合并项目总结 – 讨论后整理

# 需求

* 尽量减少定制需求。要求销售，产品组在和客户讨论时，不做定制需求
* 定义详细需求文档的规范。可以找一个范本，需要包括但不限于：功能，接口，报错，不同场景下的功能描述。
* 独立成立一个小组，去做客户提出来的小需求/线上Bug修复这样的小版本。可以做到一个月发布一波。让客户的需求在这些版本中得到实现。
* 减少并行项目，并且在两个项目之间的间隙拉开，至少要隔开一到两周。
* 小版本和独立部署定制功能，也都要写清楚详尽的需求文档和测试用例。这样之后就可以有据可依。
* 独立部署定制功能合并到共享平台时，需要重新做需求和设计，需要考虑合并到共享平台会不会有问题。？ --现在一发布就合并, 这个问题不会有了
* 需求文档更新要及时，做到1更新需求文档，2发送邮件或群消息通知，3安排好开发修改代码/测试修改用例，4如有必要调整计划
* 希望在前一个项目结束的前2个星期就能拿到下一个项目的需求, 项目后期一般事情不多了

# 设计

* 一定要有设计文档，需要包含
  + 数据库设计文档。这个最好保存在一个地方，有一个完整的数据库设计，就像之前的Comm100.Database工程一样。

1. 总的文档很久没有更新了, 2. 应是项目结束以后项目组把新的内容更新到总的文档里面去）
   * 如果有ChatServer改动，需要有ChatServer设计文档，
   * 如果有接口的开发，需要有接口设计文档

* 最好包括流程图，类设计文档，动效交互设计。其中，动效交互设计需要由@Lolly产品小组提供。
* 上述的设计文档也需要进行评审
* 产品组提供的设计图中，界面样式会出现不统一的情况，需要保持统一。@Lolly

# 开发

* **开发自测一定要做**
  + 将开发自测安排到计划中去
  + 增加Unit Test。目前访客端开发已经有单元测试在做，@Roy接下去会去做ChatServer的单元测试，希望增加更多的单元测试。
  + 自测时，建议参考测试组提供的冒烟测试用例
  + 新的技术，依赖注入这些都没用，新项目里面可以用
* **代码评审一定要做**
  + 将代码评审安排到计划中去
  + 现有两种方案，一个是在开发自测之后，提测之前做。一个是在开发完成了一个小功能后就进行。现在大家偏向于后者。
  + 改完之后，要做好检查的工作，不是做完评审就完了。由当时Review的人进行检查
  + 现在进行Review的人可能本身比较忙，要做Code Review时经常会和自己本身的项目有冲突。主要集中于ChatServer模块的评审，这个需要与William/Roy/Allon/Michael/Davy提前沟通安排好
  + 代码评审需要包含Framework, Public的改动
  + 可以考虑
    - 在TFS中开启评审之后的代码才允许签入的规则。

---这个要验证一下

* + - 使用更好的工具支持导出到PDF – @Yan可以去研究一下
* 之后的项目遵循多分支开发，主干发布的方式
* 提交测试 - 需要由项目经理发邮件，发给RandD，需要包含提测模块，Release.txt，修改了什么Bug。

--冒烟测试的时候才需要写这个邮件.

--每次发布的时候要说一下发布了哪些模块, 开发在测试环境验证一下再改成fixed, 测试看到

* 针对合并出来的问题，遵循谁开发谁合并的原则。当合并代码出现冲突时，需要确认清楚。合并完之后，需要做好充分的自测。
* 希望公司抽出一点资源来研究一些更高效的技术如.net core、redis等，首先从技术层面就公司产品最主要的一些问题提出可行的解决方案
* 完善公司公共框架相关的文档，大家都不太清楚公共模块有什么，有时候还会自己实现一次，应该要补充

---公共框架大家都能改，这是个大问题, 是否要做限制

* 可以考虑以下：
  + 代码库LiveChat下东西太多了，考虑将ChatServer, LiveChatReport分开。 @Roy
  + 用GIT代替TFS，chatserver和report新增部分数据库访问可以用轻量ORM代替原先的体系，用websocket及长轮询代替目前的轮询，专业log工具代替自写的LogHelp等
  + 每个人只看到一部分的代码，有时候修改功能时，不可见的代码没法进行修改，也没法知道细节。希望在可控的情况下，放开代码权限
  + 对Warning要像Error一样重视，中长期降到warning为0。属于编译器误报的地方加warning suppress来忽略警告。新工程的warning要为0。

# 测试

* 项目中各里程碑都需要发邮件通知到RandD组
  + 冒烟测试
    - 没有通过需要打回。打回时，发邮件通知。
  + 其他各个里程碑都需要发邮件通知，提测，冒烟测试，第一轮结束，单元测试结束，开始IT测试，进入RC，RC结束。（管理层可以选择在每个里程碑对项目进行审查）
  + 测试结束后，需要有一个测试报告，包含多少bug，各类型的统计…

测试报告的模版是什么，需要确定一下

* **测试用例**
  + 不管项目大小，都需要有测试用例
  + 需要对现有用例进行一次全面的整理
    - 删掉冗余用例，废旧用例
    - 更新过时用例
    - 补充缺少的用例
    - 标记出回归测试用例
    - 用例不能写的太大，Steps不要超过5个
    - 测试用例标题可以使用中文
    - 考虑写的更新详细一些，可执行性好一点
    - 哪些是冒烟的（优先级是1的），哪些是回归的用例（优先级是1，2），评审的时候要定一下
    - 在不理解需求的前提下单靠看测试用例就知道怎么做是很难的. 需要整体的需求文档, 每次项目结束后再更新整体文档.
  + 测试用例更新要及时，可以考虑项目RC结束或发布之后留一段时间做用例更新的工作
* 严格控制发布的版本，测试负责人和测试人员都可以把好关，不要频繁发布。

1. 可以通过项目经理用jenkins 来打包，控制发布
2. ....

