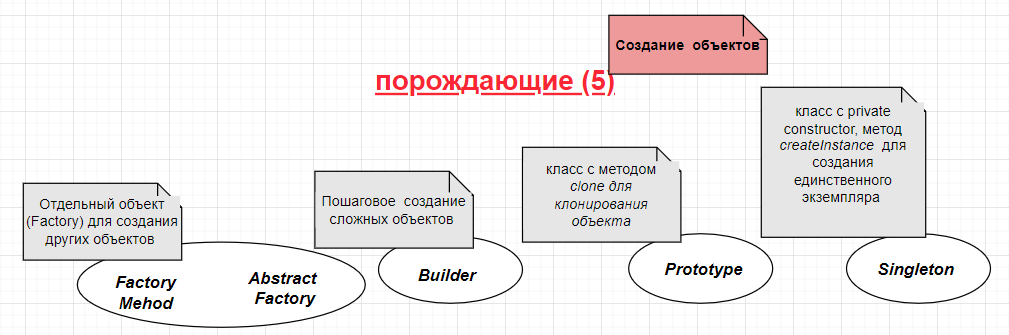
**Паттерны проектирования**

**ИСиТ,ПОИБМС-2**

**Лекция 11**

**Итоги**

**Всего 22 паттерна**

***Factory Method***

1. *IProduct, ConcreteProduct\_:IProduct*
2. *ICreator, ICreator.createProduct():IProduct*
3. *ConcreteCreator\_:ICreator, ConcreteCreator\_.createProduct():IProduct*

***Abstract Factory***

1. *IProduct\_,…*
2. *ConcreteProduct\_:IProduct\_,…*
3. *IAbstractFactory, IAbstractFactory.CreateProduct\_:IProduct\_,…*
4. *ConcreteFactory\_:IAbstractFactory*

***Builder***

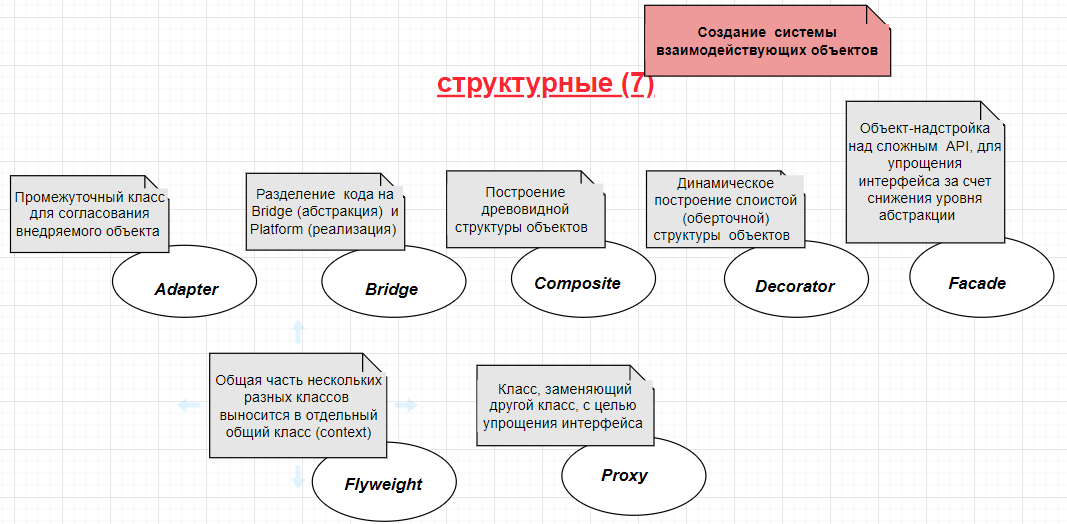
1. *Product\_,…*
2. *IBuilder, IBuilder.step\_(),…*
3. *ConcreteBuilder\_:IBuilder,…, ConcreteBuilder\_,getResult():Product\_*
4. *Director, new Director.Director(typeBuild), Director.builder:IBuilder*

***Prototype***

1. *IProduct, IProduct.clone(id):IProduct*
2. *Product\_:IProduct…, Product\_.clone(id):IProduct*

***Singleton***

1. *Singleton, private Singleton.Singleton(), static Singleton.createInstance(): Singleton.*

****

***Adapter***

1. *IExternal, IExtenal.met(X):Y*
2. *Client, new Client.Client(IExternal)*
3. *S, S.metS(Z):W*
4. *Adapter:IExternal, Adapter.met(X){W =S.metS; …,return Y}*
5. *New Client(Adapter)*

