

공통운영환경 세그먼트 개발 방법

손명근, 고현민, 윤경아, 배두환
한국과학기술원 전자전산학과 전산학전공
{mgson, hmko, kayoon, bae}@salmosa.kaist.ac.kr

A Development Method for Common Operating Environment Segments

Myung-Geun Son, Hyun-Min Ko, Kyung-A Yoon, and Doo-Hwan Bae
Div. of CS, Dept. of EECS, Korea Advanced Institute of Science and Technology

요 약

공통운영환경 세그먼트 개발 방법론으로 I&RTS 에서 제시하는 개발 절차는 상위수준의 프로토타입으로 실제 세그먼트를 개발하는 개발자들에게 구체적인 지침으로서의 역할을 하지 못한다. 이러한 문제점을 극복해 보고자 현재 일반 객체지향/컴포넌트 개발 프로세스로 각광 받고 있는 RUP 를 도입하여 I&RTS 에서 제시하는 개발 프로세스를 보다 구체화하여 공통운영환경 세그먼트 개발 방법을 제안하고, 세그먼트 개발사례를 통해 제안된 개발 방법의 적합성을 검증한다.

1. 서론

지금까지 구축되어진 대부분의 국방 정보 체계들의 대다수가 유사하거나 동일한 기능들을 독립적으로 개발하였기 때문에 정보 체계간 상호운용성과 통합이 미흡하고 중복 투자로 인한 비용이 증가되고 있는 실정이다. 이러한 정보 체계들을 통합하고 변화되는 환경에 효과적으로 대응 및 진화하기 위해 공통 기반을 공유하여 높은 수준의 상호 운용성을 제공하는 것이 필수적이다. 공통운영환경(Common Operating Environment, COE)은 이러한 요구에서 시작되었다.[1]

이미 미국과 영국에서는 공통운영환경에 대한 연구가 많이 진행되고 있고, 특히 공통운영환경 상에서 소프트웨어 또는 데이터 단위들의 집합인 세그먼트를 개발하기 위해, 미국에서는 I&RTS에서 제시하는 개발 절차를 채택하고 있다. 그러나 I&RTS에서 제시하는 개발 절차는 구체화된 개발 방법론이 아닌 개방된 상위 수준의 프로토타입 형태로 제시되고 있으므로 실제 공통운영환경 세그먼트를 개발하는 개발자를 위한 지침서로서의 역할이 어렵다.

위와 같은 문제점을 해결하기 위해 본 논문은 I&RTS에서 제시한 개발 절차에 컴포넌트 기반의 개발 방법론으로 최근에 주목 받고 있는 RUP를 결합하여 공통운영환경에 적합하고 보다 구체적인 개발 방법을 제안하였다.

본 논문의 구성은 다음과 같다. 2장에서는 공통운영환경 세그먼트 개발 절차 및 대표적인 컴포넌트 개발 방법론인 RUP에 대해 설명하고, 3장에서는 2장의 개발 방법론 및 프로세스를 적용한 구체적인 공통운영환경 세그먼트 개발 방법을 제안한다. 4장에서는 3장에서 제안된 개발 방법에 따라 세그먼트를 개발한 사례를 보여주고, 마지막 5장에서는 결론 및 향후 연구 방향을 제시한다.

2. 관련연구

2.1. I&RTS (Integration and Runtime Specification)

국방 정보 체계간 상호 운용성, 재사용, 통합을 증진시키기 위해 채택한 공통운영환경 세그먼트 개발 방법론은 I&RTS에 기반하도록 제안하고 있다. I&RTS에서 제시하는 개발 절차는 구체화된 개발 방법론이 아닌 다른 어떤 개발 방법론과도 결합하여 사용할 수 있도록 하는 개방된 형태의 상위수준의 프로토타입이다. 그림 1에서는 I&RTS의 개발 절차를 간략히 보이고 있다.

