

6시그마와 MDA 기반 모델 변환 기법을 이용한 프로젝트 품질과 생산성 향상에 대한 연구

주태중⁰, 김남욱, 황종선
고려대학교 컴퓨터과학기술 대학원
taejoong@korea.ac.kr⁰, zigzag@samsung.com, hwang@disys.korea.ac.kr

A Study for Improving Project Quality and Component Productivity using Six Sigma with MDA based Model Transformation Technique

Tae-Joong Joo⁰, Nam-Wook Kim, Chong-Sun Hwang
Graduate of Computer Science and Technology, Korea University

요 약

소프트웨어 산업 사회에서 프로젝트의 품질과 생산성을 높이는 문제에 대한 여러 가지 방안들을 제시하고 있으며, 컴포넌트 기반 개발 프로세스 역시 생산성 향상에 대한 가능성을 제시하고 있다. 하지만 고객 니즈에 맞는 고품질의 프로젝트를 수행하기 위해서는 체계적인 관리 프로세스가 필요하며 더욱이 컴포넌트 호환 및 상호 운용에 대한 문제점을 해결하기 위한 많은 노력이 필요하다. 본 논문에서는 이러한 문제점들을 해결하기 위한 노력들 중의 하나로 Six Sigma와 OMG의 MDA(Model Driven Architecture) 기술을 이용하여 특정 도메인의 모델 변환(Model Transformation) 구조를 구현하고, 최적화된 Six Sigma 관리 프로세스를 적용한 실제 사례를 통하여 프로젝트의 품질과 생산성 향상에 대한 결과를 측정하여 그 효과에 대해서 검증하였다.

1. 서론

프로젝트 생산성과 품질의 향상, 고객의 신뢰도를 높이는 일은 소프트웨어 산업 사회에서 가장 중요한 일이며 특히 고객의 니즈를 정확히 파악하고 그에 맞는 고품질(High Quality)의 서비스를 제공하는 것은 프로젝트 수행의 최대의 목표가 된다. 그러나 현재까지는 표준화된 프로젝트 관리 프로세스 미흡으로 인하여 고객의 요구사항 변경과 프로젝트 진행상황, 변경사항, 프로세스 준수율 및 생산성 등에 대한 파악이 쉽지 않아, 결과적으로 고객이 요구하는 고품질의 프로젝트를 수행하는데 어려움이 발생한다. 소프트웨어 컴포넌트 기술이 발전함으로써 소프트웨어 산업의 생산성 향상에 새로운 가능성을 제시하고 있고, 실행 가능한 컴포넌트들을 개발하고, 특정한 목적에 따라 컴포넌트들의 조립을 통하여 새로운 소프트웨어 시스템을 손쉽게 구축할 수 있게 되었다. 하지만 컴포넌트 기술의 낮은 성숙도로 인해서 많은 문제점들도 발생하고 있다. 많은 경우 단순한 UML을 이용한 모델링, CBD 방법론의 적용, EJB나 COM+과 같은 컴포넌트 기술을 이용한 바이너리 컴포넌트의 구현 자체를 컴포넌트 기술의 최종 목표로 오해하고 있다. 이로 인해 지금까지 컴포넌트 간의 상호 운용은 동일 플랫폼 상에서 바이너리 컴포넌트 수준으로 운용되고 있다. 이 경우 이기종 플랫폼이나 기반 기술과 같은 다양한 환경에 대한 표준화된 접근을 간과함으로써 실제 컴포넌트 기술에 기대했던 효과를 얻지 못하고 있는 것이다.

본 논문에서는 프로젝트 품질을 향상 시킬 수 있는 해결방안으로 Six Sigma의 DFSS 방법론을 제시하고 컴

포넌트 개발 생산성에 대해서는 OMG의 MDA 기술의 응용을 통해서 제시하였다.

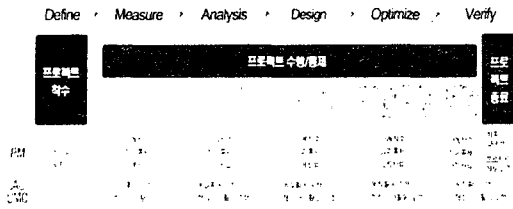
특정 도메인의 소프트웨어 아키텍처에서 요구하는 기반 환경 요소 및 품질 요소들을 효율적으로 컴포넌트 시스템에 반영할 수 있는 모델 변환(Model Transformation)을 Six Sigma의 DFSS 방법론과 더불어 실제 적용한 프로젝트 사례를 통해서 그 효과에 대한 검증을 수행하였다.

