软件验证的一般原则;工业和FDA工作人员的最终指南

General Principles of Software Validation - Final Guidance

文件发布于：2002年1月11日

本文件取代1997年6月9日的草案文件“软件验证的一般原则，1.1版”。

美国卫生与人类服务部

食品和药品管理中心设备和

放射健康中心生物制品评估和研究

前言

公众意见

可随时提交评论和建议，供原子能机构审议，管理系统和政策司，人力资源和管理事务厅，食品和药物管理局，5630 Fishers Lane，1061室（HFA-305），Rockville，MD，20852。提交评论时，请参阅本指导文件的确切标题。在下次修订或更新文件之前，原子能机构不得对评论采取行动。

有关使用或解释涉及设备和放射健康中心（CDRH）的本指南的问题，请致电(301)594-4659或发送电子邮件至John F. Murray。jfm@cdrh.fda.gov

有关使用或解释涉及生物制品评估和研究中心（CBER）的本指南的问题，请联系Jerome Davis，电话：（301）827-6220或发送电子邮件至davis@cber.fda.gov。

额外的副本

小精灵
其他副本可从以下网址获得： 小精灵 DeviceRegulationandGuidance / GuidanceDocuments / UCM085281.htm。
您也可以发送电子邮件请求 dsmica@fda.hhs.gov 接收指南的电子副本或发送传真请求至301-847-8149以获得硬拷贝。请使用凭证编号（938）来确定您要求的指导。

克伯
其他副本可从以下网址获得：http://www.fda.gov/cber/guidelines.htm, 写信给CBER，传播，培训和制造商协助办公室（HFM-40），1401 Rockville Pike，Rockville，Maryland 20852-1448，或致电1-800-835-5709或301-827-1800。

目录

软件验证的一般原则

本文档旨在提供指导。它代表了原子能机构目前对该主题的看法。它不会为任何人创建或授予任何权利，也不会约束食品药品管理局（FDA）或公众。如果此类方法满足适用法规和规定的要求，则可以使用替代方法。

第1节。 目的

本指南概述了食品药品管理局（FDA）认为适用于医疗器械软件验证或用于设计，开发或制造医疗器械的软件验证的一般验证原则。该最终指导文件2.0版取代了1997年6月9日的软件验证通用原则1.1版草案。

第2节。 范围

本指南描述了医疗器械质量体系法规的某些规定如何适用于软件和该机构目前评估软件验证系统的方法。例如，本文件列出了FDA可接受的软件验证要素;但是，它没有列出在所有情况下都必须用于遵守法律的所有活动和任务。

本指南的范围比该术语最严格定义中的验证范围更广泛。本指南中讨论的规划，验证，测试，可追溯性，配置管理以及良好软件工程的许多其他方面都是重要的活动，它们共同有助于支持软件验证的最终结论。

本指南建议整合软件生命周期管理和风险管理活动。根据预期用途和与要开发的软件相关的安全风险，软件开发人员应确定具体方法，要使用的技术组合以及要应用的工作量。虽然本指南不建议任何特定的生命周期模型或任何特定的技术或方法，但它建议在整个软件生命周期中进行软件验证和验证活动。

如果软件是由设备制造商以外的其他人开发的（例如，现成的软件），则软件开发人员可能不直接负责遵守FDA法规。

在这种情况下，负有监管责任的一方（即设备制造商）需要评估现成软件开发人员活动的充分性，并确定需要做出哪些额外努力才能确定软件是否符合设备制造商的预期要求使用。

2.1. 适用性

本指南适用于：

用作医疗设备的组件，部件或附件的软件;
软件本身就是一种医疗设备（例如血液建立软件）;
用于生产设备的软件（例如，制造设备中的可编程逻辑控制器）;和
用于实现设备制造商质量系统的软件（例如，记录和维护设备历史记录的软件）。

本文档基于公认的软件验证原则，因此可应用于任何软件。出于FDA目的，本指南适用于与受管制医疗器械相关的任何软件，如“联邦食品，药品和化妆品法案”（法案）第201（h）节以及现行FDA软件和法规政策所定义。本文档未明确指出哪些软件受到监管或未受到监管。

2.2. 听众

本指南为以下人员提供了有用的信息和建议：

- 受医疗器械质量体系监管的人员
- 负责医疗设备软件设计，开发或生产的人员
- 负责设计，开发，生产或采购用于设计，开发或制造用于实施质量体系本身的医疗设备或软件工具的自动化工具的人员
- FDA调查员
- FDA合规官
- FDA科学评论员

2.3. 最麻烦的方法

我们认为我们应该考虑在医疗器械监管的所有领域中最不繁琐的方法。本指南反映了我们对相关科学和法律要求的仔细审查，以及我们认为您遵守这些要求的最不繁琐的方式。但是，如果您认为替代方法不那么繁琐，请联系我们，以便我们考虑

你的观点。您可以将您的书面意见发送给本指南前言中列出的联系人或CDRH监察员。有关CDRH监察员的全面信息，包括与他联系的方式，可以在互联网上找到：

http://www.fda.gov/cdrh/resolvingdisputes/ombudsman.html.

2.4. 软件验证的监管要求

FDA对1992年至1998年间进行的3140次医疗器械召回的分析显示，其中242起（7.7％）归因于软件故障。在那些与软件相关的召回中，192个（或79％）是由软件缺陷引起的，这些缺陷是在软件初始生产和分发后进行更改时引入的。本指南中讨论的软件验证和其他相关的良好软件工程实践是避免此类缺陷和结果召回的主要方法。

软件验证是质量体系法规的要求，该法规于1996年10月7日在联邦纪事上公布，并于1997年6月1日生效。（参见标题21联邦法规（CFR）第820部分和61联邦注册（FR）52602。验证要求适用于用作医疗设备中的组件的软件，本身为医疗设备的软件，以及用于生产设备或实施设备制造商质量体系的软件。

除非在分类法规中特别豁免，否则1997年6月1日之后开发的任何医疗器械软件产品，无论其器件类别如何，均受适用的设计控制规定的约束。（见21CFR §820.30。）此要求包括完成当前开发项目，所有新开发项目以及对现有医疗设备软件所做的所有更改。有关设备软件验证的具体要求，请参阅
21CFR §820.30（g）。医疗设备软件需要其他设计控制，例如计划，输入，验证和评审。（参见21CFR §820.30。）这些活动的相应记录结果可为医疗器械软件验证的结论提供额外支持。

根据21CFR §820.70（i）的要求，任何用于自动化设备生产过程的任何部分或质量体系的任何部分的软件都必须根据其预期用途进行验证。此要求适用于任何用于自动化设备设计，测试，组件验收，制造，标签，包装，分发，投诉处理或自动化质量系统的任何其他方面的软件。

此外，用于创建，修改和维护电子记录以及管理电子签名的计算机系统也受
验证要求的约束。

（参见21CFR§11.10（a）。）必须验证此类计算机系统，以确保准确性，可靠性，一致的预期性能，以及识别无效或更改记录的能力。

上述申请的软件可以在内部或根据合同开发。但是，软件经常为现有的特定用途购买。所有生产和/或质量系统软件，即使是现成购买的，也应具有完全定义其预期用途的文件要求，以及可与之比较测试结果和其他证据的信息，以表明软件已经过验证它的用途。

