

개발 방법론의 요구 사항 변경 관리를 개선하기 위한 프로세스 모델

(A Process Model to Improve the Requirements Change Management for the Development Methodologies)

정 규 장 * 신 종 철 ** 구 연 설 ***
(Kyu-Jang Chung) (Jong-Cheol Shin) (Yeon-Seol Koo)

요약 폭포수 모형을 기반으로 하는 구조적 개발 방법론에서는 요구 분석 단계에서 요구 사항들이 일단 명세화되고 나면, 이들은 단지 후속의 개발 단계를 위한 중간 산출물로만 사용되고 더 이상 요구 사항 자체를 관리 대상으로 취급하지 않기 때문에 설계 단계 이후에 발생하는 요구 사항의 변경을 관리할 수 있는 절차가 미흡하다. 그러나 현실적으로는 정보 기술의 발전, 시장 환경이나 적용 환경의 변화 등으로 인하여 개발 기간 중 요구 사항은 끊임없이 변화하게 된다. 따라서 이러한 요구 사항의 지속적인 변경을 지원하기 위해서는 전체 개발 생명 주기에 걸쳐 요구 사항을 관리하고 특히 설계 단계 이후의 요구 사항 변경을 지원할 수 있는 요구 사항 변경 관리 프로세스가 필요하다.

이 논문에서는 하향식(top-down)의 구조적 개발 방법론에 적용할 수 있는 요구 사항 변경 관리 프로세스 모델을 제안하여 설계 단계 이후에 발생하는 요구 사항의 변경을 체계적으로 관리하고 요구 사항 자체를 모든 개발 생명 주기에서 활용하기 위한 방안을 제시한다. 제안 프로세스는 마르미 방법론의 개발 프로세스와 산출물 측면의 적용 검토를 통하여 개발 방법론의 요구 사항 변경 및 관리에 대한 개선 효과를 평가한다.

키워드: 요구 공학, 요구 사항 관리, 변경 관리, 개발 생명 주기, 개발 방법론

Abstract In conventional development methodologies, requirements are considered to be not changing after analysis phase, and requirements specifications are used for the next step system design purpose. But in the real world, requirements can be changed and modified throughout 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. So there needs a requirements change management process that can extend requirements management over the entire development life cycle and can support managing changes to the requirements after design phase.

In this paper, a requirements change management process that can be integrated into conventional development methodologies is proposed to support the extension of requirements life cycle and managing changes to the requirements after design phase. This process was evaluated through an verification test with a widely used development methodology 'MaRMI'.

Key words: Requirements Engineering, Requirements Management, Change Management, Development Life-cycle, Development Methodology

1. 서론

* 정 회 원 : 한국지리정보산업협동조합
magis@hanmir.com
** 비 회 원 : 송우아이엔티(주)
sis@songwoo.co.kr
*** 종신회원 : 충북대학교 컴퓨터과학과 교수
yskoo@cbucc.chungbuk.ac.kr
논문접수 : 2001년 12월 24일
심사완료 : 2003년 3월 4일

현재 널리 적용되고 있는 하향식의 구조적 개발 방법론에서는 요구 사항의 추출 및 명세화가 분석 단계에서 이루어지고, 다음 단계인 설계 및 구현을 위한 기준으로 사용된다. 그러나 일단 설계 요소로 변환된 요구 사항들은 더 이상 요구 사항 자체로 관리되지 않고, 개발 방법론 내부에는 설계 단계 이후의 요구 사항 변경을 지원할 수 있는 절차가 미흡하다. 따라서 설계 단계 이후에 요구 사항의 변경이 발생하면 프로젝트 관리뿐만 아니라

라 개발 기간과 비용에도 심각한 영향을 초래하여, 특히 외주 개발의 경우에는 발주자와 개발자 사이의 분쟁을 유발하는 원인이 되기도 한다.

시스템 개발의 근간이 되는 요구 사항을 보다 체계적으로 관리하기 위하여 요구 사항 관리 모델[15], 시스템 및 소프트웨어 요구 사항 명세화 지침(IEEE Std 1233 [22] 및 Std 830-1998[21]), Volere 프로세스와 템플릿[10], 요구 사항 추적표[12, 15] 등의 요구 사항 관리 기법과, DOORS, RTM, RequisitePro, SACHER, Core, PROTEUS, CRADLE, GMARC 등과 같은 상용화 된 요구 사항 관리 도구[23]들이 존재한다. 그러나 이러한 요구 사항 관리 도구와 기법들은 적용 기반의 미비, 전문 인력의 부족, 현재 사용되고 있는 개발 방법론과의 통합이 이루어지지 않음 등으로 인하여 실제 프로젝트에 대한 적용에 어려움이 많은 실정이다.

이 논문에서는 요구 사항의 관리 범위를 확장하고 변경 관리를 개선할 수 있는 요구 사항 변경 관리 프로세스를 통하여 구조적 개발 방법론의 요구 사항 변경 관리를 개선하는 방안을 제시한다. 프로세스는 요구 사항 관리를 위한 객체-관계 모형과 요구 사항 변경 관리 절차와 활동, 영향 범위 추적 모형, 그리고 산출물을 통하여 정의한다. 일반적으로 요구 공학과 개발 방법론, 특히 프로세스 분야는 프로젝트의 규모와 특성에 따라 조정과 변형이 불가피하기 때문에 논리적 검증이나 계량화된 측정이 어렵고 실무적 경험과 지식을 필요로 한다. 따라서 제안 요구 사항 변경 관리 프로세스의 평가를 위하여, 대표적 개발 방법론의 하나인 마르미 방법론의 개발 프로세스와 산출물에 대한 적용 검토를 통하여 적용 효과를 분석한다.

이 논문의 제 2 장에서는 요구 공학 및 개발 방법론에서의 요구 사항 관리 절차 및 기법들을 분석하고, 제 3 장에서는 요구 사항 변경 관리를 위한 모형, 활동 및 산출물의 정의를 통하여 프로세스 모델을 제안한다. 그리고 제 4 장에서는 제안 프로세스 모델을 검증하고 평가하기 위하여 마르미 방법론에 대한 적용 검토와 정보 시스템 개발 경험이 풍부한 정보처리 기술사 및 정보 시스템 감리인에 대한 설문 조사를 통하여 요구 사항 변경 관리의 적용 효과를 분석한다.

2. 관련 연구

2.1 개발 방법론의 요구 사항 관리

ISO 12207에서는 5개의 기본 공정 중 개발 공정에서 소프트웨어의 개발과 관련된 13개의 주요 활동들을 정의하고 있는데[5], 요구 사항 분석 활동은 시스템 요구

분석(5320)과 소프트웨어 요구 분석(5340)으로 나누어진다. 시스템 요구 분석은 시스템 요구 명세 작성(5321)과 시스템 요구 사항 평가(5322)로 세분화되며 소프트웨어 요구 분석은 소프트웨어 요구 명세 작성(5341), 소프트웨어 요구 사항 평가(5342), 합동 검토 및 베이스라인 설정(5343)으로 세분화된다. 그리고 시스템 요구 분석 활동은 시스템 구조 설계 활동(5330)으로, 소프트웨어 요구 분석 활동은 소프트웨어 구조 설계 활동(5350)으로 변환된다.

마르미 방법론에서는 요구 분석 단계를 단계 준비, 사용자 요구 사항 정의, 프로세스 모형 구성, 프로세스/엔티티 연관 분석, 인터페이스 분석, 엔티티 모형 구성, 분산 모형 구성, 테스트 요건 정의, 단계 점검의 9개 활동으로 구분한다[3, 5]. 이들 중에서 사용자의 요구 사항을 도출하고 분석하는 작업은 사용자 요구 사항 정의 활동 단계에서 주로 이루어지는데, 이 사용자 요구 사항 정의 활동은 사용자 요구 사항 정리(D2202), 현행 시스템 분석(D2204), 사용자 요구 사항 분석(D2206)의 3개 세부 업무로 구분된다. 그리고 사용자 요구 사항 정리(D2202) 작업과 현행 시스템 분석(D2204) 작업은 사용자 요구 사항 분석(D2206) 작업으로 통합되며, 사용자 요구 사항 정의 활동은 다음 단계의 활동인 프로세스 모형 구현, 프로세스/엔티티 연관 분석, 엔티티 모형 구성으로 변환된다[5].

