UML 클래스 저작도구를 위한 메타데이터의 정보 구축 및 처리

김재훈* · 김윤호**

Constructing and Processing of the Metadata Information for UML Class Authorization Tool

Jae-Hoon Kim* · Yun-Ho Kim**

요 약

본 논문에서는 UML 클래스 저작도구를 위한 메타데이터의 정보를 구축하고 처리하는 방법을 제시하고자 한다. 저작도구의 메타데이터 정보 구축은 UML의 클래스 다이어그램을 구성하는 요소들 바탕으로 클래스 (Class)와 관계 (Relationship)를 정의한다. 클래스의 정보 정의는 클래스의 가시성과 클래스의 이름과 클래스의 속성 그리고 클래스의 오퍼레이션이다. 관계의 정보 정의는 관계의 이름과 관계의 유형과 시작클래스 그리고 도착 클래스이다. 본 논문에서 제시하는 클래스 저작도구를 위해 구축된 정보를 바탕으로 정보 저장소에서 정보를 저장하는 방법과 불러오는 방법을 제시한다.

ABSTRACT

This paper presents a technique of contstructing and processing the metadata information, which is for UML class authorization tool. Construction of the information authorization tool defines Classes and Relationship based on the elements of the consisting of UML class diagram. Definition of the information class is a visibility of Class, a name of Class, an attribution of Class and an operation of Class. Definition of the relationship with information is a name of Relationship, a type of Relationship, a FromClass and a ToClass. In this paper, for a class authorization tool, it suggests a technique to save and read information from informationStorage based on the constructed information.

키워드

UML, 클래스 다이어그램, 클래스, 관계

Key word

UML, Class Diagram, Class, Relationship

* 준회원 : 안동대학교 컴퓨터공학과 석사과정(zoom50210@gmail.com)
** 정회원 : 안동대학교 컴퓨터공학과 정교수

접수일자 : 2010. 10. 11
심사완료일자 : 2010. 12. 03

Ⅰ. 서 론

많은 객체지향 모델링언어가 존재한다. 하지만 각각의 모델링 언어의 표기법이 달라서 사용자간에 의사소통이 힘든 문제점이 발생하여 UML[1-4]이 1997년 발표되었고, 표준 통합 모델링 언어이다. UML의 궁극적인 목적은 많은 객체지향 모델링 언어들을 통합하기 위한 것이며, 그래픽 언어를 제공하는 것이다.

클래스 다이어그램은 시스템의 정적인 구조를 모델하며 개발자와 개발자 간에 또는 사용자간에 상호작용을 돕기 위한 수단으로 청사진 역할을 하기도 한다. 즉, 시스템의 구조를 분석하여 모델링 하는 기술이다. 시스템의 구조를 모델 하는 방법은 시스템에 사용되는 명사를 추출하여 각각의 클래스를 나타낼 수 있다. UML에서 클래스 다이어그램을 구성하는 요소는 클래스와 관계가 있다. 클래스는 name과 attribute 그리고 operation을 정의하고 있으며, 관계는 의존, 연관, 일반화, 실체화, 집합, 합성을 정의하고 있다. 따라서 본 논문에서 UML 클래스 저작도구는 UML의 클래스 다이어그램을 구성하는 요소들 바탕으로 클래스와 관계의 정보를 구축하고 처리하는 방법을 제시하고자 한다.

Ⅱ. UML 클래스 저작도구를 위한 메타데이터의 정보 구축

본 장은 UML 클래스 저작도구의 메타데이터 정보 구축을 기술하였다. UML에서 클래스 다이어그램의 큰 특징은 클래스의 집합, 클래스들의 관계를 표현하는 것이다. 클래스 다이어그램을 구성하는 요소의 예를 그림 1과 같이 보인다.

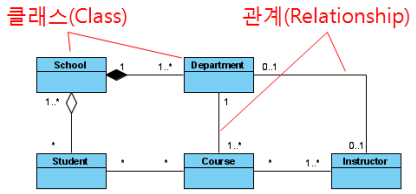


그림 1. UML에서 클래스 다이어그램을 구성하는 요소의 예
Figure 1. The Example of Element consisting Class Diagram in the UML

클래스는 시스템 개발에서 시스템의 용어를 획득하여 클래스를 표현할 수 있고, 개념적인 것을 표현한다. 관계는 시스템에서 어떠한 기능을 수행하기 위해서 클래스들 간에 협력하여 수행하는 경우가 있는데, 이러한 경우 클래스들 간에 협력하기 위해서 클래스와 클래스 사이에 관계를 나타낼 수 있다. 그러므로 저작도구를 위한 메타데이터의 정보는 클래스의 정보, 관계의 정보로 분류하였다. 분류한 정보의 클래스들은 그림 2와 같다.

