KITA

Inclusion Polymorphism과 UML 클래스 다이어그램 구조에 의거한 디자인패턴 해석

이랑혁, 이현우, 고석하

rang2guru@gmail.com, westminstor@naver.com, shkoh@cbnu.ac.kr

충북대학교 경영정보학과

충북 청주시 흥덕구 개신동 12번지 충북대학교
학연산공동기술연구원 843호

Tel:043-272-4034

Keyword : Design patterns, Inclusion Polymorphism, UML Class diagram, Classification

- Abstract -

디자인 패턴은 새롭게 만들어 지는 것이 아니라 기존의 검증된 지식, 관용법, 원칙들을 체계화한 것이다. 다시 말하면 디자인 패턴은 특정한 문제를 해결하기 위한,검증된 설계 방법에 이름을 붙인 것이다. 그러므로 적절한 디자인 패턴 사용은 1) 개발자들간의 원활한 의사소통에 도움을 주며, 2) 하급자가 고급기술을 쉽게 익힐 수 있도록 할 수 있다. 3) 또한 사용된 디자인이나 아키텍처를 재사용할 수 있도록 하고, 4) 만들어진 시스템의 유지 보수를 보다 쉽게 할 수 있는 등의 장점을 얻을 수 있다. 반면에 필요하지 않은 곳에 까지 디자인패턴을 사용하게 되면 소프트웨어를 복잡하고, 유지보수도 어렵게 만들 수 있다.

디자인 패턴의 분류는 수 많은 패턴을 비슷한 속성을 지닌 그룹들로 조직화 하는 것이다. 이는 개발자가 특정 문제에 맞는 디자인 패턴을 쉽게 선택 할 수 있도록 도와 줄 뿐만 아니라, 디자인 패턴의 주요특성을 빠르게 이해하고 간파 할 수 있게 한다. 그래서 Beck이 디자인패턴을 소개한 이후 GoF, Buschmann, Grand, Antoy 등은 디자인 패턴을 단순히 열거를 통해 소개하지 않고,각자의 기준에 따라 체계적으로 분류하여 패턴을 설명 하고 있다. 본 연구는 객체지향 설계의 근본 개념인 Polymorphism (Inclusion Polymorphism)과 '객체 지향 소프트웨어 설계 원칙' 그리고 이 근본 원칙들이 UML 클래스 다이어그램에 나타나는 구조적 특징에 의거해 디자인 패턴 해석을 수행 하였다. 본 연구의 목적은 1)객체지향의 근본 원칙으로 표현 되는 패턴과 2)설계자의 전문적인 Art를 포함하고 있는 패턴으로 분류하는데 있다.

목 차

연구목적

- 본 연구는 소프트웨어 설계 시 적용되는 '객체지향의 근본 개념'이 'UML클래스 다이어그램'으로 표현된 '일반적 형태'에 의거해 기존의 디자인 패턴을 해석 한다.

  이를 통해 디자인 패턴을

  - 1)단순히 객체지향 근본 개념을 설명하는 패턴
  - 2) 전문가적 Art가 존재하는 패턴

  으로 그룹화 한다.

연구범위

- 객체지향 설계의 기본 원칙
  - Inclusion Polymorphism (Cardelli1985)
  - 객체지향 소프트웨어 설계원칙(Martin2004)
  - UML Class-Diagram 적용 (OMG2005)

- 해석 대상
  - GoF 디자인 패턴 (GoF1995)
  - GRASP 디자인 패턴 (Larman1998)

관련연구 – 다형성(Polymorphism)의 분류

- Universal Polymorphism [Cardelli et 1985]
  - Inclusion Polymorphism
  - Parametric Polymorphism
    - Bounded Polymorphism
    - F-Bounded Polymorphism [Canning 1989]
- Ad-hoc Polymorphism [Cardelli et 1985]
  - Overloading
  - Coercion

