논문 12-37C-11-10

한국통신학회논문지 '12-11 Vol.37C No.11
http://dx.doi.org/10.7840/kics.2012.37C.11.1094

하이브리드 데이터 통신 방식을 적용한 IEEE 1516.1-2000 표준의 구현

심 준 용◆, 위 성 혁°

An Implementation of IEEE 1516.1-2000 Standard with the Hybrid Data Communication Method

Jun-yong Shim◆, Soung-hyouk Wi°

요 약

최근 국방 소프트웨어 산업은 다양한 무기체계 사업을 통해 모델링 및 시뮬레이션 기술을 적용한 시뮬레이션 시스템 개발을 늘리고 있으며, 특히 이기종 시뮬레이터 간 이식성 및 상호 연동성 확보를 위해서 분산 시뮬레이션 표준 프레임워크인 HLA(High Level Architecture)의 적용을 규정하고 있다. HLA는 분산 환경에서 시뮬레이터 간 데이터 교환 및 순서화를 제공하기 위한 서비스를 정의하며, HLA 규칙, Federate 인터페이스 표준 그리고 객체 모델 템플릿의 주요 컴포넌트로 구성된다. RTI(Run-Time Infrastructure)는 Federate 인터페이스 표준을 구현한 소프트웨어로써 Federation 환경에 참여 중인 Federate들이 정보를 교환할 수 있도록 기능을 제공한다. RTI 기술은 워 게임, 가상 시뮬레이션, 훈련 및 무기체계 소프트웨어 연동과 같은 다양한 분야에서 사용되고 있다. 하지만 국내에서 개발된 사례가 없어 모두 외산 제품에 의존하고 있는 실정이다. 본 논문은 국내에서 개발된 IEEE1516.1-2000 표준의 RTI 구현을 소개한다. 특히, Federate 간 데이터 교환 성능을 높이기 위해서 서버-클라이언트 방식과 단대단 방식을 혼합한 하이브리드 데이터 통신 방식의 적용 방법을 기술하고, 상용 RTI와 데이터 처리율 및 네트워크 지연 시간의 비교를 통해 성능이 개선되었음을 보인다.

Key Words : HLA, RTI, IEEE1516.1-2000, Federation, Federate, Distributed Simulation, Interoperability

ABSTRACT

Recently, software industry regarding national defense increases system development of distributed simulation system of M&S based to overcome limit of resource and expense. It is one of key technologies for offering of mutual validation among objects and reuse of objects which are discussed for developing these systems. RTI, implementation of HLA interface specification as software providing these technologies uses Federation Object Model for exchanging information with joined federates in the federation and each federate has a characteristic that is supposed to have identical FOM in the federation. This technology is a software which is to provide the core technology which was suggested by the United state's military M&S standard framework. Simulator, virtual simulation, and inter-connection between military weapons system S/W which executes on network which is M&S's core base technology, and it is a technology which also can be used for various inter-connection between S/W such as game and on-line phone. These days although RTI is used in military war game or

◆ 주저자 : LIG넥스원(주) 소프트웨어연구센터, junyong.shim@lignex1.com, 정회원
° 교신저자 : LIG넥스원(주) 소프트웨어연구센터, sounghyouk.wi@lignex1.com, 정회원
논문번호 : KICS2012-08-400, 접수일자 : 2012년 8월 31일, 최종논문접수일자 : 2012년 11월 14일

tactical training unit field, there is none in Korea. Also, it is used in mobile-game, distribution game, net management, robot field, and other civilian field, but the number of examples are so small and informalized. Through this developing project, we developed the core technique and RTI software and provided performance of COTS level to improve communication algorithms.

Ⅰ. 서 론

대규모 분산 환경(Large Scaled Distributed Environment)에서 사용자간 다양한 형태의 상호작용을 시뮬레이션 할 요구가 증가함에 따라 미 국방성(Department of Defense, DoD)은 Simulator Networking(SIMNET, 1985), Distributed Interactive Simulation(DIS, 1989), Aggregate Level Simulation Protocol(ALSP, 1992)과 같은 분산 시뮬레이션 아키텍처를 연구해왔다. 1996년 산하기관인 Defense Modeling and Simulation Office(DMSO, 현재 MSCO[1]로 변경)를 설립하여 기존 아키텍처의 제한사항 극복과 함께 분산 시뮬레이션 환경의 상호운용성(Interoperability)을 보장하고, 모의 소프트웨어를 포함한 실 시스템과의 연동을 용이하게 하며, 시뮬레이션 소프트웨어의 재사용성을 촉진시키기 위한 방안으로 High Level Architecture(HLA)를 발표한다. HLA 표준은 DMSO가 제정한 HLA 1.3과 IEEE가 제정한 IEEE 1516-2000이 있으며, 최근 IEEE1516-2010 버전이 발표되면서 분산 환경을 위한 다양한 IT 기술들이 적용되어 발전하고 있다. Run-Time Infrastructure(RTI)는 HLA 표준을 구현한 소프트웨어로서 시뮬레이션 개발에 필요한 기반 소프트웨어를 제공하며, 워 게임, 가상 시뮬레이션, 훈련 및 무기체계 소프트웨어 연동과 같은 국방 분야뿐만 아니라 원격 가상교육, 온라인 게임, 의료 시뮬레이션 등의 민수 분야에서도 다양하게 응용되고 있다. 주요 제품으로는 Pitch 社의 pRTI[2], MÄK社의 High Performance RTI[3], Raytheon VTC社의 RTI-NG Pro[4] 등이 시장을 형성하고 있다.

