論文
5-2-15

닷넷 기반의 소프트웨어 RAD 프로세스
.NET Based Software Rapid Application Development(RAD) Process

노재우(Jae-Woo Noh)[1] 조현훈(Hyun-Hoon Cho)[2] 류성열(Sung-Yul Rhew)[3]

요 약

소프트웨어의 개발은 여러 번의 반복(Iteration)을 거치며 각각의 반복은 요구사항 분석, 분석 및 설계, 구현, 그리고 테스트 및 평가 과정을 포함하고 있어 자체로서도 하나의 개발주기를 이룬다. 이러한 반복적인 개발 방법에서는 반복마다 실행 가능한 릴리즈가 산출되고 이는 반복이 거듭될수록 향상되어 결국 최종 시스템으로 발전된다. 전통적인 프로세스와 비교했을 때 빠르고 반복적인 개발 방법이 갖는 장점은 초기에 위험요소를 줄일 수 있고 변경에 대한 관리가 용이하다. 그리고 보다 높은 수준의 재사용이 가능하며, 프로세스가 진행됨에 따라 프로젝트 팀원의 기술을 향상시킬 수 있다. 이러한 결과로 전반적인 고품질을 얻을 수 있다.

본 논문에서는 고객 중심의 요구를 빠르게 적용할 수 있고 짧은 기간에 개발 산출물을 제공할 수 있도록 프로세스 개선의 초점을 둔 빠른 개발 프로세스와 사례연구를 목적으로 하고 있으며, 이를 위해서 제안 프로세스는 .NET 기반에서 마르미III, MSF/CD, XP, Agile, PSP, TSP의 공통적이고 핵심적인 활동을 중심으로 구성하였다.

Abstract

Software development undergoes a number of iterations and each iteration forms its own cycle going through requirement analysis, scheme and design, implementation and finally test and evaluation. In this iterated development process, executable releases are produced, improved and eventually developed to a complete system, going through this particular development cycle. Compared to the conventional process, the advantage of rapid iterative development process lies in reducing risk factors in early stage and responding to changes very flexibly. In addition, highly reusable, the process can improve capabilities of the development team while the project is being carried out. As a result, overall balance in quality is secured.

The objective of this paper is the research of rapid development process and its case studies showing how to adapt the rapidly changing customer requirements and to transform those requirements into the project timely and adequately. The proposed process is focused on the common and core activities of .NET-based MarMIIII, MSF/CD, XP, Agile, PSP and TSP

1) 정회원 : 정평모비컴(주) 대표컨설턴트
2) 정회원 : 신흥대학 컴퓨터 정보계열 겸임교수
3) 정회원 : 숭실대학교 대학원 컴퓨터학과 주임교수

논문접수 : 2004. 2. 9.
심사완료 : 2004. 2. 17.

*** 논문은 2003년 12월부터 2004년 3월까지 정평모비컴㈜의 닷넷기반의 소프트웨어 RAD방법론 개발 프로젝트의 연구비 지원에 의한 연구 산출물임.**

1. 서 론

현 시장 환경은 공급자 중심에서 수요자 즉, 고객 중심으로 변화고 있으며, 고객의 요구는 무척이나 다양해지면서 빠르게 변하고 있다. 또한 특정 기술과 제품에만 의존한 경쟁력의 확보는 점점 어려워지고 있으며, 경쟁력 유지가 가능한 시간 또한 점점 짧아지고 있다. 이를 위해 기업은 내부적으로 고객의 요구를 정확하고 빠르게 수집하고, 이에 따른 변화요구를 제품에 빠르게 반영시킬 수 있는 프로세스를 갖추어야 한다. 기존 시스템의 구축 방향은 인지된 비즈니스 프로세스의 자동화에 초점이 맞추어져 있었고, 반복적인 업무처리 시간 단축이라는 가시적인 성과를 통해 어느 정도 기업 경쟁력 제고에 일조를 해 왔다고 할 수 있다. 그러나 이러한 제한적인 접근 방법으로는 향후 비즈니스 변화에 대한 빠른 수용력 측면에서 경쟁력을 갖추기는 어려울 것이다[9][11][12][13][14][15]. 이러한 시도는 최근 유행을 하고 있는 CBD(Component Based Development)에서 방향성을 찾을 수 있다. CBD란 업무 생활에서 사람이 가지고 있는 역할과 그 역할에 따른 일들을 컴포넌트의 클래스 및 서비스란 형태로 구현하여 제공한다. 이는 인간 생활에서의 개인 서비스의 유연한 변화 모델을 컴퓨터 시스템에서 확장 가능하고 또 재사용이 가능한 컴포넌트 서비스 모델에 투영함으로써, 인간 생활이 가지는 변화에 대한 융통성을 시스템에서도 가지려고 하는 시도로 볼 수 있다[5][6][8][10].

이를 위해 21세기에 들어서면서 Microsoft는 급변하는 비즈니스의 경쟁상황에서 미래의 컴퓨터 시스템이 갖추어야 할 비전으로서 .NET을 발표하였고, 이를 구현 할 수 있도록 수 많은 제품군을 체계화하고 있다. Microsoft는 .NET을 웹서비스 기반의 시스템 구현을 위한 기반 기술로서 정의하고 있으며, 고객이 원하는 서비스를 빠르게 제공한다는 강점을 제공하고 있다. 이러한 서비스는 .NET이라는 특정 기술과 제품에 의존적이지만 생산성 향상 측면에서 효율성을 가져다 줄 수 있다[1][[7].

소프트웨어 개발에 대한 인식체계가 변화하고 있는 현실에서 기업에 프로세스에 압력을 가하고 팀원들의 초과 근무를 강요하는 전통적인 접근 방식으로는 업무를 제대로 수행해 낼 수 없다는 사실을 깨닫게 됨에 따라 개발 그룹의 팀 능력과 개인 개발자의 역량이 보다 중요시 되고 있다. 즉, 개인 개발자의 부담을 가중시키기 보다 어떻게 하면 프로세스를 변경하여 팀을 보다 생산적으로 운영하고 품질을 향상 시킬 수 있는 프로세스가 절실히 필요하다[2][3][9][13].

그러므로 .NET이라는 특정 기술을 기반으로 고객의 요구사항 변화를 중단 없이 반영할 수 있어야 하고 단기간에 반복적인 진화를 통해 국내 환경에 적합한 프로세스의 확립 또한 개발에 필수적이라 할 수 있다.

