Journal of Knowledge Information Technology and Systems

ISSN 1975-7700

http://www.kkits.or.kr

Development of an Auto-Build and a Version Management System for a Fly-By-Wire Model of a Rotorcraft.

Ho-Chae Jung[1], **Jong-Min Bae***[2]

[1]*APPIA ENGINEERING*
[2]*Computer Science, GyeongSang National University*

ABSTRACT

The dynamics models and the flight control models are developed on non-real time operating system environment in its first development step when we build a fly-by-wire flight control system. These models are built on a Handling Quality Simulator(HQS) in the next step which is a virtual simulator to be operated in real-time environment that is similar to the real system. A pilot makes quantitative and qualitative assessment for testing the controllability of the models operating in the HQS. In this case the model build process is complex and time consuming. Moreover it is difficult to compare the previous version of models because of the continuous model modifications. In this paper, we present design and implementation result of an auto-build tool which is used for building the flight control models and dynamic models on the top of the HQS, and also present the version management system of models which can be used for comparing models immediately. The auto-build tool can reduce model build time and enhance system reliability. It makes engineers concentrate on only model development itself. The version management system can reduce model testing time and make model insertion, modification and deletion easily. Thus our system can improve development productivity by reducing the model build and testing time.

© 2017 KKITS All rights reserved

KEYWORDS: Fly-by-wire, Model build, Rotorcraft, Flight control model, Version management, HQS, Model based design

ARTICLE INFO: Received 10 July 2017, Revised 7 August 2017, Accepted 11 August 2017.

*Corresponding author is with the Department of Computer Science, Gyeongsang National University, 501 Jinju-daero Jinju, 52828, KOREA.

E-mail address: jmbae@gnu.ac.kr

www.kci.go.kr

1. 서 론

전자식 비행제어장치(Fly-By-Wire)는 항공기 비행 조정 시스템의 하나이다. 전통적인 비행 조정 시스템은 기계구조와 유압에 의존한 반면, 전자식 비행제어장치는 조종신호를 컴퓨터가 해석하여 전기적인 신호를 유압 시스템에 제공하여 조정사의 임무량 감소, 항공기 구조 안전성 증가, 무게 감소를 통한 연비절감효과를 가질 수 있다. 이 장치의 구현을 위하여 모델기반 설계 기법이 사용된다.

모델기반 설계 기법(Model Based Design)은 항공기, 자동차 및 방위산업 등 다양한 산업분야에 널리 사용되는 설계 기법이다.[1] 본 논문에서 다루는 한국형헬기 시뮬레이터는 모델기반 설계 기법으로 개발되었다. 모델기반 설계기법은 제어법칙의 설계와 동시에 시뮬레이션의 정성적인 테스트를 시행함으로써 모델의 기능을 검증과 개발 시간을 단축시킬 수 있다. 또한 설계된 제어법칙을 추가 작업 없이 실시간 시뮬레이터에 자동생성코드 기능을 활용하여 탑재프로그램에 적합한 소스코드를 생성할 수 있다.[2-4] 이러한 기능은 모델기반 설계기법이 가지는 주요한 장점이다. 모델기반설계 기법은 MIL, SIL, PIL, HIL 테스트를 지원함으로써 각 단계에서의 개발 시간을 크게 단축시킬 수 있다.[5]

모델기반 시뮬레이션 모델은 다음과 같은 요건을 갖추어야 한다.[6-7] 첫째, 모델 자체가 이해하기 쉽고 또한 다루기 쉬워야한다. 둘째, 모델에 쓰이는 변수들을 변경하거나 주고받기가 용이해야 한다. 셋째, 설계서/명세서가 직접 시뮬레이션 가능해야 한다. 넷째, 구조적인 오류는 별도로 분리되어 관리될 수 있어야 한다. 다섯째, 정확한 실시간 작동이 가능해야 한다. 여섯째, 외부에서 획득한 데이터로부터 모델을 직접 시뮬레이션 가능해야 한다. 일곱째, 시뮬레이션 수행 오류가 감지되고 평가될 수 있어야 한다. 그리고 중요한 변수들은 자동으로 수집 가능해야 하며, 마지막으로 이해하기 쉬운 성능 보고서가 자동 생성 가능해야 한다.

