韓國IT서비스學會誌
第7卷 第4號
2008年 12月

사용성 문제의 분류 체계 : 문헌분석 및 새로운 개념적 프레임워크

함 동 한*

Classification Scheme of Usability Problems : Literature Review and New Conceptual Framework

Dong-Han Ham*

■ Abstract ■

It is widely known that usability is a critical quality attribute of IT systems. Many studies have developed various methods for finding out usability problems. Usability professionals have emphasized that usability should be integrated into the development life cycle in order to maximize the usability of systems with minimal cost. To achieve this, it is essential to classify usability problems systematically and connect them into the activities of designing user interfaces and tasks. However, there is a lack of framework or method for these two problems and thus remains a challengeable research issue. As a beginning study, this paper proposes a conceptual framework for addressing the two issues. We firstly summarize usability-related studies so far, including usability factors and evaluation methods. Secondly, we review seven approaches to identifying and classifying usability problems. Based on this review and opinions of usability engineers in real industry as well as the review results, this paper proposes a framework comprising three viewpoints, from which more sound classification scheme of usability problems can be inductively developed.

Keyword : Usability, Usability Problems, Software Quality, IT Quality

논문투고일 : 2008년 07월 20일 논문수정완료일 : 2008년 09월 09일 논문게재확정일 : 2008년 09월 10일

* 영국 Middlesex University, School of Engineering and Information Sciences

1. 서 론

IT 시스템의 중요한 품질 특성의 하나인 사용성(usability)은 지난 20년간 여러 관점에서 많은 연구가 이루어져 왔다. 높은 사용성을 갖는 시스템은 측정 가능한 비즈니스적 이익을 많이 가져다 준다는 것은 널리 알려져 있다[21]. 사용성에 대한 절대적인 정의는 없지만 현재 널리 받아들여지고 있는 사용성의 개념은 '특정 상황에서 사용자로 하여금 쉽게 배우고 이용할 수 있게 하고 사용자에게 매력적인 느낌을 주는 시스템의 능력 내지는 속성'이다[24]. 사용성과 관련된 주요 연구 분야는 사용성 요소(차원), 사용성 평가 방법 및 척도, 사용자 인터페이스 설계 원칙 및 가이드라인, 사용성 문제의 분류, 사용자 중심의 설계 방법론 등이다[25].

사용성을 바라보는 관점에 따라 사용성을 구성하는 세부 요소 혹은 차원이 달라지게 된다. 지금까지 많은 연구들이 사용성 요소를 정의하고 분류하고 그들간의 관련성을 연구해왔다. 예로 ISO/IEC 9241[13]은 사용성이 효과성(effectiveness), 효율성(efficiency), 만족성(satisfaction)의 세 가지 요소로 구성된다고 규정하고 있다 반면에 Nielson [20]의 연구는 사용성을 학습성(learnability), 효율성(efficiency of use), 기억성(memorability), 오류(errors), 만족성(satisfaction)의 다섯 요소로 특성화하고 있다. 사용성 요소는 IT 시스템의 종류 및 특성에 의하여 다르게 정의될 수 있지만 모든 요소는 일반적으로 두 개의 그룹으로 분류가 된다[10]. 객관적 요소(objective factors)는 사용자가 자신의 직무를 얼마나 잘 수행하는가와 밀접한 관련이 있는 반면에 주관적 요소(subjective factors)는 사용자가 실제로 시스템의 사용성을 어떻게 느끼는가에 초점을 둔다[2].

사용성 요소들을 참조해서 사용성을 평가하고 측정하기 위한 여러 방법들이 개발되어 왔다. 사용성 평가 방법은 평가자의 역할 및 실제 사용자의 참여 여부 등에 의해 일반적으로 세 개의 그룹으로 분류된다[27]. 사용성 테스팅(usability testing)은 통상적으로 실험실 환경에서 사용자로 하여금 프로토타입 혹은 구현된 시스템을 실제로 사용해서 일련의 직무를 수행하도록 한다. 이 과정에서 사용자가 얼마나 자신의 직무를 효과적이고 효율적으로 달성하는가를 측정한다. 사용성 테스팅의 대표적인 방법으로 협력에 의한 발견적 학습법(co-discovery learning), 질문-답변 프로토콜(question-asking protocol), 따라 말하기 방법(shadowing method) 등이 있다. 사용성 심의(usability inquiry)는 실제 환경에서 사용자가 시스템을 사용하는 상황을 관찰하고 사용자들이 시스템 사용성에 대해 어떻게 느끼고 어떠한 정보요구를 갖고 있는가를 설문한다. 현장 관찰(field observation), 포커스 그룹(focus groups), 설문지 조사(questionnaire survey) 등이 사용성 심의의 대표적인 방법들이다. 사용성 점검(usability inspection)은 일반적으로 사용성에 관한 특정 이론에 기반해서 분석적으로 사용성에 관련된 특성을 조사한다. 이 과정에서 사용성 체크리스트나 가이드라인이 많이 활용된다. 대표적인 방법으로 인지적 시찰법(cognitive walkthrough), 다원적 시찰법(pluralistic walkthrough), 휴리스틱 평가(heuristic evaluation) 등이 있다. 모든 상황에서 항상 우수한 사용성 평가 방법은 존재하지 않기 때문에 평가자는 사용성 평가의 목적, 획득해야 하는 사용성 척도, 이용가능한 자원 등을 종합적으로 고려해서 상황에 맞게 적절한 평가 방법을 선택할 필요가 있다. 위에서 설명한 세 가지 그룹에 속하는 방법들은 개발 과정에서 사용성 문제를 파악하기 위한 형성평가(formative evaluation)와 사용성을 종합적으로 판단하는 종합적평가(summative evaluation)에 모두 적합하다. Ivory and Hearst [14]는 위의 세 그룹외에 두 개의 그룹을 더 추가해서 사용성 방법을 총 다섯 개의 그룹으로 구분하고 있다. 추가적인 두 개의 그룹은 분석적 모형화(analytical modeling)와 모의실험(simulation)으로 사용자와 사용자 인터페이스 모형을 이용해서

사용성을 예측하는데 초점을 두고 있다. 분석적 모형화의 대표적인 방법으로는 인지적 직무 분석(cognitive task analysis), 직무-환경 분석(task-environment analysis), GOMS(Goals, Operators, Methods, and Selection Rules)분석 등이 있다. 모의실험의 경우 페트리 네트 모형 및 유전자 알고리즘 모형을 활용한 방법이 대표적이다.

다양한 설계 특성(design features)이 사용성에 영향을 미치지만 사용자 인터페이스가 가장 중요한 특성일 것이다. 이러한 이유로 인터페이스 설계자를 지원하기 위한 많은 인터페이스 설계 원칙과 가이드라인이 개발되어 왔다. 설계 원칙은 특정 직무나 사용상황(contexts)에 의존하지 않고 일반적으로 적용되는 상위 수준의 설계 목표라 할 수 있다. 반면에 설계 가이드라인은 직무 특성 및 사용상황에 종속적이면서 설계 원칙을 구현하기 위한 구체적인 설계 규칙이라 할 수 있다[21]. 일관성(consistency)은 설계 원칙의 대표적인 예이고 일관성을 지키기 위한 설계 가이드라인의 한가지 예로 생각할 수 있는 것이 '홈 버튼은 항상 좌측 상단부에 위치시킨다'이다.

사용성 공학(usability engineering)은 시스템 개발 주기의 전 과정에 걸쳐서 사용성을 정량적으로 정의하고 측정하는 일련의 조직화된 공학적 프로세스 및 방법을 일컫는다[20]. 사용성 공학은 사용성 특성이 개발 초기단계부터 명확하게 정의되어야 하고 사용성이 개발 활동의 중심 개념이 되어야 함을 강조한다. 사용성 공학에 관련된 활동들은 모두 중요하고 단일화되고 통일된 프레임워크 하에서 통합될 필요가 있다[6]. 즉 사용성이 시스템의 모든 개발 활동과 유기적으로 연결되어야 하는데 이를 위해 반드시 필요한 것이 사용성 문제의 특성을 파악하고 그 문제들의 원인을 설계 활동과 연결하여 제대로 진단을 하는 것이다[4, 11]. 이를 위해 소프트웨어 공학과 인간-컴퓨터 상호작용(human-computer interaction, HCI) 분야를 중심으로 사용성 문제를 객관적으로 분류하기 위한 방법 및 분류 체계가 개발되어 왔다. 본 논문에서는 사용성 문제의 분류 체계와 관련된 기존의 연구들을 분석한다. 분석 결과 설계 활동과 보다 밀접하게 연결되어 활용될 수 있는 새로운 사용성 문제의 분류 체계를 개발할 필요가 있음을 주장한다. 새로운 분류 체계가 사용성 문제를 종합적인 관점에서 다룰 수 있는 일종의 개념적 프레임워크(conceptual framework) 내지는 모형에 기반해서 이해되고 개발될 필요가 있음을 주장하고 이러한 개념적 프레임워크를 제안한다.

