

A Distributed Interactive Combat Simulation Model

홍윤기*, 권순중*

Abstract

최근 산업체, 공공기관 등 거의 모든 분야에서 인터넷 기술을 이용한 클라이언트/서버환경으로 시스템을 구축하고 있다. 본 논문에서는 이러한 환경에서 DIS와 ADS의 개념을 이용한 전투 시뮬레이션을 수행하여 기존 모델의 비현실적인 가정과 문제의 난이도로 인한 여러 제약을 해결할 수 있으며 시간 및 시스템의 활용면에서 효율의 증가에 대한 모델을 구현하였다. 본 전투 시뮬레이션 모델은 인터넷 프로토콜의 표준으로 자리하고 있는 TCP/IP를 이용한 것이기 때문에 이동통신을 이용한 무선 데이터는 물론, 인터넷 접속이 가능한 곳이라면 어떠한 환경에서도 수행이 가능하다. 노드에서 시뮬레이션이 종료되면 여기서 발생한 생존 대수, 발사시간 분포 등의 정보는 서버에서 적절한 의사결정을 수행한 후 다른 노드로 전달하게 된다.

1. 서론

최근 국내외를 막론하고 산업체, 공공기관, 서비스 분야 등 전반적인 부문에서 기업환경이 인터넷 기술을 활용한 클라이언트/서버 환경으로 시스템을 구축하는 추세로 가고 있다. 과거 중앙 집중식 환경에서는 모든 사항을 메인 프레임에서 관리하였으므로 이로 인해 종종 시스템이 정지하거나 데이터 처리에 오류를 가져오게 되었다. 이러한 중앙 집중식 환경의 단점을 보완하기 위해 나온 개념이 분산처리환경이다. 여기서 말하는 분산처리환경이란 간단히 말해 [그림 1]과 같이 네트워크 시스템에서 자원을 공유하는 시스템의 집합을 말한다. 이는 원격지에서의 명령 처리를 말하며 원격지는 지역적, 기능적 혹은 속성과 관련되어 구분된다[1]. 이와 같은 분산처리는 특정 시스템에 가해지는 부담을 여러 시스템으로 분산

시킴으로써 운영상 처리시간의 단축과 데이터 처리에 신뢰성을 증가시킬 수 있는 것 등 많은 장점이 있다.

본 논문에서는 위에서 언급한 분산처리의 이점을 확률적 전투모형 시뮬레이션에 적용하였다. 기존의 전투모델이 지니고 있는 비현실적인 가정과 문제해결의 난이도로 인하여 근사적 접근 방법들이 사용되어 왔다. 이와 같은 문제의 해결방안으로 DIS(Distributed Interactive Simulation)와 ADS(Advanced Distributed Simulation)의 활용이 미국방성이 주도하는 국방관련 시뮬레이션 연구에서 중요한 위치와 역할을 하고 있다.

각 전투노드(이하 노드)에서 수행된 시뮬레이션 정보를 노드에서 참고하여 활용하게 된다. 이는 결과적으로 동시에 모든 상황의 정보를 시뮬레이션에 반영하게 되고 총체적

* 한성대학교 산업공학과

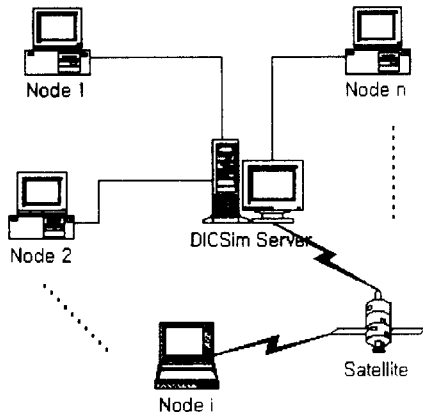


그림 1. 분산 시뮬레이션 환경

인 수행능력의 향상과 임무수행을 위한 각종 정보의 공유로 한결 수준 높은 모의실험의 결과를 기대할 수 있게 된다.

실제 전투상황이 지형, 기상상태, 각종 다양한 장비의 조합, 시간적 차이 등 결코 동일할 수 없는 상황하에서 진행되는 것을 감안한다면, 이와 같이 특정 전장의 형태가 분산 처리되어 수행되는 확률적 전투모형 시뮬레이션은 종래의 재래식 모델들을 혁신적으로 보완하고 개선 및 발전시키는 데에 중요한 계기가 될 것으로 기대된다[1]. 또한 이 연구는 특정 규모의 전투모형을 수립하는데 있어 이를 다수의 소규모 전투들의 네트워크로 간주하여 모델링하는 연구와 병행하여 발전을 모색하는 방안도 의미를 주시할 필요가 있겠다.

