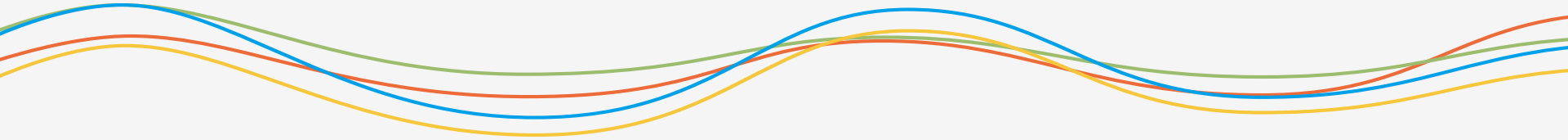
ThoughtWorks®

APMCon

打造复杂事务性能的智能引擎

金明
ThoughtWorks



关于我

金明

ThoughtWorks中国
ScaleWorks云产品总监
持续交付与DevOps Lead

我的一些演讲——
“Up! To Docker PaaS”
“Docker in Action”
“从零开始，构建企业IT云平台”
“从虚拟化到私有云之实践”
“DevOps: 组织级敏捷落地与推广之钥”
“云化运维：持续交付落地的基石”
……

ThoughtWorks 是一家以融合创新、技术、设计的咨询公司，成立于 1993 年，在全球 12 个国家拥有 29 间办公室。目前中国有6个办公室，超过800名员工。

12	30	6	800+
个国家	个办公室	个在中国	名中国员工

“敏捷”和“精益”创新的领导者

APMCon

大 纲

APMCon

- 一个移动App的故事
- 为什么我们都在谈APM?
- APM工具为什么沦为摆设?
- APM本质是Business Performance Management
- 面向复杂事务性能的BPM智能引擎方案
- BPM实施方法
- BPM真正解决IT-业务对齐的案例

一个移动App的故事

IT开发的App在App Store里面反馈不稳定，用户评分低

IT开发的App性能很慢，操作后要等很久

IT的App开发能力低，很难定位故障原因

……？

Business Leader

Apps/IT Leader

IT提供的App性能指标

APMCon

JVM指标

gc.cms.count
gc.parnew.time
heap_memory
heap_memory_committed
heap_memory_init
heap_memory_max
non_heap_memory
non_heap_memory_committed
non_heap_memory_init
non_heap_memory_max
thread_count
catalina.jsp_count
catalina.jsp_queue_length
catalina.jsp_reload_count
catalina.jsp_unload_count

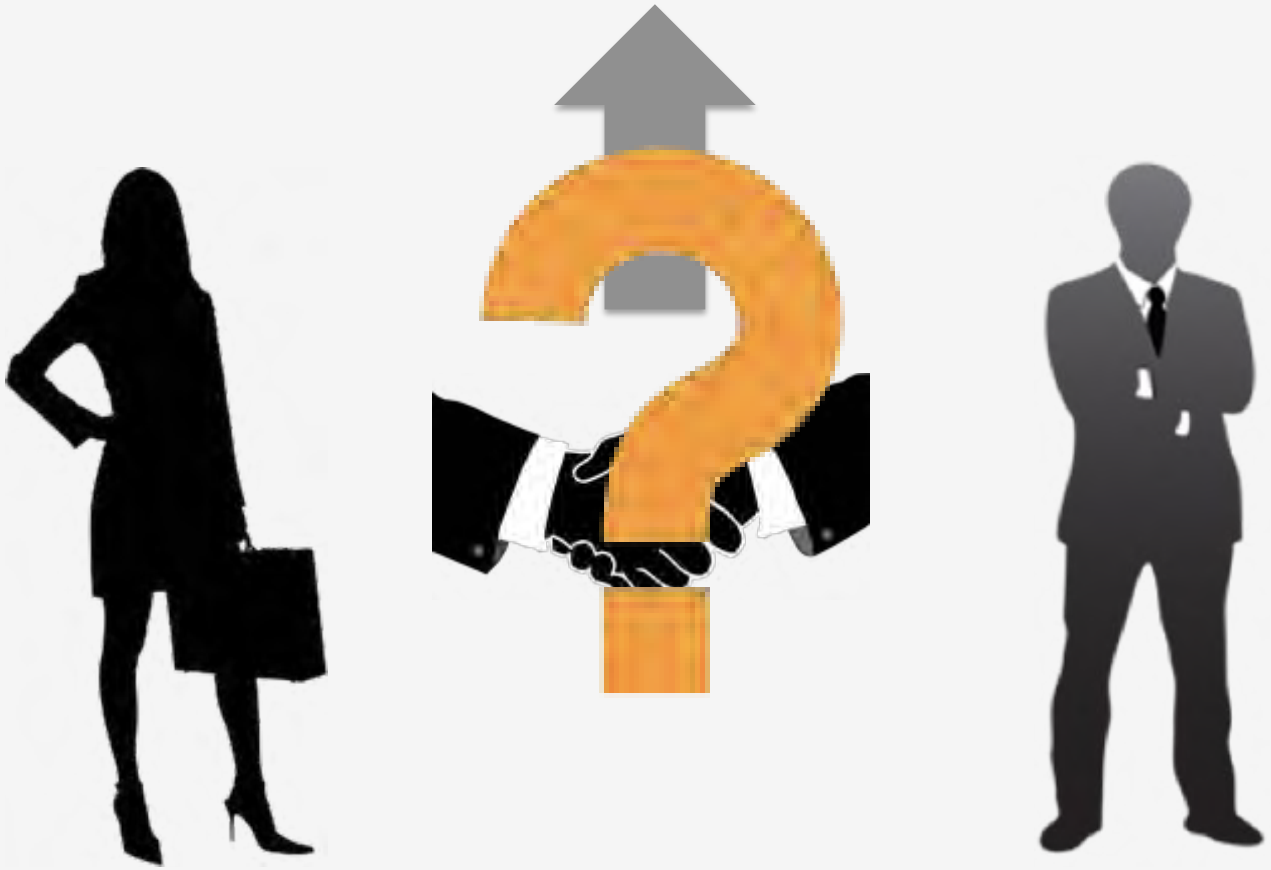
Tomcat指标

bytes_rcvd
bytes_sent
error_count
max_time
processing_time
request_count
servlet.error_count
servlet.processing_time
servlet.request_count
threads.busy
threads.count
threads.max

MySQL指标

innodb.buffer_pool_size
innodb.data_reads
innodb.data_writes
innodb.os_log_fsyncs
net.connections
net.max_connections
performance.open_files
performance.queries
performance.questions
performance.slow_queries
performance.table_locks_waited
performance.threads_connected

IT-Biz对齐、持续交付



Business Leader

Apps/IT Leader

为什么我们都在谈APM?

数字经济时代，业务与技术的融合越来越深、越来越快

APMCon

加速地交付产品和服务

90亿　全世界的可连接设备*

客户驱动创新

56%　客户期望更多的自服务能力

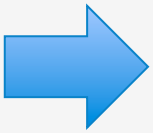
全球化价值链

16万亿　2010年的全球出口总额（美元）

数据需求增长率

10倍　从2007到2011数字数据的增长

50%



根据受访企业CIO，50%IT预算花费在持续运营和维护费用上**

**来源: Forrester Research, Inc. "2012 IT Budget Planning Guide For CIOs," October 27, 2011 by Craig Symons

微服务、移动等技术带来企业的IT架构越来越复杂

APMCon

现代IT系统架构：

- 微服务化
- 容器化
- API接口调用
- 云平台
- 大数据平台
- 移动应用

为什么APM沦为“摆设”?

