Software Design Patterns

Lecture 10

BIL428 Software Architectures

Mustafa Sert
Asst. Prof.
msert@baskent.edu.tr

Department of Computer Engineering, Başkent University
Ankara 06810 TURKEY

Outline

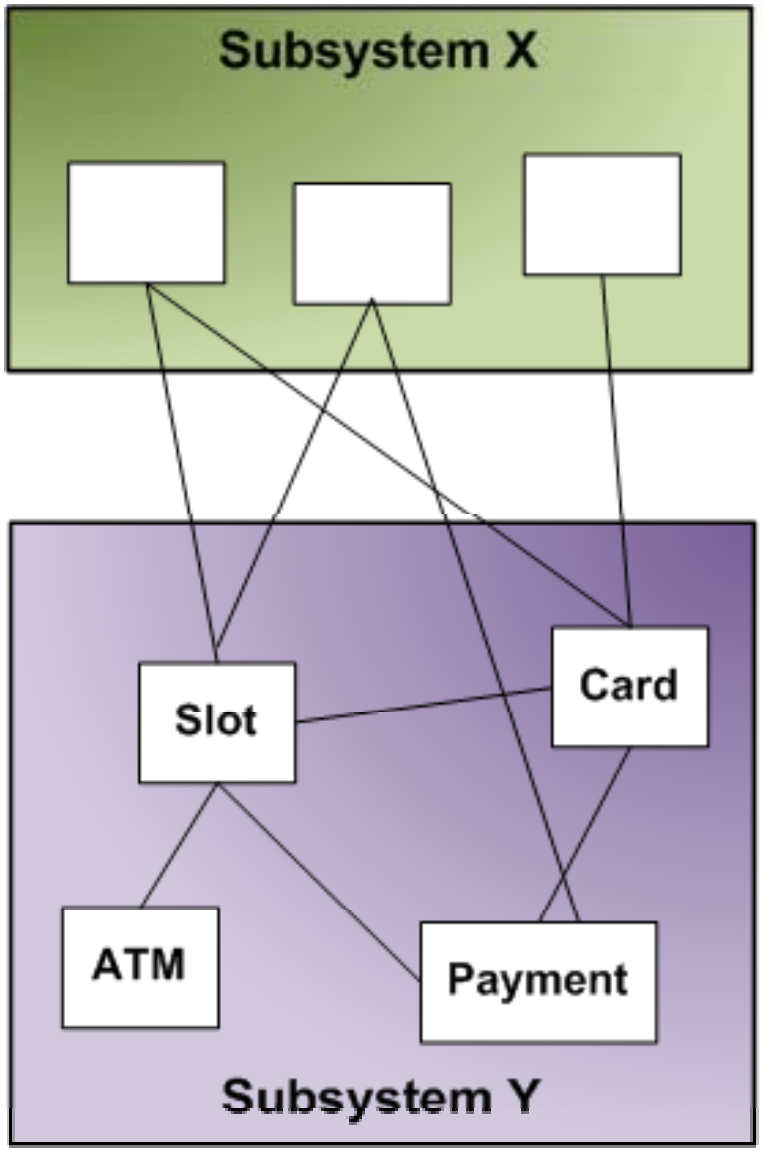
- Structural Patterns
  - Adapter (we have already seen that)
  - Controller (we have already seen that)
  - Façade
  - Proxy
- Behavioral Patterns
  - Observer
  - Strategy

Facade

- A kind of **structural** design pattern
- Provides a unified interface to a set of objects in a subsystem.
- A facade defines a higher-level interface that makes the subsystem easier to use
- Facades allow us to provide a closed architecture by encapsulating a subsystem with a simple unified interface

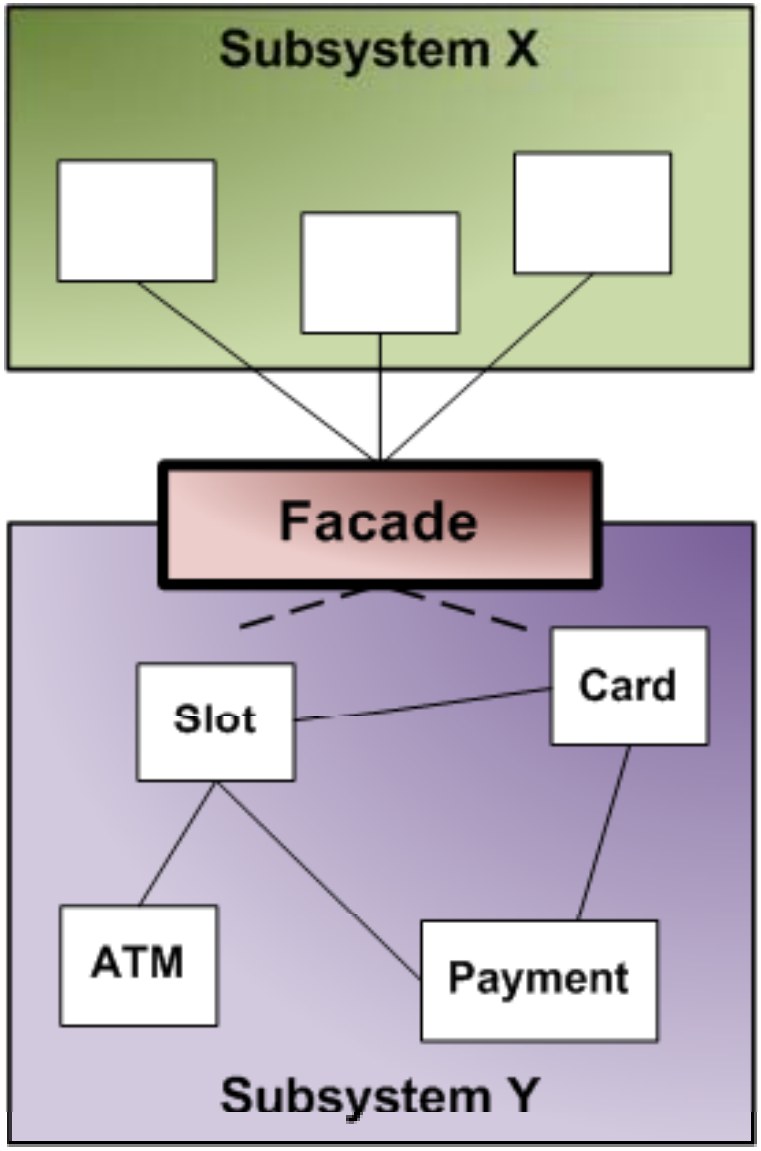
Facade – Design Example

- Subsystem X can look into the Subsystem Y (ATM subsystem) and call on any component or class operation..
- This is a "Ravioli Design"
- Why is this good?
  - **Efficiency!**
- Why is this bad?
  - **Can't expect the caller to understand how the subsystem works or the complex relationships within the subsystem**
  - **May lead to non-portable code!**



