

컴포넌트 기반 국방 소프트웨어 개발 프로세스간 산출물 공유 기법

조혜경⁰ 고인영 이준기 박승진⁺

한국정보통신대학교(ICU), 국방과학연구소⁺

{hgcho⁰, iko, haedal2}@icu.ac.kr, hanearl@dreamwiz.com⁺

An Artifact-sharing Method across Multiple Component-based Military Software Development Processes

Hye-Kyeong Jo⁰ In-Young Ko, Junki Lee, Sungjin Park⁺

Information and Communications University (ICU), Korean Agency for Defense Development⁺

요약

RUP과 ADDMe는 국내 국방 도메인에서 가장 많이 사용되는 컴포넌트 기반 소프트웨어 개발 프로세스이다. ADDMe는 RUP을 기반으로 만들어졌고 국방 컴포넌트 기반 소프트웨어 개발에 표준화되었다. 국방 도메인에서 소프트웨어 재사용 증진과 소프트웨어 이해를 돋기 위해서는 RUP과 ADDMe로 개발된 프로젝트 산출물을 정보들의 경색 서비스를 제공해야 한다. 또한, 기존 이미 RUP로 개발된 것과 유사한 기능을 일부 제공하는 프로젝트를 ADDMe로 개발할 경우 현재 작성중인 ADDMe 산출물과 일정히 연관된 기존 RUP 산출물을 검색한 후 재사용할 가능성이 크다. 이를 위해서는 RUP과 ADDMe 프로세스에 존재하는 산출물들 간의 상관관계 매핑이 먼저 요구된다. 그러나 이 두 프로세스들에 존재하는 산출물 연관성 매핑 연구가 현재 존재하지 않는다. 본 논문은 위의 두 프로세스들을 분석·비교하여 온톨로지 기반 산출물 분류를 수행한다. 그리하여, 본 연구는 두 프로세스 내의 산출물들이 의미적으로 서로 어떠한 연관 관계에 있는지 어떠한 정보를 포함하는지 파악할 수 있게 하여 산출물 검색을 통한 컴포넌트 재사용을 돋는다.

1. 서 론

국내 국방 도메인에서는 소프트웨어 개발 비용의 절감을 위해 많은 소프트웨어 컴포넌트들이 개발되어 왔다. 그러나 다양한 컴포넌트를 개발하였지만 이것들을 효율적으로 재사용하기 위한 환경은 구축되지 않았다. 본 논문은 국방 소프트웨어 컴포넌트의 재사용 환경인 '국방 소프트웨어 컴포넌트 그리드'의 구축 과정에서 기본 연구로 수행된 것이다.

국내 국방 도메인에서 대표적인 소프트웨어 재사용 연구로는 MND-AF(Ministry of National Defense Architecture Framework), COE(Common Operating Environment), ADDMe(Advanced Defense component Development Methodology)가 있다. 이때, MND-AF는 국방 아키텍쳐 프레임워크, COE는 공통운용환경, ADDMe는 국방 컴포넌트 기반 개발 방언론을 의미한다. MND-AF는 국방 분야의 정보 시스템들 간의 통합을 위해 각종 아키텍처 산출물을 모델링하고 설계하기 위한 지침을 제공한다^[1]. 즉, MND-AF는 국방 아키텍처 모델, 산출물, 개발 절차, 참조 모델 등을 제공한다. COE는 모든 국방 정보 시스템들 간에 소프트웨어와 데이터 재사용과 구성 요소간의 통합을 위한 정보체계 기반 환경으로써 COE 아키텍처, 런 타임 환경, 데이터 환경, 참조 구현, 표준과 스펙 집합 등을 제공한다^[2]. ADDMe는 국방과학연구소에서 개발한 컴포넌트 기반 개발 프로세스로써 분석, 설계, 구현 및 테스트, 인도 단계로 구성되며 컴포넌트 개발과 컴포넌트 기반 애플리케이션 개발에 관한 지침 및 여러 산출물의 작성 기법들을 제안한다^[3].

국방 도메인에서 개발자들이 컴포넌트 기반 개발을 수행할 때 위의 재사용 연구들 중에서 가장 많이 사용하는 것이 ADDMe이다. ADDMe는 2006년에 개발이 완료되었고 국방 도메인에서 표준화되었다. 그리하여, ADDMe는 국방 도메인에서 컴포넌트 기반 개발을 수행할 때 반드시 사용해야 하는 프로세스가 되었다.