当前的APM工具

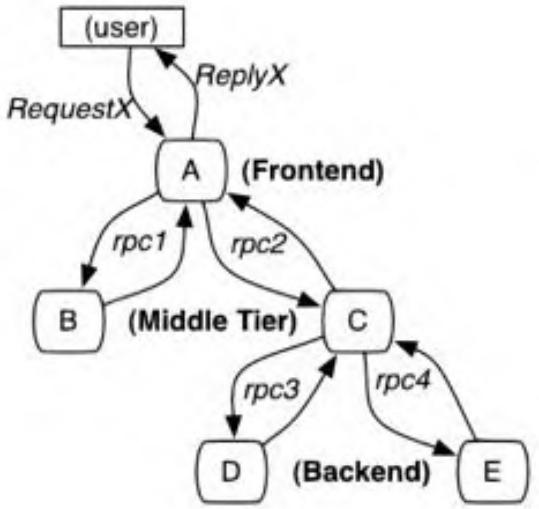
APMCon

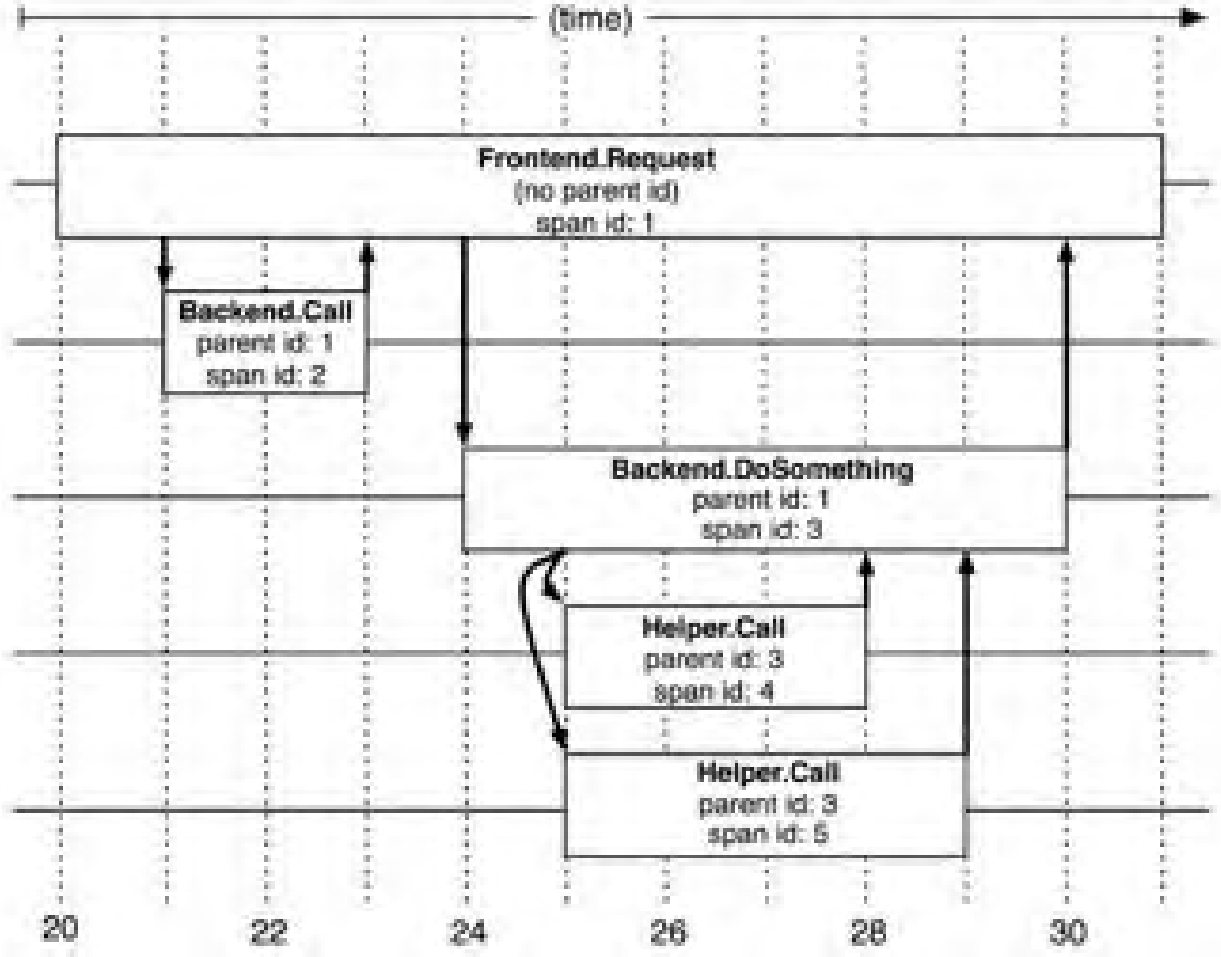
商业化产品

开源产品

- ELK
- Pinpoint
- CAT
- Zipkin by Twitter

APM工具的理论基石：Google Dapper 论文





真实的世界：异构系统、复杂依赖关系制约运维自动化

APMCon

异构的基础设施

企业*IT*基础设施环境与前几年相比前所未有复杂，往往混杂公有云、私有云、混合云以及多地数据中心等

复杂的依赖关系

企业运营一套系统往往有数套甚至数十套后台软件做支撑，其中包括数据库、*Web*服务器、*ERP*以及企业的其他应用信息系统等

割裂的数据

企业*IT*各种系统和应用系统中，包括大量的数据、资源使用指标等，它们反映了应用的数据变化、流程流转等，但这些数据往往分散在各个系统内部，形成了割裂的“数据孤岛”。

APM工具遇到的挑战

APMCon

- 商业遗留系统，无法承受改造的工作量和风险
- 基于探针式数据埋点得出的信息太琐碎，大部分信息没有用
- 数据关联逻辑太弱，只能维护父子关系
- 异构系统的数据无法关联，缺乏关联维度（存储、主机、虚拟机、应用、数据库等）
- 数据分析能力不够，只有简单的汇总聚合
- 数据报表展示的基本上都是已知的信息，很少能够发现业务深度的信息
- 强烈依赖于运维人员的分析和处理能力
- 分析的结果与异常处理脱节，未能端到端完成自动化运维

