극한 프로그래밍의 사용성 향상 방안

이 상 준† · 배 석 찬††

요 약

극한 프로그래밍은 빠르게 변화되는 사업 환경에 기민하게 대응하는 애자일 소프트웨어 개발 방법론 중에서 가장 대표적인 방법론이다. 소프트웨어 산업이 성숙됨에 따라 소프트웨어 품질 요소중 사용성이 점차 강조되고 있으나, 극한 프로그래밍에서의 사용성을 확보하기 위한 노력이 부족하다. 본 논문에서는 극한 프로그래밍에 부족한 3가지 사항을 보강한다. 사용자 인터페이스 설계자의 역할을 정의 및 제시하고, 사용성 평가 방법 도입 방안 제시하고, 개발 프로세스 및 산출물을 제안한다. 제안 방안의 타당성 분석을 위해 4가지 방법을 사용하였다. 첫째, 소프트웨어 개발 생명주기 지원 여부 분석, 둘째, CMM 핵심 프로세스 영역 만족도 분석, 셋째, CMM 규정 실무 만족도 향상 분석, 넷째, 녹차 쇼핑몰을 개발하는 사례를 분석하였다. 녹차 쇼핑몰은 사용성 평가 방법의 도입으로 예측된 실행시간이 23%, 학습용이성이 21% 향상되었다. 또한, 휴리스틱 평가 점수가 18%, 설문지 평가 점수는 16% 향상되었다.

A Plan for Improvement of Usability in Extreme Programming

Sang-Jun Lee† · Seok Chan Bae††

ABSTRACT

Extreme programming is the most representative methodology among agile software development methodologies that is agile in business environment which change fast. As software industry is matured, usability of software quality characteristics is emphasized gradually, but effort to obtain usability in extreme programming is insufficient. In this paper, three things lacked in extreme programming are reinforced. First, roles of user interface expert are defined. Second, usability testing method to extreme programming are introduced. Third, development process and products are proposed. The proposed plan is validated by four methods, which analyze supporting software development life cycle, analyze satisfaction of CMM key process areas, analyze satisfaction of CMM practices, and analyze development of green tea shopping mall. Green tea shopping mall is improved 23% in the estimated running time, 21% in the learnability. Also, usability is improved 18% in the heuristic evaluation and 16% in the questionnaire method.

키워드 : 소프트웨어 개발 방법론(Software Development Methodology), 소프트웨어 품질(Software Quality), 애자일 소프트웨어 개발(Agile Software Development), 극한 프로그래밍(Extreme Programming), 소프트웨어 사용성(Software Usability)

1. 서 론

소프트웨어 프로세스 모델에 있어서 전통적인 폭포수 모델은 사용자가 개발자에게 한 번에 모든 요구사항을 정확하게 전달하는 것을 가정하고 있으나, 현실적으로 사용자들은 자신들이 심지어 무엇을 원하는지 정확히 모를 뿐만 아니라 요구사항을 한 번에 모두 이야기하지도 않고 자주 마음을 바꾸기도 한다. 이런 문제를 해결하고자 애자일(Agile) 소프트웨어 개발 방법론이 등장하게 되었다[11, 15, 27]. 소프트웨어 개발의 수명주기를 짧게 반복하는 반복형 모델은 요구사항의 변경에 대한 소프트웨어 프로세스의 적응성을 개선하는 효과를 얻을 수 있으며, 극한 프로그램(XP : eXtreme Programming)은 반복형 모델의 개발주기를 극단적으로 짧게 함으로써, 프로그래머가 설계, 구현, 시험 활동을 전체 소프트웨어 개발 기간에 걸쳐 조금씩 자주 실시하도록 하고 소프트웨어 변경의 비용을 최소화하는 애자일 기법이다[36].

소프트웨어 산업이 성숙될수록 단순한 개발보다는 제품의 품질을 강조하는 것과 같이, 소프트웨어 품질 특성 관점에서는 기능성 중심에서 신뢰성과 사용성을 점차 강조하고 있고, 점점 더 많은 조직이 사용성을 중요하게 받아들이기 시작했다[23]. 보통 개발자는 사용성 개념을 채택하지 않아서, 소프트웨어 제품의 사용성 레벨이 향상되지 못하고 있다[14]. 소프트웨어 사용성 향상을 위하여 사후처리 대신에 개발 프로세스의 핵심 컴포넌트로 사용자 중심의 설계 및 평가를 도입할 수 있다. 사용성 중심의 개발이란 초기 설계 향상과 재작업을 줄이도록 가능한 개발 단계의 시작 부분

※ 본 논문은 정보통신부 정보통신연구진흥원에서 지원하고 있는 정보통신기초기술연구지원사업의(03-기초-0057) 연구결과입니다.
† 종신회원 : 서남대학교 컴퓨터정보통신학과 교수
†† 종신회원 : 군산대학교 컴퓨터정보과학과 교수
논문접수 : 2003년 8월 22일, 심사완료 : 2004년 3월 2일

에 사용성을 확보하도록 하는 것이다. 보호활동 프로세스로 사용성을 제공하는 것보다는 사용성 테스팅을 소프트웨어 개발 프로세스에 완벽하게 통합해야 좋다[6].

XP 실무기법에 사용성 기술을 도입하여 강력하고, 단일한 소프트웨어를 개발 환경을 구축하는 연구가 필요하다. 본 논문은 XP에 사용성 테스팅 방법의 도입을 제안하고, XP의 개발 방법론적 측면, 프로세스(활동) 측면, 품질 측면의 미비한 점을 개선하는데 **목표**를 두고 연구하였다.

본 논문은 XP가 갖는 사용성 관련 문제를 식별하고, 해결 방안을 제시하고, 해결 방안을 프로세스 자체에 대한 정성적인 평가 방법과 예제에 적용한 정량적인 평가 방법으로 타당성 분석하는 **방법**으로 연구하였다.

기존의 XP 연구에서는 소프트웨어 사용성 관점에서 다음 네 가지 **문제점**을 갖고 있다.

- 방법론을 구성하는 역할 영역 중 사용자 인터페이스 설계자의 역할이 정의되지 않았다.
- XP에서의 사용성에 대한 연구의 사례가 없다.
- CMM의 관리 및 하부구조 지원 프로세스를 지원하지 못하고 있다.
- XP 프로젝트의 활동에서 사용성에 대한 고려가 없다.

XP의 사용성 향상을 위하여 본 논문에서는 다음 세 가지 사항을 **제안**한다.

- 사용자 인터페이스 설계자의 역할을 정의한다(3.2절).
- 사용성 평가 방법 도입 방안을 제시한다(3.3절).
- 개발 프로세스 및 산출물을 정의한다(3.4절).

본 논문에서 제안한 극한 프로그래밍의 사용성 향상 방안에 다음 4가지 방법을 적용하여 **연구의 타당성**을 보인다.

- 소프트웨어 개발 생명주기 지원 여부(3.5절)
- CMM 핵심 프로세스 영역 만족도 분석(3.5절)
- CMM 규정 실무 만족도 향상(3.5절)
- 적용 사례 분석(4장)

인터넷 쇼핑몰(녹차 쇼핑몰)은 많은 사용자들이 접속하는 인터넷 사이트로서 쇼핑몰 상품의 변화, 소비자들의 기호 변화에 따라 짧은 시간에 민첩하게 변화되어야 하기 때문에, 극한 프로그래밍 방법론을 적용하는 것이 좋다. 또한, 사용자와 상호작용이 많아서 소프트웨어 사용성이 강조된다.

논문의 2장에서는 극한 프로그래밍과 사용성 테스팅의 관련 연구를 소개하고, 3장에서는 극한 프로그래밍에서 사용성 향상 방안을 제시한다. 4장에서는 사용성 향상 방안을 형상 관리 시스템에 적용한 사례를 소개한다. 5장에서는 연구의 결론을 맺는다.

