Overview of the Design Patterns

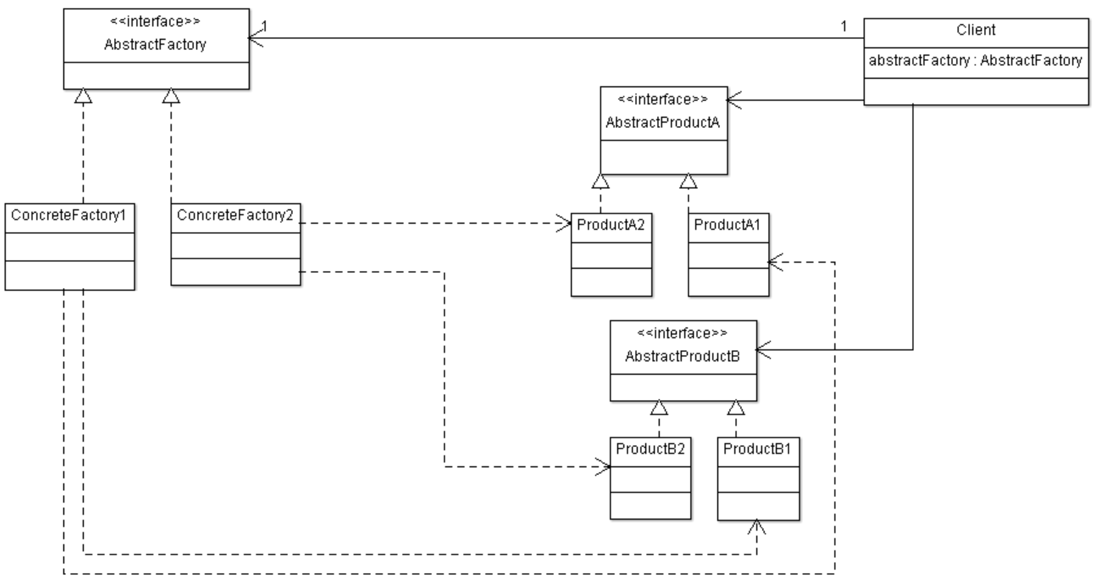
This document gives an overview of the Design Patterns used in the Design Pattern Detection software.

1. Abstract Factory

1.1 participants
AbstractFactory, ConcreteFactory, AbstractProduct, ConcreteProduct, Client

1.2 Collaborations
Client has one attribute of type AbstractFactory, with cardinalities 1 at each side.

1.3 A Diagram



1.4 the pattern xml

```
<?xml version="1.0" encoding="UTF-8"?>
<patterns>
<pattern name="Abstract Factory" family="Abstract Factory">
<nodes>
<node id="Client">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
<node id="AbstractFactory">
<node.rule nodeType="INTERFACE"/>
```

```
</node>
<node id="AbstractProductA">
<node.rule nodeType="INTERFACE"/>
</node>
<node id="AbstractProductB">
<node.rule nodeType="INTERFACE"/>
</node>
<node id="ConcreteFactory1">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
<node id="ConcreteFactory2">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
<node id="ProductA1">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
<node id="ProductA2">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
<node id="ProductB1">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
<node id="ProductB2">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
</nodes>
<relations>
<relation node1="Client" node2="AbstractFactory">
<relation.rule relationType="ASSOCIATES_WITH" cardinalityLeft="1"
cardinalityRight="1"/>
<relation.rule relationType="HAS_ATTRIBUTE_OF" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="Client" node2="AbstractProductA">
<relation.rule relationType="ASSOCIATES_WITH" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="Client" node2="AbstractProductB">
<relation.rule relationType="ASSOCIATES_WITH" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="ProductA1" node2="AbstractProductA">
<relation.rule relationType="IMPLEMENTS" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="ProductA2" node2="AbstractProductA">
```

```
<relation.rule relationType="IMPLEMENTS" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="ProductB1" node2="AbstractProductB">
<relation.rule relationType="IMPLEMENTS" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="ProductB2" node2="AbstractProductB">
<relation.rule relationType="IMPLEMENTS" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="ConcreteFactory1" node2="AbstractFactory">
<relation.rule relationType="IMPLEMENTS" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="ConcreteFactory2" node2="AbstractFactory">
<relation.rule relationType="IMPLEMENTS"  cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="ConcreteFactory1" node2="ProductA1">
<relation.rule relationType="DEPENDS_ON" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="ConcreteFactory2" node2="ProductA2">
<relation.rule relationType="DEPENDS_ON" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="ConcreteFactory1" node2="ProductB1">
<relation.rule relationType="DEPENDS_ON" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="ConcreteFactory2" node2="ProductB2">
<relation.rule relationType="DEPENDS_ON" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
</relations>
</pattern>
</patterns>
```

1.5 Other recommendations (GoF p. 102)

a. Concrete factories are singletons. You can implement them like this.
b. Abstract factory should be an interface and not an abstract class.