APM本质是Business Performance Management

业务系统的典型日志样本

APMCon

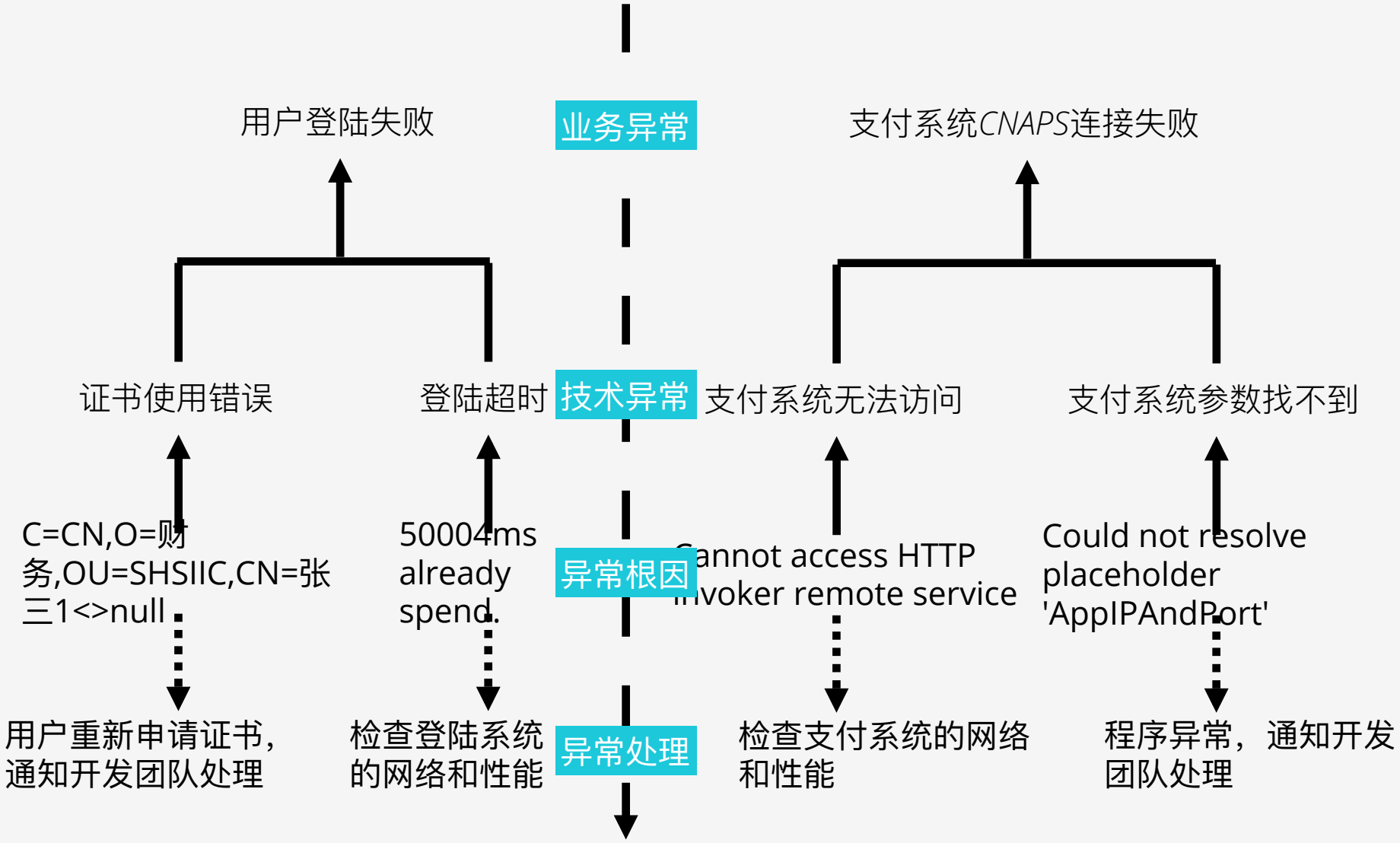
```
00:00:00 ERROR [QuartzShell] Job DEFAULT.rqs.BuildRecordJobDetail threw an unhandled Exception:
org.springframework.scheduling.quartz.JobMethodInvocationFailedException: Invocation of method 'execute' on
target class [class com.nstc.terraQuartz.scheduling.quartz.AutoJobProxyImpl] failed; nested exception is
java.lang.RuntimeException: java.lang.reflect.InvocationTargetException
    at
org.springframework.scheduling.quartz.MethodInvokingJobDetailFactoryBean$MethodInvokingJob.executeInternal(Met
odInvokingJobDetailFactoryBean.java:268)
    at org.springframework.scheduling.quartz.QuartzJobBean.execute(QuartzJobBean.java:86)
    at org.quartz.core.JobRunShell.run(JobRunShell.java:205)
    at org.quartz.simpl.SimpleThreadPool$WorkerThread.run(SimpleThreadPool.java:520)
Caused by: java.lang.RuntimeException: java.lang.reflect.InvocationTargetException
    at com.nstc.terraQuartz.scheduling.quartz.AutoJobProxyImpl.execute(AutoJobProxyImpl.java:117)
    at sun.reflect.GeneratedMethodAccessor932.invoke(Unknown Source)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:37)
    at java.lang.reflect.Method.invoke(Method.java:611)
    at org.springframework.util.MethodInvoker.invoke(MethodInvoker.java:276)
    at
org.springframework.scheduling.quartz.MethodInvokingJobDetailFactoryBean$MethodInvokingJob.executeInternal(Met
odInvokingJobDetailFactoryBean.java:260)
    ... 3 more
Caused by: java.lang.reflect.InvocationTargetException
    at sun.reflect.GeneratedMethodAccessor933.invoke(Unknown Source)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:37)
    at java.lang.reflect.Method.invoke(Method.java:611)
    at org.springframework.util.MethodInvoker.invoke(MethodInvoker.java:276)
    at com.nstc.terraQuartz.scheduling.quartz.AutoJobProxyImpl.execute(AutoJobProxyImpl.java:112)
    ... 8 more
Caused by: org.springframework.transaction.CannotCreateTransactionException: Could not open Hibernate Session
for transaction; nested exception is org.hibernate.exception.GenericJDBCException: Cannot open connection
```

日志中典型异常的分析

APMCon

日志文件中的程序异常*	业务异常	可能的影响
证书使用错误：C=CN,O=财务,OU=SHSIIC,CN=黄静1<>null 50004ms already spend.token clear: sid=....	用户登陆失败	• 影响用户的正常登陆和体验 • 导致用户无法完成操作
09:50:41 INFO [ExecuteQuery] Exec[0]: cnapsService.selectCnapsByOpBankName() 09:51:38 WARN [ExecuteQuery] Method execution failed	支付系统CNAPS连接失败	• 无法正常完成支付 • 支付失败影响用户体验
通知预约过期后自动撤销失败 UndoAppointmentExecutor.execute Could not get JDBC Connection	业务操作失败	程序操作失败，系统预约等状态可能不一致，需要人工校正
org.springframework.remoting.RemoteAccessException: Cannot access HTTP invoker remote service at [http://172.28.50.12:7001/EDC/remoting/remoteService]	转账POS系统连接失败	转账失败，账户金额不一致，需要人工对账和确认

针对异常的关联抽取处理模型



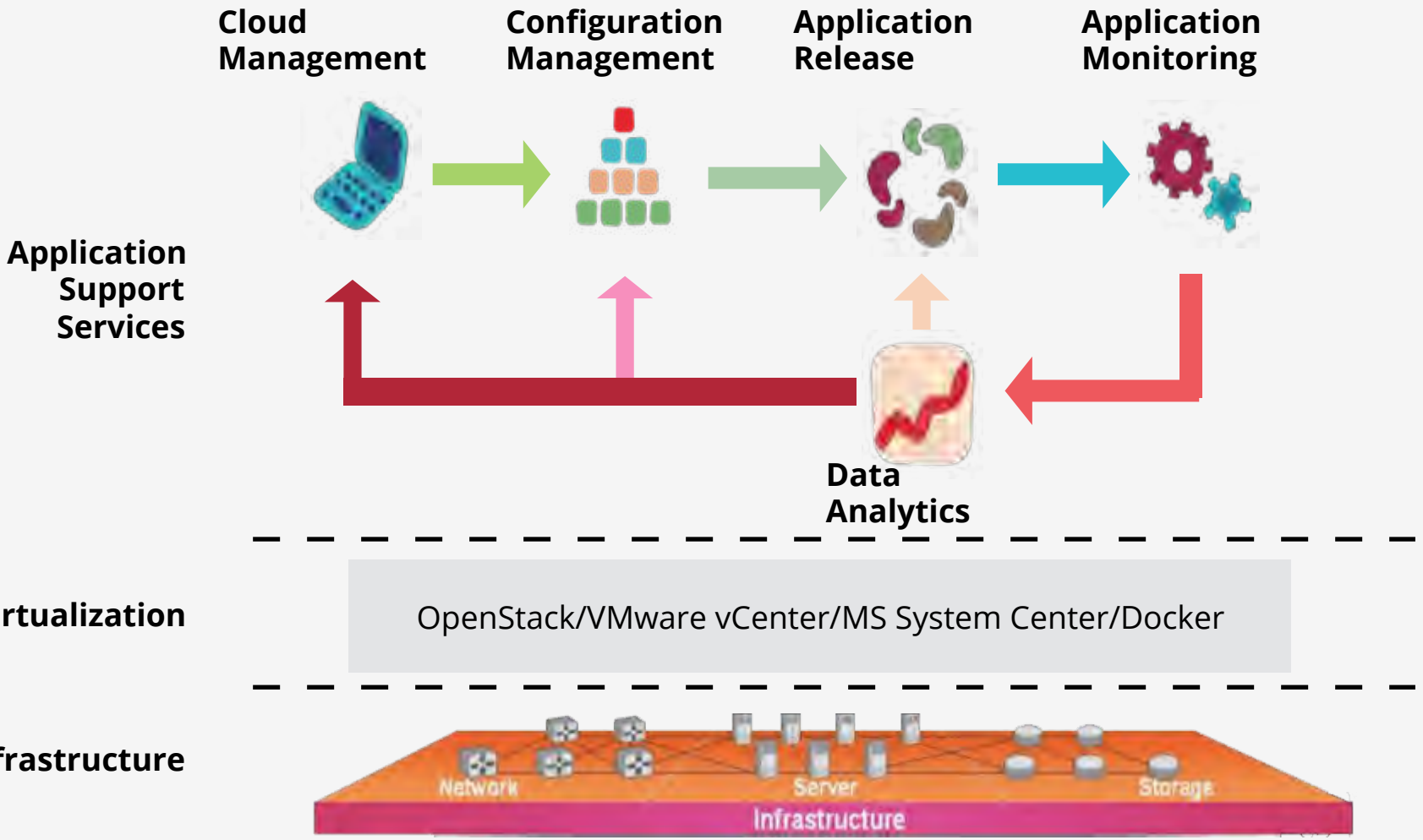
- APM如何形成、表达复杂事务的关联关系，打通“监控、分析和处理”的全流程?

