한국지리정보학회지 9권 3호 : 193~206(2006)

# 객체지향 설계 유형에 의한 지오센서 인터페이스 구현

백정호[1] · 이홍로[1※]

# Implementation of Geosensor Interface using Object Oriented Design Pattern

Jeong-Ho BAEK[1] · Hong-Ro LEE[1※]


## 요 약

본 논문은 객체지향 디자인 패턴에 기반하여 지오센서 모델을 설계하고, JBuilder를 이용하여 지오센서 인터페이스 시스템을 구축하는 효율적인 방법을 제안하고자 한다. 이와 같은 지오센서 기술은 다량의 현장 정보를 실시간으로 원거리에서 네트워크를 구성하여 관리할 수 있는 지리정보시스템의 새로운 연구 분야가 될 것이다.

지오센서 네트워크와 지리정보시스템을 접목시키는 기술은 나날이 증가되고 있는 지리정보의 기능을 충족시키는데 필요한 요소이다. 따라서 본 논문은 객체 지향 지오센서 미들웨어를 개선하기 위해 GoF 디자인 패턴 활용하고자 한다.

본 논문은 객체지향 디자인 패턴의 활용함으로써 재사용, 모듈화, 그리고 수정이 용이한 소프트웨어를 개발할 수 있는 최적화된 지오센서 인터페이스의 구현에 기여할 것이다.


주요어 : 객체지향, 디자인패턴, 지오센서, 인터페이스, 지리정보시스템, JBuilder


## ABSTRACT

This paper proposes The Efficient method that should design Geosensor model based on object oriented design pattern and implement the Geosensor network interface system using JBuilder. Such as geosensor technology will be to a new research paradigm of GIS which can manage a great quantity of field information by means of constructing the real time remote sensing network.

The technology that integrates object oriented design pattern Geosensor interface network with GIS will be necessary elements that satisfy the function of GIS increasing day by day. Therefore, we would like to utilize GoF design pattern in order to change for the better object oriented Geosensor middleware.

This paper shall contribute to implementing the optimal Geosensor interface that can develop reusable, modular and modifiable software by using the object oriented design pattern


*KEYWORDS : Object Oriented, Design Pattern, Geosensor, Interface, GIS, JBuilder*

2006년 8월 29일 접수 Received on August 29, 2006 / 2006년 9월 22일 심사완료 Accepted on September 22, 2006

1 군산대학교 자연과학대학 컴퓨터정보과학과 Department of Computer Information Science, Kunsan National University

※ 연락저자 E-mail : leehongro@kunsan.ac.kr

## 서 론

현재까지 다양한 분야에 필요한 디자인 패턴(Design Pattern)들이 발견되어지고 그 활용을 증명하는 연구들이 진행되고 있다(Gamma 등, 1995; T. H. Ng 등, 2006). 하지만 본 논문의 센서기반 지리정보 인터페이스를 개발하는데 있어서 디자인 패턴을 체계적으로 적용하기 위한 연구는 많이 부족한 상태이다. 그 이유는 디자인 패턴이 일부 시스템이나 특정 목적에 국한된 소프트웨어 설계의 부가적인 수단으로써 활용되며, 그에 대한 명세와 활용은 주로 개발자의 수작업에 의존하여 다양한 형태로 적용되고 있기 때문이다. 그 결과 지리정보시스템(GIS: geographical information system)에서 일관된 형태의 분석과 활용이 어렵고 오류 발생 빈도가 높을 뿐 아니라 프로그램 개발에 많은 시간이 요구된다. 이러한 문제점을 해결하기 위해 지리정보시스템 소프트웨어 개발 시 디자인 패턴의 활용을 위한 체계적이고 효과적인 접근 방법이 요구된다. 본 연구에서 설계 및 구현된 지오센서 기반 인터페이스는 환경 감시/예측/예방, 헬스 케어, 텔레매틱스 등에 응용할 수 있다.

소프트웨어 개발의 안정성과 재사용성을 증가시키기 위해 개발 경험의 효율적인 접근 방법은 체계적이고 구체적인 설계 정책을 이끌어 낼 수 있어야 한다. 그러나 특정 목적을 달성하기 위한 방법만을 제시하거나 디자인 패턴 정보는 소프트웨어 시스템에 적용될 때 관련 패턴의 정보가 소멸되는 경향이 있는 문제가 있다. 그러므로 디자인 패턴 정보를 일관되고 효율적으로 시스템에 적용하여 시스템의 오류 발생 비율을 줄이고 분석능력을 향상시키는 것이 필요하다(김운용과 최영근, 2002; 백정호 등, 2005; Gordilo 등, 1997, 1999; Camara, 2001).

본 연구에서는 GoF 디자인패턴 23가지 중 Creational Pattern, Structural Pattern, Behavioral Pattern 3가지 측면에서 5가지 (Abstract Factory, Factory Method, Template Method, Composite pattern, adapter pattern)를 활용하여 설계하였으며(Gamma 등, 1995; Yacoub 등, 2004), 지오센서 인터페이스의 센싱 부분은 자료 복구(Data Discovery)를 제공하기 위한 일반적인 센서 정보를 제공(Stefanidis와 Nittel, 2005), 센서 측정 데이터의 처리와 분석 기능 지원, 측정된 데이터의 지리적 위치 제공, 정확성 등의 높은 성능 제공, 센서에 관한 기본적인 속성 기록 및 추정 등의 표준에 맞춰 실시간 데이터의 획득하는 인터페이스 구현에 목적을 두었다.

복잡한 GIS 구조에 대해 디자인패턴의 적용으로 GIS 인터페이스를 설계하고 지형정보 위에서 센서데이터의 결과를 모니터링 할 수 있는 지오센서 인터페이스의 융합, 컴포넌트 기반 설계유형에 의해 지오센서 인터페이스에 수치지도를 이용한 벡터기반 인터페이스와 위성영상을 이용한 래스터기반 인터페이스 그리고 논문에서 제안하는 센서데이터를 이용한 센서기반 인터페이스로 설계 및 구현하고자 한다.

Gamma 등은 디자인패턴의 정의와 분류 그리고 쓰임에 대해서 상세히 정의 하였으며(Gamma 등, 1995), Gordillo 등은 디자인패턴을 적용하여 GIS어플리케이션의 구조를 일반화 하였다(Gordilo 등, 1997,1999). 양인태와 최영재는 래스터기반인 위성영상으로 GIS응용시스템을 개발하였으며(양인태와 최영재, 2004), Xiong zhan wu 등은 GIS인터페이스 개발에 디자인패턴을 적용하여 Java기반으로 구현하였고(Xiong 등, 2006), 이재봉 등은 객체지향 디자인패턴 기반으로 온도 변화 탐지에 대한 인터페이스 설계 및 구현 하였다(이재봉 등, 2005).

