효율적인 요구사항 분류방법에 관한 연구

정찬일*, 이동현**, 인호**
*고려대학교 컴퓨터정보통신대학원 디지털정보공학과
**고려대학교 컴퓨터학과
e-mail:{bankplus,tellmehenry,hoh_in}@korea.ac.kr

A Study on Efficency Requirement Classification

Chan-Yil Jung*, Dong-Hyun Lee**, Hoh Peter In**
*Dept of Digital Information Engineering, Korea University
**Dept of Computer Science & Compute Engineering, Korea University

요 약

정보시스템의 개발에 있어서 요구사항의 체계적인 관리가 중요시되는 가운데, 요구사항에 대한 세부적인 정보들을 계량적으로 관리하기 위한 체계적인 분류방법이 부족한 것으로 판단되어, 요구사항을 업무로직, 데이터, 기술기반 관점 및 CEO/CIO, 책임자, 설계자 시각에 따른 요구사항 분류매트릭스를 제공함으로써, 요구사항을 좀더 체계적으로 수집, 관리할 수 있는 기반을 마련하였다.

1. 서 론

정보시스템의 구축에 있어서, 명확한 요구사항의 수집 및 변경관리는 필수적인 성공요소이다. standishgroup에 의하면 소프트웨어 프로젝트의 성공요인의 절반 이상이 요구사항과 관련이 있다.[1] 이에 따라 90년대에 들어오면서 요구사항에 관련되는 모든 활동과 원칙들을 요구공학(Software Engineering)이라는 이름으로 많은 연구가 진행되고 있다.[2][3]

소프트웨어 개발의 핵심인 요구사항의 수집 및 관리를 위해 대화분석(dialogue analysis), 요구사항범주화(categorization), 클러스터링(clustering)등 다양한 요구공학적 접근방법들이 연구 되어지고 있으나, 이러한 방식들은 기본적으로 자연어처리 기법(natural language processing technique)을 중심으로 하고 있어 정량적인 분석을 위해서는 좀더 체계적인 분류기준이 필요할 것이다.[2]

본 논문에서는 소프트웨어 개발에 관여되는 다양한 종류의 이해당사자(Stakeholder)들에 의해 도출되는 다양한 형태의 요구사항들을 체계적으로 수집하고, 관리하기 위해 Enterprise Architecture(EA)의 프레임워크와 같이 관점과 시각이라는 기준에 의한 요구사항 분류 매트릭스를 제시하고자 한다.

본 논문의 구성은 다음과 같다. 먼저 2장에서 기존의 요구공학 논문들에서의 요구사항 분류 현황을 되짚어 보고, EA의 프레임워크와 비교하며, 3장에서 요구사항에 대한 구체적인 분류 모델을 제시함으로써 요구사항 수집 및 관리의 프레임워크로서 요구사항분류매트릭스를 정의하고자 한다. 마지막으로 4장에서 결론 및 향후 연구과제를 제시한다.

2. RELATED WORK

2.1 기존의 요구사항 분류(Classification)

요구사항정의에 대한 국제 표준으로서 IEEE 830은 프로젝트 수행시 Software Requirement Specification (이하 SRS)에 대한 내용 및 작성법이 정의되어 있다.[4] SRS는 엄밀히 말해 명확한 요구사항의 분류체계라기 보다는 문서 목차에 가까운데, 요구사항을 Introduction, Overall description, Specific requirements의 순서로 상세화 되지만, 각각의 목차들은 요구사항에 대한 분류가 혼합되어 나타나고 있다.

Soren Lauesen의 경우 조금 더 구체적으로 요구사항을 Contents를 기준으로 데이터, 기능, 품질, 관리, 이행 요구사항 등으로 분류하고, 요구사항의 Goal-driven Scale을 기준으로 Goal-level, Domain-level, Product-level, Design-level등으로 분류하였다.[14] 이러한 Scale 혹은 Level별 분류는 일반적으로 Stakeholder의 분류와 일맥상통한다고 볼 수 있다. Goal-level은 상위의 의사결정자들의 시각이고, Design-level로 내려 갈수록 보다 실질적인 담당자, 설계자 수준의 요구사항들이 정의되기 때문이다.

