UML 기반 객체 지향 개발을 위해 ISO 12207을 조정한 객체지향 프로세스

이 상 준† · 김 병 기††

요 약

소프트웨어 품질은 프로세스 품질과 제품의 품질로 구분할 수 있는데, 소프트웨어 품질 관리의 경험에 의하면 제품 품질의 수준은 프로세스와 조직의 좋고 나쁨에 강하게 의존하는 것으로 알려졌고, 그 결과 소프트웨어 프로세스의 개선이 중요한 과제가 되어 왔다. 국제 표준화 기구에서는 이와 같은 흐름에 발맞춰 소프트웨어 생명주기 프로세스의 표준으로 ISO 12207을 발표하였다. UML을 기반으로 하는 객체지향 개발 프로세스를 위하여 ISO 12207을 조직이나 프로젝트의 성격에 따라 추가하거나 삭제하거나 조정하여, 활동 및 태스크를 구체적으로 정의하는 연구가 필요하다. 본 논문에서는 ISO 12207 소프트웨어 생명주기 프로세스를 따르며 UML을 기반으로 하는 객체지향 개발 프로세스를 제안한다. 이 프로세스는 신뢰성 중심의 품질을 향상시킬 수 있도록 개발 단계, 활동, 산출물을 포함하는 7단계 19개의 세부 작업으로 구성된다. 소프트웨어 품질 향상을 위한 객체지향 프로세스의 유용성은 프로세스 특성의 비교 분석, SPICE 프로세스 품질 평가, SPICE 위험 분석의 세가지 방법으로 증명한다.

ISO 12207 Tailored Object-Oriented Process for UML Based Object-Oriented Development

Sang-Jun Lee† · Byung-Ki Kim††

ABSTRACT

Software quality is classified by quality of process and product. In experience of Quality Management, it is known that quality level of product as it depends on goodness and badness of process and organization. As a result, improvement of software process has been important subject. According as this trends, ISO 12207 is publicated as standard of software life cycle process by ISO. For UML based object oriented development process, it is necessary that we should research detailed definition of activity and task of ISO 12207 process which is added, deleted or tailored in according to organization and project characteristics. In this thesis, by according with ISO 12207 software life cycle process, UML based object oriented development process is proposed. This process is composed of 7 steps and 19 activities including development phase, activity and product to improve quality of reliability. Usefulness of object oriented process for improvement of software quality is proved at three ways, which are comparative analysis of process characteristics, SPICE process evaluation and SPICE risk analysis.

† 정 회 원 : 서남대학교 컴퓨터정보통신학부 교수
†† 종신회원 : 전남대학교 컴퓨터정보학부 교수
논문접수 : 1999년 1월 21일, 심사완료 : 1999년 3월 13일

1. 서 론

정보화 사회의 발전에 따라 소프트웨어 공학에서는 소프트웨어의 품질과 생산성의 향상을 가장 중요한 과제로 여기고 있다[13]. 소프트웨어 품질은 프로세스 품질과 제품의 품질로 크게 두 가지로 구분되므로, 소프트웨어의 품질향상을 위해서는 프로세스 품질의 향상 또는 소프트웨어 제품의 품질향상에 대하여 품질의 목표를 설정하고, 목표 설정을 위한 기준을 정의하여야 한다[3].

품질목표로는 정확성, 신뢰성, 효율성, 무결성, 사용용이성, 유지보수용이성, 유연성, 이식성, 재사용성, 상호운용성 등이 있다[2]. 이중에서 신뢰성은 소프트웨어를 믿고 사용할 수 있는가에 대한 가장 결정적인 목표라고 할 수 있다[11].

소프트웨어 프로세스는 소프트웨어와 관련된 프로젝트 계획, 설계문서, 코드, 시험사례, 사용자 매뉴얼 등을 개발하고 유지보수하기 위하여 사용하는 활동, 방법, 기법 등이 일련의 순서를 가지는 집합이다.

소프트웨어 개발 및 관리를 위한 표준, 절차, 방법, 도구 및 환경은 많이 개발되고 있고 증가 일로에 있다. 이러한 증가에 따라 특히 제품과 서비스가 함께 합쳐지는 경우의 소프트웨어 개발 및 관리에 많은 어려움 있어서, 소프트웨어 엔지니어가 소프트웨어를 개발하고 관리함에 있어 "의사소통의 공통공유 수단"으로 사용할 수 있는 프레임웍을 갖출 필요성이 커지게 되었다. ISO 12207 소프트웨어 생명주기 프로세스 표준은 ISO/IEC JTC1 SC7에 의해 통일된 공통 공유의 프레임웍을 제공하기 위한 목적으로 개발되었다[6].

객체지향 모델링의 표준 방안인 UML[1, 5]은 분석, 설계를 위한 표기법으로서만 역할을 담당하고 있으며, UML 개발 과정은 예비반복이라는 개괄적인 모델을 구축하는 단계로 시작하여 상세화 단계에서 점진·반복적 개발 과정을 거치며, 또 다른 관점에서는 시작, 상세화, 구축, 진화라는 4단계의 세분화 과정을 거친다.

ISO 12207 소프트웨어 개발 생명주기를 따르며, UML을 기반으로 하는 객체지향 개발 프로세스가 신뢰성 중심의 품질이 향상되도록 정의하는 연구가 필요하다.

본 논문에서는 UML을 이용한 객체지향 개발 프로세스가 ISO 12207 소프트웨어 개발 생명주기를 따르며, 신뢰성 중심의 품질을 향상시킬 수 있도록 공인활동이 강조된 개발 프로세스의 개발 단계, 활동, 산출물을 제안한다. 프로세스는 7단계 19개의 세부작업으로 구분되며 산출물은 UML 표기법을 따른다.

2장 관련 연구에서는 ISO 12207 소프트웨어 생명주기 프로세스 모델과 UML 기반 프로세스 모델을 살펴본다. 3장에서는 공인 과정과 마코프 연쇄를 이용하여 개발 단계, 작업, 산출물 중심으로 객체지향 프로세스를 제안한다. 4장에서는 제안된 프로세스의 특성 비교, SPICE 레벨 계산, SPICE 위험 분석으로 제안된 프로세스의 평가를 수행한다. 5장에서는 결론을 맺으며 앞으로 연구 방향에 대해서 논의한다.

2. 관련 연구

2.1 ISO 12207 소프트웨어 생명주기 프로세스

대부분의 조직은 내부적으로 소프트웨어를 개발하고 관리하기 위한 표준, 절차, 방법, 도구 및 환경들을 개발하여 사용하여 왔으나 무분별하게 일관성 없이 증식되어 개발자-사용자간, 개발자간의 의사소통에 오히려 많은 어려움을 발생시켰다. 물론 모든 조직, 모든 소프트웨어 유형에 똑같이 적용될 수 있는 개발 및 관리 방법이 있을 수는 없으나 소프트웨어 실무자가 개발 및 관리에 있어서 동일한 언어로 말할 수 있는 공통적인 기본틀에 대한 요구는 계속되어 왔다.

