한국항행학회 논문지 제 15 권 제 6 호 2011년 12월

검증규칙과 프레임을 이용한 객체지향 다이아그램 검증 시스템의 설계

A Design of Object-Oriented Diagram Verifying System Using a Set of Verifying Rules and Frames

김진수*

Jin-Soo Kim*

요 약

객체지향 설계에 있어서 그래픽 표현의 대표라고 할 수 있는 UML 다이아그램의 일관성과 완전성을 검증하기 위하여, 먼저 UML 다이아그램들을 분석하고 분석된 다이아그램에 검증 규칙들을 적용한다. 본 논문에서는 다이아그램을 작성하는 능력과 다이아그램의 일관성과 완전성을 검사할 수 있는 능력을 모두 갖춘 효과적인 검증 시스템을 설계하였다. 검증 시스템은 내부적인 다이아그램 정보를 표현하기 위하여 일련의 프레임들을 사용하고 있다.

Abstract

For verifying consistency and completeness of some UML diagrams as a representative of a graphical notations for object-oriented designs, I first give an analysis of some UML diagrams and apply some verifying rules to the UML diagrams. In this paper, I design effective verifying system which possesses both diagramming facility and the consistency and completeness checking capability. The verifying system uses a set of frames to represent internal diagramming information.

Key words : UML Diagram, consistency and completeness, Verifying Rules, Verifying System, Frames

Ⅰ. 서 론

최근에 객체지향 방법이 소프트웨어를 개발하는 탁월한 파라다임이 되고 있고 객체지향 방법이 기존의 개발 방법들에 비해 개발되는 소프트웨어의 품질을 향상시키고 생산성을 증가시킨다는 것은 이미 많이 알려져 있다. 일반적으로 크고 복잡한 소프트웨어 시스템은 커다란 다이아그램의 집합으로 구성되지만 이들 각각의 다이아그램들이 일관성이 있고 완전한가를 알기는 매우 어렵다. 일반적으로 다이아그램들의 일관성과 완전성을 보장하게 만드는 일은 다음과 같은 이유들로 인해 상당히 어려운 일이다. 첫째, 객체지향 설계는 개발자의 논리적 사고의 결과이기 때문에, 크고 복잡한 소프트웨어 시스템을 다른 관점과 다른 추상화 수준에서 개발자의 생각을 일관성이 있고 완전하게 유지한다는 것은 어려운 일이다. 둘째, 크고 복잡한 소프트웨어 시스템을 객체지향 방법으로 설계하는 일은 보통 여러 팀들이 각자 만들어내는

* 건양대학교 컴퓨터학과(Dept. of Computer Science & Engineering, Konyang University)
· 제1저자 (First Author) : 김진수
· 투고일자 : 2011년 11월 14일
· 심사(수정)일자 : 2011년 11월 14일 (수정일자 : 2011년 12월 15일)
· 게재일자 : 2011년 12월 30일

생산품일 경우가 많다. 셋째, 객체지향 설계는 반복적인 작업이기 때문에 설계의 변경이 발생되는 동안 개발자들이 이 모든 변화에 대해 많은 다이아그램들을 완전하게 일관성을 유지하기는 힘들다[1].

이러한 문제를 해결하기 위하여 본 논문에서는 객체지향 개발에서 많이 사용되고 있는 UML 다이아그램들에 대한 일관성과 완전성을 검사하기 위하여 검증 규칙들을 이용하여 다이아그램의 일관성과 완전성을 검증할 수 있는 검증 시스템을 설계하였다.

본 논문은 2장에서 검증 규칙을 설명하고, 3장에서 UML 다이아그램을 사용하여 검증 시스템을 설계하고, 4장에서 프레임을 이용하여 검증 시스템의 내부표현을 보이고 5장에서 결론을 맺는다.

Ⅱ. 검증 규칙

[1]에서는 각각의 다이아그램에 대하여 집합과 함수를 이용하여 정형적으로 명세한 다음 함수들의 의미에 따라 다이아그램의 일관성과 완전성을 보장하기 위한 규칙들을 유도하고 이들 규칙에 정형 명세를 주고 있다. 이들 다이아그램과 규칙들의 정형 명세는 수학적 기호로 표현되며, 다음과 같은 E와 R의 두 튜플로 정의할 수 있다.