* 数据库升级慢这样的问题暴露的比较晚。 -- 对于全库升级测试，需要尽早进行，可以在第二轮单元测试和IT测试的时候就进行
* 加入自动化测试，覆盖一部分的回归测试
* --自动化测试用例如何更新
* 考虑
  + 由专门的人来负责版本发布和环境管理
  + 统计每一轮测试执行的时间，可以统计不同人然后取一个平均数，以后就有参考依据
  + 根据测试用例数量来评估每一轮的测试时间
  + 统计每天的Bug的数据，Bug的分布
* Bug 处理的优先级要定义好, 大家对每个级别的bug定义不清楚 便于开发和测试确定任务的优先级

1 级: 阻塞测试流程走不通、报错，报黄页

2 级: 功能未实现，功能错误，但不影响总流程

3 级: 文字拼写, 提示，样式，有效性，唯一性验证，边界条件

4 级: 建议性的问题

* Bug 要进行分类: 样式问题、需求问题..., 方便对问题进行归类

# 计划，发布

* 计划还是安排的太紧
* 进入IT测试之后，就不要再接小需求，小Bug的修复，不要再合并其他项目
* 提前太久确定项目的上线日期不是非常合理。可以等到IT测试之后或者RC之后再确定。

---建议是第二轮测试结束以后

* + 其他部门可以先做好准备
  + 可以上线后，功能先不开启，其他部门准备好之后再开启
* 当项目负责人有变更，需要交接时，需要比较清楚明确，将可能的风险也沟通清楚。
* 一次合并太多不好，之后尽量做好一个合并一个。不要造成现在这样的合并版本
* 发布流程评审一定要做
* 灰度发布：
  + 分批分平台发布
  + 先将新功能在一部分客户上开启，然后再推广到更多的客户
* 项目结束后，要有时间留下来反思，沉淀，更新
* 发布完成后，一定要有专人留下来解决线上问题，及时响应客户的问题和反馈

---和运维组要做培训，上线最开始一段时间一起运维

* 考虑
  + 将大Feature分解成小Feature，滚动发布，借鉴Scrum，降低风险

# 人员

* 希望测试人员从一开始就参与项目，而不是需求做完了，开发都快做好了才参与。
* 成员换来换去还是太多了。
* 新人培训
  + 产品的培训需要更加仔细一些，比如数据库的培训
  + 在常规的产品培训和技能培训结束后，可以让新人参与到小功能的开发和测试中，比如线上用户提的一些小功能或者小的Bug，或者解决代码中的Warning等。
  + 新人参与项目时，需要给新人做项目相关的培训，让新人做好参与项目开发/测试的准备
  + 许多新人都提出来希望有一个针对整个产品的比较全的需求文档。现在我们的需求文档都是基于项目的，而且是分散的。

--长沙这边又有很多人多次提

* + 增加一份新人开发/测试时容易出错的集合的文档
* 部署自动化，解放负责人的时间
* 角色分工严重影响了研发的效率
  + 希望核心功能能有更多人去Cover，老员工可以有更多的时间指导新人
    - 观念上改变，然后项目经理/测试经理有意识的去做。
    - 老员工身上的工作压的太多了，将实际的开发测试工作少一些，多做一些前期的工作和指导新员工的事情
    - 安排计划的时候，可以让新老搭配去做一个事情，老的可以带一下新的。
    - 完善各方面的文档，可以让新人自主学习

# 环境，工具：

* **开发环境** - @William
  + 杭州29
    - 经常出现卡顿的情况
    - VS，Chrome经常性崩溃
  + 长沙开发服务器
    - 没有IIS权限 -以安装包中用过的功能为基础，做一个小系统去控制操作IIS -

--已经安排joe 在做了

* + - 非常缓慢，严重影响开发效率 - @William
    - 每次停电后恢复很慢 – 上UPS，- @Carl
  + 开发自测环境
    - 比较乱，大家都在用，希望能够有多套测试环境
    - 空间不足。
  + 没有外网开发环境，很多开发都非常麻烦
* **测试环境**太多，比较混乱，维护成本比较高
  + 需要专门讨论一下
  + 可以考虑虚拟机，Docker，或者在流程上规范

--Docker 只支持.net core 我们不能用

* 测试工具 - @Henry, William
  + 上次只买了两个Android手机，一个Mac Mini。测试人员表示不够，至少要开发测试分别配置一套iOS,Android,Mac。
* 测试用例系统TFS有一些地方不好用。希望换掉。
  + 可以考虑TestLink, 禅道，JIRA
  + TFS里对bug的功能记录太弱，不利于分析和统计（如状态比较少，没有产生阶段，产生原因，错误类型等选项。
* Teams不好用。经常拨不上，历史消息不好找。
  + 将Teams中的文件使用OneDrive同步到本地，这样文档编辑可以很方便
  + 当需要建立超过20个人的群时，可以用建立会议的方式
  + 可以考虑国内团队沟通使用钉钉，QQ
  + 长沙经常连不上teams, 需要改善网络

# 其他

* 跟IT部门的协调，需要Elei他们更多的支持，比如快速发布补丁，非工作时间发布
* 公司沉淀下来的东西太少了，比如异常处理/日志/权限等，几个小组的做法都不一样
  + 需要专门发邮件讨论一下 - @Kim
* 希望公司就一些核心模块如chatServer、billing等提供一些业务或实现方面的培训、指导（多人提及），Agent console,
* 长沙办公室这边还是缺少一些企业文化氛围，是不是也能弄个小书架和照片墙什么的

墙上还贴了上家公司的标语，给公司员工感觉不好.