GoF 디자인 패턴기반 객체지향 오염총량제 소프트웨어 설계

김 형 무† · 곽 훈 성††

요 약

객체지향 모델링에서 컴포넌트기반 방법을 사용하는 목적은 반복되는 시간 및 공간 복잡성을 줄여 연산능력을 높이는 것이다. 이러한 컴포넌트기반 방법의 많은 성과에도 불구하고 디자인 패턴과 그 표준화로 컴포넌트기반 방법의 재사용성을 높이는 연구가 필요하다. 그러나 디자인 패턴을 표준화하는 방법으로 성급하게 메타패턴언어나 패턴저장소를 구축하는 방법은 오히려 소프트웨어 개발에 패턴을 적용하는 것을 더욱 복잡하고 어렵게 만들 수가 있다. 본 연구는 환경분야의 TMDL(오염총량제) 소프트웨어의 설계에 GoF 디자인 패턴을 적용함으로써, 과도하게 추상화된 메타패턴언어나 부가적인 패턴저장소를 두지 않고도 적용패턴을 검색, 추적할 수 있도록 설계단계에서 클래스이름에 패턴이름을 명시하는 방법을 제안하였다. 따라서 본 연구결과는 환경소프트웨어 개발과정에서 빈번하게 발생하는 반복과 중복을 줄일 수 있는 이점이 있다.

GoF design patterns based object-oriented Total Maximum Daily Load software design

Hyung-Moo Kim† · Hoon-Seong Kwak††

ABSTRACT

The purpose of using CBD in the object-oriented modeling is to improve the software capability by reducing iterative time and space complexity. Despites many achievements of CBD, it is needed to study about design patterns and it's standardization for the increment of CBD design reusability. However, it is rather possible that impetuous constructing meta-pattern languages and pattern repositories make adapting patterns to software development more complicate and difficult. By applying GoF design patterns to the design of the TMDL(Total Maximum Daily Load) environmental software discipline, this study suggests a method which specifies pattern names at class names for retrieving, exploring the adapted patterns on the stage of software design without meta-pattern language which is a redundant abstraction, nor additional pattern repositories. Thus, this study can contribute on the reducing iterations and repetitions that are frequently occurred in the process of the environmental software developments.

키워드 : 4인방 디자인 패턴(GoF Design Patterns), CBD, 소프트웨어 재사용성(Software Reusability), 객체지향 오염총량제(Object-Oriented TMDL), 환경소프트웨어(Environmental Software)

1. 서 론

Gamma 등 [7]에 의해 소개된 GoF(4인방 : Gang of Four) 디자인 패턴(Design Patterns)은 소프트웨어 설계패턴을 생성, 구조, 행위 패턴으로 분류하고 객체지향설계를 위한 가장 대중적인 설계패턴 23가지를 제시한 것이다. 이를 요약한 GRASP(일반책임할당 패턴 : General Responsibility Assignment Software Pattern) 패턴은 초기의 GoF 디자인 패턴에 책임이라는 객체역할의 할당원리를 강조하여 실제 프로그래밍에 패턴을 쉽게 구현할 수 있게 한 것이다[3, 7].

오염총량제(TMDL : Total Maximum Daily Load)[2]는 용수이용목적에 맞게 목표수질을 설정하고, 해당 수계(watershed)의 배수구역(catchment)에서 배출되는 오염부하총량을 설정된 목표수질을 달성할 수 있도록 관리하는 제도이다. 기존 농도 규제로는 중. 하류에 밀집된 인구 및 산업시설을 갖고 있는 유역(basin)에서는 근본적인 수질개선이 어려운 바, 오염부하량의 양적 증가를 통제하기 위해 하천의 일정한 지점에서 오염농도만 규제하던 방식을 폐지하고 유역전체의 오염물질배수량 총량을 관리, 규제하는 것이 TMDL이다.

이 연구는 GoF 디자인 패턴기반 방법에 의해 효율성, 재사용성, 다형성이 보장된 객체지향 TMDL 환경소프트웨어 모델의 개발을 목적으로 한다. 여기서, 재사용성이 유지보수를 위해 필요하다면, 다형성은 TMDL 모델을 외부의 홍수통제, 수질관리, 상하수도 시설물관리 등에 확장시키기 위해 필요하다. 컴포넌트기반 개발의 조건인 객체지향 분석 및 설계는 객체를 기본 단위로 하여 실세계를 쉽게 인식할 수 있게 단순화하고 체계적으로 표현한 것이다. 객체지향

† 정 회 원 : 전북대학교 컴퓨터공학과 시간강사
†† 정 회 원 : 전북대학교 영상산업(학부전공) 및 영상공학과(대학원) 주임교수
논문접수 : 2004년 9월 2일, 심사완료 : 2005년 1월 31일

기술은 구조적, 정적모델이 주로 사용되다가 최근 객체의 역할을 강조한 동적, 기능적 모델을 컴포넌트 방식으로 조립하여 개발하는 방향으로 발전하고 있다. 역할을 강조한 동적, 기능적 객체지향기술의 하나가 디자인 패턴이다. 실세계를 보다 정확히 묘사하는 설계를 얻기 위해 이 연구는 객체지향원리와 디자인 패턴을 사용하고, 분석과 설계에서 표준화프로세스(UP : Unified Process[3])를 기반으로 한 일반적인 활동들을 추상화한 모델을 설계하고, 이전 과정동안 UML(Unified Modeling Language)의 표기법을 사용하여 설계를 가시화해주는 다이어그램을 생성하고 코드를 자동생성하게 함으로써 컴포넌트기반 개발의 재사용성이 강화되는 것을 보일 것이다.

본 논문의 구성은 제2장에서 관련연구 및 연구방법을 기술하고, 제3장에서 디자인 패턴 모델을 제안하고, 제4장에서 오염 총량 관리 소프트웨어를 설계하며, 제5장에서 비교 및 평가하고, 마지막으로 결론 및 향후연구에 대해 기술한다.

