온톨로지를 이용한 개념형 소프트웨어 프로세스 데이터베이스 설계 및 구현

이 준 하[†] · 박 용 범[††]

요 약

온톨로지는 인간의 사고 체계와 근접한 표현을 가능하게 해주는 형식적이고 명시적인 지식베이스로 사용 가능하다. 또한 소프트웨어 프로세스란 소프트웨어 개발 역량 성숙도가 높은 조직이 공통적으로 수행하는 모범적인 체계와 절차의 집합이다. 그러나 소프트웨어 프로세스는 복잡한 개념이 거미줄처럼 얽히어 간단한 프로세스의 도입과 개선 활동마저도 가로막는 결과를 초래하게 된다. 온톨로지를 소프트웨어 개발 프로세스간의 복잡한 사상들에 적용함으로써 프로세스간의 상관관계를 개념적으로 추론하여 결과를 보여 주게 되면 소프트웨어 개발 프로세스의 도입과 개선이 용이해질 수 있다. 본 논문에서는 온톨로지 매핑을 이용하여 국제 표준인 ISO/IEC 15504와 역량성숙도 모델인 CMMI를 개념적으로 엮음으로써 구현한 개선된 프로세스 데이터베이스를 활용하기 위한 방법을 제시한다.

키워드 : CMMI, ISO/IEC15504, 온톨로지

Design of Conceptual Software Process Database, Using Ontology

Jun-Ha, Lee[†] · Young-B, Park[††]

ABSTRACT

Ontology can be used as a formal and demonstrative knowledgebase that can express the thinking process of human. Software Development Process is a collection of ideal practices and procedural system that is performed by mature organization with high capability. Due to complexity of process, however, Software development Process often results in obstruction of introducing and improving simple process activity. While introducing and improving software development process, application of ontology to complex software development process is more approachable by showing deductive results of relationship between ISO/IEC 15504 and CMMI. In this paper, we demonstrate a methodology that utilizes the improved process database conceptually mapping between ISO/IEC 15504 and CMMI using ontology.

Key Words : CMMI, ISO/IEC15504, Ontology

1. 서 론

소프트웨어의 사용 영역이 확대되고 소프트웨어의 개발 규모가 거대해지고 복잡해짐에 따라 소프트웨어 품질과 생산성에 대한 중요성이 증가하고 있다. Watts Humphrey는 "소프트웨어 시스템의 품질은 그것을 개발하고 유지보수하기 위해서 사용된 프로세스의 품질에 전적으로 달려있다"고 하였다[7]. 이러한 상황에서 소프트웨어 프로세스 개선이 생산된 소프트웨어 시스템의 품질을 결정하는 핵심으로써 높은 관심을 끌고 있다.

실제 소프트웨어 개발 현장에서 소프트웨어 프로세스 개선 업무를 수행하기 위한 많은 노력들이 있으나, 성공적인 개선에 대한 사례는 실패 사례에 비하여 극히 적다.[3] 이러한 실패의 원인 중 하나로, 소프트웨어 개발 프로세스의 복잡 난해함을 꼽을 수 있다. 이를 해결 하기 위한 노력으로 많은 연구가 진행되고 있고, 많은 조직에서 CMMI와 ISO/IEC 15504를 상호 보완하여 사용하고 있다[2].

소프트웨어 개발 프로세스는 사람들의 머릿속에 들어있는 사상을 끄집어 내어 정리한 산물로써 개념적 사상의 응집체라 할 수 있다. 또한 온톨로지는 개념적인 지식베이스 기술이다[12]. 이러한 온톨로지는 소프트웨어 개발 프로세스의 복잡한 개념적 사상을 실제화 시키기에 적합하다. 본 논문에서는 온톨로지의 매핑을 통해 CMMI와 ISO/IEC 15504를 개념적으로 엮어 상호 보완적으로 사용하기 위해 개선된 소프트웨어 프로세스 데이터베이스를 제안한다.

2장 관련연구에서는 온톨로지에 대한 개략적인 설명과, 소프트웨어 프로세스, ISO/IEC 15504, CMMI에 대해 설명하고, 3장에서는 ISO/IEC 15504와 CMMI를 매핑한 결과에

※이 연구는 2006학년도 단국대학교 대학연구비의 지원으로 연구되었음
† 준 회 원 : 단국대학교 컴퓨터과학 제학생
†† 종신회원 : 단국대학교 컴퓨터과학 교수
논문접수 : 2006년 6월 9일, 심사완료 : 2006년 12월 7일

대해 설명한 후, 개념 형 프로세스 데이터베이스 활용 도구를 소개한다. 마지막으로 4장에서는 구현된 개념 형 프로세스 데이터베이스 활용 방안에 대해서 논하고 그에 따른 향후 과제를 기술한다.

2. 관련연구

2.1 소프트웨어 개발 프로세스

소프트웨어 프로세스란 절차 중심의 조직단위의 소프트웨어 프로젝트 관리 방법론을 뜻하며, 일련의 활동, 방법, 실행 지침, 변환 등의 집합으로 정의 된다[4].

이는 소프트웨어와 연관된 자산(표준 소프트웨어 프로세스, 소프트웨어 생명주기에 대한 설명, 표준 소프트웨어 프로세스를 조정하기 위한 기준과 가이드라인, 소프트웨어 프로세스 데이터베이스, 프로세스관련 문서들의 라이브러리)을 개발하고 유지하는데 사용된다[4].

소프트웨어 프로세스 개선(Software Process Improvement: SPI)은 비즈니스 목표와 목적을 달성하기 위하여, 개선된 소프트웨어 프로세스를 설계하거나 재정의하는 방법이다. 수익성 또는 이익 증가, 수행 비용 감소 등을 목표로 SPI를 수행하게 된다[5].

