

객체지향 설계론을 이용한 발사체 시뮬레이터 개발

최 원*, 정해욱** 서진호***, 홍일희***

The Development of Launch Vehicle Simulator Using an Object-oriented Design

Won Choi, Hae uk Chung, Jin-Ho Seo, Il-Hee Hong

Abstract

LCC(Launch Control Center) in NARO Space Center perform a data monitoring and control through the interface to the external system of launch vehicle. Launch Control function needs a high reliability and processing speed. Hence, LCC's remote control system configure a real time system. An important role of the Simulation system is discovering a risk element and minimize it When developing a launch control system. Also, secure a development technique to solve the risks. Launch Vehicle simulator is composed of various component at characteristic of the Launch Vehicle. To be like this each function component the developer will be able to develop easily in order, it using the LabVIEW which is a Graphical Program and its programs, The LabVIEW GOOP(Graphical Object-oriented Programming) which supports an Object-oriented programming it uses with the Component it develops will have a strong point which reusability and a unit test, maintenance, size of program and individual developments.

Key Words: KSLV-I, Launch Vehicle, LCS(launch Control System), Simulator, OOD(Object-oriented Design).

* (주)탑엔지니어링 연구소

** (주)탑엔지니어링 연구소

*** 한국항공우주연구원 우주발사체사업단

1. 서 론

나로 우주센터 발사관제시스템 프로토타입 개발 목적은 KSLV-I 발사체 운용 전 KSLV-I 을 모사하고 시뮬레이션을 통하여 발사관제시스템 (Launch Control System; LCS) 구축 발생 할 수 있는 위험 요소를 사전에 식별하고 그것을 해결할 수 있는 개발기술을 확보하여 위험 요소를 최소화하기 위해서이다. 아래 그림은 구축된 시뮬레이션 환경을 보여준다.

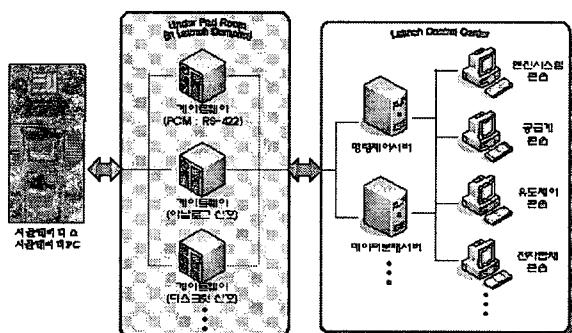


Fig. 1. 발사체 시뮬레이션 환경

기존의 로켓 소프트웨어 개발에 있어서 요구사항 설정, 설계, 제작, 모듈 레벨의 검증, 기체와의 조합 시험에 의한 시스템 레벨의 검증까지, 약 60개월의 정도의 기간이 소요되었으며 조합 시험에서 버그가 발생될 시 설계의 수정이 이루어 져야 했기에 개발기간의 증가가 초래되었다.[1]

발사관제시스템 프로토타입은 발사 관제 시스템을 구성하는 응용 소프트웨어간의 통합 후 발생할 수 있는 기술/설계상의 오류나 문제점들을 사전에 인지하도록 하기 위한 시험환경을 제공하며 개발 후 확보된 각 기능 모듈은 발사관제시스템 개발 시 단위 시험을 위한 테스트 베드로 제공하여 단위 시험 및 통합 시험 기간을 단축시키면서 보다 안정적이고 검증된 발사관제시스템 구축을 도울 수 있는 환경을 제공한다.[2~6] 시뮬레이션을 수행하기 위한 발사체 시뮬레이터는 많은 변수를 가지고 있으므로 개발함에 있어서 객체지향설계론을 적용하여 개체 개발의 편의성과 재사용을 통해 개발 시간을 단축할 수 있었다.

본 논문에서는 위와 같은 목적을 위해 개발된 발사관제시스템 프로토타입에서 시뮬레이션

을 위해 구성된 발사체 시뮬레이터의 개발 방법론과 개발된 발사체 시뮬레이터에 대해 고찰하고자 한다.

