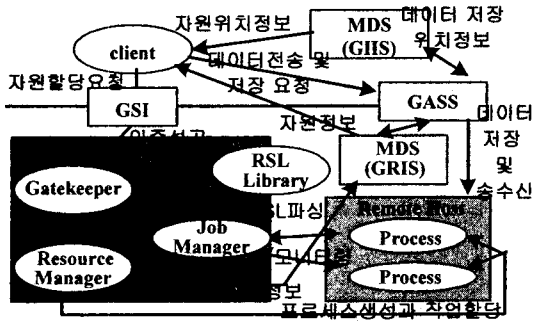
Globus의 장점은, GRID에서 필요로 하는 다양한 서비스들을 분리 가능한 독립된 모듈들로 설계했다는 점이다. [그림2]가 각각의 요소들의 작용을 보여준다.

-MDS(Metacomputing Directory Service) : LDAP (Lightweight Directory Access Protocol)을 사용해서 자원의 위치와 자원에 대한 메타데이터를 제공한다.

-GRAM(Grid Resource Allocation Management) : Globus에서 자원의 할당과 관리를 담당한다. GRAM을 통해서 사용자는 원격지의 자원들을 사용할 수가 있다.

-GSI(Grid Security Infrastructure) : GSI는 Globus의 보안을 담당한다. PKI와 SSL에 기반을 두고 Single sign-on 기능을 지원한다.

-GASS(Globus Access to Secondary Storage) : 원격지에 있는 파일의 처리나 데이터의 분산 저장을 담당한다.




[그림2] Process of Globus

### 3. Architecture of HDOWS-G

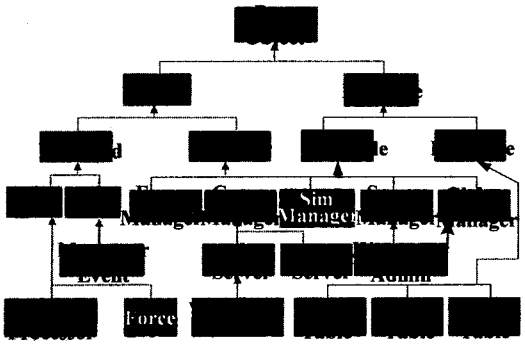
우리는 HLA 응용 프로그램 디자인을 위한 FEDEP 절차를 충분히 이용하면서도 객체 지향 설계의 강점이 발휘되도록 하였다. 또한 Globus의 리소스 활용 능력을 최대한 살릴 수 있도록 프로그램 자체에 자동 구성 능력을 가지도록 설계하였다.

### 3.1 Design of simulation

전체적인 구조는 client-server구조로 이루어져 있다. client는 windows 기반으로서 부대의 이동, 공격 명령이나 경로 설정과 같은 이벤트들을 발생시킨다. 또한, 위치정보나 탐지 정보, 전투력 정보를 visualization하는 역할을 담당한다. server는 Linux 기반으로서 simulation을 수행하고 결과값을 주기적으로 client에게 전송한다.

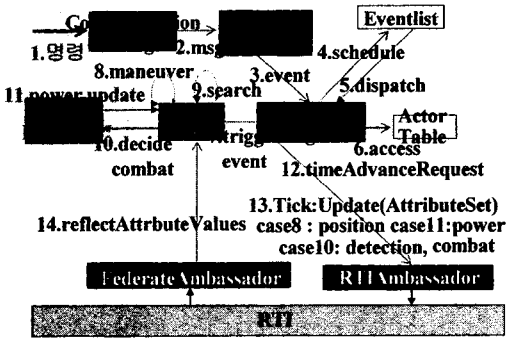
server는 master와 slave의 두 가지 종류로 생성된다. client는 처음에 master server로 접속하지만 master는 Globus를 사용해서 여러 개의 slave server를 생성시키고, 각각의 slave들이 필요한 객체의 생성을 나누어서 수행하게 한다. 각 slave들은 하나의 federate로서 RTI에 join하고, 모든 simulation data는 RTI를 통해 전송된다. master는 client로부터의 메세지를 slave로 전달하고, RTI로부터 전해진 data를 client로전송한다.