SPI는 이와 같은 이익을 얻기 위한 수단으로 개선된 소프트웨어 프로세스를 만들어내는 일련의 작업이다. 다시 말해서 이익을 일정 수준 달성하기 위해 개선된 소프트웨어 프로세스를 만들어 내는 것이다. 이익은 수익성 또는 이윤 증가, 비용 절감, 상당한 비용 절약을 말한다[4].

SPI에서 일찍이 시도된 것은 적은 비용을 들여 품질을 개선하고 신뢰성을 향상시키도록 설계하는 것이다. 목표는 효율성 증가, 비용 절감, 이윤 증가이다. 전략적인 목표를 가지고 사이클 타임을 빠르게 하고, 고객 만족을 높이는 결과를 나타낸다. SPI는 새롭고 개선된 소프트웨어 프로세스를 만들어 내는 주요한 수단이기 때문에 중요하다. 이것은 가능한 적은 비용으로 중요한 경제적 이익을 이루기 위해 수행된다[4, 5].

2.1.1 ISO/IEC 15504 (Software Process Improvement Capability dEtermination:SPICE)

ISO/IEC 15504는 조직의 프로세스를 개선하기 위한 활동을 지원하기 위해서 현재의 프로세스 상태를 측정하여 성숙한 능력수준을 측정한다[14].

ISO/IEC 15504는 전체 9개의 프로세스(조달, 공급, 공학, 관리, 프로세스 개선, 자원 및 인프라, 재사용, 품질관리, 품질보증)로 구성되어 있다. 각각의 프로세스는 각각의 프로세스에는 표준에 따라 소프트웨어 심사에서 반드시 지켜져야 할 내용을 담고 있다. 프로세스 능력차원에 정의된 각각의 프로세스에 대한 능력은 Level 0에서 Level 5로 정의 되어 있으며, 각 Level은 프로세스 영역(PA: Process Area)으로 표현되며 프로세스 능력 수준 단계의 달성 정도는 프로세스 속성에 의해 측정된다[14].

(그림 1)은 ISO/IEC 15504의 프로세스 차원 속성을 나타내고 있다.

<table>
<tr><th colspan="2">PRIMARY Life Cycle Processes</th><th>ORGANIZATIONAL Life Cycle Processes</th></tr>
<tr><td>1. Acquisition Group
ACQ.1 Acquisition preparation
ACQ.2 Supplier selection
ACQ.3 Supplier monitoring
ACQ. 4 Customer acceptance</td><td>2. Supply Group
SPL.1 Supplier tendering
SPL.2 Contract agreement
SPL.3 Software release
SPL.4 Software acceptance</td><td>1. Management Group
MAN.1 Organizational alignment
MAN.2 Organization management
MAN.3 Project management
MAN.4 Quality Management
MAN.5 Risk Management
MAN.6 Measurement</td></tr>
<tr><td colspan="2">3. Engineering Group
ENG.1 Requirement elicitation
ENG.2 System requirement analysis
ENG.3 System architectural design
ENG.4 Software requirement analysis
ENG.5 Software design
ENG.6 Software construction
ENG.7 Software integration
ENG.8 Software testing
ENG.9 Software installation
ENG.10 System integration
ENG.11 System testing
ENG.12 System & software maintenance</td><td>2. Process Improvement Group
PIM.1 Process establishment
PIM.2 Process assessment
PIM.3 Process improvement
3. Resource & Infrastructure Group
RIN.1 Human resource management
RIN.2 Training
RIN.3 Knowledge management
RIN.4 Infrastructure
4. Reuse Group
REU.1 Asset management
REU.2 Reuse program management
REU.3 Domain engineerin</td></tr>
<tr><td colspan="2">4. Operation group
OPE.1 Operational use
OPE.2 Customer support</td><td></td></tr>
<tr><td colspan="3">SUPPORTING Life Cycle Processes</td></tr>
<tr><td colspan="2">1. Configuration control Group
CFG.1 Documentation Management
CFG.2 Configuration Management
CFG.3 Problem Management
CFG.4 Change Request Management</td><td>2. Quality Assurance Group
QUA.1 Quality assurance
QUA.2 Verification
QUA.3 Validation
QUA.4 Joint review
QUA.5 Audit
QUA.6 Product Evaluation</td></tr>
</table>

(그림 1) ISO/IEC 15504 프로세스 차원 속성

2.1.2 CMMI(Capability Maturity Model Integrate)

CMMI는 CMM모델이 SW CMM, SE-CMM, IPD-CMM 등으로 늘어나 사용자에게 혼선을 주어 여러 제품의 유지보수와 관리가 곤란하였다. SEI에서는 이들을 각각의 CMM모델을 통합시켰고 나아가 SPICE와의 호환성도 고려해서 개발하였다.[15] CMMI에서는 기존의 CMM에 비해 중요한 부분들이 더욱 세분화 되어 SE-CMM과 SW-CMM을 기준으로 하여 18개 프로세스영역에서 22개 프로세스영역으로, 316개의 실행에서 382개의 실행으로 늘어났다[15].

CMMI모델은 단계적 표현방법(Staged representation)과 연속적 표현방법(Continuous representation)으로 나뉠 수 있다. 단계표현방법은 5단계의 레벨구성으로 구성되어 있는데, 한번에 한 단계씩 프로세스를 개선할 수 있도록 체계적이고 구조적인 방법을 제시한다. 연속표현방법은 0에서 5까지의 6단계로 구성되어 있고, 필요한 PA 영역에 대해서 선택적으로 개선 하고자 할 때 유용한 방법이다.

CMMI에서는 소프트웨어 개발 프로세스에서 해당 성숙도를 달성하기 위해 25개의 프로세스 영역(PA: Process Area)이 규정되어 있다. <표 1>은 CMMI 단계적 표현방법의 각 레벨에 정의되어 있는 프로세스 영역이다[10].