한편, 국내에서는 RTI 기술과 관련하여 연구 목적의 Prototype이 개발 된 사례가 있지만 상업 목적의 RTI 소프트웨어가 개발된 사례가 없기 때문에 외산 제품에 의존하고 있는 실정이다. RTI 기술은 국방 모델링 및 시뮬레이션 환경 구축을 위한 핵심 기술이며, 각 군의 독자 모델 간 연동을 위한 기반 기술로 활용될 수 있기 때문에 국내 고유의 기술 확보가 필요하다. 특히, 향후 전시작전권 전환에 따라 미군의 전장관리 정보체계와의 연동 및 상호운용 수준의 향상이 중요한 문제로 부각되고 있기 때문에 독자적으로 각 국의 전장관리정보체계 간 연동 문제를 해결하는 것이 앞으로의 중요한 과제라고 할 수 있다[5].

본 논문은 분산 시뮬레이션 환경의 상호 연동 표준인 HLA와 HLA의 핵심 요소인 RTI 기능을 살펴보고, 국내에서 개발된 RTI의 기능 및 주요 특징을 소개한다. 특히, 시뮬레이터 간 데이터 교환 성능을 높이기 위해 제안된 RTI 구성요소 간 하이브리드 데이터 통신 방법을 기술하고, 제안 구조를 적용한 RTI와 상용 RTI의 네트워크 지연시간(Latency) 및 데이터 처리량(Throughput)을 비교함으로써 성능의 우수함을 보여준다. 논문의 구성은 다음과 같다. 2장은 HLA / RTI 기술 개요 및 서비스를 살펴보고, 3장에서 RTI 구성 요소 및 시뮬레이션의 연동 구조와 특징을 설명한다. 4장은 RTI 개발을 위한 핵심 기술과 데이터 교환 성능을 높이기 위해 제안된 하이브리드 데이터 통신 방법을 기술한다. 5장에서 제안 방법을 적용하여 개발한 RTI와 상용 RTI의 성능을 비교 및 분석한 후, 마지막으로 6장에서 결론을 맺고 향후 과제를 제시한다.

Ⅱ. HLA / RTI 기술 개요 및 서비스

2.1. HLA 개요

HLA는 HLA 규칙[6], Federate 인터페이스 표준(Interface Specification)[7] 및 객체 모델 템플릿(Object Model Template, OMT)[8]의 요소로 구성되며, 연합관리, 선언관리, 객체관리, 시간관리, 소유권관리 및 데이터분산 관리의 6가지 서비스를 제공한다. 또한 시뮬레이션 소프트웨어의 재사용 단위로서 구현 모델을 제공하는 Federate와 시뮬레이션 수행 시간에 Federate 그룹의 통신 환경을 제공하는 Federation을 정의한다.

2.1.1. HLA 규칙

HLA 규칙은 Federation과 Federation에 참여하는 Federate가 HLA기반 시뮬레이션 구현을 위해 준수해야 하는 10가지 규칙을 정의한다. 크게

Federation 구성에 필요한 5가지 규칙과 Federate 개발에 필요한 5가지 규칙으로 나뉜다.

2.1.2. Federate 인터페이스 표준

인터페이스 표준은 Federate 개발에 필요한 6가지의 주요 서비스들을 정의하며, 각 서비스마다 제공되는 기능들의 API를 기술한다. 해당 표준을 구현함으로써 RTI를 개발할 수 있다.

2.1.3. 객체 모델 템플릿

객체 모델 템플릿은 Federate 간 데이터 교환에 필요한 리코딩(Recording) 규칙을 제공하며, 객체 모델(Object Model)을 정의하여 개별 Federate의 정보를 기술하는 Simulation Object Model(SOM)과 Federation에서 다수의 Federate 간 상호작용의 집합을 기술하는 Federation Object Model(FOM)을 작성할 수 있다.

2.2. RTI 개요

RTI는 HLA의 인터페이스 표준을 구현한 미들웨어로서 RTI 실행 프로세스(RTI Executive Process, RtiExec), Federation 실행 프로세스 (Federation Executive Process, FedExec) 그리고 C++ 클래스 라이브러리(Local RTI Component 이하, LRC)로 구성된다. 각 구성 요소는 TCP/UDP/IP 프로토콜을 기반으로 분산 객체 통신 기능을 구현하며, Publish-Subscribe 방식[9]의 데이터 교환 메커니즘을 제공한다. 그림 1은 RTI의 구조를 보여주며, Federate를 가진 각각의 실행 프로세스들은 LRC를 사용하여 FedExec와 통신한다. Federate는 독립적인 프로세스로 존재할 수도 있고, 한 개 이상의 프로세스들의 그룹으로 이루어질 수도 있다.

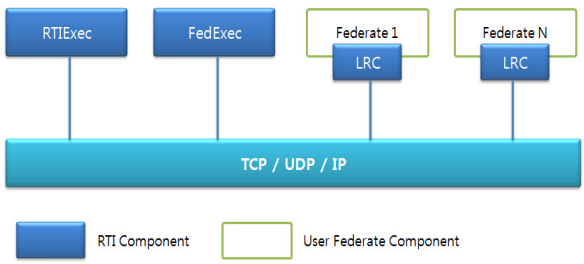

그림 1. RTI 구조
Fig. 1. RTI Structure

2.2.1. RtiExec

RtiExec는 Federation에 참여하는 모든 Federate들이 통신할 수 있도록 환경을 제공하며, 각 Federate는 LRC의 통신 정보를 구성하기 위해 RtiExec와 상호작용해야 한다. 주요 기능으로 FedExec의 생성 및 삭제를 수행하며, 다수 Federation에 대한 관리 기능을 제공한다. 특히, Federation에 참여하고자 하는 Federate들에게 적절한 Federation 수행을 연결시킨다.

2.2.2. FedExec

FedExec는 오직 하나의 Federation을 관리하며, Federate들이 Federation에 참여하거나 또는 탈퇴하는 기능을 제공한다. 특히, 데이터 교환이 수행될 수 있도록 Federation에 존재하는 객체의 상태 및 정보를 관리한다. Fedexec는 createFederationExecution 함수를 통해 생성되고, destroyFederationExecution 함수를 통해 삭제된다.

