이원적 품질속성을 이용한 효율적인 요구분석 방법

* 조장희, 서성채, 김귀연, 김병기
전남대학교 전산학과 소프트웨어공학연구실
veronica@natural.chonnam.ac.kr, {scseo, gykim, bgkim}@chonnam.ac.kr

Efficient Requirements Analysis using Dual Approach of Quality Attribute

* Jang Hee Cho, Seong Chae Seo, Gwi Yeon Kim, Byung Ki Kim
Dept. of Computer Science, Chonnam National University

요 약

소프트웨어 개발에서 요구사항 분석에 대한 관리는 품질과 생산성에 중요한 역할을 한다. 기존 연구는 요구사항 분석단계에서 기능 중심으로 문제분석을 시도하고, 시험 및 구현단계에서 품질문제를 고려하고 있다.
본 논문에서는 요구사항을 추출하고 분석하는 단계에서 품질속성을 고려하는 요구분석 모델을 제안한다. 품질평가모형인 ISO/IEC 9126 품질속성으로 분석된 요구사항을 개발시스템의 이해와 사용자의 요구사항에 대한 만족도를 높일 수 있도록 카노(Kano)의 이원적 품질이론을 통해서 재분류함으로써 새로운 요구분석 방법을 제안한다.

1. 서론

현대사회에서 과학기술의 발달은 인류에게 큰 편의를 제공하게 되었다. 특히, 과학기술에 소프트웨어가 차지하는 비율은 점차 증가하고 있다. 기존의 사무 자동화를 위하여 사용되었던 소프트웨어가 이제는 인간의 생활에 없어서는 안되는 위치까지 오게 되었다. 따라서 소프트웨어가 실생활에 미치는 파급 효과는 매우 크다[1]. 특히 소프트웨어 개발 단계에서 요구사항들의 구체적인 정의는 성공적인 소프트웨어 개발의 필수 요건이다. 설계와 구현이 잘되더라도 사용자의 요구가 반영되지 못한 소프트웨어는 실패작일 뿐이다. 최근에는 요구사항에 대한 관리가 소프트웨어 시스템 개발의 중요한 성공요인으로 등장하게 되어 요구공학(Requirement Engineering)이 활발히 연구되고 있다.

요구공학이란 요구분석단계에서 행해졌던 요구사항 분석 및 서술뿐 아니라 이들의 추출, 관리, 검증, 유지 등을 포함하여 요구사항에 관계되는 모든 활동과 원칙들에 대한 공학적 접근을 의미한다[2]. 이와 같이 초기단계의 사용자 요구사항을 정확히 파악하고 이해하는 것은 성공적인 소프트웨어 개발에 중요한 척도가 된다.

또한 요구분석단계에서 고려할 점은 품질속성의 중요성이 언급되어진다. 소프트웨어 품질 속성은 시스템이 만족해야 하는 품질에 대한 요구사항을 의미한다. 즉, 품질 속성을 만족하지 못하는 자원이 시스템 구축에 사용됨으로써 시스템 전체가 품질에 대한 요구를 만족하지 못하는 경우가 발생하게 된다. 이러한 이유로 시스템 구축 전에 품질에 대한 요구를 만족하는지를 평가하고 결정하는 것은 매우 중요하며 이는 소프트웨어 아키텍처의 평가에도 많은 영향을 미친다. 본 논문에서는 요구공학에서 요구사항을 추출하는 방법에 대해 살펴보고 효율적인 요구사항 추출에 관한 새로운 접근방법을 제시한다.

본 논문의 구성은 다음과 같다. 2장에서는 요구사항의 중요성과 현재 국제적으로 사용의 효율성을 인정받고 있는 품질평가모형 ISO/IEC 9126에 대해서 살펴본다. 또한 요구공학 프로세서를 살펴보고 카노(Kano)가 제시한 품질의 이원적 품질이론에 대해 살펴본다. 3장에서는 기존의 요구공학 방법에 카노의 이원적 품질요소를 이용한 품질속성의 재분류와 분류된 속성 메트릭스의 매핑을 통하여 정보를 추출, 분석함으로서 요구사항에 대한 명확한 이해와 시스템의 품질 향상을 가져다 줄 수 있는 새로운 접근방법을 제시한다. 마지막으로 4장에는 결론 및 향후 연구방향을 기술한다.