2. 객체지향(Object-oriented)

2.1 객체지향설계(Object-oriented Design)

객체지향은 소프트웨어 모듈의 재사용과 독립성을 중요시 하고 객체는 고유한 책임할당을 통한 특화된 역할을 담당한다. 객체지향은 객체들이 메시지(Message)를 통하여 통신함으로써 원하는 결과를 얻는다. 각 객체는 고유의 데이터와 데이터를 처리할 수 있는 메소드로 구성된다. 이러한 객체지향은 문제를 쉽고 자연스럽게 모델링 할 수 있으므로 현실세계의 사고방식을 그대로 적용함으로써 현실세계의 문제를 자연스럽게 표현이 가능한 장점이 있다. 또한, 쉬운 프로그램의 개발로 인한 생산성 향상 및 프로그램 모듈을 재사용 할 수 있으며 프로그램의 확장 및 유지 보수가 용이한 장점을 가지고 있다. 객체지향설계는 개념적인 모델을 시뮬레이션 모델로 변환하는 과정이다. 객체지향접근법은 프로그래머와 분석가가 초기에 생성된 모델을 재사용할 수 있게 하여 복잡한 모델을 만들어 낼 수 있도록 데이터의 추상화, 캡슐화 정보은닉, 상속성, 동적결합등을 제공하므로 프로그래밍과 모델링에 아주 유용하다. 데이터 추상화와 캡슐화는 모듈화를 달성하기 위해 객체지향에서 사용되는 개념이며, 상속성과 동적결합은 재사용성을 확보하는 방법이다.[7] 따라서 발사체와 같은 복잡한 시스템을 시뮬레이션하기 위해서 객체지향에서 제공되는 모듈화와 재사용성의 두 가지 특성을 이용하는 것이 효과적이다. 발사체와 같이 복잡한 시스템에서 변화에 신속하고 쉽게 변경될 수 있는 시뮬레이션 모델을 구축하기 위하여 확고한 시뮬레이션 도구가 요구된다. 본 연구에서는 위와 같은 재사용성과 모듈가능성을 제공하는 객체지향 설계론을 사용하여 발사체 시뮬레이터를 개발하였다.

2.2 개발 프로세스

발사체 시뮬레이터의 개발 프로세스는 요구분석(Requirement Analysis), 분석(Analysis), 설계(Design), 구현(Implementation), 테스트(Test)의 절차로 이루어졌다.

요구분석은 발사체 시뮬레이터 개발 초기 운영자가 시스템에 요구하는 기능을 분석하는 단계이다. 분석은 실제 발사체 시뮬레이터 개발에서 풀어야 할 문제에 대한 분석과정이다. 이 분석은 세부적인 기술이나 특정 기술을 배제하고 실세계의 존재물에 해당하는 모델들에 관한 것이다. 설계 단계는 분석단계의 결과물에 어떻게 구현할 것인지의 기술적인 부분을 첨가하여 확장한다. 구현단계는 실제 소스코드를 생성하는 단계이다. 테스트의 목적은 코드에서의 에러를 발견하고자 함이다.

2.2.1 UML

발사체 시뮬레이터는 객체지향 분석과 설계를 위한 Modeling Language인 UML(Unified Modeling Language)을 이용하여 설계하였다. UML은 객체지향 분석(Analysis)과 설계(Design)를 위한 모델링 언어로 시스템의 기능, 요구사항, 상호 연관관계 등을 가장 명확히 표현 할 수 있다.[8] 발사체 시뮬레이터 설계 과정에서 Use Case, Sequence, Class, Component, Deployment 다이어그램 등을 산출하였으며 UML툴은 Borland 사의 Together를 이용하였다. 구현을 위해서는 NI(National Instruments)사의 LabVIEW를 이용하였으며 LabVIEW의 경우 LabVIEW GOOP(Graphical Object-oriented Programming) 모듈을 이용하여 객체지향 프로그래밍을 수행하였다.

