

让设计与变化共舞 ACT敏捷教练工具箱之 柔性设计

麦宇安 2018 中国·上海



目录

- 1 ACT简介
- 2 敏捷转型中的设计现状
- 3 柔性设计概念
- 4 柔性设计的实践
- 5 Q&A



什么是ACT?

- ACT (Agile Coach Toolbox) 是一个敏捷教练导入工具箱
- 它来自于一线教练的实践总结
- 目的是辅助教练从全生命周期的视角展开工作
- 让团队快速形成60分的战斗力
- 给敏捷教练指明成长通道



ACT Group的使命

- 服务转型中的企业
 - 有效落地敏捷实践，
- 服务成长中的教练
 - 快速提升实战技能。



自我介绍



-2013

2014 2015

2016

2017

2018

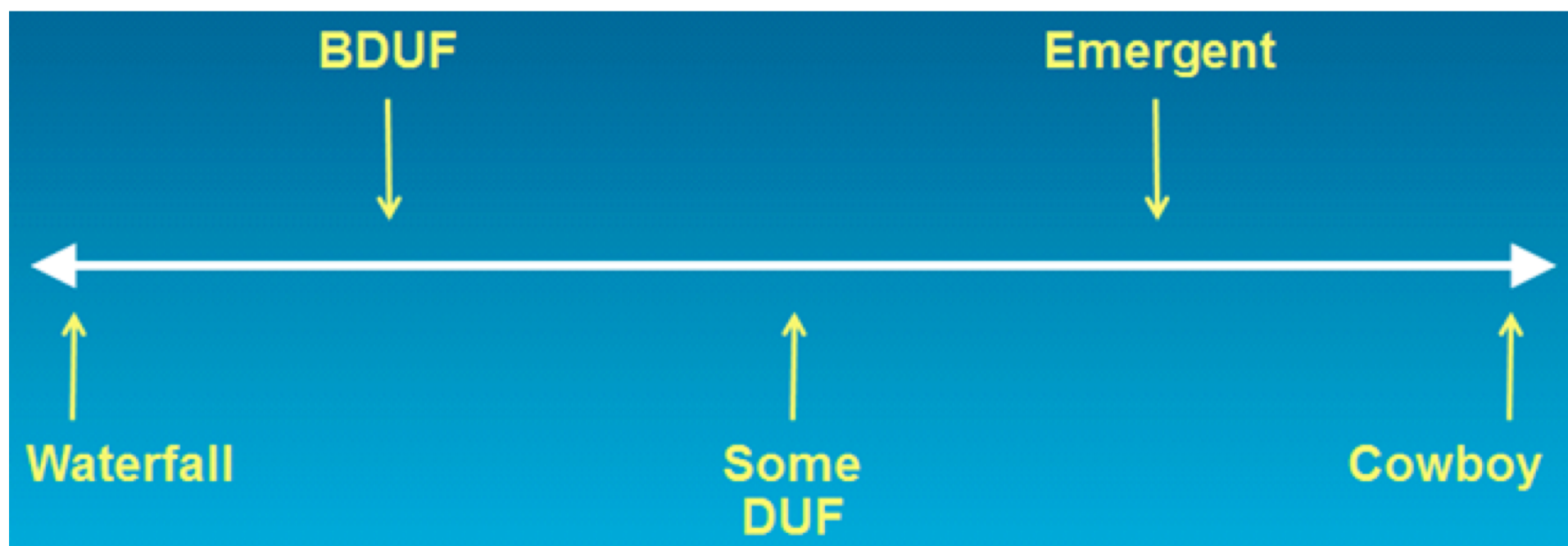


没有银弹

我们预见，从当前开始的十年之内，将不会看到任何银弹，无论是在技术上或管理上，都不会有任何单一的重大突破，能够保证在生产力、可靠度或简洁性上获得改善，甚至，连一个数量级的改善都不会有。

。

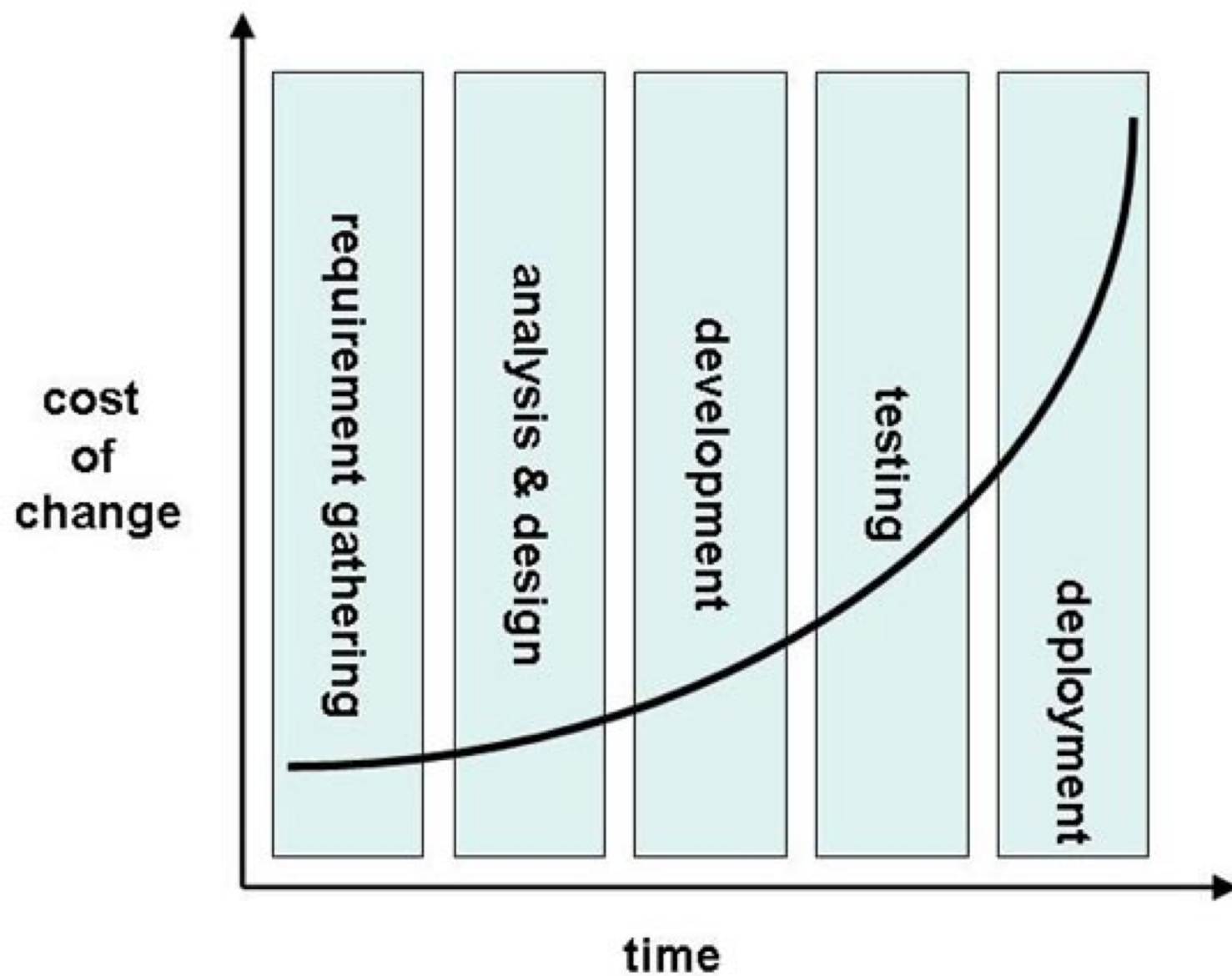




前期大规模设计

需求变化方式	复杂度	前置时间	返工	浪费
不变化	过度复杂	慢	无	前期的过度设计
按预期方式变化	合适	慢	无	无
没按预期方式变化	过度复杂	慢	重新设计	前期的过度设计