* 본 연구는 방위사업청과 국방과학연구소의 지원으로 수행되었습니다. (2007-SW-12-DM-01)

ADDMe가 개발되기 이전에 국방 도메인에서 소프트웨어 통합 업체들은 RUP(Rational Unified Process)을 이용하여 컴포넌트들을 개발해 왔다. 그리하여, ADDMe로 새로운 컴포넌트 기반의 개발 프로젝트를 수행할 때 밀접히 관련된 기존 RUP를 활용해 수행된 프로젝트 결과물인 산출물을 참고하거나 재사용할 필요성이 대두되었다. 그러나, 국방 도메인에서 개발자들은 ADDMe와 RUP 프로세스에 존재하는 여러 산출물들 간의 관련성을 명확히 파악하기 어렵고 각 프로세스에 산출물들의 내용 파악에 어려움이 존재하게 되었다. 이러한 문제 해결을 위해서는 RUP과 ADDMe에 존재하는 여러 산출물들 간의 의미적인 연관성에 따른 산출물 분류가 필요하다.

그리하여, 본 연구는 RUP과 ADDMe 프로세스를 분석하여 각 프로세스에 존재하는 산출물들의 내용 파악을 통해 여러 산출물을 연관성에 따라 의미적 분류를 수행하였다. 이를 위해 서로 다른 프로세스에 존재하는 산출물의 분류 기법을 제시하였으며 그 산출물 분류 결과를 온톨로지 기반으로 표현하였고 프로토타입 구현으로 그 분류 결과의 활용 방법을 보여준다. 본 연구의 결과는 RUP과 ADDMe이외의 다른 컴포넌트 기반 개발 프로세스에 쉽게 확장될 수 있으며 프로세스들 간의 관련성 매핑, 프로세스 이해 및 적용에 도움이 될 것이다.

본 논문의 구성은 다음과 같다. 2절에서는 관련 연구로 소프트웨어 개발 프로세스, 프로세스의 산출물 명세 방법에 대해 기술한다. 3절에서는 ADDMe와 RUP를 분석하고 두 프로세스의 구조를 상세히 기술한다. 4절에서는 ADDMe와 RUP에 존재하는 산출물을 시맨틱 네트워크 구조로 의미적 분류를 수행한다. 5절에서는 결론 및 향후 연구와 본 연구 기대 효과에 대해 기술한다.

2. 관련 연구

2.1 소프트웨어 개발 프로세스

소프트웨어 개발 방법론(또는 프로세스)들을 분류하면 객체지향 방법론, 컴포넌트 기반 개발 방법론, 제품 라인 공학 방법론, 객체지향 방법론의 프레임워크로 크게 나눌 수 있다^[4]. 각

체지향 방법론으로 대표적인 것은 OMT(Object Modeling Technique), Fusion, ROOM (Real-Time Object-Oriented Modeling), HOOD (Hierarchical Object-Oriented Design), OORAM 등이 있다. CBD 방법론의 대표적인 것으로는 Catalysis, Select Perspective, UML components가 있다. 제품라인 공학 방법론으로 대표적인 것은 COPA, FAST, FORM, Kobra and QADA가 있다. 객체지향 방법론의 프레임 워크는 RUP, OPEN(Object-Oriented, Process, Environment, and Notation)이 있으며 이것들은 현대 모든 소프트웨어 공학적 최적의 실제(Best Practice)를 포함하고 있어 범위가 너무 넓기 때문에 명백한 사용자 지침이 제공되지 않는 단점이 있다.

위의 소프트웨어 개발 방법론 분류에 따라 대표적인 컴포넌트 기반 개발 방법론에 대해 간단히 기술한다. Catalysis는 문제 영역 또는 비즈니스, 컴포넌트 상세화, 컴포넌트 구현의 세 가지 수준에서 모델링을 정의한다. 문제 영역 또는 비즈니스는 시스템의 사용자와 관련된 개념을 기술하는 시스템의 외부적인 것을 의미한다. 컴포넌트 상세화는 외부적으로 보여지는 행동을 캡처하는 시스템 또는 컴포넌트의 경계를 나타낸다. 컴포넌트 구현은 하나의 컴포넌트가 작은 부품들로부터 만들어지는 방법을 기술하는 컴포넌트 내부를 가리킨다.