2. 관련 연구

2.1 Six Sigma(DFSS)

Six Sigma의 DFSS 방법론은 프로젝트 프로세스 설계 단계에서 CTQ(Critical to Quality) 하위 프로세스에 대한 최적의 디자인을 함으로써 프로젝트 오류를 제로화하고 Six Sigma 수준의 품질을 달성하게 하는 방법론이다. DIFF의 구성은 [그림 1]과 같이 총 6가지 시스템으로 이루어진다.

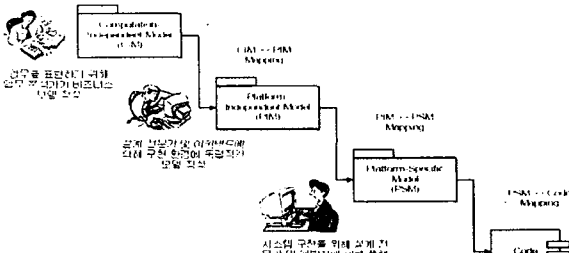
- Define: 프로젝트 착수, 프로젝트 선정, 프로젝트 승인 프로젝트 계획서 작성
- Measure: 요구분석 및 프로젝트 수행통제, 고객의 요구사항을 정의하고 우선순위화, CTQ도출
- Analysis: 분석 및 프로젝트 수행통제, 요구사항에 맞는 설계방안 정의, 최적안 선정, 개념설계
- Design: 설계 및 프로젝트 수행통제, 상세 설계
- Optimize: 개발 및 프로젝트 통제, 신뢰성, 오류방지 등 설계변경, 튜닝을 통한 성능 최적화
- Verify: 구현 및 프로젝트 종료, 이관 및 결과 정리



[그림 1] Six Sigma DFSS 개발 프로세스

2.2 MDA (Model Driven Architecture)

MDA(Model Driven Architecture)는 OMG(Object Management Group)에서 2001년 9월에 기술 표준으로 제정한 소프트웨어 아키텍처 표준 모델로서, 각종 플랫폼 및 기술 요소에 대한 상호 운용성(Interoperability) 문제를 효과적으로 해결하기 위한 새로운 시스템 통합화 방안이다. MDA는 OMG가 현재까지 추진한 여러 표준들(UML, MOF, CWM, XMI, CORBA)을 기반으로 모든 컴포넌트 기술 요소의 표준 메타 모델(Meta Model)을 정의하고 각 구성요소를 정의함으로써 모델 중심의 시스템을 개발하여 모든 컴포넌트 기술 요소들에 대한 호환성과 상호 운용성을 보장한다. 그리고 그 핵심은 [그림 2]와 같이 메타 모델을 기반으로 구현 플랫폼 환경에 독립적인 모델(Platform Independent Model : PIM)을 각 플랫폼 환경에 종속적인 모델(Platform Specification Model : PSM)로 자동으로 변환할 수 있는 구조를 정의한다.

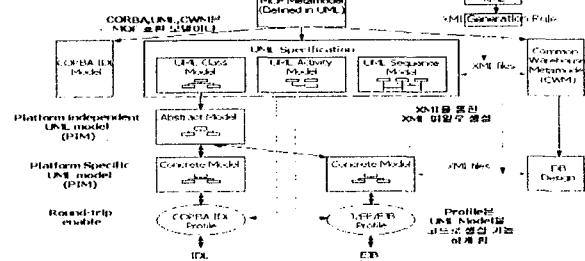


[그림 2] MDA 모델 관계성 및 개발 프로세스

2.3 UML Profile 및 Meta Modeling

UML 모델을 이용하여 특정 도메인에 존재하는 특징들을 효과적으로 표현하기 위해 확장 메커니즘을 정의할 수 있으며, 이것이 UML Profile 확장이다. 그리고 UML Profile 확장에서는 모델 요소들의 스테레오 타입(Stereotype) 및 Tagged Value 등의 요소들을 이용하여 다른 메타 모델의 모델들이나 도메인들을 서로 시각적으로 차별화 시킬 수 있다. [그림 3]과 같이 UML 모델에서 특정 구현 언어 기반의 소스 코드를 자동 생성하거나, 생성된 어플리케이션들에 요구되는 각종 패턴에 대한 품질 요소들을 보장하기 위해 UML Profile 확장을 통해 실현할 수 있다. 즉, UML 명세를 기반으로 확장을 원하는 형태로 추상적(PIM), 구체적 모델(PSM)을 작성하여 UML Profile

을 확장한다. 이 과정에서 각 모델 및 모델 간의 변환 구조는 메타 모델링을 통해서 이루어진다. 따라서 UML Profile의 확장 및 메타 모델링을 통해서 특정 구현 언어에 대해 요구되는 형태의 소스코드를 자동 생성하고, 기반 프레임워크 등과 같은 구현 환경 요소에 대한 정보들을 모델 및 소스코드에 자동 반영할 수 있으며, 이를 통해 어플리케이션에 대한 일관된 품질을 만족시킬 수 있다.