2. 모델

2.1. 전투 환경

임의의 지역에서 실제 전투가 벌어지고 있다고 가정하면 이 곳에는 각각의 국지전을 수행하는 지역이 있고, 이들을 관리하는 중앙지휘본부가 있다. 중앙지휘본부는 모델

의 중추적인 역할을 담당하게 되고 이는 구현되는 소프트웨어에서는 서버로 그 역할을 수행하며, 실제 전투가 진행되고 있는 지역들은 노드들로 정의한다. 여기서 말하는 노드는 실제 시스템 환경상에서 [그림 1]과 같이 네트워크로 연결된 각 클라이언트 시스템을 의미한다.

중앙 지휘본부 즉, DICSim(Distributed Interactive Combat Simulation) 서버에서는 각 DICSim 노드에서 발생하는 정보를 참고로 하여 의사결정을 한 후 특정 노드에 명령을 전달하는 역할을 한다. 노드는 DICSim 서버에서 수신한 명령을 대상으로 일련의 시뮬레이션을 수행한다.

2.2. 분산 시뮬레이션 과정

각 DICSim 서버와 노드에서 벌어지는 상황은 다음과 같다.

Step 1. 서버 활동 개시와 노드의 접속

노드는 시뮬레이션을 수행하기 전에 DICSim 서버에 접속한다. 접속 시에는 서버에서 각 노드를 구별할 수 있도록 노드 ID와 시뮬레이션 시작 시간을 서버로 전송한다.

Step 2. 노드 접속완료와 시뮬레이션 준비

DICSim 서버에서는 노드에서 전송된 ID와 시간을 보관한 후, 모든 노드의 접속이 완료되면 시뮬레이션을 시작한다.

Step 3. 시뮬레이션 시작

서버에서 시뮬레이션이 시작되면 자체 시간을 진행시켜 미리 보관된 각 노드의 시

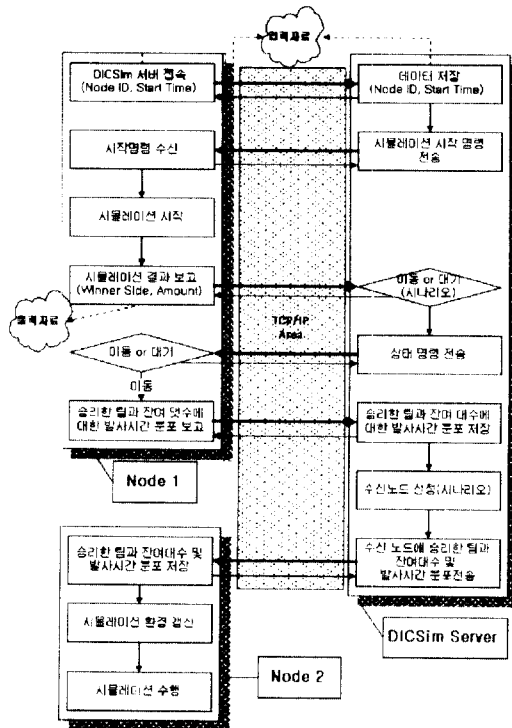


그림 2. 서버와 노드간 데이터 흐름

작시간과 비교한다. 이에 특정 노드의 시작 시간이 되면 해당 노드로 시작 메시지를 전송하여 노드에서의 시뮬레이션 시작을 명령한다.

Step 4. 시뮬레이션 결과정보의 전달

시작 명령을 받은 노드에서는 미리 설정된 시뮬레이션 상황 - A Team 과 B Team 의 초기 설정 대수, 발사시간 분포, 명중 확률 - 에 따라 시뮬레이션을 시작하며, 시뮬레이션이 종료되면 시뮬레이션 종료시간, 승리한 팀과 잔여대수 및 발사시간 분포에 관한 정보를 DICSIM 서버에 전송한다.

Step 5. 시나리오 수행 및 노드에 정보전달

DICSIM 서버에서는 수신한 정보를 대상으로 미리 설정된 시나리오에 따라 현재 시뮬레이션이 수행되고 있거나 준비중인 다른

노드에 수신한 정보를 전송한다.

Step 6. 시뮬레이션 환경 갱신

다른 노드의 정보를 수신한 노드는 이 정보를 자기 시뮬레이션 환경에 반영한 후 상황을 갱신하여 계속적으로 시뮬레이션을 수행한다.