ISO 12207 표준은 이러한 문제를 해결하기 위하여 소프트웨어 산업계에서 참고할 수 있는 잘 정의된 용어를 사용하여 소프트웨어 제품이나 서비스의 획득, 소프트웨어 제품의 공급, 개발, 운영 및 유지보수에 적용될 수 있는 소프트웨어 생명주기 프로세스들에 대한 공통의 기본틀을 제공하고 있다. 생명주기 기본틀은 프로세스, 활동, 작업 수준으로 계층화되어 있고 기본(primary), 지원(supporting), 조직(organizational) 프로세스로 구성되어 있다[6].

ISO 12207은 소프트웨어 프로젝트에 따라 조정될 수 있는 프로세스, 활동 및 태스크의 집합으로 구성되어 프로젝트의 필요성에 따라 선택적으로 적용할 수 있게 하였다.

ISO 12207의 개발 프로세스는 요구사항 분석, 설계, 구현, 통합, 시험, 설치 및 인수등 소프트웨어 제품 개발과 관련된 활동을 포함한다. 개발 프로세스에서 수행할 각 활동과 그에 대한 세부작업 내용은 프로세스 구현, 시스템 요구사항 분석, 시스템 구조 설계, 소프트웨어 요구사항 분석, 소프트웨어 구조 설계, 소프트웨

어 상세 설계, 소프트웨어 구현 및 시험, 소프트웨어 통합, 소프트웨어 자격 시험, 시스템 통합, 시스템 자격 시험, 소프트웨어 설치, 소프트웨어 인수 지원이 있다.

본 논문에서는 프로세스의 주체가 개발자가 되어 개발자가 수행할 활동 및 태스크가 제시된 ISO 12207의 개발 프로세스를 UML을 기반으로 하는 객체지향 개발 프로세스에 도입하도록 연구하였다.

2.2 UML 기반 프로세스 모형

UML(Unified Modeling Language)은 Rumbaugh의 OMT방법론, Booch의 Booch방법론[4], Jacobson의 OOSE방법론을 통합하여 만든 객체지향 통합 모델링 기법이다. 기존의 방법론이 프로세스 흐름 중심으로 모델링 했던 것에 반해 UML은 객체 뿐만 아니라 객체간의 상호관계를 표현함에 있어 기능적인 면과 행위적인 면, 상태적인 면을 쉽게 표현할 수 있게 하는 모델링 언어이다. 기능적인 면을 표현하기 위해 사용사례 다이어그램과 활동 다이어그램을 제공하고, 정적인 상태를 표현하기 위해 클래스 다이어그램(class diagram), 패키지 다이어그램, 배치 다이어그램(deployment diagram)을 제공한다. 그리고 동적인 행위를 표현하기 위해서는 순차 다이어그램, 협력 다이어그램, 상태 다이어그램을 제공한다.

UML은 표준 프로세스를 정의하지 않고 있으며, 모델링 기법을 이용하여 어떤 프로세스로 소프트웨어를 개발하는 것이 품질을 향상시킬 수 있는가 하는 연구는 아직 부족하다[10].

개발 방법론은 다이어그램 작성 도구와 개발 프로세스로 이루어져 있다. 그림만 그린다고 설계가 되는 것이 아니고, 방법과 순서에 맞추어 도구를 사용해야 한다. 이 순서는 매우 유기적으로 연결되어 있어서 앞 단계가 있어야 뒤의 단계를 계속할 수 있다. 그렇지 않은 경우에는 산출물의 품질이 매우 떨어지게 된다.

UML의 개발 과정에서는 개괄 모델의 구축과 점진적 개발이라는 분류와 함께 또 다른 관점에서 전체 공정을 "시작→상세화→구축→진화" 라는 4단계로 분류하고 있다[1]. UML의 개발 과정은 소프트웨어가 프로젝트의 마지막 단계에서 하나의 최종 산출물로 개발되는 것이 아니라 여러 개의 부분 산출물로 개발되는 점진·반복적 개발 과정이다. 구현 단계는 많은 반복으로 이루어지며, 각각의 반복에서는 프로젝트의 부분 요구 사항들을 만족시키기 위하여 시험되고 생산성과 품질 중심의 통합된 소프트웨어를 구축한다. 또한 각각의 반복은 분석, 설계, 구현 및 시험 등의 일반적인 생명 주기의 모든 단계를 포함한다. 개발 과정에서 프로젝트의 목표와 범위를 설정하는 시작(inception) 또는 예비반복(preliminary iteration)으로 표현된 부분은 개괄적인 모델을 구축하는 단계이다. 상세화(elaboration) 단계에서는 시스템 구조의 베이스라인을 설정하고 높은 수준의 분석과 설계를 실시하기 위하여 보다 구체적인 요구 사항을 수집하며, 구현을 위한 계획을 수립한다. 나머지 단계들은 점진·반복적 개발에 해당한다. 구축(construction) 단계에서는 일련의 반복을 통하여 산출물을 구현한다. 진화(transition) 단계에서는 베터 시험, 성능 조율 및 사용자 교육등을 실시하고, 사용자 환경에서 이용할 수 있도록 필요한 작업을 수행한다.

3. 제안한 객체지향 프로세스

제안한 프로세스는 점진 반복적인 구조를 갖고 있으며 <표 1>과 같이 제품 출고 계획을 수립하여 조사, 계획, 구축, 배치를 반복한다. 제품 구축 단계는 분석, 설계, 구현, 테스팅이 포함된 개발 주기를 반복적으로 수행하여 구축된다. 이전 단계에서의 테스팅 결과는 다음 단계의 분석, 설계 단계에 적용된다.

〈표 1〉 제안된 프로세스의 개요

제품 출고	출고 세부사항	구축 주기	구축 세부사항
제품 출고 1	조사		
	계획		
	구축	ISO 12207를 따르는 개발주기 #1	분석, 설계, 구현, 테스팅
		ISO 12207를 따르는 개발주기 #2	분석, 설계, 구현, 테스팅
		. . .	. . .
		ISO 12207를 따르는 개발주기 #n	분석, 설계, 구현, 테스팅
	배치		
제품 출고 2	조사		
	계획		
	구축	ISO 12207를 따르는 개발주기 #1	분석, 설계, 구현, 테스팅
		ISO 12207를 따르는 개발주기 #2	분석, 설계, 구현, 테스팅
		. . .	. . .
		ISO 12207를 따르는 개발주기 #n	분석, 설계, 구현, 테스팅
	배치		
. . .	. . .	. . .	. . .

