논문 2009-3-11

컴포넌트 모델링을 위한 객체의 정보관계 추상화 방법

Information Relation Abstraction Method of Objects for Component Modeling

임명재*, 이기영**, 권영만***, 강정진****

Myung-Jae Lim, Ki-Young Lee, Young-Man Kwon, Jeong-Jin Kang

요 약 본 논문에서는 객체지향 설계 정보를 이용한 컴포넌트 기반 모델링 방법을 제안한다. 클러스터링, 캡슐화, 상속성 개념과 같은 재사용성과 확장성을 향상시키는 객체지향 기술을 기반으로 기존에 개발된 객체 정보를 이용하여 추상화함으로서 정보간 관계를 단순화, 명세화 한다. 또한 정보관계의 이해를 지원할 수 있도록 정보관계 추상화 방법을 제시하고, 특정 도메인에 관계없이 시스템 레벨에서 지원하는 공통적인 서비스들을 재사용 단위로 가공하여 시스템 서비스 컴포넌트로 구분하여 모델링 할 수 있다. 이를 통해 재사용성과 확장성을 실현할 수 있으며 개발기간 단축과 품질 향상을 이룰 수 있다.

Abstract In this paper, we propose component modeling method using object-oriented design information. it will be supplied to simplify and specify the relationship between informations that to be developed technology based on clustering, encapsulation and inheritance concepts. Also, we propose abstraction method, it will be support to understanding information relation, and it can modeling on system level without particular domain through dividing common service by reuse unit. Thus It is possible reusability and scalability by this concept, and shorten development period and enhance quality.

Key Words : CBD, Component modeling

Ⅰ. 서 론

현재 소개된 많은 컴포넌트 기술 표준들은 객체지향 방법을 기반으로 하고 있으며, 컴포넌트 기술의 핵심 개념들이 객체지향에서 비롯된 개념들로 구성되어 있다. 따라서 컴포넌트를 개발하는 데 있어서 객체지향 기술을 바탕으로 하는 것이 효과적일 수 있다.

컴포넌트 개발방법은 소프트웨어개발과 유지보수 비용을 절감하고 개발 생산성을 향상시킬 수 있는 효과적인 방법으로 여러 범주에서 적용되고 있다[1]. 특히 컴포넌트 개념은 객체지향의 정보은닉 개념을 기반으로 하고 있다. 컴포넌트 내의 정보는 외부에 나타내지 않는다. 단지 외부에서 접근할 수 있도록 컴포넌트가 제공하는 서비스를 정의한 인터페이스만을 제공한다. 즉 인터페이스가 컴포넌트에 접근하기 위한 유일한 통로가 된다. 따라서 컴포넌트 사용자는 인터페이스 정보만을 가지고 해당하는 용도에 적용하여 사용하면 된다. 그러나 컴포넌트 개발자 입장에서 컴포넌트의 설계 및 모델링 정보를 알고 있어야 한다[2]. 따라서 본 논문에서는 클러스터링, 캡슐화, 상속성 개념 등을 기반으로 재사용성과 확장성을 향상시키는 객체지향 기술을 기반으로 기존에 개발된 객체 정보를 이용하여 추상화함으로서 정보간 관계를 단순화하고 명세화하여 이를 기반으로 컴포넌트를 모델링 할 수 있는 방법을 제안하였다.

*종신회원, 을지대학교 의료산업학부 전산전공
**정회원, 을지대학교 의료산업학부 전산전공(교신저자)
***정회원, 을지대학교 의료산업학부 전산전공
****종신회원, 동서울대학 정보통신과
접수일자 2009.04.15, 수정완료.2009.06.01

본 논문의 구성은 다음과 같다. Ⅱ장에서는 관련 연구를 설명하고, Ⅲ장에서는 정보관계 추상화 표현 방법을 제시하며, Ⅳ장에서 관련방법의 비교평가, Ⅴ장에서 결론을 기술한다.

