韓國 OA學會 論文誌
第6卷 第4號, 2001. 12.

2001-6-4-1-5

# 요구사항 관리범위 확대를 위한 명세화 개선방안

신 종 철*

# Improving Requirements Specification to extend Requirements Management over the Development Life Cycle

Jong-Cheol Shin*

## 요 약

전통적 개발방법론에서는 요구분석단계에서 요구사항들이 한번 정리되면 이들은 개발이 완료될 때까지 변경이 없을 것으로 간주한다. 그러나 현실적으로는 개발기간 중 목표 시스템의 구체화, 정보기술의 발전, 적용환경의 변화 등으로 인하여 요구사항은 끊임없이 변화하게 된다.

본 논문에서는 개발방법론에서 미흡한 요구사항 관리를 개선하기 위하여 요구사항의 관리를 전체 개발 생명주기로 확대하고, 지속적으로 발생하는 요구사항의 변경을 효율적으로 관리할 수 있는 요구사항의 명세화 방안을 제시한다.

## Abstract

In the conventional development methodologies, requirements are considered to be not changing after the analysis phase. But in the real world, requirements can be changed and modified through out the development life cycle according to end-user's more understanding about the target system, new IT technologies, changes of customer environment and market situation, and so on.

In this paper, an idea of improving the requirements specification to extend requirements management over the whole development life cycle is proposed to resolve the requirements management problem of development methodologies.

* 송우아이엔티(주) 기술연구소장

# Ⅰ. 서론

현재 널리 적용되고 있는 구조적 개발방법론에서는 요구사항의 추출 및 명세화가 분석단계에서 이루어지고, 다음 단계인 설계 및 구현을 위한 기준으로 사용된다. 따라서 한번 명세화된 요구사항들은 적어도 개발단계 내에는 변경되지 않을 것이라는 가정을 바탕으로 한다. 그러나, 현실적으로는 사용자가 개발될 시스템을 좀 더 이해하게 됨에 따라 사용자의 요구사항이 변경되거나 새로운 요구사항이 추가되는 경우, 혹은 정보기술의 급속한 발전으로 인하여 개발기간 중 적용기술이나 서비스를 변경해야 하는 경우, 그리고 외부적인 시스템의 적용환경 변화에 따른 변경 요구 등 요구사항의 변경 요인이 끊임없이 발생한다. 그리고 설계단계 이 후에 요구사항의 변경이 발생하면 프로젝트 관리뿐만 아니라 개발기간과 비용에도 심각한 영향을 초래하여, 외주 개발의 경우에는 발주자와 개발자 사이의 분쟁을 유발하는 원인이 되기도 한다.

따라서 본 논문에서는 개발방법론에서 미흡한 요구사항의 변경과 관리를 효율적으로 지원하기 위하여 각각의 요구사항을 별도의 관리대상으로 설정하고, 개발생명주기 동안 요구사항의 지속적인 변경과 보완이 가능하며, 변경의 영향을 곧 바로 파악할 수 있도록 요구사항 관련 산출물을 요구사항 기술서, 요구사항 변경이력서, 요구변경 영향분석서의 3 가지로 구분하여 명세화하고, 이들을 활용하여 요구사항을 관리하는 방안을 제시하며 개발방법론과 프로젝트 관리 측면의 적용효과를 분석한다.

본 논문의 제 2 장에서는 요구공학 및 개발방법론에서의 요구사항 명세화 방법을 분석하고, 제 3 장에서는 개발방법론에 통합 적용이 가능한 요구사항 명세화의 개선방안을 제시한다. 그리고 제 4 장에서는 제안된 요구사항 명세화 개선 및 관리방안의 적용 효과를 분석한다.

