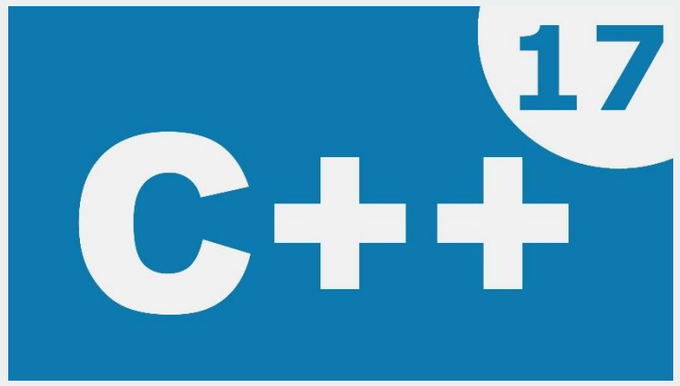
OTUS

ОНЛАЙН-
ОБРАЗОВАНИЕ

OTUS

Десятиминутка C++17



… самостоятельно

https://ps-group.github.io/cxx/cxx17

Сегодня

GRASP

- **Слабое зацепление (Low Coupling)**
- **Высокая связность (High Cohesion)**
- **Создатель (Creator)**
- **Информационный эксперт (Information Expert)**
- **Контроллер (Controller)**

Продолжаем пополнять копилку умных слов. Вот коллеги удивятся.

Сегодня

GRASP

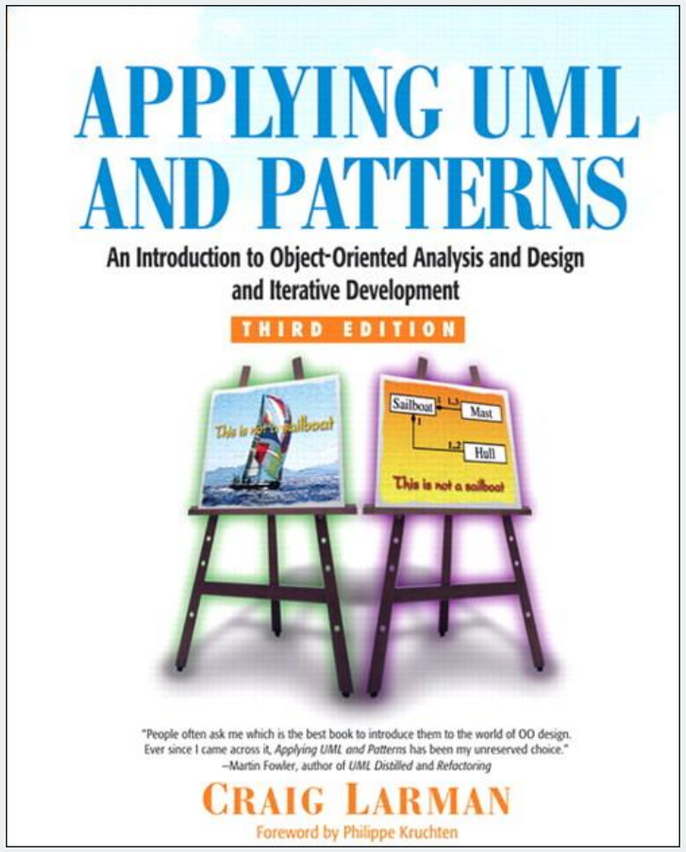
- **Полиморфизм (Polymorphism)**
- **Чистая выдумка (Pure Fabrication)**
- **Посредник (Indirection)**
- **Устойчивость к изменениям (Protected Variations)**

1 **Продолжаем пополнять копилку умных слов. Вот коллеги удивятся.**

GRASP

General Responsibility Assignment Software Patterns (Principles)

