**摘要:** 本文以航天嵌入式实时软件测试的过程，讨论了项目开发中测试所使用的方法和工具。在该项目中，我担任测试经理工作。嵌入式实时软件因其特殊性，在测试时有一定的困难。目前国内和军队已就软件产品的质量要求及其测试制定了多项标准，但缺乏符合国内标准的本土化的测试工具，因而在工具的选取上，主要考虑的是符合国际上航天或军用行业标准的测试工具，及软件测试工具增强软件测试的自动化程度，和对文档支持的自动化程度。在编码过程中，我们选用PC-Lint作为代码静态分析工具，辅助编译器进行代码审查。在进行单元和集成测试时，我们使用了LDRA公司的Testbed/TBrun工具。我认为，提高测试技术并不单指使用了测试工具进行测试，测试工具的使用可以提高效率，但过多的强调工具的使用反而会使项目陷入困境。

众所周知，在航天领域安全是最关键的问题，软件的微小瑕疵就可能造成天文数字的巨额财产损失，甚至对国家安全造成严重威胁。这就使保证软件的质量，保证软件的高度可靠性，面临巨大的挑战。 我们单位是航天科技集团某院下属的一个研究所的软件中心，主要从事航天器上的嵌入式实时软件的研制。我们的软件是典型的嵌入式强实时软件，基于桌面PC为宿主机，x86系列的386EX为目标机。软件计算量大，任务间时序要求严格，逻辑条件控制复杂，属于大型嵌入式软件。随着软件工程思想的引入，CMM和RUP模型的推行，测试技术和测试方法也逐渐被重视起来。近年我一直从事软件测试的工作，在项目中担任测试经理角色一职，主要负责测试计划的制定、编写测试用例和组织软件测试。   
当前我们大部分的测试还处于半手工的方式，没有形成一套自动化系统的软件测试标准和规范。在软件审查中测试用例简单，不全面。为了尽可能多地找出程序中的错误，生产出高质量的软件产品，我把测试工作重点放在加强对测试的组织和管理上。 按照软件工程的思想，中间过程产品的高质量，才能导致最终产品的高质量，软件开发过程中越到后期发现的错误，需要修复而付出的代价就越高。为了确保软件的质量，较理想的做法是对软件的开发过程，按软件工程各阶段形成的结果，分别进行严格的测试。保证每个开发过程以最小的缺陷进入下一阶段的开发。为此我们为测试活动的定义了以下的过程：制定测试计划、设计用例、实施测试、执行测试、评估测试。并使这些过程协同作用、互相促进，主要目标是在设定的条件限制下，尽可能发现和排除软件缺陷。  
在组织软件测试上，我们定义了以下的规程。 1、开发人员负责自己开发模块的代码静态分析； 2、开发人员组成小组，相互之间进行代码检视； 3、开发人员负责自己开发模块的单元测试； 4、开发部内部集成测试； 5、独立的测试部门进行系统集成测试； 6、系统测试，较全面的对最终目标系统进行功能验证，做尽可能多地覆盖； 在具体测试过程中，我们分阶段分步骤地采用了一些先进的工具和方法策略。为我们测试成功奠定了基础。 目前国内和军队已就软件产品的质量要求及其测试制定了多项标准，但缺乏符合国内标准的本土化的测试工具，因而在工具的选取上，主要考虑的是符合国际上航天或军用行业标准的测试工具，及软件测试工具增强软件测试的自动化程度，和对文档支持的自动化程度。   
在编码过程中，我们选用PC-Lint作为代码静态分析工具，辅助编译器进行代码审查。通过PC-Lint我们能发现一般的语法错误，检查出那些通过了编译，虽然符合语法要求但不易发现的潜在错误。如没有被适当检验的数组下标，未初始化的变量，使用空指针，冗余的代码等等。PC-Lint可以添加用户自定义的检查规则，用来适应新的编码规范。此外还支持命令行和MAKEFILE方式，可以集成到UltraEdit和Tornado2中开发环境中，或固化到软件开发测试流程中去，方便开发人员的使用。功能强大，易于使用，性价比高是我们选择PC-Lint的主要原因。我们发现利用PC-Lint进行代码静态分析，在程序动态测试之前发现编码错误，可以防止低级错误引入到后期的开发中。通过实践证明这样可以有效的降低软件除错的成本。   