2. 관련 연구

사용성 평가 방법과 사용성 문제 분류 체계를 보다 과학적인 방법으로 연구하기 위해 유럽연합에서는 COST action 294-MAUSE(towards the MAturation of information technology USability Evaluation)라고 하는 컨소시움을 구성하였다(www.cost294.org)[18]. MAUSE는 사용성 문제와 관련해 가장 활발한 연구를 수행하고 있는 단체중의 하나로 이 곳에서 이루어지는 연구에 항상 주목할 필요가 있다. MAUSE는 유럽연합의 소프트웨어 및 IT 시스템 관련 업체의 사용성 전문가들로부터 다양한 의견을 수렴한 후에 사용성 분야에서 향후 집중적인 연구가 이루어질 필요가 있는 연구 주제를 다음과 같이 도출하였다. 이러한 주제들은 기존의 사용성 연구들이 탄탄한 이론적 모형을 갖고 연구되었다기 보다는 실제 발생하는 문제 중심으로 단편일률적으로 연구되어온 사실에 기인한다[6, 10]. 또한 이러한 연구 주제들로부터 발견된 사용성 문제를 정확히 이해하고 다양한 관점에서 그 원인을 진단하는 것이 얼마나 중요한지 알 수 있다.

- 관찰된 사용성 문제 상황을 설명하기 위한 이론적 프레임워크의 부족
- 사용성 문제를 특성화하기 위한 실제적이고 널리 받아들여지는 기준의 부족
- 핵심적인 사용성 테스트 매개변수의 값을 추정하기 위한 표준적 접근법의 부족

- 사용자와 평가자의 영향을 체계적으로 관리하기 위한 효과적인 전략의 부족
- 사용성 문제를 분석하기 위한 종합적 분류 체계의 부족
- 시나리오 기반의 사용성 평가를 위한 직무 선정의 가이드라인의 부족
- 사용성 및 다른 품질 특성(예 : 신뢰성)과의 관련성을 표현하기 위한 정확한 통계 모형의 부족
- 사용성 평가에서의 문화적 요소의 영향력에 대한 이해의 부족

사용성 평가 방법 연구와 사용성 문제 분류 연구는 구별될 필요가 있다. 사용성 평가 방법의 목적은 실제 사용성 문제(usability problem data)를 파악하고 수집하는 것이다. 이와는 별도로 수집된 사용성 문제를 어떻게 해석하고 활용할 것인가의 문제가 존재한다. 이러한 문제를 다루기 위한 목적으로 연구되는 것이 사용성 문제의 분류 체계이다[11]. 사용성 평가 방법과 사용성 문제 분류 체계는 밀접한 관련이 있지만 분명히 다른 활용 목적을 갖고 있음을 주의할 필요가 있다.

본 논문에서는 사용성 문제를 특성화하고 분류하는데 유용하게 활용할 수 있는 아래에 나열된 7개의 분류 체계를 설명한다. 이 7개 이외에 다른 분류 체계도 있지만 여기에서 소개하는 7개의 분류 체계가 현재 가장 많이 활용되고 있는 것이라 할 수 있다[18]. 이 중에서 앞의 4개는 원래 소프트웨어 공학 분야에서 소프트웨어의 결함 문제를 파악하기 위해 연구된 것인데 이 것을 사용성 공학에서 도입해 응용한 것이다. 나머지 3개는 HCI 분야에서 사용자의 인지 모형 및 사용자와 시스템 간의 상호작용 모형을 기반으로 사용성 문제를 연구한 분류 체계다.

- Orthogonal Defect Classification Scheme (ODC)[5]
- Root Cause Analysis(RCA)[19]
- Hewlett Packard Defect Classification Scheme(HP-DCS)[12]
- Classification of Usability Problem Scheme (CUP)[26]
- Usability Problem Taxonomy(UPT)[15]
- User Action Framework(UAF)[1]
- Usability Problems Classification using Cyclic Interaction(CI)[23]

2.1 소프트웨어공학으로부터 응용된 분류 체계

사용성 문제를 분류하고자 하는 주요 목적이 문제의 원인을 제대로 진단하고 이를 바탕으로 설계 프로세스를 향상하는 것이다. 어떤 결함적 문제의 현상을 발견하는 것보다 그것의 원인을 파악하고 이를 설계 활동에 적용하는 과정은 매우 어렵다고 알려져 있다[8]. 특히 이 과정에서 필수적인 것이 문제를 분류하기 위해 필요한 속성이 무엇이고 이 속성들이 어떻게 종합적으로 활용될 것인가를 결정하는 것이다. 이를 위해 소프트웨어공학 분야에서 연구되어온 결함 분류 체계가 유용한 정보를 제공해줄 것으로 판단된다. 이러한 이유로 COST action 294에서도 기존의 소프트웨어 결함 분류 체계가 어떻게 사용성 문제 분류에 활용될 수 있는가를 많이 연구하고 있다. 이러한 관점에서 사용성 문제 분류와 기존의 소프트웨어 결함 분류 체계와의 관련성을 논할 수 있을 것이다.

ODC는 소프트웨어 개발 프로젝트에서 개발자들에게 유용한 피드백을 제공하기 위해 IBM에서 개발된 분류 체계이다[5]. ODC는 기존의 소프트웨어 결함의 통계적 모형이 결함(defect)의 원인 분석과 연결되어 있지 않음을 지적하였다. 이러한 문제의식을 바탕으로 ODC는 발견된 결함과 그 결함이 개발활동에 미치는 영향에 대한 관련성을 원인-결과의 관점에서 파악하려고 노력한다. 즉 ODC는 결함이 갖는 특성을 파악하고 이로부터 개발활동이 얼마나 건전한가를 추론한다. ODC는 결함이 영향을 미친 정도 및 부분에 대한 주관적 의견보다는 결함형태(defect type) 및 트리거(trigger)와

같은 객관적 속성에 기반해서 결함을 분류한다. 결함의 의미를 특성화하기 위해 <표 1>과 같이 8개의 속성이 활용된다.

<표 1>에 설명된 것처럼 8개의 속성은 2개의 프로세스 단계에 의해 구분된다[7]. '진행중 (open)' 단계는 결함이 발견되고 새로운 결함 리포트가 추가적으로 결함 추적 시스템에 의해 적용될 때 실행된다. 반면에 '완료(close)' 단계는 발견된 결함이 올바르게 수정되어 그 결함 리포트가 더이상 결함 추적 시스템에 있지 않을 때 실행된다. 각 속성은 각각 고유의 속성 값들을 지니게 된다. 예로 목표대상(target)은 속성 값으로 요구사항(requirements), 설계(design), 코드(code) 등을 갖는다. 결함형태(defect type)의 경우 할당(assignment), 체킹(checking), 알고리즘(algorithm), 기능(function) 등을 속성 값으로 갖는다. 8개 속성 값들의 조합은 하나의 결함을 여러 관점에서 설명해주는데 ODC는 특히 결함형태와 트리거를 중요한 두 개의 속성으로 간주한다.

RCA는 광통신 네트워크의 일부분인 네트워크 요소의 개발 및 테스팅 과정에서 발견된 결함의 수정 요청에 대한 회고적(retrospective) 분석을 위해 사용된 분류 체계이다[19]. RCA는 5개의 특성을 활용해서 다양한 관점에서 결함을 해석하고 있다. 5개의 특성에는 발견 단계(phase detected), 결함형태(defect type), 실제적 결함장소(real defect location), 결함트리거(defect trigger), 방책분석(barrier analysis)이 포함된다. 발견단계(phase detected)는 시스템 개발 주기상에서 어떤 단계에서 결함이 발견되었는가를 표현한다. RCA에서는 시스템 개발 주기를 10개의 단계(예 : 시스템 정의, 하드웨어 설계, 소프트웨어 설계)로 정의하고 있다. 즉 발견단계는 10개의 특성 값을 갖는다고 해석할 수 있다. 결함형태에서는 결함의 형태를 세 가지 클래스로 구분한다. 세 가지 클래스는 구현(implementation), 인터페이스(interface), 외부(external)로 구분되는데 각 클래스 내에서 일련의 구체적인 결함형태가 정의되고 있다. 예로 구현 클래스는 8개의 구체적 형태를 포함한다(예 : 데이터 설계, 자원 할당, 예외 처리). 실제적 결함장소는 결함이 발견된 장소를 의미하는데 세 가지(문서, 하드웨어, 소프트웨어)의 속성 값을 갖는다. 결함트리거는 결함의 실제적인 근본 원인을 의미하는데 RCA는 하나의 결함의 경우 여러개의 근본적인 원인이 있을 수 있음을 인정한다. 이를 위해 RCA는 완전 독립적이지는 않은 네 가지의 근본적 원인의 클래스를 정의하고 있다. 네 가지의 클래스는 단계(phase-related), 인적요소(human-related0, 프로젝트(project-related), 리뷰(review-related)를 포함한다. 결함형태와 마찬가지로 각 클래스에는 여러 가능한 특성 값들이 존재한다. 예로 인적요소 클래스에 해당하는 결함트리거로 시스템 지식의 부족, 도구 지식의 부족, 개인적 오류 등이 정의된다. 방책분석은 결함의 조기 발견 및 방지를

〈표 1〉 ODC에서 결함의 의미를 설명하는 8가지 속성[7]

속성 이름	설 명	프로세스 단계
활동(activity)	언제 결함을 발견했는가?	진행중(open)
트리거(trigger)	어떻게 결함을 발견했는가?	
영향도(impact)	결함이 문제해결 없이 그대로 방치되었으면 어떤 영향을 미쳤을까?	
목표대상(target)	무슨 상위수준의 대상(entity)이 고쳐졌는가?	완료(close)
근원(source)	누가 목표대상(target)을 개발했는가?	
이력(age)	목표대상(target)의 이력은 어떻게 되는가?	
결함형태(defect type)	무엇이 고쳐져야 하는가?	
결함특성자(defect qualifier)	결함형태가 생략(omission), 실행(commission), 외재성(extraneous)중에서 어떤 것인가?	