A Stakeholder Win-Win Approach의 경우 각각의 Stakeholder들은 동등한 입장에서 상호 협상과 중재를 시도한다.[15] 그러나 의사결정자의 요구사항과 설계자의 요구사항이 갖고 있는 상하관계 및 요구사항의 종속성에 대해서는 명확히 언급되지 않았다. 이러한 상하관계에 대한 이해는 요구사항의 적합성 판단에 도움을 줄 수 있다. 즉 개별 이해당사자간에 합의를 이끌어냈다고 하더라도, CEO의 요구사항, 즉 소프트웨어 개발의 근본적 목적, 전략에 어긋난다면, 그러한 합의는 의미가 없어질 수도 있기 때문이다.

일반적으로 국내 소프트웨어 프로젝트의 경우 “요구사항정의서”라는 형태로 요구사항목록을 정의하고 있다. 이러한 요구사항목록은 요구사항의 분류 구분이 거의 전무하며, 있다고 하더라도 단지 “기능요구사항”과 “비기능요구사항”정도로 요구사항을 구분하고 있다.

실제 소프트웨어 개발 프로젝트에서 요구사항 정의 단계는 이해당사자들간의 대화를 통한 수집이 중심이므로, 자연어를 통한 서술이 주된 작업 방법이 된다. 이에 따라 요구사항에 대한 분류 역시 서술된 문장들을 키워드 중심으로 분류해 나가면서 유사한 내용들을 하나로 묶어 나가다 보니 요구사항의 분류는 특정한 표준에 따르기보다는 프로젝트별로 상이한 경우가 대부분이다. 또한 Stakeholder들에 대한 분류 역시 명확한 기준이 있다기보다는 개별 프로젝트에서 파악된 이해당사자들을 기준으로 묶어나가는 방식이기 때문에 프로젝트에 따라 ad-hoc하게 정의되는 것이 일반적이다.

2.2 Enterprise Architecture의 아키텍처 분류

John Zachman은 1987년 IBM 시스템 저널에서 “A framework for information systems architecture”라는 논문을 통해 EA를 개척하였다.[7][8]
자크만은 프레임워크를 통해 하나의 시스템은 두 개의 축 즉, 이해당사자별 관점(″perspective″)과 모델링 및 구현을 위한 대상(″focus″)에 의해 (5x6) 매트릭스로 표현하였다. 여기에서 '관점'은 planner, owner, designer, builder, sub-contractor 등의 5개로, '대상'은 6하 원칙에 해당되는 what, how, where, who, when, why의 6 개로 구분되며, 국내에서는 정보통신단체표준인 TTA.KO-10.0153를 통해 “공공부문 전사적 아키텍처 프레임워크 표준”을 제정하였다.[9][11]

이러한 EA의 프레임워크들은 기업의 전체 시스템을 이해당사자별 관점(″perspective″)과 모델링 및 구현을 위한 대상(″focus″, “view″)에 의해 2차원 매트릭스형태로 표현함으로써 전체 시스템의 현황을 한 장의 그림으로 표현한다. 이러한 EA 프레임워크의 기본 사상을 요구사항의 수집 및 관리에 적용하여, 요구사항들 한눈에 조감할 수 있고, 관리할 수 있는 2차원 형태의 매트릭스를 작성함으로써 효율적인 요구사항의 수집 및 관리를 위한 분류기준을 제시하고자 한다.

3. 요구사항 분류 매트릭스

3.1 요구사항의 시각(Perspective)별 분류

요구사항의 수집은 일반적으로 Top-Down형태로 진행된다. 물론 경우에 따라 기존 시스템의 고도화와 같은 경우 Bottom-Up방식으로 요구사항을 분석하는 방식을 사용할 수도 있겠으나, SRS의 목차는 1. Introduction을 통해서 프로젝트의 목적, 범위등을 정의하고, 2. Overall description을 통해서 제품의 기능들을 검토하며, 3. Specific requirements에서 더욱더 자세한 사항들을 기술하는 Top-Down방식을 취하고 있다.[4]

이러한 방식에 따라 요구사항을 소프트웨어 개발을 둘러싼 다양한 Stakeholder들을 시각(Perspective)을 중심으로, CEO/CIO 시각(Planner, Contextual), 관리자(Owner, Conceptual), 시각, 설계자(Designer, Logical) 시각으로 분류하였다.