2. 관련연구 및 연구방법

2.1 관련연구

디자인 패턴은 객체지향 시스템에서 반복적으로 발생하는 디자인 문제를 체계적으로 명명하고, 동기를 유발하고, 설명함으로써 문제, 해결책, 언제 해결책을 적용할 것인지와 그 연장을 기술하고 또한, 구현 힌트와 구현 예를 제시한다. 이때 해결책은 그 문제를 해결하는 일반적인 객체와 클래스들의 배치이다. 해결책은 특수한 문법에 의해 해결되게 하기 위해 맞춤화 되고 소프트웨어 공학의 UP에 구현된다.

〈표 1〉 UP의 분야와 부산물[3]

분야 (Discipline)	부산물 (Artifact)
비즈니스 모델링	도메인 모델 (Domain Model)
요구사항	사용사례 모델 (Use-Case Model)
	비전 (Vision)
	보충명세서(Supplementary Specification)
	용어집 (Glossary)
설계	설계 모델 (Design Model)
	소프트웨어 아키텍처 문서
	데이터 모델 (Data Model)

디자인 패턴은 Booch 등 [5]에 의해 소프트웨어 설계에 필요한 다이어그램 작성 및 문서관리가 가능해진 이후인 1990년대 중반 Gamma 등 [7]이 객체지향설계를 위한 가장 대중적인 설계패턴인 GoF 패턴을 제안한 데서 시작되었다. 그러나 Gamma 등의 방법은 패턴의 분류에 머물러 있어 실제 소프트웨어 설계시 어떤 패턴을 적용할지 모르게 되고, 동시에 너무 상세한 분류로 인해 표준화하기 곤란하여 패턴의 해결책으로서의 본질이 구현되지 못하는 단점이 있다[1]. 또한, Gamma[8]의 패턴은 다양한 도메인에서 일반적으로 사용될 수 있는 경험들을 추상적으로 표현한 것이므로 적용영역에 대한 제한은 없지만 이 분류는 너무 광범위한 것이어서 주어진 설계상황에서 소프트웨어 설계자를 위한 유용성은 떨어진다. Larman[3]은 객체에 책임을 할당하는 정책을 GRASP 패턴으로 정의하고 Information Expert, Creator, High Cohesion, Low Coupling 및 Controller 등의 책임할당원리를 제시하였으나 5대 책임할당원리가 상호간에 중복성이 있고 원리에 머물러 객체지향 소프트웨어의 개발 시 구체적 유용성을 제시하지 못하였다[1]. Budinsky 등 [4]은 디자인 패턴으로부터 코드자동생성기법을 제안하였으나 설계단계에 이 패턴정보가 반영되지 않았다. 설계이전 단계부터 패턴정보를 반영하기 위해 Eden등 [6]은 메타언어를 이용한 어플리케이션 소스코드 생성기법을 제안하였으나, 메타언어에 대한 사전 지식이 필요하였고 반대로 설계정보의 구체적인 활용규칙이 제시되지 않았다. 이 문제를 해결하기 위해 Sefica 등[10]은 디자인 패턴에 규칙을 정의하여 설계정보가 그 규칙을 만족하는지를 검증하는 방법을 제안하였으나 패턴에 대한 규칙관리가 곤란하고 설계정보에 디자인 패턴 자동화나 적용방법에 대한 제시가 부족하여 사용된 패턴에 대한 추적이 곤란하다. 김운용 등[1]은 패턴관리부, 소프트웨어 설계부, 소스코드 관리부로 구성된 XML기반 패턴정보관리기를 제안하여 설계단계에서 다이어그램 생성시 적용시킬 패턴이 자동으로 검색되어 이 디자인 패턴을 설계와 구현에 반영하고 이 반영 기록을 저장하여 패턴 추적을 가능하게 하는 방법을 제시하였으나 기존 디자인 패턴의 저장 추적과 자동화 소스코드 생성이 가능한 반면 디자인 패턴의 다형성 즉 파생패턴의 자동추출, 자동생성에 까지 이르지 못하였다.

이와 같은 선행연구를 바탕으로 본 연구는 특정 개발프로세스 중에서 반복적 개발 프로세스(Iterative Development Process) 방법의 표준으로 널리 알려진 UP에 맞게 디자인 패턴을 객체지향 환경소프트웨어 개발 사례연구에 적용한다. 객체지향 분석 및 설계의 입장에서 TMDL 시스템의 디자인 패턴기반 데이터모델은 분석과 설계의 문제를 해결하기 위해 고려해야 하는 사항들을 규정하고 패턴에 대한 정확한 정의를 바탕으로 TMDL소프트웨어에 적용할 5개의 패턴을 중심으로 객체지향 원리에 부합하도록 설계를 정형화하였다.

2.2 연구방법

본 연구는 디자인 패턴기반 동적 개발방법에서 UML 설계문서에서 패턴의 이름을 클래스 이름 뒤에 첫 글자를 대문자로 붙여 표시함으로써 패턴의 추적, 검색, 적용, 생성, 저장, 표준화를 가능하게 한다. 이는 객체지향 패턴정보관리를 자동화하고 적용된 디자인 패턴의 적합성을 분석하고, 적용패턴을 추적가능하게 하고, 객체들 간 책임할당을 통한 소프트웨어 설계정보를 유지할 수 있게 한다.

적용할 패턴 정보는 설계 문서에 이미 반영되어 있으므로 UML 개발 툴이 이를 소스코드로 자동 생성함으로써 설계와

구현의 자동화가 가능하다. 요구사항분석 후 쓰임새도와 같은 설계 초기단계 UML 다이어그램 생성시 패턴요소를 검토하여 적합한 패턴과 유사패턴을 비교하여 최적 패턴을 선정하고 명시하면 설계의 복잡도를 낮추는 결과를 가져온다.

