软工实验总结

Redis

作者：陈志伟

2015/6/6

1. 项目总结

1.1. 系统概述

Redis 是 NoSQL 系列中的一种 key-value 类型的轻量级内存数据库，全名为远程字典服务(REmote DIctionary Server)。它是由 Salvatore Sanfilippo 使用 ANSI C 语言编写的，支持网络、可基于内存亦可持久化的开源日志型、Key-Value 数据库，并提供多种语言的 API。从 2013 年 3 月 15 日起，Redis 的开发工作由 VMware 主持。从 2013 年 5 月开始，Redis 的开发由 Pivotal 赞助。

Redis 定位于一个内存数据库，但事实是其并不是将所有的数据都存储在内存中，其持久性体现在在硬盘上进行写操作，它不仅仅是一种简单的 key-value 存储。Redis 支持存储的 value 类型有 string(字符串)、lists(链表)、sets (集合)、zsets(有序集合)和 hash 等。这些数据类型均支持 push、pop、add、remove，取交集、并集和差集及更丰富的操作，且这些操作都是原子性的。它支持各种不同方式的排序，为保证效率，它的数据都保存在内存中，但是 Redis 会周期性地把更新的数据写入磁盘或把修改操作写入追加的记录文件，并且在此基础上实现了 master-slave 同步。因为是纯内存操作，Redis 的性能非常出色,每秒可以处理超 10 万次读写操作，是已知性能最快的 Key-Value 数据库。

1.2. 项目计划

古语亦云：谋定而动。“谋”就是做计划，也就是做任何事情之前，都要先计划清楚。软件工程项目更是如此，有人说软工项目就是制定计划，执行计划，监控计划的过程。项目管理泰斗科兹纳更是一针见血：不做计划的好处，就是不用成天煎熬的监控计划的执行情况，直接面临突如其来的失败与痛苦。可见项目计划在项目管理中的重要性。

在实验中，按照老师的建议，经过多次的修改，最终我们在 MS Projec 中完成了整个项目的计划。通过及时地记录每周项目实际完成情况，找到了项目实践中的出现的一些问题，如计划时间不合理，没有留下处理意外情况的时间等，并在后续项目中积极改正，使整个达到更好的效果。做项目计划确实是一件艰难又耗时的事，但一段完成它对整个项目的监督和促进作用将是巨大的。

总结一下项目计划阶段，我们得出的经验如下：

（1）准备筹划工作要到位，分析要精准；

（2）对想法进行测试快速实验，可行的继续执行；

（3）执行时记录下工作的重要细节；

（4）具体方案出来后，画出工作流程和需要做的细节；

（5）对流程图中可能存在优化空间的流程进行再优化；

（6）后期不断执行，不断得出新方案，对旧方案进行替换修改优化。

最后，对于每一个项目的分析一定要做到充足和充分，考虑到项目实行过程中可能遇到的所有问题，并且一定在写之前要考虑可行性，做好流程的记录和细节的优化。

1.3. 需求分析

需求分析是指对要解决的问题进行详细的分析，弄清楚问题的要求，包括需要输入什么数据，要得到什么结果，最后应输出什么。需求分析是软件工程中的一个关键过程。在这个过程中，系统分析员和软件工程师确定顾客的需要。只有在确定了这些需要后，他们才能够分析和寻求新系统的解决方法。需求分析阶段的任务是确定软件系统功能。

在实验中，我们并不是按照标准定义一样，针对一个要解决的新问题给出其功能定义，而是对一个已有的具有一定规模的开源软件，从其源码和使用说明书出发反过来推出其需求定义，这是一个逆向的分析过程。由于源代码等其它资料已经给定，我们也不用经历最恐怖的需求反复变更过程。即便如此，我们在实际进行需求分析时依旧是困难重重，因为这个阶段也依旧包含了太多需要学习的东西。

需要阅读开源软件的源代码，但这并不简单。那些复杂的上下调用函数关系可以把人绕晕，这时就需要冷静下来看看其他的辅助资料和图片。理解一个软件是困难的，特别是深入了源码级别之后，但一旦闯过了这个过程，就会发现这个软件的美妙之处尽收眼底。更重要的是，在这个过程中所培养出的阅读源码的能力对我们以后的工作大有裨益。