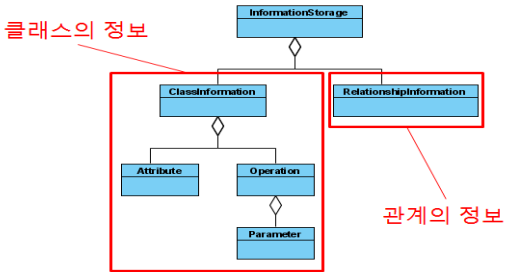


그림 2. UML 클래스 저작도구를 위한 메타데이터의 정보를 구성하는 클래스
Figure 2. The Class consisting Information of Metadata for UML Class Authorization Tool

저작도구를 위해 구축된 정보 클래스들의 역할은 표 1과 같이 보인다.

표 1. UML 클래스 저작도구를 위한 정보 클래스 요약
Table 1. The Summary Class Information for the UML Class Authorization Tool

구분	내용
InformationStorage	UML 클래스 저작도구에서 클래스의 정보와 관계의 정보를 묶어서 관리하는 곳
ClassInformation	클래스의 정보
Attribute	속성 정보
Operation	오퍼레이션 정보
Parameter	파라미터 정보
RelationshipImformation	관계의 정보

2.1 저작도구의 클래스 정보 구축

본 절에서는 UML에서 클래스 다이어그램의 요소 중 하나인 클래스의 특징을 바탕으로 저작도구를 위한 클

래스의 정보를 구축하였다. UML에서 정의된 클래스는 name, attribute 그리고 operation를 나타내고 있으며 그림 3과 같다.

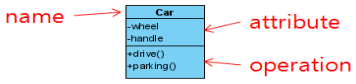


그림 3. UML에서 정의한 클래스
Figure 3. The Class defined in The UML

저작도구의 클래스 정보 구축은 클래스의 가시성, 클래스의 이름, 클래스의 속성 그리고 클래스의 오퍼레이션 이렇게 4가지로 구성하였다. 클래스의 가시성은 클래스의 접근성을 나타내기 위하여 추가하였고, 클래스의 이름과 속성 그리고 오퍼레이션은 UML에서 클래스를 구성하는 요소들 바탕으로 정의하였다. 클래스의 이름은 각각의 클래스를 식별하기위해 필요하며 유일하도록 하였다. 클래스의 속성과 클래스의 오퍼레이션은 각 클래스 마다 하나 이상을 가지거나 없을 수 있기 때문에 만약 하나 이상의 속성 또는 오퍼레이션을 가질 경우를 위해 각각의 속성과 오퍼레이션을 가지도록 클래스로 분류하고 그림 4와 같이 보인다.

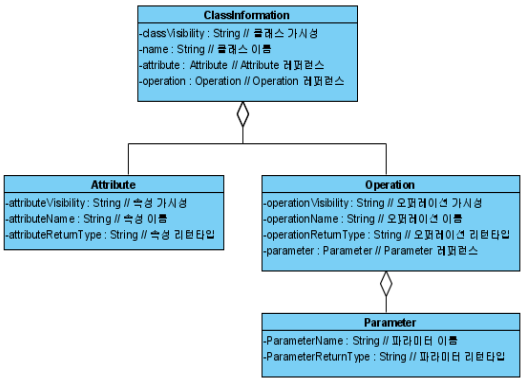


그림 4. UML 클래스 저작도구의 클래스 정보를 구성하는 구조
Figure 4. The Structure consisting Class Information of the UML Class Authorization Tool

클래스의 가시성과 클래스 이름을 표현한 표기법에 따른 예는 그림 5와 같고, 클래스 정보를 구성하는 요소의 요약은 표 2와 같다.

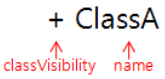


그림 5. 클래스의 이름과 가시성 표기법에 따른 예
Figure 5. The Example of Notation according to Name and Visibility of the Class

표 2. 클래스 정보를 구성하는 요소의 요약
Table 2. The Summary of Element consisting Class Information

이름	내용
classVisibility	클래스의 가시성
name	클래스의 이름
attribute	클래스의 속성
operation	클래스의 오퍼레이션

2.1.1 Attribute 정보 정의

클래스의 속성은 속성의 가시성과 이름 그리고 타입을 정의하였다. 클래스의 속성을 표현한 표기법에 따른 예는 그림 6과 같고, 정보를 구성하는 요소의 요약은 표 3과 같다.