Select Perspective는 개발의 시작점으로서 비즈니스 프로세스 모델링의 중요성을 강조하고 시스템 독립적인 비즈니스 프로세스를 관심이 있는 시스템의 구현 지향적인 모델로 차례대로 변형하는 명확한 프로세스를 따른다^[5]. Select Perspective에서 모델링 순서는 비즈니스 프로세스 모델 → 사용 사례 모델과 상호작용 다이어그램 → 클래스 다이어그램과 상태 머신 다이어그램의 작성이다.

UML components는 RUP에 기반한 컴포넌트 기반 개발 방법론이다^[6]. UML components는 요구사항과 상세화를 두 가지 주요 작업 흐름으로 식별하였고 각 작업흐름에서 생성되는 모델들은 다음과 같다. 요구사항 작업 흐름에는 비즈니스 개념 모델과 사용 사례 모델이 작성되며 상세화 작업 흐름에는 비즈니스 타입 모델, 인터페이스 모델, 컴포넌트 상세화 모델, 컴포넌트 아키텍처가 작성된다. 앞의 상세화 작업 흐름은 다시 3개의 서브 작업 흐름으로 구성되는 데 그것들은 컴포넌트 식별, 컴포넌트 상호작용, 컴포넌트 상세화 작업 흐름들이다.

2.2 프로세스 산출물의 명세 방법

컴포넌트 기반 소프트웨어 개발 프로세스를 이용하여 소프트웨어를 개발할 때 가장 중요하게 취급되는 것은 컴포넌트이다. 그리하여, 컴포넌트 기반 소프트웨어 개발 프로세스에 존재하는 여러 산출물들 중에서 가장 대표적인 산출물은 컴포넌트 스펙이 된다. 기타, 다른 산출물들은 프로세스 내에서 대부분 모델 및 텍스트 형태로 명세가 수행된다. 그래서 본 절은 컴포넌트 스펙에 포함되는 내용과 그 내용들을 표현하는 명세 기법에 대한 관련 연구들을 기술한다.

컴포넌트 스펙은 외부적으로 시작화될 수 있는 컴포넌트의 특성을 정의하고 그 컴포넌트의 외부 환경에 대한 기대를 정의한다^[4]. 즉, 컴포넌트 스펙은 컴포넌트 인터페이스와 요구사항들을 정의한다. 컴포넌트 스펙은 CASE(Computer Aided Software Engineering) 도구들을 이용하여 모델들로부터 소스코드를 자동적으로 생성하는 데 가장 빈번히 이용된다.

Beugnard는 컴포넌트의 클라이언트와 공급자 사이의 약속인 계약(Contracts) 개념을 도입하여 컴포넌트를 정적, 행동, 동기화 등의 관점에서 기술하였다^[7]. Fettke는 Beugnard 연구를 확장해 인터페이스, 행동, 상호작용, 품질, 전문용어, 임무, 마케팅 측면에서 컴포넌트를 기술하였다^[8]. 또한, Bachmann은 컴포넌트를 기능적 측면과 비 기능적 측면에서 기술하였다^[9].

위와 같이 컴포넌트 스펙에 포함되는 내용들이 정의된다면

그 내용들을 표현하는 기법들이 고려되어야 한다. 이러한 접근 방법은 컴포넌트 스펙뿐만 아니라 컴포넌트 기반 개발 프로세스에 있는 다른 산출물에도 적용된다. 컴포넌트 스펙의 표현 기법에는 CBD 96과 Catalysis와 같은 방법론에서 제공하는 기법, EJB, CCM, COM+와 같은 컴포넌트 구현 플랫폼에서 제공하는 기법, Componentware, Piccola, Object-Z와 같은 정형 언어를 이용한 기법, 온톨로지를 이용한 기법 등이 있다.

2.3 기존 연구의 문제점

지금까지 많은 소프트웨어 개발 프로세스들이 출현하였다가 사라지기를 반복해 왔다. 그리하여, 소프트웨어 프로세스 표준화를 위해 ISO/IEC 12207^[10]과 같은 연구가 수행되었으나 이것은 생생주기별 프로세스 활동에 관한 것이다. 또한, OMG (Object Management Group)의 프로세스 산출물 명세 연구인 SPAM(Software Process Engineering Metamodel Specification)이 있으나 SPAM은 단순히 프로세스 산출물의 내부 구성 요소에 대한 메타모델을 제공하고 새로운 프로세스를 정의하고자 할 때 이 메타모델의 사용을 권장하고 있다.