2.2.3. LRC

C++ 라이브러리인 Local RTI Component(LRC)는 Federate 개발자에게 HLA 인터페이스 표준에 지정된 RTI 서비스를 제공하며, RTIAmbassador와 FederateAmbassador 클래스를 통해 Federate가 HLA 서비스를 사용할 수 있도록 기능을 구현한다. HLA 인터페이스 표준은 LRC가 Federate들에게 제공해야 하는 서비스와 Federate가 Federation과 상호작용을 위해 구현해야 할 서비스를 명시하고 있다. 그림 2에서 볼 수 있듯이 RTIAmbassador는 LRC 내에서 구현 형태로 존재하며, Federate Code에서 데이터 송신을 수행할 경우 RTIAmbassador 객체를 생성하여 해당 함수를 호출한다. FederateAmbassador의 경우 각 Federate Code가 구현해야 하는 콜백 함수(callback function)를 정의하며, 추상 클래스의 형태로 제공된다. 따라서 Federate Code는 FederateAmbassador를 상속받아 FederateAmbassador에 선언된 함수들을 구현해야 한다. Federate의 구현된 함수를 통해 RTI로부터 제공되는 정보를 수신할 수 있다.

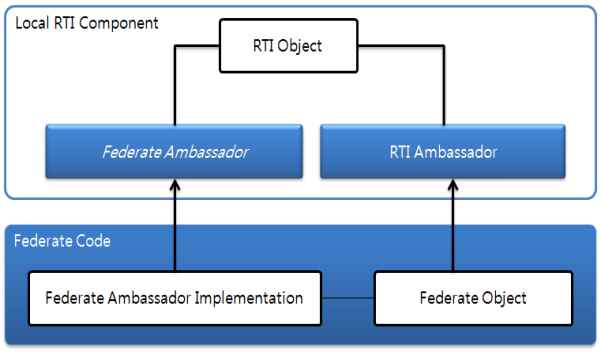

그림 2. RTI와 Federate 관계
Fig. 2. the relation of the federate to RTI

2.2.4. SOM / FOM

OMT 표준에 준하여 작성된 객체 모델인 'Object Class'와 'Interaction Class'를 통해 시뮬레이션 수행 동안 교환할 정보를 선언한다. Object Class는 공중위협, 미사일 등의 플랫폼 정보를 정의하며, Interaction Class는 미사일 발사, 폭파 등의 행위 정보를 정의한다. 개별 Federate의 SOM을 통합하여 FOM을 생성하고, XML 형태의 FDD(FOM Document Data, IEEE 1516표준) 파일로 작성된다. 각 객체 모델의 공유(Sharing) 속성을 'Publish' 또는 'Subscribe'로 설정하여 교환 여부를 결정한다.

2.3. 표준 서비스

RTI의 표준 서비스는 HLA의 인터페이스 명세에 기술되어 있으며, 정리하면 다음과 같다.

2.3.1. 연합 관리(Federation Management)

연합 관리 서비스는 Federation의 생성, Federation에 Federate의 가입, Federate 동기화, Federation에 대한 저장 및 복구, Federation으로부터 Federate 탈퇴 그리고 Federation 제거와 같은 기능을 포함한다.

2.3.2. 선언 관리(Declaration Management)

선언 관리 서비스는 Federation 내에 존재하는 Federate들이 상호작용하기를 원하는 Object(공유객체) 및 Interaction(상호작용) 정보를 Publication(발행) 또는 Subscription(구독)할 수 있는 기능을 제공한다. 각각의 Federate는 정보를 송신하기 위해서 해당 Object나 Interaction에 대해서 Publish를 수행해야하며, 수신하기 위해서는 해당 Object나 Interaction에 대해서 Subscribe를 수행해야 한다.

2.3.3. 객체 관리(Object Management)

객체 관리 서비스는 객체 생성 측면에서는 Instance의 등록(registration), 갱신(update) 그리고 삭제(delete), 객체 소비 측면에서는 Instance의 발견(discovery), 반영(reflection) 그리고 제거(remove) 등을 포함한다. Federate가 정보를 주고받을 수 있는 통신 기능을 제공하기 때문에, 데이터의 인코딩 및 디코딩에 대한 수행이 이루어진다.

2.3.4. 소유권 관리(Ownership Management)

RTI는 Federate에게 몇 가지 제한사항을 주면서 객체 Instance에 대한 갱신과 삭제에 대한 책임을 공유하도록 한다. 객체 Instance에 포함된 Attribute는 Federation 내에서 오직 하나의 Federate에 의해서만 소유될 수 있으며, 소유권 관리 서비스는 각 Attribute에 대한 소유권을 획득 또는 이양할 수 있도록 기능을 제공한다. 소유권 획득 또는 이양은 크게 Acquisition 모델과 Divestiture 모델을 사용하여 소유권 교환을 수행한다. Acquisition 모델은 해당 Attribute에 대해서 소유권을 가져오는 메커니즘을 제공하는데 Intrusive와 non-intrusive 모델이 존재한다. Intrusive 모델은 Attribute를 소유하고 있는 Federate에게 소유권을 Release할 것을 요구하며, non-intrusive 모델은 Attribute를 소유하고 있는 Federate가 소유권을 포기할 것을 협상하는 모델이다. Divestiture 모델은 소유권을 포기하는 메커니즘을 제공한다. 모든 생성 Instance는 HLAprivilegeToDeleteObject라는 Attribute를 갖는다. 해당 Attribute를 소유한 Federate만이 Instance를 삭제할 수 있으며, 이 Attribute의 소유권 또한 교환이 획득 및 이양이 가능하다.

2.3.5. 시간 관리(Time Management)

시간 관리 서비스는 Federate 내에 전달되는 메시지의 순서(Total Ordering)를 보장할 수 있는 관련 메커니즘과 서비스를 포함한다. 크게 Time Regulating과 Time Constrained 상태를 정의한다. 시간 정보가 포함된 정보를 송신하기 위해서 해당 Federate는 Regulating 상태를 선언해야 하며, 시간 정보가 포함된 정보를 수신하기 위해서 해당 Federate는 Constrained 상태를 선언해야 한다.