모델기반설계에는 자동생성코드기능과 코드랩퍼 기술이 사용된다. MATLAB Simulink[1]에서 만들어진 모델기반 알고리즘을 타겟기반의 소스코드를 생성하고 이를 탑재해서 실행시킨다. 이때 코드랩퍼기술을 적용하여 모델을 실행가능하게 만든다.

헬리콥터용 전자식 비행제어 시스템은 ADOCS(Advanced Digital Optical Control System)에서 개발된 비행제어법칙을 활용한다.[8-12] 그리고 헬리콥터용 전자식 비행제어법칙을 효율적으로 개발하기 위한 개발환경에 대한 많은 연구가 있다.[13-14] 그리고 개발된 전자식 비행제어시스템의 조종성을 평가하기 위해서는 ADS-33E-RFP[7] 국제표준기준을 만족해야 한다.

개발된 비행제어시스템을 평가하기 위하여 동역학모델들과 비행제어법칙모델을 가상의 시뮬레이션 환경에서 조종사의 정성적인 조종성 테스트와 수치해석상의 정량적인 테스트를 수행한다. 이를 기반으로 HILS(Hardware in the Loop Simulator) 환경에서 실장비와 연동을 통하여 비행 적합성 테스트를 수행하여 실체계로 전자식 비행제어 시스템을 탑재하기 위한 사전 검증과정을 수행한다. 이와 같이 비행제어 시스템을 검증함에 있어서 동역학 모델의 안전성, 성능의 정량적 그리고 정성적인 평가가 요구된다.

비행 시스템 엔지니어가 역학 모델들과 비행제어법칙을 개발함에 있어서 수치해석상의 정량적인 평가만으로는 검증에 한계가 있기 때문에 가상 시뮬레이터 HQS(Handling Quality Simulator)[15]를 활용하여 정성적인 조종성 평가를 수행한다. 이때 몇 가지 제약사항이 있다. 첫 번째로 개발된 비행제어 모델과 동역학 모델들을 가상의 시뮬레이터로 빌드함에 있어 오랜 시간이 소요되어 개발의 생산성이 떨어진다. 두 번째로 버전관리 기능이 없어 버

www.kci.go.kr

전별로 모델들을 비교함에 즉각적인 피드백이 어렵다. 이에 본 논문에서는 헬리콥터용 전자식 비행제어 시스템 개발을 위하여 자동화된 빌드환경을 제공함으로써 모델빌드시간을 단축시키고, 모델버전관리시스템을 제공함으로써 이전 버전의 모델들과의 즉각적인 상호비교를 가능하게 함으로써 모델 테스트 시간을 단축시킴으로써 개발 생산성을 향상시키는 시스템 개발 환경을 제시한다.

2. Handling Quality Simulator

HQS(Handling Quality Simulator)는 헬리콥터 전자식 비행제어시스템의 평가를 위한 환경으로서, 비행제어컴퓨터, 서버모델, 항공기 장비와의 연동을 통해 비행제어시스템에 필요한 데이터를 생성하고, 그 결과를 검증할 수 있는 시스템이다.[15] HQS를 구성하는 호스트 컴퓨터의 소프트웨어 형상은 <그림 1>과 같다.

호스트컴퓨터는 동역학 모델과 제어법칙, 이들을 스케줄링 하는 메인 스케줄러, 시스템을 통제하는 통제 시스템, 로그조회도구, 그리고 본 논문에서 제안한 자동 빌드 도구로 구성된다. 여기서 모델자동 빌드 도구는 항공엔지니어가 제작한 제어법칙과 동역학 모델 소스를 HQS에 빌드하는 과정을 자동화하는 도구이다. 이때 사실적인 조종성 평가를 위하여 실시간성이 요구된다. 호스트 컴퓨터는 내부 모듈 및 서브시스템 모델의 추가, 변경 및 삭제에 대해 효율적으로 대처할 수 있도록 표준화된 인터페이스 드라이버를 통해 제어될 수 있도록 설계된다. 호스트 컴퓨터의 소프트웨어는 제어법칙, 동역학 모델, 영상처리 모델, H/W 인터페이스 모델 그리고 로그 모델로 구성되며, 주 제어프로그램과 서브시스템 사이의 표준화된 인터페이스를 통해서 각 모델을 효율적으로 관리한다.