본 연구에서는 단순한 패턴적용과 RF형식의 유비쿼터스 센서 네트워크(USN: ubiquitous sensor network) 구성하지 못하고 단순 센서데이터의 활용(Stefanidis와 Nittel, 2005)을 위한 센서와 서버와의 커뮤니케이션 인터페이스, 기존의 인터페이스에 플러그인 형식의 끼워 넣기

인터페이스로 개발에 한계를 두었다.

디자인 패턴을 이용하면 좋은 설계나 아키텍처를 재사용하기 쉬워지고, 입증된 기술을 디자인 패턴으로 표현하면 유용하게 사용할 수 있다. 디자인 패턴은 설계자로 하여금 재사용을 가능하게 하는 설계를 선택하고, 방해하는 요소는 배제하도록 도와준다. 이미 만든 시스템의 유지보수나 문서화도 개선해 주고, 패턴화를 통해서 클래스의 명세를 정확하게 하며, 객체간의 상호작용 또는 설계의 의도 등을 명확하게 정의할 수 있다. 즉, 소프트웨어 개발에서 디자인 패턴의 활용은 생산성을 향상시키고 재사용과 신뢰성을 가질 수 있는 시스템을 구축할 수 있게 하며, 소프트웨어 개발에서 발생되는 문제 해결을 위한 좋은 수단으로 활용되고 있다.

본 논문은 객체지향 디자인 패턴기반 지오센서 모델을 분석 및 설계하고, 지오센서 인터페이스를 구축 및 검토하며, 마지막으로 디자인 패턴 설계에 의한 센서기반 지리정보 인터페이스의 결론을 내리고, 향후 지오센서 인터페이스의 연구방향에 대해 기술하고자 한다.

## 객체 지향 디자인패턴 기반 지오센서 모델

### 1. 객체 지향 디자인 패턴

객체지향 소프트웨어의 구성은 여러 개의 작은 구조들의 집합으로 구성될 수 있으며, 소프트웨어 개발 시스템 내에서 계층적 구조와 객체 사이의 상호 작용은 자주 반복되는 객체지향 설계 구조이다. 디자인 패턴은 소프트웨어 공학자의 경험에 의하여 이루어지며, 이 작은 구조를 객체지향 디자인 패턴이라 한다. 이 디자인 패턴은 디자인 문제에 대한 추상적인 해결방법을 제시하고 있다. 이 객체지향 소프트웨어에 필요한 디자인 패턴요소를 기술하는 패턴으로 GoF 패턴이 있다.

객체지향 소프트웨어 개발의 주된 요소로 시스템을 구성하는 객체들의 효율적인 구조와 행위를 표현하기 위한 객체의 책임 할당 정책을 들 수 있다. 이러한 책임 할당에 대한 기본 원칙을 정리하는 패턴이 GRASP(general responsibility assignment software patterns) 패턴이다(Gamma 등, 1995; Yacoub 등, 1994).

패턴들의 검색 및 관리의 효율성을 증가시키기 위한 패턴을 분류하는 법들에 대한 다양한 연구가 이루어지고 있다. 소프트웨어의 뼈대를 형성하는 패턴에 대한 부분을 구조패턴으로 정의하고, 각각의 컴포넌트를 형성하는 디자인패턴과 프로그램 언어를 이용한 코드 명세에 대

**TABLE 1.** GoF design pattern space

| | | 목 적 | | |
|---|---|---|---|---|
| | | **생성 패턴** | **구조 패턴** | **행위 패턴** |
| **범주** | **클래스** | Factory Method | Adapter(class) | Interpreter<br>Template Method |
| | **객체** | Abstract Factory<br>Builder<br>Prototype<br>Singleton | Adapter(Object)<br>Bridge<br>Composite<br>Decorator<br>Facade<br>Flyweight<br>Proxy | Chain of Responsibility<br>Command<br>Iterator<br>Mediator<br>Memento<br>Observer<br>State<br>Strategy<br>Visitor |

한 코드 패턴을 정의할 수 있다. 아래의 표1은 GoF 패턴을 목적과 범주에 따라 구분하여 정의하였다.

본 논문에서는 layer 관리에 필요한 구체적인 클래스를 지정하지 않고 서로 독립적으로 관련 있는 객체들의 집합을 생성하는 부분 때문에 생성패턴의 Abstract Factory를 사용, 상위 객체로부터 상속을 받아 하위 클래스에서 재정의 함으로서 객체를 생성하기 위해 Factory Method를 사용하고, point, line, polygon등의 기본 객체들을 복합적으로 사용하기 위해 복합객체를 정의하여 지리정보시스템에 적용하기위해 구조패턴의 Composite Pattern을 사용, 지리정보시스템을 위해 일치하지 않는 맵 인터페이스와 센서 인터페이스를 데이터가 상호 연동되도록 중간 역할을 하기 위해 Adapter Pattern을 사용하며, 소스 코드의 중복을 피하기 위해 모든 하위 클래스에서 동일하게 나타나는 모듈을 추출해 하나의 클래스 내에 정의하기위해 행위패턴 중 Template Method를 사용한다.

## 2. 디자인 패턴 기반 지오센서 모델

### 2.1 지오센서 개요

필요한 모든 것(곳)에 전자태그를 부착하고 이를 통하여 사물의 인식정보를 기본으로 주변의 환경정보(온도, 습도, 오염정보, 균열정보 등)까지 탐지하여 이를 실시간으로 네트워크에 연결하여 정보를 관리하는 것으로 궁극적으로 모든 사물에 컴퓨팅 및 커뮤니케이션 기능을 부여하여 Anytime, Anywhere, Anything 통신이 가능한 환경을 구현하기 위한 것이다(Stefanidis와 Nittel, 2005). USN은 먼저 인식정보를 제공하는 전자태그를 중심으로 발전하고 이에 센싱 기능이 추가되고 이들 간의 네트워크가 구축되는 형태로 발전할 것이다. 이러한 USN의 개념이 지리정보시스템에 적용되어 지리현상에 센서를 통한 데이터 활용하는 부분을 지오센서라 한다.

### 2.2 디자인 패턴 기반 지오센서 모델 설계

#### 1) 추상 팩토리(Abstract Factory)

지오센서 인터페이스 중에서 지도 화면에 대해 지오센서를 위한 추상 팩토리 패턴의 객체 생성과 기능을 다음과 같이 규정한다.

[패턴] ***LayerFactory*** : 지오센서를 위한 추상화 팩토리 패턴 객체

[정의] Layer 관리에 필요한 구체적인 클래스를 지정하지 않고 서로 독립적으로 관련 있는 객체들의 집합을 생성한다.

[기능] 추상 팩토리인 LayerFactory의 인터페이스를 정의한다. 생성할 집단 EDIT, VIEW와 DATA 각각에 대해 구체 팩토리 클래스를 정의한다.