掌握逆向的获取需求的方法，利用 RUCM 工具进行需求描述，并且学习编写需求规格说明书。需求获取阶段出现的最有争议的问题，就是用例图和对应规格中的 actor 的确定。由于分析过程主要基于的是源码，很多时候我们所给出的用例及其对应的规格描述是系统内部的功能，而不是外界的用户需要系统提供的功能，这

就造成了内部系统功能所对应的用例常常找不到明显的 actor，于是便把其命名为“系统”，但这个“系统”不符合软件工程中对 actor 的定义，这就成为了错误。我们小组在这一点上，主要是使用活动图或状态图来代替用例图来描述系统的内部功能。效果也很不错。

最后，通过需求分析阶段的实验，我们应该了解怎样才能清楚的描述出一个系统的需求，注意在需求中需要描述一个系统的哪些方面，在需求建模的过程中发现容易犯的错误，积累经验。

1.4. 测试阶段

软件测试是使用人工操作或者软件自动运行的方式来检验它是否满足规定的需求或弄清预期结果与实际结果之间的差别的过程。它是帮助识别开发完成（中间或最终的版本）的计算机软件（整体或部分）的正确度(correctness) 、完全度(completeness)和质量(quality)的软件过程，是 SQA(software quality assurance)的重要子域。

小组在测试阶段完成的工作有编写测试需求规格说明书，设计测试用例和进行测试等。

在测试需求规格说明书的编写中，主要的错误是没有审清要求加入了测试的结果，因而把其写成了测试规格说明书，经过修改后及时的更正了错误。同时，在性能测试这一方面由于进行的实验力度不够大，得出的结果不够明显，按照老师的建议重新进行了实验设计。

RUCM4test 中的测试用例描述有 5 部分的内容：基本信息、setup、基本流、分支流、Oracle verification flow。在得到所有的需求用例后，通过分析各个用例的前置条件、后置条件、基本流、分支流，对应的分析设计测试用例中的 setup，test oracle 和 测试用例的基本流和分支流。

按照测试用例的要求，编写了测试脚本。测试员按照测试用例中规定的流程对特定的测试结果进行检查和验证。由于测试环境的限制，在实际中也不可能完全按照测试用例的要求来进行测试，同时测试过程中也会出现一些超过预期的事情，需要灵活的进行处理。测试最终要达到的效果是：在系统满足前置条件的情况下，用户发出的请求后，系统能够正确执行需求流程中的要求，并且满足后置条件，实现功能的正确执行。如果测试在正常执行的情况下，测试结果与预期结果相同，则说明测试通过。

1.5. 评审阶段

如果有人问我，经过一学期的软件工程实验的学习，你觉得哪一阶段对你提升最大？我一定毫不犹豫的回答：评审阶段，这一阶段包括课堂上由老师主持的评审和网络上不同小组之间的互评。

每周课堂上由每个小组轮流进行演讲，然后由老师和同学指出问题，这一过程简单、粗暴、直指要害；被评审的同学还能做出解释，形成了一个良好的交流互动环境，但受限于课堂时间的限制，每个小组所能展示的只能是最有代表性的部分，其它部分的隐含的优点和缺点还需要我们利用额外的时间去发掘。网络上小组的互评却可以很好的弥补这一缺憾。

在网络上小组互评阶段，我们可以接触到每个小组上传的文档，通过横向比较每个小组的成果，可以清楚的看到其他人文档中的优点，如有的文档格式规范、描述清晰，同时对于缺点，如缺少引用文献、废话太多等，可以引以为戒。最后，给出评审意见，需要一定的洞察力，并评出最优秀的文档，也是对其他人工作的肯定。

现在还是觉得大家对评审阶段的重视不够，导致我们很多时候只是做完了就交差。后面即使被指出了问题，在缺少一种强有力的追踪机制下，由于人的惰性，我们很少会进行文档的再次修改，自然也缺少了一个提升自己的机会，比较可惜！