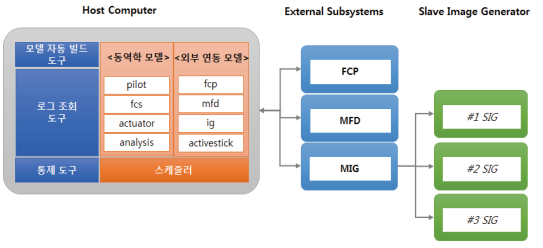


그림 1. HQS 소프트웨어 형상
Figure 1. Software Configuration for HQS

FCP(Flight Control Panel)은 각종 스위치 입력값을 제어법칙 모델에 전달하는 기능을 한다. FCP는 항공기의 각종 게이지(Air Speed, Altitude, Heading 등) 정보를 화면에 전시하고, 모의 통제도구에 있는 기능 중 시뮬레이터 제어기능(Load, Unload, Start, Stop, Restart)을 조종석에서 바로 수행할 수 있는 편의기능을 제공한다. 모의조종간 스틱(Cyclic/Collective Stick, Pedal) 제어, 영상 UCE(Use Cue Environment) 레벨 설정기능을 포함한다.

MFD는 조종사에게 다양한 항공기 정보를 제공하는 장치이다. 항공기의 자세, Heading, Air Speed, Altitude, X/Y 방향 가속도, 이착륙 가속도 그리고 엔진 RPM 등 다양한 정보를 제공한다. 조종사는 작전형태에 따라 적절한 화면 페이지를 선택하여 화면에 전시한다.

영상시스템은 프로젝터로 실제 영상을 내보내는 채널 1, 2, 3번의 부 영상시스템(Slave Image Generator)이 있는데, 1대의 주 영상시스템으로써 3 채널의 부 영상시스템을 제어한다.

3. 모델 자동빌드 도구

여기서는 모델 비행 엔지니어가 개발한 동역학 모델과 비행제어법칙 모델을 자동빌드하는 도구에 대하여 논한다. 모델 자동빌드라 함은 비행 엔지니

www.kci.go.kr

어가 개발한 모델들을 실시간 환경인 HQS에 맞게 소스코드를 컴파일하고 모델들을 실행가능하게 만드는 과정을 말한다.

<그림 2>는 모델 자동 빌드 과정을 나타낸다. 비행제어법칙모델 개발자는 동역학모델과 비행제어법칙모델의 소스코드와 이들의 입출력 관계를 표시하는 설정파일(MDF 파일)을 제공한다. 자동빌드 도구는 이 자료를 바탕으로 크게 3가지 기능을 수행한다. 첫째, 실시간 환경의 시뮬레이터로 복사한 동역학 모델과 제어법칙을 정적 라이브러리로 구성한다. 둘째, 모델 입출력정보가 저장된 MDF 파일을 바탕으로 모델을 실제로 호출하는 드라이버를 생성하여 정적라이브러리와 링크시켜 실행 파일을 구성한다. 마지막으로 설정 정보가 저장된 설정파일을 생성한다.

3.1 정적 라이브러리 생성

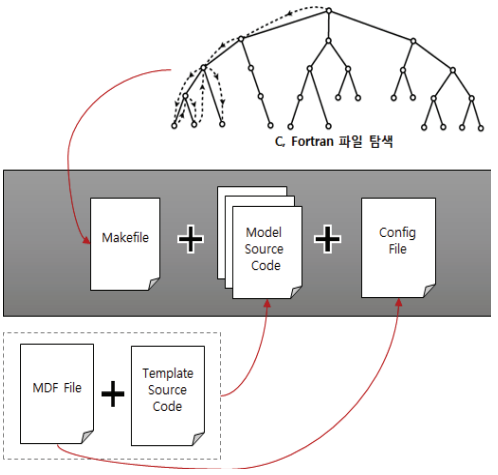


그림 2. 모델 자동 빌드 과정
Figure 2. The Auto-Build Process of the Model

동역학 모델들과 제어법칙 모델들을 시뮬레이터의 지정된 위치로 가져와서 각 모델들을 검색하여 컴파일 과정을 묶은 makefile을 자동으로 생성하고 이를 바탕으로 모델들을 하나의 정적 라이브러리로 묶는다. 이는 나중에 실행파일을 생성하는 단계에서 사용된다.