# Ⅱ. 관련 연구

## 2.1 개발방법론의 요구분석 절차와 산출물

ISO 12207에서는 소프트웨어의 생명주기동안 수행되어야 할 여러 가지 활동들을 5개의 기본 공정(Primary Process)과 8개의 지원 공정(Supporting Process), 그리고 4개의 조직 공정(Organizational Process)로 구분한다. 그리고 기본공정 중 개발공정에서 공정 구현, 시스템 요구사항분석, 시스템 구조설계, 소프트웨어 요구사항분석, 소프트웨어 구조설계, 소프트웨어 상세설계, 소프트웨어 구현 및 시험, 소프트웨어 통합, 소프트웨어 자격시험, 시스템 통합, 시스템 자격시험, 소프트웨어 설치, 소프트웨어 인수지원 등 소프트웨어의 개발과 관련된 13개의 주요 활동을 정의하고 있다[13]. 여기서 요구사항 분석활동은 시스템 요구분석과 소프트웨어 요구분석으로 구분되며, 이들은 각각 요구명세 작성과 요구사항 평가로 세분화된다.

마르미방법론에서는 요구분석 단계를 단계 준비, 사용자 요구사항 정의, 프로세스 모형 구성, 프로세스/엔티티 연관분석, 인터페이스 분석, 엔티티 모형구성, 분산 모형구성, 테스트 요건 정의, 단계 점검의 9개 활동으로 구분한다[9]. 이들 활동 중에서 사용자의 요구사항을 명세화하는 작업은 사용자 요구사항 정의활동 단계에서 이루어지며, 이 사용자 요구사항 정의활동은 사용자 요구사항 정리, 현행 시스템 분석, 사용자 요구사항 분석의 3개 세부 업무로 구분된다. 그리고 이 단계에서 작성되어야 할 산출물로는 양식명세서, 양식 총괄표, 업무기술서, 면담 및 회의록, 현행 데이터베이스 기술서, 현행 어플리케이션 기술서, 현행 시스템 구성도, 현행 시스템 사양서, 현행 시스템 사양 설명서, 현행 업무 절차서, 요구사항 기술서, 요구사항 분석서, 요구사항 추적표 등이 있다.

관리기법/1에서는 프로젝트의 규모, 기술적 범위, 업무절차 등을 고려하여 정보계획 수립, 클라이언트/서버 시스템 개발, 호스트 시스템 개발, 패키지 시스템 개발, 소규모 프로젝트 개발, 고속 개발 등으로 개발경로를 구분하는데[9], 클라이언트/서버 개발경로가 대규모 프로젝

트를 기준으로 전체 단계를 포함하고 있으므로 이를 기준으로 사용한다. 이 개발경로에서는 사용자 요구사항 세그먼트에서 요구사항들을 명세화하게 되는데, 이 사용자 요구사항 세그먼트는 작업흐름 및 조직 확인, 사용자 요구사항 파악, 현행설계 복구의 3개 타스크로 구분된다. 그리고 이 세그먼트에서 작성되어야 할 산출물은 엔티티 관계도, 작업 흐름도, 조직도, 조직별 프로세스, 면담 비망록, 요구사항 설명, 현행 시스템 설명, 관계유형 설명, 엔티티 유형설명, 속성 유형설명, 요구사항 설명 등이 있고, 적용도구 및 기법으로는 면담과 JAD 등이 있다.

구조적 개발방법론인 ISO 12207, 관리기법/1, 그리고 마르미방법론에서는 공통적으로 파악된 요구사항들을 별도의 관리대상으로 설정하지 않고, 후속의 요구사항 명세서나 설계서 작성을 위한 중간 단계로 활용한다. 따라서 분석된 요구사항 중에서 오류가 발생하거나 누락된 요구사항이 발생할 경우에는, 후속의 개발단계에서는 역으로 추적하거나 확인할 수 있는 원천 정보를 상실함으로써 검증 및 변경을 위한 근거를 잃어버리게 되는 문제점이 있다. 그리고 요구분석이 완료된 다음에는 이후의 개발기간 동안 요구사항이 변경되지 않을 것이라는 가정을 공통적으로 적용하고 있어, 실제 프로젝트에서 전체 개발 노력(비용)과 기간의 80% 이상[11, 12]을 차지하는 나머지 개발단계에서 발생할 수 있는 요구사항의 변경을 체계적으로 수용할 수 있는 절차가 미흡한 문제점이 있다.

