위성시험운영 통합 절차서 언어 설계 접근

곽남이*, 허윤구**, 최종연***

Design Approach to Satellite Test and Operations Common Procedure Languages

Nam-Yee Kwak*, Yun-Goo Huh**, Jong-Yeoun Choi***

Abstract

In order to develope a common ground system, a general procedure language that can be used in both EGSE and MCS is primarily needed. As the first step in developing a common test and integration procedure language, KARI's ATS for AIT and MCE for GS were compared to some of the most representative languages such as PLUTO regarded European standard, STOL and ELISA and PIL. Based on the analysis, design features of developing a common test and integration procedure language were presented.

초 록

위성 개발 및 운영의 주체가 달라서 이중으로 소모되는 에너지를 줄이기 위해 공통지상시스템의 통합 개발이 절실히 필요하다. 위성의 개발, 시험, 통합, 발사, 궤도 운영, 임무수행의 전주기를 아우르는 공통지상시스템을 개발하기 위해서는 모든 단계에서 사용될 수 있는 언어를 개발해야 한다. 이를 위해 위성 시험운영언어의 유럽표준인 PLUTO와 가장 대표적인 언어중 하나인 STOL과 ELISA 및 PIL과 현재 항공우주연구원 AIT의 위성 시험언어인 ATS와 지상국 운영시스템인 MCE를 자세히 분석하고, 이를 토대로 통합 절차서 언어 개발을 위한 설계 요건들을 제안한다.

키워드 : 통합 절차서 언어(common procedure language), ATS(Automatic Test Scripts), MCE(Mission Control Element), PLUTO(Procedure Language for Users in Test and Operations), STOL(Satellite Test and Operations Language), ELISA(Extended Language for Integrations of Spacecrafts and Avionics), PIL(Powerful Operations Language)

접수일(2008년12월17일), 수정일(1차 : 2009년 9월 15일, 2차 : 2009년 10월 20일, 게재 확정일 : 2009년 11월 1일)

* 위성기능시험팀/namyee@kari.re.kr ** 위성기능시험팀/perfect@kari.re.kr *** 위성기능시험팀/jycho@kari.re.kr

1. 서 론

AIT(Assembly, Integration & Test ; 위성체 총조립 시험)를 위한 EGSE(Electronic Ground Support Equipment; 전기지상지원장비)와 위성 임무 운영을 위한 MCS(Mission Control System; 관제시스템)의 통합은 미국과 유럽의 위성 관련 기관 및 산업체에서 이미 지난 20세기 후반부터 활발히 연구되고 있다.[1~5] 두 시스템은 기술적 측면에서 많은 공통점과 호환성을 가지고 있으나, 사용 시기 및 개발 주체가 달라서 독립적으로 개발되어 왔다. 이에 따라 위성 개발 주체와 운영 주체는 유사한 기능의 두 시스템을 중복하여 개발하게 되고, 이 두 시스템은 검증 절차도 개별적으로 수행해야만 한다.

두 시스템의 유사성과 호환 가능성에 근거한 통합시스템의 개발은 이러한 비효율성을 줄이기 위해 논의되기 시작했다. 통합시스템 개발의 효과는 적지 않다. 우선 시스템 개발과 운영 및 교육에 필요한 사업비용을 절감할 수 있다. 또한 위성 개발 단계에서 이미 검증한 위성 운영 단계의 시나리오를 다시 검증할 필요가 없이 다시 재사용할 수 있음에 따라 운영 위험도를 낮출 수 있다. 그 외에도 업무 수행의 효율성이 높아지고, 시스템의 유지보수 부담은 낮아진다.

이러한 많은 장점을 가지는 통합 시스템을 개발하기 위해서는 두 시스템의 절차서 언어와 데이터베이스 구조를 통합하는 것이 핵심적이다.[6] 본 연구에서는 통합 절차서 언어 개발에 주안점을 두고, 현재 항공우주연구원의 위성 총조립 시험단계와 운영단계에서 사용되고 있는 절차서 언어인 ATS (Automatic Test Scripts)와 MCE (Mission Control Element)와 대표적인 통합 절차서 언어인 PLUTO (Procedure Language for Users in Test and Operations)와 STOL (Satellite Test and Operations Language), ELISA (Extended Language for Integrations of Spacecrafts and Avionics)와 PIL(Powerful Operations Language)를 분석하고 이를 토대로 통합시스템의 절차서 언어의 주요 설계 요건들을 제안한다.