Facade – Design Example (*cont.*)

- Realization of the Interface Object: **Façade**
- The subsystem decides exactly how it is accessed
- No need to worry about misuse by callers
- If a façade is used, the subsystem can be used in an early integration test
  - **We need to write only a driver**



Facade – Summary

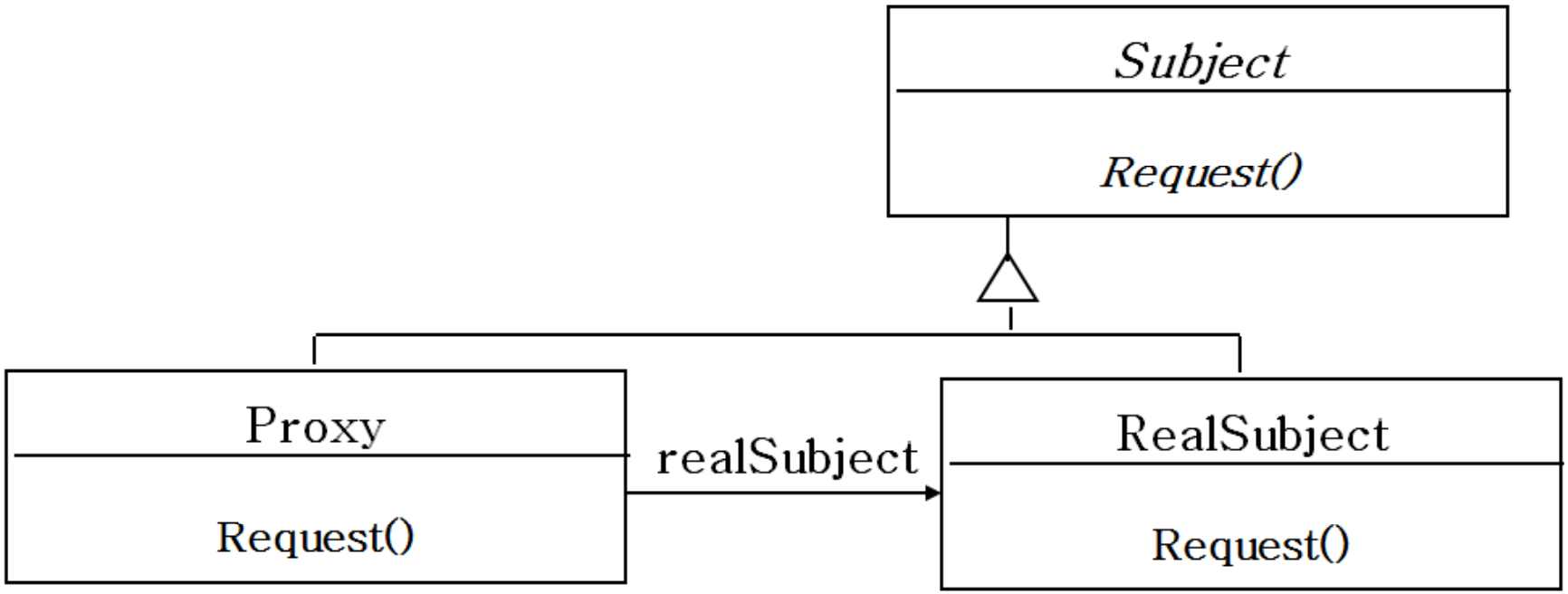
- **Purpose**
  - The Facade pattern **reduces dependencies** among classes **by encapsulating a subsystem with a simple unified interface**
- **Description**
  - A single Facade class **implements a high-level interface** for a subsystem by invoking the methods of lower level classes
  - A Facade is **opaque** in the sense that a caller does not access the lower level classes directly
  - The use of Facade patterns **recursively yields a layered system**

Proxy

- A kind of **structural** design pattern
- What is expensive?
  - Object Creation
  - Object Initialization
- Defer object creation and object initialization to the time you need the object
- Proxy pattern:
  - Reduces the cost of accessing objects
  - Uses another object ("the proxy") that acts as a stand-in for the real object
  - The proxy creates the real object only if the user asks for it

Proxy

- Interface inheritance is used to specify the interface shared by **Proxy** and **RealSubject**
- Delegation is used to catch and forward any accesses to the **RealSubject** (if desired)
- Proxy patterns can be used for lazy evaluation and for remote invocation



Proxy – Applicability

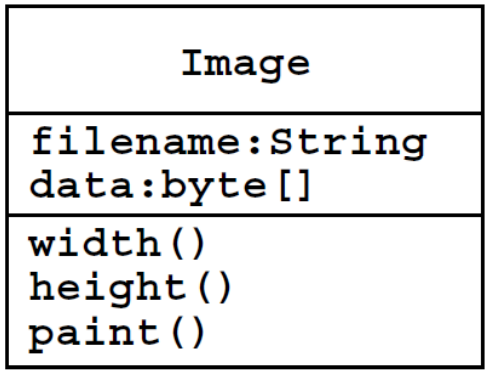
- **Remote Proxy**
  - Local representative for an object in a different address space
  - Caching of information: Good if information does not change too often
- **Virtual Proxy**
  - Object is too expensive to create or too expensive to download
- **Protection Proxy**
  - Proxy provides access control to the real object
  - Useful when different objects should have different access and viewing rights for the same document.
  - Example: Grade information for a student shared by administrators, teachers and students.

