

컴포넌트 명세의 Z 활용

장종표[†] · 이상준^{††} · 김병기^{†††}

요 약

소프트웨어 생산성이 사용자들의 서비스에 대한 요구를 만족시키지 못했고, 소프트웨어 품질이 향상되지 않았으며, 유지보수가 어려운 문제점을 가지고 있다. 이러한 위기를 해결하기 위한 한가지 대응책으로 최근 컴포넌트기반 소프트웨어 공학 혹은 컴포넌트웨어가 등장하였다. 컴포넌트기반 소프트웨어공학을 이루는 관련 기술들 중 하나인 소프트웨어 아키텍처는 시스템의 구조적 기술로서 시스템을 구성하는 컴포넌트와 그 컴포넌트들 사이의 상호작용을 기술한 것이다. 소프트웨어 아키텍처는 컴포넌트기반 소프트웨어공학에서의 컴포넌트 사이의 조합에 대한 기술 및 방법론을 제 공함으로써, 컴포넌트기반 소프트웨어공학의 핵심 기술 중 하나로 자리잡고 있다.

본 논문에서는 컴포넌트 명세를 위해서 필요한 정보들을 분석하고, 분석력과 논리성이 검증된 정형 명세 언어 Z를 이용하여 컴포넌트를 명세하는 활동을 제안하였다. 제안 활동은 9개의 태스크 들로 구성되며 11개의 산출물을 구체적으로 제시함으로써 재사용을 위한 컴포넌트 기반 소프트웨어 개발의 기초가 되는 컴포넌트 명세 활동을 제안하였다

Component Specification Using Z

Jong-Pyo Jang[†] · Sang-Jun Lee^{††} · Byung-Ki Kim^{†††}

ABSTRACT

Software productivity doesn't satisfy the need of service of software users and software quality isn't improved. Moreover, we still have difficulty in software maintenance. As a plan to counteract solving this crisis, the technology that is called CBSE or Componentware is introduced. Component Based Software Architecture, one of the technologies associated with CBSE, as a structural description of system, describes both the components composed system of and interaction among these components. Software Architecture provides the technology and the methodology of composition among components in the field of CBSE. is accepted one of core technologies.

In this thesis, we analyse information necessary for component specification and then proposed that component specification activities using formal specification language Z which is verified with the ability of analysis and logicity. The proposed activities are composed of 9 tasks. By presenting specific 11 products, it is also proposed that component specification activities which is the base of CBSD(ComponentBased Software Development) for reusing.

1. 서 론

소프트웨어 위기 문제에 대한 대안으로 소프트웨어 재사용 기술이 소개되면서 소프트웨어 공장 개념이 도입되었다. 80년도부터 이와 같은 소프트웨어 재사용 시장을 구축하기 위한 기술로 객체지향 프로그래밍이 본격적으로 도입되기 시작했다. 그러나

