TiD 2017
质量竞争力大会

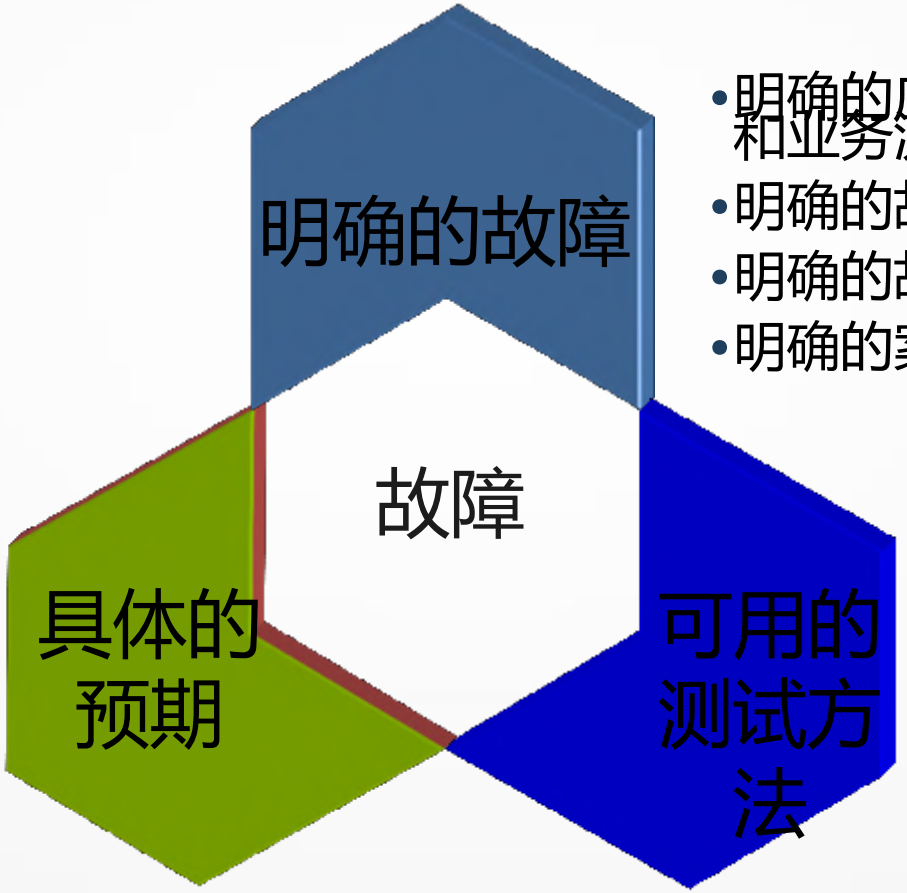
云化系统可靠性测试实践

华为软件 唐硕

下一代软件研发
SOFTWARE DEVELOPMENT

- 当前可靠性测试的主要方法
- 云化趋势下的变化
- 云化下可靠性测试方法改进
- 总结