2. 관련 연구

2.1 마르미Ⅲ

마르미Ⅲ는 국내 표준 컴포넌트 기반의 개발 방법론으로 개발 배경은 체계적인 시스템 개발 및 관리의 필요성과 프로젝트 구성원의 효율적인 역할 할당과 동일한 의사소통 수단을 제공함에 있으며, 최신의 정보기술을 반영하고자 개발이 되었다. 이러한 필요성에 의해 국내 기술진에 의한 축적된 기술과 경험적 지식의 체계화를 통해 구체적인 개발 및관리 절차를 제시하고 있으며, 실용적인 개발 기법과 지침을 제공하고 있다[4].

2.2 MSF/CD(Microsoft Solution Framework/Component Design)

MSF(Microsoft Solution Framework)는 Microsoft내 제품 개발 그룹의 25년 이상의 소프트웨어 개발 및 프로젝트 노하우와 내부 전산 그룹에서 진행했던 다양한 프로젝트 경험, 컨설팅 그룹의 경험 등을 체계화시켜 2-3명이 참여하는 소규모 프로젝트에서 윈도우2000개발 프로젝트와 같은 대규모 프로젝트에 모두 적용할 수 있도록 유연성과 적용 능력을 가지고 있는 프로젝트 관리 프레임워크라 할 수 있다[16][17][18]. MSF는 다음과 같이 4개의 구성요소로 이루어져 있다.

- MSF/AD(Application Development): 응용 프로그램 시스템 개발 방법론

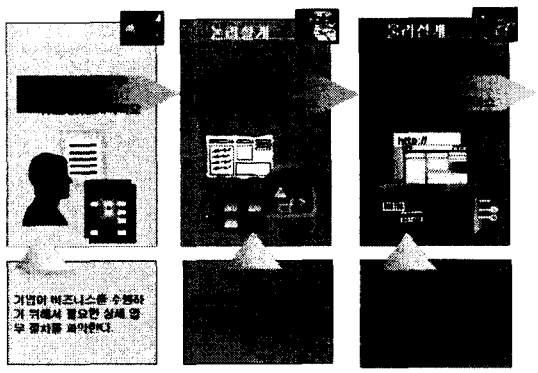
- MSF/CD(Component Design): 컴포넌트 설계 방법론

- MSF/ID(Infrastructure Deployment): 시스템 기반 기술 구축 방법론

- MSF/EA(Enterprise Architecture): 기업 전략 기반 시스템 아키텍쳐 계획 수립 방법론

4가지 요소 중 컴포넌트 설계 방법론인 MSF/CD가 본 논문의 참조 방법이라 할 수 있다.

MSF/CD는 Microsoft의 오랜 소프트웨어 개발 경험 및 현장 개발 및 구현 경험에 바탕을 두고 만들어진 다양한 방법중의 하나로 컴포넌트 설계에 관한 방법론이라 할 수 있다. MSF/CD는 많은 부분이 UML(Unified Modeling Language)에 기초하여 만들어졌으며, 여기에 Microsoft의 오랜 컴포넌트 기반 제품 개발 경험으로부터의 실전적인 가치가 반영된 것이다. MSF/CD의 개념 및 논리설계는 기존의 방법론들과 큰 차이는 없지만, 기존 객체 지향 모델링 방법론들의 취약 부분이 물리설계 부분에서는 구체성과 실효성 면에서 상당한 강점을 가지고 있다. 즉, MSF/CD는 객체지향 모델링을 기반으로 한 컴포넌트 기반의 개발(CBD)방법론이라 할 수 있다.



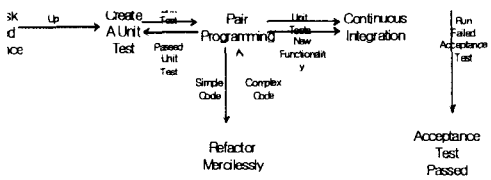
[그림 1] MSF/CD의 3가지 활동
[Fig. 1] Three Activity of MSF/CD

2.3 Agile 소프트웨어 개발

Agile 소프트웨어 개발은 직접 실행해보고 다른 사람이 이를 실행하도록 도와줌으로써 소프트웨어 개발에 대한 더 나은 방법을 밝히고자 하는 데 목적이 있다. Agile 소프트웨어 개발이 갖는 특징은 프로세스와 도구 보다는 개인과 상호작용하는데 있으며, 이해를 돕는 문서화보다는 제대로 움직이는 소프트웨어를 만들어야 하며, 고객과의 협력에 기반을 두고 고객의 요구 변화에 대응하는데 개발의 가치와 원칙을 정의하고 있다.[2]

2.4 XP(eXtreme Programming)

XP는 Agile 소프트웨어 개발의 원칙을 따르는 방법 중에 하나로, 위험요소들을 짧은 기간의 단위 기능 개발과 여러 차례의 테스트 작업, 그리고 고객을 개발의 중심에 두고 문제를 해결하기 위해 제안된 방법론이다. 고객, 프로그래머, 관리자의 역할을 정의하고 pair programming 기법을 통해 고객이 원하는 소프트웨어를 단기간에 신속히 개발하는 데 목적이 있다[3][15]. 그림 2는 XP의 개발 프로세스를 나타낸다.

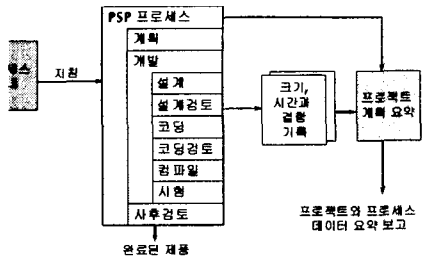


[그림 2] XP 개발 프로세스
[Fig. 2] XP Development Process

2.5 PSP(Personal Software Process)

PSP는 엔지니어 개인의 효율성을 정의하고, 사용하고, 향상시키는데 필요한 원칙과 방법을 정의하여 소규모 소프트웨어 업무 기반에서 개개인이 활용하도록 만든 프로세스이다. 이를 통해 대규모 소프트웨어 시스템 개발을 지원하도록 확장할 수 있다. PSP를 통해서 프로젝트 추정 및 계획 능력이 향상되며, 과도한 목표설정을 방지할 수 있고, 이를 기반으로 품질 목표 달성 및 지속적으로 프로세스 개선에 엔지니어

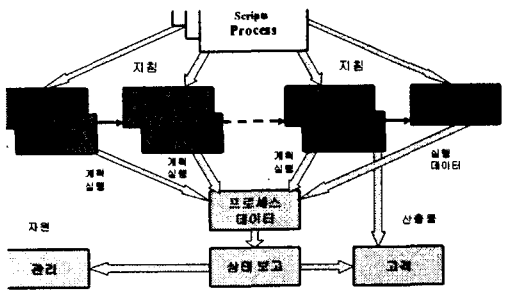
가 적극적으로 참여할 수 있다[9][11][13]. 그림3은 PSP에 대한 흐름이다.