보장하기 위한 수단 내지는 방법을 제안한다.

HP-DCS는 휴렛팩커드에서 장시간에 걸쳐 소프트웨어 품질 결함의 수를 최소화하기 위해 개발 프로세스를 향상시키기 위한 목적으로 개발된 분류 체계이다[12]. HP-DCS는 결함을 특성화하기 위해 세 가지의 기술자(descriptor)를 정의한다. 첫 번째 기술자인 유발(origin)은 개발 주기의 단계 중에서 실제로 결함이 발견된 단계가 아니고 제대로 방지했다면 결함이 발견되지 않았을 첫 번째 단계를 의미한다. 두 번째인 형태(type)는 관련된 유발 내에서 해당 결함에 책임이 있는 영역을 의미하다. 세 번째로 모드(mode)는 결함이 왜 어떤 형태로 발생했는가에 대한 해석이다. ODC나 RCA와 비슷하게 HP-DCS에서도 각 기술자는 여러 속성 혹은 속성 그룹으로 구성되며 각 특성 값들의 조합으로 결함을 해석하고 분류한다. 유발(origin) 기술자는 여섯 가지의 속성을 갖는데 이는 HP-DCS에서는 개발주기를 6개의 단계로 해석함을 의미한다. 여섯 가지의 속성은 명세/요구사항, 설계, 코드, 환경/지원, 문서화, 기타활동으로 구분된다. 형태(type) 기술자는 6개의 속성 그룹으로 구성되는데 하나의 예가 로직, 계산(computation), 자료 처리, 모듈 인터페이스 등으로 구성된 그룹이다. 모드(mode) 기술자는 다섯 가지의 속성으로 결함이 발생한 형태적 원인을 규명한다. 정의된 다섯 가지의 모드 속성은 무엇이 빠졌는지(missing), 무엇이 불분명한지(unclear), 무엇이 잘못되었는지(wrong), 무엇이 변경되었는지(changed), 더 나은 방법이 있었는지(better way)로 구성된다. HP-DCS에서 특이한 점은 유발(origin) 기술자에서의 어떤 속성을 선택하는 가에 따라 형태(type) 기술자에서의 속성 그룹의 선택이 영향을 받는다는 점이다. 즉 형태(type) 기술자에서 하나의 속성 그룹은 시스템 개발 주기의 특정 단계에서만 의미가 있는 것으로 해석할 수 있다.

CUP는 ODC, RCA, HP-DCS와 같은 기존의 분류체계를 종합적으로 검토한 후에 사용성 문제에 적용할 수 있도록 제안된 분류 체계이다[26]. RCA나 HP-DCS와 비슷하게 CUP는 사용성 문제를 특성화하기 위해 10개의 속성을 정의하고 있다. 10개의 속성은 식별자(identifier), 기술(description), 결함제거 활동(defect removal activity), 트리거(trigger), 영향도(impact), 예측단계(expected phase), 실패 특성자(failure qualifier), 원인(cause), 실제단계(actual phase), 에러방지(error prevention)를 포함한다. 또한 각 속성은 구체적인 속성값을 가지는데 예로 원인(cause) 속성의 경우 인적 요소, 기술적 요소, 방법론적 요소, 관리적 요소 등으로 구성된다. 각 속성 및 속성 값의 활용은 다른 분류 체계와 비슷하다.

2.2 HCI관점에서 연구된 분류 체계

UPT는 텍스트 컴포넌트를 포함하고 있는 그래픽 사용자 인터페이스에서의 사용성 문제를 설계물(artifact)과 직무의 두 가지 관점에서 구분할 수 있도록 개발된 분류 체계다[15]. UPT는 실제 산업체의 개발 프로젝트에서 수집된 400개의 사용성 문제의 체계적인 검토결과를 바탕으로 개발되었다. UPT는 사용성 문제를 분류하기 위해 [그림 1]과 같이 세 개의 수준을 갖는 계층적 구조를 지닌다. 최상위 수준은 설계물 컴포넌트와 직무 컴포넌트로 구성된다. 설계물 컴포넌트는 사용성의 문제를 개별적인 인터페이스 객체 내지는 설계 요소의 관점에 초점을 둔다. 직무 컴포넌트는 반면에 사용자가 직무를 수행하는 과정에 어떠한 방식으로 사용성 문제가 발생하는 가에 보다 밀접하게 관련된다. 각 컴포넌트는 하위 단계에서 주요 범주를 갖게 되며 각 주요 범주 역시 다시 구체적인 하위 범주로 구체화되어 사용성 문제를 분류하게 된다. UPT를 활용해서 실제로 사용성 문제를 분류할 때 각 문제는 [그림 1]에 나타난 범주에 대하여 어떻게 분류가 되는가에 따라 세 가지의 결과를 낳는다. 즉 사용성 문제가 특정 범주에 완전히 분류되는지(full classification, FC), 부분적으로 분류되는지(partial classification, PC), 분류가 되지

최상위수준	5개 주요 범주	하위범주	
설계물 컴포넌트 (artefact component)	시각화(visualness)	객체(스크린) 배치	
		객체 모양	
		객체 이동	
		정보/결과의 표현	
		비메시지 피드백	
	언어(language)	네이밍/레이블링	
		기타 워딩(wording)	피드백 메시지
			에러 메시지
			기타 시스템 메시지
			스크린상의 텍스트
			사용자 요청 정보/결과
	조작(manipulation)	인지적 측면	시각적 큐(visual cues)
			직접 조작(direct manipulation)
		물리적 측면	
직무 컴포넌트 (task component)	직무매핑(task-mapping)	상호작용	
		네비게이션	
		기능성	
	직무편의화(task-facilitation)	대안성	
		직무/기능 자동화	
		사용자 행위 취소화	
		사용자 직무흐름 유지	

[그림 1] Usability Problem Taxonomy(UPT)의 사용성 문제 분류

않는지 (null classification, NC)를 파악하게 된다. NC의 경우 정의된 하위 범주의 어느 것과도 관련성은 없지만 최상위 수준의 설계물 컴포넌트와 직무 컴포넌트의 의미를 고려했을 때 그 컴포넌트와 관련 있음을 의미한다.

UAF는 원래 사용성 개념, 이슈, 설계 특성, 사용성 문제 등을 조직화하기 위한 상호작용 모형 기반의 구조다[1]. UAF는 사용성 공학에서 널리 활용하기 위해 사용성 점검, 사용성 문제 보고, 사용성 데이터 관리 및 설계 가이드라인의 효과적 사용을 지원하기 위한 통합된 프레임워크를 목표로 개발되었다. 특히 UAF의 주요한 기능 중의 하나는 사용성 문제의 원인을 일관성있게 이해하고 보고할 수 있도록 해주는 것이다. UAF는 사용성 문제를 분류하기 위해 상호작용 사이클 모형을 활용한다. 이 모형은 Norman의 7단계 행위 모형을 수정하고 확장한 것이다. 상호작용 사이클 모형은 사용자가 IT 시스템과 상호작용하는 과정에 대한 모든 것을 포함한다. 즉 이 모형은 사용자의 인지적 행위(cognitive actions), 물리적 행위(physical actions), 인식적 행위(perceptual actions)를 모두 포함한다. 구체적으로 이 모형은 세 개의 행위단계로 구성되는데 구성 요소는 계획(planning; 어떻게 의도한 바를 행위로 옮길 것인가?), 물리적 행위(physical actions; 계획된 행위의 실행), 상황판단(assessment; 시스템의 피드백에 대한 해석)이다. 계획은 상위 수준의 계획(high-level planning)과 하위 수준의 계획(translation)으로 더 구분되기도 한다. UAF는 사용성 문제와 상호작용 사이클 모형의 세 가지 행위 단계와 연관시켜서 사용성 문제의 특성을 파악한다. 예로 '읽기 어려운 에러 메시지'라는 사용성 문제는 상황판단(assessment)과 관련되어 있고 '사용자가 마스터 문서의 구조를 이해할 수 없다'는 사용성 문제는 계획(planning)과 관련되어 있으며 '다른 버튼의 클릭'이라는 사용성 문제는 물리적 행위(physical actions)과 대응된다. 이러한 방식으로 UAF는사용자의 상호작용 사이클에서 어느 부분에서 문제가 발생하

는가를 이해함으로써 보다 사용자 관점에서 사용자 문제를 분석하려는 입장을 취한다.