不要给我说什么
底层原理、框架内核！

老夫敲代码就是

一把 **梭**！

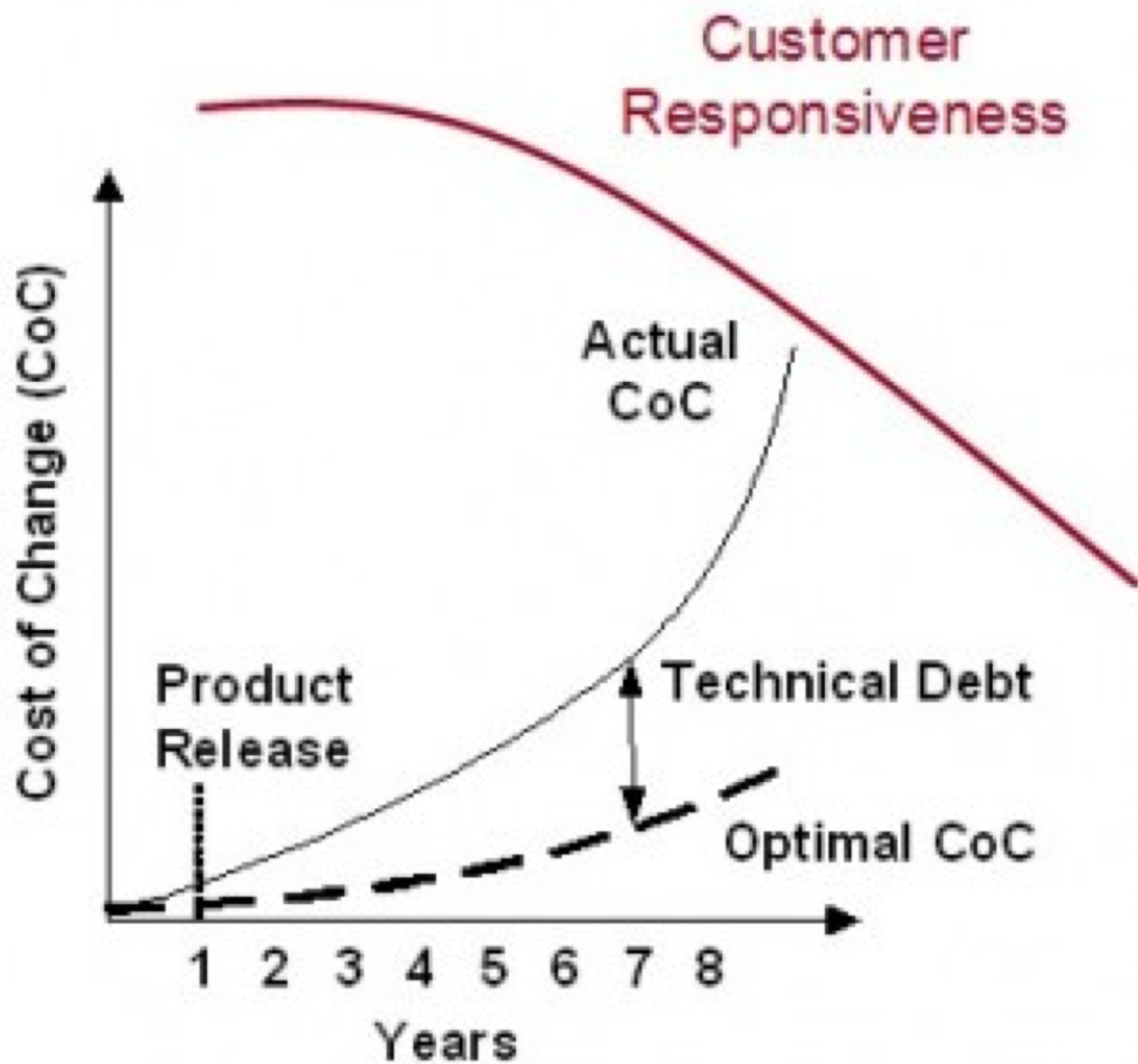
复制！

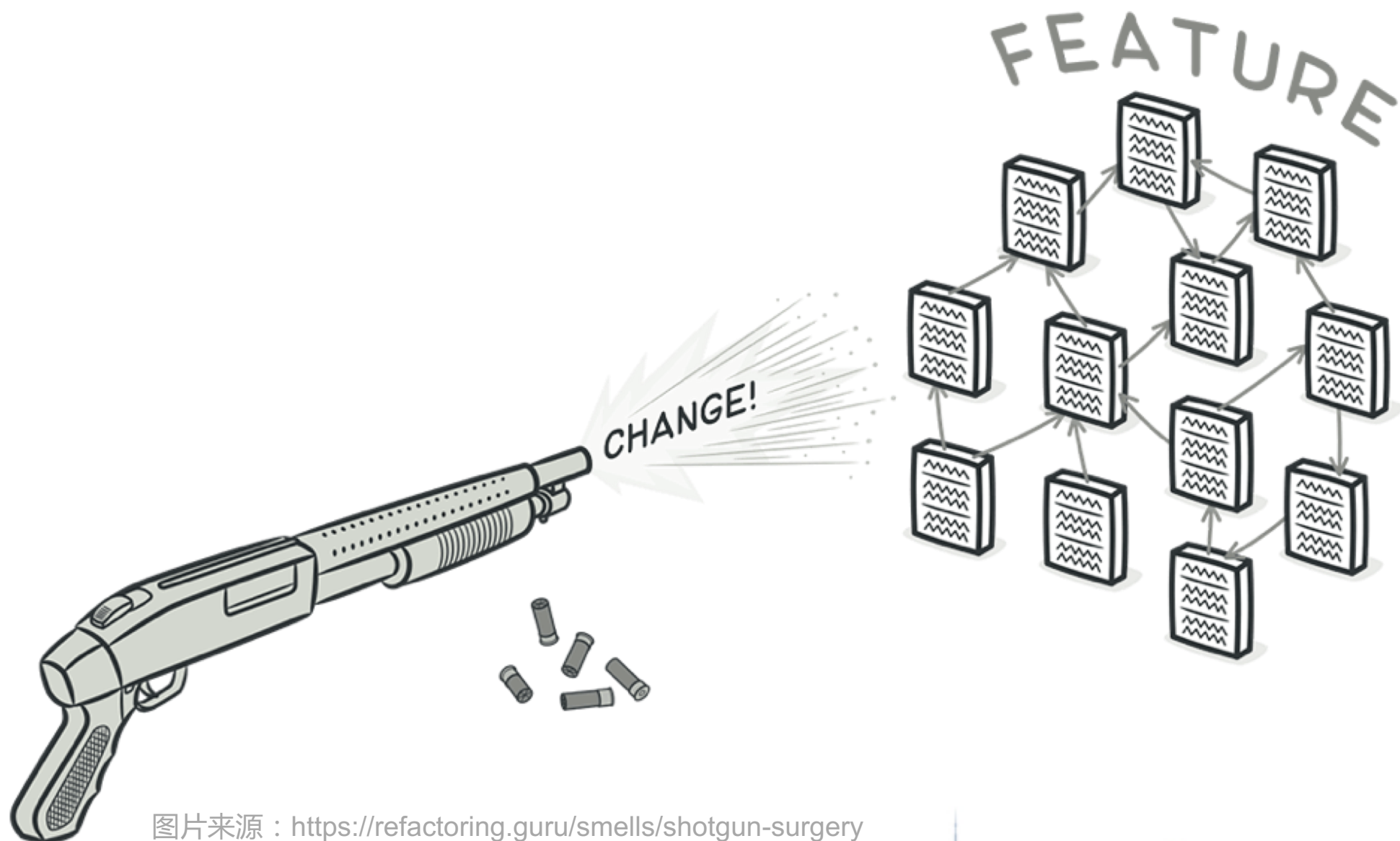
粘贴！

拿起键盘就是

干！







图片来源：<https://refactoring.guru/smells/shotgun-surgery>



匿名发布: [REDACTED]