### 2.2 시스템 요구분석과 소프트웨어 요구분석

ISO 12207을 바탕으로 IEEE 표준에서는 시스템 요구명세서 작성지침과 소프트웨어 요구명세서 작성 권고안을 제시하고 있는데, 시스템 요구명세서(SyRS : System Requirements Specification) 작성을 위한 지침은 IEEE Std 1233, 1998 Edition[2]에서, 그리고 소프트웨어 요구명세서(SRS : Software Requirements Specification) 작성을 위한 권고사항은 IEEE Std 830-1998[1]에서 기술하고 있다. IEEE Std 1233는 개발될 시스템이 만족시켜야 할 명시적 요건들을 요구사항으로 개발하는 지침을 제공하며, 시스템 요구사항의 수집이 만족시켜야 할 속성으로 유일성, 정규성, 연결성, 완전성, 일관성, 구분성, 변경 가능성, 구성 관리, 추상화 수준 등을 제시한다. 그리고 IEEE 830은 좋은 소프트웨어 요구사항 명세서(SRS)가 갖추어야 할 내용과 품질을 기술하고 소프트웨어 요구사항의 명세화를 위한 접근 권고안을 제공하며, 소프트웨어 요구사항 명세서가 만족시켜야 할 특성으로 정확할 것, 애매 모호하지 않을 것, 완전할 것, 일관성이 있을 것, 중요도 / 안정성에 따른 등급화, 검증 가능할 것, 수정이 용이할 것, 추적 가능할 것 등을 제시한다.

그러나 IEEE의 시스템 요구명세서(SyRS) 및 소프트웨어 요구명세서(SRS)에서는 요구사항을 시스템 요구사항과 소프트웨어 요구사항으로 구분함으로써,

첫째, 시스템 요구사항을 추출하고, 이를 바탕으로 시스템 중에서 소프트웨어 부분의 요구사항을 추출함으로써 사용자 관점의 개별 요구사항을 도출하기 어렵고,

둘째, 시스템 요구명세서 및 소프트웨어 요구명세서는 시스템 전체적인 측면을 중심으로 기술하기 때문에 각각의 요구사항을 별도의 관리대상 객체로 설정하지 않으며,

셋째, 요구사항의 변경에 대한 체계적인 처리절차가 수립되어 있지 않은 문제점이 있다.

### 2.3 요구사항 명세화 템플리트

개별 요구사항을 관리대상 객체로 설정하기 위해서는 먼저 각각의 요구사항에 대한 독립적인 명세화가 필요하다. 이러한 요구사항의 독립적인 명세화에는 요구사항 템플리트가 사용될 수 있는데, 대표적인 템플리트로는 Volere 요구명세 템플리트와 관리기법/1의 '요구사항 설명' 템플리트 등이 있다.

Volere 요구명세 템플리트(Requirements Specification Template)는 각각의 요구사항에 대하여 일련번호를 부여하고, 정해진 형식에 따라 카드 형태의 셸(shell)에 기록하여 관리함으로써 개별 요구사항의 객체화를 지원한다. 관리의 편의를 위하여 요구사항을 요구형식(Requirement Type)에 따라 구분하는데, 요구형식은 크게 프로젝트 구동요소, 프로젝트 제약사항, 기능적 요구사항, 비기능적 요구사항, 프로젝트 쟁점사항의 5가지 종류로 구분하고 다시 27가지의 세부 형식으로 분류한다. 그리고 이 템플리트의 구성 항목은 요구사항 번호, 요구사항 형식, 사건 / 유스케이스 번호, 내용 설명, 근거, 출처, 적용 기준, 고객 만족도, 고객 불만정도, 의존 관계, 상충되는 요구사항, 관련 문서, 이력 등으로 이루어진다[4].

관리기법/1에서는 사용자 요구사항을 파악하여 '요구사항 설명'을 작성하도록 오브젝트 템플리트를 정의하고 있으나[10], 필수 산출물이 아니라 다음 단계의 작업을 위

한 중간 산출물로 사용한다. 이 '요구사항 설명'은 요구사항을 정의하고, 응용이 무엇을 어떻게 하는지를 정의하며, 기능이 충족되었을 때의 효과와 기능 요구사항으로 인해 프로젝트팀에 부과되는 제약사항을 파악하는데 사용된다. 이 템플리트에 포함되는 구성 항목들은 요구사항 ID, 정의, 유형, 현황, 목표, 순차이, 효과, 제약, 중요성, 이해 관련자, 후원자, 변경 방법 등이 있다.