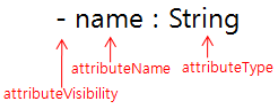


그림 6. 클래스의 속성 표기법에 따른 예
Figure 6. The Example of Notation according to an Attribute of the Class

속성의 가시성은 속성의 접근성을 나타내며, 속성의 이름은 클래스마다 각각의 속성을 구별하기 위해 정의하였다. 속성의 타입은 각각의 속성마다 타입을 지정할 수 있다.

표 3. 클래스의 속성 정보를 구성하는 요소의 요약
Table 3. The Summary of Element consisting Information an Attribute of the Class

이름	내용
attributeVisibility	속성의 가시성
attributeName	속성의 이름
attributeType	속성의 타입

2.1.2 Operation 정보 정의

클래스의 오퍼레이션은 오퍼레이션의 가시성과 이름과 리턴타입을 정의하였고, 오퍼레이션에서 표현하는 파라미터는 없을 수도 있거나 하나 이상을 가질 수 있도록 하였다. 오퍼레이션을 표현한 표기법에 따른 예는 그림 7과 같고, 정보를 구성하는 요소의 요약은 표 4와 같다.

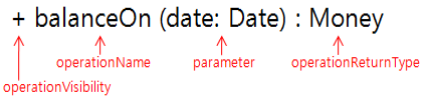


그림 7. 클래스의 오퍼레이션 표기법에 따른 예
Figure. 7. The Example of Notation according to an operation of the Class

오퍼레이션의 가시성은 오퍼레이션의 접근성을 타나내며, 오퍼레이션의 이름은 클래스마다 각각의 오퍼레이션을 식별하기 위해 정의하였다. 오퍼레이션의 리턴타입은 오퍼레이션이 수행되고 나서 리턴 값을 돌려주며, 파라미터는 오퍼레이션에서 파라미터를 넘겨받는 경우에 표현한다.

표 4. 클래스의 오퍼레이션 정보를 구성하는 요소의 요약
Table 4. The Summary of Element consisting Information an operation of the Class

이름	내용
operationVisibility	오퍼레이션의 가시성
operationName	오퍼레이션의 이름
operationReturnType	오퍼레이션의 리턴타입
parameter	파라미터

2.1.3 Parameter 정보 정의

파라미터는 파리미터의 이름과 타입을 정의하였다. 파라미터의 표기법에 따른 예는 그림 8과 같고, 정보를 구성하는 요소의 요약은 표 5와 같다.

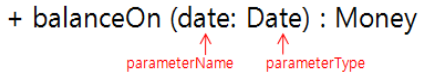


그림 8. 파라미터의 표기법에 따른 예
Figure 8. The Example of Notation according to the Parameter

파라미터 이름은 오퍼레이션에서 각각의 파라미터를 식별하기 위해 정의하고, 파라미터 타입은 각각의 파라미터마다 타입을 지정 할 수 있다.

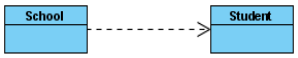
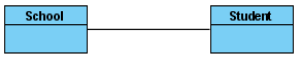
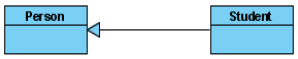
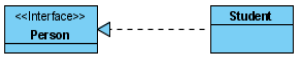
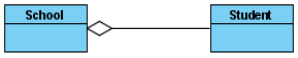
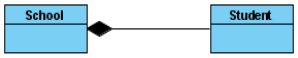
표 5. 파라미터 정보를 구성하는 요소의 요약
Table 5. The Summary of Element consisting a Parameter Information

이름	내용
parameterName	파라미터 이름
parameterType	파라미터 타입

2.2 저작도구의 관계 정보 구축

본 절에서는 UML에서 클래스 다이어그램의 정의된 관계를 바탕으로 저작도구의 관계 정보 구축을 정의 하였다. 정의한 관계는 그림 6과 같다.

표 6. UML에서 정의된 관계
Table 6. The Relationship defined in the UML

관계	표기법
의존 (dependency)	
연관 (association)	
일반화 (generalization)	
실체화 (realization)	
집합 (aggregation)	
합성 (composition)	

저작도구의 관계 정보 구축은 각각의 관계를 식별하기 위해서 관계 이름을 정의하였고, 관계의 방향성을 가질 수 있기 때문에 시작클래스와 도착클래스를 나타내었다. 그리고 클래스와 클래스의 관계가 어떠한 관계인지 표현하기 위해서 관계의 유형으로 정의하였다. 저작도구의 관계 정보를 구성하는 구조는 그림 9와 같고, 정보를 구성하는 요소의 요약은 표 7과 같다.