在自动化医疗设备以及自动化制造和质量系统操作中使用现成的软件正在增加。现成的软件可能具有许多功能，设备制造商只需要其中的一些功能。设备制造商负责其设备中使用的软件的充足性，并用于生产设备。当设备制造商购买“现成的”软件时，他们必须确保其在所选应用中按预期运行。对于制造或质量体系中使用的现成软件，附加指南包含在章节中本文档的6.3。对于设备软件，还有其他有用的信息可*以在FDA的工业指南，FDA审查员和医疗器械中现有软件使用合规性中找到。*

2.4.质量体系监管与上市前提交

本文档涉及涉及软件验证实施的质量体系法规问题。它为软件验证过程的管理和控制提供指导。软件验证过程的管理和控制不应与任何其他验证要求混淆，例如自动化制造过程的过程验证。

设备制造商可以使用相同的程序和记录来符合质量体系和设计控制要求，以及向FDA提交的上市前提交。本文档未涵盖与软件验证相关的任何特定安全性或功效问题。本文档未涉及受管制软件的上市前提交的设计问题和文档要求。与安全性和有效性相关的具体问题以及上市前提交的文件应提交给设备评估办公室（ODE），器械和放射健康中心（CDRH）或血液研究和审查办公室，生物制品评估与研究中心（CBER）。有关上市前提交的适用的FDA指导文件，请参阅附录A中的参考文献。

第3节。 软件验证的上下文

许多人要求就FDA期望他们采取哪些措施来确保在软件验证方面符合质量体系法规的具体指导。本文档中提供的有关软件验证的信息并不新鲜。使用第4节和第5节中列出的原则和任务对软件进行验证已经在软件行业的许多领域进行了20多年。

由于医疗设备，流程和制造设施种类繁多，因此无法在一个文档中声明适用的所有特定验证元素。但是，几个广泛概念的一般应用可以成功地用作软件验证的指导。
这些广泛的概念为构建全面的软件验证方法提供了可接受的框架。附录A中列出的许多参考资料中提供了其他具体信息。

3.1. 定义和术语

除非在质量体系法规中定义，或在下面另有说明，否则本指南中使用的所有其他术语均在当前版本的FDA计算机化系统和软件开发术语表中定义。

医疗器械质量体系法规（21 CFR 820.3（k））将“建立”定义为“定义，记录和实施”。如果出现在本指南中，“建立”和“已建立”应理解为具有相同的含义。

与软件行业中常用的术语相比，医疗设备质量体系法规中的一些定义可能会令人困惑。
示例是需求，规范，验证和验证。

3.1.1 要求和规格

虽然质量体系法规规定必须记录设计输入要求，并且必须验证指定的要求，但法规并未进一步阐明术语“要求”和“规范”之间的区别。要求可以是任何需求或期望。系统或其软件。要求反映了客户的明示或暗示的需求，可能是基于市场的，合同的或法定的，以及组织的内部要求。可以有许多不同类型的要求（例如，设计，功能，实现，接口，性能或物理要求）。软件需求通常源自已分配给软件的系统功能的那些方面的系统要求。软件需求通常以功能术语表示，并随着开发项目的进展而定义，改进和更新。准确完整地记录软件需求的成功是成功验证所得软件的关键因素。

规范被定义为“声明要求的文件。”（参见21CFR§820.3（y）。）它可以引用或包括图纸，图案或其他相关文件，并且通常表明符合以下内容的方法和标准。要求可以检查。有许多不同类型的书面规范，例如系统要求规范，软件要求规范，软件设计规范，软件测试规范，软件集成规范等。所有这些文件都建立了“特定要求”并且是各种形式的设计输出验证是必要的。

3.1.2 验证和验证

质量体系法规与ISO 8402：1994协调一致，将“验证”和“验证”视为单独和不同的术语。另一方面，许多软件工程期刊文章和教科书互换使用术语“验证”和“验证”，或者在某些情况下，将软件“验证，验证和测试（V＆T）”称为单一概念，三个术语之间没有区别。

软件验证提供客观证据，证明软件开发生命周期的特定阶段的设计输出满足该阶段的所有指定要求。软件验证在软件及其支持文档的开发过程中查找软件及其支持文档的一致性，完整性和正确性，并为随后的软件验证结论提供支持。软件测试是旨在确认软件开发输出满足其输入要求的众多验证活动之一。其他验证活动包括各种静态和动态分析，代码和文档检查，演练和其他技术。

软件验证是成品设备的设计验证的一部分，但未在质量体系法规中单独定义。出于本指南的目的，FDA认为软件验证是“通过检查确认并提供客观证据，证明软件规范符合用户需求和预期用途，并且通过软件实现的特定要求可以始终如一地实现。”在实践中，软件验证活动可以在软件开发生命周期的过程中以及在软件开发生命周期结束时进行，以确保满足所有要求。由于软件通常是较大硬件系统的一部分，因此软件验证通常包括所有软件要求已正确且完整地实施且可追溯到系统要求的证据。软件验证的结论高度依赖于在软件开发生命周期的每个阶段执行的全面的软件测试，检查，分析和其他验证任务。在模拟使用环境中测试设备软件功能，以及用户站点测试通常作为软件自动化设备的整体设计验证程序的组件包括在内。

软件验证和验证很困难，因为开发人员无法永远测试，而且很难知道有多少证据就足够了。在很大程度上，软件验证是一种“信心水平”的问题，即设备满足软件自动化功能和设备功能的所有要求和用户期望。诸如规范文件中的缺陷，剩余缺陷的估计，测试覆盖率和其他

技术等措施都被用于

在运输产品之前建立可接受的置信水平。信心水平以及所需的软件验证，验证和测试工作水平将根据设备自动化功能带来的安全风险（危险）而变化。补充指导有*关软件安全风险管理的信息，请参见FDA关于医疗器械软件上市前提交内容指南的第4节，以及*附录A中引用的国际标准ISO / IEC 14971-1和IEC 60601-1-4。

3.1.3 IQ/OQ/PQ

多年来，FDA和受监管行业都试图在流程验证术语的背景下理解和定义软件验证。例如，行业文档和其他FDA验证指南有时会在安装资格（IQ），操作资格（OQ）和性能认证（PQ）方面描述用户站点软件验证。这些术语的定义和有关IQ / OQ / PQ的其他信息可在FDA中找到*指南 流程验证的一般原则，*1987年5月11日和FDA的*词汇表 计算机化系统和软件开发术语，*1995年8月。

虽然IQ / OQ / PQ术语很好地实现了它的目的，并且是在用户站点组织软件验证任务的许多合法方法之一，但许多软件专业人员可能无法很好地理解这个术语，并且在本文档的其他地方没有使用它. .但是，FDA人员和设备制造商都需要了解术语中的这些差异，因为他们要求并提供有关软件验证的信息。

3.2. 软件开发作为系统设计的一部分

使用软件实现系统功能的决定通常是在系统设计期间进行的。软件需求通常源自整个系统要求和系统中将使用软件实现的那些方面的设计。成品设备有用户需求和预期用途，但用户通常不指定硬件，软件或两者的某种组合是否满足这些要求。因此，必须在系统的整体设计验证的上下文中考虑软件验证。

文档化的需求规范代表了用户的需求和开发产品的预期用途。软件验证的主要目标是证明所有已完成的软件产品符合所有已记录的软件和系统要求。系统要求和软件要求的正确性和完整性应作为设备设计验证过程的一部分来解决。软件验证包括确认所有软件规范的一致性，并确认所有软件要求均可追溯到系统规范。确认是整体设计验证的重要部分，以确保医疗设备的所有方面符合用户需求和预期用途。