Общие принципы распределения обязанностей в ПО

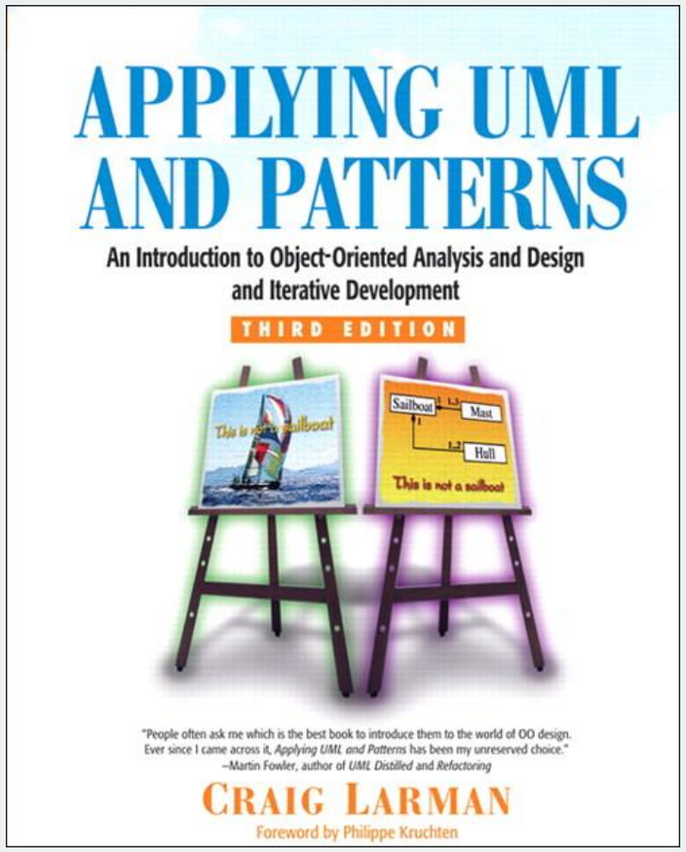


Принципы или паттерны? Паттерны или принципы?

GRASP

General Responsibility Assignment Software Patterns (Principles)

Общие принципы распределения обязанностей в ПО

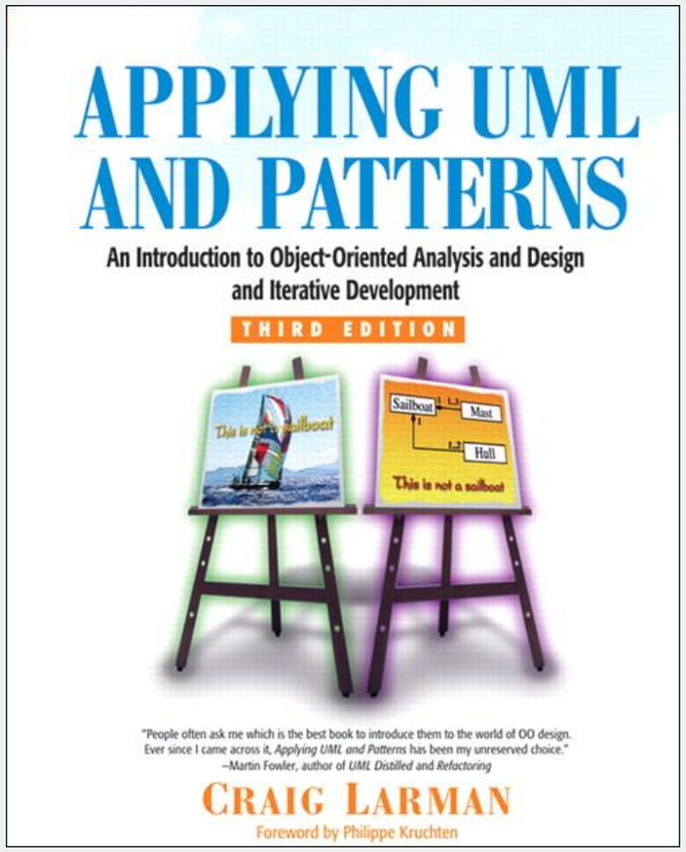


- Маркеры для мозга в начале проектирования

GRASP

General Responsibility Assignment Software Patterns (Principles)