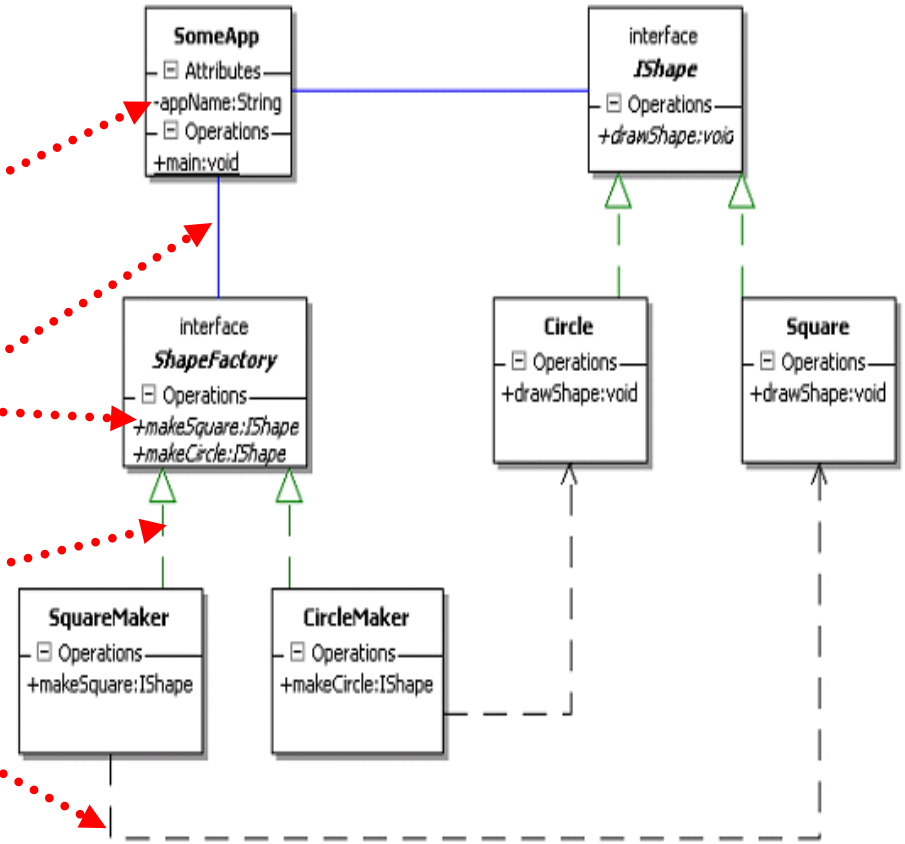
관련연구 – 객체지향 소프트웨어 설계원칙

객체지향 소프트웨어 설계원칙 (Martin 2004)

- SRP (Single Responsibility Principle)
- OCP (The Open-Closed Principle)
- LSP (Liskov Substitution Principle)
- DIP (Dependency Inversion Principle)
- ISP (Interface Segregation Principle)
- RDP (Repetitiveness Dependency Principle)

관련연구 – UML Class Diagram

- 시스템의 객체의 타입과 그들 간의 존재하는 다양한 정적 관계에 대해서 기술한다.(Fowler2003)
  - 속성(attribute)
  - 오퍼레이션(operation)
  - 연관(association)
  - 일반화(Generalization)
  - 의존(dependency)