2.2 온톨로지 (Ontology)

온톨로지란 "관심영역 내 공유된 개념화에 대한 형식적이고 명시적인 명세화 - T. Gruber"라고 정의 할 수 있다[19]. 사람들은 세상에 있는 사물이나 사건을 경험하면서, 같은 속성을 공유하고 있는 특정 범주를 추상화시키게 된다. 그리고 이 과정을 거쳐 생성된 범주들 및 그들 간의 관계를 이용하여 세상을 파악하고, 이해하게 되는데 이러한 과정을 개념화라고 하며, 이 때의 범주를 개념이라고 한다. 다시 말해, 온톨로지는 개념간의 계층구조와 관계, 제약이 표현된 데이터로 만든 개념적 지식베이스라고 할 수 있다.[12]

지식을 효과적으로 처리하기 위해 개발된 대부분의 시스템은 논리적 추론 기능을 요구한다. 해당시스템에 입력으로 들어오는 질의에 대해 적절한 답을 주기 위해서는 명시적으로 표현된 지식들을 단순히 검색하는 것만으로는 부족하고 표현되지 않은 채로 존재하는 암묵적인 지식을 추론해 낼 필요가 있는 경우가 많기 때문이다. 온톨로지는 특정 도메인을 개념 적으로 연관시켜 줌으로서 이러한 암묵적인 지식의 추론을 가능하게 한다.

〈표 1〉 성숙도레벨과 프로세스 영역

성숙도레벨	프로세스 영역
성숙도레벨 2	요구관리(REQM) 프로젝트 계획책정(PP) 프로젝트 감시와 제어(PMC) 공급자 합의 관리(SAM) 측정과 분석(MA) 프로젝트와 제품 품질 보증(PPQA) 형상관리(CM)
성숙도레벨 3	요구개발(RD) 기술적 해결(TS) 제품 통합(PI) 검증(VER) 타당성 확인(VAL) 조직 프로세스 중시(OPF) 조직 프로세스 정의(OPD) 조직훈련(OT) 통합 프로젝트 관리(IPM) 리스크 관리(RSKM) 통합 팀 편성(IT) 통합 공급자 관리(ISM) 결정분석과 해결(DAR) 통합을 위한 조직환경(OEI)
성숙도레벨 4	조직프로세스 실적(OPP) 정량적 프로젝트 관리(QPM)
성숙도레벨 5	조직 개혁과 실적(OID) 원인분석과 해결(CAR)

2.2.1 웹 온톨로지 언어

웹 온톨로지 언어는 현재 RDF(S)와 OWL을 중심으로 발전하고 있다[9]. 2004년 2월에 W3C(World Wide Web Consortium)에서는 자원 서술 프레임웍과 웹 온톨로지 언어에 관한 표준안으로 RDF와 OWL을 발표하였다[19].

RDF(Resource Description Framework)는 W3C의 가장 기본적인 시맨틱 웹 언어로서 웹에 있는 자원(resource)에 관한 메타정보를 표현하기 위한 언어이다. 이러한 RDF는 메타데이터의 구조의 상이함으로 인한 상호운용의 어려움을 해결하기 위하여 주로 사용되고 있다.

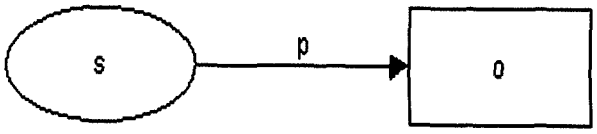
DAML+OIL은 DAML(DARPA Agent Markup Language) 프로그램의 DAML-ONT와 주로 유럽에서 개발된 OIL(Ontology Inference Language)의 결합을 통하여 만들어졌다. DAML+OIL은 관심영역(Domain)의 구조를 서술하기 위한 목적을 갖는다. 이러한 구조는 객체지향적인 방법으로 클래스와 속성을 써서 표현된다. W3C의 표준 중 하나인OWL(Web Ontology Language)은 DAML+OIL을 계승하여 발전시킨 형태이다[13].

자원이(resource) DAML+OIL이나 OWL 클래스의 인스턴스(Instance)이거나 특정속성을 갖는 것을 표현 할 때에는 이러한 목적에 적합한 RDF를 사용한다[11, 13].

2.2.2 RDF 모델

RDF문(Statement)은 특정 자원과 지정된 특성, 그리고 그 특성의 값으로 이루어진다. 이 사실을 RDF에서 표현하기 위해서는 원래의 문장을 네 개의 특성을 가진 자원으로 모델화해야 한다[17]. 위의 네 가지 특성은, Subject 특성, Predicate 특성, Object 특성, Type 특성이다. Subject특성은 모델화된 문장에 의해 기술되는 자원을 식별하고, Predicate 특성은 모델화된 문장의 원래의 특성을 식별하며, Object 특성은 모델화된 문장의 특성 값을 식별한다. 또한, Type 특성 값은 새로운 자원의 타입을 기술한다.[6]

(그림 2)은 RDF 모델에서 문장을 방향 그래프로 표현한 것이다.



(그림 2) RDF 모델의 형태

위의 (그림 2)에서 s는 RDF 문의 Subject를 나타내고, p는 predicate를, 그리고 o는 Object를 나타낸다.

2.2.3 RDQL

RDF에서는 N-Tripple이라는 구문형태를 사용하는데, N Tripple이란 RDF문장의 Subject, Predicate, Object를 말한다[16]. RDQL은 W3C에서 표준으로 제정된 RDF를 위한 질의 언어로서 SQL과 유사한 구문을 제공한다. SELECT 절에서는 N-Tripple에서 사용된 변수를 기술함으로써 변수의 결과값을 SQL의 테이블과 같은 형태로 결과값을 얻도록 하고, WHERE 절에서는 N-Tripple을 기술하여 RDF 그래프 모델상에 있는 특정한 Tripple을 검색할 수 있게 한다[18].