1天内

有了这个杯垫，需求变得越来越清晰明了，pm的话也少了很多



❤️ 138

↻ 38

💬 32



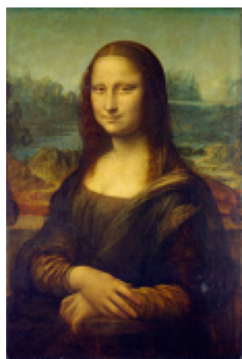
价值



柔性



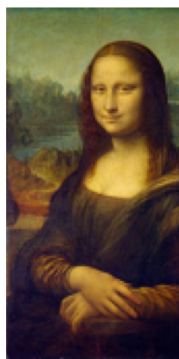
价值



柔性



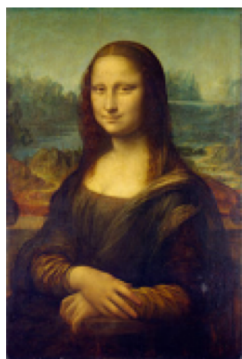
价值



柔性



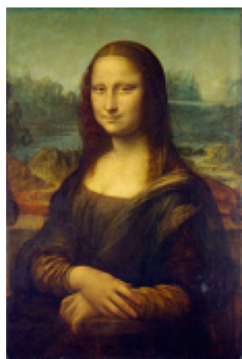
价值



柔性



价值



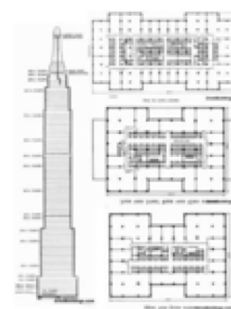
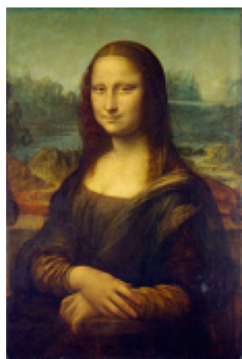
媒介



柔性



价值



柔性



价值



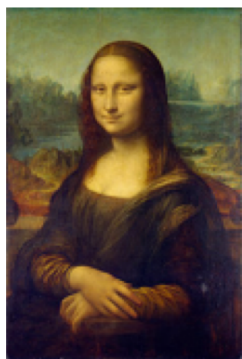
代码
代码



柔性



价值



代码



柔性



价值

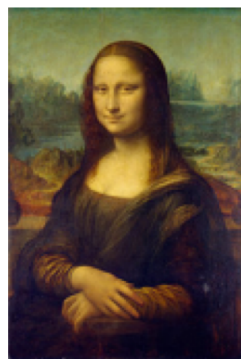


UML 设计文档 UML 设计文档

柔性



价值



代码

?

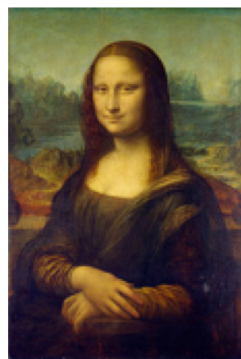


UML
设计文档

柔性



价值



代码

可重构
媒介



UML
设计文档

柔性



可执行
高性能
易部署

为机器
写的代码

可重构
媒介

设计文档

易理解、易修改
低耦合、反馈快



为机器写的代码

- 嗯，这段代码写不清楚，我加点注释
- 还是读不懂？没事，不是有文档吗？
- 命名？我已经尽力了：IJobProcessManager, JobProcessManagerImpl, process(), temp
- 方法？分分钟上百行
- 逻辑？if else for break 随便几层嵌套
- 嗯，响应CTO不养生的号召，加班到1点，发个朋友圈先



为人写的代码

```
def Option(expiry, strike, underlying, alternative):  
    Wait(expiry, Choice(underlying - strike, alternative))
```

```
def EuropeanOption(expiry, strike, underlying):  
    Option(expiry, strike, underlying, 0)
```

```
def EuropeanStockOption(expiry, strike, stock):  
    EuropeanOption(expiry, strike, IndexAtMaturity(stock))
```

```
def IndexAtMaturity(stock):  
    Settlement(ObservationDate(), ForwardMarket(ObservationDate(),  
stock))
```



此码非彼码

- 代码需要可读吗？
- 代码需要整洁吗？
- 代码需要静态扫描？
- 代码即设计/可执行模型？
- 浮现式设计？



如何让机器码更容易理解？

- 有意义的注释命名
- 提取方法、类
- 提高信噪比，代码瘦身
- 把一部分代码变成衍生制品



代码瘦身前

```
import java.util.Arrays;

public class DataExample {
    private final String name;
    private int age;
    private double score;
    private String[] tags;

    public DataExample(String name) {
        this.name = name;
    }

    public String getName() {
        return this.name;
    }

    void setAge(int age) {
        this.age = age;
    }

    public int getAge() {
        return this.age;
    }

    public void setScore(double score) {
        this.score = score;
    }
}
```

```
public double getScore() {
    return this.score;
}

public String[] getTags() {
    return this.tags;
}

public void setTags(String[] tags) {
    this.tags = tags;
}

@Override public String toString() {
    return "DataExample(" + this.getName() + ", " +
        this.getAge() + ", " + this.getScore() + ", " +
        Arrays.deepToString(this.getTags()) + ")";
}

protected boolean canEqual(Object other) {
    return other instanceof DataExample;
}
```

```
@Override public boolean equals(Object o) {
    if (o == this) return true;
    if (!(o instanceof DataExample)) return false;
    DataExample other = (DataExample) o;
    if (!other.canEqual((Object) this)) return false;
    if (this.getName() == null ? other.getName() != null :
        !this.getName().equals(other.getName())) return false;
    if (this.getAge() != other.getAge()) return false;
    if (Double.compare(this.getScore(), other.getScore()) != 0)
        return false;
    if (!Arrays.deepEquals(this.getTags(), other.getTags())) return
        false;
    return true;
}

@Override public int hashCode() {
    final int PRIME = 59;
    int result = 1;
    final long temp1 = Double.doubleToLongBits(this.getScore());
    result = (result * PRIME) + (this.getName() == null ? 43 :
        this.getName().hashCode());
    result = (result * PRIME) + this.getAge();
    result = (result * PRIME) + (int)(temp1 ^ (temp1 >>> 32));
    result = (result * PRIME) +
        Arrays.deepHashCode(this.getTags());
    return result;
}
```



通过Lombok瘦身

```
import lombok.AccessLevel;  
import lombok.Setter;  
import lombok.Data;  
import lombok.ToString;
```

```
@Data public class DataExample {  
    private final String name;  
    @Setter(AccessLevel.PACKAGE) private int age;  
    private double score;  
    private String[] tags;  
}
```



为什么使用Lombok ?

