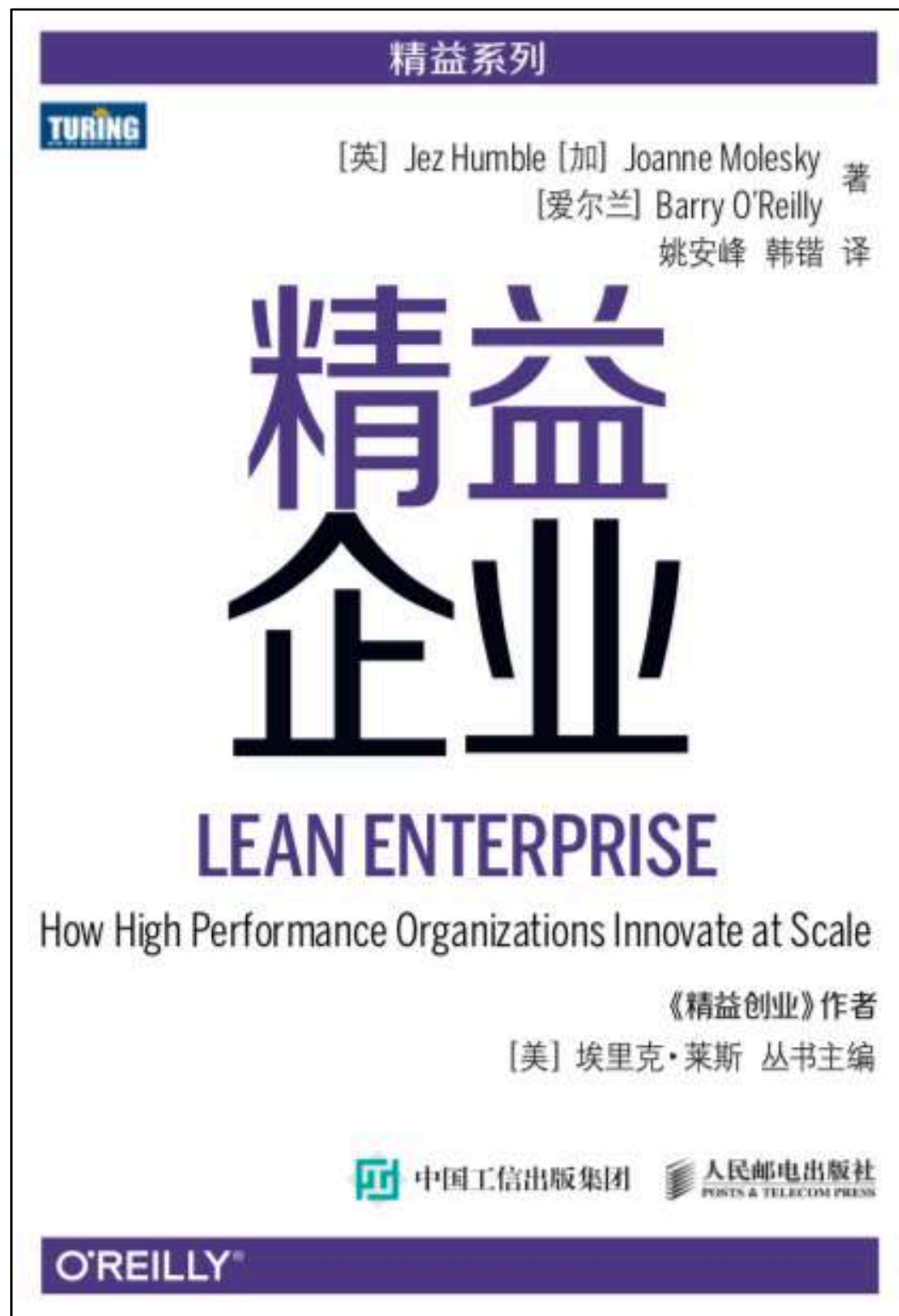




演进式架构的平台落地

ThoughtWorks 首席咨询师 姚安峰

关于讲师



ThoughtWorks首席咨询师。从事精益、敏捷等软件产品研发方法的践行与推广，对从业务探索、产品与服务设计、演进式架构、敏捷开发、持续交付、数据运营等端到端数字化业务的方法与实践及其规模化应用有深入研究和丰富实践经验。

目前致力于帮助各行业客户实现数字化转型，建立数字时代的精益研发体系，提升规模化创新能力。近两年来，研究的重点放在数字化业务管理的组织治理与动态投资组合管理，最大化投资成效。

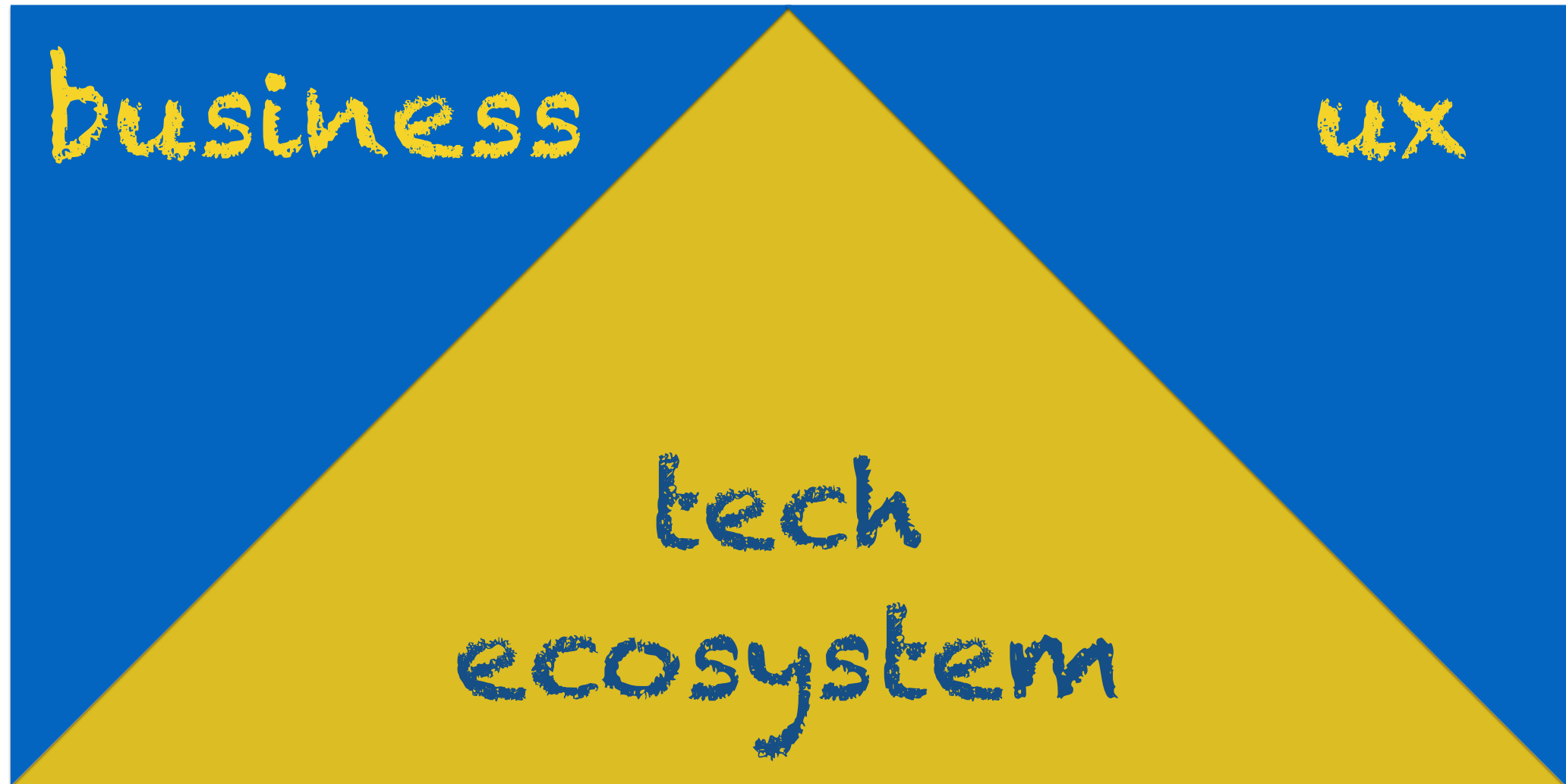
2016年翻译出版了著作《精益企业》。

演进式架构

让架构可以快速跟上业务发展与技术生态变化，并保持稳定

变化

—



everything
changes all
the time!



当事物在以不可预期的方式持续改变时，怎么可能进行有效的预先计划？

动态平衡

accessibility
accountability
accuracy
adaptability
administrability
affordability
agility
auditability
autonomy
availability
compatibility
composability
configurability
correctness
credibility
customizability
debugability
degradability
determinability
demonstrability
dependability
deployability
discoverability
distributability
durability
effectiveness
efficiency

reliability
extensibility
failure transparency
fault-tolerance
fidelity
flexibility
inspectability
installability
integrity
interchangeability
interoperability
learnability
maintainability
manageability
mobility
modifiability
modularity
operability
orthogonality
portability
precision
predictability
process capabilities
producibility
provability
recoverability
relevance

repeatability
reproducibility
resilience
responsiveness
reusability
robustness
safety
scalability
seamlessness
self-sustainability
serviceability
supportability
securability
simplicity
stability
standards compliance
survivability
sustainability
tailorability
testability
timeliness
traceability
transparency
ubiquity
understandability
upgradability
usability

evolvability

auditability



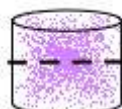
performance



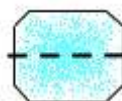
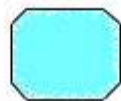
security



data



legality



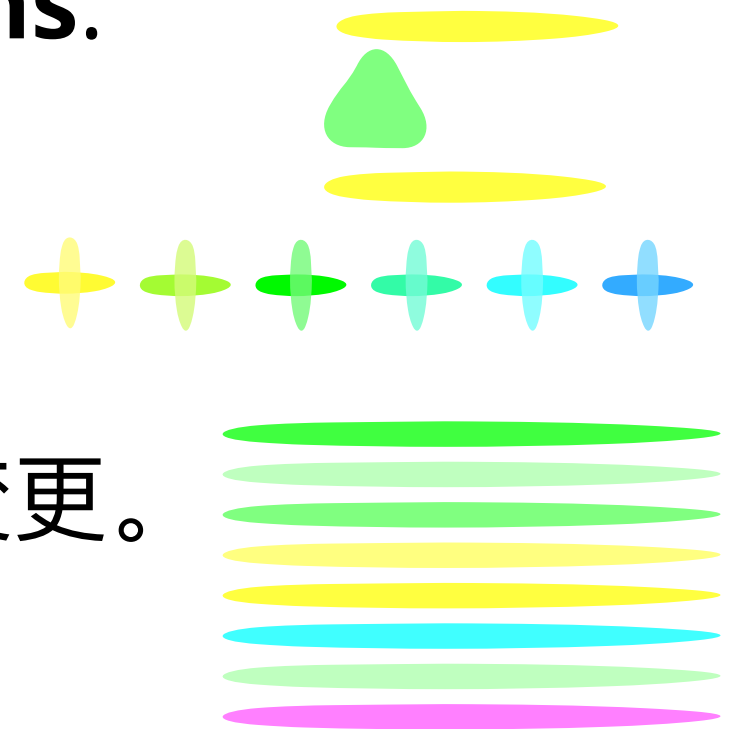
scalability



演进式架构

An evolutionary architecture supports
guided, incremental change
across **multiple dimensions**.