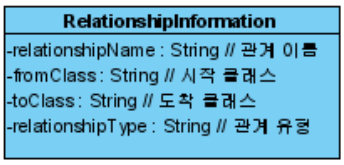


그림 9. UML 클래스 저작도구의 관계 정보를 구성하는 구조
Figure 9. The Structure consisting a Relationship Information of UML Class Authorization Tool

표 7. 관계 정보를 구성하는 요소의 요약
Table 7. The Summary of Element consisting a Relationship Information

이름	내용
relationshipName	관계의 이름
relationshipType	관계의 유형
fromClass	시작 클래스
toClass	도착 클래스

Ⅲ. UML 클래스 저작도구를 위한 메타데이터의 정보 처리

본 장에서는 2장에서 구축한 정보를 바탕으로 정보저장소에서 클래스와 관계의 정보를 처리하는 방법을 기술하였다.

3.1 저작도구의 구조

본 절에서는 저작도구의 구조를 기술하였으며. 저작도구의 컴포넌트 구조는 [5]와 같으며 MVC패턴에 기반하였다. MVC 패턴[6-8]을 구성하는 컴포넌트 구조는 View 컴포넌트, Controller 컴포넌트, Model 컴포넌트로 구성되어있다. View 컴포넌트 목적은 UML클래스 저작도구의 화면 출력을 담당하며 GUI에 관련된 컴포넌트이다. Controller 컴포넌트는 입출력을 담당하며, View 컴포넌트와 Model 컴포넌트사이에서 중개하는 목적을 가진다. Model 컴포넌트의 목적은 비즈니스 로직에 따라 데이터를 가공하고 처리하며 아래의 그림 10과 같다.

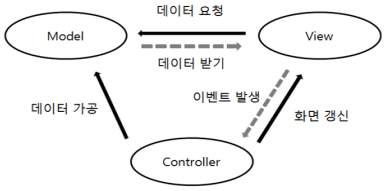


그림 10. MVC 패턴의 구조
Figure 10. The Structure of MVC Pattern

저작도구의 메타데이터를 처리하기 위해서 적용한 각각의 컴포넌트를 나타내었으며 그림 11과 같다. 각각의 컴포넌트를 구성하는 클래스들의 역할을 기술한 것은 표 8과 같다.

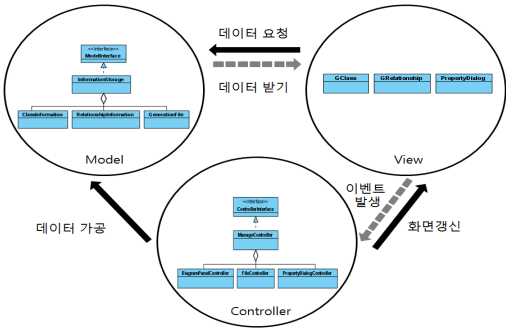


그림 11. 저작도구의 컴포넌트 구조
Figure 11. The Structure Component of Authorization Tool

표 8. 저작도구의 컴포넌트 역할
Table 8. The Role Component of Authorization Tool

컴포넌트	클래스 이름	클래스 역할
View	GClass	GNode 인터페이스의 실체화이며 클래스를 그리는 역할
	GRelation ship	GEdge 인테페이스의 실체화이며, 관계를 그리는 역할
	Property Dialog	노드, 연결선의 속성을 표현하는 영역
Controller	Controller Interface	Controller 컴포넌트의 인터페이스 정의
	Manage Controller	ControllerInterface의 실체화이며, 각각의 컨트롤러에게 데이터 입력 또는 발생된 이벤트를 위임하는 역할
	Diagram Panel Controller	다이어그램을 그리기 위해서 DiagramPanel에서 발생하는 이벤트 수행

컴포넌트	클래스 이름	클래스 역할
	File Controller	클래스의 정보와 관계의 정보를 파일로 저장하기 위해서 파일에 관련된 이벤트 수행
	Property Dialog Controller	클래스의 이름, 클래스의 속성, 클래스의 오퍼레이션의 입력된 데이터를 처리하기 위해서 PropertyDialog에서 발생하는 이벤트 수행
Model	Model Interface	Model 컴포넌트의 인터페이스 정의
	Information Storage	클래스와 관계 그리고 파일의 정보를 묶어서 관리하는 저장소
	Class Information	클래스의 정보를 저장하기 위한 클래스의 정의
	Relationship Information	관계의 정보를 저장하기 위한 관계의 정의
	Generation File	클래스와 관계의 정보를 파일로 저장하기 위한 파일 정보의 정의

3.2 정보저장소에서 클래스 정보의 처리

3.2.1 클래스 정보저장소에서 클래스정보 저장

새로운 클래스의 정보를 입력 했을 때, 그 정보를 저장하는 과정을 나타내기 위해서 클래스의 이름 저장을 시퀀스 다이어그램으로 나타내고 그림 12와 같다.