[†] 준 회원: 전남대학교 전산학과 박사과정

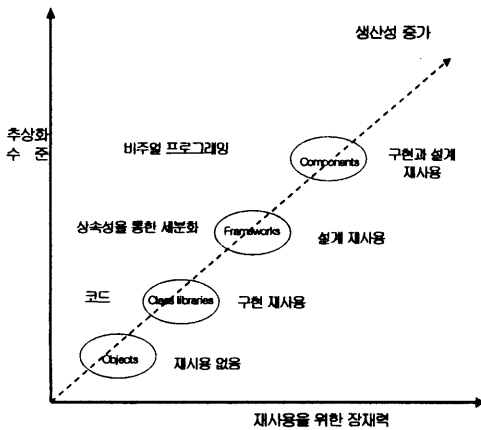
전남대학교 정보통신연구소 연구원

^{††} 준 회원: 서남대학교 컴퓨터 영상 정보통신학부 교수

^{†††} 준 회원: 전남대학교 컴퓨터 정보학부 교수

논문접수: 2000년 8월 24일, 심사완료: 2000년 11월 10일

클래스 라이브러리 단위의 재사용도 충분한 소프트웨어 재사용성을 제공하지는 못하고 있다. 소프트웨어 개발자들은 여전히 어플리케이션 코드의 대부분을 새롭게 작성하고 있다. 클래스 라이브러리 단위로 재사용을 하는 데 있어서 복잡성이나 중복성, 그리고 제어 흐름으로 인한 여러 한계점이 따르기 때문이다. 이런 한계점을 해결하기 위한 한가지 대응책으로 최근 컴포넌트웨어(Componentware) 혹은 컴포넌트기반 소프트웨어 공학(Component-Based Software Engineering, CBSE)이 등장하였다(그림 1).



(그림 1) 소프트웨어 재사용 기술의 변화

CBSE를 이루는 관련 기술들 중 하나로써 컴포넌트기반 소프트웨어 개발 프로세스들이 제안되고 있다[2][3][4]. 특히, 재사용을 목적으로 하는 컴포넌트기반 소프트웨어 개발 프로세스에서 컴포넌트의 품질은 매우 중요하다. 컴포넌트 개발 프로세스의 초기활동, 즉 명세활동에서 명세의 정확성과 명세의 검증을 통하여 에러를 찾아낸다는 것은 컴포넌트의 전체 품질에 매우 중요한 의미를 갖는다. 그러나, 기존의 컴포넌트기반 소프트웨어 개발 프로세스에서는 무엇이 전달되고 있는가에 대하여 명확성이 결여된 자연어를 이용하여 명세활동을 수행하였고, 명세활동에 있어서 품질관리는 검사(review)만을 통하여 이루어졌다. 이러한 문제의 해결책으로 명세활동의 정형화가 필요하다.

본 논문에서는 컴포넌트 명세를 위해서 필요한 정보들을 분석하고, 분석력과 논리성이 검증된 정형 명세 언어 Z를 이용하여 컴포넌트를 명세하는 활동에 관하여 연구하였다. 최근 교육용 소프트웨어가 많이 개발되고 있지만, 실제 재사용의 사례를 거의 찾아보기 힘들다. 그러나, 본 논문에서 제안하는 활동을 이용한다면 재사용 가능한 교육용 소프트웨어를 개발할 수 있을 것이다.

2. 관련연구

2.1. 컴포넌트기반 소프트웨어 개발 방법론

2.2.1 Catalysis

Catalysis[2][3]는 UML, the OMG, RM-ODP를 포함하는 표준위에 생성된 객체와 프레임워크를 가진 컴포넌트 기반 개발을 위한 지원을 제공하는 방법론이다.

(그림 2) Catalysis 프로세스

2.2.2 CBD 96

CBD 96[2] 개발 방법론은 Catalysis를 기반으로 하고 자체적으로 컴포넌트 모델링 프로세스를 포함하고 있으면 여러 가지 도구에 의해서 지원된다.

- 정형적 명세서의 개발은 소프트웨어 요구 사항과 소프트웨어 설계에 대한 고찰과 이해를 제공한다.
- 정형화된 소프트웨어 명세서를 수학적인 개체로 표현하고, 수학적 방법론을 사용하여 분석될 수 있다.
- 정형적 명세서는 자동적으로 처리될 수 있다.
- 정형적 명세서는 구성 요소의 검사자가 적당한 검사를 할 수 있도록 도와주는 안내자 역할을 할 수도 있다.

(그림 3) CBD 96 프로세스

2.2.3 Fusion

Fusion[2][4] 방법론은 글자 그대로 Booch, Rumbaugh, 다른것뿐만 아니라 Objectory를 포함하는 다중 방법의 가장 좋은 컴포넌트의 정성들여 결합한 것이다.

<표1>은 대수적 접근방법과 모델기반 접근방법에 사용되는 정형 명세 언어를 정리한 것이다.