표 4 이해관계자 시각(Perspective)별 분류설명

Perspective	설명
의사결정자 시각 -Planner, Contextual	구축하고자 하는 소프트웨어의 의미, 개괄적인 설명, 비즈니스적인 필요성 및 전략 등에 대한 개괄적인 요소를 정의한다.
관리자 시각 -Owner, Conceptual	소프트웨어가 준수하여야할 표준, 비기능적인 필요사항, 개발 제약사항등에 대한 개념적인 요소를 기술한다.
설계자 시각 -Designer, Logical	구축하고자 하는 소프트웨어의 구체적인 기능 및 성능 상의 필요조건. 각각의 상세한 설명을 포함한다.

이해관계자간의 시각(Perspective)별 분류는 기존의 여타 요구사항 수집방안에서 존재하였던 일반적인 Stakeholder별 요구사항 분류라는 측면과는 다르게, 각각의 시각들은 개별적인 자신의 입장을 대표하는 것 보다는 상위에서 하위로의 논리적인 흐름을 대표한다고 볼 수 있다. 즉 의사결정자와 관리자들 간의 입장차이를 해소하기 위한 분류가 아니라, 요구사항의 개괄적 측면(큰그림, Big Picture)과 개념적 측면, 그리고 실제 설계를 하기 위한 논리적 측면에 대한 분류로서 각각의 시각들은 종속관계를 갖게 된다.

또한 의사결정자시각의 요구사항은 관리자 시각의 요구사항들에 의해 구체화되며, 이것이 다시 설계자 시각에서 상세화 되므로, 각 단계의 요구사항은 상위수준의 요구사항 정의에 위배하거나, 상호대치 되지 않아야 한다. 이러한 요구사항에 대한 연관관계(Relationship)을 정의함으로써, 개별 요구사항이 전체의 큰그림에 부합하는가를 확인할 수 있을 것이다.

3.2 요구사항의 관점(View)별 분류

기존의 요구사항 분류방법들은 크게 Function, Non-Function 요구사항으로 요구사항을 분류하고, 더욱더 세부적으로 Purpose, Scope, Functional requirement, External interface requirement, Performance requirement, Design constraints, Attribute, Other requirement 등으로 요구사항을 분류하였다.[4]

그러나, 이러한 분류방법은 실질적으로 요구사항들을 수집하는데 있어서, 각각의 프로젝트별로 수집된 요구사항들을 분류하여 정리함으로써, 작성자의 자유도가 너무 높고 표준화가 미흡한 단점이 있었다. 실제로 작성된 SRS들을 살펴보면 대부분의 요구사항들은 Other requirement에 기술되는 경우가 많으며, Function Requirement와 Design constraints와 같은 경우 동일한 수준의 입도(Granularity)라고는 보기 어려운 것으로서, 어느 한부분의

내용은 너무 많고, 다른 쪽은 너무 적은 요구사항이 담김으로써, MECE 원칙에 어긋나는 것을 확인할 수 있었다.

일반적인 소프트웨어의 구현을 위한 대상(Focus)을 구분하는데 있어서, 가장 역사가 오래되었으며, 동시에 가장 익숙한 개념인 육하원칙(5W1H)의 개념에 따라 분류 해보면, What은 어떠한 정보에 관한 소프트웨어 인가를 표현한다. 즉 데이터관점의 소프트웨어 특성을, When은 소프트웨어의 시간적인 특성, 즉 시간상의 제약사항 및 진행형태를 표현하고, Where은 소프트웨어의 Network적인 관점에서의 요구사항을 표현한다고 정의할 수 있다. Who는 소프트웨어의 사용자, 관리자 등의 이해당사자들에 대한 요구사항을 표현한다. 즉 이해당사자들이 제시하는 요구사항이 아니라 이해당사자 자체에 대한 내용을 기술하는 것이다. Why는 소프트웨어 개발의 Motivation에 대한 기술로서, 비즈니스적인 요청 및 동인에 대한 설명이다. 마지막으로 How는 일반적인 소프트웨어의 기능 요구사항, 즉 Function에 대한 정의 및 프로세스를 기술한다.