2. 관련연구

2.1 요구분석의 중요성 및 요구공학 프로세서

소프트웨어 요구사항은 개발에서부터 구현까지 납기, 비용, 품질에 결정적인 영향을 미치는 요소 중의 하나이다. 세계적인 소프트웨어 생산성 연구기관인 SPR에서는 소프트웨어 프로젝트가 실패하는 가장 근본적인 원인을 소프트웨어 요구공학의 미성숙과 부적절한 견적 및 계획으로 보고하고 있다. 즉, 올바른 요구사항을 조기에 추출하지 못함으로 인해 소프트웨어 개발에 있어서 오류가 늦게 발견될수록 문제해결이 더욱 어려워지며 오류수정 비용도 커진다. 어떠한 소프트웨어 개발방법론을 사용하더라도 모든 설계의 기초는 요구사항이며 이는 적절히 정의되어야 한다[6].
고객의 요구사항을 정확히 정의하면 다음 단계에서 발생하

는 오류 또는 부적합사항은 줄어들게 된다. 요구사항의 정의 및 검증은 고객의 요구사항을 정의하는 것으로부터 출발한다. 개발의 각 단계마다 고객의 요구사항은 요구명세로 변환되고 이 명세는 역으로 고객의 요구사항과 일관성이 있어야 한다. 정확한 요구사항의 정의는 고객의 제품에 대한 만족도를 향상시키고 유지보수 및 지원비용을 절감시킨다. 또한 개발과정에서의 재작업을 축소함으로써 개발주기를 단축시키고 높은 생산성과 품질향상을 가져온다.
그러나 고객의 요구사항을 만족스럽게 기술한다는 것은 어려운 일이므로 고객의 요구사항 명세서는 오류의 원인이 될 수 있다. 부적절한 요구사항은 설계를 불가능하게 하고, 목표가 명확히 정의되지 않음으로 인해 생산과 관리를 효율적으로 수행시킬 수 없게 한다.

이러한 요구분석단계의 중요성에 대한 인식이 높아짐에 따라 제품개발을 위한 요구사항 설정단계에서부터 제품개발과 테스트, 생산에 이르기까지 개발공정의 매 단계마다 초기에 정한 개발 요구사항들은 물론 이후의 상세요구사항들이 제품설계와 구현단계에서 제대로 지켜지고 있는지를 검증해 나가는 요구공학이라는 기법이 대두되었다.

요구공학 프로세서는 크게 요구사항을 추출, 분석, 명세서, 검증, 유지보수의 5단계 공정으로 분류된다. 그 내용을 살펴보면 다음과 같다[2].

① 요구사항 추출 : 고객, 시스템 사용자 그리고 시스템 개발에 관련된 사람들과 의견을 교환함으로써 개발하고자 하는 시스템에 대한 요구들을 찾아내는 공정.
② 요구사항 분석 : 사용자의 필요사항를 이해하고 문제해결을 위한 제약사항을 정리하는 공정.
③ 요구사항 명세서 : 분석된 요구사항을 명확하고 정확하게 기록하여 문서화하는 것.
④ 요구사항 검증 : 소프트웨어 요구사항이 사용자들의 요구사항에 맞게 정확하고 완벽하게 그리고 연계성 있게 명세되었는지를 검증하는 것.
⑤ 요구사항 유지보수 : 요구사항의 변경들을 체계적으로 관리하는 것.