관리기법/1에서는 프로젝트의 규모, 기술적 범위, 업무 절차 등을 고려하여 정보 계획 수립, 클라이언트/서버 시스템 개발, 호스트 시스템 개발, 패키지 시스템 개발, 소규모 프로젝트 개발, 고속 개발 등으로 개발 경로를 구분하는데[5], 클라이언트/서버 개발 단계가 대규모 프로젝트를 기준으로 전체 경로를 포함하고 있으므로 이를 기준으로 적용하면 사용자의 요구 사항은 사용자 요구 사항(350C) 세그먼트에서 명세화가 이루어진다. 사용자 요구 사항(350C) 세그먼트는 작업 흐름 및 조직 확인, 사용자 요구 사항 파악, 현행 설계 복구의 3개 타스크로 구분되며, 이 세그먼트는 다음 단계인 요건 분석(370C) 세그먼트로 변환된다[5].

ISO 12207, 관리기법/1, 그리고 마르미 방법론과 같은 하향식의 구조적 개발 방법론에서는 공통적으로 파악된 요구 사항들을 각각의 관리 대상으로 설정하지 않고, 후속의 요구 사항 명세서나 설계서 작성을 위한 중간 단계로 활용한다. 따라서 분석된 요구 사항 중에서 오류가 발생하거나 누락된 요구 사항이 발생할 경우, 후속의 개발 단계에서는 역으로 추적하거나 확인할 수 있는 원천 정보를 상실함으로써 검증 및 변경을 위한 근

거를 잃어버리게 되는 문제점이 있다. 그리고 요구 분석이 완료된 다음에는 이후의 개발 기간 동안 요구 사항 자체를 관리 대상으로 취급하지 않기 때문에, 실제 프로젝트에 있어서 전체 개발 노력(비용)과 기간의 70 - 80% 이상[7, 9, 12]을 차지하는 나머지 개발 단계에서 발생할 수 있는 요구 사항의 변경을 체계적으로 수용할 수 있는 절차가 없다.

2.2 Volere 요구 사항 프로세스 모델

Volere 요구 사항 프로세스 모델은 요구 사항을 수집하고 명세화하며 검증하는 포괄적인 활동들로 구성되며, Volere 요구 명세 템플릿을 사용하여 시험 가능한 요구 사항을 발견하고 문서화하는 지침을 제공한다[13]. 이 프로세스 모델은 각 활동의 수행 순서나 절차보다는 각 활동과 연결된 산출물을 중심으로 정의하기 때문에 필요에 따라 일부 활동의 조합으로 재구성하거나, 적용하는 프로젝트의 규모나 특성에 따라 조정(tailoring)이 가능하다. 그리고 그림 1에서 원으로 표시된 각 활동들은 데이터 흐름도(DFD)와 같이 하위 단계로 계층적 상세화가 가능하다[13].

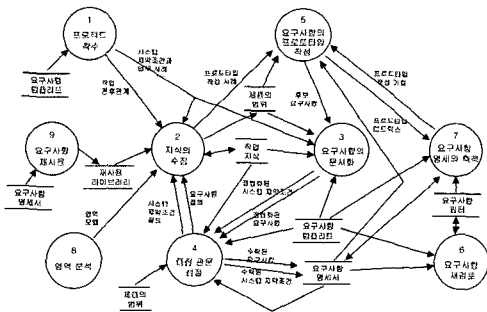


그림 1 Volere 요구 사항 프로세스 모델 (요약)

이 모델은 프로젝트 착수, 요구 사항의 도출을 위한 지식의 수집, 요구 사항의 문서화, 요구 사항의 품질 향상 및 누락 방지를 위한 품질 관문의 설정, 요구 사항의 프로토타입의 작성, 요구 사항의 재검토, 요구 사항 명세의 추적, 영역 분석, 요구 사항의 재사용 등의 활동으로 구성되며, 각 활동간의 의존 관계는 산출물을 사용한 인터페이스로 정의한다.

이 프로세스 모델에서 사용하는 요구 명세 템플릿(requirements specification template)[10]은 Atlantic Systems Guild가 프로젝트 수행 경험과 요구 공학적 연구를 바탕으로 작성한 요구 사항 명세화를 위한 템플

리트르, 요구 사항을 명세화하는데 필요한 틀(framework)을 제공한다. 각각의 요구 사항은 템플리트를 사용하여 별도로 명세화하게 되는데, 요구 사항들은 시스템이 수행할 역할과 기능에 관한 기능적 요구 사항, 시스템이 갖추어야 할 비기능적 요구 사항, 프로젝트 구동 요소, 프로젝트 제약 사항, 프로젝트 쟁점 사항 등으로 구분된다.

이 Volere 요구 사항 프로세스 모델은 각각의 요구 사항을 관리 대상 객체로 선정하도록 요구 명세 템플리트를 사용하고 절차보다는 산출물을 통하여 프로세스 모델을 정의함으로써, 다양한 개발환경에서 적용이 용이하다. 그러나 정의된 산출물이 요구사항의 관리적 측면만 반영하고 있어 개발 프로젝트에 널리 사용되는 개발 방법론에 직접 적용하기 어렵고, 프로젝트 관리에 활용하기 위해서는 추가적인 적용 방안의 설정이 필요하다.

2.3 요구 사항의 객체화 관리

Larry Boldt는 전체 요구 사항을 시스템 요구 명세서(SyRS)나 소프트웨어 요구 명세서(SRS)와 같은 통합된 문서로 관리하지 않고, 각각의 요구 사항을 속성과 행위를 가지는 관리 대상 객체로 정의하는 방안을 제시하였다[18]. 개별 요구 사항을 독립적인 객체로 설정하면, 각각의 요구 사항에 대한 별도 관리가 가능하고, 요구 사항 변경 시 버전 부여 및 변경 이력 관리가 용이하며, 요구 사항에 대한 검증 및 확인 기준을 설정하여 개발 초기 단계부터 요구 사항에 대한 테스트를 시작할 수 있으며, 요구 사항으로부터 설계 및 구현으로 전환에 대한 개별적 추적이 가능하고, 변경으로 인한 영향 범위 및 파급 효과를 분석하기 용이한 장점이 있다[18]. 그런데 각각의 요구 사항을 관리 대상 객체로 설정하기 위해서는 먼저 요구 사항들을 각각 독립적으로 명세화해야 하는데, Volere의 요구 명세 템플리트나 관리기법/1의 '요구 사항 설명' 템플리트 등이 요구 사항의 독립적인 객체화에 적용될 수 있다.

Volere 요구 명세 템플리트[10]는 각각의 요구 사항에 대하여 관리 번호를 부여하고 정해진 형식에 따라 카드 형태의 셀(shell)에 기록하여 관리함으로써, 요구 사항의 점진적인 구체화를 가능하게 하며, 요구 사항을 전체 개발 생명 주기에 걸쳐 관리하고 추적할 수 있도록 하고, 적용 기준(fit criteria)을 통하여 요구 사항의 명세화 단계에서부터 요구 사항에 대한 테스트를 시작할 수 있도록 지원한다. 템플리트의 구성 항목은 요구 사항 번호, 요구 사항 형식, 사건 / 유스케이스 번호, 내용 설명, 근거, 출처, 적용 기준, 고객 만족도, 고객 불만 정도, 의존 관계, 상충되는 요구 사항, 관련 문서, 이력

등으로 이루어지며, 관리를 목적으로 요구 사항의 형식은 5개 종류 / 27개 세부 형식으로 세분화된다.