UMLOOD = < E, R >
E : 일반화된 엔티티 타입들의 집합
R : 일반화된 엔티티 타입들간의 관련성의 집합

통합된 다이아그램에서 각 일반화된 엔티티 타입은 집합 E의 한 요소이다. 따라서 첫 번째 튜플 E는 다음과 같은 일반화된 엔티티 타입의 집합을 이용하여 정형적으로 정의된다.

E = {C,R,Cao,S,T,Sa,Ta,O,L,Oa,Lm,I,Im}

여기에서 제시된 집합 E의 요소인 엔티티 타입의 집합들은 추상자료형 집합을 사용하여 정형적으로 명세될 수 있다. 다음 표 1.은 집합 E의 각 요소들에 대한 설명이다.

표 1. 집합 E의 요소에 대한 설명
Table 1. Description of Elements of Set E

요소	설 명
C	일반적인 Class 엔티티 타입을 표현하는 구체적인 엔티티 타입들의 집합
R	일반적인 Relationship 엔티티 타입을 표현하는 구체적인 엔티티들의 집합
Cao	일반적인 Class-Attribute-Operation 엔티티 타입을 표현하는 클래스들의 속성과 오퍼레이션들의 집합
S	일반적인 State 엔티티 타입을 표현하는 구체적인 엔티티 타입들의 집합
T	일반적인 Transition 엔티티 타입을 표현하는 상태 전이들의 집합
Sa	일반적인 State-Action 엔티티 타입을 표현하는 상태-행위들의 집합
Ta	일반적인 Transition-Action 엔티티 타입을 표현하는 전이-행위들의 집합
O	일반적인 Object 엔티티 타입을 표현하는 객체들의 집합
L	일반적인 Link 엔티티 타입을 표현하는 객체 링크들의 집합
Oa	일반적인 Object-Attribute 엔티티 타입을 표현하는 객체-속성들의 집합
Lm	일반적인 Link-Message 엔티티 타입을 표현하는 링크-메시지들의 집합
I	일반적인 Interaction 엔티티 타입을 표현하는 객체 상호작용들의 집합
Im	일반적인 Interaction-Message 엔티티 타입을 표현하는 상호작용-메시지들의 집합

UMLOOD의 정형화된 정의의 두번째 투플은 R이다. 투플 R은 다음과 같은 함수들로 정형적으로 정의된다.

R = {RfC[1], RtC[2], TfS[3], TtS[4], LfO[5], LtO[6], IfOi[7], ItOi[8], CaoC[9], SaS[10], TTa[11], OaO[12], LLm[13], IIm[14], SaCao[15], TaCao[16], OaCao[17], LmCao[18], OiO[19], IL[20], ImLm[21], ScC[22], OcC[23], LcR[24]}

집합 R의 각 요소는 다이아그램에서 엔티티 타입들간의 관련성을 설명하는 함수들로 정의된다. 다음 표 2.는 집합 R의 각 요소들에 대한 설명이다.

표 2. 집합 R의 요소에 대한 설명
Table 2. Description of Elements of Set R

요소	설 명
RfC	R → C, Class 엔티티 타입으로부터 Relationship 엔티티 타입으로의 from 일대다 관련성을 표현하는 R 집합에서 C 집합으로의 함수
RtC	R → C, Class 엔티티 타입으로부터 Relationship 엔티티 타입으로의 to 일대다 관련성을 표현하는 R 집합에서 C 집합으로의 함수
TfS	T → S, State 엔티티 타입으로부터 Transition 엔티티 타입으로의 from 일대다 관련성을 표현하는 T 집합에서 S 집합으로의 함수
TtS	T → S, State 엔티티 타입으로부터 Transition 엔티티 타입으로의 to 일대다 관련성을 표현하는 T 집합에서 S 집합으로의 함수
LfO	L → O, Object 엔티티 타입으로부터 Link 엔티티 타입으로의 from 일대다 관련성을 표현하는 L 집합에서 O 집합으로의 함수
LtO	L → O, Object 엔티티 타입으로부터 Link 엔티티 타입으로의 to 일대다 관련성을 표현하는 L 집합에서 O 집합으로의 함수
IfOi	I → Oi, Object(i) 엔티티 타입으로부터 Interaction 엔티티 타입으로의 from 일대다 관련성을 표현하는 I 집합에서 O(i) 집합으로의 함수
ItOi	I → Oi, Object(i) 엔티티 타입으로부터 Interaction 엔티티 타입으로의 to 일대다 관련성을 표현하는 I 집합에서 O(i) 집합으로의 함수
CaoC	Cao → C, Class 엔티티 타입으로부터 Class-Attribute-Operation 엔티티 타입으로의 adorned_by 일대다 관련성을 표현하는 Cao 집합에서 C 집합으로의 함수
SaS	Sa → S, State 엔티티 타입으로부터 State-Action 엔티티 타입으로의 adorned_by 일대다 관련성을 표현하는 Sa 집합에서 S 집합으로의 함수
TTa	T → Ta, Transition-Action 엔티티 타입으로부터 Transition 엔티티 타입으로의 adorned_by 일대다 관련성을 표현하는 T 집합에서 Ta 집합으로의 함수
OaO	Oa → O, Object 엔티티 타입으로부터 Object-Attribute 엔티티 타입으로의 adorned_by 일대다 관련성을 표현하는 Oa 집합에서 O 집합으로의 함수