***Bridge***

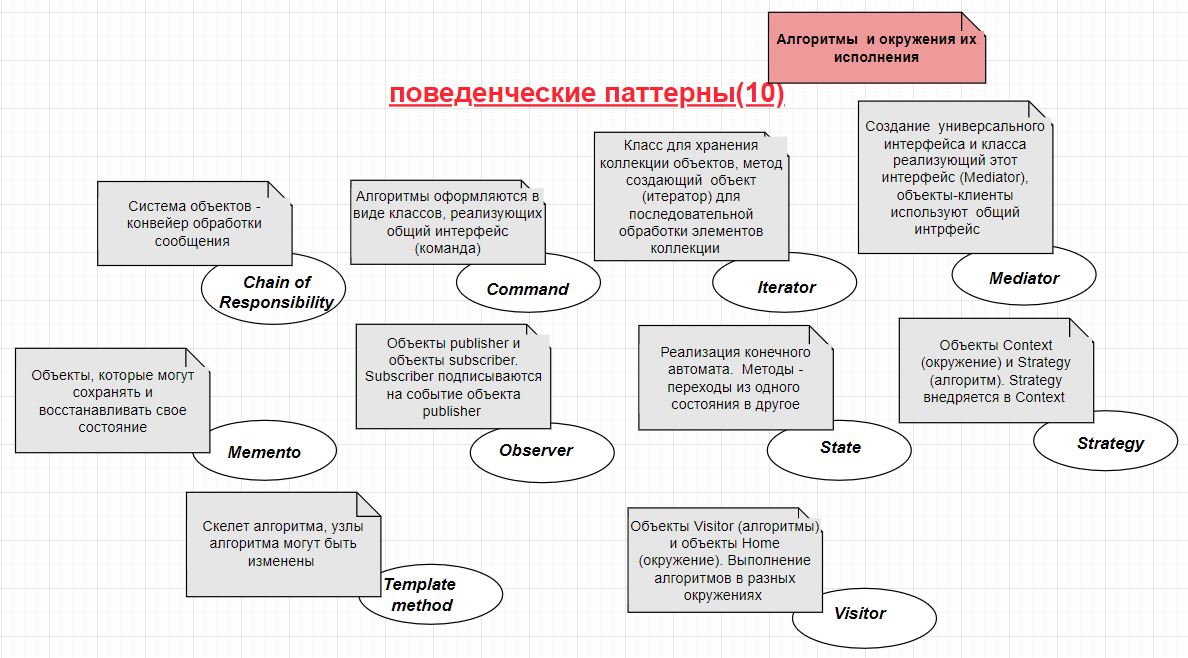
1. Bridge =  *IProperty, IAbsraction, IAbstraction.IProperty*
2. Platform = *Implementation:IPoperty, Realisation:IAbstraction*,
3. *Client, new Client(IAbstraction)*

***Composite***

1. *IComponent, IComponent.execute()*
2. *Composite:IComposite, Composite.children:IComposite[], Composite.add(IComponent), Composite.del(IComponent), getChildren():IComponent[].*

***Decorator***

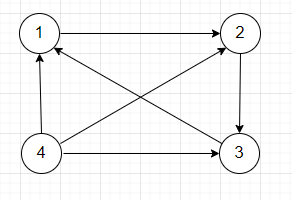
1. *IComponent, IComponent.execute()*
2. *Component\_:IComponent*
3. *Decorator:IComponent, Decorator.wrapped, new Decarator.Decarator(IComponent)*
4. *ConreteDecorator\_:Decorator*

****

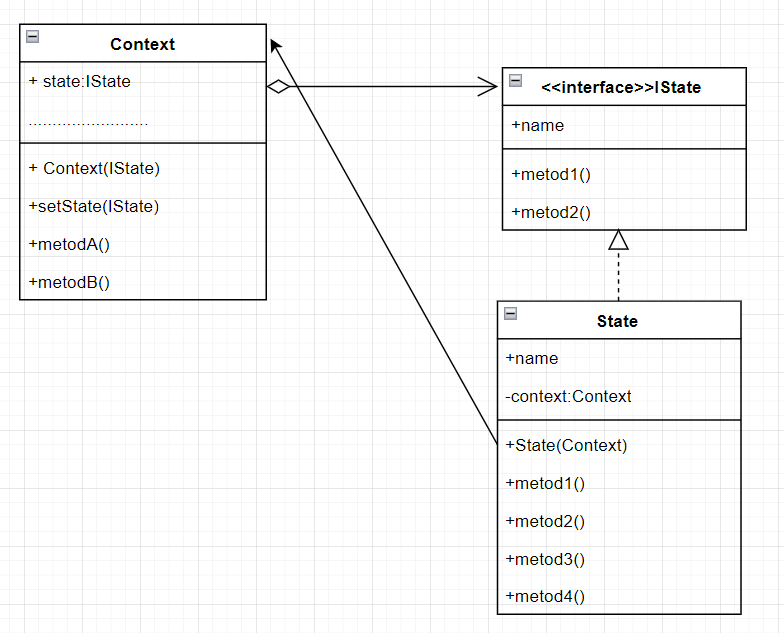
1. **Поведенческие паттерны:** взаимодействие программ.
2. **Поведенческие паттерны:** Chain of Responsibility (цепочка обязанностей), Command (Команда), Iterator (Итератор), Mediator (Посредник), Memento (Снимок), Observer (Обозреватель), State (Состояние), Strategy (Стратегия), Template method (шаблонный метод), Visitor (Посетитель).