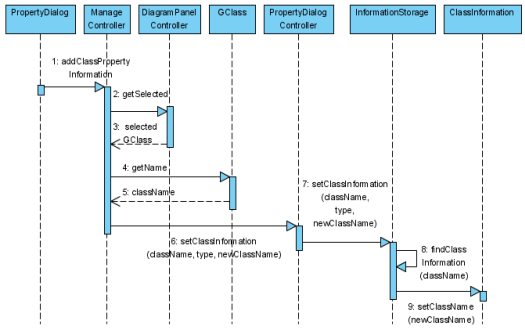


그림 12. 클래스의 정보 저장
Figure 12. The Save of the Class Information

PropertyDialog에서 새로운 클래스의 이름을 입력받는다. 입력받은 그 정보를 Controller 컴포넌트의 인터페이스 역할을 하는 ManageController에게 넘겨준다. Manage zController는 Diagrma Panel 의 이벤트를 처리하는 DiagramPanelController에게 선택된 노드(클래스)를 요청하고 선택된 노드를 반환받는다. 그리고 클래스 이름은 각각의 클래스를 구별하기 위해서 유일하며 중복되어 있지 않기 때문에 이름으로 선택된 클래스를 식별할 수 있도록 하였다. 그러므로 선택된 노드의 이름을 식별하기 위해 DiagramPanelController는 GClass에게 이름을 요청하고 반환받는다. 반환 받은 이름과 정보의 타입과 입력받은 새로운 클래스의 이름을 PropertyDialog의 이벤트를 수행하는 PropertyDialogController에게 넘겨준다. 정보의 타입은 클래스의 정보를 가지고 있는 ClassInformation에게 클래스의 어떠한 정보를 뜻하는지 식별하기위한 것이다. PropertyDialogController는 클래스의 정보와 관계의 정보 그리고 파일의 정보를 저장하고 있는 InformationStorage에게 요청을 위임한다. InformationStorage는 선택된 노드의 이름으로 저장되어 있는 클래스를 식별한다. 마지막으로 식별된 ClassInformation에게 새로운 클래스의 이름을 저장하고 마친다.

3.2.2 클래스 정보저장소에서 클래스정보 불러오기

파일에 클래스의 정보를 구성하는 클래스의 이름과 클래스의 속성 그리고 클래스의 오퍼레이션을 저장하는 경우, 클래스 정보저장소에서 클래스정보를 불러와야 한다. 먼저, DiagramPanelController가 GClass에게 노드의 이름을 요청하고 반환받는다. 그리고 반환받은 이름으로 InformationStorage에게 선택된 노드의 이름으로 저장되어 있는 클래스를 식별한 다음 식별된 클래스를 DiagramPanelController에게 반환한다.

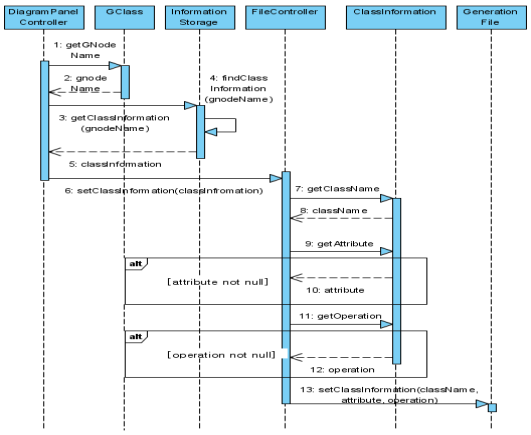


그림 13. 클래스의 정보 불러오기
Figure 13. The Read Information of The Class

DiagramPanelContrller는 파일의 정보를 저장하고 있는 FileController에게 반환받은 클래스를 넘겨준다. FileController는 ClassInformation에게 클래스의 이름을 요청하고 반환받으며 클래스의 속성과 오퍼레이션은 만약 정보가 저장되어 있을 경우에 클래스의 속성과 오퍼레이션을 반환한다. 마지막으로 반환받은 클래스의 정보를 GenerationFile에게 넘겨주고 끝으로 클래스 정보저장소에서 클래스정보 불러오기를 마친다. 그림 13와 같이 보인다.

3.3 정보저장소에서 관계 정보의 처리

3.3.1 관계정보 저장소에서 관계정보 저장

새로운 관계의 정보를 저장하기 위해서 크게 3가지로 나눌 수 있다. 먼저 fromClass의 정보를 식별하여 저장한 다음, toClass의 정보를 식별하여 저장한다. 마지막으로 fromClass와 toClass의 관계 유형의 정보를 식별하고 저장한다. 그림 14는 새로운 관계의 정보를 저장하기 위해서 관계정보 저장소에 저장하는 순서를 A, B, C프레임으로 나눠서 시퀀스 다이어그램으로 표현하였다.