2. 관련 연구

2.1 극한 프로그래밍

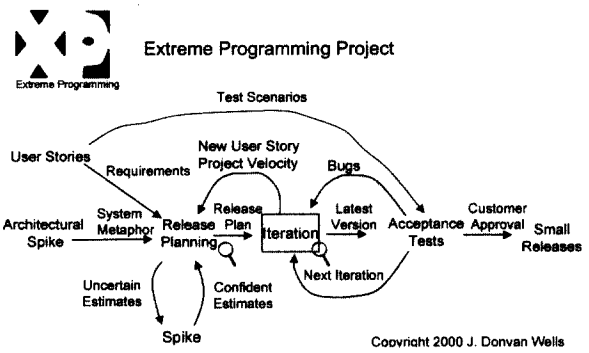
소프트웨어 개발자는 구현된 소프트웨어를 빨리 만들어야 한다는 강한 압박을 받고 있다[7, 12]. 애자일 소프트웨어 개발 방법론은 "인터넷 시간"에 소프트웨어를 개발할 수 있는 더 가벼우며 빠르고 민첩한 소프트웨어 개발 프로세스에 대한 해답을 제공하기 위해 개발되었기 때문에 인터넷 소프트웨어 산업 뿐만 아니라 모빌 어플리케이션 환경에 좋다. 소프트웨어 개발 프로세스의 앞으로의 동향을 논의하는 2001년 모임에서 "애자일 소프트웨어 개발"의 가치와 목표를 정의하고, "애자일 연합 헌장"으로 12가지 공통된 원리를 제시하였다[3]. 애자일 연합에서는 소프트웨어 개발 방법론에서의 애자일을 "메소드의 경량"과 같은 의미로 정의하였다. 애자일의 정도를 나타내는 기민성(Agility)은 조직의 기민성, 사업의 기민성, 민첩한 제조, 애자일 소프트웨어 개발에서부터 유래되었다. 기민성은 소프트웨어개발 조직에 대하여, 환경의 변화와 환경에 의해 부과된 요구에 신속하게, 적당하게, 채택하게, 반응하게 하는 능력으로 정의되었다[25].

2002년 11월부터 2003년 1월 동안에 Shine Technology에서 애자일 방법론에 대한 시장 관심을 측정하기 위해 웹기반 조사를 실시하였다. 10개의 질문에 정확하게 응답한 131개의 타당한 결과를 분석한 결과 애자일 방법론이 생산성에 도움이 됐다는 응답이 93%, 품질이 향상되었다는 응답이 88%, 비용이 적게들었다가 49%, 사업적 만족도 83%, 2003년에 적용할 것인가에 94.7%로 응답하였다[4]. 지난 2002년 3월 Giga Information Group Inc.는 응용 소프트웨어 개발 컨퍼런스에서 18개월 이내에 전세계 IT 기업의 2/3 이상이 일종의 애자일 방법론을 사용하게 될 것이라고 예측했다[16].

애자일 방법론으로는 Extreme Programming[8], Scrum[31], Crystal Mehods[11], Feature-Driven Development[29], Rational Unified Process[23], Dynamic Systems Development Method[33], Adaptive Software Development[17], Open Source Software Development[18], Agile Modeling[5], Pragmatic Programming[19] 등이 있다. 설문 조사에 의하면 응답자의 59%가 10여개의 애자일 개발 방법론 중에서 극한 프로그래밍(XP)을 가장 많이 사용하였다[3]. XP는 다양한 애자일 소프트웨어 개발 방법론의 최초 연구로 널리 알려져 있다.

XP는 의사소통, 피드백, 단순성, 용기의 4가지 가치를 기반으로 고객, 관리자, 프로그래머에 대한 역할과 권한을 중시하고, "고객에게 최고의 가치를 가장 빨리" 전달하도록 하는 경량 소프트웨어 개발 방법론이다. 요구사항 등의 변화가 자주, 많이 있거나 개발자가 소규모(10명 내외)이고

같은 공간을 사용하는 경우에 높은 효과를 볼 수 있다고 알려져 있고, 다른 규모나 원거리 XP 등의 적용이 꾸준히 시도되고 있다. XP에서는 12개의 실무로서 계획, 작은 릴리즈, 메타포, 단순 설계, 리팩토링, 테스팅, 짝 프로그래밍, 공동 소유권, 지속적 통합, 주당 40시간, 작업에 고객 동참, 코딩 표준을 사용하고 있다. XP가 갖는 다른 소프트웨어 개발 방법론과의 가장 큰 차이점은 테스팅을 강조한다는 점이다. XP에서 테스팅은 모든 개발의 기초이다. XP 프로그래머는 코드를 작성한 것과 같이 테스트를 작성하여야 한다. 테스팅이 빌드 프로세스에 통합됨으로서, 매우 안정적이고 확장가능한 프로덕트를 얻게된다. Don. Well에 의해 표현된 일반적인 XP 프로젝트의 전체적인 구성은 (그림 1)과 같다[35].



(그림 1) XP 프로젝트

2.2 소프트웨어 사용성

정보화 사회로의 전환이 급속하게 이루어지면서 사회적으로 컴퓨터 및 소프트웨어에 대한 의존도도 크게 증가하고 있다. 인터넷과 정보화 사회의 발전에 따라 소프트웨어 개발 분야에서는 높은 생산성과 품질의 향상을 가장 중요한 과제로 여기고 있다. 소프트웨어 개발의 목표를 우수한 품질의 소프트웨어를 획득하는 것으로 했을 때, 목표로 하는 품질에 대한 정의를 해야하고, 현재의 품질과 목표치를 비교하여 만족할 만한 품질이 되도록 지속적으로 품질을 향상시키는 노력을 해야 한다. 이때 현재 품질이 어떤 상태인지 평가하는 일은 꼭 필요하다.

소프트웨어 제품 자체의 품질은 기능성, 신뢰성, 사용성, 효율성, 유지보수성, 이식성의 6가지 특성을 바탕으로 평가할 수 있다[21]. 사용성은 명시적 또는 암시적 사용자가 사용하기 위해 필요한 노력으로 각각의 사용 결과에 의한 평가를 나타내는 속성의 집합을 의미한다. 사용성은 이해성, 습득성, 운용성, 친밀성의 품질 부특성을 갖는다. 이해성은 소프트웨어의 논리적인 개념을 사용자가 이해할 수 있는 노력 정도를 나타내는 소프트웨어 속성이며, 습득성은 소프트웨어 운용관리, 입출력 방법 등을 습득하기 위한 사용자의 노력을 나타내는 소프트웨어 속성이며, 운용성은 소프트웨어의 운용과 운용관리하기 위한 사용자 노력 정도를 나타내는 소프트웨어 속성이다.

ISO 9241-11에 따르면 사용성은 제품이 특정 사용자에 의해 특정 사용 문맥에서 효과성, 효율성, 만족도의 특정 목표에 성취하는 정도이다[20]. 인간 공학의 관점에서는 효율성과 효과성도 사용성의 구성 요소로 간주한다. 사용성은 학습 용이성, 사용 효율성, 기억 용이성, 최소한의 에러를 포함하며, 넓게는 미학적 구성에 의해 상당 부분 영향을 받는 주관적 즐거움 까지도 포함한다[10, 28]. 좋은 소프트웨어가 되기 위해서는 익히기 쉬워야 하고, 한번 익혀진 것은 높은 사용 효율을 발휘해야 하며, 시간이 흐른 뒤에도 기억하기 쉽고, 사용자의 실수를 유발하지 않아야 하는 사용성이 강조된다.

사용성을 평가하는 방법으로는 사용자 관찰, 성능 측정, 질문지, 생각 크게 말하기, 모델기반 방법, 전문가 평가등 여러 방법이 있다. 적용단계, 스타일, 객관적 유무, 측정값

〈표 1〉 Dix에 의한 사용성 평가 방법의 특성

분 류	분석적 평가 방법				실증적 평가 방법			실중적 평가 방법		
	인지 검토회	휴리스틱 평가	검토 기반 평가	모델 기반 평가	실 험	질의 평가 기법		관찰 평가 기법		
					실 험	인터뷰	질문지	생각 크게 말하기	프로토콜 분석	past-walk 워커스루
적용 단계	설계,구현	설계,구현	설계,구현	설계,구현	설계,구현	설계,구현	설계,구현	구현	구현	구현
스타일	실험실	실험실	실험실	실험실	실험실	실험실,필드	실험실,필드	실험실/필드	실험실/필드	실험실/필드
객관적 유무	비객관적	비객관적	소스 의존	비객관적	객관적	비객관적	비객관적	비객관적	비객관적	비객관적
측정값 유형	정성적	정성적	소스 의존	정성적	정량적	정성/정량	정성/정량	정성적	정량적	정량적
정보수준	저수준	고수준	소스 의존	저수준	저/고	고수준	고수준	고/저	고/저	고/저
응답의 긴박	n/a	n/a	소스 의존	n/a	긴박	비긴박	비긴박	긴박	긴박	비긴박
사용자 행동	변화 없음	변화 없음	변화 없음	변화 없음	변화 있음	변화 없음	변화 없음	변화 있음	변화 있음	변화 없음
시간 소비	중수준	저수준	저-중	중수준	고수준	저수준	저수준	고수준	고수준	중수준
장비 필요	저수준	저수준	저수준	저수준	중수준	저수준	저수준	저수준	고수준	저수준
전문성 수준	고수준	중수준	저수준	고수준	중수준	저수준	저수준	중수준	고수준	중수준