2.3.6. 데이터분배 관리(Data Distribution Management)

데이터분배 관리 서비스는 관련 없는 송수신 데이터를 줄이기 위한 기능을 포함한다. 선언 관리 서

비스가 Class의 Attribute 수준의 데이터 연관을 제공했다면, 데이터분배 관리 서비스는 Instance의 Attribute 수준의 데이터 연관을 제공한다. 따라서 교환 영역(Region)을 설정한다면 상호작용하고 있는 Federate 간 불필요한 정보의 발생을 줄일 수 있다.

2.3.7. 지원 서비스(Support Services)

지원 서비스는 Name-to-Handle 및 Handle-to-Name 변환, Advisory Switch의 설정, Region 수정 및 RTI 시작 및 종료와 관련된 서비스들을 제공한다.

2.3.8. 관리 객체 모델(Management Object Model)

관리 객체 모델은 Federation이 수행되는 동안 RTI의 운영 정보를 접근할 수 있도록 서비스를 제공한다. Federation 내에 존재하는 Federate들을 제어하고, 감시할 수 있도록 HLAfederate 및 HLAfederation과 같은 MOM 객체를 정의한다.

Ⅲ. RTI 시뮬레이션 구조

3.1. RTI 구조

RTI의 일반적인 구조는 그림 3과 같이 한 개의 RtiExec를 중심으로 최대 허용 개수의 Federation(FedExec)들과 각각의 Federation에 가입될 수 있는 최대 허용 개수의 Federate들로 구성된다.

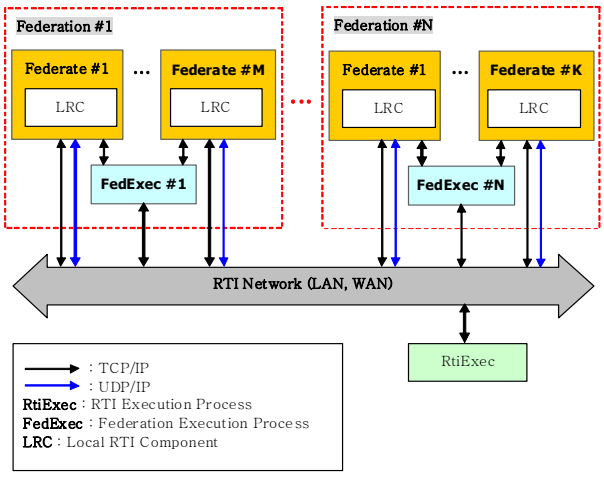

그림 3. RTI 구조
Fig. 3. RTI Structure

생성된 Federation은 서로 독립적이기 때문에 메시지 교환과 같은 동작으로 서로에게 영향을 미치지 않는다. RtiExec와 FedExec의 통신 프로토콜은 TCP/IP 유형 한 가지만을 제공하는 반면, Federate는 TCP/IP 및 UDP/IP의 두 가지 프로토콜 유형을 제공한다. RTIExec는 모든 Federation들과 Federate들의 IP 주소 및 ID 정보를 관리하기 때문에 각 LRC들이 서로 통신하기 위해서는 RtiExec를 통해 해당 정보들을 요청해야 한다. RtiExec 및 FedExec의 구조를 살펴보면 다음과 같다.

3.1.1. RTIExec 구조

RTIExec는 그림 4와 같이 RTI 개발 프레임워크인 RDK(RTI Development Kit) 모듈, Protocol 모듈, Federation 관리 모듈, Support Services 모듈 및 Main 모듈로 구성된다.

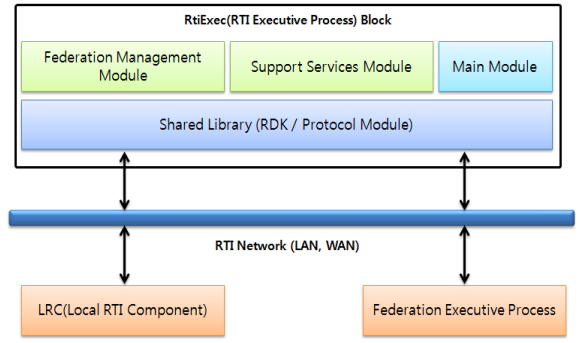

그림 4. RtiExec 블록 모듈 구성도
Fig. 4. RtiExec Block Module Map

RDK와 Protocol 모듈은 RTI 기능 구현을 위해 LRC와 FedExec 모듈이 통신할 수 있는 기반 서비스를 제공하며, Federatopm 관리 모듈은 FedExec의 생성 및 삭제를 관리한다. Support Services 모듈은 Federation 생성 시 필요한 객체들의 핸들 및 이름 정보에 대한 접근 서비스를 제공하고, Main 모듈은 RTIExec를 실행할 수 있는 환경을 구성한다.

3.1.2. FedExec 구조

FedExec는 그림 5와 같이 RDK 모듈, Protocol 모듈, 7가지의 관리 서비스 모듈 및 Main 모듈로 구성된다.

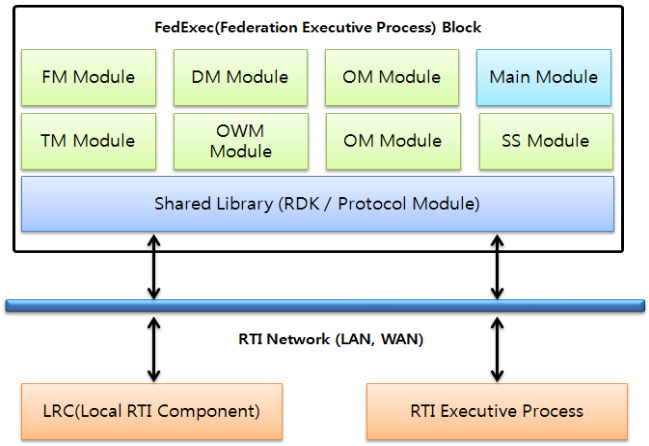

그림 5. FedExec 블록 모듈 구성도
Fig. 5. FedExec Block Module Map

FedExec 블록은 RtiExec 블록과 동일한 모듈 외에 HLA의 핵심 서비스 모듈을 구현한다. 특히, Federation에 가입된 Federate가 서비스 별 기능을 제공받을 수 있도록 서버 역할을 수행한다.