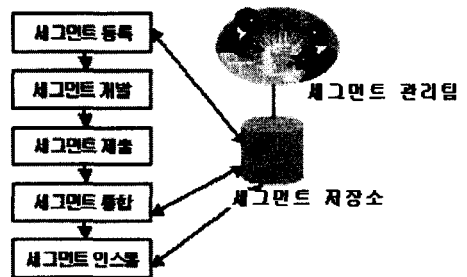


그림 1. I&RTS 개발 절차

2.2 RUP (Rational Unified Process)

RUP[3]는 객체 지향 기법을 근간으로 하는 프로세스이며 UML을 기본적으로 사용한다. 이것은 소프트웨어 개발에 필요로 하는 모든 관련 활동들을 총 망라한 매우 포괄적인 절차이며 유즈케이스를 기반으로 아키텍처 중심의 반복적이고 점진적인 방법론이라는 특징을 가지고 있다. 이러한 특징과 함께 RUP는 공통운영환경의 세그먼트 개발 방법론에 적용 가능하다.

3. 공통운영환경 세그먼트 개발 방법

이 장에서는 본 논문에서 제안한 RUP를 결합한 구체화된 형태의 공통운영환경 세그먼트 개발방법에 대해 설명하고자 한다.

그림 2에서와 같이 I&RTS에서 제시하고 있는 개발 절차를 기반으로 RUP를 결합하여 공통운영환경 세그먼트 개발 절차를 구체화 하였다. 각 단계는 반복적이고 점진적으로 진행되도록 하였고, 개발팀과는 별도로 모든 세그먼트에 대한 등록, 테스트, 형상 관리를 담당하는 별도의 기구를 편성, 운영하였다. 특히 세그먼트 관리팀의 역할을 강조하였고, 세그먼트 개발 전 과정에 참여하도록 하였다.

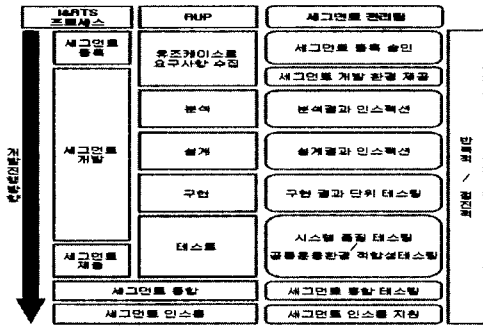


그림 2. 공통운영환경 세그먼트 개발 방법

3.1. 세그먼트 등록

세그먼트 개발에 앞서 세그먼트 정보를 등록한 후 개발에 필요한 COE-컴포넌트 세그먼트를 다운로드 받음으로써 공통운영환경을 만족시키는 세그먼트 개발이 시작될 수 있도록 한다.

3.2. 세그먼트 개발

세그먼트 개발자들에게 친숙한 UML을 모델링 언어로 채택하고, RUP의 특징 중 하나인 아키텍처 중심의 분석, 설계를 통하여 공통운영환경을 고려한 분석, 설계가 이루어 질 수 있도록 한다. 또한 개발의 모든 단계가 점진적이고, 반복적으로 진행될 수 있도록 하며, 분석 및 설계, 구현 단계가 끝날 때 마다 별도의 기구를 통해 세그먼트가 공통운영환경에 적합한지를 평가한다. 이를 위해 분석, 설계 산출물에 대한 인스펙션을 실시하고, 미 국방 공통운영환경 기준표준[1]에 의해 체크리스트를 작성하여 확인 한다.

3.3. 세그먼트 제출

세그먼트 저장소에 제출하도록 하여 공통운영환경 세그먼트의 통합 및 관리가 체계적으로 이루어질 수 있도록 한다. 세그먼트를 압축하고 암호화하여 제출하고, 개발 중의 테스트 결과 역시 함께 제출하도록 한다.

3.4. 세그먼트 통합

각기 다른 환경 하에서 개발된 세그먼트들을 공통 운영환경에서 상호 운용성이 보장될 수 있도록 통합하는 단계이다. 세그먼트 통합 후에는 시스템 통합

테스팅을 실시하여 세그먼트의 안정성을 입증한다.

3.5. 세그먼트 설치

운영자에게 다운로드 될 수 있는 세그먼트의 목록을 제공하여 주고, 세그먼트가 자동으로 설치될 수 있도록 하여야 한다는 제약 사항을 두어 사용자 편의성을 보장한다.

4. 사례 연구

이 장에서는 3장에서 제안한 공통운영환경 세그먼트 개발 절차에 따라 실제 세그먼트를 개발한 사례와 이를 통해 얻을 수 있었던 교훈에 대해 설명하고자 한다. 개발된 세그먼트 TASS(Tactical Air Support System)는 공군의 항공기 지원 관리 시스템으로 J2EE기반으로 구현되었고, 미 국방 COE의 기준에서 세그먼트의 적합성 수준으로 4레벨 획득을 목표로 하였다.