〈표 2〉 ISO/TR 16982 사용성 평가 방법 선택 첫 번째 기준

프로젝트 특성	메소드 범주											
	사용자 관찰	성능 측정	치명적인 사건 분석	질문지	인터뷰	생각 크게 말하기	협력 설계 및 평가	창조적 방법	문서기반 방법	모델기반 방법	전문가 평가	자동화된 평가
생명주기 프로세스												
획득-공급	++	+	+	+	+		+		++		+	
개발-요구 분석	++	+	+	++	++	++	+	+	+	+	+	
개발-아키텍처 설계	+	++		+	+	++	+	++	++	+	+	+
개발-자격 시험	+	++	+	++	++	+	+		+	+	+	+
유지보수-운영	+	+	+	+	+		+				+	
프로젝트 환경 제약사항												
치밀한 시간 규모		−	−	−		−	−		+	−	++	+
비용/가격 제어		−	−		−	−		−	++	−	+	
고품질레벨	++	++	+	++	++	+	+	+	+	+	+	+
초기 정보/피드백/진단 필요	+			+	++		+	+			+	
진화적 명세	+	+	+	+	+	+	++	+				

범례 ++ : 권장함 + : 적당함 빈칸 : 보통 − : 권장 안함 NA : 적용 불가

〈표 3〉 ISO/TR 16982 사용성 평가 방법 선택 두 번째 기준

특 성	메소드 범주											
	사용자 관찰	성능 측정	치명적인 사건 분석	질문지	인터뷰	생각 크게 말하기	협력 설계 및 평가	창조적 방법	문서기반 방법	모델기반 방법	전문가 평가	자동화된 평가
사용자 특성												
포함/접근 불가	NA	NA	NA	NA	NA	NA	NA	NA	+	+	+	+
포함/접근 불가	++	++	+	++	++	+	++	+	+	+	+	+
현저한 불가능성	++	+	+	+	++	+	++	+	+	−	+	−
태스크 특성												
매우 복잡	+	+	++	+	++	++	+	+		+		
에러가 심각한 결과 유도	++	++	++	+	+	+	+		+	++	+	
사용자에게 완전히 새로운 작업	+		NA				++	++	+	+	+	
광범위 태스크 스펙트럼	+	+	+	++	+	+	+	+	++	+	+	++
조직/작업/기술의 큰 변화	+	+	+	+	+	+	++	++	+	−	+	−
높은 시간과 정확도 제약사항	+	++	++			−	−	−	−	+	−	−
제품 특성												
기존 시스템/제품 적용	+	++	++	++	+	+	+		++	++	+	+
제한되고 간단한 잘 이해된 제품	+	+		++	+		+		++		++	
고차원 적응성	+	+	+	+	++	+	++	+				
기술 논점												
인간공학 기술/경험 소유	++	++	++	++	++	++	++	++	++	++	++	++
제한된 능력 소유	+	−	−	+	−	−	+	+	+	−	NA	+

범례 ++ : 권장함 + : 적당함 빈칸 : 보통 − : 권장 안함 NA : 적용 불가

유형, 정보 수준, 응답의 긴박, 사용자 행동, 시간 소비, 장비 필요, 전문성 수준에 따른 사용성 평가 방법의 비교가 <표 1>에 있다[13].

ISO/TR 16982에서의 사용성 평가 방법은[22] <표 2>와 같이 어떤 생명주기 프로세스에서 사용하는 평가 방법인지, 프로젝트 환경 제약사항으로는 어떤 것이 있는가를 제1 선택 기준으로 삼는다.

제1 선택기준을 우선 고려한 후 사용자 특성, 태스크 특성, 제품 특성에 따라 사용성 평가 방법을 선택한다. <표 3>은 제2 선택기준이다.

3. 극한 프로그래밍에서 사용성 향상 방안

3.1 극한 프로그래밍의 문제점

기존의 XP 연구에서는 소프트웨어 사용성 관점에서 다음 네 가지 문제점을 갖고 있다.

- 방법론을 구성하는 역할 영역 중 사용자 인터페이스 설계자의 역할이 정의되지 않았다.
- XP에서의 사용성에 대한 연구의 사례가 없다.
- CMM의 관리 및 하부구조 지원 프로세스를 지원하지 못하고 있다.
- XP 프로젝트의 활동에서 사용성에 대한 고려가 없다.

방법론의 범위는 생명주기 영역, 역할 영역, 활동 영역의 3축으로 특성화 할 수 있다. XP는 방법론의 범위로 볼 때 사용자 인터페이스 설계자의 역할 영역이 정의되어 있지 않음으로 사용성에 대한 고려가 없는 실정이다[11]. 사용자 인터페이스는 소프트웨어 개발 노력의 40~60%을 차지할 만큼 중요하게 다뤄야할 부분이다. 사용자 인터페이스 설계자는 사용자 인터페이스 요구사항을 파악하고, 워크스루 태스크를 수행하여 설계의 문제점을 찾고, 실제 사용자 인터페이스를 설계한 후, 사용성을 평가해야 한다.

XP와 애자일 개발 방법론을 제창했던 Beck은 소프트웨어 개발의 4가지 변인으로 비용, 시간, 품질, 범위를 소개하였다[9]. 소프트웨어 개발 현장의 선임 관리자는 고객 만족도 향상의 필요성을 인식하고, 사용성 실무를 주요한 수단으로 도입하고 있으나[6], XP에서는 소프트웨어 품질특성중 기능성 중심으로 품질을 확보하는 노력을 할뿐, 신뢰성, 사용성, 효율성, 유지보수성, 이식성에 대한 고려가 구체적이지 못하고 미비하다. 소프트웨어 개발을 개발자만이 수행하는 것이 아니라 프로젝트 초기부터 사용자를 참여시키자는 실무기법의 특성을 갖는 XP에서 사용성은 중요하게 다뤄야 할 부분임에도 불구하고 XP에서의 사용성에 대해 연구된 사례가 없다.

XP가 단순한 프로그래밍 기법 이상의 소프트웨어 개발 패러다임으로 존재하기 위해서는 프로세스에 대한 정의가 있어야 하지만, XP의 많은 연구자들이 합의한 프로세스는 없고, 4개의 가치와 12개의 실무기법만을 정의하여 이를 바탕으로 프로세스를 정의하여 사용하도록 하고 있다. 소프트웨어 프로세스 품질을 성숙도로 나타내는 CMM의 핵심 프로세스 영역(KPA : Key Process Area)들을 XP가 얼마나 만족하는가에 대한 연구에 의하면 XP는 관리 및 하부구조의 지원이 필요하다. XP와 관련된 많은 책과 논문과 코스, 웹사이트에는 부분적으로 CMM의 조직 프로세스 정의와 훈련 프로그램등의 KPA를 삽입하여 보완하는 연구를 수행하고 있다[30].

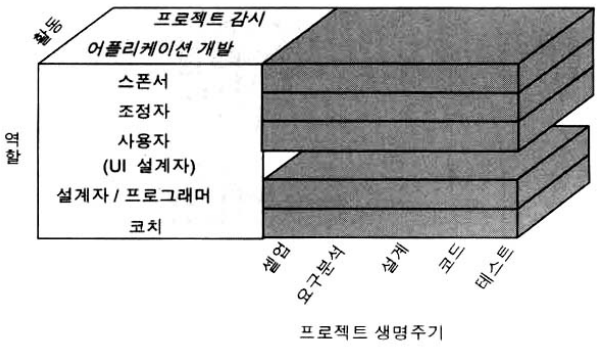
XP에 대한 대표적인 사이트인 extremeprogramming.org의 홈 페이지에는 XP/Agile Universe 2002 국제 학술회를 주관했던 Don. Well에 의해 표현된 일반적인 XP 프로젝트의 흐름이 소개되어 있다[35]. 프로젝트를 구성하는 사용자 스토리, 아키텍처 스파이크, 릴리즈 계획, 스파이크, 반복, 인수 시험, 스몰 릴리즈와 같은 요소들에서 사용성에 대한 어떠한 고려도 하고 있지 않다.

XP의 사용성 향상을 위하여 본 논문에서는 다음과 같은 세 가지 사항을 **제안**한다.

- 사용자 인터페이스 설계자의 역할을 정의한다(3.2절).
- 사용성 평가 방법 도입 방안을 제시한다(3.3절).
- 개발 프로세스 및 산출물을 정의한다(3.4절).

3.2 극한 프로그래밍에서 사용자 인터페이스 설계자 역할 정의

XP를 일종의 방법론으로 간주할 때, 특성을 지원하는 프로젝트 생명주기, 정의된 역할, 활동의 3차원으로 표현할 수 있다. (그림 2)와 같이 XP에서는 사용자 인터페이스 설계자의 임무에 대한 정의가 없다. 그만큼 사용성에 대한 고려가 부족하다.