[그림 3] PSP 흐름
[Fig. 3] PSP Flow

2.6 TSP(Team Software Process)

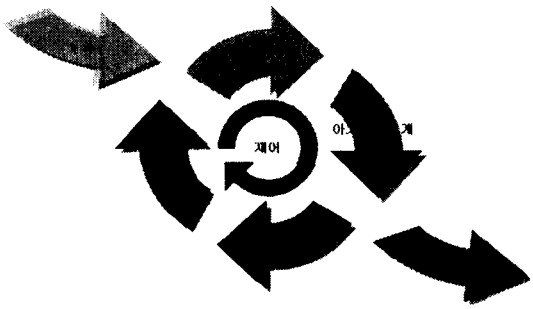
TSP는 효율적인 팀의 구성과 운영을 위해 만들어진 프로세스로서, 1996년 Watts S. Humphrey 에 의해 개발되었다. 이 프로세스는 효율적인 팀 구성에 대한 지침을 제공 하고 있으며, 관리자가 팀을 효과적으로 이끌 수 있는 방법을 제시하고 있다. TSP를 위해서는 개인적인 훈련과 팀의 목표, 개인적인 역할, 행동에 대한 공통된 계획, 그리고 성공적인 프로젝트를 수행하기 위한 관리 방법을 고려해야 한다. TSP는 3명에서 15명의 엔지니어로 구성된 소규모 팀들의 성공적인 프로젝트 진행을 도와주기 위해 정의된 프로세스로서 다음 그림4와 같은 흐름으로 구성된다[9][11][13].



[그림 4] TSP 흐름
[Fig. 4] TSP Flow

3. 닷넷 기반의 소프트웨어 RAD 프로세스

제시한 RAD 프로세스는 마르미Ⅲ라는 국내 표준 컴포넌트 기반의 개발 방법론을 바탕으로 단기간에 고객의 요구를 빠르게 반영할 수 있는 Agile개발 원칙과 극단적인 프로그래밍 기법인 XP기법, 그리고 엔지니어들 개인의 효율성을 정의하고 사용하고, 향상시키는데 필요한 원칙과 방법을 정의한 PSP 및 PSP로 훈련된 엔지니어로 구성된 구성원간의 효율적인 운영을 위해 만들어진 TSP와 같은 프로세스적인 측면과 MSF/CD방법에서 컴포넌트 설계 활동을 복합적으로 접목하여 짧은 기간에 반복적으로 개발할 수 있는 프로세스 프레임워크이라고 정의할 수 있다. RAD 프로세스에서 정의한 단계별 세부 활동과 작업을 개념적으로 그림5에서 보여주고 있다.

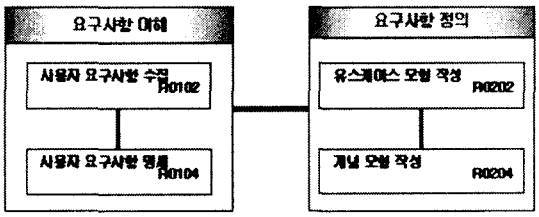


[그림 5] 개발 프로세스 프레임워크
[Fig. 5] Development Process Framework

제안 프로세스는 크게 요구사항분석단계, 아키텍쳐 설계단계, 점진적 개발 단계 및 인도 단계의 4단계로 구성되며, 요구사항분석 단계에서 고객의 요구를 빠르게 반영할 수 있도록 개발자가 준비한 스토리 카드를 통해 고객의 요구를 반영하도록 하고, 아키텍쳐 설계 단계에서 도메인의 시스템 아키텍쳐와 기능 아키텍쳐를 정의한후, 컴포넌트를 획득한다. 점진적 개발 단계에서는 XP 프로그래밍 기법과 MSF/CD방법의 3가지 주요 컴포넌트 설계 활동을 통하여 짧은 기간에 개발에 용이하도록 재사용 가능한 컴포넌트 중심으로 점진적 개발과 테스트를 병행함으로써 품질 측정을 진행한다. 마지막 인도 단계에서는 사용자 교육 및 시스템 설치, 그리고 인수 테스트를 통해 시스템을 인도하는 활동으로 구성된다. 또한 각 단계별 지침서와 산출물에 대한 템플릿을 정의하고 세부 활동을 정의한다.

3.1 요구사항분석 단계

요구사항분석 단계의 목적은 실제 시스템의 사용자로부터 요구사항을 수집하고, 수집된 요구사항을 기반으로 구현할 기능을 정의한다. 이를 통해 개발 시스템의 범위를 명확히 하고 사용자 요구사항을 유스케이스 모형과 개념 모형으로 정의한다. 다음은 요구사항분석 단계의 세부 활동을 정의한 것이다. 이 단계에서의 산출물은 비전기술서, 사용자요구수집서, 사용자정의서, 면담서, 설문서, 수집자료정의서, 요구사항기능정의서, 유스케이스모형기술서, 개념모형기술서, 용어집, 스토리 카드가 해당한다.



[그림 6] 요구사항분석 단계의 활동
[Fig. 6] Activity of Requirement Analysis Phase

3.1.1 요구사항이해 활동

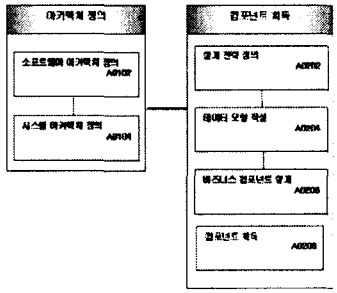
본 활동은 제안 시스템에 대한 시스템의 비전을 개발하고 실제 사용자와의 면담, 설문 등을 통하여 상세한 요구사항을 수집한다. 필요한 경우, 사용자의 요구사항을 기반으로 개선 업무 모형을 작성함으로써 시스템의 개발 배경을 명확히 이해한다.

3.1.2 요구사항정의 활동

본 활동은 사용자 관점의 텍스트 기반 요구사항을 분석가 관점으로 모형화 하고 UI 프로토타입 구축과 시연을 통해 사용자 요구사항을 명확히 한다.