演进式架构支持
在各个架构设计维度上，
沿着特定方向进行频繁增量式变更。





guided



沿着特定方向 (guided) 的演进

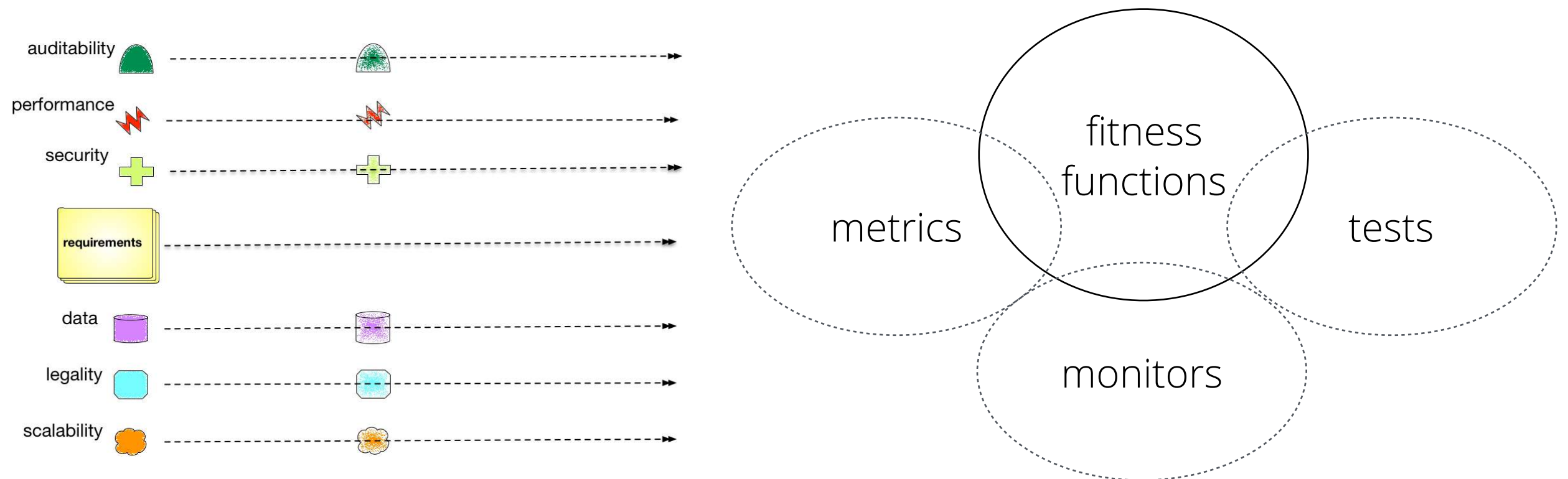


guided

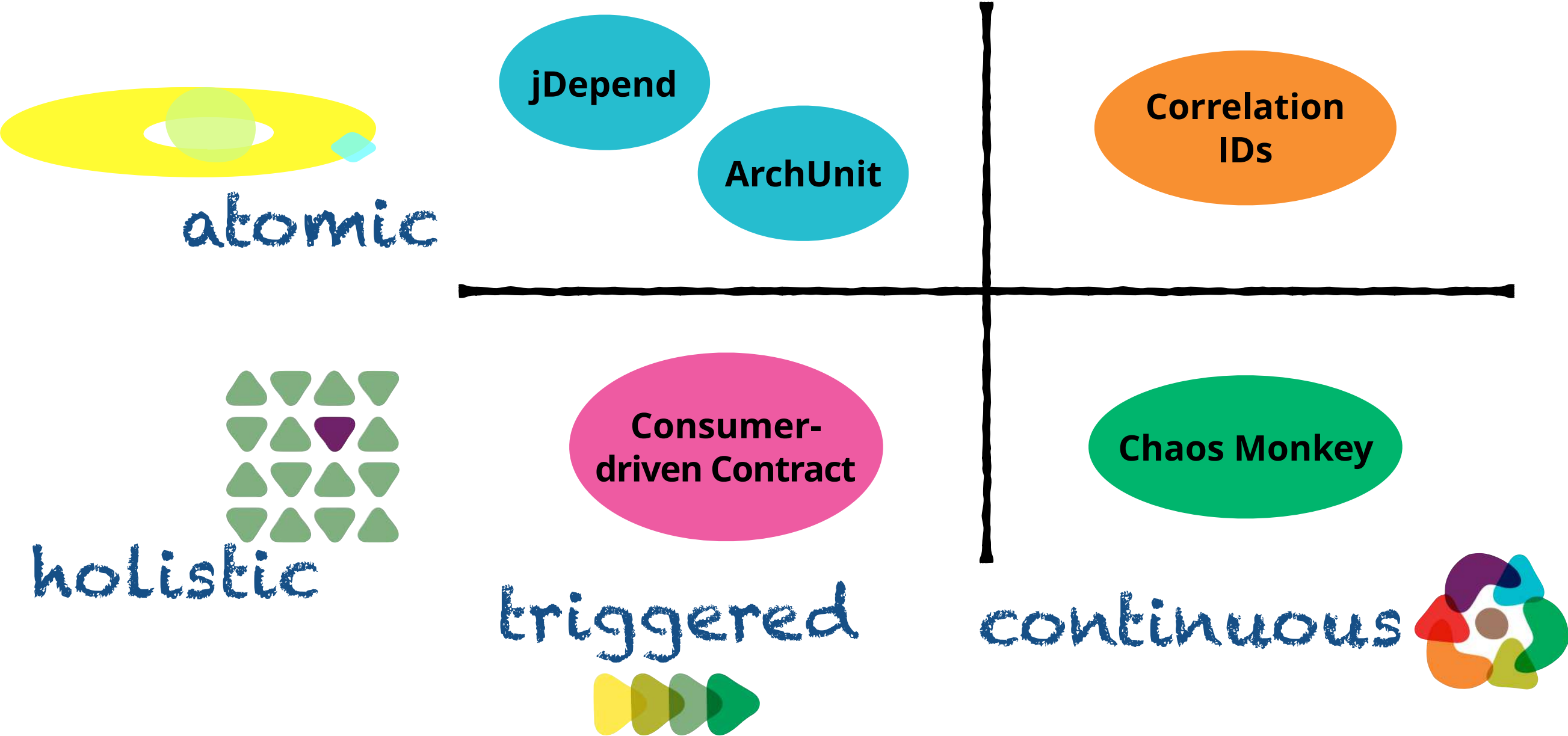
架构“适应性函数

(Architectural Fitness Functions) ”

对一些架构特征提供客观的一致性评估。

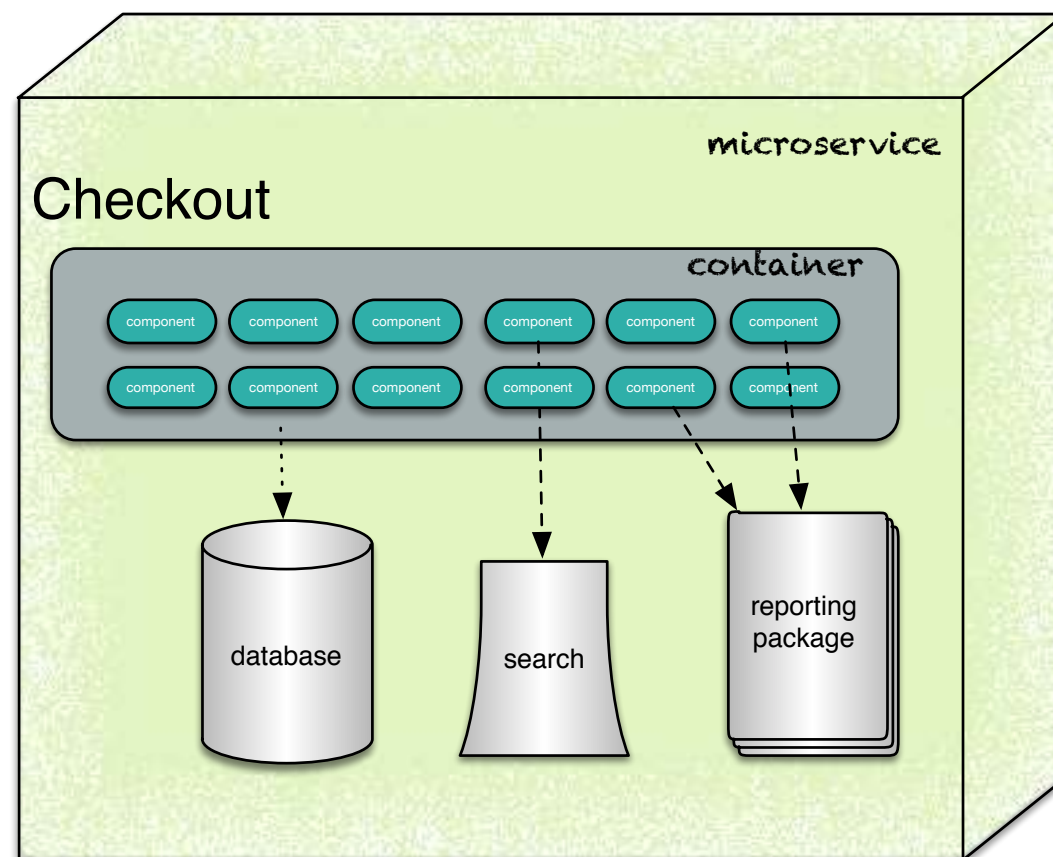


适应性函数分类

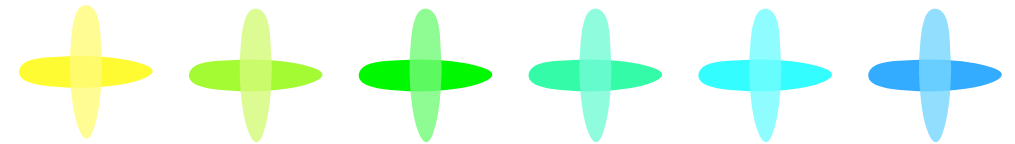
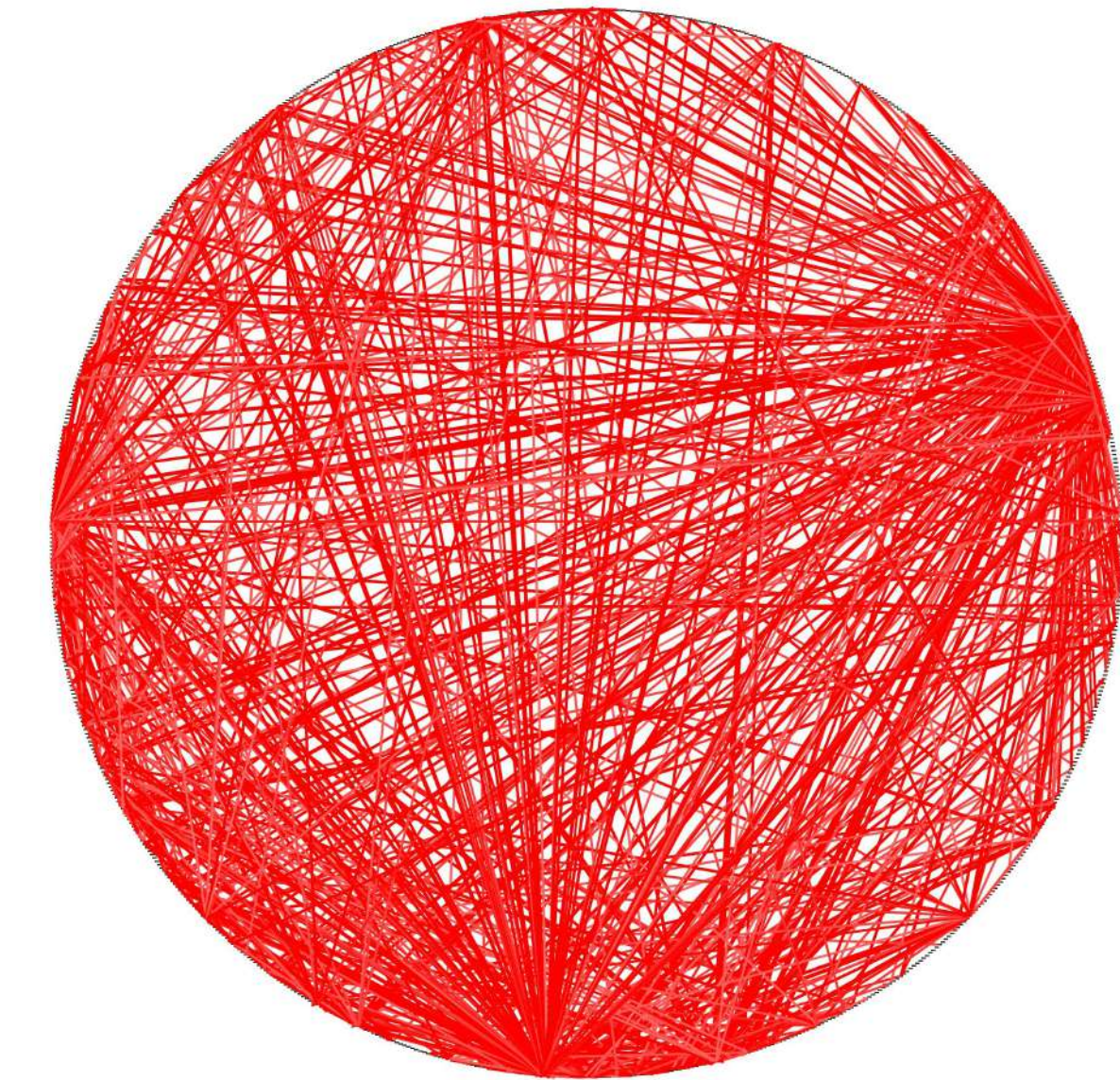


