

跨国交易平台的风控和反 欺诈技术

内容

- 背景介绍
 - 跨国交易平台
 - 跨国交易存在的风险
 - 风控和反欺诈的现状
 - 我们做了哪些工作
- 风控和反欺诈的技术
 - 业务规则引擎
 - 业务规则的局限性
 - 基于统计学的模型
 - 模型的可扩展性

内容 – Cont'

- **风控系统的平台技术**
 - 服务化架构
 - 服务化进程
 - 高性能服务框架
 - 海量数据
 - 其他的
- **一些感触**
 - 架构师做什么？
 - 程序员的结局
 - 可伸缩性代表什么？

背景介绍

- 跨国交易平台
- 跨国交易存在的风险
- 风控和反欺诈的现状
- 我们做了哪些工作

背景介绍 – 跨国交易平台

- 跨国交易平台 – www.aliexpress.com
- 2009年 4 月公测版
- 2010年 4 月正式版
- 2010年中国本土跨国交易平台第一
 - Alexa 排名 < 1000
- 平台的特点
 - 单个订单一般在 500 美元以下
 - 来自全球 190个国家的买家
 - 买卖双方在线沟通，下单支付一步到位，国际快递全球配送
 - 直接向国外零售终端或者网店供货，拓展利润空间

背景介绍 – 跨国交易存在的风险

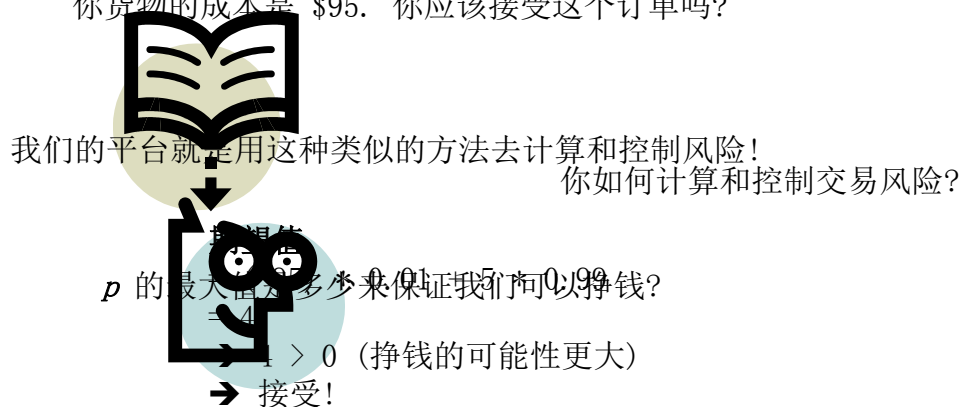
我应该接受这个买卖吗？



我可以买这个货物吗？

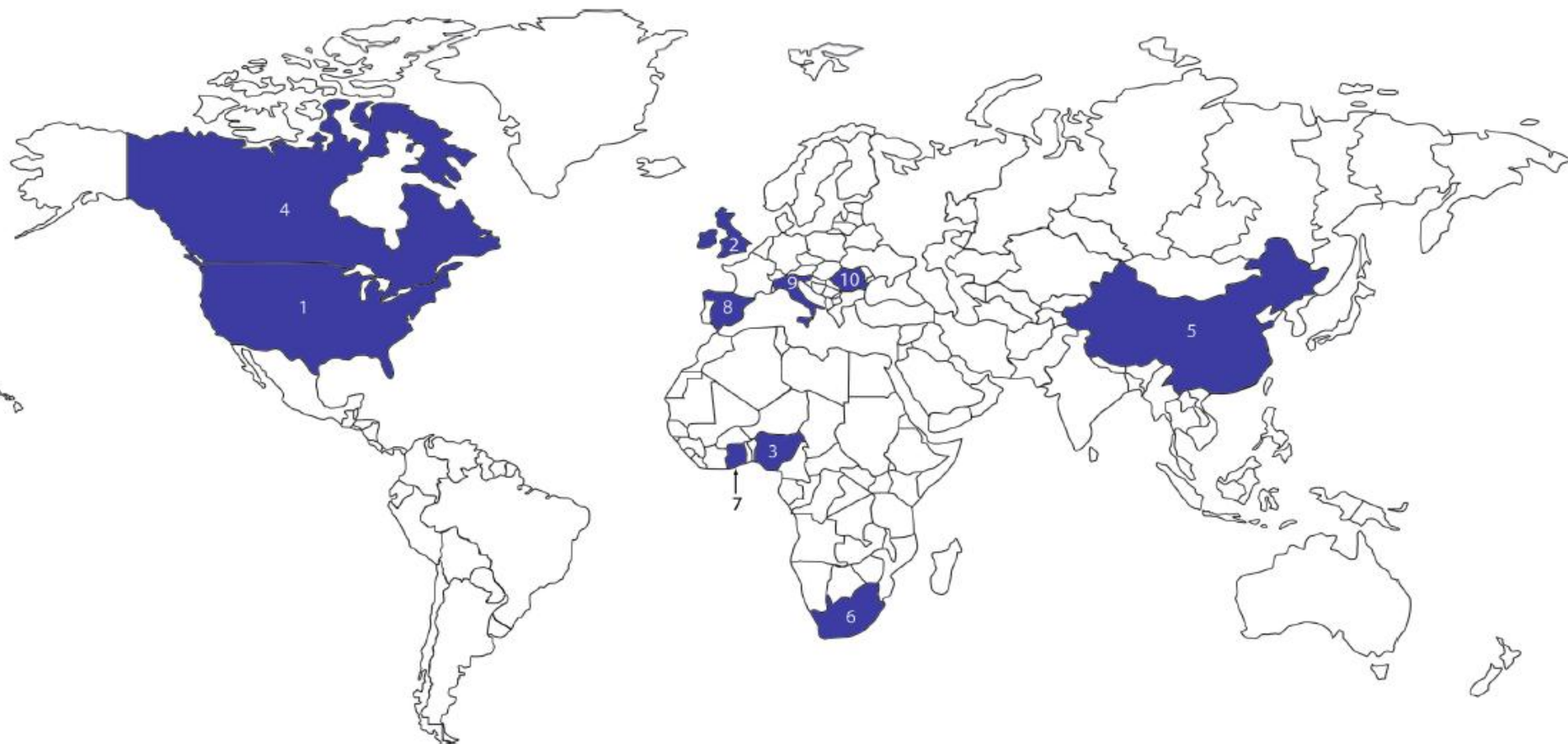


假设一个订单(\$100)是欺诈行为的可能性是 p , 比如说 1%, 你货物的成本是 \$95. 你应该接受这个订单吗？



背景介绍 – 风控和反欺诈的现状

Top Ten Countries By Count: Perpetrators



Map 2 – Top Ten Countries By Count (Perpetrators)