예를들어, 집합 R의 처음 원소는 RfC이다. RfC는 다음과 같은 함수로 정의된다.

RfC : R → C

여기에서 R은 이 함수의 정의역인 클래스 관련성의 집합이고 C는 이 함수의 치역인 클래스들의 집합이다. 함수 RfC는 클래스 엔티티 타입으로부터 관련성 엔티티 타입으로의 일대다 관련성에 대한 정형화된 정의이다. 집합 R의 다른 원소들도 동일한 방법으로 각각의 함수들로 정의된다[1].

Ⅲ. 검증 시스템의 설계

3-1 검증 시스템의 구성

2장에서 제시한 검증 규칙을 기반으로 다이아그램들간의 일관성과 완전성을 검사하기 위한 검증 시스템은 다음 그림 1.과 같이 크게 4개의 서브시스템으로 구성된다.

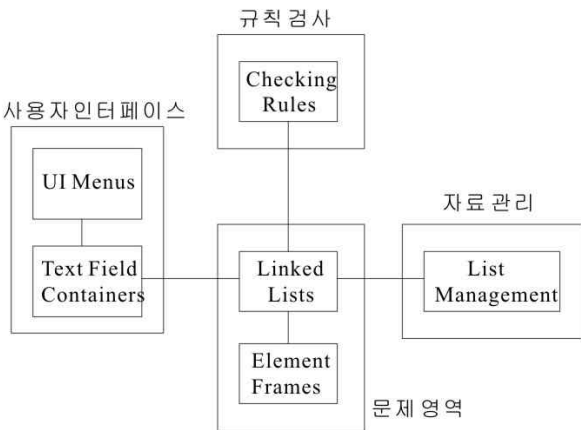

그림 1. 검증 시스템의 구성
Fig. 1 Configuration of Verifying System

먼저 사용자 인터페이스 서브시스템은 Text_ Field_Containers와 UI_Menus로 구성되는데 Text_ Field_Containers는 각 항목이 텍스트 필드를 갖는 텍스트 필드 항목들의 집합이라고 할 수 있고, UI_Menus는 사용자의 입력을 도와주는 기능들을 제공한다. 다음으로 문제영역 서브시스템은 Linked_ Lists와 Element_Frames로 구성되는데 Linked_ Lists는 single-linked-list를 생성하기 위해서 Element_Frames의 서비스를 이용하고 Text_Field_ Containers로부터 받은 정보를 single-linked-list의 요소에 저장한다. 자료관리 서브시스템의 요소인 List_Management는 single-linked-list를 외부 파일로 저장하기도 하고 저장된 파일로부터 다시 생성하기도 한다. 규칙 검사 서브시스템의 요소인 Rule_ Based_Checking은 single-linked-list로부터 자료를 얻기 위해서 Linked_Lists

의 서비스를 이용한다.

Text_Field_Containers는 다음 그림 2.와 같이 네 종류의 텍스트 필드 항목들을 가지고 있으며 프레임의 슬롯에 값을 넣는 것에 따라 종류가 결정된다.

