UML 2.0 모델 기반의 교전통제 소프트웨어 아키텍처 개발

Development of the Engagement Control Software Architecture Based on UML 2.0 Model

유 명 환* 배 정 일* 신 진 화* 조 길 석*
Yoo, Myong-Hwan Bae, Jung-Il Shin, Jin-Hwa Cho, Kil-Seok

ABSTRACT

The engagement control software embedded in the weapon control computer of the fire control center for air defense missile system is large-scale real-time software. The use of typical software development methodologies is not appropriate to develop such large-scale embedded software in terms of reusability, reliability, and productivity for the reason that it is significantly complicated, and highly dependent on hardware platforms and developers. In this paper, a model-based software architecture using components based on UML 2.0 for the engagement control software is presented in order to solve these problems. This software architecture is verified using the black-box test, the scenario-based test, and the Ethernet packet monitoring test methods. The results demonstrate that the developed software architecture can be employed to enhance reusability, maintainability, and productivity of large-scale embedded software.

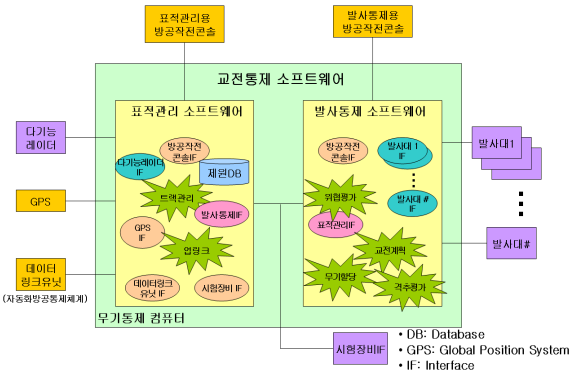
주요기술용어(주제어) : UML 2.0, Engagement Control, Software Architecture, Software Model, Model Based Architecture, Embedded Software

1. 머리말

대공유도무기체계(Surface-to-Air Missile Weapon System) 사격통제소(Fire Control Center)는 상위지휘체계(High Echelon)인 자동화방공체계(自動火防空體系)와 전술데이터링크(Tactical Data Link)로 연동되어 상위지휘체계의 지휘통제 하에서 지능화된 교전결심 지원과 자동화된 교전통제를 수행하며, 트랙을 탐지·추적하는 센서체계와 다수의 유도무기체계를 통제하여 다표적 동시교전을 수행하는 작전통제소(Operational control Center)이다.

사격통제소에 탑재되는 교전통제 소프트웨어(Engagement Control Software)는 그림 1과 같이 센서체계(다기능레이더)의 통제 및 트랙 정보 수집, 다수 발사대의 통제, 각 발사대에 탑재된 유도탄의 동시 발사통제, 전술데이터링크 장비 통제 및 상위지휘체계로의 명령 수신을 처리한다. 또한, 자동화된 교전절차, 운용상태 통제, 운용자의 의사결정을 지원하기 위한 트랙관리, 위협평가, 최적의 무기할당, 교전계획 수립, 격추평가 등의 알고리즘들이 내장되고 다기능

† 2007년 9월 11일 접수~2007년 10월 18일 게재승인
* 국방과학연구소(ADD)
주저자 이메일 : ymh@add.re.kr



[그림 1] 교전통제 소프트웨어의 구성도

레이더를 포함한 다수의 부체계(subsystem) 또는 컴퓨터들과 연동되어야 한다. 사격통제소의 교전통제 소프트웨어는 복잡도가 높은 대규모의 내장형 실시간 소프트웨어이다.

이와 같이 교전통제 소프트웨어는 부체계 및 장비와의 연동과 계산 복잡도가 높은 알고리즘들을 포함하므로 시스템 분석 및 설계에 다음과 같은 어려움이 있다. 첫째, 소프트웨어 요구조건이 방대하여 구조가 매우 복잡하므로 구조적 측면의 구성요소 식별·분석·설계 및 구성요소간의 연관관계 분석·설계 등에 정형화된 기법의 적용이 요구된다. 둘째, 기술 분야가 다양한 다수 개발자가 참여하며, 개발자간 원활한 정보공유가 요구된다. 셋째, 개발기간동안 기능적 요소에 대한 사용자 요구조건 변경이 빈번함에 따라 소프트웨어 변경을 원활하게 하며, 장비간 통신 방법과 타겟 컴퓨터(target computer)의 변경에 보다 유연하게 대처할 수 있어야 한다. 마지막으로, 무기체계 개발 패러다임 변경으로 인해 COTS(Commercial Off The Shelf) 하드웨어 및 소프트웨어 기술을 활용한 저비용의 개발이 필요하다.

그러나 기존의 내장형 실시간 소프트웨어 개발 방법은 다음과 같은 문제점이 있다. 첫째, 코드 기반의 개발 방법은 설계와 구현이 분리되지 않고 장비에 적합한 최적의 설계는 가능하나 타겟 컴퓨터에 의존성이 강하고 개발자의 능력에 따라 품질이 좌우되며, 확장성·유지보수성·재사용성이 떨어진다[1]. 둘째, 개발자간의 정보 공유가 기술자료와 소스코드 기반으로 이루어져 오류가 발생할 여지가 많이 존재한다. 이와 같은 오류는 통합하여 시험하는 단계에서 대부분 발견되기 때문에 수정보완에 많은 자원이 소요된다[1,2]. 셋째, 상용 소프트웨어(Commercial Software) 설계에 많이 사용되는 UML 1.x[2](Unified Modeling Language)의 모델 설계 방법은 소프트웨어 아키텍처(Software Architecture) 측면의 설계가 용이하지 않으며, 내장형 실시간 소프트웨어의 특성인 동시성(concurrency)을 위한 다중 쓰레드(multi-thread)와 이벤트 기반 방식(Event driven method)에 대한 분석·설계가 어렵다[3]. 또한 UML 1.x는 분석·설계 위주의 모델링 언어로써 실질적으로 소프트웨어 설계와 구현이 통합되지 않아 생산성이 높지 않다.