II. 관련연구

1. 컴포넌트 분류

컴포넌트는 바라보는 시각 또는 관점에 따라 다양한 형태로 표현된다. 또한 컴포넌트에 대한 개념들을 혼돈하는 이유 중의 하나는 한쪽 측면에서만 컴포넌트를 정의함으로써 비롯된다고 볼 수 있다.

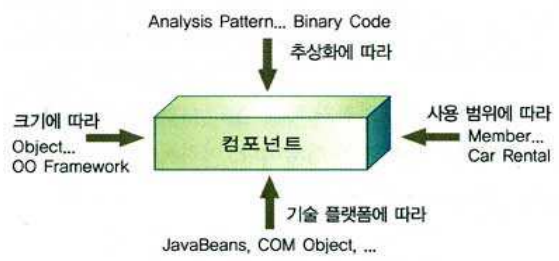


그림 1. 컴포넌트 분류
Fig. 1 Classification Component

모델링 방법에 의하면 요구사항을 분석하기 위하여 서로 다른 관점인 정적인 측면, 동적인 측면 그리고 기능적인 측면에서 분석해야 한다[3].

2. 모델링 방법분류

- 정적 모델링 : 시스템에서 요구되는 객체를 찾아내 객체들의 특성과 객체간 관계를 규명하는 것으로 UML의 클래스 다이어그램 및 컴포넌트 다이어그램으로 표현한다 [4].
- 동적 모델링 : 정적 모델링에서 규명된 객체들의 행위화 상태를 포함하는 라이프사이클을 보여주는 단계로 UML의 시퀀스 다이어그램 및 상태 다이어그램을 표현한다[6].
- 기능모델링 : 각 객체의 형태 변화에서 새로운 상태로 들어갈 때 수행되는 동작들을 기술하는 데 사용하며 UML의 유즈케이스 다이어그램과 활동 다이어그램으로 표현한다[6].
- 컴포넌트 모델링 : 컴포넌트의 생성, 서비스, 소멸의 전 단계를 통제하는 컨테이너안에서 홈 객체에 의해 객체들이 생성되고 컨테이너에서 관리되며, 각각의 컴포넌트들은 인터페이스만을 통해서 서로 호출한다[7].

3. 정보관계 클러스터링

합성 오퍼레이션과 서브시스템 식별을 이용하여 서브시스템과 상속 관계는 초기 호출 그래프상에 표현된다. 이 시스템에 사용된 클러스터링 방법은 프로그램의 중요 엔트리 포인트를 찾는 것이다[8]. 그리고 이 노드로부터 도달 가능한 모든 루틴의 의존도 트리를 생성하고 중심 컴포넌트를 식별한다.

III. 정보관계 추상화 표현방법

1. 연구 목적

각 객체간 정보관계 추상화을 통해서 프로그램의 재사용과 유지보수를 지원하도록 원시코드를 클래스 표현으로 재구성한다[9]. 또한 객체지향 프로그램의 동적인 면의 중요성을 인식하고 이를 이해하기 위한 방법으로 각 객체간 호출관계와 객체 내부의 정보관계의 흐름을 보여주는 정보관계 흐름도(Event Flow Diagram)를 재구성한다. 정보관계 흐름도에서 내부메소드간 연관성과 객체간 정보관계 추상화 방법을 적용하여 표현한다[10].

2. 정보관계 추상화

분석과 식별의 대상이 되는 객체지향 시스템의 동적 요소를 정리하면 메소드, 내부메소드, 인터페이스 메소드, 정보관계 및 메시지이다.