incremental change

“架构量子（architectural quantum）”是一个可独立变化并部署的单元组件，其具有高功能内聚性，包含一个系统正常工作所需的所有结构要素。



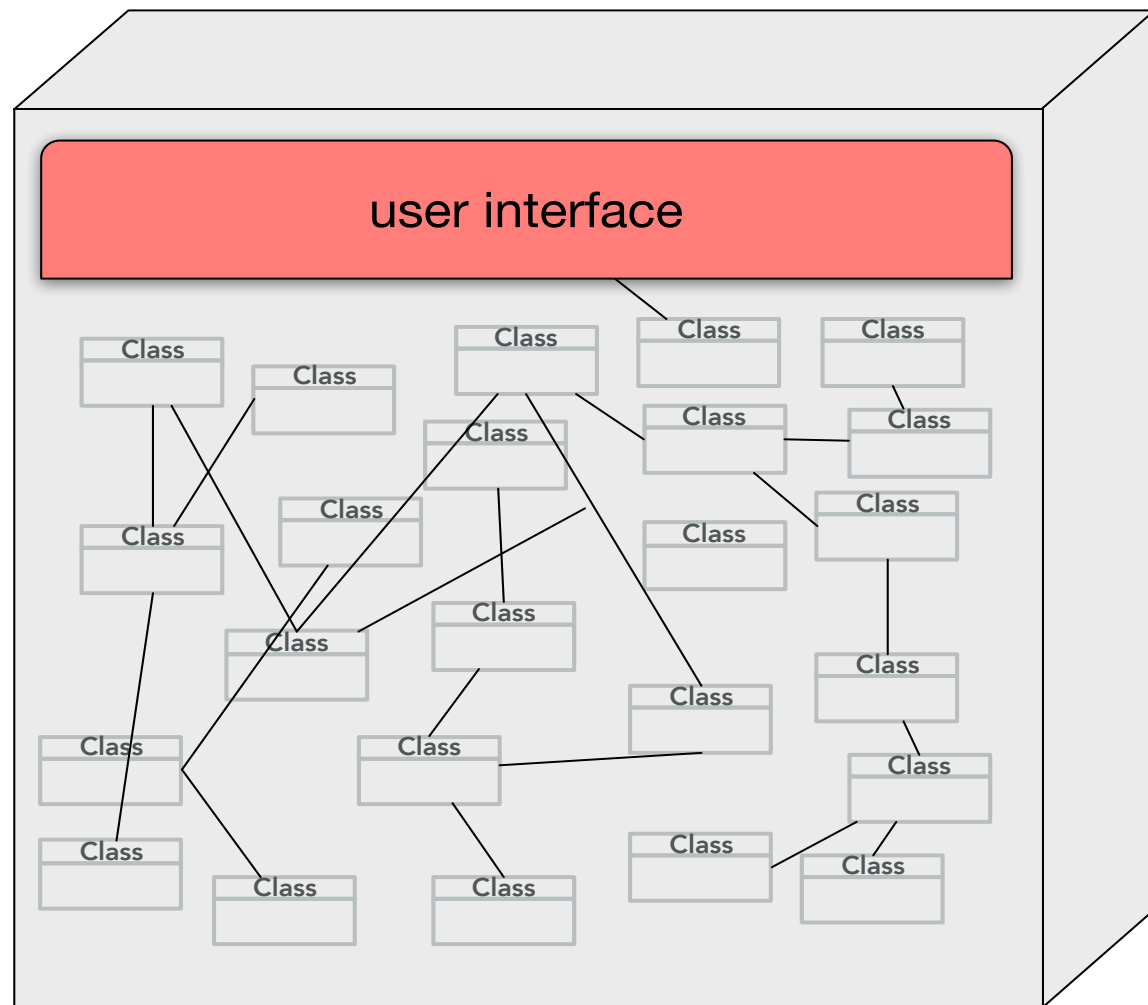
大泥球



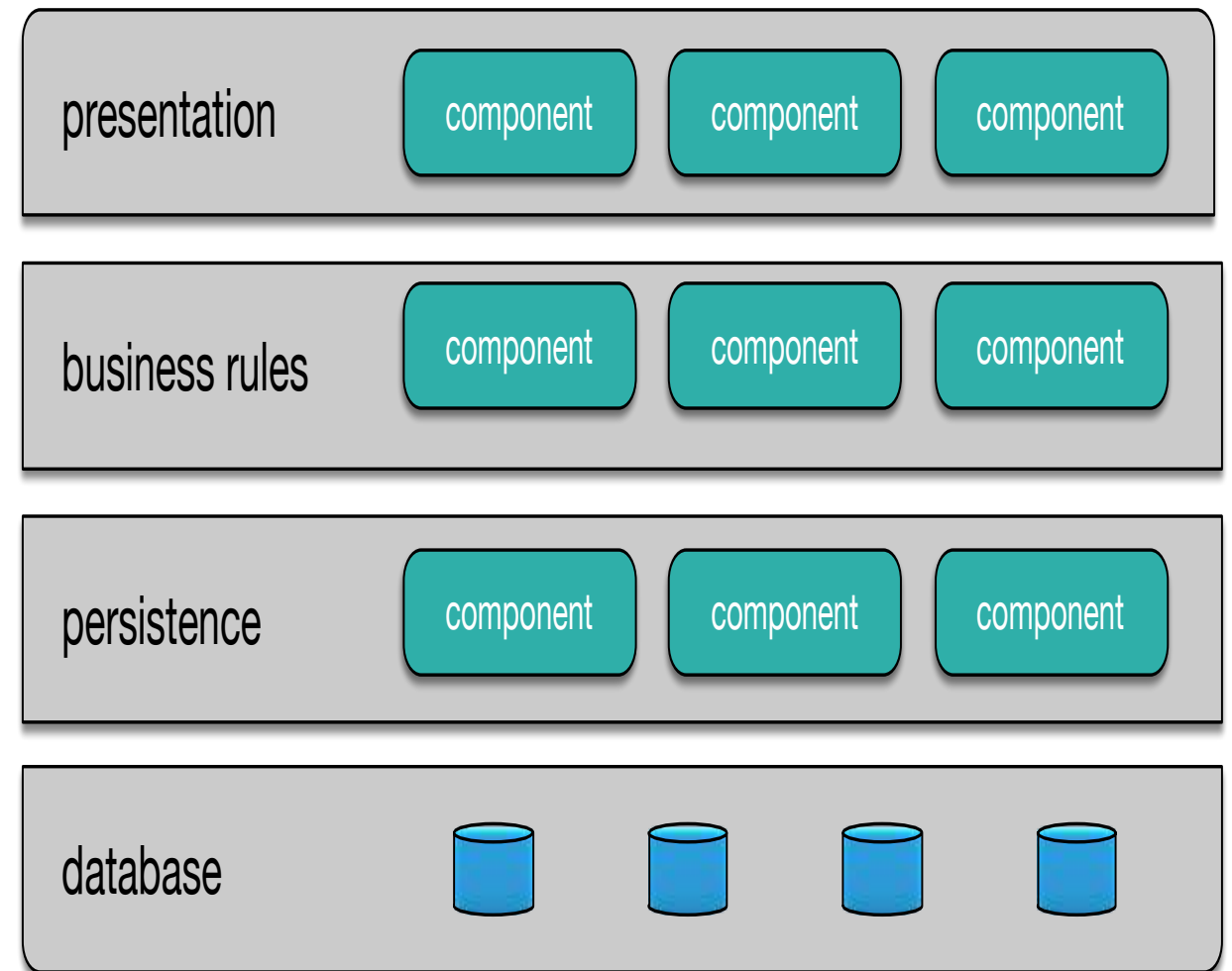
大量子尺寸的应用阻碍频繁增量变更，因为高耦合性需要一次部署大量代码大量的应用。

单例架构

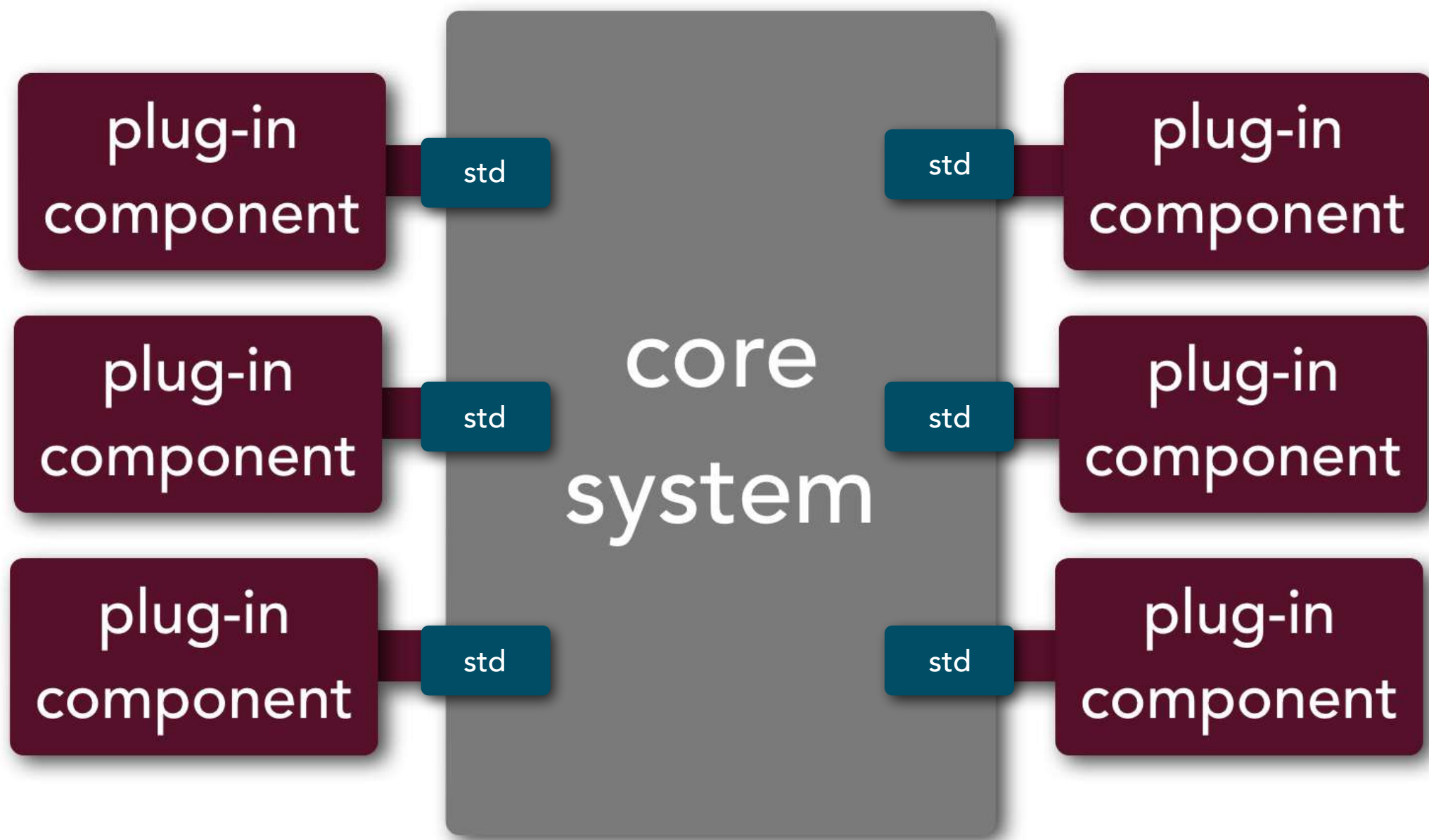
非结构的单例架构



分层单例架构

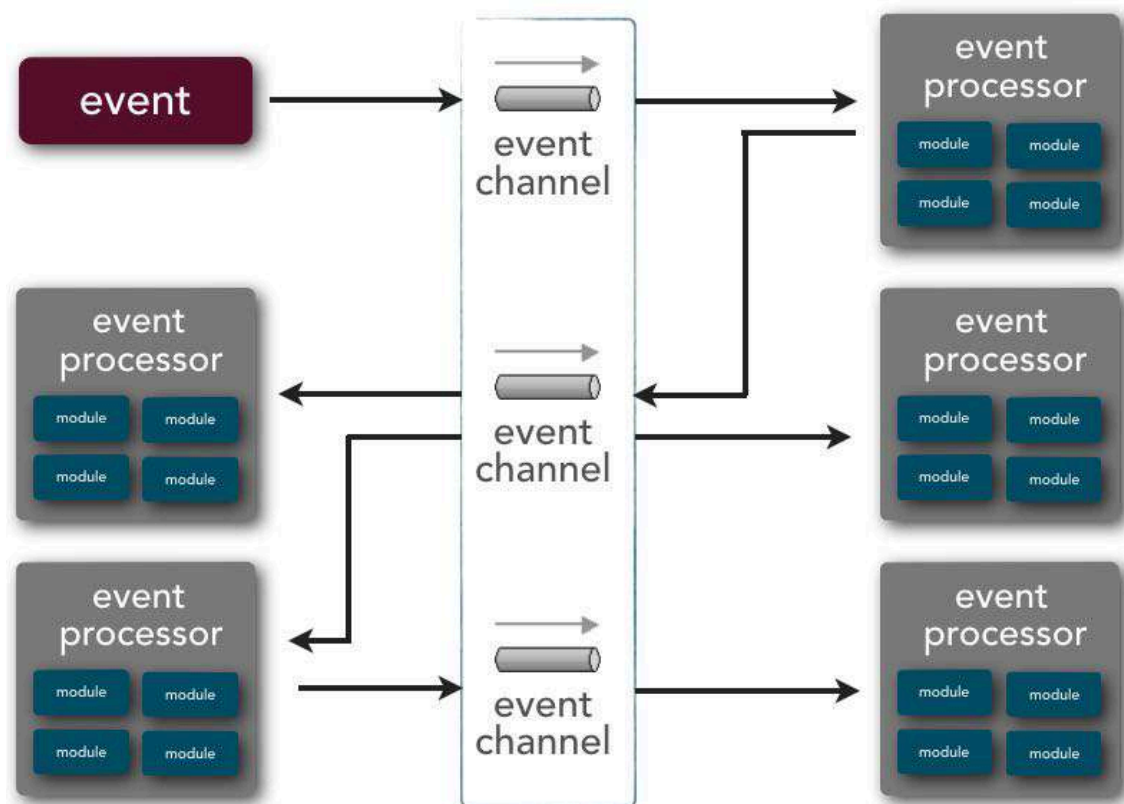