(그림 2) XP의 범위

본 논문에서는 사용자 인터페이스 설계자의 역할을 다음과 같은 임무, 설계 규칙, 접근법으로 정의한다.

- 사용자 인터페이스 설계자의 임무를 다음과 같이 정의한다
  - 사용자 인터페이스 요구사항을 파악한다.
    - 사용자, 작업, 환경 등의 분석 및 모델 제작
  - 워크스루 태스크를 수행하여 설계의 문제점을 찾는다.
  - 실제 사용자 인터페이스를 설계한다.
    - UI 정의, 그들간의 연결
  - 인터페이스를 구축한다.
  - 사용성을 평가해야 한다.
- 사용자 인터페이스 설계 시에는 다음과 같은 규칙을 지켜야 한다.
  - 사용자가 제어할 수 있어야 한다.
  - 사용자 기억 부담을 줄여야 한다.
  - 일관성 있게 만들어야 한다.
- 인터페이스 설계 접근법으로 다음을 이용한다.
  - 각 작업의 목적과 의도를 정립하라.
  - 각 목적과 의도를 특정 동작의 순서로 대응시켜라.
  - 작업 및 하위작업이 인터페이스에서 실행될 때의 동작 순서를 명세하라.
  - 시스템 상태를 지적하라.
  - 제어체계를 정의하라.
  - 제어체계가 시스템 상태에 영향을 미치는 방법을 보여라.
  - 인터페이스에서 제공되는 정보를 보고 사용자가 시스템 상태를 판단할 수 있는지 지적하라.

3.3 극한 프로그래밍에서 사용성 평가 방법 도입

많은 종류의 사용성 평가 방법은 특정 방법 1개만을 사용해도 되고, 각 평가 방법의 단점을 보완하기 위해 몇 개의 평가 방법을 같이 사용할 수 있다. 2.2절 사용성에서 소개된 평가 방법 중에서 XP에 적용하기 부적합한 방법들이 있다. XP가 소프트웨어 개발 생명주기에 사용가능해야 함으로, <표 2>의 기준에 의하여 치명적인 사건분석 방법, 창조적 방법은 사용할 수 없다. 프로젝트 환경 제약사항중 전문적인 사용성 평가 실험실을 마련하기 위한 비용과 시간이 많이 드는 사용자 관찰 방법, 성능 측정 방법, 생각 크게 말하기 방법은 제외한다. XP는 가능한한 문서화를 하지 않기 때문에 문서기반 방법을 사용할 수 없다.

본 논문에서는 <표 4>와 같은 3가지 사용성 평가 방법을 선택하여 적용하고자 한다.

- 인수 시험 활동에서 사용성 테스팅을 수행하기 위해 질문지를 이용하도록 제안한다.
- 명세, 구현, 테스팅 과정에서 사용성 테스팅 수행을 위해 GOMS(Goals, Operators, Methods, Selection rules), 휴리스틱 평가 방법의 도입을 제안한다.

사용성 평가 방법중 학문적으로 가장 정리가 잘 되어있으며 사용성 평가 결과를 이용하여 설계를 수정할 수 있는 가이드라인까지 제공할 수 있는GOMS를 이용한 모델 기반 방법과, 사용자의 느낌을 가장 쉽게 평가할 수 있는 질문지 방법, 전문가에 의해 숨겨진 문제점까지 찾아내올 수 있는 휴리스틱 평가 방법을 도입하고자 한다.

3.3.1 GOMS 방법 활용

GOMS란 목표, 조작자, 방법 및 선택 규칙에 기초를 두고 컴퓨터 시스템과 함께 업무를 수행할 때의 사용자 지식과 작업을 기술하는 것이다. 이 방법은 전문가를 참가자 대상으로 하며 수행능력과 수행시간을 정밀하게 검사한다. 논리적인 수행 과정을 검증하고자 하거나 반복적인 사용 시간을 단축시키는 것이 목적일 경우에 매우 유용한 방법이다[24].

GOMS 모델을 구축 절차는 다음과 같다.

〈표 4〉 사용성 평가 방법 선택 기준 및 선택 결과

기준	사용자 관찰	성능 측정	치명적인 사건 분석	**질문지**	인터뷰	생각 크게 말하기	협력 설계 및 평가	창조적 방법	문서기반 방법	**모델기반 방법**	**전문가 평가**	자동화된 평가
개발 생명주기			×					×				
소요비용	×	×				×						
문서화 위주									×			
학문적 체계화										○		
사용자 느낌 쉽게 평가 가능				○								
숨겨진 문제 발견가능											○	

Step A : 최상위 레벨 사용자 목표 선택
Step B : 다음 반복 절차를 수행 :
B1. 각각의 목표를 수행하기 위한 메소드 초안 작성
B2. 초안을 완성한후, 일관성과 가이드라인에 대한 일치에 대한 요구를 이용하여 점검하고 검토
B3. 필요하다면, 고수준 연산자를 수행-목표 연산자로 변경함으로서 분석의 낮은 수준으로 이동하고, 적합한 목표에 대한 메소드를 제공
Step C : 분석을 문서화하고 점검
Step D : 판정과 가정에 대한 민감성 점검

GOMS 방법에 인지복잡도 이론을 근거로 한 NGOMSL (Natural GOMS Language)이 있다. NGOMSL은 모형 구축에 필요한 규칙과 용어를 단순화하고, 모형 구축을 용이하게 하며, 사용성 관련 자료를 수치적으로 예측할 수 있도록 하였다. XP의 설계 결과를 검증할 때 NGOMSL 모델을 적용하도록 한다.

〈표 5〉 NGOMSL 실행시간 예측 공식

- 실행시간 예측 = NGOMSL 문장 시간 + 원시 외부 연산자 시간
  + 분석자 정의 정신 연산자 시간
  + 시스템 응답 시간
- NGOMSL 문장 시간 = 실행된 NGOMSL 문장수 × 0.1초
- 원시 외부 연산자 시간 = 외부 연산자에 대한 전체 시간
  키 스트로크 : 0.28초
  마우스버튼 누름 : 0.1초
  마우스 이동 : 1.1초
  키보드나 마우스로 손을 가져감 : 0.4초
- 분석가 정의 정신 연산자 시간 = 분석가에 의해 정의된 정신 연산자에 대한 전체 시간
  card의 모델에 의해 정신 연산자 시간 : 1.35초
- 시스템 응답 시간 = 사용자가 쉬고 있을 때 전체 시간

NGOMSL에 의한 예측 결과를 이용하여 설계를 수정할 수 있고, 수정한 후 성능예측을 다시 계산할 수 있다. GOMS 방법은 다른 사용성 평가 방법에 비해 사용성을 향상시킬 수 있는 7개의 가이드라인을 제공하며, 설계의 정성적 평가, 원시연산자를 이용한 실행시간 예측, 사용자 학습 시간 계산이 가능하다. 예를 들어 NGOMSL 문장으로 표현할 때 <표 5>와 같은 공식을 이용하여 실행시간을 예측할 수 있다.

3.3.2 휴리스틱 평가 방법 활용

휴리스틱 평가는 주로 4~5명 정도의 소수의 평가 전문가가 사용성 평가 대상 시스템(또는 시제품)에 대한 사용평가를 수행하여 그 인터페이스가 사용성 원칙에 따르는지 등의 문제점을 도출한 후, 모여서 수합하는 방식으로 진행된다. 일반적으로 평가자가 많을수록 더 많은 사용성 문제를 탐지하지만 그 부가적 효율성은 5명 이상이 되면 급감한다.

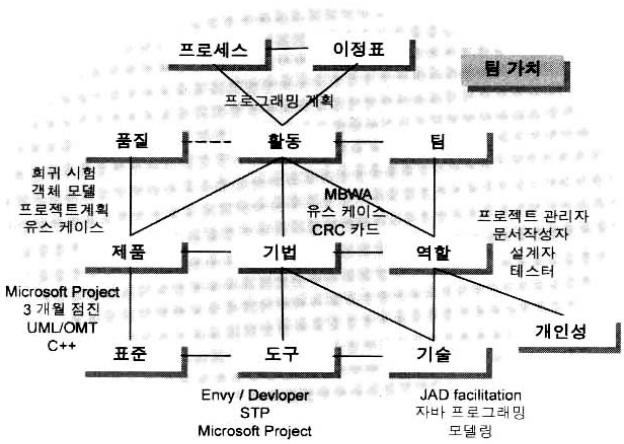
휴리스틱 평가는 UI 전문가들이 UI의 각 요소가 확립된 사용성 원칙을 따르고 있는지 판단할 수 있는 사용성 검증법 중 하나이다. 각 프로젝트에 맞게 구성된 전문가 집단은 휴리스틱 평가 항목에 의해 가시적인 평가를 하고 이것을 수치적으로 나타날 수 있다. 전문가의 검증이 필요한 기초 분석단계 또는 프로젝트 수행 마지막 단계에 효과적이다. 휴리스틱 평가 방법을 제안한 Nielson의 10개 평가 항목은 참고문헌 [28]에 제시되어 있다.