基于故障模式库的可靠性测试设计

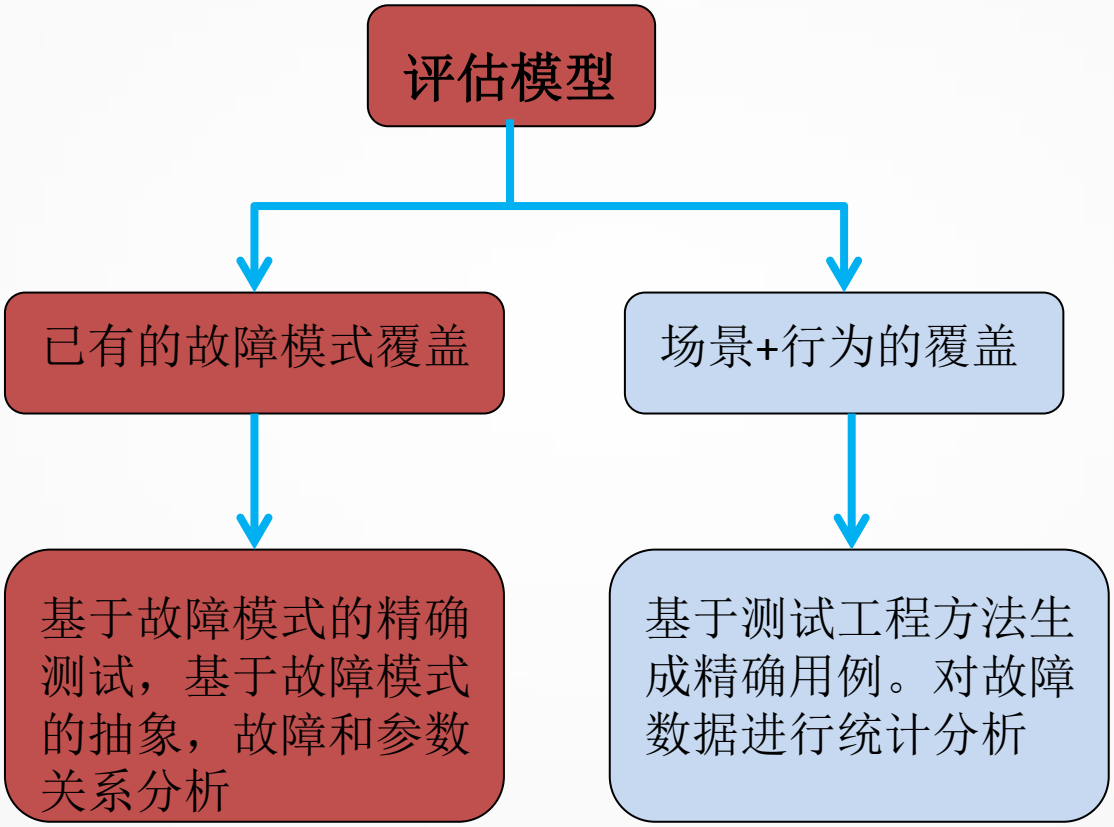


- 明确的应用场景和业务流
- 明确的故障原因
- 明确的故障来源
- 明确的案例档案

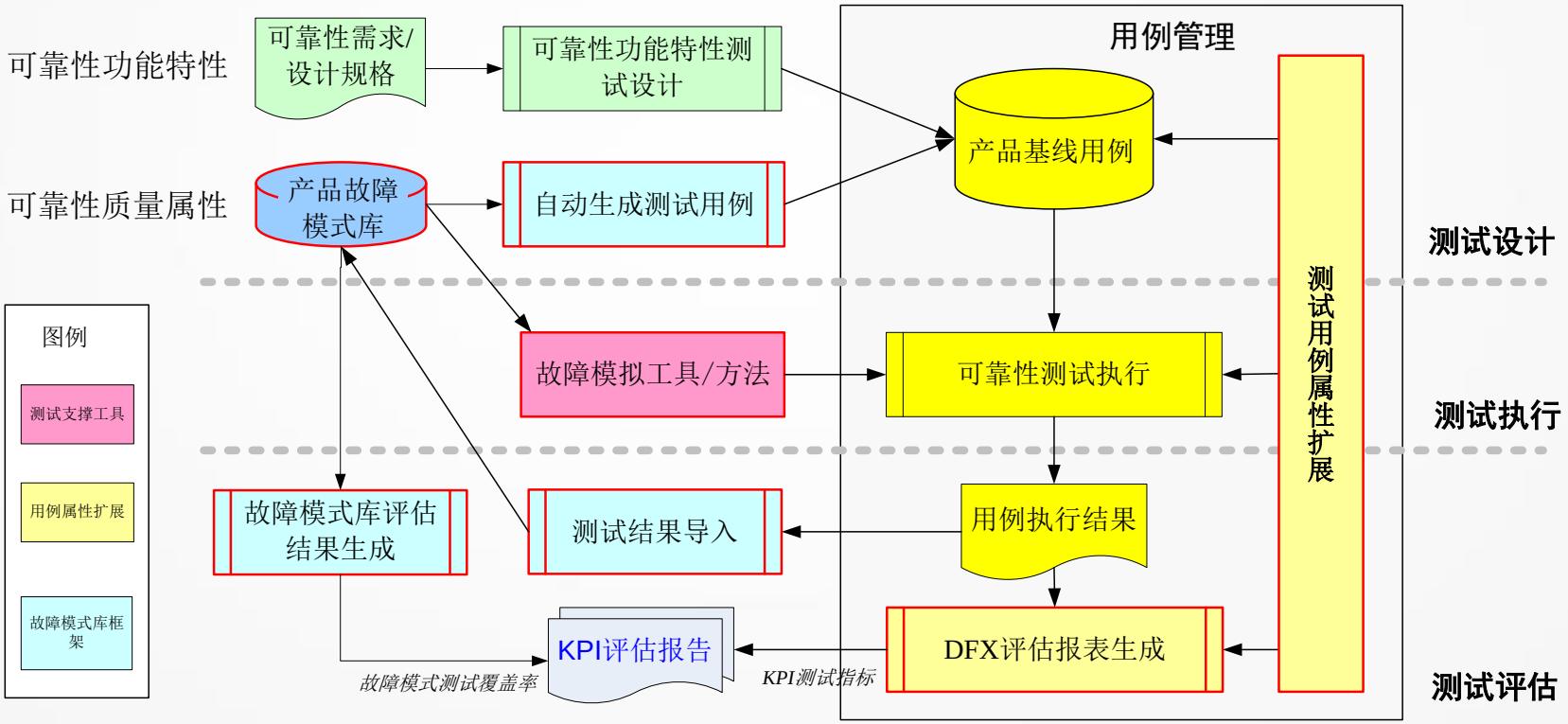
- 告警类型
- 如何容错保护
- 是否需要修复
- 是否自动修复
- 修复措施

- 故障模拟方法
- 故障恢复方法
- 设计参考

可靠性测试主要方式


评估模型
已有的故障模式覆盖
场景+行为的覆盖
基于故障模式的精确测试，基于故障模式的抽象，故障和参数关系分析
基于测试工程方法生成精确用例。对故障数据进行统计分析