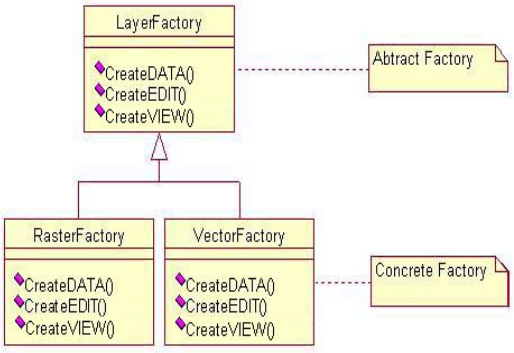


FIGURE 1. 지오센서 인터페이스의 자료관리 중 지도화면을 위한 추상 팩토리

위 그림 1에서 공장의 역할을 수행한 것은 LayerFactory 클래스이다. 이 클래스는 추상 클래스로서 하위 클래스(RasterFactory, Vector Factory 등)에서 실제로 객체를 생성하는 메소드를 구현한다. 이렇게 특정 객체가 다른 객체들을 전문적으로 생성하는 패턴을 추상 팩토리라 한다.

추상 팩토리 패턴은 객체들을 집단적으로 생성할 때 사용하는 패턴으로서 실제 반환되는

객체의 클래스 타입을 알 필요가 없이 원하는 객체를 생성하는 인터페이스를 제공한다. 이 인터페이스는 객체를 만들어내는 공장(Factory)이라고도 부른다.

객체 생성 패턴은 GoF 패턴 중 하나로써, 객체 생성 패턴을 위해 본 논문에 적용된 팩토리는 두 개의 인터페이스를 가진다. 첫째, 추상 팩토리를 위한 인터페이스로 다양한 제품 타입을 생성하기 위한 것과 둘째, 팩토리에 의해 생성된 제품을 위한 인터페이스 클래스로서 이 인터페이스는 팩토리 객체에 의해 생성된 모든 다양한 제품들을 위한 인터페이스이다.

제품 라이브러리는 인터페이스에 접근하며 구현부가 드러나지 않는다. 추상 팩토리로 부터 상속 관계에 있는 구체 팩토리(Concrete Factory)가 제품을 생성한다. 제품의 교체는 구체 팩토리 클래스의 객체 생성 부분의 변경만으로 가능하다. 구체 팩토리에 의해 제품이

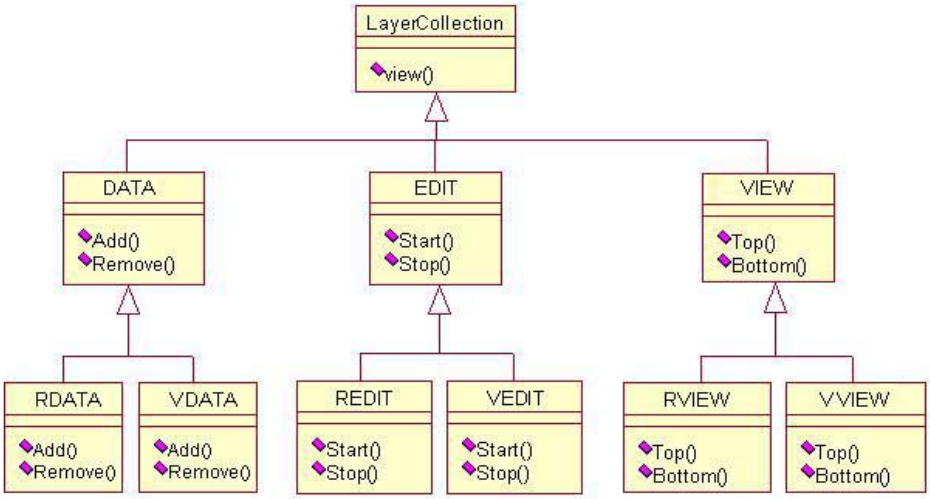


FIGURE 2. 지오센서 인터페이스를 위한 지도화면 관리 클래스 계층

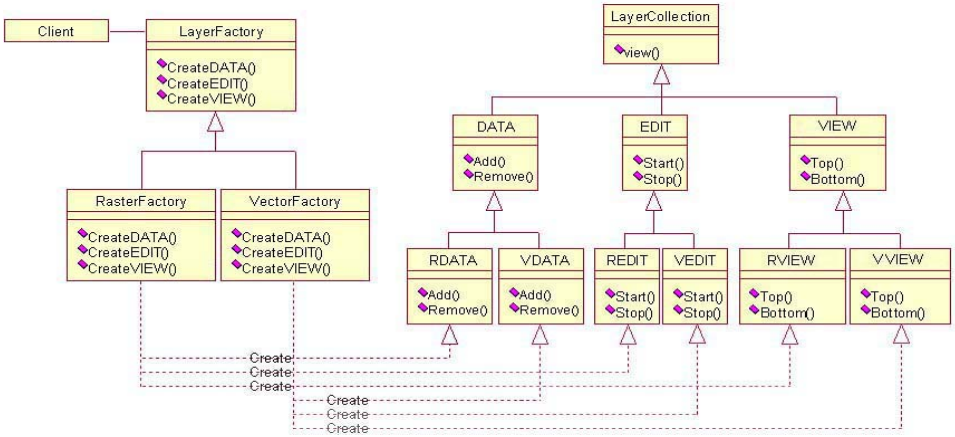


FIGURE 3. 지오센서 인터페이스를 위한 지도화면 관리 팩토리와 프러덕트

생성되므로 여러 제품이 혼합되는 것이 기본적으로 방지된다.

그림 2는 지오센서 인터페이스를 위한 지도 화면 관리를 위한 자료구조를 보인다. Layer Collection은 지도화면의 레이어의 집합을 위한 추상 클래스이다. LayerCollection에 필요한 지리정보 자료는 상속관계에 있는 DATA, EDIT와 VIEW 클래스에 의해 관리된다. 래스터 자료형(data type)을 RDATA, REDIT와 RVIEW에 의해 정의되고 벡터 자료형은 VDATA, VEDIT와 RVIEW에 의해 정의된다.

그림 3에서 LayertFactory는 객체 생성을 위한 인터페이스를 집단적으로 선언한다. 즉, 클라이언트가 사용할 객체를 생성하는 메소드인 "CreateDATA()" 등을 선언하며 이 메소드는 DATAProduct 객체를 반환한다.

CreateDATA() 등과 같은 관련 메소드는 생성해야 할 객체 종류의 숫자대로 여러 개가 존재할 수 있다. 구체 팩토리는 추상 팩토리에서 상속된 CreateDATA() 메소드를 오버라이딩하여 DATAProduct 클래스로부터 상속받은 Product 객체를 생성하는 코드를 구현한다.

그림 3에서 RasterFactory와 VectorFactory가 구체 팩토리이다. 각 CreateDATA() 메소드는 프러덕트인 RDATA, VDATA 객체들을 생성하여 트리 형(Tree type)으로 반환한다.

### 2) 합성 패턴(Composite Pattern)

지리정보시스템을 위한 인터페이스 중에서 지도 화면에 대해 원격 탐사를 위한 합성 패턴의 객체 생성과 기능을 다음과 같이 규정한다.

[패턴] *CompositeGeo* : 지리정보시스템을 위한 부분-전체 관계의 복합객체

[정의] 객체의 구성에 관계하며 동적으로 정의할 때 부분 객체가 전체 객체를 대표할 때 사용한다.