3.2. 시뮬레이션 연동 구조

RTI 시뮬레이션의 연동 구조는 그림 6과 같다. 송신 Federate는 LRC의 RTI 함수를 통해서 데이터를 RTI 서버에 제공하고, 수신 Federate는 구현한 콜백 함수를 통해서 RTI 서버로부터 데이터를 수신한다.

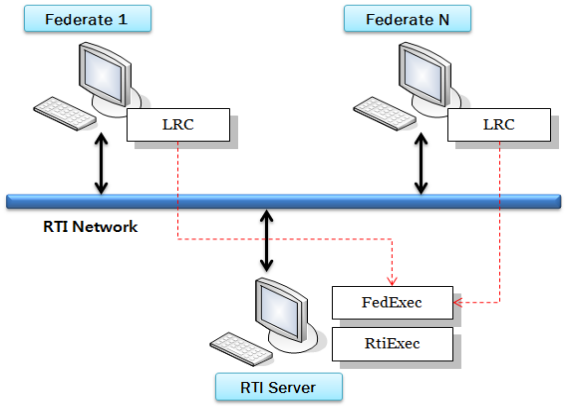

그림 6. RTI 시뮬레이션 연동 구조
Fig. 6. RTI Simulation interconnection

일반적으로 RTI는 Federate 간 연동을 수행하기 위해서 Federation 정보를 관리하는 RTI 서버를 통해 데이터 교환이 이루어지며, 따라서 가입된 Federate가 증가할수록 데이터 교환을 위한 부하가 발생하게 된다.

Ⅳ. 구현 기술 및 하이브리드 데이터 통신 방법

RTI 기술 개발을 위해서 메시지 전달 방식, 시간관리 알고리즘, 데이터 필터링을 위한 데이터분산 알고리즘 그리고 RTI 개발 프레임워크를 구현한다. 각 기술의 내용은 다음과 같다.

4.1. 메시지 전달 방식

RTI는 구성 요소 간 효율적 인터페이스를 제공하기 위해서 그림 7과 같이 Messaging 구조를 적용한다. 메시지는 어플리케이션 간에 혹은 어플리케이션과 사용자 간 네트워크를 통해서 교환되는 데이터를 가리킨다. 이러한 메시지를 통해서 RTI는 분산 환경에서 구동하는 각각의 어플리케이션이 서로 연동할 수 있도록 하는 환경을 제공한다.

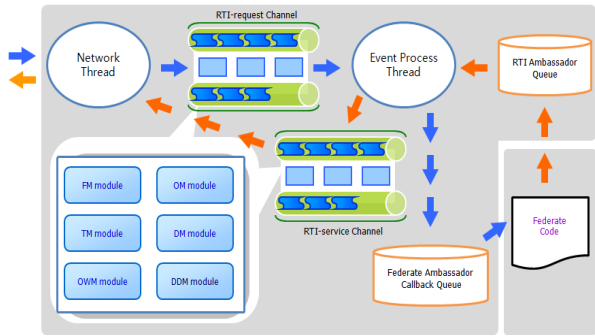

그림 7. 모듈 간 메시지 흐름
Fig. 7. Message Flow between Modules

4.2. 시간관리 알고리즘

시간관리 알고리즘은 HLA의 시간관리 서비스의 기능을 구현하기 위한 기술로서 분산 시뮬레이션에서 각 Federate의 시간 진행이나 메시지 송수신에 있어서 인과관계(Causality)를 보장해주는 방법을 제공한다. 알고리즘의 핵심은 Federate 간 교환되는 메시지의 논리 시간(Logical Time)을 생성하고, 해당 논리 시간을 통해서 Federate가 진행할 수 있는 최소 시간을 계산하는 것(GALT)이다. 시간 관리를 위한 구조는 그림 8과 같다.

4.3. 데이터분산 관리 알고리즘

데이터분산 알고리즘은 송신 Federate와 수신 Federate의 관심 관리를 이용하여 전달할 데이터를 필터링함으로써 네트워크 트래픽을 줄이는 방법을 제공한다. 구조는 그림 9와 같다.

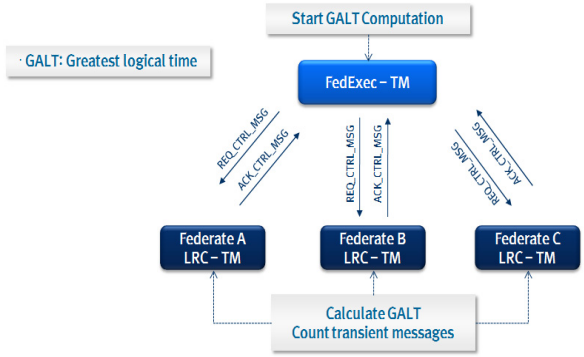

그림 8. 시간관리 알고리즘
Fig. 8. Time Management Algorithm

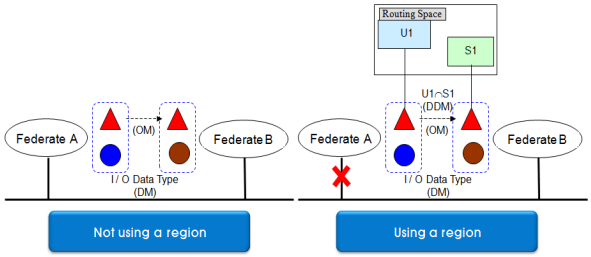

그림 9. 데이터 분산 알고리즘
Fig. 9. Data Distribution Algorithm

일반적으로 FedExec를 거쳐 통신하는 RTI의 경우 송신 Federate가 수신 Federate의 정보를 알 수 없기 때문에 데이터 필터링 시 수신 측 필터링을 사용하는 반면, 개발 RTI는 Federation에 존재하는 각 Federate들이 서로의 정보를 공유하기 때문에 송신 측 필터링을 사용한다. 이는 기존 RTI의 방식보다는 네트워크의 데이터 전송 량을 더 줄일 수 있다.

