

CBD 방법론 비교 분석

한국과학기술원 전자전산학과 전산학 전공

전상욱, 김인규, 김정윤, 윤경아, 배두환

목 차

1. 서 론	4. 결 론
2. CBD 방법론 비교를 위한 기준	5. 참고문헌
3. CBD 방법론 비교 결과	

1. 서 론

컴포넌트 기반 개발(Component Based Development, CBD)은 소프트웨어 개발 패러다임 진화의 최첨단에 위치하며 높은 품질의 소프트웨어를 신속하고 효과적으로 개발할 수 있는 방법으로써 각광 받아 왔다.

이러한 높은 생산성과 재사용성을 제공하는 CBD 방법론에 대한 산업계의 관심이 커지기 시작한 요즈음, 많은 회사들은 CBD 방법론을 도입하기 시작하였다. 하지만 방법론마다 적용 분야, 개발 시스템의 크기 등 적용 대상이 다양해서 각 회사마다 회사 업무의 특성을 제대로 반영하는 방법론을 선택하기 위해 기존 CBD 방법론들을 충분히 검토하고 비교해야 하는 필요가 발생하게 되었다.

본 논문에서는 각 회사나 조직에 적합한 방법론을 선정하는데 도움을 주고자 CBD 개발 방법론을 비교 및 분석하였다. 알려진 여러 컴포넌트 기반 개발 방법론들[2,5,6,8,9] 중에 널리 사용되는 네 가지 CBD 방법론을 대상으로 비교/분석을 실시하여 각 방법론을 평가함으로써 각 방법론의 장단점 및 특성을 파악하고자 하였다. ‘UML Component’는 방법론이라기 보다는 디자인 기법에 더 가까우므로 본 논문의 대상에서 제외되었다.

본 논문의 구성은 다음과 같다. 2장에서는 CBD 방법론의 비교를 위한 기준을 설명하고 비교 대상이 되는 주요 방법론에 대해 간단히 소개한다. 3장에서 이 기준을 가지고 기존 네 가지 CBD 방법론을 비교/분석한 결과를 기술한다. 마지막으로, 4장에서 CBD 방법론의 추후 방향에 대한 토론으로써 결론을 맺는다.

2. CBD 방법론 비교를 위한 기준

이 장에서는 CBD 방법론을 비교 및 분석하기 위한 기준과 비교 대상으로 선정한 주요 CBD 방법론 네 가지에 대해서 간략하게 설명한다.

2.1 CBD 방법론 비교기준

CBD방법론 비교 기준은 크게, 방법론 개요(Overview), 개념지원(Concept), 표기법

(Notation), 프로세스 지원(Process), 방법론에 대한 지원(Support), 기타(Others)로 나누어 볼 수 있다.

2.1.1 방법론 개요

방법론의 비교에 앞서서 해당 방법론의 기본적인 사항에 대한 항목들이 포함된다. 방법론 개요에 포함되는 항목은 <표 1>에 나타나 있다.

<표 1> 방법론 개요의 세부 항목

항목	설명
저자	방법론을 만든 사람 또는 단체
정의	저자가 방법론에 대해 정의한 내용
목적	방법론의 목적
영향 받은 방법론	이전 방법론과의 관계
특징	중심 아이디어, 다른 방법론과의 차이점 또는 특징, 주도기관, 장단점
방법론의 크기	방법론의 구성 및 구체적 성격(heavy or light)

2.1.2 개념 지원

객체지향 방법론에서도 각 방법론마다 객체(Object), 클래스(Class) 등의 개념을 조금씩 다르게 정의하고 있다. 마찬가지로 CBD 방법론에서도 방법론마다 컴포넌트(Component)나 컴포넌트 조합(Component composition)등에 대한 개념을 달리 하고 있다[7]. 각 방법론에서 정의하는 이들 개념을 먼저 살펴볼 필요가 있다. <표 2>에 개념 지원 항목에 포함되는 세부 항목들이 나타나 있다.

<표 2> 개념 지원의 세부 항목

항목	설명
컴포넌트 개념지원	컴포넌트 정의, 컴포넌트 명세 방법, 표기법에 대한 정의
컴포넌트 조합 개념 지원	컴포넌트들을 조합하여 어플리케이션으로 만들 때 필요한 조합 또는 협동(Collaboration) 개념에 대한 정의
아키텍처 개념 지원	아키텍처 관점(Architectural view), 아키텍처 설계 방법에 대한 설명
재사용 대상	재사용 가능한 산출물에 대한 정의

2.1.3 표기법

각각의 방법론에서 요구하는 모델, 그리고 각 모델에서 사용하는 표기법이 각각 다르다.