微内核与插件架构

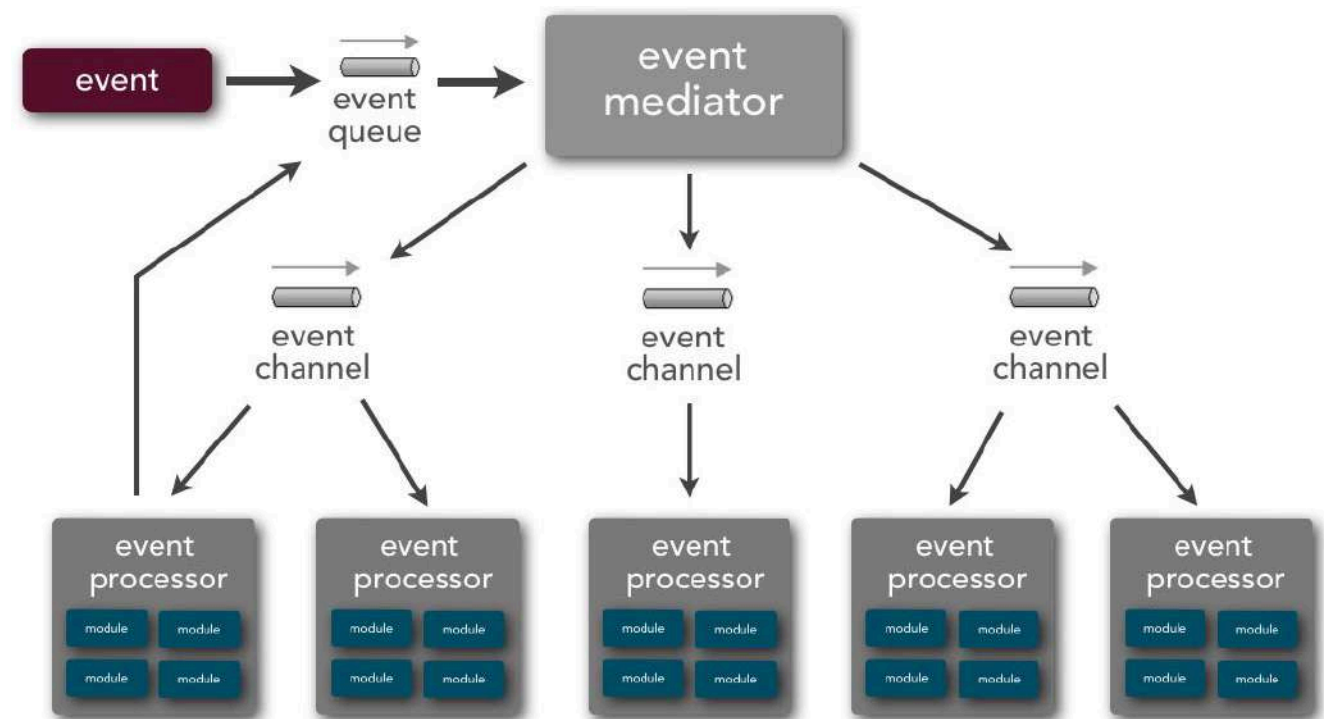


事件驱动架构

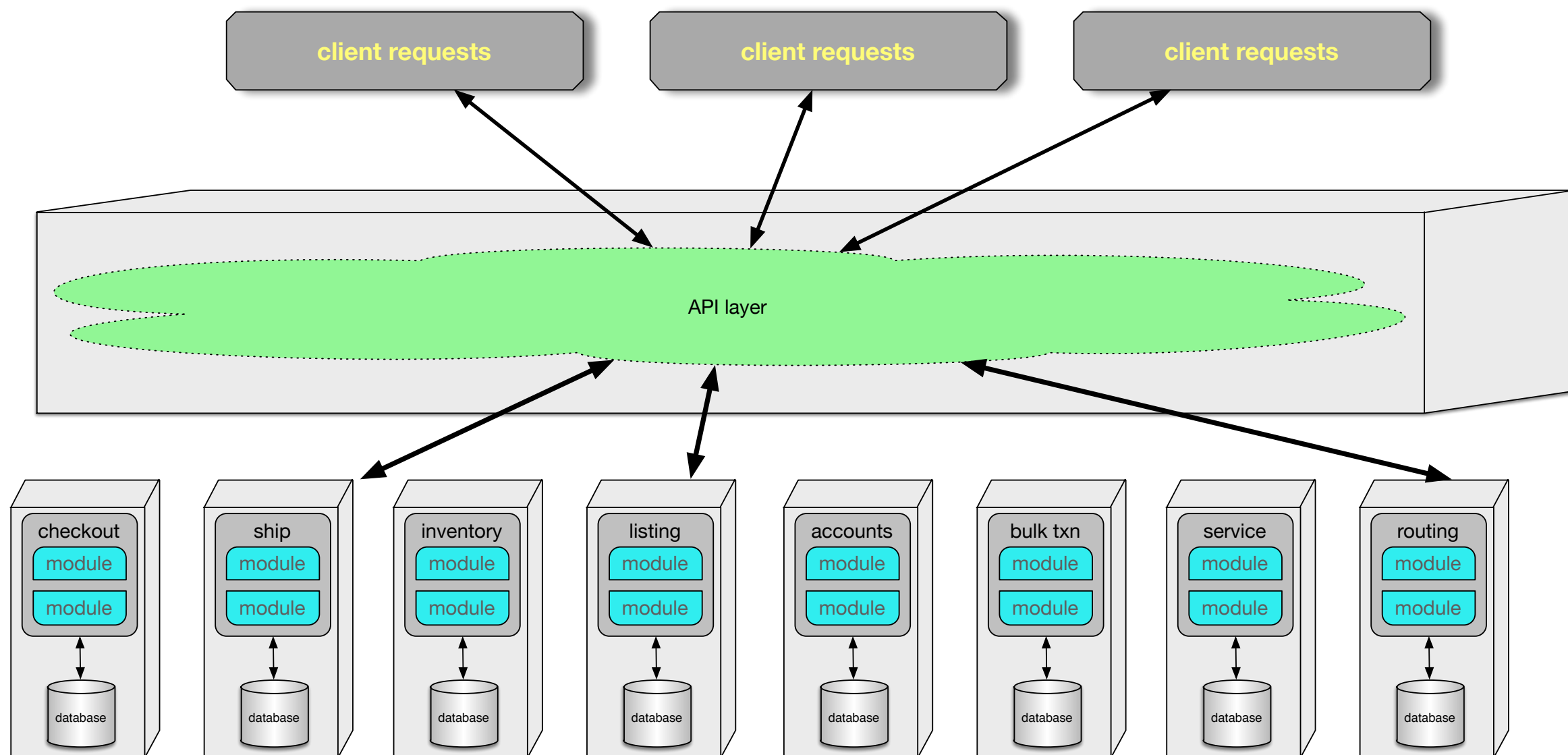
Broker



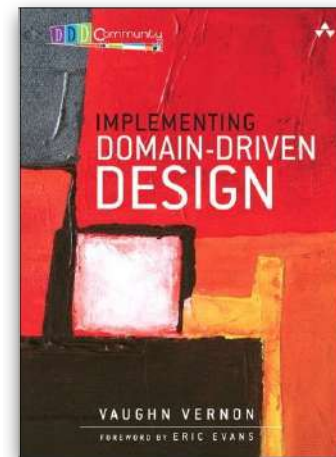
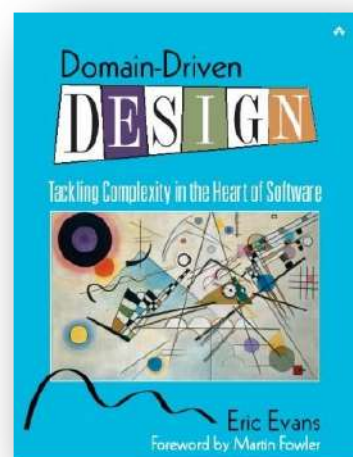
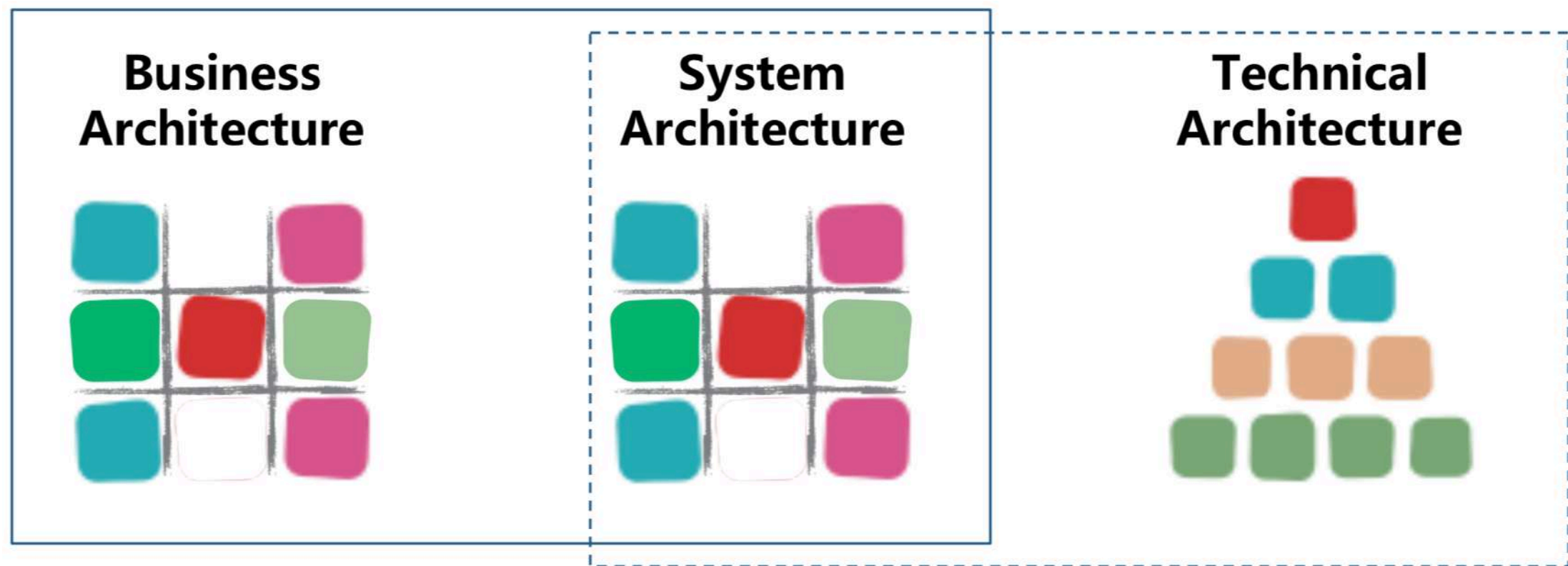
Mediator



微服务架构

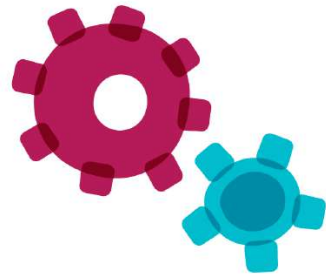


向以“领域”为中心的架构迁移



演进数据库 (**like code**)