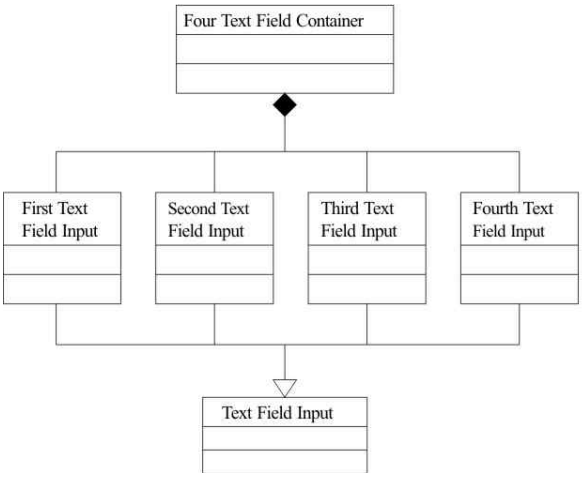

그림 2. Text_Field_Containers에 대한 클래스 다이아그램
Fig. 2. Class Diagram of Text_Field_Containers

사용자 인터페이스 서브시스템의 UI_Menu는 다음 그림 3.에서와 같이 다이아그램 메뉴, 다이아그램 요소 메뉴, 오퍼레이션 메뉴와 같은 클래스들로 구성되어 있다.

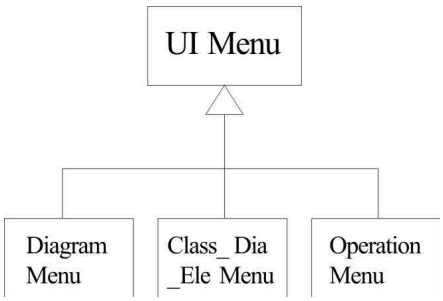

그림 3. UI_Menu에 대한 클래스 다이아그램
Fig. 3. Class Diagram of UI_Menu

문제 영역 서브시스템의 Linked_Lists는 다음 그림 4.와 같이 SL_List 클래스와 SL_List_Element 클래스로 구성되어 있다.

그림 4. Linked_Lists에 대한 클래스 다이아그램
Fig. 4. Class Diagram of Linked_Lists

문제 영역 서브시스템의 Element_Frame은 다음 그림 5.와 같이 네 개의 클래스로 구성되어 있다.

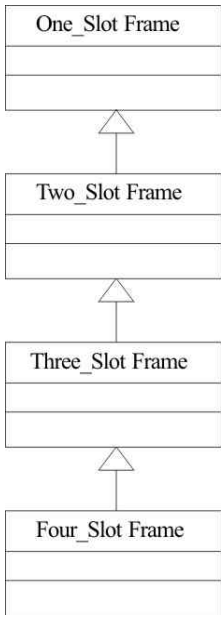

그림 5. Element_Frame에 대한 클래스 다이아그램
Fig. 5. Class Diagram of Element_Frame

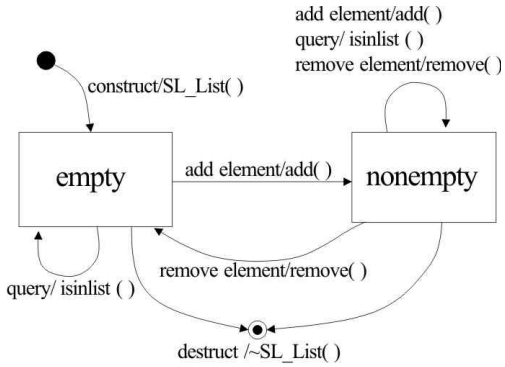

그림 6. SL_List 클래스에 대한 상태 다이아그램
Fig. 6. State Diagram of SL_List Class

SL_List 클래스는 그림 6.과 같이 empty와 non-empty의 두 상태를 가지는데 클래스 생성자 함수인 SL_List()가 수행되면 비어있는 single-linked-list가 생성되고 그 상태는 empty 상태가 된다.

먼저 다이아그램 메뉴에서 클래스 다이아그램이 선택되고, 다이아그램 요소 메뉴에서 클래스 다이아그램의 요소인 relationship이 선택되고, 오퍼레이션 메뉴에서 오퍼레이션 add()가 선택되면, 네 개의 슬롯을 갖는 하나의 프레임이 생성된다. 먼저 사용자로부터 관련성의 이름을 입력받아 프레임의 처음 슬롯에

넣고 다음으로 관련성의 subset 이름을 받아 프레임의 두 번째 슬롯에 넣는다. 관련성이 시작되는 클래스의 이름을 입력받아 관련성 프레임의 세 번째 슬롯에 넣고 마지막으로 관련성이 종료되는 클래스의 이름을 입력받아 관련성 프레임의 네 번째 슬롯에 넣으면 이 프레임은 single-linked-list인 ClassList에 하나의 요소로 추가된다. 다음 그림 7.은 이러한 시나리오를 보여주는 객체 다이아그램이다.