Общие принципы распределения обязанностей в ПО

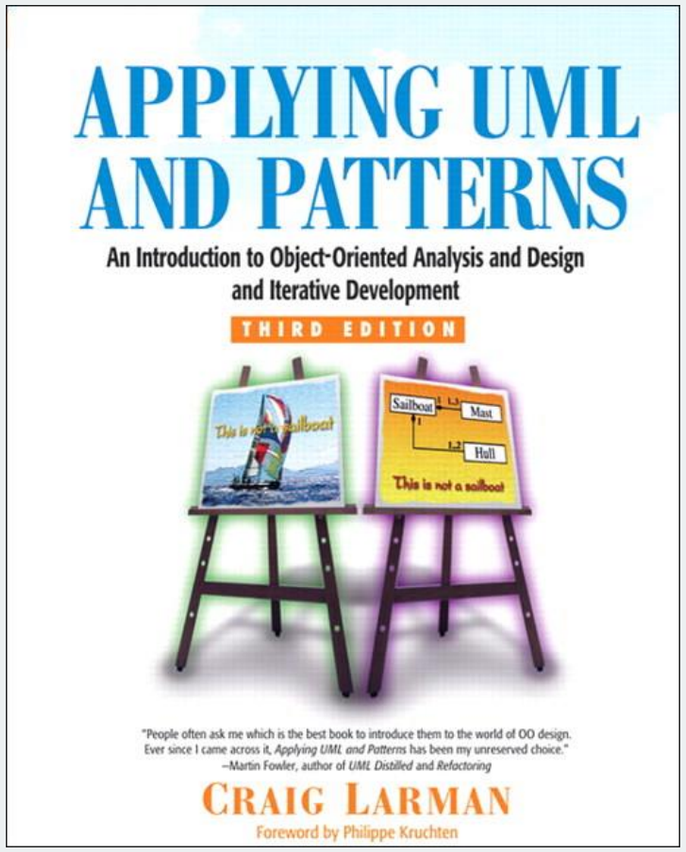


- Маркеры для мозга в начале проектирования
- Авторитет среди коллег

GRASP

General Responsibility Assignment Software Patterns (Principles)

Общие принципы распределения обязанностей в ПО

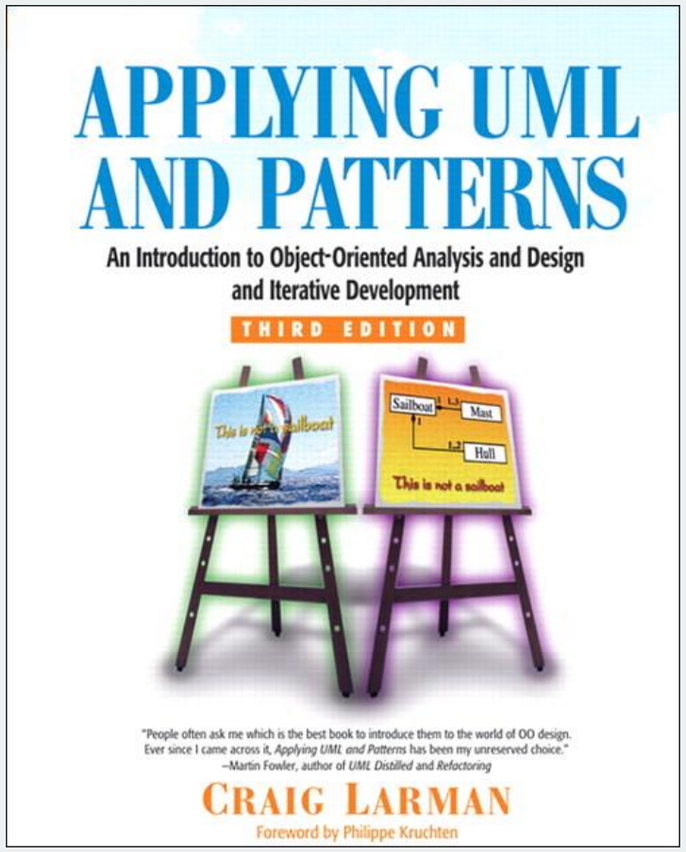


- Маркеры для мозга в начале проектирования
- Авторитет среди коллег
- Чтение соответствующей литературы (редко)

GRASP

General Responsibility Assignment Software Patterns (Principles)

Общие принципы распределения обязанностей в ПО



- Маркеры для мозга в начале проектирования
- Авторитет среди коллег
- Чтение соответствующей литературы (редко)
- Чтение документации (крайне редко).