@Data = @ToString +
@EqualsAndHashCode +
@Getter +
@Setter +
@RequiredArgsConstructor



代码瘦身前

```
List<Transaction> groceryTransactions = new ArrayList<>();
for(Transaction t: transactions) {
    if(t.getType() == Transaction.GROCERY){
        groceryTransactions.add(t);
    }
}
Collections.sort(groceryTransactions, new Comparator(){
    public int compare(Transaction t1, Transaction t2){
        return t2.getValue().compareTo(t1.getValue());
    }
});
List<Integer> transactionIds = new ArrayList<>();
for(Transaction t: groceryTransactions){
    transactionIds.add(t.getId());
}
```



通过Stream瘦身

```
List<Integer> transactionsIds = transactions.parallelStream().  
    filter(t -> t.getType() == Transaction.GROCERY).  
    sorted(comparing(Transaction::getValue).reversed()).  
    map(Transaction::getId).  
    collect(toList());
```



数据访问层作为代码

```
public interface BookDaoInterface<T, Id  
extends Serializable> {
```

```
    public void persist(T entity);
```

```
    public void update(T entity);
```

```
    public T findById(Id id);
```

```
    public void delete(T entity);
```

```
    public List<T> findAll();
```

```
    public void deleteAll();
```

```
}
```

```
public class BookDao implements BookDaoInterface<Book, String> {
```

```
    public void persist(Book entity) {  
        getSession().save(entity);  
    }
```

```
    public void update(Book entity) {  
        getSession().update(entity);  
    }
```

```
    public Book findById(String id) {  
        Book book = (Book) getSession().get(Book.class, id);  
        return book;  
    }
```

```
    public void delete(Book entity) {  
        getSession().delete(entity);  
    }
```

```
    @SuppressWarnings("unchecked")  
    public List<Book> findAll() {  
        List<Book> books = (List<Book>) getSession().createQuery("from Book").list();  
        return books;  
    }
```

```
    public void deleteAll() {  
        List<Book> entityList = findAll();  
        for (Book entity : entityList) {  
            delete(entity);  
        }  
    }
```

```
}
```



数据访问层作为衍生制品

```
import java.util.List;
import org.springframework.data.repository.CrudRepository;

public interface CustomerRepository extends CrudRepository<Customer,
Long> {
    List<Customer> findByLastName(String lastName);
}
```



成功案例某DevOps云平台

- 2016初立项，上线半年，基于Docker的DevOps云可以随时发布到 产环境。已接入持续交付项目28个、流水线186条、环境146个、 用户211人、构建16081次
- 2017年，集团层面组织40余个省级公司到该省公司参观和学习经验



成功案例某DevOps云平台

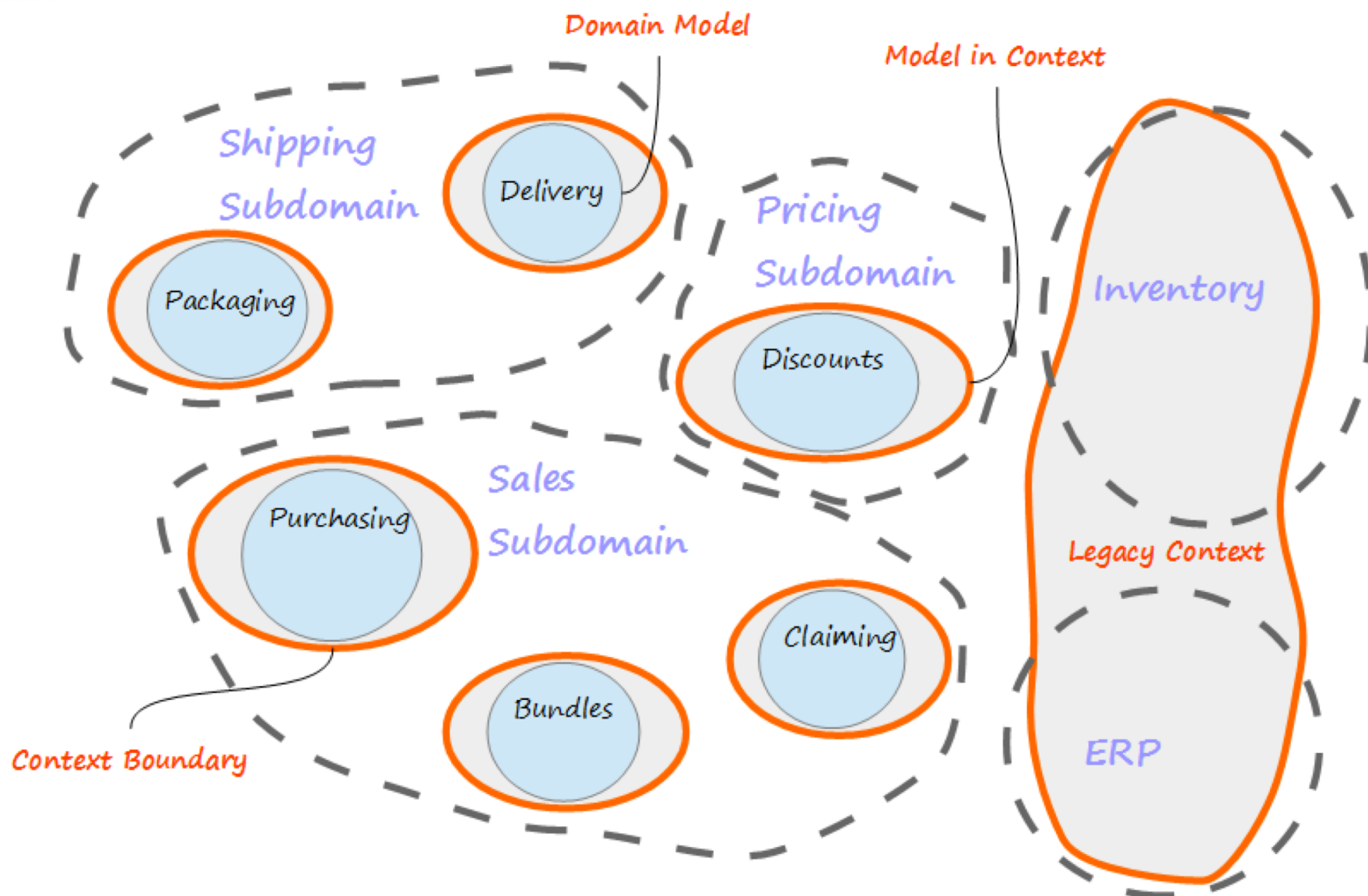
- 2016初立项，上线半年，基于Docker的DevOps云可以随时发布到 产环境。已接入持续交付项目28个、流水线186条、环境146个、 用户211人、构建16081次
- 2017年，集团层面组织40余个省级公司到该省公司参观和学习经验
- 后台代码仅3000行，不是ruby、不是python，是Java