Ryu and Monk[23]은 UAF의 상호작용 모형과 비슷한 사이클 모형에 기반해서 사용성 문제를 분류하는 방법을 제안하였다. 이 사이클 모형은 사용자와 시스템간의 이루어지는 인식(recognition) 기반의 상호작용을 모형화하고 있다. 이 모형의 세 가지 요소는 사용자의 목적(goal), 사용자의 행위(action), 시스템의 결과/피드백(effect)이다. 각 요소들간의 연결 고리라 할 수 있는 세 가지의 경로(행위-피드백 경로, 피드백-목적 경로, 목적-행위 경로)를 기준으로 사용성 문제를 분석하고 분류한다. 세 가지의 상호작용 경로는 세 가지의 사용성 문제 형태(행위-피드백 문제, 피드백-목적 문제, 목적-행위 문제)와 대응된다. 행위-피드백 문제는 동일한 사용자 행위가 다른 시스템 결과를 초래하는 모드(mode) 문제와 밀접한 관련이 있다. 일반적으로 모드에 관련된 사용성 문제는 다음의 세가지 그룹으로 분류된다 - 숨겨진 모드 문제(hidden mode problems), 부분적으로 숨겨진 모드 문제(partially hidden mode problems), 모드 신호의 오류(misleading mode signals). 피드백-목적 문제는 피드백에 기반한 사용자의 목적 설정과 관련되어 있다. 효과적이지 못하고 부적절한 사용자 목적의 형성은 네 가지 형태의 사용성 문제로부터 발생한다-목적 설정을 위한 큐(cues)의 부재, 목적 설정과 관련 없는 큐(cues)의 강한 존재, 필요없는 목적 제거에 힌트를 주는 큐(cues)의 부재, 필요없는 목적 제거와 관련 없는 큐(cues)의 강한 존재. 목적-행위 문제는 사용자가 계획한 목적를 달성하기 위해 예측하지 못한 행위를 수행해야 하는 상황에서 발생한다. 이와 관련된 사용성 문제는 행위지원성(affordance)과 밀접한 관련이 있고 두 가지 형태로 구분된다 - 올바른 행위를 위한 약한 행위지원성, 올바르지 않은 행위를 위한 강한행위지원성. Ryu and Monk[23]는 사이클 모형에 기반한 이러한 사용성 문제들을 파악하기 위해 사용성 점검(usability inspection)의 한 방법인 시찰법(walkthrough)이 효과적으로 이용될 수 있다고 하였다.

2.3 새로운 분류 체계의 필요성

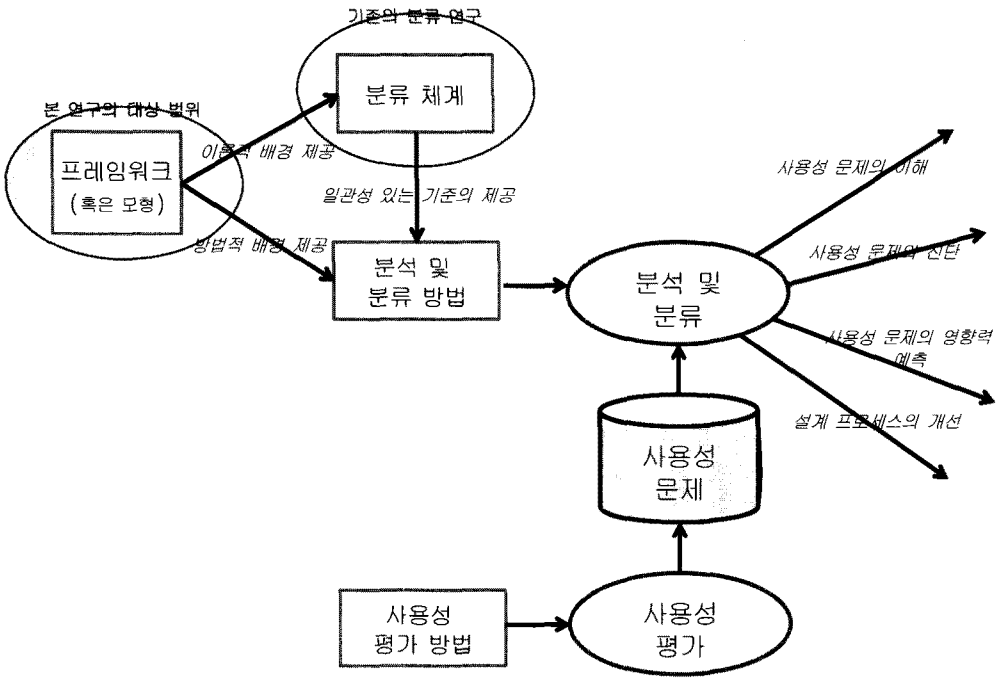
앞에서 소개한 7개의 분류 체계를 8개의 판단 기준을 갖고 <표 2>와 같이 비교할 수 있다. 소프트웨어공학 분야에서 개발된 결함 분류 체계(ODC, RCA, HP-DCS, CUP) 혹은 이를 사용성에 적용한 분류 체계(CUP)는 사용성 문제를 개발과정을 포함한 보다 큰 맥락에서 해석함과 동시에 개발자 관점에서 사용성 문제를 분류하는데 도움이 된다. 이 분류 체계들의 장점은 사용성 문제를 다양한 측면에서 이해하기 위해 여러 속성과 그 속성들이 취할 수 있는 값을 정의하는데 도움을 준다는 점이다. 그러나 왜 그러한 속성이 중요한지 어떠한 기준에 의해 그러한 속성이 도출되었는가에 대한 이론적인 배경이 부족하다는 단점을 갖고 있다. 또한 실제 환경에서 사용자가 경험할 수 있는 사용성 문제에 대해서는 협소한 시각에서의 분석결과만을 도출할 가능성이 크다는 것도 단점으로 지적된다. 그러나 발견된 사용성 문제를 개발 과정 및 개발 활동에 연결시켜 개발 과정의 생산성을 향상시키기 위한 정보를 제공하려는 측면은 가장 큰 장점이라 할 수 있다. 반면에 HCI에서 발전해온 UPT나 UAF 등은 사용자의 상호작용 모형을 활용해서 사용자 관점에서 사용성 문제를 해석하려는 특성을 지닌다. 그래서 사용자가 왜 그러한 사용성 문제를 갖게 되었는가에 대해 보다 상세하게 파악할 수 있다는 장점을 지닌다. 하지만 HCI 분야에서 개발된 분류 체계들은 개발 과정에 연결될 수 있는 방법 내지는 수단에 대해서는 아직까지 좋은 아이디어를 제공하지 못하고 있다. 또한 상호작용 사이클 모형과 같은 HCI 모형에 익숙하지 않은 소프트웨어나 시스템 개발자들 입장에서는 이러한 모형에 기반해서 분석된 사용성 문제를 직접적으로 개발 활동에 활용한다는 것이 쉬운 일이 아니다.

〈표 2〉 기존 분류 체계들의 비교

	ODC	RCA	HP-DCS	CUP	UPT	UAF	CI
사용성 관련 이론적 배경	-	-	-	-	++	++	++
문제 특성화를 위한 다양한 속성 제공	++	++	++	++	-	-	-
사용자 활동 관점에서의 문제 특성화	-	-	-	+	+	++	++
사용성 문제가 발생한 사용상황의 파악	-	-	-	-	+	+	+
설계활동과의 연결고리 제공	-	-	-	-	-	-	-
설계 관점에서의 사용성 문제의 원인 파악	+	+	+	+	+	-	-
사용성 문제의 영향력 평가	+	-	-	+	+	-	+
사용성 문제의 해결방안 제시	-	-	-	+	-	-	-

주) - : 취약, + : 보통, ++ : 좋음.

사용성 문제의 분류가 보다 실제적인 의미를 갖고 설계 활동에 효과적으로 활용되기 위해서는 대별되는 두개의 관점(사용자 및 설계자 관점)을 동시에 고려해야 한다. 또한 사용성 문제의 원인을 체계적으로 진단해서 이를 설계 활동의 개선과 연결시키기 위해서는 사용성 문제와 설계 활동을 종합적으로 바라보는 개념적 프레임워크이 필요하다. 그러나 기존의 분류 체계는 이런 측면에서 일면 부족함을 알 수 있다. [그림 2]에 설명된 바와 같이 사용성 문제의 분류 체계를 보다 객관성 있게 이론적으로 확고하게 개발하기 위해서는 일종의 개념적 프레임워크 내지는 모형이 필요하다. 분류 체계는 그 이면에 어떤 이론 내지는 개념을 갖고 개발되었는가에 따라 크게 달라질 수 있으므



[그림 2] 분류 체계 개발을 위한 연구 방법

로 분류 체계의 개발과 이를 활용하는 방법의 개발은 [그림 2]에 의한 체계대로 이루어질 필요가 있다[9]. 이러한 이유로 프레임워크의 정립은 새로운 사용성 문제 분류 체계를 개발하는데 중요한 의미를 지닌다. 본 연구에서는 상술한 문제 의식에 기반하여 사용성 문제의 분류 체계를 보다 논리적으로 개발하는데 도움이 될 수 있는 개념적 프레임워크를 제안한다.