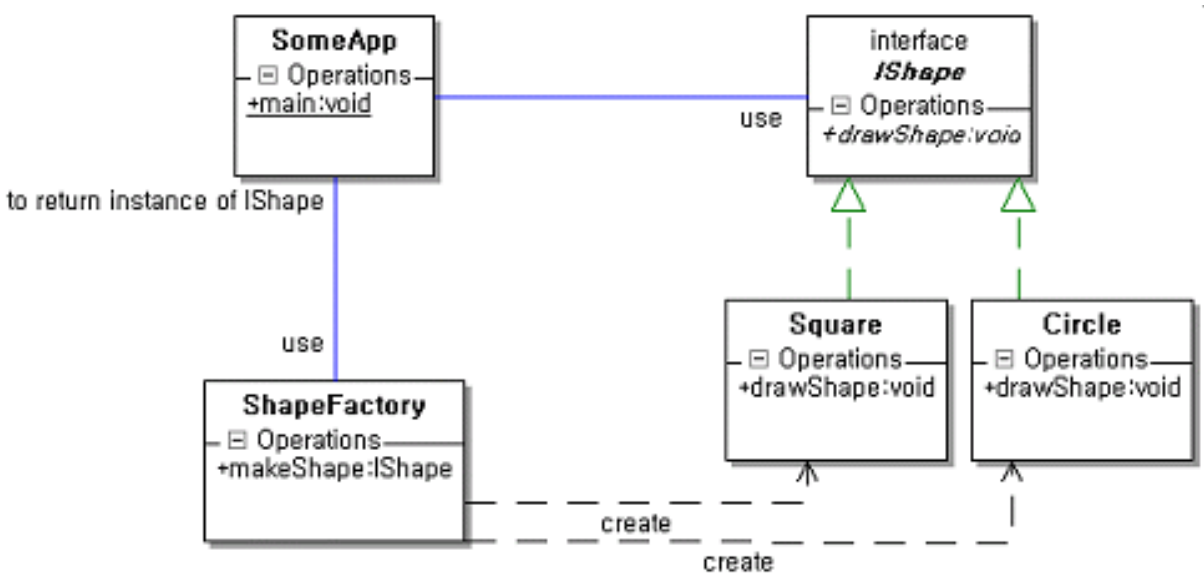


관련연구 – 디자인 패턴 분류

- GoF : Design patterns are classified by their scope and purpose
- Buschmann : Design patterns are classified by their purpose(problem categories) and granularity(pattern categries).
- Zimmer : GoF patterns are Classified by their relationship.
  - Zimmer defines three categories of relationships:
    - Pattern X uses pattern Y in its solution
    - Pattern X is similar to pattern Y.
    - Pattern X can be combined with pattern Y

객체지향 기본 개념을 이용한 디자인 패턴 해석

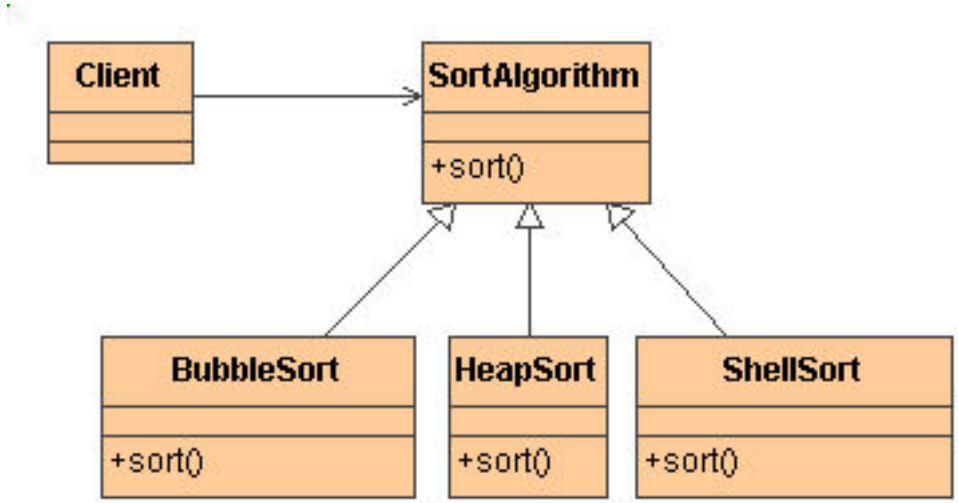
- 객체지향 설계의 기본개념
  - Inclusion Polymorphism
  - UML Class diagram
  - 객체지향 설계원칙