1. United States	66.1%	6. South Africa	0.7%
2. United Kingdom	10.5%	7. Ghana	0.6%
3. Nigeria	7.5%	8. Spain	0.6%
4. Canada	3.1%	9. Italy	0.5%
5. China	1.6%	10. Romania	0.5%

背景介绍 – 风控和反欺诈的现状

Top Ten States by Count: Individual Perpetrators



Map 1 - Top Ten States (Perpetrators)

1. California	15.8%	6. Washington	3.9%
2. New York	9.5%	7. Illinois	3.3%
3. Florida	9.4%	8. Georgia	3.1%
4. Texas	6.4%	9. New Jersey	2.8%
5. D.C.	5.2%	10. Arizona	2.6%

背景介绍 – 风控和反欺诈的现状

- 欺诈造成的损失中只有 50% 申请了退款
- 年交易额 > \$100M 的网商平均会使用大约 7.7 个风控和反欺诈的工具
- 盗卡是网上交易欺诈主要的方式之一
- 在美国近 3 年的网上欺诈总额达到 \$45 Billion
- PayPal 对外 “宣称” 0.5% 的欺诈率

背景介绍 – 风控和反欺诈的经典案例

- **国际欺诈组织在保加利亚被逮捕: 2006年1月23日 – 布加斯, 保加利亚.** 一个由八个年轻人组成的网络犯罪团伙被逮捕, 因为盗卡并且成功购买超过 5 万美元的产品. 他们先用钓鱼网页获取 eBay 和 PayPal 的用户信用卡和个人信息, 然后用盗卡信息购买贵重的衣服, 电器产品和软件
- **礼物卡诈骗: 2006年1月21日 – 艾普顿. Kelly 和 Steven Groat 在 eBay 上售卖礼物卡但是确不发送给买家. 用户总共被骗 \$78,000. Steven Groat 和 Kelly Groat 都受到相应法律制裁**
- **2006年3月: 通过 eBay 和 PayPal 的风控调查, 罗马尼亚警方逮捕了 11个涉嫌在 eBay 上发布产品并且要求客户直接通过 Western Union 转现钱给自己的诈骗嫌疑人**

背景介绍 – 风控和反欺诈 – 我们的案例

- 2009年12月22日，我们的风控服务在处理客户海外信用卡支付申请时发现一笔可疑交易，由海外客户Alex提交，金额超过\$1000，这张VISA信用卡支付 IP 所在地显示是在香港，而该信用卡的发卡地为美国加州，而且前一天刚刚在美国境内进行过网上支付，以前历史交易的平均金额在\$100左右
- 风控系统将这次支付划入线下人工审核案例：
 - 客服人员通过客户提供的资料与其联系，但始终得不到Alex的回复，通过以前数据库中此卡联系人留下的联系方式与客人取得联系，方得知持卡人并不认识 Alex，且信用卡资料未告知过他人，卡主领取信用卡后未更改密码，平时只用于网上支付，使用的银行初始密码
 - 风控部人工分析后认为，持卡人的账户卡号有很多种途径可以被犯罪分子取得，而且其密码过于简单，很容易被人猜中，然后被人在网上进行恶意支付，因为发现得及时，该交易被止付，资金被追回，从而保护了持卡人与商户的利益

背景介绍 – 我们做了哪些工作

■ 自建风控和反欺诈系统

自建风控基础技术和可扩展架构

效果不输于世界领先的风控解决方案

专业的风控运营和服务团队

风控和反欺诈的技术

- **风控和反欺诈的技术**
 - **风控平台概览**
 - **业务规则引擎**
 - **业务规则的局限性**
 - **基于统计学的模型**
 - **模型的可扩展性**