2.2 소프트웨어의 품질을 정의하는 소프트웨어 품질모형

소프트웨어의 품질을 정의하는 소프트웨어 품질 모형에 대한 연구가 1976년과 1977년 Boehm[7]과 McCall[8]에 의해 각각 시작되었고, 그 이후에도 다양한 품질 모형들이 제시되었다. 이러한 기존의 소프트웨어 품질 모형들은 '제품 특성을 측정하고 제어함으로써 제품의 품질을 향상시킬 수 있다'고 가정한다. 그러나 기존의 모형에서 이러한 가정은 이론적인 근거가 뒷받침되어 있지 않으므로 소프트웨어의 품질과 메트릭스와의 관계는 모호하다. 따라서 소프트웨어의 품질 속성이 만족되기 위해서 어떤 메트릭스 집합이 만족되어야 하는지, 반대로 메트릭스가 만족되었을 경우 품질의 어떤 측면이 개선되는지 등을 알수 없다. 기존의 소프트웨어 품질모형은 소프트웨어의 품질관리에 실제로 사용하기에는 부족한 점이 있다. Dromey[9]는 개발 단계별로 소프트웨어의 구성 요소를 식별하고, 이러한 구성요소의 특성으로 단계별 품질모형을 구성하였다. 이 방법은 제품 특성에 기반을 두고 소프트웨어 품질을 제어할 수 있는 기능성을 보여주지만, 이 방법으로 각각의 품질 속성이 향상되었는지, 향상된다면 어떻게 향상되는지를 보여주지 못한다.
ISO/IEC 9126의 소프트웨어 품질 모형은 국제 표준으로서 소프트웨어의 품질 정의에 대한 보편적인 모형이다. 이 모형은 품질 속성을 세분화된 품질 속성으로 나누어 명시하므로 제품이 완성된 후의 품질 측정에 적합하다. 그러나 개발 프로세스에서는 직접적인 품질 속성, 즉 제품의 외부적인 행위를 평가할수 없으므로 프로세스에서의 품질을 검증하는 기준 모형으로 사용되기는 불충분하다. 따라서 이제까지의 소프트웨어의 품질 관점에서 단계별 메트릭스를 결정하기 어렵다. 이렇듯 개별적으로 이루어지는 메트릭스의 연구는 제품 특성을 정량적으로 측정하는 방법을 발전시키고 있으나, 이렇게 측정되는 메트릭스가 소프트웨어의 품질에 있어서 가지는 의미를 제대로 보여주지 못하고 있다.

2.3 품질속성 평가

품질속성은 양이나 질로 관찰하여 수치로 측정할 수 있는 시스템의 속성을 말한다. 품질속성은 이해관계자들의 관심사와 요구사항을 그대로 반영하고, 아키텍처는 이해관계자들이 원하는 수준으로 품질속성을 달성해야 한다.
단순히 성능이 좋다거나 오류가 없다거나 쓰기 쉽다는 것으로 소프트웨어 시스템의 품질을 규정할 수 없다. 품질모델은 품질속성을 분류하고 정의하여 누구나 인정할 수 있는 품질 측정 기준을 정의한 것이다. 품질 모델을 도입하면 소프트웨어 시스템의 품질을 정형화(formalization)할 수 있다. 국제표준 기구에서는 ISO/IEC 9126 소프트웨어 품질 모형과 ISO/IEC 14598 소프트웨어 제품 평가를 국제표준으로 제시하였다. ISO/IEC 9126은 소프트웨어의 품질 속성을 기능성, 신뢰성, 사용성, 효율성, 유지보수성, 이식성으로 구분하고 있다[3].

① 기능성(Functionability)
기능 집합과 사양화된 내용을 실현하는 특성의 집합으로써 명시적 또는 암시적 요구를 만족하는 속성의 집합을 의미한다. 부 속성으로는 적합성, 정확성, 상호호환성, 유연성, 보안성이 있다.
② 신뢰성(Reliability)
명시된 조건하에서 명시된 기간, 소프트웨어의 실행 레벨을 유지하기 위한 능력을 만족하는 속성의 집합을 의미한다. 부 속성으로는 성숙성, 오류허용성, 회복성이 있다.
③ 사용성(Usability)
명시적 또는 암시적으로 사용자가 이용하기 위해 필요한 노력으로 각각의 사용 결과에 의한 평가를 나타내는 속성의 집합을 의미한다. 부 속성으로는 이해성, 습득성, 운용성이 있다.
④ 효율성(Efficiency)
명시적인 조건하에서 소프트웨어의 실행 레벨과 사용되는

자원양자간의 관계를 나타내는 소프트웨어 속성의 집합을 의미한다. 부 속성으로는 실행효율성, 자원효율성이 있다.