이러한 6하원칙에 근거한 분류를 MECE에 입각하여 중복을 줄이고, 일반적으로 소프트웨어에 적용될 수 있도록 업무(Business Logic), 데이터(Data), 기술기반(Technology Infra)의 3가지 유형으로 관점을 분류하였다. 업무영역은 Why와 How를 데이터 영역은 What, Who을 기술기반 영역은 Where 및 When등의 영역을 포함하며 아래와 같은 의미를 지닌다.

표 6 관점(View)에 따른 분류 설명

View	설명
업무(Business Logic)	소프트웨어의 개발 동인 및 기능(Function), 프로세스에 대해 기술한다. 이때 소프트웨어를 사용하지 않는 일반적인 업무 내용까지를 포함한다.
데이터(Data)	소프트웨어를 사용하고 관리할 대상 및 데이터의 특징에 대해 기술한다.
기술기반(Technology Infra)	소프트웨어를 개발하기 위한 제약조건 및 하드웨어, 기반 솔루션 등의 인프라 및 사용기술등에 대해 기술한다.

3.3 요구사항 분류 매트릭스

상기한 요구사항에 대한 관점 및 시각별 분류에 근거하여 아래와 같은 3X3의 분류 매트릭스를 작성할 수 있었다.

표 7 요구사항 분류 매트릭스

	업무로직(Business Logic)	데이터(Data)	기술기반(Technology Infra)
의사결정자(Contextual)	BV1. 업무정의	DV1. 이해관계자 정의 DV2. 데이터 정의	TV1. 기반구조 정의
관리자(Conceptual)	BV2. 업무기술	DV3. 데이터 기술	TV2. 기반구조 기술
설계자(Logical)	BV3. 업무설계	DV4. 데이터 설계	TV3. 기반구조 설계

위의 요구사항분류 매트릭스상의 각 셀들은 정의, 기술, 설계에 해당하는 구성요소들을 갖고 있으며, Data View의 경우에는 Contextual수준에서 이해관계자 정의와 데이터 정의의 2가지 요소를 갖고 있지만, 대부분의 경우에는 셀별로 1개 정도의 구성요소를 갖고 있다. 각 셀의 구성요소는 Contextual 수준의 경우 정의, Conceptual수준의 경우 기술, Designer수준의 경우에는 설계로 정의 되었다.

요구사항 분류 매트릭스 상의 각각의 셀들에 해당하는 요구사항들은 아래와 같다.