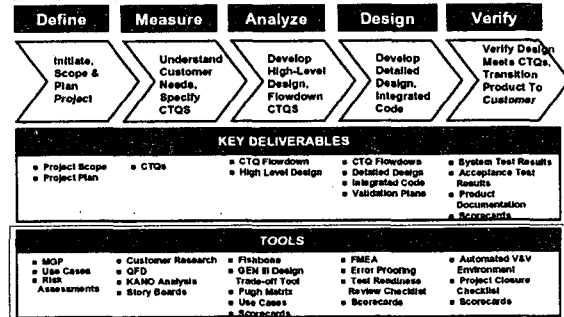


[그림 3] Meta Modeling과 UML Profile을 통한 어플리케이션 구현

3. 적용 사례

3.1 S사 영업정보 S/W 개발 프로세스 분석

일반적인 S/W 개발방법론과 사내 통제 프로세스를 가지고 있으나 고객 요구사항에 대한 잘못된 이해와 잦은 변경, 정량적인 데이터에 근거하지 않은 프로젝트관리, 프로세스 및 품질에 대한 평가능력 부재등으로 프로젝트의 개발원가와 납기준수에 대한 위험과 저품질의 상품을 생산하는 결과가 발생하였다. 이러한 비효과적인 S/W 프로세스를 개선하기 위해 [그림 4]와 같이 통제 표준 프로세스에 따라 프로젝트를 수행하였다.



[그림 4] S/W 개발 단계 및 통제 프로세스 정의

3.2 S사 영업정보 시스템 분석

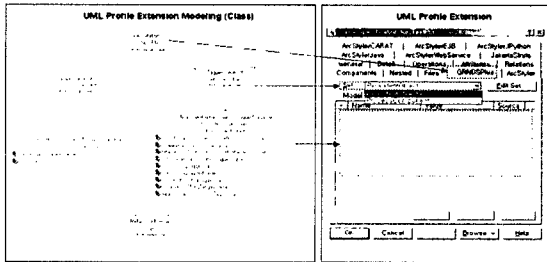
지금까지 진행되어 오던 구현 종속적 개별 플랫폼 지향 개발의 경우 모델을 통해 자동 생성된 소스의 구현 이외에 데이터베이스, 어플리케이션 서버, 환경 파일, 각종 기반 프레임워크, 개발 IDE 환경 등에 대한 모든 기반 환경 요소들에 대한 비용이 소요되어야 했으며, 이는 곧, 자동 생성된 소스와 기반 프레임워크 간의 불일치, 수작업량 증가, 프로젝트 기간 연장 및 S/W 품질 저하 등의 여러 가지 문제점들을 유발하는 원인이 되었다. 따라서 이와 같은 문제점들에 대한 해결 방안으로 S사는

CBD 기반 프레임워크 적용 기술 아키텍처를 정의하고 다음과 같은 작업들을 수행하였다.

- 기반 컴포넌트 프레임워크의 모델링 스타일 적용을 위한 MDA 기반의 UML Profile 확장 모델링 수행
- 생성해야 할 Artifacts에 대한 메타 모델링 수행 및 확장된 UML Profile 메타 모델과의 변환 구조 설계
- 표준화된 모델 변환 구조를 자동화 도구에 플러그인 하여 자동화 실현
- 표준 모델 변환 구조를 근간으로 하는 개발 프로세스를 통해서 컴포넌트의 요구 품질 및 생산성 증대

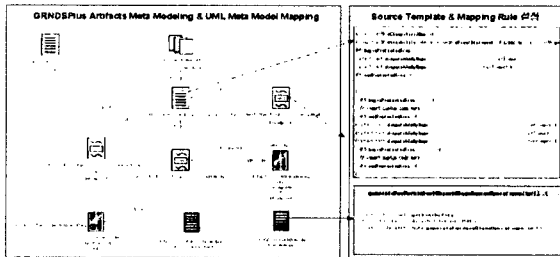
3.3 UML Profile 확장 및 메타 모델링

전사적으로 사용하게 될 컴포넌트 프레임워크 및 컴포넌트 모델링 패턴의 효율적인 적용 및 표준화를 위해서 UML Profile의 확장 모델링 및 메타 모델링 작업을 통해서 모델 변환 구조를 설계하고 구현하였다. [그림 5]와 같이 기반 프레임워크 요소들을 앞으로 개발될 각 어플리케이션 컴포넌트들에 자동 반영 시키기 위해 스테레오 타입 및 Tagged Value를 이용하여 클래스, 속성, 관계, 기능 등의 메타 모델 요소들에 대해서 UML Profile을 확장하여 모델링 하였다.