3.2 아키텍처설계 단계

아키텍처설계 단계의 주목적은 사용자 요구사항을 기반으로 소프트웨어 및 비즈니스 아키텍처를 설계하고 이러한 아키텍처 기반에서 재사용 가능한 컴포넌트를 조사하여 획득하며, 점진적 개발단계에서 개발되는 시스템에 대한 데이터 이관 및 전환 계획을 통해 컴포넌트 설계가 이루어진다. 이 단계에서의 산출물은 소프트웨어아키텍처정의서, 매커니즘기술서, 데이터모형설계서, 컴포넌트명세서, 시스템아키텍처정의서가 해당된다.



[그림 7] 아키텍처설계 단계의 활동
[Fig. 7] Activity of Architecture Design Phase

3.2.1 아키텍처정의 활동

본 활동은 사용자 요구사항에 따라 개발시스템의 구현관점에서 소프트웨어 아키텍처와 시스템 아키텍처를 정의한 후 정의한 아키텍처를 고려한 비즈니스 컴포넌트 아키텍처와 시스템 아키텍처를 작성한다.

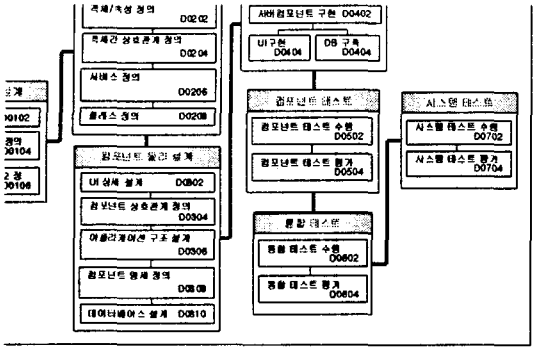
3.2.2 컴포넌트 획득 활동

본 활동은 정의된 소프트웨어 아키텍처와 시스템 아키텍처를 고려하여 설계전략을 정의하고 데이터 모형과 비즈니스 컴포넌트 아키텍처를 작성하며, 비즈니스 중심의 컴포넌트 설계에 대한 설계 전략을 정의하고 데이터모형을 작성, 비즈니스 컴포넌트를 설계한다.

3.3 점진적개발 단계

점진적개발 단계에서의 주목적은 프로젝트 수행계획에 따라 컴포넌트, 데이터베이스, GUI 등을 구현하여 점진적으로 시스템을 개발한다. 그리고 프로젝트에서 유스케이스를 구현하고 나면 사용자의 검토를 받아 요구사항을 만족하는지 확인하며, 계획된 모든 프로젝트를 수행하고 나면 통합된 시스템의 기능성 및 성능이 사용자 요구사항을 만족하는지 확인하기 위한 시스템 테스트를 수행하고 본 단계의 수행결과를 평가한다. 이 단계에서 주요 산출물은 컴포넌트 개념설계서, 컴포넌트 논리설계서, 컴포넌트 물리설계서, 정제된 컴포넌트 명세서, 데이터베이스

설계서, 컴포넌트 코드가 해당된다.



[그림 8] 점진적개발 단계의 활동
[Fig. 8] Activity of Incremental Development Phase

3.3.1 컴포넌트 개념설계 활동

이 활동에서는 설계하고자 하는 업무에서 현재 사용자가 하는 일이 무엇이고, 비즈니스 요구사항이 무엇인지에 대해 이해하여 업무를 정의하는데 그 목적이 있다. 그러므로 실제 업무를 담당하는 현업 담당자가 현 업무의 내용을 기술하는 것이 이상적이나, 그렇지 못할 경우 설계 담당자가 현업 담당자의 입자에서 설계를 진행한다. 이 단계의 대표적인 산출물은 시나리오이다.

3.3.2 컴포넌트 논리설계 활동

이 활동에서는 개념설계 단계에서 나열된 업무들을 대상으로 누군가 수정해야 할 역할 및 그에 따른 서비스들을 정의하는 것이다. 이 설계 단계는 시스템 설계자의 입장에서 이루어져야 한다. 개발할 시스템의 객체 구조를 정의하고, 정의된 객체간의 관계를 기술하며, 객체간의 상호작용 및 인터페이스를 정의하는 데 그 목적이 있다.

3.3.3 컴포넌트 물리설계 활동

이 활동은 논리설계에서 작성된 논리적 클래스/서비스 모델을 대상으로 하여 실제 운영 환경에서 나타날 수 있는 각종 문제점들을 사전에 찾아내고, 이에 대한 대책을 반영함으로써 운영을 위한 최적의 물리적 구성요소 모델을 완성해 나가는 과정이다. 이 단계는 설계자와 더불어 실제 개발자가 같이 참여하여 논리 설계 내용을 이해하고 개발자가 인지하고 있는 현 개발 환경의 제약사항(기술구조, 네트워크, 시스템 자원 등)하에서 발생할 수 있는 문제점에 대한 대책을 사전에 반영할 수 있도록 한다.

3.3.4 컴포넌트 및 시스템 구현 활동

이 활동은 플랫폼에 맞게 설계된 컴포넌트와 UI를 프로그래밍 언어로 구현하고 데이터베이스를 구축한다. 기존 업무 정보 또는 기존 시스템의 정보를 새로 설계된 데이터베이스 또는 기타 지속성 저장소로 옮기기 위한 준비 작업을 수행한다.

3.3.5 컴포넌트 테스트 활동

이 활동은 구현된 컴포넌트를 대상으로 테스트 케이스를 설계하여 테스트를 수행한다. 필요시 테스트 드라이버, 스텁 등을 추가 개발하여 테스트를 수행할 수 있다.

3.3.6 통합 테스트 활동

이 활동에서는 테스트계획서를 기반으로 통합 테스트 환경을 구축하고 테스트를 수행한다. 통합 테스트는 해당 미니프로젝트 활동에서 개발한 컴포넌트 간의 통합 및 기 개발된 서브 시스템과의 통합 테스트로 이루어진다.

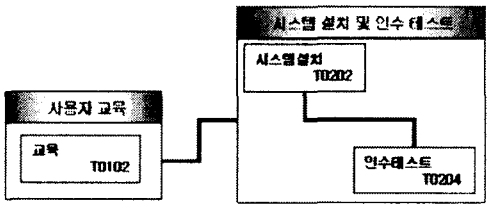
3.3.7 시스템 테스트 활동

시스템 테스트는 모든 미니프로젝트가 완료된 후 개발된 각 서브 시스템들을 통합한 전체 시스템에 대해 사용자 요구사항을 만족하는 지를 테스트한다.

