이산사건 모델링 및 시뮬레이션 기반 LYNX-ESM 체계 시뮬레이션에 관한 연구

A Research on the LYNX-ESM System Operating and Performance Prediction Simulation Based on DEVS

신 동 조* 　　 **윤 기 천***
Shin, Dong-Cho 　　 Yun, Ki-Cheonn

ABSTRACT

This paper is to describe LYNX-ESM Simulation System to simulate for EW operating environment analysis and system performance verification of LYNX-ESM system using Discrete Event Simulation (DEVS) Methodology. This system consists of 3 PC with TCP/IP network. Each PC is loaded with Modeling & Simulation program based DEVS. Each connected program conducts EW simulation. As a result, we analyze the operating environment of the maritime EW threat, simulate the EW threat discrimination and geolocation capability, and estimate the LYNX-ESM system effectiveness before real LYNX-ESM system development.

주요기술용어(주제어) : LYNX-ESM System Development(해상작전헬기용 ESM 체계개발), DEVS(이산사건 시뮬레이션), EW Modeling & Simulation(전자전 모델링/시뮬레이션)

1. 머리말

현재 아군 함정의 전자전 위협 신호를 원거리에서 탐색하여 모함에 전송, 아군 함정의 생존성을 극대화시키는 LYNX-ESM 체계 개발 시, 체계 개발의 특성이 탐색 개발 단계 없이 바로 실용 체계 개발을 수행하여야 하므로 체계 M&S가 반드시 필요하다. 이를 위하여 해상 전자전 운용환경을 Discrete Event Simulation(DEVS) 기반의 모델링 및 시뮬레이션 시스템을 개발, LYNX-ESM 체계의 전자전 운용환경 분석 및 체계 성능 검증을 위한 전자전 모델링 및 시뮬레이션 시스템으로 3개의 PC가 상호 네트워크로 연결하여 각각 DEVS 기반의 모델링 및 시뮬레이션이 실시간 연동되어 LYNX-ESM 체계의 전자전 운용환경 분석 및 체계 성능 예측을 위한 시뮬레이션을 수행하여 체계 개발 전 체계의 성능을 예측하여 기술적 위험요소를 찾아 제거하여 실제 체계 개발 시 활용하였다.

이산사건 모델링 및 시뮬레이션 방법론인 DEVS 형식론을 군 체계 시뮬레이션을 위해 최적화하였으며 실시간 비주얼 모니터링을 위해 3차원 그래픽 소프트웨어 연동 기술을 개발하였다.

† 2006년 7월 26일 접수~2006년 11월 20일 게재승인
* 국방과학연구소(ADD)
주저자 이메일 : jdcshin@add.re.kr

현재 시뮬레이션 기법을 활용하는 큰 응용분야중의 하나가 군사용 전략 시뮬레이션 분야이다. 군사용 체계의 특성상 시스템 간 상호 연동 기능이 요구되며 각 시스템 구성요소를 개발하기 위해 많은 비용이 투자되어야 하는 국방 체계 구축에서 시뮬레이션 기법의 도입은 필수적이다.

특히, 전자전 체계 구축과정에서 적용되는 시뮬레이션 기법은 시스템 자체의 적용 가능성 확인 단계에서부터 개발 완료 후 성능 및 기능 검증단계까지 모든 개발 사이클에서 응용이 가능한 형태로 진화되고 있다. 시스템 설계 단계에서 구성 요소들의 성능 지수 추출을 위해서는 시뮬레이션 기법이 유일한 방법론이 되며 개발 시에도 각 단위 구성요소의 기능을 검증하기 위해 시뮬레이션 기법을 활용하는 것이 매우 효율적인 검증 방법이 된다.

대부분의 군사용 시뮬레이션 기법은 시뮬레이션 방법론에 기반되기 보다 시뮬레이션 모델을 프로그래밍 언어를 이용하여 개발하는 방법을 적용하여 개발되어 왔다. 최근 들어 프로그래밍 언어를 기반으로 하는 시뮬레이터 환경에서의 모델 개발 및 확장, 모델 재사용성에 대한 어려움 때문에 이론적 기반을 갖춘 전략 시뮬레이터를 개발하려는 요구가 점증하고 있다.

