

CBD의 Process 개선을 위한 Opportunity Tree 설계

최승아⁰, 이경환, 이은서

중앙대학교 컴퓨터 공학과 소프트웨어공학연구소
{ sachoi⁰, kwlee, eslee }@object.cau.ac.kr

Opportunity for CBD Process improvement

Songah Choi⁰, Kyungwhan Lee, Eunseo Lee
Lab. of Software Engineering, ChungAng university

요 약

컴포넌트 기반 기술의 장점을 최대한 살리기 위해서는 컴포넌트 기반 개발 프로세스 및 관리 프로세스가 체계적으로 통합 관리 되어야 한다. 체계적인 개발 프로세스의 부재 속에서 개발된 컴포넌트는 자신의 최대 이점인 소프트웨어의 재사용과 품질의 향상을 끌어내지 못하기 때문이다. 이에 본 리포트에서는 체계적인 개발 프로세스에 대한 Opportunity Tree를 제안함으로써 컴포넌트를 개발하고자 하는 이들의 CBD에 대한 이해를 높이고, 개발 과정의 주요 프로세스를 효과적으로 수행할 수 있도록 한다.

1. 서론

컴포넌트 기반 기술은 기존의 소프트웨어 개발의 생산성을 향상 시키고, 소프트웨어의 품질을 증대시키며, 개발 비용 및 일정의 단축을 유도한다. 이러한 컴포넌트 기반 기술의 장점을 최대한으로 살리기 위해서는 컴포넌트 기반 개발 프로세스 및 관리 프로세스가 체계적으로 통합 관리 되어야 한다. 체계적인 개발 프로세스의 부재 속에서 개발된 컴포넌트는 자신의 최대 이점인 소프트웨어의 재사용과 품질의 향상을 이끌어내지 못하기 때문이다.[4]

이에 본 리포트에서는 CBD의 개발의 주요 issue들을 바탕으로 개발의 핵심 프로세스에 대해 Tree 형식의 단계적 접근을 제시한다. Opportunity Tree를 통해 CBD 개발의 핵심 프로세스에 대한 이해를 높이고, 이의 응용을 용이하게 하여 개발 활동에서 생길 수 있는 결함을 줄이고, 컴포넌트의 재사용성과 품질을 높일 수 있는 방법을 제안하고자 한다.

2. 기초연구

2.1. CBD

CBD(Component Based Development)는 프로그래밍 위주의 전통적인 정보공학 개발 방법론과는 달리 이미 만들어진 컴포넌트를 기본 아키텍처와 설계도에 따라 조립하는 방식의 새로운 개발 형태이다. 소프트웨어 컴포넌트를 조립해 새로운 어플리케이션을 만들 수가 있으므로, 개발 기간을 단축 할 수 있고, 기존의 컴포넌트를 재사용함으로써 생산성과 경제성을 높일 수 있다. 개발하려는 시스템의 전체를 컴포넌트화 할 수 있는 여러 개의 단위로 분리해 개발하므로 복잡성을 단순화 시킬 수 있고, 초기 개발을 실질적으로 수행 할 수 있도록 해준다. 컴포넌트는 캡슐화 되어 있어 로직상의 에러나 런타임 에러 등의 범위를 컴포넌트로 한정할 수 있어 유지보수가 용이하다. 이러한 장점이 부각되면서 현재 컴포넌트들의 재사용과 품질을 체계적으로 지원하기 위한 방법이 다각적으로 모색되고 있다.

2.2 OT(Opportunity Tree)

OT는 사용자에게 Formal한 방법을 통해 효율적인 결

정을 내리게 하여 SE의 목적을 달성하게 하는 것이다. 조직 내에서 어떤 문제가 발생하여 이를 개선하기 위한 전문가가 원하는 벤치마킹과 정보를 효율적으로 찾고, 그 해결책을 Tree 형식으로 제안하는 방법이다. OT는 문제정의 범위와 해결책을 더욱 명확히 제시하며, 문제를 해결할 수 있는 방안들의 우선순위를 통한 최적의 해결책을 제시한다. 이를 위해서는 OT의 Goal을 정의하고 Goal을 이끌어내기 위한 문제점들과 주요 과제를 정리하여, 이에 부합하는 OT를 제시한다.