표 8 프레임워크 구성요소 설명

구분	설명
BV1. 업무정의	소프트웨어가 수행하여야할 기본적인 비즈니스에 대하여 정의한다. 세부적인 기능 전체를 서술하는 것이 아니라, 개괄적인 요구사항을 기술한다. - 소프트웨어의 필요성, 기대효과, 구축 전략 등
BV2. 업무기술	소프트웨어가 수행하여야할 기능 및 프로세스에 대한 요구사항들을 기술한다. 이때 소프트웨어를 사용하지 않고 수행되는 활동에 대해서도 기술하여야 한다. - 기능 요구사항, 프로세스 요구사항
BV3. 업무 설계	BV2에서 정의된 소프트웨어의 기능적인 요구사항 및 개별 기능들간의 연관관계에 대한 세부사항을 기술한다. - 기능상세, 기능간 연관관계 및 선후관계 등
DV1 이해관계자 정의	소프트웨어의 사용자 및 관리자 등 개발 이후의 이해관계자들을 정의 한다. 매트릭스상의 Perspective와는 달리 실제 프로젝트에서 파악된 사항을 기술한다. - 이해관계자 분류 및 특성
DV2. 데이터 정의	소프트웨어에서 다루고자 하는 정보에 대한 개괄적인 주제영역 및 데이터 그룹을 정의한다. - 데이터 주제영역 구성, 데이터 그룹
DV3. 데이터 기술	소프트웨어에서 관리, 활용되는 데이터구조에 대해 기술한다. 이때 BV2. 업무기술에서 정의된 기능과의 연관관계(CRUD)를 기술하여야 한다. - 주요 데이터, 엔티티 후보 키워드
DV4. 데이터 설계	소프트웨어 설계시 논리데이터 모델을 작성하기 위한 기반 자료를 명세한다. 데이터 엔티티들을 정의하고 엔티티간의 관계 및 카디날리티를 기술하여야 한다. - 논리데이터모델(ERD), 엔터티 목록
TV1. 기반구조 정의	소프트웨어의 물리적 제약사항을 기술한다. - 필요응답시간, 하드웨어/소프트웨어 제약사항 등
TV2. 기반구조 기술	소프트웨어 구축시 사용될 아키텍처 요구사항을 정리하여 기술한다. - 물리적 아키텍처, 소프트웨어 아키텍처, 기술분류
TV3. 기반구조 설계	사용될 기반구조(하드웨어,소프트웨어, 적용기술)에 대한 상세한 내용을 기술한다. - 적용 하드웨어, 소프트웨어, 적용기술 표준

매트릭스는 단순히 구성요소들의 정의뿐 아니라 각 구성요소들 간의 연관관계(Relation)를 정의하여야 한다. 이때 연관관계는 동일한 시각(Perspective)에 속하는 요구사항들 간에 맺어지는 것을 원칙으로 한다.

4. 요구사항분류 매트릭스 적용 사례

4.1 적용분야 선정 및 적용방법

공공기관에서 실제 수행중인 “정보화사업관리” 구축사업의 요구사항을 대상으로 요구사항 분류 매트릭스를

적용하여 그 효과를 측정해 보았다. 요구사항의 수집 및 관리라는 측면에서 기존 프로젝트에서 일반적으로 사용되는 "요구사항정의서"를 통해 수집된 요구사항과 일부 "화면설계서"의 내용을 본 논문에서 제시한 요구사항분류 매트릭스에 매핑하여 요구사항 수집의 적정성을 검토하였으며, 요구사항의 변경 발생시 연관관계를 추적하여 영향도를 측정였다.

4.2 적용결과 및 성과

기존에 수집된 요구사항들을 분류 매트릭스를 통해 재정리해 본 결과, 전체적인 요구사항들에 대한 MECE적인 분류가 가능하여, 기존에는 분석되지 못한 기술적인 이슈(웹접근성 이슈 등)에 대한 부분이 추가적으로 파악할 수 있었다. 이러한 부분은 기존의 방법을 이용하게 되면, 프로젝트 설계완료 시점 혹은 개발시점에 이슈화가 될 수 있는 부분을 미리 발견한 사례라고 할수 있다.

또한 수집된 요구사항들이 매트릭스를 통해 한 화면에 조회될 수 있기 때문에, 요구사항 수집 단계에서 전체적인 수집 진도를 파악하거나, 요구사항이 좀더 파악되어야 하는 부분에 대해 쉽게 파악될 수 있었다. 예를 들어 BV2에서는 30여 가지의 기능정의 내용이 채워져 있으나, DV2에는 20여개의 데이터만이 담겨져 있었기 때문에, 기술인프라 부분에 대한 요구사항이 추가적으로 수집되어야만 한다는 것을 판단할 수 있었고, 추가 요건정의를 수행하였다.

또한 각 관점/시각별로 요구사항이 정리됨으로써, 업무 / 데이터/ 기반구조에 대해 유사한 부분들에 대한 의사결정자(CEO/CIO)들의 시각에서부터 개발자들의 시각이 정리되어 한눈에 상/하간의 연관관계를 분석할 수 있는 효과를 제공하였다.