3.2 드라이버 생성

모델기반설계기법에서는 코드래퍼기술을 사용하여 개발된 모델들을 목적 플랫폼에 맞추어 소스코드를 생성한다. 본 논문에서도 코드래퍼기술을 적용한 드라이버를 생성한다. 드라이버 파일은 실제적으로 모델의 함수를 호출하는 소스코드이며 하나의 실행 가능한 프로세스가 된다. 예를 들어 FCS, Actuator, Sixdof라는 모델이 있다면 이들의 드라이버 파일을 생성하고 이전단계에서 생성된 라이브러리와 함께 컴파일하게 되면 세 개의 독립적인 실행파일이 생성된다. 이들은 주프로그램으로부터 인터럽트를 받아 시뮬레이션이 종료 될 때까지 수행된다. 모델 프로세스의 입출력 데이터는 공유메모리를 통하여 대화한다.

드라이버 파일을 생성하기 위해서는 비행엔지니어가 사전에 작성한 MDF 파일이 필요하다. MDF 파일에는 모델의 이름, 수행주기, 입력목록, 출력목록이 저장되어 있다. 이를 바탕으로 모델 자동빌드 도구는 모델을 호출하는 랩퍼함수와 이를 호출하는 메인 함수를 포함하는 드라이버 파일을 생성한다. 랩퍼함수의 매개변수는 MDF에 정의된 입출력 목록 정보를 해석하여 해당 파라미터가 저장되는 공유메모리에 대한 포인터를 넘겨준다. 이때 Fortran과 C간의 호출규약을 맞추어 주어야 한다.

<그림 3>은 템플릿 코드와 MDF를 이용하여 드라이버를 생성하는 과정을 나타낸 것이다. 그림에서 보는 바와 같이, 템플릿 코드에는 공유메모리 초기화, 인터럽트 핸들러, CPU 동작 위치설정 등 기본적인 구조가 정의되어 있다. MDF에는 입출력

www.kci.go.kr

데이터와 사용 가능한 모델함수의 이름이 정의되어 있다. 이를 바탕으로 <그림 4>와 같은 드라이버를 생성한다. 생성된 드라이버와 모델 컴파일 과정에서 생성된 라이브러리 정보를 바탕으로 모델 별로 하나의 실행파일을 생성한다.

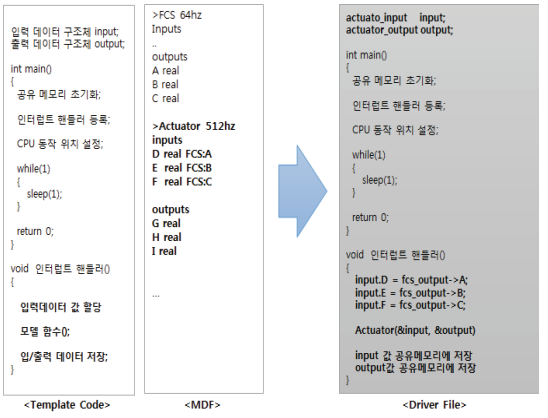


그림 3. 드라이버 생성 과정
Figure 3. The Process of a Driver Generation

3.3 설정파일 생성

설정파일은 두 가지 파일로 구성되어 있다. 하나는 버전관리를 위한 파일로 버전이름과 모델이 저장된 기준 폴더정보를 포함하고 있다. 이 파일은 시뮬레이터 통제 도구에서 버전을 선택할 때 사용된다. 두 번째 설정파일은 모델의 이름, 동작주파수, 입출력 정보가 저장되어 있다. 버전별로 하나씩 생성된다. 파일의 정보를 바탕으로 버전별로 실시간 데이터 모니터링이 가능하다. 만약 설정파일 없이 버전별로 모델의 입출력 목록이 상이하다면 버전별 입출력 데이터 관리가 어렵다.

4. 버전관리

버전관리 기능은 버전별로 동역학 모델이나 비행제어법칙을 전환해 시뮬레이션 해봄으로써 즉시 조종사의 정성적인 테스트 수행 시 상호비교가 가능하게 한다. 만약 버전관리 기능 없이 모델을 전환하면서 테스트를 수행하면 시뮬레이션을 종료하고 새로운 모델로 교체를 해야 한다. 그 후 다시 시뮬레이션을 실행시켜 초기화 과정을 거치는 번거로운 과정을 다시 해야 한다. 이러한 과정 없이 버전관리 기능을 사용하게 되면 시간지연 없이 시뮬레이션이 가능하다.