2.3 메타데이터의 상호운영

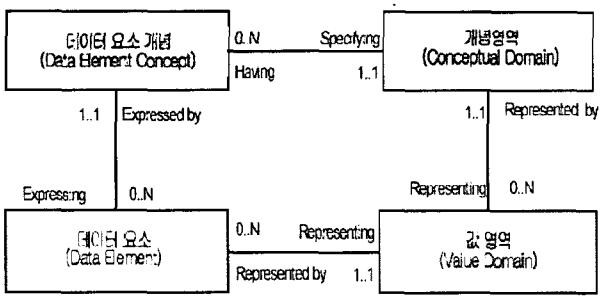
메타데이터의 상호운영성은 이용자 측면에서 볼 때 다양한 출처에서 생성되는 정보자원을 성공적으로 검색하고 그 결과를 확신할 수 있게 하는 것이다.

메타데이터의 상호운영을 확보하기 위한 방법으로 더블린 코어와 같은 하나의 표준적인 메타데이터로 통합하여 표현하는 방법과, 메타데이터 형식과 기술구조를 인정하고 RDF등을 이용하여 상호매핑을 통해 해결하는 방법, 메타데이터 온톨로지 즉 메타데이터 레지스트리에 의한 해결 방법이 있다[8].

2.3.1 메타데이터 레지스트리

메타데이터 레지스트리는 서로 다른 데이터베이스가 같은 개념에 대해 서로 다른 식별자 혹은 서로 다른 단어를 사용할 경우 이를 해결해 주기 위해 온톨로지를 이용한 메타데이터 관리시스템이다[8].

(그림 3)는 메타데이터 레지스트리의 구성요소인 데이터 요소 개념과 데이터 요소, 개념영역과 값 영역간의 관계를 보여준다.



(그림 3) 메타데이터 레지스트리의 구성요소

데이터 요소는 값 영역의 데이터 타입으로 정의될 수 있으며, 값 영역의 값은 실제 메타데이터로 정의될 수 있다.

3. 제시모델

3.1 시스템 구현 목적

CMMI는 SPICE와의 호환성을 고려하여 개발되었다. 그러므로 각각의 모델은 상호 보완하여 사용될 수 있다.

상호 보완하여 사용했을 경우, 조직은 더욱 구체적으로 프로세스를 관리할 수 있다. 예를 들어, CMMI에는 특정실행을 수행했을 때의 산출물(Typical Product)과 하부실행(Sub Practice)이 명시되어있지만, SPICE에는 명시되어있지 않다. 또 CMMI의 프레임워크에 비해, SPICE의 프레임워크가 더 상세히 명시되어있다. 그러므로 이 두 모델을 상호 보완하여 사용하였을 경우, 조직은 더욱 구체적으로 프로세스를 관리 할 수 있을 것이다.

한편, 소프트웨어 프로세스 데이터베이스란, 소프트웨어 프로젝트 산출물 및 가이드 등, 조직적으로 프로젝트를 관리 하는데 필요한 모든 데이터가 저장된 장소이다. 일반적으로 마감된 프로젝트의 결과물은 모두 프로세스 데이터베이스에 취합되어 누적되고, 그것의 분석 결과는 다음 프로젝트 개시에 영향을 미친다[3].

본 연구에서 제안되는 프로세스 데이터베이스 디자인은 CMMI와 SPICE를 서로 매핑시킴으로써 각각의 프레임워크에 맞게 데이터베이스 구조를 변형시켜 보여줌으로써 프로세스 데이터베이스를 보다 구체적으로 관리할 수 있도록 도와준다.

또한, 본 시스템은 CMMI와 SPICE중 어느 한 모델의 심사만 받은 조직이 다른 모델의 심사를 받고자 할 때, 다른 모델의 심사를 받기에 미비 된 점이 무엇인지를 보다 쉽게 판단할 수 있도록 도와준다. 나아가 프로세스데이터베이스를 재 정비할 필요 없이, 단순히 그 구조만 변형시킴으로써, 심사 시에 소요되는 시간과 비용을 절감할 수 있다.

3.2 CMMI와 SPICE의 매핑

<표 2>는 CMMI와 SPICE의 상호 보완을 위해서 CMMI의 특정실행(Specific Practice)과 SPICE의 기본실행(Base Practice)을 매핑시킨 결과의 일부이다.

<표 2> CMMI와 SPICE간의 매핑의 예

CMMI	Specific Goal	ISO/IEC 15504(SPICE)	Specific Practice	Base Practice
요구사항관리	SG 1 요구사항 관리	ENG.1 요구사항선출	SP 1.1 요구사항의 이해 획득	BP1 : 고객 요구사항 및 요청 확득
			SP 1.2 요구사항의 커미트먼트 획득	BP3 : 요구사항에 대한 합의
				BP4 : 고객 요구사항에 대한 베이스라인 수립
			SP 1.3 요구사항 변경 관리	BP2 : 고객의 기대사항을 이해
				BP5 : 고객 요구사항의 변화를 관리
			SP 1.4 요구사항의 양방향 추적가능성 유지	BP6 : 고객 질문에 응답할 수 있는 메커니즘 수립
			SP 1.5 프로젝트 작업과 요구사항 간의 불일치한 사항 식별	BP2 : 고객의 기대사항을 이해
				BP4 : 고객 요구사항에 대한 베이스라인 수립

<표 2>는 요구사항 관리 도메인의 CMMI와 SPICE간의 매핑된 결과를 보여주는 예이다. 이것을 정리하면 다음과 같다.[2]

SP 1.1과 BP1은 고객의 요구사항을 요청하여 획득하는 단계이다. SP 1.2와 BP3, BP4와의 매핑은 고객과 조직간의 요구사항에 대한 합의와, 그에 따른 기준선이 확립되는 단계이다. SP 1.3과 BP2, BP5와의 매핑은 고객 요구사항의 변경을 수렴하여 기준선을 재확립하는 단계이다. SP 1.4와 BP6과의 매핑은 요구사항과 프로젝트 계획, 산출물간의 양방향 추적이 가능하도록 유지하는 것과, 요구사항과 요구사항으로부터 도출된 요구사항간의 양방향 추적 가능하도록 유지하는 단계이다.

