**SOA立项白皮书**

【立项背景】

1.系统缺乏整体规划和设计，导致代码结构不清晰，到处充斥过期或无效代码，系统维护成本居高不下

2.重复造车轮，系统功能无法有效的复用，浪费了大量的业务及技术资源。

3.系统核心流程高耦合低内聚，系统随时处在牵一发而动全身的不利局面

4.系统架构伸缩性不足，无法有效的利用好硬件资源

5.对业务的支持力度有限，难以满足业务复杂性和多样性的需求

# SOA架构设计

## 架构演进路线

### 任务描述

架构升级版本迭代路线，对比出每次升级的内容，对比旧版本的差异性，解决问题或者改进点。

### 解决问题

1. 清晰架构发展路线。
2. 根据实际资源和成本，采用不同版本的架构。
3. 解决当前业务下的架构问题。

### 对比优势

1. 对未来的规划清晰。成本资源可控。
2. 可以提前预演不同业务场景下的架构设计，及时解决当前问题。
3. 可以动态选择不同架构来支撑业务体系。

### 实现业务指标

1. 成本可控。
2. 资源最大化规划使用。
3. 风险预判。

### 资料数据存档

http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=21532981

## SOA现有架构

### 任务描述

基于环球易购多站点业务背景下的SOA分布式架构第一版本。

### 解决问题

1. 分布式多节点，解决单点故障问题。
2. 解决高峰低谷压力下的扩容和缩容的问题。
3. 解决多站点增量接入成本高的问题。
4. 解决多站点背景下风险监控预判等问题。
5. 解决多站点差异性业务迭代冗余繁杂问题。

### 对比优势

1. 动态简单的扩容和缩容，几乎无成本和耗时,并且可以针对某项目某节点扩容。
2. 多站点接入服务器成本曲线趋于平稳，增长极少，并不需要一站一服务器申请。
3. 大量配套监控，组件化接入监控，监控粒度。
4. 更细，范围更小，减少故障排查问题成本。
5. 代码复用性极高，无需冗余重复代码，较大降低。
6. 项目维护成本。