따라서 방법론을 비교하기 위해서는 이들 표기법들을 고려해야 한다[4]. 이에 해당하는 세부 항목은 <표 3>과 같다.

<표 3> 표기법 항목의 세부 항목

항목	설명
사용하는 다이어그램	방법론에서 사용하고 있는 다이어그램
모델링 기술	모델링을 위해 필요한 기술(UML, UML extension 등)

2.1.4 프로세스 지원

이 항목은 각 방법론이 사용되기 적합한 개발 도메인은 어떤 곳인가, 소프트웨어 개발 라이프 사이클의 어디까지를 지원하는가, 그리고 각 방법론의 프로세스에 대해 어떤 형태의 맞춤화가 가능한가 등을 통해 각 방법론의 특징을 파악하기 위함이다[4, 6]. 세부 항목은 <표 4>와 같다.

<표 4> 프로세스 지원 항목의 세부 항목

항목	설명
라이프사이클 지원정도	방법론의 전체 프로세스 및 해당 프로세스가 지원하고 있는 소프트웨어 개발 단계 기술
세부 단계별 행위	방법론의 프로세스를 구성하는 세부 단계 및 각 단계마다의 행위와 산출물
CBD 프로세스 요구사항 지원 정도	재사용 컴포넌트 확인, 컴포넌트 명세 등 CBD 프로세스가 갖추어야 하는 항목들에 대한 지원 정도
특징	비기능적 요구사항에 대한 처리, 추적성 등 해당 프로세스의 특징

2.1.5 방법론에 대한 지원

각 방법론들마다 예제나 튜토리얼 등의 교육 자료 또는 해당 방법론을 지원하는 도구들은 어떤 것이 있는지를 알아보기 위함이다. 세부 항목은 <표 5>와 같다.

<표 5> 방법론에 대한 지원 항목의 세부 항목

항목	설명
지원 자료	예제, 튜토리얼, 교재 등 방법론을 지원하고 있는 자료
지원 도구	방법론을 사용할 때 이용할 수 있는 도구

2.1.6 기타

기타 항목은 <표 6>과 같다.

<표 6> 기타 세부 항목

항목	설명
방법론 사용 현황	방법론이 얼마나 사용되고 있는가 또는 방법론이 얼마나 성숙했는가
요구되는 배경 지식	방법론을 익히기 위해 필요한 기본 지식

2.2 주요 CBD 방법론

산업계나 학계에서 많이 사용되고 있고 비교적 자료의 획득이 쉬운 아래의 네 가지 CBD 방법론을 분석하였다.

- RUP[2]: Rational Company에서 개발한 방법론으로 산업분야의 목적에 부합되는 특성을 가짐으로써 많은 업체에 의해 채택되어 활용되고 있다.
- Catalysis[3]: 주로 학계에서 연구대상으로 채택된 방법론으로서 이 방법론이 포함하고 있는 다양한 기초개념과 기법은 기타 다른 방법론들의 형성에 영향을 주었다.
- Select Perspective[5]: 산업분야의 목적에 의해 개발된 방법론으로서 다양한 모델링 방법론들을 채택하고 있어 다양한 비즈니스 모델링을 지원한다.
- 마르미-III[8]: 한국전자통신연구원에서 만든 컴포넌트 기반 소프트웨어 개발 방법론이다. 마르미-III는 컴포넌트의 개발 및 컴포넌트 기반의 시스템 개발에 필요한 작업과 작업 수행에 필요한 기법 및 작업별 산출물을 정의하고, 작업에 따른 상세한 개발 절차와 지침을 제공한다

3. CBD 방법론 비교 결과

2장에서 제시한 기준에 따라 주요 CBD 방법론으로 뽑히는 4가지 방법론을 비교해 보았다. 비교 결과는 <표 7>과 같다.