Virtual Proxy Example

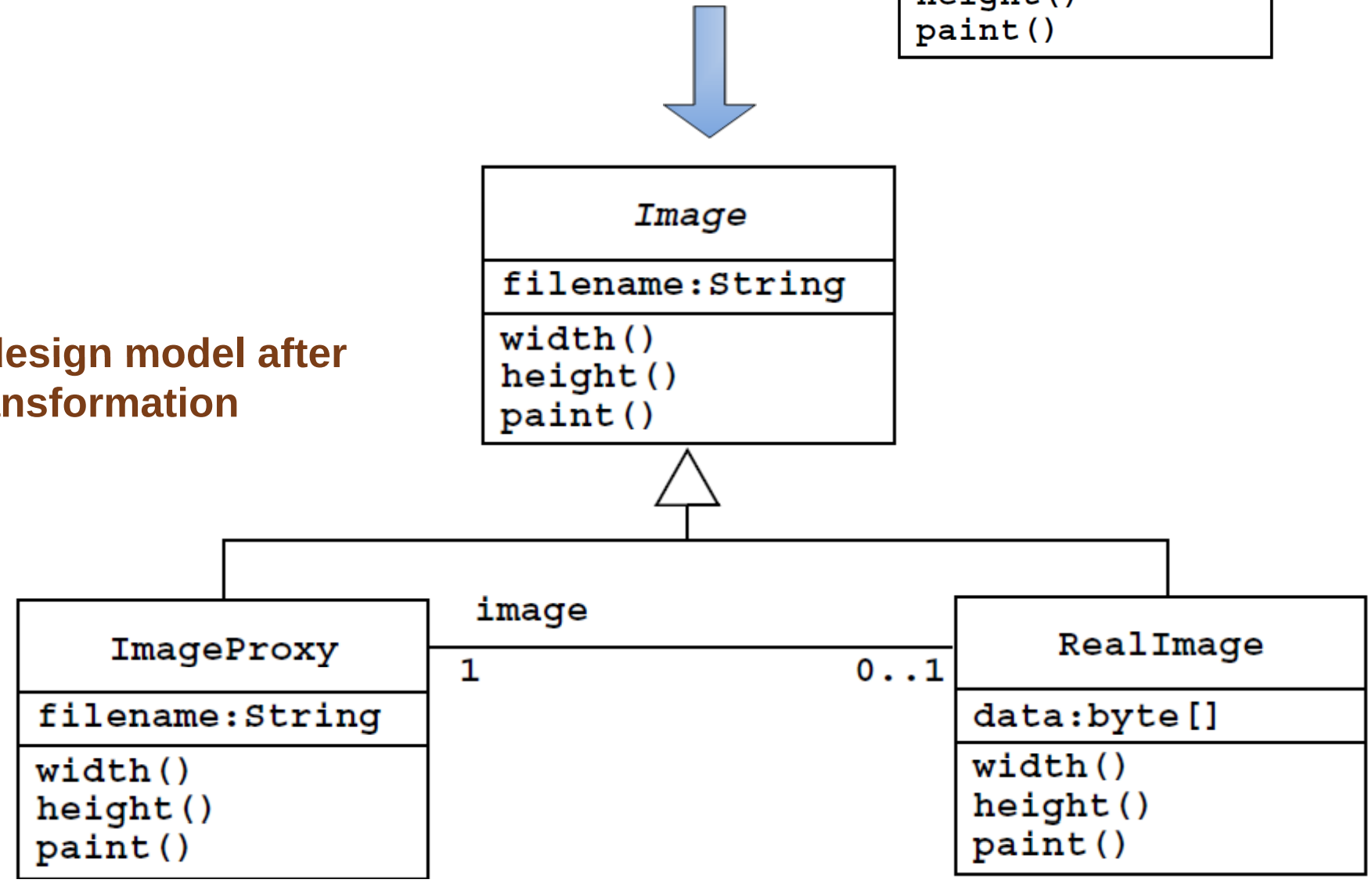
- An **ImageProxy** object acts on behalf of an Image stored on disk
  - The **ImageProxy** contains the same information as the Image (e.g., width, height, position, resolution) except for the Image contents
  - The **ImageProxy** services all contents independent requests
  - Only when the Image contents need to be accessed (e.g., when it is drawn on the screen), the **ImageProxy** creates the **RealImage** object and loads its contents from disk.

Virtual Proxy (*cont.*)