TiD2017 基于故障模式库的测试闭环



- 当前可靠性测试的主要方法
- 云化趋势下的变化
- 云化下可靠性测试方法改进
- 总结

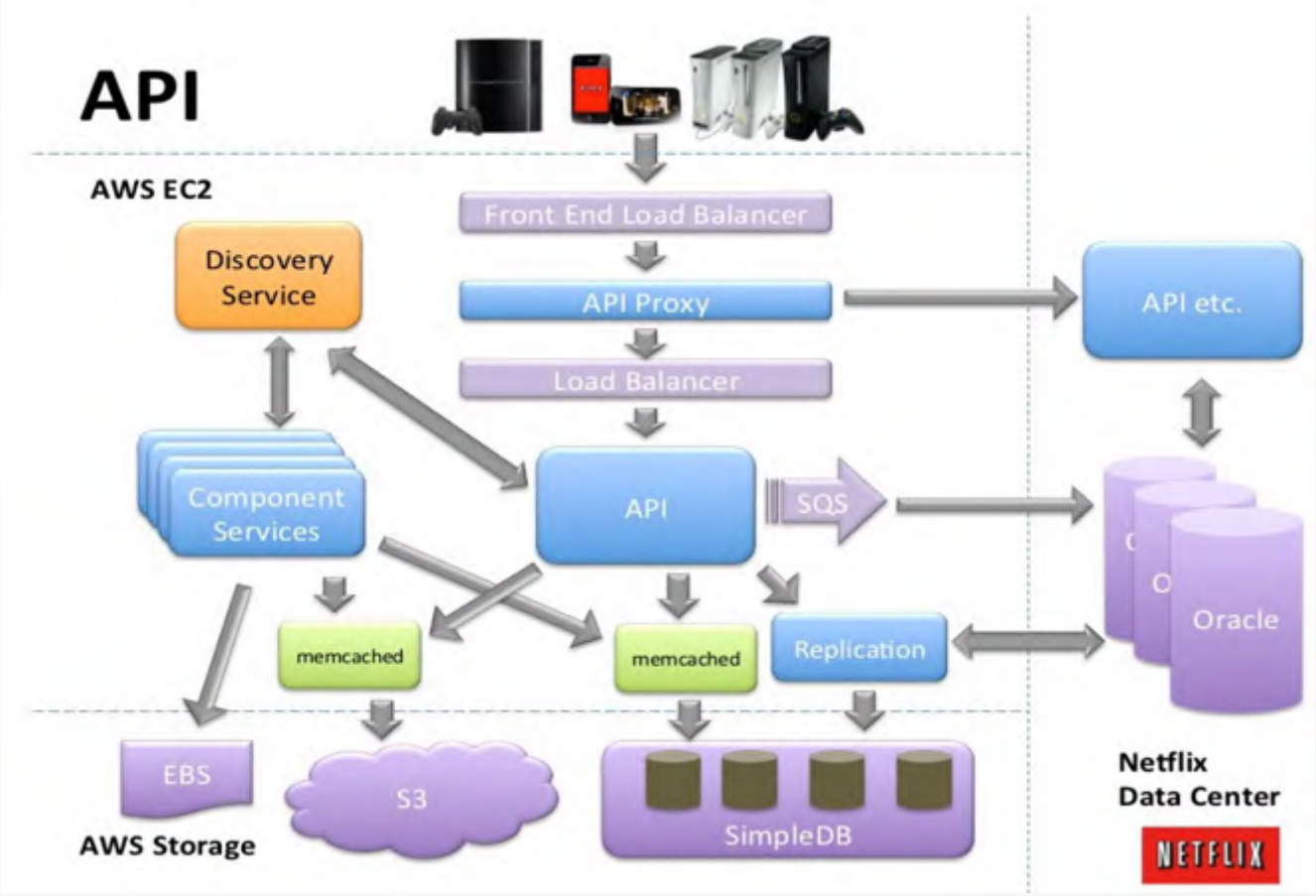
产品的变化

API开放层

服务层

数据层

云基础服务



微服务架构

敏捷
持续交付

客户对可靠性要求变化

用户要求：

设备不出故障

出了故障不影响主要功能和业务

功能受影响能尽快定位和恢复

可用度A	9的个数	年停机时间（分钟）	适用产品
0.999	三个9	500	电脑或服务器
0.9999	四个9	50	企业级设备
0.99999	五个9	5	一般电信级设备
0.999999	六个9	0.5	更高要求电信级设备

TiD2017

当前可靠性测试问题

基于X86系统提供云服务，组网规模大，故障率高

- XX混合云的**伸缩组服务**基于X86服务器为应用提供集群&伸缩能力；
- XX私有云的**服务**基于X86 PC服务器设计，支持200台设备，提供分布式数据库能力；
- PC服务器硬件成本低，但故障率高；