2.3 SPI(S/W Process Improvement)

체계적인 절차와 방법을 적용하면 효과적으로 고품질의 소프트웨어를 개발할 수 있다는 가정에서 출발한 것이 프로세스 개선 관점이다. 즉, 조직은 소프트웨어 개발 프로세스를 개선시킴으로써 효율성을 높일 수 있고 결과적으로 고품질의 소프트웨어를 얻을 수 있으리라는 것이다. 소프트웨어 개발에서의 프로세스는 소프트웨어 개발 조직의 목표를 달성하기 위해 사용되는 자원(Resource), 활동(Activity), 방법(Method), 실무지침(Practices)을 일컫는다. 소프트웨어 개발 조직은 조직의 정의된 업무 요구를 효과적으로 만족시킬 수 있는 프로세스들을 체계적으로 수립할 필요가 있다. 소프트웨어 프로세스 개선(Software Process Improvement)활동은 프로세스에 대한 이해를 높이기 위해 수행되는 모든 노력 및 활동을 말한다.

3. CBD의 PI를 위한 OT설계

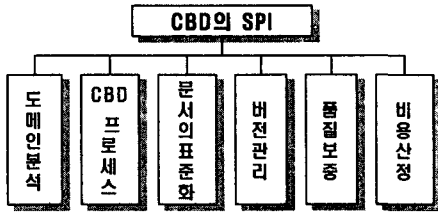
3.1. 개념

본 리포트에서 제시하고자 하는 OT는 컴포넌트를 개발하는 개발자 입장을 중심으로 소프트웨어 프로세스 개선 관점에서 CBD 프로세스의 주요 issue로 대두되고 있는 프로세스들에 대한 이해를 높이고, 이의 효과적인 수행을 돕는다. 나아가서 개발 시 제기되는 의문점이나 문제에 대한 해결 방법에 대한 접근을 돕는다.

3.2. OT 대상 항목

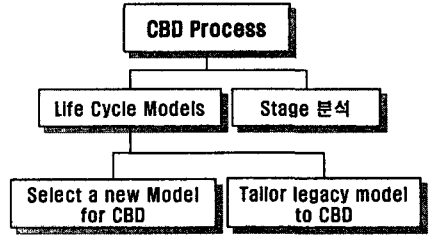
OT의 대상 항목은 크게 6개로 나뉜다. 엔지니어링분야를 위한 도메인 분석과 CBD Process 과정과

Management분야를 위한 문서의 표준화, 버전관리, 품질 보증, 비용 산정이다. 이를 OT 형식으로 표현하면 [그림1]과 같다.



[그림 1 OT의 1레벨]

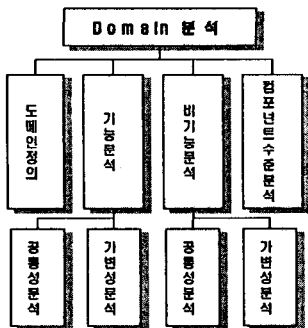
의 설계 등을 이뤄낼 수 있다. 정확한 작업 분석과 설계야말로 좋은 소프트웨어를 만들기 위해 꼭 필요한 조건이다.



[그림 3 CBD Processes]

3.2.1. 도메인 분석(Domain Analysis)

도메인 분석은 도메인에서 해결해야 할 문제, 즉 도메인 요구사항을 인식하고 추출하는 과정이다. 도메인 정의에서는 도메인의 목적과 범위를 정의하고, 도메인 설계에서는 도메인 요구사항을 해결하기 위해 컴포넌트를 추출하고, 그들로 구성된 도메인 아키텍처를 개발한다. 여기서 제안하는 것은 도메인 정의와 설계 시 고려해야 하는 4가지의 프로세스를 제시한다.



[그림 2 도메인 분석]

도메인 정의는 도메인의 목적과 범위를 정의하는 것이다. 기능 분석은 컴포넌트가 달성해야 할 기능적인 요소를 추출해 내고, 비기능 분석은 개발 시 기능적인 요소가 아닌 환경적인 요소, 즉 Resource나 개발 환경, 조직의 Vision등을 분석한다. 컴포넌트의 수준 분석은 개발 환경에 대한 분석과 함께 현재 개발하고자 하는 컴포넌트의 레벨을 설정하는 것이다. 기능, 비기능의 분석에는 공통성과 가변성 분석이 공통적으로 수행되어야 한다.

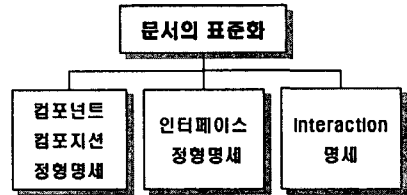
3.2.2. CBD Process

CBD Processes는 개발 과정에서 수행되어야 할 프로세스를 제시한다. 이는 컴포넌트의 개발뿐만 아니라 일반 소프트웨어 개발에서도 중요한 과정으로써 본 레포트에서는 CBD 개발자들이 영두 해두어야 할 과정을 다 음[그림 3]과 같이 제시하였다.