[기능] Point, Line, Polygon등의 기본 객체들을 복합적으로 사용하기 위해 복합객체를 정의하여 지리정보시스템에 적용한다.

합성 패턴을 사용하려면, 기본 클래스와 이를 포함하는 컨테이너 클래스를 구분하지 않고 처리하는 재귀적 합성을 이용할 수 있다. 재귀적 합성이란, 컨테이너 클래스가 기본 요소뿐만 아니라 기본 요소를 포함하고 있는 컨테이너 자체도 포함할 수 있음을 의미한다. 즉, 폴더가 파일 및 폴더도 포함할 수 있다는 사실을 생각하면 된다.

위의 그림 4에서 합성 패턴의 가장 중요한 요소는 기본 클래스와이들의 컨테이너를 모두 표현할 수 있는 하나의 추상화 클래스를 정의

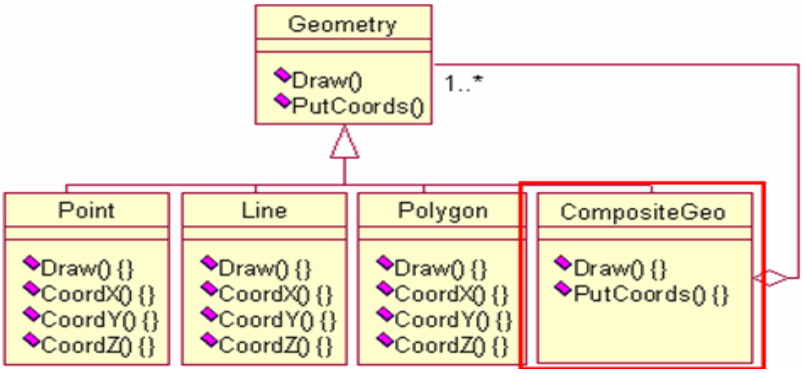


FIGURE 4. 지오센서 인터페이스의 지도화면 관리를 위한 합성 패턴

하는 것으로, 그림 4에서 본다면 추상클래스로 Geometry 클래스를 정의하는 것이다. Geometry 클래스의 기본 클래스 오퍼레이션인 Draw()와 PutCoords()가 정의되어 있다. 즉, 복합 객체의 합성 요소들을 다룰 오퍼레이션들까지 정의하고 있다.

Geometry 클래스의 서브클래스인 point, line, polygon은 기본적인 지리정보 객체들로서 이들 클래스는 Draw() 오퍼레이션을 구현하나, 기본 지리정보들은 어떠한 요소 지리정보를 포함하지 않으므로 이들 서브클래스는 합성 요소를 다루는데 필요한 오퍼레이션을 구현하지 않는다. CompositeGeo 클래스는 Geometry 객체들과 집합 관계가 성립되어 있어, Geometry 객체들을 포함할 수 있는 복합 객체로 정의되어 있다. CompositeGeo 클래스도 Draw() 오퍼레이션을 구현하고 있다. 이는 여러 지리 정보 요소가 포함된 복합 그림도 실제로 그려지는 기본 행위를 제공하고 있기 때문이다. CompositeGeo 클래스에는 Draw() 이외에 PutCoords()가 실제로 구현된다.

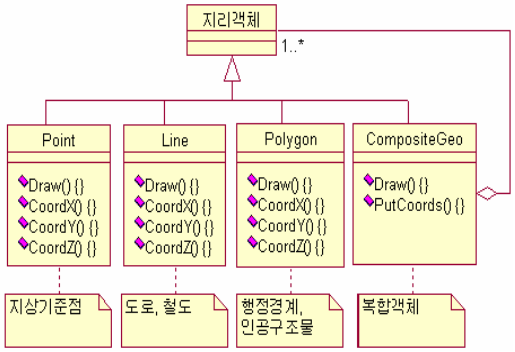


FIGURE 5. 지오센서 인터페이스의 Geometry 적용 관계도

그림 5는 그림 4에서 합성 패턴을 이용하여 실제 본 논문에서 사용된 인터페이스 구축한 내용이다. Geometry class는 인터페이스에서 지리객체의 정보를 가지고 있는 객체로 Point class는 지상기준점의 정보를 가지고 있는 객체로 Line class는 도로, 철도 등의 정보를 가지고 있는 객체로 Polygon class는 행정경계, 인공구조물 정보를 가지고 있는 객체로서 사용되어지며 CompositeGeo class는 본 논문에서 설명하는 합성 패턴에 핵심내용으로 부분이면서 전체로 활용되어지는 객체이다. CompositeGeo는 지리객체의 한 부분으로 사용되어지지만 반면에 지상기준점의 속성인 point와 도로, 철도의 속성인 line, 행정경계, 인공구조물의 속성인 polygon을 모두 가지고 있는 전체로 활용되어지는 복합객체이다.

### 3) 어댑터 패턴(Adapter Pattern)

클래스의 인터페이스를 클라이언트가 기대하는 형태의 인터페이스로 변환한다. 어댑터 패턴은 서로 일치하지 않는 인터페이스를 갖는 클래스들을 함께 동작시킨다.

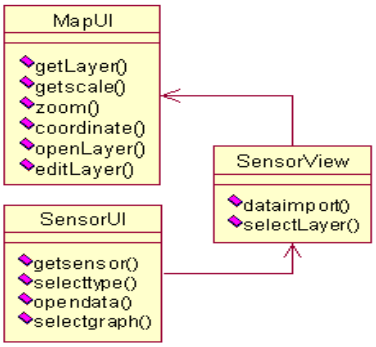


FIGURE 6. Geosensor 인터페이스의 어댑터 패턴 관계도

[패턴] *SensorView* : 지리정보시스템을 위해 일치하지 않는 맵 인터페이스와 센서 인터페이스를 데이터가 상호 연동되도록 중간 역할을 해 준다.

[정의] 클래스의 인터페이스를 클라이언트가 기대하는 형태의 인터페이스로 변환한다. 어댑터 패턴은 서로 일치하지 않는 인터페이스를 갖는 클래스들을 함께 동작시킨다.

[기능] MapUI와 SensorUI상에 서로 상호 연동되지 않는 바이너리 형식의 센서 데이터를 point, line, polygon의 GIS인터페이스에 연동되도록 중간 역할을 해준다.

#### 4) 팩토리 메소드(Factory Method)

지리정보시스템을 위한 인터페이스 중에서 지도 화면에 대해 원격 탐사를 위한 팩토리 메소드 패턴의 객체 생성과 기능을 다음과 같이 규정한다.