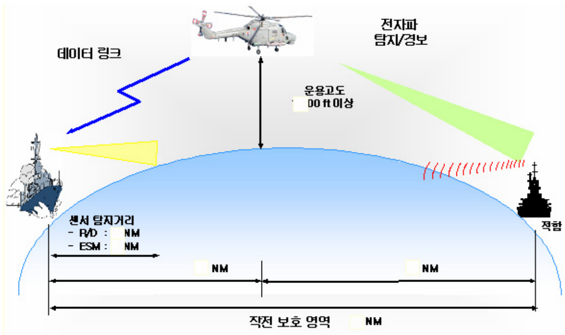
이러한 환경에서 본 연구팀은 이산사건 모델링 및 시뮬레이션 방법론을 제공하는 DEVS 형식론을 LYNX-ESM 체계 개발을 위해 구축된 전자전 실시간 분산 시뮬레이션 시스템을 연구 개발하였다.

본 연구를 통하여 LYNX-ESM 체계 시뮬레이션 시스템을 개발함으로써, LYNX-ESM 체계 개발에 지대한 공헌을 하였을 뿐만 아니라, 향후 유사한 전자전 체계 개발 시 보다 효율적으로 개발하고, 체계 개발 전 체계의 성능을 분석할 수 있을 것으로 기대된다.

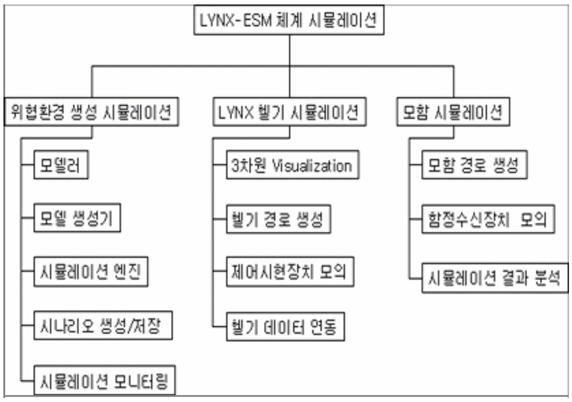
2. 시뮬레이션 체계의 구성 요소 분석

본 연구는 아군 함정의 전자전 위협 신호를 원거리에서 탐색하여 모함에 전송, 아군 함정의 생존성을 극대화 시키는 LYNX-ESM 체계 시제 개발 전, 체계의 실전 전자전 운용환경 분석 및 개발될 체계 성능 예측을 위한 시뮬레이션 수행이 요청되었다.(그림 1 참조) 이러한 체계 시뮬레이션을 통하여 실제 시제 개발 시 존재할 수 있는 개발 위험요소를 제거하였으며, 또한 새로운 요구 사항 및 기존 체계 구상의 문제점 등을 미리 검증함으로써 본 시뮬레이션 연구 필요성을 재인식하였다.

본 시뮬레이션 체계 구성은 3개의 PC로 구성된다. 위협환경을 모의하는 위협환경 생성 시뮬레이션 PC, LYNX-ESM 장치를 탑재한 헬기를 모의하는 LYNX-ESM 헬기 시뮬레이션 PC, 그리고 모함을 모의하는 모함 시뮬레이션 PC로 구성된다. 체계시뮬레이션 장치를 구성하는 주요 구성품 간의 통신은 TCP/IP를 기반으로 하여 LAN 네트워크 환경에서 이루어진다. 이렇게 구성된 분산 실시간 시스템은 실제 운용 환경



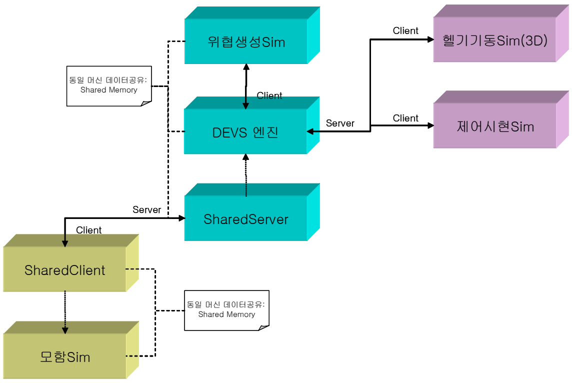
LYNX-ESM 운용개념



모델링

[그림 1] LYNX-ESM 운용개념 및 이에 근거한 모델링 요소 식별

[그림 2] 분산 실시간 체계시뮬레이션 시스템 구성



[그림 3] 체계시뮬레이션 소프트웨어 모듈 구성도

과 동일하게 LYNX-ESM 체계 성능 및 운용 환경 시뮬레이션을 수행 할 수 있다.

3. LYNX-ESM 체계 M&S 기본 구성

LYNX-ESM 체계 시뮬레이션은 전자전 교전모델에 가장 적합한 이산사건 시뮬레이션을 사용하며, 이산사건의 모델링 규격인 DEVS 형식론과 이산사건 시뮬레이션 엔진인 DEVSim++를 표준으로 적용하여 개발함으로써 모델링의 확장성 및 표준성을 유지케 하였다.