### 实现业务指标

1. 降低成本。
2. 资源复用。
3. 服务有效管控。

### 资料数据存档

http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=21532985

## SOA架构扩展性

### 任务描述

1. 通过很少的改动，就能实现整个。
2. 系统处理能力的线性增长，实现。
3. 高吞吐量和低延迟高能力。

### 解决问题

1. 降低大量改动给系统带来的负面影响，并且解决。
2. 大量冗余，不能细粒度模块化扩展和迭代等问题。

### 对比优势

1. 细粒度的扩展和迭代功能需求，代码复用性强，扩展。
2. 成本低，实现目标速度快，影响范围小。

### 实现业务指标

1. 提升系统稳定性。
2. 较少成本快速迭代。

### 资料数据存档

http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=21532987

## SOA架构性能

### 任务描述

核心体现在TPS/QPS，以及增量式扩容对性能影响。

### 解决问题

随着站点或者业务增量扩充下性能不高，抗压能力弱等问题。

### 对比优势

随着节点增加可以增量式提升,并且可以动态削峰填谷。

### 实现业务指标

1. 提高用户体验。
2. 流量转化率提升。

### 资料数据存档

http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=21532987

## SOA可靠可用性

### 任务描述

主要体现在系统的稳定性，在高峰压力下无单点故障，保持系统永远可用，故障节点可自动恢复，数据保证最终可用。

### 解决问题

1. 降低线上系统故障率以及故障时长，解决出现故障。
2. 不能自动恢复，用户不能正常使用问题。

### 对比优势

天生集群模式，故障背景下自动流量转移，可以实现数据最终一致性，保证系统始终可用。

### 实现业务指标

1. 减少或者杜绝故障带来的利益损失。
2. 提升用户体验。

### 资料数据存档

http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=21532992

## SOA架构安全性

### 任务描述

系统对于风险预控，以及在风险攻击下的稳定性，网站服务的可用性。

### 解决问题

解决不能动态有效地统一入口进行外部风险预判和管控，造成利益直接或者间接损失。

### 对比优势

1. 入口可以统一在proxy层面，缩小管控范围。
2. 可以动态定制可插拔规则引擎拦截或者记录外部风险。

### 实现业务指标

1. 提升系统稳定性。
2. 降低外部风险对底层服务数据的攻击。

### 资料数据存档

http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=21532994

## SOA平滑发版

### 任务描述

不停机下的需求上线发版。

### 解决问题

减少或者杜绝快速业务迭代背景下网站停机带来的利益损失。

### 对比优势

1. 无状态多节点可以手动上下线,并且可以将部分流量或者测试用户流量切入新节点运行。
2. 上线测试不通过只影响局部节点，不影响线上正常使用，无需新服务器环境。

### 实现业务指标

1. 降低风险。
2. 减少成本。
3. 快速迭代发版。

### 资料数据存档

http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=21532996

## SOA站点增量接入

### 任务描述

快速增量接站,主要体现在服务器，耗时，代码改动的影响。

### 解决问题

解决快速接站带来的机器资源不能复用，代码冗余频繁改动，成本大的问题。

### 对比优势

1. 机器资源可最大程度复用。
2. 代码复用性强。
3. 增量接站耗时时间较短。
4. 不影响已有网站或者系统使用。

### 实现业务指标

1. 降低成本。
2. 提升效率。
3. 提高稳定性。

### 资料数据存档

http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=21532998

## 服务治理平台蓝图

### 任务描述

SOA服务化终极架构方案。

### 解决问题

架构蓝图目标为奋斗方向,从技术层面驱动业务发展。

### 对比优势

1. 清晰终极架构方案。
2. 以技术引导业务流程优化设计。

### 实现业务指标

驱动业务设计优化，功能改善。

### 资料数据存档

http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=21533000

# 组件化接入

## 组件化接入

### 任务描述

1. 按照插件形式可插拔的接入或者。
2. 卸载功能，独立于业务代码。

### 解决问题

1. 解决业务代码和第三方系统强耦合问题。
2. 可以独立专一的解决对应组件问题，降低维护成本。
3. 可插拔形式可以降低接入卸载风险。

### 对比优势

1. 独立于业务代码,较大的维护业务系统稳定性。
2. 独立的插件，灵活接入扩展。
3. 独立的组件开发，专注对应的技术。

### 实现业务指标

1. 提高系统稳定性。
2. 快速接入和卸载，降低成本。

### 资料数据存档

http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=21533004

## soa-proxy

### 任务描述

服务化功能接口流量统一网关。

### 解决问题

路由转发，服务监听，服务熔断与降级，安全管理，数据监控，流量管控。

### 对比优势

1. 服务接口粒度管控。
2. 流量统一入口安全管理。
3. 灵活的服务治理，服务上线与下线。

### 实现业务指标

1. 系统安全。
2. 系统稳定。
3. 服务自动化治理，降低成本。

### 资料数据存档

http://wiki.hqygou.com:8090/display/MY/soa-proxy

## soa-fireye

### 任务描述

soa服务化平台全链路跟踪，可以跟踪到接口，db，redis,mq等输入输出。

### 解决问题

1. 解决不能细化到接口层面监控服务运行状态问题。
2. 解决无法根据以往的日志跟踪到每个请求的上下文问题。
3. 解决不能监控所有接口层面的性能和耗时统计分析问题。

### 对比优势

全链路请求跟踪，在复杂庞大的调用过程跟踪每个细节，可以得到性能数据，更好的管控治理服务。

### 实现业务指标

1. 服务治理监控。
2. 提升故障排查恢复速度以及效率。

### 资料数据存档

http://wiki.hqygou.com:8090/display/MY/soa-fireye

## soa-cache

### 任务描述

1. redis缓存客户端，屏蔽redis。
2. 内部实现处理细节，并且方便链路埋点接入。

### 解决问题

1. 屏蔽redis api客户端处理细节，方便业务接入。
2. 加入埋点,监控redis的key以及value。

### 对比优势

可以灵活针对redis操作可拆卸式加入redis数据链路监控，对业务代码零侵入。

### 实现业务指标

1. 提升业务代码稳定性。
2. 加强数据监控,降低风险。

### 资料数据存档

http://wiki.hqygou.com:8090/display/MY/soa-cache

## soa-search

### 任务描述

1. 网站后台搜索引擎，提供多维度。
2. 数据查询，优化后台运营管理流程，提升后台管理效率。

### 解决问题

解决后台复杂数据多维度查询性能问题。

### 对比优势

1. 定制化的搜索api，方便业务方使用。
2. 动态根据DB定制索引结构，方便多个数据系统接入搜索。

### 实现业务指标

1. 降低运营管理成本，提升运营工作效率。
2. 提升数据查询效率。

### 资料数据存档

http://wiki.hqygou.com:8090/display/MY/soa-search

## soa-disconf

### 任务描述

统一配置管理中心，将复杂的配置文件统一化管理，动态下发配置，并且进行二次开发，定制化个性化需求。

### 解决问题

1. 解决配置文件分布范围广，运维开发部署复杂困难。
2. 持续接入开发成本大等问题。

### 对比优势

1. 配置文件可以统一中心管理。
2. 可以动态下发配置文件，动态加载内存，无需重启服务。

### 实现业务指标

1. 提升开发运维效率。
2. 降低系统维护成本。

### 资料数据存档

http://wiki.hqygou.com:8090/display/MY/soa-disconf

## soa-mq

### 任务描述

多元mq客户端组件，屏蔽其内部实现细节，方便业务方接入，灵活扩展第三方消息系统api接入。

### 解决问题

解决各种第三方mq系统繁琐的api接入，对业务侵入大的问题，并且可以接入链路埋点。

### 对比优势

可以灵活扩展第三方mq api接入，并且可拆卸式的加入链路监控，对业务代码零侵入。

### 实现业务指标

1. 提升业务代码稳定性。
2. 降低mq系统风险。

### 资料数据存档

http://wiki.hqygou.com:8090/display/MY/soa-mq

## ejob

### 任务描述

管理复杂庞大的定时任务job，实现高可用分片等功能，人工干预job的运行状态，并且完整体系监控job运行状态，保证业务数据以及功能的可靠性。

### 解决问题

解决复杂数量庞大的job维护困难，可靠性难以保证，无法监控实时运行状态，导致业务功能异常以及不稳定等问题。

### 对比优势

1. 人工页面定制化job。
2. 完善的监控体系，及时报警。
3. 高可用，无单点故障。
4. 实现分片功能，提升任务处理速度。

### 实现业务指标

1. 降低维护成本。
2. 提升系统稳定性。

### 资料数据存档

http://wiki.hqygou.com:8090/display/MY/ejob

## elog

### 任务描述

随着公司业务日益扩张，应用服务器规模也成倍增长，相对应的也会产生大量的系统日志和业务日志，系统日志有助于我们了解系统的运行情况，业务日志能帮助我们了解运营情况，除此之外还有一些metrics用来度量服务调用次数的日志。在分布式系统中，向外界提供服务的往往是多台服务器，对于运维和开发而言，如果出现一次调用错误，到底是哪台服务器出了问题，只能一台台去找；对于数据部门而言，用传统方法分析日志数据也已经不可能。

### 解决问题

1. 构建应用系统和分析系统的桥梁，并把他们之间的关联解耦。
2. 支持近实时的在线分析系统和类似于hadoop之类的离线分析系统。
3. 具有高可扩展性。即当数据量增加时，可以通过增加节点进行水平扩展。