SP 1.5와 BP2, BP4와의 매핑은 프로젝트 계획과 산출물, 그리고 요구사항 사이의 불일치한 사항을 식별하고 불일치한 원인을 분석하여, 시정 조치하는 단계이다.

CMMI와 SPICE간 매핑은, 요구사항관리 도메인의 매핑에서 보는 것처럼 언제나 일대일로 매핑되거나, 일대이로 매핑되는 규칙적인 매핑이 될 수 없다. 더구나 언제나 한 CMMI PA의 특정실행(SpecificPractice)들이 한 SPICE PA의 기본실행(BasePractice)들에만 매핑되지도 않는다. 어떤 CMMI와 SPICE의 연관은 일대다 매핑이 될 수도 있고, 다대일 매핑이 될 수도 있다. 또한 한CMMI PA의 특정실행이 여러 SPICE의 PA의 기본실행과 매핑될 수도 있다. 아래 (알고리즘 1)은 CMMI와 SPICE간의 매핑 방법을 보여 준다.

Let
  SP_i = i'th Specific Practice of CMMI
  BP_i = i'th Base Practice of SPICE
  SP = { SP_i; i = 1..n, *where* n = number of Specific Practice of CMMI}
  BP = { BP_i; i = 1..m *where* m = number of Base Practice of SPICE}
Find
  CP = SP ∩ BP
  SBP = SP - BP
  BSP = BP - SP
Make a relation between SP_i and BP_i for each element of CP
Find
  BP^k where $BP^k \subset BP$ and $BP^k \supseteq \{SP_k\}$
  SP^k where $SP^k \subset SP$ and $SP^k \supseteq \{BP_k\}$
Make 1-to-n relation from SP_k to each element of BP^k
Make 1-to-n relation from BP_k to each element of SP^k

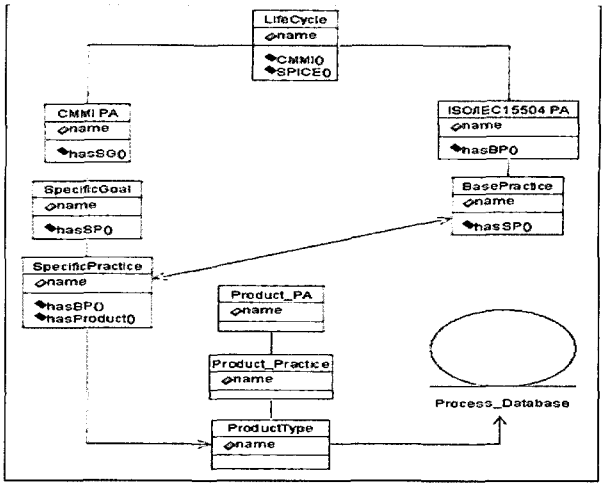
(알고리즘 1) CMMI와 SPICE와의 매핑

즉 CMMI와 SPICE의 매핑은 불규칙적이다. 이 사실은 관계 형 데이터베이스의 테이블만을 사용하여 매핑시키는 것은 부적절하다는 것을 말해준다. 따라서, 불규칙적인 CMMI와 SPICE간 매핑에는, 인간의 사고체계와 흡사한 구조를 가지는 온톨로지의 매핑을 사용하는 것이 적절하다.

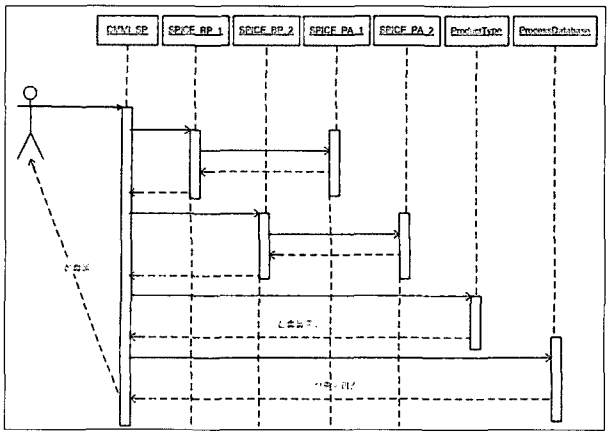
또한 프로세스는 테일러링 과정을 거치지 않고 사용하기 어렵다는 점을 고려하여, 암묵적인 추론은 불가능하지만, 다른 언어에 비해 손쉬운 확장이 가능한 RDF를 이용하였다. 그러므로 프로세스의 테일러링 과정을 거칠 때, 소스코드의 수정 없이 RDF문서를 수정하는 것만으로도 프로세스 테일러링 시의 수정이 가능하다.

3.3 온톨로지 디자인

CMMI와 SPICE를 매핑시킨 것을 토대로 온톨로지를 설계하였다. (그림 4)는 설계된 온톨로지의 기본 구조를 보여준다.



(그림 4) 온톨로지 설계



(그림 5) 시퀀스 다이어그램

(그림 4)의 클래스 다이어그램에서 속성은 OWL문서의 DataProperty와 대응되고, 메소드는 OWL문서의 ObjectProperty가 대응된다. 먼저 온톨로지의 줄기가 될 수 있는 LifeCycle 클래스에서는 CMMI속성과 SPICE속성을 통하여 CMMI PA 클래스와 SPICE PA클래스에 매핑된다.