객체 지향적 시뮬레이션을 적용하여 실제 전자전 환경과 유사하게 함으로써 모델링의 타당성(Model Verification)과 시뮬레이션 결과의 적합성(Validation)을 향상시켜 전자전 운용환경 분석 및 ESM 체계의 성능 예측을 위한 시뮬레이션 체계로 개발하였다.

DEVS 형식론에 입각하여 개발된 모델의 동작특성을 이산사건에 의한 상태천이도 기술 방법을 이용하여 컴퓨터 프로그램으로 표현 하였다. 또한 이러한 시뮬레이션 모델을 적군/아군으로 편성하여 여러 개의 객체를 생성하여 시나리오를 작성하여 전자전 조우 시뮬레이션을 수행하였다.

가. 모델러

시뮬레이션 시에 사용할 새로운 객체를 생성하는 기능으로 모델의 이름을 부여하면 모델 관련 정보를 기록할 데이터 구조를 생성하도록 설계되었다. 설계된 데이터 구조에 모델러가 지정한 파라미터 들이 일괄 관리되며 모델링이 완료되면 이를 파일에 저장할 수 있는 기능이 제공된다.

나. 시뮬레이터 기본 기능

본 시스템에서 사용된 시뮬레이션의 기본 기능은 다음과 같다.

- 각각의 모델과 그 경로를 지정
- 모델 파라미터 변경 기능
- 시뮬레이션 시작/중지/일시정지 기능
- 시뮬레이션 저장/재생 기능
- 시뮬레이션 Time Schedule 조정 기능(Fast Forward/Backward)
- 시뮬레이션 도중에 모델의 상태변수 확인 기능
- 모델의 이동 경로 디스플레이 기능 등

다. 모델링

생성된 모델을 DEVS 형식론에 입각하여 모델러가 각 파라미터를 지정하는 기능을 제공한다. 초보자를 위한 Tabular 입력 형식과 고급 사용자를 위한 Text 기반의 모델링 기능이 모두 제공된다. 입력되어야 할 파라미터는 다음과 같다.

- 입력, 출력 이벤트
- 상태변수
- 내부천이함수와 외부천이함수
- 시간전진함수
- 출력발생함수

1) 모델 객체들

가) 위협환경

(1) ESM 신호원 : 레이더 및 미사일 탐색기

RF 신호
- 종류 : 고정 및 이동 레이더(해안, 지상, 함정, 항공기, 잠수함, 미사일)
- 모델링 요소 : 동적특성(위치, 고도, 속도) 및 신호특성(주파수, PRI, PW, 스캔, 출력, 주파수특성, PRI 특성, 스캔특성, 피아구분) 등.
- 최대 모의 객체 수 : 64개

(2) 레이더 표적 : 수상 및 항공 표적
- 종류 : 함정, 항공기
- 모델링 요소 : 종류(함정, 항공기), 동적특성(위치, 고도, 속도), 피아구분
- 최대 모의 객체 수 : 64개

(3) 소나 표적 : 수중 표적
- 종류 : 잠수함
- 모델링 요소 : 종류(잠수함), 동적특성(수평거리, 수평각도)
- 최대 모의 객체 수 : 64개

나) LYNX-ESM 헬기 모의

(1) LYNX-ESM 헬기 :
- 종류 : LYNX-ESM 헬기
- 모델링 요소 : 종류(LYNX헬기), 동적특성(위치, 고도, 속도, 헬기 기동정보 등)
- 최대 모의 객체 수 : 2개

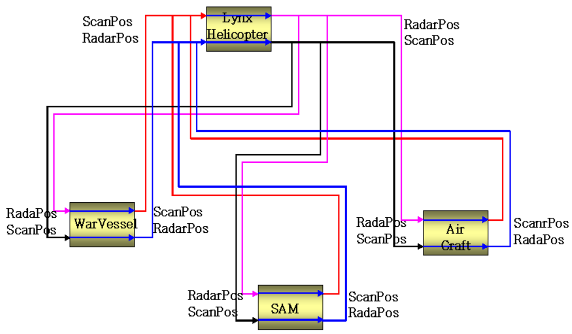
다) 모함 모의

(1) 모함 :
- 종류 : 모함
- 모델링요소 : 종류(모함), 동적특성(위치, 고도, 속도, 함 기동 정보 등)
- 최대 모의 객체 수 : 1개

2) LYNX-ESM 모델링

본 M&S 시스템에서 구현된 여러 모델중 가장 대표적인 모델인 LYNX-ESM 모델을 보다 구체적으로 살펴보면 다음과 같다.