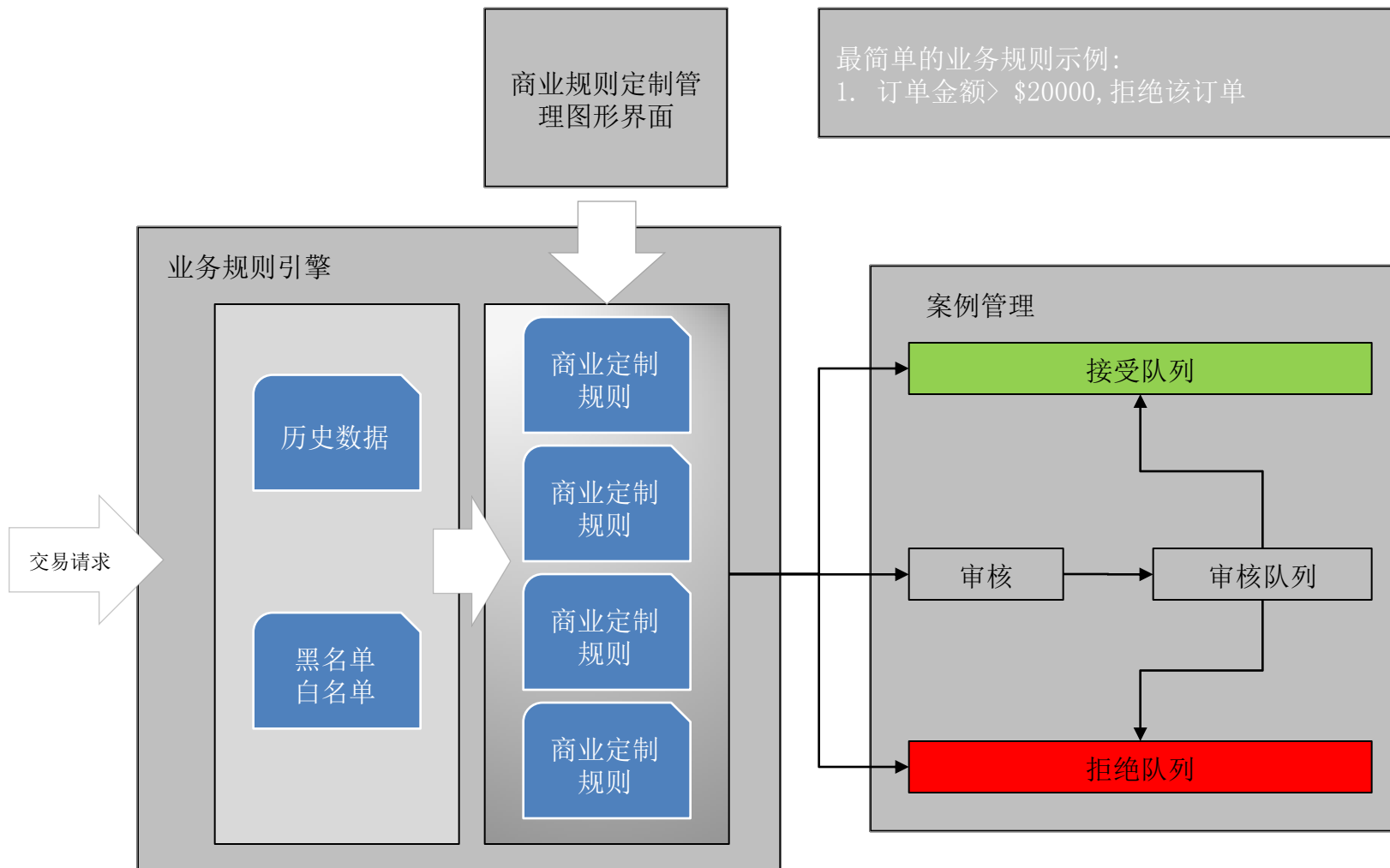
风控平台概览

- 交易网站风险控制的平台
- 7 *24 提供事前、事中和事后的风险分析
- 使用风控平台的角色
 - 风控运营人员
 - 风控业务分析人员
 - 风控模型研究员
 - 风控系统管理员