관리기법/1에서는 사용자 요구 사항을 파악하여 '요구 사항 설명'을 작성하도록 오브젝트 템플릿을 정의하고 있으나[8], 필수 산출물이 아니라 다음 단계의 작업을 위한 중간 산출물로 사용한다. 이 '요구 사항 설명'은 요구 사항을 정의하고, 응용이 무엇을 어떻게 하는지를 정의하며, 기능이 충족되었을 때의 효과와 기능 요구 사항으로 인해 프로젝트팀에 부과되는 제약 사항을 파악하는데 사용된다. 작성자는 기능 분석자나 사용자이며, 작성 시기는 기능 분석자가 기능 요구 사항의 확인이 필요할 때, 그리고 작성량은 주요 요구 사항별(기능, 품질, 정량적 효과, 정성적 효과)로 작성하며, 작성 도구는 저장소 중심 워드프로세스가 사용된다.

2.4 요구 사항 추적표

요구 사항 추적표는 요구 사항과 요구 사항 발생 출처와의 관계, 요구 사항과 발생 근거와의 관계, 요구 사항과 요구 사항간의 관계, 요구 사항과 설계와의 관계, 요구 사항과 시스템 구현과의 관계, 요구 사항과 인터페이스와의 관계 등을 나타낼 수 있다[15]. 요구 사항과 요구 사항간의 관계는 요구 사항 추적표를 사용하지 않고 개별 요구 사항에 하나의 속성으로 정의할 수도 있으나, 요구 사항의 변경 관리를 위해서는 추적표 형식으로 기술하는 것이 바람직하다. 이 논문에서는 특히 요구 사항과 요구 사항간의 관계를 정의한 요구 사항 추적표를 요구 사항 관계표라고 명명하여 다른 요구 사항 추적표와 구분하기로 한다.

요구 사항 관계표는 서로 영향을 주거나 받는 요구 사항간의 상호 관계를 표시할 수 있는데, 영향 유무만 단일 기호로 표시하거나 복수의 기호(U : Use, R : Related 등)를 사용하여 영향 관계의 성질까지 표시할 수 있다. 표 1은 요구 사항 관계표의 예로, RQ-2는 RQ-3과 RQ-5에 영향을 주고, RQ-4의 영향을 받는 관계를 나타낸다.

표 1 요구 사항 관계표

구분	RQ-1	RQ-2	RQ-3	RQ-4	RQ-5
RQ-1			○	○	
RQ-2			○		○
RQ-3	○				
RQ-4		○	○		
RQ-5				○	

표 2 요구 사항 추적 리스트

구분	관련 요구사항
RQ-1	RQ-3, RQ-4
RQ-2	RQ-3, RQ-5
RQ-3	RQ-1
RQ-4	RQ-2, RQ-3
RQ-5	RQ-4

표 1과 같은 요구 사항 관계표는 요구 사항의 수가 많아지면 관리하기 어려운 문제점이 있다. 따라서 요구 사항의 수가 많아지면 요구 사항을 Volere 템플릿의 요구 사항 형식을 기준으로 하거나 혹은 단순하게 기능적 요구 사항과 비기능적 요구 사항으로 범주화하여 그룹별로 관리하거나, 보다 단순한 형태인 표 2와 같은 요구 사항 추적 리스트를 적용할 수 있다. 그러나 이 요구 사항 추적 리스트는 사용하기 간편하고 요구 사항의 수가 많아지더라도 관리가 가능하나, 영향을 미치는 하나의 방향만 표시할 수 있기 때문에 역방향의 영향 관계를 파악하기 위해서는 새로운 추적 리스트를 다시 만들어야 하는 문제점이 있다.

3. 요구 사항 변경 관리 프로세스 모델

제 2 장에서 살펴본 바와 같이, 개발 방법론에서는 요구 사항을 도출하고 분석하며 명세화하여 설계에 반영하는 절차와 방법을 제시하지만 설계 단계 이후의 요구 사항 변경은 지원하지 못하는 문제점이 있고, Volere 요구 사항 프로세스 모델을 비롯한 요구 사항 관리 모델과 요구 공학적 기법들은 요구 사항 자체에 대한 관리 방안만 제시하기 때문에 시스템 개발을 위한 기본 틀로 사용되는 개발 방법론에 적용하기 위해서는 별도의 절차와 노력이 필요하다. 따라서 이 논문에서는 폭포수 모형을 기반으로 하는 구조적 개발 방법론의 관점을 바탕으로 하는 요구 사항 변경 관리 프로세스 모델을 통하여 기존 개발 방법론에서 미흡한 요구 사항 변경 관리를 개선하는 방안을 제시한다.

일반적으로 프로세스는 입력을 출력으로 변환하는 상호 연관된 활동들의 집합을 의미한다[15]. 간단한 프로세스의 경우에는 상세한 절차를 규정하기도 하지만, 좀더 복잡한 프로세스의 경우에는 이를 적용하는 사람의 개인적 배경이나 기술 수준, 적용 환경에 따라 적용 방법이나 결과가 달라질 수 있으므로 상위 수준의 절차나 결과물만 규정한다. 따라서 이 논문에서는 요구 사항 변경 관리 프로세스를 설계하기 위하여 우선 요구 사항의 객체화와 관계 표시에 의한 관리 모형을 설정하고, 이를

바탕으로 요구 사항 변경 관리에 필요한 활동과 산출물을 정의한다.

3.1 요구 사항 관리 모형의 정의

요구 사항의 변경을 관리하기 위해서는 우선 요구 사항 관리 모형을 정의하여야 한다. 그런데 Gerald Kotonya 와 Ian Sommerville의 요구 사항 관리 모델[15]이나 요구 사항 관리 모델을 웹 기반으로 확장한 ‘웹 기반 요구 사항 관리 모델’[1]과 같은 일반적인 요구 사항 관리 모델들은 요구 사항 관리의 개략적인 흐름이나 전체적인 기능 구성만을 정의하고 있어 개발 방법론에 통합 가능한 요구사항 변경 관리 프로세스를 정의하기에 부적합하다. 따라서 여기서는 요구 사항 전체를 각각의 요구 사항 객체(entity)와 요구 사항간의 관계(relation)로 표시하는 객체-관계 모형으로 정의한다. 이 객체-관계 모형에서는 각각의 요구 사항을 독립적으로 관리할 수 있는 객체로, 그리고 요구 사항간의 영향 관계를 관계에 대한 속성(attribute) 형식으로 정의한다.

각 요구 사항 객체는 요구 사항 기술서의 형식으로 표시하는데, 이 객체의 속성은 프로젝트의 성격이나 규모에 따라 달라질 수 있으나 대표적인 속성으로는 요구 사항 번호, 요구 사항 종류, 요구 사항 설명, 근거, 출처, 충족 요건, 중요도, 참조 문서, 관련자, 생성 일자, 변경 일자, 변경 이력 등이 있다. 그리고 요구 사항간의 관계는 제 2.4 절에서 살펴 본 요구 사항 관계표를 사용하여 표시하며, 이 관계가 가지는 주요 속성은 영향을 미치는 방향과 영향을 미치는 정도를 수치화한 가중치의 두 가지가 있다.

그림 2는 각 요구 사항을 객체로 표시하고 객체간의 관계를 링크로 표시한 객체-관계 모형으로, 링크는 방향과 가중치를 갖는 그래프를 사용하여 표시한다.

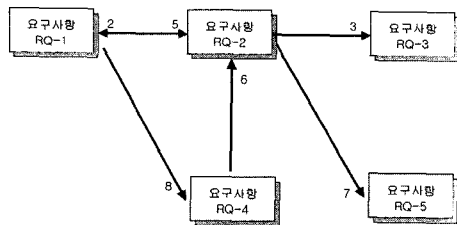


그림 2 객체-관계 요구 사항 관리 모형

그리고 그림 2의 객체-관계 모형을 테이블 형식으로 나타내면 표 3과 같은 요구 사항 관계표로 변환된다. 이 요구 사항 관계표에서 행으로는 영향을 미치는 요구 사항들을 표시하고 열로는 영향을 받는 요구 사항들을 나

표 3 요구 사항 관계표

구 분	RQ-1	RQ-2	RQ-3	RQ-4	RQ-5
RQ-1		5		8	
RQ-2	2		3		7
RQ-3					
RQ-4		6			
RQ-5					

타낼 수 있는데, 예를 들어 RQ-1은 행을 기준으로 RQ-2와 RQ-4에 영향을 미치고 열을 기준으로 RQ-2의 영향을 받음을 표시한다.