3.3. 软件不同于硬件

虽然软件与硬件共享许多相同的工程任务，但它有一些非常重要的差异。例如：

- 绝大多数软件问题都可追溯到设计和开发过程中出现的错误。虽然硬件产品的质量高度依赖于设计，开发和制造，但软件产品的质量主要取决于设计和开发，而对软件制造的关注最小。软件制造包括可以轻松验证的复制。制作数千个与原版完全相同的程序副本并不困难;难以让原始程序满足所有规格。

- 软件最重要的特征之一是分支，即基于不同输入执行备选系列命令的能力。此功能是软件另一个特性的主要因素 - 它的复杂性。即使是简短的程序也可能非常复杂，难以完全理解。

- 通常，仅靠测试无法完全验证软件是否完整和正确。除了测试之外，还应结合其他验证技术以及结构化和文档化的开发过程，以确保采用全面的验证方法。

- 与硬件不同，软件不是物理实体，也不会磨损。事实上，软件可能随着年龄的增长而改善，因为潜在的缺陷被发现并被删除。但是，随着软件不断更新和更改，这些改进有时会在更改期间引入软件中的新缺陷中得到抵消。

- 与某些硬件故障不同，软件故障在没有高级警告的情该软件的分支允许它在执行期间遵循不同的路径，可能会隐藏一些潜在的缺陷，直到软件产品被引入市场很久。

- 软件的另一个相关特性是它可以更改的速度和简便性。这个因素可能导致软件和非软件专业人员相信软件问题可以轻松纠正。结合对软件缺乏了解，它可以使管理人员相信严格控制的工程对于软件和硬件都不需要那样多。事实上，情况正好相反。由于其复杂性，软件的开发过程应该比硬件更严格地控制，以防止在开发过程中稍后无法轻易检测到的问题。

- 软件代码中看似无关紧要的变化可能会在软件程序的其他地方产生意外和非常重要的问题。应充分规划，控制和记录软件开发过程，以检测和纠正软件变更带来的意外结果。

- 鉴于对软件专业人员和高度移动的劳动力的高需求，对软件进行维护更改的软件人员可能没有参与原始软件开发。因此，准确而全面的文档是必不可少的。

- 从历史上看，软件组件并不像硬件组件那样经常标准化和可互换。但是，医疗设备软件开发人员开始使用基于组件的开发工具和技术。面向对象的方法和现成的软件组件的使用有望实现更快，更便宜的软件开发。但是，基于组件的方法在集成期间需要非常小心。在集成之前，需要时间来完全定义和开发可重用的软件代码，并充分了解现成组件的行为。

由于这些原因和其他原因，软件工程需要比硬件工程更高级别的管理审查和控制。

3.4. 软件验证的好处

软件验证是用于确保设备软件和软件自动化操作质量的关键工具。软件验证可以提高设备的可用性和可靠性，从而降低故障率，减少召回和纠正措施，降低患者和用户的风险，降低设备制造商的责任。软件验证还可以通过使可靠地修改软件和重新验证软件更改变得更容易，成本更低来降低长期成本。软件维护在整个生命周期中占软件总成本的很大一部分。已建立的综合软件验证流程通过降低软件每个后续版本的验证成本，有助于降低软件的长期成本。

3.5 设计回顾

设计评审是对设计进行记录，全面和系统的检查，以评估设计要求的充分性，评估设计满足这些要求的能力，以及发现问题。虽然在软件项目期间可能会在开发团队中发生许多非正式技术评审，但正式的设计评审更加结构化，并包括开发团队以外的其他人的参与。正式的设计评审可以参考或包括其他正式和非正式评论的结果。在软件与硬件集成到系统中或两者之后，可以单独为软件进行设计评审。设计评审应包括检查开发计划，需求规范，设计规范，测试计划和程序，与项目相关的所有其他文档和活动，定义生命周期的每个阶段的验证结果以及整个设备的验证结果。

设计评审是管理和评估开发项目的主要工具。例如，正式的设计审核允许管理层确认软件验

证计划中定义的所有目标

已经实现了。质量体系法规要求在设备设计过程中至少进行一次正式的设计审查。但是，建议进行多次设计评审（例如，在每个软件生命周期活动结束时，准备进行下一个活动）。在将主要资源提交给特定设计解决方案之前，正式设计审核在需求活动结束时或接近结束时尤其重要。此时发现的问题可以更容易地解决，节省时间和金钱，并减少错过关键问题的可能性。

在正式的设计评审期间，应记录一些关键问题的答案。这些包括：

- 是否为每个软件生命周期活动建立了适当的任务和预期结果，输出或产品？
- 执行每个软件生命周期活动的任务和预期结果，输出或产品：
  - ✓ 在正确性，完整性，一致性和准确性方面是否符合其他软件生命周期活动的要求？
  - ✓ 满足该活动的标准，实践和惯例？
  - ✓ 为下一个软件生命周期活动启动任务建立适当的基础？

第4节。 软件验证原则

本节列出了软件验证应考虑的一般原则。

4.1. 要求

文档化的软件需求规范为验证和验证提供了基准。没有既定的软件要求规范（参考：21 CFR 820.3（z）和（aa）和820.30（f）和（g）），软件验证过程无法完成。

4.2. 缺陷预防

软件质量保证需要专注于防止将缺陷引入软件开发过程，而不是试图在软件代码写入后“测试质量”。
软件测试在揭示软件代码中所有潜在缺陷的能力方面非常有限。例如，大多数软件的复杂性使其无法进行详尽的测试。软件测试是必要的活动。但是，在大多数情况下，软件测试本身并不足以确定软件是否适合其预期用途。为了建立这种信心，软件开发人员应该使用各种方法和技术来防止软件错误并检测确实发生的软件错误。方法的“最佳组合”取决于许多因素，包括开发环境，应用程序，项目规模，语言和风险。

4.3. 时间和精力

要构建软件验证的案例需要时间和精力。软件验证的准备应该尽早开始，即在设计和开发规划和设计输入期间。软件验证的最终结论应基于从整个软件生命周期中进行的计划工作中收集的证据。

4.4. 软件生命周期

软件验证在已建立的软件生命周期的环境中进行。软件生命周期包含支持软件验证工作所必需的软件工程任务和文档。此外，软件生命周期包含适用于软件预期用途的特定验证和验证任务。本指南不建议任何特定的生命周期模型 - 只应选择它们并用于软件开发项目。

4.5. 计划

通过使用计划来定义和控制软件验证过程。软件验证计划定义了通过软件验证工作完成的“内容”。软件验证计划是一种重要的质量系统工具。软件验证计划指定范围，方法，资源，计划以及活动，任务和工作项的类型和范围等方面。

4.6. 程序

通过使用程序执行软件验证过程。这些程序建立了“如何”进行软件验证工作。这些过程应确定完成单个验证活动，任务和工作项必须采取的特定操作或操作顺序。

4.7. 更改后的软件验证

由于软件的复杂性，看似很小的本地变化可能会对全球系统产生重大影响。当对软件进行任何更改（甚至是小的更改）时，需要重新建立软件的验证状态。无论何时更改软件，都应进行验证分析，不仅要验证单个更改，还要确定更改对整个软件系统的影响程度和影响。基于此分析，软件开发人员应进行适当级别的软件回归测试，以显示系统中未更改但易受攻击的部分未受到不利影响。设计控制和适当的回归测试可确保软件在软件更改后得到验证。