Low Coupling

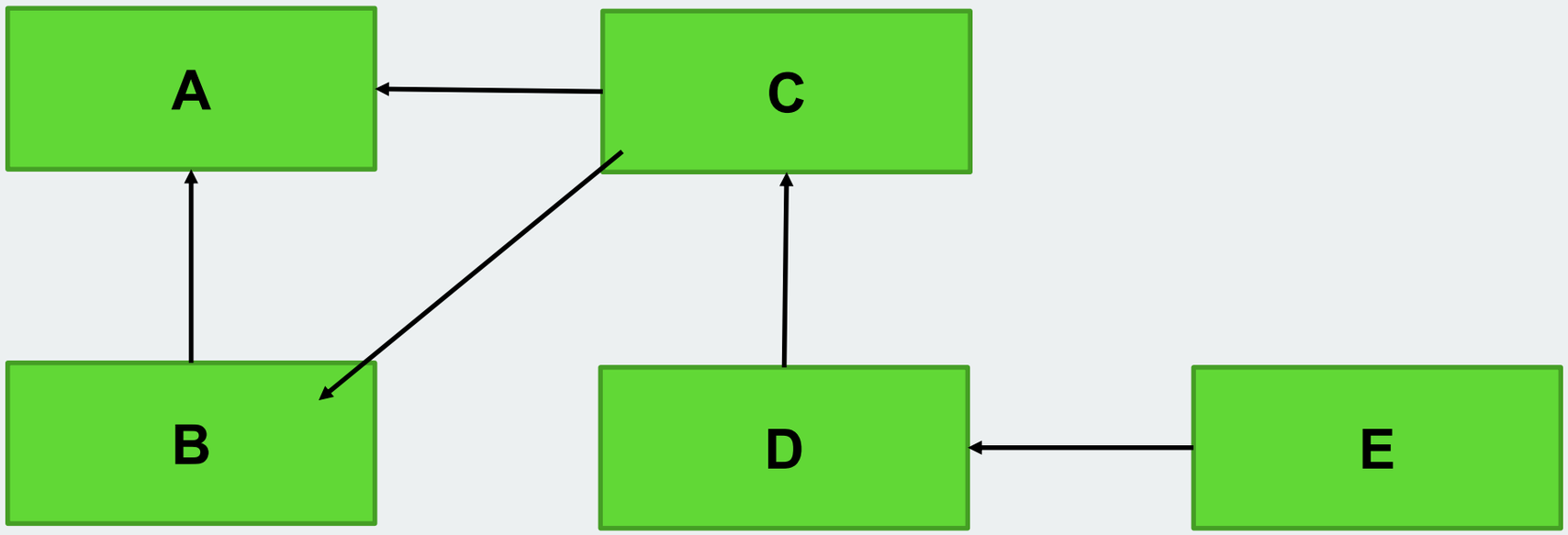
Слабое зацепление

- Снежный ком из изменений
- Наводит на мысли о связях вообще
- Позволяет сравнить разные реализации
- А как снизить количество связей?
- Стоит ли доводить до абсурда?

Зацепление – звучит не особо, но по сути верно. Можно представить себе репейник.