### 对比优势

1. 低成本可拆卸式将第三方系统接入elog。
2. 全面的日志数据分析，得出各个维度分析数据，为运营决策，线上维护提供支持。
3. 日志统一收集分析系统，避免重复造轮子。

### 实现业务指标

1. 降低开发运维成本。
2. 数据分析，为运营决策提供支持。

### 资料数据存档

http://wiki.hqygou.com:8090/display/MY/elog

## esearch

### 任务描述

面向用户全文多维度搜索。

### 解决问题

解决用户搜索商品数据不准确，权重不清晰，性能不高，系统不稳定等问题。

### 对比优势

1. 快速接站，最大化复用资源。
2. 提供实时数据增量更新，为用户搜索到最新数据。
3. 搜索性能高，系统高可用，保证用户搜索功能稳定性。

### 实现业务指标

1. 提升用户转化率。
2. 提升系统稳定性。
3. 提升用户体验。

### 资料数据存档

http://wiki.hqygou.com:8090/display/MY/esearch

# 业务代码设计

## 业务模块化

### 描述

根据不同的业务边界，soa服务化会按功能进行不同粗细粒度的拆分。方便各模块之间的管理，开发。解耦各模块之间的依赖。达到模块之间的水平扩展，对整个系统而言提供更好的可扩展性，系统的抗压性。从人员资源利用来看，由于边界划分清楚，划分粒度较传统架构更小,开发人员更容易了解并关注业务，人员之间的配合更密切。可以大大提高开发人员工作效率。

### 解决问题

1. 提高工作效率，职责更为明确;
2. 搞高系统可扩展性，抗压性;
3. 更好的适应业务变化。

### 对比优势

1. 模块化部署，降低运维成本；
2. 业务模块化，降低开发成本。一处业务的变更与其它业务模块无关；
3. 风险可控，一个模块出现问题不影响其它模块；
4. 高并发，如果一个业务模块并发过大，只需要单独对并发大的模块进行水平扩展；
5. 业务模块之间更好隔离，降低模块之间的耦合度。减少业务相互之间的影响。

### 业务指标

1. 有效降低运维成本；
2. 降低开发成本；
3. 降低开发与线上运行风险。

## 微服务接口设计

### 描述

服务接口是soa最小粒度的划分。soa对外提供统一的接口，业务调用方无需关心接口内部实现，这样可以把业务变化点控制在接口范围内。

### 解决问题

1. 业务更明确可控，职责更为清晰;
2. 更好的应对业务变化；
3. 接口服务简单，只关注一个业务；
4. 更好的提高系统可扩展性，可以根据业务接口提供不同的扩展。

### 对比优势

1. 每个接口单一职责职划分更为清晰;
2. 一个接口对应一个业务点，粒度更细，系统松耦合。有更好的扩展性；
3. 接口更可控，线上运行的出错的几率更小；
4. 更多接口复用。

### 业务指标

1. 开发成本会下降;
2. 减少系统运行风险点,风险可控。

## 环球多站点问题设计

### 描述

soa是解决多站点架构的利器。因为soa可以做到最小粒度的复用。多个站点之间的相同业务提供原子接口复用，不同站点之间提供聚合服务适应不同的业务规则。目标是降低聚合服务的应用，所有站点业务向原子服务靠拢。在接入越来越多的站点的时候，会体现出更明显的优势。soa更像一个服务仓库，任何站点需要任何的服务直接调用soa接口，在需要快速建立一个新站的时候只需要前端提供页面,业务处理调用soa后端的服务仓库及可。在已经形成的服务仓库下 ，可以达到快速适应业务变化，与快速建立站点。甚至可以提供CMS系统自动生成站点。

### 解决问题

1. 结束copy代码的时代;
2. 提高站点之间业务复用;
3. 理清业务代码与结构;
4. 更合理的业务架构设计;
5. 更好的系统扩展性，并实现高可用，高并发的设计。

### 对比优势

1. 多个站点共用服务,节省开发时间提升开发效率；
2. 对业务进行剥离，soa只管后台业务；
3. 向多个站点提供统一服务，统一维护；
4. soa提供整套的电商体系解决方案，走向电商平台化。

### 业务指标

1. 降低运维管理成本；
2. 后续接入更多站点可以降低机器成本；
3. 接入越多的站点业务规则更统一时，可大大降低开发成本。

# 技术接口规范

## 接口文档标准

### 任务描述

规范接口文档规范,保证接口文档完善度和一致性。

### 解决问题

1. 接口无文档调用全靠猜；
2. 接口和文档不一致；
3. 由于人员离职接口功能无人懂；
4. 离职人员交接麻烦,交接不合格；
5. 无知识累积,每次新做功能服务。

### 对比优势

1. Confluence树形目录管理文档便于管理；
2. 文档规范,便于新来人员编写和查看；
3. 严格要求文档质量定期核查责任到人。

### 实现业务指标

1. SOA API 接口文档覆盖率100%；
2. 第三方依赖系统接口记录100%；
3. 接口及时更新率达到90%(开发中接口不要求接口和文档一致)。

### 资料数据存档

## 外部接口对接协议规范

### 任务描述

规范外部接口对接协议,完善外部接口文档,提高外部接口稳定性。

### 解决问题

1. 外部接口文档维护和后续人员接手时成本；
2. 外部接口对接人员不稳定；
3. 外部接口稳定性。

### 对比优势

1. 外部接口通讯稳定性好；
2. 文档完善；
3. 新手上手快。

### 实现业务指标

外部接口调用文档率100%。

### 资料数据存档

## 代码标准规范

### 任务描述

提高代码质量,提高代码可读性,提高代码健壮性,提高开发人员码代码能力。

### 解决问题

1. 代码可维护性；
2. 代码可读性；
3. 代码可扩展性；
4. 对比优势。

### 对比优势

1. 代码可读写性好；
2. 易于维护；
3. 问题查找良好。

### 实现业务指标

1. 每周进行代码评审；
2. 注释覆盖率90%。

### 资料数据存档

1. [**Java开发规范及注意事项**](http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=3277149)**；**
2. [**工程站点多数据源设计规范**](http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=9601300)**；**
3. [**站点sitecode规范**](http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=12616945)**。**