如何给机器码降耦合？

- 用领域驱动设计，给系统划分子域和限界上下文
- 在子系统之间采用事件驱动进一步降低耦合





限界上下文(Bounded

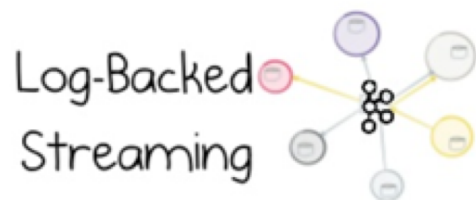
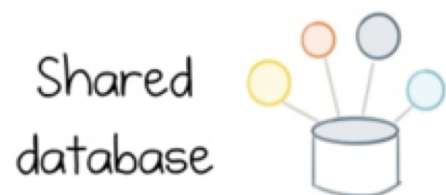
- 一个由显式边界限定的特定职责。领域模型便存在于这个边界之内。在边界内，每一个模型概念，包括它的属性和操作，都具有特殊的含义。
- 一个给定的业务领域会包含多个限界上下文，想与一个限界上下文沟通，则需要通过显式边界进行通信。系统通过确定的限界上下文来进行解耦，而每一个上下文内部紧密组织，职责明确，具有较高的内聚性。



耦合
(自治)

易访问

一致性

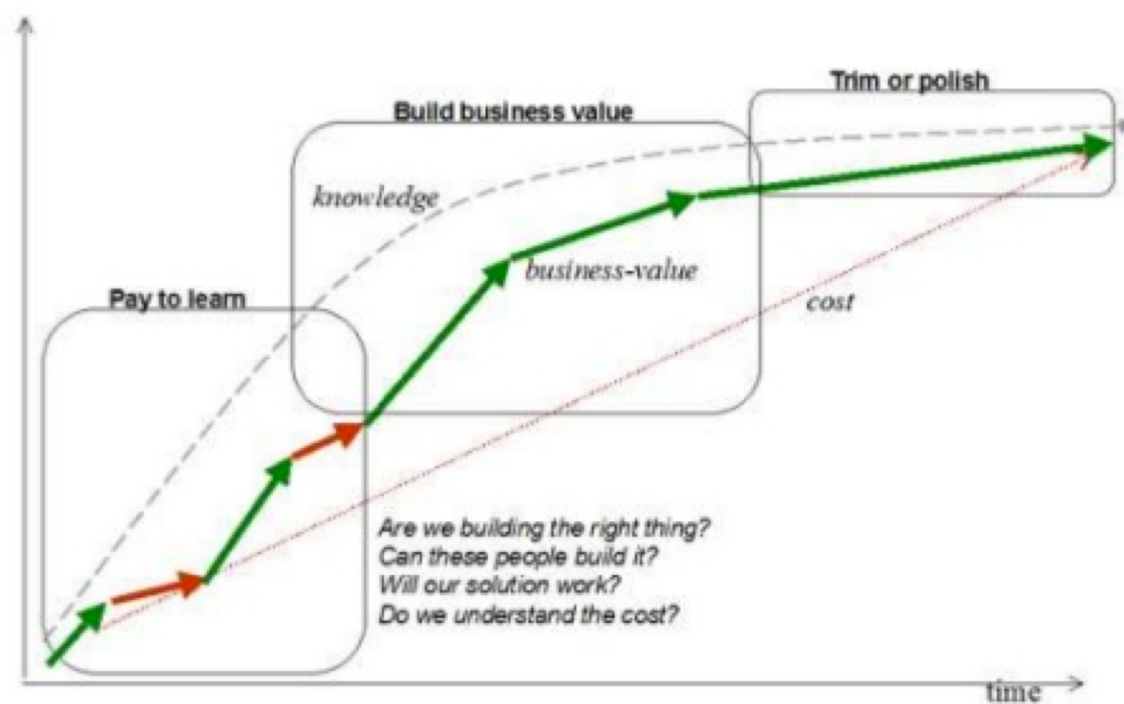


如何让机器码更容易修改？

- 通过构造行走骨架快速验证风险点
- 使衍生制品变得可重构
- 通过git effort贯彻开放关闭原则
- 掌握常用重构手法



Learn – Value - Tail



The Knowledge-Acquisition Curve

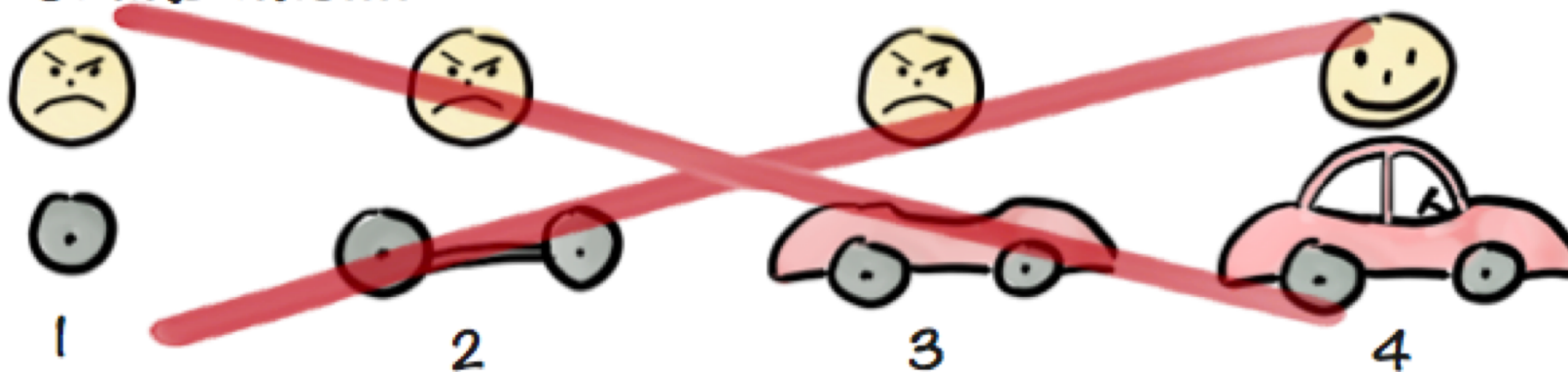
©Alistair Cockburn 2011



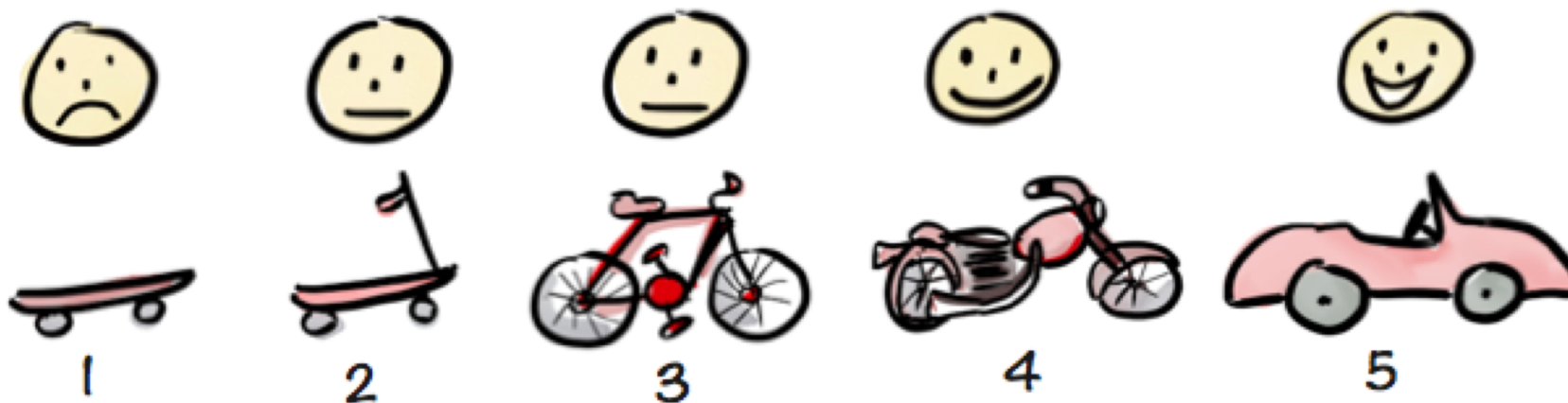
Knowledge Acquisition Curve - <http://alistair.cockburn.us/>



Not like this....