4.4. RTI 개발 프레임워크(RTI Development Kit, RDK)

RDK는 RTI의 구성 요소인 RtiExec, FedExec 그리고 LRC를 구현하기 위한 공통의 서비스들의 집합이다. Windows와 Linux 플랫폼을 지원하며, 기본적으로 네트워크, 쓰레드, 메모리 관리 및 로깅 서비스 등을 제공한다. RDK의 구조는 그림 10과 같다.

RDK기반 개발의 장점은 각 구성 요소가 HLA의 주요 서비스를 구현하기 위해 상호작용하기 때문에 공통 기능들을 공유함으로써 재사용을 늘리고, 특히 성능을 최적화 시킬 수 있다는 것이다.

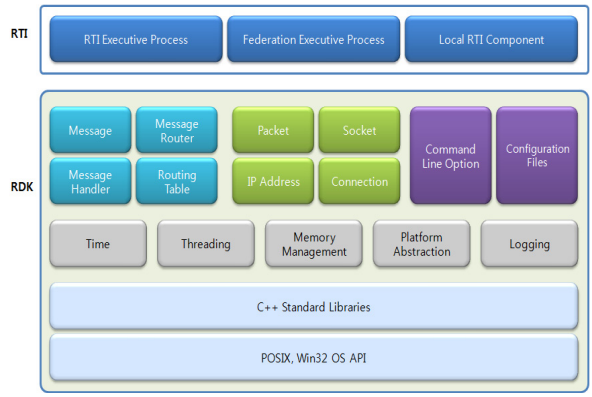

그림 10. RDK 구조
Fig. 10. RDK Structure

4.5. 하이브리드 데이터 통신 방법

일반적으로 서버-클라이언트(Server-Client) 구조는 교환되는 메시지의 크기나 클라이언트의 개수가 늘어남에 따라 서버 측 과부하가 발생할 수 있다. HLA는 RtiExec 및 FedExec가 서버 역할을 수행하고, LRC가 클라이언트 역할을 수행하므로 기존 서버-클라이언트 구조의 문제점을 갖게 된다.

4.5.1. 제안 구조

본 논문은 기존 구조가 갖는 문제를 해결하면서 동시에 Federate 간 데이터 교환 성능을 높이기 위해서 일대일(Peer-to-Peer) 구조를 혼합한 하이브리드 데이터 통신 방식을 적용했다. 이 방식은 Federation에 가입한 Federate가 FedExec의 라우팅 테이블로부터 객체 모델의 선언 정보를(Publish 또는 Subscribe) 확인해서 관계된 Federate의 위치 정보(IP 및 ID)를 가져온 후, 데이터 교환을 일대일로 수행하는 것이다. 그림 11은 그림 6의 기존 방식을 개선한 통신 구조를 보여준다.

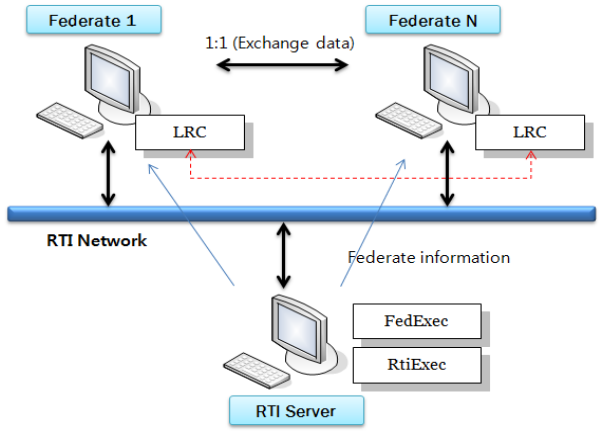

그림 11. 하이브리드 데이터 통신
Fig. 11. Hybrid data communication

제안 방법은 Federate 간 실제 정보를 주고받을 경우 FedExec를 거치지 않기 때문에 다수의 Federate로부터 집중적으로 발생할 수 있는 데이터 교환 횟수를 줄일 수 있는 반면, 일대일 통신을 구현함으로써 시간관리, 소유권관리 및 데이터분산 관리에 대해서 구현 이슈가 발생한다. 시간관리 및 소유권 관리의 경우 관련 정보들이 해당 Federate에만 관여하지만 데이터분산 관리의 영역 설정 기능의 경우 동일 영역을 사용하는 모든 Federate에게 반영되어야 하기 때문에 구현 방식의 개선이 필요하다.

4.5.2. 교환 영역 설정 동기화

데이터를 교환하는 Federate는 시뮬레이션 수행 중에 데이터분산 관리 서비스를 통해서 교환 영역 설정을 변경할 수 있다[10]. 하지만 하이브리드 데이터 교환 방식은 데이터 교환은 일대일 방식으로 이루어지고 영역 정보 설정은 서버-클라이언트 방식으로 이루어지기 때문에 그림 12와 같이 변경된 영역 정보가 서버를 통해 다른 Federate에 도달하기 전에 데이터 교환이 먼저 발생할 수 있다.