业务规则引擎

商业规则定制管理图形界面

最简单的业务规则示例：
1. 订单金额 > \$20000, 拒绝该订单



业务规则的缺陷

- **业务规则的缺陷**
 - 数以千计的业务规则
 - 互相冲突、难于维护
 - 实时计算、性能不高
 - 可复用性差

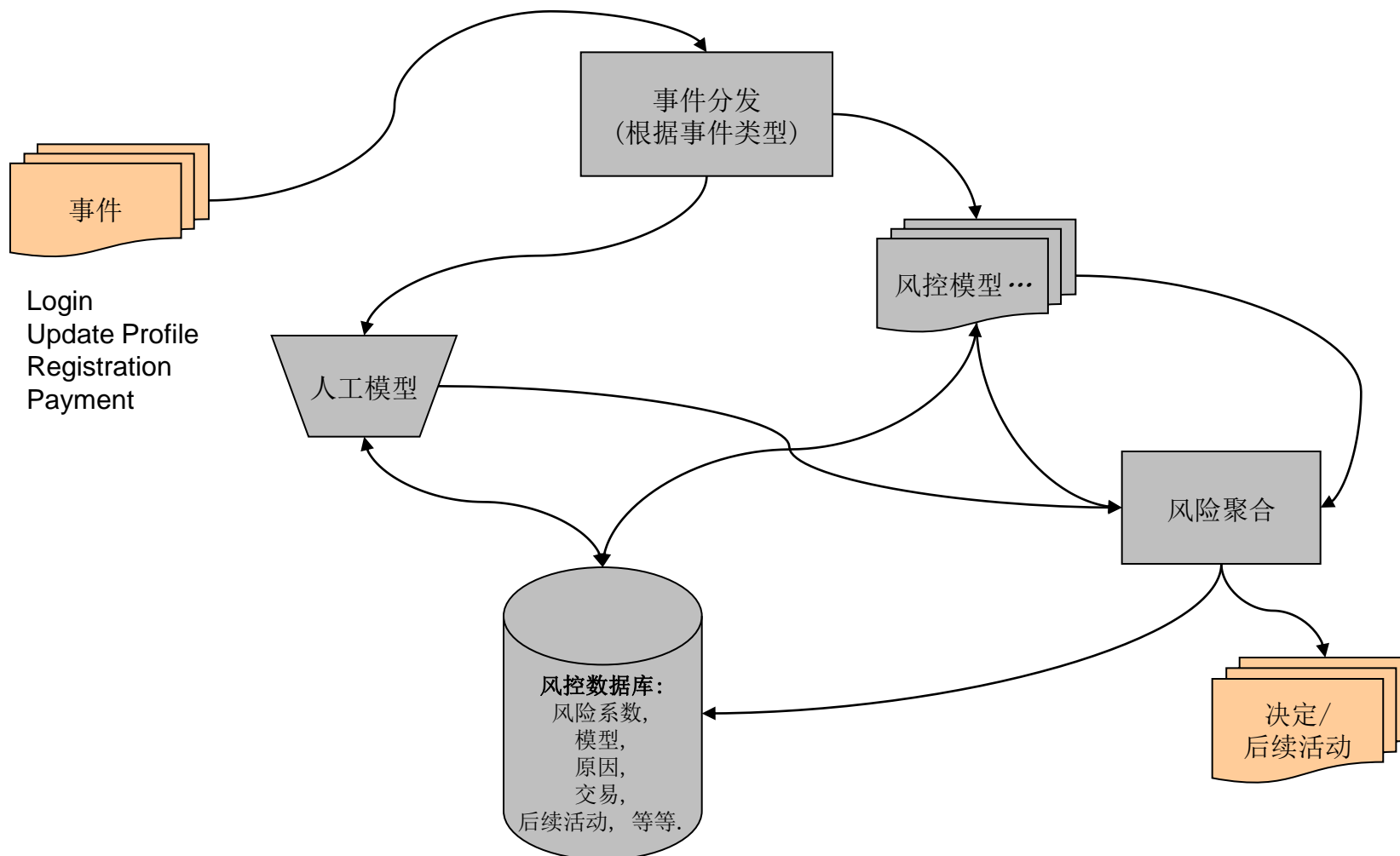
业务规则的缺陷



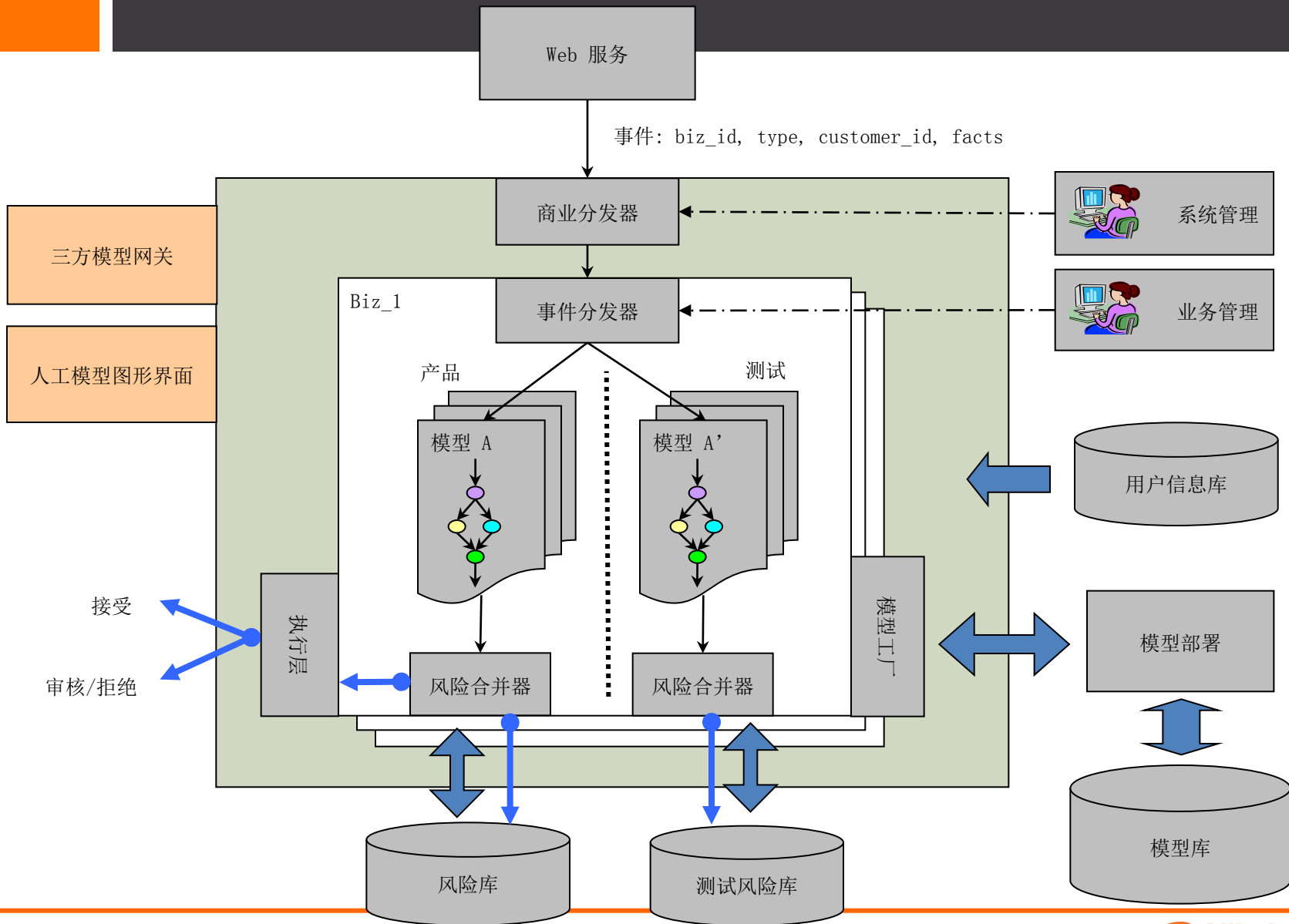
基于统计学的模型

- **统计学模型的优点:**
 - **不会互相冲突**
 - **风险识别精确高**
 - **一定的预测性**
 - **性能比较好 – 之前有训练**

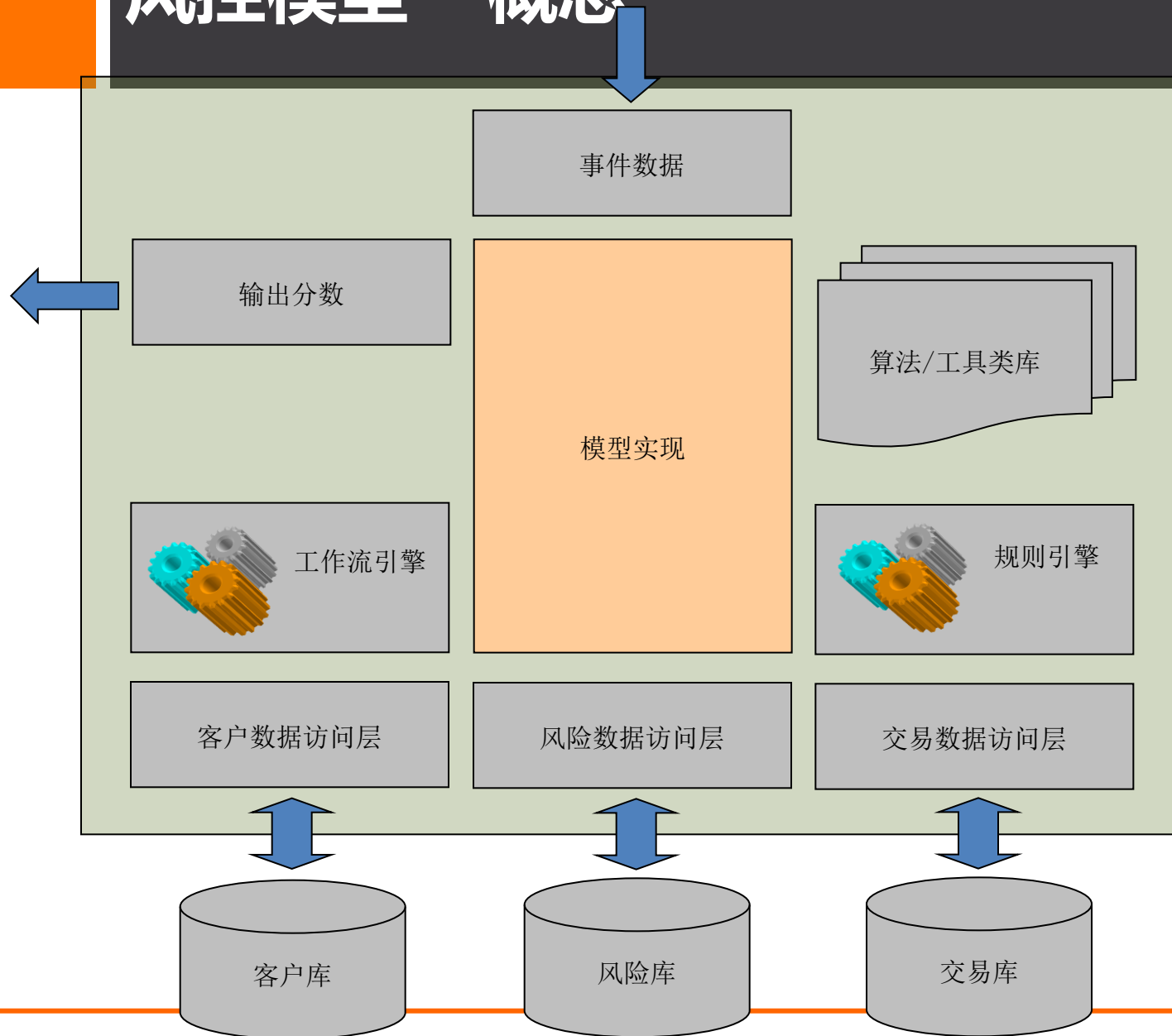
基于统计学的模型



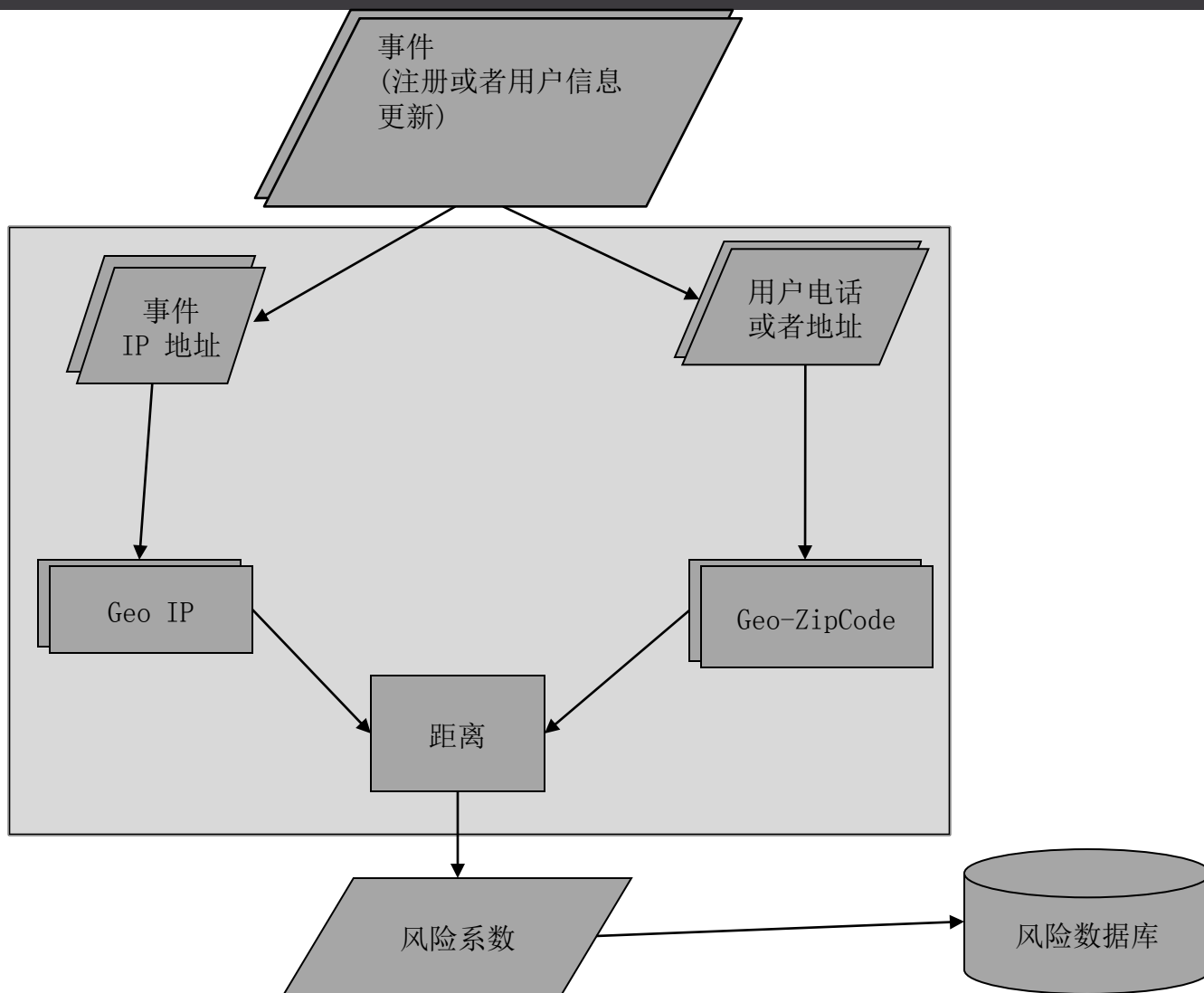
基于统计学的模型



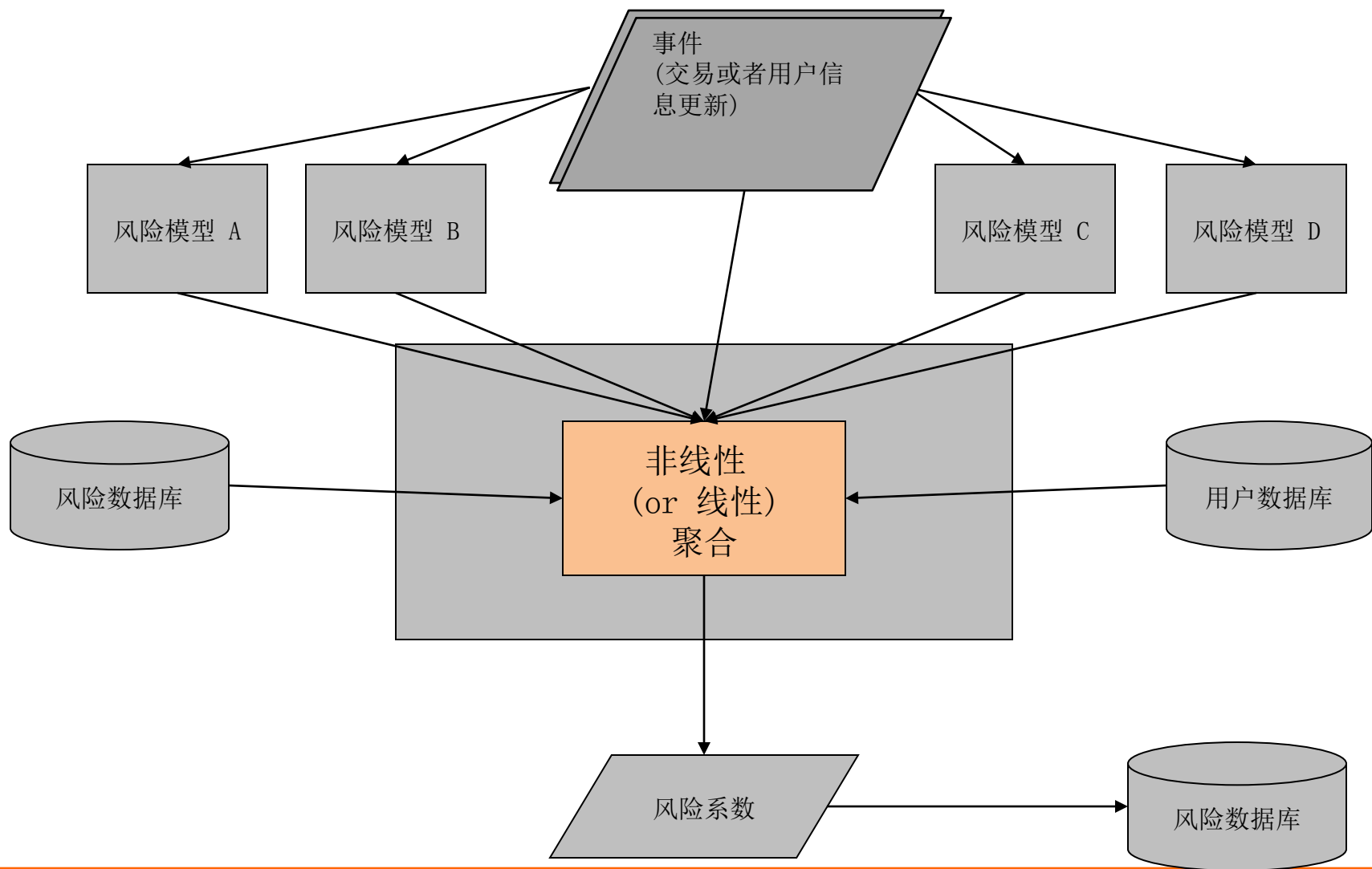