Low Coupling

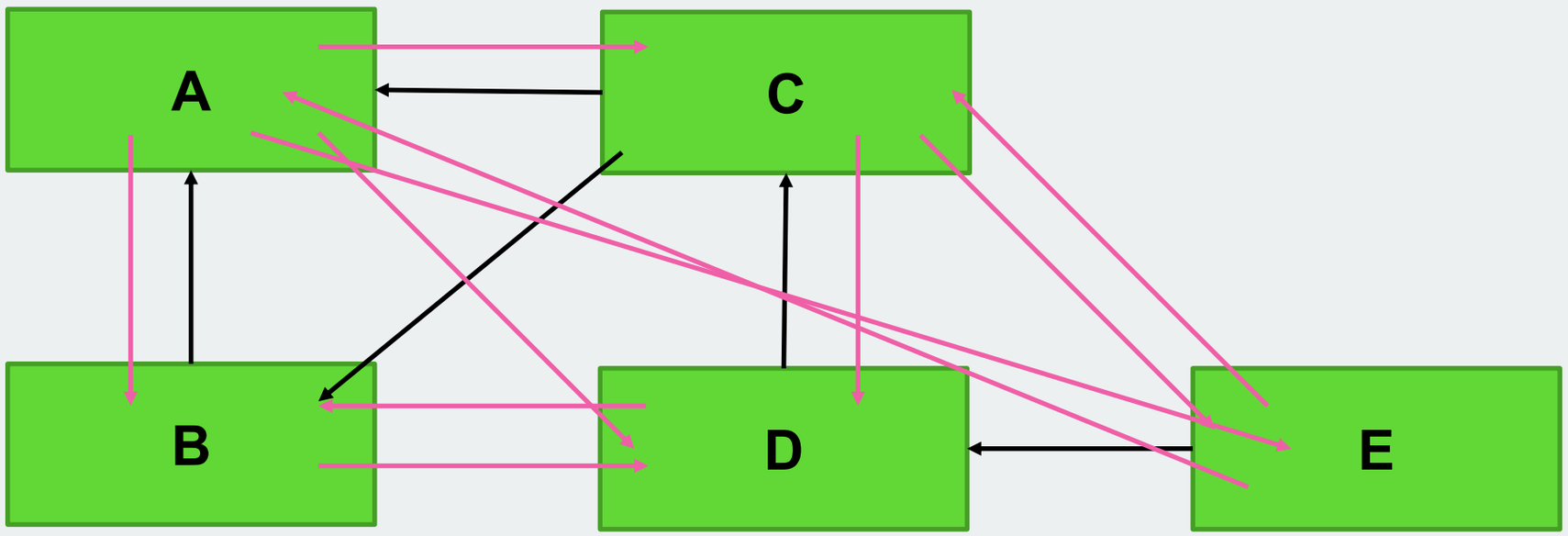
Слабое зацепление



3 **Зацепление – звучит не особо, но по сути верно. Можно представить себе репейник.**

Low Coupling

Слабое зацепление



3 **Зацепление – звучит не особо, но по сути верно. Можно представить себе репейник.**

Low Coupling

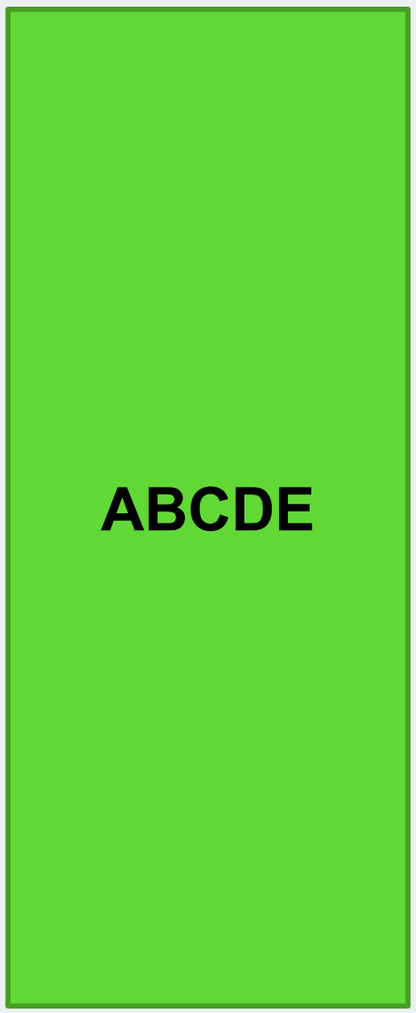
Слабое зацепление

- Дружба (friend)
- Наследование (наследник -> базовый класс)
- Агрегация / композиция
- Передача через параметр
- Глобальная переменная
- Параметр шаблона

3 **Зацепление – звучит не особо, но по сути верно. Можно представить себе репейник.**