面对复杂事务性能的BPM解决方案

从基础设施到应用的自动化运维

APMCon

DevOps & Continuous Delivery extremely depends on '***Infrastructure as the Code***', which includes the automation capability of virtual server, configuration, release and monitoring.



自动化运维体系建设蓝图

APMCon

PaaS, consists of ***FIVE*** products, builds best practices of DevOps in with FOSS, aims to provide 'Out-of-Box' DevOps services to our clients.

Visualization Collaboration

ChatOps
Slack-Style Team Collaboration & Operation Automation

Platform

CMP
Cross-Cloud, Self-Service

SCP
Puppet-based Configuration Orchestration

ARP
Blue-Green, Canary Release

Analytics
Log Collection, Data Analytics, Visualization

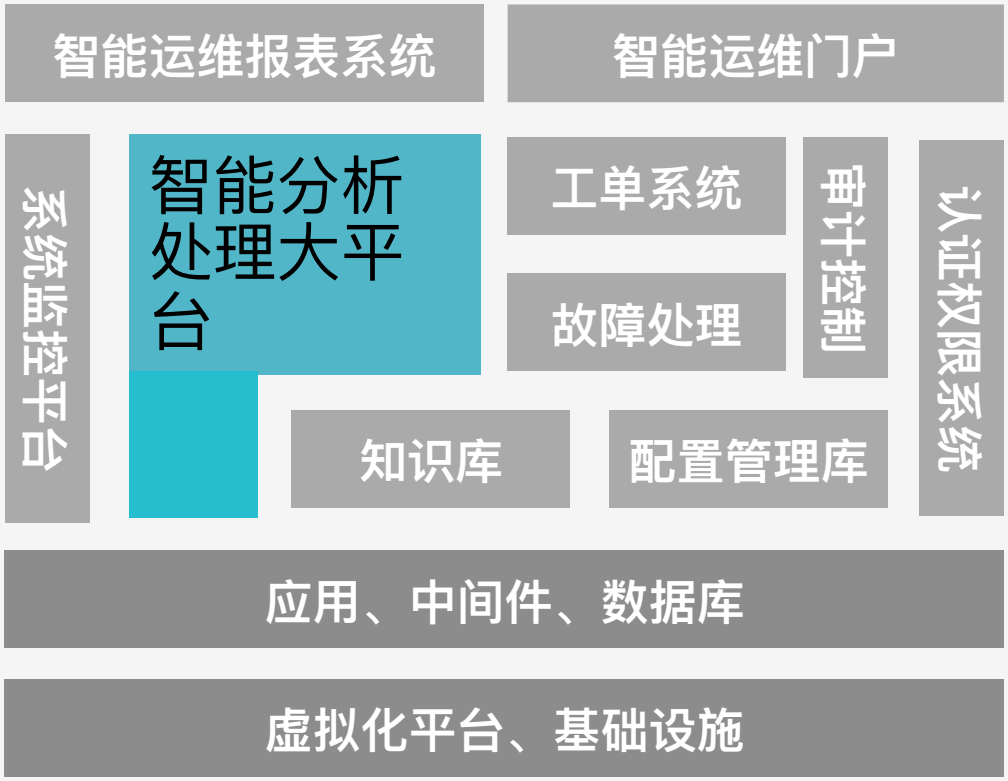
Virtualization

OpenStack/VMware vCenter/MS System Center/Docker

Infrastructure

打造统一的智能分析处理大平台

统一分析处理平台目的在于打通“应用－监控－优化”系统的“孤岛”，形成快速实时的响应闭环，从而帮助企业有效地提升大运维体系的效率。



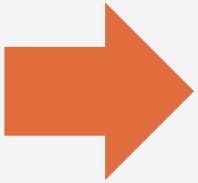
平台功能：

- 实时采集应用、中间件、数据库以及基础设施的信息
- 根据知识规则触发告警，提交给统一监控平台，触发告警处理机制，生成运维工单
- 通过数据网关，向统一智能运维报表系统提供实时、细粒度的运维数据
- 根据认证权限系统、审计规则，实时监控不合规的用户访问等异常

BPM四大核心

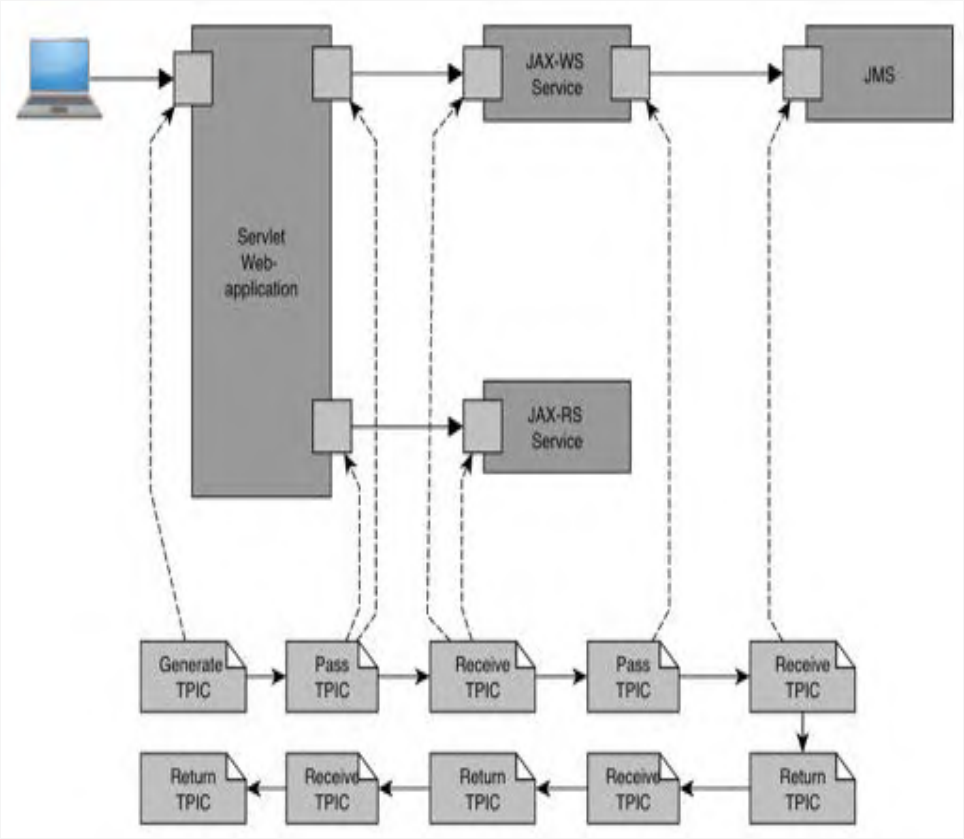
APMCon

- 探针+机器埋点
- 业务+技术专家梳理
- 机器学习
- 自动化处理



- 基于标准协议接口，而非方法粒度的Tracer SDK
- 支持结合业务的“灵活关系模型”，而非“单一关系模型”的规则引擎
- 支持大数据机器学习的数据模型优化
- 整合不同运维对象的API接口，实现自动化