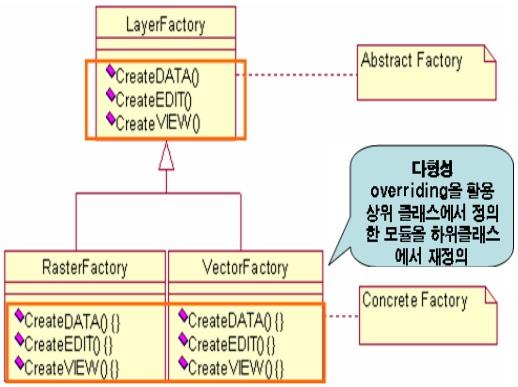


FIGURE 7. 지도화면 관리를 위한 Factory Method

그림 7은 지리정보시스템을 위한 팩토리 메소드 패턴을 말한다. 팩토리 메소드 패턴은 객체를 생성하기 위해 인터페이스를 정의하지만, 어떤 클래스의 인스턴스를 생성할지에 대한 결정은 서브클래스에서 이루어지도록 서브클래스에게 인스턴스 생성의 책임을 미룬다. 생성할 객체 타입을 예측할 수 없거나 생성할 객체를 기술하는 책임을 서브클래스에 정의하고자 하는 경우에 사용된다.

[패턴] ***CreateDATA(), CreateEDIT(), CreateVIEW()*** : 객체 생성을 대행해 주는 함수

[정의] 객체를 생성하되 직접 객체 생성자를 호출해서 객체를 생성하는 것이 아니라 대행 함수를 통해 간접적으로 객체를 생성하는 방식

[기능] 상위 객체로부터 상속을 받아 하위 클래스에서 재정의 함으로서 객체를 생성

그림 7에서 LayerFactory, RasterFactory, VectorFactor의 각 클래스의 메소드를 그룹화 {CreateDATA(),CreateEDIT(),CreateVIEW()} 하여 객체 생성을 간접적으로 생성하는 방식이다. LayerFactory 클래스의 서브클래스가 인스턴스화 되면 애플리케이션에 따른 문서의 인스턴스가 된다.

#### 5) 템플릿 메소드(Template Method)

지리정보시스템을 위한 인터페이스 중에서 지도 화면에 대해 원격 탐사를 위한 템플릿 메소드 패턴의 객체 생성과 기능을 다음과 같이 규정한다.

[패턴] ***Putcoords()*** : 두 개 이상의 알고리즘이 공통된 부분을 가질 때 하나의 모듈로 관리하는 모듈

[정의] 전체적인 흐름이 동일한 알고리즘에 대해 클래스 상속관계를 이용해서 그 기본 골격을 하나의 모듈로 작성, 관리하게 만든 설계를 말한다.

[기능] 소스 코드의 중복을 피하기 위해 모든 하위 클래스에서 동일하게 나타나는 모듈을 추출해 하나의 클래스 내에 정의한다.

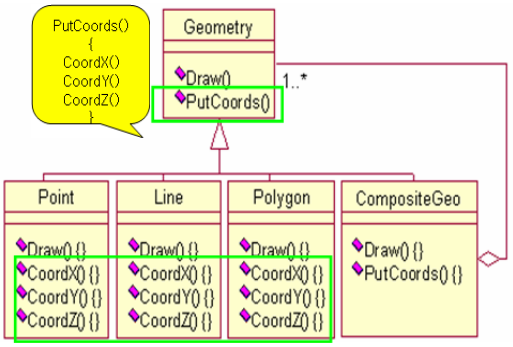


FIGURE 8. 지도화면 관리를 위한 Template Method

위 그림 8에서 Geometry 클래스의 PutCoords() 오퍼레이션을 템플릿 메소드로 정의하여 CoordX()와 CoordY(), CoordZ()를 정의하여 point, line, polygon의 상태로 세팅한다, PutCoords()는 CoordY 전에 CoordX를 호출하고 있다. 이전 상태를 복구하기 위해 CoordZ()를 다시 호출한다. 이렇듯이 템플릿 메소드 패턴은 가장 기본이 되는 패턴으로 대부분 추상 클래스에 사용되는 패턴이다.

## 지오센서 인터페이스

### 1. 지오센서 인터페이스 구현

### 1.1 디자인 패턴 기반 지오센서 인터페이스 기능

그림 9는 Abstract Factory, Factory Method, Composite, Adapter 그리고 Template Method의 디자인 패턴을 지오센서 인터페이스의 자료관리, 편집관리, 보기관리와 센서관리 기능에 적용한 유스케이스 다이어그램을 보여주고 있다.

최초 관리자가 맵관리 쓰임새와 센서관리 쓰임새에 따라 요청을 하면 맵관리 쓰임새는 자료관리와 편집관리 그리고 보기관리 쓰임새로 나뉘어 이벤트 요청을 하게 된다. 또한 센서 관리 쓰임새는 센서 데이터 관리와 상황인식 관리 쓰임새로 이벤트 요청을 하게 되어 각 쓰임새에 맞는 저장, 열기, 닫기, 삽입, 수정 및 삭제 등의 목적에 맞는 요청 및 응답을 하게 된다.