2. 절차서 언어

2.1 PLUTO (Procedure Language for Users in Test and Operations)

PLUTO는 ESTEC (European Space Research and Technology Centre)과 ESOC (European Space Agency Space Operations Center)이 개발한 유럽의 표준 절차서 언어(ECSS-E-70-32A)로, 위성 개발 전주기에 사용할 수 있는 절차서 언어의 정의를 목적으로 개발되어 2000년대 이후 대부분의 ESA 위성 운영에 적용 및 검증되어 온 가장 대표적인 통합 절차서 언어이다.[7), 8)] 위성의 개발 및 운영단계에서 모두 사용할 수 있으며, ESA가 1970년대부터 축적해온 30여 년의 경험을 바탕으로 개발한 핵심지원모듈인 SCOS-2000의 절차서 언어이기도 하다.[9), 10)] 이 표준안에서는 통합 절차서 언어를 문법과 구조, 절차서의 체계의 관점에서 자세하게 정의하고, 이를 모두 만족시키는 언어로 PLUTO를 제시하고 있다.

PLUTO는 절차서가 선언부 (Declaration Body), 전제부 (Preconditions Body), 메인부 (Main Body), 감시부 (Watchdog Body), 확인부 (Confirmation Body)의 다섯 개의 서브 구조로 이루어진다는 구조적 특징을 갖는다. 선언부는 절차서를 실행하면서 발생할 수 있는 이벤트를 사전에 선언하여 감시부에서 활용할 수 있도록 관리하는 부분이다. 전제부는 절차서 또는 하위 프로세스의 실행 여부를 결정하기 위한 트리거 역할을 하는 조건을 정의하는 부분이다. 이 초기 조건을 만족하지 않으면 절차서는 즉시 중단된다. 메인부는 절차서의 목적과 직결된 명령 코드들의 집합으로써 하나 이상의 독립적인 하위 단계들로 구성된다. 절차서의 하위 단계는 변수에 값을 인가하고 데이터 흐름을 제어하며 다양한 함수를 사용하여 연산 및 저장하는 것과 같은 실질적인 작업을 수행한다. 전제부에서 사전에 선언한 우발상황들은 모두 감시부에서 관리한다. 감시부는 하나 이상의 '감시 단계'들로 이루어지며 이 단계들은 동시에 실행된다. 하나 이상의 감시 단계가 시작되고 중단 명령이 반환되면 상

위 절차서(단계)를 즉시 취소하며, 그렇지 않은 경우에는 감시 단계가 모두 종료되기 전까지 대기한다. 확인부에서는 수행결과를 평가하는 조건문을 사용하여 절차서의 목표달성여부를 확인 및 평가한다.

PLUTO는 불리언형, 상수나열형(enumerated constant), 정수, 실수, 문자열, 상수, 절대시간 상수, 상대시간 상수의 데이터형과, if문과 for문, case문, while문, repeat문, wait until문과 같은 기본적인 데이터 흐름 제어문을 제공한다.

2.2 STOL(Satellite Test and Operations Language)

STOL은 NASA가 개발하여 ITOS (Integrated Test and Operations System)와 EPOCH에서 사용하고 있는 통합 절차서 언어다.[11] ITOS는 NASA 고다드 우주비행센터에서 위성과 위성 컴포넌트의 개발, 운영단계를 위해 개발한 소프트웨어로써[11] 보드 및 박스레벨 개발단계부터 운영에 이르는 위성 전주기를 지원하는 소프트웨어다. 이식성이 뛰어나며 구조가 체계적이다. 1990년에 최초의 소형탐사선임무인 SAMPEX의 위성체 총조립 시험을 지원하기 위한 EGSE로부터 발전되었으며, 이후 GLAST (Gamma-Ray Large Area Space Telescope), ULDB (Ultra Long Duration Balloon Program), 감마선 폭발 연구용 위성 SWIFT, SMEX (Small Explorer)의 6기 위성과 지구기후연구용 위성 Triana, Spartan 251, Spartan 401 등 12개 임무를 지원하고 있다. ITOS와 STOL은 NASA와 계약 관계의 기관 및 업체에 한해서 무료 사용이 허용되고 있어 미국 업계에서 활용도가 높다. EPOCH이 대표적인 예인데, EPOCH은 1982년에 설립된 위성 지상 시스템 분야의 선두 업체인 Integral Systems의 위성 명령 및 관리에 관련된 최초의 상용 소프트웨어로, STOL을 절차서 언어로 사용한다. 현재에 이르기까지 205개 이상의 위성 임무를 지원한 상용 위상지상시스템이다.[12]