<표 7> CBD 방법론 비교 결과

비교기준 방법론		RUP	Catalysis	Select Perspective	마르미-III
overview (개요)	저자	Ivar Jacobson, Grady Booch, James Rumbaugh	Desmond Francis D'Souza	Stuart Frost, Paul Allen	한국전자통신연구원

비교기준 방법론		RUP	Catalysis	Select Perspective	마르미-III	
	이론	매우 큰 소프트웨어 시스템이나, 여러 다른 어플리케이션 영역, 다양한 조직, 다양한 프로젝트 크기에 전문화될 수 있는 포괄적인 프로세스 프레임워크이다	UML 표기법을 기반으로 한 방법론으로서 컴포넌트 기반 방법론	변화에 대해 유연하며 적용 가능한 방법론을 제공하기 위해 컴포넌트 기술을 채택하고 있으며, 아키텍처 프레임워크 내의 프세스 템플릿을 활용한다. 또한 업계에서의 가장 실질적으로 받아들여진 모델링 기술들의 집합이라고 볼 수 있다.	컴포넌트 기반 개발 방법론	
	프레임	예측할 수 있는 계획과 예산의 범위 내에서 사용자의 요구 사항을 만족시킬 수 있는 고품질의 소프트웨어를 생산하는데 도움을 준다	개방적 분산 시스템의 모델링과 구축을 지원	어플리케이션 개발의 실질적 접근방법의 제공	체계적인 컴포넌트 기반 개발 및 관리 절차 확립	
	이론, 프레임, 방법론	Rational Objectory Process, Object-Oriented Modeling and Design, The Ericsson Approach	초기에 객체 중심의 방법론으로 시작, Rumbough의 OMT 기법을 수용하고 UML을 받아들여 컴포넌트 중심으로 개선됨	소프트웨어 엔지니어링 뿐만 아니라 데이터 베이스 등 다양한 분야의 방법론에 영향을 받음. UML, BPM, ER 등.	마르미-I, 마르미-II	
	프레임	유즈케이스(Use case)를 기반으로 모든 프로세스가 진행된다. 또한 아키텍처 중심의 디자인을 수행하며, 반복적인(Iterative) 과정을 거쳐 점진적인(Incremental) 디자인을 한다.	학계에서 주도되어 비즈니스 중심의 개발방법론으로 발전되었으며 많은 기술적 개념들을 포함. 실제 업계에서 많이 활용되지는 않음.	업계에서 주도되어 발전하였으며, 실무적용에 성공적인 기법들을 적극적으로 포용함	- 개발 프로세스 분리 정의(CDM, CBD) - 구체적이고 실용적임 - 어플리케이션 패밀리 개념 - EJB 명세 기반 - 기타: UML 사용, 유즈케이스 기반, 반복적 및 점진적 개발, 아키텍처 중심, 위험관리	
	이론, 프레임	Heavy weight	실질적으로 적용할 경우 heavy weight, 그러나 프로세스의 맞춤화 결과에 따라 light weight형태도 가능.	Heavy weight	Heavy weight	
개념지원(Concept)	컴포넌트	이론	구현물을 포장하고, 주어진 인터페이스에 따르며 그를 제공하는, 시스템의 교체될 수 있는 부분. 코드(소스, 바이너리 또는 실행파일)나 스크립트와 같은 시스템 구현의 물리적인 부분이다.	상호작용(Interactive)을 하는 객체들의 집합. 컴포넌트는 단순히 소프트웨어의 집합일 뿐만 아니라 하나의 비즈니스 행위가 될 수도 있다.	서비스와 관련된 블랙박스 캡슐화(Encapsulation)를 제공하는 코드의 실행 가능한 단위	시스템의 재사용 및 대체 가능한 최소단위의 부품
		프레임	몇 개의 스테레오 타입(Stereotype)으로 표현	프레임워크(Framework), 패턴(Pattern) 등 객체의 상호작용을 중심으로 기술된 컴포넌트	UML기반으로 컴포넌트 기술	컴포넌트 명세서 존재