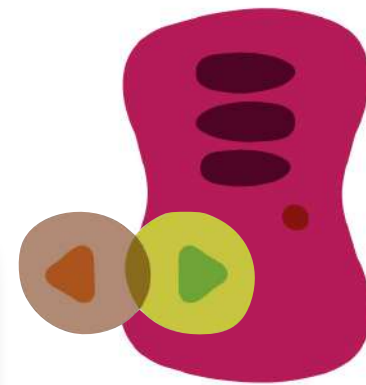
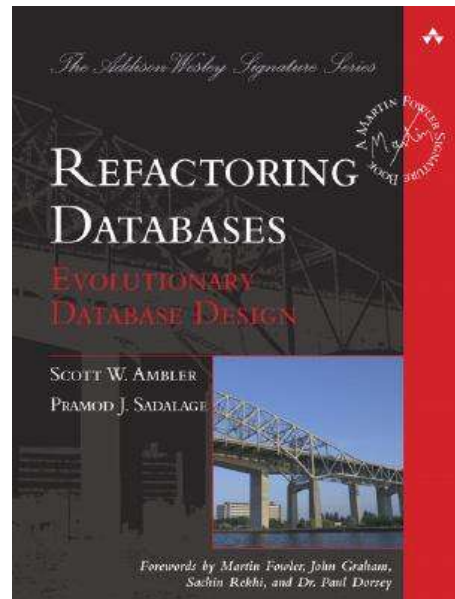
— Tested



scripting all db changes
incrementally

— Versioned

— Incremental

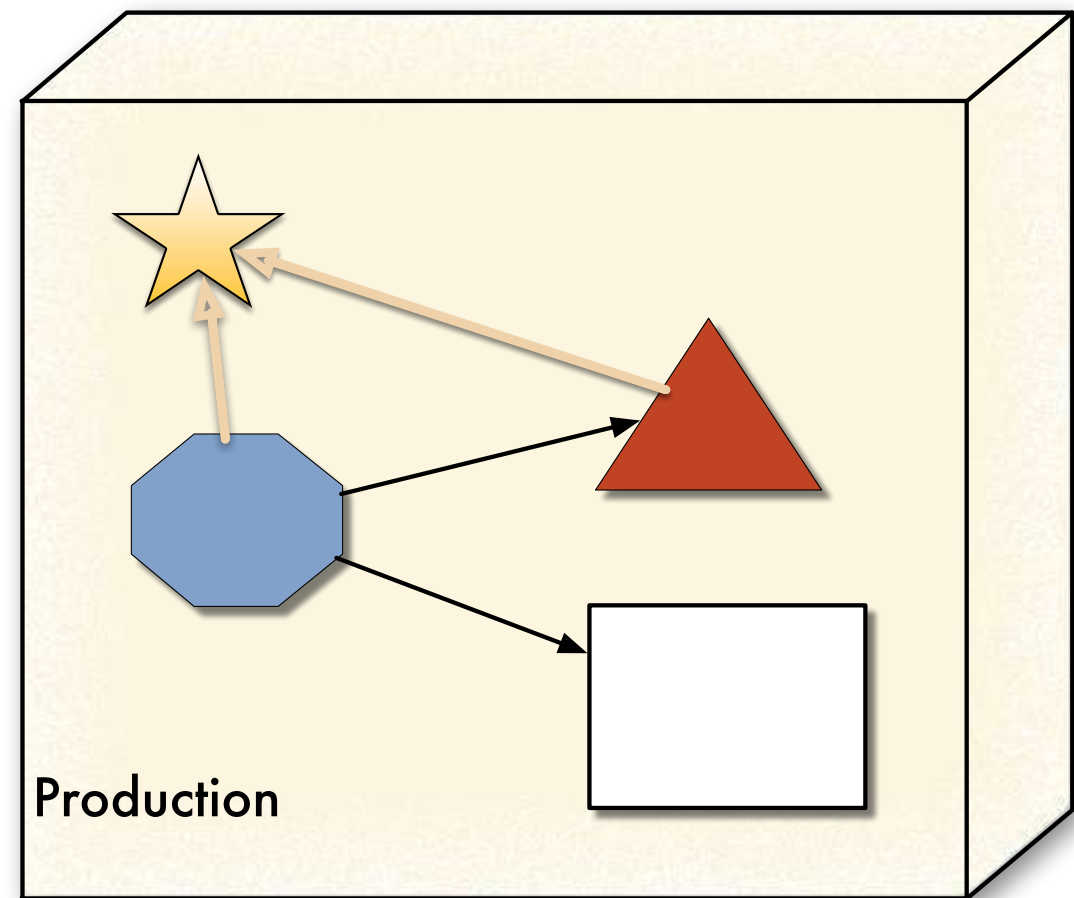


db
refactoring

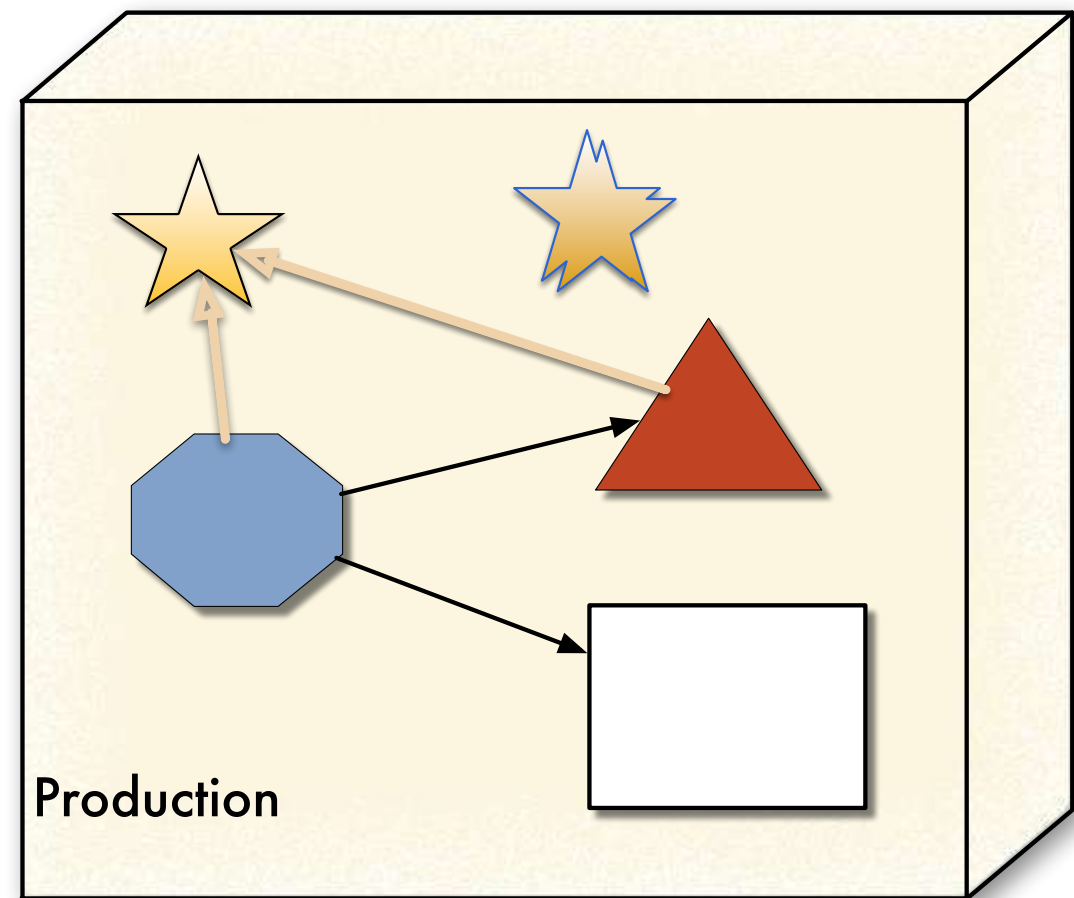


**decouple db
migration
from app
migration**

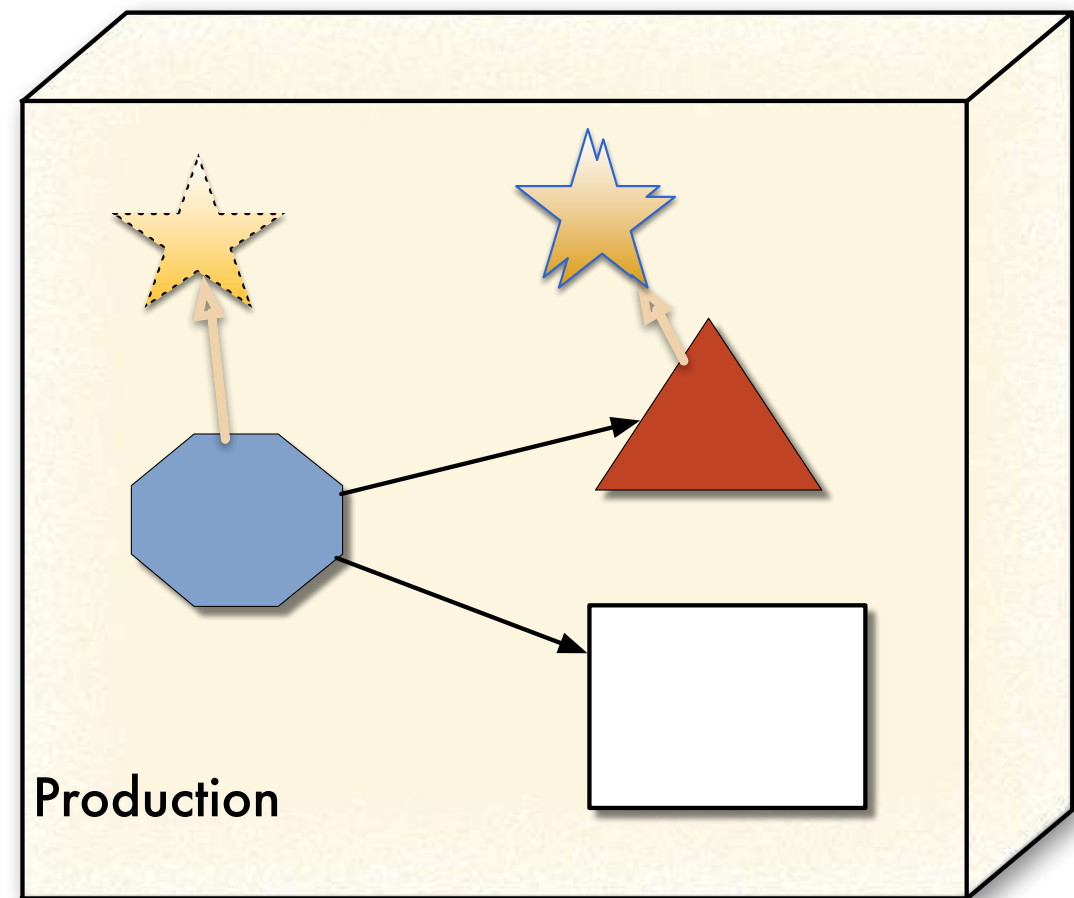
演进接口 (**forward compatible**)



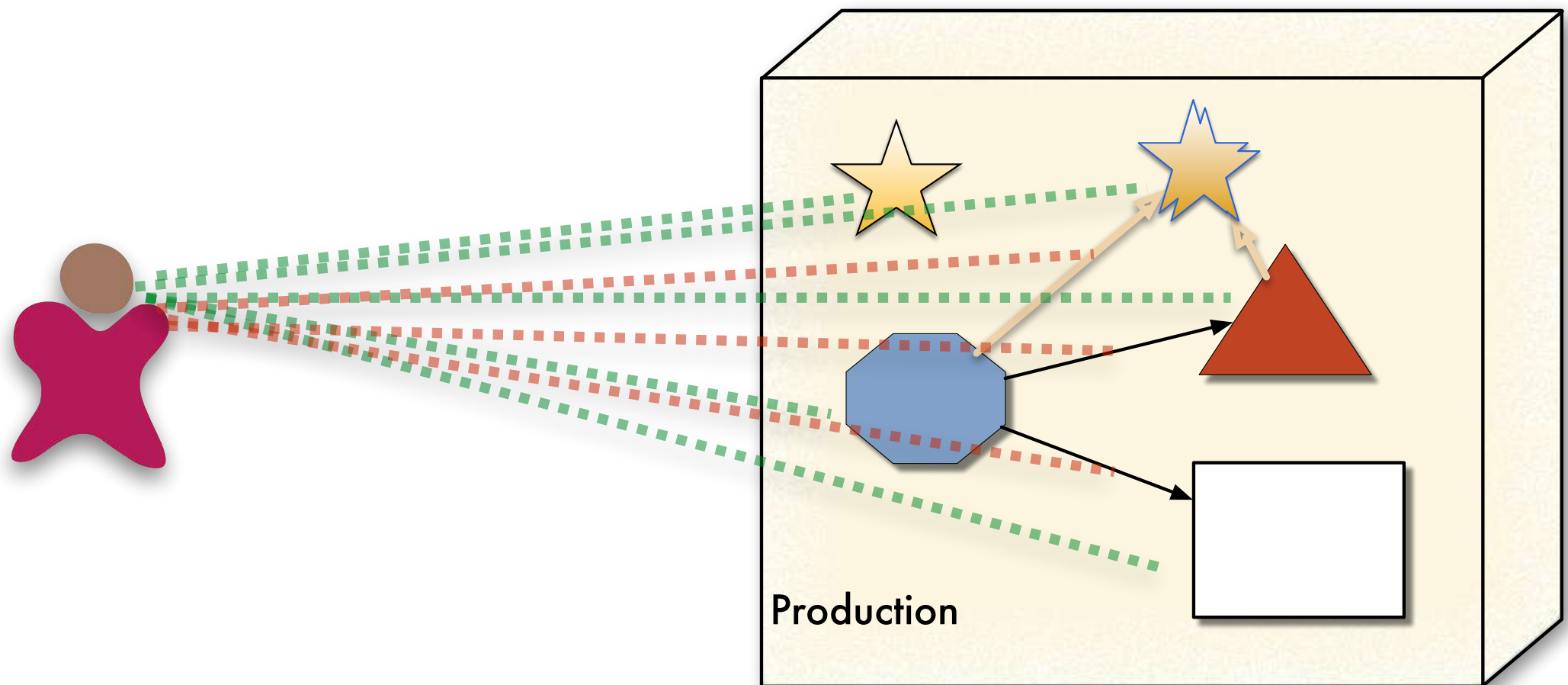
演进接口 (**forward compatible**)



