# 目录

[目录 1](#_Toc395539456)

[版本修订 2](#_Toc395539457)

[1 目标 3](#_Toc395539458)

[2 工作方向及思路 3](#_Toc395539459)

# 版本修订

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 日期 | 描述 | 修订人 |
| V1.0 | 2014-8-11 | 创建 | 陈绪绍 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 目标

1. 技术架构松散耦合
2. 分层架构模式，每一层职责清晰

如表现层、控制层、业务逻辑层、持久层、领域模型层等。

1. 引入开源的轻量级组件
2. 针对不同行业给能够给相应技术架构建议方案
3. 提供一些通用的技术组件

包括通用技术组件或业务组件。如SSO组件，人员/权限组件。

1. 通用技术架构组件的组合demo
2. 业务场景的技术解决方案

如面对高并发，架构可能是涉及多个系统的，如数据库系统(如主从加，读写分享)，前端(CDN)加速，集群(负载均衡，反向代理等)，Session的统一管理等等。

# 工作方向及思路

1. 引入开源组件，有针对性的对其进行预研，形成相应的产出物，形成team技术资产。
2. 还原开源组件的本来面目

不对开源组件进行过多的包装。

1. 不重新发明轮子，重用已有的口碑好的开源组件
2. 平稳升级相关开源组件版本

比如Spring组件的版本。比如quartz版本，现在已经是2.x了，Biz中还是1.6。不盲目追新，但一般来讲，较新的东西总是修复了bug，提升了性能的。

1. 引入更先进的项目构建工具

比如maven，这也是行业标准。

1. 保留Biz当前一些有用的工具

比如文件上传api，前端ui组件等。

1. 允许自由组合相关组件

如仅将biz当前的dao层可以只作为一个持久层组件。允许选择其它的持久层组件比如spring jdbc，spring hibernate,jpa等。对web mvc组件也提供可选支持。

1. 针对一些业务场景引入其它一些缓存系统，比如Redis，Memcached
2. 针对一些业务场景，引入nosql，比如mongodb
3. 针对一些业务场景，引入大数据
4. 同一种技术方案，提供多个组件供用户选择

比如邮件解决方案，可以给出spring mail，jodd mail等

1. 引入一些通用的高性能web优化技术

如前端脚本压缩(biz中实际已经有了)，cdn加速等。

1. 引入一些集群，负载均衡技术。以应对更大用户量的业务场景。
2. 对业内一些较通用的、较流行(业内可能已经验证了其可行性)的技术架构进行总结，并给出基本的demo。

r1团队未来可能更多的是技术框架、组件培训(付费的)；或有更优异的付费组件。

1. 总结最佳实践

为增强代码质量，对企业级开发的最佳实践进行总结，并推广。

1. 业务场景预研及总结

面对高并发，架构可能是涉及多个系统的，如数据库系统(如主从加，读写分享)，前端(CDN)加速，集群(负载均衡，反向代理等)，Session的统一管理等等。