**0.我们有一套完整的软件说明书，但是并不是一个文档，是很多个文档，可以从这些文档惊醒搭建环境，编译，以及单元测试。**

**1.你的团队的源代码控制在哪里？用的是什么系统？如何处理文件的锁定问题？**  
　　我们团队使用GitHub作为代码管理工具进行代码的管理。  
  
　　使用的win7系统。  
  
　　文件没有被锁定，所有团队内部人员都可以自由签出文件，每个人可根据自己的需要修改自己负责的模块。  
  
**2、如何看到这个文件和之前版本的差异？ 如何看到代码修改和工作项 （work item），缺陷修复 （bug fix) 的关系？**  
　　进入github团队项目仓库，点开项目的提交记录，点击文件详情，便可看到这个文件与之前版本的差异。  
  
我们在上面图片里面可以看到的是”+”标注的是在原文件的基础上增加的代码的记录，”-“标注的是在原文件的基础上删掉的代码的部分，颜色显示也不同。   
　　其实我们团队是以任务为单位和模块进行的开发，这种开发模式在任务分配之处就已经给该任务提供了描述。

1. 通常情况下，可以在git中执行合并即可实现自动合并Git修改的部分。但是，也存在被修改的部分发生冲突或无法自动合并的情况。也存在无法自动合并的情况。当修改的内容冲突时，应查看冲突内容，手工修改冲突，完成提交，然后使用git merge命令合并。通过git 版本控制工具既可完成。对于无法完成自动合并的原因在于远程数据库和本地数据库的同一个地方都发生了修改，系统无法自动判断要选择远程数据库还是选择本地数据库进行修改，所以会发生冲突。当然，git会显示本地数据库和远程数据库同一个地方的不同修改，所以我们可以根据git的显示手动解决问题。  
      
   **4.**   
     git作为一个成熟的源代码版本管理系统，可以保证git服务器远程提供的修改操作具有原子性，这样就保障了整体修改的原子性。当然，在签入之前，一定要先对比处理有冲突的文件。在本地仓库中修改的文件都要统一经过commit为新的本地版本后，再push至远程分支。  
      
   **5.** 你的PC 上有关于三个功能的修改， 但是都没有完成，有很多文件处于半完工的状态，这时你要紧急修改一个新的 bug，如何把本地修改放一边，保证在干净的环境中修改这个 bug, 并成功地签入你的修改 --- changelist management。  
     在本地新建一个分支，然后在新的分支上进行bug的修复就好。当前分支的内容就被保存在原地。

**6. 规范操作和自动化**  
  
你的团队规定开发者签入的时候要做这些事情：  
  
    - 运行单元测试，相关的代码质量测试。  
  
    - 代码复审 （要有别的员工的名字）  
  
    - 和这次签入相关的issue 编号， 任务/task， 缺陷IMG_256g 编号，等等， 以备查询。  
  
    请问你的团队有这样的自动化工具让开发者方便地一次性填入所有信息然后提交么？  （高级功能， 代码提交之后， 相关bug 的状态会改动为  “fixed”， 并且有链接指向这次签入。）  
  
答：我们团队没有使用这样的工具，我们在开发之前对编码做了规范，我们每次完成一个功能会自己测试好久，进行完单元测试后，将没有bug的代码提交Github。  
  
  
**7. 如何给你的源代码建立分支？**  
  
   上传一个与MASTER相关的分支（该分支是从MASTER中git clone 得到，相关信息在 .git 文件中）  
  
    修改后源码后，在进行如下操作  
  
1、git add .  
  
2、git commit -m "test"  （”test“为分支名）  
  
3、git branch test（创建分支）  
  
4、git checkout  test （切换分支）  
  
5、git push origin test:test

**8. 一个源文件，如何知道它的每一行都是什么时候签入的，为了什么目的签入的 （解决了哪个任务，或者哪个bug）？**  
   场景： 一个重要的软件历经几年，几个团队的开发和维护，忽然出现在某个条件下崩溃的事故， 程序员果冻经过各种debug手段，发现问题是在某一个文件中有一行代码似乎显然出了问题， 但是这个模块被很多其他模块调用，  这行代码是什么时候，为了什么目的，经过谁签入的呢？  如果贸然修改， 会不会导致其他问题呢？  怎么办？  
回答：对于我们团队来讲每一次的任务都是由同一个人负责上传源文件，他在上传时会给文件打上标签，而且在签入的时候会有commit的记录保留，所以每一次的提交都可谓是目的明确，特征显著。至于“追责”问题，github上面有每一次的提交的记录，对应着非常容易找到相关负责人。  
  
   
**9. 如何给一个系统的所有源文件都打上标签，这样别人可以同步所有有这个标签的文件版本？**  
   代码每天都在变， 有时质量变好，有时变差，我们需要一个 Last Known Good （最后稳定的好版本） 版本， 这样新员工就可以同步这个版本， 我们如果需要发布，也是从这个版本开始。  那么如何标记这个 Last Known Good 版本呢？   
回答：这个在每一次提交的时候没有特别的标记。我们团队一般会根据任务提交时的commit记录的时间推测这个版本是哪一个。还有就是我们团队都是大家讨论后，然后确定哪个版本是最好的，这样所有人就知道是哪个版本了。

**10.你的项目的源代码和测试这些代码的单元测试，以及其他测试脚本都是放在一起的么?**  是的     
修改源代码会确保相应的测试也更新么?  是的     
你的团队是否能部署自动构建的任务?能  
  在签入之前，程序员能否自动在自己的机器上运行自动测试，以保证本地修改不会影响整个软件的质量? 可以  
  在程序员提交签入之后，服务器上是否有自动测试程序，完成编译，测试，如果成功，就签入  否则,就取消签入?  没有  
   
**11.团队是否配置了服务器，它自动同步所有文件，自动构建，自动运行相关的单元测试，碰到错误能自动发邮件给团队**    尚未配置

github:非 Windows 环境下的开源项目，我认为 GitHub 是一个理想选择——特别是它有杀手级的 pull request 功能。如果是团队合作的私有项目，感觉 GitHub 的优势就没有那么明显了.  
 gitcafe.com:GitCafe是一个基于代码托管服务打造的技术协作与分享平台，程序开发爱好者们可以通过使用代码版本控制系统Git来将他们所写的开源或商业项目的代码托管在GitCafe上，与其他程序员针对这些项目在线协作开发。你可以在这里阅读到当今最流行的代码版本控制系统Git的基础介绍与使用教程。  
 VSS:Visual Source Safe 是微软搭建的一个集成小组开发环境,他可以让一堆人在一个工程中协同工作(当然是在网络环境中),而不用等每个人做完后在重新组合,Visual Source Safe 对开发小组来说是一个很好的工具！可以备份，还可以资源共享，但提交大文件容易崩溃。