그러나, 프로세스들 간의 연관성 특히 프로세스 산출물들 간의 연관성 연구는 아직 수행되지 않았다. 만일, 특정 도메인에서 사용되는 프로세스들이 한정되어 있다면 그 프로세스들의 산출물 재사용을 증진하기 위해 의미적인 산출물 연관성 연구는 특히 필요하다. 이에 본 논문은 국방 도메인에서 컴포넌트 개발을 위해 가장 많이 사용되는 개발 방법론들을 분석하여 의미적인 산출물 분류를 수행하였다. 이를 위해 온톨로지에서 지식 표현 기법 중 하나인 시멘틱 네트워크를 활용하여 국방 도메인의 프로세스 산출물들을 분류하였다.

3. 프로세스의 구조

RUP과 ADDMe는 한국의 국방 도메인에서 컴포넌트 개발을 위해 가장 많이 사용된다. 그래서 본 절은 RUP과 ADDMe에 대해 기술한다.

3.1 RUP(Rational Unified Process)

RUP은 사용 사례 기반 프로세스이다^[11]. 이것은 RUP에 존재하는 프로세스 단계의 대부분 활동에서 사용 사례를 가지고 현재 활동의 작업이 시작된다는 것이다. RUP은 아키텍처 중심 적이고 반복적이고 점진적인 프로세스이다. 그러나, RUP은 각 산출물의 내용을 명백히 정의하지 않는다. 그리하여, RUP을 이용하는 각 개발자들은 각 단계에 존재하는 산출물들의 정의에 맞게 자신이 임의대로 산출물 양식을 만들고 그 내용을 기술해야 한다. 이것은 RUP이 산출물 작성의 융통성을 제공하나 기술해야 하는 명확한 산출물 양식이 없기 때문에 개발자들에게 오히려 혼란을 증가시킬 수 있다. 그러나, ADDMe는 모든 산출물의 상세한 양식을 제공하여 이러한 문제를 해결한다.

또한, RUP은 컴포넌트 기반 개발 프로세스가 아니고 객체지향 방법론의 프레임 워크이다. RUP은 컴포넌트 보다는 클래스나 객체의 식별 및 기술에 더 많은 초점을 둔다. 소프트웨어 개발 현장에서는 개발 조직에 고유한 컴포넌트 기반 개발 프로세스가 존재하지 않으면 컴포넌트를 개발할 때 RUP을 많이 사용해 왔다. 국방 도메인에서도 마찬가지이다. 그러나, RUP은 컴포넌트 식별 및 컴포넌트 스펙을 제외하고는 현존하는 여러 컴포넌트 기반 소프트웨어 개발 프로세스들과 매우 유사하다.

3.2 ADDMe

ADDMe는 한국의 국방 도메인에서 개발된 컴포넌트 기반 개발 프로세스이다^[12]. ADDMe는 국방 도메인에서 엄격한 사용을 위해 표준화되었다. ADDMe는 RUP와 여러 컴포넌트 기반 개발 프로세스들로부터 영향을 받았다 다른 컴포넌트 기반

개발 프로세스들의 많은 특징들이 ADDMe에서 응용되었다. 그리하여, ADDMe는 국방 도메인뿐만 아니라 다른 도메인에서 폭넓게 사용될 수 있다. ADDMe의 개발은 2006년 8월에 종료되었고 ADDMe는 명백한 지침 및 가이드라인이 제공되어 적용이 쉽다. ADDMe는 ISO/IEC 12207과 MIL-STD-498에 순응한다. ISO/IEC 12207은 소프트웨어 개발 생명 주기 프로세스의 국제 표준이다. MIL-STD-498은 개발자들이 국방 도메인에 있는 시스템들을 개발하는 동안에 생성해야 되는 산출물을 정의한 것이다.

3.3 RUP과 ADDMe 구조

그림 1은 RUP의 업무모델링, 요구분석, 분석과 설계와 같은 워크플로우를 보여준다. 각 워크플로우는 작업자, 산출물, 활동으로 구성되며 하나의 워크플로우는 여러 활동으로 세분화된다. 각 작업자는 워크플로우의 어느 활동에서 그 활동의 결과로 산출물을 생성하게 된다. 그림 1에서 RUP의 분석과 설계 워크플로우는 아키텍처 설계, 서브시스템 설계, 데이터베이스 설계와 같은 활동의 결과로 소프트웨어 아키텍처 문서, 설계 서브시스템, 데이터 모델과 같은 산출물을 생성하게 된다.