4.1. COE 세그먼트 개발 방법의 적용사례

먼저 세그먼트 등록단계에서는 TASS를 개발하기 위한 환경으로 J2EE환경을 구축하였다. 또한 세그먼트 등록 정보를 작성하여, 세그먼트 저장소에 등록하고 개발팀과는 별도로 세그먼트 관리 팀을 구성하여 상호 운용성 및 재사용이 개발절차 상에서 보장될 수 있도록 하였다. 그림 3은 세그먼트 등록단계에서 사용되었던 TASS의 등록정보이다.


세그먼트 등록정보의 항목명	내용
세그먼트 명	TASS
세그먼트 용도	항의 명칭이 없다.
세그먼트 디플로이 명	C:\TASS
세그먼트 형식	소문자영어
시스템 지원	Enterprise Java Beans에서 사용하던 reserved port
용기된 디스크 지원용 요구	약 1.2-1.5Mbytes
복용 플랫폼의 맥그린종도 플랫폼	복용 플랫폼의 맥그린종도 플랫폼은 모두 Enterprise Java Beans)와 호환된다.
HTT 세그먼트를 위한 파일 불합치	HTT 세그먼트를 위한 파일 불합치 없다.
세그먼트 용인 시 불합치	Enterprise Java Beans)가 설치되어 있어야 한다.
대상 플랫폼	종류별 개정이 없다.
세그먼트에 대해 기밀한 내용을 포함	종교, 핵무기 지(核)상(核)수행 시 운영에 대해 방출지침을 신중히 하되, 세그먼트이다.
세그먼트 사용자 인터페이스의 구현된 방법	
공통된 세그먼트 인스톨	AME

그림 3. TASS 세그먼트 등록 정보

두 번째, 세그먼트 개발단계는 RUP의 개발단계를 주로 결합한 단계로, 먼저 요구사항 정의 단계에서 유스케이스를 통해 요구사항을 수집하고, 유스케이스 모델을 만들었다. 이후 분석 단계에서는 요구사항으로부터 시스템 사양을 추출해 내기 위하여 시스템 아키텍처, 유스케이스, 클래스, 그리고 패키지 분석을 실시하였다. 특히, 시스템 아키텍처 분석 시도 메인 모델링을 통해 시스템 아키텍처를 분석하였고, 동시에 공통운영환경에 대한 만족여부도 고려하였다. 분석 단계의 산출물들을 기반으로 시스템 디자인을 실시하였는데, 이 단계에서는 아키텍처 디자인, 유스케이스 디자인, 클래스 디자인, 그리고 서브시스템 디자인을 하였다. 디자인 단계에서는 Deployment 모델을 통해 EJB를 이용한 클라이언트/서버 구조의 전체 아키텍처를 도출하여 플랫폼 독립적인 분산환경 지원에 대한 요구사항을 고려하였다. 그리고 디

자인 산출물들의 COE환경 적합성 여부를 인스펙션을 통해 확인하고, 확인 절차를 개발팀 자체뿐 아니라 세그먼트 관리를 팀에서도 실시하게 함으로써 개발 진행 도중 발생할 수 있는 문제점들을 보완하였다. 구현단계에서는 개발환경 설정을 위해 COE를 로드하였고, TASS의 클래스와 객체를 컴포넌트로 구현하여 시스템을 완성하였다. 개발단계의 마지막인 테스트 단계에서는 품질에 관한 테스트 이외에도 COE환경에서 요구하는 조건들을 세그먼트가 만족하는지 여부를 판별하는데 지원하는 "VerifySeg"라는 도구를 개발하여 사용함으로써 TASS에 대한 COE환경 만족성 평가를 실시하였다.

세째, 세그먼트 제출 단계에서는 TASS의 인터페이스와, 충돌 여부를 테스트한 후 이를 세그먼트 관리팀으로 보냈고, 넷째, 세그먼트 통합단계에서는 세그먼트 관리 팀에서 I&RTS에서 제공하는 적합도 점검리스트를 기반으로 TASS의 적합성 수준을 평가한 후 공통운용환경 하에서 AMIS(Aircraft Maintenance Information System)라는 다른 세그먼트와의 통합을 실시하여 상호 운용성이 보장되는지 확인하였다. 마지막 다섯째, 세그먼트 설치단계에서는 세그먼트의 등록정보가 기술된 XML 디스크립트 파일을 이용하여 자동설치를 지원하기 위해 본 연구에서 개발되어진 COEInstaller라는 도구를 이용하여 TASS를 지정된 운영사이트에 자동 설치 하였다.