3.4 인도단계

인도단계의 주목적은 개발자 환경에서 개발된 결과물을 컴포넌트인 경우에는 컴포넌트 리포지터리에, 컴포넌트 기반 시스템인 경우에는 실제 시스템이 운영될 사용자 환경에 설치한다. 기존에 운영되고 있는 리포지터리나 시스템이 있을 경우 신규 시스템으로 전환하여 원활한 운영이 가능하도록 한다. 개발된 컴포넌트 또는 시스템에 대하여 최종적으로 사용자 요구사항과의 일치 여부에 대하여 승인을 얻고 프로젝트의 모든 전달물을 사용자에게 전달하고 인계한다. 본 단

계는 사용자의 업무에 영향을 주게 됨으로 사용자의 적극적인 참여가 필수적이며, 사용자는 인수와 함께 시스템을 운영할 수 있도록 준비하여야 한다. 본 단계의 주요 산출물에는 사용자교육보고서, 시스템설치보고서, 인수테스트결과보고서가 해당된다.



[그림 9] 인도단계의 활동
[Fig. 9] Activity of Transition Phase

3.4.1 사용자교육 활동

본 활동은 새로운 시스템을 사용자가 운영할 수 있도록 준비를 하고 교육을 실시하는 활동이다. 이는 최종적으로 인수 테스트가 완료된 후에 새로운 시스템으로 전환하여 운영하기 위한 내용들을 교육하게 되는데 일반적으로 이러한 교육을 통하여 전 조직에 보급 및 확산이 이루어지게 된다.

3.4.2 시스템설치 및 인수테스트 활동

시스템 설치에서는 하드웨어, 시스템 소프트웨어, 응용 프로그램 설치뿐만 아니라, 기존 데이터의 전환 등을 실시한다. 하드웨어 설치와 시스템 소프트웨어 설치 작업은 조달 계획에 의거하여 전문 요원이나 공급업체에 의해서 이루어지며, 하드웨어 설치, 시스템 소프트웨어 설치가 완료되면 신규 응용 프로그램이 설치된다. 설치가 완료되고 앞 단계의 테스트를 통하여 요구한 내용이 정확하고 빠짐 없이 개발되었다는 것이 확인되면 사용자가 주체가 되어 테스트를 수행하게 된다. 시스템 설치가 완료되면 실제 운영환경 하에서 신규 시스템이 제대로 운영되는지를 일정 기간 모니터링 하여 문제가 없다는 것을 검증하여야 한다. 설치 후 정상적으로 운영되는 시스템들도 일정 기간 운영 상에서 문제가 발생되는 경우가 있으므로 시스템 운영을 전제로 하여 일정 기간 관찰을 통하여 이러한 문제들을 찾아 내고 해결하여야 한다.

4. 사례 및 평가

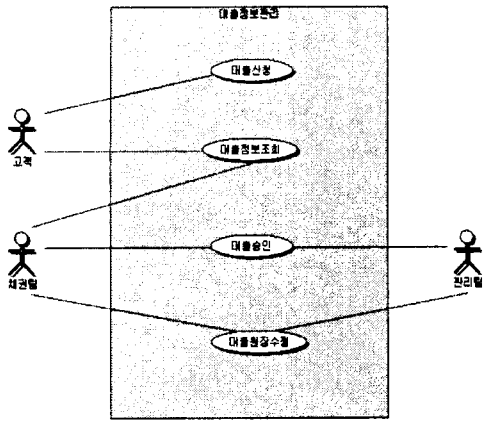
제시된 프로세스의 타당성 검증을 위해 대부업 도메인에서 대출정보관리를 적용한다. 이를 위해 대출정보관리에 필요한 세부 기능을 정의하고 RAD 프로세스의 각 단계별 지침서와 산출물을 정의한다.

4.1 요구사항분석 단계

대부업이란 돈을 빌려주는 사금융업(私金融業)으로, 은행, 협동조합, 보험회사, 금융회사, 신용금고 등 공적인 금융기관이 아닌 금전의 대부, 또는 대부차용의 중개를 업무로 하는 영업으로 현재 대부업의 상황은 2002년 8월 정부의 대금업법 시행으로 인해 대부업자의 제도권 진입이 이루어졌으며, 전국 4만여 대부사업자 중 약 9천여개사가 대부업 등록을 완료한 상태이다. 또한 금융시장 개방과 전자금융시대의 도래 등 급변하는 금융환경의 변화가 대부업 시장을 활성화 시키고 있는 실정이다. 우선 업무 표준화 및 전산화를 통해 사업자 특성에 맞는 컴포넌트를 제공할 수 있어야 하고, 이를 통해 대내외적으로 대부업계에 대한 경쟁우위를 확보를 차지할 수 있다. 대부업 전산화에 필요한 세부 기능은 다음과 같다.

- 고객정보관리: 회원관리 및 맞춤 정보 제공
- 신용정보관리: 개인 및 법인의 신용 평가 정보 제공
- 대출정보관리: 대출신청, 대출정보조회, 대출승인, 대출원장관리 제공
- 경영정보관리: 재무제표 분석 자료 제공
- 회계정보관리: 회계전표 및 대차대조표, 손익계산서, 연말결산등의 다양한 정보 제공
- 채권관리: 채권추심에 관련된 정보를 실시간 조회할 수 있는 여러 가지 기능 제공

이러한 세부 기능중에 대부업의 핵심업무로서 대출정보관리를 적용하여, 요구사항분석 단계의 세부 활동을 통해 다음과 같은 산출물의 일부를 정의한다.

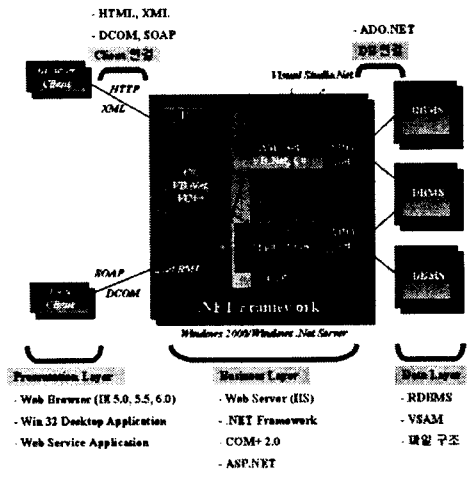


[그림 10] 대출정보관리 유스케이스 다이어그램
[Fig. 10] Loan Information Management Use Case Diagram

4.2 아키텍처설계 단계

아키텍처단계의 주 목적은 시스템에 필요한 컴포넌트 획득에 필요한 아키텍처를 수집하는데 있으며, 획득할 컴포넌트를 개발 시스템 아키텍처상에서 표현한다.