특히 이러한 연관관계에 대한 관리는 향후 요구사항 관리도구(tool)과의 연계를 통해서 시스템적으로 관리된다면, 요구사항 변경 발생시 파급되는 영향도를 분석하여 효과적인 의사결정을 위한 자료를 제공할 수 있을 것이다.

5. 결론

지금까지 요구사항을 수집하기 위한 체계를 수집하기 위하여 다양한 이해관계자들의 시각과 요구사항의 대상에 따른 요구사항 분류 프레임워크를 작성하고 적용하여 보았다. 요구사항분류 매트릭스는 기존의 SRS를 이용한 요구사항 관리에 비하여 보다 시각적이고, 체계적으로 요구사항을 관리할 수 있는 장점이 있으며, 인터뷰 등을 통한 요구사항 수집 시 누락되거나 중복된 부분을 쉽게 파악할 수 있었다. 특히 요구사항들 간의 연관관계를 통해서 요구사항 변경요인 발생시 영향도를 파악하고, 변경내용의 타당성에 대한 의사결정에 도움을 줄 수 있었다.

그러나, 요구사항분류 매트릭스는 요구사항의 수집 및 관리에 관한 구체적인 절차까지는 제공하지 못하고 있다. 즉 What에 대한 부분을 중심으로 정의하였기에, How에 해당하는 요구사항 관리 프로세스에 대한 부분은 명확히 정의되지 않았다. 추후 요구사항분류 매트릭스에 기반하여 요구사항 수집 및 관리에 대한 표준 프로세스를 정의함으로써, 요구사항 수집 및 관리 방법론으로까지 발전시킨다면, 요구공학 및 소프트웨어 개발 분야의 유효한 도구로서 활용될 수 있을 것으로 전망한다.

6. 참고문헌

[1] The Standish Group, Chaos Chronicles III, www.standishgroup.com, 2003

[2] 고영중외3인, 요구 사항 문장 범주화를 이용한 웹 기반의 요구 사항 추출 지원 시스템, 정보과학회논문지, 제27권, 제4호, 384~385p, 386-388p, 2000

[3] Ian Sommerville and Pete Sawyer, Requirements Engineering, Wiley, 1997 (상기 논문[2]에서 재인용)

[4] IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications, http://standards.ieee.org, 1998

[5] Renaud Lecoeuche, Chris Mellish, and Dave Robertson, "A Framework for Requirements Elicitation through Mixed-Initiative Dialogue", International Conference on Requirement Engineering(ICRE '98), 1998

[6] 한국전산원 이승한,신신애,이헌중, 정보기술아키텍처(ITA) 기반의 전자정부구현, 정보과학회지 제22권 제11호, 2004

[7] Zachman, John. A. "A Framework for Information Systems Architecture", IBM System. J. ([9]에서 재인용)

[8] Zachman Framework, http://www.zifa.com/

[9] 김덕현, Enterprise Architecture 기반의 전자상거래 표준화 프레임워크, 한국전자거래학회 학술대회 발표집, 2002

[10] Chief Information Officers Council, "Federal Enterprise Architecture Framework(Version 1.1)", 1999

[11] 한국정보통신기술협회, 정보통신단체표준 TTAS.KO-10.0153 "공공부문 전사적 아키텍처 프레임워크 표준", 2003

[12] 인호,김상수,이병정,박수용, 미래의 소프트웨어 공학기술, 정보과학회지 제24권 제12호, pp 6-12, 2006

[13] 황만수, 요구공학 기반이 통합된 소프트웨어 요구사항 관리 프레임워크 설계, 숭실대학교 박사학위 논문, 2001

[14] Soren Lauesen, Software requirement: Styles and techniques, Addison-Wesley, 2001, pp 13

[15] Boehm, B., Egyed, A., Port, D., Shah, A., Kwan, J., and Madachy, R., "A Stakeholder Win-Win Approach to Software Engineering Education", Annals of Software Engineering, 1999