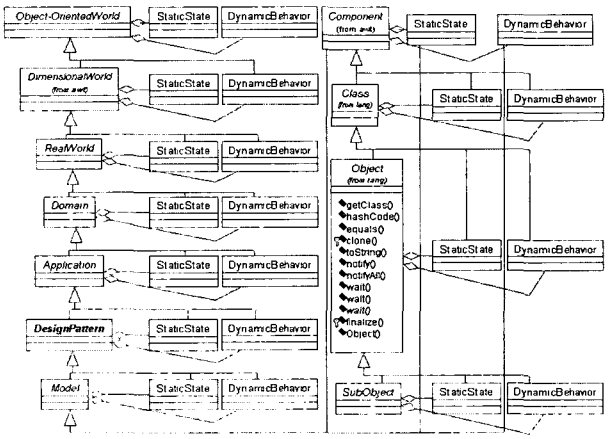
디자인 패턴이라는 용어는 "객체패턴"과 "패턴객체"라는 두가지 상반된 관점이 혼용되고 있는 실정이다. 첫째, 객체패턴(Object Patterns)은 패턴기반 하나의 객체 또는 객체집합의 유사속성으로서의 일정한 유형을 의미하고 그 구현방법으로서 패턴 저장소, 패턴 라이브러리 등의 패턴분류기가 모색된다. 둘째, 패턴객체(Pattern Objects)는 객체지향에 중점을 두고 하나의 객체 또는 객체집합의 고유속성으로서 패턴을 갖는 새로운 객체와 객체집합을 의미하고 그 구현방법으로서 기존 객체지향 모델링 언어를 이용하여 표준화된 새 객체지향 개발모델로서 패턴지향 모델을 만들기 위한 패턴 생성기, 패턴모델 생성기가 모색된다. 본 연구는 후자, 즉 패턴객체방법의 관점에서 기존 객체지향의 대안으로서의 디자인 패턴 개념을 수용하여 기존 개체-관계 데이터모델과 객체지향 데이터모델의 장점을 상속하고 패턴객체의 재사용성을 최대한 보장하는 관점에 기반하여 분석하고 설계하고자 한다.

구체적으로 본 연구의 제안방법은 TMDL 환경소프트웨어 도메인모델의 설계에 GoF 23개 디자인 패턴중 P_Singleton : Singleton Pattern(싱글톤 패턴), P_State : State Pattern(상태 패턴), P_Composite : Composite Pattern (합성기 패턴), P_Observer : Observer Pattern(관측기 패턴, P_Visitor : Visitor Pattern(방문자 패턴)을 적용하기 위해 정의를 내리고, 객체요소(OE : Object Elements), 객체행위요소(OBE : Object Behavior Elements), 객체관계성유형(ORT : Object Relationship Types), 기타 관계성 이름, 다중성, 역할, 책임, 그리고 디자인 패턴에 관한 부가사항 표기법을 명세화 하여 설계함으로써 소프트웨어 개발과정에서 불필요한 반복과 중복을 줄이고자 한다. 제안방법의 TMDL 응용도메인에 대한 적용 절차는 다음과 같다.

a. 설계단계에서 클래스이름에 적용패턴을 명시하여 패턴추적이 가능하도록 하여 따로 패턴저장소, 패턴공유기, 패턴언어들을 사용하여 사전지식이 필요해지는 것을 방지한다.
b. 요구사항분석단계에서 사전 선언된 적용패턴 원리에 따라 UML을 작성, 저장하고 부족한 패턴추적은 주석을 활용한다.
c. 설계모델을 컴파일 이후까지 보존하고 배포단계에서 제외시킨다.
d. 케이스 스터디 연구대상은 물의 순환환경 특히 유역과 하천에서 TMDL 환경소프트웨어모델 설계이다.

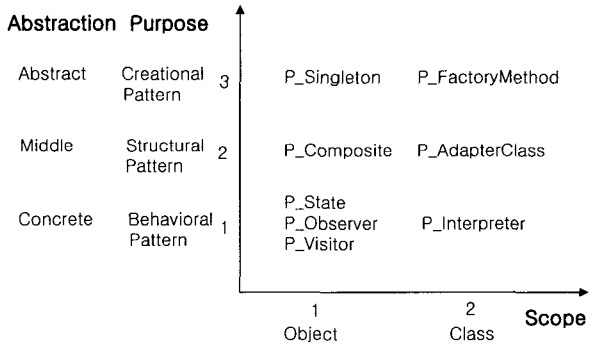
3. 디자인 패턴기반 객체지향 소프트웨어 설계 모델

디자인 패턴에 기반 한 객체지향의 기본 개념은 (그림 1)의 실세계 추상화수준, 적용범위와 층위에 따른 정적상태와 동적행위 패턴과 같이 객체를 실세계의 원소로 파악하여 객체모델로 실세계를 재구성하는 것이다.



(그림 1) 디자인 패턴의 실세계 추상화 객체모델

본 연구에서는 빠른 이해를 위해 각 개념적 층위를 편의상 클래스 계층구조화 하여 클래스도로 표기하였다. 여기서 디자인 패턴은 Model 층위의 일반화이며 Application을 상속한다. 동시에 디자인 패턴의 구체화이자 요소인 각 패턴들은 StaticState와 DynamicBehavior의 집단화로 구성되며 역시 Application을 상속하므로 Application의 개념적으로 속성과 메서드 들을 재사용할 수 있다.



(그림 2) GoF 디자인 패턴의 추상화, 목적과 적용범위

(그림 2) GoF 디자인 패턴의 추상화수준과 적용범위에 따른 분류와 기하, 위상 척도 개념도는 GoF 패턴에 기하, 위상 개념을 부여하여 패턴을 2차원 유클리드 평면에 사상하는 방식으로 가시화한 것이다. 먼저 세로축은 패턴의 적용목적과 패턴의 추상화수준 기준 축으로서, 추상적이며 주로 정태적 생성에 관계되는 패턴을 상단에, 구체성이 있고 동태적 행위에 관계된 패턴은 하단에 배치되도록 한 것이다. 가로축은 패턴이 위치하여 적용되는 범위 또는 층위 기준 축으로서, 상위클래스 수준은 우측에 하위클래스나 객체 인스턴스 수준은 좌측에 배치되도록 한 것이다.

(그림 2)에 분류한 23개의 GoF 패턴 중 본 논문에 적용된 5개 디자인 패턴을 TMDL 소프트웨어 모델의 구축에

적용하기 위해 다음과 같이 각각의 패턴객체에 이름을 선언하여 규정하고 그 적용대상문맥의 문제와 해결책의 쌍을 기술하고자 한다.

[패턴 1] P_Composite : Composite Pattern(합성기 패턴객체)

부분과 전체의 계층을 표현하기 위해 복합객체를 트리구조로 만든 것을 합성기 패턴객체라고 한다.