〈표 1〉 계속

제품 출고	출고 세부사항	구축 주기	구축 세부사항
제품 출고 n	조사		
	계획		
	구축	ISO 12207를 따르는 개발주기 #1	분석, 설계, 구현, 테스팅
		ISO 12207를 따르는 개발주기 #2	분석, 설계, 구현, 테스팅
		. . .	. . .
		ISO 12207를 따르는 개발주기 #n	분석, 설계, 구현, 테스팅
	배치		

제안된 프로세스는 <표 2>와 같이 7단계의 19 세부 작업으로 구성된 개발 단계가 점진·반복적으로 수행되도록 하였다. 제시된 각 단계별 세부 작업에 대하여 작업명, 목적, 태스크, 입력, 출력을 정의하고, ISO 12207 소프트웨어 생명주기 프로세스의 어떤 부분과 관련이 있는지 조사하였고, 평가하기 위한 프레임웍으로 SPICE를 고려하여 평가 관련 부분을 조사하였다. 각 단계들은 간략히 표 형식으로 제시하고, 프로세스의 특성이 되는 품질 확보를 위한 공인 단계와 관련된 활동들을 집중적으로 설명하고, 일반적인 활동의 자세한 설명은 생략한다.

〈표 2〉 단계별 세부 작업 내용

단계명	세부 작업
100 조사	110 시스템 조사
200 계획	210 요구 명세서 작성 220 개발 계획서 작성
300 분석	310 사용 사례도 및 사용법 연쇄 작성 320 시스템 분석 및 개념적 모델링 330 사용자 인터페이스 분석 340 분석서 작성
400 설계	410 시스템 설계 및 상세 모델링 420 사용법 프로파일 유도 430 사용자 인터페이스 및 DB 설계 440 설계서 작성
500 구현	510 클래스 구현 520 시험 사례 생성 및 단위 시험 530 사용자 인터페이스 구현
600 테스팅	610 통합 시험 620 신뢰도 공인 630 테스팅 결과 보고서 작성
700 배치	710 인수 시험 720 배치 및 운영

3.1 조사 단계

조사단계에서는 시스템 전반에 대한 조사를 수행하여 목표 시스템과 관련된 기초 자료를 수집한다.

〈표 3〉 시스템 조사

항 목	내 용
작 업 명	110 시스템 조사
목 적	목표시스템과 관련된 기초 자료 조사
태 스 크	비 전산화 업무 분석 기존 전산화 업무 분석 목표 시스템 일반 특성 분석
입 력	각종 원천 문서 기존 소프트웨어
출 력	비 전산화 업무 현황 문서 기존 전산화 업무 현황 문서 목표 시스템 일반 특성 문서
ISO 12207 관련 항목	프로세스 구현
SPICE 관련 항목	CUS.1.1 , SUP.1 , MAN.1.1

3.2 계획 단계

계획단계에서는 요구 명세서 작성, 개발 계획서 작성을 통하여 요구 명세서와 개발 계획서를 작성한다.

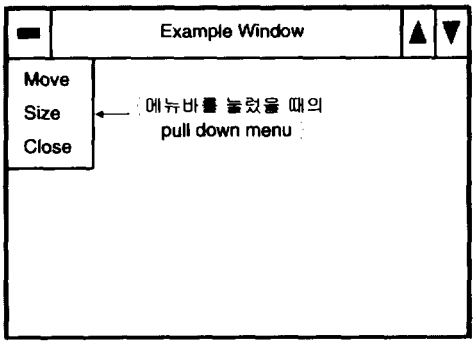
〈표 4〉 요구명세서 작성

항 목	내 용
작 업 명	210 요구 명세서 작성
목 적	목표시스템에 대한 사용자의 요구사항 작성
태 스 크	목표 시스템의 전반적인 문제 영역 정의, 목표 설정
입 력	비 전산화 업무 현황 문서 기존 전산화 업무 현황 문서 목표 시스템 일반 특성 문서
출 력	요구 명세서
ISO 12207 관련 항목	시스템 요구사항 분석
SPICE 관련 항목	ENG.1.1 , SUP.1 , MAN.1.1

〈표 5〉 개발 계획서 작성

항 목	내 용
작 업 명	220 개발 계획서 작성
목 적	목표시스템 개발 계획 작성
태 스 크	시스템 정의를 타당하게 실현할 수 있는 최적의 개발 전략을 인적, 기술적, 자금 투자 등으로 구체화하는 시키는 작업
입 력	요구 명세서
출 력	개발 계획서
ISO 12207 관련 항목	프로세스 구현(개발 계획서 작성)
SPICE 관련 항목	ENG.1.1 , ENG.2 , SUP.1 , MAN.1.1

3.3 분석 단계



(그림 1) 윈도우 예제

분석 단계에서는 공인 작업을 위한 기초 활동으로 사용 사례도 및 사용법 연쇄를 작성한다[15, 16]. 예를 들어 간단한 윈도우에서 메뉴의 선택에 따른 사용법 모델링을 사용 사례도 및 사용법 연쇄 작성으로 설명하면 다음과 같다. (그림 1)은 목표로 하고 있는 윈도우 예제이다.

〈표 6〉 사용 사례도 및 사용법 연쇄 작성

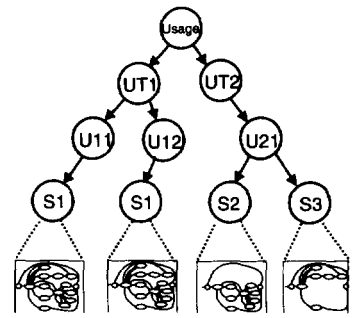
항 목	내 용
작 업 명	310 사용 사례도 및 사용법 연쇄 작성
목 적	사용자 위주의 시스템 분석
태 스 크	행위자 식별 행위자별 사용 사례 분석 시나리오 작성 사용법 모델링
입 력	요구 명세서
출 력	사용 사례도, 사용법 연쇄
ISO 12207 관련 항목	시스템 요구사항 분석, 소프트웨어 요구사항 분석
SPICE 관련 항목	ENG.1.2 , SUP.1 MAN.1.1 , MAN.2

〈표 7〉 시스템 분석 및 개념적 모델링

항 목	내 용
작 업 명	320 시스템 분석 및 개념적 모델링
목 적	시스템 분석 시스템의 정적, 동적, 기능적 면을 모델링
태 스 크	시스템 구조 분석 클래스 식별 클래스 속성/연산 식별 클래스간의 관계 식별
입 력	요구 명세서, 사용 사례도
출 력	클래스도, 상태도, 순차도, 활동도
ISO 12207 관련 항목	시스템 요구사항 분석, 소프트웨어 요구사항 분석
SPICE 관련 항목	ENG.1.2 , SUP.1 MAN.1.1

윈도우는 메뉴 막대에서 Move, Size, Close 라는 세 개의 하위 메뉴를 제공하는데 Move는 윈도우의 위치를 옮길 때, Size는 윈도우 크기를 조절할 때, Close는 윈도우를 종료할 때 이용한다. ▲버튼은 윈도우의 크기를 화면 전체 크기로 바꿀 때, ▼버튼은 윈도우를 아이콘화 할 때 이용한다. 윈도우 사용법은 사용자 타입에 따라 각기 달리 사용할 수 있을 것이며, 개별적인 사용자는 시스템 서비스들을 이용할 수 있다.