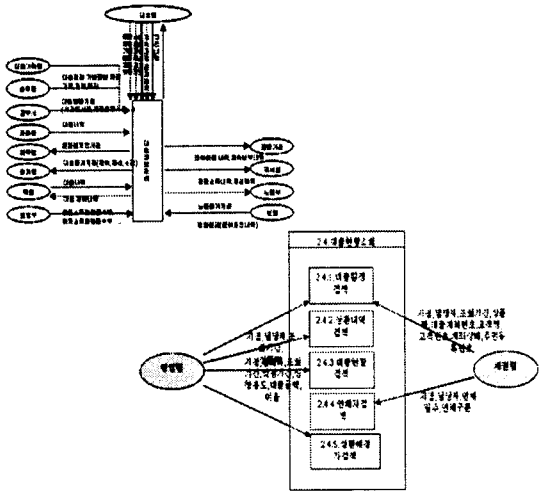
본 단계의 세부 활동에서는 표준 아키텍처 어플리케이션을 개발하고자 할 때 기존의 MS DNA(Distributed interNet Architecture)를 기반으로 했던 3-Tier의 개념을 그대로 적용하고 있다. 하지만 기존의 3-Tier에서 각각 계층의 컴포넌트나 계층을 연결하는 많은 요소들이 닷넷 환경에 맞도록 변경하였고, 새로운 개념을 추가하였다.



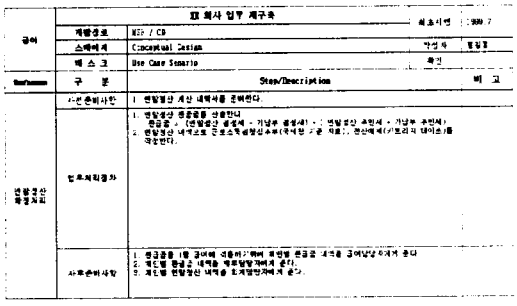
[그림 11] 닷넷 기반의 어플리케이션 아키텍처
[Fig. 11] .NET Based Application Architecture

4.3 점진적개발 단계

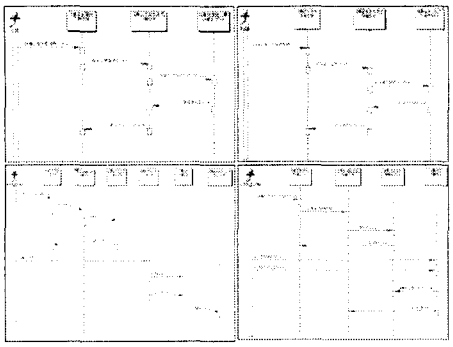
점진적개발단계에서는 MSF/CD의 주요활동인 개념 설계, 논리설계, 물리설계를 통해 컴포넌트를 정제하고 시스템에 적용한다. 컴포넌트 개발은 유스케이스 중심으로 만들어지며, 다음과 같은 산출물을 도출한다.



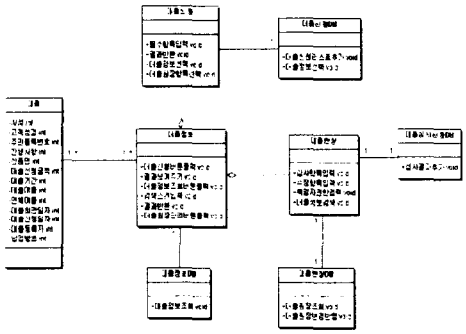
[그림 12] 비즈니스 문맥도와 웍플로우 프로세스 모델
[Fig. 12] Business Context Diagram and Workflow Process Model



[그림 13] 유스케이스 시나리오
[Fig. 13]Use Case Scenario



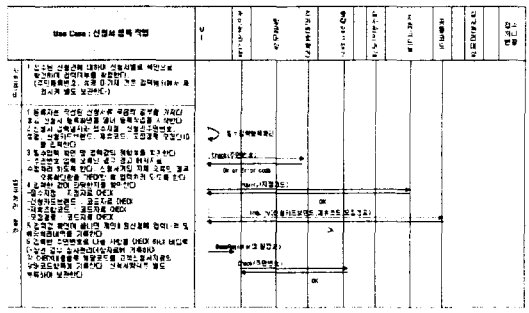
[그림 14] 작업순차도
[Fig. 14] Task Sequence Diagram



[그림 15] 대출정보관리 컴포넌트 클래스
[Fig. 15] Loan Information Management Component Class

고객등록/변경	신용정보	고객신용정보	
고객등록/변경	신용평가자	고객신용정보	신용등급
대금지불요청	대금지불요청여부	대금청구건수내역	대금지불요청여부
대금청구건수	정상청구여부	대금청구건수내역	정상청구여부
대금청구건수	대금청구요청내역	대금청구건수내역	
대금청구건수	대금지불요청	대금청구건수내역	
대금청구건수	수량	발주	발주수량
대금청구건수	단가	발주	발주단가
대금청구건수	일자	발주	발주일자
대금청구건수	발주 번호	발주	발주 번호
대금청구건수	발주내역	발주	
대금청구건수	생산자명	생산처관리	생산자명

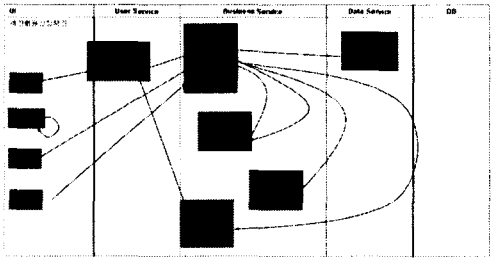
[그림 16] 객체 및 속성 정의
[Fig. 16] Object and Attribute Definition