⑤ 보수성(Maintainability)

사양화된 개정을 처리하기 위해 필요로 하는 노력을 나타내는 속성의 집합을 의미한다. 부 속성으로는 해석성, 변경성, 안정성, 시험성이 있다.

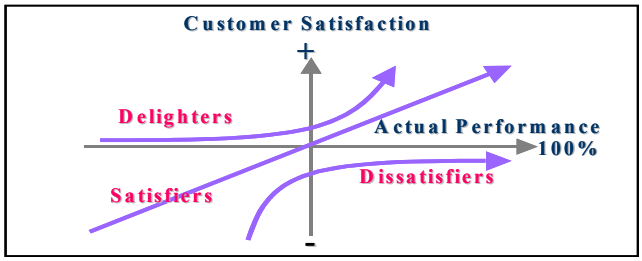
⑥ 이식성(Portability)

임의의 환경에서 다른 환경으로 소프트웨어를 이식하기 위한 속성의 집합을 의미한다. 부 속성으로는 환경적응성, 이식작업성, 일치성, 치환성이 있다.

요구사항을 추출하고 분석하는 단계에서 전체 시스템에서 차지하는 요구사항 항목들을 정하고 분류하는 것은 향후 설계 및 아키텍처 생성, 구현단계에서 개발자와 사용자 모두에게 폭 넓은 개발 방안을 제공하게 된다. 따라서 추출 단계에서 품질을 고려한다면 시스템 개발 시 상당한 비용 감축을 가져올 수 있다.

2.4 카노(Kano)의 이원적 품질이론

대부분의 소비자들은 제품의 미비한 부분에 대해서는 불만을 가지면서도, 품질수준이 충분한 경우에는 당연하다고 느낄 뿐 만족감을 가지지 않는 경향이 있다[4,5].



(그림 1) Kano's Diagram

카노(Kano)는 (그림 1)과 같이 품질의 이원적 인식방법을 제시하였고, 품질수준이 높아지면 사용자는 당연히 만족하게 되지만(일원적 품질), 고객이 미처 기대하지 못했던 것을 충족시켜주거나 고객이 기대했던 것이라도 고객의 기대를 훨씬 초과하는 만족을 주는 품질요소(매력적 품질)를 보여주고, 마땅히 있어야 하지만 없으면 불만이 되는 품질요소(당연적 품질)를 설명하고 있다.

또한 사용자와 개발자가 시스템의 자원을 공유할 때 요구사항의 검증이 사용자 관점에서 품질 만족요소로 작용할 수 있다는 전제로 요구사항에 따른 시스템 구현에 적용하면 만족/불만족 이라는 주관적 측면(개발자)과 충족/불충족 이라는 객관적 측면(사용자)의 관계로 적용될 수 있다.

Kano's Model에 의한 품질요소 및 특성을 살펴보면 <표 1>과 같다.

