소프트웨어 요구사항 관리 사례 연구

최정은[0], 최순규, 이선아
삼성전자 CTO전략실 소프트웨어센터
(jeunchoi, soonkew, salee)@samsung.com

The Case Study of Software Requirement Management

Jeong-Eun Choi[0], Soon-Kew Choi, Seon-Ah Lee
Software Center, Corporate Technology Operations, Samsung Electronics Co.,Ltd.

요 약

최근 국내 기업들은 S/W Process 심사를 통하여 기업의 Process 수준 향상과 개발 제품의 품질을 향상시키고자 하는 관점에서 CMM , SPICE와 같은 표준에 많은 관심을 보이고 있다. 이러한 표준들에서는 소프트웨어 품질 향상을 위한 기반으로 요구사항 관리를 중요시하고 있다. 많은 기업들이 요구사항을 추출하여 분석하고, 관리하는 방안에 대해 관심을 가지고 적용하고 있다. 요구사항 관리는 요구사항 개발, 변경 및 연계성과 관련된 모든 활동을 포함한다. 즉, 상호적이고 협력적인 공정을 통해서 요구사항을 추출, 분석하고 분석된 결과를 문서화하고, 요구 사항들 간의 연계성을 설정 관리하고, 변경 사항을 관리해야 한다. 본 논문에서는 요구사항 추출 단계에서부터 개발 완료 시점까지 요구사항을 관리하는 소프트웨어 요구사항 프로세스를 제안하고, 이를 적용한 사례를 제시한다.

1. 서 론

최근 국내 기업들은 S/W Process 심사를 통하여 기업의 Process 수준을 향상 시키고자 하는 관점에서 소프트웨어 제품의 품질 향상에 대한 관심이 높아지고 있다. SEI(Software Engineering Institute)의 CMM(Capability Maturity Model)[1], ISO /IEC 15504 TR2(Technical Report Type 2)인 SPICE (Software Process Improvement Capability dEtermination)[2]등에서 소프트웨어 품질 향상을 위한 기반으로 요구사항 관리를 중요시하고 있다. CMM Level 2의 Key Process Area를 보면, Requirement Manage 항목이 있다. 이에 CMM Level 2를 달성하기 위해선 요구사항 관리가 필수적이다. SPICE 심사를 받는 프로세스 중 ENG.1. 개발 프로세스를 가장 많이 심사 받는데 이 중 ENG.1.2가 S/W 요구 분석(Software Requirement Analysis)이다. 따라서, 많은 기업들의 요구사항 관리에 대한 관심은 지속적으로 증대되고 있다. 그러나 이들 표준에는 특정 기술이나 방법론을 반영하지 않은 일반적인 프로세스를 제공하며, 그 수준 역시 개괄적이라서 실제 소프트웨어 개발에 적용하기에 부족함이 있다.

본 연구에서는 소프트웨어 품질 향상을 위해 개발 전 단계를 통한 체계적인 요구사항 프로세스를 제시하여 요구사항 관리의 실용화를 가능하게 한다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 요구사항 프로세스에 대해 알아보고, 3장에서는 실용화 할 수 있는 체계적인 소프트웨어 요구사항 프로세스인 SRP를 제시한다. 4장에서는 이를 적용한 사례를 소개하고, 끝으로 5장에서 적용 효과 및 결론을 제시한다.

2. 기존 연구

요구사항이란 어떤 기능을 가진 시스템을 원하는지, 기능을 수행하는 데에 있어서 제약 받는 사항이 무엇이 있는지를 표현한 것이다.[3,4] 이러한 요구사항을 추출하고 관리하는 것이 요구사항 프로세스이다. Boehm[5], Davis et al.[6], Karl E. Wiegers[7] 등 많은 사람들이 요구사항 프로세스에 대한 모델을 그림 1과 같이 제시하였다.

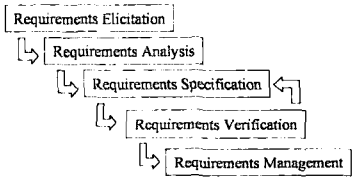

그림 1. 요구사항 프로세스

이러한 요구사항 프로세스에서는 구체적으로 해야 할 작업에 대해 명시하고 있지 않아, 실제 소프트웨어 개발에 적용하기에 어려움이 있다.

Rational의 RUP[8]에서는 요구사항 프로세스 모델을 문제 분석, 고객 요구 파악, 시스템 정의, 시스템 개발 범위 관리, 시스템 세부 내용 정의, 변경 관리로 제시하고 있다. RUP에서 제시하는 프로세스는 세부적인 절차가 있어 많이 적용이 되어 지고 있지만, 각 기업의 적용하기에는 Tailoring이 필요하다.