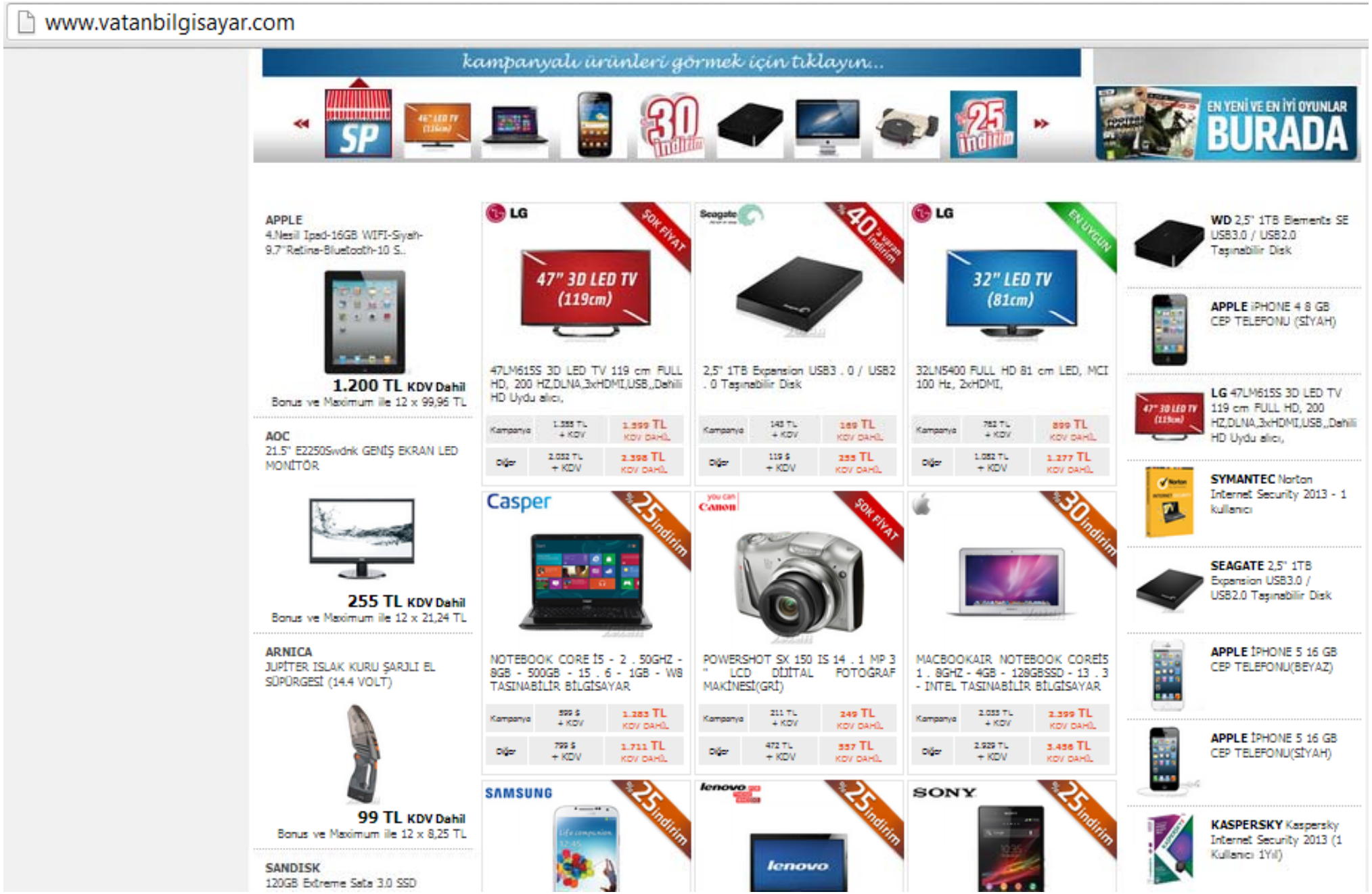
Object design model before transformation



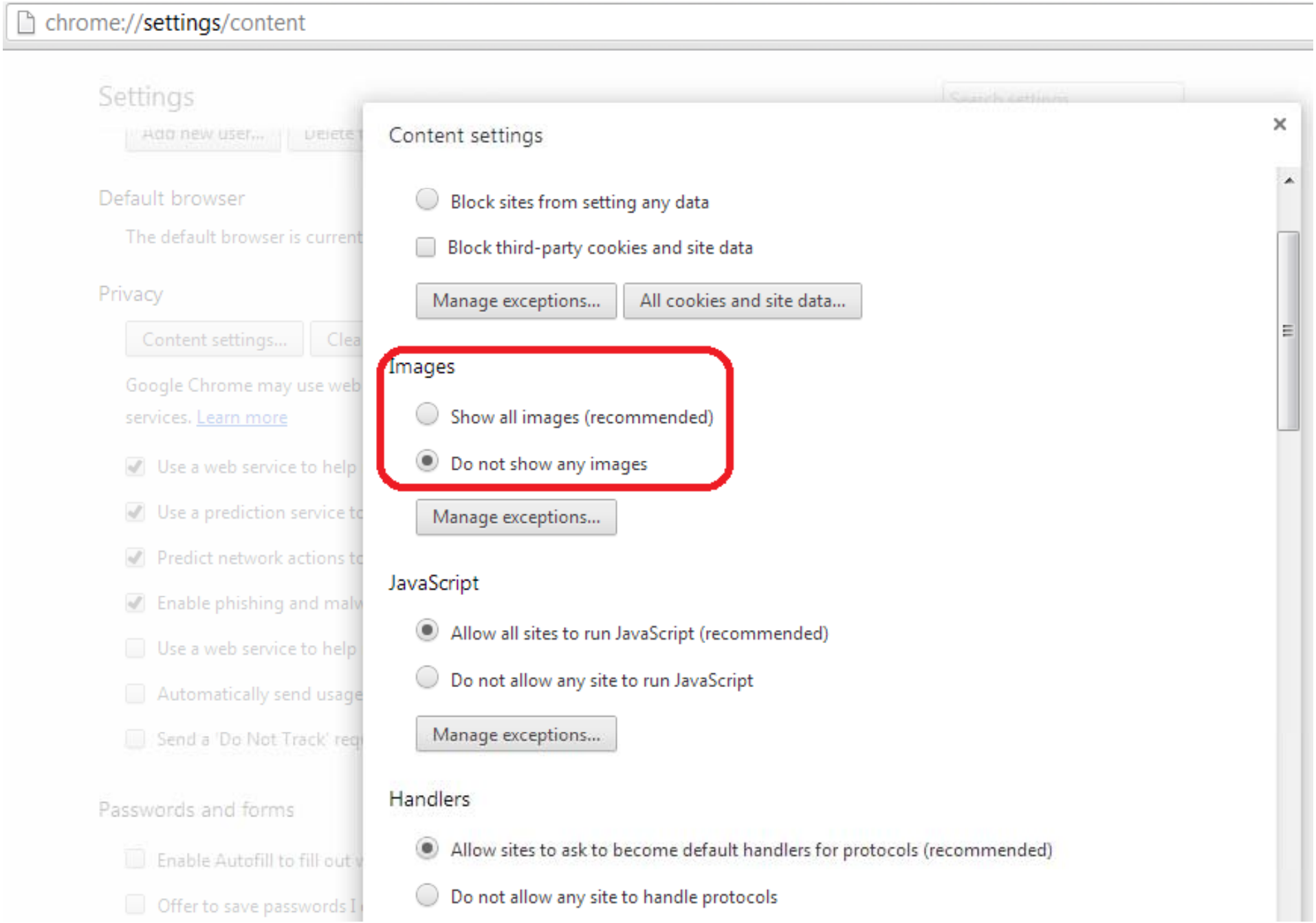
Object design model after transformation