비교기준 방법론			RUP	Catalysis	Select Perspective	마르미-III
		표기법과 모델	탭(Tab)이 붙어 있는 사각형	- 다이어그램 : 인터페이스 명세, 컴포넌트 명세, 컴포넌트 아키텍처 - 스테레오 타입 사용 예) <<comp spec>>	UML 컴포넌트 다이어그램	컴포넌트 명세서, 컴포넌트 인터페이스 정의서, 컴포넌트 협력도, 비즈니스컴포넌트 모형 기술서, 컴포넌트 설계서
		컴포 지변	객체들간의 협동 다이어그램	컴포넌트 킷(kit) 개념을 적용하여 조립	배치(deployment) 다이어그램에서 하나의 노드에 할당되는 컴포넌트들을 나열하여 표시	컴포넌트 협동 다이어그램과 시퀀스 다이어그램
		재사용 대상	소프트웨어 컴포넌트	모든 산출물은 재사용 대상이며 프레임웍이라는 일반화된 업무형태를 재사용 형태로 컴포넌트 재사용을 지원.	모든 산출물이 재사용 대상이며 'Select Component Manager' 와 같은 툴을 이용하여 컴포넌트를 관리	모든 산출물이 재사용 대상이다(특히, 재사용 컴포넌트)
	아키텍처	아키텍처관련	Elaboration 단계에서 각 워크플로우(Workflow)마다 각각의 모델의 관점에서 본 아키텍처를 가지고 있다.	소프트웨어의 구조를 의미할 뿐만 아니라 각 구조물간의 코드수준에서 활용될 수 있는 관계에 관한 내용을 포함	점진적 개발에 의해 추가되는 구성요소	컴포넌트 아키텍처, 구현 중심의 응용 아키텍처, 최종 구현될 시스템 아키텍처
		아키텍처 설계 방법	부분적인 가이드라인 제시	디자인 단계에서 개발자와 유지 관리자 들의 임의 개발이나 수정을 할 수 없도록 한 결정들의 조합이란 원칙을 제시	서비스에 기초하여 구축하도록 제안	설계 절차나 지침을 비교적 자세히 제공
	표기법(Notations)		사용 다이어그램	활동 다이어그램(Activity diagram), 클래스 다이어그램, 협동 다이어그램, 컴포넌트 다이어그램, 협동 다이어그램, 상호작용 다이어그램, 시퀀스 다이어그램, 상태 다이어그램(Statechart diagram), 유즈케이스 다이어그램	타입, 협동, 프레임웍, 시퀀스 다이어그램, 유즈케이스 다이어그램, 상태 다이어그램	비즈니스 프로세스 모델(BPM), UML, BPM, UML, 논리적 데이터 모델 (CCTA)
모델링 기술			UML	UML 기반이나 다소 변형된 형태의 표기법을 개발	UML, BPM, ER 다이어그램, CCTA 등 다양한 표기법 사용	UML, PERT/CPM, Gantt Chart, 피시본 다이어그램, 파레토 다이어그램

비교기준 방법론		RUP	Catalysis	Select Perspective	마르미-III	
프로세스(Process)	라이프사이클 지원 정도	비즈니스 모델링 요구사항 분석 설계 구현 테스팅	사이클을 대략 비즈니스 모델 확립(Build), 컴포넌트 명세, 아키텍처 디자인, 컴포넌트 구현 의 네 단계로 나눌 수 있으나 선형적이지 않음	전체적으로 ‘정렬’, ‘설계’, ‘조립’의 세 과정으로 나누어지며 각 과정은 복수의 반복단계로 구성되어 있으며, 각 단계는 다시 ‘비즈니스 계획’, ‘기술 아키텍처’, ‘컴포넌트 관리’, ‘컴포넌트 개발’, ‘솔루션 개발’의 여섯 단계의 활동으로 나뉜다.	계획단계, 아키텍처단계, 점진적 개발단계, 인도단계	
	세부단계별 행위 및 기술	요구사항	비즈니스 모델링 요구사항	비즈니스 모델 확립에 해당 (산출물: 스토리 보드(Story board), 배경도(Context map), 유즈케이스 다이어그램)	‘비즈니스 계획’ 활동에 해당되며 매 단계마다 반복된다. (산출물: BPM, 유효성 연구(Feasibility study) 리포트, 유즈케이스 다이어그램)	계획단계 (요구사항이해, , 요구사항 정의, 개발전략수립, 프로젝트계획)
		분석	분석	비즈니스 모델 확립에 해당(산출물: 유즈케이스 다이어그램, 타입 다이어그램, 협동 다이어그램)	‘비즈니스 계획’ 과 ‘기술 아키텍처’ 활동에 해당되며 매 단계마다 반복된다. (산출물: 유즈케이스 다이어그램, 클래스 다이어그램)	아키텍처단계 (요구사항분석, 컴포넌트 식별, 아키텍처정의, 컴포넌트 명세 작성, 아키텍처 프로토타이핑, 점진적 개발계획)
		설계	설계	컴포넌트 명세, 아키텍처 디자인에 해당 (산출물: 타입 다이어그램)	‘기술 아키텍처’ 및 ‘컴포넌트 개발’에 해당되며 매 단계 마다 반복 (산출물: 컴포넌트 다이어그램)	점진적 개발 단계 (요구 사항 및 아키텍처 정제, 컴포넌트 설계, 컴포넌트 구현, 지침서 개발, 컴포넌트 테스트, 컴포넌트 통합테스트, 시스템 테스트)
		구현	구현	컴포넌트 구현에 해당	‘컴포넌트 개발’ 및 ‘솔루션 개발’에 해당되며 매 단계 마다 반복	
		테스팅	테스팅	프로세스 전 과정에서 수행됨	프로세스 전 과정에 반복	인도단계 (시스템설치, 사용자교육, 설치 후 관리, 사용자인수테스트)
	CBD 프로세스 요구사항 지원정도	재사용 컴포 넌트 활용	X	프레임워크 (일반화 된 협동)를 이용, 컴포넌트의 재사용성 지원.	‘Select Component Manager’와 같은 자동화된 도구 이용	재사용 컴포넌트 선정 작업과 컴포넌트 획득 작업에 대해 간단히 기술
		컴포넌트 기술	X	협동, 타입 다이어그램을 통해 행위 중심의 컴포넌트 표현	UML 컴포넌트 모델 활용	인터페이스 식별, 컴포넌트 상호작용 분석, 컴포넌트 명세화 작업에 대해 자세히 기술
		컴포넌트 구현	소스코드 패키지 수준에서 서술	초기 단계에서 선정된 비즈니스 모델을 컴포넌트로 구축하는 과정 명시	반복을 통해 컴포넌트를 생산하며 관리도구에 의해 추적, 관리됨.	클래스 구조 설계, 상호 작용 설계, 가변성 적용 설계, 패턴 적용 설계, UI 설계, DB 설계, 컴포넌트 구현 설계, 코딩, UI 구현, DB 구축, 데이터 전환 준비 작업의 일부 기술