分布式/集群可靠性测试方法短板

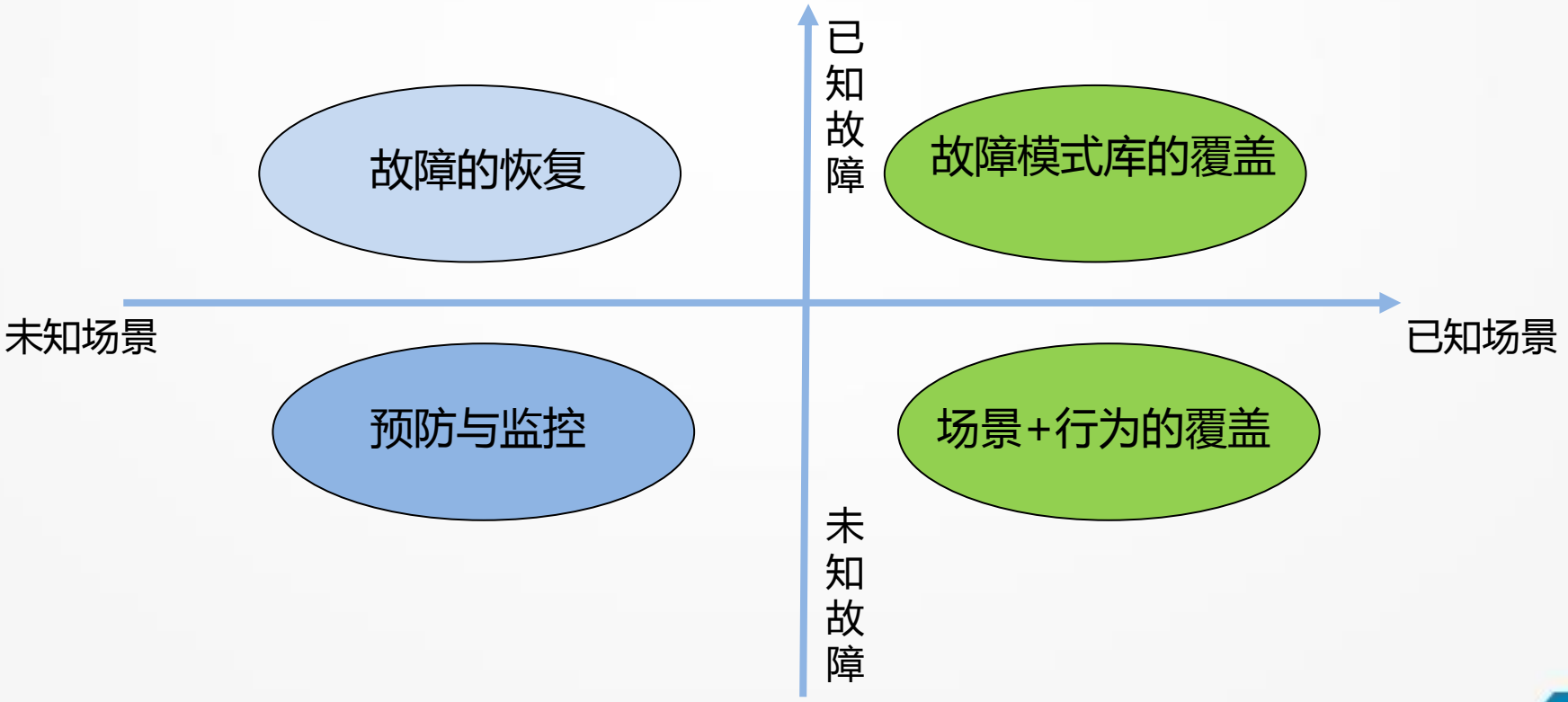
- 传统可靠性测试在固定节点进行故障注入，与实际场景不符合；
- 混合故障场景验证不足，测试工作量大且覆盖不全；
- 缺少大数据下的结果校验方法；