그림 4와 같이 본 시뮬레이션을 구성하는 여러 모델 간의 관계에서 각 모델들은 일정 주기로 LynxHelicopter에 ScanPos 메시지를 전송하도록 설계 되었다. 또한 LynxHelicopter는 일정한 주기로 ScanPos 메시지를 모든 모델들에 전송한다. SAM, AirCraft, WarVessel은 ScanPos메시지를 받을 경우 LynxHelicopter에 각각 RadarPos 메시지를 전송한다. LynxHelicopter가 ScanPos 메시지를 받을 경우 해당 모델에게 RadarPos를 전송한다. 체계 전체를 구성하는 여러 모델 중 대표적으로 LYNX 헬기에 대하여 상세적으로 살펴 보면 다음과 같다.



[그림 4] LYNX-EMS 모델 구성도

- LynxHelicopter 모델링(DEVS 형식론)

  Input : RadarPos(from Any to LynxHelicopter)
  Input : SonarPos(from SubSurface to LynxHelicopter)
  Input : ScanPos(from Any to LynxHelicopter)
  Input : Stop(from Any to LynxHelicopter)

  Output : ScanPos(from LynxHelicopter to Any)
  Output : RadarPos(from LynxHelicopter to Any)

- LynxHelicopter의 상태함수
  - Y = { ACTIVE, ESM, STOP }

- LynxHelicopter의 외부변이 함수
  - Ext (ACTIVE and NOT STOP, RadarPos) = ACTIVE ; RADARFIND

  RadarPos Input Event가 발생하면 아래와 같은 로직에 의해서 해당 Radar 정보를 얻는다.

  GetRadarData(const CMessage& message)

```
{
        int nObjID,nObjIndex,nObjType;
        float fDegree,fLat,fLon;
        1. 상대의 객체 정보를 포인터로 받아들임.
        RADAR*model= (RADAR*)
message.GetSource()
        2. 상대의 위치 산출
        model->GetPositionFn(&fLat,&fLon);
        3. 모델의 고유 번호
     model->GetObjectInfoFn(&nObjID,
&nObjIndex);
        4. 모델의 유형
        nObjType=model->GetObjType();
        5. 헬기와 상대객체의 위치를 통해서 거리
계산
        float dx=fLat-m_pData->S_lat;
        float dy=fLon-m_pData->S_lon;
        float distance=(float)sqrt(dx*dx+dy*dy);
        6. 만일 Radar 검색 유효 반경인 경우 해
당 정보를 반영.
        if(m_pData->S_RadarRange>distance)
        {
        6.1 두 점으로부터 각도를 구함.
  fDegree=RAD2DEG(atan2(dx,dy));
        }
        else
        {
                fDegree=-1.0;
        }
        7. 해당 Radar의 ID를 통해서 ESM의 정
보에 데이터를 반영.
m_pData-> S_RadarDegree[nObjIndex]=fDegree;
        m_pData->S_RadarDistance[nObjIndex]=
distance;
        m_pData->S_RadarID[nObjIndex]=
nObjIndex
  }
```

• Ext (ACTIVE and NOT STOP, SonarPos) = ACTIVE ; SONARFIND

SonarPos Input Event가 발생하면 아래와 같은 로직에 의해서 해당 Radar 정보를 얻는다.

```
  GetSonarData(const CMessage& message)
  {
        int nObjID,nObjIndex,nObjType;
        float fDegree,fLat,fLon;
        1. 상대의 객체 정보를 포인터로 받아들임.
        SONAR*model=(SONAR*)
message.GetSource();
        2. 상대의 위치 산출
        model->GetPositionFn(&fLat,&fLon);
        3. 모델의 고유 번호
     model->GetObjectInfoFn(&nObjID,
&nObjIndex);
        4. 모델의 유형
        nObjType=model->GetObjType();
        5. 헬기와 상대객체의 위치를 통해서 거리
계산
        float dx=fLat-m_pData->S_lat;
        float dy=fLon-m_pData->S_lon;
        float distance=(float)sqrt(dx*dx+dy*dy);
        6. 만일 Sonar 검색 유효 반경인 경우 해
당 정보를 반영.
        if(m_pData->S_SonarRange>distance)
        {
        6.1 두 점으로부터 각도를 구함.
fDegree=RAD2DEG(atan2(dx,dy));
        }
        else
        {
                fDegree=-1.0;
        }
        7. 해당 Radar의 ID를 통해서 ESM의 정
보에 데이터를 반영.
m_pData->S_SonarDegree[nObjIndex]=fDegree;
        m_pData->S_SonarDistance[nObjIndex]=
distance;
  }
```