3. 새로운 분류 체계

3.1 새로운 분류 체계 개발을 위한 요구사항

제 2장에서 설명한 기존의 분류 체계의 고찰 결과와 실제 산업계에 종사하는 HCI 엔지니어들의 의견을 종합해서 새로운 분류체계의 개발을 위해 고려해야 하는 요구사항을 정리하였다. 첫째로 사용성 문제를 분류하려고 하는 목적에 맞게 사용성의 개념과 범위가 명확하게 정의되어야 한다. 전통적으로 사용성 개념은 시스템의 기능적 측면인 유용성(usefulness)을 포함하지 않는다. 그러나 최근에는 유용성뿐만 아니라 사용자의 감성적 느낌까지 심지어 사용성에 포함되고 있는 추세다[25]. 만일 사용성 분류의 주 목적이 설계 과정의 개선이라면 유용성이 포함된 넓은 의미의 사용성 개념을 적용할 필요가 있다. 하지만 그 목적이 단순히 어떤 사용자 인터페이스 요소나 인터페이스 기반의 오퍼레이션의 사용성 테스팅과 관련되어 있다면 전통적 의미의 사용성을 적용하는 것이 더 바람직할 것이다.

두 번째로 사용성 문제는 기본적인 5W1H 질문을 활용해서 다음과 같이 해석하고 특성화시킬 필요가 있다. 이러한 질문을 염두에 두고 사용성 문제를 바라볼 때 다양한 관점에서 사용성 문제를 고려하는 것이 가능해진다. 또한 사용성 문제의 분석 및 분류의 목적이 이러한 질문에 대한 해답을 구하는 과정의 일환이라고 해석할 수 있다.

- 사용성 문제로부터 영향 받는 혹은 문제를 발생시킨 기능은 무엇인가(what)?
- 누가 사용성 문제를 경험하는가(who)?
- 언제 사용성 문제를 경험하게 되는가(when)?
- 사용자가 어떻게 사용성 문제를 경험하게 되는가(how)?
- 사용자 인터페이스의 어떤 설계 요소가 사용성 문제와 밀접한 관련이 있는가(where)?
- 사용성 문제가 왜 발생하였는가(why)?

세 번째로 설계 활동의 결함으로부터 발생되어진 사용성 문제와 특정 사용자 혹은 특정 사용자 그룹에 특화되어 발생하는 사용성 문제는 구분되어야 한다. 특정 사용자에 특화된 사용성 문제의 경우 사용자의 매우 주관적인 측면과 밀접한 관련이 있을 가능성이 있으므로 다른 사용자에게서는 크게 문제가 되지 않을 수 있다. 이러한 문제는 설계 활동과 연결하여 생각할 필요성이 높지 않으므로 이러한 문제는 다른 시각에서 접근할 필요가 있다(이러한 사용성 문제를 다루는 연구주제는 본 논문의 범위 밖이다).

네 번째로 두 번째의 요구사항과 연결해 고려할 때 사용성 문제를 분류할 때 사용할 수 있는 기준으로 6개를 생각해볼 수 있으며 이들의 적절한 조합으로 새로운 사용성 문제 분류 체계를 개발할 수 있다. 6개의 기준은 시스템 기능(what), 사용자(그룹)(who), 사용자 직무(when), 사용자와 시스템의 상호작용 단계(how), 사용자 인터페이스 객체(where), 설계 원칙(partly why)을 포함한다. 예로 워드 프로세스 프로그램의 사용성 문제를 사용자 직무의 관점에서 분류한다면 어떤 문제는 '편집' 직무와 관련이 있을 것이고 어떤 문제는 '문서 포맷' 직무와 관련이 있을 것이다. 만일 인터페이스 객체 기준을 사용한다면 어떤 사용성 문제는 '메뉴'와 관련이 있을 것이고 어떤 문제는 '아이콘'과 관련이 있을 것이다. 이러한 방법으로 6개의 기준은 사용성 문제를 다양하게 분류할 수 있는 유연성을 제공해준다.

다섯 번째로 사용성 문제를 설계 과정 및 설계

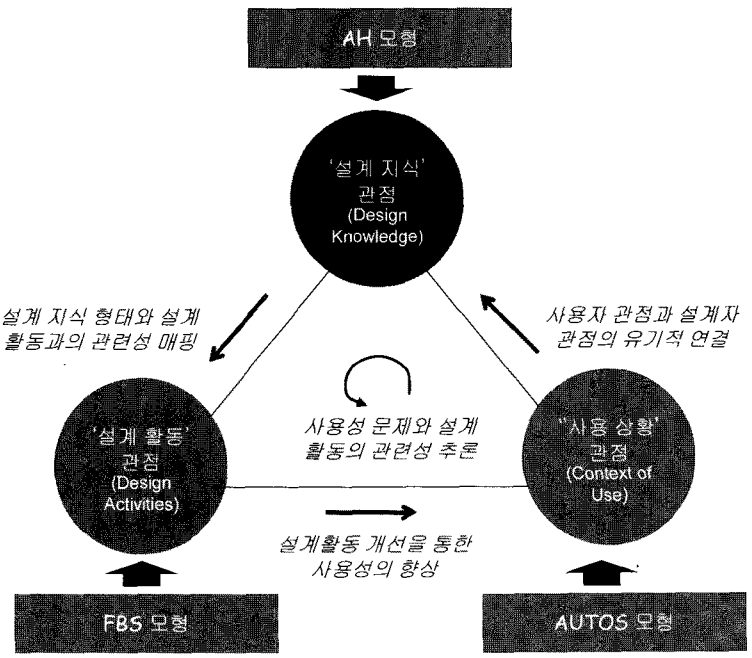
활동과 연결해서 유용하게 활용하기 위해서는 설계 활동과 사용성 문제가 동일한 언어 혹은 모델링 개념을 갖고 설명될 필요가 있다. 뒤에서 설명되겠지만 본 연구에서는 이 요구사항을 위해 설계 지식의 추상화 계층의 사용을 제안한다.

3.2 개념적 프레임워크의 제안

앞에서 정리한 요구사항에 기반해서 기존의 분류 체계를 비교 분석할 수 있고 새로운 분류 체계를 연역적으로 개발하는데 도움이 되는 개념적 배경을 제공하는 프레임워크를 정립하였다 [그림 3]. 제안된 프레임워크는 세 가지 관점으로 구성되며 각 관점에서 사용성 문제를 다루기 위해 효과적으로 활용할 수 있는 모형도 함께 포함하고 있다. 이 프레임워크는 사용성 문제를 사용자의 광범위한 맥락에서 사용자의 인지적(cognitive)관점에 초점을 맞추어 해석할 필요가 있음을 강조한다. 이를 바탕으로 설계자의 시스템 설계 지식과 사용자가 시스템을 바라보는 시스템 지식간의 대응관계를 통해 시스템 설계 활동과의 연관성을 추론할 수 있는 토대를 제공한다. 이러한 대응관계는 절대적일 수 없고 완전하게 설명될 수도 없으나 사용성 엔지니어나 시스템 설계자들이 사용성 문제를 시스템 설계 활동 및 시스템의 설계 지식과 연결해서 종합적으로 생각할 수 있는 이론적 배경을 제공해준다는 점에서 그 의의를 찾을 수 있다. 기존의 분류 체계는 대부분 특정 기준을 정한 후에 사용성 문제를 구분하는데만 중점을 두고 있기 때문에 전체 시스템 개발 활동에서 사용성 문제를 유용하게 활용하기에는 부족한 점이 많았다는 점을 고려할 때 본 논문에서 제안하고 있는 프레임워크는 그러한 단점을 보완하기 위한 노력의 일환으로 해석할 수 있다.

3.2.1 사용상황(context of use) 관점

사용성 문제가 발생했을 때 그 문제가 발생한 상황(context of use)을 올바로 이해하지 못하면

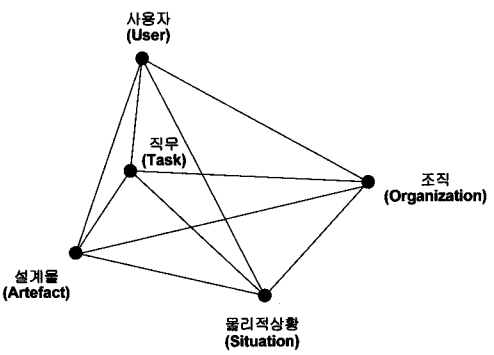


[그림 3] 제안된 개념적 프레임워크

사용성 문제를 제대로 진단하고 활용할 수가 없다[17]. 이러한 이유로 사용상황에 대한 체계적인 분석은 사용성 문제 해석의 가장 기본적인 과정이라 할 수 있다. 사용상황은 일종의 창발현상(emergent features)이기 때문에 완벽하게 예측할 수도 없고 사용상황을 규정하는 요소를 분석적으로 찾아낸다는 것에는 항상 한계가 있게 마련이다. 그럼에도 불구하고 사용상황에 기여한다고 판단되는 중요한 요소와 이들간의 관련성을 파악하는 것이 사용성 문제의 본질에 대해 가장 체계적으로 접근하는 방법이라는 점에는 많은 연구들이 동의하고 있다[18, 25].