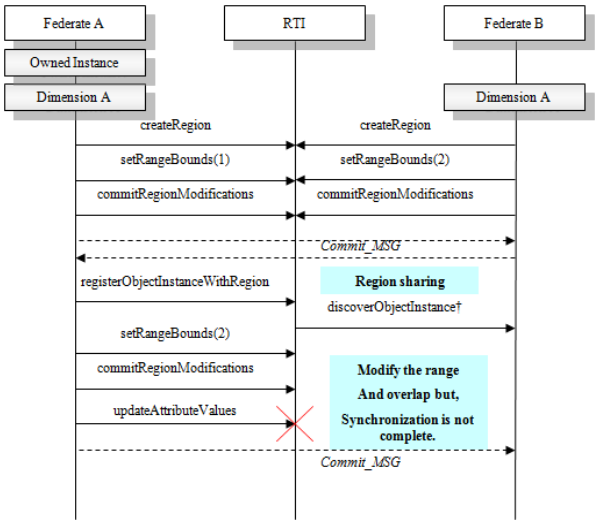

그림 12. 제안 방식의 영역 설정 동기화 문제
Fig. 12. Out of sync problem in dimensions

Federate A가 영역 값을 commitRegionModifications 함수를 통해 서버에 통보 한 후 서버가 Federate B에 반영하기 전에 데이터 교환 기능인 updateAttributeValues 함수가 수행될 수 있다. 따라서 Federate A의 영역 정보와 맞지 않는 잘못된 교환이다. 이를 해결하기 위해서 그림 13과 같이 영역 설정 시 데이터 교환 전에 Hand-Shaking을 수행하도록 했다[11]. Hand-Shaking은 네트워크 환경에서 보낸 데이터의 응답을 수신한 후 다음 데이터를 보내는 방식을 제공하는데, 제안 구조에서 이를 적용하여 Federate A가 영역 값을 설정 한 후 Federate B가 해당 영역 값을 반영할 수 있도록 기다린다. Federate B로부터 영역 설정 완료 여부에 대한 응답 메시지가 발생하면 Federate A는 필터링을 통해 데이터를 갱신하도록 했다.

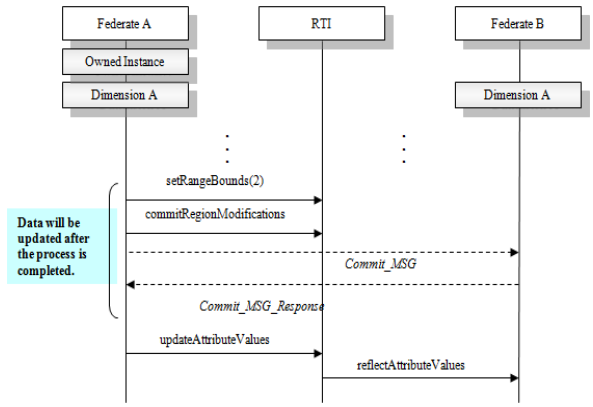

그림 13. 영역 설정 동기화 방법
Fig. 13. Synchronized region solution

위 그림과 같이 updateAttributeValues 함수를 호출하기 전에 Commit 메시지를 교환하고 관련된 Federate들이 영역 정보를 모두 수신했는지 확인함으로써 서버-클라이언트 방식에서 일대일 방식으로 변경할 경우 발생하는 문제점을 해결했다.

V. 성능 시험

개발 RTI의 데이터 교환 성능을 분석하기 위해서 상용 RTI인 RTI-NG Pro를 비교했다. RTI는 네트워크 통신과 FedExec의 다수 Federate에 대한 데이터 처리율이 성능의 중요한 요소이므로 네트워크 지연시간(Response Time) 및 데이터 처리율(Throughput Rate)을 선정했다. 시험을 통해서 상용 수준의 성능을 만족시키는지 검증하고, 개발 RTI에 적용된 하이브리드 데이터 교환 방식이 기존 방식보다 성능에 대한 우수함을 보인다.

5.1. 시험 환경

시험 구조는 그림 14와 같이 Sender Federate와 Receiver Federate 간 1 대 1로 수행하며, 조건은 각 시험에서 100Byte 크기의 Object Instance 수를 1개부터 100개까지 늘려간다.