4.2. 개발방법의 검증

본 연구에서는 산출물인 세그먼트의 COE 표준에 대한 만족도를 확인해 봄으로써 제안한 개발방법의 COE에 대한 적합성을 간접적으로 검증해보았다. 평가 항목은 I&RTS의 적합도 점검리스트[1]에서 제공되는 항목들로 상호운용성, 재사용성 보장여부와 규정된 개발절차를 준수했는지에 관한 내용들로 구성되어 있다. 표 1은 TASS 세그먼트를 평가한 결과로 이를 통해 TASS가 미 국방 COE기준 레벨 4를 만족하고 있음을 알 수 있다. 따라서 RUP를 결합한 형태의 개발방법이 미 국방 COE기준에 부합하고 있음을 간접적으로 확인할 수 있다.

표 1. 적합성 평가결과

적합도 레벨	평가 항목 수	T (True)	F (False)	N/A (Non-Applicable)	False 비율
1	16	9	0	7	0
2	15	7	0	8	0
3	18	9	0	9	0
4	17	10	0	7	0
5	151	41	47	63	31.1
6	101	15	41	45	40.6
7	54	10	23	21	42.6
8	23	3	12	8	52.2
Others	15	2	2	11	13.3
계	410	106	125	179	

4.3. 교훈

사례연구를 통해 얻은 교훈은 다음과 같다.
 - 개발 절차 프레임워크와 개발방법론의 결합을 통한 구체적인 지침 제공 :
 I&RTS에서 제안한 세그먼트 개발 절차 프레임워크에 RUP를 테일러링한 개발방법을 적용함으로써 I&RTS에서 제한하는 사항들을 만족하며 쉽고 효율적으로 세그먼트를 개발할 수 있었다.
 - 세그먼트 통합 단계의 구체화 필요 :
 RUP와의 테일러링 시 I&RTS의 개발절차 프레임워크의 세그먼트 개발단계에 집중하는 경향이 있어 세그먼트 통합 단계에서 주로 하는 테스트이나 적합성 평가에 대한 보다 구체화된 지침이 필요하였다.
 - 개발 절차 상에서 상호운용성과 재사용성의 보장 :
 개발 절차 상에서 상호운용성과 재사용성을 보장하기 위한 방법 중 하나로 세그먼트 관리팀을 운영하였다. 이를 통해 여러 단계에서 세그먼트의 상호운용성과 재사용성을 검증하게 되어 절차 상에서 이를 철저히 보장할 수 있게 되었다.

5. 결론

본 논문에서는 I&RTS의 세그먼트 개발 절차 프로토타입에 상호운용성과 재사용성과 같은 공통운용환경의 목적에 적합한 RUP를 결합하여 구체적인 개발방법을 제안하였다.
 제안한 개발 방법에 따라 사례연구로 TASS라는 세그먼트를 개발하였고, 개발방법의 검증을 위해 개발된 세그먼트가 COE 기준을 따르는지 여부를 확인하였다. 사례연구를 통해 얻은 교훈을 바탕으로, 향후 연구에서는 본 논문에서 제안된 세그먼트 개발방법을 다양한 사례에 적용하고 부족한 면에 대해서는 다른 개발 방법론을 결합하여 이를 보완함으로써 COE의 특징인 상호운용성과 재사용성의 보장을 강화할 것이다.

참고문헌

[1] US DoD, "Defense Information Infrastructure (DII) Common Operating Environment(COE) Integration and Runtime Specification(I&RTS)," DISA, 1999.
 [2] "Defense Information Infrastructure (DII) Common Operating Environment(COE) Java Development Guidelines, Version 1.0," May.
 [3] Jack GreenField "UML Profile for EJB", Rational Software Corporation, 1999
 [4] I. Jacobson, G. Booch, J. Rumbaugh, "The Unified Software Development Process," Addison Wesley, 1998.
 [5] Gregory Frazier, "The DII COE: An Enterprise Framework," Science Applications International Corporation(SAIC), Oct 2001.
 [6] KAIST, 국방정보체계 공통운용환경의 효과적 구축방안 연구, 2000