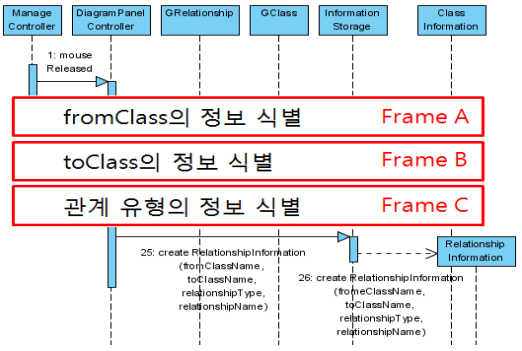


그림 14. 새로운 관계의 정보를 관계 정보저장소에 저장
Figure 14. The Save new Relationship in Relationship InformationStorage

ManageController는 마우스 이벤트를 감지하고 DiagramPanelManager에게 이벤트 유형에 따른 이벤트 처리를 요청한다. 다음으로 fromClass의 정보를 식별하고 획득하는 방법은 그림 15와 같으며, toClass의 정보를 식별한다. 식별하는 방법은 그림 16과 같다. 그리고 관계 유형의 정보를 식별하고 획득하는 방법은 그림 17과 같다. 이 세 가지의 정보를 획득하여 마지막으로 클래스와 관계 그리고 파일의 정보를 저장하고 있는 InformationStorage에게 넘겨준다. 그리고 관계의 정보를 저장하는 RelationshipInformation을 생성하기 위해서 InformationStorage는 새로운 RelationshipInformation을 생성하고 생성된 관계에 저장을 끝으로 새로운 관계의 정보를 관계 정보저장소에 저장하는 것을 마친다.

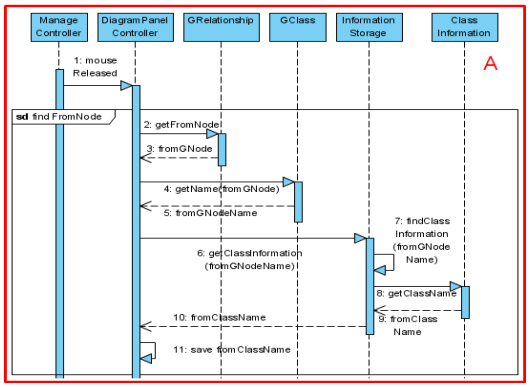


그림 15. fromClass의 정보식별(Frame A)
Figure 15. The Information Identify of a fromClass

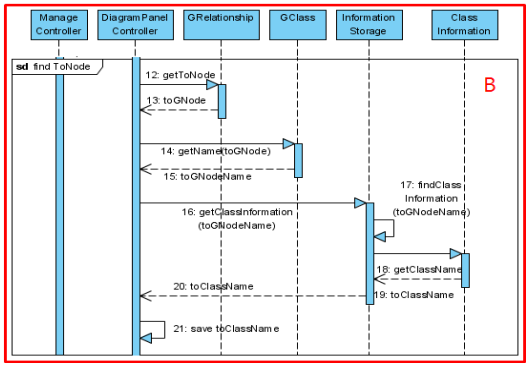


그림 16. toClass의 정보식별(Frame B)
Figure 16. The Information Identify of a toClass

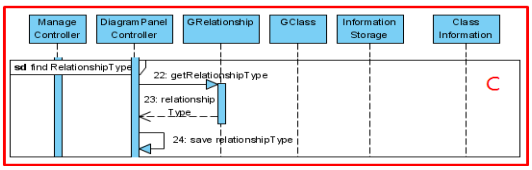


그림 17. 관계 유형의 정보식별(Frame C)
Figure 17. The Information Identify of the Relationship Type

fromClass의 정보를 획득하는 방법은 DiagramPanel Controller는 GRelationship (선택된 관계)에게 fromNode를 요청하고 반환 받는다. 연결선은 저작도구의 다이어그램에서 관계를 표현한다. 그리고 반환 받은 fromGNode의 이름을 가지기 위해서 GClass에게 이름을 요청하고 반환받는다. DiagramPanelController는 InfromationStorage에게 반환받은 이름을 넘겨주고, InformationStorage는 fromClass이름으로 저장되어진 클래스를 식별한다. ClassInformation 에게 클래스의 이름을 요청하고 반환받는다. 이것을 DiagramPanelController에게 위임하고 DiagramPanelController는 fromClass의 이름을 획득한다. toClass의 정보를 획득하는 방법은 위의 FromClass의 정보를 획득하는 것과 같은 방법이다. 관계 유형의 정보를 획득하는 방법은 ManageController는 GRelationship에게 관계의 유형을 요청하고 반환받는다. 그리고 관계의 유형을 획득한다.