风控模型 - 概念



风控模型 – 模型样例



风控模型 – 模型的扩展性



风控系统面临的技术挑战

- **风控系统面临的技术挑战:**
 - **服务化，包括服务治理和性能**
 - **大数据量的实时处理**
 - **统计学模型的离线训练和在线处理**
 - **可灵活配置的模型引擎**
 - **统计学模型的精度**

风控的技术架构

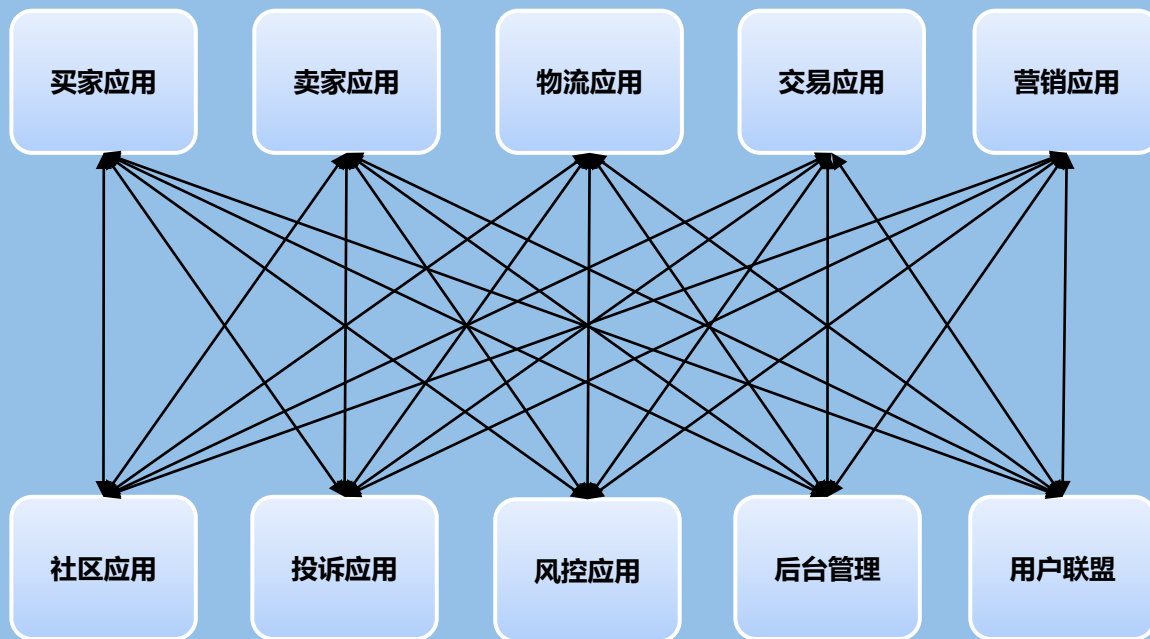
- 我们做过的一些技术工作
 - 服务化推进
 - 海量数据处理
 - 统计学模型训练和调优

技术架构 – 服务化

- **服务化进程**
 - **服务化之前 – 2009 年**
 - 服务的概念不强
 - 应用之间耦合度高
 - 应用功能划分不明确
 - **服务化进程 – 2010 ~ 2011 年**
 - 高性能服务框架
 - 服务依赖治理
 - 服务多版本管理
 - 服务高内聚低耦合

技术架构 – 服务化之前

网站应用



依赖混乱
网状、循环

基础服务

搜索服务

缓存服务

分布式文件存储

分布式数据管理

消息中心

监控中心

技术架构 – 服务化之后

网站应用

产品应用

交易应用

支付应用

物流应用

类目应用

社区应用

营销应用

投诉应用

后台管理

用户联盟

开放平台

开发者
社区

信任登陆

API

会员和信用 服务

核心业务服务

交易服务

物流服务

风控服务

支付服务

营销服务

产品服务

基础服务

搜索服务

缓存服务

高性能服务框架

分布式文件存储

分布式数据管理

消息中心

监控中心

技术架构 – 服务化

■ 服务化

- 服务化的基础 – 高性能服务框架
- 服务治理
 - 服务注册和管理
 - 服务版本管理
 - 监控、容错、异常处理
- 交易平台中有多个服务 (比如: 用户、产品、交易、支付、投诉、后台)、服务之间经常交互
 - 场景1: 交易系统通知支付系统付款
 - 场景2: 物流系统通知支付系统付款
 - 场景3: 风控系统发现产品信息侵权, 产品下架