4.8. 验证范围

验证范围应基于软件的复杂性和安全风险 - 而不是基于公司规模或资源限制。验证活动，任务和工作项的选择应与软件设计的复杂性以及与指定预期用途的软件使用相关的风险相称。对于风险较低的设备，只能进行基线验证活动。随着风险的增加，应增加额外的验证活动以弥补额外的风险。验证文档应足以证明所有软件验证计划和程序已成功完成。

4.9. 审查的独立性

验证活动应使用“审查独立性”的基本质量保证准则进行。自我验证极其困难。在可能的情况下，独立评估总是更好，特别是对于风险较高的应用。一些公司外包给第三方独立

验证和验证，但此解决方案可能并不总是可行的。另一种方法是指派内部工作人员不参与特定设计或其实施，但有足够的知识来评估项目并进行验证和确认活动。较小的公司可能需要在如何组织和分配任务方面具有创造性，以保持审查的内部独立性。

4.10. 灵活性和责任感

这些软件验证原则的具体实现可能因应用程序而异。设备制造商可以灵活选择如何应用这些验证原则，但仍然负责证明软件已经过验证。

软件在各种环境中设计，开发，验证和管理，适用于各种风险级别的设备。FDA监管的医疗设备应用包括以下软件：

- 是医疗器械的组件，部件或附件;
- 本身就是一种医疗器械;要么
- 用于制造，设计和开发，或质量体系的其他部分。

在每个环境中，来自许多来源的软件组件可用于创建应用程序（例如，内部开发的软件，现成的软件，合同软件，共享软件）。此外，软件组件有许多不同的形式（例如，应用程序软件，操作系统，编译器，调试器，配置管理工具等等）。在这些环境中验证软件可能是一项复杂的工作;因此，在设计软件验证过程时，应考虑所有这些软件验证原则。由此产生的软件验证过程应与系统，设备或过程相关的安全风险相称。

软件验证活动和任务可以分散，发生在不同的位置，由不同的组织进行。但是，无论任务分配，合同关系，组件来源或开发环境如何，设备制造商或规范开发人员都有最终责任确保软件得到验证。

第5节。 活动和任务

软件验证是通过在软件开发生命周期的各个阶段计划和执行的一系列活动和任务来完成的。这些任务可能是一次出现或可能多次迭代，具体取决于所使用的生命周期模型以及随着软件项目的进展而进行的更改范围。

5.1. 软件生命周期活动

本指南不建议使用任何特定的软件生命周期模型。软件开发人员应建立适合其产品和组织的软件生命周期模型。所选的软件生命周期模型应涵盖从出生到退役的软件。典型软件生命周期模型中的活动包括以下内容：

- 质量策划
- 系统要求定义
- 详细的软件需求规范
- 软件设计规范
- 建筑或编码
- 测试
- 安装
- 运营和支持
- 保养
- 退休

在每个活动期间都会进行支持软件验证的验证，测试和其他任务。生命周期模型以各种方式组织这些软件开发活动，并提供用于监视和控制软件开发项目的框架。FDA中定义了几种软件生命周期模型（例如，瀑布，螺旋，快速原型，增量开发等）*计算机系统和软件开发术语表，* 这些和许多其他生命周期模型在附录A中列出的各种参考文献中有所描述。

5.2. 支持验证的典型任务

对于每个软件生命周期活动，某些“典型”任务支持软件验证的结论。但是，要执行的具体

任务，其性能顺序以及性能的迭代和时间将由所选的特定软件生命周期模型和与软件应用程序相关的安全风险决定。对于风险极低的应用程序，可能根本不需要某些任务。但是，软件开发人员至少应该考虑这些任务中的每一个，并且应该定义和记录哪些任务适合或不适合

他们的具体应用。以下讨论是通用的，并不旨在规定任何特定的软件生命周期模型或执行任务的任何特定顺序。

5.2.1. 质量策划

设计和开发计划最终应制定一个计划，确定必要的任务，异常报告和解决程序，必要的资源和管理评审要求，包括正式的设计评审。应识别软件生命周期模型和相关活动，以及每个软件生命周期活动所需的那些任务。该计划应包括：

- 每个生命周期活动的具体任务;
- 列举重要的质量因素（例如，可靠性，可维护性和可用性）;
- 每项任务的方法和程序;
- 任务验收标准;
- 确定和记录产出的标准，以便评估其对投入要求的符合性;
- 每项任务的输入;
- 每项任务的输出;
- 每项任务的角色，资源和责任;
- 风险和假设;和
- 用户需求的文档。

管理层必须确定并提供适当的软件开发环境和资源。（参见21CFR§820.20（b）（1）和（2）。）通常，每项任务都需要人员和物质资源。该计划应确定每项任务的人员，设施和设备资源，以及风险（危险）管理将发挥的作用。应开发配置管理计划，以指导和控制多个并行开发活动并确保正确的通信和文档。必须进行控制以确保包含软件系统的规范文档，源代码，目标代码和测试套件的所有批准版本之间的正确和正确的对应关系。控制还应确保准确识别和访问当前批准的版本。

应创建程序以报告和解决通过验证或其他活动发现的软件异常。管理层应识别报告并指定每份报告的内容，格式和负责的组织要素。审核和批准软件开发结果也需要程序，包括此类审核和批准的负责组织元素。

典型任务 - 质量计划

风险（危害）管理计划

配置管理计划

软件质量保证计划
- 软件验证和验证计划
  - 验证和验证任务以及验收标准
  - 计划和资源分配（用于软件验证和验证活动）
  - 报告要求
- 正式的设计审查要求
- 其他技术评审要求

问题报告和解决程序

其他支持活动

5.2.2. 要求

需求开发包括识别，分析和记录有关设备及其预期用途的信息。特别重要的领域包括系统功能分配到硬件/软件，操作条件，用户特征，潜在危险和预期任务。此外，要求应明确说明软件的预期用途。

软件需求规范文档应包含软件功能的书面定义。没有预先确定和记录的软件要求，无法验证软件。典型的软件要求指定以下内容：

- 所有软件系统输入;
- 所有软件系统输出;
- 软件系统将执行的所有功能;
- 软件将满足的所有性能要求（例如，数据吞吐量，可靠性和时序）;
- 所有外部和用户界面的定义，以及任何内部软件到系统接口;
- 用户如何与系统交互;
- 什么构成错误以及如何处理错误;
- 所需的响应时间;
- 软件的预期操作环境，如果这是设计约束（例如，硬件平台，操作系统）;
- 软件将接受的所有范围，限制，默认值和特定值;和
- 将在软件中实施的所有安全相关要求，规格，功能或功能。

软件安全要求源自与系统需求开发过程紧密集成的技术风险管理过程。软件需求规范应明确指出系统中软件故障可能导致的潜在危险以及软件中要实施的任何安全要求。应评估软件故障的后果，以及减轻此类故障的方法（例如，硬件缓解，防御性编程等）。从这一分

析中，应该可以确定防止伤害所需的最适当措施。

质量体系法规需要一种机制来解决不完整，模糊或冲突的要求。（参见21 CFR 820.30（c）。）应评估软件需求规范中确定的每个要求（例如，硬件，软件，用户，操作员界面和安全性）的准确性，完整性，一致性，可测试性，正确性和清晰度。例如，应评估软件要求以验证：