<표 1> 정형 명세 언어

접근방법	순차적 시스템	병렬 시스템
대수적 접근방법	Larch (1985) OBJ (1985)	Lotos (1987)
모델기반 접근방법	Z (1992) VDM (1980)	CSP (1985) Petri Nets (1981)

2.3. Z [6]

Z는 표준 집합론에 기반하는 정형표기이고, 시스템의 특성을 표현하기 위해서 양식화된 수학적인 표기를 사용한다. Z 명세는 스키마를 사용하여 구조화된다. 스키마는 묘사의 정형적인 부분을 포함할 뿐 아니라, 비정형적인 표현으로부터 정형 묘사를 분리하는 것을 도와준다. Z는 3가지 표기법을 가지고 있는 상호교환언어이다. 하나는 특별한 도식적인 기호들을 사용하고, 다른 2개는 전적으로 문자에 기반한다.

(그림 4) Fusion Process

2.2. 정형 명세 언어

정형적 소프트웨어 명세서란 어휘, 구문, 의미가 정형적으로 정의된 언어로 표현된 명세서이다. 정형 명세 언어[5]는 자연어에 기초할 수 없기 때문에 정형적 의미 정의가 필요한 것이고, 수학에 기초를 둔다. 이러한 정형 명세를 사용함으로써 얻을 수 있는 잠재적인 기술적 이점들은 다음과 같다.

3. Z를 이용한 컴포넌트 정형 명세

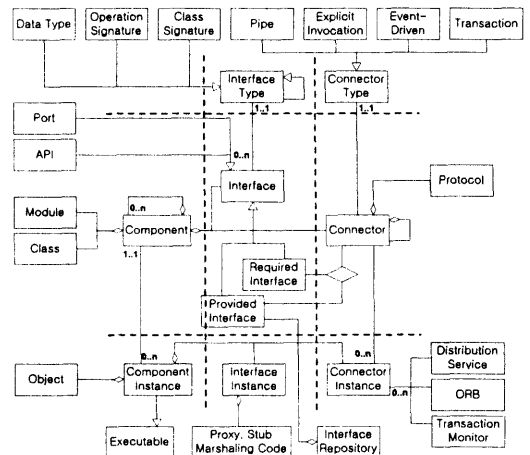
3.1. 컴포넌트 명세 정보 분석

컴포넌트 개발을 지원하는 대표적인 컴포넌트 모델에는 COM/DCOM, JavaBeans/EJB, CORBA 컴포넌트 모델 등이 있다. 이들 각각의 모델에서 컴포넌트 명세를 위한 사용하고 있는 정보는 (그림5)(그림6)(그림7)(그림8)(그림9)와 같다[7][8][9].

<표 2> 주요 컴포넌트 모델 비교

구분	COM	JavaBeans	CORBA
현상형	NT에서 제공	Java에서 제공	계속 개발중
구현	OS	Java virtual machine	다양한 third-party 제품
언어 독립성	좋음	없음	좋음
플랫폼 독립성	제한됨	좋음	좋은
이벤트 모델	없음	Java 언어	제품에 따름
인트로스펙션	개발자가 직접제공	항상 가능	아직지원안함 (3.0에서 지원예정)

또한 컴포넌트에 관해 많은 연구와 토론이 있지만, 아직까지 좋은 컴포넌트 모델이 제시되지 못하고 있다. 잘 정의된 컴포넌트 모델은 다른 개념들로부터 컴포넌트 개념들을 구별하는데 쉽게 하며, 기존 기술의 문맥 속에서 어떻게 존재하는지를 이해하는데 도움을 준다[10]. (그림 10)은 컴포넌트의 구성요소를 잘 나타내고 있는 컴포넌트 모델이다.



(그림 10) 컴포넌트 개념 모델

그러나, 모든 컴포넌트 모델에서 제공하고 있는 정보는 플랫폼에 독립적이거나 특별한 구현에 종속되기 때문에 컴포넌트를 명세를 위해 불충분한 정보를 제공하고 있다. <표2>는 주요한 컴포넌트 모델들을 비교한 것이다.