본 연구에서는 앞서 언급한 문제점을 극복하고 대규모의 내장형 실시간 소프트웨어인 교전통제 소프트웨어를 개발하기 위하여 UML 2.0[4~6] 모델 기반의 교전통제 소프트웨어 아키텍처를 제안하였다. 제안한 소프트웨어 아키텍처를 기반으로 교전통제 소프트웨어를 분석, 설계, 구현을 하였다. 또한, 기능 검증을 위한 시나리오 기반의 시험을 실시하여 성능을 입증하였으며, 소프트웨어의 재사용성을 제고시켰다.

본 논문의 구성은 2장에서는 교전통제 소프트웨어 아키텍처 모델링 과정과 재사용 가능한 컴포넌트(component) 설계 방법에 대해서 설명하며, 3장에서는 교전통제 아키텍처와 컴포넌트를 통합하여 구현한 교전통제 소프트웨어의 시험 결과를 기술하였으며, 4장에서는 개발한 아키텍처와 컴포넌트의 재사용성에 대해서 분석하였으며, 마지막으로 이에 대해서 결론을 맺는다.

2. UML2.0 모델 기반의 소프트웨어 설계

UML 2.0 모델 기반의 설계방법은 분석·설계 도메인에 적합한 모델링이 가능하다. 또한, 내장형 소프트웨어에 적합한 다양한 관점의 모델링 언어를 제공해 주고 있다[4]. 교전통제 소프트웨어는 UML 2.0에서 제공해주는 다양한 모델링 언어를 적용하여 하드웨어와 운영체제의 변경에 유연하게 대처가 가능한 PIM(Platform Independent Model)[3]로 설계하여 이기종 컴퓨터 환경에서도 설계된 모델을 쉽게 적용할

수 있도록 하였다. 교전통제 소프트웨어 아키텍처를 단순화시키고 알고리즘의 성능개량·확장성이 유리하도록 모델링 기법을 적용하였다.

가. 교전통제 소프트웨어 아키텍처 설계

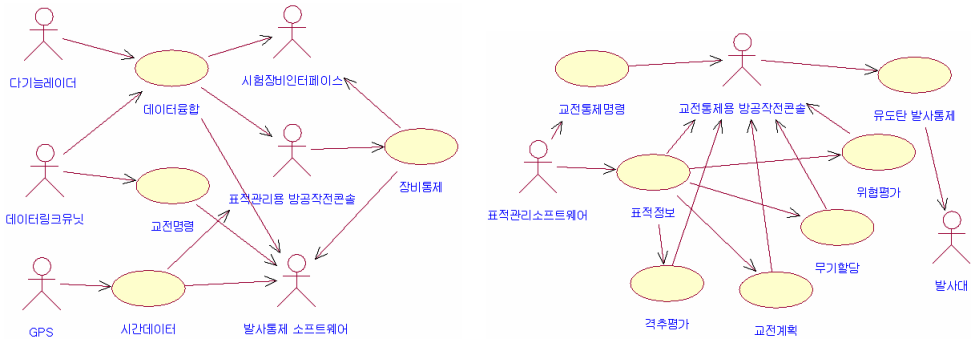
1) 요구조건 분석 모델링

UML의 유즈케이스 다이어그램(usecase diagram)을 활용하여 교전통제 소프트웨어 기능요소와 외부 연동요소를 식별하였다. 그림 2에서와 같이 기능요소는 유즈케이스로, 외부 연동요소는 행위자(actor)로 표시하였다.

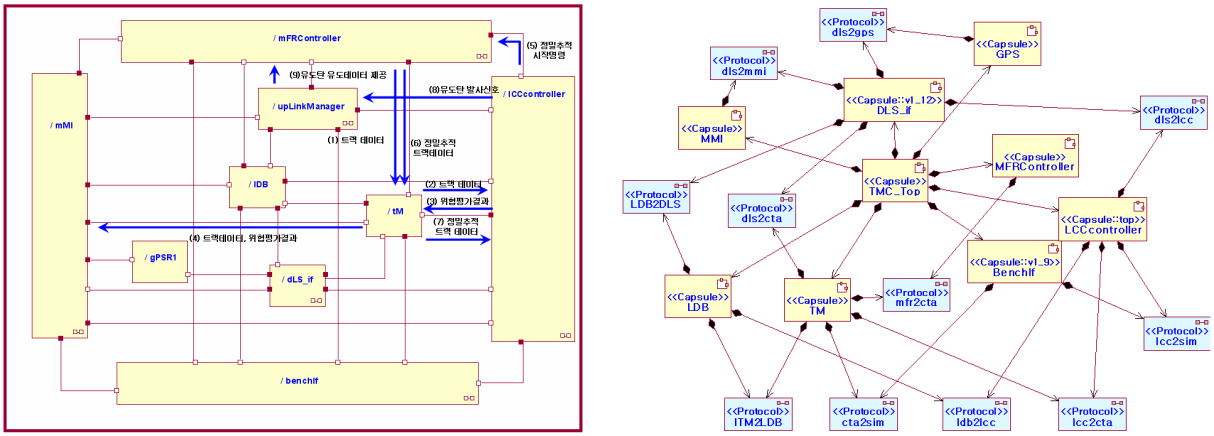
2) 정적 구조 설계 모델링

소프트웨어 아키텍처는 시스템 구성요소의 유기적 결합을 위해 구조적 연관성과 상호작용을 정의해 놓은 추상화(abstraction)된 프레임으로 개발 원칙과 지침을 제공함으로써 일관성 있는 소프트웨어 개발을 가능하게 한다[7,8].