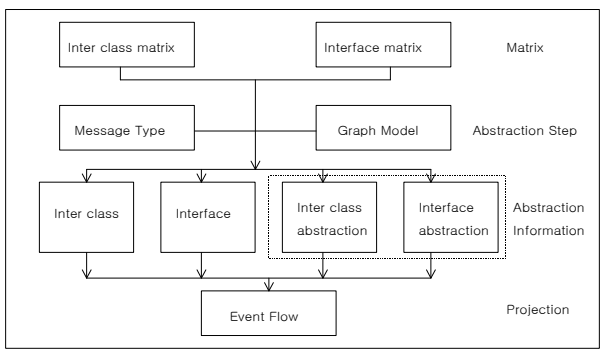


그림 1. 정보관계 추상화 모델
Fig. 1 Abstraction model of Information

또 다른 요소인 인터페이스 메소드는 다른 클래스의 메소드를 호출하거나 호출되는 메소드이며, 정보관계나 메시지는 메소드들간 호출관계와 한 클래스 내에서나 클래스들간 호출관계에 대응하는 요소이다. 시스템의 동적 정보는 본 논문에서 제안하는 그래프 모델의 적합여부에 의해 추출된다. 정보관계 추상화를 개념적으로 살펴보면 <그림 1>과 같다. 원시코드를 입력으로 받아들이면 이를 분석한 정보가 메트릭스에 저장된다. 이때 메소드의 호출관계와 메시지의 타입을 분류하고 이를 메트릭스에 저장한다. 처리대상이 되는 요소는 내부메소드, 인터페이스 메소드가 있으며 이들 값은 해당 메트릭스에 저장된다.

프로그램 분석시 메소드간 또는 클래스간 메시지 전달 관계를 이해하고 이들 사이의 복잡도를 줄이기 위하여 정보관계를 추상화시킨다. 따라서 메트릭스에 저장된 정보를 본 논문에서 정의한 그래프 모델 방법에 의해서 추출하여 이를 저장한다. 추출하고자 하는 그래프 모델은 평행관계, 연쇄관계 그리고 조건관계, 순차관계 등이다.

3. 메트릭스의 정의

원시 코드에서 분석된 정보는 내부메소드 메트릭스와 인터페이스 메소드 메트릭스로 분류되어 저장된다. 메트릭스는 행과 열을 이용하여 호출하는 메소드와 호출되는 메소드의 관계를 효율적으로 나타낼 수 있으므로 메트릭스를 사용하여 호출관계를 저장한다.

```
Let matrix M = a [i, j] be n * m matrix
    Matrix Definition
      if a [i, j] ≠ 0 then
        [0, j] is callee of [i, 0]
        [i, 0] is caller of [0, j]
```

위와 같은 내용은 다음 관계를 유도한다.

```
i = {1, . , n}, K = { i, n  | a [i, n]  = true } 일 때
  N(K) is outdegree (callers) of a [i, 0]
j= {1, . , m}, K = { 1, j  | a [1, j]  = true}  일 때
  N(K) is indegree (callees) of a [0, j]
a[i, j] = true ∧ a [j, i]  = true 일때
   a [i, j]  ≠ a [j, i]
```

즉 열은 호출된 메소드를 나타내며 행은 호출하는 메소드를 나타낸다. 예를 들면 a→b는 메소드 a 를 호출한 다음 b 메소드를 호출하는 관계를 나타낸다고 할 때 <표 1>에서 나타난 것과 같이 a→b→c 의 호출관계를 나타낸다.. 그리고 a→b≠b→a 이므로 비대칭적이다. 메트릭스의 가로 열은 호출하는 함수를 나타내며, 세로 열은 자신을 호출하는 메소드를 나타낸다.

표 1. 내부, 인터페이스 메소드
Table.1 Inter, Interface method

	a	b	c	d
a		1		
b			1	
c				
d				

	A:a	B:b	C:c	D:d
A:a		1		
B:b				
C:c		1		
D:d			1	

<표 2>는 정보관계 추상화 표현시 사용될 타입을 정의하였다. 정보관계 타입은 메소드의 호출순서와 함께 메트릭스에 저장됨으로서 그래프 모델의 추출시 조건으로 사용된다. 메트릭스의 cell의 구조는 <그림 2>와 같다. 호출순서는 메소드에서 호출되는 순서를 나타내고 정보관계의 타입은 i, r은 true와 false로 나타낸다.