그러나 표 3과 같은 요구 사항 관계표는 일반적으로 요구 사항의 수가 많아지면 관리하기 어렵게 되므로, 규모가 큰 프로젝트에서는 기능 요구 사항과 비기능 요구 사항의 두 가지 범주로 구분하거나, Volere 템플리트와 같이 기능 요구 사항, 비기능 요구 사항, 프로젝트 제약 사항, 프로젝트 구동 요소, 프로젝트 쟁점 사항의 5 가지 범주로 구분하여 관리할 수 있다.

3.2 요구 사항 변경 관리 절차

일반적으로 개발 프로젝트의 진행 과정에서 요구 사항의 변경 요구가 발생하면, ① 이 요구에 대한 타당성을 평가하고, ② 이 변경으로 인한 다른 요구 사항들의 영향 범위를 분석하며, ③ 이 요구를 구현하기 위한 소요 자원(개발 기간, 인력, 비용 등)을 산정하여, ④ 발주자와 개발자측 관리자의 승인을 거쳐 요구 사항의 변경이 이루어지게 되는데, 이러한 요구 사항 변경 관리 절차를 도식화하면 그림 3과 같다.

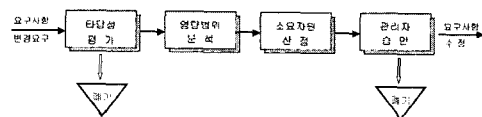


그림 3 요구 사항 변경 관리 절차

요구 사항의 타당성을 평가하는 단계에서는 요구 사항 자체의 타당성뿐만 아니라 개발중인 목표 시스템과의 부합 여부 및 전체 시스템에 미치는 영향을 평가하여 변경 요구를 시스템 개발에 반영할 것인지를 결정하며, 영향 범위 분석 단계에서는 앞 절에서 기술한 요구 사항 관리 모형을 이용하여 해당 요구 사항의 변경에 따라 영향을 주거나 받는 요구 사항들을 파악한다. 그리고 소요 자원 산정 단계에서는 이 변경 요구를 구현하기 위하여 필요한 인력, 기간, 비용을 산정하며, 관리자 승인 단계에서는 요구 사항 자체의 타당성과 산정된 소요 자원의 규모에 따라 계약 변경이 필요할 수도 있기

때문에 발주자와 개발자 측 관리자가 참여하여 이의 승인 여부를 결정한다.

이러한 요구 사항 변경 관리 절차는 일반적인 변경 관리 절차와 유사하나, 운영 및 유지 보수 단계와는 달리, 관리자 승인 단계에서 계약 범위에 대한 확인을 위하여 개발자뿐만 아니라 발주자측 관리자까지 함께 참여해야 하고, 변경 요구가 발생하는 개발 작업 단계에 따라 재 작업을 위한 소요 자원이 달라지게 되는 차이가 있다. 특히, 소요 자원 산정 단계에서는 영향 범위와 함께 재 작업 거리를 고려해야 하는데, 개발 작업의 진행 단계에 따라 분석 단계로 되돌아가는 재 작업 거리가 달라지고 소요되는 비용이 기하 급수적으로 증가하기 때문이다[17].

$$R = f(d, s)$$

여기서 R은 소요 자원(인력, 기간, 비용)을 의미하고, d는 현재 단계에서 분석 단계까지의 재 작업 거리를, 그리고 s는 영향을 받는 요구 사항들의 범위를 의미한다. 특히, 요구 분석 단계의 문제 해결을 위한 노력을 1이라고 할 때, 설계 단계는 3~6, 구현 단계는 10, 테스트 단계는 15~40, 검수 단계는 30~70, 그리고 운영 단계는 40~1000 정도의 노력이 소요되는 것으로 알려져 있으므로[14], 재 작업 거리 d는 소요 자원 산정의 중요한 영향 요소로 작용하게 된다.

또한 영향 범위 s는 제 3.1 절의 요구 사항 관리 모형에 의하여 직접 영향을 받는 요구 사항들을 포함한다. 그러나 이들 직접 영향을 받는 요구 사항만으로는 전체 시스템에 미치는 영향 범위를 모두 파악할 수 없기 때문에, 요구 사항의 변경이 발생하면 이에 따라 영향받는 설계 / 구현 / 테스트 요소까지 분석이 필요하다.

3.3 영향 범위 추적 모형

요구 사항은 개발 작업의 단계가 진전됨에 따라 설계, 구현, 테스트의 산출물로 변환되기 때문에, 요구 사항의 변경으로 인한 영향 범위를 분석하기 위해서는 요구 사항에 대한 추적성(traceability)이 필요하다. 요구 사항에 대한 추적성은 수직 및 수평 추적성으로 구분되며, 방향성 그래프를 사용하여 표시할 수 있다[11].

그러나 이 그래프에는 요구 사항 상호간의 영향 관계를 표시하는 수직 추적성이 누락되어 있으며, 수평 추적성은 일대 다(1:n)의 관계를 가지므로 모든 링크의 가중치가 동일하지 않으나 이를 표시할 수 없는 문제점이 있다. 그리고 이 그래프는 유지 보수 단계를 기준으로 하기 때문에 개발 단계의 요구 사항 추적을 위하여 각 구현 요소와 일대 일(1:1) 대응 관계를 갖는 테스트 단계는 생략할 수 있다.

따라서 여기서는 요구 사항 단계의 수직 추적성을 추가하고, 각 링크에 가중치를 부가하며, 테스트 단계를 생략하여 그림 4와 같은 영향 범위 추적 모형을 정의한다. 이 모형은 개발 작업의 단계가 진행함에 따라 수평적 구성 요소뿐만 아니라 수직적 구성 요소들도 단계적으로 추가할 수 있는 성장형 모형으로, 요구 사항의 변경이 발생할 때마다 모형의 내부 구성이 변경된다.

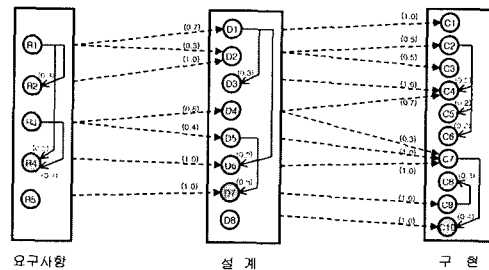


그림 4 요구 사항 영향 범위 추적 모형

이러한 영향 범위 추적 모형은 알고리즘의 적용을 위하여 매트릭스나 테이블 형태로 표시할 수 있는데, 매트릭스 형태로 표시하면 역 방향 추적성까지 나타낼 수 있으나, 수직 구성 요소간의 연관 관계가 비교적 단순할 경우 최소 매트릭스가 생성되기 때문에 테이블 형태로 표시하는 것이 취급하기 간편하다. 그림 4와 같은 추적성 모형을 테이블로 변환하면, 요구 사항간의 영향 관계를 표시하는 RV(수직 요구 사항) 테이블, 요구 사항과 설계 요소간의 영향 관계를 표시하는 RH(요구 사항-설계) 테이블, 설계 요소간의 영향 관계를 표시하는 DV(수직 설계) 테이블, 설계 요소와 구현 요소간의 영향 관계를 표시하는 DH(설계-구현) 테이블, 구현 요소간의 영향 관계를 표시하는 CV(수직 구현)테이블 등으로 표시할 수 있다.

3.4 활동과 산출물의 설계

제 3.2 절에서 기술한 요구 사항 관리 절차에 따라 요구 사항 변경 관리 활동은 요구 사항 변경 요구의 타당성 평가, 변경으로 인한 영향 범위 분석, 변경을 구현하기 위하여 필요한 소요 자원 산정, 관리자 승인의 4가지 세부 활동으로, 그리고 이들 활동의 결과로 생성되는 산출물은 요구 사항 기술서, 요구 사항 관계표, 요구 사항 변경 내역서, 요구 사항 변경 영향 분석서의 4 가지로 표 4와 같이 정의한다.