[문제] 비-합성(non-composite, atomic)객체를 다루는 것과 동일한 방법으로 다형성을 이용하여 객체들의 합성구조를 어떻게 다룰 수 있는가?

[해결] 합성 객체와 비-합성 객체를 위한 클래스를 정의하되 이들이 동일한 인터페이스를 구현(예, ≪Interface≫Basin_Component_P_Composite, (그림 7) 하도록 한다.

[패턴 2] P_Observer : Observer Pattern(관측기 패턴객체)

객체들 간의 종속성을 관리한다. 특정 객체에 종속된 모든 객체들은 한 객체의 상태가 변경되면 이를 자동으로 통지 받게 한 것을 관측기 패턴객체라고 한다.

[문제] 하나의 객체상태가 변화되었을 때 의존관계에 있는 다른 객체들에게 알려주고 자동적으로 업데이트 되도록 할 수 있는가?

[해결] 주제를 다루는 객체는 그것의 관측기 객체를 알고 있고 관측기 객체는 하위 관측기 객체를 관찰하되 "update()" 메서드(예, update(), (그림 8))를 가지고 한 Subject 객체의 공지되어야 할 변화를 업데이트하는 인터페이스를 구현한다.

[패턴 3] P_Singleton : Singleton Pattern(싱글톤 패턴객체)

클래스가 오직 하나의 인스턴스를 가지도록 한다. 모든 접근은 이 하나를 통해서만 가능하다. 소스코드 어디에서나 호출할 수 있는 전역변수와 같은 접근성을 가지면서도 전역변수같이 여러 개의 인스턴스를 갖지 않고 오직 하나의 인스턴스만을 유지하는 클래스를 싱글톤 디자인 패턴 또는 싱글톤 패턴객체라고 정의한다.

[문제] 클래스의 인스턴스가 단 하나만 허용된다. 어떻게 객체들의 전역적인 단일 접근 점을 만들 수 있는가?

[해결] singleton을 반환하는 클래스의 정적 메서드(static method : 예, P_Singleton, P_SingletonCounter(), (그림 5))를 정의한다.

[패턴 4] P_State : State Pattern(상태 패턴객체)

객체가 자신의 내부 상태에 따라 다른 행위를 하게 한다. 객체의 내부 상태변화에 따라 행위를 다르게 수행시키고자 할 때 각각의 상태를 속성으로 가지며 메서드는 공유하는 각각의 클래스를 만들고 이것들을 인터페이스 ≪Interface≫ 클래스에서 상속하도록 함으로써 특정 상태에 종속적인 행위를 다른 상태와 분리시켜 길고 복잡하며 쉽게 분리할 수 없는 If 또는 다중 선택 내포(switch nest)를 생략한 간결한 코드로 구성시키는 것으로 상태의 전이를 명시적으로 선언하여 여러 클래스에서의 공유가 가능하게 한 클래스 배치 방법을 상태 패턴객체라고 한다.

[문제] 객체의 행위가 상태에 의존하며 실행시의 상태변화에 따라 행위를 반드시 변화시킬 수 있는가?

[해결] Context 객체가 클라이언트들에게 관계되는 인터페이스를 정의하도록 하되 현재의 상태를 표현하는 ConcreteState 하위클래스의 인스턴스를 유지하게 하고, State 객체가 각각의 상태 Context로 구성된 행위를 포함하는 인터페이스를 정의하고 ConcreteState 하위클래스들로 하여금 각 Context 상태로 구성된 행위를 구현(예, getObserver(),(그림 6))하게 한다.

[패턴 5] P_Visitor : Visitor Pattern(방문자 패턴객체)

방문자 패턴객체는 객체구조의 요소들 위에서 수행되는 연산을 표현한다. 방문자는 새 연산이 수행되는 요소 클래스들의 변화 없이 새 연산을 정의할 수 있게 한다.

[문제] 관계된 인터페이스들과 함께 많은 클래스객체들을 포함하는 객체구조를 만들었는데 그 클래스들에 의존하는 객체들 위에서 연산을 수행시켜야하고 특히 이 연산은 자주 변화하는 경우에 어떻게 해야 하는가?

[해결] NodeVisitor라는 방문자 패턴객체와 이것을 상속하는 ConcreteVisitor를 만들고 방문자 클래스에 visit() 메서드(예, visitSimpleFlow(), (그림 9))를, 그리고 구체화클래스 요소에는 accept() 메서드를 선언하고 구현한다.