사용상황을 분석하기 위해서는 사용자와 IT 시스템간의 발생하는 상호작용에 대한 가정이 필요하다. 어떠한 가정을 하느냐에 따라 고려할 사용상황의 요소의 선택이 영향을 받는다. 사용자와 시스템간의 상호작용의 형태는 사용자의 목적 형성과정과 밀접한 관련이 있다. 이와 관련해 대별되는 두 개의 가정을 생각해볼 수 있다. 하나는 사용자의 목적이 사용자 행위에 대한 시스템의 반응 내지는 피드백에 주로 의존해 형성된다고 하는 가정이다. 앞에서 설명한 UAF[1]나 Ryu and Monk [23]의 사이클 모형이 이러한 가정을 적용했다고 할 수 있다. 그러나 HCI의 최근의 연구동향은 사용자의 목적 형성은 시스템의 결과 외에 다른 요소들에 의해서도 많이 영향을 받는 상황의존적(context-dependent) 성격을 지님과 동시에 사용자 스스로 시스템의 결과와는 관계 없이 목적 형성을 한다는 가정이 더 타당할 수 있음을 보여준다[3]. 이는 사용상황을 단순하게 사용자와 시스템만을 이용해서 해석하기 보다는 보다 큰 맥락에서 이해할 필요가 있음을 의미한다.

이러한 이유로 제안된 프레임워크에서는 설계물(artifact)-사용자(user)-직무(task)-조직(organization)-상황(situation)(AUTOS) 모형[3]을 활용한다. [그림 4]에 보여진 바와 같이 이 모형은 사용상황을 5개의 요소들이 만들어내는 일종의 조합으로 설명하려 한다. 또한 AUTOS 모형을 활용해 사용자와 시스템간의 상호작용을 인지적 정보처리 관점에서 더 구체적으로 이해할 수 있도록 해주는 방법으로 인지적 기능분석(cognitive function analysis)도 개발되어 있다. AUTOS 모형을 사용한 사용상황의 분석은 4개의 기준(직무, 사용자, 인터페이스 객체, 상호작용 단계)을 동시에 고려해서 사용성 문제를 접근해갈 수 있는 수단을 제공해준다. 사용성 문제는 특정 사용자에 의해 발생한 것일 수 있고 모든 사용자에 적용될 수 있는 문제일 수도 있다. 또한 사용성 문제는 시스템의 특성과는 관계 없이 시스템을 사용하는 조직이나 물리적 상황에 영향을 받아서 발생한 것일 수도 있다. 특정 직무만 관련 있는 사용성 문제도 발생할 수 있으며 특정 사용자가 특정 직무만 사용할 상황에서만 발생하는 사용성 문제도 있을 것이다. 이러한 방식으로 사용성 문제는 다각도로 해석할 수 있지만 설계 활동과 사용성 문제의 연결 고리에 보다 중점을 두는 본 논문의 프레임워크에서는 직무 및 설계물(인터페이스 객체 및 기타 설계 요소)의 두 기준에 밀접하게 관련되어 있는 사용성 문제에 보다 많은 관심을 갖게 한다. 직무 및 설계물의 설계 결함으로 발생한 사용성 문제는 뒤에 설명할 설계 지식과 연결되어 설계자 관점과 사용자 관점을 동시에 고려할 수 있는 기회를 제공한다.



[그림 4] AUTOS 모형

3.2.2 설계지식(design knowledge) 관점

앞에서 설명한 바와 같이 제 3.1절에서 사용성

문제 분류의 기준으로 이용할 수 있는 6개의 기준 중에서 4개의 기준은 사용상황 관점에서 다루어진다. 나머지 2개의 기준인 시스템 기능과 설계원칙은 설계지식 관점에서 고려될 수 있다. 또한 이 기준들은 시스템의 설계 지식을 고려했을 때 시스템 설계물과 직무의 기준과 의미적으로 연결이 될 수 있다. 시스템 설계물과 직무의 설계상의 결함은 시스템의 기능 설계 및 설계 원칙의 오류와 관련이 있고 이들은 시스템의 기능, 행위, 구조 등이 어떻게 설계되었는가에 대한 설계지식으로부터 분석이 가능하다. 설계지식의 모형으로 이용가능한 것이 여러 가지가 있을 수 있지만 제안된 프레임워크에서는 시스템의 추상화 계층(abstraction hierarchy; AH) 모형을 활용한다.

추상화 계층은 설계된 시스템의 기능적 지식을 다양한 추상화 수준에서 표현할 수 있도록 해주는 지식표현의 모형으로 인지 시스템공학(cognitive systems engineering) 및 HCI 분야에서 그 유용성이 널리 인정되어 왔다[22]. 시스템의 추상화 수준을 몇 개의 수준으로 정해야 한다는 절대적인 기준은 없으나 보통 5개의 수준이 널리 이용되고 있다. 추상화 계층에서 가장 중요한 특징은 각 추상화 수준들간의 관계가 '목적-수단(goal-means)'으로 정의되고 있다는 점이다. 즉 상위 수준은 어떤 기능(what)이 구현된 이유(why)에 대한 정보를 제공하고 하위수준은 어떤 기능이 어떻게(how)구현되었는가에 대한 정보를 제공한다. 추상화 수준은 보통 공학시스템의 분석에서 널리 이용되는 전체-부분(part-whole)관계에 근거한 물리적 분해 수준과는 개념적으로 구분된다. [그림 5]는 시스템의 추상화 수준과 물리적 분해 수준으로 구성된 설계 지식의 표현을 위한 매트릭스를 보여준다. 이러한 매트릭스를 이용해 시스템의 최상위 기능이 목적-수단 관계를 이용해서 어떻게 가시적인 시스템의 설계물 내지는 인터페이스 객체로 설계되는가를 명확하게 표현하는 것이 가능하다.

추상화 계층을 구성하는 다섯 수준의 의미를 소프트웨어에 특화해서 설명하면 다음과 같다.

- 기능적 목표(FP) : 소프트웨어가 설계된 궁극적 목표

전체-부분 / 목표-수단	전체 시스템 (Total system)	하위시스템 (Subsystem)	기능적유닛 (Functional unit)	하위유닛 (Subassembly)	부품 (Component)
기능적 목표 (Functional purpose)	Why				
추상적 기능 (Abstract function, priority, measure)	Why	What			
일반적 기능 (General function)		What	How		
물리적 기능 (Physical function)			How		
물리적 형태 (Physical form)					

[그림 5] 시스템의 추상화 계층 모형

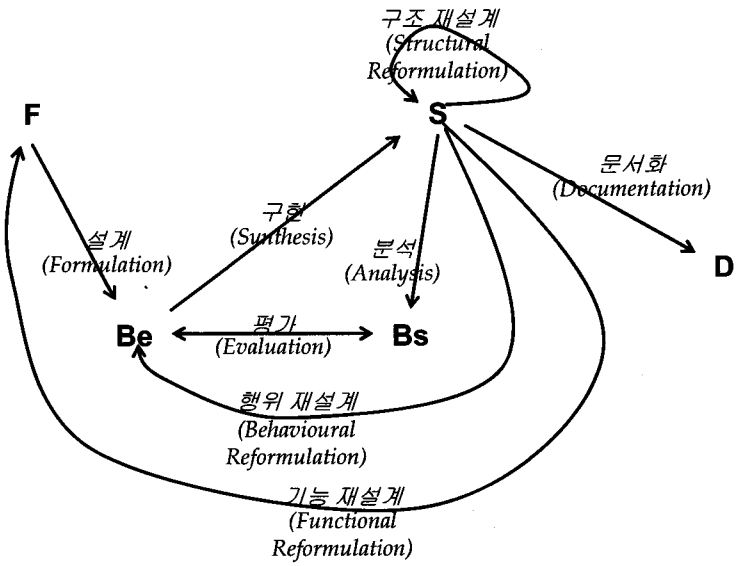
- 추상적 기능(AF) : 정보의 인과적 관계로 설명되는 소프트웨어의 기능 및 일반적 기능의 구현에 관련된 설계 원칙 및 우선순위
- 일반적 기능(GF) : 기능적 목표를 달성하기 위해 구현된 기능으로 도메인에 독립적임
- 물리적 기능(PF) : 일반적 기능을 달성하기 위해 구현된 기능으로 통상 인터페이스 객체의 가시적 기능으로 설명됨
- 물리적 형태(P) : 인터페이스 객체가 가시적으로 보여지는 형태 및 외관

시스템의 기능(function), 행위(behavior), 구조(structure) 관점에서 본다면 추상화 수준의 상위 세개의 수준이 기능과 관련 있다고 할 수 있다. 물리적 기능은 행위와 물리적 형태는 구조와 대응이 된다. 이러한 대응 관계는 뒤에 설명할 설계활동 관점의 연결고리로 작용한다.

사용성 문제를 추상화 수준의 관점에서 어떻게 고려하는지 하나의 단순한 예를 들면 다음과 같다. 워드 프로세스의 '파일 처리' 기능은 일반적 기능 수준에서의 기능이다. '파일 처리'기능을 구현하기 위해 물리적 기능 수준에서의 여러 기능(파일 열기, 파일 닫기, 프린팅, 저장 등)이 필요하다. '파일 열기'와 '저장' 기능에서 사용성 문제가 발생했다면 이는 추상화시켜서 '파일 처리' 기능의 설계가 잘못되었을 가능성이 높다고 추측이 가능하다.