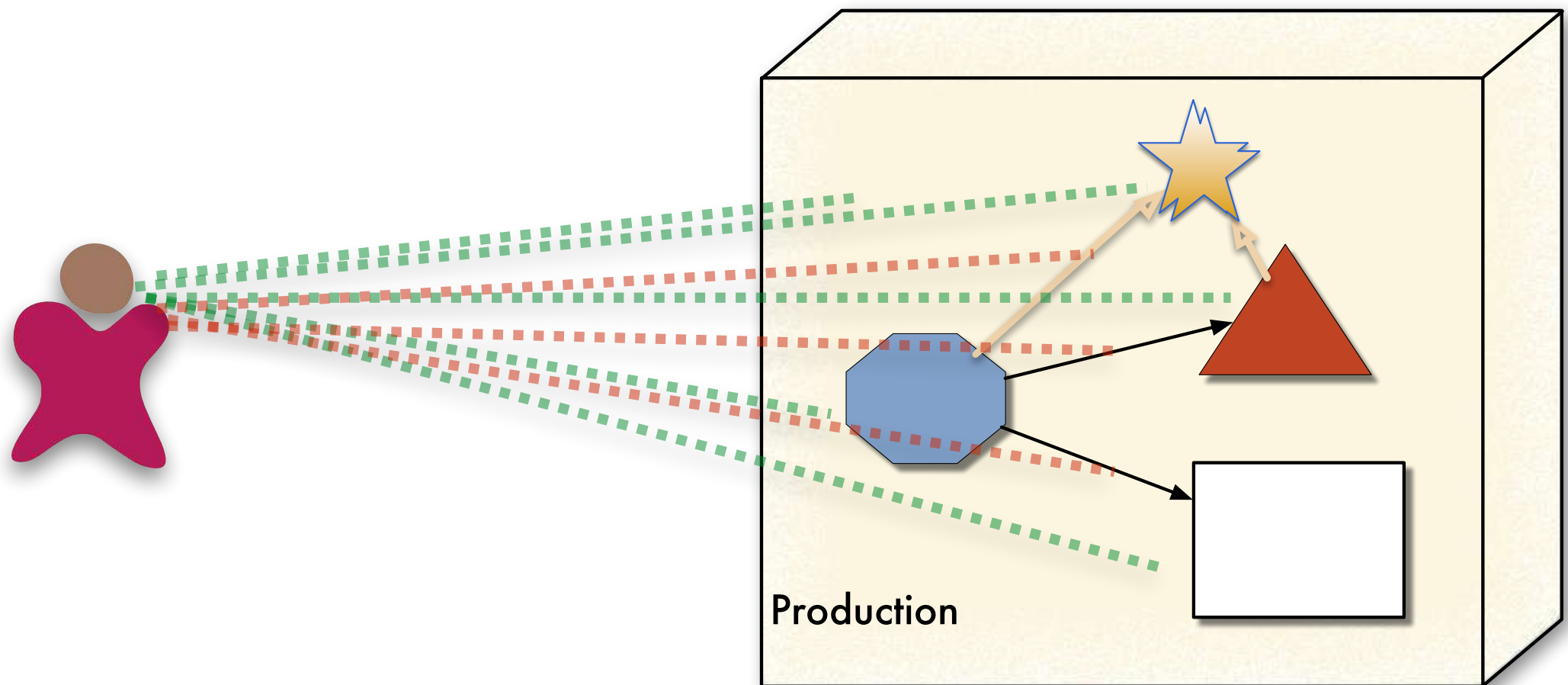
演进接口 (**forward compatible**)



演进接口 (**forward compatible**)



演进接口 (**forward compatible**)