요구 사항 기술서는 기존의 개발 방법론에서와 같이 설계서나 명세서 작성을 위한 중간 산출물이 아니라 개발 생명 주기의 전 단계에서 작성 관리해야 하는 주요

산출물로 정의하며, 프로젝트의 초기 단계에서는 요구사항의 상세화가 어려움을 감안하여 개발 프로세스의 진행에 따라 점진적으로 상세화할 수 있도록 요구사항간의 관계를 이용하여 서로 다른 상세화 수준을 지원한다. 요구사항 기술서는 각각의 요구사항에 대하여 작성하며, 비전문가인 사용자나 개발 참여자도 쉽게 이해할 수 있도록 가급적 자연어로 작성하고, 각각의 요구사항을 하나의 관리 단위(객체)로 설정한다. 이 객체가 가지는 주요 속성은 Volere 요구명세 템플리트나 관리기법[1]의 '요구사항 설명' 템플리트 등을 기준으로 프로젝트의 성격이나 규모, 특성에 따라 조정할 수 있으며, 여기에 변경에 따른 버전 부여와 버전별 간략한 변경내용 기술 항목을 추가하여 향후 요구사항 변경 관리, 특히 변경 이력 관리를 위하여 활용한다.

표 4 요구사항 변경 관리 활동과 산출물

활동	세부 활동	산출물
요구사항 변경관리	타당성 평가	요구사항 기술서
	영향 범위 분석	요구사항 관계표
	소요 자원 산정	요구사항 변경 내역서
	관리자 승인	요구사항 변경 영향 분석서

요구사항 관계표는 요구사항 관리 모형에 의하여 각 요구사항의 다른 요구사항에 대한 영향 관계를 테이블 형식으로 표시하며, 다른 요구사항에 대한 영향 정도를 가중치로 나타낸다. 가중치는 영향을 받거나 미치는 정도에 따라 제 3.3 절의 영향 범위 추적 모형에서와 같이 0.0-1.0 사이의 수치로 부여하되, 하나의 요구사항이 다른 요구사항에 의하여 유도된 것이거나 완전히 종속적인 경우에는 최고의 가중치(1.0)를 부여한다. 이 관계표는 요구사항의 변경이 발생할 때마다 수시로 수정되어 항상 최신의 영향 관계를 나타내게 된다.

요구사항 변경 내역서는 각 요구사항의 변경 내역을 관리하기 위한 산출물로, 소스 코드처럼 정형화하기 어렵고 체계적인 버전 관리가 어려운 요구사항의 특성을 고려하여 항상 현재 유효한 요구사항들을 파악하고 요구사항의 변경 이력을 추적하기 위하여 사용한다. 그리고 관리의 편의를 위하여 추가, 변경, 수정된 요구사항별로 구분하거나 별도 목록을 추가할 수 있다.

요구사항 변경 영향 분석서는 발주자와 개발자간에 요구사항 변경으로 인한 영향에 대한 공통의 이해와 합의를 유도하기 위하여 사용되며, 추가, 변경, 혹은 삭제된 각각의 요구사항에 대하여 관련된 요구사항 목록을 명시하고, 이로 인하여 발생할 수 있는 영향 범

위를 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 영향 범위를 평가하기 힘들고, 간접적인 영향까지 분석하기는 매우 어려우므로 파악 가능한 범위에서 기술적인 영향 범위와 추가로 소요되는 인력, 비용, 기간 등의 자원을 산정하여 기술하고, 프로젝트 관리자와 발주자측의 승인을 거치도록 한다.

4. 평가 및 검토

요구공학 분야와 개발 방법론 분야는 실제 시스템 개발 과정에서 발생하는 다양한 현상을 대상으로 하고, 특히 개발 방법론에서는 프로젝트의 규모와 성격, 특성에 따라 프로세스의 활동과 산출물의 조정을 전제로 한다. 따라서 제 3장에서 설계한 요구사항 변경 관리 프로세스 모델에 대한 논리적 검증이나 계량화된 평가가 어렵기 때문에, 여기서는 현재 널리 사용되고 있는 개발 방법론에 대한 적용 검토를 통하여 프로세스 모델의 적용성과 개선 효과를 평가한다.

4.1 객체-관계 모형에 대한 평가

제 3.1 절의 요구사항 관리 모형은 각각의 요구사항을 객체로 정의하기 때문에 개별 요구사항의 관리 대상 객체화를 지원한다. 그리고 다른 요구사항과의 영향 관계를 가시화하여 요구사항 기술 문서의 한 항목으로만 표시되는 요구사항 상호간의 의존 관계를 보다 체계적으로 나타낼 수 있다. 그런데 요구사항은 업무 수준의 요구사항에서부터 설계와 코딩에 사용되는 상세 요구사항으로 단계별로 상세화되거나, 다른 요구사항으로부터 파생되는 경우와 같이 다른 요구사항과 연관 관계를 가질 수 있다. 그러나 그 영향 정도는 서로 다르기 때문에 가중치의 개념을 도입하면 향후 변경으로 인한 영향 범위 분석과 소요 자원 산정에 활용할 수 있다.

각각의 요구사항을 독립적인 객체로 설정하면 요구사항에 대한 가시성, 재사용성, 시험성, 추적 및 대체성, 유지 보수 및 보안성, 완전성 등을 개선할 수 있다[18]. 요구사항의 개별적 관리를 통하여 검토, 정렬, 필터링 등 가시적 취급이 용이하며, 요구사항에 대한 버전을 부여하여 변경 관리 및 다음 프로젝트에서의 재사용이 가능하게 된다. 그리고 각 요구사항별로 확인 및 검증 기준을 부여하여 개발 초기 단계부터 테스트를 시작할 수 있으며, 요구사항으로부터 설계 및 구현으로의 추적

이 가능하고, 변경으로 인한 영향 범위 및 파급 효과를 분석하기 용이한 장점이 있다[18]. 각 요구 사항별로 변경 이력을 유지하고 보안 수준을 설정하여 유지 보수와 보안성을 향상할 수 있으며, 객체의 속성으로 정의한 요구 사항의 세부적 기술과 다른 객체와의 연결성 등을 통하여 요구 사항 자체의 완전성을 향상할 수 있다.

4.2 마르미 방법론 적용 평가

국내에서 널리 적용되고 있는 구조적 개발 방법론에는 관리기법/1, 마르미 방법론, 정보공학 방법론 등이 있다. 그런데 관리기법/1은 프로젝트 규모나 적용 대상에 따라 다양한 개발 경로가 존재하기 때문에 어느 하나의 개발 경로에 대하여 대표성을 부여하기 어렵고, 정보공학 방법론은 구조적 기법을 프로젝트 단위가 아닌 기업 전체에 대한 정보 전략의 관점으로 적용하고 데이터 모델을 중심으로 분석과 설계를 진행하며 프로그램의 자동 생성을 전제로 하여 프로세스 모델이 정교하지 않기 때문에 요구 사항 관리 프로세스의 적용 검토 대상으로 부적절하다. 반면, 마르미 방법론(마르미-D 기준)은 전형적인 구조적 기법을 사용하고 ISO 12207을 기반으로 하며 각 단계별 활동과 산출물의 정의가 정형화되어 있으므로 이 논문에서는 적용 검토의 대상으로 마르미 방법론을 선정한다.

4.2.1 개발 프로세스에 대한 적용

마르미 방법론에는 정보 시스템 개발 방법론인 마르미-D, 정보 계획 수립 방법론인 마르미-P, 객체 지향 개발 방법론인 마르미-II, 컴포넌트 기반 개발 방법론인 마르미-III 등이 있으며[3], 적용 검토의 대상으로 선정된 마르미-D는 일반적인 경영 정보 시스템(MIS)을 주 대상으로 하는 하향식의 구조적 개발 방법론으로, 개발 계획 단계, 요구 분석 단계, 기본 설계 단계, 상세 설계 단계, 구축 단계, 테스트 단계, 설치 및 인도 단계 등의 7 단계로 구성되어 있다.