(그림 2)는 윈도우 예제를 사용법(Usage), 사용자 타입(UT), 사용자(U), 서비스(S) 수준으로 구분하여 상태 계층 모델로 사용법 모델링을 수행한 결과이다. 그림을 쉽게 이해하기 위해 UT1(교수), UT2(학생), U11(김교수), U12(이교수), U21(홍길동)이라 표시해 보면, 사용자 타입은 교수와 학생으로 구분되고, 사용자는 김교수, 이교수, 홍길동이 있고, 각 사용자에 따라 사용하는 서비스가 같거나 다를 수 있다.



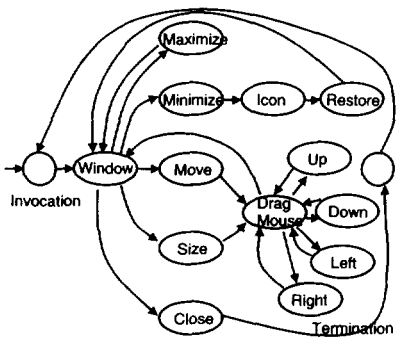
(그림 2) 윈도우 예제에 대한 사용법 모델

〈표 8〉 윈도우의 자극과 응답

자극	응답
Invocation	(그림 3)의 윈도우가 나타남
▲ 선택	윈도우 크기를 화면 전체 영역이 덮어지도록 확장
▼ 선택	윈도우를 제거하고 합당하는 아이콘으로 대체
▬ 선택후 pull down menu에서 Move선택	마우스 입력에 의해 직접적으로 윈도우가 이동
▬ 선택후 pull down menu에서 Size 선택	마우스 입력에 의해 직접적으로 윈도우 크기 조절
▬ 선택후 pull down menu에서 Close선택	화면에서 윈도우를 제거
아이콘을 선택하고 놓았을때	화면으로부터 아이콘을 제거하고 윈도우를 복원

<표 8>은 윈도우에서 사용자의 자극과 자극에 따른 응답 결과를 나타내었다. 자극과 응답간의 관계는 사용자의 움직임에 따른 소프트웨어의 예상되는 행동으로 볼 수 있다.

(그림 2)를 구성하는 각 서비스의 마코프 체인[14]을 중복함이 없이 나타내면 (그림 3)와 같이 사용법 연쇄로서 사용법 모델이 통합될 수 있다.



(그림 3) 예제의 사용법 모델링

〈표 9〉 사용자 인터페이스 분석

항 목	내 용
작 업 명	330 사용자 인터페이스 분석
목 적	시스템을 사용할 사용자의 이용 환경 분석
태 스 크	사용자 인터페이스 요구 사항 분석, 사용자 인터페이스 시나리오 작성, 최적의 메타포 정의
입 력	요구 명세서
출 력	사용자 인터페이스 시나리오, 메타포 정의
ISO 12207 관련 항목	시스템 요구사항 분석, 소프트웨어 요구사항 분석
SPICE 관련 항목	ENG.1.1 , ENG.1.2 , SUP.1 MAN.1.1

〈표 10〉 분석서 작성

항 목	내 용
작 업 명	340 분석서 작성
목 적	분석 결과의 문서화
태 스 크	310, 320, 330 작업을 하나의 문서로 문서화
입 력	요구 명세서, 사용 사례도, 클래스도, 상태도, 순차도, 활동도 사용자 인터페이스 시나리오, 메타포 정의
출 력	분석서
ISO 12207 관련 항목	시스템 요구사항 분석, 소프트웨어 요구사항 분석
SPICE 관련 항목	ENG.1.1 , ENG.1.2 , SUP.1 , MAN.1.1

3.4 설계 단계

설계 단계의 작업중 사용법 프로파일 유도 작업은 품질향상과 관계된다. 사용법 프로파일은 행동 레벨상의 모든 천이와 함께 사용법 계층에서 각 분기에 대한 확률을 할당하여 사용법 모델을 보완한다.

〈표 11〉 시스템 설계 및 상세 모델링

항 목	내 용
작 업 명	410 시스템 설계 및 상세 모델링
목 적	시스템 구조 설계 소프트웨어 상세 설계
태 스 크	시스템 구조 설계 상세 모델링
입 력	분석서
출 력	사용 사례도, 클래스도, 상태도, 순차도, 활동도, 협력도
ISO 12207 관련 항목	소프트웨어 구조 설계, 소프트웨어 상세 설계
SPICE 관련 항목	ENG.1.3 , SUP.1 , MAN.1.1

〈표 12〉 사용법 프로파일 유도

항 목	내 용
작 업 명	420 사용법 프로파일 유도
목 적	실제 사용 사례 정의 테스트 사례 기초 자료 파악
태 스 크	고객 프로파일 찾기 사용자 프로파일 설정 시스템 모드 프로파일 설정 기능적 프로파일 결정
입 력	분석서(사용법 연쇄)
출 력	사용법 프로파일
ISO 12207 관련 항목	없음
SPICE 관련 항목	MAN.1.1 , MAN.2

〈표 13〉 사용법 프로파일 유도 단계

단계	프로파일 유도단계	상태 계층 모델 레벨
1	고객 프로파일 찾기	사용자 타입 레벨
2	사용자 프로파일 설정	사용자 레벨
3	시스템-모드 프로파일 설정	서비스 레벨
4	기능적 프로파일 결정	행동 레벨
5	운용적 프로파일 결정	적용 결과 보관

확률은 이전 시스템으로 부터의 경험과 실제 시스템에 관계된 기대 변화 또는 상품화된 시스템의 기대 사용법을 바탕으로 유도되어 진다.

사용법 프로파일을 유도하기 위한 5단계는 <표 13>과 같다[12]. 상태 계층 모델 중심으로 소프트웨어 공인을 수행하고자 할 때, 테스팅 하기 위한 기초 자료를 프로파일 유도 단계를 이용하여 획득할 수 있다.