절차서는 아규먼트 선언부, 변수 선언부, 명령어부의 세 섹션으로 구성되며, 이 순서가 반드시 지켜져야 한다. 절차서들 간에 아규먼트를 전달할 수 있으며, 이 값에 의해 절차서의 실행 내용이 바뀔 수 있다. 광역 변수와 지역 변수가 모두 정의되어 있다. 또한 현재 실행 중인 절차서에서 하나 이상의 절차서를 반복적으로 호출할 수 있다. 사용자가 지정한 시간에 따라 절차서 실행을 연기하거나 대기시킬 수 있으며, 사용자와 절차서간의 입력과 출력을 통한 상호 작용이 가능하다. STOL은 기본 데이터 형으로 정수, 실수, 논리상수, 문자열, 시간상수를 지원한다. EPOCH T&C 스트림 변수를 참조할 수 있으며, UNIX 명령 발행이 가능하다. STOL은 단순한 구조와 명령어가 50여개로 적어 사용자의 학습 및 활용이 용이하면서도 강력한 기능을 제공하는 통합 절차서 언어이다.

2.3 ELISA 와 PIL

ELISA와 PIL은 Matra Marconi의 후신인 Astrium(프랑스 툴루즈)의 시험 절차서 언어, 운영 절차서 언어이며,[13), 14)] 현재 항공우주연구원의 통신해양기상위성 개발에 사용되고 있다.

2.3.1 ELISA(Extended Language for Integrations of Spacecrafts and Avionics)

ELISA는 PLUTO와 함께 유럽의 대표적인 절차서 언어 중 하나이다. Astrium의 위성 개발단계는 'Open Center'라는 소프트웨어가 주관하는데, 사용자는 'Open Center'의 하위 컴포넌트인 'LNG(Test and Interface Language)'가 제공하는 사용자환경에서 ELISA를 이용하여 시험 시나리오를 작성하고 실행한다. ELISA는 매우 구조화된 언어로 시험 시퀀스부(Test Sequences), 루틴부(Routines), 외부 루틴부(External Routines), 전역 시퀀스부(Global Sequence)의 4가지 모듈로 구성된다.

시험 시퀀스부는 ELISA 절차서의 메인 실행 프로그램으로 LNG에서 순차 및 병렬 실행이 가능하다. 시험 시퀀스부는 다음 그림과 같이 다시 광역부(Global), 시퀀스부(Sequence), 루틴(Routine)

세 부분으로 나눠진다.

시험 시퀀스부에서 사용하는 모든 광역변수 및 상수, 루틴들은 반드시 광역부에 선언해야 한다.

```
GLOBAL
CONST
.....        definition of global constants used by the Test Sequence
VAR
.....        definition of global variables used by the Test Sequence
END GLOBAL
```

```
SEQUENCE <name_of_sequence>
PROTO
.....        definition of the arguments of the Test Sequence
END PROTO
CONST
.....        definition of constants used in the Test Sequence
VAR
.....        definition of variables used in the Test Sequence
MAIN
.....        source code of the Test Sequence
.....
END SEQUENCE
```

```
ROUTINE <name_of_routine_1>
.....
END ROUTINE
.....
ROUTINE <name_of_routine_n>
.....
END ROUTINE
```

그림 1. ELISA 시험 시퀀스부와 루틴부의 내부 구조

호출된 아규먼트를 시퀀스부의 'PROTO'에 정의하고, 시험 시퀀스부의 내부에서 사용되는 상수와 변수를 'CONST'와 'VAR'에 선언한다. 변수들을 먼저 선언한 후, 'MAIN'에 주요 기능을 수행하는 프로그램을 작성하고 마지막으로 루틴부에 루틴 코드를 작성한다.

```
GLOBAL
CONST
.....        definition of global constants used by the External Sequence
VAR
.....        definition of global variables used by the External Sequence
END GLOBAL
```

```
ROUTINE <name_of_routine>
PROTO
.....        definition of the arguments of the Routine
END PROTO
CONST
.....        definition of constants used in the Routine
VAR
.....        definition of variables used in the Routine
MAIN
.....        source code of the Routine
.....
END ROUTINE
```

그림 2. ELISA 외부 루틴부의 구조

ELISA의 가장 큰 특징 중 하나는 외부 루틴들을 따로 분리해서 이를 시험 시퀀스들 간에 공통적으로 사용할 수 있게 모듈화 하는 것이다. 위의 그림은 외부 루틴부의 구조를 보여준다.