4.1 버전관리를 위한 메모리 구조

시스템에는 총 네 가지 종류의 공유메모리가 정의되어 있다. 즉, 전체 시스템을 통제하기 위한 공유메모리, 모델들의 입출력 값을 저장하는 공유메모리, 로그 데이터를 저장하기 위한 공유메모리 그리고 모델들의 실행시간을 저장하는 공유메모리가 있다. 이들은 서로 다른 메모리 맵핑 파일로 정의되어 있으며 시스템의 오류가 발생하여 멈추는 경우 설정파일과 메모리 맵핑 파일을 통하여 오류발생 직전의 입출력 값을 알 수 있다.

모델 입출력 공유메모리는 각 모델별로 4Kbyte 크기의 고정크기로 할당되어 있으며 이들은 다시 1Kbyte의 서브 섹션으로 나누어진다. 첫 번째 섹션은 입력 데이터, 두 번째는 출력 데이터, 세 번째는 datapump 데이터, 네 번째는 fix 데이터 영역으로 정의된다. 모델이 수행되기 전 설정파일에 정의된 값을 찾아 입력 데이터 섹션에 저장하고 모델을 수행한다. 수행 된 결과 값을 다시 자신의 출력 데이터 섹션에 저장한 후 모델 수행을 다른 모델로 넘긴다. 이러한 메모리 구조는 버전별로 입출력 목록이 바뀌더라도 설정파일에 따라 실시간 데이터 모니터링이 가능하다는 장점을 가지고 있어 시스템이 보다 유연해 질 수 있다.

www.kci.go.kr

4.2 모델 로드와 실행

HQS의 버전관리를 위한 주요 폴더는 다음과 같이 구성된다. config 폴더에는 각종설정파일이 저장되고,. version.xml에는 버전의 이름, 기준경로, 모델입출력 설정파일이 저장된다. sim 폴더는 주프로그램의 소스코드가 있다. 주프로그램은 모델들에 종속적이지 않도록 설계되었다. 그 결과, 모델이 바뀌더라도 주프로그램의 변경은 발생하지 않는다. 동역학모델과 비행제어법칙은 버전별로 hetlas 폴더 밑에 위치한다. 버전이 증가할 때마다 hetlas 폴더 밑에 새로운 폴더를 생성하고 모델들을 복사한다. 메인 스케줄러가 시뮬레이터 통제도구로부터 로딩 하고자 하는 버전의 모델들의 로드명령을 받으면 설정파일에서 모델들의 기존폴더를 찾아서 해당 모델들을 차례대로 실행시킨다.

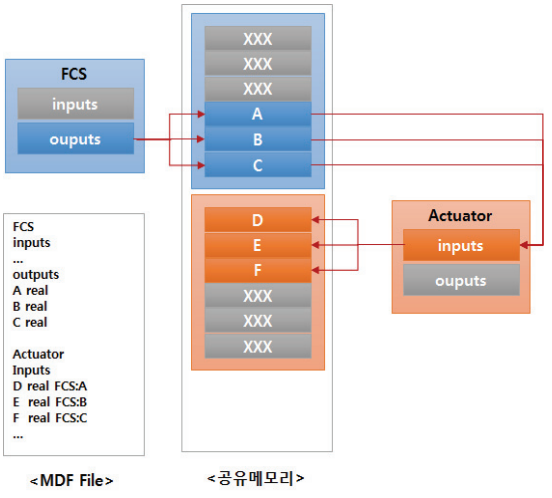


그림 4. 모델 입출력
Figure 4. The Models I/O

<그림 4>는 모델들의 입출력 관계가 메모리에서 사상되는 방법을 나타낸다. 각 모델들은 하나의 독립적인 실행파일이다. 이들 모델들은 모델입출력파일에 정의된 내용에 맞추어서 공유메모리에 접근하고 각 함수의 입출력 내용을 결정한다. 예를 들어 FCS 모델의 출력 값으로 A, B, C라는 값이 정의되어 있고 이 값이 Acuator 모델의 입력 값 D, E, F로 할당되어 있으면 Actuator 모델은 공유메모리를 통하여 FCS 모델의 출력 값 A, B, C를 가져와 자신의 입력 값 D, E, F에 할당한다.