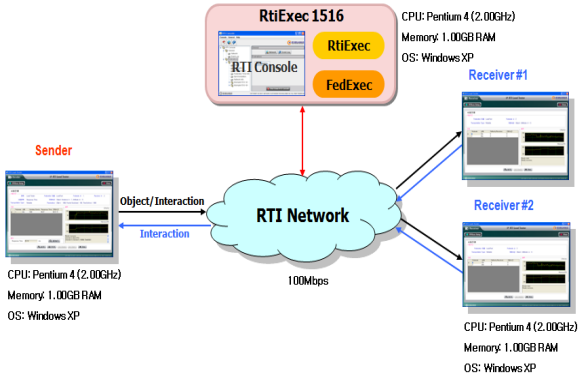

그림 14. 성능시험 환경
Fig. 14. Configuration for performance testing

5.2. 시험 결과

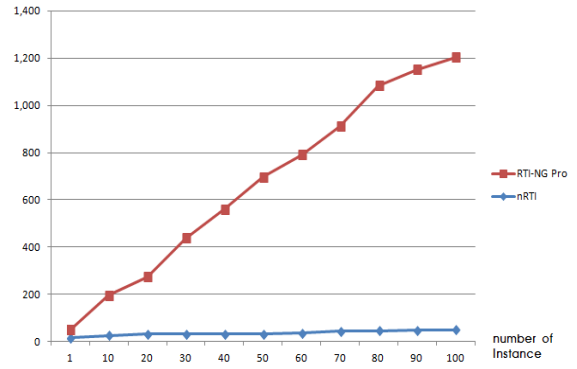

그림 15. 응답 시간
Fig. 15. Response Time

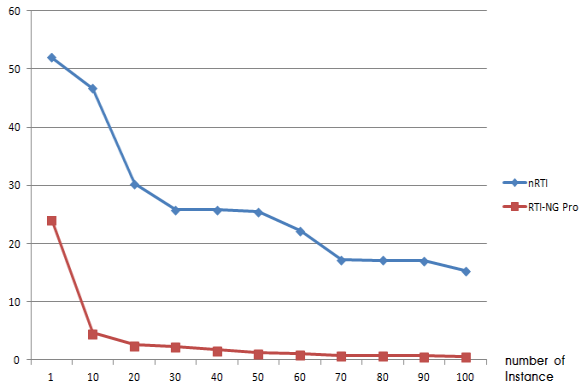

그림 16. 처리율
Fig. 16. Throughput Rate

그림 15에서 볼 수 있듯이 개발 RTI는 데이터 교환 단위인 Object Instance의 개수 증가와 관계없이 네트워크 응답시간의 변화가 거의 없음을 확인할 수 있다. 즉, 상용 RTI는 통신 시 FedExec를 거쳐 통신하는 반면, 개발 RTI는 제안된 하이브리드 구조를 사용함으로써 데이터 교환 시점에는 1:1로 통신을 수행하기 때문이다. 가령, 상용 RTI는 객체 100개를 갱신하기 위해서 송신 Federate가 FedExec에게 100번 송신하고, FedExec는 수신 Federate에게 또 다시 100번 송신해야 한다. 하지만 개발 RTI는 송신 Federate가 수신 Federate에게 직접 100번 송신하면 된다. 데이터 처리율의 경우 그림 16과 같이 감소하는 경향을 보이지만 동일 조건에서 RTI-NG와 비교하여 더욱 많은 데이터를 처리함을 보이고 있는데, 데이터 패킷 생성, 인코딩 및 디코딩 구현 시 개발 RTI가 RDK 라이브러리를 사용함으로써 RTI 서비스 간 최적화된 기능을 공유하기 때문이다.

Ⅵ. 결론 및 향후과제

최근 국방 소프트웨어 분야에서는 모델링 및 시뮬레이션 기술에 대한 관심이 높아지면서 개발 시뮬레이션의 상호 연동 및 재사용을 늘리기 위한 방안으로 분산 시뮬레이션 표준인 HLA의 적용을 권고하고 있다. HLA는 미 국방성에서 제정되었으며 IEEE에서 표준화되어 현재까지 발전하고 있다. 특히, HLA기반 시뮬레이션의 핵심 소프트웨어인 RTI는 모의 소프트웨어 간 통신 환경을 제공하기 때문에 시뮬레이션 구축을 위한 중요한 기술로 자리 잡고 있다. 한편, 국내 RTI 기술 개발은 전무한 상황으로 국내에서 사용된 RTI는 모두 외산에 의존하고 있다. 본 논문은 국내에서 최초로 개발한 IEEE1516.1-2000 표준의 RTI 구조 및 구현 방법을 기술했다. 특히, 데이터 교환 성능을 개선하기 위한 방안으로 하이브리드 데이터 통신 방법을 제안하고, 제안 방법에서 발생하는 데이터 교환 영역 동기화 문제를 해결하는 방법을 보였다. 또한 제안 방법의 성능을 검증하기 위해 상용 RTI인 RTI-NG Pro와 비교함으로써 실험 환경에서의 우수함을 보였다.

향후, 개발 RTI는 기능 및 성능에 대한 신뢰성을 향상시키기 위해서 다양한 상용 RTI와의 벤치마킹 시험이 수행되어야 한다. 특히, 성능 시험에서 구성한 일대일 방식뿐만 아니라 다수의 Federate가 정보를 교환할 경우 제안 데이터 통신 방법이 더 나은 성능을 보이는지 시험되어야 한다.

References

[1] Modeling & Simulation Coordination Office, Retrieved Oct., 31, 2012, from http://www.msco.mil

[2] Pitch pRTI, Retrived Oct., 31, 2012 fromhttp://pitch.se/products/prti

[3] HLA RTI - Run Time Infrastructure - MAK RTI, Retrieved Oct., 31, 2012, from http://mak.com/products/link-simulation-interoperability/hla-rti-run-time-infrastructure.html

[4] RTI NG Pro, Retrived Oct., 31, 2012, from http://www.raytheon.com/capabilities/products/rti/

[5] Eui Soon Kim, “Interconnection structures of U.S.-led coalition information system and implications”, *KIDA*, 2009.5.

[6] IEEE, “EEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Framework and Rules, IEEE Std 1516”, *Institute of Electrical and Electronics Engineers, New York, December 11*, 2000.

[7] IEEE, “EEE Standard for Modeling and Simulation(M&S) High Level Architecture (HLA) - Federate Interface Specification, IEEE Std 1516.1-2000”, *Institute of Electricaland Electronics Engineers*, New York, March 9, 2000.

[8] IEEE, “EEE Standard for Modeling and Simulation(M&S) High Level Architecture (HLA) - Object Model Template, IEEE Std 1516.2-2000,” *Institute of Electrical and Electronics Engineers*, New York, March 9, 2001.

[9] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, “Pattern-Oriented Software Architecture A System of Patterns”, *Wiley & sons*, 1996.

[10] Mark Hyett, “Implementation of the Data Distribution Management in the RTI-NG”, *02S-SIW-044*, March 2002

[11] Keith W. Ross James F. Kurose. “Computer Networking”, *Pearson Education*, 2009.

심 준 용 (Jun-yong Shim)

2005년 2월 우석대학교 컴퓨터공학과 졸업

2007년 2월 한양대학교 컴퓨터공학과 석사

2011년 1월~현재 LIG넥스원(주)소프트웨어 연구센터 선임연구원

<관심분야> 분산객체, HLA/RTI, 분산시뮬레이션, 이기종 연동

위 성 혁 (Soung-hyouk Wi)

2001년 2월 전남대학교 컴퓨터공학과 졸업

2011년 2월 한남대학교 국방 M&S 학과 석사

2011년 1월~현재 LIG넥스원(주)소프트웨어 연구센터 수석연구원

<관심분야> SBA, 시스템 아키텍처, 모델링&시뮬레이션