그림 3은 교전통제 소프트웨어의 일부인 표적관리 소프트웨어의 아키텍처를 나타내는 구조도와 클래스 다이어그램이다. 표적관리 소프트웨어의 주요 실행흐름은 다음과 같다. 다기능레이더로부터 수신한 트랙데이터는 다기능레이던 연동 컴포넌트(mFRController)을 통해 트랙관리를 책임지는 컴포넌트(tM)으로 전달되고(1) 트랙처리 후 발사통제 소프트웨어와의 연동을 담당하는 컴포넌트(lCCcontroller)를 통해 발사통제 소프트웨어로 전달하면(2) 위협평가결과를 받아(3) 저장한 후 운용자에 알려주기 위해 운용자 컴퓨



[그림 2] 교전통제 소프트웨어의 유즈케이스 다이어그램

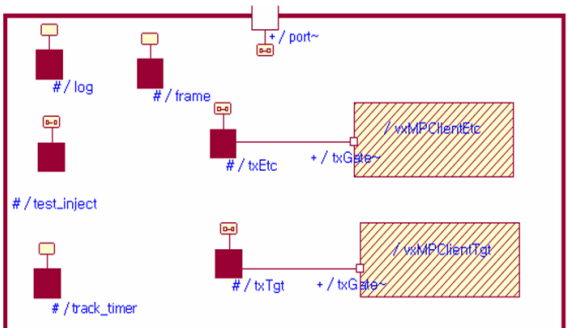


[그림 3] 표적관리 소프트웨어 아키텍처의 구조도(structure diagram)와 클래스 다이어그램(class diagram)

터 인터페이스 컴포넌트(mMI)로 전달(4)한다. 발사통제 소프트웨어에서 교전을 하기위해 트랙에 대해서 정밀추적 명령을 내리면(5) 정밀추적된 트랙데이터를 수신(6)한다. 발사통제 소프트웨어에서는 수신한 정밀추적된 트랙데이터(7)에 대해서 유도탄 발사를 수행한 후 유도탄 발사신호를 유도탄 유도 컴포넌트(upLinkManager)에게 보내면(8) 유도탄의 유도 데이터를 생성하여 다기능레이더에게 전송(9)한다.

교전통제 소프트웨어 아키텍처에서는 이러한 일련의 처리 순서와 데이터만을 정의하고 각 구성요소들의 세부 동작 및 기능은 각 컴포넌트에게 위임(delegation)시켰다. 그림 3의 구조도와 같이 장비연동 컴포넌트와 알고리즘 관련 컴포넌트들의 유기적 관계만을 정의하였다. 구조도의 각 컴포넌트들의 연관관계(relation)를 나타내는 실선( )은 그림 3의 클래스 다이어그램에서 <<Protocol>> 스테레오타입의 프로토콜 클래스(protocol class)의 인스턴스(instance)인 포트(port)를 나타낸 것이다. 각 컴포넌트들은 프로토콜 클래스를 매개체로 하여 연관관계를 가지며 프로토콜 클래스에서는 교환되어야할 데이터 및 시그널을 정의하였다. 이를 통해 각 컴포넌트의 독립성을 보장할 수 있으며, 프로토콜 클래스로 상호작용이 정의되기 때문에 컴포넌트 변경이 이루어져도 서로 영향을 주지 않으며, 유지보수가 용이하다.

또한, 교전통제 소프트웨어 아키텍처는 독립적으로 실행이 가능하고 재조립이 가능한 UML 2.0의 활성 클래스(active class)로 설계하고 각 활성 클래스 사이의 관계는 프로토콜 클래스의 인스턴스인 포트로 연결시켜 각각의 활성 클래스들 사이의 인터페이스가 가능토록 하고, 상호교환 데이터 및 이벤트들은 프로토콜의 시그널(signal)과 데이터 클래스로 정의하였다. 각 활성 클래스의 내부에도 각 세부 기능에 따라 내부 활성 클래스를 식별하여 역할 재분배 및 계층적 구조[6,8,9]를 통해 내부 구현을 숨기고, 하드웨어 의존성을 낮추었다. 또한 다중 쓰레드 관리를 위해 교전통제 소프트웨어 아키텍처에서는 그림 4의 빗금 표시 부분과 같이 논리적 쓰레드(logical thread)를 정의하여 관리하고 실제 쓰레드(physical thread)는 컴파일 시점에 결정되도록 하여 쓰레드의 분석 및 재구성이 용이하도록 하였다.