测试效率低，成本高

- 项目周期短，大量的时间花费在已知故障覆盖上；
- 当前的故障影响范围大，需要花费大量时间在分析上；

当前可靠性测试问题

故障模式库和场景+行为的可靠性测试可以理论上保证已知场景和已知故障的覆盖，但是由于敏捷和持续交付，无法保证进行完备的可靠性测试



- 当前可靠性测试的主要方法
- 云化趋势下的变化
- 云化下可靠性测试方法改进
- 总结

Simian Army

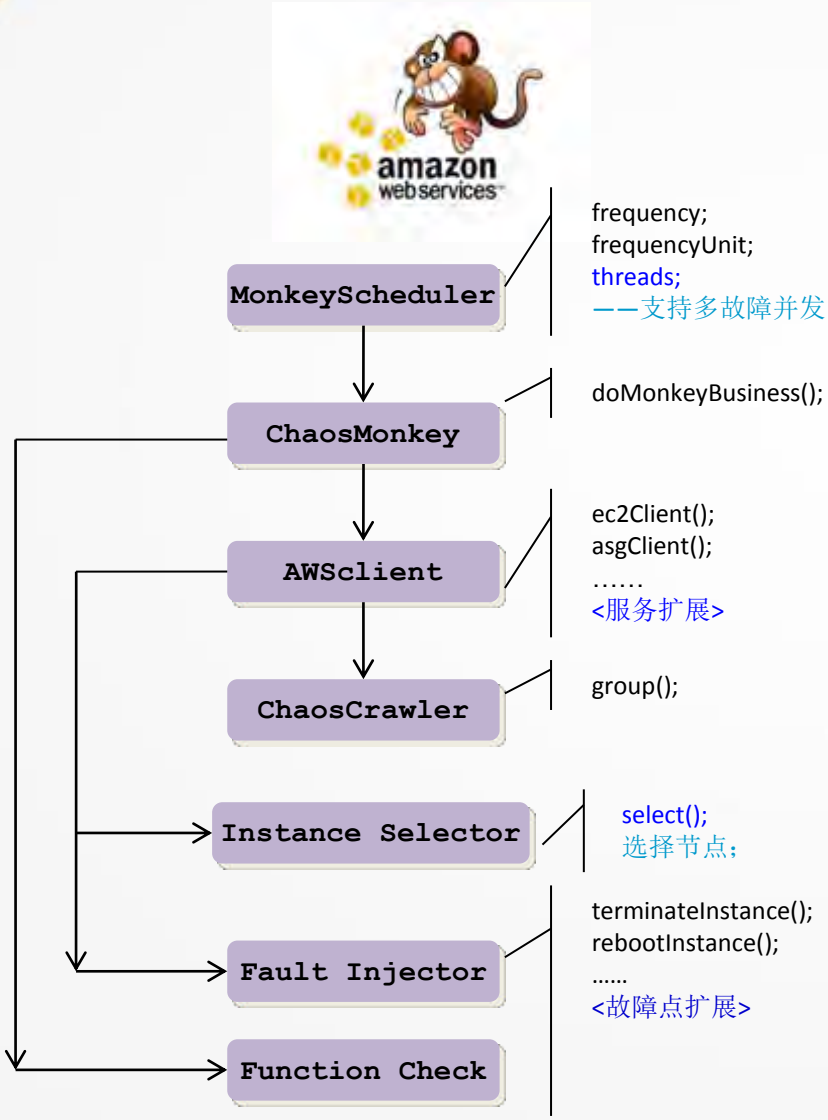
2012年8月，Netflix开源了他们那被人惦记好一阵子的“Chaos Monkey”，这是一套用来故意把服务器搞下线的软件，可以测试云环境的恢复能力。

Netflix团队让Chaos Monkey亮相的时间，最早是在2010年12月的一篇官博文章，文章内容是他们在AWS云上托管其热门视频流服务所得到的经验教训。文中总结了一点，叫做“避免失败的最好办法是经常失败”，反映Netflix通过主动破坏自身环境来发现弱点的做法。