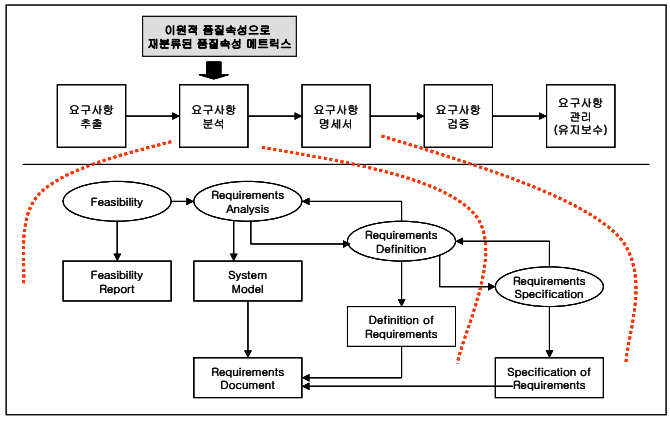
<표 1> Kano's Model의 품질요소 및 특성

품질요소	특 성
당연적 품질 (Dissatisfiers)	- 당연히 존재하여야 하거나 기대되는 제품의 특성으로 결여될 경우 고객의 불만을 초래 - 고객은 이러한 제품의 특성에 대해서는 요구하지 않음 - 고객의 불만을 최소화하기 위해서는 최대한 줄어야함. 이를 줄이기 위해서는 고객의 불만사항의 수집과 처리가 필수적. 단 이는 최상의 품질을 달성하기 위한 충분조건이 될 수 없음
매력적 품질 (Satisfiers)	- 요구품질(Desired Quality)을 충족시킴. 일반적으로 고객이 요구함 - 제품의 특성 중 Satisfiers(비용의 저하, 신뢰도의 향상, 속도의 향상 등)가 많으면 많을수록 고객의 만족도는 향상됨 - 경쟁도 분석을 위한 Benchmark로 사용됨
일원적 품질 (Delighters)	- 잠재적인 요구를 충족시킴 - 고객이 처음 접했을 때, 흥분과 즐거움을 주는 제품의 특성 - 제공되지 않더라고 불만족은 발생하지 않음. 고객이 요구하지 않음 - 이러한 특성은 특이하며, 어떠한 패턴도 존재하지 않는 것으로서 새로운 시장을 형성시킴

3. 카노의 이원적 품질속성을 이용한 품질속성 재분류

요구사항을 추출하고 분석하는 단계에서 체계적으로 아키텍처와 품질속성간의 관계를 잘 정리하면 분석 및 설계과정에서 큰 향상을 기대할 수 있다.

아키텍처는 시스템이 달성해야 하는 품질들을 실현할 때 필요한 큰 그림을 결정하기 때문에 중요하다. 따라서 품질을 달성할 수 있도록 아키텍처를 분석하고 설계해야 한다.



(그림 2) 제안모델을 적용한 요구공학 다이어그램

제안 모델은 (그림 2)과 같이 일반적인 요구공학 프로세스의 요구사항 추출분석단계에서 국제 표준으로서 소프트웨어의 품질 정의에 대한 보편적인 모형인 ISO/IEC 9126의 세분화된 품질 속성으로 나누어 명시하고 세분화된 품질속성을 카노(Kano)의 이원적 품질속성으로 재분류한다. 카노의 이원적 품질 인식은 품질을 구성하는 개개의 품질요인에 대하여 사용자의 만족으로 나타나는 주관적인 측면과 제품이나 서비스의 요구조건에 대한 일치성을 나타나는 객관적인

측면이 대응 관계를 이루고 있음을 보인다. 과거의 품질에 대한 인식이 모든 품질요인들이 만족과 불만족을 함께 내포하고 있다는 일원적 인식을 가지고 있었다고 보았으나 품질요소는 물리적인 충족과 주관적인 만족 간에 대응관계를 이루며 만족과 불만족의 연속선상에 위치를 하지 않고 개별적으로 차이점을 가진다고 주장을 하였다. 이는 소프트웨어의 성공적인 개발을 위해서는 매력적 품질요소의 극대화와 당연적 품질요인의 불만족을 제거하는 수준으로의 향상을 유도할 필요성이 있음을 암시하고 있다.

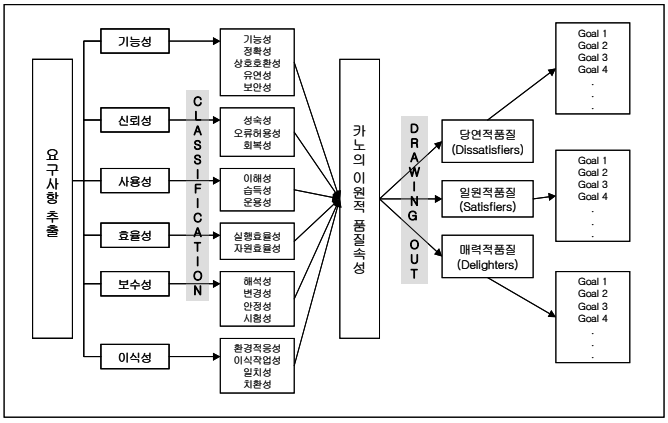
본 논문에서는 이원적 품질속성을 이용하여 분류된 속성 메트릭스의 매핑을 통하여 정보를 추출, 분석함으로서 명확한 요구사항에 대한 이해와 시스템의 품질 향상을 가져다 줄 수 있는 새로운 요구분석방법을 제시한다.
즉 요구분석단계에서 제안된 이원적 품질속성 메트릭스를 적용함으로서 요구사항을 추가적으로 세분화하여 추출할 수 있으며 요구공학에서의 최상위 레벨에서 요구사항을 정확히 분석해줌으로써 효율적으로 품질을 분석하고 평가할 수 있는 요구사항 분석을 가능하게 한다.