비교기준 방법론			RUP	Catalysis	Select Perspective	마르미-III
		어플리케이션 수준	X	컴포넌트의 인터페이스를 바탕으로 커넥터 등을 이용하여 조립	어플리케이션의 기능이 반복이 진행되면서 컴포넌트가 추가 확장되는 형식으로 진전됨.	컴포넌트 식별, 아키텍처 정의, 컴포넌트 명세 작성, 컴포넌트 통합 테스트 활동에 대해 간단히 기술
		컴포넌트	X	비즈니스 모델이 타입, 협동 등으로 표현되면서 컴포넌트로 이어진다.	비즈니스 요구사항 명세를 투입하고 이를 구현한 컴포넌트를 수확한다.	사용자설명서 작성, 컴포넌트 인증, 등록 및 배포 작업에 대해 부분적으로 기술
	종류	대기 이용 사항	Concurrency	도메인 모델과 시스템 배경 분석과정에서 고려할 것을 명시.	유효성 연구에서 유즈케이스를 선정하는 단계에서 고려.	사용자 인터페이스(UI)
		추진	명확하다	비즈니스 모델부터 구현까지 연결되어 있다	비즈니스 모델, 생성된 컴포넌트, 기능이 추가된 어플리케이션까지 연결	단계, 활동, 작업 별로 각각 입력물과 산출물을 명시
		제한	반복의 횟수	개발 대상의 특성에 따라 프로세스를 맞춤화하여 계획한다. (탄력적으로 프로세스를 구성)	반복의 횟수	현재는 직접적 언급 없다. 그러나, 미니프로젝트 개념을 이용하여 반복적이고 점진적인 작업을 할 수 있다. 향후, “방법론 조정 방안” 제시
		어플리케이션 / 컴포넌트 개발	컴포넌트 기반 어플리케이션 개발 < 컴포넌트 개발 (객체지향개발)	컴포넌트 기반 어플리케이션 개발 < 컴포넌트 개발	컴포넌트 기반 어플리케이션 개발 > 컴포넌트 개발	컴포넌트 기반 어플리케이션 개발 > 컴포넌트 개발
	지원(Support)	지원 자료	Tutorial(Rational Homepage) 교재[Rumbaugh1999]	저자의 책, 홈페이지(http://www.catalysis.org/)	저자의 책, 홈페이지(http://www.selectst.com/)	적용사례, 설명 자료, 절차서, 기법, 산출물 양식 등 제공
		지원 도구	Rational Suite series	직접적인 도구는 거의 없으나 Smart Draw등이 활용됨	SELECT, SELECT Enterprise, SELECT Component manager™, Perspective Patterns™	기존의 UML 도구
	기타 (Others)	비즈니스 요구	가장 잘 알려지고 사용됨	업계에서 일부 사용됨	다수 주요기업에 의해 사용됨	아직까지 활발히 사용되고 있지는 않음
		요구되는 배경지식	UML	Catalysis에서 변형, 확장된 UML 기반의 표기법 및 프레임워크, 협동, 패턴에 대한 이해.	BPM, UML, CCTA 등 Select에서 도입하여 활용하고 있는 기법에 대한 이해. 이미 광범위하게 활용되고 있는 방법들이 Select에서 채택되어 있어 개발자들의 숙련에 유리함.	UML, 아키텍처, EJB, 어플리케이션 패밀리에 대한 지식 요구됨