# SOA模块化业务设计

## 业务模块化

### 任务描述

SOA是面向服务的体系结构，是一个组件模型，它将应用程序的不同功能单元（称为服务）通过这些服务之间定义良好的接口和契约联系起来。

### 解决问题

1. 系统缺乏整体规划和设计，导致代码结构不清晰，到处充斥过期或无效代码，系统维护成本居高不下；
2. 重复造车轮，系统功能无法有效的复用，浪费了大量的业务及技术资源；
3. 系统核心流程高耦合低内聚，系统随时处在牵一发而动全身的不利局面；
4. 系统架构伸缩性不足，无法有效的利用好硬件资源；
5. 抗风险能力较差；
6. 对业务的支持力度有限，复杂的逻辑和功能无法实现。

### 对比优势

1. 可复用；
2. 松耦合；
3. 明确定义的接口；
4. 无状态的服务设计；
5. 基于开放标准。

### 实现业务指标

1. 可供复用的10个模块（结算和店铺规划中），降低开发成本；
2. 提升基于SOA建站的效率。

### 资料数据存档

## 模块系统化（趋势）

### 任务描述

基于各个模块业务设计的低耦合性和各模块对于业务的深度挖掘和优化，最终实现每个模块都是一个可以独立接入站点的系统或平台，针对模块所负责的业务线对网站进行优化，提升效率，降低开发成本。

### 解决问题

1. 解决因原有系统架构原因导致的不能处理的问题；
2. 为外部合作者提供一站式解决方案。

### 对比优势

1. 更专业的模块化业务解决方案；
2. 可供公司内外部系统或站点单独接入的系统化服务。

### 实现业务指标

1. SOA物流模块已单独接入GB提供物流服务；
2. 降低开发成本；
3. 可对外提供基于电商各个业务模块的服务，增加商业变现维度。

### 资料数据存档

# 更聚焦的模块业务优化

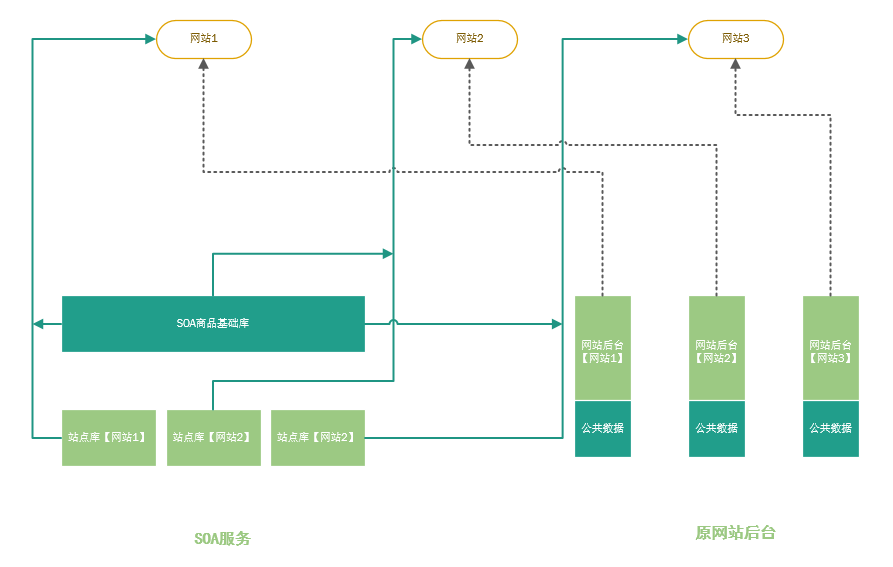
## 商品模块业务优化

### 任务描述

提供必要的商品资料数据以及业务管理功能，满足网站运营、商品推广等相关业务和系统的要求，提供更加完整、灵活、高效的商品资料服务。

### 解决问题

1. 商品资料按照基础库（共用资料）和站点库（个性化资料）分别存储，解决多个网站商品共用资料部分的冗余问题；
2. 商品基础资料提前同步，减轻因资料变更数据过多产生的延迟问题；
3. 提供统一的服务和统一的功能，商品资料管理通用化，减轻涉及的周边系统的数据获取压力；
4. 通过提供通用化服务，重新梳理网站业务，保证网站的主干业务清晰明了，也更便于功能维护。



### 对比优势

1. 数据统一化；

通过对商品资料的统一化管理，大大减少商品资料同步到各个网站时，商品资料在各个网站资料冗余的情况，由SOA统一管理后，大大减少商品资料同步成本和维护管理成本。

1. 服务统一化；

通过梳理网站针对商品资料的业务需求，将商品服务作为SOA底层基础服务，提供统一公共化服务，在更加稳定的支撑和实现业务需求的同时，可以更加稳定的提供底层服务，保证网站的售卖主干业务的稳定、流畅。

1. 平台多元化；

针对网站多仓库、多平台的业务场景下，针对商品资料管理中的部分业务：商品状态、商品价格等方面，可以针对不同仓库、不同平台进行单独管理，为网站提供更加多元化的商品资料管理。

1. 业务明确化；

作为网站管理后台，按照后台管理职能、使用场景和操作业务的维度，更加清晰和明确的支撑相关业务，跟周边系统明确业务边界，提供更新精准、高效的功能服务。

### 实现业务指标

1. 可以针对商品进行跨仓库、跨平台、跨站点（未来）的多维度批量操作，大大节约操作成本、缩短设置流程、提高工作效率；
2. 更规范的商品价格计算，根据业务设置多个价格因子、考虑供应商限价、最低利润率等，提升商品价格准确性，保障网站利益和供应商权益。