2. 工作总结

2.1. 收获体会

在这次的软工综合实验中，我负责的主要是服务器模块及项目管理。对于服务器模块的工作，完成了从需求分析到软件测试这一整套过程，使我的软工建模和分析能力得到了极大的锻炼。就完成的质量来说，个人认为已经尽力做到最好，完成时较为认真，撰写的文档质量也较高。虽然也会出现一些小问题，但在老师和同学的监督和指出后，会及时的改正。总之，基准的实验任务自己做的还行，这一点可以从工作量统计表中看出。

对于项目管理，这是我作为组长应该完成的任务。老师每周布置任务后，我把老师的任务分解成小的任务，然后考虑每个组员的能力和个性，把任务合理的分给他们，并给出统一的格式模板，最后定出完成的截至时间。当然还得监督组员的完成情况，并把每个人完成的任务汇总。

项目管理的工作比较琐碎，但却比较花时间，面临的最头疼的问题是：如何调动组员的积极性？既要在规定的时间内未完成任务时，给予他们一定的惩罚，但惩罚太轻起不到作用，同时惩罚也不能过重，否则会严重的伤害大家的积极性，影响后续实验任务的进行，这是一个艰难的过程。总的来说，对于基准实验任务，自己还是较为“仁慈”的，给出的记录比较保守，但对于部分同学的过多的额外付出也会有相应的奖励，于是便有了额外工作量一项，它占总贡献率 15%。

最后，我想在每次的小组合作中，都存在能做事的人，同时也有浑水摸鱼之辈。对于尽心尽力的人给予他们一定的奖励，让他们再创辉煌，否则每个人都只是应付，而若所有小组完成的任务质量都很低，那么整个软工实验从某种意义上来说就是失败的。有时也经常觉得自己不喜欢这种小组完成任务的方式，若由自己来做肯定比他们做的都好，也不用再被各种麻烦缠身。但老师说的也有道理，个人能力再强也不可能在未来由你一个人独自完成一个项目，你一定需要和人合作甚至会管理别人，所以这次实验对我来说也是一次良好的锻炼我管理与合作能力的机会。

2.2. 个人建议

总之，这学期的软工实验虽然也有一些小的瑕疵，但总体来说还是非常成功的，达到了预期的目的。最后也想提出一些建议：

（1）合理划分实验每一阶段的时间。这在需求分析阶段表现的最为突出，当时只有一周的时间，需要我们对一个刚拿到手的软件，完成需求分析、建模和需求规格说明书的撰写的整个过程，十分的困难，特别是对于上学期没有上过高级软件工程课程的同学来说，他们还需额外学习 RUCM 的相关知识。这导致了很多同学其实根本没有认真分析软件，而是通过其它资料和想象完成了这一部分。所以，建议老师在以后的实验中，一定要在开始放慢节奏，多和同学沟通，给大家充足的时间学习新的知识和理解软件，只有开始准备充足才能为后面的实验进行奠定一个好的基调。

（2）重视评审阶段。评审阶段是项目总结、反馈和提高的最好时机，利用好了这一阶段才能事倍功半。

（3）小组管理。这主要是结合自己在本次实验中的亲身经历，所提出的一点想法。回想起实验开始之际，大家一致选我为组长，虽然知道组长会很辛苦，但我也没推脱。但在后面的任务中，自己每次布置下去的任务，根本没法控制大家做或不做、什么时候做和做得如何，因为你若催促或追问组员，他们最经典的回答是“这周好忙，完全没时间做”。即使他们最后侥幸做了，也是临时抱佛脚随便赶出来的，和那些认真做的同学相比质量差太多，这也会影响整个团队的水平，间接损害其他人的利益。这一切的根源我觉得在于组长没有什么权利，自然没法约束组员。

所以，我建议以后的实验中，每个小组合力推选出一位组长，他负责整个小组的任务完成情况，当小组任务未完成时，他就得买单，其成绩可由老师结合整个小组最后的完成情况给出。每个组员贡献率由组长依据其表现给出，同时为了防止组长乱用职权，若一组内有一半以上的成员反对组长的统治，则可选出新的组长等。这里只是我建议的一种方式，还有更多好的方式值得我们去思考。

最后，祝大家梦想成真！