이에 본 논문에서는 요구사항의 세부적인 절차가 있는 소프트웨어 요구사항 프로세스인 SRP에 대해 제시한다.

3. 소프트웨어 요구사항 프로세스 (SRP)

요구사항을 체계적으로 관리하는 소프트웨어 요구사항 프로세스(SRP)의 전체적인 모습은 그림 2와 같다. SRP는 크게 요구사항 개발 단계와 요구사항 관리 단계가 있다. 요구사항 개발 단계는

요구사항 추출, 분석, 검증하는 절차이다. 요구사항 관리 단계는 요구 사항들이 구현되어 제품화 될 때까지 변경 사항을 유지 관리 하는 절차이다.

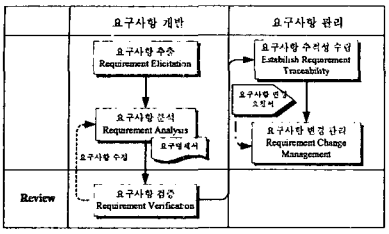

그림 2. SRP

3.1 요구사항 개발 단계

요구사항 개발단계에는 요구사항 추출, 분석, 검증의 작업이 있다. 각각의 세부적인 절차는 다음과 같다.

3.1.1 요구사항 추출

고객이 요구하는 문제를 해결하기 위한 방안을 찾고, 소프트웨어의 개발 범위와 수행해야 하는 기능들에 대해 명시하는 절차로 요구사항 추출 작업의 Workflow는 그림 3과 같다. Interviews[8], Brainstorming[8], 설문 조사[8], Requirements Review[9]등의 요구사항 추출 기법들을 통해 다양한 요구사항을 수집 하고, 수집된 결과를 나열하여 요구사항 리스트를 작성한다. 요구사항 리스트의 내용들이 고객이 요구한 문제를 해결하는 사항들인지를 검증하고, 부적절한 요구사항이 추출되었을 때에는 문제 인식 부족으로 문제 분석 과정부터 다시 수행한다.

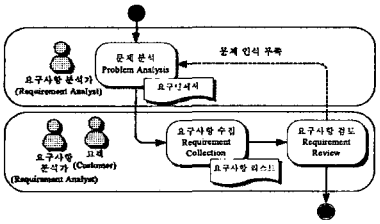

그림 3. 요구사항 추출 Workflow

3.1.2 요구사항 분석

요구 사항들을 구현 여부와 기능적인 사항과 비기능적인 사항들로 구분하는 분석 작업을 하고, 요구사항들간의 우선 순위를 결정하고 요구사항을 문서화하는 절차이다. SRP에서는 요구사항 분석 과정을 구조적 방법론적인 것과 객체지향 방법론적인 Use Case Driven 방식[9,10]으로 제시한다. Use Case Driven 방식으로 요구사항을 분석하면, Actor와 Use Case를 추출하고, Use Case Diagram을 작성하는 절차를 수행한다. 요구사항 분석 작업의 Workflow는 그림 4와 같다. 요구사항 분석 작업을 수행 시에는 요구사항 타입과 속성에 대한 지식, 우선 순위를 결정하는 기법[8], UML[8,9]에 대한 지식들이 적용된다. 요구사항 분석 작업에서 작성하는 요구명세서는 IEEE Std 830 표준[11]의 내용을 기 반으로 조직의 특성에 맞는 요구명세서 Template을 적용한다.

3.1.3 요구사항 검증

요구사항이 반영되어 요구명세서에 기술되었는지를 검토하고, 개발 소프트웨어의 전체 모습을 볼 수 있는 Use Case Diagram이 고객이 원하는 사항 대로 그려졌는지를 검토하는 절차이다.

이 작업에서는 Inspection[8], Walkthrough[8]등과 같은 검토 기법을 통하여 요구명세서를 검토하고, 요구사항 검증을 위한 테스트 케이스를 선정하는 작업을 한다. 이때 선정된 테스트케이스는 개발된 소프트웨어를 검증하는 것과 요구사항 검증 수단으로 사용된다. 요구사항 검증 작업에 대한 Workflow는 그림 5와 같다.