객체지향의 기본 개념이 적용된 일반적 설계 모양

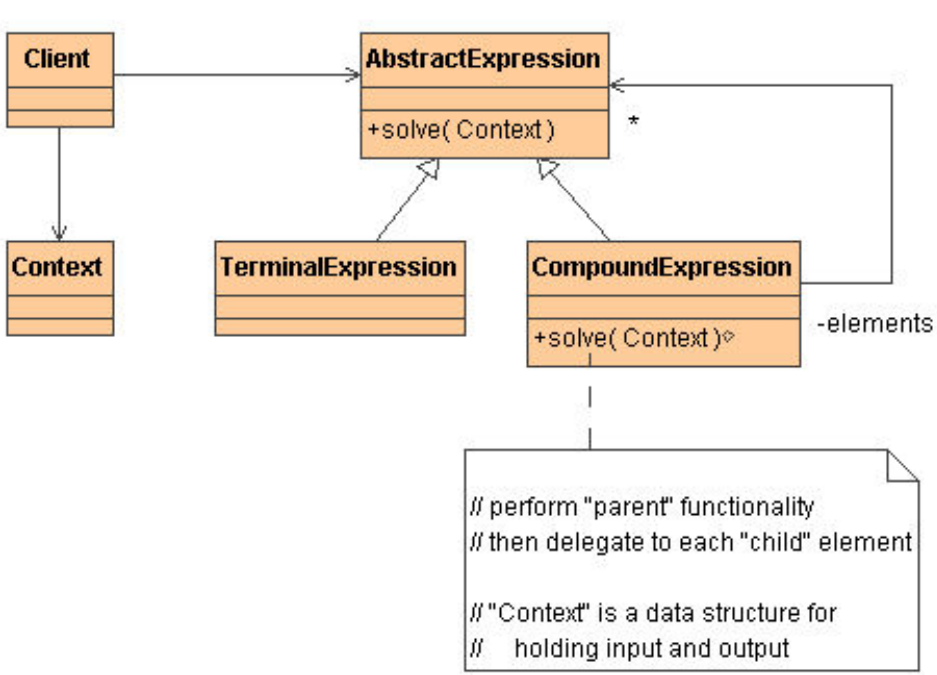
객체지향 기본 개념을 이용한 디자인 패턴 해석

- 객체지향 설계의 기본개념 만으로 표현 가능한 디자인 패턴 예시
  - Strategy 패턴은 Inclusion polymorphism의 전형적인 형태이다.



객체지향 기본 개념을 이용한 디자인 패턴 해석

- 설계 전문가의 Art가 적용되어 있는 패턴 예시
  - Interpreter 패턴은 객체지향 설계원칙과 Art가 필요한 패턴이다.



결 론

- 객체지향의 근본 개념에 포함되는 다형성(Inclusion Polymorphism)과 객체지향 설계 원칙 그리고 이 근본 개념에 의해 표현된 UML 클래스 다이어그램에 의거해 디자인 패턴을 분류한 결과 1) 근본 개념이 설계에 단순이 적용된 형태와 2) 객체지향 설계의 근본 원칙에 전문가적 Art가 적용된 패턴으로 그룹화 할 수 있었다.

참고문헌

[Buschmann1996] Frank Buschmann, R. Meunier, H. Rohnert, P. Sommerland, M. Stal, "Pattern-Oriented Software Architecture: A System of Patterns", Addison-Wesley, 1996.

[Cardeli1985] Luca Cardelli, Peter Wegner, "On Understanding Types, Data Abstraction, and Polymorphism", Computing Survey, Vol17 n.4, pp471-522, December 1985.

[Canning1989] Peter Canning, William Cook, Walter Hill, Walter Olthoff, "F-Bounded Polymorphism for Object-Oriented Programming",ACM, 1989.

[Fowler2004]Martin Fowler, "*UML Distilled Third Edition : A Brief Guide To The Standard Object Modeling Language*", Addison-Wesley, 2004.

[GoF1995] Erich Camma, Richard Helm, Ralph Johnson, John Vlissides, "*Design Patterns : Elements of Reusable Object-Oriented Software*.",Addison Wesley, 1995.

[Larman1998] Craig Larman, "*Appying UML And Patterns* ", Prentice Hall PTR, 1998.

[Liskov1987] Babara Liskov, "Data Abstraction and Hierachy", OOPLSA '87 Addendum to the Proceedings, 1987.

[Liskov2000] Babara Liskov, John Gutag , *"Program Development In Java : Abstraction, Specification, and Object-Oriented Design"*, Canada: Addison-Wesley, 2000.

[Martin2004] Robert C. Martin,이용원, 정지호, 김정민 역, *"소프트웨어 개발의 지혜 : 원칙, 디자인 패턴, 실천방법"*, 야스 미디어,2004.

[Zimmer1996] Zimmer, W., "Relationships Between Design Patterns", Pattern Languages of Programs 2, Vlissides et. Al., eds., Addison Wesley, 1996

[OMG2005] OMG, "*Unified Modeling Language : Superstructure*", OMG,2005.