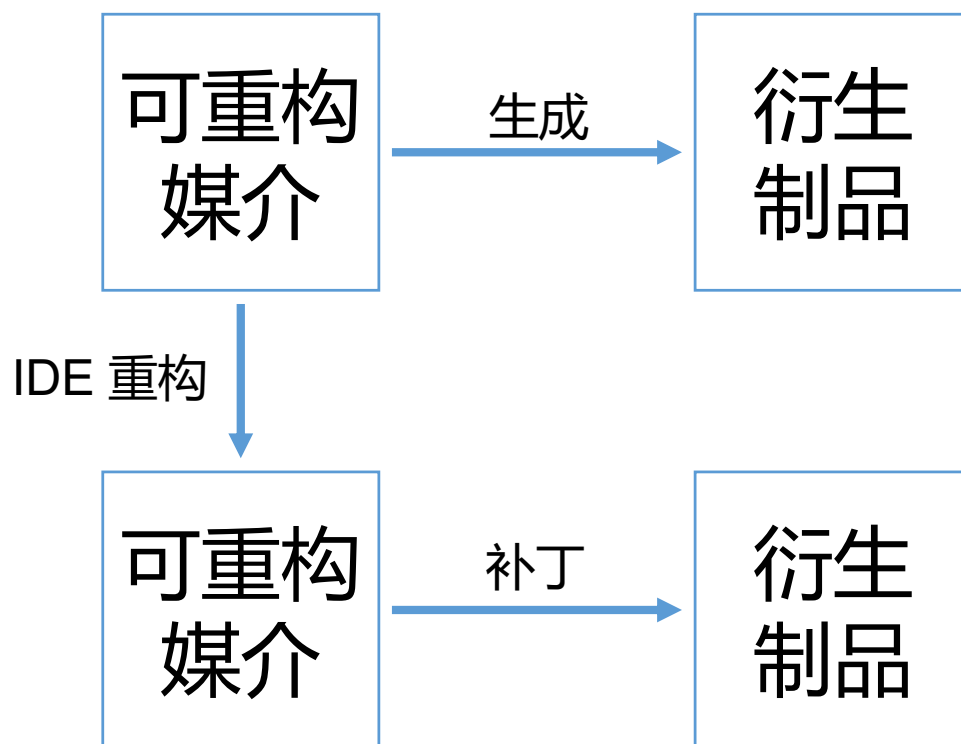
Like this!



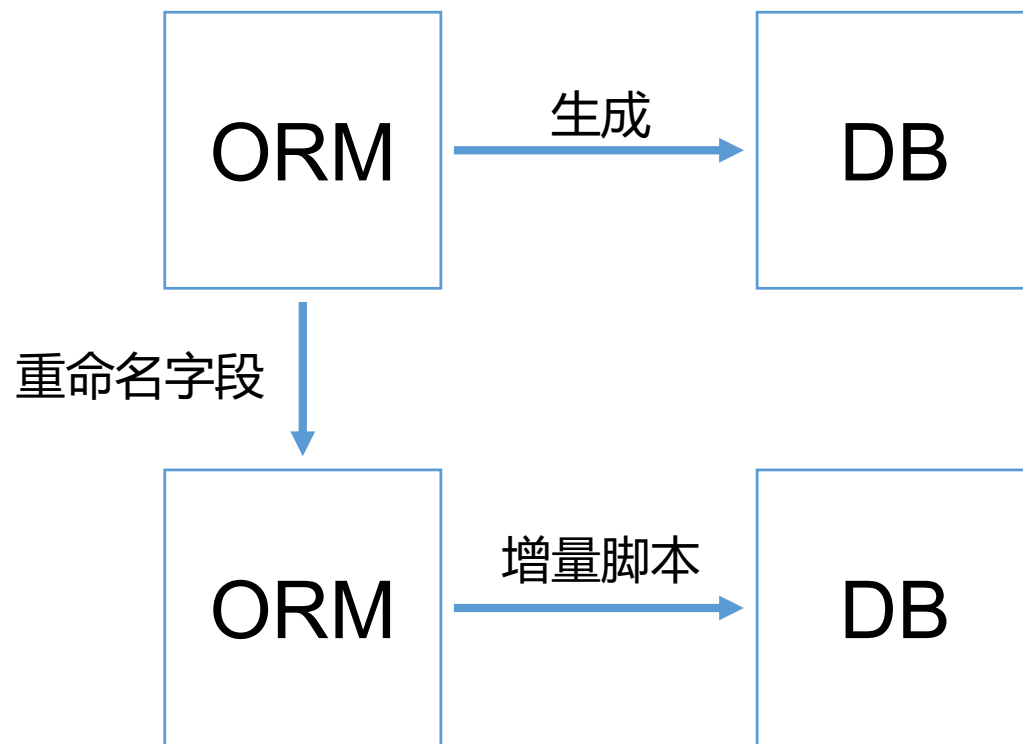
<http://blog.crisp.se/2016/01/25/henrikkniberg/making-sense-of-mvp>



使衍生制品变得可重构



使数据库变得可重构



通过git effort贯彻开放关闭原则

path	commits	active days
src/services.....	14	5
src/index.ts.....	9	5
src/services/slackHelperSpec.ts.....	8	4
src/services/slackHelper.ts.....	8	4
src/providers.....	7	2
src/apps/appSpec.ts.....	6	3
src/apps.....	6	3
src/providers/rdsProviderSpec.ts.....	4	2
src/models.....	4	3
src/apps/app.ts.....	4	3
src/providers/resourceType.ts.....	3	2
src/providers/rdsProvider.ts.....	3	2
src/providers/elasticCacheProvider.ts.....	3	2