3.3.3 질문지 이용 평가 방법 활용

사람들이 가지고 있는 일반적인 생각과 그 변화를 이해하는데 효율적이어서 시스템이나 서비스에 대한 사용자들의 만족도, 불편사항, 건의사항 등을 파악할 때 가장 많이 사용되는 방법이다. 질문조사의 결과는 중요한 요인 항목을 찾아내고, 그 항목들간의 상관관계를 밝히는 것 이상의 세부적인 사용성 정보를 주지 못하는 문제점도 있다. 하지만 개발에 참여하는 사용자에게 사용하기에 편리한가라는 개괄적이고 단편적인 질의 응답보다는, 척도가 있어야 상태를 알 수 있는 품질 평가의 기본 특성을 구체적인 질문으로 정량적으로 사용성을 평가할 수 있기에 질문지를 이용하고자 한다. 특히, 개발과정의 요구분석 단계와 자격시험, 고품질 레벨이 요구되는 평가에 우수한 방법이다.

3.4 극한 프로그래밍에서 개발 생명주기 지원 강화

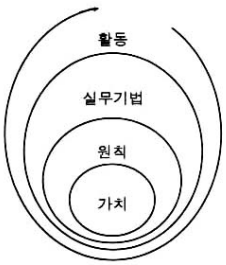
소프트웨어 개발 방법론은 (그림 3)과 같이 13개 요소로 구성된다. 이들은 역할, 기술, 팀, 기법, 활동, 프로세스, 작업 산출물, 이정표, 표준, 품질, 팀 가치, 개인성, 툴이다[11]. XP 프로세스가 갖는 문제점을 찾기 위해 XP 프로세스를 구성하는 가치, 원칙, 실무기법, 활동을 분석하고, 이를 해결하기 위해 활동과 활동의 세부 태스크, 산출물을 개선하고자 한다. 방법론의 유형은 규범적인(Normative) 방법론, 이성적인(Rational) 방법론, 참여적인(Participative) 방법론, 휴리스틱(Heuristic) 방법론 범주가 있다. 지식이 증가됨에 따라 방법론 영역은 휴리스틱 방법론에서 규범적인 방법론으로 옮겨지고, 표준 문제에 대한 표준 해결책으로 체계화되어 진다. XP를 포함한 대부분의 소프트웨어 개발은 휴리스틱 방법론의 적당한 수준에 있다.



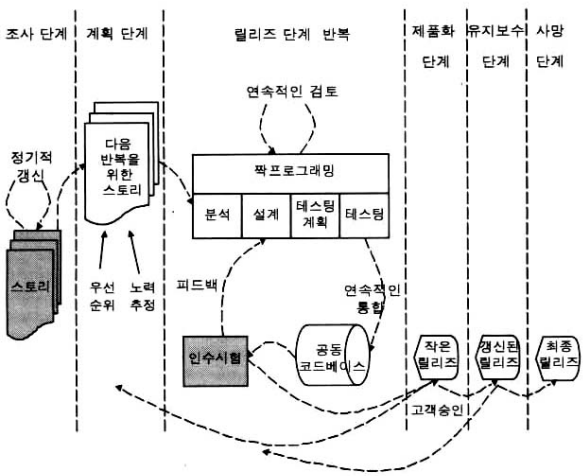
(그림 3) 방법론 구성요소

XP는 (그림 4)와 같이 4가지 핵심 가치에 기반하고 있으며 5가지 원칙, 12가지 실무기법, 활동에 의해 지탱된다. 활동이 전체의 사이클을 포괄하고 있다. XP가 소프트웨어 개발 생명 주기의 어느 부분을 지원하고 있는지는 XP의 가치, 원칙, 실무기법, 활동에 의해 파악될 수 있다. XP의 소프트웨어 개발 생명 주기 지원 현황을 보면 개념 생성, 인수시험, 동작중 시스템 생명주기에 대한 지원이 안되고 있으며, 프로젝트 관리는 전체 생명주기에서 모두 지원되지 않고 있다[1, 2].

3.3절의 사용성 평가 방법의 도입을 (그림 5)과 같이 XP 프로세서와 연계할 수 있다. 본 논문에서 제안하고 있는 사용성 향상 방안은 조사 단계의 스토리 작성 활동과 릴리즈 단계 반복에서 인수 시험 활동과 설계 활동에 적용된다.



(그림 4) XP 핵심 가치, 원칙, 실무기법, 활동 포함관계



(그림 5) XP 프로세스

사용자 스토리는 사용자의 태스크를 카드와 같은 적은 분량의 문서로 작성하고 있는데, 사용성에 대한 요구 사항을 사용자 스토리에 추가하여 넣도록 제안한다. 사용자 스토리를 바탕으로 테스트 시나리오와 테스트 케이스를 만들기 때문에, 이때 목표로 하는 사용성을 명시하도록 한다. 예를 들면, 특정 태스크의 GOMS 실행 예측시간이 5.5sec 이하여야 한다든지, 명령을 내릴 때 잘못할 가능성을 5% 미만으로 한다든지, 필드 리스트의 식별 용이성이 10점 척도에서 8점 이상이 되도록 한다는 등이다. 스토리 카드의 구조는 <표 6>과 같이 사용하도록 제안한다.

〈표 6〉 제안된 사용자 스토리카드 형식

프로젝트 이름 :

사용자 스토리 번호 :
간단한 설명 :
개발자 :
개발 시작일 :
개발 완료일 :

태 스 크	설 명	추 정 치

사용성 세부 항목	설 명	목 표 치

코멘트 :

날 짜	코 멘 트	개 발 자

코드 완성 여부 :
통합 여부 :
소요 시간 :

반복 개발의 설계 활동에서는 GOMS 모델을 이용하여 사용성 평가하도록 한다. XP에서는 매분 매시간마다 단위 시험을 할 수 있기 때문에 전문가에 의한 GOMS평가가 과도한 오버헤드가 될 수 있기 때문에, 통합 테스트에 들어가기 전에 테스팅 하도록 한다.

XP의 인수시험에서는 기능시험 위주로 되어있다. 사용자가 해당 시스템을 받아들일 것인지를 결정하는 시험에서, 사용자에 의한 사용성 평가 방법이 유용할 것이다. 이때 질문지를 이용한 평가 방법을 사용하도록 한다.

3.5 제안 방안의 타당성

본 논문에서 제안한 극한 프로그래밍의 사용성 향상 방안에 다음 4가지 방법을 적용하여 **연구의 타당성**을 보인다.

- 소프트웨어 개발 생명주기 지원 여부(3.5절)
- CMM 핵심 프로세스 영역 만족도 분석(3.5절)
- CMM 규정 실무 만족도 향상(3.5절)
- 적용 사례 분석(4장)

XP의 소프트웨어 개발 생명 주기 지원 현황을 보면 <표 7>과 같이 개념 생성, 인수시험, 동작중 시스템 생명주기에 대한 지원이 안되고 있으며, 프로젝트 관리는 전체 생명주기에서 모두 지원되지 않고 있다[1, 2]. 본 논문에서는 인수시험 활동을 위한 프로세스를 지원하고, 실무/활동/산출물 정의를 시도하였다.