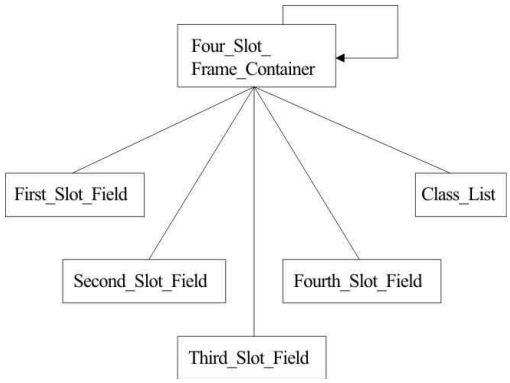

그림 7. 관련성을 추가하는 객체 다이아그램
Fig. 7. Object Diagram appending Relationship

ClassList 리스트에 하나의 관련성을 추가하는 시나리오는 다음 그림 8.과 같이 순차 다이아그램으로도 나타낼 수 있다.

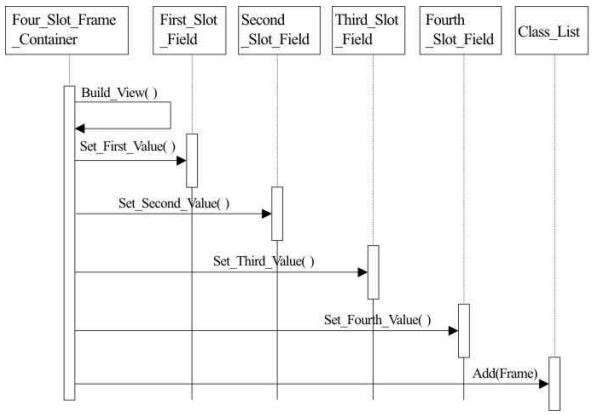

그림 8. 관련성을 추가하는 순차 다이아그램
Fig. 8. Sequential Diagram appending Relationship

3-2 검증 시스템의 검증 과정

다음 그림 9.는 검증 시스템의 자료흐름도를 보여주고 있다.

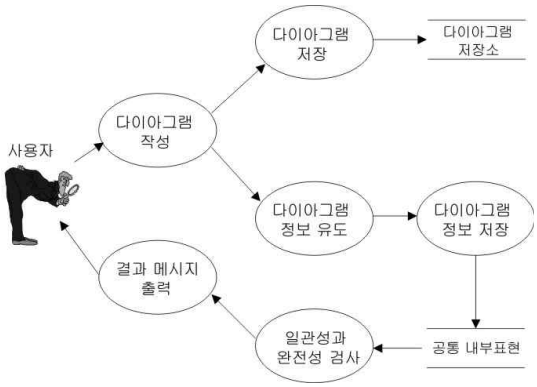

그림 9. 검증 시스템의 자료흐름도
Fig. 9. Data Flow Diagram of Verifying System

본 논문에서 설계한 검증 시스템은 먼저 사용자가 다이아그램을 선택할 수 있는 다이아그램 메뉴, 다이아그램 요소를 선택할 수 있는 다이아그램 요소 메뉴, 이러한 다이아그램 요소의 오퍼레이션을 선택할 수 있는 메뉴를 제공한다. 사용자가 다이아그램, 다이아그램 요소, 오퍼레이션을 선택하면 검증 시스템은 이러한 다이아그램의 요소에 대한 필요한 정보를 가져와서 공통 내부표현으로 저장할 수 있도록 한다. 이렇게 저장된 공통 내부표현을 이용하여 일관성과 완전성에 대한 검사를 수행하며 수행된 결과는 사용자에게 메시지로 제공될 수 있다.