〈표 14〉 전이 확률과 이벤트

From-State	To-State	확률	간격	이벤트
Usage	UT1	0.7	0-69	▶
Usage	UT2	0.3	70-99	▶
UT1	U11	0.2	0-19	▶
UT1	U12	0.8	20-99	▶
UT2	U21	1.0	0-99	▶
U11	S1	1.0	0-99	▶
U12	S1	1.0	0-99	▶
U21	S2	0.5	0-49	▶
U21	S3	0.5	50-99	▶
Invocation	Window	1.0	0-99	▶
Window	Maximize	0.08	0-7	▲
Window	Minimize	0.08	9-15	▼
Window	Move	0.17	17-32	■ M
Window	Size	0.17	34-49	■ S
Window	Close	0.5	50-99	■ C
Maximize	Window	1.0	0-99	▶
Minimize	Icon	1.0	0-99	▶
Icon	Restore	1.0	0-99	▶
Restore	Window	1.0	0-99	▶
Move	Drag Mouse	1.0	0-99	◉
Size	Drag Mouse	1.0	0-99	◉
Drag Mouse	Window	0.27	0-26	◉
Drag Mouse	Up	0.07	27-33	◉
Drag Mouse	Down	0.33	34-66	◉
Drag Mouse	Left	0.2	67-86	◉
Drag Mouse	Right	0.13	87-99	◉
Up	Drag Mouse	1.0	0-99	◉
Down	Drag Mouse	1.0	0-99	◉
Left	Drag Mouse	1.0	0-99	◉
Right	Drag Mouse	1.0	0-99	◉
Close	Termination	1.0	0-99	▶
Termination	Invocation	1.0	0-99	▶

▶ : 특정 이벤트 없이 다음상태로 자동전환
▲ : 윈도우 위젯중 전체 화면으로
▼ : 윈도우 위젯중 화면을 아이콘으로
■(M,S,C) : 메뉴막대를 누른후 서브 메뉴 선택
◉ : 마우스를 조작함

상위 4단계까지는 시스템을 점차적으로 더 자세히 나누고, 마지막 5단계에서는 시스템에 의해 수행된 직무를 나타낸다. 시험 노력을 할당하고, 시험을 선택하고, 어떤 시험이 수행되어야 하는지 결정하기 위해 운용적 프로파일을 사용한다. 운용적 프로파일은 시스템을 실제 수행한 후의 자료에 의해서 생성되므로 구현 작업 이후에 이용할 수 있다.

프로파일을 생성하는 과정에서는 확률을 계산하게 된다. 사용법 모델은 마코프 연쇄 형태인 사용법 연쇄이며 사용법 모델에 확률을 결정하는 무정보 방법, 정보 방법, 의도 방법의 세가지 접근 방법이 있다. 윈도우 예제에서 가능한 사용법에 대한 순차를 <표 15>와 같다고 할 때, 이런 정보를 가지고 의도 접근법으로 <표 14>와 같이 전이 확률을 계산할 수 있다.

〈표 15〉 가정된 사용법

순번	가정된 순차
1	<Invocation><Window><Maximize><Window><Close><Termination>
2	<Invocation><Window><Minimize><Icon><Restore><Window><Close><Termination>
3	<Invocation><Window><Move><Drag-Mouse><Down><Drag-Mouse><Right><Drag-Mouse><Down><Drag-Mouse><Window><Close><Termination>
4	<Invocation><Window><Size><Drag-Mouse><Left><Drag-Mouse><Up><Drag-Mouse><Left><Drag-Mouse><Window><Close><Termination>
5	<Invocation><Window><Move><Drag-Mouse><Down><Drag-Mouse><Left><Drag-Mouse><Down><Drag-Mouse><Window><Close><Termination>
6	<Invocation><Window><Size><Drag-Mouse><Down><Drag-Mouse><Right><Drag-Mouse><Window><Close><Termination>

〈표 16〉 사용자 인터페이스 및 DB 설계

항 목	내 용
작 업 명	430 사용자 인터페이스 및 DB 설계
목 적	사용자 인터페이스 설계 DB 설계
태 스 크	화면 윈도우 설계 사용자 인터페이스 프로토타입 설계 DB 스키마 구성
입 력	사용자 인터페이스 시나리오 사용자 인터페이스 메타포 클래스도
출 력	사용자 인터페이스 설계 절차서
ISO 12207 관련 항목	소프트웨어 구조 설계, 소프트웨어 상세 설계
SPICE 관련 항목	ENG.1.3 , SUP.1 , MAN.1.1

〈표 17〉 설계서 작성

항 목	내 용
작 업 명	440 설계서 작성
목 적	설계 결과의 문서화
태 스 크	410, 420, 430 작업을 하나의 문서로 문서화
입 력	사용 사례도, 클래스도, 상태도, 순차도, 활동도, 협력도 사용법 프로파일 사용자 인터페이스 설계 절차서
출 력	설계서
ISO 12207 관련 항목	소프트웨어 구조 설계, 소프트웨어 상세 설계
SPICE 관련 항목	ENG.1.3 , SUP.1 , MAN.1.1

구현단계를 거치면서 <표 20>과 같은 6개의 시험 사례를 통해서 예제의 모든 상태를 적어도 한번은 시험하도록 할 수 있다.

3.5 구현 단계

구현단계에서는 품질 향상을 위해서 적은 비용으로 최적의 시험 사례를 생성하는 작업을 수행한다. 더불어 클래스와 사용자 인터페이스의 구현도 수행한다.

마코프 연쇄는 통계적 확률 과정 모델중 하나로서 모델에 대한 분석을 하기 위한 몇가지 계산식이 있다 [14, 16]. 특히 유용한 분석식으로는 특정 상태에 도달하는데 필요한 중간 상태의 개수를 계산하는 $x_i = \frac{1}{\pi_i}$ 이 있다. 이때 π_i는 가정된 사용법에서 i상태가 몇 번 사용되었는지를 나타내는 누적 분포 확률이다. 또, 상태 i가 나타날 때 까지의 평균 순차 개수를 나타내는 $s_i = \frac{x_1}{x_{TERM}} = \frac{\pi_{TERM}}{\pi_i}$ 식이 있다. 윈도우 예제에서 시험 사례의 평균 길이는 14(11+3)개이며, 시험 사례의 개수는 6개가 필요하다. 이와 같은 시험 사례의 개수 및 평균 길이에 맞춰 0부터 99 사이의 정수 난수를 84(14*6)개 발생시킨다. <표 20>에서 발생된 난수에 따른 시험 사례를 나열하였다.