[그림 17] 객체상호관계다이어그램
[Fig. 17] Object Interaction Diagram

CLASS	서비스명	Param.In	Param.Out	Use Case Name
거래은행	TransferReq	Name,Account,금액	Status	진행
거래처	inquiry()	요청항목들, 조건 명세	요청항목값들	지불방식결정 진행 결재
거래처	QueryAccount	BuyerName,CompanyID	Bank,Account	진행
거래처	QueryGrade	BuyerName	Grede	지불방식결정
거래처	QueryRisk	BuyerName	Credit,Bankrupt	결재
결재	Approve	BuyerName	Status	결재
결재	Inquiry()	요청항목들, 조건 명세	요청항목값들	
결재	PayTypeRegister	BuyerName,CashFlag(지불방식)	Status	지불방식결정
결재	Query	일	Confirm 대상들	결재
고객	CheckExists	주민번호	결과메시지	고객정보등록/변경

[그림 18] 서비스 일반화
[Fig. 18] Services Generalization



[그림 19] 컴포넌트 상호관계 다이어그램
[Fig. 19] Component Interaction Diagram

System	Application	Component	Class	Services	Parameter	
					Input	Output
DB	배송처리	배송관리 (CRUDcreate)	배송상세Processor chkCRProcessorDB	Inquiry()	String strContents String strCondition	RS
				Register()	String strContents String strContentValues	String strResultMessage
				Modify()	String strContents String strContentValues	String strResultMessage
				Cancel()	String strContents String strContentValues	String strResultMessage
			배송상세상태 chkCRProcessorDB	Inquiry()	String strContents String strCondition	RS
				Register()	String strContents String strContentValues	String strResultMessage
				Modify()	String strContents String strContentValues	String strResultMessage
				Cancel()	String strContents String strContentValues	String strResultMessage
			DB기관명과 chkCRProcessorDB	Inquiry()	String strContents String strCondition	RS
				Register()	String strContents String strContentValues	String strResultMessage

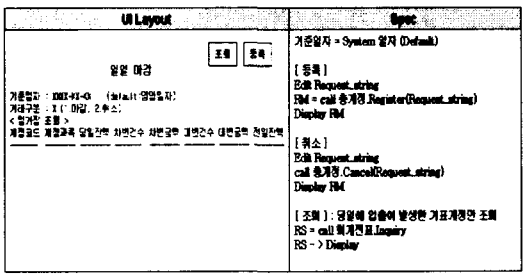
[그림 20] 어플리케이션 구조
[Fig. 20] Application Structure

SYSTEM		대출정보관리
Application		CheckIn
Component		대출
Class		clsReservation
service	public	private

Check	Public Check
	connect to DB using CommDB
	IF table.rows >0 then
	rerurn "Y"
	ELSE
	return "N"
	END IF
	End Public	

Inquiry	Public Inquire Connect to DB using CommDB select 대출목록모두 (+) from input prameter (대출번호 or 고객명) End Public	

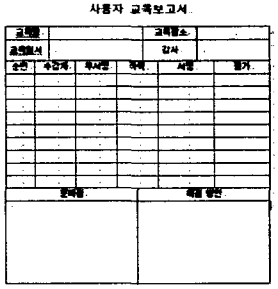
[그림 21] 컴포넌트 명세
[Fig. 21] Component Specification



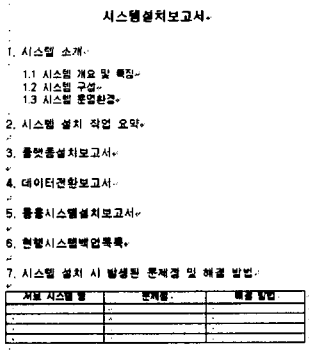
[그림 22] UI 명세
[Fig. 22] UI Specification

4.4 인도 단계

개발된 컴포넌트 또는 시스템에 대하여 최종적으로 사용자 요구사항과의 일치 여부에 대하여 승인을 얻고 프로젝트의 모든 전달물을 사용자에게 전달하고 인계한다. 인도단계에서 산출물은 다음과 같다.



[그림 23] 사용자 교육보고서
[Fig. 23] User Education Report



[그림 24] 시스템 설치보고서
[Fig. 24] System Setup Report

4.5 평가

본 논문에서 제안하는 RAD 프로세스를 통해 다음과 같은 단계별 세부 산출물 목록을 정의하였으며, 각 산출물에 대한 상태를 정의하였다. 이를 통해 프로젝트에 필요한 자원의 효율적 사용 및 지침을 제공할 수 있으며, 관련 프로젝트 진행시 참조 모델로 적용할 수 있다.

단계	활동	산출물	상태
R0000 요구 사항 분석	R0100 요구사항의 이해	사용자요구수집서	기존
		면담서	기존
		설문서	기존
		수집자료정의서	기존
		요구사항기능정의서	추가
		용어집	기존
	R0200 요구사항 정의	유스케이스모형기술서	기존
		개념모형기술서	기존
A0000 아키텍처 설계	A0100 아키텍처 정의	소프트웨어 아키텍처 정의서	기존
		시스템 아키텍처 정의서	기존
	A0200 컴포넌트 획득	메커니즘 기술서	기존
		데이터 모형 설계서	기존
		컴포넌트 명세서	기존
D0000 점진적 개발(.Net)	D0100 컴포넌트 개념설계	비즈니스문맥다이어그램	추가
		웍플로우프로세스모델	추가
		유스케이스시나리오	추가
		작업순차도	추가
	D0200 컴포넌트 논리설계	객체 추출 및 정의서	추가
		객체상호관계 다이어그램	추가
		클래스 다이어그램	추가
		Psuedo Code	추가
		UI Scratch	추가
	D0300 컴포넌트 물리설계	UI명세서	추가
		컴포넌트 상호관계 다이어그램	추가
		어플리케이션 구조도	추가
		컴포넌트 명세서	추가
		데이터베이스 정의서	추가
		데이터베이스 설계서	추가
	D0400 컴포넌트 및 시스템 구현	컴포넌트 코드	기존
		DLL정의서	기존
		UI코드	기존
		데이터 베이스 정의서(보완)	기존
		데이터 베이스 설계서(보완)	기존
	D0500 컴포넌트 테스트	컴포넌트테스트설계서	기존
		컴포넌트테스트수행보고서	기존
		컴포넌트테스트결과보고서	기존
	D0600 통합 테스트	통합테스트설계서	기존
		통합테스트수행보고서	기존
		통합테스트결과보고서	기존
	D0700 시스템 테스트	시스템테스트설계서	기존
		시스템테스트수행보고서	기존
		시스템테스트결과보고서	기존
T0000 인도	T0100 사용자 교육	교재	기존
		사용자 교육보고서	기존
	T0200 시스템 설치 및 인수 테스트	시스템 설치보고서	기존
		플랫폼설치 보고서	기존
		현행 시스템 백업 목록	기존
		응용 시스템 설치 보고서	기존
		인수 테스트 설계서	기존
		인수테스트 결과 보고서	기존