외부 루틴부는 독자적 파일에 선언하되, 하나의 파일에 단 하나의 외부 루틴만 선언하도록 정의되어 있다. 외부 루틴은 시험 시퀀스부에서 '@루틴명(아규먼트들)'의 명령어로 호출한다.

ELISA는 절차서간 호출이 빈번하여 그에 따른 혼동이 없도록 하기 위해서 모든 시험 시퀀스가 공유하는 광역변수를 광역 시퀀스부에 따로 모아 관리한다. 광역시퀀스부의 구조는 다음 그림 3.과 같다.

```
GLOBAL SEQUENCE <global_sequence_name>
CONST
.....        definition of global constants
VAR
.....        definition of global variables
END GLOBAL
```

그림 3. ELISA 광역 시퀀스부의 구조

ELISA는 정수, 실수, 불리언, 문자열, 시간상수 등의 데이터형 및 2차 배열 변수를 지원하며, 기본 연산자 및 데이터 흐름 제어문을 제공한다. 또한 광역 변수를 사용할 수 있으며 모든 광역변수들은 하나의 광역 시퀀스부에 선언되어야 한다. ELISA는 사용자가 프로젝트 고유 특성에 따라 새로운 명령문을 추가해야 할 때에 활용할 수 있도록 공개 표준 절차서 소스를 제공한다. ELISA의 구조는 PLUTO에 비견할 수 있을 만큼 체계적이다. 사용자는 손쉽게 외부 루틴부를 활용해서 자주 사용하는 작업들을 함수화할 수 있다. 그러나 PLUTO, STOL과는 달리 위성 개발단계에서 시험 언어로만 사용되고 있다.

2.3.2 PIL(Powerful Operations Language)

PIL은 Astrium의 운영소프트웨어인 'OPSWARE'의 절차서 실행프로그램인 'OPSEXECUTER'에서 사용되는 운영언어이다. PIL은 다른 위성시험운영언어와 유사하게 선언부와 메인부 결과확인부 등으로 이루어져 있으며 기능에 따라 위성 온보드상의 매스메모리를 삭제하고 새롭게 업로드하

거나 시퀀스 파라메터 값을 초기화하고 업로드하는 부분이 있다.

PIL은 코드를 'ACTION', 'TC CODE', 'TM CODE', 'NAME', 'TM VALUE'의 다섯 개의 필드로 나눠진 표 형식으로 작성하는 독특한 구조적 특징을 가진다. 'ACTION'은 변수를 정의하고 초기값을 인가하고 조건에 따라 연산을 수행하는 필드이고, 'TC CODE', 'TM CODE'는 각각 위성과 통신하며 주고받고자 하는 원격명령과 원격측정의 코드명을 입력하는 필드이다. 원격측정 값의 예상치는 'TM VALUE' 필드에 작성한다.

위성을 운영하는 입장에서는 지상국과 위성의 컨택타임의 길이와 주기가 중요한데, 통상 저궤도 위성은 하루 한번 약 10여분, 지구궤도 위성은 24시간 컨택이 가능하다. 따라서 지구궤도 위성은 지상국의 명령에 대해 즉각 응답할 수 있으나, 보통 저궤도 위성은 그렇지 않다. 그래서 저궤도 위성은 원격측정을 바로 측정하고 확인하는 부분은 사용이 불가능하다. 이에 위성의 종류와 관계없이 같은 형식의 운영 절차 코드를 작성하고자 한 시도가 이 필드를 분리하한 표 형태의 운영 언어라고 할 수 있다.

3. 언어요소 비교 분석

3.1 데이터 흐름 제어문

위성을 시험하고 운영할 때 논리구조를 세우고, 이를 자유롭게 시험하고 확인하는 것은 프로그래밍 언어의 기본적인 기능이다. 이를 가능하게 하는 것이 데이터 흐름 제어문이다. 일반적으로 알려진 if문, for문, while문 등이 이 구문에 해당한다. PLUTO는 if문, case문, repeat문, while문, for문을 지원하며 이들을 서로 중첩하여 다양한 데이터 실행 흐름을 만들 수 있다. 주어진 조건을 만족하는 동안 데이터의 흐름은 해당 구문 안에서 제어된다. STOL은 PLUTO와 마찬가지로 if문 do-while문을 지원한다. 또한 start문, pause문, stop문, step<n>문, continue문, goto문, gototime<expression>문, return문 등 보다 직접적인 방법으로 실행 제어문을 제어한다. STOL은 PLUTO와 마찬가지로 시험과 운영 모두를 지원하는 시험운영언어이지만, 이 두 언어는 그 스타일에 있어 차이를 보인다. PLUTO는 STOL에 비하여 좀 더 구조적인 특성을 가진다. STOL은 명령실행지점을 통제하여 원하는 데이터의 실행 흐름을 형성한다. ELISA는 if문, swith-case문, do문, for문, goto문, exit if문을 지원하며 PLUTO와 유사한 구조적 언어의 특징을 보인다. ATS는 시험만을 위해 사용되는 위성 시험언어로 데이터 흐름 제어를 위해서 if문, while문, goto문, run문, action문, exit문, wait문을 제공한다.