개발 계획 단계는 사용자가 요청한 개발 업무에 대하여 개략적인 업무 분석 과정을 거쳐 사용자 업무의 문제점을 도출하고, 사용자로부터 상위 수준의 요구 사항을 끌어내어 이를 해결할 수 있는 개략적인 정보 시스템의 구성을 정의하는 단계이다. 외주 개발의 경우에는 제안 요청서 준비와 제안서 작성 단계로 구분할 수 있는데, 이 단계는 요구 사항을 수집하고 구체화하는 단계이기 때문에 요구 사항의 변경 관리가 적용되지 않는다.

요구 분석 단계는 개발 영역과 수행 업무 및 현행 시스템에 대한 이해와 사용자 요구 사항 분석을 통한 신규 업무 절차 및 필요 정보를 정의하는 단계이다. 따라서 요구 사항 자체의 예러나 요구 사항간의 상충, 설계

나 구현상의 문제 등으로 인하여 제안서 단계의 요구 사항에 대한 변경이 필요하거나 새로운 요구 사항이 발생하는 경우에만 요구 사항 변경 관리를 적용할 수 있다.

기본 설계 단계는 요구 분석 단계에서 파악된 요구 사항을 컴퓨터 시스템의 구현 관점에서 해결 방안을 결정하는 단계로, 설계 작업의 진행 도중 발견되는 요구 사항의 문제점이나 사용자의 변경 요구 등을 지원하기 위하여 요구 사항 변경 관리를 실시하고 요구 사항을 보완한다.

상세 설계 단계는 기본 설계 단계에서 파악된 논리적인 설계안을 토대로 물리적 설계안을 도출하는 단계로, 설계 작업의 진행 도중 발견되는 요구 사항의 문제점이나 사용자의 변경 요구 등을 지원하기 위하여 요구 사항 변경 관리를 실시하고 요구 사항을 보완한다.

구축 단계에는 설계 명세를 바탕으로 프로그램을 원시 코드로 구현하고, 작성된 원시 코드를 단위 테스트를 통하여 검증하는 단계로, 구축 작업의 진행 도중 발견되는 요구 사항의 문제점이나 사용자의 변경 요구 등을 지원하기 위하여 요구 사항 변경 관리를 실시하고 요구 사항을 보완한다.

테스트 단계는 시스템에 대한 모든 요구 사항과 분석, 설계 명세서의 내용이 시스템에 정확히 반영되고 기능들이 정상적으로 수행되는지를 개발자 측면에서 검증하고 사용자가 시스템을 테스트하여 인수하도록 보장하는 단계로, 테스트 작업의 진행 도중 발견되는 요구 사항의 문제점이나 사용자의 변경 요구 등을 지원하기 위하여 요구 사항 변경 관리를 실시하고 요구 사항을 보완한다.

설치 및 인도 단계에는 사용자 승인 테스트가 완료된 후에 기존 운영 시스템을 새로운 시스템으로 전환하고 프로젝트의 모든 산출물을 사용자에게 전달하고 인계하는 단계로, 지금까지 변경된 모든 요구 사항을 반영하여 요구 사항을 재정립하고 요구 사항의 기준선을 다시 설정한다. 여기서 다시 정립된 요구 사항은 발주자에게 인도되어 향후 운영 및 유지 보수에 활용될 수 있다.

4.2.2 산출물에 대한 적용

개발 계획 단계에는 사용(예정)자의 요구 사항을 수집하여 상위 수준의 요구 사항 명세서를 작성한다. 따라서 요구 사항에 대한 변경 관리는 적용되지 않지만, 설계 단계 이후의 요구 사항 변경 관리에 필요한 산출물의 준비를 위하여 상위 수준의 요구 사항 명세서를 보완하여 요구 사항 기술서(개략)를 작성하고 요구 사항 관계표를 추가한다.

요구 분석 단계에는 상세한 요구 사항을 도출, 분석,

명세화하고 테스트 요구 사항을 정의한다. 따라서 이 단계에서도 요구 사항에 대한 변경 관리는 적용되지 않지만, 설계 단계 이후의 요구 사항 변경 관리에 필요한 산출물의 준비를 위하여 기존의 요구 사항 정의서와 테스트 요구 사항 정의서를 보완하여 요구 사항 기술서(보완)를 작성하고 요구 사항 관계표를 추가한다. 그러나 외주 개발의 경우에는 제안 요청서 및 제안서 작성 단계에서 요구 사항이 이미 정리되고 개발자와 발주자간의 계약의 기준으로 사용되었으므로 요구 사항 변경 관리 절차를 적용할 수도 있다.

기본 설계 단계는 단계 준비(D3100), 응용 시스템 설계(D3200), 예비 분산 설계(D3300), 분할 및 할당 설계(D3400), 분산 어플리케이션 할당 설계(D3500), 아키텍처 설계(D3600), 업무 설계(D3700), 단계 점검(D3800)의 8개 활동과 24개 작업으로 구성되어 있으나, 마르미 방법론에는 요구 사항 관련 활동이나 작업이 정의되어 있지 않다. 따라서 이 단계에서는 표 6과 같은 요구 사항 변경 관리 활동과 작업, 그리고 산출물을 추가한다.

상세 설계 단계는 단계 준비(D4100), 데이터베이스 상세 설계(D4200), 프로그램 상세 설계(D4300), 테스트 및 이행 설계(D4400), 단계 점검(D4500)의 5개 활동과 11개 작업으로 구성되어 있으나, 마르미 방법론에는 요구 사항 관련 활동이나 작업이 정의되어 있지 않다. 따라서 이 단계에서도 표 6과 같은 요구 사항 변경 관리 활동과 작업, 그리고 산출물을 추가한다.

구축 단계는 단계 준비(D5100), 프로그래밍(D5200), 단위 테스트(D5300), 지침서 및 교재 개발(D5400), 단계 점검(D5500)의 5개 활동과 14개 작업으로 구성되어 있으나, 마르미 방법론에는 요구 사항 관련 활동이나 작업이 정의되어 있지 않다. 따라서 이 단계에서도 표 6과 같은 요구 사항 변경 관리 활동과 작업, 그리고 산출물을 추가한다.

테스트 단계는 단계 준비(D6100), 통합 테스트(D6200), 시스템 테스트(D6300), 사용자 승인 테스트(D6400), 단계 점검(D6500)의 5개 활동과 15개 작업으로 구성되어 있으나, 마르미 방법론에는 요구 사항 관련 활동이나 작업이 정의되어 있지 않다. 따라서 이 단계에

서도 표 6과 같은 요구 사항 변경 관리 활동과 작업, 그리고 산출물을 추가한다.

설치 및 인도 단계는 단계 준비(D7100), 사용자 교육 실시(D7200), 시스템 설치(D7300), 설치 후 관리(D7400), 단계 점검(D7500)의 5개 활동과 13개 작업으로 구성되어 있으나 마르미 방법론에는 요구 사항 관련 활동이나 작업이 정의되어 있지 않다. 따라서 전달물 검토 및 갱신(D7404) 작업에서 향후 유지 보수를 위하여 지금까지의 모든 요구 사항 변경을 정리하여 요구 사항 기술서를 재 정립하고 요구 사항 기준선을 다시 설정하는 업무를 추가하고, 산출물로 요구 사항 기술서(재정립)와 요구 사항 관계표(수정)를 추가한다.

이러한 방법으로 제안 요구 사항 관리 프로세스를 마르미 방법론에 적용하면, 기존의 각 단계별 활동이나 절차를 변경하지 않고 단지 요구 사항 변경 관리 활동을 추가하거나 보완함으로써 쉽게 적용할 수 있고, 기본 설계 단계 이후에 대한 요구 사항 관리를 보완하며, 설치 및 인도 단계에서 요구 사항 기준선을 다시 설정할 수 있도록 요구 사항 관리의 개선이 가능하다.

