**摘要:** 2010年8月，我参加了公司一个对原有产品进行升级的项目，担任项目经理一职。 这个项目是在原有的产品上进行升级，系统架构由二层改为三层，并在功能上对原有系统进行扩充。同时考虑到以前我们做开发时，软件过程不规范，原有系统在设计、开发、维护、实施上存在一些弊端，开发出来的软件总存在这样那样的问题。我们在这个项目进行过程中，对软件过程进行了改进。在这个项目里，我们努力做好项目规划、加强版本控制，采取了一些质量控制手段，如同行评审、专家评审、加强测试等，使软件的质量得到保证，同时也降低了开发成本。现在看来，这个项目在一定程度上是取得了成功，但还是存在一些问题，如采用工具方面，过程改进制度化方面，过程标准化与工程进度方面的矛盾等都有不如意的地方，后来公司都对这些问题有所认识，也在不断地进行改进，公司的软件过程已日渐成熟，并于去年顺利地通过了CMM3认证。

软件过程的改进是一项需要长期不断地进行下去的工作。以我们公司来说，作为一家软件公司，过去对软件的开发过程并不重视，接到项目后首先考虑的是如何用尽量短的时间来完成，而忽略了软件的质量，结果也往往是欲速则不达，甚至因为产品的不成熟使工期一拖再拖。 在2010年8月，我们要对公司原有的《城镇职工基本医疗保险》产品进行升级。这个产品是为了配合国家的公费医疗制改革，由公费医疗改为基本医疗保险而开发的。产品由各地劳动部门下属的医疗保险管理中心组织建设，中心机房设在医保中心，通过城域网连接到各定点医疗机构（医院和药店），参保职工可持医保卡在定点医疗机构直接消费，由医疗机构为病人垫付费用，事后再与医保中心结算。  
原产品为两层C/S结构，各定点医疗机构的前台为脱机消费，定时传送数据。这次产品升级首先将系统架构由原来的两层C/S升级为三层C/S，中间层采用比较流行的Weblogic或SilverStream做应用服务器，用J2EE实现业务逻辑，并采用标准的程序接口与各前台程序连接，以方便与第三方软件与我们的系统集成。网络传输也由原来的定时传送改为实时连接，软件功能上也需要突破由于原产品设计时没考虑周全而带来的限制，如原系统为实现脱机消费，采用了IC卡带金方式，但系统设计时没有考虑到IC卡本身可能存在质量问题，而经常发生医保个人帐户透支问题，在新系统中我们就采用了个人帐户在医保中心统一管理，IC卡只是作为一个身份识别的工具来解决这个问题。因此在项目起动时，对这个产品的定位就是一个全新的产品，而不是对原有产品进行修改。在这个项目里,我担任了项目经理一职。   
一、做好项目规划   
在项目的规划阶段，我们意识到公司原有的软件过程存在很大的弊端，首先，原来的软件过程中，设计与开发职责不分，甚至存在分析、设计、开发、测试全由一个人承担的做法，这样做不但是对人力资源的浪费，同时软件质量也得不到保证。开发和测试由一人承担，不利于测试出软件中存在的错误，整个过程由一个人来做，做出来的软件究竟对不对，没有一个说法，只有到最后程序拿给用户去用时问题才能暴露出来。再者在这样的过程中，开发人员往往会忽略文档的重要性，这对后期的维护也会带来一些问题。针对这一点，我们首先将项目组分为设计、开发、测试三个组，设计和开发组由系统总设计师负责，测试组有一个专门的组长。设计组负责软件的分析和设计，形成设计文档，设计文档首先要做同行评审，评审内容一般是文档的规范性以及对开发人员的指导性方面，同行评审后由系统总设计师来做专家评审，评审的内容是设计是否符合业务需求。开发组负责根据设计人员的设计文档编写出代码，代码编写出来后要通过同行评审，评审内容是代码的编写是否符合编码规范、是否具有可读性和可维护性。测试组负责根据需求和设计文档编写测试用例，并对开发出来的代码进行测试。通过这样的改进，我们充分调动了各员工的积极性，也明确了各自的责任，使得整个过程处于受控状态。  
二、加强版本控制   