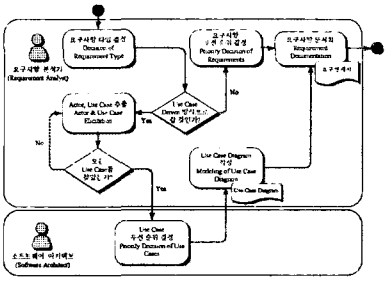

그림 4. 요구사항 분석 Workflow

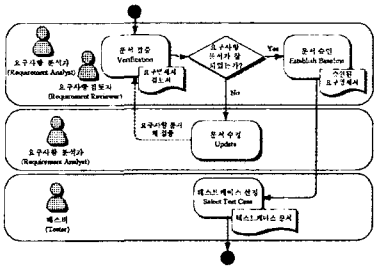

그림 5. 요구사항 검증 Workflow

3.2 요구사항 관리 단계

요구사항 관리 단계에는 요구사항 추적성 수립과 요구사항 변경 관리의 작업이 있다. 요구사항 관리 단계에서는 요구사항 관리 자동화 도구를 사용 할 수 있다. 자동화 도구에는 Doors[12], RequisitePro[13], CaliberRM[14] 등이 있다. 자동화 도구를 사용하여 요구사항을 관리하면 요구사항 관리에 드는 시간과 비용을 절감할 수 있고, 요구 사항들을 데이터베이스로 관리하여 기업의 자산이 될 수 있다.

3.2.1 요구사항 추적성 수립

승인된 요구 사항들을 전자 매체에 저장하고, 요구사항 추적 매트리스(RTM:Requirements Traceability Matrix)를 작성하여 요구사항을 관리한다. 이 작업에 대한 Workflow는 그림 6과 같다. 이 작업에서 작성될 수 있는 RTM에는 다음과 같은 것들이 있다.

- 요구사항 서로간의 관계 RTM
- 요구사항과 Use Case 간의 관계 RTM
- 요구사항/Use Case와 구조 설계 모듈간의 관계 RTM
- 요구사항/Use Case와 테스트케이스간의 관계 RTM
- 요구사항/Use Case와 상세 설계의 클래스간의 관계 RTM

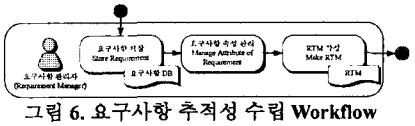

그림 6. 요구사항 추적성 수립 Workflow

3.2.2 요구사항 변경 관리

요구사항 변경이 요구되었을 때 소프트웨어의 개발 범위와 구현 가능성을 고려하여, 요구사항을 변경하여 개발되어지고 있는 소프트웨어의 변경된 요구사항이 잘 반영되도록 하는 작업이다. 이 작업에 대한 Workflow는 그림 7과 같다.

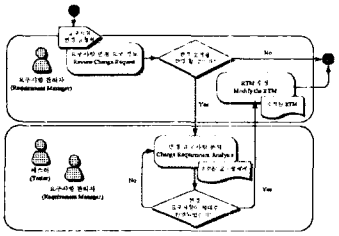

그림 7. 요구사항 변경 관리 Workflow

4. SRP 적용 사례

본 장에서는 3장에서 제시한 SRP를 소프트웨어 개발에 적용한 사례를 소개한다. SRP를 적용하여 수행한 활동들을 정리한 것은 그림 8과 같다.

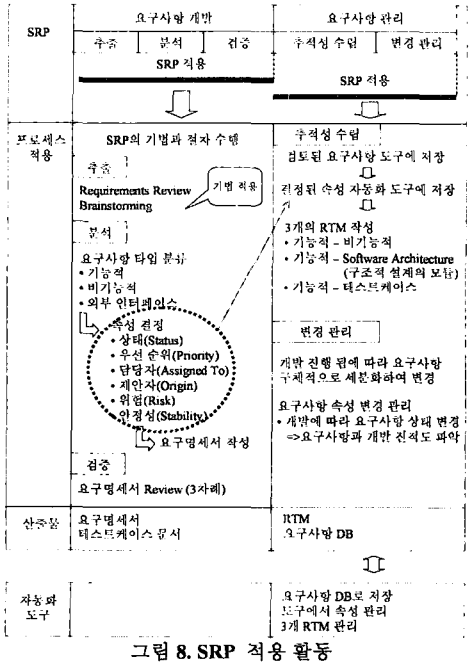

그림 8. SRP 적용 활동

SRP를 적용하여 요구사항을 관리함으로써 프로젝트 관리측면에서 얻은 효과가 있다.