### 资料数据存档

## 库存模块业务优化

### 任务描述

即提供统一的库存数据以及业务管理功能，满足网站运营、推广等相关业务和系统的要求，对内提升效率，对外提供更加实时、准确、高效的库存服务。

### 解决问题

1. 通过建立销售仓的的虚拟联合库存运作管理模式，在接入多站点的库存相关数据后，解决在供应链系统中各节点相互独立库存运作模式导致的库存同步不及时的问题；
2. 通过对售前库存变更中多节点的精准控制，提前占用库存，提供更实时精确库存数据，解决网站售卖超卖断货，影响客户体验的问题；
3. 通过提供通用接口与统一服务，支持快速响应网站库存业务相关流程的变化，减少了业务应用实现的系统限制。

### 对比优势

1. 实时共享库存服务

相比之以往各站点的相互独立库存运作模式导致的需求变异放大现象，SOA所提供的虚拟联合库存方式是解决库存同步化沉痼的一种有效方法。通过销售仓关联、公私库存划分、实时库存变更，使供应过程中的每个库存所属方都从相互之间的协调性考虑，保持相邻两个节点之间的库存所属者对需求的预期一致，从而实现各个库存点的信息共享，以及各库存点间稀缺资源的快速调配，从而大大提高了库存实际利用效率。

1. 降低开发维护成本

库存服务作为SOA底层基础服务，基于标准的服务构建并提供通用接口。对于被服务的相关模块与周边系统而言，无需再考虑底层的集成与复杂的业务逻辑，而是考虑如何复用现有的库存服务或检查如何让服务被重新利用，更加明确业务边界，避免重复开发，从而节省了大量的资金和时间。

1. 灵活响应业务需求

相比之以往因业务变动而被迫进行大规模的应用代码开发方案，SOA库存服务这种基于消息机制的模式，可以通过分析服务间的交互，帮助应用单位了解在哪个节点什么业务逻辑被切实的执行，从而更有针对性的去局部优化调整已有服务，保障业务需求务在商业环境许可的时间内对变化的市场做出快速的响应。

### 实现业务指标

1. 库存数据准确性，减少超卖。
2. 综合后端仓配系统考虑对库存周转率的提升。

### 资料数据存档

## 会员模块业务优化

### 任务描述

负责网站会员数据的收集管理，为上下游系统提供数据支持；完善用户画像，为数据化营销提供精准的用户群体；用户分层，发展有价值用户，限制恶意用户。

### 解决问题

1. 提供更完善的用户基础数据
2. 为会员营销提供用户分群数据及更多玩法

### 对比优势

1. 数据完备

更多维度记录用户基础数据，数据涵盖面更广。

1. 功能通用

功能设计上，通用兼容，能支持网站不同运营策略。

1. 风险管理

多维度管理用户数据，不仅能保证用户账号安全，而且针对恶意用户能进行区分和限制。

### 实现业务指标

1. 会员活跃度

会员分层找出活跃用户数据，通过运营策略提升活跃用户生命周期。

1. 风险管控

找出风险用户和恶意用户，通过限制操作，降低恶意注册和恶意下单。

### 资料数据存档

## 营销模块业务优化

### 任务描述

营销模块通过丰富营销活动玩法、设立多种coupon形式、控制促销价格等来最大化的刺激网站业务，促进商品销售、拉动新用户注册、保障老用户留存。

### 解决问题

1. 自由化设计解决营销活动扩展性差，复用性低，维护难问题。
2. 易用性结构降低运营学习、系统开发成本。
3. 统一标准保证数据统一、系统开放。
4. 数据完整保存，保障数据可追溯。

### 对比优势

1. 自由易用设计，多元化可扩展
   1. 营销模块遵循自由、易用设计原则，将业务拆分成基础条件，以模板化形式自由组合，增加基础条件的复用性，支持的场景以平方级增加，业务扩展只需增加一两个条件，即可完成对复杂场景的支撑
2. 业务数据统一
   1. 营销活动统一管理，活动全平台通用，避免各网站需求重复开发，数据规格不一致的情况发生，减少功能开发、系统对接成本
   2. 营销数据完整保存，只开放逻辑删除，支持后续业务重现，数据统计
3. 系统开放、业务独立
   1. 营销模块业务提供开放接口，支持上下游系统自由调用，除常规售前流程外，还可以支持售中、售后流程中部分业务

### 实现业务指标

1. 稳定支持营销业务
   1. 第一阶段承接GB现有营销功能，保障现有站点业务不受后台切换影响
   2. 第二阶段扩充营销功能，根据业务需求扩展营销活动
   3. 逐步建立规则引擎优化内部逻辑
2. 数据化运营支撑

结合大数据量化营销成本、营销效果，为站点运营提供数据化支持

### 资料数据存档

## 购物车模块业务优化

### 任务描述

提供网站必要的购物通道以及业务功能，满足支撑网站活动、售卖及结算等相关业务和系统要求，提供更加完整、灵活、高效的购物结算服务。

### 解决问题

1. 通过提供通用化服务，重新梳理网站业务，保证网站的主干业务清晰明了，也更便于功能维护
2. 通过可购买校验、库存校验、占用和扣减，减少各网站超卖现象
3. 满足网站下分仓下单，且通过拆单，配合支撑物流核算精准运费

### 对比优势

1. 服务统一化

通过梳理网站针对购物车模块的业务需求，将购物车模块作为SOA底层基础服务，提供统一公共化服务，在更加稳定的支撑和实现业务需求的同时，可以更加稳定的提供底层服务，保证网站的售卖主干业务的稳定、流畅。