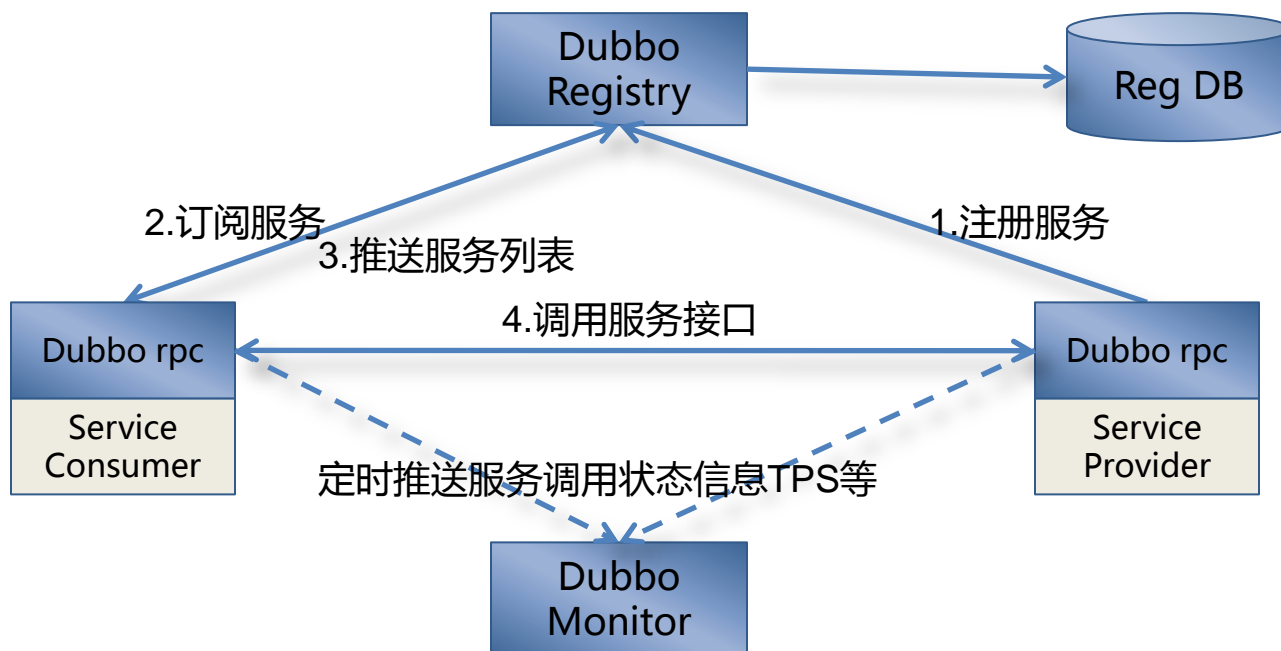
技术架构 – 服务化的好处

- 应用之间的耦合降低，拓展新业务变得容易，更多时候成为服务的一个组合
- 开发人员更加关注业务，不需要考虑一些低层技术细节，比如：服务之间的通讯协议，服务的调用性能，服务依赖的管理和监控