4.3 요구 사항 추적 모형의 검증

제 3.3 절의 그림 4와 같은 요구 사항 추적 모형의 수직 및 수평적 상관 관계를 테이블로 변환하면 표 7~표 11과 같은 5개의 테이블이 생성된다.

표 7 수직 요구 사항 테이블

요구 사항	관련 요구 사항
R1	(R2, 0.3) (R4, 0.2)
R2	
R3	(R4, 0.4)
R4	
R5	

표 8 요구 사항 - 설계 테이블

요구 사항	관련 설계 요소
R1	(D1, 0.7) (D2, 0.3)
R2	(D2, 1.0)
R3	(D4, 0.6) (D5, 0.4)
R4	(D6, 1.0)
R5	(D7, 1.0)

표 6 요구 사항 변경 관리 활동의 작업과 산출물

활동	작업	산출물
요구 사항 변경 관리	타당성 평가 영향 범위 분석 소요 자원 산정 관리자 승인	요구 사항 기술서 (보완) 요구 사항 관계표 (수정) 요구 사항 변경 이력서 요구 사항 변경 영향 분석서

표 9 수직 설계 테이블

설계 요소	관련 설계 요소
D1	(D3, 0.3) (D6, 0.2)
D2	
D3	
D4	
D5	(D7, 0.5)
D6	
D7	
D8	

표 10 설계 - 구현 테이블

설계 요소	관련 구현 요소 *
D1	(C1, 1.0)
D2	(C2, 0.5) (C3, 0.5)
D3	(C4, 1.0)
D4	(C4, 0.7) (C7, 0.3)
D5	(C7, 1.0)
D6	(C7, 1.0)
D7	(C9, 1.0)
D8	(C10, 1.0)

표 11 수직 구현 테이블

구현 요소	관련 구현 요소
C2	(C4, 0.2) (C5, 0.2) (C6, 0.2)
C7	(C10, 0.4)
C9	(C8, 0.3)

여기서 수직 상관 관계는 동일한 작업 단계 요소(요구 사항과 요구 사항, 설계 요소와 설계 요소, 구현 요소와 구현 요소)간의 영향 관계를 표시하고, 수평 상관 관계는 다음 작업 단계 요소(요구 사항과 설계 요소, 설계 요소와 구현 요소)와의 영향 관계를 표시한다.

요구 사항의 변경으로 인한 영향 범위를 분석하기 위하여 이들 5개의 테이블에 대하여 넓이-우선 탐색을 적용하면,

- ① RV(수직 요구 사항) 테이블의 탐색을 통하여 변경되는 요구 사항(Ri)으로부터 영향받는 파생 요구 사항과 영향 정도(가중치)를 파악할 수 있다.
- ② RH(요구 사항-설계) 테이블의 탐색을 통하여 ①의 결과로 추출된 모든 요구 사항에 대한 관련 설계 요소와 영향 정도(가중치)를 파악할 수 있다.
- ③ DV(수직 설계) 테이블의 탐색을 통하여 ②의 결과로 추출된 설계 요소들로부터 영향받는 모든 파생 설계 요소와 영향 정도(가중치)를 파악할 수 있다.
- ④ DH(설계-구현) 테이블의 탐색을 통하여 ③의 결과로 추출된 모든 설계 요소에 대한 관련 구현 요소와

영향 정도(가중치)를 파악할 수 있다.

- ⑤ 그리고 CV(수직 구현) 테이블의 탐색을 통하여 ④의 결과로 추출된 구현 요소들로부터 영향받는 모든 파생 구현 요소와 영향 정도(가중치)를 파악할 수 있다.

그런데 개발 단계의 추적 모형은 구현이 완료될 때까지 미완성의 상태로 개발 작업의 진행 단계에 따라 설계와 구현 부분이 점진적으로 완성되게 되므로, 아직 해당 테이블이 존재하지 않으면 그 시점에서 탐색을 중단한다.

이러한 요구 사항 추적 모형에 대한 탐색을 통하여 어떤 요구 사항이 변경되면 이에 파생하는 요구 사항과 영향 정도, 관련된 설계 요소와 영향 정도, 그리고 최종적으로 관련되는 구현 요소와 영향 정도까지 파악할 수 있다. 그러나 이러한 영향 범위 추적 모형은 역 방향의 추적성을 제공하지 못하고, 요구 사항의 변경이 발생할 때마다 전체 테이블의 재구성이 필요하기 때문에 실용적 측면에서는 한계가 있다[20].

4.4 적용 효과 분석

일반적인 요구 사항 관리 프로세스의 적용 효과는 TBI 보고서[19]에 자세하게 기술되어 있는데, 이 논문에서 제안하는 요구 사항 변경 관리 프로세스의 특징에 따른 추가적인 적용 효과는 다음과 같다.

- 각각의 요구 사항을 관리 대상 객체로 설정하여,
 - 요구 사항의 관리 범위를 개발 계획 단계에서부터 마지막 설치 및 인도 단계까지 전체 개발 생명 주기로 확장하고
 - 개발이 진행됨에 따라 구체화되는 요구 사항의 점진적인 명세화를 지원하며,
 - 필요한 경우 각각의 요구 사항을 형상 관리 항목으로 지정할 수 있다.
- 요구 사항 추적 모형에 영향 정도에 따른 가중치 개념을 도입하여,
 - 요구 사항 변경으로 인한 영향 범위를 분석하고 영향 정도를 계량화할 수 있으며,
 - 따라서 보다 정확한 소요 자원 산정을 위하여 활용할 수 있다.
- 소요 자원 산정에 재 작업 거리 개념을 도입하여,
 - 일반적인 변경 관리와 다른 개발 단계 요구 사항 변경의 특징을 반영하고
 - 개발 단계의 진행에 따라 기하급수적으로 증가하는 소요 자원의 산정을 위한 구조화된 틀을 제공한다.

더구나 이 논문에서 제안하는 요구 사항 관리 프로세

스는 개발 방법론적 관점을 바탕으로 하기 때문에, 제 4.2 절에서 살펴 본 바와 같이 별도의 추가적인 절차나 적용 방안의 수립 없이도 폭포수 모형을 기반으로 하는 구조적 개발 방법론에 요구 사항 변경 관리를 직접 적용할 수 있는 방안을 제공한다.

그러나 현행 개발 방법론에 제안 프로세스와 비교 가능한 프로세스가 존재하지 않기 때문에, 여기서는 요구 사항 변경 관리 프로세스를 개발 방법론에 적용하기 위한 조건들을 조사하고 이에 대한 충족 여부의 확인을 통하여 제안 프로세스의 효율성을 평가한다.

풍부한 개발 프로젝트 경험(10년 이상 86.3%, 15년 이상 81.8%)과 실제 정보 시스템 개발 감리 수행 경험(90.9%)을 가진 정보처리기술사 및 정보시스템감리인을 대상으로 한 설문 조사(100명 중 22명 응답) 결과에 의하면, 요구 사항 변경 관리 프로세스를 개발 방법론에 적용하기 위한 주요 요건으로는 ① 요구 사항 변경을 정확하고 효율적으로 관리할 수 있도록 지원하는 기능의 완전성 (36.4%), ② 추가적인 산출물을 최소화하는 적용의 용이성 (22.7%), 그리고 ③ 관련되는 요구 사항에 대한 영향 범위를 추적할 수 있는 변경 영향의 관리 (18.2%) 등이 있다[4]. 그런데 이 논문에서 제안하는 요구 사항 변경 관리 프로세스는,

- 세 가지의 추가적인 산출물(요구 사항 관계표, 요구 사항 변경 내역서, 요구 사항 변경 영향분석서)을 필요로 하지만, 요구 사항 관계표는 요구 공학에서 일반적으로 정의하고 있는 요구 사항 추적표를 보완함으로써, 그리고 요구 사항 변경 내역서는 형상 관리에서 적용되는 버전 부여와 각 버전에 대한 변경 내역 설명을 활용하여 쉽게 작성할 수 있다.
- 그리고 요구 사항 변경 영향 분석은 실제 프로젝트에서 비정형적으로 수행되고 있는 작업들을 문서화하기 위한 절차와 산출물을 제공함으로써, ② 적용의 용이성과 ③ 변경 영향 관리의 요건을 충족한다.
- 그러나, ①의 기능의 완전성 요건은 실제 프로젝트에 대한 적용 시험을 거쳐야 보다 정확한 결론을 도출할 수 있기 때문에 이 논문에서의 검증 및 평가는 미흡하다고 할 수 있다.