표. 2 타입 정의
Table. 2 Type definition

표시형식	내 용
i	t : 조건문에 의한 분기 메소드 f : 조건문에 의한 비분기 메소드
smt	s : 정보관계의 시점 메소드 m : 정보관계의 중간 경로 메소드 t : 정보관계의 종점 메소드
r	t : 함수 반환값이 있는 메소드 f : 함수 반환값이 없는 메소드

호출순서		정보관계 타입		
seq#	if_else	i	r	smt

그림 2. cell의 구성
Fig. 2 Structure of cell

4. 추상화 정보의 정의

메트릭스에 저장된 정보에서 추상화를 하기 위해서 그래프 모델을 추출한다. 이때 정점은 내부메소드이거나 클래스이며 간선은 정보관계를 나타낸다. 간선의 방향은 호출관계를 나타낸다. <그림 3>은 추출되는 그래프 모델을 나타낸다. 생성된 정보를 내부메소드 메트릭스로

구성하고 이를 기반으로 인터페이스 메소드와 내부메소드의 추상화 정보, 인터페이스 메소드의 추상화 정보를 생성할 경우 메트릭스를 구성하는 규칙을 정형화하여 표현한다.

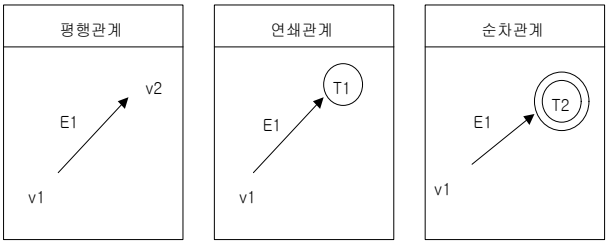


그림 3. 그래프 모델의 추상화
Fig. 3. Abstraction of Graph Model

표 3. 추상화 정보의 정의
Table. 3 Definition of Abstract Information

	내부 메소드	인터페이스 메소드
평행관계	source, destination, parallel	source class name::method destination class name::method parallel
연쇄관계	source, destination, chain	start class name::method, stop class name::method, chain
순차관계	first, last, serial	first class name :: method. last class name :: method, serial

평행관계 즉, 정점 v1 과 정점 v2 사이에 두개 이상의 간선이 존재한다면 <그림 4>에서와 같이 간선 e1과 e2는 E1으로 표현된다. 연쇄관계, 정점 v1과 v2, v3가 연쇄관계에 있다면 v1->v3로의 간선을 e2라 하고 T1을 v2 + v3로 추상화 하면 e1과 e2는 E1이 된다.

Element	Definition
e_1	v_1 -> v_2
e_2	v_1' -> v_2'
E1	e_1 + e_2

Element	Definition
e_1	v_1 -> v_2
e_2	v_2 -> v_3
T1	v_2 + v_3
E1	v_1 -> T1
E1	e_1 + e_2 + e_3

그림 4. 요소관계 정의
Fig. 4 Element relation Definition

순차관계, 즉 간선 e1, e2, e3가 순차관계에 있을 때 v2, v3, v4를 T1으로 추상화 되면 e1, e2, e3는 E1이 된다. 내부 클래스 메소드를 기반으로 하여 인터페이스 메소드 메트릭스를 생성한다. 내부 클래스 메소드 메트릭스의 정보중에서 정보관계의 시작점을 찾는다. 왼쪽 행에 있는 메소드가 호출하는 메소드를 알기 위해 열을 검색한다. 수평방향의 메소드는 가장 왼쪽의 메소드가 호출하는 메소드이며 수직 방향의 메소드는 가장 상위의 메소드를 호출하는 메소드이다. 각 cell을 검사하여 정보가 들어 있다면 해당 정보를 호출 순서에 따라 연결 리스트에 저장하고, 동시에 정보의 caller와 callee에 관한 정보도 함께 저장한다. 연결 리스트에 저장된 정보의 순서대로 추적하면서 정보관계 타입과 caller, callee를 참조하여 그래프 모델에서 추출한 정의의 만족 여부를 판단하고 이를 기반으로 추상화 정보를 생성한다.