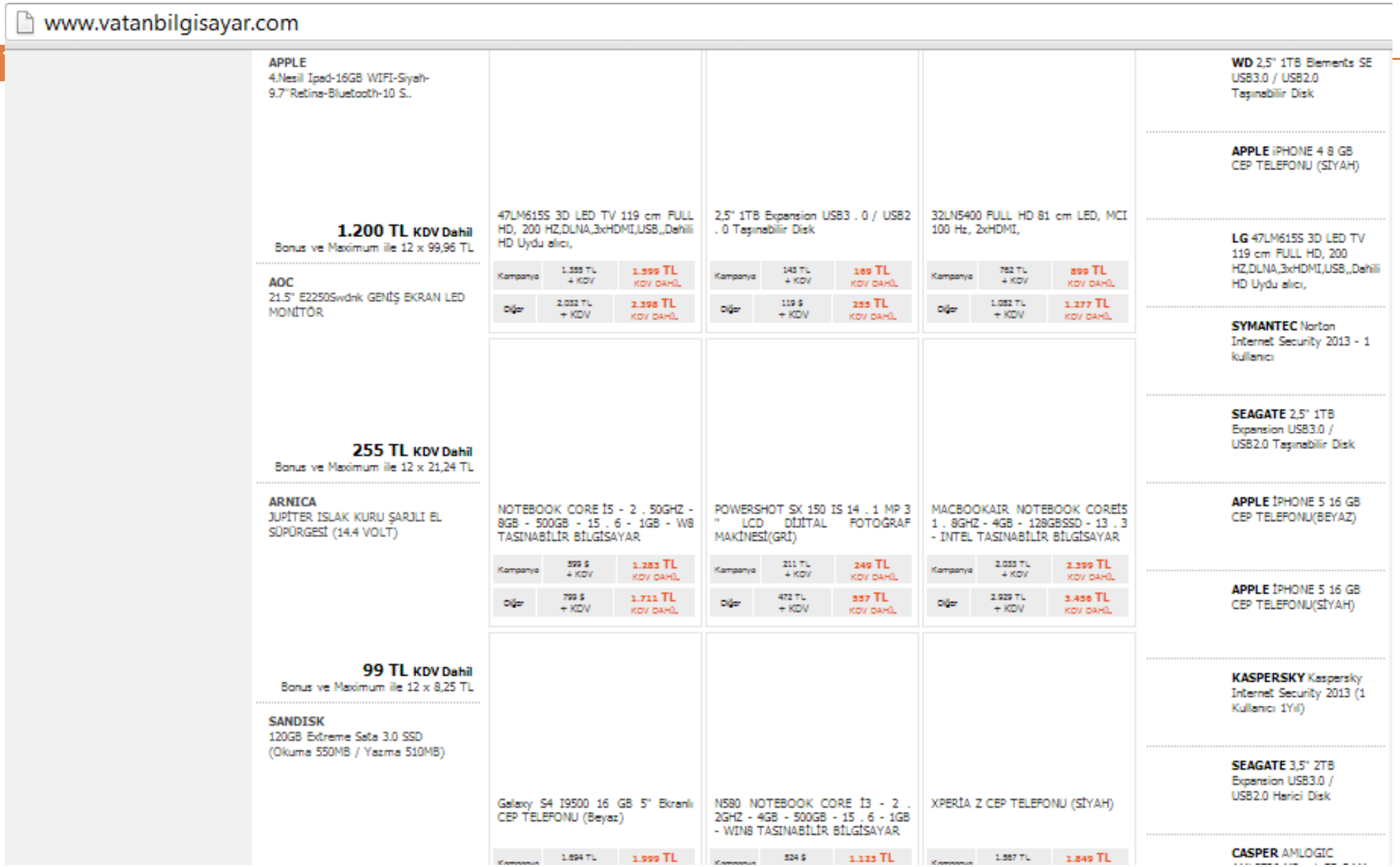
Before..


www.vatanbilgisayar.com
kampanyalı ürünleri görmek için tıklayın..
EN YENİ VE EN İYİ OYUNLAR BURADA
APPLE
4.Nesil Ipad-16GB WIFI-Siyah-9.7"Retina-Bluetooth-10 S..
1.200 TL KDV Dahil
Bonus ve Maximum ile 12 x 99,96 TL
AOC
21.5" E2250Swdnk GENİŞ EKRAN LED MONİTÖR
255 TL KDV Dahil
Bonus ve Maximum ile 12 x 21,24 TL
ARNICA
JUPİTER ISLAK KURU ŞARJLI EL SÜPÜRGESİ (14.4 VOLT)
99 TL KDV Dahil
Bonus ve Maximum ile 12 x 8,25 TL
SANDISK
120GB Extreme Sata 3.0 SSD
LG
ŞOK FİYAT
47" 3D LED TV (119cm)
47LM615S 3D LED TV 119 cm FULL HD, 200 HZ,DLNA,3xHDMI,USB,,Dahili HD Uydu alıcı,
Seagate
%40 Yarın İndirim
2,5" 1TB Expansion USB3 . 0 / USB2 . 0 Taşınabilir Disk
LG
EN UYGUN
32" LED TV (81cm)
32LN5400 FULL HD 81 cm LED, MCI 100 Hz, 2xHDMI,
Casper
%25 İndirim
NOTEBOOK CORE İ5 - 2 . 50GHZ - 8GB - 500GB - 15 . 6 - 1GB - W8 TASINABİLİR BİLGİSAYAR
Canon
ŞOK FİYAT
POWERSHOT SX 150 IS 14 . 1 MP 3 " LCD DİJİTAL FOTOĞRAF MAKİNESİ(GRİ)
%30 İndirim
MACBOOKAIR NOTEBOOK CORE İ5 1 . 8GHZ - 4GB - 128GBSSD - 13 . 3 - INTEL TASINABİLİR BİLGİSAYAR
SAMSUNG
%25 İndirim
lenovo
%25 İndirim
SONY
%25 İndirim
WD 2,5" 1TB Elements SE USB3.0 / USB2.0 Taşınabilir Disk
APPLE iPHONE 4 8 GB CEP TELEFONU (SİYAH)
LG 47LM615S 3D LED TV 119 cm FULL HD, 200 HZ,DLNA,3xHDMI,USB,,Dahili HD Uydu alıcı,
SYMANTEC Norton Internet Security 2013 - 1 kullanıcı
SEAGATE 2,5" 1TB Expansion USB3.0 / USB2.0 Taşınabilir Disk
APPLE İPHONE 5 16 GB CEP TELEFONU(BEYAZ)
APPLE İPHONE 5 16 GB CEP TELEFONU(SİYAH)
KASPERSKY Kaspersky Internet Security 2013 (1 Kullanıcı 1Yıl)

Controlling Access



After..