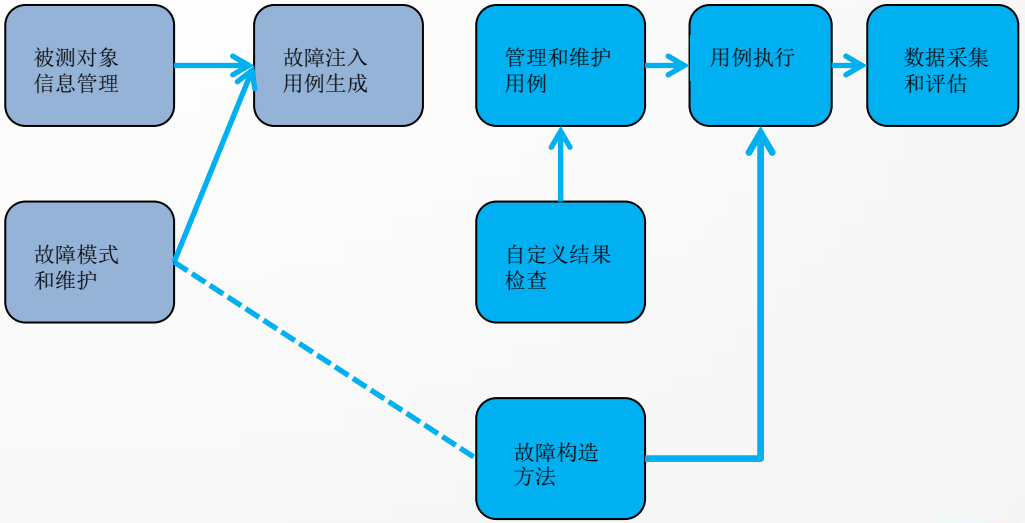
选1个例子：Chaos Monkey的19个函数

```
BlockAllNetworkTrafficChaosType()   //阻塞网络
BurnCpuChaosType()          //消耗CPU
BurnIoChaosType()           //IO
DetachVolumesChaosType()    //解挂磁盘卷组
FailDnsChaosType()          //DNS失效
FailDynamoDbChaosType()     //失效DynamoDB实例
FailEc2ChaosType()          //失效EC2实例
FailS3ChaosType()           //失效S3实例
FillDiskChaosType()         //磁盘满
KillProcessesChaosType()    //杀进程
NetworkCorruptionChaosType() //网络腐化
NetworkLatencyChaosType()   //网络时延变长
NetworkLossChaosType()      //网络丢包
NullRouteChaosType()        //空路由
ShutdownInstanceChaosType()  //shutdown某个实例
ScriptChaosType()           //跑一个指定的脚本
ChaosEmailNotifier()        //email通知
sendTerminationNotification(); //termination通知
sendTerminationGlobalNotification(); //全局termination通知
```

基于CHAOS Monkey的拓展



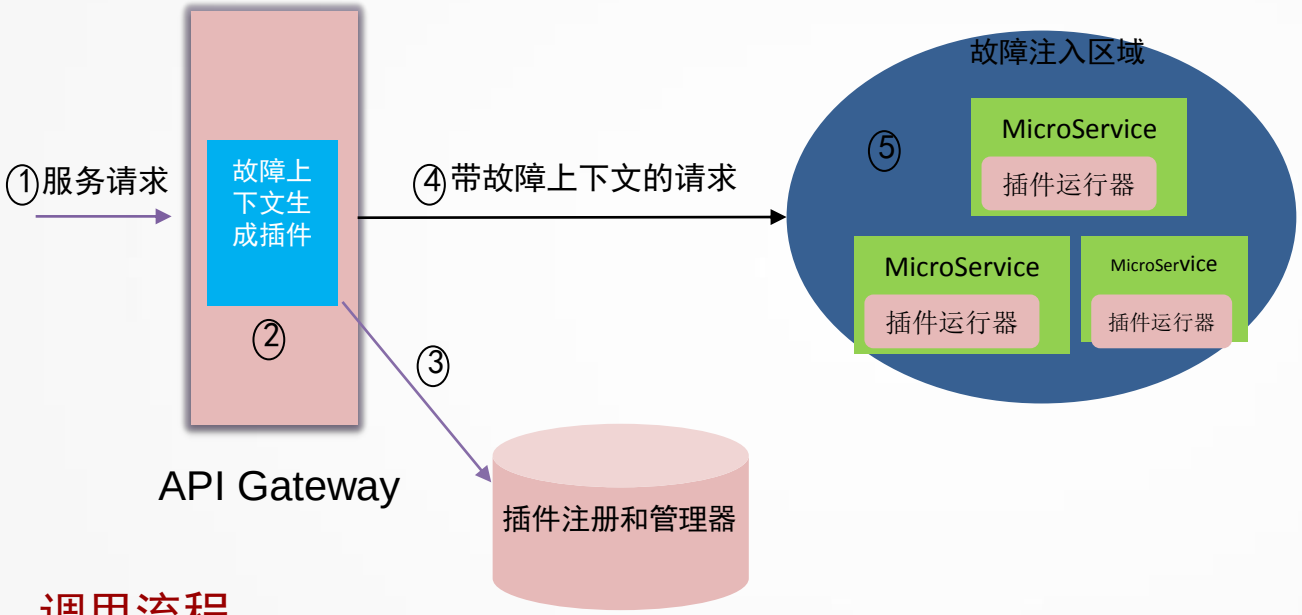
- **框架改造：**基于Chaos Monkey工具扩展支持的服务类型和故障类型；
- **功能扩展：**系统后续新增API接口后可灵活扩展故障类型；
- **测试用例：**增加对用例的自动生成和用例的管理，并能自定义结果检查；
- **可靠性评估：**基于故障注入执行结果对被测系统进行评估



