小米的开源参与之道

冯宏华

云平台存储组

大纲

- 小米的开源现状
- 小米对开源的态度
- 小米的开源参与方式
- 一点思考

小米的开源软件使用列表

Linux MySQL Redis PHP Nginx Resin Rabbit-MQ

Discuss Storm MemCache Supervisord Tomcat Java

Lucene Tornado MongoDB Jetty Spring Thrift Python

…

开源软件在小米的服务领域

◆ **所有业务部门**：硬件，MIUI，电商，电视，路由器，云平台，智能家居，互娱…

◆ **几乎所有业务**：云服务(Mi Cloud)，小米推送，主体市场，小米账号，应用市场，消息系统，阅读，支付，米聊，米币，浏览器，黄页，广告，搜索…

小米的开源软件贡献列表

HBase HDFS Zookeeper YARN MapReduce SenseiDB

Spark Impala Hive Kudu Storm Kafka Scribe Thrift

…

小米的开源重要贡献数字

- ✓ HBase : 提交了347个patch，被社区接受了195个
- ✓ Hadoop : 提交了85个patch，被社区接受了42个
- ✓ Spark : 提交了63个patch，被社区接受了58个
- ◆ 3个HBase Committer，2个SenseiDB Committer

小米的开源现状小结

- 与公司业务成长息息相关，相辅相成
- 开源社区贡献：
  - 2012年下半年开始发力
  - 渐入正轨，初见成效，大有可为

大纲

- 小米的开源现状
- 小米对开源的态度
- 小米的开源参与方式
- 一点思考

取自社区，回馈社区

- 对社区已有较成熟实现的系统，不另起炉灶，不重造轮子
- 对基于社区版本开发的新功能新改进，饮水思源，及时回馈

积极参与，保持同步

- 对社区重要改进的讨论/实现积极参与，多发声，增强影响力
- 线上/线下保持与社区重要参与者的联络与沟通

业务第一，开源次之

◆ 业务的需求优先级高于开源社区(即使不能merge进社区导致合并版本时很辛苦)

◆ 不以提交/被接受的patch数目为目标(不助长“墙内开花墙外香”)

◆ 不凭空想象和开发“可能会有用”的新功能

保证质量，诚信负责

- 不把自己未在实际业务验证或不满足实际业务需求的产品/改进放出去
- 对所有开源的产品/改进提供详尽描述(对不足不隐瞒不回避)
- 对开源出去的产品/改进的问题/反馈第一时间响应

大纲

- 小米的开源现状
- 小米对开源的态度
- 小米的开源参与方式
- 一点思考

“先斩后奏”(先做后说)

◆ 很多业务提出的新需求，不能期望社区有人帮你做，也不能承担先和社区讨论清楚的时间成本

◆ 做好并在实际业务场景得到验证后的patch提交时更有说服力(场景/背景/测试/效果…)

社区版本：保持距离，及时同步

- 不亦步亦趋：完全同步则失去弹性，吃力不讨好
- 不自成一统：完全分道扬镳则失去选择开源的初衷

认理不认人，并保持礼貌

◆ 社区参与者来自世界各地五湖四海，文化/习惯迥异，但在代码面前人人平等，这是一个基准

◆ 要能忍受你认为是“愚蠢”的声音，就事论事，保持克制和礼貌(社区就是一个人群，和现实中的人群一样有无奈有妥协)

投桃报李，互帮互助

◆ 有些功能/代码可能当前业务用不到，被人请求时仍然要尽力提供帮助：review代码、问题分析等

◆ 自己提的patch，没有人回应时，可以直接给对这块代码熟悉的committer发信请求review

保持中立，谋定后动

- 对任何针对设计/实现上的“争吵”，不存私心，不站队
- 对任何分歧或讨论，先想清楚再发声，尽量周全/清晰

保持耐心，贵在坚持

- 重大的patch几乎都是需要长时间反复的讨论、review和修改
- 对正确的事情要有耐心和坚持(即使刚开始不被理解和尊重)

大纲

- 小米的开源现状
- 小米对开源的态度
- 小米的开源参与方式
- 一点思考

“溯洄从之，道阻且长”

- 开源软件开发速度(甚至质量)常常不能令人满意
  - 开发者不是一个真正的“团队”(同一段时间内有着同一目标的一群人)
  - 极难做到所有人在“同一页”(same page)
  - 常会走弯路(无论设计还是实现，甚至bug-fix)

设计/概念的一致性问题(concept integrity)

- 开源软件相较闭源软件更难做到concept integrity
  - 人员、空间、时间等情况使此问题恶化
  - 超大型系统此问题导致“盲人摸象”情况严重

千般不如意，但终归向好

◆ 开源不是最好的多人协作开发的模式，但从大时间尺度衡量却足够好(good enough)

- ➢ 理论上所有的错误/不足终能(eventually)被纠正/改进(虽慢但总是朝好的方向前进)
- ➢ “凡走过必留痕迹”(即使失败的项目，其有价值的地方/代码仍可被借鉴和参考)

Q & A