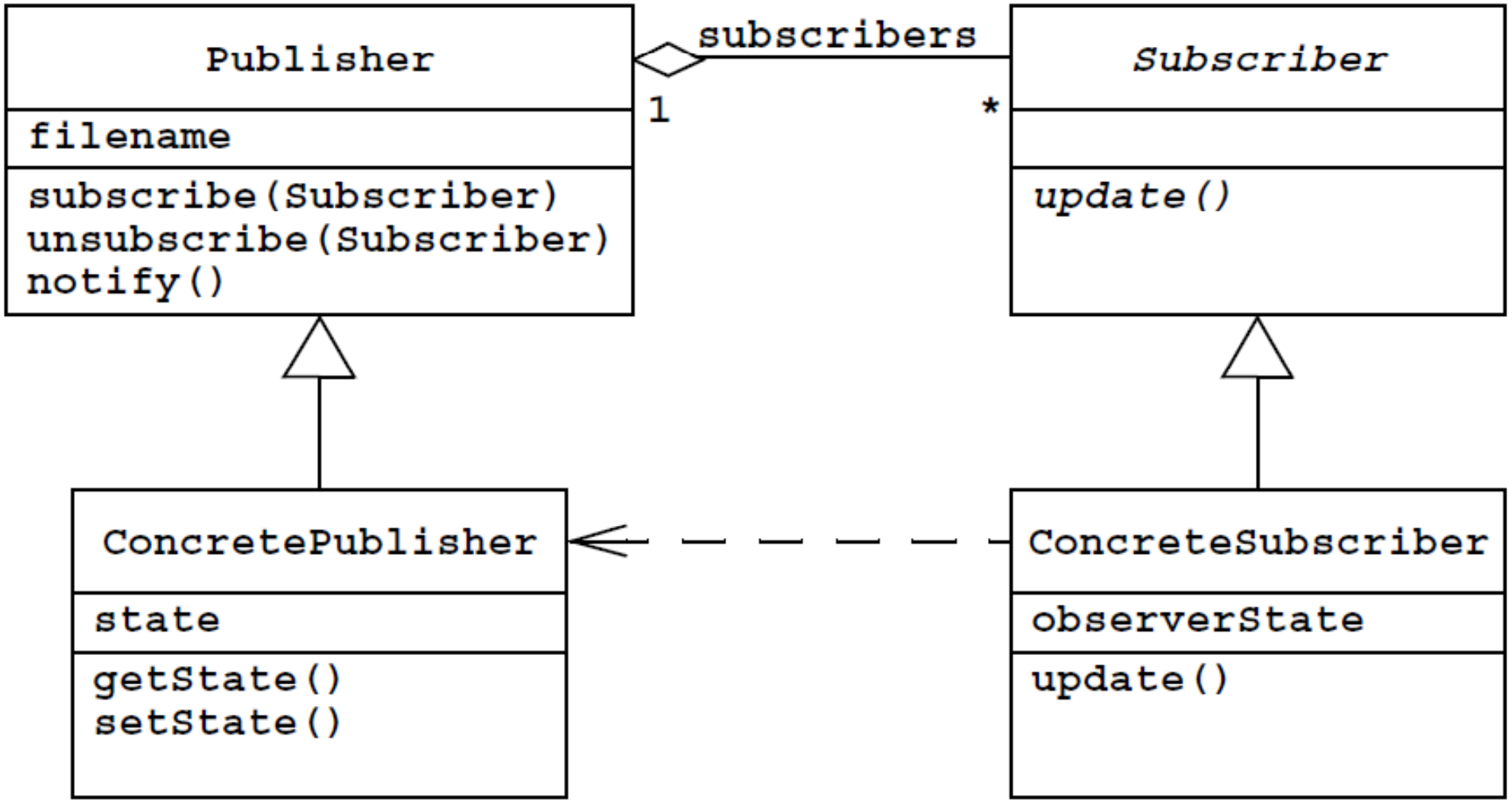

www.vatanbilgisayar.com
APPLE
4.Nesil Ipad-16GB WIFI-Siyah-9.7"Retina-Bluetooth-10 S..
1.200 TL KDV Dahil
Bonus ve Maximum ile 12 x 99,96 TL
AOC
21.5" E2250Swdnk GENİŞ EKRAN LED MONİTÖR
255 TL KDV Dahil
Bonus ve Maximum ile 12 x 21,24 TL
ARNICA
JUPİTER ISLAK KURU ŞARJLI EL SÜPÜRGESİ (14.4 VOLT)
99 TL KDV Dahil
Bonus ve Maximum ile 12 x 8,25 TL
SANDISK
120GB Extreme Sata 3.0 SSD (Okuma 550MB / Yazma 510MB)
47LM615S 3D LED TV 119 cm FULL HD, 200 HZ,DLNA,3xHDMI,USB,,Dahili HD Uydu alıcı,
Kampanya 1.555 TL + KDV 1.599 TL KDV DAHİL
Diğer 2.052 TL + KDV 2.398 TL KDV DAHİL
2,5" 1TB Expansion USB3 . 0 / USB2 . 0 Taşınabilir Disk
Kampanya 145 TL + KDV 169 TL KDV DAHİL
Diğer 119 $ + KDV 255 TL KDV DAHİL
32LN5400 FULL HD 81 cm LED, MCI 100 Hz, 2xHDMI,
Kampanya 762 TL + KDV 899 TL KDV DAHİL
Diğer 1.082 TL + KDV 1.277 TL KDV DAHİL
NOTEBOOK CORE İ5 - 2 . 50GHZ - 8GB - 500GB - 15 . 6 - 1GB - W8 TASINABİLİR BİLGİSAYAR
Kampanya 599 $ + KDV 1.283 TL KDV DAHİL
Diğer 799 $ + KDV 1.711 TL KDV DAHİL
POWERSHOT SX 150 IS 14 . 1 MP 3 " LCD DİJİTAL FOTOĞRAF MAKİNESİ(GRİ)
Kampanya 211 TL + KDV 249 TL KDV DAHİL
Diğer 472 TL + KDV 557 TL KDV DAHİL
MACBOOKAIR NOTEBOOK CORE İ5 1 . 8GHZ - 4GB - 128GBSSD - 13 . 3 - INTEL TASINABİLİR BİLGİSAYAR
Kampanya 2.033 TL + KDV 2.399 TL KDV DAHİL
Diğer 2.929 TL + KDV 3.456 TL KDV DAHİL
Galaxy S4 I9500 16 GB 5" Ekranlı CEP TELEFONU (Beyaz)
1.694 TL 1.999 TL
N580 NOTEBOOK CORE İ3 - 2 . 2GHZ - 4GB - 500GB - 15 . 6 - 1GB - WIN8 TASINABİLİR BİLGİSAYAR
524 $ 1.123 TL
XPERİA Z CEP TELEFONU (SİYAH)
1.567 TL 1.849 TL
WD 2,5" 1TB Elements SE USB3.0 / USB2.0 Taşınabilir Disk
APPLE iPHONE 4 8 GB CEP TELEFONU (SİYAH)
LG 47LM615S 3D LED TV 119 cm FULL HD, 200 HZ,DLNA,3xHDMI,USB,,Dahili HD Uydu alıcı,
SYMANTEC Norton Internet Security 2013 - 1 kullanıcı
SEAGATE 2,5" 1TB Expansion USB3.0 / USB2.0 Taşınabilir Disk
APPLE İPHONE 5 16 GB CEP TELEFONU(BEYAZ)
APPLE İPHONE 5 16 GB CEP TELEFONU(SİYAH)
KASPERSKY Kaspersky Internet Security 2013 (1 Kullanıcı 1Yıl)
SEAGATE 3,5" 2TB Expansion USB3.0 / USB2.0 Harici Disk
CASPER AMLOGIC

Observer Pattern

- A kind of **behavioral** design pattern
- "Define a one-to-many dependency between objects so that when one object changes state, all its dependents are notified and updated automatically."
- Also called "**Publish and Subscribe**"

- **Uses:**
  - Maintaining consistency across redundant state
  - Optimizing batch changes to maintain consistency

Observer – Class Diagram



Observer

- **Purpose**