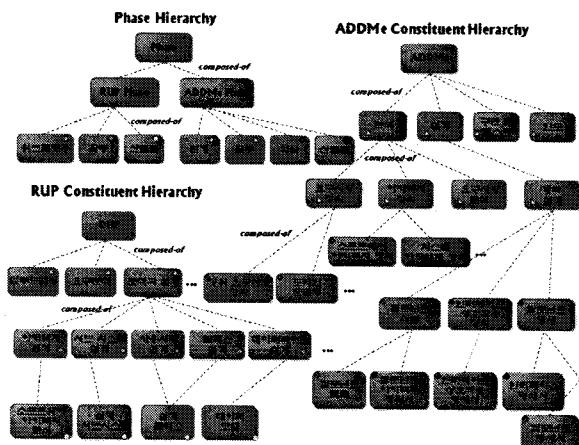


그림 1 RUP와 ADDMe 프로세스의 구조

ADDMe는 4 단계, 12 활동, 37 작업으로 구성된다. ADDMe는 분석, 설계, 구현 및 테스트, 인도 단계를 가진다. ADDMe의 각 단계에는 여러 활동들이 존재하며 그 활동들은 다시 세부적인 작업으로 분할되고 특정 작업의 수행 결과로 산출물들이 생성된다. 그림 1에서 ADDMe의 설계 단계에는 개략 설계 활동이 있고 개략 설계 활동에는 다시 컴포넌트 식별, 컴포넌트 명세의 작업이 존재하며 그 작업들의 결과로 컴포넌트 목록, 컴포넌트 아키텍처 정의서, 인터페이스 명세서, 컴포넌트 명세서 등의 산출물이 생성되게 된다.

4. 프로세스의 산출물 분류

4.1 산출물의 분류 절차

산출물의 분류 결과는 개발자들이 서로 다른 소프트웨어 개발 프로세스를 이용하여 개발된 산출물을 효율적으로 브라우징하게 하고 수행된 프로젝트 결과를 개발자들이 서로 공유할 수 있게 돋는다. RUP과 ADDMe 프로세스의 산출물 분류는 아래 절차에 따라 수행되며 그 결과를 이용하여 산출물들 간의 상호관계를 파악할 수 있다. 또한, 아래의 산출물 분류 절차는 다른 프로세스들에 적용하여 여러 프로세스를 위한 산출물 분류 기법으로 쉽게 일반화될 수 있다.

- ① 서로 다른 소프트웨어 개발 프로세스의 서로 다른 단계들(Phases)에서 공통 단계를 정의한다.
- ② 서로 다른 프로세스의 산출물 관계를 분석한 후에 산출물 분류를 위한 새로운 개념(Concept)들을 생성한다.
- ③ 서로 다른 프로세스에 있는 산출물들의 동의어를 식별한다.
- ④ 산출물들을 분류하고 각 산출물 개념의 속성(Property)들을 실제 산출물에서 추출한다.
- ⑤ 분류된 모든 각 산출물이 다른 프로세스에 어느 산출물과 밀접히 연관되는지 동의어 리스트를 참조해 정의한다.

위에서 ①번은 가장 주의 깊게 수행되어야 한다. 그 이유는 프로세스마다 최상위 단계를 가리키는 말이 서로 다르기 때문이다. 예를 들면, 그것을 RUP에서는 작업 흐름(Workflow)으로 ADDMe에서는 단계(Step)로 또 다른 프로세스들에서는 국면(Phases)로 부른다. 또한, 프로세스에 따라 산출물도 인공물(Artifact) 또는 작업 결과(Workproduct)로 부른다. 이 문제는 SPAM의 도움을 받아 프로세스 공통 용어로 해결할 수 있다.

4.2 산출물의 분류 결과

그림 1의 RUP와 ADDMe 구조로부터 두 프로세스의 공통 단계로 요구 분석, 분석 설계 구현, 테스트가 식별되었다. 분석 설계 구현에 너무 많은 산출물들이 존재하게 되나 이 단계에서 RUP와 ADDMe 산출물들이 서로 밀접히 연관되므로 이렇게 구분한다. 본 절에서 산출물의 분류 결과는 이 최상위 공통 단계에 따라 상세히 기술된다.