## Ⅲ. 요구사항 명세화의 개선

### 3.1 개선의 필요성

시스템 개발의 모든 생명주기에서 새로운 요구사항이 생겨나고 기존의 요구사항에 대한 변경이 발생할 수 있으며, 전체 요구사항의 50% 이상이 실제 서비스 단계로 구현되기 전에 변경되는 경우가 자주 발생한다[3]. 그런데 대부분의 요구사항 변경은 경제상황이나 시장환경의 변화, 경쟁제품의 등장 및 법률과 규정의 변경 등과 같은 외부 환경의 변화로 인하여 발생하는 경우가 많고, 요구사항이 변경되면 시스템의 설계와 구현이 변경되어야 하며, 이러한 요구사항의 변경은 시스템 개발기간 내내 계속될 수 있다. 그리고 James Martin에 의하면 그림 1과 같이 개발 프로젝트에서 발생하는 절반 이상의 결점이나 오류가 요구사항과 관련 있는 것으로 알려져 있으므로[5], 요구사항의 관리범위 확대와 변경관리 방안이 필요하다.

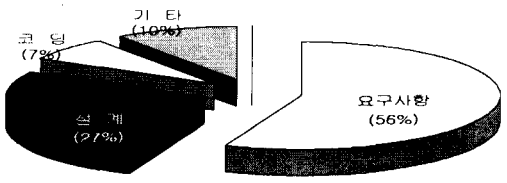


그림 1. 프로젝트 결점/오류 발생원인

그러나 기존의 개발방법론에서는 요구사항의 명세화가 여러 산출물로 분산되어 이루어지기 때문에 요구사항의 집중 관리가 불가능하다. 그리고 적용하는 개발방법론이나 대상 프로젝트의 규모와 특성에 따라 프로세스가 서로 다르므로, 본 논문에서는 명세화를 개선하기 위한 산출물의 정의를 통하여 요구사항의 관리범위를 확대하고 변경관리를 지원하는 방안을 제시한다.

### 3.2 산출물의 정의

기존의 개발방법론이나 요구사항 명세화 기법들의 문제점을 개선하기 위하여 새로운 요구사항 명세화 산출물들이 갖추어야 할 요건들은 다음과 같다.

첫째, 요구사항의 관리범위를 전체 개발생명주기로 확대하기 위하여 각각의 요구사항을 별도의 관리대상 객체로 설정하도록 지원하고,

둘째, 요구사항의 변경관리를 위하여 변경에 따른 영향분석과 변경이력의 관리를 지원하며,

셋째, 개발방법론에 대한 용이한 적용을 위하여 최소한의 산출물과 작성 노력을 요구하여야 한다.

이러한 요건들을 만족시킬 수 있도록 본 논문에서는 필요한 산출물을 요구사항 기술서, 요구사항 변경이력서, 요구변경 영향분석서의 세 가지로 정의한다. 제 2.3 절의 요구사항 템플리트를 확장한 요구사항 기술서를 사용하면 개별 요구사항의 명세화 및 관리대상 객체화가 가능하고, 요구사항의 변경은 요구사항 기술서의 수정과 변경이력 관리 및 영향분석으로 명세화가 가능하기 때문에, 이들 세 가지 산출물을 통하여 요구사항 명세화의 개선이 가능하다고 판단할 수 있다.

### 3.3 산출물의 구성

요구사항 기술서는 하나의 요구사항을 단위로 작성하되, 비전문가인 사용자도 쉽게 이해할 수 있도록 자연어로 작성하고, 각각의 요구사항을 하나의 관리단위(객체)로 설정한다. 이 객체가 가지는 주요 속성은 요구사항 번호, 요구사항 종류, 요구사항 설명, 출처 및 근거, 관련자, 중요도, 다른 요구사항과의 관계, 변경 이력, 충족 요건, 기대 효과, 참조 문서, 작성 일자 등이 있다. 그런데 요구사항의 종류는 제 2.3 절의 Volere 요구명세 템플리트처럼 너무 세분화하면 요구사항들이 중첩되거나 애매한 요구사항이 발생하면 처리가 곤란한 문제점이 있으므로, 기능, 성능, 제약조건, 인터페이스, 희망사항 등의 대분류로 구분한다. 그리고 각각의 요구사항에는 필요시 표나 테이블, 혹은 다이어그램 등을 첨가하여 보충적인 설명을 기술하며, 기술방식은 가급적 수치화하여 모호성을 배제하고, 적용하는 프로젝트의 규모나 성격에 따라 조정하여

적용할 수 있도록 한다.