Ⅳ. 프레임을 이용한 검증 시스템의 내부표현

일반적으로 프레임은 값을 넣을 수 있는 여러 개의 슬롯으로 구성되는데 타입의 값들을 나타내기 위하여 사용되기도 한다. 예를들어 2장에서 제시된 Relationship 타입의 각 값은 관련성들간에 구별될 수 있는 유일한 이름을 갖고 있는 하나의 관련성이라고 할 수 있다. 따라서 Relationship 타입에 대한 프레임은 Relationship-Name이라는 슬롯을 가지는데 이 슬롯의 값은 Relationship 타입에서 유일하다. 관련성이 어떠한 부분 집합을 가지는가를 나타내기 위하여 Relationship 타입에 대한 프레임은 SubsetName 이라는 슬롯을 가지게 되고 이 슬롯에 부분 집합 이름이 들어가면 이 구체적인 관련성은 해당 부분 집합의 원소가 된다는 것을 나타낸다.

타입은 함수의 정의역이나 치역의 타입을 나타낼

수 있다. 예를들어, 2장에서 Relationship 타입이 RfC1 함수와 RtC^2 함수의 정의역 타입으로 사용되었기 때문에 Relationship 타입에 대한 프레임은 추가적으로 FromClassName과 ToClassName의 두 슬롯을 가지게 된다. FromClassName 슬롯은 Class 타입이 RfC^1 함수의 치역이므로 관련성이 시작되는 클래스의 이름이 들어가고 ToClassName 슬롯에는 RtC2 함수의 치역이므로 관련성이 종료되는 클래스의 이름이 들어가게 된다.

이와 같이 Relationship 타입에 대한 프레임은 RelationshipName, SubsetName, FromClassName, ToClassName의 네 개의 슬롯으로 구성되어 있다. 다음 그림 10.은 Relationship 타입에 대한 슬롯을 보여주고 있다.

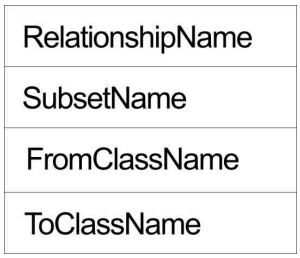

그림 10. Relationship 타입에 대한 슬롯
Fig. 10. Slot of Relationship Type

이와 유사하게 각 타입들에 대한 프레임을 얻을 수 있다. 다음 그림 11.은 각 타입들에 대한 프레임을 보여주고 있다.

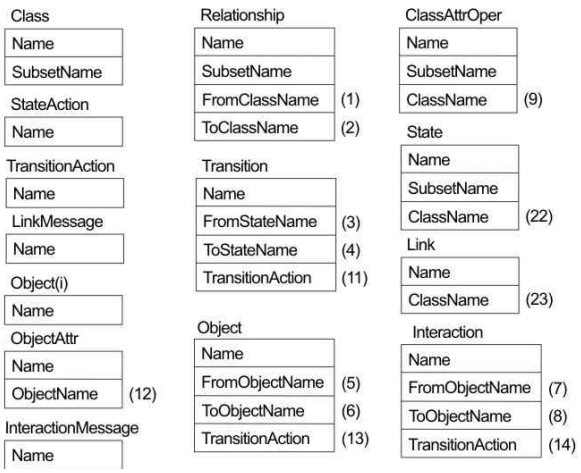

그림 11. 각 타입에 대한 프레임들
Fig. 11. Frames of Each Type

그림 11.에서 각 슬롯의 우측에 기재된 번호는 2장에서 제시된 집합 R의 원소인 함수에 붙여진 번호로써 해당 함수의 치역 타입의 값을 가진다는 의미이다. 결국 각 타입들은 single-linked-list 형태로 표현될 수 있으며 다음 그림 12.는 그림 2., 그림 3., 그림 4., 그림 5.의 클래스 다이아그램에서 유도된 정보를 가지고 있는 검증 시스템의 공통내부 표현이라고 할 수 있고 이러한 내부표현들을 입력으로 일관성과 완전성을 검증할 수 있다.