〈표 18〉 클래스 구현

항 목	내 용
작 업 명	510 클래스 구현
목 적	클래스 구현
태 스 크	상세 설계 내용 구현 클래스 속성/연산 정의 구현
입 력	설계서(클래스도, 상태도, 순차도, 활동도, 협력도)
출 력	코드 작성

〈표 18〉 계속

항 목	내 용
ISO 12207 관련 항목	소프트웨어 구현 및 시험, 소프트웨어 통합
SPICE 관련 항목	ENG.1.4 , SUP.1 , MAN.1.1

〈표 19〉 시험 사례 생성 및 단위 시험

항 목	내 용
작 업 명	520 시험 사례 생성 및 단위 시험
목 적	단위 시험을 위한 시험 사례 생성
태 스 크	시험 사례의 길이, 개수 결정 시험 사례 생성 단위 시험 수행
입 력	사용법 연쇄
출 력	시험 사례, 단위시험 결과서
ISO 12207 관련 항목	소프트웨어 구현 및 시험
SPICE 관련 항목	ENG.1.4 , SUP.3 , MAN.2

〈표 20〉 시험 사례 생성

일련번호	구분	난수/이벤트													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	난	22	19	16	78	30	93	78	23	15	58	57	48	61	36
	이	UT	U11	S1	W	Mo	Dm	Le	Dm	W	C	T	I		
2	난	18	88	09	12	85	38	53	40	02	95	35	26	77	46
3	난	37	61	93	21	95	97	69	04	61	85	21	15	02	87
4	난	98	10	47	22	67	27	03	13	58	47	23	69	35	55
5	난	69	90	83	74	39	15	90	95	39	16	52	56	21	23
6	난	00	87	20	40	73	38	48	55	44	95	19	65	51	17

6개의 사례 중에서 첫 번째 사례를 간단히 살펴보면 Usage, UT1, U11, S1, Window, Move, Drag Mouse, Left, Drag Mouse, Window, Close, Termination, Invocation 상태로 진행될 때 이전 상태에서 다음 상태로 천이되는 입력 이벤트를 나열하면 시험 사례가 된다. 쉽게 풀어쓰면 윈도우 시작, 윈도우 위짓중 ▲선택, 왼쪽으로 마우스 움직임, 윈도우로 마우스 움직임, Close 메뉴 선택순으로 사례가 생성된다.

〈표 21〉 사용자 인터페이스 구현

항 목	내 용
작 업 명	530 사용자 인터페이스 구현
목 적	사용자 인터페이스 프로그램 작성
태 스 크	사용자 인터페이스 설계 내용을 프로그램으로 구현 사용자 인터페이스 부분을 프로그램으로 개발

〈표 21〉 계속

항 목	내 용
입 력	사용자 인터페이스 절차 설계서
출 력	코드 작성
ISO 12207 관련 항목	소프트웨어 구현 및 시험
SPICE 관련 항목	ENG.1.4 , SUP.1 , MAN.1.1

3.6 테스팅 단계

테스팅 단계에서는 통합 시험과 신뢰도 공인, 결과 보고서 작성에 관한 작업을 수행한다. 생성된 시험 사례를 가지고 실제적인 시험을 수행하며, 고장 상태를 사용법 연쇄에 삽입하여 테스팅 연쇄를 구성한다. 고장 사이의 간격(MTBF)을 조사할 수도 있다.

〈표 22〉 통합 시험

항 목	내 용
작 업 명	610 통합 시험
목 적	시스템의 기능 및 성능의 통합 시험
태 스 크	시스템 특성을 반영한 통합 시험 계획 작성 통합 시험 수행 및 결과 분석
입 력	시스템 구조, 통합 시스템
출 력	시험 계획서, 시험 결과 보고서
ISO 12207 관련 항목	소프트웨어 구현 및 시험, 시스템 통합 소프트웨어 자격 시험, 시스템 자격 시험
SPICE 관련 항목	ENG.1.5 , ENG.1.6 , ENG.1.7 , SUP.1 , MAN.1.1 , MAN.2

일반적인 소프트웨어 컴포넌트 개발에서는 경험 또는 계약에 따라 신뢰도가 결정되어 있을때 신뢰도가 목표치를 달성하거나 초과할 때까지 고장 수정 및 반복 작업이 필요할 것이다. 하지만 신뢰도 공인 입장에서는 개발된 컴포넌트를 아예 기각 시킬 것인지, 또는 지속적인 품질 향상 작업을 들어갈 것인지 결정하고, 목표치를 초과할 경우는 수락하도록 공인 결과를 만들 수 있다.

〈표 23〉 신뢰도 공인

항 목	내 용
작 업 명	620 신뢰도 공인
목 적	테스트 결과를 이용하여 신뢰도 공인
태 스 크	확인 검증 공인
입 력	시험 결과 보고서

〈표 23〉 계속

항 목	내 용
출 력	신뢰도 공인 결과
ISO 12207 관련 항목	소프트웨어 구현 및 시험, 시스템 통합 소프트웨어 자격 시험, 시스템 자격 시험
SPICE 관련 항목	ENG.1.5 , ENG.1.6 , SUP.3 SUP.4 , SUP.5 , MAN.2

〈표 24〉 테스팅 결과 보고서 작성

항 목	내 용
작 업 명	630 테스팅 결과 보고서 작성
목 적	테스팅 결과 보고서 작성
태 스 크	테스팅 결과 보고서 작성
입 력	시험 결과
출 력	테스팅 결과 보고서
ISO 12207 관련 항목	소프트웨어 구현 및 시험, 시스템 통합 소프트웨어 자격 시험, 시스템 자격 시험
SPICE 관련 항목	ENG.1.5 , ENG.1.6 , SUP.1 , SUP.3 , MAN.2

경험 또는 계약에 따라 신뢰도가 정확한 수치로 결정되어 있지 않을때는 사용법 연쇄와 테스팅 연쇄 모두가 마코프 연쇄이므로, 통계적 속성이 허용된 한도 안에서 두 연쇄의 차이가 없음을 근거로 소프트웨어 컴포넌트를 수락할 것인지 결정할 수 있다.

3.7 배치 단계

배치 단계에서는 완성된 시스템을 고객의 요구 사항을 근거로 최종 시험하는 인수 시험과 시스템을 사용자에게 설치 후 운영하게 하는 배치 및 운영 작업을 수행한다.

〈표 25〉 인수시험

항 목	내 용
작 업 명	710 인수 시험
목 적	완성된 시스템을 고객의 요구사항을 근거로 최종 시험
태 스 크	고객의 요구사항을 근거로 하여 고객이 시험 인수 결정
입 력	요구 명세서, 통합 시험
출 력	인수 시험 결과서
ISO 12207 관련 항목	소프트웨어 인수 시험
SPICE 관련 항목	ENG.1.6 , ENG.2 , SUP.3 MAN.2

〈표 26〉 배치 및 운영

항 목	내 용
작 업 명	720 배치 및 운영
목 적	시스템을 사용자에게 설치 후 운영할 수 있게 한다
태 스 크	시스템 패키지화 사용자에게 설치 사용자 매뉴얼 전달 시스템 사용 교육
입 력	통합 시스템
출 력	소프트웨어 패키지, 사용자 메뉴얼, 운영자 메뉴얼
ISO 12207 관련 항목	소프트웨어 설치
SPICE 관련 항목	ENG.2 , SUP.1 , MAN.2

4.3 프로세스의 평가

4.3.1 제안 프로세스의 도출 근거

제안된 프로세스를 평가하기 전에 도출 근거를 보임으로서 연구의 논리적 배경을 명확히 하고자 한다. ISO 12207 표준안에서 기본 생명 주기 프로세스의 개발 프로세스 관련 항목을 수용하고, UML의 기본 프로세스를 따르고 있으며, 품질향상 방안을 도입하여 프로세스가 제안되었음을 <표 27>에서 도표화 하였다.