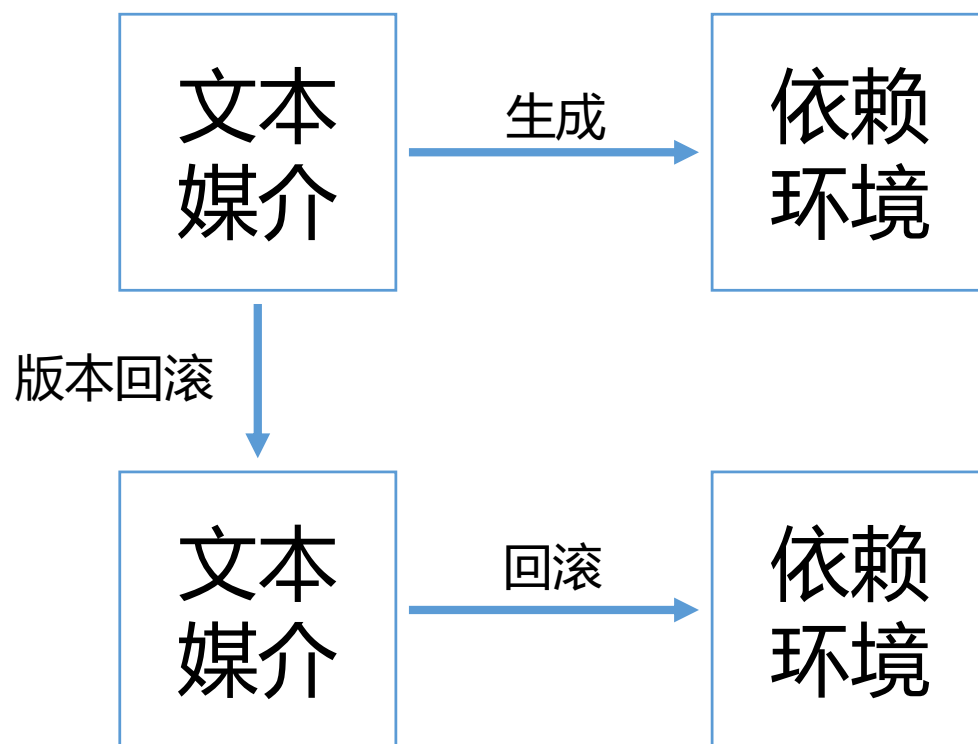


如何加快机器码反馈？

- 把环境/依赖变成可回滚媒介
- 反馈工程(feedback engineering)
 - 并行
 - 资源池
 - 即时反馈



把环境/依赖变成可回滚媒介



测试数据库作为衍生制品

```
@Before public void setUp() {  
    db = new EmbeddedDatabaseBuilder()  
        .setType(EmbeddedDatabaseType.H2)  
        .addScript("db/sql/create-db.sql")  
        .addScript("db/sql/insert-data.sql")  
        .build();  
}  
  
@Test public void testFindByName() {  
    NamedParameterJdbcTemplate template = new NamedParameterJdbcTemplate(db);  
    UserDao userDao = new UserDao();  
    userDao.setNamedParameterJdbcTemplate(template);  
  
    User user = userDao.findByName("da xian");  
  
    Assert.assertNotNull(user);  
}
```



手工管理Jenkins Job

Jenkins > Demo

 [Back to Dashboard](#)

 [Status](#)

 [Changes](#)

 [Workspace](#)

 [Build Now](#)

 [Delete Project](#)

 [Configure](#)

 [Copy project](#)

Build History [\(trend\)](#)

 [RSS for all](#)  [RSS for failures](#)

Project name

Description

[Preview](#)

- ☐ Discard Old Builds
- ☐ This build is parameterized
- ☐ Disable Build (No new builds will be executed until the project is re-enabled.)
- ☐ Execute concurrent builds if necessary



Advanced Project Options

[Advanced...](#)

Source Code Management

- ☐ CVS
- ☒ None
- ☐ Subversion
- ☐ Use SCM from another project

Build Triggers

- ☐ Build after other projects are built
- ☐ Build periodically
- ☐ - " - - - -



Job作为Jenkinsfile的衍生制

pipeline
agent any

```
stages {  
  stage('Build') {  
    steps { echo 'Building..' }  
  }  
  stage('Test') {  
    steps { echo 'Testing..' }  
  }  
  stage('Deploy') {  
    steps { echo 'Deploying....' }  
  }  
}
```



甚至更简单的travis.yml

```
#.travis.yml
```

```
language: java
```

```
jdk:
```

```
- oraclejdk8
```

```
script:
```

```
- mvn test
```