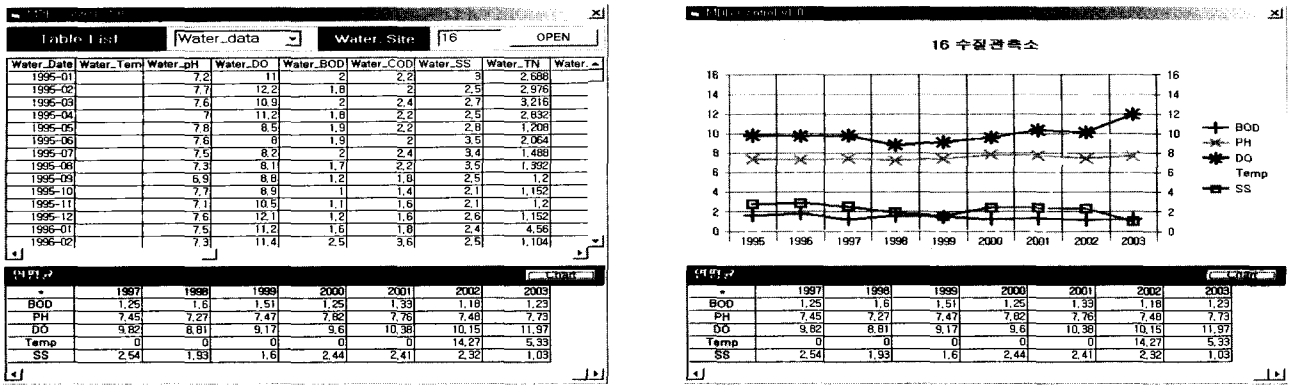
4. 디자인 패턴기반 오염총량제 소프트웨어 설계

4.1 응용 영역 분석 및 요구사항 분석

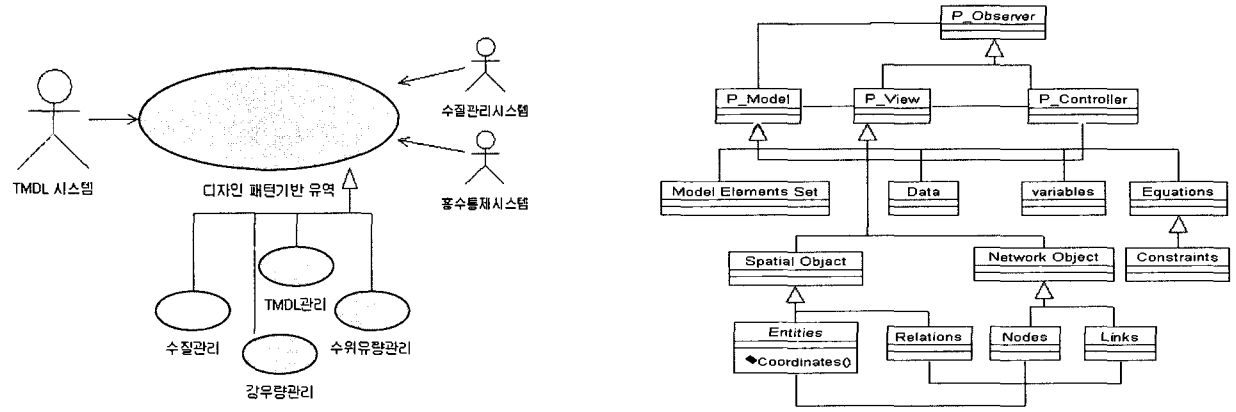
2004년을 기점으로 시행되는 TMDL 즉 오염총량제는 기존의 농도규제가 하천 및 호소 등의 모든 구역에서 수질기준이 지켜지더라도 하구에 이르러서는 자정능력한계를 초과하여 급속히 부영양화하는 문제를 해결하기 위해 유역단위로 기준배출량을 설정하여 지역의 지속가능한 개발을 유도하기 위해 도입된 오염부하량기준 수질관리정책이다.

오염총량제 소프트웨어에 대한 요구사항은, 제1차 TMDL 기간은 2004년부터 2010년이며, 이 기간 동안에는 우선 BOD(생물화학적 산소요구량)만을 TMDL 시행 대상물질로 한정하고 향후 COD(화학적 산소요구량), T-N(총질소), T-P(총인), DO(용존산소농도) 등이 대상물질로 추가 될 예정이므로 향후 대상물질이 추가될 경우에도 재사용 및 유지보수가 용이하고 인접한 도메인인 홍수통제시스템이나 수질관리시스템과의 다형성이 보장된 소프트웨어를 설계하는 것이다.

이와 같은 응용영역 및 요구사항에 대한 분석에 따라 우선 소프트웨어가 처리해야할 자료에 대한 스키마를 규정하는 작업을 하였고, (그림 3)은 수위 유량관측소, 수질관측소, TMDL관측소에 위치, 시간 종속되는 데이터로서 TMDL 소프트웨어에서 처리해야하는 TMDL 1차대상물질 BOD와 추가대상물질 자료 스키마의 속성을 일부 가시화한 것이다.



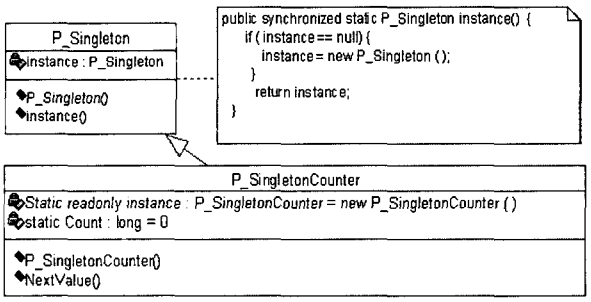
(그림 3) TMDL 1차대상물질 BOD의 자료 스키마



(그림 4) TMDL 유역 쓰임새도와 패턴객체

4.2 오염총량제 소프트웨어의 설계

디자인 패턴기반 객체지향 TMDL 환경 소프트웨어 모델 설계는 (그림 4)의 유역 컴포넌트 쓰임새도에 나타난 도메인분석과 요구사항분석 선정결과에 따라 유역 컴포넌트의 패턴객체 요소 모델의 패턴요소를 바탕으로 유역 누가유량 계산 컴포넌트에 싱글튼(Singleton) 패턴, 상태(State) 패턴, 합성기(Composite) 패턴, 관측기(Observer) 패턴, 방문자(Visitor) 패턴을 적용하는 순서로 진행되었다.



(그림 5) 유역 컴포넌트의 싱글튼 패턴

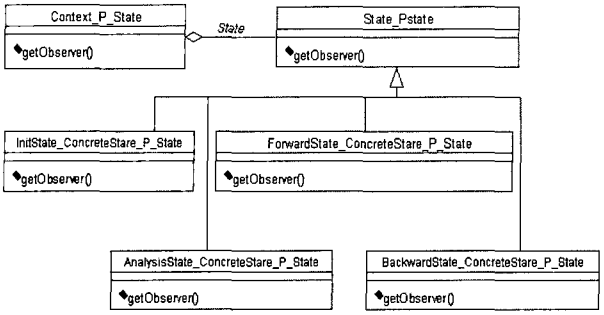
(그림 4)는 디자인 패턴기반 TMDL 유역 컴포넌트의 쓰임새도와 패턴객체 요소 모델을 설계문서화 표현한 것이다. 디자인 패턴기반 TMDL 유역 컴포넌트 모델 패턴요소는 관측기 패턴과 조절기 패턴의 구성요소인 뷰, 모델, 모델요소집합, 자료, 변수, 방정식(제약사항), 공간객체, 개체, 관계, 망객체, 노드, 링크로 구성되어 있다.

싱글튼 패턴객체를 상속한 싱글튼 카운터 객체는 (그림 5) 유역 컴포넌트에서의 싱글튼(Singleton) 패턴에서처럼 재귀적 수계순회를 통해 누가유량을 계산하게 한다. 싱글튼 카운터에 의한 수계순회는 정확히 마지막 수계번호에서 카운트를 종료하게 하며 복수 카운터의 생성에 의한 유량의 중복을 미연에 방지하여 재귀적 유역 분석알고리즘의 정확성을 보장한다.