3.2.3 설계활동(design activities) 관점

설계활동 관점은 설계지식 관점과 연계되어 사용성 문제에 기반해서 설계활동 및 설계과정을 향상할 수 있는 가능성을 제공해준다. 설계이론에서는 전통적으로 설계활동은 시스템의 기능을 행위를 거쳐 구조화에 이르는 일련의 변환과정으로 설명하고 있다. 이는 앞에서 설명한 추상화 계층의



F : 기능(Functions);
S : 구조(Structure);
Be : 예측된 행위(Expected Behaviours);
Bs : 실제 행위(Set of Behaviours);
D : 설계 문서(Design Description)

[그림 6] FBS 모형

관점에서 본다면 최상위 수준에서 최하위 수준까지의 변환과정이라고도 할 수 있다. 그러나 이러한 일련의 변환과정이 순차적으로 상위 수준에서 하위 수준으로만 이루어지는 것이 아니고 그 반대의 경우도 가능하다. 이러한 변환과정을 체계적으로 설명한 모형이 기능-행위-구조(Function-Behaviour-Structure; FBS) 모형이다[16].

[그림 6]에 설명된 바와 같이 FBS 모형은 5개의 기본적인 시스템 설계 상태를 정의하고 있으며 이들간의 연결 상태를 8개의 추상적인 공학적 설계 활동(혹은 설계 과정)으로 정의하고 있다. 예로 설계(formulation)는 시스템에서 구현되어야 하는 기능(F)을 시스템의 예측된 행위(Be)로 변환하는 과정으로 생각할 수 있다. FBS 모형과 설계지식 관점에서 설명한 추상화 계층모형과는 완전한 일대일 대응 관계를 정의할 수 없지만 각 추상화 수준의 의미를 고려했을 때 F와 Be 는 상위 3개의 수준에 대응된다고 할 수 있다. 또한 Bs는 물리적 기능 수준에 대응되고 S는 물리적 형태 수준과 대응된다. 따라서 어떤 사용자 문제를 분석했을 때 우선적으로 설계지식 관점에서 어떤 추상화 수준에서 문제가 발생했는지 판단한 후에 FBS 모형을 통해 어떤 설계 활동이 그 사용성 문제에 직접적 관련이 있는가를 추론해볼 수 있다. 이러한 추론 과정을 통해 얻은 사용성 문제와 설계 활동과의 연관 관계는 절대적일 수 없고 그 가능성만을 제공할 수 밖에 없다. 하지만 사용성 문제를 사후적으로 설계 활동에 반영해서 사용성을 향상시키기 위해서는 이러한 분석 및 추론 과정은 필수적이다. 기존의 분류 체계가 이러한 측면에서 부족한 점이 있었다는 점에서 제안된 프레임워크의 의의를 찾을 수 있다.

〈표 3〉 사용성 문제 분류 체계가 지녀야할 최소한의 속성

그 룹	속 성	설 명	필수여부
일반	식별자	문제의 식별번호	필수
	개요	문제에 대한 설명	필수
사용상황	인터페이스객체	문제에 관련된 인터페이스 객체들 (예 : 버튼, 폼, 메뉴)	필수
	사용자그룹	문제에 관련된 사용자 그룹 (예 : 특정 기능 사용 초보자, 특정 문화적배경 그룹)	필수
	시스템기능 혹은 직무	문제에 관련된 시스템 기능 혹은 직무 (예 : 화면이동, 에러방지, 폼 작성)	필수
	조직적 요소	문제에 관련이 있는 조직적 요소 (예 : 커뮤니케이션 관련 사용성 문제)	선택
	상황적 요소	문제에 관련이 있는 상황적 요소 (예 : 시간적 제약이 있는 직무의 수행)	선택
	직무의 세부 인지 단계	사용자 직무의 세부적 인지 단계 (예 : 목표 형성, 계획 수립, 시스템 상태 인식)	선택
설계지식	인터페이스 설계원칙	문제에 관련이 있는 인터페이스 설계 원칙 (예 : 일관성, 가시성, 에러방지성)	선택
	설계지식의 추상화계층	문제에 관련이 있는 것으로 판단되는 추상화계층에서의 수준	선택
	추상화계층에서의 전환	문제에 관련이 있는 추상화계층에서의 전환 (예 : FP→GF; FG→PF)	선택
설계활동	FBS상의 설계 활동	문제에 관련이 있는 것으로 판단되는 FBS모형에서의 설계 활동(예 : 구현, 평가)	선택
	설계 프로세스	문제에 관련이 있는 것으로 판단되는 사용성 관련 설계 프로세스	선택

3.3 프레임워크의 활용의 예

제 3.1절에서 기술한 요구사항과 제안한 프레임워크에 기반해서 사용성문제의 분류 체계가 갖고 있어야 할 최소한의 속성들을 <표 3>과 같이 정리할 수 있다. 일반 두 개의 속성을 제외한 나머지 속성들은 프레임워크에서 제안하고 있는 세 개의 관점별로 그룹핑 될 수 있다. 필수적 속성으로 분류되는 속성들은 사용성 문제를 사용상황 관점에서 특성화하기 위해 필수적으로 파악되어야 한다. 반면에 선택적 속성으로 분류되는 속성들은 사용성 문제의 특성에 따라 파악될 필요성의 여부가 결정된다. 필수적 속성들을 활용해 사용성 문제가 설계 지식의 오류와 연결되어 생각해볼 필요가 있는지 아닌지 판단할 수 있다. 사용성 문제가 설계 활동의 오류보다는 다른 요인에 의한 가능성이 높다고 판단되면 설계 지식이나 설계 활동에 관련된 속성들을 파악하는 것은 무의미하다. 예로 버튼이나 레이블의 색깔이 특정 색맹 그룹이 인식하기에 부적합하다라고 하는 사용성 문제의 경우 설계 활동과 연결해 생각해볼 수도 있지만 그보다는 접근성(accessibility)의 문제로 분리해 생각해보는 것이 바람직하다. 반면에 폰트 사이즈의 크기가 너무 작다라고 하는 문제의 경우는 모든 사용자 그룹에 해당될 수 있는 문제이므로 설계 활동과 연결해 생각해보는 것이 더 바람직할 것이다.

설계 활동과 연결해 생각해볼 필요가 있는 사용성 문제의 경우 그 문제를 세 가지의 속성을 활용해서 설계 지식 관점에서 고려하게 된다. 특히 이를 통해 고려중인 사용성 문제가 추상화 계층간의 전환 과정 중에서 어느 과정과 밀접한 연관관계가 있는지 추측을 하는 과정이 중요하다. 이 과정은 사용성 문제를 설계 활동에 연결하는 매개체 역할을 하게 된다. 예로 폰트 사이즈의 크기가 너무 작

〈표 4〉 FBS과정과 사용성관련 설계 활동의 대응관계

추상화계층에서의 관계	FBS 과정	활 동	사용성 관련 설계 활동
FP → AF → GF	설계 (Formulation)	F → Be	사용자 및 직무 분석(user and task analysis) 사용상황 분석(Analysis of context of use) 사용성 요구사항 명세(usability requirements specificication)
FG → PF → P	구현 (Synthesis)	Be → S	인터페이스 명세(interface specification) 메타포 설계(metaphor design) 다이얼로그 설계(dialogue design) 네비게이션 설계(navigation design) 정보표현 설계(presentation design) 도움말 설계(help design)
P → PF	분석 (Analysis)	S → Bs	테스팅(testing) 검토활동(review activities)
PF ↔ GF	평가 (Evaluation)	Be ↔ Bs	요구사항 검증(requirements validation) 사용성평가(usability evaluation)
GF → P	문서화 (Documentation)	S → D	사용자 메뉴얼 설계(user manual design) 도움말 설계(help design)
P ↔ P	구조 재설계 (Structural Reformulation)	S → S	설계 및 코드의 수정(refinement of design and code) 설계 및 코드의 결함 제거(fixing defects in design and code)
PF ↔ PF	행위 재설계 (Behavioural Reformulation)	S → Be	사용성 요구사항 변경(requirements change)
GF ↔ GF	기능 재설계 (Functional Reformulation)	S → F	사용자의 요구사항 변경(change in needs)

다라는 사용성 문제는 물리적 기능(PF)에서 물리적 형태(P)로의 전환 과정과 관련이 있을 수 있다 (PF → P).

사용성 문제를 설계 지식 관점에서 분석을 한 후에 <표 4>에서 보여주는 대응 관계를 이용해 사용성 문제가 어느 설계 활동에서부터 발생할 가능성이 높은가를 추측해볼 수 있다. <표 4>는 사용성 관련 설계 활동과 FBS 모형에서의 8가지 활동을 대응시킨 결과를 보여준다. 이 관계는 절대로 완전하게 배타적으로 일대일의 대응관계가 될 수 없다. 그러나 사용성 관련 설계 활동의 목적을 고려하고 관련문헌[16, 25]을 참고해 이러한 대응 관계를 정립해볼 수 있다. 폰트 사이즈의 크기 문제를 'PF → P'와 관련 있는 것으로 판단을 내렸으면 이는 FBS 활동 중에서 구현과 대응이 되고 사용성 관련 활동 중에서 정보표현 설계와 가장 연관성이 높다고 추측해볼 수 있다.