十万级回归测试并行案例

- C移植到Java
- DSL -> Java->Byte Code -> ClassLoader -> 执行



并行运行
40分钟



In Memory
Java
Compiler



-XDuseUnsharedTable



SQL DDL DML持续集成案例

增量

增量

基线

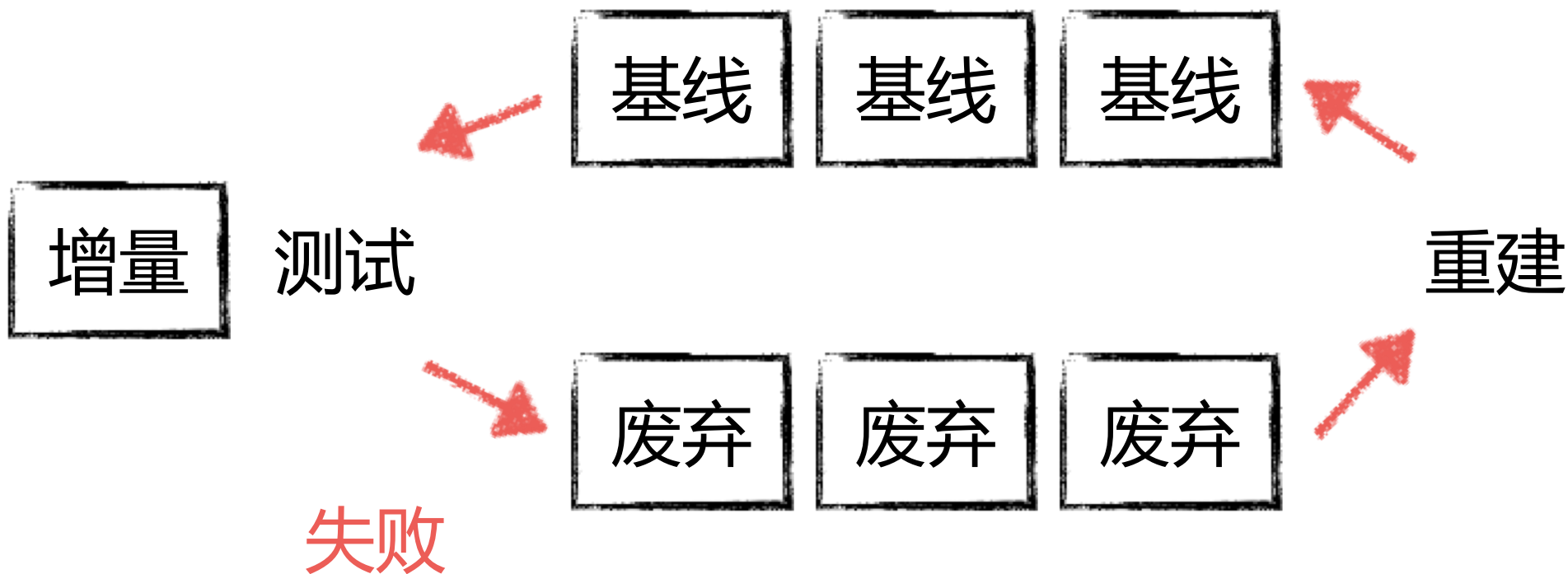


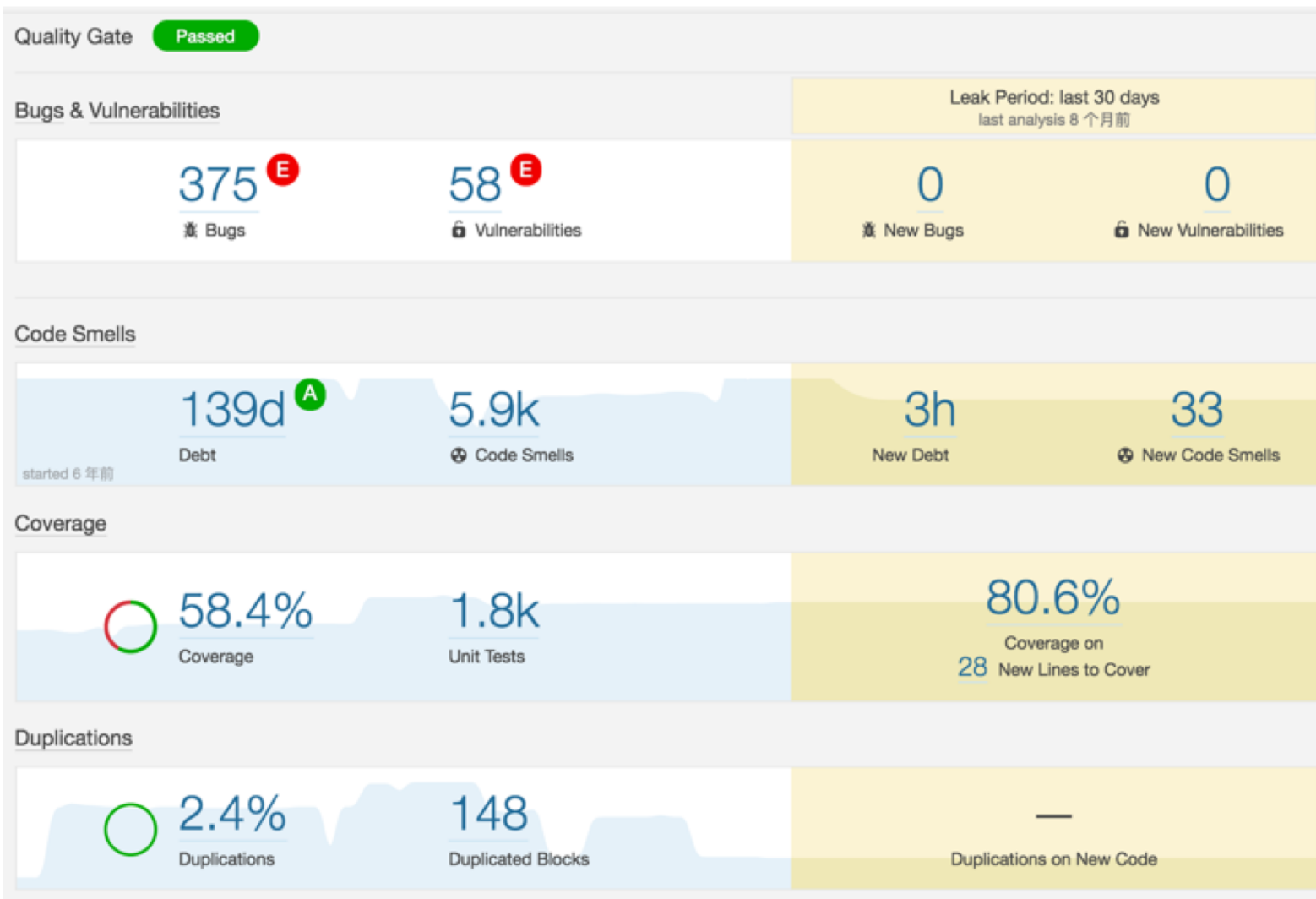
挑战

- 基线构建>30分钟
- 跨团队持续集成
- 增量频繁提交
- 无法回滚到基线



用数据库池来降低延迟





The screenshot shows the SonarLint IDE interface. On the left, a list of issues is displayed, categorized by file. The file 'AnnotatedSpecScanner.java' is selected, showing 7 issues. The issue 'Refactor this method to reduce its Cognitive Complexity' is highlighted. On the right, the detailed view of this issue is shown, including the rule name 'Cognitive Complexity of methods should not be too high', the rule ID 'squid:S3776', and a description of the rule. The bottom status bar shows 'SonarLint' and '6: TODO'.

可执行
高性能
易部署

为机器
写的代码

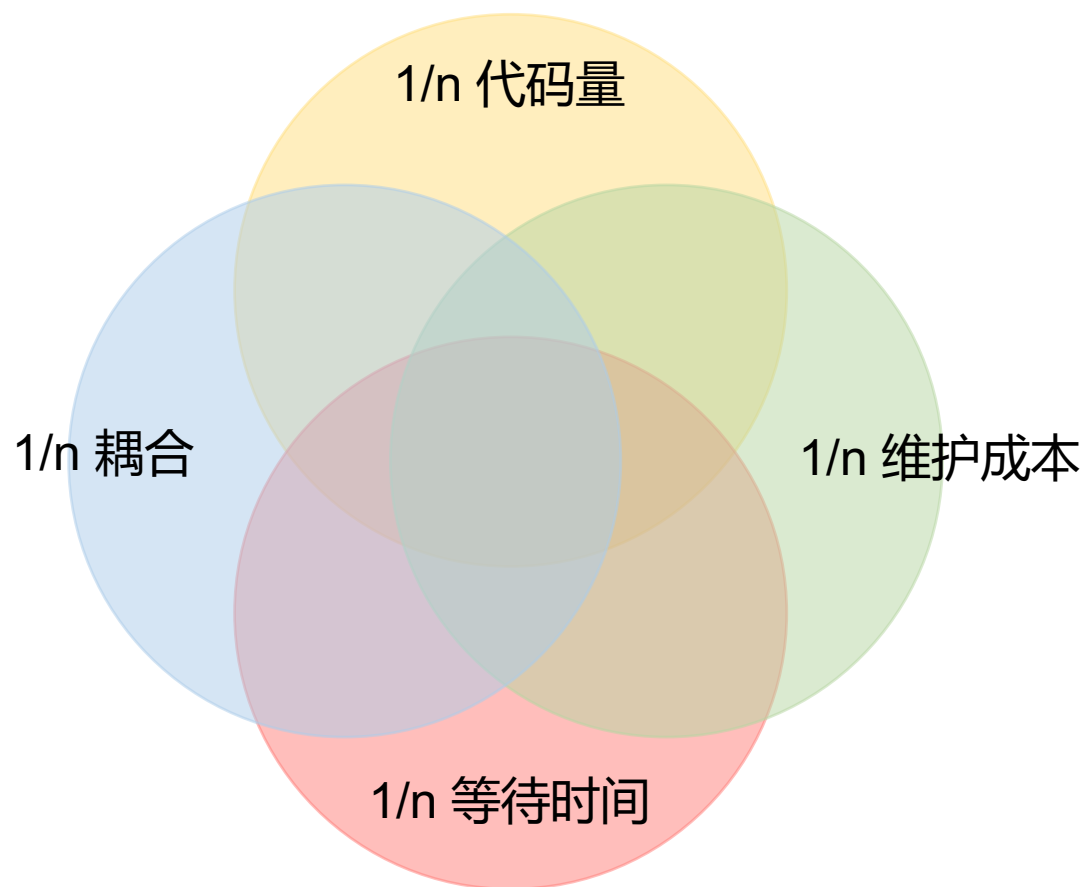
可重构
媒介

设计文档

易理解、易修改
低耦合、反馈快







简书



麦宇安
向你推荐星球

知识星球



星主：伟丹

ACT实战型敏捷教练圈



微信扫码预览星球详情



THANKS

Website :
chinadevopsdays.org/

Global Website:
www.devopsdays.org/events/2018-shanghai/

Official Email:
organizers-shanghai-2018@devopsdays.org



Official Wechat