Low Coupling

Слабое зацепление

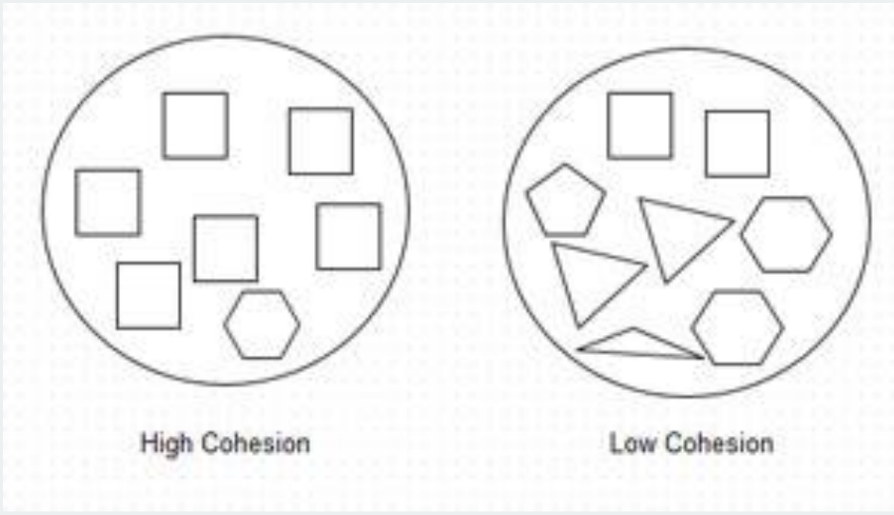


Зацепление – звучит не особо, но по сути верно. Можно представить себе репейник.

High Cohesion

Высокая связность

Single Responsibility (SOLID)



- Один супер-класс на все случаи жизни
- «И швец, и жнец, и на дуде игрец»
- Кластеризация функционала на этапе проектирования
- Чем меньше кирпичик, тем проще его менять

Связность элементов внутри модуля – это хорошо. В отличии от зацеплений модулей.

Creator

«Тебе нужно – ты и создавай»

Класс должен создавать экземпляры:

- агрегированных классов
- классов, которые содержит
- классов, который активно использует
- классов, для инициализации которых обладает наибольшей информацией

Создание объектов – так себе акт творения. Но всё-таки делать это надо умеючи.

Creator

«Тебе нужно – ты и создавай»

Класс должен создавать экземпляры:

- агрегированных классов
- классов, которые содержит
- классов, который активно использует
- классов, для инициализации которых обладает наибольшей информацией

Потому что иначе придётся создавать дополнительные связи!

Создание объектов – так себе акт творения. Но всё-таки делать это надо умеючи.

Creator

«Тебе нужно – ты и создавай»

- Кто создаёт экземпляры?
- Где они хранятся?
- В случае композиции всё понятно.
- А кому они вообще нужны (внутренние)?
- Кто обладает информацией о том, как их создавать?
- Иногда даже есть специальный объект – фабрика.
- Удачное применение снижает количество связей.

Создание объектов – так себе акт творения. Но всё-таки делать это надо умеючи.

Information Expert

Информационный эксперт

- Кто должен выполнять операцию?
- Тот, у кого есть для этого максимум информации.
- Повышает связность, снижает зацепление.
- Повышает инкапсуляцию – не нужно отдавать информацию.
- Если их несколько – что-то не так с архитектурой (DRY).
- Довольно очевидный принцип.

Information Expert

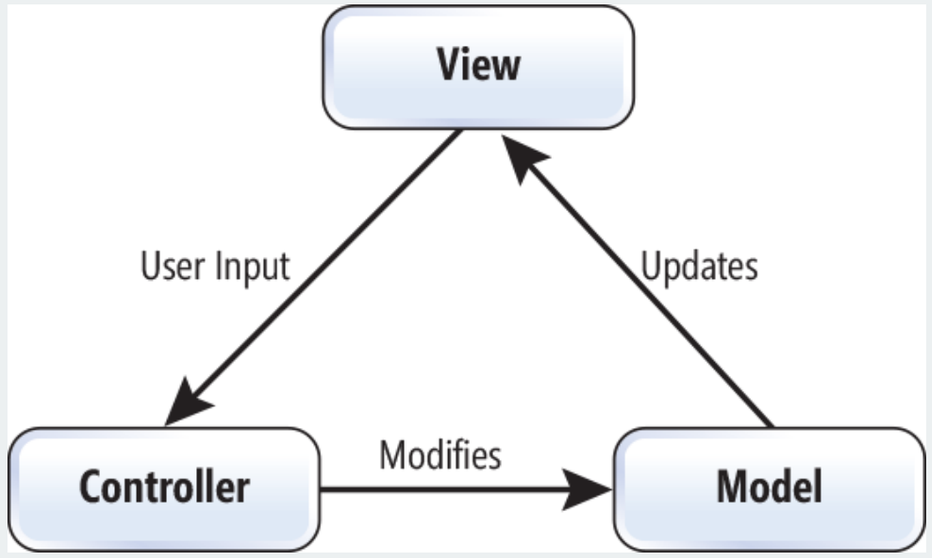
Информационный эксперт

- Кто должен выполнять операцию?
- Тот, у кого есть для этого максимум информации.
- Повышает связность, снижает зацепление.
- Повышает инкапсуляцию – не нужно отдавать информацию.
- Если их несколько – что-то не так с архитектурой (DRY).
- Довольно очевидный принцип.