在进行单元和集成测试时，我们使用了LDRA公司的Testbed/TBrun工具。我们测试的是航天器的控制软件，语句逻辑控制结构复杂，进行白盒测试时，测试的输入量大，是一件十分枯燥且重复性的工作。通过TBrun提供的灵活的测试数据输入方法，测试数据可按范围、最大、最小及等步长或不等步长方式输入，实现了自动化的环境建立和测试输入。TBrun还提供了可视化管理用例的方法，我们发现这些方法极大的提高了软件测试人员的积极性，使测试效率有了大幅度提高。 在测试中我们会常常遇到这样一个问题：在测试初期，一般主要针对功能点语句设计用例，以检查所期望的功能是否已经实现，这时覆盖率迅速增加，但一般达到70%左右的覆盖率后，要再提高覆盖率是十分困难的，因为新的测试往往覆盖了相同的测试路径。这时需要对测试策略做一些改变：从功能性测试转向结构化测试。也就是说，针对没有执行过的路径，构造适当的用例来覆盖。Testbed有效的解决了这个问题，Testbed通过提供调用图与控制流程图，显示被测系统的调用关系及每个子程序的控制流程，在测试阶段，实时显示测试覆盖率，提供检查分析关键因素和优化测试路径的必要信息，帮助用户分析未测试的代码，实现以最少测试用例即可达到覆盖率要求。 Testbed支持自动隔离测试，可自动产生驱动和桩模块。在我们按组合集成策略进行渐增式集成测试时，在程序结构的高层使用自顶向下的策略，在下面的较低层试用自底向上的策略时，自动产生桩模块和驱动模块，让我们把主要精力放在了测试用例的设计上。此外，被测代码修改后对测试用例自动验证，使得回归测试非常的轻松。  
实践证明了好的测试工具对于激励测试人员的积极性和创造性，提高测试效率是非常有效的。 嵌入式实时系统的性能测试，我们使用了CodeTest。CodeTest在真实软硬件环境下使用非采样性测试，可以精确计算出每个函数或任务的执行时间或间隔，所以其测试具有很高的可靠性。我们仿真实时系统并按照外部事件的序列检查其行为，使用类似于等价类划分的技术，对事件如中断，控制信号和数据分类测试，并在测试了每种事件后，以随机顺序和概率将事件传给系统，检查系统行为是否有行为错误。实现了以往难以进行的系统行为测试。 在隔离了任务内部和系统行为错误后，我们用不同的数据率和处理负载来测试系统中通信的同步和互斥任务，确定任务时序是否正确，根据CodeTest测出的时间作为对应用程序优化的依据，使开发人员可以有针对性地优化某些关键性地模块或任务，以及改善整个软件的总体性能。  
总之，软件测试方法和工具的使用，可以提高测试技术水平，对软件质量保证起很大的作用。但是提高测试技术并不单指使用了测试工具进行测试，测试工具的使用可以提高效率，但过多的强调工具的使用反而会使项目陷入困境。通过这次软件测试，对我启示很大。觉得目前我们的测试活动中存在以下的不足和可改进措施。 让软件测试走向规范，应用过程方法和系统方法来建立软件测试管理体系。其主要目标是让测试活动中的过程协同作用、互相促进，从而使它们的总体作用大于各过程作用之和。 缺乏对测试工作进行度量，对测试效果进行评估，原因是缺少一个有效的度量数据收集和分析机制。对测试进行度量和评估会极大的促进软件过程改进的效果。