우리는 상속을 이용한 객체 지향적 설계를 통해 보다 큰 프로그램에서의 재사용성을 극대화하고, 독립된 모듈이 주는 장점을 살렸다. [그림 3]이 서버의 class구조를 보여준다. 짙은 바탕으로 처리한 class들이 RTI와의 interaction을 담당한다. SimManager는 force의 객체와 data들을 RTI에 선언하고, 시간의 전진을 담당한다. RTI는 Force에 주기적으로 data의 변화를 반영한다. 빗금으로 그려진 class들은 globus의 동작에 관여한다. GlobusManager가 모든 globus process를 총괄하고 나머지 클래스들이 이에 대한 메시지 처리를 담당한다.




[그림 3] Class Diagram of HDOWS-G server

simulation이 수행하고자 하는 것은 기동과 탐지, 근접전투 알고리즘이다. 사용자의 명령에 의해 기동하던 부대는 계속된 탐지를 통해서 적을 식별하고 전투를 벌이게 된다. 이러한 일련의 사건들은 모두 이벤트로 구성되어 이벤트 리스트에 의해 통제되며, 이들 간의 통신은 각 이벤트에 대한 메시지들과 메시지 큐가 담당한다. [그림 4]는 시스템이 초기화된 후의 메시지 작용을 나타낸 그림이다.



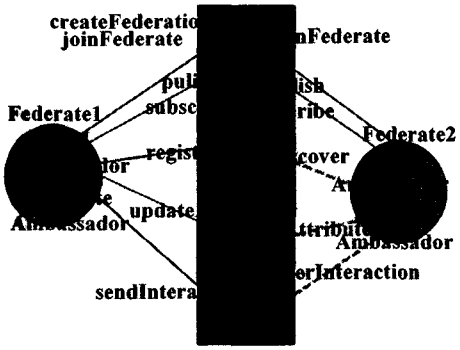

[그림 4] Interaction Diagram of HDOWS-G server

### 3.2 Operation of HLA components

HDOWS-G 시스템의 실행이 시작되면, 프로그램의 초기화가 이루어진다. 이때, 각 manager들의 setup이 이루어지는데, 앞서 말했듯이, SimManager에서 HLA process가 시작된다.

FOM은 Force를 Object Class로 두고, side, power, pos_x, pos_y, detected_flag를 attributes로 하여 구성하였다. 각 federate(server)는 federation을 만들고, join하고, FOM에 명시된 교환할 data를 publish하고 subscribe한다. 또, force에 대한 객체의 핸들을 생성해

객체들을 통제하는데, 이 과정이 register이다. 이후에는 [그림 4]에서 보여지듯이, 시간의 전진과 함께 data에 대한 주기적인 update와 reflect가 이루어진다. [그림 5]는 RTI의 process를 나타내는데, 점선은 callback을 의미한다. HLA에서는 이러한 일련의 과정을 통하여 federate간에 원활한 data의 교류를 가능케 하고, 시간의 동기화를 이룩할 수 있다. 또한 simulation과 통신을 분리하여 효율적인 simulation을 도모하고, 프로그램의 재사용성을 높이고 있다.



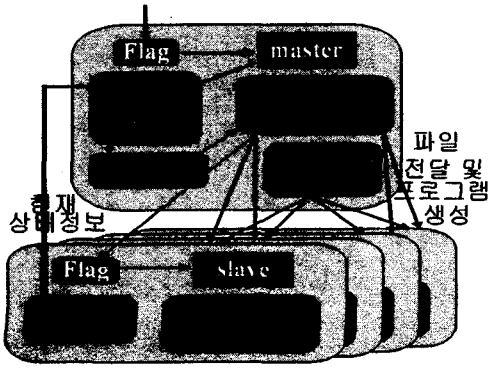

[그림 5] RTI process

### 3.3 Globus Process of HDOWS-G

Globus의 사용은 유휴 자원을 이용해서 하나의 서버에 집중되는 computing load를 줄이기 위해 이루어졌다. 이는 스스로 자원을 찾아내고 그 자원에 프로그램을 배포해서 실행시킴으로써 분산 simulation을 수행하는 강력한 확장성을 내포한다.