• Ext (ACTIVE and NOT STOP, ScanPos) =

ESM; ESMFIND

ScanPos Input Event가 발생하면 아래와 같은 로직에 의해서 해당 ESM 정보를 얻는다.

```
GetEsmData(const CMessage& message)
{
        int nObjID,nObjIndex,nObjType;
        float fDegree,fLat,fLon;
        1. 상대의 객체 정보를 포인터로 받아들임.
        ESM* model=(ESM*)
message.GetSource();
        2. 상대의 위치 산출
        model->GetPositionFn(&fLat,&fLon);
        3. 모델의 고유 번호
      model->GetObjectInfoFn(&nObjID,
&nObjIndex);
        4. 모델의 유형
        nObjType=model->GetObjType();
        5. 헬기와 상대객체의 위치를 통해서 거리
계산
        float dx=fLat-m_pData->S_lat;
        float dy=fLon-m_pData->S_lon;
        float distance=(float)sqrt(dx*dx+dy*dy);
        6. 만일 ESM 검색 유효 반경인 경우 해당
정보를 반영.
        if(m_pData->S_EsmRange>distance)
        {
        6.1 두 점으로부터 각도를 구함.
fDegree=RAD2DEG(atan2(dx,dy));
        6.2 해당 각도의 Random 함수에 의해서
확률을 반영.
                fDegree=CRandom::Normal
(fDegree,5.0,1);
                if(fDegree<0)
                        fDegree+=360;
        }
        else
        {
                fDegree=-1.0;
        }
        7. 해당 ESM의 ID를 통해서 ESM의 정보
에 데이터를 반영.
        m_pData->S_EsmDegree[nObjIndex]=fDegree;
        m_pData->S_EsmType[nObjIndex]=nObjType;
        }
```

- Ext (ESM and NOT STOP, RadarPos) = ESM ; ESMFIND, RADARFIND

위의 GetEsmData(),GetRadarData()를 참조.

- Ext (ESM and NOT STOP, SonarPos) = ESM ; ESMFIND, SONARFIND

위의 GetEsmData(),GetSonarData()를 참조.

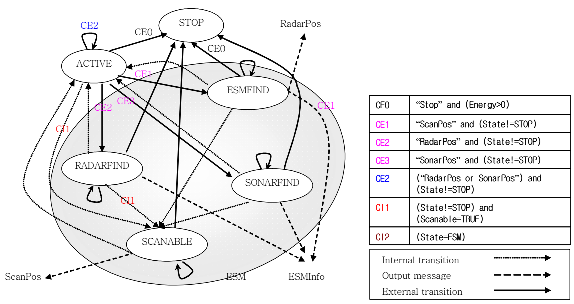
- Ext (ESM and NOT STOP, ScanPos) = ESM; ESMFIND

위의 GetEsmData()를 참조.

- LynxHelicopter의 내부 변이 함수
  - Int (ACTIVE, Energy>0) = ACTIVE
  - Int (ESM, Energy>0) = ACTIVE

- LynxHelicopter 출력함수
  - Out (ACTIVE, Scanable=TRUE) = ScanPos, ESMInfo
  - Out (ESM, Scanable=TRUE) = ScanPos, ESMInfo
  - Out (ESM,) = RadarPos

- Time Advance 함수
  - Ta(STOP) = Infinity
  - Ta(ACTIVE) = 3000
  - Ta(ESM) = 10



[그림 5] Lynx헬기 Phase Transitions Diagram

라. 분산 실시간 시뮬레이션

본 LYNX-ESM 체계 시뮬레이션에서는 3대의 PC가 연동하여 수행하도록 설계 구현 하였다. 그러한 구성 요소 중 LYNX 헬기가 3차원 환경에서 실시간으로 수행하도록 설계되었으므로, 시뮬레이션 체계를 구성하는 다른 논리적 모델들이 실시간 연동할 수 있도록 메커니즘을 제공하였으며, 시뮬레이션의 시간 전진은 실제 시간에 의존하여 진행하도록 구성되었다.

실시간으로 진행하는 모델은 입출력과 4개 모델 함수를 실시간으로 연동할 수 있도록 처리하며, 다른 모델과의 인터페이스는 변경하지 않도록 구성되었다. LYNX-ESM 체계 시뮬레이션의 모델링 기능은 모델 함수를 사용자가 직접 프로그래밍하여 적용하는 것이 가능하므로 다음과 같은 4개 특성 함수를 하드웨어 연동 시에 고려하였다.