그림 2, 3, 4에서 개념(즉, 타원)에 △나 ○가 없는 것은 산출물 분류를 위해 새롭게 생성된 것이며 실제 산출물이 아니다. 반대로, 개념에 △나 ○가 있는 것은 실제 산출물이며 한 개념 안에 △나 ○가 모두 표시된 것은 그 산출물이 RUP와 ADDMe에 모두 존재함을 의미한다. 각 개념의 속성들은 타원 바로 밑에 기술되거나 또는 타원과 연결된 박스 안에 기술된다.

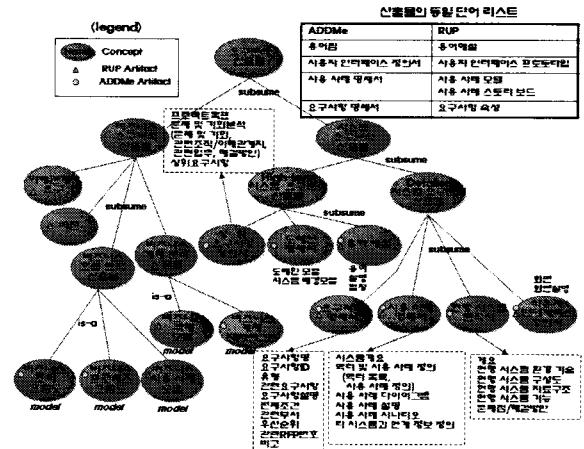


그림 2 요구분석 단계의 산출물 분류

그림 2에서 'model' 표시된 산출물은 산출물 정보가 클래스 다이어그램 또는 사용 사례 다이어그램과 같은 형태로 존재하나 특정한 속성 값이 없는 경우이다. 또한, 그림 2, 3, 4에서 산출물의 동일 단어 리스트의 테이블은 ADDMe와 RUP의 산출물들을 비교하여 서로 다른 이름을 가지고 있으나 포함하는 정보가 거의 동일한 경우의 산출물들을 정리한 것이다. 이때, 서로 다른 프로세스로 개발되거나 유사한 기능을 일부 제공하는 프로젝트인 경우 산출물의 동일 단어 리스트를 활용해 프로젝

트들 간의 산출물 재사용을 도울 수 있다.

또한, 그림 4에서 컴포넌트 테스트 설계서, 통합 테스트 설계서, 시스템 테스트 설계서는 공통적으로 테스트 단위 테스 대상 및 범위, 테스트 케이스 설계, 테스트 케이스 수행 절차의 속성들을 가진다. RUP과 ADDMe는 다른 단계들 보다 테스트 단계에서 몇 가지 차이점을 보인다. RUP의 테스트 컴포넌트 산출물은 테스트 코드를 의미하며 RUP의 결점(Defects) 산출물은 테스트 평가 결과로 탐지된 결점들을 나열한 것이다. 그러나, ADDMe는 컴포넌트 테스트 설계서 및 결과서, 통합 테스트 설계서 및 결과서, 시스템 테스트 설계서 및 결과서의 산출물을 제공하여 RUP에 비해 테스트 관련 산출물이 단순하다. 결론적으로, ADDMe는 하나의 산출물 내의 정보가 RUP보다 더 상세하고 RUP은 ADDMe보다 여러 도메인에서의 많은 내용들을 다루기 위해 다양한 산출물들을 제시하고 있다.