TMDL은 기존 농도규제가 유역의 오염 임계치를 초과하는 환경오염에는 해결책을 제시하지 못하는 단점을 극복하기 위해 전체유역을 대상으로 오염발생량을 하천의 유량에 맞게 통제하는 것으로서 유역의 실시간 유량과, 오염발생량, 그리고 하천의 수용능력에 대한 방대한 수질모델링과 기존 홍수통제시스템, 수질관리시스템 같은 인접 응용도메인에 재사용될 수 있으며 실행시 외부 유역환경현상의 예상되지 않은 변화에 효율적으로 대응할 다형성까지 보장되는 설계가 요구된다.

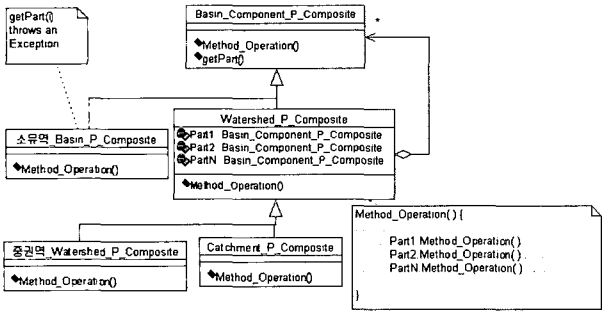
이때 특정 유역의 정확한 실시간 유량을 미리 계산하는 유량함수의 인수인 수계유량카운터는 하나의 수계를 정확히 한번만 순회하여 누가유량을 계산하도록 하지 않으면 최하류 유량에 최상류유량, 차상류 유량이 중복 계산된다.

유역누가유량을 재귀호출함수로 구할 경우 싱글톤패턴이 해당 유역유량의 중복계산을 방지한다.



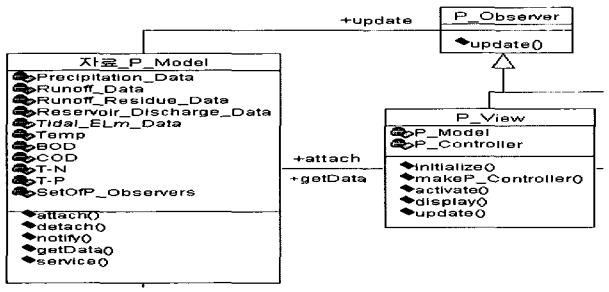
(그림 6) 자료, 관측자 컴포넌트의 상태 패턴

수부 또는 수체(Water-body)는 대기 중에서, 수계에서, 하천에서, 호수에서, 하구에서, 그리고 바다에서 6개의 상태 변화를 갖는다. 각각의 상태변화를 추적하되 수위-유량속성과 메서드를 유지하려면 자료, 관측자 컴포넌트에서는 (그림 6)과 같이 상태패턴을 적용하는 것이 유리하다.



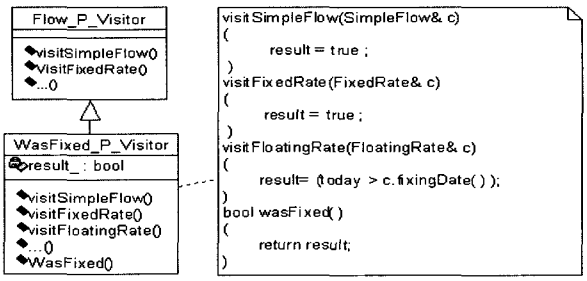
(그림 7) 유역 컴포넌트의 합성기 패턴

(그림 7)은 유역 컴포넌트를 구성하는 합성기 패턴이다. 본 연구에서는 ZEUS의 Polygon 시스템클래스를 상속하여 유역을 생성하였으며 구성수계의 위계적 구조를 유역, 대권역 수계, 중권역 수계, 표준유역 수계, 그리고 소유역(소유역_Basin)으로 5개 계층을 두어 조립함으로써 소유역_Basin_P_Composite의 예상할 수 없는 재분할과 재조립 시에도 유역 컴포넌트 전부를 수정하지 않고 소유역_Basin_P_Composite만을 분리 보수할 수 있게 하였다.

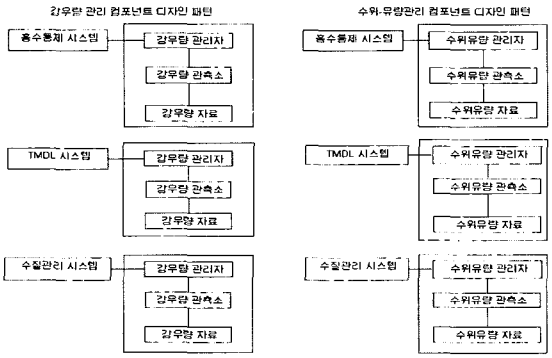


(그림 8) 자료와 관측자 컴포넌트의 관측기 패턴

(그림 8) 자료와 관측자 컴포넌트에서의 관측기(Observer) 패턴은 관측자(강우량 관측자, 수위-유량 관측자, 수질 관측자, TMDL 관측자, 홍수통제 관측자)와 사용자 인터페이스 뷰, 그리고 자료에 적용된다. 메서드 update()는 "뷰"에 상속되어 "뷰"가 하나의 관측자가 되어 자료모델로부터 데이터를 받아들이고 관측기 클래스는 그 자료의 사용자 인터페이스로의 송출을 업데이트 한다.



