课程学习过程以及学到的内容：

利用四象限法可以针对我们做的project做出合理的时间规划，根据四象限法，把我们的project进展分成四个象限，分别是重要紧急的，重要不紧急的，不重要紧急的，不重要不紧急的，如图所示：

http://note.youdao.com/noteshare?id=ec551a61eeb174400743400814b07a8a

按照“要事第一”的法则，所有任务分为四类：

重要紧急：需要尽快处理，最优先。

重要不紧急：可暂缓，但要加以足够的重视，最应该偏重做的事。

紧急不重要：不太重要，但需要尽快处理，可考虑是否安排他人。

不重要且不紧急：不重要，且也不需要尽快处理，可考虑是否不做、委派他人、或推迟。

基于我们做的project的规划为：

重要紧急：做出CanTool装置是最重要和紧急的。

重要不紧急：由于我们组做的是装置和测试，所以要等其它小组做好CanToolAPP之后才可以进行测试，故虽然是重要的但不是紧急的。

紧急不重要：实现传感器与CanToolAPP之间信息的交流是紧急不重要的。

不重要且不紧急：CanToolAPP不是我们的任务，所以是不重要且不紧急的。

个人觉得自己要完成的任务属于第二象限的内容，在组长完成CanTool装置的设计之后，实现与下位机的信息的交流。

#CanTool项目需求分析：

1、CanTool装置与上位机通过USB串口或蓝牙RFComm实现UART串口通信。

2、上位机与CanTool装置之间的信息传送方式使用ASCII码格式+ \r（即0x0d）方式进行信息交换。

3、CanTool装置接收CanToolApp发送的CAN信息并显示并发送到CAN总线。

4.进行单元测试－功能测试－集成测试－场景测试－系统测试－Alpha/Beta Test。

5.以Arduino微控制器为基础设计CanTool硬件装置，实现CanTool装置信号的发送和接收。

招数：设计变更（Design change Request）

经过了Alpha/Beta阶段，移山团队就收到了不少用户的反馈，原来的设计就有不少要改进的地方。此外，有了用户反馈，大家也有了很多的新想法于是就需要做设计的变更，即DCR。

重构—在尽量保持原有界面的基础上优化部分代码。

重写—重新实现原有功能，同时，要分清是全部重写原有功能，还是偷偷加上许多新的功能（Feature Sneak）

DCR的过程：

1、如何提出DCR？

a.问题在哪里，问题的影响；b.如果不修改，会有什么后果？

c.几种修改方案，各种方案的优缺点和成本。

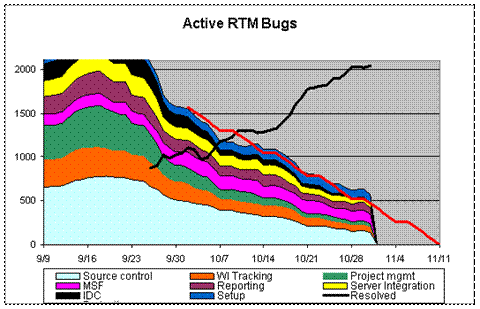
2、如何决定DCR的执行次序？

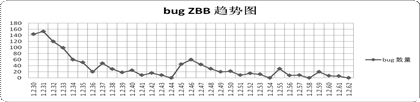
1）会诊所有DCR。

2）按照影响、成本排序，得到一个自上而下的名单，根据现有资源，按照名单执行。

招数：ZBB

ZBB=Zero Bug Build，顾名思义，就是要把这一版本的Bug都解决掉。左图是一个200人的团队的“预想ZBB 进军图”。每个小组的Bug数量累加起来，就是团队的Bug总量，图中的黑线表明修复的Bug总量。右图表示随着每次ZBB 过程中质量的加强，Bug 的数目会越来越少，并经过几次反弹，逐渐到0的情况。



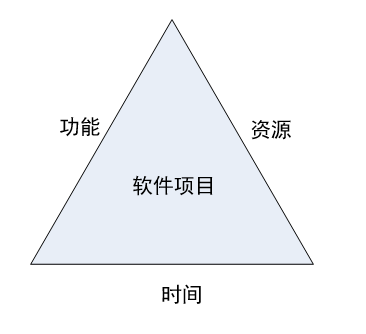


招数：最后回归测试

项目临近结束时，所有的人员（开发、管理、测试）都要回归测试所有的Bug。回归测试是指重复以前的全部或部分的相同测试，每个人都要帮助团队确保这些Bug的确被修复了，而且别的更改没有导致功能的“回归”。

招数：砍掉功能

如果有一个模块看来不能实现预期的设计需求，时间快到了，怎么办？回顾第九章的功能/资源/时间的平衡图, 我们要不断保持这些因素的平衡，所以这时候就需要把这一模块的功能砍掉，避免时间或者金钱等成本的浪费。



招数：修复Bug的门槛逐渐提高

在Alpha，如果开发人员拿到了一个Bug，那他/她就可以马上拿去修复，之后告知大家做了什么样的修改就可以了。这个做法叫作Tell Mode。

在Beta阶段，在修改之前先要告诉会诊小组这个修改潜在的风险是什么，如何应对，这个做法称为Ask Mode，即请求别人同意你修改代码，没同意之前，不能动代码。

在RC阶段，开发人员在拿到Bug进行修复工作之前，就要和会诊小组沟通，看看这个Bug是否值得花时间。

招数：逐步冻结

随着程序功能的完善，功能进入稳定阶段，我们就可以让程序的各个方面有次序地“冻结”，这样才能把稳定的软件交付给用户。