- Internal transition function : 지정된 주기 별로 하드웨어의 상태 값을 polling한다.
- External transition function : 다른 모델로부터 전달된 이벤트를 하드웨어로 전달한다.
- Time advance function : 실제 하드웨어 동작 시간에 맞도록 설정한다.
- Output function : 하드웨어가 인식해야 할 값을 출력한다.

본 시뮬레이션 시스템에서 연동은 TCP/IP 기반으로 LAN 환경에서 구현 되었으며, 시뮬레이션 알고리즘에 의존하지 않도록 독립적으로 모델링 하였다. 이러한 기능을 제공하는 실시간 스케줄러는 최소 스케줄링 interval을 고려하여 시스템에 인터럽트를 등록한 후에 인터럽트 처리 함수에서 모델들의 스케줄에 따라 수행하도록 구현하였다.

마. 3차원 Visualization

LYNX 헬기 시뮬레이션 PC에서는 다음과 같은 성능의 3차원 Visualization을 구현하였다.

- OpenGL 기반(VTree) C++ 프로그램
- 최소24Frame/Sec 이상 3D 렌더링 속도 유지
- 지형 DB : 한반도 전역을 위성영상(20m급)을 이용하여 3차원 지형을 구축하였다. 이러한 지형 정보를 바탕으로 작전 및 임무 분석에 의한 교전 시나리오를 모의하여 컴퓨터에 의한 헬기 조우 환경을 실시간으로 시뮬레이션 할 수 있다.

4. 위협환경 생성 시뮬레이션

환경모델링 시에는 위협환경을 신규입력, 수정편집 등의 관리를 수행하며, 시뮬레이션 기간에는 헬기의 위치 정보를 받아서 해당 위협신호를 탐지거리를 기준으로 산출 선정하여 해당 위협정보를 On-line 메시지 형태로 헬기 객체에 전송한다. 시뮬레이션의 주 서버 역할을 수행하고 각종 상태변수를 기록저장하며, 재현 요구 시 재현하는 기능을 한다.

주요 기능은 다음과 같다.

- 위협환경 모델링 입력 및 수정
- 전체 시뮬레이션의 서버 기능
- 위협환경 동시 모의발생 관리
- 헬기위치 및 기동정보 On-line 입력
- 함정위치 및 기동정보 On-line 입력
- 탐지위협 산출 및 산출된 위협정보 헬기에 전달 기능(산출기준 : 탐지거리, 전파의 수평선 Line-Of-Sight)
- 시뮬레이션 중 모델 상태변수 표시
- 시뮬레이션 전체 기간 중의 모든 객체 상태 저장 및 재현

5. LYNX 헬기 시뮬레이션

Lynx 헬기 시뮬레이션 PC에서는 2대의 헬기의 이동 경로를 실시간으로 조이스틱을 이용하여 3차원 공간상에서 기동 및 이동 경로를 생성하며, 위협환경 생성 시뮬레이션 PC에서 생성된 전자전 대상위협 신호들을 헬기에 장착된 Radar, Sonar, ESM 등의 장비 모의를 통해 얻어진 각 데이터들을 헬기 디스플레이 모의로 도시하며, 상기 장비로 얻어진 데이터와 GPS, GYRO, 항법장비 등의 모의 데이터를 모함으로 전송하는 시뮬레이션 기능을 수행한다.

- 최대 2대의 헬기객체의 위치정보를 모의한다. (위도, 경도, 고도, 속도)

- 2대 중 1대는 운용자에 의해 선택적으로 수동조정 비행이 가능하다.(조이스틱 사용)
- 헬기 위치정보 및 기동정보를 시뮬레이션 서버로 On-line 전송한다.
- 제어 시현 모의 화면에 위협들을 도시한다.
- 수신된 위협정보와 헬기 위치정보를 통합한 데이터링크 전송정보를 모함객체로 On-line 전송한다.
- 3차원으로 비행상태 및 지형상태를 시현한다.