5. 컴포넌트 모델링

전 단계에서의 추상화 정보를 기반으로 컴포넌트 모델링을 한다. 컴포넌트 모델링을 위해 내부메소드 메트릭스를 기반으로 인터페이스 메소드 메트릭스를 생성한다. 다음은 <그림 5>의 원시코드를 입력으로 내부메소드를 통해서 다음 정보관계를 추상화시키는 과정이다. 결과의 내용은 <그림 6>과 같이 나타낼 수 있다.

```
_
class FileAction : Virtual public StrAction
virtual public Border{
        LineAction *Crt ;
        FileAction *next ;
        int cx, cy ;
public :
        void Dpl (LineAction *p, int) ;
        void Dpl_Ln( ) ;
        void Dpl_Sc( ) ;
}
void FileAct::Delact
{
int length1, length2 ;
int i ;
   if(currentcol < currentline → length)
        save = 0 ;
        Crt→del(currentcol) ;
        Dpl(currentline, cy) ;
        Dpl_Ln( ) ;
   else
```

```
        Dpl_Sc( ) ;
}
void Dpl_Sc( ) ;
{
   Dpl(currentline, cy) ;
   Dpl_Ln( ) ;
   Space( ) ;
   Move To( );
}
```

그림 5. 원시 코드 리스트
Fig. 5 Source code List

	DelAct		Dpl		Dpl_Ln		Dpl_Sc		Space		MoveTo	
DelAct			112	it	113	it	2El	im				
Dpl_Sc			1	it	4	it			2	it	3	it
Dpl_Ln												

예) FileAction에 대한 내부메소드 메트릭스

분석된 정보를 각각의 내부메소드와 인터페이스 메트릭스에 추상화하여 저장한다.

	FileAct::DelAct		LineAct::del	
FileAct			il1	itf

예)인터페이스 메소드 메트릭스

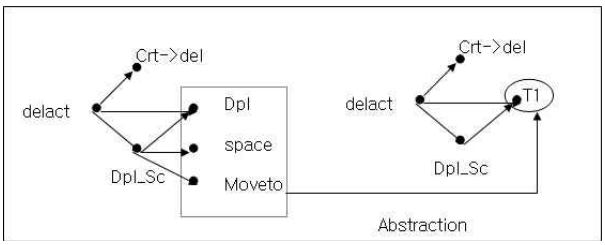


그림 6. 추상화 모델 생성
Fig. 6 Create Abstraction Model

T1	FileAct::Dpl	FileAct::Moveto	serial	Dpl

예)추상화 저장정보

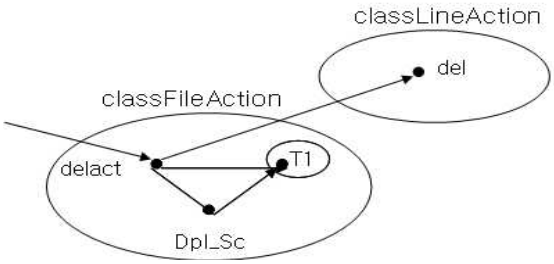


그림 7. 컴포넌트 모델링을 위한 추상화결과
Fig. 7 Abstraction result for Component modeling

다음 이들 두 가지 정보를 추상화의 입력으로 사용하며, <그림 8>와 같이 이후에 컴포넌트 추상화 분류에 적용하여 정보를 생성한다. 이때 주로 사용되는 데이타의 구조는 메소드, 정보관계 타입, 호출 순서, callee의 포인터를 저장할 포인터 배열을 갖는 메소드 리스트와 추상화 정보를 저장하는 추상화 정보 리스트 등이 있다.