Life Cycle Model을 선정하는 것은 개발과정의 체계를 세우고, 소프트웨어 개발 과정에서 어떤 점을 중요시 할 것인지 우선순위를 세운다. Life Cycle Model은 새로운 모델을 도입하거나 만드는 방법과 기존의 쓰고 있던 모델을 CBD나 현재 프로젝트에 맞게 Tailoring하는 방법이 있다. 두 번째 항목은 Stage분석으로 개발 작업을 세분화하여 분석하는 것이다. 이 과정을 통해 WBS

3.2.3. 문서의 표준화

컴포넌트의 재사용성을 높이기 위해 수반되어야 할 작업이 바로 각종 문서의 정형화 과정이다. 더 나아가서 표준화를 통해 컴포넌트를 자산화 하는 것이 가능하다. 표준화된 문서들을 통해 컴포넌트 가변성간에 존재하는 불일치성을 줄이고, 가변성의 추적을 용이하게 한다. 정형영세와 컴포넌트 모델링의 산출물은 컴포넌트의 생산성을 높이고 컴포넌트의 재사용성을 높일 수 있다. 문서의 표준화에 대한 항목은 [그림4]와 같다.

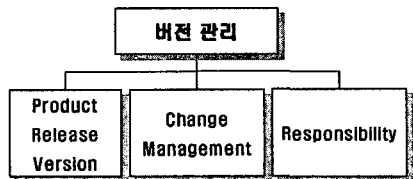


[그림 4 문서의 표준화]

컴포넌트의 정형 영세에는 컴포넌트의 기능과 구조, 역할을 설명하는 컴포지션 영세와 인터페이스들에 대한 인터페이스 정형영세가 필요하다. 그리고 컴포넌트와 인터페이스, 컴포넌트와 다른 컴포넌트와의 관계를 기술하는 Interaction 영세가 존재한다. 각각의 정형화된 문서는 개발되는 컴포넌트에 대한 이해와 활용도를 높 이는데 중요한 역할을 한다.

3.2.4. 버전관리

컴포넌트를 개발하고 이를 상품화 하거나, 재사용하고자 한다면 빠질 수 없는 것이 버전의 관리이다. 체계적인 버전의 관리와 Release Plan이 필요하다. 버전관리의 주요 항목은 [그림 5]와 같다.



[그림 5 버전관리]

가장 우선적으로 생각되어 질 수 있는 것은 제품의 Release 버전 관리이다. 소프트웨어를 개발하고, 어떤 식으로든 변화 혹은 Upgrade 사항이 발생하면 즉각적

으로 version Up이 되어야 하고 이렇게 업그레이드된 version의 배포에 대한 전략이 수립되어야 한다. 또한 그 변화들의 히스토리나 관리가 이루어져야 한다. 이러한 변화들과 배포 전략간의 관계도 미리 정의해 놓고 관리해야 한다. 마지막으로 이런 변화와 배포에 대한 책임 관계를 명확히 해야 한다. 이것은 더 나아가 상품화된 컴포넌트의 Licensing 문제와도 관련이 있으므로, 그 책임 관계를 명확히 하고 책임의 우선순위를 따지는 것은 중요한 일이다.

3.2.5. 품질 보증

품질 보증은 해당된 프로젝트의 특성에 맞도록 관리하고 품질 보증 지침에 의거하여 실행된다. 소프트웨어 품질 활동은 제품 및 프로세스를 대상으로 평가하고 공급자와 획득자의 입장에서 평가한다. ISO 9126 모형에서 제시한 품질 특성에 따라서 공급자의 제품 품질 평가는 적합성 및 기능성을 평가하고, 획득자가 정보처리 기능을 획득하기 위해서 타당한가를 평가한다.[3]



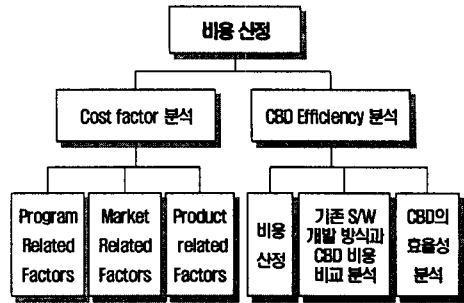
[그림 6 품질 보증]

본 OT에서는 품질 보증을 위한 기본 2단계를 제시한다. 품질 목표 설정과 품질 Item 선정이 그것이다. 소프트웨어의 품질 보증 과정 중에서 컴포넌트 개발을 위해서 특히 신경을 써야 할 부분이다. 단순한 일회용의 소프트웨어 개발이 아니고 지속적인 재사용과, 다른 시스템이나 컴포넌트와 통합되기 위해서는 세밀하게 품질의 수준을 결정하는 일이 필요하다.

level 1의 아래 항목의 6가지 요인은 ISO 9126 모델을 참고 하였다. 각 품질 요인별로 목표를 정하고, 그 요인별 목표를 이루기 위해서 필요한 목표 Item을 선정한다. 프로젝트의 특징에 따라 품질 평가 요인은 다른 모델을 사용해도 무방하다.