그러나, 아키텍처 모델링 언어에 의해 아키텍처의 무슨 측면이 모델화되어야 하는지에 대한 연구계에서 의견 일치가 부족하다. 아키텍처 묘사가 제공해야만 하는 최소한의 요소들이 제안되었다[11]: 주요한 요소는 컴포넌트, 커넥터, 환경이 될 수 있다. 모든 ADL(Architecture Description Language)은 묘사에 있어서 드러나는 컴포넌트를 모델링하기 위한 능력을 제공해야만 한다. 또한 ADL은 컴포넌트들

사이의 통시를 위한 채널로서 명확하거나 함축적으로 커넥터를 선언해야만 한다. 환경은 컴포넌트가 커넥터를 통해서 다른 컴포넌트와 통신하는 것을 보여준다. 이를 기반으로 본 논문에서는 “컴포넌트 명세란 시스템을 구성하는 컴포넌트와 그 컴포넌트들 사이의 상호작용을 기술한 것”으로 정의하고 컴포넌트 정확한 명세를 위해 <표3>과 같이 컴포넌트 명세 항목을 정의하였다.

<표 3> 컴포넌트 명세 항목 정의

명세항목	정의
컴포넌트	컴포넌트 인터페이스 명세 컴포넌트 기능 명세
커넥터	컴포넌트들의 집합 사이의 상호작용 명세
환경	커넥터를 통해서 결합한 컴포넌트 실제 명세

이러한 명세를 하기위한 기본적인 컴포넌트 명세 분석 정보는 <표4>와 같고, 세부적인 컴포넌트 명세정보를 <표5><표6><표7>과 같이 정의하였다.

<표 4> 기본적인 컴포넌트 분석 정보

명세구분	명세에 필요한 정보
컴포넌트 기술	컴포넌트 이름 컴포넌트 인터페이스
커넥터 기술	내부 연결 규칙 외부 연결 규칙
환경 기술	내부 컴포넌트 집합 내부 인터페이스 집합 내부 연결의 집합

<표 5> 컴포넌트 명세 정보

Component (name, structure, principle_obj, Interface, Internal_obj)	
name	컴포넌트에 대한 유일한 이름
structure	내부 객체들과 그들간의 관계 structure = (Obj, →)
principal_obj	principal_obj ∈ Interface
Interface	Interface ⊆ Obj 인터페이스 객체들의 집합
Internal_obj	Internal_obj = Obj \ Interface

<표 6> 커넥터 명세 정보

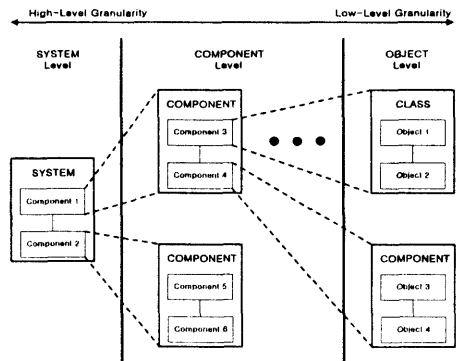
Connectors (LinkRules, con_type)	
LinkRules	커넥터와 커넥터, 커넥터와 컴포넌트 사이의 결합 규칙들의 집합
con_type	컴포넌트 통신, 협상을 캡슐화한 추상화 단위

<표 7> 시스템 구조 명세 정보

Structure (Inner_com, Inner_con, Link)	
Inner_com	구조내에 포함된 컴포넌트들의 집합
Inner_con	구조내에 포함된 커넥터들의 집합
Link	구조내에 포함된 결합들의 집합

3.2. Z를 이용한 컴포넌트 명세 활동

컴포넌트는 구성요소의 세분정도에 따라서 (그림 11)과 같이 분류할 수 있으므로, <표8>과 같이 컴포넌트 타입을 정의하였다[12][13].