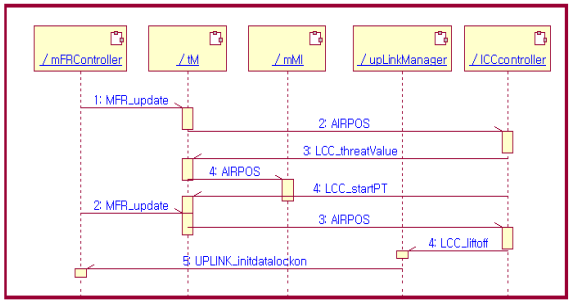


[그림 4] 활성 클래스의 논리적 쓰레드 정의

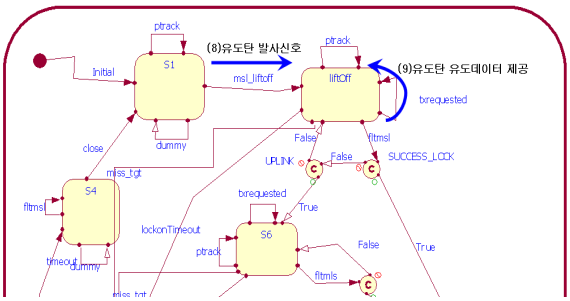
3) 동적 구조 설계 모델링

동적인 측면에서 각 활성 클래스들의 행동(behavior)를 설계하기 위해 그림 5와 같이 시퀀스 다이어그램(sequence diagram)을 활용하여 각 활성 클래스들 사이에 교환되는 메시지의 종류, 시간적 순서 등을 분석·설계하였다. 앞 절에서는 설명한 구조도와 클래스 다이어그램은 각 컴포넌트들의 연관관계 및 교환데이터에 대해서 정의하였다면 그림 5는 주요 흐름을 각 시그널을 시간 순서별로 도식한 시퀀스 다이어그램이다.

이를 바탕으로 하여 연동장비에서 발생한 이벤트와 메시지의 교환을 분석하였으며, 이렇게 분석된 이벤트와 메시지 교환에 따라 활성 클래스의 내부 상태를 도출하고 정의하였다. 활성 클래스 수준에서 이벤트와 메시지의 동적 처리를 위해 정형화 도구인 FSM (Finite State Machine)[10]을 적용하였다. 이를 통해 소프트웨어 활성 클래스의 동적 처리뿐만 아니라 소프트웨어 내부의 각 활성 클래스의 상태 천이에 대하여 분석·설계하였다. 그림 6은 그림 3과 그림 5에서



[그림 5] 각 컴포넌트의 시퀀스 다이어그램



[그림 6] 유도탄 포착·교신 알고리즘 컴포넌트의 Finite State Machine

설명한 컴포넌트들 사이의 상호작용 중 유도탄의 발사신호와 유도데이터 제공에 처리를 담당하는 컴포넌트인 upLinkManager의 이벤트 처리방법을 FSM으로 도시한 실례이다.

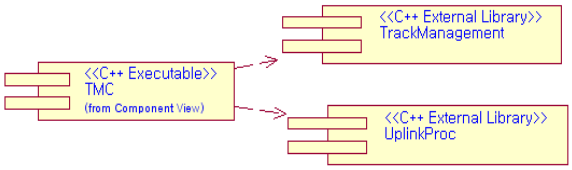
나. 재사용 가능한 컴포넌트의 설계

교전통제 소프트웨어의 개발에 필요한 알고리즘, 장비간 연동 인터페이스 및 IPC(Interprocessor Communication) 메커니즘을 재사용 가능한 컴포넌트로 설계하였다. 이렇게 설계된 컴포넌트는 교전통제 소프트웨어 아키텍처의 부품처럼 필요시 교체가 가능하다. 컴포넌트는 운용통제 및 이벤트 처리와 같은 로직(logic)과 알고리즘 또는 하드웨어 의존적인 부분을 분리할 수 있도록 계층적 구조로 설계하여 타겟 컴퓨터에 독립적이고, 알고리즘이 변경되더라도 상호 영향을 미치지 않도록 하였다.

1) 알고리즘 컴포넌트의 설계

교전통제 소프트웨어에서 요구되는 트랙관리, 위협평가, 무기할당 등의 알고리즘은 각종 이론에 의한 공학적 검증과 지속적인 성능개선이 필요하다. 교전통제 소프트웨어 아키텍처에서는 알고리즘 내장을 위한 인터페이스만을 제공하고 알고리즘은 별도의 라이브러리로 개발하기 위하여 그림 7과 같이 별도의 컴포넌트로 구성하였다.

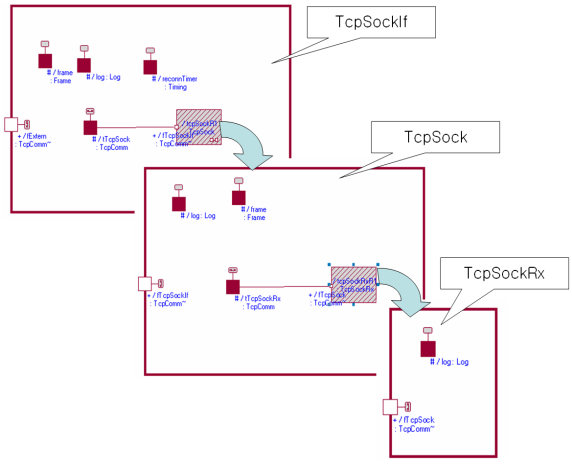
이와 같이 알고리즘 관련 컴포넌트를 설계함으로써 알고리즘 개발자와 교전통제 소프트웨어 개발자의 독립적 설계가 가능토록 하며, 소프트웨어 구조를 단순화 시켰다.