4. 결 론

본 논문에서는 현재 널리 알려져 있는 주요 CBD 방법론 4개를 비교해 보았다. 각각의 방법론마다 적용 가능한 대상이 있고 장단점이 다르기 때문에 특정 회사나 조직에 맞는 방법론을 분류할 수는 있으나 절대적으로 다른 방법론에 비해 효과적인 방법론을 단정

짓기는 어렵다. 하지만 주요 방법론의 비교를 통해서 각 방법론의 특징을 파악함으로써 회사 및 프로젝트의 성격에 맞는 CBD 방법론을 선택하는데 도움을 줄 수 있다. 또한 회사나 조직에서 사용하는 방법론과도 비교함으로써 해당 방법론의 향상에 도움을 줄 수 있을 것이다.

향후 방법론의 추세는 특정 방법론을 선정하여 모든 프로젝트에 같은 방법론을 획일적으로 적용하는 것이 아니라, 필수적인 요소로 기본적인 방법론을 표준화하고, 각 프로젝트의 특성에 따라 맞춤화(Customizing)하는 방향으로 진행되고 있다. 이는 과거에 비해 개발하는 시스템의 종류가 다양해지고, 단순한 방법론의 적용으로 많은 문서를 도출해내는데 만족하지 않고, 프로젝트의 규모, 인원, 기간, 목적, 제한 조건 등을 반영하여 효율적인 소프트웨어 개발을 하고자 하는 경향이 진행되고 있음을 알 수 있다. 또한 요즘 소규모 프로젝트에 개발자들 사이에 관심이 높아져 가고 있는 light weight 방법론에 대한 연구 및 적용도 새로운 추세 중에 하나이다. 다만, 컴포넌트 기반 개발의 경우 정확한 명세 작업이 필수적인데, 기존의 light weight 방법론은 이러한 요구를 수용하기에는 부적절한 면이 없지 않아 이에 대한 연구가 더 필요할 것이다.

과거 소프트웨어 개발 자체에 목적을 둔 방법론의 적용에 비해, 향후는 비즈니스 목적과 소프트웨어 개발, 특히 소프트웨어 설계를 통합하는, 다시 말하면 소프트웨어 개발 자체보다는 결과물이 비즈니스 목표에 부합이 잘 될 수 있는 방향으로 방법론이 적용될 전망이다. 마지막으로 국내의 경우 단순한 방법론의 적용이 아닌 CMM(Capability Maturity Model)의 특정 level, 예를 들면 level 3 수준에 부합하는 방법론의 적용도 점차 확대될 것으로 예상된다.

5. 참고문헌

- [1] N. Boertien, M. Steen, and H. Jonkers, "Evaluation of Component-Based Development Methods", Sixth CAiSE/IFIP8.1 International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design, 4-5 June, 2001
- [2] I. Jacobson, G. Booch, and J. Rumbaugh, The Unified Software Development Process, Addison-Wesley, 1999.
- [3] D. D'sousz and A. Wills, Objects, Components, and Frameworks with UML: The Catalysis Approach, Addison-Wesley, 1998.
- [4] G. Karam and R. Casselman, "A Cataloging Framework for Software Development Methods", IEEE Computer, 26(2):34-46, February 1993.
- [5] "Select Perspective," White Paper, Princeton Softech, Jan., 10, 2000, <http://www.pricetonsoftech.com/index.asp>.
- [6] K. Ring and N. Ward-Dutton, *Componentware: Building it, Buying it, Selling it*, Ovum Ltd, 1998.

- [7] “*A Comparison of Object-Oriented Methodologies*”, The Object Agency, 1995,
<http://www.toa.com/shnn?oosearticles>.
- [8] 마르미-III, <http://www.component.or.kr>
- [9] J. Cheesman and J. Daniels, UML Components: A Simple Process for Specifying Component-Based Software, Addison-Wesley, 2000.