4.3.2 프로세스의 정성적인 평가

프로세스 평가는 간단하게는 특징, 장점, 단점과 같은 정성적인 특성 비교로 가능하다.

〈표 27〉 제안 프로세스의 도출 근거

단 계	ISO 12207 관련항목	UML 관련항목	품질향상 방안	제안된 프로세스
조 사	프로세스 구현	사업 모델링	관련사항 없음	110 시스템 조사
계 획	시스템 요구사항 분석 프로세스 구현	요구사항 분석	관련사항 없음	210 요구 명세서 작성 220 개발 계획서 작성
분 석	시스템 요구사항 분석 소프트웨어 요구사항 분석	요구사항 분석	소프트웨어 사용법 모델링	310 사용 사례도 및 사용법 연쇄 작성 320 시스템 분석 및 개념적 모델링 330 사용자 인터페이스 분석 340 분석서 작성
설 계	소프트웨어 구조 설계 소프트웨어 상세 설계	구조 계층 설계 클래스 계층 설계	사용법 프로파일 유도	410 시스템 설계 및 상세 모델링 420 사용법 프로파일 유도 430 사용자 인터페이스 및 DB 설계 440 설계서 작성
구 현	소프트웨어 구현 및 시험 소프트웨어 통합	구현	시험 사례 생성	510 클래스 구현 520 시험 사례 생성 및 단위 시험 530 사용자 인터페이스 구현
테스팅	소프트웨어 구현 및 시험 시스템 통합 소프트웨어 자격 시험 시스템 자격 시험	테스트	시험 사례 수행 및 고장 데이터의 수집	610 통합 시험 620 신뢰도 공인 630 테스팅 결과 보고서 작성
배 치	소프트웨어 인수 시험 소프트웨어 설치	배치	신뢰성 공인	710 인수 시험 720 배치 및 운영

〈표 28〉 제안한 프로세스의 특성 비교

항목 \ 프로세스	ISO 12207 프로세스	UML 프로세스	제안된 프로세스
특 징	· 국제 표준 · 권장 사항 · 적용자의 조정작업 필요 · 작성되어야 할 문서의 명확한 내용, 형식, 명칭까지는 기술 않음 · 특정한 생명주기 모델이나 소프트웨어 개발 방법론을 전제 않음	· 4단계의 일반적 사항만 명시됨 · 구체적인 프로세스가 없음 · 점진 반복적인 객체지향 개발	· 구축단계를 7단계 19개 세부 작업으로 구성 · 국제 표준 도입 · UML 기반으로하는 객체지향 방법에 이용 · 점진 반복적인 객체지향 개발 · 품질 향상 방안 도입
단 계	· 단계가 설정되지 않음	· 시작, 상세화, 구축, 발전 단계 · 예비반복, 상세화 단계	· 7단계(조사, 계획, 분석, 설계, 구현, 테스팅, 배치)

항목 \ 프로세스	ISO 12207 프로세스	UML 프로세스	제안된 프로세스
활 동	· 각 프로세스별로 활동 지정 · 기본 생명주기 프로세스[획득(5), 공급(7), 개발(13), 운영(4), 유지보수(6)] · 지원 생명주기 프로세스 · 조직 생명주기 프로세스	· 구체적인 활동은 정의되지 않음 · 활동의 종류로는 사업 모델링, 요구사항, 분석 및 설계, 구현, 시험, 배치	· 각 단계의 활동 정의됨 [조사(1), 계획(2), 분석(4), 설계(4), 구현(3), 테스팅(3), 배치(2)]
산출물	· 정해진 산출물 없이 적용자가 지정	· 사용사례도 · 정적구조 다이어그램(클래스 다이어그램) · 상태 다이어그램(활동도,상태도) · 상호작용 다이어그램(순서도, 협력도) · 구현 다이어그램(패키지도, 컴포넌트 다이어그램)	· 요구명세서, 개발 계획서 · 사용사례도, 사용법 연쇄 · 클래스도, 상태도, 순차도, 활동도, 협력도 · 분석서, 설계서, · 시험 사례, 테스팅 결과서
장 점	· 국제 표준 · 모든 종류의 프로젝트 적용 가능	· 점진 반복적인 객체지향 개발	· 국제 표준 도입 · UML 기반으로하는 객체지향 방법에 이용 · 품질 향상 방안 도입
단 점	· 자세한 프로세스는 적용자가 조정해야 하므로 업무 분야, 적용자의 능력에 따라 변화	· 구체적인 프로세스가 없어서 개발에 대한 정해진 지침이 없음	· 구축 단계만을 대상으로 구체적으로 프로세스를 조정

제안된 프로세스와 널리 알려진 많은 프로세스와의 비교는 생략하고, 관심이 되는 ISO 12207 프로세스와 UML 프로세스만을 <표 28>에서 비교하였다.

<표 28>에서 제시된 것처럼 제안된 프로세스는 ISO 12207의 표준 프로세스를 이용하면서 구체적으로 프로세스를 구현하는 과정을 도입한 객체지향 프로세스로서 UML을 지원할 수 있다. 또한, 공인과정을 도입하여 품질 보증 활동이 되도록 하였다.

4.3.3 SPICE 프로세스 레벨 평가

<표 29>는 프로세스 속성 전체에 대한 프로세스 레벨을 표시하였다. 제시된 표에 의하면 SPICE 레벨 5 수준으로 프로세스 능력 수준이 최적화되어 있다고 평가할 수 있다[7, 8, 9].