(그림 11) 컴포넌트 타입

<표 8> 컴포넌트 타입 정의

종류	정의
System Type	서브시스템(컴포넌트 타입)의 집합으로 이루어진 컴포넌트
Component Type	여러 컴포넌트들의 집합으로 이루어진 컴포넌트
Object Type	최소단위인 object들과 그들 사이의 관계집합으로 이루어진 컴포넌트

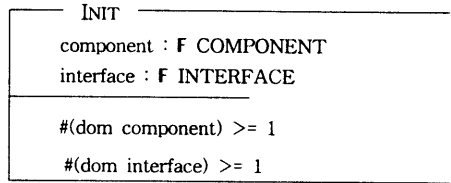
시스템 타입(System Type)에서는 전체 시스템 구성에 대한 기술을 중점적으로 표현하고, 컴포넌트 타입(Component Type)에서는 커넥터 중심으로 컴포넌트를 기술하고, 객체 타입(Object Type)에서는 컴포넌트 내부의 기술을 중점적으로 기술한다.

3.1에서 분석한 컴포넌트 명세 정보를 가지고 제안하는 컴포넌트를 명세하는 활동은 <표9>와 같다. 제안하는 컴포넌트 명세 활동은 컴포넌트 명세, 커넥터 명세, 환경명세의 3개의 활동으로 나뉘어지며, 9개의 구체적인 태스크를 통하여 11개의 산출물을 제시하는 컴포넌트 명세 활동을 제안하였다. 내부 연결 규칙 식별과 외부 연결 규칙 식별 태스크에서는 컴포넌트 명세가 반드시 제공해야 하는 제공하는 인터페이스와 요청하는 인터페이스를 고려하여야 한다.

<표 9> 컴포넌트 명세 활동

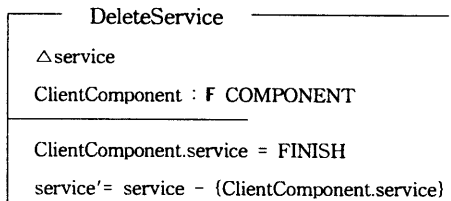
활동	태스크	산출물
컴포넌트 명세	컴포넌트집합 정의	컴포넌트 집합
	인터페이스집합 정의	컴포넌트 인터페이스 집합
	컴포넌트상태스키마정의	컴포넌트 상태스키마
	내부 연결 규칙 식별	내부 연결 규칙 집합
	컴포넌트연산스키마정의	컴포넌트 내부연산스키마
	스키마 조합	컴포넌트 스키마
커넥터 명세	외부 연결 규칙 식별	외부 연결 규칙 집합
	컴포넌트연산스키마정의	컴포넌트 외부연산스키마
환경명세	컴포넌트 타입 식별	시스템타입 컴포넌트명세 컴포넌트타입 컴포넌트명세 객체타입 컴포넌트명세

컴포넌트 집합과 컴포넌트 인터페이스 집합은 Z에서 제공하는 집합 표기법을 이용하여 쉽게 정의할 수 있고, 컴포넌트 상태 스키마 정의 태스크에서는 컴포넌트 각각에 대한 초기 상태 스키마를 정의하여야 한다. 초기 상태스키마의 예는 (그림 12)와 같다. 이 상태스키마에서는 컴포넌트는 1개 이상의 컴포넌트들의 집합으로 하나의 컴포넌트가 구성되며, 컴포넌트는 반드시 1개이상의 인터페이스를 지녀야 한다는 제약조건들을 포함하고 있다.



(그림 12) 초기 상태 스키마의 예

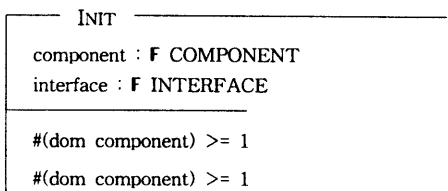
내부 연결 규칙 식별, 컴포넌트 연산스키마 정의 태스크에서는 컴포넌트의 상호작용과 관련된 인터페이스를 연산스키마로 정의하는 것으로 올바른 연산스키마와 예외처리 연산스키마 각각을 정의해야 한다. 외부 연결 규칙 식별, 컴포넌트 연산스키마 정의 태스크에서는 컴포넌트들의 집합 사이의 상호작용을 명세하는 것으로 올바른 연산스키마와 예외처리 연산스키마 각각을 정의해야 한다. (그림 13)은 서비스가 종료되었을 때의 연산스키마의 예이다. 서비스가 종료되었을 때 반드시 요청된 서비스가 제거되어야 한다는 제약조건을 포함하고 있다.