〈표 7〉 XP의 소프트웨어 개발 생명 주기 지원

생명주기 / 지원 유형	개념 생성	요구 사항 명세	설계	코딩	단위 시험	통합 시험	시스템 시험	인수 시험	동작중 시스템
프로젝트 관리 유무	×	×	×	×	×	×	×	×	×
프로세스 지원 유무	×	○	○	○	○	○	○	×	×
실무/활동/산출물 정의 유무	×	○	○	○	○	○	○	×	×

XP의 프로세스가 CMM 핵심 프로세스 영역을 만족하는 정도는 <표 8>과 같다[30]. <표 8>에서 레벨 2의 하청 관리는 XP가 중소 소프트웨어 개발 조직에서 사용되기 때문에 쓸모가 없고, 레벨3의 프로그램 훈련은 XP 자체의 범위 밖이다. **본 논문에서 제시한 프로세스를 통하여 레벨 4의 정량적 프로세스 관리와 소프트웨어 품질 관리 영역을 보완 할 수 있다.**

<표 9>는 XP가 핵심 프로세스 영역에 대하여 규정 실무를 만족하는 정도를 보여주고 있다. **본 논문에서 제시한 프로세스를 통하여 규정 실무의 소프트웨어 품질 보증이라는 실무를 보완할 수 있다.**

〈표 8〉 XP의 CMM 핵심 프로세스 영역 만족도

레벨	핵심 프로세스 영역	만족도
2	요구사항 관리	++
2	소프트웨어 프로젝트 계획	++
2	소프트웨어 프로젝트 추적 및 감독	++
2	소프트웨어 하청 관리	−−
2	소프트웨어 품질 보증	+
2	소프트웨어 형상 관리	+
3	조직 프로세스 촛점	+
3	조직 프로세스 정의	+
3	프로그램 훈련	−−
3	통합 소프트웨어 관리	−−
3	소프트웨어 프로덕트 공학	++
3	조직간 조정	++
3	동료에 의한 검토	++
4	**정량적 프로세스 관리**	−−
4	**소프트웨어 품질 관리**	−−
5	결함 예방	+
5	기술 변경 관리	−−
5	프로세스 변경 관리	−−

+ : XP에서 부분적 언급
++ : XP에서 대체적으로 언급
−− : XP에서 언급되지 않음

〈표 9〉 XP와 규정 실무

공통 특징(각각의 KPA)	실 무	만족도
수행에 대한 일치	정책 리더쉽과 스폰서쉽	−− −−
수행에 대한 능력	조직적 구조 자원과 자금 훈련	+ + +
측정과 분석	측정	+
구현 검증	선임 관리자 감독 프로젝트 관리 감독 **소프트웨어 품질 보증**	−− ++ +

+ : XP에서 부분적 언급
++ : XP에서 대체적으로 언급
−− : XP에서 언급되지 않음

제안한 방안의 타당성을 <표 10>과 같은 여러 가지 유형으로 증명할 수 있다[32]. 소프트웨어 공학 분야에서 국제적으로 가장 명망있는 학술대회인 ICSE(International Conference on Software Engineering) 2002년 발표 논문중에서 예제를 이용하여 타당성을 제시한 방법이 가장 많은 비중을 차지했다. 본 논문에서도 예제를 이용하며, 제4장 적용 사례를 통해 제안 방안의 타당성을 정량적으로 평가하고자 한다.

〈표 10〉 ICSE2002 게재 논문의 타당성 제시 스타일

타당성 유형	타당성 제시 예제/스타일	ICSE2002 게재율
분석	결과를 분석하고, 분석별과로부터 만족하다는 것 발견 • 정형모델 : 유도 및 증명 • 경험모델 : 통제 환경에서 사용상 자료 통제된 실험 : 통계적으로 유의한 결과로 실험 설계	26%
계산	결과가 • 설명 모델 : 적절하게 현상 설명 • 품질 모델 : 횟수 • 경험 모델 : 예측 가능 타당성 연구, 파일럿 프로젝트 포함	2%
실험	다른사람에 의해 수행된 실제 예제에 사용하여, 정확성/유용성/효율성이 • 품질 모델 : 이야기체 • 경험 모델 혹은 도구 : 데이터, 통계적, 실무 • 표기법 혹은 기법 : 실제 사용하는 시스템 비교	19%
예제	어떻게 일하는지 예제 • 기법 혹은 절차 : 실제 시스템에 기반한 "slice of life" 예제 • 기법 혹은 절차 : 내가 개발한 시스템 • 기법 혹은 절차 : 현실에서 추진하게된 토이 예제 "slice of life" 예제 : 간단한 예제가 해결하고자 하는 문제의 필수사항을 가지고 있음에 대한 설명이 있어야 한다	37%
설득	나는 타당성 증명이 어렵다고 생각하고, 다음을 열렬히 믿고 있다. • 기법 : 다음 방법을 수행하면… • 시스템 : 이와 같은 시스템이 구축되면… • 모델 : 이 예제는 내 아이디어가 어떻게 동작되는지 보여준다. 연구 논문으로 불충분하다.	2%
주제넘은 주장	결과를 평가하는데 중요한 시도가 없다. 통과되기 매우 어렵다.	0%

참고) 부족한 6%는 쉽게 알아볼 수 없는 상태였음

4. 적용사례

인터넷 쇼핑몰은 많은 사용자들이 접속하는 인터넷 사이트로서 쇼핑몰 상품의 변화, 소비자들의 기호 변화에 따라 짧은 시간에 민첩하게 변화되어야 하기 때문에, 극한 프로그래밍 방법론을 적용하는 것이 좋다.또한, 사용자와 상호작용이 많아서 소프트웨어 사용성이 강조된다.

4.1 녹차 쇼핑몰

본 논문에서는 (그림 6)과 같은 녹차 쇼핑몰의 구축과정에 제안한 극한 프로그래밍의 사용성 향상 방안을 적용하였다. 일반적인 인터넷 쇼핑몰과 같이 회원 가입, 상품 검색, 주문 기능 등을 갖고 있다.

(그림 6) 녹차 쇼핑몰

4.2 스토리 카드 작성

본문 3.4절 <표 6>에서 제안한 사용자 스토리 카드를 녹차 쇼핑몰에서는 <표 11>과 같이 정의하였다.

〈표 11〉 녹차 쇼핑몰에 적용된 사용자 스토리카드

프로젝트 이름 : greentea 쇼핑몰

사용자 스토리 번호 : 쇼핑-1
간단한 설명 : 홈페이지 방문객이 짧은 시간내에 원하는 물건을 주문할 수 있도록 한다
개발자 : 김남석, 한휴
개발 시작일 : 2003.06.01
개발 완료일 : 2003.06.30

태 스 크	설 명	추 정 치
회원 가입	쇼핑몰 회원 관리	
녹차 주문	녹차 주문 판매	
장바구니 보관	회원들의 재방문 지원	

사용성 세부 항목	설 명	목 표 치
녹차 주문 GOMS 실행시간 예측	녹차 주문 목표 달성을 위한 예측된 실행시간	420초 이하
녹차 주문 GOMS 순수 학습시간	녹차 주문 순수 학습시간	17분 이하
휴리스틱 사용성	휴리스틱 평가 점수	85점 이상
질문지 사용성	질문지 평가 점수	85점 이상

코멘트 :

날 짜	코 멘 트	개 발 자
2003.07.01		

코드 완성 여부 :
통합 여부 :
소요 시간 :

4.3 사용성 평가 방법 적용

4.3.1 GOMS 방법

GOMS 방법에서는 목표, 연산자, 메소드, 선택 규칙을 파악하고, 실행시간이나 인지 부하를 예측할 수 있다. XP 프로세스의 설계 활동이 끝나고, 통합 시험 이전에 사용한다.

〈표 12〉 녹차 쇼핑몰에서 GOMS 연산자 리스트

구 분	종 류	내 용	시간(sec)
모델러 정의 연산자	Wait for System Response	사용중인 시스템의 반응시간	횟수
정신 연산자	Make Up Your Mind(MUYM)	물품 선정	-
기억 연산자	Retrieve Long-term Memory	장기 기억 단위의 상기	-
	Forget Long-term Memory	장기 기억 단위의 망각	-
	Recall Working Memory	단기 기억 단위의 회상	-
	Retain Working Memory	단기 기억 단위의 망각	-
외부 연산자	Find Location	모니터 상에서 위치 찾기	1.20
	Move Cursor	커서 이동	1.10
	Click Mouse	마우스 클릭	0.20
	Type Keyboard	키보드 타이핑	0.28
	Home-hand-to mouse	마우스로 손 이동	0.40
	Home-hand-to keyboard	키보드로 손 이동	0.40
제어 연산자 흐름	Accomplish Goal(AG)	실행순서의 제어를 해당 목표로 이동	-
	Return with Goal Accomplished(RGA)	해당 메소드 완료후 메소드를 호출한 상위 목표로 이동	-

GOMS에서 어려운 일은 연산자를 파악하는 일이다. <표 12>에서 녹차쇼핑몰에 필요한 연산자 리스트를 나열하였다.