  - This pattern allows to maintain consistency across the states of one **Publisher** and many **Subscribers**
- **Description**
  - A **Publisher** (called a **Subject** in [Gamma et al., 1994]) is an object whose primary function is to maintain some state; for example, a matrix
  - One or more **Subscribers** (called **Observers** *in [Gamma et al., 1994]) use the state maintained by a* **Publisher***; for* example, to display a matrix as a table or a graph
  - This introduces redundancies between the state of the **Publisher** and the **Subscribers**

Observer

- **Description** (*cont.*)
  - To address this issue, **Subscribers** invoke the **subscribe()** method to register with a **Publisher**
  - Each **ConcreteSubscriber** also defines an **update()** method to synchronize the state between the **Publisher** and the **ConcreteSusbcriber**
  - Whenever the state of the **Publisher** changes, the **Publisher** invokes its **notify()** method, which iteratively invoke each **Subscriber.update()** method

Observer Example


```
interface Observer {
        public void update(float interest);
}

interface Subject {
        public void registerObserver(Observer observer);

        public void removeObserver(Observer observer);

        public void notifyObservers();
}
```

Observer Example


```
class Loan implements Subject {
        private ArrayList<Observer> observers = new ArrayList<Observer>();
        private String type;
        private float interest;
        private String bank;
        ...

        @Override
        public void registerObserver(Observer observer) {
                observers.add(observer);
        }

        @Override
        public void removeObserver(Observer observer) {
                observers.remove(observer);
        }

        @Override
        public void notifyObservers() {
                for (Observer ob : observers) {
                        System.out.println("Notifying Observers on change in
                                                Loan interest rate");
                        ob.update(this.interest);
                }
        }
}
```

Observer Example


```
class Newspaper implements Observer {
	@Override
	public void update(float interest) {
		System.out.println("Newspaper: Interest Rate updated, new Rate is: "
				+ interest);
	}
}

class Internet implements Observer {
	@Override
	public void update(float interest) {
		System.out.println("Internet: Interest Rate updated, new Rate is: "
				+ interest);
	}
}
```