3. 발사체 시뮬레이터 설계

3.1 Use Case Diagram

발사체 시뮬레이터 설계에서 Use Case Diagram은 발사체 시뮬레이터와 사용자가 상호 작용을 하는 경우를 나타낸다. 아래 그림은 발사체 시뮬레이터의 Use Case Diagram을 보여준다.

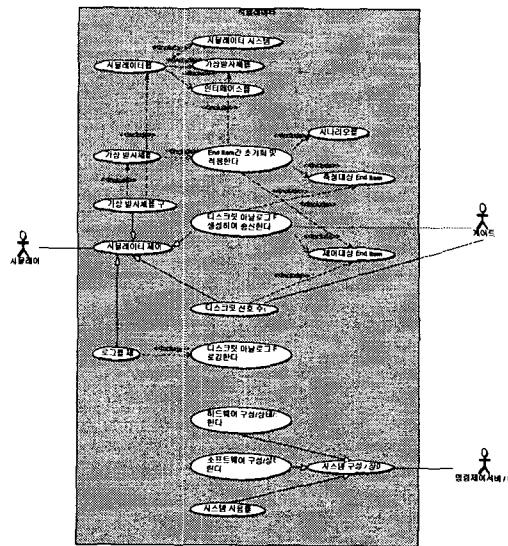


Fig. 2. 발사체 시뮬레이터 Use Case Diagram

3.2 Class Diagram

발사체 시뮬레이터 설계에서 Class Diagram은 여러 가지 객체들의 타입, 즉 클래스들을 표현하고 그 클래스들의 정적인 관계를 표현한다. 아래 그림은 초기화 기능에 관련된 클래스들을 나타낸다.

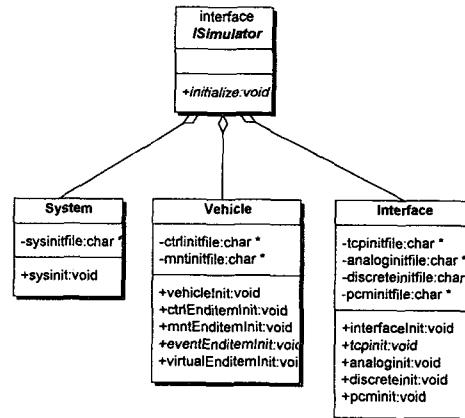


Fig. 3. 초기화 클래스

3.3 Sequence Diagram

발사체 시뮬레이터 설계에서 Sequence Diagram은 시스템의 동적인 구조, 즉 객체와 객체그룹사이, 객체와 객체사이, 객체그룹과 객체그룹사이의 동적인 행위를 기술한다.