- 명확한 요구명세서가 작성됨으로써 프로젝트 관리자가 프로젝트 개발 일정을 명확히 파악하고, 계획 할 수 있었다.
- 요구사항과 모듈간의 RTM을 작성하여 관리 함으로써 개발자 각자의 역할에 대해 요구사항을 기반으로 분명히 파악 할 수 있었다.
- 개발 완료 시점에서 시작되는 테스트케이스 작성을 개발 초기에 요구사항을 기반으로 작성 할 수 있었고, 요구사항과 테스트케이스간의 RTM을 작성하여 요구사항에 대한 검증도 할 수 있었다.
- 개발이 진행됨에 따라 요구사항 속성의 상태를 변경하면서 개발 진척도를 알 수 있었다.

그림 9는 요구사항 관리를 시작한 시점부터 요구사항 속성의 상태 변경 상황을 그래프로 표현한 것이다. 요구사항 속성의 상태에는 Proposed, Approved, Canceled, Developing, Incorporated, Postponed, Developed, Validated 8가지 상태를 두었다. 그림 9에서 보면, 요구사항이 제시되면, Proposed/Approved 상태로 되고, 개발이 됨에 따라 Developing / Developed 상태로 변경되면서 요구사항이 개발에 반영된다는 점을 알 수 있다.

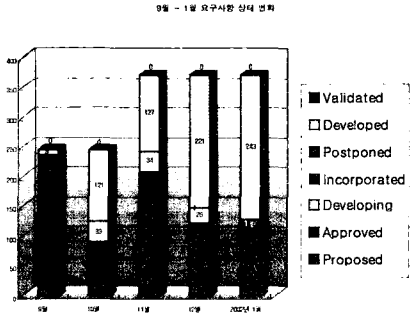

그림 9. 요구사항 상태 변화 Graph

5. 결론

본 논문에서는 소프트웨어 요구사항 프로세스 즉, SRP에 대한 구체적인 작업 과정과 이를 적용한 사례를 소개하였다. SRP를 적용하여 요구사항을 관리 함으로써 첫째, 프로젝트 개발 일정을 명확히 할 수 있었다. 둘째, 프로젝트 관리자가 개발자 각자의 역할에 대해 분명히 파악 할 수 있었다. 셋째, 개발이 진행되는 척도를 요구사항 상태 변화로 알 수 있었다. 넷째, RTM을 작성하여 요구사항과 관련이 되어 있는 모듈, 테스트케이스를 파악 할 수 있었다.

본 논문에서 소개한 사례는 자동화 도구를 사용하여 요구사항을 관리하였다. 이번 사례에서는 RTM 작성을 구조적 설계의 모듈까지 연계하였다. 향후 실제 RTM 작성 범위를 상세 설계의 클래스 단계까지 요구사항과 매핑하여 관리할 것이다.

[참고 문헌]

[1] Paulk, M. C. et al, *The Capability Maturity Model: Guidelines for Improving the Software Process*, Addison-Wesley Pub Co., 1995.

[2] ISO/IEC 15504 TR2, *Software Process Assessment and Capability determination*, ISO/IEC, 1998.

[3] Richard H. Thayer and Merlin Dorfman, *Software Requirements Engineering*, Second Edition, IEEE Computer Society, 2000.

[4] Roger S. Pressman, *Software Engineering-A Practitioner's Approach*, Fourth Edition, McGrawHill, SciTech, 1998.

[5] Boehm, B. W., "A Spiral Model of Software Development and Enhancement," *Computer*, Vol. 21, No. 5, May 1988, pp.61-72.

[6] Davis, A.M., E.H. Bersoff, and E.R. Comer, "A Strategy for Comparing Alternative Software Development Life Cycle Models," *IEEE Trans. Software Eng.*, Vol. 14, No. 10, Oct.1988, pp. 1453-1461.

[7] Karl E.Wiegers, *Software Requirements*, Microsoft Press, 1999.

[8] Dean Leffingwell, Don Widrig, *Managing Software Requirements, A Unified Approach*, Addison-Wesley, 2000.

[9] Grady Booch, Jams Rumbaugh, Ivar Jacobson, *The Unified Modeling Language User Guide*, Addison-Wesley, 2000.

[10] Bruce Powel Douglass, *Real-Time UML second Edition: Developing Efficient Objects for Embedded Systems*, Addison-Wesley, 2000.

[11] IEEE Std 830-1993, *Recommended Practice for Software Requirements Specifications*, Software Engineering Standards Committee of the IEEE Computer Society: New York, NY, 1993.

[12] G. Gorman, "The Benefits of an Extensible, Customizable Environment", Telelogic North America Inc., April 1998.

[13] http://rational.co.kr/products/requisitepro.asp

[14] http://www.tbi.com/caliberrm/rmintegrations.cfm.