[그림 2]는 위의 과정을 도식적으로 나타낸 것이다. 이 그림에서 보면 정보 전송노드와 정보수신노드는 분리되어 서로 다른 역할을 수행하는 것처럼 보이나, 실제로 동일한 DICSIM 노드 프로그램에서 이 기능을 수행할 수 있어야 한다. 또한 굵은 화살표는 실제 데이터의 전송을 나타내며, 얇은 화살표는 데이터의 수신에 대한 call back 의 역할을 한다. 데이터에 대한 수신 확인은 상당히 중요한 역할을 하며, 각 노드와 DICSIM 서버에서 데이터 전송에 대한 수신을 확인 후에야 비로소 다음 단계로 넘어갈 수 있다.

3. 적용 예

본 모델의 개발은 Borland 사의 Delphi 3.0 C/S Version 을 사용하였으며, 이 경우 자체적으로 제공하는 Server Socket 과 Client Socket Component 를 사용하여 DICSIM 서버와 노드간의 데이터전송을 관리하였다. 전송되는 데이터를 관리하는 프로토콜은 인터넷 프로토콜의 기본인 TCP/IP 를 이용하여 유선 네트워크망은 물론, 최근 상용화된 PCS 같은 휴대전화를 이용한 무선 통신도 가능하도록 구현하였다.

초기 인원수는 아군(A Team)과 적군(B Team) 각각 10 대로 정하며, 시나리오 1 의

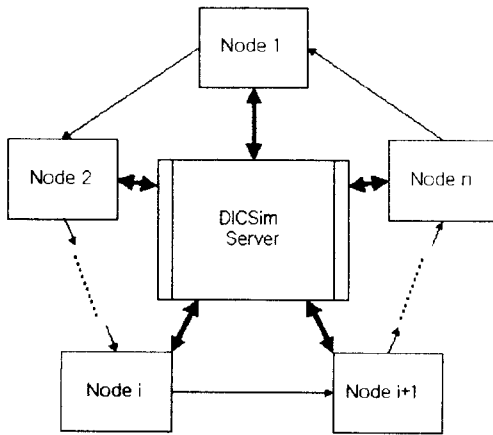


그림 3. 데이터 전송과정

노드 수는 3 개, 시나리오 2 의 노드 수는 5 개로 설정하였다. 발사분포는 Exponential 분포와 Uniform 분포에서 선택할 수 있게 하였으며 명중확률 또한 상황에 따라 결정할 수 있다. 특정 노드에서 시뮬레이션이 종료된 후 인접 노드로 이동하게 되며, 이때 이동은 전투개시 시간이 가장 빠른 노드로 정해진다.

기본적인 이동 시나리오의 상황은 [그림 3]에 나타나 있다. 노드 i에서 시뮬레이션이 종료되면 노드 i+1로 데이터를 전송하는 상황이다. 이 그림에서 굵은 선의 양방향 화살표는 물리적인 네트워크의 연결을 나타내며 상대적으로 가는 선의 단방향 화살표는 실제 데이터 흐름의 방향을 나타낸다.

이때 수신 노드에서 시뮬레이션이 진행되고 있다고 하더라도 수신된 정보들은 진행 상황에 즉시 갱신되어 새로운 환경의 시뮬레이션을 실행하게 된다. 이 상황은 실제 전투가 진행되는 상황에서 지원군이 도착하여 함께 전투를 계속 진행시켜나가는 개념과 동일하다. 만일 수신 노드에서 시뮬레이션이 종료한 후에 정보가 수신된다면 이미

상황이 종료된 후이므로 수신 노드의 승리한 팀과 동일한 팀이 도착했을 경우는 단순히 잔존 대수만 증가시키며, 상대팀이 갱신되었을 경우는 새로운 상황의 전투가 시작된다.

[그림 4]와 [그림 5]는 각각 DICSIM 노드의 실행화면과 DICSIM 서버의 실행화면이다.