요구사항 변경이력서는 분석단계 이후에 발생하는 요구사항의 변경을 체계적으로 관리하기 위하여 작성한다. 변경은 추가, 변경, 삭제로 구분하며, 변경이 필요한 이유와 변경으로 인하여 영향을 받는 다른 요구사항들의 목록을 명시한다. 관리의 편의를 위하여 추가, 변경, 삭제된 요구사항별로 테이블 형식의 별도 목록을 유지할 수 있으며, 요구사항 변경이력서에 포함될 항목은 요구사항 번호, 변경 일자, 변경 내역, 변경 구분, 변경 사유, 영향을 미치는 요구사항 목록, 담당자 등이 있다.

요구변경 영향분석서는 추가, 변경, 혹은 삭제된 각각의 요구사항에 대하여 관련되는 요구사항 목록을 명시하고, 이로 인하여 발생할 수 있는 영향범위를 기술적 측면과 관리적 측면으로 나누어 기술한다. 기술적 측면에는 직접 혹은 간접으로 영향을 미치는 요구사항과 시스템의 범위와 영향 정도를 기술하고, 관리적 측면에는 프로젝트의 진행에 영향을 미치는 소요 인력, 비용, 기간 등을 기술한다. 실제로는 정확한 영향범위를 평가하기 힘들고, 간접적인 영향까지 분석하기는 매우 어렵기 때문에 파악 가능한 범위에서 기술적 영향범위와 추가되는 인력, 비용, 기간을 산정하여 기술하고, 프로젝트 관리자와 발주자측의 검토를 거치도록 한다. 요구사항 영향분석서에 포함될 항목은 요구사항 번호, 변경 영향범위, 예상 소요비용, 예상 소요기간, 예상 소요인력, 위험요소 등이 있다.

표 1. 개발작업 단계별 산출물

| 개발 단계 | 담 당 자 | 산 출 물 | 비 고 |
|---|---|---|---|
| 제안요청서 준비 | 시스템기획자 사용(예정)자 | 요구사항 기술서 (개략) | |
| 제안서 작성 | 개발자 | 요구사항 기술서 (보완) | |
| 분 석 | 개발자 (사용자) | 요구사항 기술서 (보완)<br>요구사항 변경내역서 (필요시)<br>요구변경 영향분석서 (필요시) | 요구사항 기준선 설정 |
| 설 계 | 개발자 (사용자) | 요구사항 기술서 (보완)<br>요구사항 변경내역서 (필요시)<br>요구변경 영향분석서 (필요시) | |
| 구 현 | 개발자 (사용자) | 요구사항 기술서 (보완)<br>요구사항 변경내역서 (필요시)<br>요구변경 영향분석서 (필요시) | |
| 테스트 | 개발자 사용자 | 요구사항 기술서 (보완)<br>요구사항 변경내역서 (필요시)<br>요구변경 영향분석서 (필요시) | |
| 설치 및 인도 | 개발자 사용자 | 요구사항 기술서 (재정립) | 요구사항 기준선 재설정 |

### 3.4 산출물의 적용

앞 절에서 정의한 산출물들을 개발방법론에 통합 적용하기 위하여 개발작업 단계별로 작성할 산출물의 종류를 표 1과 같이 제안한다.

표 1에 의하여 개발작업의 진행 흐름에 따른 각 산출물의 작성 방법과 시기를 아래와 같이 적용할 수 있다.