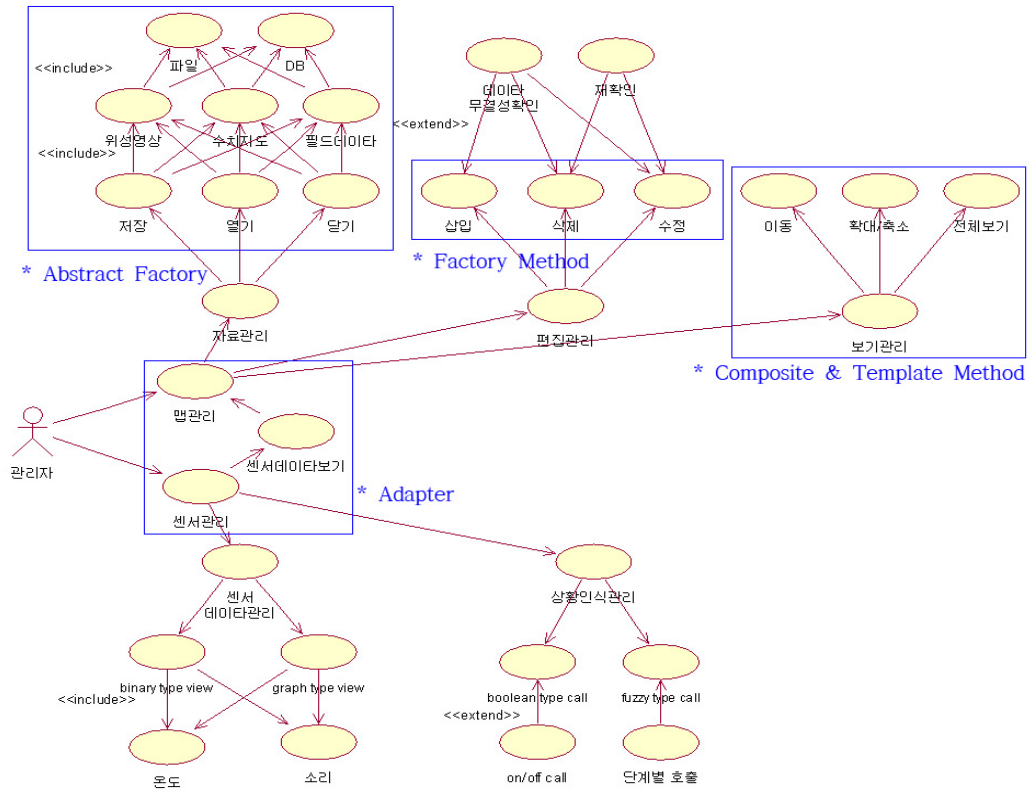


FIGURE 9. 지오센서 인터페이스를 위한 유스케이스 다이어그램

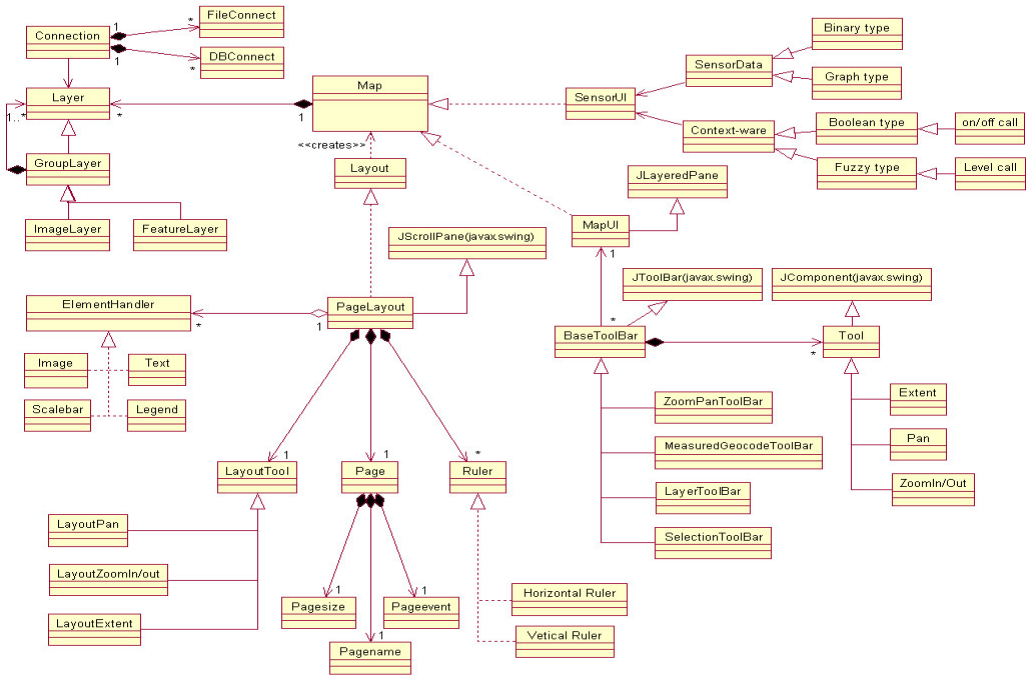


FIGURE 10. 지오센서 인터페이스를 위한 클래스 다이어그램

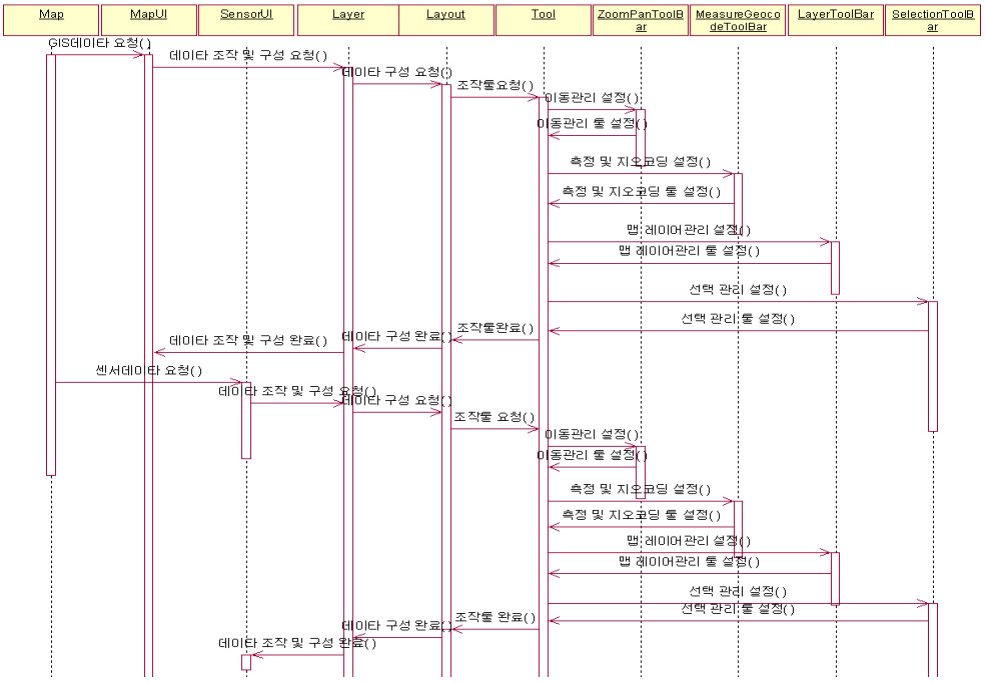


FIGURE 11. 이벤트 흐름에 따른 순차 다이어그램

### 1.2 디자인 패턴 기반 지오센서 클래스 관계도

그림 10은 지오센서 인터페이스에서 ArcGIS 9.1의 ArcObject 모델에서 “Map”을 Root 클래스로 하고 있다(선도소프트, 2005). 사용자 인터페이스 기능을 하는 SensorUI와 MapUI 클래스는 상속관계, 출력양식 기능의 Layout 클래스는 종속관계, 그리고 지도 중첩을 형성하는 Layer 클래스는 집단화관계를 형성한다. 세부적으로 Pagelayout 클래스는 JScritPane 클래스로부터 상속을 받으며, SensorUI는 Sensordata와 contextware클래스를 종속시킨다. layer는 GroupLayer로 세분화되며, 또한 Grouplayer는 ImageLayer와 FeatureLayer로 세분화된다.

### 1.3 디자인 패턴 기반 지오센서 순차도

그림 11은 지오센서 인터페이스 클래스 다이어그램에서 생성되는 각 객체들 사이의 순차적인 메시지 교환에 의해서 관리자 액터(actor)가 요청에 따른 흐름을 맵 관리와 센서관리로 나누어 각각의 관리자에게 요청을 보내면 각 관리자들이 이벤트 흐름에 따라 시스템의 요청을 처리하는 형식으로 이루어져 있다.