故障影响范围

Service A
Service B
Request 1
Request 2
Request 3
故障粒度：粗→细
覆盖率 ：低→高
Service A
Service B
Request 1
Request 2
Request 3

精确控制故障范围



调用流程

1. 服务请求发至API Gateway.
2. API Gateway截获请求后，根据预定义故障场景判断是否注入故障。故障的元数据被加入到统一上下文并往下传递。
3. API Gateway会获取最新的故障场景定义。
4. 带有故障上下文的请求被发送到微服务（故障注入区域）。
5. 内嵌的插件运行器执行故障注入逻辑

- 精确的控制故障范围，不影响非相关接口
- 框架可以被扩展支持其他业务需求，例如性能监控，数据收集等

TiD2017

- 当前可靠性测试的主要方法
- 云化趋势下的变化
- 云化下可靠性测试方法改进
- 总结

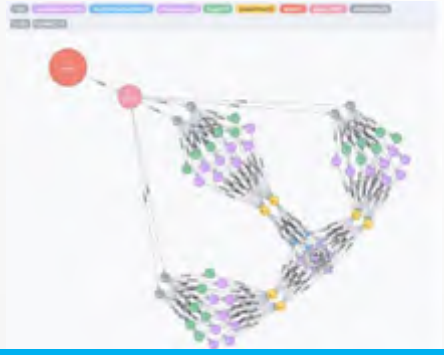
总结

■可靠性用例随着故障模式的增多会膨胀，产品的持续交付无法保证可靠性的完备测试

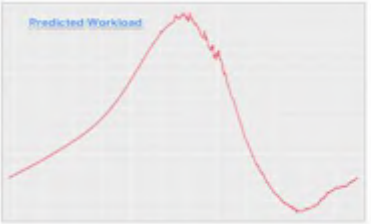
基于机器学习的动态生成方式来进行用例生成，这种方式的核心思想是基于遗传算法

基于产品以往发现的问题（包含研发和现网的）使用大数据挖掘算法，反复迭代获取潜在风险较大的故障和模块

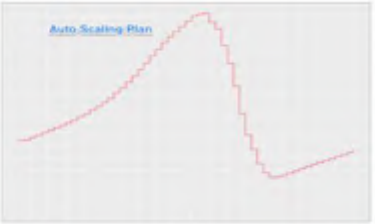
■当前的可靠性测试并没有解决未知故障+未知场景问题



自动监控工具实时监控（如：Simianviz/原名Sipgo ）



预测算法得到的负载模型（如：Netflix Scryer）



生成的伸缩执行计划

Q&A

请备注：姓名 公司
欢迎交流软件测试，
骑车的话题

部门的微信公众号

软件质量报道：朱少民老师的公众号
软件质量、软件测试有深入的理解