技术架构 – 高性能服务框架

■ 高性能服务框架 Dubbo

- 场景: 服务交互、服务治理、服务注册、订阅和推送



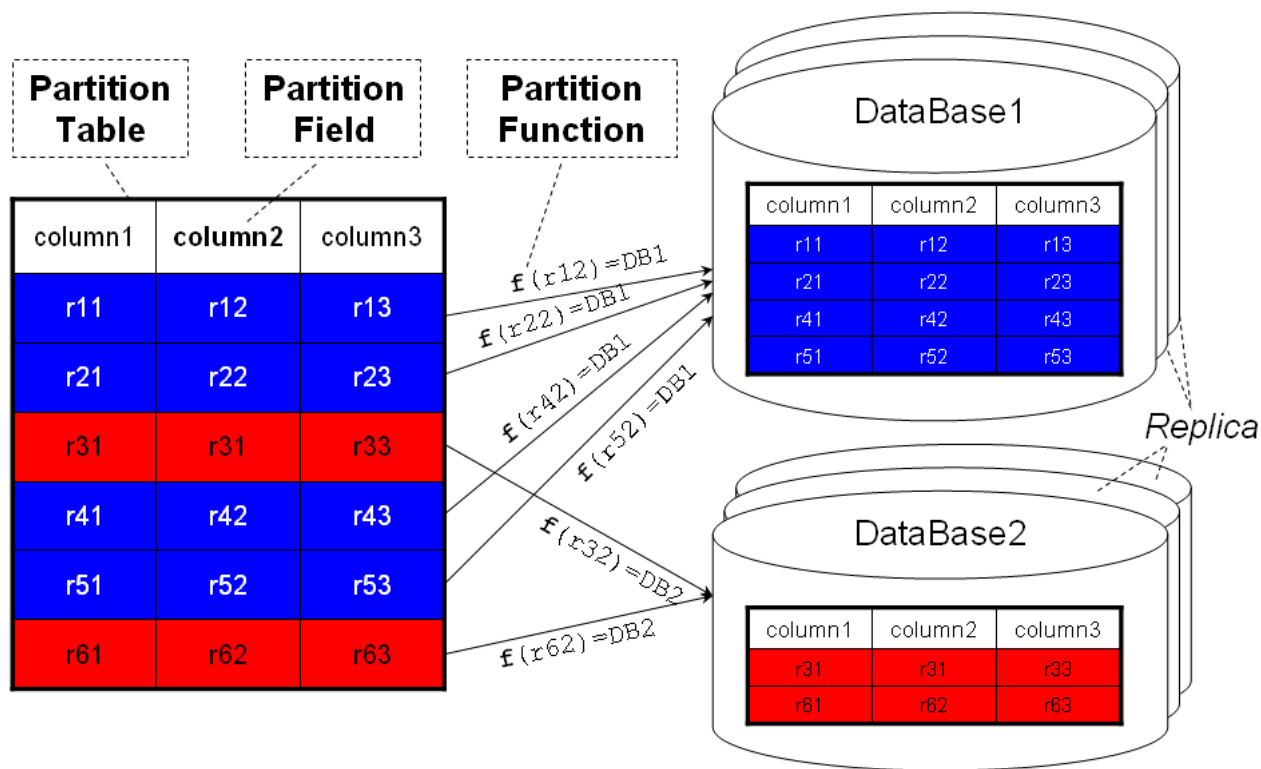
技术架构 – 高性能服务框架

- 1 天 > 1亿次调用

技术架构 – 海量数据

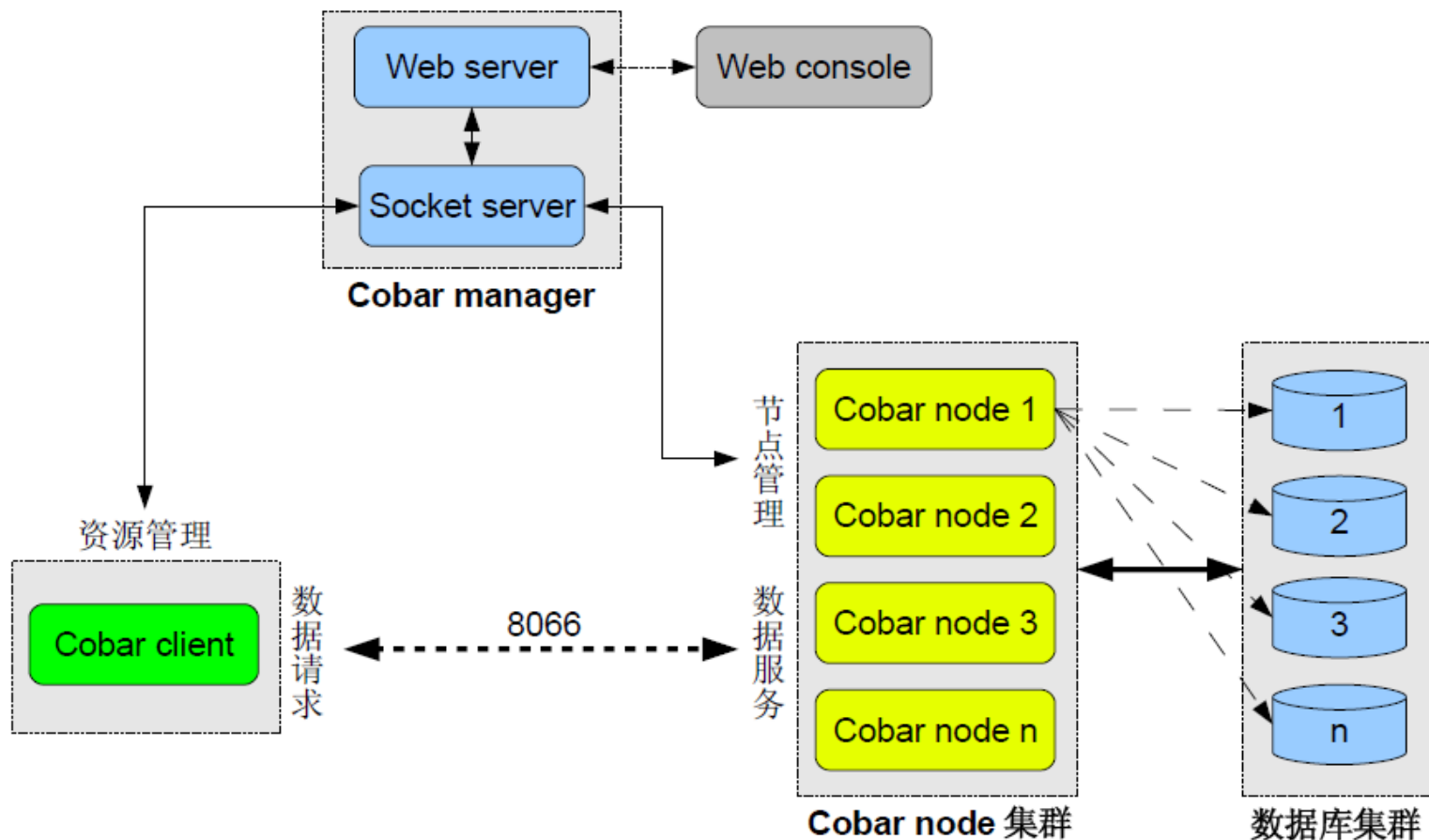
■ 海量数据 – 水平拆分

- 场景: 用户行为数据、黑白名单数据、产品数据



技术架构 – 海量数据

■ 海量数据 – 水平拆分 – 逻辑部署



技术架构 – 其他的风控技术

- 机器学习、分类器
 - C5
 - Bayes
 - 神经网络
- 机器指纹识别 – 精确识别网络设备
-

一些轻松的事情

- 架构师做什么?
- 程序员的发展
- 可伸缩性代表什么?

架构师做什么

■ 架构师 A

- 经常给方向指导
- 不写代码
- PPT 写得挺好
- 架构设计原理很了解
- 知道如何应用这些原理
- 不会有线上故障
- 一天到晚开会
- 有创造性思维但少付出实施

■ 架构师 B

- 偶尔给些建议
- 到处都是他的代码
- PPT 写得很差
- 经常不测试直接放上产品环境
- 有时候导致一些故障
- 热心帮工程师解决问题
- 会议很多但经常不参加
- 想到了就干哪怕就一个人

- 10 年后你会成为？

程序员的发展

■ 民工?



程序员的发展

■ 大师?

MARTIN FOWLER



Intro Bliki Design Agile Refactoring DSL Delivery About Me ThoughtWorks  



photo: Adewale Oshineye

I am an author, speaker... essentially a loud-mouthed pundit on the topic of software development. I've been working in the software industry since the mid-80's where I got into the then-new world of object-oriented software. I spent much of the 90's as a consultant and trainer helping people develop object-oriented systems, with a focus on enterprise applications. In 2000 I joined ThoughtWorks.

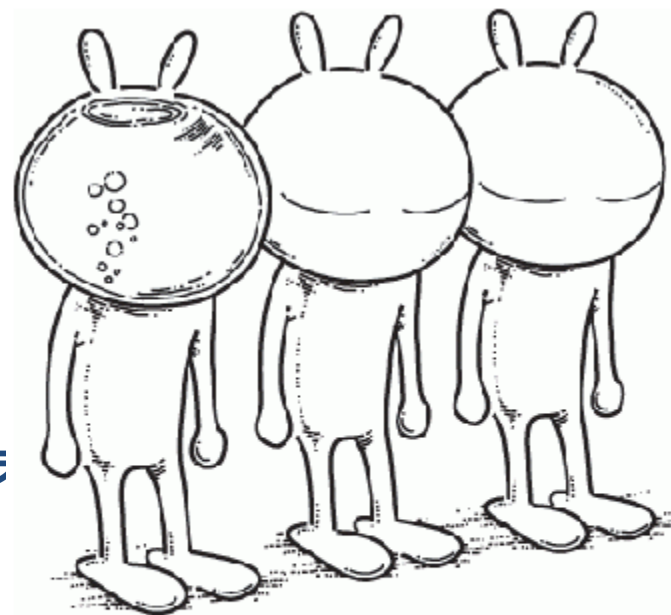
My main interest is to understand how to design software systems, so as to maximize the productivity of development teams. In doing this I've looked to understand the patterns of good software design, and also the processes that support software design. I've become a big fan of agile approaches and the resulting focus on evolutionary software design.

程序员的发展

- 成为大师的途径:
 - 坚持在第一线
 - 从问题出发，着眼解决问题
 - 持续学习，提高技能包括理论水平
 - 创造性工作

可伸缩性代表什么

- 什么是可伸缩性?
- 什么情况下系统的可伸缩性有问题了
 - 平时正常，高并发下性能有问题
 - 不能通过加资源来提高性能
- 达到可伸缩性
 - 负载均衡 – 分布式负载均衡
 - 应用层 – 分布式缓存，服务器集群，异步
 - 数据库 – 分布式数据库
 - 存储层 – 分布式文件系统
 - 每一层都要搞



真正的悲伤是不能以天来计算的
而是浸透在每时每刻里的

可伸缩性代表什么

- 所以建议:
 - 不要为了可伸缩而可伸缩
 - 着重解决问题，平衡投入和产出
 - 好的架构不是规划出来的，而且演化出来的

Q& A



Email: kevin.jiangt@alibaba-inc.com