CMMI PA클래스에서는 hasSG속성을 통하여 SpecificGoal 클래스에 매핑되고, SpecificGoal클래스는 다시 hasSP속성을 통하여 SpecificPractice클래스로 매핑된다. 한편 SPICE PA클래스에서는 hasBP속성을 통하여 BasePractice클래스로 매핑되며, SpecificPractice클래스와 BasePractice클래스는 서로 hasBP와 hasSP 속성을 통해 매핑된다. 또한 CMMI의 SpecificPractice클래스에서는 hasProduct속성을 통해 ProductType클래스에 매핑된다. 마지막으로 ProductType클래스는 직접적으로 프로세스 데이터베이스로 매핑된다. 그러므로 현재 CMMI의 구조로 되어있는 데이터 구조를 SPICE의 구조로 바꾸어 보여주기 위해서는 SPICE의 BP(BasePractice)에 매핑되는 SP(SpecificPractice)를 찾고 각각의 SP에 매핑되는 프로세스데이터베이스의 산출물을 가져오면 된다.

(그림 5)는 사용자가 CMMI의 구조로 보여지고 있는 프로세스데이터베이스를 SPICE의 구조로 프로세스 데이터베이스 형태를 바꾸기 위한 시퀀스다이어그램이다.

4. 실험 및 고찰

4.1 구현 도구

4.1.1 Protégé 3.1

Protégé 3.1은 미국 Stanford대학교의 의료정보 학 그룹(Stanford Medical Information, SMI)에서 개발한 온톨로지 저작 도구 이다. Protégé 3.1는 클래스와 속성의 계층구조, 속성과 인스턴스, 그리고 클래스의 제약사항과, 네임스페이스 등을 쉽게 작성할 수 있게 도와주고, 자동적으로 OWL문서나 RDF문서로 변환해 주어서, 손쉬운 온톨로지 문서 작성에 도움을 준다. (그림 6)은 Protégé3.1을 사용하여 온톨로지 문서를 작성하는 화면이다.

(그림 6)은 각각의 클래스가 매핑되는 것을 보여준다. 위의 클래스 다이어그램에서 설명한 것처럼 LifeCycle클래스의 인스턴스는 CMMI_PA의 인스턴스와 SPICE_PA클래스의 인스턴스에 매핑되고, CMMI_PA클래스의 인스턴스는 SpesificGoal클래스의 인스턴스와 매핑되며, SpesificGoal클래스의 인스턴스는 SpecificPractice클래스의 인스턴스와 매핑되는 것을 보여준다. 한편, SPICE_PA클래스의 인스턴스는 BasePractice클래스의 인스턴스와 서로 매핑되고, BasePractice클래스의 인스턴스는 SpecificPractice클래스의 인스턴스와 서로 매핑되어 거미줄처럼 얽힌 관계를 보여준다.

4.2 구현

구현은 Protégé 3.1을 이용하여 온톨로지 문서를 작성한 후, JENA2를 이용하여 작성된 온톨로지 문서를 파싱했다. 그리고 RDQL을 이용하여 온톨로지 문서에 질의 하였다. 또한 JENA2를 사용하기 위해서 JSP로 구현하였다.

4.2.1 온톨로지 문서 작성

(그림 7)은 Protégé 3.1을 이용하여 작성된 온톨로지 문서의 일부를 보여준다.

(그림 7)에서 나타난 OWL문서를 간단하게 설명하면 다음과 같다.

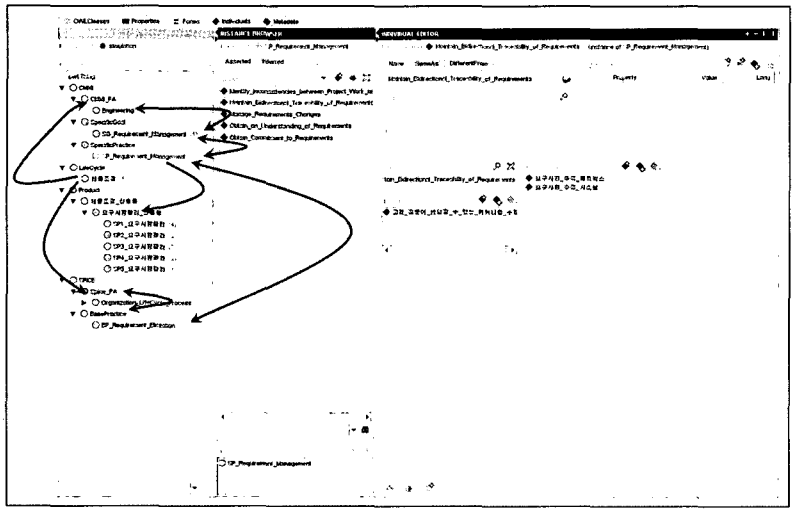
```
<hasSP>
  <SP rdf:ID="SP_1.1-1_요구사항의_이해_획득">
   <name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >SP_1.1-1_요구사항의_이해_획득</name>
   <hasBP>
    <BP rdf:ID="BP1.고객_요구사항_및_요청_획득">
     <name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >BP1.고객_요구사항_및_요청_획득</name>
     <hasSP rdf:resource="#SP_1.1-1_요구사항의_이해_획득"/>
    </BP>
   </hasBP>
   <hasProduct rdf:resource="#기준_대비_분석결과"/>
   <hasProduct>
    <PDB_요구관리 rdf:ID="요구사항_제공자들을_적절하게_구분하기_위한_기준_리스트">
     <name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >요구사항_제공자들을_적절하게_구분하기_위한_기준_리스트</name>
    </PDB_요구관리>
   </hasProduct>
   <hasProduct>
    <PDB_요구관리 rdf:ID="요구사항의_평가와_수락_기준">
     <name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >요구사항의_평가와_수락_기준</name>
    </PDB_요구관리>
   </hasProduct>
   <hasProduct>
    <PDB_요구관리 rdf:ID="합의된_요구사항_목록">
     <name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">합의된_요구사항_목록</name>
    </PDB_요구관리>
   </hasProduct>
  </SP>
</hasSP>
```