2. Builder

3. Factory Method

4. Prototype

5. Singleton

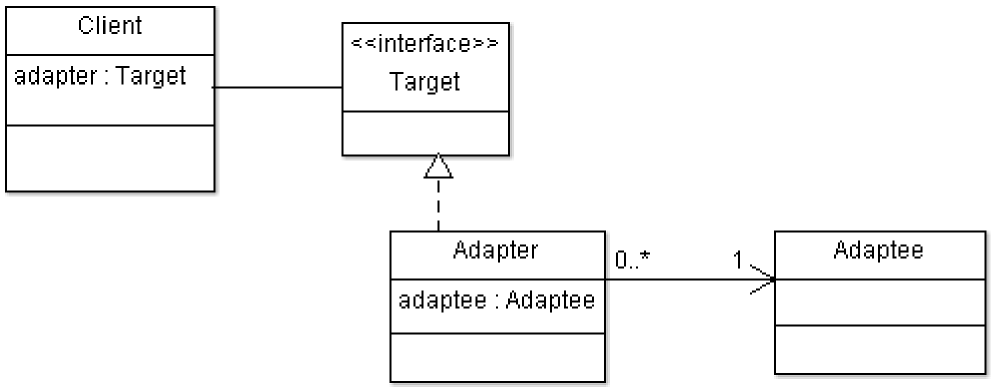
6. Adapter

A. Object Adapter

6.1 Participants
Client, Target, Adapter, Adaptee

6.2 Collaborations
The Client has an attribute of type Target.
The Adapter has one or more attribute of type Adaptee.

6.3 Diagram



6.4 pattern xml

```
<?xml version="1.0" encoding="UTF-8"?>
<patterns>
```

```
<pattern name="Object Adapter" family="Adapter">
<nodes>
<node id="Client">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
<node id="Target">
<node.rule nodeType="INTERFACE"/>
</node>
<node id="Adapter">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
<node id="Adaptee">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
</nodes>
<relations>
<relation node1="Client" node2="Target">
<relation.rule relationType="ASSOCIATES_WITH" cardinalityLeft="1"
cardinalityRight="1"/>
<relation.rule relationType="HAS_ATTRIBUTE_OF" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="Adapter" node2="Target">
<relation.rule relationType="IMPLEMENTS" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="Adapter" node2="Adaptee">
<relation.rule relationType="ASSOCIATES_WITH"
cardinalityLeft="0..*" cardinalityRight="1"/>
<relation.rule relationType="HAS_ATTRIBUTE_OF" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
</relations>
</pattern>
</patterns>
```

6.5. Recommendations

B. Class Adapter

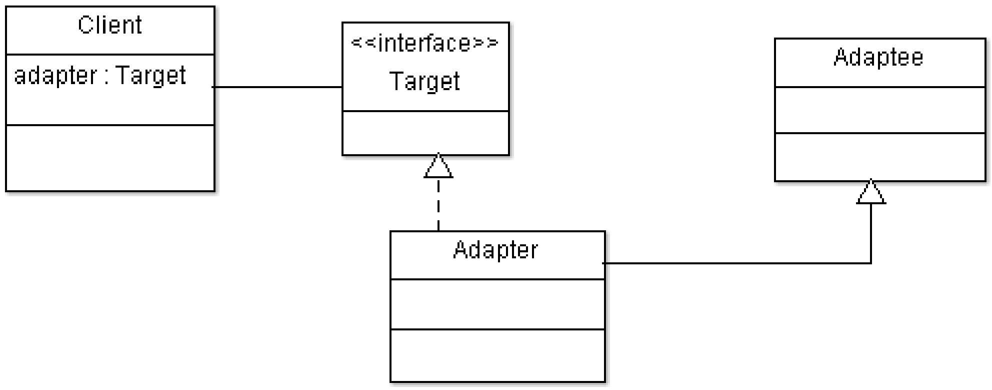
6.1 Participants

Client, Target, Adapter, Adaptee
In the case of a class adapter, the target must be an interface in a java implementation. The Adapter implements from the Target and inherits from the Adaptee.