녹차쇼핑몰의 녹차구매에 관한 메소드는 다음과 같다. 메소드를 구성하는 연산자들에 대한 수행 시간을 <표 10>을 기준으로 더하면 녹차구매에 소요되는 수행 시간을 예측할 수 있다.

```
Method for goal : 녹차 구매.
        Step 1. Accomplish goal : 녹차 검색.
        Step 2. Make up your mind(MUYM) 상품 종류, 갯수
        Step 3. Accomplish goal : 녹차 주문.
        Step 4. Return with goal accomplished(RGA)
Selection rule set fo : 녹차 검색.
        If 녹차 종류별 특성을 모름, then Accomplish goal : 녹차
          특성 이해
        If 녹차 종류별 특성을 앎, then Accomplish goal : 녹차
          상품 읽기
        RGA
Method for goal : 녹차 특성 이해.
        Step 1. Find Location 녹차 소개관련 메뉴
        Step 2. Click Mouse 메뉴중 녹차 소개 메뉴
        Step 3. Accomplish goal : 녹차 특성 읽기
        Step 4. RGA
Method for goal : 녹차 특성 읽기
        Step 1. If 웹문서가 화면1개 분량보다 많을 경우, then
                Click Mouse Wheel button
        Step 2. RGA
Method for goal : 녹차 상품 읽기
        Step 1. If 다른 페이지에 있는 상품 정보 선택할 경우,
                then Click Mouse 해당페이지
        Step 2. If 녹차상품이 화면1개 분량보다 많을 경우, then
                Click Mouse Wheel button
        Step 3. If 녹차상품 읽기가 끝나지 않은 경우, then Goto 1
        Step 4. RGA
Method for goal : 녹차 주문.
        Step 1. Recall Working Memory 주문할 녹차
        Step 2. Click Mouse 상품이미지(상품명)
        Step 3. Click Mouse 수량
        Step 4. Click Mouse 장바구니 담기 버튼
        Step 5. If 쇼핑 계속, then Click button 쇼핑 계속, Goto 1
        Step 6. Accomplish 주문자-수신자 정보 입력
        Step 7. Accomplish 결재수단 입력
        Step 8. RGA
Selection rule set for : 주문자-수신자 정보 입력.
        If 비회원 주문, then Accomplish goal : 비회원 주문
        If 회원 주문, then Accomplish goal : 회원 주문
RGA
Method for goal : 비회원 주문.
        Step 1. Input 이름
        Step 2. Input 연락처
        Step 3. Input 휴대폰
        Step 4. Input 전자우편
        Step 5. Accomplish 수신자 정보 입력
        Step 6. RGA
Method for goal : 회원 주문.
        Step 1. Verify 이름
        Step 2. Verify 연락처
        Step 3. Verify 휴대폰
        Step 4. Verify 전자우편
        Step 5. Accomplish 수신자 정보 입력
        Step 6. RGA
Method for goal : 수신자 정보 입력
        Step 1. Input 이름
        Step 2. Input 연락처
        Step 3. Input 휴대폰
        Step 4. Input 우편번호
        Step 5. Input 상세주소
        Step 6. Input 요청사항
        Step 7. RGA
...
Method for goal : Input 우편번호
        Step 1. Find Location
        Step 2. Move Cursor to 우편번호
        Step 3. Wait for System Response 우편번호 자동 검색
        Step 4. Home-hand-to Keyboard
        Step 5. Type Keyboard 동 이름
        Step 6. Move Cursor to 동 선택
        Step 7. Home-hand-to mouse
        Step 8. RGA
Method for goal : Input 상세주소
        Step 1. Home-hand-to Keyboard
        Step 2. Retrieve Long-term Memory
        Step 3. Type keyboard 45 character
        Step 4. Forget working memory
        Step 5. Home-hand-to mouse
        Step 6. RGA
...
Selection rule set for : 결재수단 입력.
        If 온라인 입금, then Accomplish goal : 온라인 입금
        If 카드 결재, then Accomplish goal : 카드 결재
        RGA
...
Method for goal : 카드 결재
        Step 1. Input 카드 선택
        Step 2. Input 카드 번호
        Step 3. Input 유효기간
        Step 4. Input 주민등록번호 뒷자리(7)
        Step 5. Input 비밀번호 앞자리(2)
        Step 6. RGA
...
```

4.3.2 휴리스틱 평가

Neilson의 평가 기준을 수용하여[28], 아래와 같은 항목을 XP의 통합테스팅 이전에 전문가에게 검토하도록 하였다.

설문지 : 녹차 쇼핑몰의 평가를 아래 10개의 문항에 답해주세요. 각 문항의 점수는 10점 만점이고, 그 문제점과 이유를 답해주세요.

[문항 1] 시스템 상태의 가시성 : 녹차 쇼핑몰을 네비게이션하면서 현재 문서 위치와, 조작한 행동에 대한 피드백을 잘 표시해주지 못한 점이 있나요? 즉, 문서페이지의 이름, 주소, 상태표시란에 페이지 이동의 결과를 보여주거나, 마우스 동작의 지표, 링크 표시의 확실성 등 현재의 홈페이지의 전후좌우 상태에 문제가 있으면 그 문제점을 구체적으로 기술하세요.

[문항 2] 시스템과 실세계의 대응 : 녹차 쇼핑몰에 사용된 용어, 단어, 그림, 정보등이 부자연스럽고 이해하기 어려운 점이 있나요? 형상관리 시스템 목적이나 방문자에 맞지 않는 그림디자인, 아이콘, 메뉴 방식, 문서내용들이 있었나요?

[문항 3] 사용자 통제와 자유 : 원하는 기능을 선택하기 곤란했거나, 잘못 선택했을 때 뒤로가기를 하기 불편한 점이 있었나요? 백(back), 홈(home)버튼 등이 필요한 곳에 있나요?

[문항 4] 일관성과 표준 : 녹차 쇼핑몰의 디자인이 일반적 관행과 표준을 따르지 않고 있는 점이 있나요? 문서의 배경, 구조, 아이콘, 메뉴 등이 일관적으로 구성되지 못한 점이 있나요? 전체적으로 모든 문서들이 통일된 느낌을 주고 있나요?

[문항 5] 오류방지 : 당신이 실수를 범하기 쉬웠나요? 실수하지 않도록 하는 디자인의 노력이 보이나요? 양식을 잘못 입력하지 않도록 하는 기능이 들어있나요? 마우스를 잘못 누르지 않도록 세심하게 디자인 되어있나요?

[문항 6] 회상대신 재인식 : 특정 작업으로 가려고 할 때 메뉴 항목들을 일일이 기억해 내어야 했나요? 메뉴들을 시각적으로 계속 볼 수 있었나요? 링크들이 무엇을 의미하는지 잘 나타내지 못하는 점이 있나요?

[문항 7] 사용의 유연성과 효율성 : 당신이 원하는 방식으로 녹차 쇼핑몰을 네비게이션 하기가 어려웠나요? 북마크나, 저장, 인쇄 등을 하기 어려운 점이 있나요? 초보자나 숙련자 모두 각기 원하는 방식으로 사용하기 어렵게 디자인 된 점이 있나요? 단축키 등을 사용하기 어려운가요?

[문항 8] 심미적이고 최소화된 디자인 : 녹차 쇼핑몰이 부분적으로 지저분한 점이 없나요 필요하지 않거나 덜 중요한 메뉴나 링크가 많아서 혼란스러운 점은 없나요? 중요하고 새로운 정보를 효과적으로 배열하지 못한 점이 없나요? 문서를 일일이 읽지 않고 대충 보기만 해도 중요한 내용을 금방 찾아낼 수 있고, 내용의 주안점을 금방 파악하게 하는 주제어, 제목 등이 명시되어 있나요?

[문항 9] 에러진단과 회복 : 사용상의 에러가 났을 때 문제점을 정확하게 지적하지 못하거나, 해결책을 잘 제시하지 못한점이 있나요?

[문항 10] 도움말과 지침서 : 사용법 도움말, 사이트 맵(문서지도), 검색엔진 등이 제공되지 않았나요? 특정 기능을 사용하기 위한 도움말이 제공되어 있나요? 이러한 도움말이 구체적 단계로 기술되지 못한 점이 있나요? 지침서나 도움말이 지나치게 길거나 장황한 점은 없나요?

4.3.3 질문지 평가(다시 작성요망, 그룹등)

XP의 인수 시험에서 사용자에 의해 평가 되도록 한다. 아래는 녹차 쇼핑몰에 적용 가능한 질문지 형식이다.

설문지 : 녹차 쇼핑몰의 평가를 아래 10개의 문항에 답해주세요. 각 문항의 점수는 10점 만점입니다.