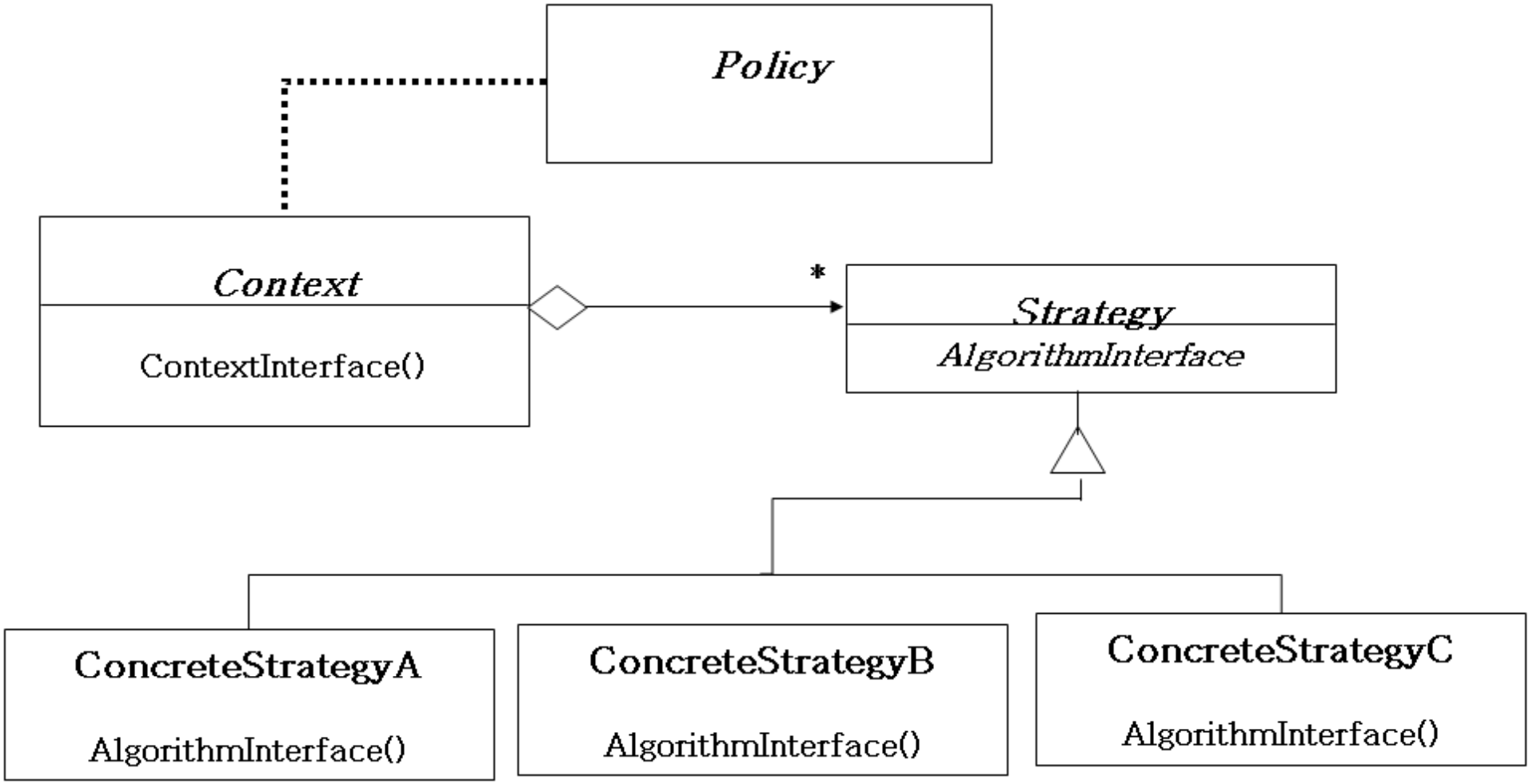
Observer Example


```
public class ObserverTest {

	public static void main(String args[]) {
		// this will maintain all loans information
		Newspaper printMedia = new Newspaper();
		Internet onlineMedia = new Internet();

		Loan personalLoan = new Loan("Personal Loan", 12.5f,
				"Standard Charterd");
		personalLoan.registerObserver(printMedia);
		personalLoan.registerObserver(onlineMedia);
		personalLoan.setInterest(3.5f);

	}
}
```

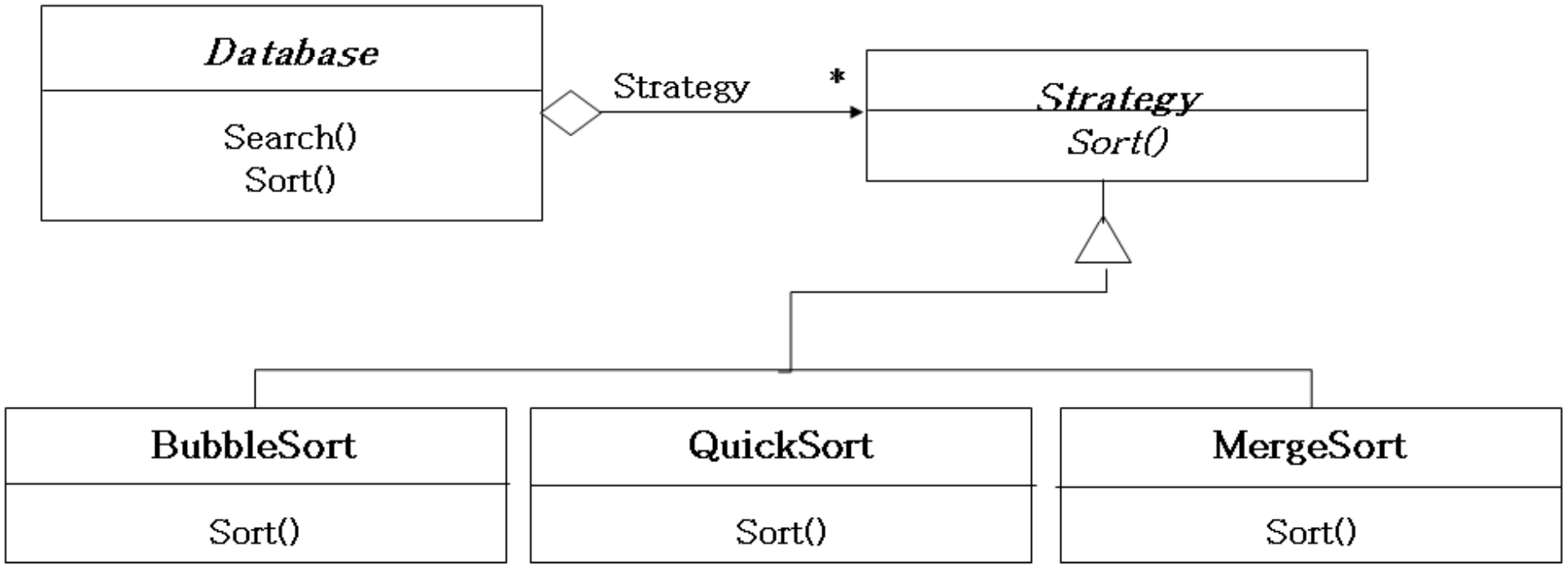
Strategy Pattern

- A kind of **behavioral** design pattern
- Many different algorithms exists for the same task
- **Examples:**
  - Breaking a stream of text into lines
  - Parsing a set of tokens into an abstract syntax tree
  - Sorting a list of customers
- The different algorithms will be appropriate at different times
  - Rapid prototyping vs. delivery of final product
- We want to decouple an algorithm from its implementation(s)
- If we need a new algorithm, we want to add it easily without disturbing the application using the algorithm

Strategy – Class Diagram



Strategy – Example 1



Strategy – Example 2

- An example of Strategy pattern encapsulating multiple implementation of the IDEA encryption algorithm
- The Message and *IDEA classes cooperate to realize the* encryption of plain text. The selection of an implementation can be done dynamically