在原来的软件过程中，我们对软件的版本控制不严密，没有采用必要的工具，而是完全由版本控制员手工进行操作，且版本控制员还要兼一部分开发任务。在这种情况下，版本控制经常出问题，有时同一代码被不同的人员同时修改，有时将本应发给甲用户的程序发给了乙用户，又或者开发人员自以为自己手上的代码是最新的，而出现已改过的BUG又重复出现的现象。这样做的另一个问题是版本的历史很难追踪，由什么人在什么时候做了什么样的修改完全没法掌握。在这个项目里，我们意识到这一点，首先，设立了专门的版本控制人员，同时使用了ClearCase版本控制软件，所有对文档和代码的修改必须先从版本控制服务器上Check Out，改完后再Check In。这样做就杜绝了版本的覆盖问题，而且版本历史也是一目了然，任何修改都会形成日志，这也为问题责任的追究提供了依据。   
三、加强测试工作   
在这个项目里，我们特别加强了测试人员的作用。在这之前，公司也设立过测试部，但由于存在部门之间的沟通问题，测试部很难参与到项目中来，即使参与进来也发挥不了应有的作用，测试部曾一度被撒消。这一次参与测试的是新成立的测试部，而测试人员加入到项目组，业务上测试组是受项目经理领导，人事上仍受测试部领导并考核。这样做，首先消除了测试与开发之间的沟通隔阂，而测试人员也少了其他项目的打扰，可以专心只为一个项目做测试。而以前出现的因部门间隔不让测试人员参与直接由开发人员自己测试的情况也就不存在了。 由于以前的软件过程存成那么多的问题，使我们的产品不是一个成熟的产品，不成熟的产品后期施工的成本是很高的，因为存在太多的问题，维护人员要做大量的维护，而前期开发并没有留下什么文档，也给后期的维护带来很多困难，维护人员每修改一段代码首先需要读懂原来的程序，往往读不懂时就直接在原来的程序上加上一段通过设置条件来跳过原来的代码，这样使得程序越来越难读懂，问题也越改越多。这样的产品拿到一个点去施工时往往需要二个月甚至更长的时间。  
在这次的升级中，由于采用了较好的软件过程，产品的成熟度得到了很大的提高，而设计文档也是我们这一次重点控制的对象。这样的产品为后期的施工提供了很好的条件，现在，产品在一个点的实施时间可以缩短到四十天以内，大大地减少了施工成本。 而好的设计文档也为产品的本地化修改提供了好的条件，维护人员读懂设计文档比读懂程序要容易得多，在这样的基础上做修改出现的问题也越来越少。   
尽管在这个项目里我们做了这么多的改进，但也存在不少的问题，首先我们使用的ClearCase版本控制软件存在问题，这个软件要求所有开发人员将自己的机器加入到由服务器控制的域里，否则，就只能取到版本快照而不能进行版本更新。由于这样做，域管理员具有比本机超级用户更高的权力来控制每台机器，使得开发人员不愿意这样做，于是出现了多人用服务器超级用户远程控制服务器来取版本的现象，使得版本的责任追究出现问题。而我们使用的ClearCase版本不支持Windows XP，也使这个版本控制软件的使用出现了问题。 另外，我们的软件过程制度化方面也没做好，在项目的早期，各项工作流程都被很好的执行，各种文档也非常完整。由于我们这一次的升级只是针对的整个产品的一个部分进行的，在这之后我们又对这个产品进行了一次更大的升级，使得我们的产品能覆盖更大的范围。但后面的这次升级由于规模比这一次大，人员也大量的增加了。而新加入进来的人员并没有很好地进行规范培训，好的软件过程标准也没有形成有效的制度，再加上项目工期非常紧，包括同行评审、专家评审这样的流程都开始有些流于形式甚至被忽略。开发组编码时也没有完全按制定的规范进行。因此，产品质量上就出来了一些反复。我们这个产品是个可分可合的产品。在后来的产品实施上出现了这样一种情况：如果一个点只实施前一次升级的那部分，施工难度很小，能在短期内完工，本地化开发工作也很好完成。而要全面实施整个产品的话，工期就会被拖得很长，本地化开发工作也存在很大的问题。   
针对出现的这种情况，我们公司意识到了软件过程改进的重要性，针对版本控制软件问题，我们改用了功能虽然没有ClearCase强，但更适合于我们的VSS。而在制度化方面，更是下了大力气，从印度请来了专家为我们的改进做参谋，现在公司的情况已有很大改观，各项制度已不再流于形式，而公司更是在去年年底顺利地通过了CMM3的认证。软件过程改进的路还很长，但有一点是不变的，只有通过软件过程的改进，我们的产品才能不断地走向成熟。也只有产品成熟了，我们才能在竞争中永远立于不败之地。