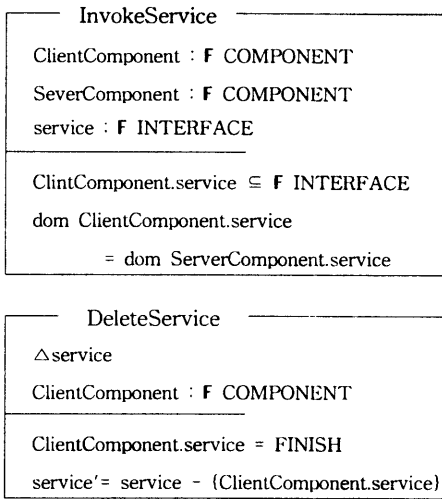


(그림 13) DeleteService 스키마의 예

스키마 조합 태스크에서는 주어진 집합들과 상태 스키마와 연산스키마를 합성하여 컴포넌트 명세를 완성한다. (그림 14)는 집합, 상태스키마와 연산스키마의 합성 예를 보여준다.

[COMPONENT, INTERFACE]
STATE ::= RUNNING | FINISH





(그림 14) 스키마 합성의 예

3.3. 컴포넌트 명세 활동의 비교

컴포넌트기반 소프트웨어 개발 프로세스의 명세활동을 정량적으로 비교하기 어렵기 때문에 각 프로세스의 명세활동들을 명세시기, 명세도구, 품질평가, 그리고 특징 등을 정성적으로 비교하였다<표 10>.

<표 10> 명세 활동의 정성적 비교

프로세스 항목	Catalysis	CBD/e	Fusion	제안활동
명세시기	시스템명세	실 계	분 석	명 세
명세도구	영역모델 형 모델	사용사례	객체모델 인터페이스 모델	Z
품질평가	정 제	정 제	검 사	검 증
기능요구	O	O	O	O
비기능요구	X	X	X	O
특 징	자연어 다이어그램	자연어 다이어그램	자연어 다이어그램	Z

4. 결론 및 향후연구방향

소프트웨어 생산성이 사용자들의 서비스에 대한 요구를 만족시키지 못했고, 소프트웨어 품질이 향상되지 않았으며, 유지보수가 어려운 문제점을 가지고 있다. 이러한 위기를 해결하기 위한 한가지 대응책

으로 최근 컴포넌트기반 소프트웨어공학 혹은 컴포넌트웨어가 등장하였다.

본 논문에서는 컴포넌트 기반 소프트웨어 개발에 있어서 중요한 컴포넌트 명세 정보를 분석하고, 이를 기반으로 분석력과 논리성이 검증된 정형 명세 언어 Z를 이용하여 컴포넌트를 명세하는 활동을 제안하였다.

제안 방법은 3개의 세부활동으로 구성되며, 이 활동들은 9개의 태스크들로 구성되고 11개의 산출물을 구체적으로 제시함으로써 재사용을 위한 컴포넌트 기반 소프트웨어 개발의 기초가 되는 컴포넌트 명세 활동을 제안하였다.