(1) 요구사항 기술서 : 제안요청서 준비단계에서 발주자의 의도와 사용(예정)자의 요구사항을 수집하여 목표 시스템이 만족시켜야 할 기본적인 요구사항들을 개략적으로 기술하며, 제안서 작성단계에는 개발될 목표 시스템의 개념 모델을 구체화하기 위하여 제안요청서에서 발주자가 제시한 요구사항들을 보완하고 구체화하여 요구사항 기술서를 보완한다. 일반적으로 제안서는 발주자와 개발자간 계약 체결의 기준으로 사용되므로, 개발자는 구현될 목표 시스템을 구체화할 수 있는 수준까지 요구사항 기술서를 보완한다. 그리고 분석단계에는 개발자가 요구분석 과정을 통하여 시스템이 만족시켜야 할 모든 요구사항들을 구체적으로 정리하여 요구사항 기술서를 보완하며, 이 문서는 시스템 개발의 전체 생명주기에서 요구사항의 기준선으로, 그리고 다음 설계단계의 검증을 위한 기준으로 사용될 수 있다. 나머지 설계, 구현 및 테스트 단계에서는 진행과정에서 발견되는 요구사항의 문제점이나 사용자의 변경요구 등을 반영하여, 필요한 경우 요구사항 기술서를 보완한다. 또한 설치 및 인도단계에서 최종적으로 확정되는 요구사항 기술서는 개발된 시스템이 발주자에게 인도되어 운영 및 유지보수단계로 전환되는 시점을 기준으로 한 시스템의 새로운 요구사항 기준선으로 사용될 수 있다.

(2) 요구사항 변경내역서 : 요구사항의 변경이 발생하여 이미 작성된 요구사항 기술서에 대한 수정 및 보완이 필요한 경우, 변경되는 각각의 요구사항에 대하여 변경내역서를 작성한다. 이 산출물은 소스 코드처럼 정형화하기 어렵고 체계적인 버전 관리가 어려운 요구사항의 특성을 고려하여 각 개발단계에서 유효한 요구사항들을 파악하고 요구사항의 변경을 추적하기 위하여 사용한다. 그리고 작성시기는 분석단계에서 요구사항의 기준선이 설정된 이후에 설계, 구현, 혹은 테스트 단계에서 요구사항의 변경이 발생하면 작성하고, 변경된 각각의 요구사항에 대하여 요구변경 영향분석서를 함께 작성한다.

(3) 요구변경 영향분석서 : 요구사항의 변경이 발생하는 경우, 요구사항 변경이력서와 함께 작성한다. 여기에

는 요구사항의 변경으로 인한 기술적 영향뿐만 아니라 비용이나 프로젝트 관리와 같은 관리적 측면까지 파급효과와 영향을 분석하여 기술한다. 이 산출물은 발주자와 개발자간의 요구사항 변경으로 인한 영향에 대한 공통의 이해를 증진하기 위하여 사용한다. 그리고 작성시기는 분석단계에서 요구사항의 기준선이 설정된 이후에 설계, 구현, 혹은 테스트 단계에서 요구사항의 변경이 발생하면 각각의 요구사항 변경에 대하여 작성한다.

## Ⅳ. 평가 및 검토

IEEE의 SyRS와 SRS는 시스템 요구사항과 소프트웨어 요구사항을 전체 시스템적 관점에서 기술하고, 시스템 요구사항 수집이 만족시켜야 할 속성, 요구사항의 범주화 기준, 소프트웨어 요구명세서가 만족시켜야 할 특성 등을 규정하고 있지만, 분석단계에서 설계단계로 이행하기 위하여 수행하는 일회성 과정으로 정의하고 있다. 반면에 본 논문에서 제안하는 요구사항 명세화 개선방안은 요구사항을 전체 개발생명주기에 걸쳐 점진적으로 상세화하거나, 변경 요구를 체계적으로 수용하고 변경으로 인한 영향을 곧 바로 파악할 수 있도록 각각의 요구사항을 관리대상 객체화할 수 있는 장점이 있다.