6.2 Collaborations

The Client has an attribute of type Target.

6.3 Diagram



6.4 pattern xml

```
<?xml version="1.0" encoding="UTF-8"?>
<patterns>
<pattern name="Object Adapter" family="Adapter">
<nodes>
<node id="Client">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
<node id="Target">
<node.rule nodeType="INTERFACE"/>
</node>
<node id="Adapter">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
<node id="Adaptee">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
</nodes>
<relations>
<relation node1="Client" node2="Target">
<relation.rule relationType="ASSOCIATES_WITH" cardinalityLeft="1"
cardinalityRight="1"/>
<relation.rule relationType="HAS_ATTRIBUTE_OF" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="Adapter" node2="Target">
```

```
<relation.rule relationType="IMPLEMENTS" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="Adapter" node2="Adaptee">
<relation.rule relationType="INHERITS_FROM" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
</relations>
</pattern>
</patterns>
```

6.5 Recommendations

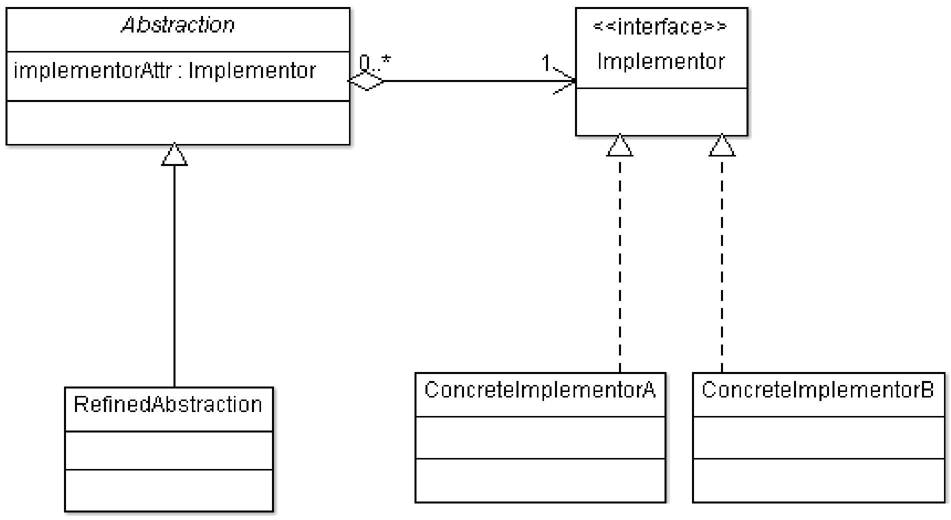
Set the adaptee attribute of the object adapter class Adapter with private visibility.

7. Bridge

1.1 Participants
Abstraction, Implementor, RefindeAbstraction, ConcreteImplementorA, ConcreteImplementorB
As the Abstraction class has an attribute of type Implementor, it must be an abstract class.

1.2 Collaborations
The Abstraction class can have multiple instances of the Implementor class (cardinalities 0..* and 1)

1.3 Diagram



1.4 xml scheme

```
<?xml version="1.0" encoding="UTF-8"?>
<patterns>
<pattern name="Bridge" family="Bridge">
<nodes>
<node id="Abstraction">
<node.rule nodeType="ABSTRACT_CLASS"/>
</node>
<node id="Implementor">
<node.rule nodeType="INTERFACE"/>
</node>
<node id="RefinedAbstraction">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
```

```
<node id="ConcreteImplementor">
<node.rule nodeType="CONCRETE_CLASS"/>
</node>
</nodes>
<relations>
<relation node1="Abstraction" node2="Implementor">
<relation.rule relationType="ASSOCIATES_WITH"
cardinalityLeft="0..*" cardinalityRight="1"/>  <relation.rule
relationType="HAS_ATTRIBUTE_OF" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="ConcreteImplementor" node2="Implementor">
<relation.rule relationType="IMPLEMENTS" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
<relation node1="RefinedAbstraction" node2="Abstraction">
<relation.rule relationType="INHERITS_FROM" cardinalityLeft="1"
cardinalityRight="1"/>
</relation>
</relations>
</pattern>
</patterns>
```

1.5 Recommendations

8. Composite
9. Decorator
10. Facade
11. Flyweight
12. Proxy
13. Chain Of Responsibility
14. Command
15. Interpreter
16. Iterator
17. Mediator
18. Memento
19. Observer
20. State
21. Strategy

22. Template Method

23. Visitor