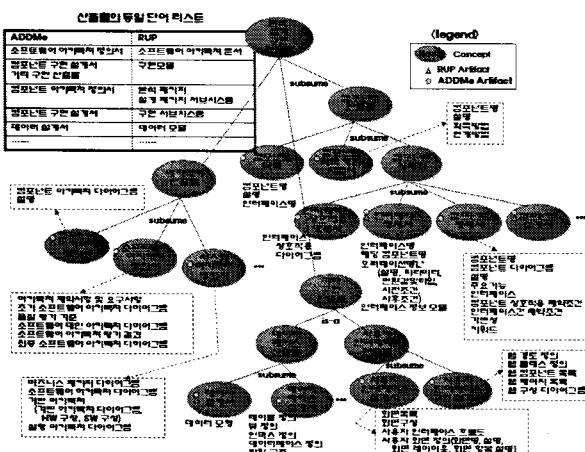


그림 3 분석·설계·구현 단계의 산출물 분류

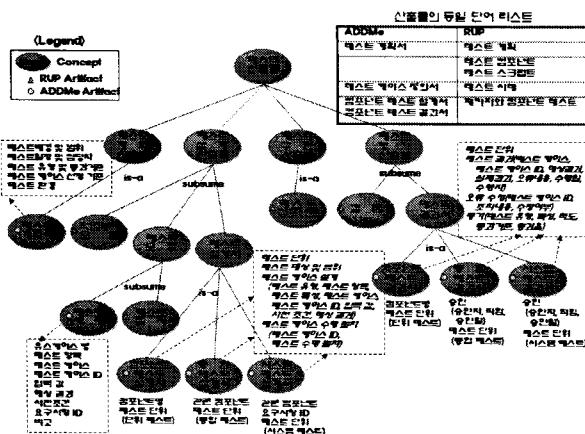


그림 4 테스트 단계의 산출물 분류

4.3 구현

온톨로지는 “공유하는 개념화의 형식적이고 명확한 형세”이다[13]. 온톨로지는 세상의 특정 분야에 관련된 용어(개념)들을 정의하고 이들 간의 관계들로 구성된 일종의 사전이다. 온톨로지의 지식 표현 기법에는 시맨틱 네트워크, 프레임, 규칙, 일차 논리, 기술 논리(Description Logic)가 있다.

본 논문은 이해의 용이성을 위해 시맨틱 네트워크 기법을 사용해 산출물 분류 결과를 앞 절과 같이 기술하였다. 그 시맨틱 네트워크 표현들은 또한 Protégé를 이용해 기술되었다. 다음 그림 5는 본 논문의 산출물 분류 결과를 적용한 것이다. 그림 5는 국방 도메인에서 산출물 재사용 저장소의 메인 사용자 인터페이스 화면이다. 현재 사용하는 프로세스가 ADDMe 또는 RUP인가에 따라 그림 5의 화면 왼쪽 프레임 아래 내용이 변경된다. 그리고, 현재 수행 중인 프로젝트의 결과 산출물들을 그 산출물 계층구조를 활용해 탐색할 수 있으며 현재 작성 중인 산출물의 내용은 화면 오른쪽 프레임에 디스플레이 되게 된다.

온톨로지의 구성요소로는 개념, 속성, 관계, 제약조건, 공리(Axiom), 인스턴스가 있다. 온톨로지는 비형식적 온톨로지의 경우처럼 최소한의 형식적 구조를 가지고 개념과 그 개념에 대한 정의만으로도 구성될 수 있다¹⁴⁾. 그리하여 본 논문의 산출물 분류 결과는 개념 분류(Classification)에 관한 온톨로지이다. 본 논문의 산출물 온톨로지를 이용한 완벽한 추론을 제공하기 위해서 개념 및 속성 정제(Refinement), 관계 정제, 인스턴스 생성 등의 작업이 현재 진행되고 있어 프로세스의 산출물 온톨로지는 계속 발전될 것이다.