[그림 6] LYNX 헬기 시뮬레이션 화면

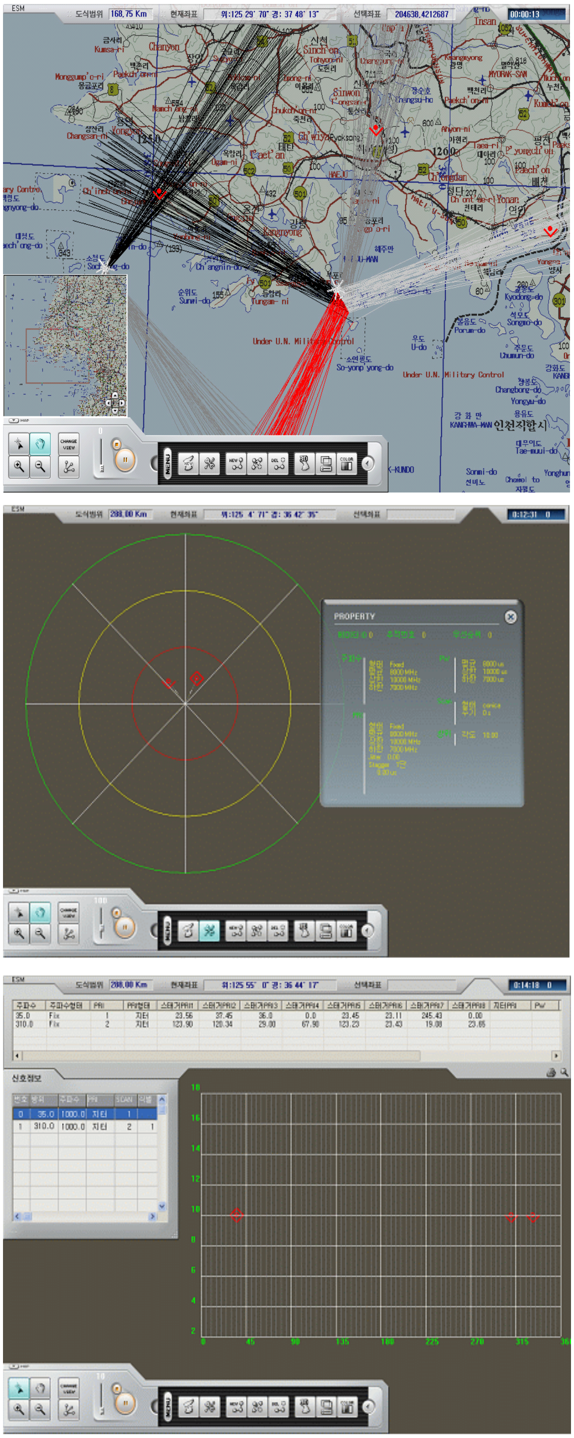
[그림 7] 체계 시뮬레이션을 수행 후 실제 체계에 적용된 제어시현장치 화면

- 수신된 위협 정보군을 시뮬레이션 Node 별로 시간정보와 함께 저장한다.
- 시뮬레이션이 종료되면 다음과 같은 성능분석 자료를 출력한다.
  ▷ Excel 출력기능 : 시뮬레이션시간, 주파수, PRI, PA, PW, SCAN 정보를 Excel 차트로 출력
  ▷ 누적 PRI 도시 기능 : 각 Node 별 위협신호의 PRI 합계를 도시
  ▷ 주파수 분포 도시기능 : 각 Node 별 위협신호의 주파수 분포를 도시
- 그림 7은 본 체계 시뮬레이션을 수행하여 최적의 운용환경을 분석 한 후 실제 체계에 적용된 사례이다.