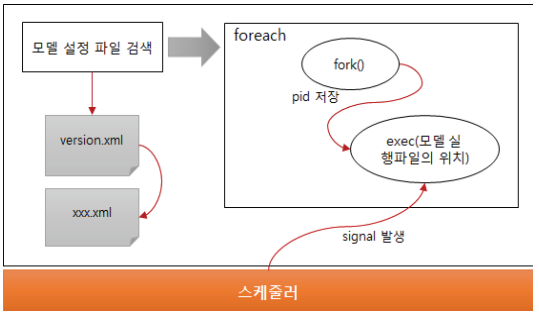


그림 5. 모델 로드과정과 실행
Figure 5. The Process of Model Load and Execution

<그림 5>는 모델 로드해서 실행시키는 절차이다 주프로그램은 프로그램 구동 시 버전 설정 파일을 읽어 버전정보를 저장한다. 시뮬레이터 통제 도구에서 원하는 버전을 선택하면 네트워크를 통하여 메인 스케줄러에 버전 번호를 전송한다. 메인 스케줄러는 수신된 버전 번호로 모델들의 정보를 조회한다. 버전 설정 파일에는 모델들의 정보가 저장된 모델 입출력 파일의 경로가 저장되어 있다. 모델 입출력 파일을 읽어 해당 버전의 모델의 실행파일들 로드 할 수가 있다. 이어서 새로운 자식 프로세스들을 생성하고 새로운 프로세스에 실행흐름을 할당한다. 이런 과정을 모든 모델에 적용하여 모델 프로세스를 로드한다. 또한 로드될 때 생성된 프로세스의 PID정보를 저장하고 있다가 스케줄링 시 PID를 활용하여 모델들에게 시그널을 보낸다. 모델들은 시그널 인터럽트 핸들러를 등록해 두고 있다가 자신에게 시그널이 발생하게 되면 모델 수행을

www.kci.go.kr

반복한다. 이 과정을 프로그램이 종료될 때 까지 반복하며 시뮬레이션을 수행 한다

4.3 버전별 데이터 모니터링

버전별로 데이터의 입출력 목록이 다를 수 있기 때문에 데이터를 조회하기 위한 유연한 설계가 필요하다. 본 논문에서는 버전별로 유연하게 데이터를 모니터링하기 위하여 설정파일을 기반으로 데이터를 모니터링 할 수 있는 기능으로 구현하였다.

<그림 6>은 모델함수 실행과정과 입출력 자료의 흐름을 나타낸 것이다. 먼저 모델 자동빌드 도구가 MDF 파일(모델의 입출력 정보가 저장되어 있는 최초 파일)을 기반으로 랩퍼소스코드를 생성하면서 모델설정파일을 함께 생성한다. 랩퍼소스코드에는 입출력 데이터와 모델함수호출부가 정의되어 있는데, 입출력 데이터는 모델함수의 매개변수로 사용된다. 함수가 호출되기 전에 MDF 파일에 저장된 다른 모델의 출력 값을 조회하여 자신의 입력 데이터에 배정한다. 모델 호출결과, 출력 데이터에 결과 값이 결정되면 자기의 입출력 자료와 덤프데이터를 자신에게 할당된 공유메모리 영역에 저장한다.

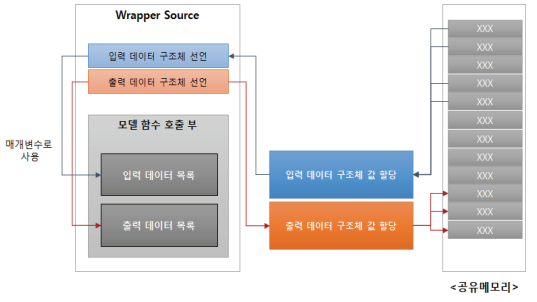


그림 6. 모델함수 실행과정
Figure 6. The Execution Process of a Model Function

<그림 7>은 실시간으로 모델들의 입출력 정보를 모니터링하고 출력데이터를 사용자가 임의로 고정시킬 수 있는 GUI이다. 시뮬레이터를 로드할 때 버전정보를 입력하면 시스템은 해당버전의 입출력 설정파일을 읽어서 모델별로 입출력 정보를 트리 형태로 제시한다. 사용자가 실시간으로 감시할 데이터를 선택하면 해당 데이터의 공유메모리 번지를 계산하여 화면 중앙에 출력한다. 또한 왼쪽 화면에서 사용자가 원하는 출력 값을 임으로 고정할 수 있는 기능도 제공한다.