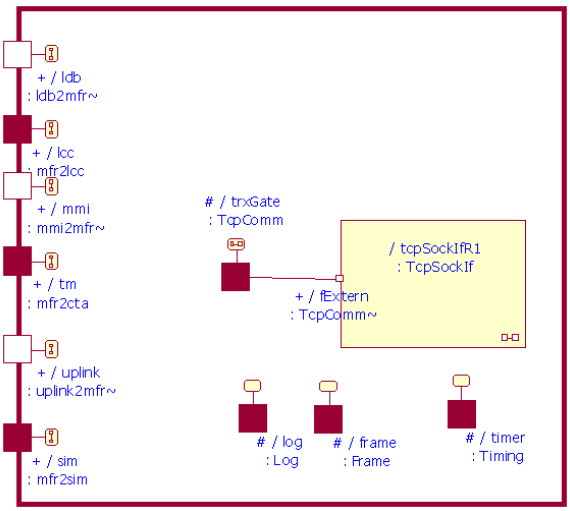
[그림 7] 알고리즘의 별도 라이브러리 연동을 위한 컴포넌트 다이어그램

2) 장비 간 연동 컴포넌트

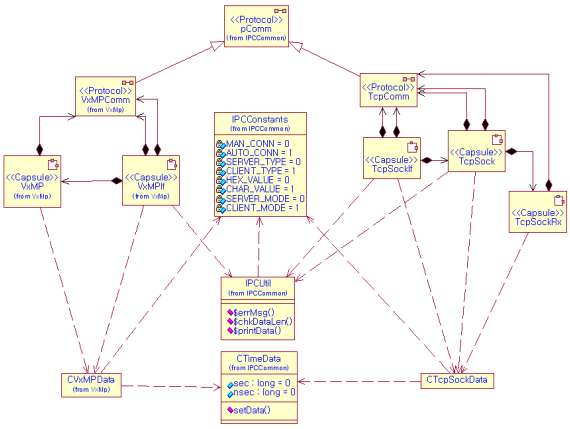
교전통제 소프트웨어는 다양한 장비와의 연동을 하기 위하여 장비 특성과 통신 매체에 따라 다양한 프로토콜을 지원하는 IPC 메커니즘이 필요하다. 그림 8과 같이 장비간 연동을 하는 컴포넌트와 통신을 담당하는 IPC 메커니즘 컴포넌트를 계층적 구조로 설계하였다. TcpSockIf 활성 클래스는 TCP/IP의 통신 개설, 재접속, 데이터 송신주기를 담당하고, TckSock 활성 클래스는 TCP/IP(Transmission Control Protocol /Internet Protocol)의 API(Application Programming Interface)를 사용한 통신개설을 담당한다. 그리고 TckSockRx는 데이터 수신을 담당한다. 그림 9의 다기능레이더 연동 컴포넌트와 같이 IPC 메커니즘을 활용하는 각 연동 컴포넌트는 연동을 위한 운용통제 로직만을 전담하고 통신부분은 IPC 메커니즘 컴포넌트에 위임시킴으로써 운용통제 로직과 IPC 메커니즘을 분리시켜 타겟 컴퓨터에 종속적인 IPC 메커니즘의 변경에 유연하게 대처하도록 하였다.



[그림 8] TCP/IP IPC 단계별 구조도



[그림 9] 다기능레이더 연동 컴포넌트의 구조도



[그림 10] VxMP와 TCP/IP IPC 클래스 다이어그램

IPC 메커니즘 컴포넌트는 상위의 장비간 연동 컴포넌트가 통신매체, 프로토콜, 통신 개설 관리, 데이터 전송 주기와 무관하게 동작할 수 있고 공용으로 사용할 수 있도록 설계하였기 때문에 각종 장비 연동 컴포넌트에서는 IPC 메커니즘 변경에 유연하게 대처가 가능하다. 즉, 그림 10에 도시된 VxMP와 TCP/IP 인터페이스 클래스 다이어그램에서 알 수 있듯이 서로 다른 IPC 메커니즘임에도 불구하고 다른 컴포넌트와 인터페이스 되는 프로토콜 클래스를 pComm으로 추상화시키고 이를 상속(inheritance)받는 유사한 구조로 설계함으로써 pComm을 상속받은 VxMpComm과 TcpComm의 프로토콜을 인스턴스로 사용하기 때문에 IPC 메커니즘이 변경되더라도 이를 사용하는 장비 연동 컴포넌트에 영향이 미치지 않도록 설계하였다.