- 要求之间没有内部不一致;
- 已经详细说明了该系统的所有性能要求;
- 容错，安全和安全要求是完整和正确的;
- 软件功能分配准确，完整;
- 软件要求适用于系统危害;和
- 所有要求均以可衡量或客观可验证的术语表示。

应进行软件需求可追溯性分析，以跟踪（和）系统需求和风险分析结果的软件需求。除了用于验证软件要求的任何其他分析和文档之外，建议进行正式的设计评审，以确认在广泛的软件设计工作开始之前，要求是完全明确和适当的。
需求可以逐步批准和发布，但应注意软件（和硬件）需求之间的交互和接口得到适当的审查，分析和控制。

典型任务 - 要求

初步风险分析
可追溯性分析
– 系统要求的软件要求（反之亦然）
– 风险分析的软件要求
用户特征描述
列出主要和次要记忆的特征和局限性
软件需求评估
软件用户界面要求分析
系统测试计划生成
验收测试计划生成
歧义回顾或分析

5.2.3. 设计

在设计过程中，软件需求规范被转换为要实现的软件的逻辑和物理表示。软件设计规范描述了软件应该做什么以及应该如何做。由于项目的复杂性或启用

具有不同技术职责水平的人员可以清楚地了解设计信息，设计规范可能包含设计的高级概要和详细的设计信息。完整的软件设计规范限制程序员/编码人员遵守商定的要求和设计的意图。完整的软件设计规范将使程序员免于做出临时设计决策的需要。

软件设计需要解决人为因素。使用由于设计过于复杂或与用户对操作的直观期望相反的错误是FDA遇到的最持久和最关键的问题之一。通常，软件的设计是这种使用错误的一个因素。人因工程应该融入整个设计和开发过程，包括设备设计要求，分析和测试。在开发流程图，状态图，原型设计工具和测试计划时，应考虑设备安全性和可用性问题。此外，还应执行任务和功能分析，风险分析，原型测试和评审以及完整的可用性测试。应用这些方法时，应包括用户群体的参与者。

软件设计规范应包括：

- 软件需求规范，包括接受软件的预定标准;
- 软件风险分析;
- 开发程序和编码指南（或其他编程程序）;
- 系统文档（例如，叙述或上下文图），描述程序要运行的系统上下文，包括硬件，软件和物理环境的关系;
- 要使用的硬件;
- 要测量或记录的参数;
- 逻辑结构（包括控制逻辑）和逻辑处理步骤（例如算法）;
- 数据结构和数据流程图;
- 变量（控制和数据）的定义及其使用位置的描述;
- 错误，警报和警告消息;
- 支持软件（例如，操作系统，驱动程序，其他应用程序软件）;
- 通信链接（软件内部模块之间的链接，与支持软件的链接，与硬件的链接以及与用户的链接）;
- 安全措施（物理和逻辑安全）;和
- 上述元素中未标识的任何其他约束。

上面提到的前四个元素通常是单独的预先存在的文档，在软件设计规范中通过引用包含在内。软件需求规范在前一节中讨论过，软件风险分析也是如此。书面开发程序可作为组织的指南，书面编程程序可作为单个程序员的指南。

由于软件无法在不了解其运行环境的情况下进行验证，因此引用了系统文档。如果上述某些元素未包含在软件中，则为

如果明确说明，可能对软件的未来审阅者和维护者有帮助（例如，此程序中没有错误消息）。

软件设计期间发生的活动有几个目的。进行软件设计评估以确定设计是否完整，正确，一致，明确，可行和可维护。在设计期间适当考虑软件架构（例如，模块化结构）可以在需要软件改变时减少未来验证工作的量级。软件设计评估可以包括控制流程，数据流，复杂性，时序，大小调整，内存分配，关键性分析以及设计的许多其他方面的分析。应进行可追溯性分析，以验证软件设计是否满足所有软件要求。作为识别需求不足的技术，可追溯性分析还应验证设计的所有方面是否可追溯到软件需求。应对通信链路进行分析，以评估所提出的硬件，用户和相关软件要求的设计。应重新检查软件风险分析，以确定是否已识别出任何其他危害以及设计是否引入了任何新的危害。

在软件设计活动结束时，在开始实施设计之前，应进行正式设计评审以验证设计是否正确，一致，完整，准确和可测试。部分设计可以逐步批准和发布，以便实施;但应注意对各种要素之间的相互作用和沟通联系进行适当的审查，分析和控制。

大多数软件开发模型都是迭代的。这很可能导致软件需求规范和软件设计规范的几个版本。所有批准的版本都应按照既定的配置管理程序进行存档和控制。

典型任务 - 设计

- 更新了软件风险分析
- 可追溯性分析 - 软件要求的设计规范（反之亦然）
- 软件设计评估
- 设计通信链路分析
- 模块测试计划生成
- 集成测试计划生成
- 测试设计生成（模块，集成，系统和验收）

5.2.4. 建筑或编码

软件可以通过编码（即，编程）或通过将先前编码的软件组件（例如，来自代码库，现成软件等）组装在一起以在新应用中使用来构造。编码是将详细设计规范实现为源代码的软件活动。编码是软件开发过程的最低抽象级别。这是软件需求分解的最后阶段，其中模块规范被转换为编程语言。

编码通常涉及使用高级编程语言，但也可能需要使用汇编语言（或微代码）来进行时间关键操作。可以编译或解释源代码以在目标硬件平台上使用。选择编程语言和软件构建工具（汇编器，链接器和编译器）的决策应包括考虑对后续质量评估任务的影响（例如，所选语言的调试和测试工具的可用性）。一些编译器提供可选的级别和命令以进行错误检查，以帮助调试代码。在整个编码过程中可以使用不同级别的错误检查，并且可以记录或不记录来自编译器的警告或其他消息。
但是，在编码和调试过程结束时，通常会使用最严格的错误检查级别来记录软件中仍然存在的编译错误。如果最严格的错误检查级别不用于源代码的最终转换，则应记录使用不太严格的转换错误检查的理由。此外，对于最终编译，应该有编制过程及其结果的文档，包括来自编译器及其解决方案的任何警告或其他消息，或者决定保留未解决问题的理由。

公司经常采用特定的编码指南来建立与软件编码过程相关的质量策略和程序。应评估源代码以验证其是否符合指定的编码指南。此类指南应包括有关清晰度，样式，复杂性管理和注释的编码约定。代码注释应为模块提供有用的描述性信息，包括预期的输入和输出，引用的变量，预期的数据类型和要执行的操作。还应评估源代码以验证其是否符合相应的详细设计规范。准备进行集成和测试的模块应具有符合编码指南和任何其他适用的质量策略和程序的文档。

源代码评估通常实现为代码检查和代码演练。这种静态分析提供了在执行代码之前检测错误的非常有效的手段。它们允许单独检查每个错误，还可以帮助聚焦以后的软件动态测试。公司可以使用适当的控制手动（桌面）检查，以确保一致性和独立性。应将源代码评估扩展到验证模块和层（水平和垂直接口）之间的内部链接，并符合其设计规范。应保留所用程序的文档和源代码评估的结果，作为设计验证的一部分。