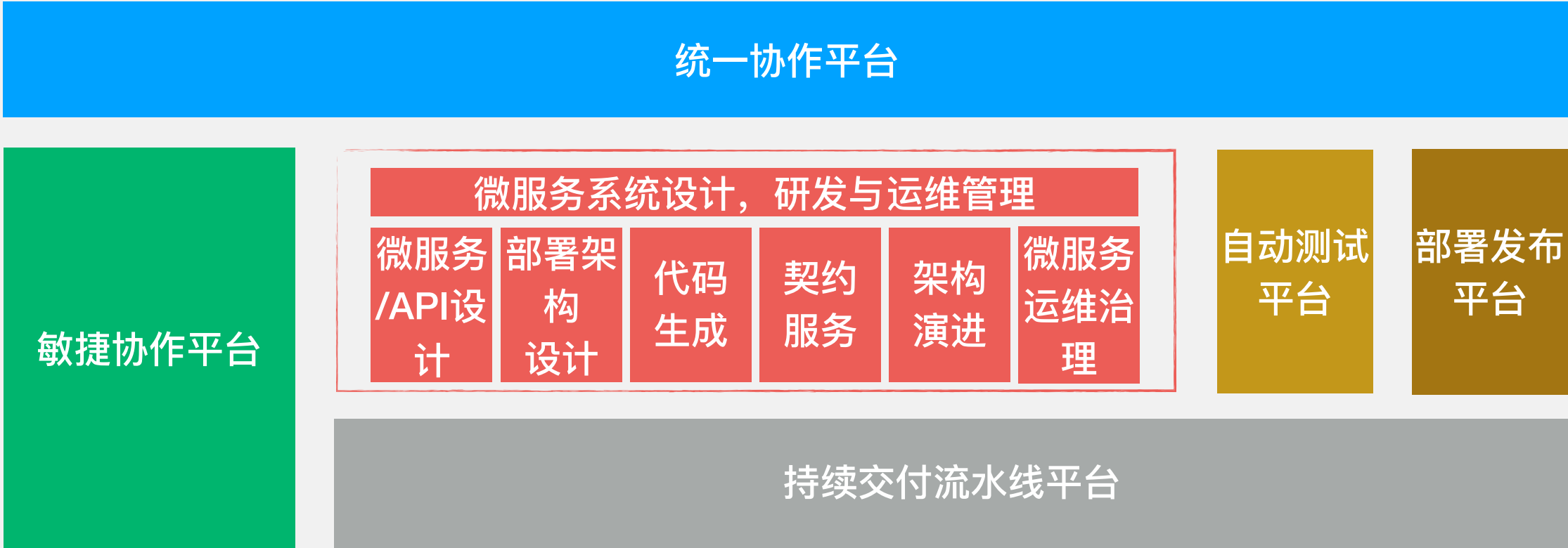


微服务研发平台落地

落地优秀实践，守护架构，降低微服务
研发学习成本

企业研发工具链生态

研发工具连



为某企业做的微服务平台规划

前台

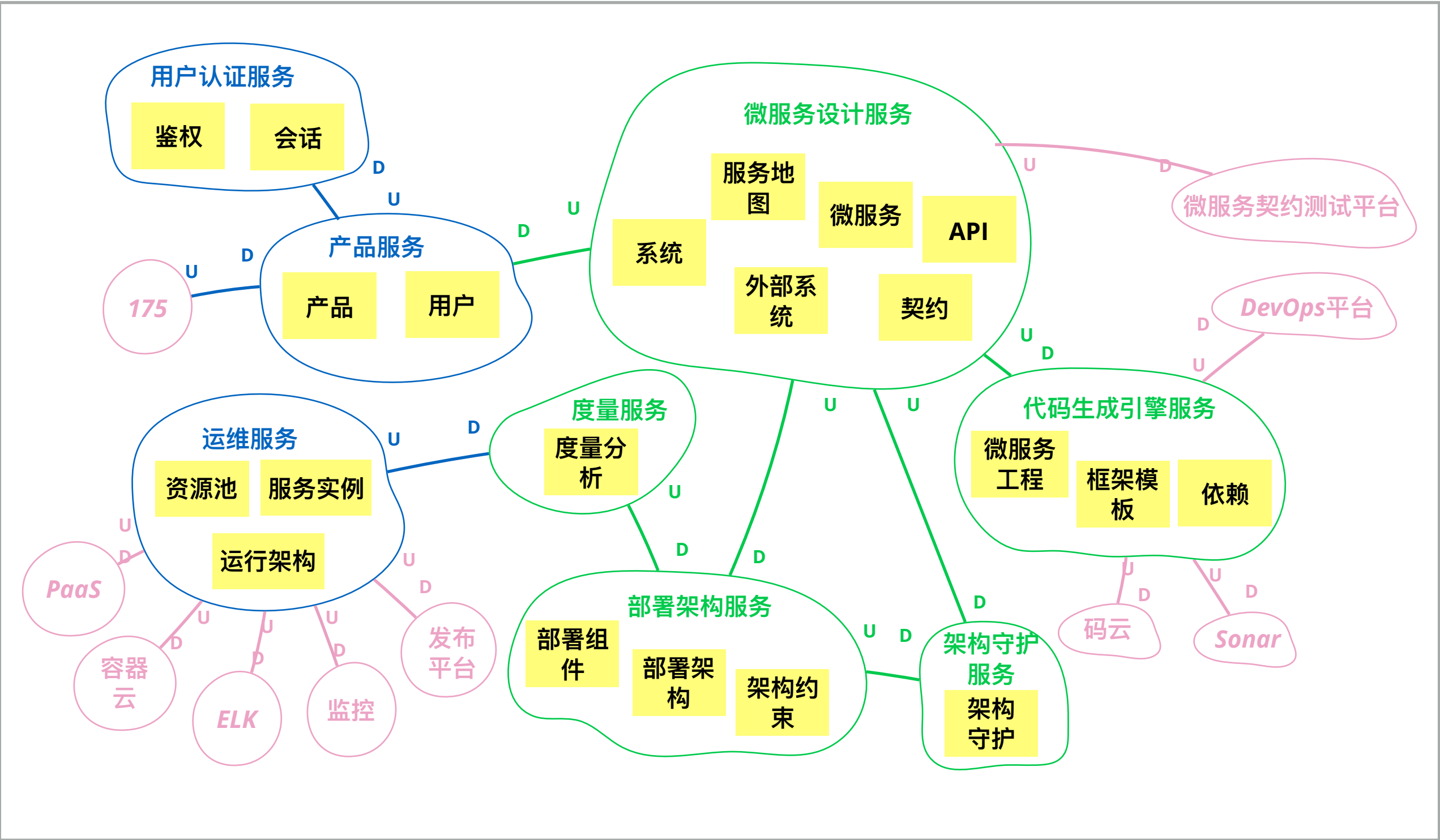
■ 为MVP包含的功能范围

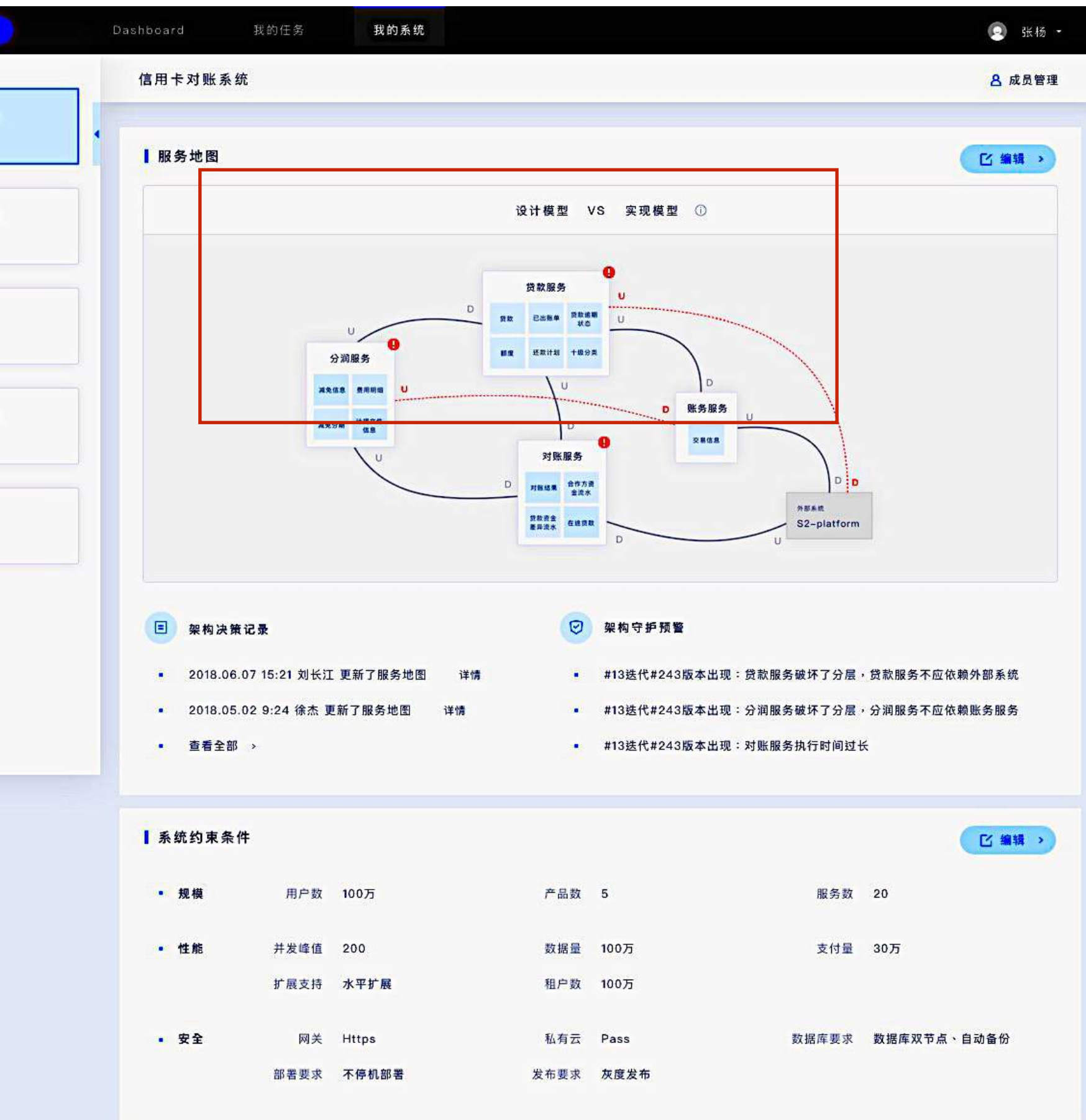
公共部分		需求	微服务架构设计		部署架构设计			微服务工程创建				服务契约	开发	架构演进				运行资源管理	服务运行治理		故障分析
产品、人员维护		查看需求	服务地图设计	API设计	定义架构约束条件	部署架构设计		开发框架	架构风格	资源配置	...	定义契约	代码评审	服务关系分析	领域模型分析	代码质量分析	数据库分析	PaaS、容器云资源管理	服务部署	运行架构治理	问题定位分析
创建系统	我的任务	可视化设计服务地图	设计API	定义通用约束条件	生产环境部署架构设计	测试环境部署架构设计	微服务开发框架定制		生成代码到GIT仓库	定义契约		架构分析-从契约反向识别实现与模型差异				已申请资源登记	服务部署脚本生成	运行时部署架构可视化	服务示例日志		
我的系统列表		服务地图变更历史	导swagger文件	定义服务约束条件	绘制部署架构，组合服务和部署元素		代码架构风格选择			契约变更评审		架构分析-架构Fitness Function检查					部署架构推到目标运行（测试、生产）	数据库状态/版本显示	调用链可视化		
组建团队		设计变更与需求关联	API清单		推荐部署架构（远期规划）		资源配置			未评审契约提示		领域模型-基于代码守护的反向可视化					单服务更新				
团队人员角色权限维护		关联变更到需求	API变更历史		部署架构变更历史		数据库版本管理			分享契约		代码质量分析-配置Sonar地址、展示分析结果									
					关联变更到需求		是否自动生成API/Dockerfile/流水线					数据结构分析-测试环境的反向工程可视化									
	应用架构变更评审			部署架构变更评审				生成流水线			数据结构分析-生产环境脚本导入可视化										

后台/支持

对接资源室获得组织结构	对接外部系统	外部系统维护	架构约束定义维护	部署元素维护	框架管理	GIT插件通信	契约测试Mocker Server	代码引入ArchUnit/log格式入手	集成Sonar Server	对接容器云PaaS	获得运行架构、实例状态	ELK方案
对接175产品定义管理					代码模版生成引擎	对接DevOps流水线	API定义生成Jar	Fitness Function执行引擎			对接流水线执行部署	
						契约配置中心统一管理						

服务地图





服务地图

- 服务定义
- 服务间及与外部系统依赖关系
- DDD对象模型

API管理

- API定义
- 服务与API关系
- 多版本API管理（API演进）

| API列表 + API + 导入Swagger文件

NO.	URI	METHOD	SCENE
1  	/integral/payment	POST	积分支付
2	/integral/payment	POST	积分支付
3	/integral/payment	POST	积分支付
4	/integral/payment	POST	积分支付
5	/integral/payment	POST	积分支付

| 服务工程



创建服务工程

创建好服务地图后可以为服务创建服务工程

查看API

2018-08-06 13:29

张三 更新了API

URI /integral/payment

METHOD POST

SCENE 积分支付

RESPONSE 成功获取200

2018-08-06 13:29

张三 更新了API

URI /integral/payment

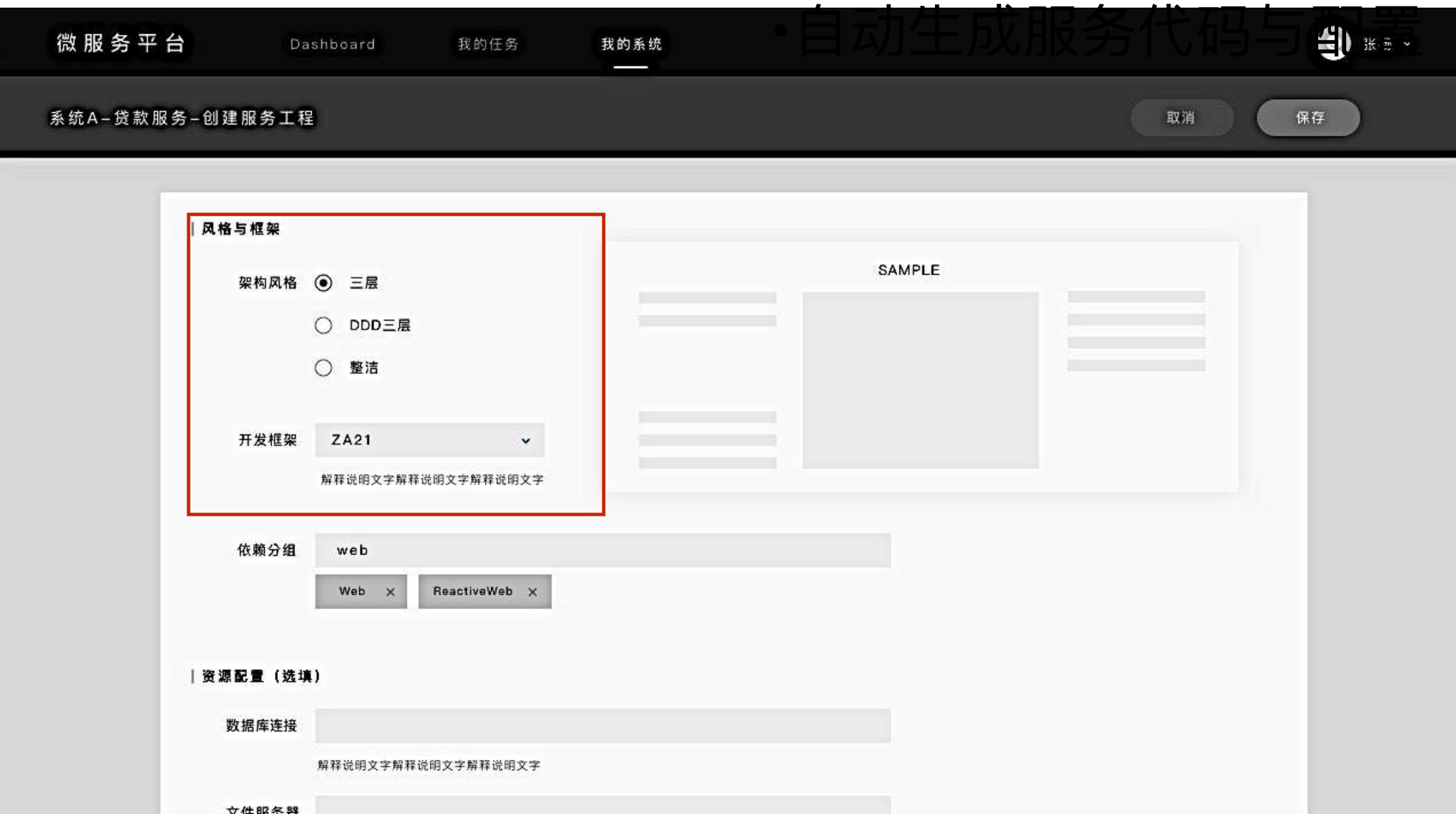
METHOD POST

SCENE 积分支付

RESPONSE 成功获取200

生成服务代码

- 多框架支持
- 多架构风格支持（分层、DDD、整洁等）



平台Dashboard我的任务我的系统张杨

服务-创建服务工程取消保存

风格与框架

架构风格

☒ 三层

☐ DDD三层

☐ 整洁

开发框架

ZA21

解释说明文字解释说明文字解释说明文字

依赖分组

web

Web ×ReactiveWeb ×

资源配置 (选填)

数据库连接

解释说明文字解释说明文字解释说明文字

文件服务器

解释说明文字解释说明文字解释说明文字

MQ

解释说明文字解释说明文字解释说明文字

数据库管理 (选填)

数据库

flyway

版本号

解释说明文字解释说明文字解释说明文字

其他配置 (选填)


☐ 生成API 解释说明文字解释说明文字解释说明文字

☐ 生成docker file 解释说明文字解释说明文字解释说明文字

☐ 创建流水线 解释说明文字解释说明文字解释说明文字

☐ 生成契约测试代码库 解释说明文字解释说明文字解释说明文字

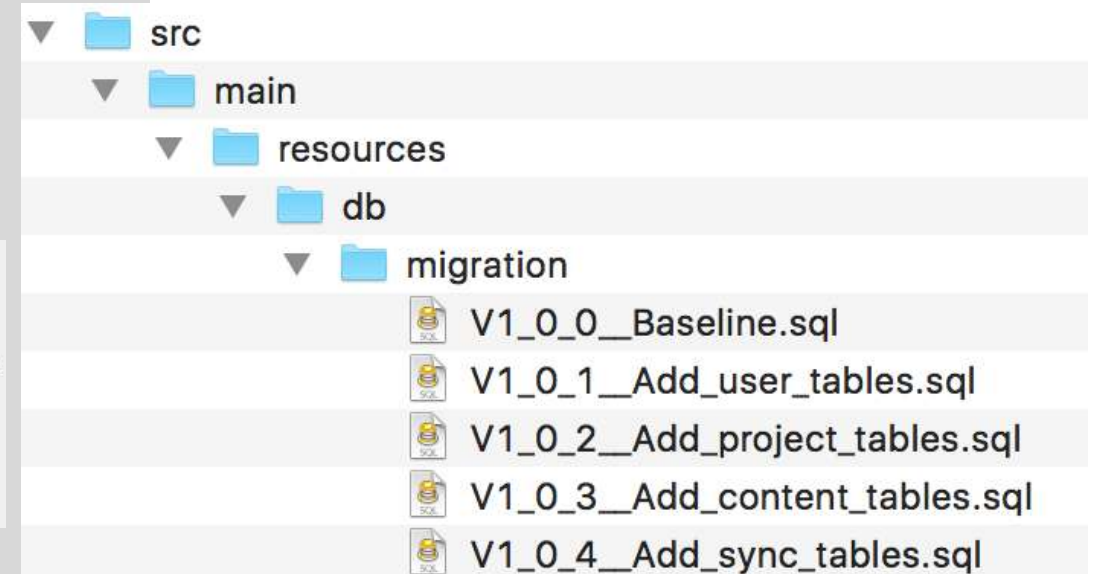
SAMPLE





数据库演进与版本化管理

- 数据库引入Flyway等
- 自动生成初始脚本库
- 自动化变更DB



查看契约



2018.08.06 13:29

+ 评审意见

张三 更新了契约定义

消费者 贷款服务 -----> 生产者 账务服务

API API /integral/payment

METHOD POST

REQUEST method:GET,uri: /app/update/1.0.0method:GET,uri: /
app/update/1.0.0method:GET,uri: /app/update/1.0.