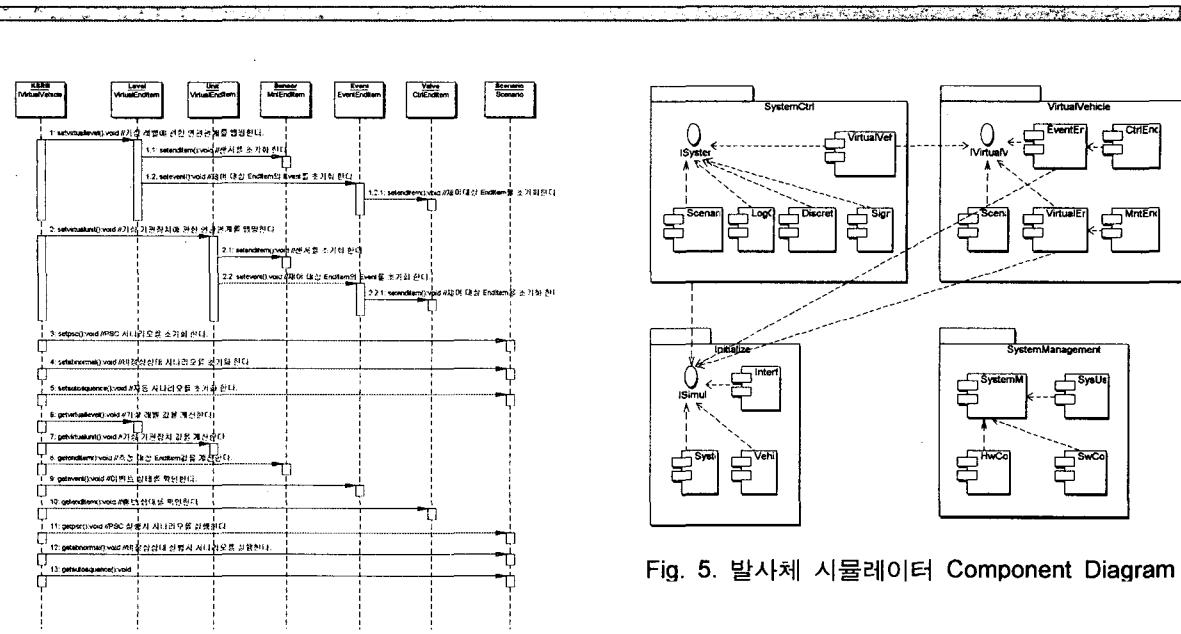


Fig. 4. 가상발사체 구동 및 종료 시퀀스

종좌표축으로 시간개념을 도입하고 횡좌표축으로 객체들을 나열하여 그 사이의 상호작용을 표시한다. 발사체 시뮬레이터의 시퀀스는 가상 발사체 구동 및 종료, 데이터 생성 및 송신, 디스크릿 신호 수신 제어, 로그 제어, 시뮬레이터 초기화, 시스템 구성 및 장애 관리로 구성되었다. 그림. 4는 가상발사체 구동 및 종료시퀀스를 보여준다.

3.4 Component Diagram

발사체 시뮬레이터 설계에서 Component Diagram은 Component들 간의 구성과 의존을 정적인 Implementation View로 모델링하는 것이다. 본질적으로 Class Diagram이며 시스템의 Component에 관점을 둔다. 아래 그림은 발사체 시뮬레이터의 Component Diagram을 나타낸다.

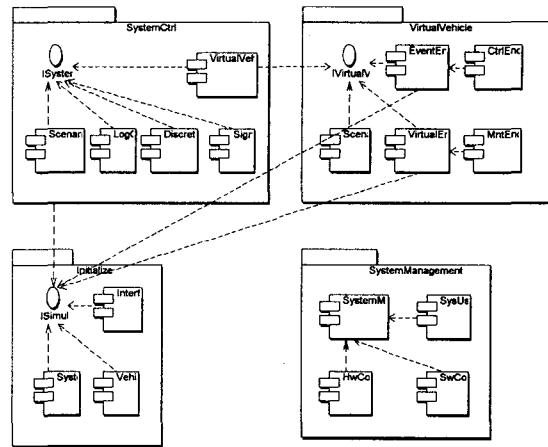


Fig. 5. 발사체 시뮬레이터 Component Diagram

3.5 Deployment Diagram

발사체 시뮬레이터 설계에서 Deployment Diagram은 하드웨어, 소프트웨어 컴포넌트들의 관계를 나타낸다. 아래 그림은 발사체 시뮬레이터와 각 컴포넌트들의 관계를 보여준다.

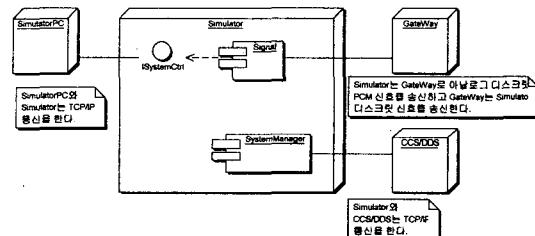


Fig. 6. 발사체 시뮬레이터 Deployment Diagram

4. 발사체 시뮬레이터 구현

4.1 LabVIEW

LabVIEW는 자동제어와 계측의 함수를 중점적으로 제공하는 프로그래밍 언어이며, 데이터의 흐름에 따라서 프로그램이 구성되는 방식을 채택하고 있다. LabVIEW는 그래픽한 아이콘을 이용하여 프로그램을 작성하는 Language이며, User Interface 뿐만이 아니라 Source Code 또한 그래픽한 환경으로 이루어져 있다. 그리하여 Source Code를 이해하기 쉽고 수정과 디버깅이 쉽다.[9]