基于中间件、消息总线标准组件协议的SDK



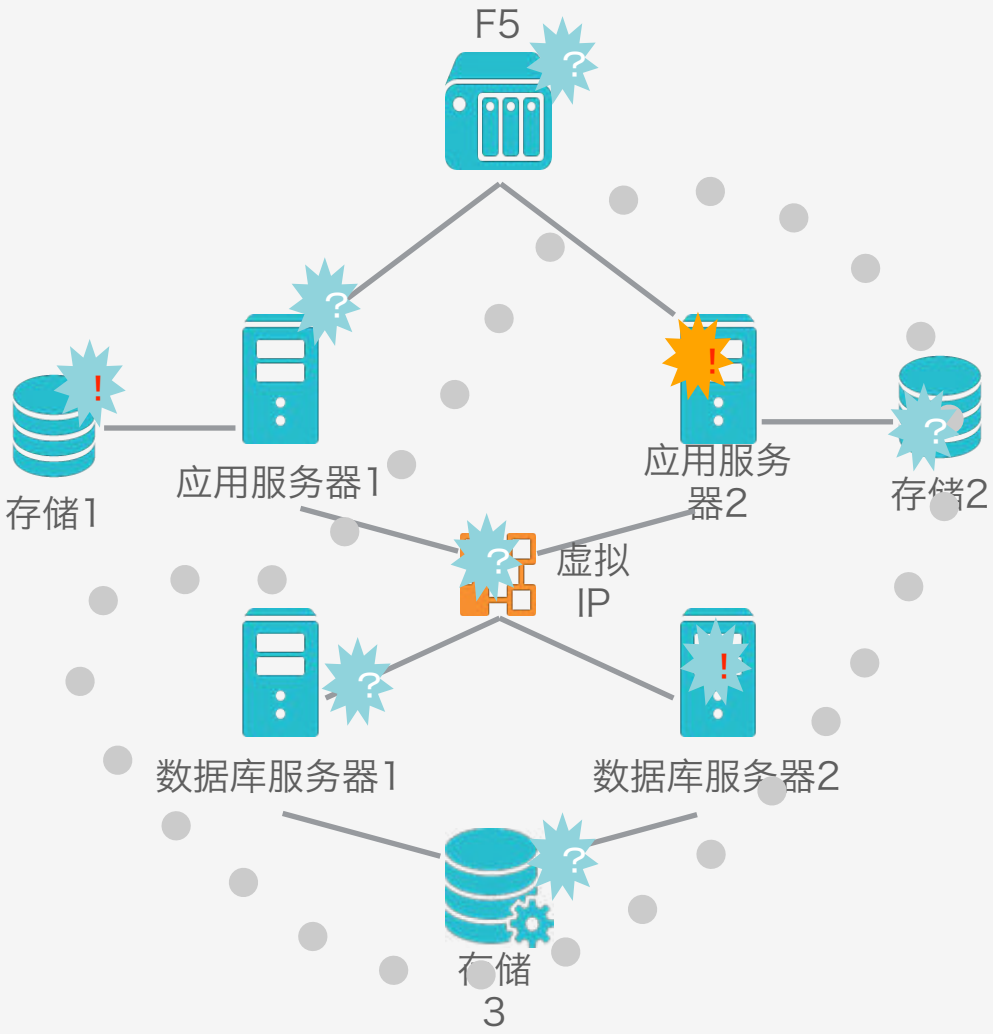
例如常见的JavaEE规范：

* servlet 2.5+
* jax-ws/jax-rs2
* jms

以及一些被广泛使用的开源框架:

* Spring MVC
* Spring Web (RestClients)
* Spring AMQP (RabbitMQ)
* Spring Web Services (SpringWS)
* Apache HttpClient 3 / 4
* Apache CXF
* Quartz Scheduler
* ……

异构系统的数据关联分析



查找“应用服务器2”上的故障根因：

仅凭时间轴关联的关联分析有如下不足：
一方面，各类数据让人目不暇接
另一方面，依靠人工罗列、对比相关组件在实际工作中过于费时费力

BPM分析规则引擎

APMCon

基于Tagging＋Tracing的元数据，实现应用调用关系、基础设施依赖的自动发现（Auto-Discovery）。通过有向图数据库的建模能力，实现数据流规则的设计和定位分析。