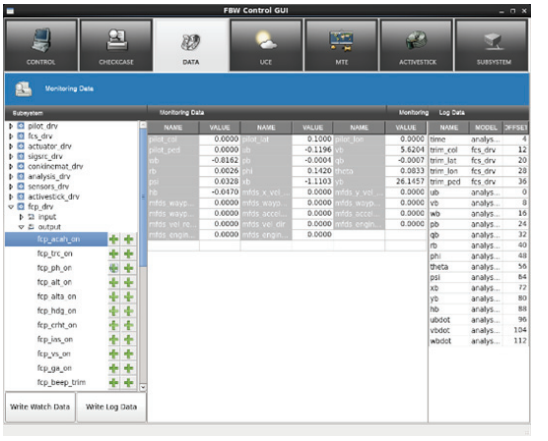


그림 7. 실시간 데이터 모니터링
Figure 7. GUI for Real-Time Data Monitoring

5. 결 론

헬기용 전자식 비행제어 모델은 항공기, 자동차 및 방위산업에서 널리 쓰이는 모델기반 설계기법으로 개발한다. 모델기반 설계에서는 자동생성코드 기능과 코드랩퍼기술을 활용하여 HQS 환경의 시뮬레이터에 동역학모델과 비행제어법칙 모델을 빌드해야 한다.

이때 모델 컴파일 과정의 복잡성으로 인하여 동역학 모델과 비행제어법칙 모델들을 HQS에 빌드하는 시간이 오래 걸려 개발 생산성 저하의 원인이 될 수 있다. 또한 지속적인 모델 변경으로 인하여,

www.kci.go.kr

이전 모델들과의 상호비교가 어려운 문제점이 있다.

본 논문에서는 이를 해결하기 위하여 자동빌드 도구와 모델들의 버전관리시스템을 제시하였다. 자동빌드도구를 사용하여 자동화된 컴파일과정을 제공함으로써 모델 빌드시간을 단축할 수 있고 이를 통해 엔지니어가 모델개발에만 집중할 수 있는 환경을 제공할 수 있어서 개발 생산성이 향상된다. 그리고 컴파일 과정에서 사용자 조작 미숙으로 인한 오류를 방지할 수 있어 시스템의 신뢰성을 향상시킬 수 있다.

또한 개발된 버전관리 기능을 통해서 모델 테스트 수행시간을 단축시킬 수 있다. 조종사는 시뮬레이션 비행시험을 하면서 이전 모델에서의 조종 감각을 잃지 않고 버전별로 모델 비교를 용이하게 할 뿐 아니라, 모델의 추가, 수정, 삭제도 용이하게 한다.

References

[1] Tom Erkkinen, *Simulink capabilities for embedded code generation*, The Mathworks, 2003.

[2] S-J An, I-J Cho, C-D Yang, and H-J Lee, *A process and tool development of auto generated code for fly-by-wire helicopter flight control law*, Proceedings of The Korea Society for Aeronautical and Space Science, pp. 1185-1189, 2012.

[3] C-S Kim, J-G Lee, S-Y Choi, E-S Jung, and I-J Cho, *A study on verification of auto code for UAV flight control law software*, Proceedings of The Korea Society for Aeronautical and Space Science, pp. 417-422, 2012.

[4] J-H Park, B-M Kim, and B-S Kim, *Evaluation technique and automating code development of handling quality for conventional aircraft,* Proceeding of The Korea Society for Aeronautical and Space Science, pp. 1278-1282, 2008.

[5] D. W. Nixon, *Flight control law development for the F-35 JointStrike flight*, Lockheed-MartinAeronautics, 2004

[6] S-J Ahn, C-S Kim, I-J Cho, and J-G Heo, *A development and verification process of auto generated code for fly-by-wire helicopter control law,* Journal of The Korea Society for Aeronautical and Space Science, Vol. 41, No. 6, pp. 488-494, 2013.

[7] *Handling quality requirements for military rotorcraft*, Aeronautical Design Standard Performance Specification ADS-33E-PRF, 2000.

