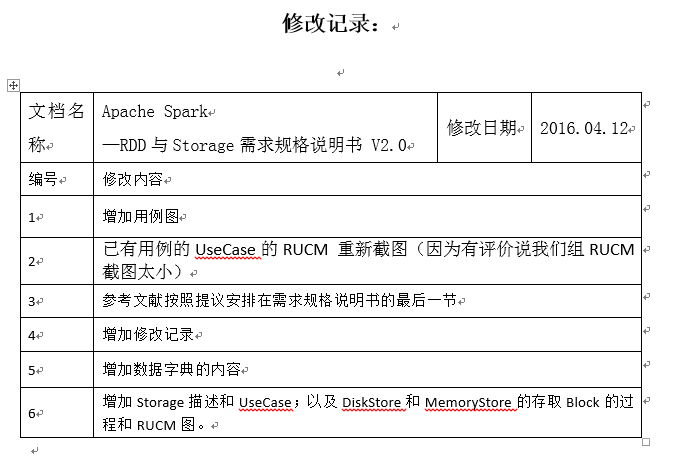
实验方法总结

1. 创建各文档间的追踪关系是必不可少的。建议可通过下面的方法建立追踪关系:
2. 建议使用表格或其它可视性强的描述方式（如绘图）等描述
3. 对同一文档不同版本的变更，要追踪其变更原因（如评审意见、个人对系统认识的增加等），变更位置（可细化到修改了文档哪一节哪一段的内容），变更结果（如修改为了什么，产出了什么文档）

例如在文档中添加修改记录：



或者单独撰写问题修改记录：



1. 对不同实验阶段文档间的关系要建立追踪。如软件测试过程需编制的测试规格说明书及其中测试用例的设计均要依赖于系统需求规格说明书。细化来说，要建立每个测试用例具体涉及到了系统需求规格说明书中的功能性/非功能性需求模块。

例如：



1. 对于同一阶段下产生的各文档间的前后关系要建立追踪表。举例来说，在测试阶段我们会产生测试规格说明书、软件测试计划、软件测试记录和问题报告。对这四个文档，可针对这四个文档的先后产出关系建立合适的追踪。

通过这种方式，我们可以从建立的追踪关系表明确同一文档在不同时间阶段的修改细节，可以清楚得理出各文档间的依赖关系，从而也有助于之后的维护。

1. 养成记录问题整理评审意见的习惯，往往会对我们实验的进行大有裨益。

建议可通过以下方法进行问题梳理:

1. 对收集到的评审意见依据其面向的评审对象分类，如可分为针对文档格式的评审、针对文档内容描述的评审及针对实验细节的评审
2. 对各评审意见对应列出其意见提出者/组，方便后续交流学习。
3. 填写出现各评审问题的原因
4. 对评审意见的受理状态分为以下几类:接受、拒绝接受、待评审和受评审双方进一步交流。
5. 描述具体的解决方案，包括方案的描述、具体实施负责人、实施记录明细。
6. 进行总结性论述，有助于发现根本原因和经验积累。
7. 在开展工作之前，首先要对工作内容的难度、每个人的个人能力以及整个团队的整体能力进行全面、认真的评估。
8. 搜集、查阅资料，对项目相关的领域有充分的了解和认知，充分衡量项目的难度和能力要求高度。我们组的项目需求阶段，对Spark的相关知识查阅很多，也找出了相关的缺陷，但是对优化Spark软件工作的难度衡量不够准确，导致刚开始的时候方向略微走偏。
9. 个人能力与工作内容的匹配：分析每个人的兴趣点、关注点以及学习、项目经验，并与工作内容的不同方面的特性、所要求的人员的能力或者特性进行匹配。
10. 多听取他人的意见，包括其他项目组的同学的意见、老师的意见、同领域研究人员的看法；对意见要多加考虑，不能一概否认，也不能全盘接受。
11. 即时发现问题，并且尽早改进
12. 所有从未动手的知识、认知对项目工作的了解、纸上谈兵都是不够深入的。必须动手尝试。
13. 在测试用例书写、测试报告、问题报告撰写的过程中，需要注意如下问题：
14. 测试用例书写应该从开始就依据需求用例来写，首先，这样的追踪关系很明确，不要后续重新建立追踪关系；其次，不会存在一些需求没有被测试到。
15. 所有测试用例中步骤以及结果描述用语应该力求准确，避免在测试过程中对测试结果是否通过无法进行明确判断，问题报告也不好书写。
16. 在测试中发现问题，并且对软件进行修改之后，有时候需要对相关测试内容进行多次重复测试和回归测试。
17. GitHub使用经验：把GitHub被当成作业提交平台，其作用就没办法发挥出来。所以，需要加强对GitHub的使用，并且使用readme文档对项目的更新情况进行阐述和记录。