源代码可跟踪性分析是验证所有代码是否与已建立的规范和已建立的测试过程相关联的重要工具。应进行源代码可追溯性分析并记录，以验证：

- 软件设计规范的每个元素都已在代码中实现;
- 在代码中实现的模块和功能可以追溯到软件设计规范中的元素和风险分析;
- 模块和功能的测试可以追溯到软件设计规范中的元素和风险分析;和
- 模块和函数的测试可以追溯到相同模块和功能的源代码。

典型任务 - 构造或编码

可追溯性分析
– 源代码到设计规范（反之亦然）
– 测试用例源代码和设计规范

源代码和源代码文档评估

源代码接口分析

测试程序和测试用例生成（模块，集成，系统和验收）

5.2.5. 由软件开发人员测试

软件测试需要在已知条件下运行软件产品，并使用已定义的输入和记录的结果，并将其与预定义的预期进行比较。这是一项耗时，困难和不完美的活动。因此，它需要早期规划才能有效和高效。

应尽可能在软件开发过程中尽可能地创建测试计划和测试用例。他们应确定时间表，环境，资源（人员，工具等），方法，案例（输入，程序，输出，预期结果），文档和报告标准。在整个测试过程中应用的努力量可以与复杂性，关键性，可靠性和/或安全性问题相关联（例如，需要通过对其容错特征进行密集测试来产生关键结果的功能或模块）。软件类别和软件测试工作的描述出现在文献中，例如：

- N*IST特刊500-235，结构化测试：使用Cyclomatic Complexity Metric的测试方法;*
- N*UREG / CR-6293，高完整性系统的验证和验证指南;和*
- I*EEE计算机学会出版社，软件可靠性工程手册。*

软件测试计划应确定在每个开发阶段要执行的特定任务，并包括其相应完成标准所代表的工作量的合理性。

软件测试具有在规划特定软件产品的测试时必须识别和考虑的限制。除了最简单的程序，软件无法进行详尽的测试。通常，使用所有可能的输入测试软件产品是不可行的，也不可能测试在程序执行期间可能发生的所有可能的数据处理路径。没有一种类型的测试或测试方法可以确保特定的软件产品已经过全面测试。测试所有程序功能并不意味着所有程序都已经过测试。测试所有程序代码并不意味着程序中存在所有必需的功能。测试所有程序功能和所有程序代码并不意味着程序100％正确！不发现错误的软件测试不应被解释为软件产品中不存在错误;这可能意味着测试是肤浅的。

软件测试用例的一个基本要素是预期的结果。这是关键细节，可以客观评估实际测试结果。从相应的预定义定义或规范获得该必要的测试信息。软件规范文档必须通过工程（即可测量或可客观验证）的详细程度来确定要通过测试确认的内容，时间，方式，原因等。有效软件测试的真正努力在于定义要测试的内容而不是测试的性能。

软件测试过程应基于促进软件产品有效检查的原则。适用的软件测试原则包括：

- 预期的测试结果是预定义的;
- 一个好的测试用例很有可能暴露错误;
- 成功的测试是发现错误的测试;
- 编码独立;
- 使用应用程序（用户）和软件（编程）专业知识;
- 测试人员使用编码员使用的不同工具;
- 仅检查通常的情况是不够的;
- 测试文档允许其重用并在随后的审查期间独立确认测试结果的通过/未通过状态。

一旦成功完成先决任务（例如代码检查），就开始进行软件测试。它从单元级测试开始，最后进行系统级测试。可能存在明显的集成测试级别。软件产品应该根据其内部结构和基于其外部规范的测试用例来挑战测试用例。这些测试应该对软件产品的功能，性能和接口定义和要求的合规性进行全面而严格的检查。

基于代码的测试也称为结构测试或“白盒”测试。它根据从源代码，详细设计规范和其他开

发文档中获得的知识来识别测试用例。这些测试用例挑战了程序的控制决策;以及程序的数据结构，包括配置表。结构测试可以识别“死”代码

程序运行时从未执行过。结构测试主要通过单元（模块）级别测试完成，但可以扩展到其他级别的软件测试。

可以使用旨在显示在结构测试期间评估了软件结构的百分比的度量来评估结构测试的级别。这些度量通常被称为“覆盖”，并且是关于测试选择标准的完整性的度量。结构覆盖的数量应与软件所构成的风险水平相当。
使用术语“覆盖”通常意味着100％的覆盖率。例如，如果测试程序已达到“语句覆盖率”，则意味着软件中100％的语句至少已执行一次。常见的结构覆盖指标包括：

- **声**明覆盖范围 - 此标准要求每个程序声明至少执行一次足够的测试用例;但是，它的成就不足以为软件产品的行为提供信心。

- **决**策（分支）覆盖范围 - 此标准要求对每个程序决策或分支执行足够的测试用例，以便每个可能的结果至少发生一次。它被认为是大多数软件产品的最低覆盖水平，但仅对于高完整性应用而言，决策覆盖率不足。

- **条**件覆盖 - 该标准要求程序决策中的每个条件都有足够的测试用例，以便至少对所有可能的结果进行一次。只有在必须评估多个条件才能做出决策时，它才与分支覆盖不同。

- **多**条件覆盖 - 此标准要求足够的测试用例在程序决策中执行所有可能的条件组合。

- **循**环覆盖 - 此标准要求对所有程序循环执行足够的测试用例，以执行零次，一次，两次和多次迭代，包括初始化，典型运行和终止（边界）条件。

- **路**径覆盖 - 该条件要求从定义的程序段的开始到退出的每个可行路径，基础路径等有足够的测试用例，至少执行一次。由于通过软件程序的可能路径非常多，因此通常无法实现路径覆盖。路径覆盖量通常基于被测软件的风险或关键性来确定。

- **数**据流覆盖范围 - 此标准要求每个可行数据流至少执行一次的充分测试用例。有许多数据流测试策略可供使用。

基于定义或基于规范的测试也称为功能测试或“黑盒”测试。它根据软件产品（无论是单元（模块）还是完整程序）的定义来识别测试用例。这些测试用例挑战了程序的预期用途或功

能，以及程序的内部和外部接口。功能测试可应用于从单元级到系统级测试的所有级别的软件测试。

以下类型的功能软件测试通常涉及增加工作量：

- **正**常情况 - 必须使用常规输入进行测试。但是，仅使用预期的有效输入测试软件产品并不能彻底测试该软件产品。就其本身而言，正常的案例测试无法对软件产品的可靠性提供足够的信心。

- **输**出强制 - 选择测试输入以确保通过测试生成所选（或所有）软件输出。

- **稳**健性 - 软件测试应证明软件产品在给出意外的无效输入时表现正常。用于识别足够的这种测试用例集的方法包括等价类划分，边界值分析和特殊情况识别（错误猜测）。虽然重要且必要，但这些技术并不能确保软件产品的所有最合适的挑战都已确定用于测试。

- **输**入组合 - 上面确定的功能测试方法都强调单个或单个测试输入。大多数软件产品在其使用条件下运行多个输入。彻底的软件产品测试应考虑软件单元或系统在操作期间可能遇到的输入组合。可以扩展错误猜测以识别输入的组合，但它是一种特殊技术。因果图形是一种功能性软件测试技术，它系统地识别软件产品的输入组合以包含在测试用例中。