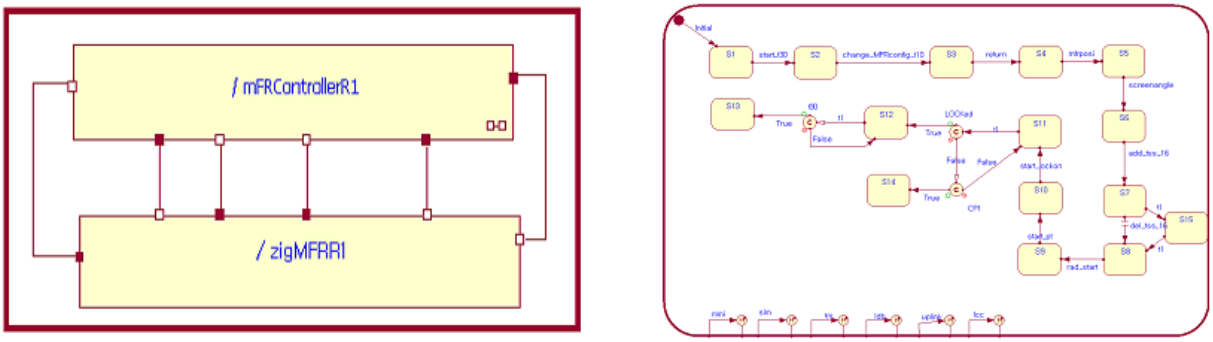
3. 교전통제 소프트웨어의 시험 및 결과

개발된 각 컴포넌트들을 교전통제 소프트웨어 아키텍처에 맞추어 통합하였다. 이렇게 완성된 교전통제 소프트웨어는 내장형 실시간 소프트웨어로 기능요소뿐만 아니라 시간 정확성과 같은 비기능 요소 성능까지 측정되어야 한다. 대부분의 내장형 소프트웨어가 탑재되는 장비에는 전시장치·분석장치가 없음으로 소프트웨어의 동작 상태를 실시간으로 파악하지 못한다. 외부 이벤트 반응시간 및 통신주기 측정을 위해 별도의 시험코드를 삽입할 경우 소프트웨어의 실시간성을 저해할 뿐만 아니라 소프트웨어 복잡도를 증가시키는 문제점이 있다.

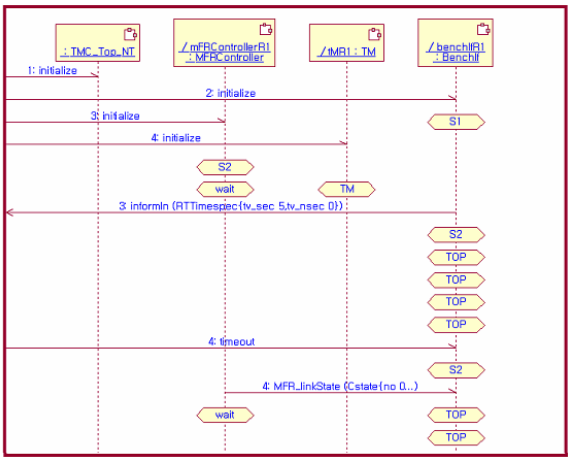
이를 해결하기 위해 블랙박스(black box) 시험방법[11]과 시나리오 기반의 검증기법[12]을 활용하여 이벤트 처리, 컴포넌트들의 동작 순서, 데이터의 흐름을 검증하였으며 소프트웨어 단위시험(unit test) 및 통합시험(integration test)에 적용하였다.

가. 기능 검증을 위한 시나리오 기반의 시험

컴포넌트 단위시험을 위해서 FSM으로 시험 시나리오를 작성하였으며, 이 시나리오에 따라 이벤트를 발생하여 이에 대한 입출력 데이터의 정확성을 검증할 뿐만 아니라 컴포넌트의 처리상태를 시험하였다. 그림 11은 다기능레이더와의 연동을 위한 컴포넌트(mFRControollerR1)의 단위시험을 하기 위한 시험치구(zigMFRR1)의 구조도와 FSM이다. zigMFR 활성 클래스에서는 mFRControllerR1의 단위시험을 위해 mFRControllerR1과 연결되는 모든 컴포넌트의 역할을 담당하면서 mFRControllerR1에서 출력되는 시그널을 수신하고 mFRControllerR1으로 입력 시그널로 FSM에 따라 다기능레이더의 초기화 정보를 제공하고 이후 트랙의 탐지, 정밀추적 시작, 유도탄 유도데이터 제공의 일련의 순서를 거치면서 이때



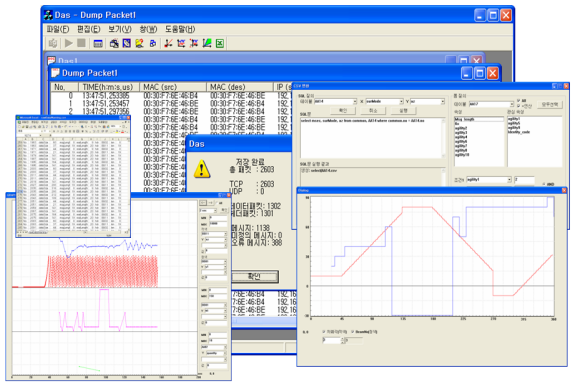
[그림 11] 다기능레이더 연동 컴포넌트의 단위시험을 위한 구조도와 시험치구(zigMFRR1)의 FSM



[그림 12] 교전통제 소프트웨어 실행으로 생성된 시퀀스 다이어그램

mFRControllerR1에서 출력되는 시그널과 그 값들을 비교하여 상태천이의 과정이 정상적으로 단계별로 이루어지면서 단말 상태까지 천이가 이루어지는지를 확인하였다. 이를 통해 출력값의 정확성과 처리과정을 검증하였다.

교전통제 소프트웨어의 통합시험은 시퀀스 다이어그램을 활용하여 시나리오 기반의 검증 방법을 사용하였다. 그림 12는 소프트웨어 실행 후 각 컴포넌트가 정해진 시간에 초기화되고 다기능레이더와의 연결상태를 보고하는 시퀀스 다이어그램으로 소프트웨어 실행 중에 각 컴포넌트 간에 교환되는 시그널들을 모니터링하여 작성한 것이다. 이 시퀀스 다이어그램을 통해 초기화 과정 중에 발생하는 시그널과 교환하는