### 1.4 지오센서 인터페이스 구성

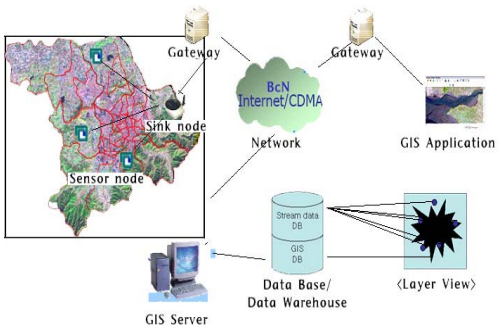


FIGURE 12. 지오센서 구성도

본 논문에서는 데이터 센싱을 위해 IEEE802.14.4. Zigbee 프로토콜을 따르고 Wi-Fi, Bluetooth와 같은 대역폭을 쓰는 Chipcon CC2420기반인 MOTEKIT을 사용하여 아래와 같은 데이터를 취득했다(백정호, 2006; Stefanidis, 2005).

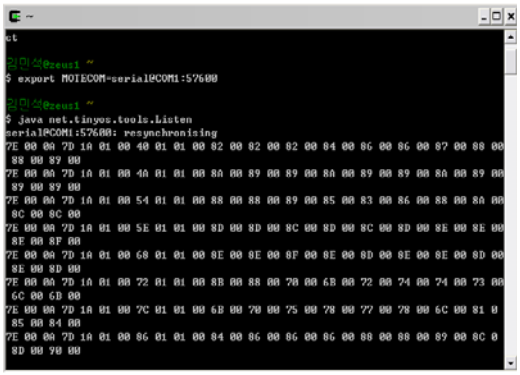

FIGURE 13. 센싱 데이터의 전송형태

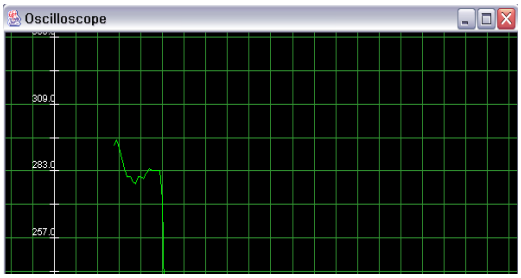


FIGURE 14. 센서 그래프 인터페이스

그림 13은 센서노드에서 측정된 아날로그 형식의 온도 데이터가 게이트보드를 통해 디지털 데이터인 바이너리 형식으로 서버에 무선으로 전송되어 서버에서 측정 및 모니터링 하는 모습을 보여주고 있다. 그림 14는 서버로 전송되어진 디지털데이터인 바이너리 형식의 데이터가 오실로스코프에 의해 도표화 되어져 사용자 중심의 센서 인터페이스를 나타내고 있음을 보여준다.

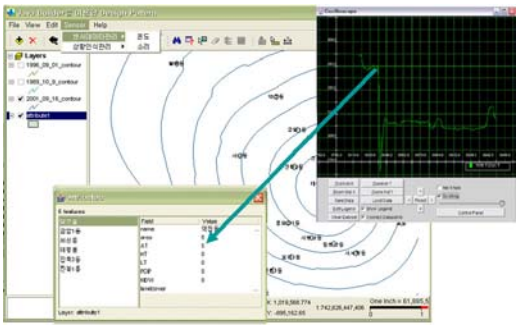

FIGURE 15. Sensor와 GIS 융합 인터페이스

본 연구에서는 Windows 2003서버 OS 환경 위에 ArcGIS 9.1 MapObject를 이용하여 Java 기반으로 지오센서 인터페이스가 구현되었다. 이 인터페이스는 Win32 API를 이용한 JBuilder를 이용하여 제작하였다. 각 디자인패턴을 적용하여 지리정보와 센서를 결합하여 위와 같은 인터페이스를 구현 하였고 전주시의 온도 변화탐지에 대한 부분을 각 행정동의 실시간 측정 온도데이터와 위성영상에서 추출한 온도 분포도의 관계를 보고 예측 분석을 하도록 구현 하였다.

그림 15에서는 실시간으로 측정된 센서데이터가 지도위에 위치값을 가진 전주시 행정동에 표현되어 온도분포도로 나타내어지는 지리정보 인터페이스에서 표현하였고, 센서 메뉴 중 온도 부분을 활용하여 실시간 온도를 하였다. 센서데이터와 지리정보 인터페이스의 결합된 형태를 보여주고 있다.

## 검 토

본 논문에서 제안하는 센서기반 GIS인터페이스는 위치탐색부분과 다기능 센싱, GIS연동, 분석, 예측/예방에 있어서 타 인터페이스보다 더 많은 기능을 제공 하였다.

객체지향 소프트웨어 개발의 주요 목적은 재상용성과 개발 시간의 단축, 비용의 절감 그리고 소프트웨어 품질 향상에 있다. 본 논문은 이러한 목적을 위해 소프트웨어 개발자들은 기존의 경험에 의해 잘 정의되고 테스트된 설계 정보. 즉, 소프트웨어 개발 시 특정상황에서 발생할 수 있는 문제에 대한 효율적인 해결책을 제시하는 디자인 패턴을 가지고 지리정보시스템의 품질 향상을 추구하였다. 이 디자인 패턴은 현재까지 다양한 분야에서 발견되어지고 있고, 이들의 활용성을 증명하는 연구가 진행되고 있다.

본 논문에서는 Java JBuilder를 이용하여 디자인 패턴을 지리정보와 센서네트워크에 적용함으로써 래스터와 벡터 자료형식에 기반한 인터페이스를 구현할 수 있는 객체지향 설계 모델을 규정하였다. OpenGIS 모델에 기반한 지리 데이터 모델을 분석 및 설계하고, 인터페이스 시스템을 구현하여 래스터와 벡터 자료의 구성 요소를 조립하고 지도 레이어 합성 객체를 생성하는 추상 팩토리 설계 유형기법을 지리정보 인터페이스 시스템에 도입하여 기존의 소스 코드와 독립적으로 새로운 요소를 추가할 수 있는 기능을 제시하였다.

본 논문은 GoF 디자인 패턴 중에서 Abstract Factory, Factory Method, Composite, Adapter 그리고 Template Method를 이용하여 컴포넌트를 통합하는 기법으로 Java JBuilder를 활용하여 지리정보시스템(GIS) 및 센서 모니터링을 위한 지도 화면(Map View)을 설계하고 구현하였다.

이에 디자인 패턴을 지오센서 인터페이스 설계 및 구현하는데 많은 문제점이 생겼다. 특히, 대행 함수를 통해 간접적으로 객체를 생성해야 하며, 메소드의 다형성과 재사용성 및 관련된 부품을 조립해서 제품을 생성해야 하는 경우, 재귀적 클래스를 구성해야 경우에 문제점이 생겼다.

이를 해결하기 위해 대행 함수를 통해 간접적으로 객체 생성하는 경우와 메소드의 다형성이 필요한 경우에는 Factory Method 패턴을 사용하여 해결하였다. 또한, 메소드의 재사용성이 필요한 경우에는 Template Method 패턴을 사용하였고, 맵 인터페이스와 센서 인터페이스의 동일한 실행을 위해 Adapter패턴을 적용하여 해결하였으며 관련된 부품을 조립해서 제품을 생성하는 경우에는 Abstract Factory, 재귀적 클래스 구성에는 Composite 패턴을 사용하여 해결하였다.