Потому что иначе придётся создавать дополнительные связи!

Controller

Контроллер



- Не выполняет работу сам
- Делегирует задачи
- Отвечает за внешнее взаимодействие
- Реализация вариантов использования
- Хранение состояния
- Может быть несколько

Потому что иначе придётся создавать дополнительные связи!

Контроллер нужен для готовности к внешним изменениям.

Polymorphism

Полиморфизм

- Обработка альтернатив
- Выделение общей логики
- Расширение базового функционала
- Возможность подменить реализацию
- Принцип подстановки Лисков
- YAGNI
- Решение принимается на этапе планирования
- Много полиморфизма тоже плохо.

Pure Fabrication

Чистая выдумка

- Что делать, когда зашли в тупик?

Не получается – включи воображение!

Pure Fabrication

Чистая выдумка

- Что делать, когда зашли в тупик?
- Любую проблему можно решить через добавление ещё одного уровня косвенности.

Pure Fabrication

Чистая выдумка

- Что делать, когда зашли в тупик?
- Любую проблему можно решить через добавление ещё одного уровня косвенности.
- … кроме проблемы слишком большого количества уровней косвенности.

Pure Fabrication

Чистая выдумка

Есть в продукте, но нет в требованиях

- handler на файл
- логгер - отладочная информация
- модуль доступа к БД

Indirection

Посредник

- Частный случай чистой выдумки
- Частный случай контроллера
- Снижает зацепление модулей
- Скрывает детали реализации
- Паттерн «Фасад»

Общение через посредника является довольно удобным, хоть и медленным.

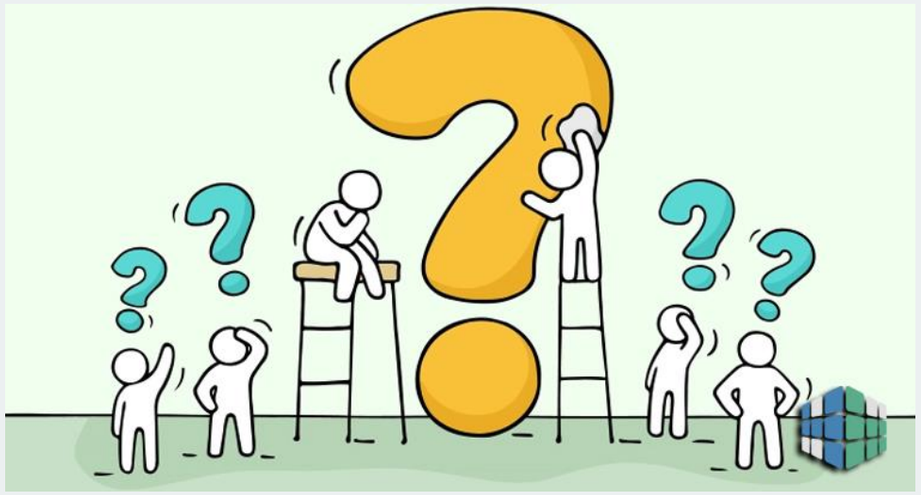
Protected Variations

Устойчивость к изменениям

- Как защититься от изменений?
- Изолирование изменяемых частей
- Защита через интерфейс взаимодействия
- Полиморфизм
- Конфиги
- Скрипты
- Система плагинов – популярное и отличное решение.

Ничто не является более предсказуемым, чем наличие изменений.

Проектирование



вопросы, на которые нужно ответить:

- нет ли лишних связей?
- нет ли модулей, которые нужно разбить на части?
- где будут создаваться бизнес-объекты?
- кто будет являться информационным экспертом для задачи?
- как можно избежать проблем при изменениях?
- какие альтернативы рассмотрены?
- не осталось ли белых пятен?

Проектирование

… никогда не заканчивается

Ответы на вопросы