(그림 7) OWL 문서

CMMI의 특정실행인 'SP 1.1-1 요구사항의 이해 획득' 항목은 hasBP속성에 의해서 SPICE의 기본실행인 'BP1. 고객 요구사항 및 요청 획득'항목과 매핑되어 있다. 이 특정실행의 산출물의 종류로는 '기준대비 분석결과', '요구사항 제공자를 적절하게 구분하기 위한 기준 리스트', '요구사항의 평가와 수락기준', '합의된 요구사항 목록'이 있는 것을 문서에서 확인할 수 있다. 이러한 매핑의 결과는 CMMI와 SPICE 모델 구조구조의 차이점을 뽑아내어 두 모델과 독립적인 잠재 모델을 형성하기도 한다.

4.2.2 RDQL 작성

(그림 8)는 작성된 온톨로지 문서에 질의하기 위한 RDQL 이다.



(그림 6) Protégé 3.1

```
SELECT ?y WHERE (?x, <http://dankook.ac.kr/zuna#name>, '요구사항_관리')
 (?x, <http://dankook.ac.kr/zuna#CMMI>, ?y)
```

(그림 8) RDQL

위의 (그림 8)는 생명주기를 선택하고, 선택된 생명주기에 CMMI속성으로 매핑된 자원을 찾는 RDQL구문이다. 생명주기 도메인에서 CMMI속성으로 매핑된 자원은 CMMI 도메인의 '요구사항_관리'이므로 위 쿼리의 결과는 '요구사항_관리'가 된다.

4.2.3 시스템 통합

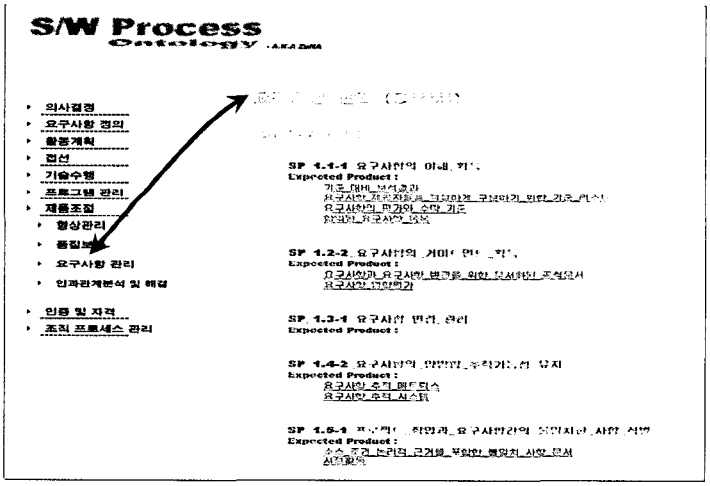
(그림 9)은 구현한 웹 페이지중 요구사항관리에 관한 페이지를 보여주고 있다.

웹 페이지의 왼쪽 프레임에는 소프트웨어 개발 프로세스의 생명주기가 나타나 있다. 사용자가 이 생명주기를 선택하게 되면, 가운데 프레임에 선택한 생명주기에 해당하는 CMMI의 특정목표와 특정실행들이 나타나고, 특정실행의 밑단에는 프로세스데이터베이스에 들어있는 산출물의 종류가 나타난다.

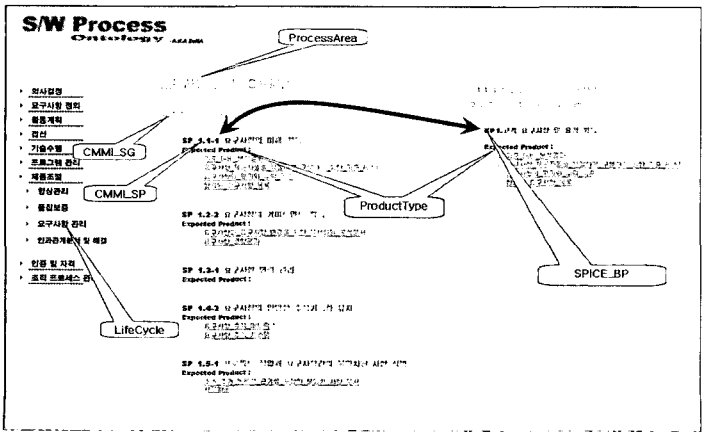
다시 CMMI의 특정실행을 선택하게 되면, (그림 10)의 오른쪽 프레임에 CMMI의 특정실행에 매핑되는 SPICE의 기본실행이 나타난다. 그 밑단에는 프로세스 데이터 베이스에 들어있는 산출물의 종류가 나타난다. 결과적으로 사용자는 특정 생명주기를 선택하고, CMMI의 특정실행을 선택함으로써 프로세스데이터베이스의 구조를 CMMI의 구조에서 SPICE의 구조로 바꾸어 검색할 수 있게 된다. (그림 10)을 보면 CMMI의 특정실행 (SP 1.1-1 요구사항의 이해 획득)과 SPICE의 기본실행 (BP1. 고객 요구사항 및 요청 획득)의 산출물의 종류가 같음을 볼 수 있다. 이 사실을 통해 같은 데이터를 CMMI 구조의 프로세스데이터베이스의 구조가 SPICE의 구조로 바뀐 것을 확인할 수 있다.

마지막으로, 사용자가 산출물의 종류를 선택하면, 프로세스데이터베이스와 직접 매핑되고, 사용자는 프로세스데이터베이스에 저장된 산출물을 볼 수 있게 된다. (그림 11)는 프로세스데이터베이스의 산출물 리스트를 보여준다.