프로그램은 flag의 설정에 의해서 자동으로 master와 slave의 역할을 인지한다. master는 MDS의 GIIS에 등록된 서버들에서 사용 가능한 자원을 찾아내어 server list를 작성한다. GASS를 이용해서 이들 server로 객체 생성 정보와 프로그램 파일, 환경 설정 파일을 전송한 후, DUROC을 이용해서 이들을 slave로 동작하게 한다.

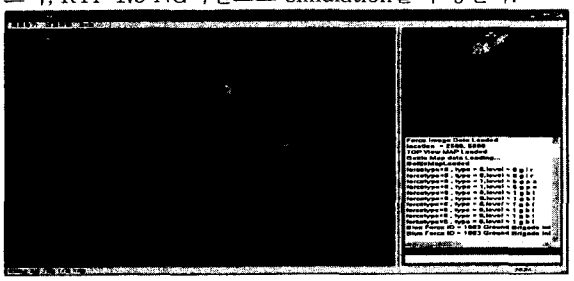
사용자는 마치 한 컴퓨터에 접속한 듯 편리한 인터페이스를 가지면서도, 자원의 제한 없이 빠르고 효율적인 작업을 할 수 있다. 또한 안전하고 편리하게 security를 보장받을 수 있다. [그림 6]에서 이러한 과정을 나타내었다.




[그림 6] Globus Process of HDOWS-G

### 4. Implementation

클라이언트의 구조는 Visual C++ 의 MFC 라이브러리를 이용하였다. 창은 크게 네 개로 나뉘어지는데, 왼쪽 큰 창은 지형과 부대의 움직임을 표시하고, 오른쪽 위 창은 우리나라 지도에서 위치를 선택하는 역할을 하며,그 아래 창은 simulation 처리 메시지를 보여준다. 마지막 작은 창은 사용자간 메시지를 교환하기 위함이다. 사용자는 메뉴에서 공격과 이동 등을 선택할 수 있다. 서버는 C++기반 프로그램으로 되어 있다. Globus 2.0 버전 API를 사용했으며, RTI-1.3 NG버전으로 simulation을 수행한다.



[그림 7] HDOWS-G client

### 5. Conclusion and Future Work

좋은 simulation이 갖추어야 할 가장 중요한 요건은 그것이 현실을 얼마나 완전하게 구현할 수 있는가이다. 이 조건을 만족시키기 위해서는 보다 풍부한 자원과 효율적인 구조가 필요하다. 대규모의 simulation이라면 그러한 환경은 더욱 필수적이다. HLA는 원활한 data 전송과 동기화를 제공하면서 뛰어난 상호 운용성과 재사용성을 갖추고 있다. Globus는 대규모의 프로젝트를 위해 필요한 자원들을 확보하고 사용하기 위한 여러 가지 강력한 기능들과 함께 안전하고 편리한 인증 기능을 제공한다. 본 논문에서는 이들을 simulation에 결합하여 보다 현실적인 simulation을 만들 수 있는 방안을 제시하였다.

물론, HDOWS-G 역시 많은 과제를 안고 있다. MDS의 모니터링을 통한 자원의 주기적인 검색, 보다 나은 자원으로의 프로그램 실행 이전, 대규모 simulation으로의 확장 등이 보다 나은 simulation 구현을 위해서 필요할 것으로 생각한다.

### 6. 참고 문헌

[1]. Wayne J. Davis, " Simulation: Technologies in the New Millennium" . Proceedings of the 1999 Winter Simulation Conference.

[2]. I. Foster, C. Kesselman, S. Tuecke , "The Anatomy of the Grid : Enabling Scalable Virtual Organizations" , International J. Supercomputer Applications, 15(3), 2001.

[3]. DMSO, "HLA RTI-1.3NG Programmer's Guide Version 3.2"