〈표 29〉 제안된 프로세스의 개발 범주에서 프로세스 능력 평가

프로세스 속성	개발프로세스						
	ENG1.1	ENG1.2	ENG1.3	ENG1.4	ENG1.5	ENG1.6	ENG1.7
PA1	F	F	F	F	F	F	F
PA2.1	F	F	F	F	F	F	F
PA2.2	F	F	F	F	F	F	F
PA3.1	F	F	F	F	F	F	F
PA3.2	F	F	F	F	F	F	F
PA4.1	F	F	F	F	F	F	F
PA4.2	F	F	F	F	F	F	F
PA5.1	L	L	L	L	L	L	L

〈표 29〉 계속

프로세스 속성	개발프로세스						
	ENG1.1	ENG1.2	ENG1.3	ENG1.4	ENG1.5	ENG1.6	ENG1.7
PA5.2	L	L	L	L	L	L	L
프로세스 레벨	Level 5	Level 5	Level 5	Level 5	Level 5	Level 5	Level 5

4.3.4 SPICE 프로세스 위험 평가

제안된 프로세스를 SPICE를 근거로 하여 <표 30>과 같이 위험 분석을 실시하였다. 위험 분석에서는 프로세스 속성의 목표값과 심사된 값을 서로 비교하여 위험 정도를 표시한 것으로 위험이 None, Low 로 평가될 때 평가되는 프로세스를 믿고 사용할 수 있다. <표 30>에 의하면 목표값을 PA5.1과 PA5.2에서만 Low Achieved 로 설정하고 나머지는 Fully Achieved 로 설정되었을 때 심사된 결과는 위험이 None 상태임을 알 수 있다.

〈표 30〉 SPICE에 근거한 위험 분석

프로세스		PA 1.1	PA 2.1	PA 2.2	PA 3.1	PA 3.2	PA 4.1	PA 4.2	PA 5.1	PA 5.2	위험
ENG 1.1	목표	F	F	F	F	F	F	F	L	L	None
	심사	F	F	F	F	F	F	F	L	L	None
ENG 1.2	목표	F	F	F	F	F	F	F	L	L	None
	심사	F	F	F	F	F	F	F	L	L	None
ENG 1.3	목표	F	F	F	F	F	F	F	L	L	None
	심사	F	F	F	F	F	F	F	L	L	None

〈표 30〉 계속

프로세스		PA 1.1	PA 2.1	PA 2.2	PA 3.1	PA 3.2	PA 4.1	PA 4.2	PA 5.1	PA 5.2	위험
ENG 1.4	목표	F	F	F	F	F	F	F	L	L	None
	심사	F	F	F	F	F	F	F	L	L	None
ENG 1.5	목표	F	F	F	F	F	F	F	L	L	None
	심사	F	F	F	F	F	F	F	L	L	None
ENG 1.6	목표	F	F	F	F	F	F	F	L	L	None
	심사	F	F	F	F	F	F	F	L	L	None
ENG 1.7	목표	F	F	F	F	F	F	F	L	L	None
	심사	F	F	F	F	F	F	F	L	L	None

5. 결 론

본 논문에서는 ISO 12207 소프트웨어 생명주기 프로세스를 따르며 UML을 기반으로 하는 객체지향 개발 프로세스를 제안했다. 이 프로세스는 신뢰성 중심의 품질을 향상시킬 수 있도록 개발 단계, 활동, 산출물을 포함하는 7단계 19개의 세부 작업으로 구성된 개발 프로세스이다. 제안된 프로세스는 분석, 설계, 구현, 테스팅 단계에서 공인을 위한 작업을 수행하여 소프트웨어 컴포넌트의 품질을 보증할 수 있다.

제안된 객체지향 프로세스의 유용성을 증명하기 위하여 프로세스 특성의 비교 분석, SPICE에 근거한 개발 프로세스의 품질 평가 및 프로세스 레벨 계산, SPICE에 의한 위험 분석을 이용하였다.

제안된 객체지향 프로세스를 활용함으로서 소프트웨어 테스팅을 위한 시험 사례를 사용자 중심으로 생성하며, 시험 사례 생성의 과비용 문제점을 해결할 수 있다. 또한, 통계적 해석을 이용하여 컴포넌트의 신뢰성을 공인함으로서 소프트웨어 품질을 향상시키고 품질을 보증할 수 있다.

참 고 문 헌

[1] 강문설, 김병기, 김태희, 이상준, *시스템 분석 및 설계*, 정익사, 1998.

[2] 이주헌, *실용 소프트웨어 공학론*, 법영사, 1993.

[3] 정기원, 윤창섭, 김태현, *소프트웨어 프로세스와 품질*, 홍릉과학출판사, 1997.

[4] G. Booch, *Object-Oriented Design with Applications*, Benjamin/Cumming, 1991.

[5] Martin Fowler, *UML Distilled Applying the Standard Object Modeling Language*, Addison-Wesley, 1997.

[6] ISO 12207, *ISO/IEC 12207 Standard for Information Technology - Software Life Cycle Processes*, ISO/IEC JTC/SC7, March, 1998.

[7] ISO 15504, *Part 2 : A Reference Model for Processes and Process Capability*, ISO/IEC JTC/SC7, 1998.

[8] ISO 15504, *Part 4 : Guide to Performing Assessments*, ISO/IEC JTC/SC7, 1998.

[9] ISO 15504, *Part 5 : An Assessment Model and Indicator Guidance*, ISO/IEC JTC/SC7, 1997.

[10] Ivar Jacobson, Grady Booch, James Rumbaugh, *Unified Software Development Process*, Addison Wesley, 1991.

[11] J. D. Musa, A Theory of Software Reliability and Its application, *IEEE Trans. Software Eng.*, Vol.SE-1, Aug. 1975.

[12] J. D. Musa, Operational Profiles in Software Reliability Engineering, *IEEE Software*, Mar. 1993.

[13] R. Pressman, *Software Engineering : A Practitioner's Approach(4th Edition)*, McGraw Hill, 1997.

[14] S. Ross, *A First Course in Probability*, Macmillan Publishing Company, 1988.

[15] James A. Whittaker and J. H. Poore, Markov Analysis of Software Specifications, *ACM Transactions on Software Engineering and Methodolody*, Vol.2, No.1, January 1993.

[16] James A. Whittaker and Michael G. Thomason, A Markov Chain Model for Statistical Software Testing, *IEEE Trans. on Software Engineering*, Vol.20, No.10, Oct. 1994.

이 상 준

e-mail : sjlee@tiger.seonam.ac.kr

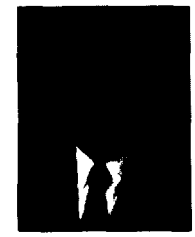
1991년 전남대학교 전산통계학과 (이학사)

1993년 전남대학교 전산통계학과 (이학석사)

1999년 전남대학교 전산통계학과 (이학박사)

1995~현재 서남대학교 컴퓨터정보통신학부 조교수

관심분야 : 소프트웨어 공학, 객체지향 시스템, 사용자 인터페이스, 분산처리 시스템



김 병 기

e-mail : bgkim@chonnam.chonnam.ac.kr

1978년 전남대학교 수학과(이학사)

1980년 전남대학교 수학과(이학석사)

1996년 전북대학교 수학과(박사수료)

1981년~현재 전남대학교 컴퓨터정보학부 교수

관심분야 : 소프트웨어 공학, 병렬처리, 소프트웨어 에이전트, 객체지향시스템