표 2. 개발방법론과의 비교 평가

| 구 분 | 기존 개발방법론 | 제안 명세화 개선방안 |
|---|---|---|
| 요구사항 관리 범위 | 요구분석단계로 한정 | 전체 개발생명주기로 확대 |
| 요구사항의 점진적 명세화 | 요구분석단계에서 요구사항 확정 | 제안요청서단계부터 점진적인 요구사항의 명세화 가능 |
| 설계단계 이후의 요구사항 변경 | 지원하지 않음 | 지원 |
| 개별 요구사항의 관리대상 객체화 | 지원하지 않음 | 설정 가능 |
| 요구사항의 집중 관리 | 지원하지 않음 | 요구사항을 전체 생명주기에 걸친 관리대상 객체로 설정 |
| 요구사항의 범주화 및 일관성 검증 | 지원하지 않음 | 지원하지 않음 |
| 요구사항의 추적성 | 지원하지 않음 | 요구사항 변경이력서를 통한 부분적 지원 가능 |
| 요구사항의 변경 영향 분석 | 지원하지 않음 | 지원 (요구사항 변경영향 분석서) |
| 형상관리와의 연계 | 지원하지 않음 | 요구사항의 관리대상 객체화로 형상관리 항목으로 적용 가능 |

그리고 전통적 개발방법론에서는 요구사항을 주요 관리대상으로 취급하지 않고 설계를 위한 중간단계로 사용하고, 분석단계에서 요구사항이 한번 명세화되면 이후의 개발기간동안에는 변경되지 않을 것이라는 가정을 적용하여 요구사항의 변경을 지원하는 절차가 미흡하다. 그런데 제안 요구사항 명세화 개선방안을 개발방법론에 통합 적용하면 표 2에서 보는 바와 같이, 개발방법론에서 미흡한 요구사항의 관리범위 확대와 점진적 명세화, 추적성, 설계단계 이후의 요구사항 변경과 이로 인한 영향분석, 그리고 형상관리와의 연계 등을 개선할 수 있다. 그러나, 요구사항의 범주화나 일관성 검증과 같은 요구사항 자체의 품질향상에는 도움이 되지 못한다.

Standish Group 보고서[6]에 의하면 정보기술 응용개발 프로젝트의 지연 및 비용초과의 원인 중에서 36.9%가, 그리고 프로젝트 중단 및 취소 원인 중에서 34.2%가 요구사항과 직접적인 연관성을 가진다. 그런데 프로젝트 지연 및 비용초과의 원인 중에서 요구사항과 관련된 부분은 ① 사용자 의견수렴 부족 (12.8%), ② 불완전한 요구사항 및 명세서 (12.3%), ③ 요구사항 및 사양서 변경 (11.8%), ④ 비현실적 기대치 (5.9%) 등이 있으며, 특히 ① ② ③은 요구사항의 관리와 직접적인 연관관계를 가진다. 따라서 이 논문에서 제안하는 바와 같이 요구사항 명세화 방법을 개선하여 개발 프로젝트의 전체 생명주기에서 적용하면, 개발 초기단계부터 사용자의 참여를 유도하고 개발자와 사용자 사이에 공통의 이해를 위한 기반을 제공함으로써 '① 사용자 의견수렴 부족'의 문제를 개선할 수 있고, 요구사항을 점진적으로 수정하고 보완하여 요구사항 및 사양서의 완성도를 향상시킴으로써 '② 불완전한 요구사항 및 명세서'의 문제를 개선할 수 있으며, 요구사항 변경의 체계적인 관리와 변경으로 인한 영향분석을 통하여 '③ 요구사항 및 사양서 변경'의 문제를 개선할 수 있으므로, 프로젝트 지연 및 비용초과의 원인 중에서 36.9%에 해당하는 부분에 대한 개선효과를 기대할 수 있다.

그리고 프로젝트 중단 및 취소의 원인 중에서 요구사항과 관련된 부분은 ① 불완전한 요구사항 (13.1%), ② 사용자 참여 부족 (12.4%), ③ 비현실적 기대치 (9.9%), ④ 요구사항 및 사양서 변경 (8.7%), ⑤ 필요성 상실 (7.5%) 등이 있으며, 특히 ① ② ④는 요구사항 관리와 직접적인 연관관계를 가진다. 이 부분에 대해서도 위에서 기술한 프로젝트 지연 및 비용초과 원인에 대한 분석과 동일한 근거에 의하여 프로젝트 중단 및 취소의

원인 중에서 34.2%에 해당하는 부분에 대한 개선효과를 기대할 수 있다.