언어별로 데이터 흐름 제어문과 명령어가 정확히 일치하는 것은 아니나 기본적인 명령문과 명령어가 유사하므로 이들을 조합을 사용하여 원하는 기능을 구현할 수 있다.

3.2 하위 프로세스

PLUTO와 ELISA는 명령문들로 구성된 하위 프로세스들로 하나의 절차서를 구성함으로써 보다 구조화되고 체계적인 특징을 갖는다. PLUTO는 절차서 내에 하위 단계를 정의할 수 있는데, 하위 단계는 순차 및 동시 실행이 가능하다.

예 1.을 보면 "Switch on Gyro3 in Fine Mode"와 "Swith on Gyro5 in Fine Mode"의 두 개의 하위 단계가 "in parallel until all complete" 명령문에 의해 동시에 실행되는 것을 볼 수 있다. 'Switch on Gyro3", "Gyro3 Fine Mode" 등 각 단계 내의 명령문이 순차적으로 실행된다. 상대적으로 하위 프로세스 및 절차서 간 연계 구조가 단순한 STOL과 ATS도 필요에 따라서 절차서 호출기능과 아규먼트 전달을 통해 같은 기능을 구현할 수는 있다.

```
Switch on Gyro3 and Gyro5 in Fine Mode
procedure
  preconditions
    wait until Gyro3 and Gyro5 Converter = "ON"
  end preconditions
  main
    in parallel until all complete
      initiate and confirm step Switch on Gyro3 in Fine Mode
        preconditions
          wait until Temperature of Gyro3 > 60 degC
        end preconditions
        main
          initiate and confirm Switch on Gyro3;
          initiate and confirm Gyro3 Fine Mode;
        end main
      end step;
      initiate and confirm step Switch on Gyro5 in Fine Mode
        preconditions
          wait until Temperature of Gyro5 > 60 degC
        end preconditions
        main
          initiate and confirm Switch on Gyro5;
          initiate and confirm Gyro5 Fine Mode;
        end main
      end step;
    end parallel;
end main end procedure
```

예 1. PLUTO의 순차, 병렬 실행의 예

3.3 데이터와 매개변수

각 언어에서 지원하는 데이터 형식에는 큰 차이가 없다. 모든 언어가 불리언, 정수, 실수, 문자열 상수, 시간 상수를 지원한다. 다만 매개변수의 정의 및 활용 정도는 언어 간에 차이를 보이는데, PLUTO, STOL, ATS의 절차서 간 데이터 전달 방법은 단순히 광역변수를 사용하는 것으로 제한된 반면에 ELISA는 절차서도 반환 값을 가질 수 있는 등 매개변수를 활발히 사용한다. 매개변수는 상수, 변수, 변수 주소 값으로 전달될 수 있으며, 주소 값으로 전달된 매개변수는 광역변수를 사용하지 않고도 현재 절차서에서 수정된 값을 상하위 절차서에서 사용할 수 있다.

3.4 함수

특정 시간동안 지연하거나 조건을 만족할 때까지 대기시키는 시간 기반 함수와 삼각함수, 절대값, 나머지연산, 버림, 제곱근, 최대・최소값, 로그, 자연로그 등의 기본적인 수학 함수는 네 언어에서 모두 사전 정의되어있거나, 명령어와 연산자의 조합으로 표현이 가능하다.

ELISA는 사용자로 하여금 사용자 정의 함수를 정의해서 활용하도록 적극적으로 구조화되어 있다. 이에 비하여 PLUTO, STOL과 ATS는 절차서 안에서 다른 절차서를 호출하는 방식으로 절차서를 구조화할 수는 있으나 ELISA처럼 매개변수를 주고받고 광역변수를 하나의 파일로 모아 정의하는 등 함수의 정의 및 사용을 적극 권장하지는 않는다.