(그림 11)는 사용자가 산출물 종류 중 '합의된 요구사항 목록'을 선택했을 때의 결과 화면이다. 여기에서는 실제 프로세스데이터베이스에 저장되어있는 산출물들의 리스트를 보여준다.



(그림 9) 생명주기와 CMMI간의 매핑



(그림 10) 결과화면

합의된_요구사항_목록

Project	Documents	Contents
단국대학교 컴퓨터 과학과	c:/요구사항명세서.pdf	합의된 요구사항에 대한 기준선입니다.
단국대학교 시스템구축	c:/RequirementMetic.pdf	요구사항을 측정, 분석하여 수치화 해놓은 메트릭스

(그림 11) 산출물 리스트

5. 결 론

본 연구에서는 시맨틱웹의 온톨로지를 이용한 CMMI와 SPICE간의 매핑을 통해 프로세스데이터베이스의 구조를 각각의 구조로 바꾸어 출력할 수 있는 시스템을 구현하였다.

과거에 수행한 프로젝트에서 산출된 데이터의 측정과 분석의 성숙도는 조직 전체 프로세스의 성숙도에 큰 영향을 미친다. 이 시스템은 과거에 수행한 데이터의 집합이라 할 수 있는 프로세스데이터베이스를 CMMI와 SPICE간의 상호 보완적 구조를 활용하여 더욱 구체적으로 측정, 분석하는데 도움을 주고, 나아가 조직이 성숙하는데 도움을 주기 위해 설계되었다. 특히, CMMI와 SPICE모델 중 한 모델의 심사를 통과한 상태에서 다른 모델의 심사를 준비하고자 할 때, 미비 된 점을 보다 간편하게 알아낼 수 있으며, 프로세스데이터베이스의 재정비시간을 감축시켜서 심사 준비 시의 시간과 비용을 절감할 수도 있을 것이다. 학술적으로는 두 모델을 매핑하면서 두 모델의 차이점으로 인해 발생한 두 모델의 내제적 표현 방법을 OMG의 Software Process Engineering Meta-model version 1.1에 비교하는 연구가 가능할 것이다.

향후에는 온톨로지의 추론을 이용한 더욱 발전된 형태의 프로세스데이터베이스로서, 조직이 현재 수행 하고 있는 활동과, 그 활동을 수행하는 중에 조직이 처해진 상황이 주어졌을 때, 각각의 활동과 상황에 도움을 줄 수 있는 문서를 지능적으로 찾아주는 시스템을 구축해 보고자 한다.

참 고 문 헌

[1] Mary Beth chrissis, CMMI Distilled, Addison Wesley Professional, 2005.

[2] Mark C. Paulk, Analyzing Conceptual Relationship bet-ween ISO/IEC 15504 and Capability Maturity Model for Software, International conference on Software Quality, 1999.

[3] Pankaj Jalote, CMM In Practice, Addison Wesley, 2002.

[4] SEI Carnegie-Mellon University, CMM CapabilityMaturity Model, Addison Wesley, 2003.

[5] SEI Carnegie-Mellon University, CMM implementation Guide, Addison Wesley, 1998.

[6] Shelley Powers, Pratical RDF, OReilly, 2003.

[7] Watts S. Humphrey, "*Discipline for SoftwareEngineering*", Addison Wesley, 1995.

[8] 고영만, 서태설, 온톨로지 기반 메타데이터 명명규칙에 관한 연구, 정보통신기술협회, 2005.

[9] 공현장, 황명권, 김원필, 김판구, 온톨로지 언어 사용 지침 표준안 제정 및 표준안 기반 온톨로지 구축 도구 설계, 한국컴퓨터종합학술대회 논문집 Vol.32 No1, 2005.

[10] 김경환, 김홍재, 박용범, CMMI와 PMBOK의 비교 분석을 통한 정량적 프로젝트 관리, 정보처리학회 논문지 제12-D권 제4호, 2005.

[11] 김현희, 안태경, "온토로지를 이용한 인터넷웹 검색에 관한 실험적 연구", 정보처리학회지 제20권, 2003.

[12] 김홍기, 시맨틱 웹 저서, 원고, 2004.

[13] 이재호, 시맨틱 웹의 온톨로지 언어, 정보과학회지 21권 제3호, 2003.

[14] 중소기업의 SW개발 역량강화를 위한 국제표준, 산업자원부 기술표준원, 2005.

[15] Carnegie-Mellon SEI site, "http://www.sei.cmu.edu /cmmi"

[16] JENA2 tutorial site, "http://jena.sourceforge,net"

[16] RDF Tutorial site, "http://www.w3.org/RDF/"

[17] RDQL tutorial site, "http://www.w3.org/Submission/RDQL/"

[18] T. Gruber Blog, "http://www.ksl.stanford.edu/kst/what-is-an-ontology.html"

[19] World Wide Web Consortium site, http://www.w3.org.

박 용 범

e-mail : ybpark@dankook.ac.kr

1985년 서강대학교 전자계산학과(학사)

1987년 N.Y. Polytechnic Univ. 전자계산학과(석사)

1991년 N.Y. Polytechnic Univ. 전자계산학과(박사)

1993년~현재 단국대학교 전자컴퓨터학부 컴퓨터과학전공 교수

관심분야 : Information Architecture, 패턴인식, 분산 에이전트

이 준 하

e-mail : diggidic@gmail.com

1999년 현대고등학교 졸업

2007년 ISO IEC 15504 예비 심사원 자격 취득

2000년 ~ 현재 단국대학교 컴퓨터과학과 재학

관심분야 : 소프트웨어 프로세스 개선, 소프트웨어 측정, 요구사항 관리