1. 平台多元化

多网站、多平台的业务场景下，购物车提供满足业务需求的核心功能，同时区分网站平台提供局部功能可配置化

1. 业务明确化

作为网站管理后台模块，按照后台管理职能、使用场景和操作业务的维度，更加清晰和明确的支撑相关业务，跟内部交互模块明确业务边界，提供更新精准、高效的功能服务。

### 实现业务指标

1. 满足原站点不能满足的业务需求，如多仓下单等
2. 降低业务处理方式缺失造成的业务问题，如超卖等

### 资料数据存档

## 物流模块业务优化

### 任务描述

物流模块的目的是物流规则校验、物流运费、商品发货路由计算，提高物流服务质量。

### 解决问题

* 下单后用户选择的物流方式不可发或与选择的物流方式不符
* 运费计算不准确，需用户补齐

### 对比优势

* 自由易用设计，多元化可扩展

物流模块遵循自由、易用设计原则

* 规则灵活

物流模块提供可以各种组合的规则，满足运营方对物流的限制

* 系统开放、业务独立

物流模块业务提供开放接口，支持上下游系统自由调用，除常规售前流程外，还可以支持售中、售后流程中部分业务

### 实现业务指标

* 稳定支持现有业务

第一阶段承接GB现有物流功能，保障现有站点业务不受后台切换影响

第二阶段扩充物流规则，根据业务需求扩展逐

* 提供运费计算准确
* 降低功能实现导致的业务问题，如不可发订单减少，减少网站不可生单数量等

### 资料数据存档

## 订单模块业务优化

### 任务描述

设计订单服务框架，支撑销售端从生单到完成支付过程中的业务逻辑和数据处理，与后端系统OMS做好数据交互，传递和更新订单数据。

### 解决问题

1）支持跨仓下单，预售商品与普通商品分离

2）生单与选择支付方式分离，生单后可变更支付方式

3）订单直接扣减SOA库存数据，更实时准确

4）订单状态记录更准确

5）订单取消时间与库存释放时间分离；订单取消时间可配置

### 对比优势

* 通过模块间明确定义的接口进行数据交互，降低业务耦合
* 与外部系统对接约定协议，清晰划分业务边界
* 统一的底层服务，可快速接入站点
* 服务原子化，可支持站点个性化需求

### 实现业务指标

1）支撑订单生单到完成支付各环节中的，库存扣减和释放、状态变更等业务；

2）为外部系统的销售数据统计提供基础数据支撑；

3）优化系统底层逻辑，支撑网站扩展新业务带来的变更。

### 资料数据存档

## 支付模块业务优化

### 任务描述

### 解决问题

1：清晰架构发展路线

整合各收款平台的集成，承接各站独立收款任务

2：根据实际资源和成本，采用不同版本的架构

3：解决当前业务下的架构问题

支付功能的重复开发，数据分散和混乱问题。

### 对比优势

1：对未来的规划清晰。成本资源可控

2：可以提前预演不同业务场景下的架构设计，及时解决当前问题

3：可以动态选择不同架构来支撑业务体系

### 实现业务指标

1:成本可控

2:资源最大化规划使用

3:风险预判

支付成功率76.4%

拒付率0.75%

### 资料数据存档

## 推动周边系统优化

### 任务描述

SOA各个模块，基于对于业务的优化和效率的提升，对当前对接的外部系统提出优化改进建议，推动外部系统进行业务改进，提升用户体验。

### 解决问题

* 功能兼容性差
* 各系统之间公共数据不统一
* 公共数据维护成本高

### 对比优势

* 开放化的标准定义接口，保证多系统兼容
* 推行统一标准，逐步完善各系统之间的共同数据的统一性
* 以SOA为出发点，将公共数据同步至CDP系统，进行统一管理维护，各系统调用，降低维护成本

### 实现业务指标

* 数据统一
* 降低公共数据维护成本
* 提升功能业务兼容

### 资料数据存档

## 更清晰的需求处理流程

### 任务描述

SOA的需求处理过程，必须从需求出发经过

需求调研，提炼核心需求并思考满足核心需求的方式；

需求设计，思考功能概要及细化功能设计及标准PRD文档输出；

需求评审，当产品设计方案确认，会快速的进行至少3轮需求评审：内部需求评审完善产品方案，业务需求评审确认产品方案，技术评审讲解和实现需求方案。

需求验收，按照模版化的需求验收文档，对需求的实现进行高标准的验收反馈。

### 解决问题

1. 产品设计方案最终不能满足需求
2. 产品设计方案不完善
3. 需求、产品设计方案、实现效果不一致
4. 为了需求为做需求的现象

### 对比优势

1. 标准流程化处理，最大程度的降低因为人员或者经验带来的差异而影响需求实现。
2. 多轮需求评审，保证需求、产品设计方案、实现保持一致并从用户体验的战略层满足真正需求。
3. 高质量的各流程模板

### 实现业务指标

1. 产品方案更能为公司或用户提供价值。
2. 降低因需求处理造成的成本浪费

### 资料数据存档

## 更专业的PRD文档

### 任务描述

PRD作为产品经理输出的必备文档，SOA所有PRD都严格按照SOA模板文档标准，基于业务线进行梳理，更专业、更清晰、更易懂。每次发版的需求（大版本），都按照版本签字打印，做到可追朔、可复查、可追责。

### 解决问题

1. 业务无法复盘
2. 业务流程不清晰
3. 功能或数据出现问题难核查、难追责
4. 因沟通成本提升，影响的整体工作效率的降低
5. 新人无法快速熟悉和掌握业务