3.5 Pre / Post 조건부

PLUTO의 가장 큰 특징은 선언부, 전제부, 메인부, 감시부, 확인부의 다섯 개 부분으로 이루어진 체계적 구조라고 할 수 있다. STOL과 ATS는 PLUTO의 메인부에 대응된다. 또다른 구조적인 언어 ELISA가 갖지 않은 PLUTO만의 특징이 있는데 그것은 절차서의 시작을 결정할 수 있는 전제부와 메인부에서 발생하는 이벤트를 병렬적으로 감독하는 감시부, 절차서 실행의 최종 결과의 적합성 여부를 확인하는 확인부가 따로 구분되어 있다는 것이다. 이러한 구조는 사용자로 하여금 보다 체계적으로 절차서를 구상하고 구현할 수 있도록 돕는 요소이다.

4. 통합시스템 설계

국제적인 추세에 발맞추어 항공우주연구원에서도 AIT의 전기지상지원장비에서 개발된 핵심 모듈인 원격명령 및 원격측정 처리기 소프트웨어와 데이터베이스를 위성관제시스템에서 재사용하는 방안을 적극적으로 고려하고 있기는 하나, 아직까지는 전기지상지원장비와 관제시스템은 서로 다른 두개의 시스템이다. 이번 장에서는 두 시스템D시스템이다언어 및 환경의 관점에서 분석하고 통합 시스템 개발D시위한 주요 설계 요소들을 제안한다.

항우연 AIT가 개발한 EGSE는 ETB (Electrical

Test Bed)의 조립시험을 통한 시험장비 자체 및 위성체 총조립 시험 기간 동안 성능 및 기능시험에 사용되고 있다.

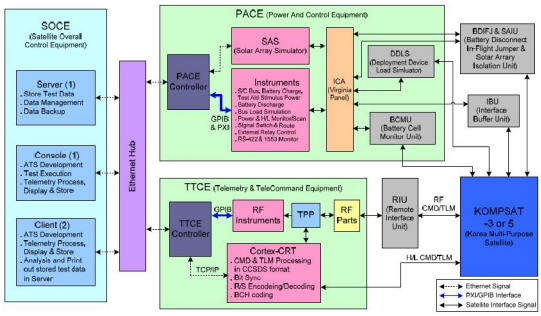

그림 4. AIT EGSE 구성

그림 4.에서 볼 수 있듯이 EGSE는 PACE와 TTCE, SOCE로 구성되고 이중 특히 SOCE는 이더넷 허브를 통해 PACE와 TTCE에 원격명령을 전달해서 시험을 전체적으로 제어한다. 그림 1.은 EGSE 구성을 보여준다.

SOCE 응용 소프트웨어는 위성체 시험을 위한 시험 시나리오에 대한 제작 환경을 제공하고 이를 자동으로 실행한다. 또한 시험을 위한 명령 및 시험 데이터를 송수신 처리 및 실시간 시현하고 위성체 시험 결과를 저장하며 분석을 위한 환경을 제공한다. 위성체 시험은 단위 시험 명령을 실행하거나 자동시험 시나리오를 실행하는 방식으로 이루어지는데, 이 자동 시나리오는 시험 절차서 언어인 ATS로 작성한다. 작성한 절차서는 EGSE 전체 시스템을 효율적으로 제어해서 그 결과 모니터링, 분석한다.

ATS는 기본적인 연산자와 함수 및 명령어가 내장 정의되어 있어 판독성과 작성력이 우수한 언어다. ATS는 정수, 자연수, 실수, 문자열, 16진수, 날짜문자열 를 지원한다. 또한 논리, 비교, 산술, 단항, 우선 순위, 배열 참조와 같은 기본적인 연산자와 삼각함수, 일반 수학함수, 시간함수와 ATS 상태 반환함수 등이 내장 정의돼있다. ATS는 구조적으로 단순하지만 ATS내에서 다른 ATS를 호출하여 순차적으로 실행할 수 있고, 광역변수를 통해 ATS간에 데이터 교환이 가능하다.

위성 총조립 및 시험단계의 절차서언어는 상대적으로 완전하지 않은 위성 데이터베이스와 시험 시나리오 검증, 자세한 디버깅 및 레포트 기능, 시험 환경 변화에 따른 빠른 대처 및 오류처리 기능 검증을 위한 유연성이 요구되는데, ATS는 이러한 시험 절차서 언어의 요구 사항에 부합한다.