[문항 1] 사용하기가 쉬운가?
[문항 2] 사용법을 숙달하기가 쉬운지?
[문항 3] 화면 메뉴의 구성이 논리적인가?
[문항 4] 화면 순서들이 명확한가?
[문항 5] 다음에 나올 화면이 예측 가능한가?
[문항 6] 메뉴, 지시문의 위치가 일관적인가?
[문항 7] 문서의 내용 및 구조가 간결, 명확한가?
[문항 8] 숙달할 때까지 시간이 적게 걸리는가?
[문항 9] 조작법 또는 명령어를 기억하기가 쉬운지?
[문항 10] 과제수행에 걸리는 단계수가 최대한 적은지?
[문항 11] 적절한 도움말을 찾기가 쉬운가?
[문항 12] 에러 메시지가 도움을 주는가?
[문항 13] 실수를 교정하기가 간단한가?
[문항 14] 사용자 숙달수준을 배려해 주는가?
[문항 15] 조작실수나 실패가 없는가?
[문항 16] 화면 되돌아가기가 용이한가?
[문항 17] 그래픽(아이콘, 그림)이 적절한가?
[문항 18] 배경색 또는 배경그림이 적절한가?
[문항 19] 긴 화면 스크롤이 편리한가?
[문항 20] 연결 안 되는 링크가 없는가?
[문항 21] 아이콘이나 그래픽 메뉴가 뚜렷이 식별되는가?
[문항 22] 사이트 맵이 유용한가?
[문항 23] 방명록이 유용한가?
[문항 24] 게시판이 유용한가?
[문항 25] 문서의 현 위치를 알려주는 정보가 유용한가?

4.4 녹차쇼핑몰 사용성 향상 분석

녹차 구매에 대한 GOMS 결과와 쇼핑몰 전체에 대한 사용성 평가 결과를 <표 13>에 정리하였다. 녹차 구매하는데 걸리는 시간과 녹차 구매 하는법을 학습하는데 걸리는 시간을 GOMS 모델로서 예측한 결과와 쇼핑몰 전체에 대한 휴리스틱 및 설문지 평가 결과를 사용성 향상 방법을 적용하기 전과 후로 비교하여 결과를 정리하였다.

〈표 13〉 녹차 쇼핑몰 사용성 향상 분석

	사용성 향상 방안 도입 전	사용성 향상 방안 도입 후	사용성 향상 효과	비 고
녹차 구매 GOMS 실행 시간 예측(초)	475	368	23%	비회원, 신용카드 결재
실험적 사용성 평가(초)	456	350	24%	녹차구매
녹차 구매 GOMS 순수 학습시간(분)	19	15	21%	준비학습시간 (30~60분)
쇼핑몰 휴리스틱 평가	76/100점	90/100점	18%	쇼핑몰 전체
쇼핑몰 질문지 평가	81/100점	94/100점	16%	쇼핑몰 전체

5. 결 론

극한 프로그래밍이 산업 실무현장에 더욱 확산될 것으로 예측되고 있다. 빠르고 쉽게 소프트웨어를 개발하고자 하는 극한 프로그래밍의 원래 취지를 준수하면서, 극한 프로그래밍의 미비점을 개선하기 위해 학문적으로 접근하는 연구가 필요하다. 소프트웨어 개발의 초기부터 개발에 사용자가 참여하도록 하는 극한 프로그래밍에서 품질의 향상을 위해서

는, 사용자 입장에서 느끼는 사용성 향상에 초점을 두고 노력해야 한다.

기존에 발표된 문헌이나 논쟁은 실무자나 컨설턴트의 애자일 방법론의 적용 사례, 적용으로부터 알게된 사실을 중심으로 연구되었다. 소프트웨어 품질, 특히 사용성 향상을 위한 학문적 연구는 본 논문에서 처음 시도되었다. 본 논문에서는 극한 프로그래밍 방법론을 이용하여 개발된 소프트웨어가 사용성이 확보되도록, 극한 프로그래밍 방법론의 개선에 노력하였다.

참 고 문 헌

[1] Abrahamsson, P. and et al., Agile Software Development Methods-Review and Analysis, VTT Publication 478, VTT, 2002.

[2] Abrahamsson, P. and et al., New Directions on Agile Methods : A Comparative Analysis, ICSE '03, IEEE, 2003.

[3] Agile Alliance Web Site : Manifesto for Agile software Development. On-line at : http://agilemanifesto.org/.

[4] Agile Methodologies Survey Results, On-line at : http://www.shinetech.com/agile_survey_results.jsp

[5] Ambler, S., Agile Modeling : Effective Practices for Extreme Programming and Unified Process, John Wiley & Sons, Inc., 2002.

[6] Anderson, J. and et al., Integrating Usability Techniques into Software Development, IEEE Software, Vol.18, No.1, Jan./Feb., 2001.

[7] Aoyama, M., Web-Based Agile software Development, IEEE Software, November/December, 1998.

[8] Beck, K., Embracing Change With Extreme Programming, IEEE Computer, Vol.32, No.10, 1999

[9] Beck, K., Extreme Programming Explained : Embrace Change, Addison-Wesley, 2000.

[10] Becker, S. A. and Mottay, F. E., A Global Perspective on Web Site Usability, IEEE Software, Vol.18, No.1, Jan./Feb., 2001.

[11] Cockburn, A., Agile software Development, Addison-Wesley, 2002.

[12] Cusumano, M. and Yoffie, D. Software Development on Internet Time, IEEE Computer, Octorber, 1999.

[13] Dix, A. J. and et al., Human-Computer Interaction, 2nd Edition, Prentice Hall, 1998.

[14] Ferre X. and et al., Usability Basics for Software Developers, IEEE Software, Vol.18, No.1, Jan./Feb., 2001.

[15] Fowler, M., The New Methodology, On-line at : http://www.martinfowler.com/articles/newMethodology.html.

[16] Giga Information Group Inc. http://www.computerworld.com/softwaretopics/software/appdev/story/0,10801,69182,00.html.

[17] Highsmith, J. A., Adaptive Software Development : A Collaborative Approach to Managing Complex Systems. Dorset House Publishing, 2000.

[18] Hightower, R. and Lesiecki, N., Java Tools for Extreme Programming, Wiley Computer Publishing, 2002.

[19] Hunt, A. and Thomas, D., The Pragmatic Programmer, Addison Wesley, 2000.

[20] ISO 9241-11, Ergonomics requirements for office work with visual display terminals(VDTs) part 11 : Guidance on Usability.

[21] ISO/IEC 9126, Information Technology-Software quality characteristics and metrics, 1998.

[22] ISO/TR 16982, Ergonomics of human-system interaction-Usability methods supporting human-centerd design, 2002.

[23] Jacobson, I., Booch, G., Rumbaugh, J., The Unified Software Development Process, Addison-Wesley, 1999.

[24] Kieras D., A Guide to GOMS Model Usability Evaluation using NGOMSL, anonymous ftp ftp.eecs.umich.edu people/kieras, 1996.

[25] Kruchten, P., Agility with the RUP, Cutter IT Journal, Vol.14, No.12, 2001.

[26] Lewis, W. E., Software Testing and Continuous Quality Improvement, Auerbach, 2000.

[27] Martin R. C., Agile Software Development, Principles, Patterns, and Practices, Prentice Hall, 2002.

[28] Nielsen, J., Usability Engineering. Morgan Kaufmann, 1993. see also http://www.useit.com/papers/heuristic/heuristic_evaluation.html.

[29] Palmer, S. R. and Felsing, J. M., A Practical Guide to Feature-Driven Development, Prentice-Hall, 2002.

[30] Paulk, M. C., Extreme Programming from a CMM Perspective. IEEE Software, Vol.18, No.6, Nov./Dec., 2001.

[31] Schwaber, K. and Beedle, M., Agile Software Development with Scrum, Prentice-Hall, 2002,

[32] Shaw, M., Writing Good Software Engineering Research Papers, ICSE '03, IEEE, 2003.

[33] Stapleton, J., Dynamic Systems Development Method -The Method in Practice, Addison Wesley, 1997.

[34] Trenner L. and Bawa, J., The Politics of Usability, Springer-Verlag, London, 1998.

[35] XP.org Extreme Programming : A gentle introduction. http://www.extremeprogramming.org/Last modified January, 2003.

[36] 권호열, 소프트웨어 개발 프로세스의 연구동향, 정보과학회지, 제20권 제3호, 2002.

이 상 준

e-mail : sjlee@seonam.ac.kr

1991년 전남대학교 전산통계학과(이학사)

1993년 전남대학교 전산통계학과(이학석사)

1999년 전남대학교 전산통계학과(이학박사)

1995년~현재 서남대학교 컴퓨터정보통신학과 조교수

관심분야 : 소프트웨어공학, 정보보호, 컴퓨터교육

배 석 찬

e-mail : scbae@kunsan.ac.kr

1983년 전남대학교 계산통계학과(이학사)

1988년 전남대학교 전산통계학과(이학석사)

1995년 전남대학교 전산통계학과(이학박사)

1983년~1985년 ROTC

1993년~1994년 서남대학교 전산통계학과 학과장

1995년~현재 군산대학교 컴퓨터정보과학과 부교수

관심분야 : 소프트웨어역공학, 트랜잭션관리, 데이터베이스보안, 객체지향시스템