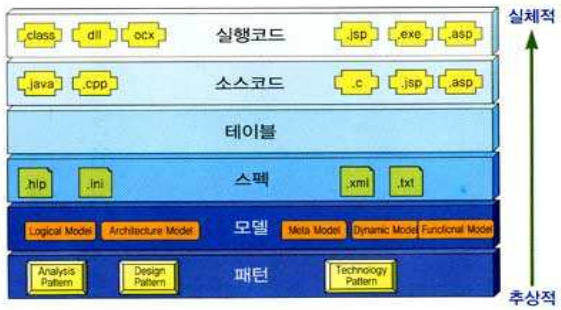


그림 8. 컴포넌트 추상화
Fig. 8 Component Abstraction

개발된 컴포넌트가 실행될 때 해당 컴포넌트의 인스턴스(Instance)인 컴포넌트 객체가 수행되며, 이때 컴포넌트 내부에는 하나 이상의 클래스에 대응하는 하나 이상의 객체들이 수행된다. 사용자 인터페이스 측면의 개발을 위해 재사용되는 컴포넌트는 사용자 인터페이스 컴포넌트라 하며, 특정 도메인에 상관없이 시스템 레벨에서 지원하는 공통적인 서비스들을 재사용 단위로 가공하여 만든 시스템 서비스 컴포넌트로 구분하여 모델링 할 수 있다.

IV. 비교 평가

평가의 대상으로는 추상화 방법, 컴포넌트 지원부분, 모델링 기술을 대상으로 하였으며, 다양한 비교 대상중 본 논문과 유사성을 갖는 Wim 방법과 phillip의 함수 추상화관련 부분에 대하여 비교하였다. 추상화 방법에 있어서 Wim의 방법은 서로 상호작용이 많은 클래스들을 클러스터링 시킴으로서 객체지향 프로그램을 몇 개의 클래스 모듈로 분할은 가능하지만 시스템의 동적인 면의 이해도를 저하시킬 수 있다. Phillip 방법은 프로그램 분할과 패턴매칭 방식을 통한 추상화 방법을 제시하고 있으나 분할영역문제, 유사패턴 선택등의 문제를 갖는다. 본 논문은 추상화 방법에서 모듈화의 기능은 Wim의 방

법에 비하여 미흡하다고 볼 수 있으나, 정보관계의 흐름을 파악할 수 있으며, 이로 인하여 클래스 내부에서의 정보관계의 흐름과 시스템을 이루고 있는 클래스들의 상호작용을 이해할 수 있다. 모델링은 비교대상 모든 방법이 UML을 공통적으로 지원하면, 본 논문에서는 pert/cpm을 지원할 수 있다.

표 4. 비교평가
Table. 4 Evaluation table

	Wim	Phillip	본 논문
동적측면	function matrix 이용	-	method, class간 관계표현
추상화방법	class -clustering	program -slicing, pattern matching	class event information relation
컴포넌트명세	some stereotype cluster -matrix	frame ,pattern	method event interface abstraction, document view
컴포넌트지원	-	business model type	user define, partial define
모델링기술	uml	uml based tool	uml, pert/cpm

V. 결론

본 논문에서는 객체지향 패러다임에 기반하여 생성된 산출물을 이용하여 컴포넌트 개발환경에 이용 가능한 컴포넌트 모델생성을 목적으로 하고 있다. 객체지향 프로그램의 특징인 객체간 상호작용을 표현하고 객체내부의 정보관계의 흐름을 나타내는 정보관계 흐름도를 재구성하였으며, 이때 정보관계 추상화를 적용하였다. 이를 통해 각 클래스들의 기능과 클래스들의 상호작용을 파악하고자 하였다. 또한 객체간 메시지의 교환관계를 시각화하여 각 객체의 역할을 분명하게 알 수 있으며, 시스템의 동적인 면의 복잡도를 시각화시켜 사용자의 이해를 용이하게 할 수 있다. 또한 특정 도메인에 상관없이 시스템 레벨에서 지원하는 공통적인 서비스들을 재사용 단위로 가공하여 만든 시스템 서비스 컴포넌트로 구분하여 모델링 할 수 있다.