<표 2>카노의 이원적 품질속성을 이용한 품질속성 재분류

ISO 9126 품질속성 \ Kano's Model		Dissatisfiers	Satisfiers	Delighters
기능성	적 합 성	○	○	○
	정 확 성	○	○	○
	상호호환성		○	△
	유 연 성			
	보 안 성		○	
신뢰성	성 숙 성	○	○	△
	오류허용성	△	△	
	회 복 성	△	○	
사용성	이 해 성			
	습 득 성			
	운 용 성			
효율성	실행효율성	○	△	△
	자원효율성			
보수성	해 석 성	△	○	
	변 경 성		○	
	안 정 성			
	시 험 성		○	
이식성	환경적응성			
	이식작업성		○	
	일 치 성	○	△	
	치 환 성	△		

※ ○ : 깊이 관련 △ : 다소 관련

시스템의 품질을 관리하기 위해 요구 사항 프로세스 단계에서 미리 해당 요구사항의 품질이 어떤지를 고려한다면 향후 시스템 구현 시 어떤 요구 사항이 반드시 구현되어야 하는지 기준을 제공하게 된다. <표 2>에서는 요구사항 추출단계에서 생성된 요구사항들을 ISO에서 제시한 품질 모델에 비추어 각 요구사항별로 어떤 품질이 나타나고 있는지를 메트릭스로 표시하였다. 이렇게 생성된 품질속성분류에 품질속성의 성격에 따라 카노의 이원적 인식방법으로 재분류시킨 것이다.



(그림3) 이원적 품질속성을 이용한 요구사항 재분류방법

이원적 품질속성을 이용한 속성재분류 방법을 통해서 요구분석을 (그림 3)과 같이 세분화시킬 수 있다.

4. 결론 및 향후 연구방향

우리의 실생활에 소프트웨어가 밀접히 관계되고 소프트웨어 품질은 이러한 소프트웨어 개발에 중요한 요소이기 때문에 소프트웨어 품질평가와 관리에 대한 관심이 높아지고 있다. 본 논문에서는 ISO/IEC 9126의 세분화된 품질 속성을 카노(Kano)의 이원적 품질속성으로 재분류하여 요구사항을 추출, 분석함으로서 명확한 요구사항에 대한 이해와 시스템의 품질 향상을 가져다 줄 수 있다. 또한 요구분석 단계에서 요구사항을 정확히 분석해줌으로써 효율적으로 품질을 분석하고 평가할 수 있는 요구사항 분석을 가능하게 한다. 향후 연구과제로 충분한 사례연구를 통한 품질속성 메트릭스의 유효성 검증과 품질 평가 기준 마련이 요구된다.

참고문헌

[1] Roger S. Pressman, "Software Engineering A Practiners' Approach", 3rd Ed. McGraw Hill.
[2] Richard H. Thayer and Merlin Dorfman, "Software Requirements Engineering", 2nd Edition IEEE Computer Society Press, 1997.
[3] ISO/IEC, "ISO/IEC 9126, Information technology software Product Evaluation-Quality Characteristic and Guidelines for Their Use", ISO/IEC, Dec. 1991.
[4] Kano, "매력적 품질과 보통품질," 品質, Vol. 14, February, pp. 39-48, 1984.
[5] 류한주, "품질개념에 대한 이원적인식방법의 고찰", 대한품질경영 심포지엄 발표문집, pp. 59-67. 1995.
[6] Ivan Jacobson, "Trends 2002-03-26", KCSE, 2002.
[7] Bohem, B. W., "software Engineering Economics", Prentice-Hall, 1981.
[8] McCall, J. A. &D. markham & M. Stosick, R. Mcgindly, "The Automated Measurement of Software Quality", IEEE, 1981.
[9] Dromey, R. Geoff. " Cornering the Chimera", IEEE Software, Vol. 13, No. 5, pp. 33-43, Jan. 1996.