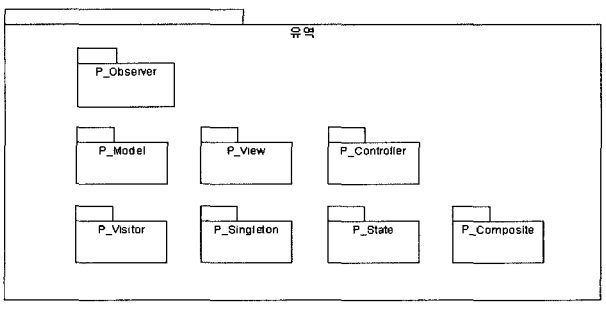
(그림 9) 자료, 관측자 컴포넌트의 방문자 패턴



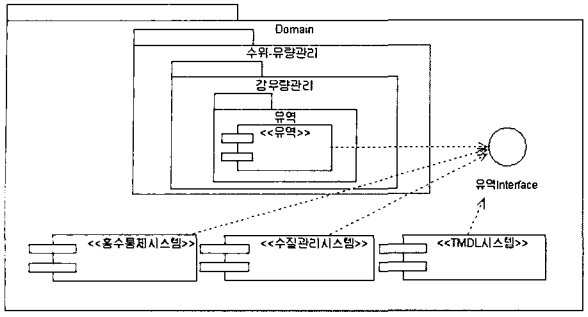
(그림 10) 디자인 패턴 모듈 재사용 및 다형성 패키지도

유량계산에서 하천계수(CN : Curve Number) 제약사항(Equation Constraints)의 적용을 위한 방문자 패턴이다. (그림 9) 자료, 관측자 컴포넌트에서의 방문자(Visitor) 패턴은 객체 구조에 속한 요소에 수행될 오퍼레이션을 정의하는 객체이다.

(그림 10)은 도메인을 달리하는 홍수통제, TMDL, 수질관리 시스템에서 강우량계산 컴포넌트, 수위-유량계산 컴포넌트가 어떻게 생성되고, 어디에 배치되어 재사용되었는지를 전체적으로 나타낸 것이다.



(그림 11) 유역컴포넌트 내부의 디자인 패턴 배치도



(그림 12) TMDL 유역 컴포넌트 재사용 및 다형성

(그림 11) 유역컴포넌트 내부의 디자인 패턴 배치도를 보면 제안한 TMDL 소프트웨어의 유역컴포넌트는, 이를 구성하는 P_Observer, P_Visitor, P_Singleton, P_State 및 P_Composite 디자인 패턴의 정태적 배치와 능동적 행위에 의해 상이한 위치, 상이한 시간의 제약사항을 감안하여 수계에 할당된 오염부하허용량을 산출하여 오염물질 제공자들에게 목표수질을 달성하기 위한 허용한계를 구체적으로 할당해줘야 하는 책임을 갖게 되었다.

(그림 12) TMDL, 수질관리, 홍수통제의 유역 컴포넌트 재사용 및 다형성은 설계한 TMDL 소프트웨어의 유역컴포넌트가 유역인터페이스를 통해 홍수통제시스템과 수질관리시스템에 다형성을 갖는 공유구조를 갖는 것을 나타냈다.

GoF 패턴 중 P_Observer, P_Visitor, P_Singleton, P_State 및 P_Composite등 5개 패턴을 TMDL 유역, 자료, 그리고 관측자 컴포넌트 설계에 적용한 결과 TMDL의 복잡한 요구사항에도 강인한 책임수행능력을 보이므로 유지보수가 용이하고 인접 도메인으로의 확장성이 자유로운 다형성이 보장된 시스템 설계를 얻을 수 있었다.

〈표 2〉 디자인 패턴기반 소프트웨어 개발 모델

대상 \ 속성	1) 패턴문서화	2) 패턴분류	3) 패턴책임할당	4) 메타패턴	5) 패턴규칙	6) 패턴저장소	7) 패턴자동검색	8) 패턴통합	9) 패턴언어 종속극복
Gamma등[7]	N/A	SCB	N/A	N/A	N/A	N/A	N/A	N/A	C, S
Gamma[8]	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	J
Booch등[5]	o	SB	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Budinsky등[4]	N/A	SB	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Eden등[6]	N/A	CB	N/A	o	N/A	N/A	N/A	N/A	J
Sefica등[10]	N/A	SCB	N/A	N/A	o	N/A	N/A	N/A	N/A
Stephen등[11]	N/A	CB	N/A	N/A	N/A	N/A	N/A	o	N/A
김운용등[1]	N/A	SCB	o	o	N/A	o	o	N/A	N/A
Larman[3]	N/A	N/A	o	N/A	N/A	N/A	N/A	N/A	N/A
본 연구	o	SCB	o	o	o	N/A	o	o	JBC#

공통범례 : o : 주요방법, N/A : 강조되지 않음., 1) 패턴문서화 : 공통, 2) 패턴분류 : S : 구조적, C : 생성적, B : 행위적, 3) 패턴책임할당 : 공통, 4) 메타패턴 : 공통, 5) 패턴규칙 : 공통, 6) 패턴저장소 : 공통, 7) 패턴자동검색 : 공통, 8) 패턴통합 : 공통, 9) 패턴언어종속극복 : J : JAVA기반 B : VB .net 기반, C# : C#기반, C : C기반, S : SmallTalk기반

5. 비교 및 검토

본 연구는 Gamma등[7]의 23개 패턴분류방법을 기반으로 패턴의 상대적 위치를 가시화하여 설계에 적용되는 패턴의 기하와 위상을 쉽게 파악할 수 있게 하였다. 또한 Larman[3]에 의해 제안된 패턴책임할당 개념을 받아들여 소프트웨어의 각 컴포넌트별로 책임을 할당하여 과업수행의 중복을 배제하였다. 패턴생성을 가능하게 하는 메타패턴에 대해서는 패턴객체와 객체패턴의 개념을 명확히 구분하여 혼용을 방지하게 하고 패턴적용에 일관성을 유지할 수 있게 하였다. 특히, 이 연구에서는 패턴객체의 관점에 서서 디자인 패턴개념을 받아들이고 "[문제]-[해결]"쌍으로 패턴을 규정하고 각 패턴규칙이 적용되게 하였다.

패턴저장소를 따로 두지 않고 클래스 이름에 패턴이름을 첨부함으로써 설계다이어그램과 소스코드가 어느 패턴을 적용할 것인지를 명시적으로 선언하여 패턴의 추적이 용이하게 하였다. 패턴자동검색은 시작패턴, 연결패턴 및 협력패턴을 설계단계에서 사전 선정된 패턴순회방식에 의해 순회하게 하여, 다음에 어떤 패턴을 적용할지를 두고 시간을 소모하지 않게 했다. 관측기 패턴 "P_Observer"에 그 연결패턴, 협력패턴인 "자료_P_Model"과 "P_View"가 통합되도록 하였다.