운영단계는 개발단계 시스템과는 다른 관점의 시스템 운용 목적과 요구도를 갖는다. 위성관제 시스템은 위성의 종류와 접촉 주기에 따른 임무 업로드 및 데이터 다운로드, 위성 상태정보 및 임무 수행 결과 모니터링, 운영단계에서의 주요 관심사는 위성이 각 부분체별 상태 모니터링 등 여러 위성의 운용 측면에서 시스템의 유연성이 요구된다. 모니터링 및 분석의 주기가 개발단계에 비해 길다.

항우연 지상국에서는 MCE (Mission Control Equipment)라는 위성관제시스템을 사용하고 있다. 이 시스템은 다음 그림과 같이 SOS (Satellite Operations Subsystem), MPS (Mission Planning Subsystem), FDS (Flight Dynamics Subsystem), SIM (Simulator), TTC (Tracking, Telemetry, and Commanding Subsystem)로 구성된다.

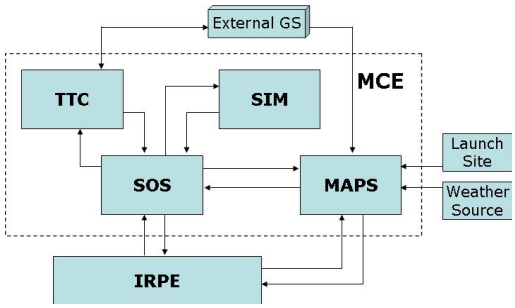

그림 5. 지상국 MCE 구성

위성체 총조립 및 시험단계에서는 위성에 전송할 명령을 ATS로 텍스트 코딩하는 것에 반해, 지상국에서는 위성의 임무에 따른 명령어 그룹들을 사전에 저장해놓아서 사용자는 단순히 해당 그룹을 선택하는 것만으로 위성에 임무를 전달할

수 있게 구현되어 있다. 이는, 위성 개발과정을 거쳐 실제적인 운영 시점에 이르게 되면 위성이 수행하게 되는 거의 대부분의 기능이 결정될 뿐 아니라, 위성에 잘못된 명령어를 전달할 가능성을 차단하고 운영자가 보다 용이하게 운영할 수 있도록 하기 위함이다.

전기지상지원장비와 관제시스템의 성공적인 절차서 언어 통합을 위해서는 두 시스템의 역할 및 목적에 따른 요구사항의 이해가 선행되어야 한다. 그리고 이를 바탕으로 먼저 두 시스템의 문법과 구조를 통일시켜야 하고, AIT나 지상국에서 작성한 절차서 콘텐츠 자체가 두 조직에서 무리 없이 사용되어야 한다. 이를 위해서는 두 시스템 모두에 적합한 사용자 인터페이스 개발이 필요하다.

또한, 지구궤도위성 운영 개념을 지원할 수 있는 시스템으로 개발되어야 한다. 지금까지 항우연이 개발한 위성은 모두 저궤도 또는 중궤도 위성으로써 저궤도 위성은 지상국과 하루에 한두 번, 약 십여 분동안 지속되는 짧은 컨택 시간을 갖는다. 컨택 시간이 짧기 때문에 지상국은 이 시간동안 위성이 수행한 임무 결과를 전송받아야 하고 또 새로운 임무를 실행 시간 태그와 함께 전송해야 한다. 반면에 지구궤도위성은 24시간 지상국과 교신하게 되므로 저궤도 위성보다는 위성과 늘 교신하는 시험단계와 오히려 유사하다.

이에 따라 AIT/지상국, 저궤도/지구궤도의 컴파일 옵션을 사용해서 AIT의 시험 및 지상국의 지구궤도위성용 사용자 인터페이스와 지상국의 저궤도위성용 사용자 인터페이스를 다르게 구현하는 통합시스템을 제안한다.

5. 결 론

본 연구에서는 AIT의 전기지상지원장비와 지상국의 관제시스템의 통합 절차서 언어의 개발을 위해서 대표적인 통합 절차서 언어인 PLUTO, STOL와, ELISA 및 PIL와 현재 항공우주연구원에서 사용하고 있는 시험언어인 ATS와 운영시스템인 MCE를 자세히 분석하고 이를 토대로 통합 절차서 언어의 주요 설계 요건들을 제안하였다.