하지만 대부분(86.4%)의 설문 응답자가 요구 사항 변경 관리 모형이 개발 방법론 자체의 보완 및 적용 효과 제고에 도움이 될 것으로 예상하고, 90.9%의 설문 응답자가 프로젝트 관리와 시스템 품질 향상에 도움이 될 것으로 예상하고 있기 때문에 전체적으로 이 논문에서

제안하는 요구 사항 변경 관리 프로세스가 개발 방법론에서 미흡한 요구 사항의 관리를 개선하는데 효과가 있을 것으로 결론지을 수 있다.

5. 결론

요구 사항은 시스템 개발의 근간을 이루며 개발 생명 주기동안 지속적으로 변화한다. 그러나 개발 방법론에는 요구 사항의 변경 관리 프로세스가 정의되어 있지 않아 변화하는 요구 사항의 관리를 체계적으로 지원하지 못하고, 요구 공학에서는 요구 사항을 개발 작업 절차와는 별도로 관리하고 구체적인 프로세스가 미흡하여 실제 시스템 개발 프로젝트에 적용하기 위해서는 추가적인 절차의 수립과 적용 노력이 필요하다.

따라서 이 논문에서는 요구 사항의 관리를 위한 요구 공학적 관점을 시스템 개발을 위한 개발 방법론에 통합할 수 있는 요구 사항 변경 관리 프로세스 모델을 제안하여, 하향식의 구조적 개발 방법론에 대한 요구 사항 변경 관리를 개선하는 방안을 제시하였다. 요구 사항 변경 관리 프로세스는 먼저 요구 사항 관리를 위한 객체-관계 모형과, 이 모형을 바탕으로 한 요구 사항 변경 관리 절차와 활동, 그리고 산출물을 통하여 정의하였다.

일반적으로 논리적 검증이 어렵고 실무적 경험과 지식을 필요로 하는 개발 방법론과 요구 공학 분야의 특성을 감안하여, 제안 프로세스를 널리 사용되고 있는 개발 방법론의 하나인 마르미 방법론에 대한 적용 검토와 정보 시스템 개발 경험이 풍부한 정보처리 기술사 및 정보시스템 감리인에 대한 설문 조사를 통하여 평가하였다. 제안 요구 사항 변경 관리 프로세스는 설계 단계 이후의 요구 사항 변경 관리를 지원함으로써, 기존의 개발 방법론에서 미흡한 요구 사항의 변경 관리를 개선할 수 있다.

그러나 이 논문의 연구 범위에는 최근 프로젝트에 적용이 확산되고 있는 객체 지향 개발 방법론과 관련된 연구가 포함되지 않고 제안 프로세스의 실증적 검증을 위한 실무 프로젝트 적용 시험이 미흡하였다. 따라서 향후 이 부분에 대한 추가적인 연구와, 실무 적용을 위하여 소요되는 노력과 비용 규모 평가, 그리고 요구 사항 자체의 품질 향상 방안에 대한 연구를 계속할 예정이다.

참고 문헌

- [1] 강기선, 김재선, 김중원, 홍태기, 박수용, “웹 기반 요구 사항 관리도구의 구현”, 정보과학회 가을 학술발표논문집, Vol. 26, No. 2, pp. 531-533, 1999.
- [2] 이원우, 박수용, 류성열, “객체지향 어플리케이션 개발을

- 위한 UML 기반의 요구공학 프로세스”, 제1회 한국소프트웨어공학 학술대회, pp. 165-172, 1999. 3.
- [3] 교육용 교재 (컴포넌트기반 시스템 개발방법론 마르미-III 공개 발표 및 마르미 / 마르미-II 개발방법론 설명회), 한국소프트웨어산업협회, 2001. 8.
- [4] 신중철, “요구 사항 변경을 지원하는 요구 공학 프로세스의 개선 모델”, 충북대학교 대학원, 박사학위논문, 2002. 8.
- [5] NCA II-AUER-97095, 시스템 개발방법론 적용기준에 관한 연구, 한국전산원, 1997. 12.
- [6] TTA.KO-11.0002, 분석단계 소프트웨어 문서 작성지침, 한국정보통신기술협회, 1998. 12.
- [7] 정통부 고시 2000-13호, 소프트웨어 사업 대가의 기준, 2000. 2. 8.
- [8] 관리기법/1 오브젝트 샘플 (버전 9.5), 한국전산원.
- [9] 왕창중, 소프트웨어 공학, 정익사, 서울, 2000.
- [10] James and Suzanne Robertson, "Volere Requirements Specification Template", Edition 6.1, Atlantic Systems Guild, 2000.
- [11] Arnold, Robert S., and Shawn A. Bohner, *Software Change Impact Analysis*, Reading, MA: Addison-Wesley, 1997.
- [12] Ian Sommerville, *Software Engineering*, 6th Edition, Addison-Wesley, 2001.
- [13] Suzanne and James Robertson, *Mastering the Requirements Process*, Addison-Wesley, 1999.
- [14] Karl E. Wiegers, *Software Requirements*, Microsoft Press, 1999.
- [15] Gerald Kotonya and Ian Sommerville, *Requirements Engineering Process and Techniques*, John Wiley & Sons, 1998.
- [16] Michael Haug, Eric W. Olsen, and Gonzalo Cuevas, *Managing the Change : Software Configuration and Change Management*, Springer Verlag, 2001.
- [17] Dean Leffingwell, "Calculating Your Return on Investment from More Effective Requirements Management", Rational Software Corp. <http://www.rational.com/products/reqpro/>, 1997.
- [18] Larry Boldt, "Managing Requirements at the Object Level", Technology Builders Inc, <http://www.tbi.com/products/asq/>
- [19] Doc. #01 WP0598, "Requirements Management : The Foundation for Solving Common Development Challenges", Technology Builder Inc., <http://www.tbi.com/products/asq/>
- [20] Glenn Stout, "Requirements Traceability and the Effect on the System Development Lifecycle (SDLC)", Nov. 29, 2001, <http://www.stickyminds.com/>
- [21] 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.
- [22] 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.
- [23] INCOSE Tools Survey : *Requirements Management (RM) Tools*, 2001.
<http://www.incose.org/tools/tooltax.htm>



정 규 장

1980년 동국대학교 전자계산학과 졸업
1994년 연세대학교 산업대학원 졸업(전자계산학 전공). 2002년 충북대학교 전자계산학과 졸업(소프트웨어공학 전공-이학박사). 1982년~1991년 서울시 지하철공사 전산실. 1991년~2001년 일신하이테크. 2001년~현재 한국지리정보산업협동조합 센터장. 관심분야는 모바일 GIS, 컴포넌트 소프트웨어, 객체지향 DB, 개발방법론, 요구공학



신 중 철

1976년 서울대학교 전기과 졸업. 1994년 연세대학교 산업대학원 졸업(전자계산 전공). 2002년 충북대학교 전자계산학과 졸업(소프트웨어공학 전공-이학박사). 1978년~1992년 현대엔지니어링 전산실/정보사업부. 1992년~2000년 (주)송우정보 대표이사. 2000년~현재 송우아이엔티(주) 기술연구소장. 관심분야는 개발방법론, 요구공학, 변경관리, 정보시스템 감리



구 연 설

1964년 청주대학교 졸업. 1975년 성균관대학교 경영행정대학원 전자자료처리 전공(경영학석사). 1981년 동국대학교 대학원 통계학 전공(이학석사). 1988년 광운대학교 대학원 전자계산학 전공(이학박사). 1994년~1995년 한국정보과학회 부회장. 1979년~현재 충북대학교 컴퓨터학과 교수. 관심분야는 객체지향 테스트, 품질관리, 정보 검색, 전자상거래