개발플랫폼간의 호환성을 보장하는 VB .net을 사용함으로써 본 연구는 JAVA와 C의 장점을 살려 코드의 재사용이 이종 플랫폼 간에도 이루어질 수 있게 하였다.

이를 <표 2> 디자인 패턴기반 소프트웨어 개발모델 주요방법 비교표에 요약하였다.

객체지향 모델링에서 컴포넌트기반 방법은 시간 및 공간 복잡성이 있는 문제를 나누어 연산능력을 높이는 것이다. 컴포넌트기반 방법의 많은 성과에도 불구하고 디자인 패턴과 그 표준화로 컴포넌트기반 방법의 재사용성을 높이는

연구가 필요하였다.

디자인 패턴을 표준화하는 방법으로 성급하게 메타패턴언어나 패턴저장소를 구축하는 방법은 오히려 소프트웨어 개발에 패턴을 적용하는 것을 더욱 복잡하고 어렵게 만들 수가 있으므로 본 연구는 과도하게 추상화된 메타패턴언어나 부가적인 패턴저장소를 두지 않고도 적용패턴을 검색, 추적할 수 있도록 설계단계에서 클래스이름에 패턴이름을 명시하는 방법을 제안하여 이를 공간적, 시간적 변화를 패턴화하여 표현하기 가장 곤란한 분야중 하나인 오염총량제, 즉 TMDL의 주요 응용도메인인 유역 컴포넌트의 설계에 적용하였다.

6. 결론 및 향후연구과제

이 연구는 GoF 디자인 패턴을 객체지향 TMDL 환경소프트웨어에 적용하여 재사용 가능한 소프트웨어 모델을 설계하는 방법을 제안하였다.

제안방법은 소프트웨어의 설계에 GoF 23개 디자인 패턴중 합성기, 관측기, 싱글톤, 상태, 그리고 방문자 패턴을 적용하기 위해 기하, 위상개념을 부여하여 규정하고, 디자인 패턴에 관한 부가사항 표기법을 명세화 하여 디자인 패턴기반 객체지향 TMDL 소프트웨어 모델을 설계함으로써 서로 다른 환경 관련 소프트웨어에 공통적으로 사용되는 유역 컴포넌트의 개발과정에서 불필요한 반복과 중복을 줄이는데 기여하였다.

향후연구과제는 설계된 TMDL 소프트웨어를 하천뿐만 아니라 대기와 해양 TMDL에도 확장시킨 디자인 패턴기반 시공간 유량, 수질 통합 관리시스템을 설계하는 것이다.

참 고 문 헌

[1] 김운용, 최영근, 디자인 패턴지향 소프트웨어 개발 지원도구, 정보과학회, 제29권 제8호, 2002.

[2] 환경부, "환경정책기본법, 환경정책기본법시행령", 환경부, 2002.12.26. 개정.

[3] Larman, Craig, 박수희외 역, Applying UML and Patterns, Hong Reung Science Pub Co., 2003.

[4] Budinsky, F., M., Finnie, Vlissides, John, and P.. Yu, Automatic code generating from design Patterns, IBM systems Journal, 35(2), 1996.

[5] Booch, Grady, Rumbaugh, James E., Jacobson, Ivar, The Unified modeling language user guide, Addison-Wesley, Reading, MA., 1999.

[6] Eden, A., A., Yehudai and J., Gil, Precise specification and automatic application of design patterns, Automated software Engineering, 1997. proceedings of 12th IEEE International Conference, 1997.

[7] Gamma, Erich, Helm, Richard, Johnson, Ralph, Vlissides, John, Design Patterns, Elements of reusable object-oriented software, Addison-Wesley, 1995.

[8] Gamma, Erich, Applying design patterns in JAVA, JAVA Report, 1999.

[9] Kim, Kangjoo, Ji Sun Lee, Chang-Whan Oh, Gab-Soo Hwang, Jinnsam Kim, Sungku Yeo, Yeongkyoo Kim, Seongmin Park, "Inorganic chemicals in an effluent-dominated stream as indicators for chemical reactions and streamflows," Journal of Hydrology, March, 2002.

[10] Sefica, M., A., Sane and R., Campbell, Monitoring compliance of a software system with its higher-level design models, Proceedings of the 18th International conference of software engineering ICSE, 1996.

[11] Stephen, S., S., Yau and Ning Dong, Integration in component-based software development using design patterns, Computer software and design patterns, Computer software and applications conference, 2000. COMPSAC 2000, The 24th Annual International conference, 2000.

김 형 무

e-mail : kimhyungmoo@chonbuk.ac.kr

1987년 서울대학교 국민윤리교육과(학사)

1990년 서울대학교 대학원 국민윤리교육과(석사)

1997년 미국 뉴욕 롱아일랜드대학교 컴퓨터교육(석사)

2003년 전북대학교 공과대학 컴퓨터공학과 과정수료(박사)

2001년~현재 전북대학교 컴퓨터공학과 시간강사

관심분야 : 객체지향모델, 지리정보시스템, 데이터베이스, 소프트웨어공학, 원격탐사, 텔리매틱스, 유비쿼터스 등

곽 훈 성

e-mail : hskwak@moak.chonbuk.ac.kr

1970년 전북대학교 전기공학과(학사)

1979년 전북대학교 대학원 전기공학과(공학박사)

1981년~1982년 미국 텍사스주립대학 연구교수

1988년~1990년 대한전자공학회평의원 및 전북지부장

1994년~1995년 국가교육연구전산망추진위원

1997년~1998년 전주영상축전조직위원장 및 전북대 영상산업 특성화사업단장

1998년 과학기술법령정비정책위원

1998년~현재 전북대학교 전자정보공학부 교수

1997년~현재 (사)영상산업연구센터 대표

1999년~현재 조달청우수제품(정보통신) 심사위원

현재 전북대학교 영상산업(학부전공) 및 영상공학과(대학원) 주임교수, 한국게임학회 종신회원

관심분야 : 영상처리(위성정보), 인공지능, 컴퓨터비젼, 컴퓨터게임 등