3.3.2 관계 정보저장소에서 관계정보 불러오기

파일에서 관계의 정보를 구성하는 관계의 유형을 저장하는 경우, 관계 정보저장소에서 관계정보를 불러오며 아래의 그림 18과 같다.

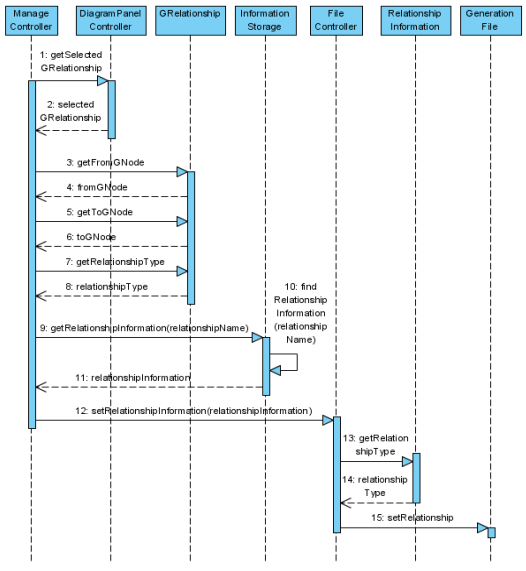


그림 18. 관계 정보 불러오기
Figure 18. The Relationship Information Read

먼저 ManageController는 DiagramPanelController에게 선택된 연결선을 요청하고 반환받는다. 그리고 GRelationship (반환받은 관계)에게 fromNode와 toNode 그리고 관계의 유형을 요청하고 반환받는다. fromNode와 toNode 그리고 관계의 유형을 식별하는 이유는 선택된 관계의 정보를 식별하기 위해서 관계의 이름을 가지고 있어야 한다. 관계의 이름은 fromNode와 toNode 그리고 관계의 유형의 조합으로 되어있기 때문이다. Manage Controller는 GRelationship에서 반환 받은 정보를 조합하여 관계의 이름으로 가지게 되고 InformationStroage에게 선택된 연결선이름으로 저장하고 있는 관계를 식별하기위해 관계의 이름을 넘겨준다.

InformationStroage는 넘겨받은 그 이름으로 저장되어 있는 관계를 식별하고 식별된 관계를 ManageController에게 반환한다. FileController에게 반환받은 관계를 넘겨준다. FileController는 그 관계의 유형을 식별하기 위해서 RelationshipInfromation에게 관계의 유형을 요청하고 반환받는다. 마지막으로 반환받은 관계의 유형을 GenerationFile에게 넘겨주는 것을 끝으로 파일에다가 관계의 정보를 불러와서 저장하는 작업을 마친다.

3.3.3 클래스정보 저장구조

저작도구의 화면에서 작성된 클래스와 관계의 정보를 파일에 저장하여 다이어그램의 일관성을 유지하도록 하였다. 2장에서 구축한 정보를 바탕으로 클래스와 관계의 정보를 파일에 저장하는 구조는 그림 19와 같이 보인다. 파일에서 클래스 정보와 관계 정보를 저장하도록 하였다. 클래스 정보는 각각의 클래스들을 식별하기 위해서 클래스 이름을 저장하고 가시성을 저장하도록 하였다. 속성은 가시성, 이름, 타입을 저장하고, 오퍼레이션은 가시성, 이름, 리턴타입을 저장한다. 파라미터는 이름, 타입을 저장한다. 그리고 각각의 클래스의 위치 정보를 저장하기 위해서 클래스의 시작점 x, 시작점 y, 너비, 높이를 저장한다. 클래스의 속성과 오퍼레이션을 구분하기 위해서 속성의 높이와 오퍼레이션의 높이도 저장하도록 하였다. 관계의 정보는 관계의 이름, 관계의 타입, 시작클래스, 도착클래스, 관계의 시작점, 관계의 도착점, 관계의 화살표 머리 모양, 선 모양을 저장하도록 하였다.