4. 결 론

IT 시스템의 기능이 다양해지고 복잡해질수록 품질특성으로서의 사용성의 중요성은 점점 높아질 것이다. 시스템의 사용성을 향상시키기 위해 사용성 문제의 체계적인 분석과 분류는 필수적이다. 사용성 문제의 분류를 위해 지금까지 소프트웨어 공학 및 HCI 분야를 중심으로 분류 체계가 개발되어 왔다. 본 논문에서는 사용성 문제의 분류 체계에 관련 있는 문헌 분석 및 고찰 결과를 요약 정리하였다. 문헌 분석 결과 및 실제 산업계에 종사하는 HCI 연구자들의 의견을 수렴하여 기존의 분류 체계의 단점을 파악하였고 새로운 분류 체계를 개발할 때 고려해야 하는 요구사항을 수립하였다. 사용성 문제의 보다 과학적인 연구 및 새로운 사용성 분류 체계의 개발을 위해서는 사용성 문제와 설계 활동을 의미 있게 통합해서 고려할 수 있는 개념적 프레임워크가 우선적으로 정립되어야 할 필요가 있음을 지적하였다.

이러한 문제 의식을 갖고 본 논문에서는 사용성 문제 분류 체계 개발에 이론적 배경을 제공할 수 있는 프레임워크를 제안하였다. 제안된 프레임워크는 사용상황, 설계지식, 설계활동의 3개의 관점으로 구성되어 있으며 각 관점에서 사용성 문제를 다루기 위해 활용할 수 있는 모형도 함께 포함하고 있다. 상황분석 관점에서는 사용성 문제가 발생한 사용상황의 광범위한 맥락에서의 분석을 통해 설계의 결함으로부터 발생한 사용성 문제와 기타 다른 요소의 영향으로 인해 발생한 사용성 문제를 구분할 필요가 있음을 강조한다. 설계지식 관점에서는 설계의 결함으로부터 발생한 사용성 문제를 시스템 기능적 구조의 추상화 수준의 관점에서 고려할 수 있는 방법을 제공한다. 설계활동 관점에서는 추상화 수준과의 연결고리를 통해 사용성 문제와 직접적 관련이 높을 가능성이 있는 설계활동을 추측할 수 있는 수단을 제공한다.

본 논문의 목적은 사용성 문제의 분류를 위한 새로운 분류 체계의 개발이 아니라 이러한 분류 체계의 개발의 이론적 토대가 되면서 동시에 다양한 사용성 문제를 보다 객관적으로 다루는데 도움이 될 수 있는 프레임워크의 개발이다. 기존의 분류 체계는 특히 사용성 문제와 설계 활동과의 유기적 연결관계에 대한 의미 있는 정보를 많이 제공하지 못하였다. 이러한 측면에서 본 논문에서 제안한 프레임워크의 의의를 발견할 수 있다.

그러나 본 논문에서 제안한 프레임워크는 시스템 개발자나 사용성 엔지니어에게 사용성 문제를 종합적으로 이해하고 분석할 수 있는 생각의 도구로써 활용이 될 수 있는 반면에 분석 및 분류를 위한 어떤 구체적인 방법은 제공하지 않고 있다. 이러한 방법은 추후 더 연구되어야 할 부분이다. 또한 프레임워크에 기반해서 새로운 사용성 분류 체계를 개발하는 것도 중요한 추후 연구가 될 것이다.

참 고 문 헌

[1] Andre, T., H. Hartson, S. Belz and F.

McCreary, "The user action framework : A reliable foundation for usability engineering support tools", *International Journal of Human-Computer Studies*, Vol.54, No.1(2001), pp.107-136.

[2] Bevan, N., "Quality in use : Meeting user needs in quality", *The Journal of Systems and Software*, Vol.49, No.1(1999), pp.89-96.

[3] Boy, G., *Cognitive function analysis*, Ablex Publishing Corporation, Conneticut, 1998.

[4] Card, D., "Learning from our mistakes with defect causal analysis", *IEEE Software*, Vol.15, No.1(1998), pp.56-63.

[5] Chillarge, R., I. Bhandari, J. Chaar, M. Halliday, D. Moebus, B. Ray, and M. Wong, "Orthogonal defect classification-A concept for in-process measurements", *IEEE Transactions on Software Engineering*, Vol.18, No.11(1992), pp.943-956.

[6] Gillan, D. and R. Bias, "Usability science I : Foundations", *International Journal of Human-Computer Interaction*, Vol.13, No.4 (2001), pp.351-372.

[7] Freimut, B., *Developing and using defect classification schemes(IESE-Report No. 072.01/E)*, Fraunhofer IESE, 2001.

[8] Hollnagel, E., "The phenotype of erroneous action", *International Journal of Man-Machine Studies*, Vol.39, No.2(1993), pp.1 -32.

[9] Hollnagel, E., *Cognitive Reliability and Error Analysis Method*, Elsevier, Oxford, 1998.

[10] Horbæk, K., "Current practice in measuring usability : Challenges to usability studies and research", *International Journal of Human-Computer Studies*, Vol.64, No.2(2006), pp.79-102.

[11] Howarth, J., T. Andre, and R. Hartson, "A structured process for transforming usability data into usability information", *Journal of Usability Studies*, Vol.3, No.1(2007), pp.7 -23.

[12] Huber, J., *A Comparison of IBM's Orthogonal Defect Classification to Hewlett Packcard's Defect Origins, Types, and Modes*, Hewlett Packard Company, 1999.

[13] ISO/IEC 9241, *Ergonomic requirements for office work with visual display terminal -Part 11 : Guidance on usability*, ISO/IEC, 1998.

[14] Ivory, M. and M. Hearst, "The state of the art in automating usability evaluation of user interfaces," *ACM Computing Survey*, Vol.33, No.4(2001), pp. 470-516.

[15] Keehan, S., H. Hartson, D. Kafura and R. Schulman, "The usability problem taxonomy : A framework for classification and analysis", *Empirical Software Engineering*, Vol.4, No.1(1999), pp.71-104.

[16] Kruchten, P., "Casting software design in the function-behaviour-structure framework", *IEEE Software*, Vol.22, No.2(2005), pp.52-58.

[17] Lavery, D., G. Cockton, and M. Atkinson, "Comparison of evaluation methods using structured usability problem reports", Vol. 16, No.4/5(1997), pp.246-266.

[18] Law, E., *Proposal for a New COST Action 294 : Towards the Maturation of IT Usability Evaluation*, COST Office, 2004.

[19] Leszak, M., D. Perry and D. Stoll, "Classification and evaluation of defects in a project retrospective", *The Journal of Systems and Software*, Vol.61, No.3(2002), pp.173-187.

[20] Nielsen, J., *Usability engineering*, AP Pro-

fessional, San Diego, 1993.

[21] Peuple, J. L. and R. Scane, *User interface design*, Crucial, Exeter, 2004.

[22] Rasmussen, J., *Information processing and human-machine interaction-An approach to cognitive engineering*, Elsevier Science, 1986.

[23] Ryu, H. and A. Monk, "Analysing interaction problems with cyclic interaction theory : Low-level interaction walkthrough", *Psychnology Journal*, Vol.2, No.3(2004), pp. 304-330.

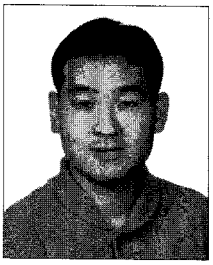
[24] Schoeffel, R., "*The concept of product usability : a standard to help manufacturers to help consumers*", ISO Bulletin, March(2003), pp.5-7.

[25] Te'eni, D., J. Carey and P. Zhang, *Human-computer interaction : Developing effective organizational information systems*, John Wiley, New Jersey, 2007.

[26] Vilbergsdottir, S., E. Hvannberg and E. Law, "Classification of usability problems(CUP) scheme : Augmentation and exploitation," *Proceedings of NordiCHI 2006*, 2006, pp. 281-290.

[27] Zhang, Z., *Overview of usability evaluation methods*, Retrieved 20 May 2008, from http://www.usabilityhome.com.

◈ 저 자 소 개 ◈



함 동 한 (d.ham@mdx.ac.uk)

인하대 산업공학과를 졸업하고 KAIST 산업공학과에서 공학석사 및 공학박사를 취득하였다. ETRI에서 선임연구원으로 근무하면서 소프트웨어 및 시스템공학 관련 연구를 수행하였다. 현재 영국 Middlesex University, School of Engineering and Information Sciences에서 종신 연구교수로 재직 중이다. 주요 연구분야는 인지시스템공학, 인간-컴퓨터 상호작용, 정보시스템 및 서비스공학이다. 현재 복잡도 기반의 인지적 직무 설계 및 평가, 의사결정 지원을 위한 정보의 시각화, 사용자 중심의 지식서비스에 관련된 연구를 수행하고 있다.