RESPONSE method:GET,uri: /app/update/1.0.0method:GET,uri: /
app/update/1.0.0method:GET,uri: /app/update/1.0.

描述 贷款服务调用对账服务的核算模块

2018.08.06 13:29

张三 创建了契约定义

消费者 贷款服务 -----> 生产者 账务服务

API API /integral/payment

METHOD POST

契约

- 契约定义（依赖、集成关系）
- 契约变更评审
- 契约变更历史记录
- 支持契约测试

架构演进守护

架构演进守护—
从契约反向识别实现与模型差异

架构演进守护—
Atomic Fitness Function检查

领域演进守护—
DDD领域模型的反向可视化

服务地图

API管理

契约定义

通过有效的契约定义推导
服务与API之间关系，并
与设计对比
(holistic fitness
function)

架构演进守护

架构演进守护—
从契约反向识别实现与模型差异

架构演进守护—
Atomic Fitness Function检查

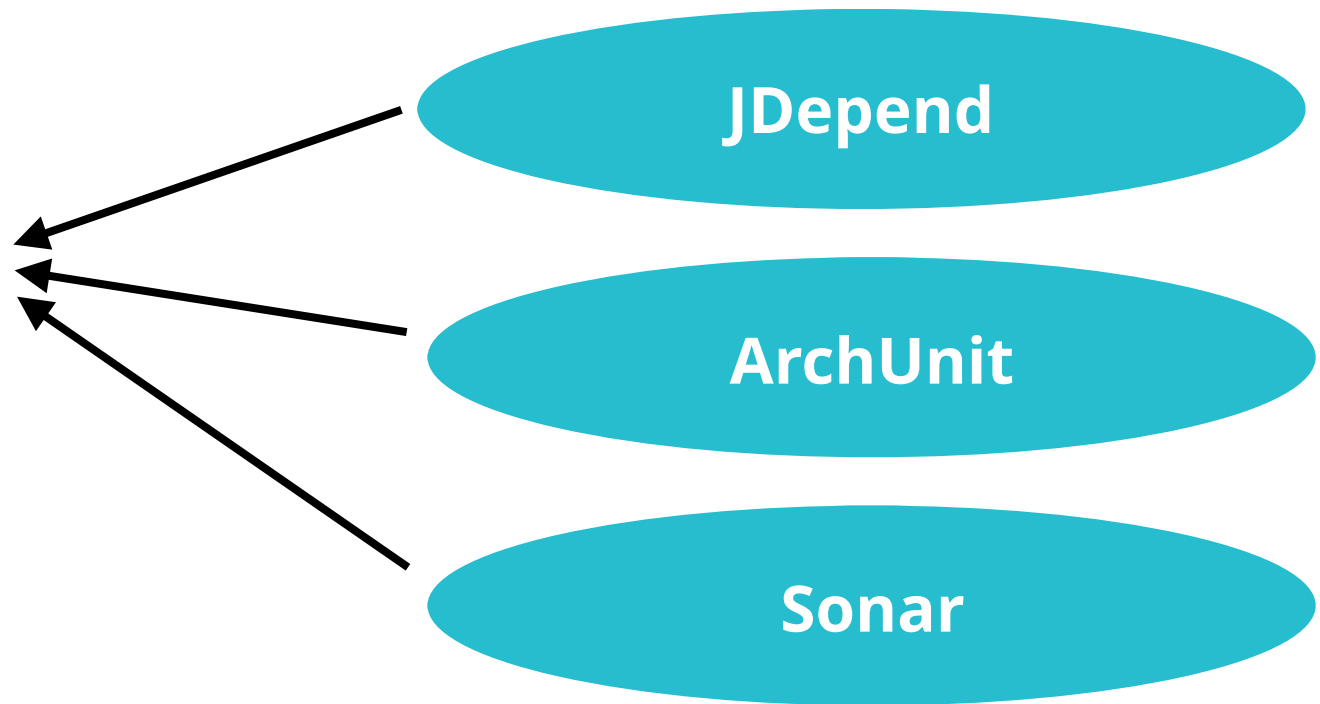
领域演进守护—
DDD领域模型的反向可视化

(atomic fitness function)

JDepend

ArchUnit

Sonar



架构演进守护

架构演进守护—
从契约反向识别实现与模型差异

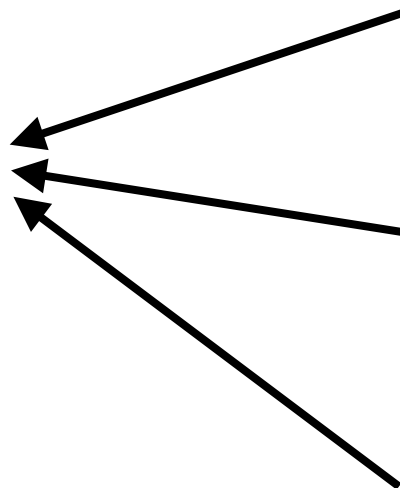
架构演进守护—
Atomic Fitness Function检查

领域演进守护—
DDD领域模型演进

DDD建模

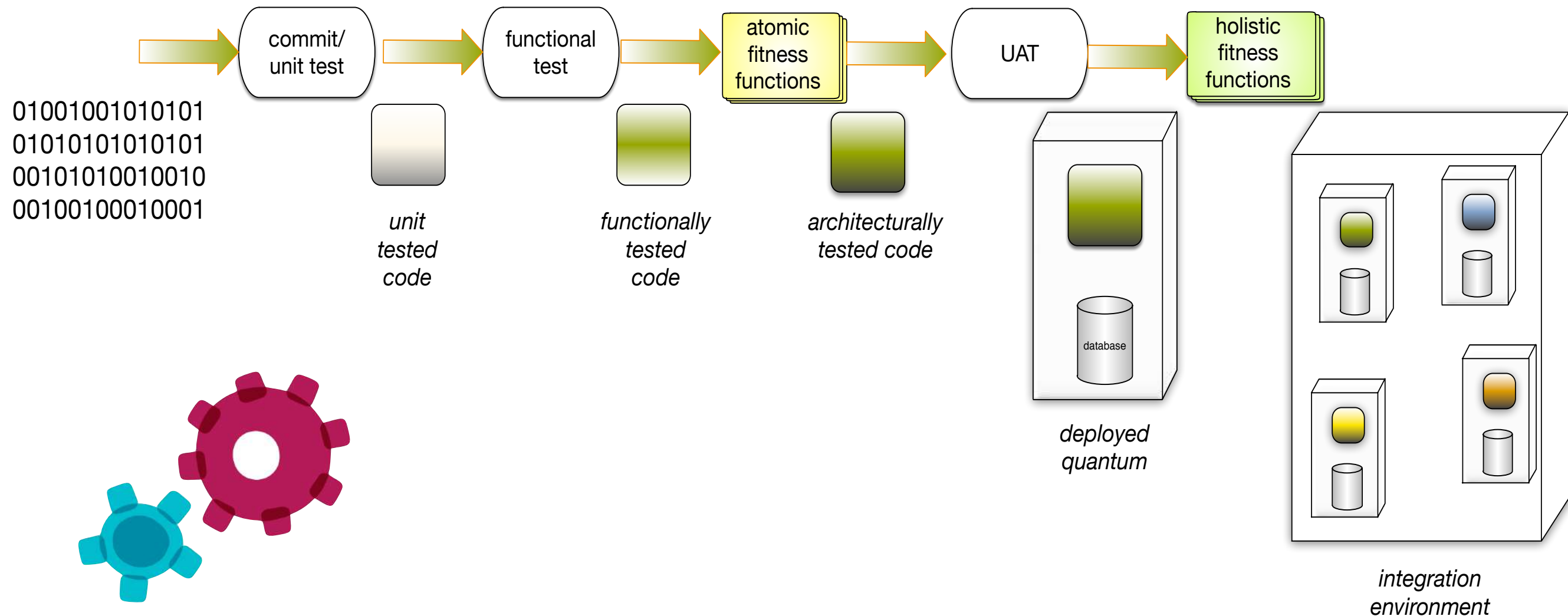
自动生成代码

代码反向生成模型



采用CD部署流水线

自动化Fitness Functions



数据库连接

解释说明文字解释说明文字解释说明文字

文件服务器

解释说明文字解释说明文字解释说明文字

MQ

解释说明文字解释说明文字解释说明文字

| 数据库管理（选填）

数据库

flyway



版本号



解释说明文字解释说明文字解释说明文字

| 其他配置（选填）



生成API

解释说明文字解释说明文字解释说明文字



生成docker file

解释说明文字解释说明文字解释说明文字



创建流水线

解释说明文字解释说明文字解释说明文字



生成契约测试代码库

解释说明文字解释说明文字解释说明文字

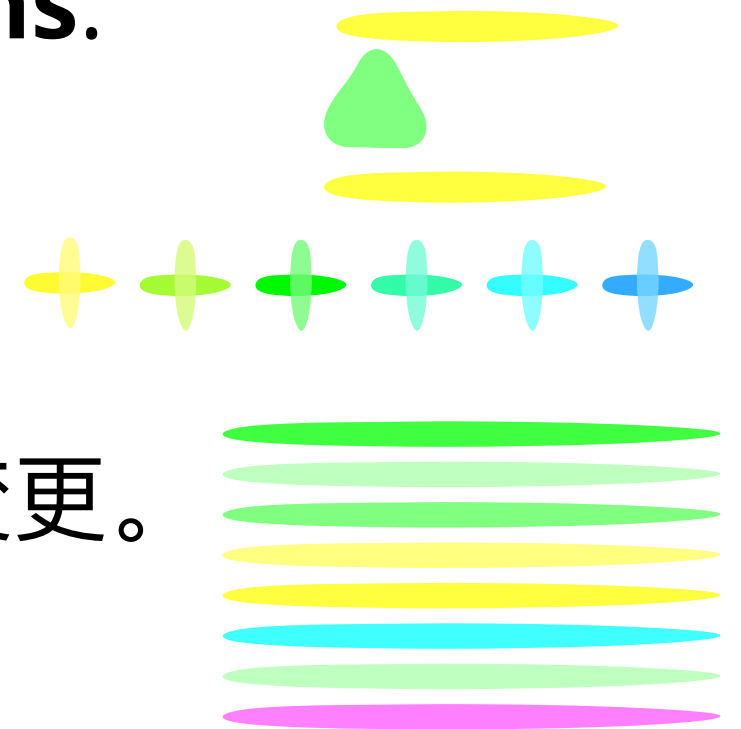
结合流水线执行 Fitness Function

- 自动生成服务的部署流水线
- 自动生成docker file
- 自动在pipeline中配置必要的fitness functions

演进式架构

An evolutionary architecture supports
guided, incremental change
across **multiple dimensions**.

演进式架构支持
在各个架构设计维度上，
沿着特定方向进行频繁增量式变更。





THANK YOU

 @neal4d
 nealford.com

 @rebeccaparsons

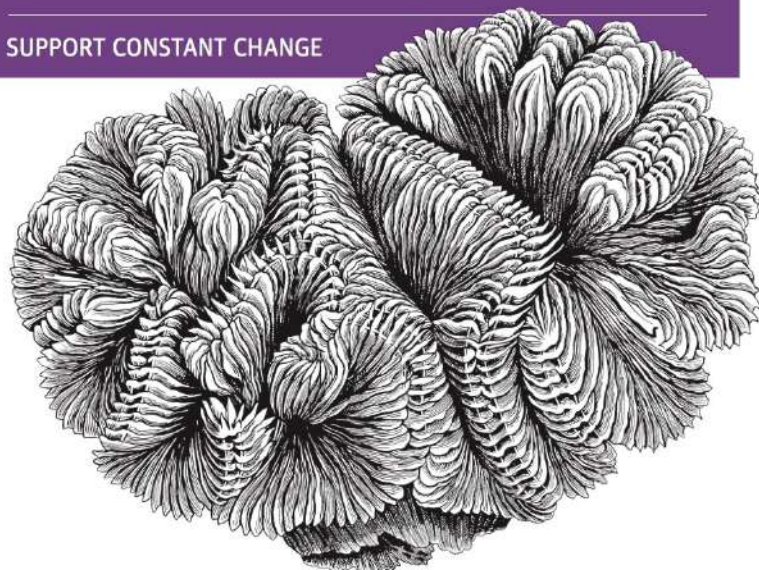
 @patkua



O'REILLY®

Building Evolutionary Architectures

SUPPORT CONSTANT CHANGE



Neal Ford, Rebecca Parsons & Patrick Kua

DDDCHINA