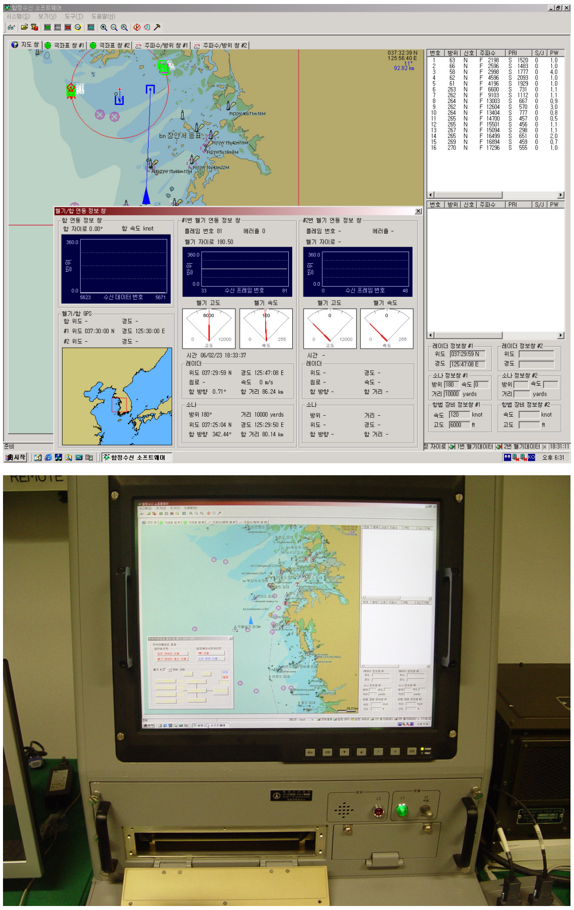
6. 모함 시뮬레이션

모함 시뮬레이션 PC에서는 해상작전(LYNX)헬기의 시뮬레이션 결과의 수신 및 Display 기능을 수행하며, 해상작전(LYNX)헬기 시뮬레이션 PC에서 전송되어온 데이터를 모함전시기에 시현한다. 시간 전진에 따른 모함의 이동 경로를 2차원으로 지도위에 도시하고, LYNX 헬기로부터 전송되는 위협 요소 데이터를 근거로 각종 디스플레이를 구현하였다. 시뮬레이션이 종료되게 되면 LYNX-ESM 헬기의 각종 효과도를 분석한다.

- 모함의 이동 경로 생성, 2차원 도시 기능
- 시뮬레이션 시에 모함의 위치를 서버에 On-Line 전달
- 헬기로부터 전송받은 데이터링크 정보를 저장 및 시현
- 시현의 재현기능
- 방향정보의 교차점분석에 의한 위협위치 추정 산출분석 기능(위치 표정(Geo Location)용)
- 시뮬레이션 전 구간의 위협 표시 기능(비행궤도 분석용)
- 그림 9는 체계 시뮬레이션을 수행하여 최적의 운용환경을 분석 한 후 실제 체계에 적용된 사례이다.



[그림 8] 모함 시뮬레이션 화면 - 지도창, 극좌표창, 주파수/방위창



[그림 9] 체계 시뮬레이션을 수행 후 실제 체계에 적용된 체계 함정 수신 장치 화면

7. 결 론

본 논문에서는 이산사건 모델링 및 시뮬레이션 방법론(DEVS)을 LYNX-ESM 체계 구축을 위해 적용한 결과를 기술하였다. 실제 체계 운용상 필수적인 3가지 요소(LYNX헬기, 모함, 위협환경)를 분류하여 시뮬레이션 시스템을 구성하였고, 이러한 모듈 간 상호작용을 위해 실시간 시뮬레이션을 수행하도록 구성하였으며, 실제적으로 LYNX-ESM 체계 개발에 적용된 연구내용을 기술하였다. 이러한 연구 개발 활동을 통하여 전장 위협 신호 환경을 분석하였고, 방탐 정확도에 따른 위협 식별 및 표정능력 예측 시뮬레이션을 수행하여 방탐 정확도에 대한 개선 방향을 제시

하였으며, 헬기 기동과 위협 위치 등에 따른 LYNX-ESM 체계 성능 분석을 할 수 있었다. 본 연구를 통하여 개발된 해상 전자전 요소 모델의 개수는 총 7종이며, 실제 체계 개발 전 다양한 설계 요구 사항을 최적화하고 검증할 수 있었으며, 탐색개발 없이 바로 체계 개발을 수행하여 체계 개발 기간을 단축할 수 있었으며, 비용 절감 효과 또한 크다고 판단할 수 있다. 또한 본 연구를 통해 개발된 3차원 가시화 프로그램은 체계 시험에서 방탐 성능시험 장치로 직접 활용하였으며, 교육 훈련 및 임무 계획용으로 확장 개발 가능하고, 향후 유사한 전자전 체계 개발 시 재활용성이 크다고 판단된다.

참 고 문 헌

[1] 김탁곤, "이산사건 시스템 모델링 시뮬레이션 기법", *전자공학회지, vol. ~19.* pp.105~114, Jan. 1992.

[2] 윤기천, "이산사건 형식론을 이용한 전자전 조우 모델링 연구", *국방과학연구소 연구 보고서*, Jul. 1998.

[3] B. P. Zeigler, *Multifacetted Modeling and Discrete Event Simulation*, Academic Press, Orlando, EL, 1984.

[4] DMSO, *DoD HLA RunTime Infrastructure Programmer's Guide*, 1997.

[5] Myung S. Ahn, *Hybrid modeling/simulation methodology for simulation speedup and preserved accuracy*, Ph.D Thesis, EE, KAIST, 1996.

[6] Tag G. Kim 외, *DEVSim++ User's Manual*, Thechnical Report, TR-CORE94-1, EE, KAIST, 1994.