4.2 LabVIEW GOOP

LabVIEW GOOP(Graphic Object-oriented Programming)는 기존의 Top-Down 설계방식의 LabVIEW 프로그래밍을 객체지향설계 방식의 Component 기반으로 프로그래밍을 할 수 있도록 지원한다.[10] LabVIEW GOOP를 사용함으로써 구성된 Component들에 대해 재사용성과 단위테스트, 유지보수, 프로그램의 크기 및 개체 개발에 대한 장점을 가지게 된다. 본 시뮬레이터 개발에 사용되었던 Component들은 게이트웨이 개발에 재사용하므로써 개발 시간을 줄이는 결과를 가져 왔다.

5. 시뮬레이션 결과

그림. 7, 8, 9, 10은 시뮬레이터 시뮬레이션 운영 화면을 보여 준다.

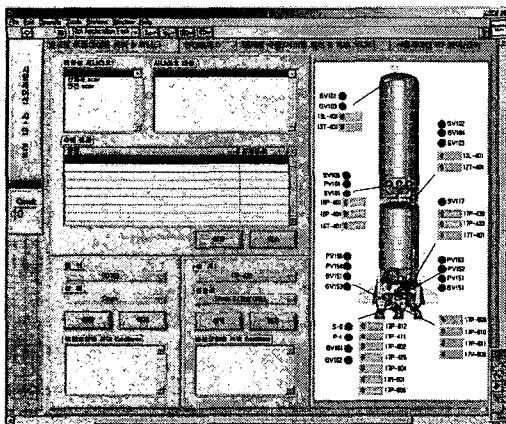


Fig. 7. 발사체 모니터링

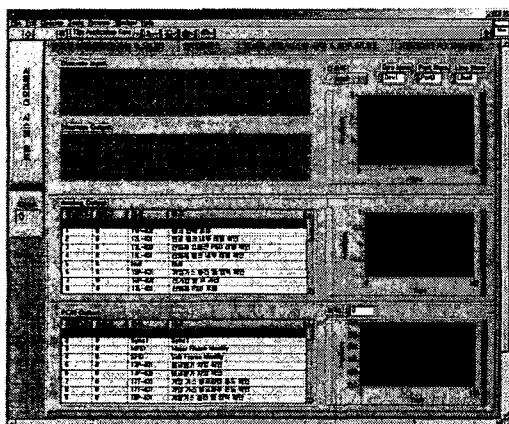


Fig. 8. Interface

그림. 7은 시뮬레이션 시나리오에 따른 발사체의 상태를 모니터링하는 화면을 보여준다.

그럼, 8은 시뮬레이션 시 End Item에서 생성하는 신호와 게이트웨이에 전달하는 데이터들의 상태를 보여준다.