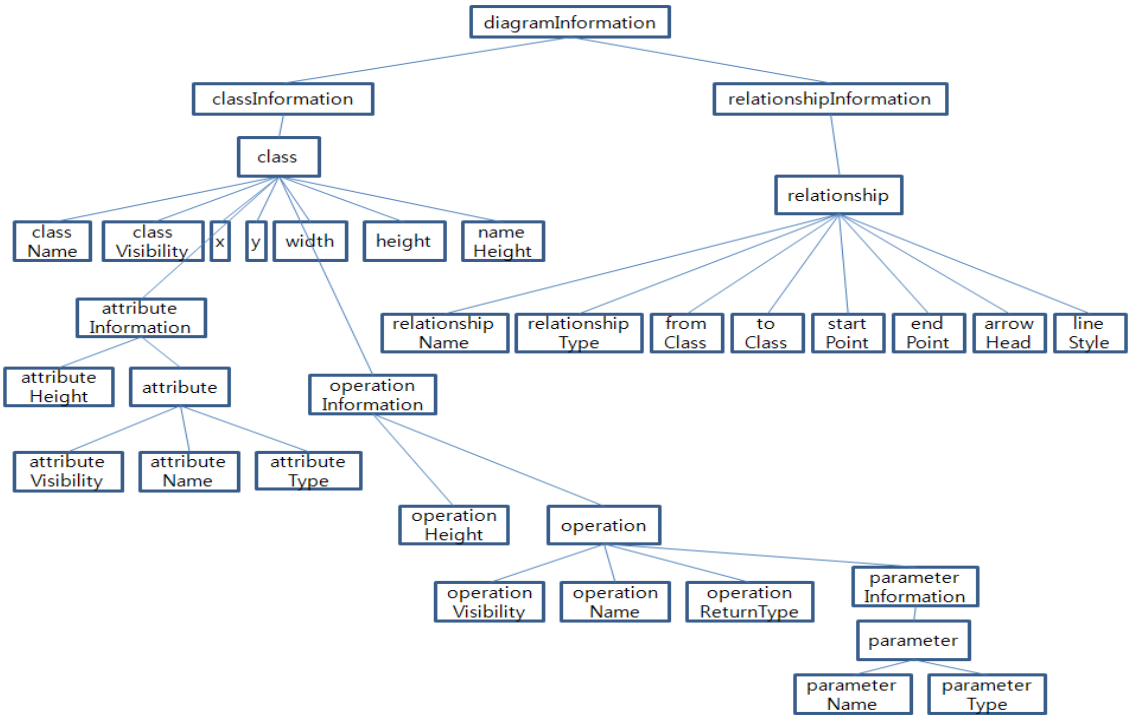


그림 19. 클래스 정보를 파일에 저장하는 구조
Figure. 19 The Structure to save Class Information into file

Ⅳ. 결 론

본 논문에서는 UML 클래스 저작도구를 위한 정보의 구축과 처리를 제시하였다. UML에서 정의된 클래스다이어그램의 요소를 식별하여 UML 클래스 저작도구에서 필요한 클래스의 정보와 관계의 정보를 구축하였다. 클래스의 정보는 클래스의 가시성, 클래스의 이름, 클래스의 속성, 클래스의 오퍼레이션을 정의하고, 관계의 정보는 관계의 이름, 관계의 유형, 관계의 이름, 시작클래스, 도착클래스를 정의하였다. 구축된 정보들 바탕으로 정보저장소에서 정보를 처리하는 방법을 연구하였다.

향후 과제로는 시스템을 개발할 때 구현하는 작업의 시간을 단축시키고 효율성을 높이기 위해서 클래스 저작도구에서 나타내는 클래스 다이어그램을 코드로 변환하는 메커니즘 설계의 연구가 필요하다.

참고문헌

[1] OMG Group, UMLspeccification, www.omg.org
[2] IBM, Rational Rose, www.ibm.com
[3] Martin Fowler, *UML Distilled,* 3rd Ed., Addison-Wesley, 2004
[4] Grady Booch, James Rumbaugh, Ivar Jacobson, *The Unified Modeling Language User Guide 2nd ED*, Addison-Wesley, 2005
[5] 김재훈, 김윤호, “MVC 디자인패턴에 기반한 클래스 다이어그램 저작도구의 설계”, 한국해양정보통신학회 논문지, Vol.14 No.12, 2010
[6] Gamma E., Helm R., Johnson., Vlissides J., *Design Patterns - Elements of Reusable Object-Oriented Software,* AddisonWesley, 1994
[7] Cay Horstmann, *Object-Oriented Design & Patterns 2nd ED,* Wiley, 2005

[8] Kathy Sierra, *Head First Design Patterns,* O'RRILLY, 2004

저자소개

김재훈(Jae-Hoon Kim)

2009.2 안동대학교 컴퓨터공학과 공학사
2009.3-현재 안동대학교 컴퓨터공학과 석사과정

※관심분야 : 객체지향 시스템, 인터넷 컴퓨팅, 리팩토링 등

김윤호(Yun-Ho Kim)

1983 경북대학교 전자공학과 공학사
1993 경북대학교 대학원 컴퓨터공학과 공학석사
1997 경북대학교 대학원 컴퓨터 공학과 공학박사
1997-현재 안동대학교 전자정보산업학부 교수

※관심분야 : 객체지향 시스템, 인터넷 컴퓨팅, 병렬처리 등