### 对比优势

1. 基于业务流程文档更清晰
2. 标准文档，降低差异（人员和经验）

### 实现业务指标

1. 提升工作效率
2. 降低沟通成本
3. 使业务清晰明确化
4. 降低人员流动时新人的业务学习成本

### 资料数据存档

# 运维优化

## 物理资源使用曲线

### 任务描述

（1）对服务器物理资源使用情况曲线进行监控。

（2）监控包括CPU负载、Memory、IO、网络等信息。

### 解决问题

服务器资源使用情况的综合负载分析掌握及应用存活状态情况。

### 对比优势

1. 可实时获取服务器物理资源使用数据。
2. 服务器负载异常时可告警通知。

### 实现业务指标

1. 资源最大化规划使用。
2. 控制并降低服务器资源的成本。
3. 支撑业务的发展，预判扩展资源需求。

### 资料数据存档

服务物理资源图表展示WIKI；

<http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=21534251>

## 模块化运维

### 任务描述

1. SOA按8个领域模块划分，上线时各模块可独立隔离部署。
2. 运维通过自动化的脚本更新发布到指定服务器。支持更新包的增量上传方式。
3. 明确运维分工：DBA运维负责Mysql、Redis；SOA运维负责SOA相关应用部署维护；PHP运维负责网站前端PHP应用部署维护。

### 解决问题

1. 应用按业务领域模块划分。
2. 对应运维工程师可独立隔离部署应用。
3. 提高运维工作的效率，降低运维成本。

### 对比优势

1. 可降低模块部署更新时间，提高发布效率。
2. 明确了运维分工提高运维协作效能。
3. 提高运维效率，降低运维支出成本。

### 实现业务指标

1. 部署发布时间风险可控。
2. 运维模块化分工互不影响。
3. 快速迭代发版。

### 资料数据存档

模块化运维说明WIKI

<http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=21534267>

## 通用性操作以及风险范围可控

### 任务描述

1. 通用性操作即是运维常规发布工作内容，采用自动化脚本实现。
2. 运维常规发布需建立规范流程，确保发布稳定，控制风险范围。

### 解决问题

1. 项目版本发布的流程不明确、上线操作混乱无章可循。
2. 发布新版本导致现有系统系统运行异常。

### 对比优势

1. 通用性操作采用自动化运维系统，可提升运维效率。
2. 自动化的脚本经过大规模测试验证，稳定可靠，减少运维事故发生。
3. 通过自动化运维降低运维操作的风险。

### 实现业务指标

（1）标准化通用操作流程规范。

（2）部署发布风险可控，减少运维事故。

### 资料数据存档

发布上线规范流程WIKI

<http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=16712728>

## SOA监控体系

### 任务描述

1. SOA的需要监控范围包括：服务器、DB、基础组件、应用。
2. 监控体系包括：elog、zabbix，链路、RMS实时监控系统、soa-canary监控。
3. 监控对象出现异常能发出告警通知。

### 解决问题

1. 按不同维度监控服务器与应用。
2. 当出现异常能及时发出告警通知到负责人，快速定位不同模块问题并解决。

### 对比优势

1. 以前的系统只有zabbix单一的监控，其范围与作用比较有限制。
2. SOA采用多方位全范围覆盖的监控系统，能确保服务器与应用指标异常触发预警。
3. 链路系统可实时监控SOA服务调用链深度与服务可靠性。
4. 多套监控系统保证服务稳定运行。

### 实现业务指标

1. 监控到异常时发出告警通知。
2. 全方位确保系统正常运行。
3. 降低服务的异常概率，提高网站可用性。

### 资料数据存档

soa监控体系WIKI

<http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=21534239>

## 风险预判及告警

### 任务描述

1. 明确系统健壮性的监控方式，对服务器资源使用率情况风险可预判。
2. 确保系统异常时能发出明确的告警通知。

### 解决问题

1. 服务器与应用出现异常时，能及时预警并发送告警通知到相关负责人
2. 能实时掌握系统运行状况，快速定位问题并解决。
3. 服务器资源利用及扩展可预判。

### 对比优势

1. 通过告警通知及时解决当前系统问题。
2. 支撑业务系统的高可用与扩展。

### 实现业务指标

1. 监控到异常时相关负责人能预判风险及时处理问题。
2. 全方位确保系统正常运行，对业务扩展所需服务器资源提供数据支撑。

### 资料数据存档

<http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=21534241>

## 组件化运维

### 任务描述

对常用的组件如elog、ejob、kibana、zk、es、zabbix监控等，采用统一规划安装部署

### 解决问题

1. 组件应用的服务器资源统一规划部署。
2. 提供基础公共的组件调用。

### 对比优势

1. 多个项目应用中调用的组件能统一规划部署，便于运维管理。
2. 运维提供统一的基础公共的组件调用，避免重复部署，减轻运维压力。
3. 提高服务器资源综合利用率。

### 实现业务指标

1. 组件化的应用统一规划部署。
2. 提高服务器资源的利用率。
3. 组件化部署的应用便于业务系统接入。

### 资料数据存档

Ejob接入 <http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=13765564>

Elog接入 <http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=14325839>

kibana平台 <http://kibana.elog.softlayer.egomsl.com>

# 项目管理优化

## 版本迭代控制

### 任务描述

需求统一经过PMO收集，根据需求优先级，整理成不同的上线版本。针对需求进行分解排期。每个版本会制定项目总控，由项目总控对项目进行推进，把控项目风险。

### 解决问题

1. 需求整合成版本，减少上线次数，保证线上稳定。
2. 需求由PMO统一跟进，防止需求遗漏。
3. 解决资源冲突，合理分工，提高项目沟通效率。
4. 每个版本制定不同总控，提高个模块的项目参与感。
5. 优化资源配置，提高迭代速度。