[8] L. R Stiles, J. Mayo, A. L. Freisner, K. H. Landis, and B. D. Kothmann, *Impossible to resist; The development of rotorcraft fly-by-wire technology*, American Helicopter Society 60th Annual Forum, 2004.

[9] S. I. Glusman, K. H. Landis, and C. Dabundo, *Handling qualities evaluation of the ADOCS primary flight control system*, American Helicopter Society 42nd Annual Forum, 1986.

[10] J. Harding, Moody, S. Jeram, G. Mansur, and M. Tischler, *Development of modern control laws for the AH-64D in hover/low speed flight*, American Helicopter Society 62nd Annual Forum, 2006.

[11] K. H Landis, J. M. Davis, C. Dabundo, and J. F. Keller, *Advanced flight control technology achievements at Boeing helicopters,* International Journal of Control, Vol. 59, No. 1, pp. 263-290. 1994.

[12] B. Boczar, D. Borgstrom, and P. Everett, *Key aspects and attributes of the RAH-66 Comanche flight control system,* American Helicopter Society 55th Annual Forum, 1999.

[13] C-S Kim, I-J Cho, S-D Lee, and H-J Lee, *Design and validation of model inversion flight control law for fly by wire helicopter*,

www.kci.go.kr

Journal of The Korea Society for Aeronautical and Space Science, Vol. 40, No. 8, pp. 678-687, 2012.

[14] C-S Kim, I-J Cho, S-D Lee, and H-J Lee, *A study on fly-by-wire flight control law development environment*, Proceedings of The Korea Society for Aeronautical and Space Science, pp. 648-653, 2011.

[15] K-C Park, J-B Seo, and H-C Jung, Development for handling quality test environment of rotorcraft fly-by-wire flight control system, Proceeding of The Korea Society for Aeronautical and Space Science, pp. 1190-1194, 2012.

헬리콥터용 전자식 비행제어 모델에 대한 자동빌드도구와 버전관리시스템 개발

정호채[1], 배종민[2]

[1]*APPIA ENGINEERING*

[2]*경상대학교 자연과학대학 컴퓨터과학과*

요 약

전자식 비행제어시스템을 개발하기 위해서는 먼저 동역학모델과 비행제어법칙 모델을 비실시간 운영체제 환경에서 개발한다. 그 후 이를 실시간 환경인 가상 시뮬레이터 HQS(Handling Quality System)에 빌드한다. 조종사는 실 체계 환경과 유사한 환경인 HQS에서 개발된 모델들의 조종성을 평가하기 위하여 정성적인 평가와 정량적인 평가를 함께 수행한다. 이때 빌드과정이 복잡하고 많은 시간이 요소 된다. 또한 지속적인 모델의 수정으로 인하여 이전 모델과의 상호비교가 쉽지 않다. 본 논문에서는 비행제어법칙모델과 동역학모델들을 가상 시뮬레이터로 자동 빌드하기 위한 도구와 즉각적으로 모델들의 상호비교가 가능한 버전관리 시스템을 제시한다. 자동빌드도구는 모델빌드시간을 감소시킬 수 있고 시스템의 신뢰성을 향상시킬 수 있다. 이는 엔지니어가 모델개발에만 집중할 수 있게 한다. 버전관리시스템 역시 모델 검증시간을 단축시킬 수 있고, 모델의 삽입, 수정, 삭제를 용이하게 한다. 이는 모델 빌드 시간과 테스트 시간을 단축시킴으로써 모델 개발의 생산성을 향상시킬 수 있다.



Ho-Chae Jung received the bachelor's degree in the Department of Business Administration from Academic Credit Bank System in 2010. He received the M.S. degree in the Department of Computer Science from Gyeongsang National University in 2015. He is currently working at APPIA Engineering, LTD. His current research interests include embedded software, avionics software and fourth industrial revolution.

E-mail address: jhc1211@gmail.com



Jong-Min Bae received the bachelor's degree in the Department of Mathematics Education from the Seoul National University in 1980. He received the M.S. degree and the Ph.D. degree in the Department of Computer Science and Statistics from Seoul National University in 1983 and 1995 respectively. He has been a professor in the Department of Computer Science at Gyeongsang National University since 1984. His current research interests include programming languages, compilers, and computer education.

E-mail address: jmbae@gnu.ac.kr

www.kci.go.kr