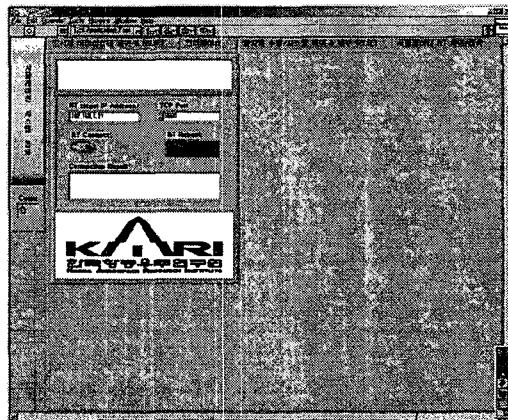


Fig. 9. 제어, 접속 여부 확인

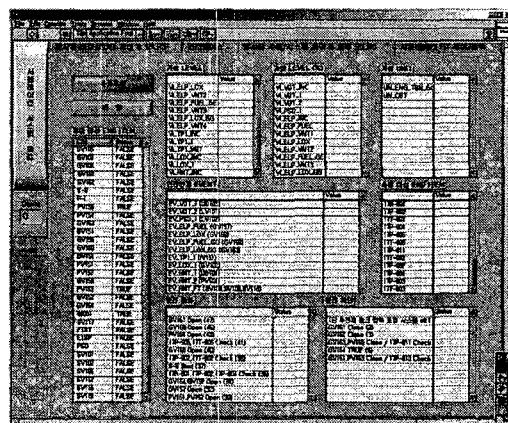


Fig. 10. 발사체 내부 제어

그림. 9는 시뮬레이션 시 시뮬레이터를 구동하기 위한 시뮬레이터 PC의 접속 여부를 확인하는 화면이다. 시뮬레이터는 제어 및 모니터링을 위한 시뮬레이터 PC가 연결되어 운용할 수 있으며, 또한 시뮬레이터 PC 없이 Embedded 되어 운용될 수 있다. 이때는 웹브라우저 기반에서 시뮬레이터 상태를 확인, 모니터링을 한다. 그림. 10은 시뮬레이션 시 End Item 각각을 제어하여 시뮬레이션을 테스트하기 위한 화면이다.

[9] 곽두영, "LabVIEW Express", Ohm사, 2003.

6. 결 론

본 논문에서는 나로 우주센터 발사관제시스템 프로토타입 시뮬레이션을 위한 발사체 시뮬레이터의 개발에 대하여 고찰하였다. 발사체 시뮬레이터의 개발에서 발사체의 각 기능 Component들을 개발자가 쉽게 개발할 수 있도록 Graphical Program인 LabVIEW를 이용하여 프로그래밍 하였다. 객체지향 프로그래밍을 지원하는 LabVIEW GOOP(Graphical Object-oriented Programming)를 사용하여 Component별로 개발함에 있어서 재사용성과 단위테스트, 유지보수, 프로그램의 크기 및 개체 개발에 장점을 가질 수 있었다.

[10] Jorgen Jehander,"Graphical Object-Oriented Programming In LabVIEW", National Instrumernt Application Note 143, 1999.10

7. 참고 문헌

- [1] 야마시타 마사토시, "로켓개발을 위한 시뮬레이터와 비행제어 소프트웨어 검증지원툴 개발소개", 일본항공우주공업회(SJAC) 회보, 2004.12.
- [2] 소형위성발사체(KSLV-I) 개발사업(I), 한국항공우주연구원, 2003.7.16
- [3] Kirk Loufheed, Wayne Prince, "Checkout and Launch Control System(CLCS) System Design Document(SDD), Volume 3A, Section 6. through 6.7. CLCS System Design Topics", NASA CLCS System Engineering Office, 2002. 2.4
- [4] 서진호, 홍일희, 정의승, 정해욱"발사관제시스템 프로토타입 개발", 한국항공우주학회 추계학술논문 발표회, 2004. 11. 19, p1019-1023
- [5] 최원,정해욱,서진호,홍일희, "KSLV- I 발사관제센터 시뮬레이션 시스템 실시간 데이터 처리를 위한 프로토타입 시험평가", 한국항공우주학회 춘계학술논문 발표회, 2004. 4. 16, p603-608
- [6] 서진호,신명호,홍일희,이영호, "KSLV- I 발사관제시스템 개발개념설계", 한국항공우주학회 추계학술논문 발표회, 2003. 11. 14, p1187-1190
- [7] 윤원영,"객체 지향 시뮬레이션 시스템 설계 및 프로토타입 시스템 개발", 지능형통합항만관리 연구센터 1998학년도 2총괄 연구결과,1998
- [8] 신인철,"UML Distilled",홍릉과학 출판사,2000