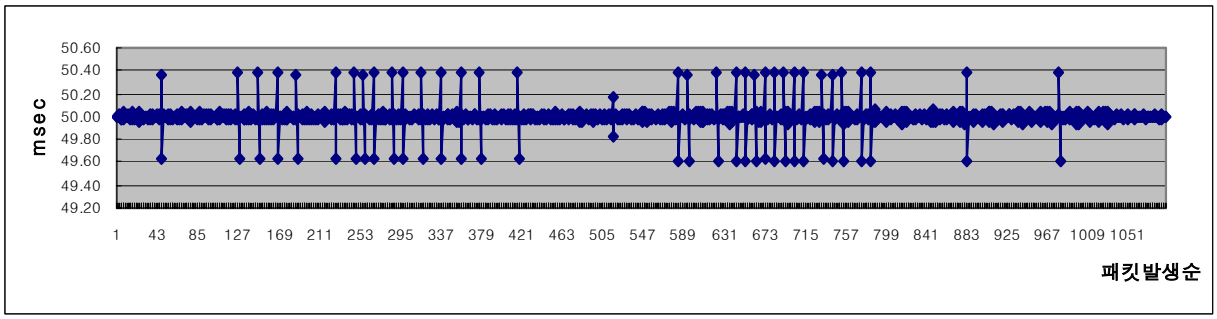


[그림 13] 데이터 분석 화면

데이터들을 설계단계에 작성한 시퀀스 다이그램과 비교하게 된다. 또한 소프트웨어 실행으로 생성된 시퀀스 다이어그램은 각 컴포넌트의 FSM의 상태 천이과정도 오각형의 도형으로 표시되기 때문에 FSM의 각 상태가 각 시그널에 따라 정상적 천이되는지, 시그널이 정해진 타임틱(time tick)에 발생하였는지, 시그널 발생순서가 예상한 순서대로 발생하는지를 확인하여 시스템 차원의 검증이 가능하고 예기치 못한 상호작용과 운용 흐름을 검증하였다.

나. 비기능적 요소 검증을 위한 시험

다중 컴퓨터 환경의 내장형 실시간 소프트웨어인 교전통제 소프트웨어의 시간 정확성, 각 장비들 사이의 통신량 분석 및 시스템 반응 시간과 같은 비기능적인 요소에 대한 시험을 위해 이더넷 패킷 모니터링 기법을 활용하였다. 교전통제 소프트웨어와 각 장비



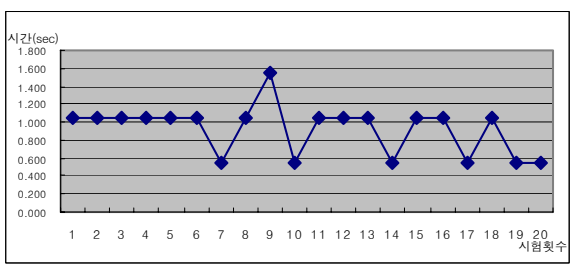
[그림 14] 표적관리 소프트웨어와 발사통제 소프트웨어의 통신 주기 그래프

들 사이에 송수신되는 모든 패킷을 수집·분석하여 시스템 반응시간, 통신 주기 및 데이터 정확성과 변화추이를 그림 13과 같이 확인하였다. 수집된 이더넷 패킷들은 데이터베이스로 구축하여 그래프 또는 CSV (Comma Separated Value)파일 포맷으로 전환하여 각 메시지의 속성값들의 변화추이를 확인할 수 있을 뿐만 아니라 통신 주기, 통신량, 프로토콜, 송수신 방향을 분석하여 비기능적 측면의 성능을 평가하였다.

다. 시험 결과

교전통제 소프트웨어에서는 50msec, 100msec, 비동기식 통신 주기가 요구된다. 장비간 이더넷 패킷을 모니터링하여 통신주기를 측정하였다. 그림 14는 50 msec의 동기식 통신 주기를 측정한 결과로 X축은 패킷이 발생한 순번이고 Y축은 패킷이 발생한 시간 차이다. 0.5msec 오차 이내로 요구되는 시간에 메시지를 전송하였다. 0.5msec의 오차는 TCP/IP의 통신 방식에서 일어나는 오차로, 통신 요구성능을 만족함을 확인하였다.

또한 교전통제 소프트웨어는 자체 성능뿐만 아니라 외부 입출력에 의한 시스템 반응시간 역시 측정이 되어야 한다. 교전통제 소프트웨어에서 최초 표적정보 수신 후 교전을 수행하는데 소요된 시간을 측정하였다. 그림 15는 시스템 반응시간의 측정 결과를 보여 주며, 반응시간은 최대 약 1.5초로 양호하게 나타났다.



[그림 15] 시스템 반응시간의 측정 결과

4. 교전통제 소프트웨어 아키텍처와 컴포넌트들의 재사용성 평가

시스템 요구조건의 변경에 유연하게 대처하기 위해 소프트웨어의 재사용율을 높여야 한다. 표 1은 본 소프트웨어의 타겟 컴퓨터가 VxWorks 운영체제의 PPC 컴퓨터에서 Windows 계열의 운영체제의 펜티엄 컴퓨터로 변경될 경우 소프트웨어에 영향을 미치는 요소를 측정한 것이다.