본 논문은 객체지향 모델의 디자인 패턴의 적용 여부에 따라 지오센서 인터페이스를 구축하는데 있어서 다음과 같은 효율성을 가짐을 알 수 있다.

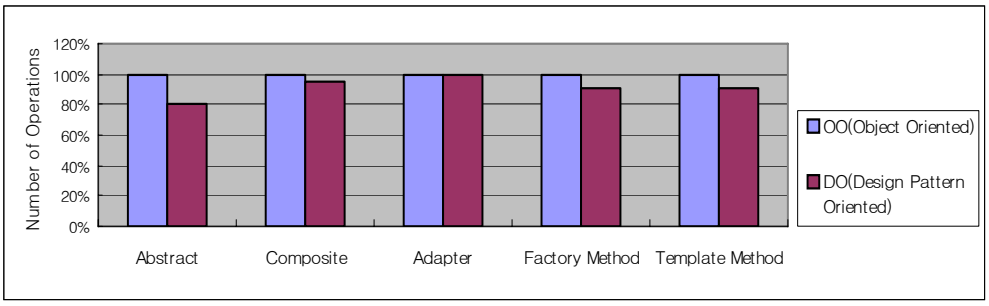


TABLE 2. 객체지향 모델과 디자인패턴 모델의 인터페이스 연산 갯수 비교

기존 객체지향 프로그래밍에서 개발한 인터페이스의 연산의 개수를 100%라 하고 디자인 패턴지향 프로그램으로 개발한 인터페이스의 효율성을 살펴보았다. Abstract Factory의 경우 20%, Composite Pattern의 경우 5%, Adapter pattern은 0%, Factory Method와 Template Method는 각각 10%씩의 절감효과를 보였다.

결국, 디자인 패턴 기반 지오센서 인터페이스 설계 및 구축은 새로운 기능이나 응용서비스를 갱신할 수 있는 인터페이스의 효율적인 구현에 기여하였으며, 소프트웨어 개발 시 제시된 디자인 패턴의 체계적인 접근과 활용은 디자인 경험과 기법을 초기 개발 단계부터 활용할 수 있게 함으로써 시스템 개발의 효율성을 증대시키고 보다 안정되고 재사용 가능한 시스템을 도출하여 개발 시간과 비용을 단축하는 효과를 제공할 수 있다.

## 결 론

본 연구에서는 5개 디자인패턴을 적용하여 기존 지리정보 인터페이스와 센서 인터페이스의 장점을 결합한 지오센서 인터페이스를 개발하였다.

Abstract Factory의 구체적 클래스를 정의하지 않고도 서로 관련성이 있거나 독립적인 여러 객체의 군을 생성하기 위한 인터페이스 제공과 함께 Composite의 부분-전체적 계층을 나타내기 위해 복합객체를 트리 구조로 만들 수 있다. 또한, Factory method의 클래스의 인스턴스를 만드는 시점을 서브 클래스로 미뤄 객체를 생성하는 인터페이스를 정의할 수 있고, Template method는 알고리즘의 처리 과정은 변경하지 않고 알고리즘의 각 단계 처리를 서브클래스에서 재정의 할 수 있게 한다.

본 논문에서는 지리정보 인터페이스 설계 및 구현에 있어서 설계단계에 필요한 디자인 패턴을 식별하여 디자인 패턴을 적용하는 기법을 제시한다. 디자인 패턴을 이용한 기법은 객체지향 소프트웨어 개발에 있어서 재사용성을 증가시킴으로써 비용을 절감 시키고, 유효성이 입증된 기존의 패턴은 안정성과 신뢰성에 확신을 갖게 한다. 유사한 문제에 대한 반복적 생성은 패턴화 된 시스템의 확장을 편리하게 한다. 복잡한 현실 세계를 기반으로 하는 원격탐사 지리정보시스템은 잘 분류된 설계 패턴에 의해 추상화됨으로써 지리정보시스템 확장을 단순화 하는데 기여한다. 또한, 실시간 센서데이터와 GIS 인터페이스를 결합함으로서 실시간 상황정보들을 지리정보시스템에 디스플레이하여 stream data의 지도표현이 가능하게 됨으로 실시간으로 변화되는 상황정보의 의사결정에 활용되게 하였다.

앞으로 USN을 구성하여 웹이나 PDA같은 무선 단말을 통하여 지리정보 데이터를 언제 어디서나 사용자에게 제공을 하고 받는 서비스 구현을 할 것이다. 또한 이 연구를 발판으로

U-korea지향 U-City에 대한 기본 인프라연구도 할 것이다.

향후 이와 같은 다양한 센서장비를 통하여 도시, 연안, 환경 등에 적용하여 감지 되어지는 데이터를 가지고 GIS분석을 통한 인간의 삶에 더 편리한 기능 및 정보를 제공 할 것이다. KAGIS

## 참 고 문 헌

김운용, 최영근. 2002. 디자인 패턴지향 소프트웨어 개발 지원도구. 정보과학회 29(8).

백정호, 문홍실, 이홍로. 2006. 실시간 상황정보 센서네트워크 기반 융합 지리정보 인터페이스 개발. 한국지리정보학회 2006 춘계학술대회 pp. 505-512,

양인태, 최영재. 2004년. 위성영상을 기반으로 한 GIS 응용 시스템 개발. 한국측량학회지 22(1):1-9.

이재봉, 백정호, 이홍로, 박기홍, 박승철, 사공호상. 2005. 객체지향 디자인 패턴 기반 전주시 온도변화 탐지 예측 시스템 설계 및 구현. 한국지리정보학회 학술발표대회 pp. 247-255.

선도소프트. 2005. What is ArcGIS 9.1? www.sundosoft.com

Xiong zhan wu, 문홍실, 이홍로. 2006. Design Pattern Based GIS Application Interface Efficient Implementation Using JBuilder. 한국지리정보학회 2006 춘계학술대회 pp. 537-548.

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

Gordillo, S., F. Balagure and F. das Neves. 1997. Generation the Architecture of GIS Applications with Design Patterns. 5th ACM GIS Symposium Las Vegas ACM Press.

Gordillo, S. and F. Balagure, 1999. CATALINA MOSTACCIOM, AND FERNANDO DASNEVES. Developing GIS Applications with Objects: A Design Patterns Approach, GeoInformatica 3(1):7-32.

Camara, G., 2001. Design Pattern in GIS Development: The TerraLib Experience. Brazilian Symposium on GeoInformatics GeoInfo2001 Rio de Janeiro.

Stefanidis, A. and Nittel, S. 2005. GeoSensor Networks. CRC Press.

T. H. Ng, S. C. Cheung, W. K. Chan, Y. T. Yu, 2006. Toward effective deployment of design patterns for software extension: a case study. Proceedings of the 2006 international workshop on Software quality WoSQ '06, Publisher: ACM Press, pp.51-56.

Yacoub, Sherif M., Ammar, Hany H. 2004. Pattern-oriented analysis and design. Addison Wesley pp.4-17. KAGIS