3.2.6. 비용산정

기업의 목표 중 중요한 부분을 차지하고 있는 것이 이 창출이다. 그렇기 때문에 S/W를 개발하는 과정 중에서 빼놓을 수 없는 부분이 바로 비용 산정이다. 비용을 미리 예상해보고, 그 비용 요인을 분석하고, 실제 비용을 산정하여 투자의 효과를 분석해 보는 것은 매우 중요한 일이다. CBD 방법을 도입 하기위해 필요한 비용이 무엇인지를 분석하고 미리 비용을 추정해 보고, 실제 소비된 비용과 비교해본다. 그리고 CBD와 다른 S/W 개발 방법의 비용을 비교하여 CBD의 효율성을 추산하는 것이 필요하다. 본 OT에서는 비용의 계획 과정 중 비용 요인의 분석과정과 효율성 분석 과정으로 비용 산정 과정을 다루었다.



[그림 7 비용산정]

Cost Factor의 분석은 크게 세 개의 카테고리로 나누었다. 개발에 관련된 비용을 주로 다루는 Program Related Factors와 판매와 마케팅 비용을 다루는 Market Related Factors, 그리고 CBD 개발 후 사용하기 위해서 드는 비용과, 재사용시 필요한 비용, Licensing등에 관한 비용을 다룬 Product Related Factors이다. 이 세 가지 카테고리로 비용을 분석하고 비용을 추정하게 된다. [2]

CBD Efficiency부분에서는 실제로 비용을 산정하고 기존의 개발 방식과 CBD 개발 방식에 소요 되었던 비용을 비교 분석한다. 이를 토대로 CBD의 효율성을 분석하는 과정을 수행하게 된다. CBD를 사용하면서 재정적인 장단점을 파악하여, 자신의 환경과 시스템에 맞게 CBD 방법론을 tailoring 해나가는 과정도 필요하겠다.

4. 결론

CBD 방법론에 대한 연구는 아직도 활발히 이뤄지고 있는 중이다. 단지 CBD의 효과적인 사용이 부재할 뿐이다. 현재 많은 단체들이나 기업체들이 CBD 방법론을 도입하려 하지만, 기초 지식이나 경험 없이는 컴포넌트의 기능을 하는 컴포넌트를 만들어 내기란 힘이 들다.[8] 컴포넌트를 효과적으로 개발하고 그 목적에 맞게 사용하는 방법에 대한 하나의 대안으로써 본 리포트에서는 Opportunity Tree를 제안하였다. 쉽게 접근하기 힘들었던 CBD의 개념이나 CBD의 프로세스를 트리 형식으로 보여주고 있다. OT는 웹을 통해 각 분야 전문가의 의견을 들을 수 있고, S/W를 개발하면서 부딪치게 되는 문제점에 대한 해결책을 제시한다. 이를 통해 CBD 프로세스를 이해하고, 효과적으로 CBD 방법론을 사용할 수 있을 것이다.

5. References

- [1] George T. Heineman, William T. Councilll "Component-Based Software Engineering"
- [2] CMU/SEI-2000-TR-010, ESC-TR-2000-010 "An Activity Framework for COTS-Based Systems"
- [3] 이경환 "소프트웨어 공학"
- [4] 장운정 "품질 개선 모델을 적용한 컴포넌트 기반 개발 및 관리에 관한 실험적 연구"
- [5] University of southern California Center for Software Engineering "USC CSE's Annual Research Review and Focused Workshop on COTS-Based Systems"
- [6] KSPICE 심사인 협회. 2001년 6월 30일 "KASPA Workshop"
- [7] 이은서 "컴포넌트의 재사용을 위한 공통성과 가변성 추출에 관한 연구"
- [8] T.Ravichandran and Marcus A. Rothenberger "Software Reuse Strategies and Component Markets"
- [9] Ali h. Dogru, Murat M. Tanik "A Process Model for Component-Oriented Software Engineering"