[그림 5] UML Profile Extension Modeling

그리고 UML Profile 확장을 통해서 UML 모델로 확장 반영된 기존 컴포넌트 프레임워크 요소들을 실제 구현 소스 및 각 종 산출물로 반영하기 위한 메타 모델링 과정을 [그림 6]과 같이 수행하였다.



[그림 6] Artifact Meta Modeling

3.4 평가

평가 방법은 Six Sigma의 DFSS 방법론을 적용하였을 때 고객 니즈 구현 및 표준 산출물에 대한 완성도를 현재의 영업시스템과 비교 검증하였으며 모델 변환 구조의 구현을 통해서 정의한 컴포넌트 프레임워크 및 컴포넌트 설계 패턴을 MDA 개발 지원 도구를 이용하여 적용 시켰을 경우와 순수 UML 모델링을 통해 컴포넌트 프레임워크

및 컴포넌트 설계 패턴을 적용 시켰을 경우를 소요 기간 및 자동화 비율의 비교를 통해서 검증하였다. 그리고 효과적인 비교 평가를 위해서 비전문가들을 대상으로 그 결과를 측정하였다.

특정 고객 니즈 구현과 산출물에 대한 결과는 [표 1]과 같고 컴포넌트 모델링에서 소스 생성 및 테스트 단계까지의 소스 및 환경 파일들에 대한 자동 생성률 및 오류 확률은 다음 [표 2]와 같다.

[표 1] 고객 니즈 구현도 및 산출물 완성도

	Six Sigma	현재
고객 니즈 구현	98%	80%
프로젝트 표준 산출물	95%이상	70%

고객 니즈 구현도는 실제 고객과의 설문으로 산출되었고 표준 산출물 완성도는 S사내 품질 심사팀이 산출하였다. 고객 니즈의 경우 20%, 산출물의 경우 25%가 향상되었으며 이는 프로젝트 품질의 향상된 결과로 간주하였다.

[표 2] 컴포넌트 자동 생성률 및 오류 확률

	MDA 기반	표준
Entity 관련 소스 생성률	100%	70%
Session 관련 소스 생성률	80%이상	50%
Build,Deploy,Tester 생성률	100%	70%
오류 확률(%)	0%	10%이상

자동 생성률에 대한 측정 기준은 자동으로 생성된 소스의 라인 수를 통해서 산출하였으며, 라인 수의 측정은 소스 파일의 라인 수 측정 프로그램을 이용하였다. 해당 소요 기간을 중심으로 생산성을 비교한 결과 표준 개발 공정을 기준으로 MDA 기반 개발 과정을 비교하였을 경우 30%의 개발 기간이 단축되었음을 검증하였으며, 이는 동일 비율의 생산성 향상 결과로 간주하였다.

4. 결론 및 향후 과제

본 연구를 통해 DFSS 방법론과 모델 변환 구조 구현을 통한 프로젝트 품질 및 생산성의 향상에 대해서 실제 프로젝트를 통하여 검증하였다. 향후 계속해서 특정 도메인에 대한 비즈니스 모델을 중심으로 실제 프로젝트 진행을 통해서 상호 운용성 향상에 대한 효율성을 측정하고, 연구할 계획이다.

참고 문헌

- [1] Service Design for Six Sigma : El-Haik, Wiley
- [2] Design for Six SIGMA in Technology and Product Development : David Antis, Jeffrey Lee Slutsky, Clyde M. Creveling , Prentice Hall PTR
- [3] Annerke Kleppe, Jos Warmer, Wim Bast : MDA Explained-The Model Driven Architecture : Practice and Promise, Addison-Wesley, 2003
- [4] David S.Frankel, Model Driven Architecture : Applying MDA to Enterprise Computing, WILEY
- [5] Recharh Hubert, Convergent Architecture : Building Model-Driven J2EE Systems with UML, WILEY, 2002