ATS와 STOL에 비하여 PLUTO와 ELISA는 고도로 구조화된 언어이다. 구조적인 차이점이 존재하기는 하나, 기능의 관점으로 볼 때에는 구현 방법의 차이일 뿐이기도 하다. PLUTO와 ELISA는 다양한 구문을 사용하는 방법으로, ATS와 STOL은 실행라인을 변경하는 방법으로 데이터 흐름을 제어한다. 또한 PLUTO와 ELISA는 하위 프로세스와 사용자 정의 함수를 다양하게 사용하는 특징을 가진다. 네 언어 모두 시간함수, 수학 함수와 각 언어의 특징에 맞는 특화된 함수들의 집합을 가진다. PLUTO의 다섯 개로 구분된 절차서 구조는 사용자로 하여금 체계적으로 절차서를 구성하게 하는 강점을 가지고, ELISA의 호출 구조에 특화된 특징들은 복잡한 임무 수행을 체계적으로 조직할 수 있게 하고, 번거로운 작업들을 함수화하여 손쉽게 이용할 수 있게 하는 장점을 가진다. 그러나 ATS와 유사한 간단한 구조만을 가지고도 NASA의 대표적인 위성시험운용 소프트웨어인 ITOS와 대표적인 상용프로그램인 EPOCH에서 사용되는 STOL을 보면, ATS의 향후 추가 개선에 있어서 구조적인 측면은 선택사항으로 보인다. 자동 레포트 기능 및 전기지상지원장비 유효성 확인의 자동화 등 사용자 관점에서 필요한 함수들을 벤치마킹하여 리스트화하고, 이를 구현해서 라이브러리로 제공하는 것은 또 다른 개선의 방법이 될 것이다.

또한 항우연은 현재 첫 지구궤도위성인 통신해양기상위성 발사를 앞두고 있으며 향후 추가적으로 지구궤도위성을 보유할 가능성이 있으므로 차세대 통합시스템은 저궤도위성과 지구궤도위성 모두를 지원할 역량을 확보해야 하며, 이를 위해 컴파일 옵션에 따라 사용자 인터페이스 다르게 구현되는 통합시스템을 제안한다.

참 고 문 헌

1. 허윤구, 최종연, “공통지상시스템 SCOS-2000 개발 동향,” 한국항공우주학회 춘계학술대회, 2008
2. 최종연, "위성 공통지상시스템 개발 동향," 항

공우주산업기술동향, 5권2호, pp. 33-42, 2007
3. M. Jones, M. Merri, F. Diekmann, S. Valera, A. Parkes, "Evolution of the ECSS-E-70 Ground Segment and Operations Standards", SpaceOps 2008
4. ECSS, Test and operations procedure language (ECSS-E-70-32A), Netherlands, ESA Publications Division, 2006, pp.1-138
5. G. Chaudhri, J. Cater, B. Kizzort, "A Model for a Spacecraft operations Language", Space Ops 2006
6. G. Chaudhri, S. Hollander, "Ground Systems - The Need for Standardization", Space Ops 2004
7. ECSS, "E-70-32A Test and operations procedure language," ESA Requirements and Standards Division, April 2006
8. Demeuse, B. & Valera, S., "PLUTO, a Procedure Language for Users in Test and Operations", DASIA 98 - Data Systems in Aerospace, Proceedings of the conference held 25-28 May, 1998 in Athens, Greece. Edited by B. Kaldeich-Schu.rmann. ESA SP-422. Paris: European Space Agency, 1998., p.307
9. J.-P. Chamoun, S. Risner, T. Beech, G. Garcia, "Bridging ESA and NASA Worlds: Lessons Learned from the Integration of hifly/SCOS-2000 in NASA's GMSEC", Aerospace Conference, 2006
10. ESA bulletin 128 - november 2006
11. J. Gal-Edd, J. A. Steck, J. C. Isaacs, III, M. J. Bredeck, C. C. Fatig, R. H. Zepp, "Selection of the Ground Segment for the Next Generation Space Telescope (NGST) Flight Demonstrator (Nexus), NTRS, 2001
12. Integral Systems, Inc., EPOCH T&C STOL Programmer's Reference Manual version 2.6.7 (ISI-EPOCH-0087), Maryland, 1992
13. EGSE User Manual, EADS/Astrium
14. P. A. Dubock, F. Spoto, J. Simpson, D. Spencer, E. Schutte & H. Sontag, "The Envisat Satellite and Its Integration", ESA bulletin 106, 2001, pp.26-45
15. KARI-SFT, SOCE (Satellite Overall Control Equipment) Operator Manual, Daejeon, 2007