향후 연구과제로는 컴포넌트 기반 설계의 명세를 기반으로 정형적인 모델링 방법의 연구가 필요한 것으로 간주되며, 이를 통해 모델 검증이 가능할 것으로 예상된다.

참고 문헌

[1] Metz, Pierre, O'Brien, John and Weber, Wolfgang, "Specifying Use Case Interaction: Types of Alternative Courses", Journal of Object Technology, Vol. 2, No. 2, Mar 2003

[2] Rumbaugh, James, Jacobson, Ivar, and Booch, Grady, "The Unified Modeling Language: Reference Manual", 2nd Ed., 2005

[3] Desmond Francis D'Souza, Alan Cameron Wills. Objects components and frameworks with UML, Addison Wesley Longman, Inc., 1999

[4] G. A. Rassati, A.M.ASCE, Component Modeling of Partially Restrained Composite Joints under Cyclic and Dynamic Loading, Journal of Structural Engineering, Vol. 130, No. 2, February 2004, pp. 343-351

[5] M, Thomas.C.Carson and J.M Hellerstein. "Creating a Customized Access Method for Bloworld". Proc. 16th International Conference on Data Engineering. pp.82-92. 2000

[6] Braganca, Alexandre and Machado, Ricardo J., "Extending UML 2.0 Metamodel for Complementary Usages of the <<extend>> Relationship within Use Case Variability Specification", SPLC'06, 2006

[7] McGibbon, B., "Status of CBSE in Europe." Component-Based Software Engineering, Addison-Wesley, 2001.

[8] Wim De Pauw, Sophia Krasikov, John F. Morar: Execution patterns for visualizing web services. SOFTVIS 2006: 37-45

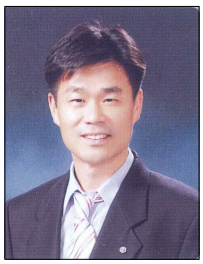
[9] Nimeta. View-based Software Architecture Reconstruction, Claudio Riva. Ph. D. Dissertation, Vienna University of Technology, Vienna, Austria, October 2004. http://www.clody.org/research/riva-dissertation-final-double.pdf

[10] Shimba. An Environment for Reverse Engineering Java Software Systems, Tarja Systä, Kai Koskimies, and Hausi Muller.

Software Practice and Experience 31(4), 2001, pp. 371-394.

저자 소개

임 명 재(종신회원)



- 1998년 2월 중앙대학교 컴퓨터공학과 공학박사
- 1992년 3월 ~ 현재 을지대학교 의료산업학부 전산전공 교수

<관심분야> : S/W 공학, CBD 방법론, 정보검색, HCI 등

이 기 영(정회원)

- 1984년 숭실대학교 전자계산학과 학사 졸업.
- 1988년 건국대학교 대학원 컴퓨터 공학과 석사 졸업.
- 2005년 건국대학교 대학원 컴퓨터 공학과 박사 졸업
- 1984년~1991년 한국해양연구원 연구원
- 1996년~1998년 한국컴퓨터정보학회 이사 및 서울동부지회장
- 1991년~현재 을지대학교 의료산업학부 부교수

<주관심분야> 공간 데이타베이스, GIS, LBS, USN, 텔레매틱스 등

권 영 만(정회원)

- 1983년 2월 광운공과대학 전자공학과 졸업
- 1985년 2월 한국과학기술원 전기및전자공학과 석사
- 1998년 8월 한국과학기술원 정보및통신공학과 박사수료
- 2007년 2월 광운대학교 전자공학과 박사
- 1993년 3월~현재 을지대학교 의료산업학부 교수

<관심분야> 영상처리, 머신비전, 운영체제

강 정 진(종신회원)

- 9권 2호 참조