**State**

1. **State:** программныйобъект, реализующий конечный автомат



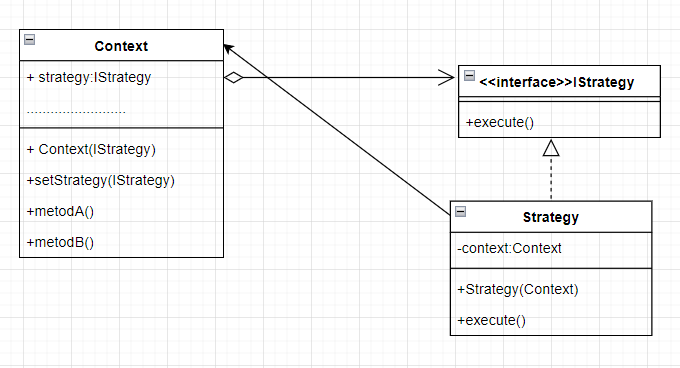
1. **State:UML**



1. **State:** автоматное программирование

**Strategy**

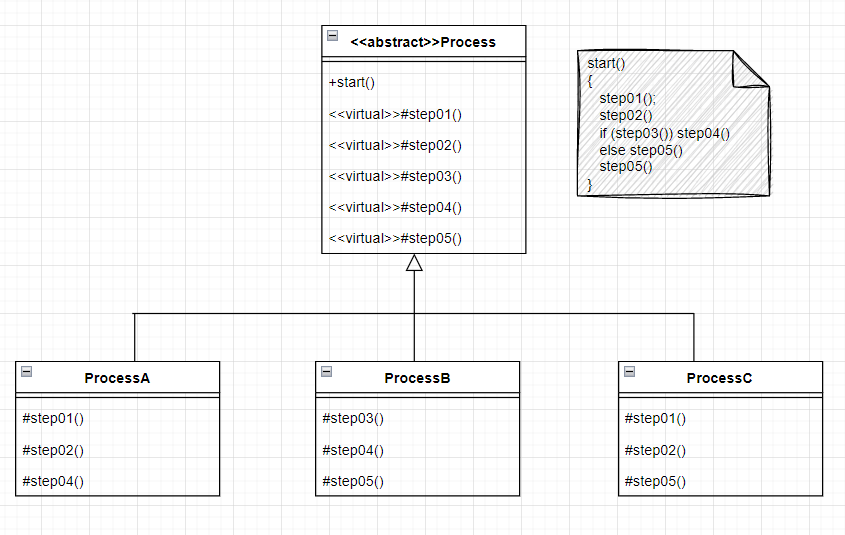
1. **Strategy:** алгоритм переносится в отдельный класс и динамически внедряется в контекст выполнения
2. **Strategy: UML**



1. **Strategy:** XMLHTTPREQUEST, callback-функция

**Template Method**

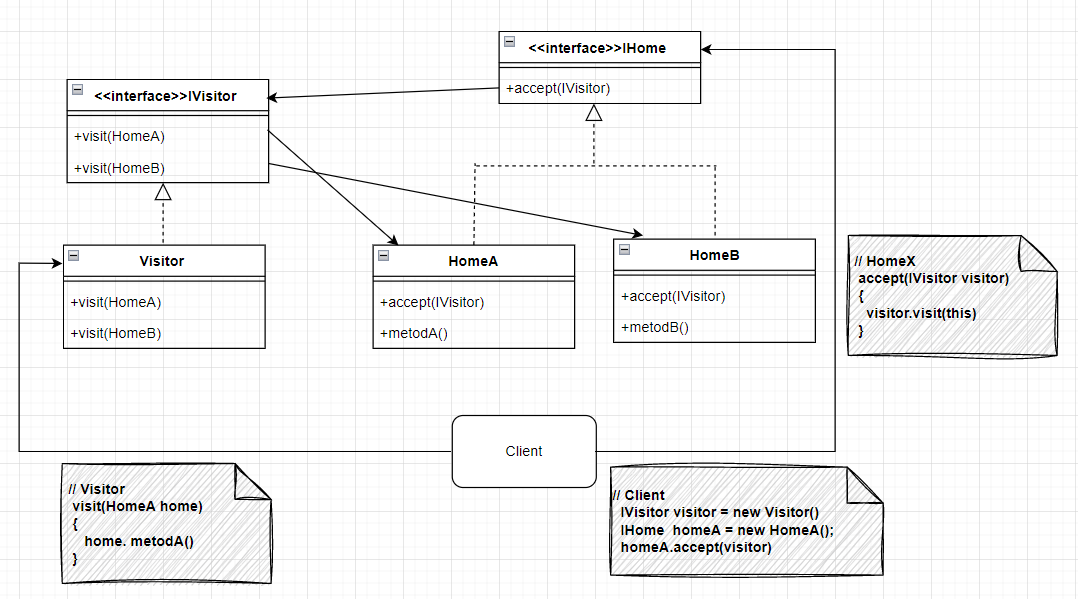
1. **Template Method:** скелет алгоритма, шаги которого могут быть переопределены



1. **Template Method:** Global.asaх**,** триггер в СУБД

**Visitor**

1. **Visitor:** набор алгоритмов (Visitor.visit()), которые могут исполнять методы разных окружений (HomeX)
2. **Visitor:** конкретный visitor «знает» в какой среде он исполняется



1. **КОНЕЦ**