功能和结构软件测试用例识别技术为测试提供特定输入，而不是随机测试输入。这些技术的一个缺点是难以将结构和功能测试完成标准与软件产品的可靠性联系起来。可以采用诸如统计测试之类的高级软件测试方法来进一步保证软件产品是可靠的。统计测试使用基于操作配置文件（例如，预期使用，危险使用或软件产品的恶意使用）的来自定义的分布的随机生成的测试数据。生成大量测试数据并且可以针对特定区域或关注点，提供识别软件产品设计者或其测试者未预期的个别和多个罕见操作条件的可能性。统计测试还提供高结构覆盖率。它确实需要稳定的软件产品。因此，结构和功能测试是软件产品统计测试的先决条件。

软件测试的另一个方面是测试软件变化。软件开发过程中经常发生变化。这些变化是1）调试发现错误并得到纠正的结果，2）新的或变更的要求（“要求蠕变”），以及3）修改的设计，因为找到了更有效或更有效的实施。一旦软件产品已经基线（批准），对该产品的任何更改都应该有自己的“迷你生命周期”，包括测试。测试已更改的软件产品需要额外的

努力。它不仅应证明更改已正确实施，而且测试还应证明更改不会对软件产品的其他部分产生负面影响。采用回归分析和测试来提供

保证变更不会在软件产品的其他地方造成问题。回归分析是基于对相关文档的审查（例如，软件需求规范，软件设计规范，源代码，测试计划，测试用例，测试脚本等）确定变更的影响，以便确定必要的要运行的回归测试。回归测试是重新运行程序先前已正确执行的测试用例，并将当前结果与先前结果进行比较，以便检测软件更改的意外影响。
当使用集成方法构建软件产品时，还应采用回归分析和回归测试，以确保新集成的模块不会对先前集成的模块的操作产生负面影响。

为了对软件产品进行全面而严格的检查，开发测试通常被分为几个级别。例如，软件产品的测试可以组织成单元，集成和系统级别的测试。

1) 单元（模块或组件）级别测试侧重于子程序功能的早期检查，并确保通过测试检查系统级别不可见的功能。单元测试可确保提供高质量的软件单元，以便集成到成品软件产品中。

2) 集成级别测试侧重于跨程序的内部和外部接口传输数据和控制。外部接口是与其他软件（包括操作系统软件），系统硬件和用户的接口，可以描述为通信链路。

3) 系统级测试表明存在所有指定的功能，并且软件产品值得信赖。此测试根据指定操作平台上显示的软件产品要求验证竣工程序的功能和性能。系统级软件测试解决了与预期用途相关的功能问题和设备软件的以下元素：

- 性能问题（例如，响应时间，可靠性测量）;
- 对压力条件的响应，例如，在最大负载下的行为，连续使用;
- 内部和外部安全功能的操作;
- 恢复程序的有效性，包括灾难恢复;
- 可用性;
- 与其他软件产品的兼容性;
- 每个定义的硬件配置中的行为;和
- 文件的准确性。

应使用控制措施（例如，可追溯性分析）来确保实现预期的覆盖范围。

系统级测试还展示了软件产品在预期操作环境中的行为。此类测试的位置取决于软件开发人

员生成目标操作环境的能力。根据情况，可以利用（潜在）客户位置的模拟和/或测试。测试计划应确定确保所需的控制措施

当计划的系统级测试在不是由软件开发者直接控制的站点进行时，实现了预期的覆盖范围并准备了适当的文档。此外，对于作为医疗设备的软件产品或在FDA批准之前将用于人类的医疗设备的组件，涉及人类受试者的测试可能需要研究设备豁免（IDE）或机构审查委员会（IRB）批准。

应以允许达到客观通过/未通过决策的方式记录测试程序，测试数据和测试结果。它们也应该适合于在运行测试之后进行审查和客观决策，并且它们应该适用于任何后续的回归测试。在发布软件之前，应记录，分类，审查和解决测试期间检测到的错误。在开发生命周期期间收集和分析的软件错误数据可用于确定软件产品对于商业分发的发布的适合性。测试报告应符合相应测试计划的要求。

在医疗设备或其生产中执行有用功能的软件产品通常很复杂。软件测试工具经常用于确保此类软件产品的测试的一致性，彻底性和效率，并满足计划的测试活动的要求。
这些工具可能包括内部构建的支持软件，以便于单元（模块）测试和后续集成测试（例如，驱动程序和存根）以及商业软件测试工具。这些工具的质量应不低于他们用于开发的软件产品。应保留适当的文件，证明这些软件工具的确认用于其预期用途（见本指南第6节）。

典型任务 - 由软件开发人员进行测试

测试计划
结构测试用例识别
功能测试用例识别
可追溯性分析 - 测试
- 单元（模块）测试详细设计
- 集成测试到高级设计
- 系统测试软件需求

单元（模块）测试执行
集成测试执行
功能测试执行
系统测试执行
验收测试执行
测试结果评估

错误评估/解决方案

最终测试报告

5.2.6. 用户站点测试

在用户站点进行测试是软件验证的重要部分。质量体系法规要求安装和检查程序（包括适当的测试）以及检查和测试文件，以证明正确安装。（参见21CFR§820.170。）同样，制造设备必须符合规定的要求，并且必须对自动化系统的预期用途进行验证。（分别参见21CFR§820.70（g）和21CFR§820.70（i）。）

有关用户站点测试的术语可能会令人困惑。诸如beta测试，站点验证，用户验收测试，安装验证和安装测试等术语都已用于描述用户站点测试。出于本指南的目的，术语“用户站点测试”包含所有这些以及在开发人员受控环境之外进行的任何其他测试。此测试应在用户的站点上进行，其中包含将作为已安装系统配置一部分的实际硬件和软件。测试是通过在其打算运行的上下文中测试的软件的实际或模拟使用来完成的。

此处包含的指导是一般性的，适用于任何用户现场测试。但是，在某些区域（例如，血液建立系统），可能存在特定的站点验证问题，需要在规划用户站点测试时加以考虑。测试计划人员应与具有相应产品管辖权的FDA中心核实，以确定是否对用户现场测试有任何其他监管要求。

用户站点测试应遵循预定义的书面计划，其中包含正式的测试摘要和正式接受记录。应保留所有测试程序，测试输入数据和测试结果的文件证据。

应该有证据表明硬件和软件的安装和配置符合规定。措施应确保在测试期间执行所有系统组件，并确保这些组件的版本是指定的。测试计划应规定在整个操作条件范围内进行测试，并应指定持续足够长的时间以允许系统遇到各种条件和事件，以便检测在更正常的活动中不明显的任何潜在故障。

开发人员网站上的软件开发人员早先执行的一些评估应在实际使用的站点重复进行。这些可能包括测试大量数据，重负载或压力，安全性，故障测试（避免，检测，容差和恢复），错误消息和安全要求的实施。开发人员可能能够向用户提供一些用于此目的的测试数据集。

除了评估系统正确执行其预期功能的能力之外，还应评估系统用户理解并正确连接它的能力。操作员应该能够执行预期的功能，并以适当和及时的方式响应所有警报，警告和错误

消息。

在用户站点测试期间，应保持记录正确的系统性能和遇到的任何系统故障。对用户现场测试期间检测到的故障进行补偿的系统修订应遵循与任何其他软件更改相同的程序和控制。

该软件的开发人员可能参与或可能不参与用户站点测试。如果开发人员参与其中，他们可以无缝地将设计级系统测试的最后部分转移到用户的站点。如果开发人员不参与，那么用户必须了解仔细测试计划的重要性，预期测试结果的定义以及所有测试输出的记录，这一点尤为重要。