앞서 그림 3에서 나타난 9개의 컴포넌트 외에도 교전통제 소프트웨어를 구성하는 컴포넌트는 총 19개이며, 표 1은 교전통제 소프트웨어 아키텍처를 이루는 모든 컴포넌트들의 재사용율을 측정한 것이다. 이들 컴포넌트 중 타겟컴퓨터에 의존성이 높은 IPC 메커니즘 관련 컴포너트의 교체 또는 일부 소스코드의 수정으로 재사용이 가능하였다. 각 컴포넌트의 경우에는 하드웨어와 독립적인 알고리즘과 데이터베이스 컴포넌트는 모두 재사용하였으며, 연동 인터페이스는 하드웨어 종속적인 부분인 IPC 메커니즘을 관련 컴포넌트에게 위임시켰기 때문에 관련 IPC 컴포넌트의 교체만 이루어졌고 연동 인터페이스 자체는 그대로 재사

[표 1] 타겟 컴퓨터의 변경에 따른 컴포넌트 재사용율

		변경 요소	변경 전 컴포넌트	재사용된 컴포넌트
알고리즘 및 데이터베이스		해당 없음	7	7
연동 인터페이스		IPC 컴포넌트 교체	9	9
IPC	VxMP	하드웨어 변경	1	0
	TCP/IP	소스 코드 수정	1	1
	UDP/IP	소스 코드 수정	1	1
계			19	18

용하였다. 하드웨어 종속적인 IPC 메커니즘은 VxMP의 경우 PPC(Power PC) 컴퓨터의 VxWorks 운영체제에서만 동작하는 고유한 IPC 메커니즘으로 PC 환경의 Windows 운영체제에서는 사용할 수 없기 때문에 재사용하지 못하였다. 재사용하지 못한 VxMP 컴포넌트는 앞의 그림 10에서 설명한 바와 같이 pComm 프로토콜 클래스를 상속을 받았기 때문에 동일한 인터페이스인 TCP/IP 컴포넌트로 교체하여 교전통제 소프트웨어가 Windows 운영체제에서도 운용이 가능하였다. 또한, TCP/IP와 UDP/IP(User Datagram Protocol/Internet Protocol)와 같은 표준 IPC 메커니즘은 각 컴퓨터와 운영체제에 적합하도록 약간의 소스코드 및 헤더파일을 변경하여 재사용이 가능하였다.

5. 결 론

무기체계 개발특성상 복잡도가 높고 시스템 요구조건 변경에 능동적으로 대처하고 신뢰성을 지닌 대규모의 내장형 교전통제 소프트웨어를 개발하기 위해 교전통제 소프트웨어 아키텍처의 설계방법을 제시하였다. UML 2.0 모델 기반의 설계 기술을 활용하여 교전통제 소프트웨어 아키텍처와 컴포넌트를 설계하고, 아키텍처를 기반으로 컴포넌트를 통합하여 교전통제 소프트웨어를 개발하여 적합성을 확인하였다.

UML 2.0 모델 기반의 교전통제 소프트웨어 아키텍처 설계와 컴포넌트의 개발을 통해 시스템에 적합한 컴포넌트를 선택하여 컴포넌트 조립방식의 교전통제 소프트웨어를 개발할 수 있는 기반을 제공함으로써 소프트웨어의 생산성과 재사용성을 향상시켰다. 또한 모델을 통한 시스템 독립적인 설계를 통해 실 타겟 컴퓨터가 아닌 개발환경 구축이 용이한 PC에서 90%이상의 개발을 수행하였으며, 블랙박스 시험방법에 의한 시나리오 기반의 검증 및 이더넷 패킷 모니터링 기법을 활용하여 소프트웨어 기능적 성능과 비기능적 성능에 대한 시험을 수행하여 교전통제 소프트웨어 신뢰성을 검증하였다.

참 고 문 헌

[1] Byeongdo Kang, Young-jik Kwon, Roger Y. Lee, “A Design and Test Technique for Embedded Software”, IEEE Third ACIS International Conference, pp. 160~165, 2005.

[2] Martin Fowler, Kendall Scott, UML Distilled 2nd Edition, Addison-wesley, 2000.

[3] Ian Oliver, “Applying UML and MDA to Real Systems Design”, IEEE Design, Automation and Test in Europe, pp. 70~71, 2005.

[4] 김현남, 생각하며 배우는 UML 2.0, 영진닷컴, 2004.

[5] Robert G. Petiti IV, “Lessons Learned Applying UML in Embedded Software System Designs”, Software Technologies for Future Embedded and Ubiquitous Systems, pp. 75~79, 2004.

[6] IBM Rational, DEV475 Mastering Object-Oriented Analysis and Design with UML2.0, 2004.

[7] Bass, Clements, Kazman, Software Architecture in Practice, Addison-Wesley, 2003.

[8] Xuejian Luan, Jing Ying, Minghui Wu, “A Heterogenous Evolution Architecture for Embedded Software”, Computer and Information Technology, pp. 901~905, 2005.

[9] D. Mathur, B. W. Edwards, J. Goldstein, H. Nguyen, Jpine, B. A. Plante, J. C. Thacker, “An Approach for Designing Reusable, Embedded Software Compenents for Spacecraft Flight Instruments”, Real Time and Embedded Technology and Applications Symposium, pp. 106~115, 2005.

[10] Haeng-Kon Kim, Roger Y. Lee, Hae-Sool Yang, “Development of Embeded Software with Component Integration Based on ABCD Architecture”, IEEE Fourth Annual ACIS International Conference, pp. 54~60, 2005.

[11] 김범모 외 5명, “임베디드 SW 블랙박스 테스트를 위한 검증 모듈의 디자인 및 구현”, KISS 추계학술대회 Vol. 31, No. 2, pp. 346~348, 2004.

[12] Hu Jun. et al., “Scenario-based Verification for Component-based Embedded Software Designs”, IEEE, The 2005 International Conference on Parallel Processing Workshops, pp. 240~247, 2005.