### 对比优势

1. 任务拆分更细，执行力更高。
2. 计划详细，分工明确，效率更高。
3. 规划更明确，迭代更快。

### 实现业务指标

1. 提高迭代速度
2. 减少沟通成本
3. 充分利用资源

### 资料数据存档

## 项目质量把控

### 任务描述

编写测试用例，测试用例评审，通过testlink统一管理。BUG和需求由jira统一跟踪跟进。多轮测试，多轮验收。确保上线发布版本都经过测试。

### 解决问题

1. 测试用例评审，提高测试的覆盖率。
2. 通过流程规避上线风险，提高协作效率。
3. 保证上线质量。

### 对比优势

1. 上线流程更规范。
2. 减少人为主观因素，测试质量更高。
3. 上线有标准要求，减少上线带来的风险。
4. 流程更清晰，代码质量更高，保证上线质量。

### 实现业务指标

1. 较少风险
2. 提升效率

### 资料数据存档

## 跨团队协作沟通

### 任务描述

每日站会保证有问题及时反馈，有信息及时同步。项目例会保证项目进度透明，阻碍性问题能得到及时解决。

### 解决问题

1. 通过PMO或者总控提供统一的服务窗口，解决跨部门对接问题。
2. 定时会议沟通，确保项目进度透明。
3. 问题及时反馈，并解决。
4. 跨部门沟通，协调资源。

### 对比优势

1. 沟通效率更高。
2. 信息更透明化。
3. 问题响应更快。

### 实现业务指标

1. 减少沟通
2. 提升效率

### 资料数据存档

## 项目管理配套设施

### 任务描述

使用主流的开源平台，通过工具提升效率，通过工具规范流程。让每个人明确自己在flow中的角色。

### 解决问题

1. jira 能实现缺陷跟踪、需求收集、任务跟踪、项目跟踪和敏捷管理等。
2. jira workflow明确各角色的职责，减少沟通。
3. wiki在线的信息协作和交流，达到信息可视化和信息共享。
4. gitlab 分布式代码管理平台，多分支开发，任务并行开发。分角色管理代码，web界面便于审阅代码。
5. Jenkins 持续集成的核心工具。使代码部署更方便快捷。
6. sonar 代码检测工具，从多维度检测代码，提高代码质量。

### 对比优势

1. 通过工具规范流程，减少人为犯错的概率。
2. 通过工具信息共享，减少交流的成本，记录历史存档，信息可溯源，减少矛盾。
3. 工作流清晰可视，进度清晰，能明确问题出在哪个环节，可通过可视化数据对资源进行调节和优化、

### 实现业务指标

1. 减少沟通成本
2. 提升工作效率
3. 优化资源，提升资源的利用率

### 资料数据存档

# 团队管理

## 团队结构及分工

### 任务描述

严格细分模块领域，或者业务领域，将责任落实到具体小团队，充分提高并发挥多个小团队并驾齐驱的高效战斗力；另外，分工细分到业务领域，专攻对应的业务流程设计以及业务技术方案，对于整体项目的推动，以及环球电商化平台的发展有非常重要的意义

### 解决问题

1：解决由于业务无拆分，团队所有人对业务掌握不彻底的问题

2：解决各个领域业务积累，技术方案积累不充分的问题

3：解决无法快速满足各个业务领域独立发展问题

4：解决无法将各个领域无法独立对接外部系统的问题

5：解决团队无法形成子团队独立管理的问题

### 对比优势

1：业务积累，技术积累更加专业和深厚

2：团队管理更高效

3：业务版本迭代发展更快速有效

4：为环球整体电商平台发展奠定基础

### 实现业务指标

1：提升工作效率

2：降低成本

3：更有针对性提升员工对业务和技术的能力，创造生产力

### 资料数据存档

http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=22087008

## 工作方式和制度

### 任务描述

摒弃小时制任务分配工作机制，倡导全民皆兵，需要兼顾需求提出，需求设计，需求评审，需求开发，需求兼容性以及扩展性。积极做到对运营，业务，开发落地，上线运营等都有所了解，希望提高自我驱动力，人人都有责任推动任务、积极沟通协作，而不是反向一直被安排任务，充分发挥每个成员的综合素质能力，提高工作效率，最终做到每个员工都是产品设计、技术开发等全面发展

### 解决问题

1：解决小时制任务分配导致任务延期、沟通成本高、开发对业务掌握不透彻、对整体项目推动不积极等问题

2：解决一直被安排任务员工自我提升、自我驱动意识不强的问题

3：解决针对开发、测试、生产环境问题员工职责不明确，导致效率慢，故障解决时间长的问题

4：解决小时制任务分配员工一直疲于完成硬性任务，忽略业务代码设计、流程优化、扩展性、性能等因素的考虑

### 对比优势

1：SOA各个团队每个成员都可以独立应对开发、测试、线上等问题

2：整体项目期间，会主动推动项目的进度，及时主动的协助沟通解决项目问题

3：善于发现运营以及产品 提出的需求或者设计问题，各抒己见，改善业务流程，提高项目质量

4：更多关注业务扩展性、代码性能、扩展性等问题，不局限于只实现业务功能

### 实现业务指标

1：提高团队稳定性

2：提高员工归属感

3：提升工作效率，项目质量，降低成本

### 资料数据存档

http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=22087011

## 团队理念及思想

### 任务描述

以产品为理念，坚决反对外包项目制 需求开发，我们提前做到业务设计、业务流程优化，而不要以bug, 故障反向驱动代码质量提升；坚决反对一味的追求数量上的增加，而忽略质量上的提升，坚决反对只追求一时利益，而忽略公司平台长远发展

### 解决问题

1：改变公司业务太重，不断重构和迁移、资源复用低的问题

2：将公司从纯业务型公司慢慢转变为技术创新型公司，以技术带动生产力

3：改变公司员工项目承包制的工作思想，转变为发展长期稳定的产品为理念

### 对比优势

1：产品理念驱动，产品更稳定

2：通过技术带动生产，避免资源浪费

3：提高团队整体性和战斗力，从思想层面统一步伐

4：有利于公司电商大平台的长期发展

5：形成电商体系生态圈

### 实现业务指标

1：提高生产力

2：提升团队稳定性

3：提高产品质量，降低成本

### 资料数据存档

http://wiki.hqygou.com:8090/pages/viewpage.action?pageId=22087014