## V. 결 론

본 논문에서 제안하는 요구사항 명세화 방안은 요구사항의 추출과 관리를 전체 개발생명주기로 확대하기 위하여 요구사항을 객체화하고 지속적으로 발생할 수 있는 요구사항의 변경을 지원함으로써, 개발방법론에서 미흡한 요구사항의 관리를 개선하고 관리범위를 확대할 수 있다. 따라서 개발 프로젝트에서 발생하는 절반(56%) 이상의 결점과 오류가 요구사항과 관련되고[5], 프로젝트 지연 및 비용초과나 중단 및 취소 원인의 35% 이상이 요구사항과 관련이 있음[6]을 고려할 때, 본 논문에서 제안하는 요구사항 명세화 개선방안은 개발 프로젝트의 결점과 오류를 감소시키고 프로젝트 관리측면의 개선을 통하여 시스템 개발의 품질과 생산성 향상에 도움이 될 것으로 기대한다.

그러나 본 논문의 연구범위에는 요구사항의 일관성 검증[7]이나 변경 시 영향 분석을 위한 요구사항 관리 모형에 대한 연구와, 최근 프로젝트에 적용이 확산되고 있는 객체지향 개발방법론과 관련된 연구가 포함되지 않았다. 따라서 향후 이 부분에 대한 추가적인 연구와, 요구사항의 변경 시 소프트웨어 개발비 산정기준과 같이 체계적으로 변경으로 인하여 추가적으로 소요되는 기간, 인력, 비용 등을 추산할 수 있는 모델에 대한 연구를 계속할 예정이다.

## 참고문헌


[1] ANSI/IEEE Std 830-1998, *IEEE Recommended Practice for Software Requirements Specifications*, IEEE, NY, 1998. Reprinted in Software Requirements Engineering (2nd Edition), pp. 207-244, edited by Richard H. Thayer and Derlin Dorfman, IEEE Computer Society Press, Los Alamitos, CA, 2000

[2] ANSI/IEEE Std 1233 (1998 Edition), *IEEE Guide for System Requirements Specifications*, IEEE, NY, 1998. Reprinted in Software Requirements Engineering (2nd Edition), pp. 245-280, edited by Richard H. Thayer and Derlin Dorfman, IEEE Computer Society Press, Los Alamitos, CA, 2000

[3] Gerald Kotonya and Ian Sommerville, *Requirements Engineering Process and Techniques*, John Wiley & Sons, 1998

[4] James and Suzanne Robertson, *"Volere Requirements Specification Template"*, Edition 6.1, Atlantic Systems Guild, 2000

[5] Larry Boldt, *"Managing Requirements at the Object Level"*, Technology Builders Inc., http://www.tbi.com/products/asq/

[6] Standish Group Research Report, *CHAOS*, 1995, http://www.standishgroup.com/visitor/ chaos.htm

[7] 고영중, 강기선, 김재선, 박수용, 서정연, *"요구사항 문장 범주화를 이용한 웹 기반의 요구사항 추출지원 시스템"*, 정보과학회논문지(소프트웨어 및 응용), 제27권, 제4호, pp. 384-392, 2000. 4.

[8] 이원우, 황만수, 박수용, 류성열, *"요구사항 관리를 위한 웹 기반 모델 설계"*, 소프트공학회지, 제11권, 제4호, pp. 35-45, 1998. 12.

[9] NCA II-AUER-97095, *시스템 개발방법론 적용기준에 관한 연구*, 한국전산원, 1997. 12.

[10] 관리기법/1 오브젝트 샘플 (버전 9.5), 한국전산원

[11] 심기보, *정보시스템 외주관리실무*, 진한도서, 서울, 2000

[12] 왕창종, *소프트웨어 공학*, 정익사, 서울, 2000

[13] 정기원, 윤창섭, 김태현, *소프트웨어 프로세스와 품질*, 홍릉과학출판사, 서울, 1997

## 저 자 소 개

신 종 철

1976년 서울대학교 전기과 졸업
1994년 연세대학교 산업대학원 (전자계산전공) 졸업
2000년 충북대학교 전자계산학과 박사과정 수료
1978~1992년 현대엔지니어링 전산실 / 정보사업부
1992~2000년 (주)송우정보 대표이사
2000~현재 송우아이엔티(주) 기술연구소장
주 관심분야 : 개발방법론, 요구공학, 정보시스템 감리