본 논문에서 제안한 방법은 세가지 기대효과가 있다. 첫째, 제안한 방법에서 이용한 정형 명세 언어 Z는 표준 집합론에 기반하는 정형표기법이고 시스템의 특성을 표현하기 위해서 양식화된 수학적인 표기를 사용한다. 특히 스키마를 사용하여 구조화할 수 있다. 이러한 Z 스키마를 이용하여 컴포넌트 기반 소프트웨어를 개발에 있어서 중요한 명세단계의 구체적인 활동을 제시함으로써 컴포넌트 개발의 생산성 향상에 기인하고, 또한 이미 검증된 정형 명세 언어를 이용함으로써 개발초기에 명세의 정확성을 검증하여 품질 높은 컴포넌트 개발을 보증할 것이다. 둘째, 제안한 방법에서 컴포넌트 명세에 필요한 정보를 분석함으로써 다른 컴포넌트 모델들에서 제공하고 있는 부족한 컴포넌트 정보의 결점을 보완할 수 있다. 셋째, 구체적인 컴포넌트 명세 활동을 정의함으로써 컴포넌트 기반 소프트웨어 개발에 있어서 중요한 컴포넌트 명세활동을 진행할 수 있을 것이다.

그러나, 구체적인 컴포넌트 명세활동이 정의되었다 하더라도 이를 지원하는 도구가 없이는 전문가도 명세활동을 하기 힘들 것이다. 또한 컴포넌트 기반 소프트웨어 개발 전 단계 프로세스에 대한 정형화연구도 필요하다.

참 고 문 헌

[1] 권오천(2000). CBD 방법론 및 핵심기술. SETC 2000 튜토리얼

- [2] 김수동(1999). 컴포넌트 개발 방법론의 구성요소 및 기술동향. SETC'99 튜토리얼
- [3] D. D. Souza, A. C. W. Catalysis:Next Generation Component-based Development from Object Frameworks. http://www.cai.com/products/platinum/wp/crc/wp_objfr.htm
- [4] K.Amako. Fusion Method: Process of Analysis. <http://arkhp1.kek.jp/managers/compu.../fusionBookProcessAnalysis.htm>
- [5] 우치수, 김갑수, 이명재(1998). 소프트웨어공학. 홍릉과학출판사. pp. 163-216.
- [6] A. Harry(1996). Formal Methods Fact File:VDM and Z. John Willey & Sons. pp. 173-258.
- [7] Microsoft, MSDN, <http://msdn.microsoft.com/library/techart/COMAddins.htm>
- [8] Sun Microsystems(1999). Enterprise Java Beans Spec. 1.1. Sun
- [9] R. Orfali, D. Harkey(1998), Client/Server Programming with Java and CORBA. Jon Wiley
- [10] J. Q. Ning(1999). A Component Model Proposal. ICSE'99 CBSE Workshop
- [11] N. Medvidovic(1996). A classification and comparison framework for software architecture description languages. UCI-ICS-97-02, Department of Information and Computer Science, Univ. of California, Irvine
- [12] 장종표, 김병기(2000). Z를 이용한 컴포넌트 명세 전략 및 방법. 한국정보처리학회 춘계학술대회.
- [13] 장종표, 문성준, 정대영, 이상준, 김병기(2000), Z를 이용한 컴포넌트 상호작용 명세. SETC 2000

장 종 표



1992 전남대학교 전산통계학과 (이학사)
1997 전남대학교 전산통계학과 (이학석사)

1997~현재 전남대학교 전산학과 박사과정
관심분야: SE, Formal Methods, CBSE, SPICE
E-Mail: jpjang@chonnam.chonnam.ac.kr

이 상 준



1991 전남대학교 전산통계학과 (이학사)
1993 전남대학교 전산통계학과 (이학석사)

1999 전남대학교 전산통계학과 (이학박사)
1995~현재 서남대학교 컴퓨터영상정보통신학부 조교수
관심분야: 소프트웨어공학, 분산 객체 시스템, 소프트웨어 품질평가
E-Mail: sjlee@tiger.seonam.ac.kr

김 병 기



1978 전남대학교 수학과 (이학사)
1980 전남대학교 수학과 (이학석사)

2000 전북대학교 수학과 (이학박사)
1981~현재 전남대학교 컴퓨터 정보학부 교수
관심분야: SE, CBSE, 객체지향 시스템
E-Mail: bgkim@chonnam.chonnam.ac.kr