4. 결론 및 추후연구

본 논문에서는 Combat Simulation 에서 기존에 수행하던 Stand alone 방식이 아닌 실제로 컴퓨터를 여러 대 분산시켜둔 후 시뮬레이션을 수행하는 방법에 대해 살펴보았다. 현재 수행하는 시뮬레이션은 과거에 수행하던 방식과 동일한 컴퓨터만으로 수행되는 Constructive Simulation 방식이다. 추후에는 DIS(Distributed Interactive Simulation)에서도 추구하고 있는 Live Simulation, Virtual Simulation, 그리고 Constructive Simulation 을 함께 포함하는 것이 바람직하며, User Interface 를 고려한 2 차원 및 3 차원 그래픽을 포함시키는 것이 필요하겠다[2]. 또한 분산 시뮬레이션에서 가장 중요한 역할을 차지하고 있는 시간 동기화(Time Synchronization) 부분에 있어서도 상당한 고려가 필요하다. 또한 DIS 와 함께 HLA(High Level Architecture)를 이용하여 시뮬레이션 모듈의 재사용성으로 시간적, 경제적 효율의 증대는 물론, 하부 시스템의 체계적인 통합에도 큰 기여를 할 수 있을 것으로 기대된다[3].

이와 동시에 전투에 참여하는 각 객체의 속성에 있어서 Heterogeneous Object 의 구현

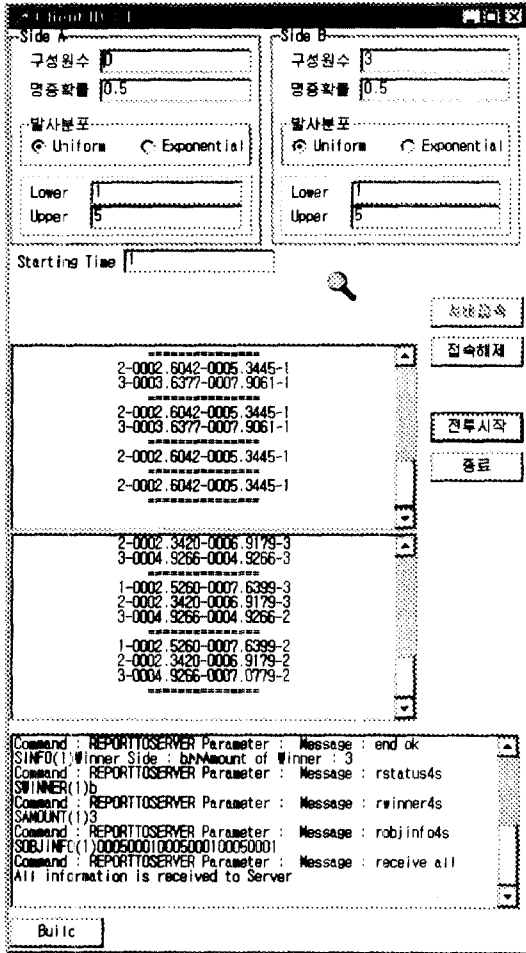


그림 4. DICSIM 노드 실행화면

은 좀더 현실적인 Combat Simulation 의 구축에 접근할 수 있을 것으로 생각된다.

5. 참고문헌

[1] John A. Hamilton, Jr., et al. "Distributed Simulation", CRC Press, New York, 1997.
 [2] Jim Sikora, Phil Coose, "What in the World is ADS?", PHALANX, Vol. 28, No 2, June 1995, pp. 1-8.
 [3] Charles E. Walters, "Use of Proto-federations to Evaluate High Level Architecture for Interoperable Simulations", PHALANX, Vol. 30,

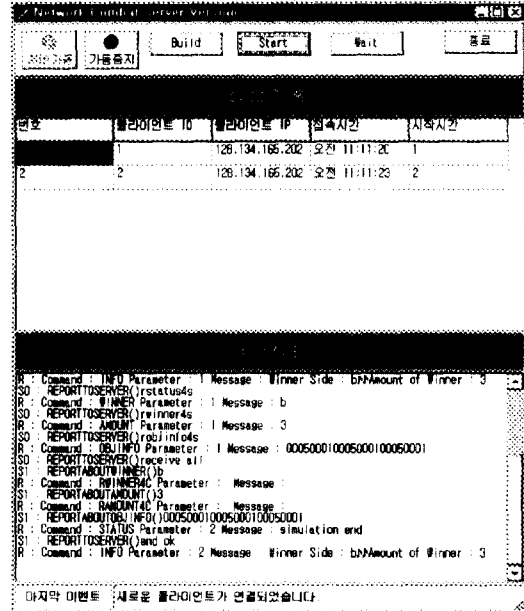


그림 5. DICSIM 서버 실행화면

No 2, June 1997, pp. 27-31.

[4] David L. Neyland, "Virtual Combat : A Guide to Distributed Interactive Simulation", 1997.
 [5] IEEE Std 1278.2, "IEEE Standard for Distributed Interactive Simulation - Communication Services", 1996.
 [6] James A. Dewar, et al, "Credible Uses of the Distributed Interactive Simulation(DIS) System", 1995.