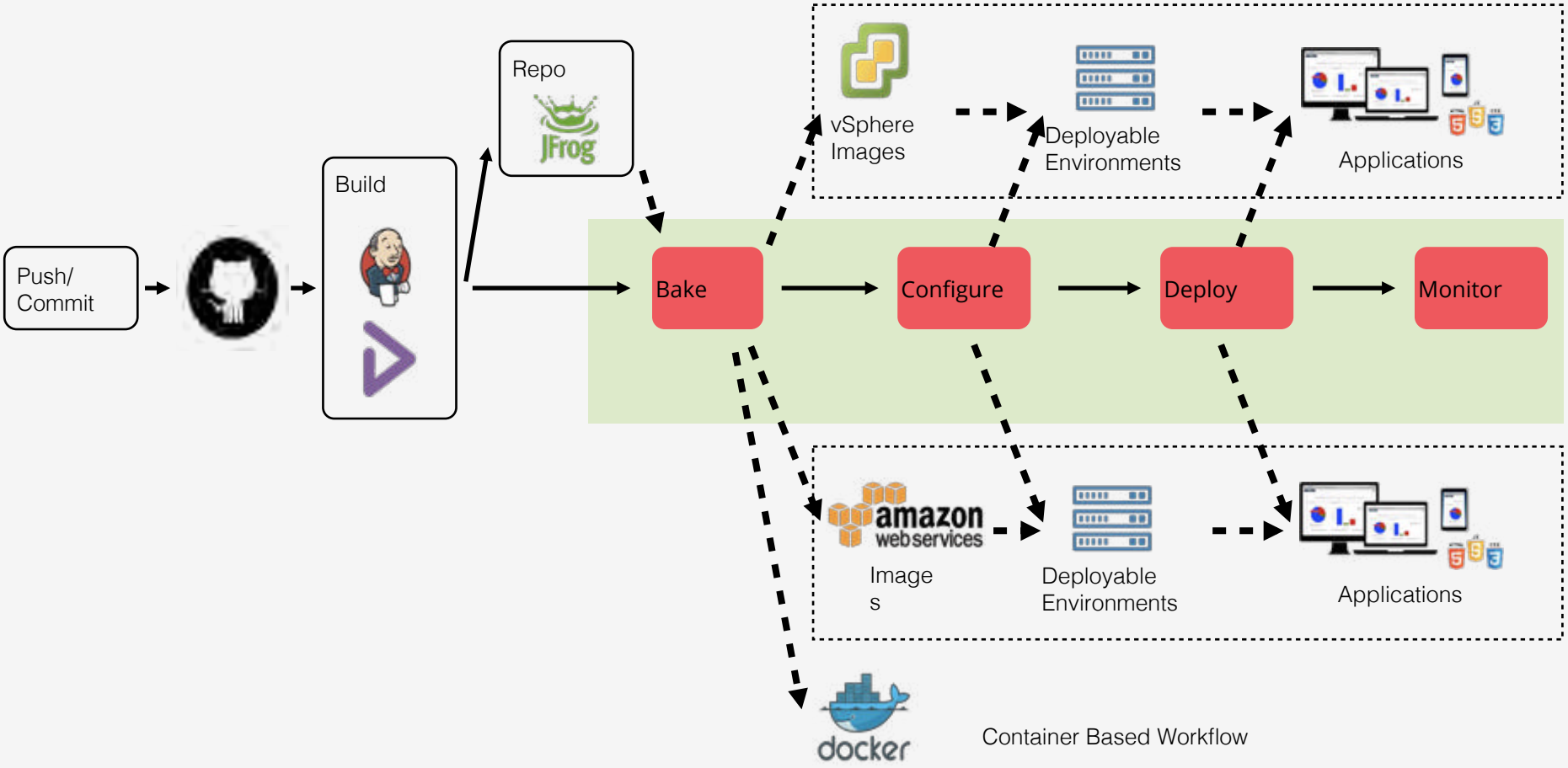
基于机器学习的数据模型优化

当机器数据采集到一定程度，可以使用大数据的机器学习，通过数据的练习，自动发现数据的关联关系，完成异常与根因、技术与业务的自动关联分析。

$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

基于DevOps的持续交付自动化流水线

APMCon



ChatOps：基于机器数据和API的运维自动化与协作

APMCon

“…While in a chat room, team members type commands that the chat bot is configured to execute through custom scripts and plugins. These can range from code deployments to security event responses to team member notifications. The entire team collaborates in real-time as commands are executed.”