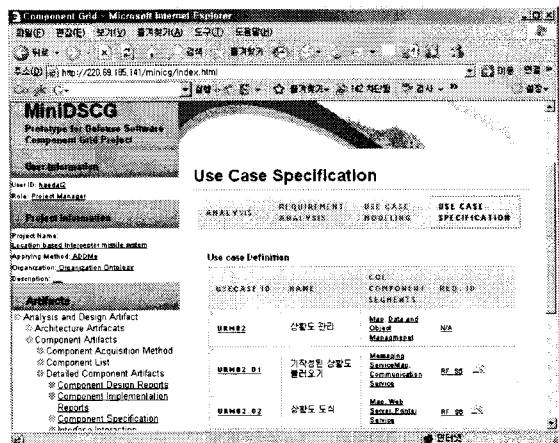


그림 5 산출물 분류 결과를 활용한 사용자 인터페이스

5. 결론

본 논문은 RUP과 ADDMe 프로세스에 존재하는 산출물들을 온톨로지를 이용해 분류하였다. RUP과 ADDMe는 국내 국방 도메인에서 컴포넌트 기반 개발을 위해 사용되는 프로세스들이다. 본 논문에서는 관련 연구로 기존 소프트웨어 프로세스들을 살펴보았고 프로세스 내의 산출물들의 내용을 식별하고 그 내용들의 표현 기법들을 컴포넌트 중심으로 알아보았다. 그리고 본 논문의 연구 배경을 설명하기 위해 국방 도메인에서 재사용 연구인 MND-AF와 COE를 기술하였다. 또한, RUP과 ADDMe를 분석한 뒤 각 프로세스의 특성을 파악해 비교 평가하였다. 그리고 제시된 산출물의 분류 결과에 따라 RUP과 ADDMe의 산출물 분류를 수행하였고 향후 산출물 저장소를 구축할 때 그 산출물 분류 결과의 이용 방법을 보여주기 위해 산출물 저장소의 사용자 인터페이스 화면을 구현하였다.

본 논문의 기본 아이디어는 대부분의 컴포넌트 기반 소프트웨어 개발 프로세스들이 비즈니스 분석, 아키텍처 설계, 컴포넌트 분석 및 설계, 클래스 분석 및 설계의 과정을 공통적으로 거친다는 점에서 출발하였다. 특히, 표준화된 UML(Unified Modeling Language)을 사용하는 프로세스들은 사용 사례 클래스, 컴포넌트, 아키텍처 관련 산출물을 공통으로 생성하기 때-

문에 여러 프로세스의 산출물들에는 많은 유사점이 존재한다.

본 논문은 산출물 브라우징을 통한 산출물 검색을 용이하게 하고 산출물 저장소 구축을 위한 산출물 분류의 기준을 제시하며 각 산출물의 소속된 범주, 각 산출물의 작성 목적, 산출물들 간의 상호 관계 파악을 용이하게 한다.

향후 연구는 다른 컴포넌트 기반 소프트웨어 개발 프로세스에 적용하여 본 논문의 산출물 분류 결과를 확장하는 것이다. 또한, 본 논문은 산출물 분류 결과 기술을 위해서 시맨틱 네트워크를 사용하였으나 그림이기 때문에 화살표들이 복잡하게 서로 얹히는 현상이 발생하였다. 그리하여, 향후에 산출물 분류 및 명세 결과를 시맨틱 네트워크 표현으로부터 서술 논리 기반으로 변환할 것이다.

6. 참고 문헌

- [1] 국방과학연구소, 국방 아키텍처 프레임워크 (MND-AF V1.0 사용자 지침서), 2005. 2.
- [2] 국방부, 공통운용환경 구축 편람, 2002.
- [3] 국방과학연구소, 국방 CBD 방법론 (ADDMe Version 1.1), 2006. 9.
- [4] Atkinson, C., Bayer, J., Bunse, C., et al., Component-based Product-line Engineering with the UML, Addison-Wesley, 2002.
- [5] Allen, P. and Frost, S., Component-Based Development for Enterprise Systems, Applying the SELECT Perspective, Cambridge University Press/SIGS, Cambridge, 1998.
- [6] Cheesman, J. and Daniels, J., UML Components: A Simple Process for Specifying Component-Based Software, Addison-Wesley, 2000.
- [7] Beugnard, A., Jézéquel, J.M., Plouzeau, N., et al., "Making Components Contract Aware", Computer 32 (7), pp. 38-45, JUL 1999.
- [8] Fettke, P. and Loos, P., "Specification of Business Components, Objects, Components, Architectures, Services, and Applications for a Networked World", Lecture Notes in Computer Science, pp. 62-75, 2003.
- [9] Bachmann, F., Bass, L., Buhman, C., et al., Volume II: Technical Concepts of Component-Based Software Engineering, CMU/SEI-2000-TR-008, 2000.
- [10] IEEE, IEEE/EIA 12207.0 Standard for Information Technology - Software Life Cycle Processes, IEEE, Mar 1998.
- [11] Jacobson, I., Booch, G., and Rumbaugh, J., The Unified Software Development Process, Rational Software Corporation, Addison-Wesley, 1999.
- [12] 정연대, 임진수, 윤희병, "국방 CBD 방법론의 현재와 미래", 정보과학회지, 제 24권, 제 9호, pp. 75-80, 2006.
- [13] Gruber, T.R., Toward Principles for the Design of Ontologies Used for Knowledge Sharing. International Workshop on Formal Ontology in Conceptual Analysis and Knowledge Representation, 1993.
- [14] 노상규·박진수, 인터넷 진화의 열쇠 온톨로지, 가즈토이, p. 26, 2007.