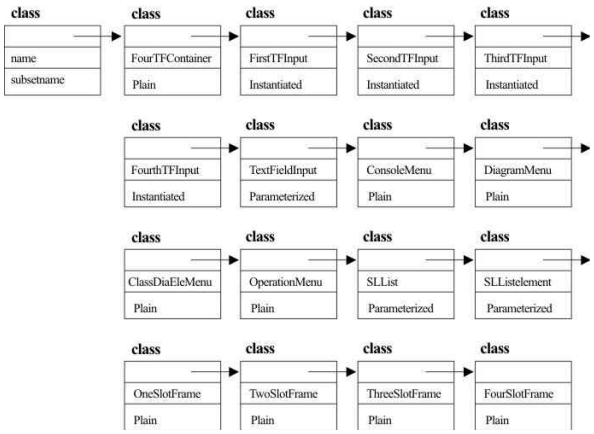

그림 12. 검증 시스템에 대한 공통 내부 표현
Fig. 12. Common Internal Representation of Verifying System

객체 다이아그램, 상태 다이아그램, 순차 다이아그램에 대한 공통내부 표현도 그림 11.의 프레임을 사용하여 그림 12.와 같이 표현될 수 있다.

Ⅴ. 결 론

본 논문에서는 [1]에서 제공된 다이아그램의 일관성과 완전성 검증 규칙을 사용하여, 작성된 UML 다이아그램의 일관성과 완전성을 검증하기 위한 검증 시스템을 설계하였다. 검증 시스템의 설계에는 UML 다이아그램을 활용하였으며 프레임을 이용하여 검증 시스템의 다이아그램에 대한 내부표현을 제시하였다.

본 논문에서 설계된 검증 시스템은 다이아그램에서 유도된 정보를 기반으로 일관성과 완전성을 검증하는데 필요한 다이아그램의 정보를 공통적인 내부표현으로 저장할 수 있도록 하기 위하여 프레임을 이

용하였고 이러한 검증과정을 보이기 위하여 다이아그램의 일관성과 완전성을 검증할 수 있는 검증 시스템에 대한 각 다이아그램을 대상으로 정보를 유도하였고 유도된 정보를 기반으로 검증에 기초가 되는 공통 내부 표현을 프레임을 이용하여 설계하는 과정을 제시하였다.

설계된 검증 시스템은 다양한 객체지향 다이아그램에 대하여 적용이 가능하며 기존의 다이아그래밍 도구와도 결합이 가능하도록 구성되어 있다.

향후연구과제로는 객체지향 소프트웨어 개발을 좀더 효율적으로 할 수 있도록 UML의 모든 다이아그램을 처리할 수 있는 그래픽 컴파일러와 같은 도구의 개발이 요구된다.

참 고 문 헌

[1] 김재웅 외, “UML 다이아그램간의 일관성과 완전성을 위한 검증 규칙 생성에 관한 연구”, *한국멀티미디어학회 논문지*, 제3권, 제3호, 290-298, 2000

[2] D. Milicev, "Automatic Model Transformations Using Extended UML Object Diagrams in Modeling Environments," *IEEE Transaction on Software Engineering,* Vol.28, No.4, pp413-431, 2002

[3] Object Management Group, *Unified Modeling Language (UML),* version 2.0, OMG Documentation, http://www.omg.org/technology /documents, 2003

[4] Van der Donckt, etc, "Synthronized model based design of multiple user interfaces", *Workshop on Multiple User Interfaces over the Internet*, 2001

[5] Terry Quatrani, *Visual Modeling with Rational Rose and UML,* Addison-Wesley, 1998

[6] Harmelan M. V.,"Object Modeling an User Interface Design", *Addison Wesley*, Reading, Mass., 2001

[7] Paul Harmon and Mark Watson, Understanding *UML : The Developer's Guide,* Morgan Kaufmann Publishers, Inc., 1997

[8] Martin Fowler, *UML Distilled,* Addison Wesley, 1997

[9] Edwards, J. M. and Henderson-Sellors, B., "A graphical notation for object-oriented analysis and design", *Journal of Object-Oriented Programming 4(9)* , 53-74, 1994

[10] Robert H. Bourdeau and Betty H. Cheng, "A Formal Semantic for Object Model Diagrams", *IEEE Trans. on Software Engineering,* Vol. 21, No. 10, pp. 799-821, 1995

김 진 수 (金鎭秀)

1986년 2월 : 중앙대학교 전자계산학과(이학사)
1988년 2월 : 중앙대학교 전자계산학과(이학석사)
1997년 8월 : 중앙대학교 컴퓨터공학과(공학박사)
1998년 3월~현재 : 건양대학교 컴퓨터학과 부교수
관심분야 : 객체지향 방법론, 소프트웨어 프로세스 개선, 소프트웨어 품질보증