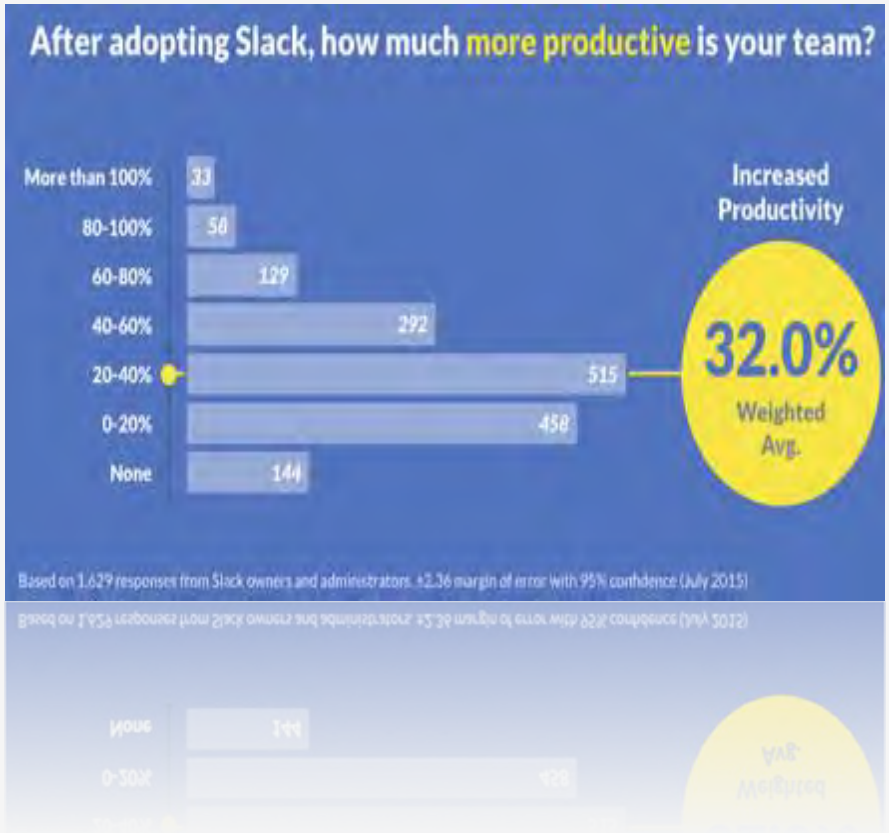
自动监控　　自动变更　　自动构建

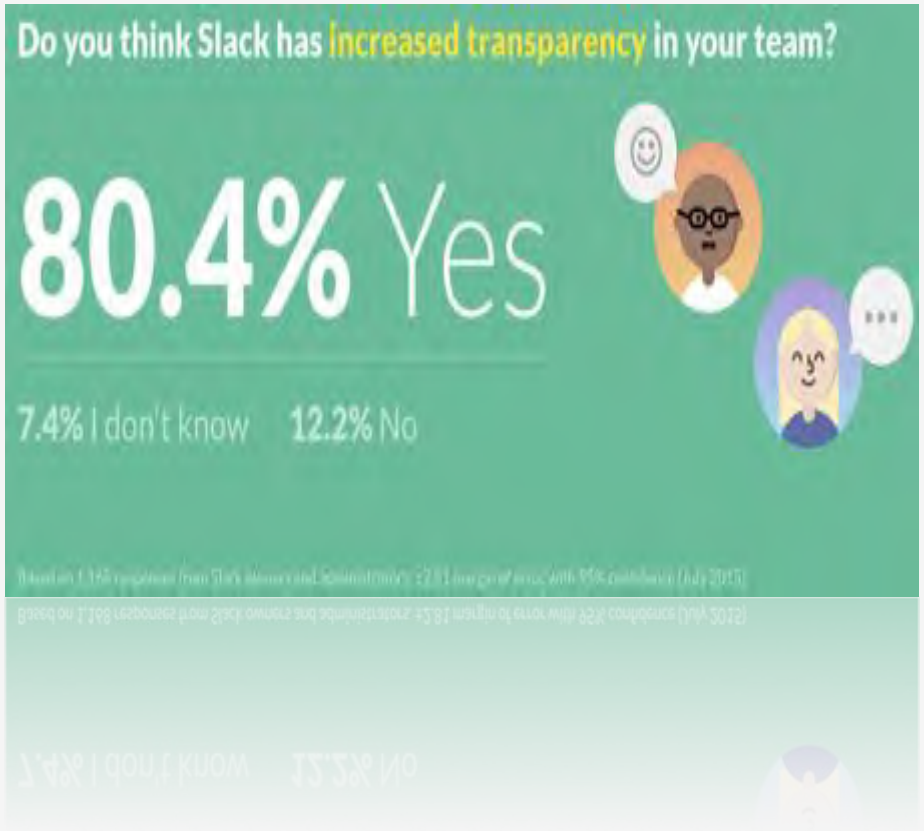
自动通知　　自动审查　　……

讨论组　　＋　　API机器人

打通机器数据与自动化，运维团队的效率得到了很大提升

APMCon





运维团队的生产率平均提升了32%

80.4%的运维团队透明度得到了提高

* https://slack.com/results

BPM实施方法

BPM实施方法ABC

- A：Architecture
- B：Business
- C：Criteria

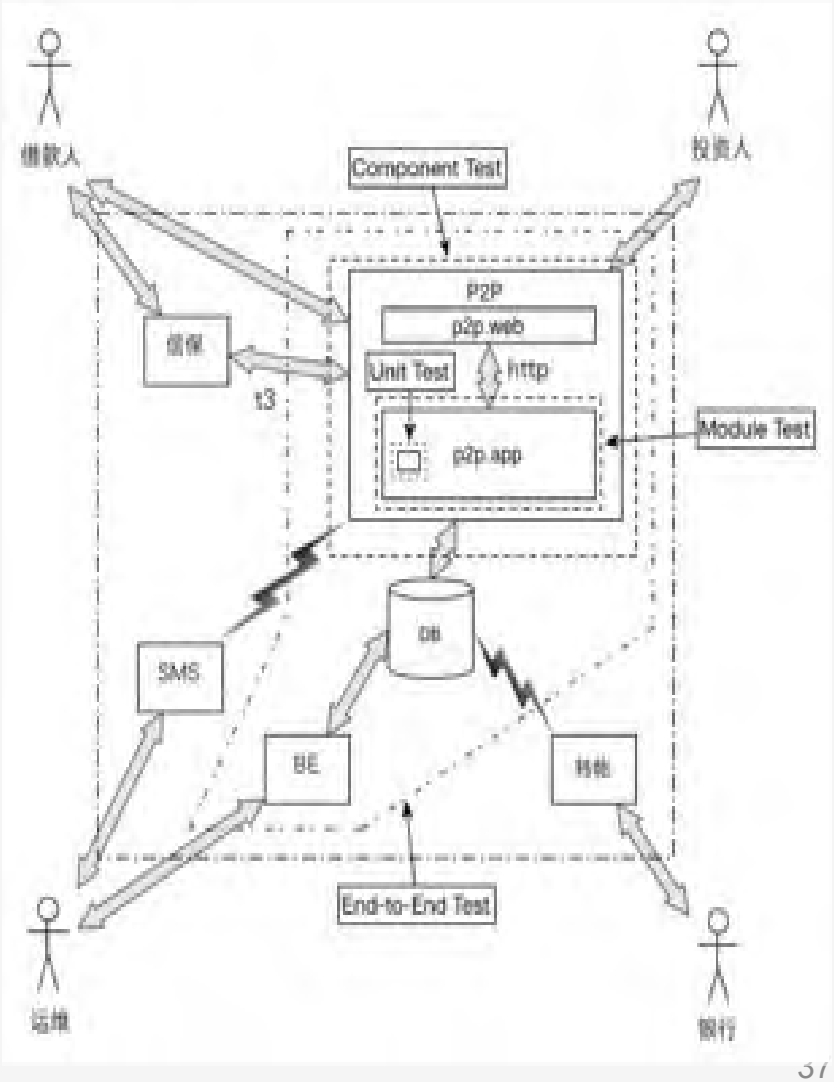
A（Architecture）梳理应用系统端到端的架构

从请求进入服务器后台开始，到服务器将结果返回给用户的端到端架构：

- 架构包含了哪些组件、模块?
- 架构包涵了哪些关键的风险点和集成点?

参考

* 边界类信息：各组件、模块的访问入口和响应出口，如controller、service、dao、SSH等
* 子系统信息：各个子系统的数据输出，如apache、tomcat
* 持久化类信息：消息队列、数据库的发送与响应的消息，如MQ、DB



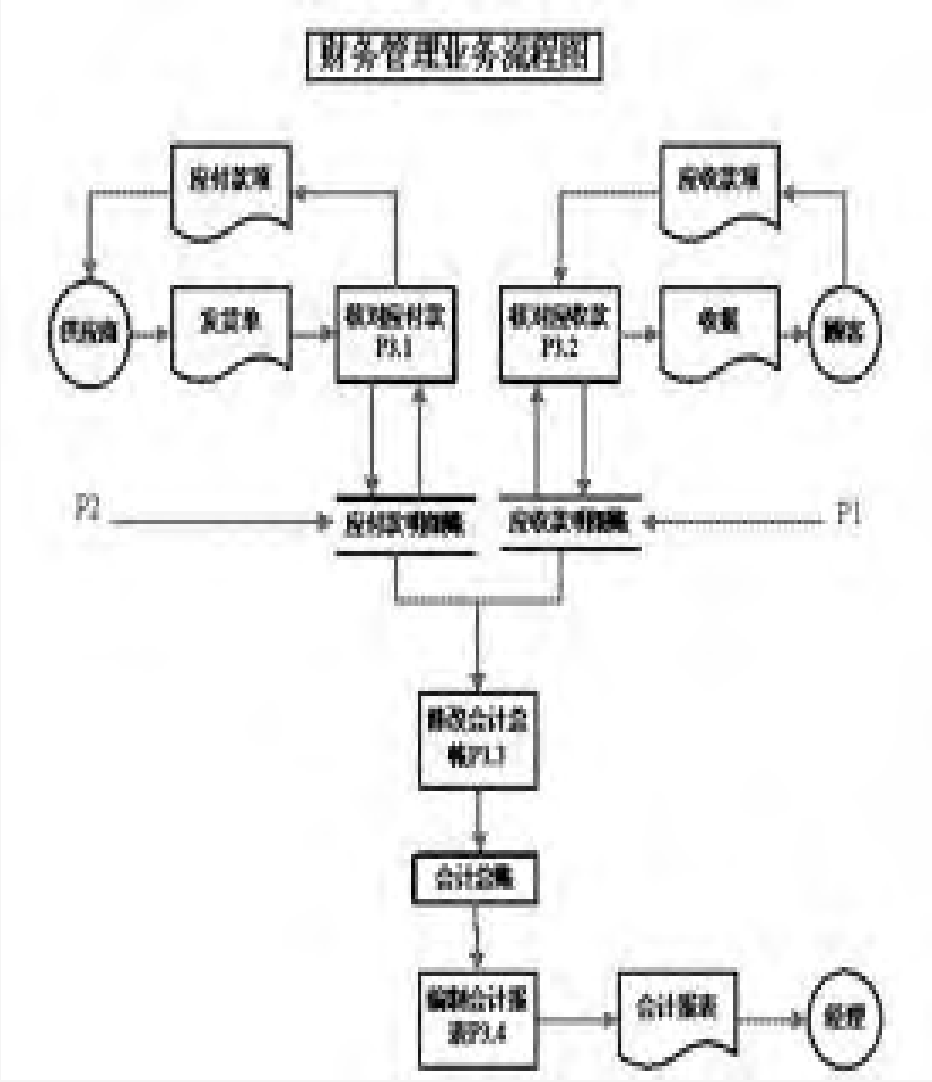
B（Business）整理业务系统的业务场景

纵向“切蛋糕”，切出比较重要的业务场景：

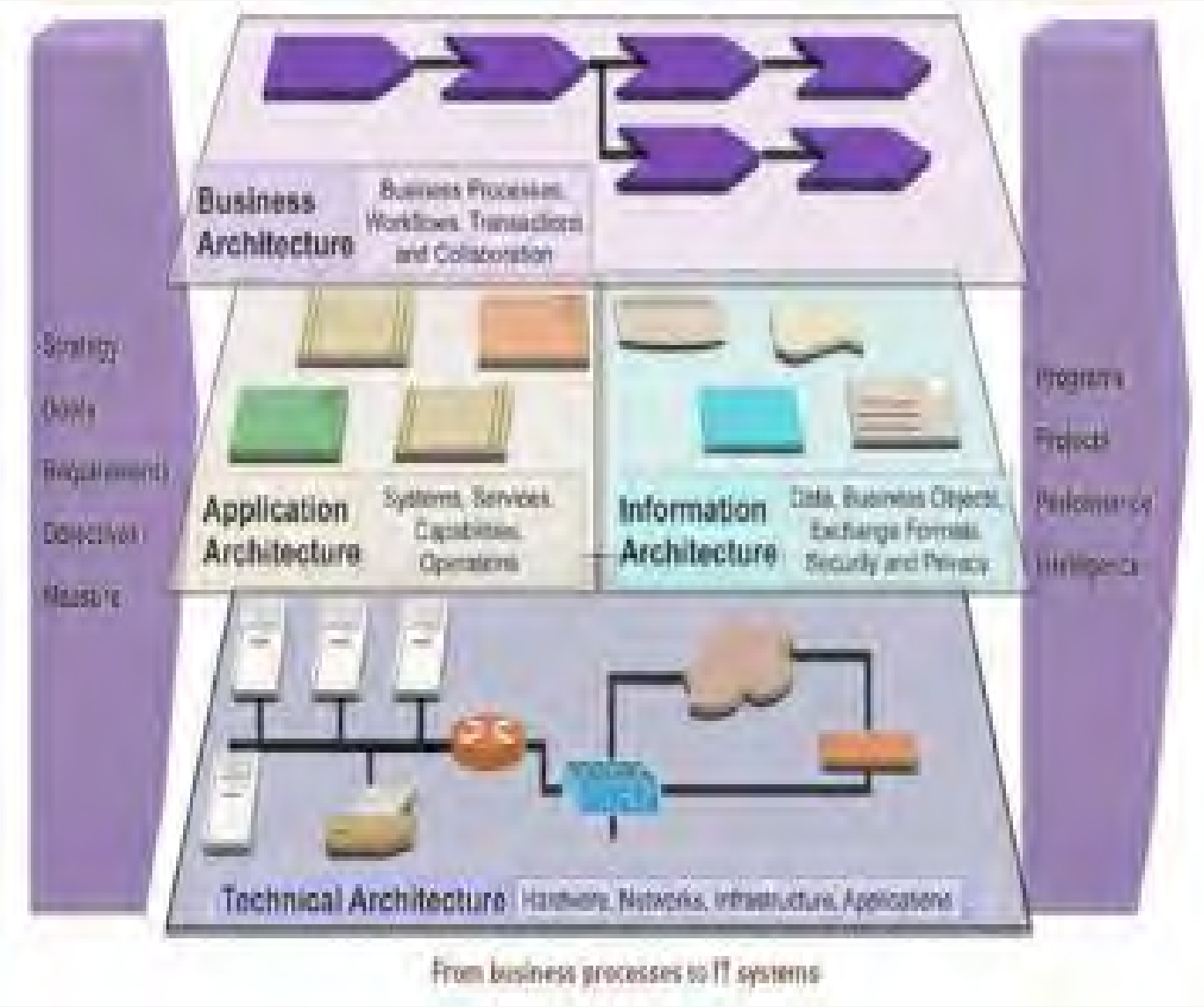
- 业务流程包括哪些业务动作?
- 业务流程有哪些关键的风险点?