5. 결론 및 향후 연구과제

본 논문에서는 개인의 역량 및 조직의 구성원 간의 의사소통을 체계적으로 관리, 통제하여 소프트웨어 생산성 및 품질을 향상시키기 위한 구체적인 절차를 체계화하여 정립하고 중소규모의 조직에서 짧은 기간에 소프트웨어 개발 프로젝트를 수행하고자 할 때 필요한 프로세스를 개발하였다. 본 개발 프로세스는 닷넷 환경에서 컴포넌트 기반의 개발을 하고자 하는데 있어서 핵심적인 기반 기술과 국내 컴포넌트 표준인 마르미Ⅲ를 프레임워크로 정의하고 커스터마이징을 통해 소규모 프로젝트 수행에 적합하도록 하였다. 더불어 각 세부 활동에서는 XP의 요구분석 기법과 프로그래밍 기법, PSP/TSP에서의 개인과 팀의 성숙도 향상 기법, 그리고 점진적 개발에서 MSF/CD의 컴포넌트 설계 기법을 적용하였으며, 프로세스 전반적으로 Agile소프트웨어 개발 방법 원칙을 따르고 있다.

마지막으로 사례 적용은 현재 본 프로세스를 적용하여 개발하고 서비스 중인 대부업 도메인의 대출정보관리 부분에서 적용 산출물을 통해 프로젝트 수행 효율성과 신뢰성, 추적성 및 수행 능력을 검증하였다.

본 논문을 통해 향후 연구과제로 더 많은 사례 적용을 통한 품질요소 평가 지표를 정의하고자 하며, 이를 통해 기업 및 조직의 프로젝트 수행 능력을 향상시키는데 필수적인 요소를 체계화하는데 적용하고자 한다.

참 고 문 헌

[1] Microsoft , .NET 구축 방법론: CBD & XML Web Services, 정보문화사, 2002.

[2] Cockburn, Highsmith, Agile Software Development, Pearson Education, Inc., 2002.

[3] Ron Jeffries, Ann Anderson, Chet Hendrickson, Extreme Programming Installed, Addison-Wesley, 2001.

[4] ETRI, MarMI, Magic and Robust Methodology Integrated, Electronics and Telecommunications Research Institute, 2001.

[5] George T. Heineman and William T. Councill, Component-Based Software Engineering: Putting the Pieces Together, Addison Wesley, 2001.

[6] John Cheesman, John Daniels, UML Components, Addison Wesley, 2001.

[7] Kirtland, M., Designing Component-Based Applications, Microsoft Press, 1999.

[8] Peter Herzum, Oliver Sims, Business Component Factory: A Comprehensive Overview of Component-Based Development for the Enterprise, WILEY, 2000.

[9] Carnegie Mellon University, Capability Maturity Model Integration(CMMI) Version 1.1, Carnegie Mellon Software Engineering Institute, 2001.

[10] David Garlan, "Software Architecture and Object-Oriented Systems," In

Proceedings of the IPSJ Object-Oriented Symposium 2000 August 2000, Tokyo, Japan.

[11] Wattts S. Humphrey, Managing Software Process, Addison-Wesley, 1989.

[12] Watts S. Humphrey, A Discipline for Software Engineering, Addison-Wesley, 1995.

[13] Lawrence · G. Jones, Albert L. Soule, "Software Process Improvement and Product Line Practice: CMMI and the Framework for Software Product Line Practice," Carnegie Mellon University, 2002.

[14] Roger S. Pressman, *Software Engineering A Practitioner's Approach, 5th Edition*, McGraw-Hill, 2001.

[15] Frank Maurer and Sebastien Martel, "Extreme Programming Rapid Development for Web-Based Applications," IEEE INTERNET COMPUTING, JANUARY·FEBRUARY 2002.

[16] Microsoft, Application Architecture for

.NET:Designing Applications and Services, Microsoft Corporation, 2002.

[17] Gunther Lenz, Thomas Moeller, .NET-A Complete Development Cycle, Pearson Education, Inc. 2004.

[18] Christian Thilmany, .NET Patterns Architecture, Design, and Process, Pearson Education, Inc. 2004.

노재우(Jae Woo Noh)

1984년 2월 숭실대학교 전자계산학과 학사(공학사)
1993년 8월 숭실대학교 정보과학대학원 정보산업학과 석사(공학석사)
2001년 8월 숭실대학교 대학원 컴퓨터학과 박사수료
1984년 4월 ~ 2003년 12월 정평모비컴㈜ 대표이사
2004년 1월 ~ 2004년 현재 정평모비컴㈜ 대표 컨설턴트
관심분야 객체지향소프트웨어공학, 소프트웨어 재사용, 컴포넌트 기반 소프트웨어공학, 닷넷 시스템 개발 방법론

조현훈(Hyun Hoon Cho)

1995년 2월 광주대학교 전자계산학과 학사(공학사)
1997년 2월 숭실대학교 대학원 컴퓨터학과 석사(공학석사)
2003년 2월 숭실대학교 대학원 컴퓨터학과 박사(공학박사)
2001년 9월 ~ 2004년 현재 ㈜크래빅스 방법론연구팀장
1999년 3월 ~ 2004년 현재 신흥대학 컴퓨터정보계열 겸임교수
관심분야 소프트웨어개발방법론, Web Application 개발, 소프트웨어 유지보수, 소프트웨어 재사용, 리엔지니어링, 컴포넌트 기반 소프트웨어 공학, 소프트웨어 테스팅, 닷넷 기반 기술 및 웹서비스

류성열(Sung Yul Rhew)

1997년 2월 아주대학교 컴퓨터학부(공학박사)
1997년 3월 ~ 1998년 3월 George Mason University 교환교수
1981년 3월 ~ 현재 숭실대학교 정보과학대학 컴퓨터학부 교수
1998년 3월 ~ 2001년 2월 숭실대학교 정보과학대학원 원장
1998년 3월 ~ 현재 숭실대학교 전자계산원 원장
2002년 9월 ~ 현재 숭실대학교 대학원 컴퓨터학과 주임교수
관심분야 리엔지니어링, 소프트웨어 유지보수, 소프트웨어 재사용, 소프트웨어 재공학/역공학, 소프트웨어 테스팅, 컴포넌트 기반 소프트웨어공학