参考

* 安全类信息：用户登录/登出或接受授权等
* 业务类信息：导致系统状态变化的操作，例如增加、修改、删除数据
* 性能类信息：对性能有要求的性能数据，如关键方法的执行时间
* 其它重要信息：其它一些重要的运行点



C（Criteria）确立应用的监控指标，整理数据体系



* http://www.managershare.com/wiki/%E4%BC%81%E4%B8%9A%E6%9E%B6%E6%9E%84

BPM真正解决业务-IT对齐的示例

移动应用的BPM指标体系

APMCon

Performance

App Crashes: Average crashes per app loads, typically 1~2%, but it varies.
API Latency: Round-trip time from a request to a response, optimal < 1s.
End-to-End Biz Transaction Latency: End-to-end response time to critical biz user flows.
App Load per period: Number of transactions or calls over a certain period of time

Usability

Battery Usage per period: Battery Usages over a certain period of time
Memory Allocation: Memory usage over a certain period of time
Network Usage: Network usage over a certain period of time
Network Request Frequency: Number of network request over a certain period of time

Engagement

Session Length: time period between app open and close
Session Interval: time between the user's first session and their next one
Retention Rate: percentage of users who return to your app

移动应用异常分析和定位的数据模型

APMCon

属性	描述
id	事件标识
timestamp	事件时间戳
host	事件发生地点
type	归档类型
input_type	采集类型
source	事件采集来源
message	事件明细
level	事件级别
logger	采集组件
thread	执行线程

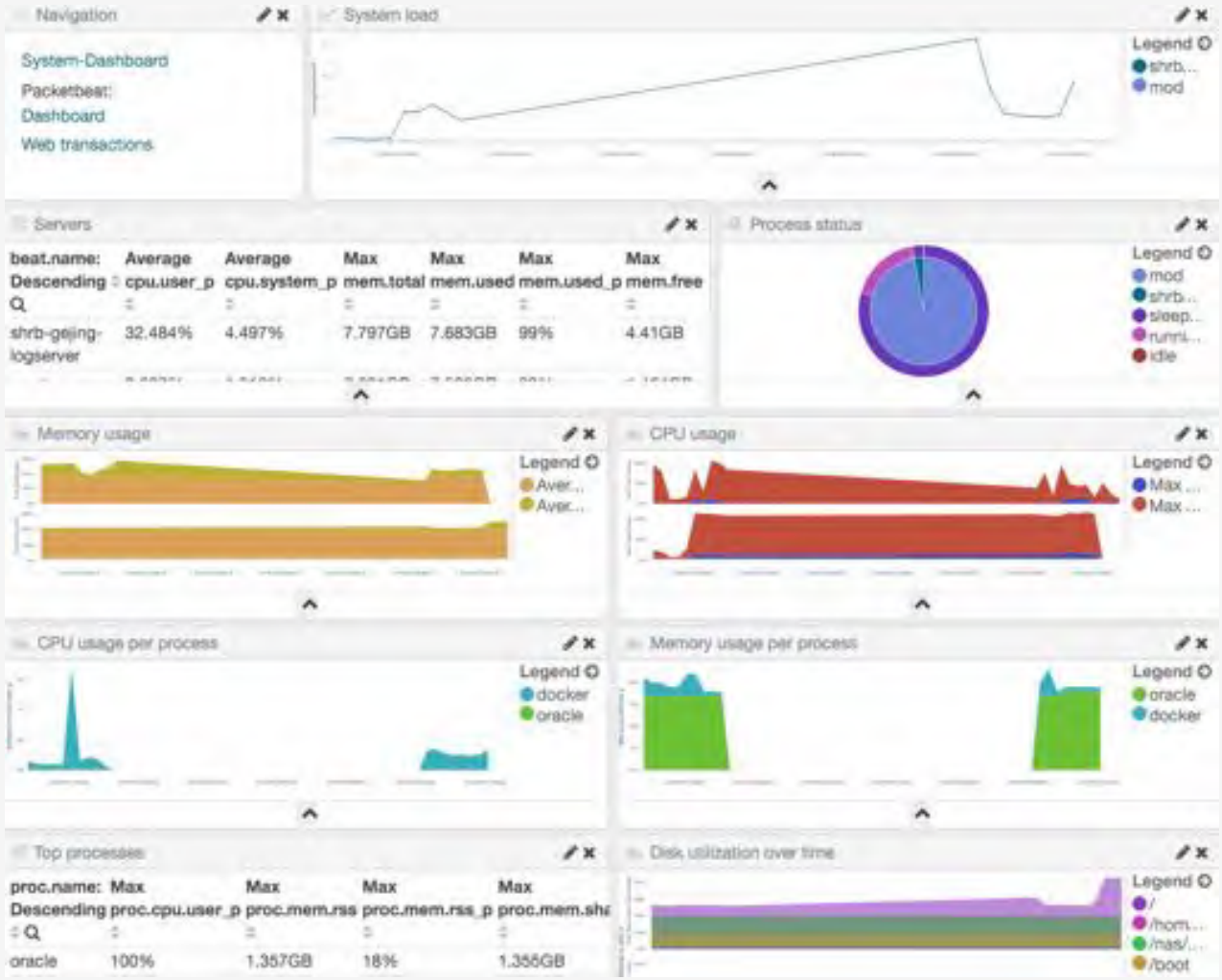
属性	描述
exception_category	异常类别
exception_description	异常描述
exception_severity	异常级别
exception_cause	异常原因
exception_impact	异常影响
exception_action	异常响应方法
TPIC.invocationId	请求标识
TPIC.user	用户标识
TPIC.target	目标类型（订单/产品）
TPIC.targetId	目标标识

移动应用异常分析与定位的规则模型

APMCon

应用性能指标：全面展示异构系统来源的数据

APMCon



- *Battery Usage*
- *Memory, CPU Usage*
- *Network Throughput*
- *Network Request Frequency*

THANK YOU

ThoughtWorks®