典型任务 - 用户站点测试

验收测试执行
测试结果评估
错误评估/解决方案
最终测试报告

5.2.7. 维护和软件变更

应用于软件时，术语维护与应用于硬件时的含义不同。硬件和软件的操作维护是不同的，因为它们的故障/错误机制是不同的。硬件维护通常包括预防性硬件维护操作，组件更换和更正。软件维护包括纠正，完善和自适应维护，但不包括预防性维护操作或软件组件更换。

对纠正软件中的错误和故障所做的更改是纠正性维护。为改善软件系统的性能，可维护性或其他属性而对软件所做的更改是完善的维护。软件更改使软件系统在变更的环境中可用是自适应维护。

当对软件系统进行更改时，无论是在初始开发期间还是在发布后维护期间，都应进行充分的回归分析和测试，以证明未参与更改的软件部分未受到不利影响。这是评估实施变更的正确性的测试的补充。

每个软件更改所需的特定验证工作取决于更改类型，受影响的开发产品以及这些产品对软

件操作的影响。设计结构的仔细和完整的文档以及各种模块，接口等的相互关系可能会限制进行更改时所需的验证工作。需要的努力程度

完全验证更改还取决于记录和存档原始软件验证的程度。例如，如果可用于执行后续回归测试，则需要存档测试文档，测试用例以及先前验证和验证测试的结果。未能存档此信息供以后使用可能会显着增加更改后重新验证软件的工作量和费用。

除了作为标准软件开发过程一部分的软件验证和验证任务外，还应解决以下附加维护任务：

- **软**件验证计划修订 - 对于先前已验证的软件，应修订现有软件验证计划以支持修订软件的验证。如果不存在先前的软件验证计划，则应建立此类计划以支持修订软件的验证。

- **异**常评估 - 软件组织经常维护文档，例如描述发现的软件异常的软件问题报告以及修复每个异常所采取的具体纠正措施。但是，通常会重复出现错误，因为软件开发人员不会采取下一步措施来确定问题的根本原因，并进行必要的流程和程序更改以避免问题再次发生。软件异常应根据其严重程度及其对系统操作和安全性的影响进行评估，但也应将其视为质量体系中过程缺陷的症状。异常的根本原因分析可以识别特定的质量系统缺陷。在确定趋势（例如，类似软件异常的重现）的情况下，必须实施适当的纠正和预防措施并记录在案，以避免再次出现类似的质量问题。（见21 CFR 820.100。）

- **问**题识别和解决方案跟踪 - 应记录在维护软件期间发现的所有问题。应跟踪每个问题的解决方案，以确保其固定，历史参考和趋势。

- **建**议的变更评估 - 应评估所有建议的修改，增强或补充，以确定每次变更对系统的影响。此信息应确定需要迭代验证和/或验证任务的程度。

- **任**务迭代 - 对于批准的软件更改，应执行所有必要的验证和验证任务，以确保正确实施计划的更改，所有文档都是完整且最新的，并且软件性能不会发生不可接受的更改。

- **文**档更新 - 应仔细检查文档以确定哪些文档受到更改的影响。应根据配置管理程序更新受影响的所有已批准文件（例如，规格，测试程序，用户手册等）。在进行任何维护和软件更改之前，应更新规范。

第6节。 验证自动化过程设备和质量系统软件

质量体系法规要求“当计算机或自动数据处理系统用作生产或质量体系的一部分时，[设备]制造商应根据既定协议验证计算机软件的预期用途。”（见21CFR§820.70（I））。自1978年以来，这一直是FDA医疗器械良好生产规范（GMP）法规的监管要求。

除上述验证要求外，实施设备制造商生产过程或质量体系的一部分（或用于创建和维护任何其他FDA法规要求的记录）的计算机系统受电子记录的约束;电子签名监管。（参见21 CFR Part 11.）当以电子方式创建或维护记录时，本法规规定了额外的安全性，数据完整性和验证要求。应仔细考虑这些附加的第11部分要求，并将其包含在任何自动记录保持系统的系统要求和软件要求中。系统验证和软件验证应证明已满足所有第11部分要求。

计算机和自动化设备广泛用于医疗设备设计，实验室测试和分析，产品检验和验收，生产和过程控制，环境控制，包装，标签，可追溯性，文件控制，投诉管理以及许多其他方面。质量体系。自动化工厂车间操作越来越多地涉及在以下方面广泛使用嵌入式系统：

- 可编程逻辑控制器
- 数字功能控制器;
- 统计过程控制;
- 监督控制和数据采集;
- 机器人;
- 人机界面;
- 输入/输出设备;和
- 电脑操作系统。

软件工具经常用于设计，构建和测试进入自动医疗设备的软件。许多其他商业软件应用程序，例如文字处理器，电子表格，数据库和流程图软件，用于实现质量系统。所有这些应用程序都需要进行软件验证，但每种应用程序使用的验证方法可能差异很大。

无论是生产或质量系统软件是由设备制造商在内部开发，由承包商开发，还是现成购买，都

应使用基本原则开发

本指南其他部分概述。设备制造商在定义如何完成该软件的验证方面具有自由度和灵活性，但验证应该是决定软件开发方式和方式以及将从谁购买软件的关键考虑因素。软件开发人员定义了生命周期模型。验证通常由以下人员支持：

- 验证软件开发生命周期每个阶段的输出;和
- 在设备制造商的预期使用环境中检查已完成软件的正确操作。

6.1. 需要多少验证证据？

验证工作的水平应与自动化操作带来的风险相称。除了风险之外的其他因素，例如过程软件的复杂性以及设备制造商依赖于自动化过程来生成安全有效设备的程度，确定作为验证一部分所需的测试的性质和程度努力。自动化过程的记录要求和风险分析有助于确定证据的范围
需要表明该软件已经过验证，符合其预期用途。例如，如果设备制造商可以证明随后在发布之前根据规范完全验证了操作的输出，则自动铣床可能需要非常少的测试。另一方面，可能需要进行广泛的测试：

- 全厂电子记录和电子签名系统;
- 用于灭菌周期的自动控制器;要么
- 自动测试设备，用于在生命维持/生命支持设备中检查和验收成品电路板。

许多商业软件应用程序可用作质量系统的一部分（例如，用于质量系统计算的电子表格或统计软件包，用于趋势分析的图形包，或用于记录设备历史记录或用于投诉管理的商业数据库）。此类软件所需的验证证据的范围取决于设备制造商对该软件的文档预期用途。例如，选择不使用供应商提供的软件所有功能的设备制造商只需要验证将要使用的那些功能，并且设备制造商依赖于软件结果作为生产或质量系统的一部分..但是，即使不使用这些软件功能，高风险应用程序也不应在具有未经验证的软件功能的相同操作环境中运行。当在同一操作环境中使用高风险应用程序和低风险应用程序时，可能需要考虑风险缓解技术，如内存分区或其他资源保护方法。升级软件或对软件进行任何更改时，设备制造商应考虑这些更改如何影响软件的“已使用部分”，并且必须重新确认所使用软件的那些部分的验证。（见21CFR§820.70（i）。）

6.2. 定义的用户需求

软件验证的一个非常重要的关键是文档化的用户需求规范，它定义了：

- 软件或自动化设备的“预期用途”;和
- 设备制造商在多大程度上依赖于该软件或设备来生产高质量的医疗设备。

设备制造商（用户）需要定义预期的操作环境，包括任何所需的硬件和软件配置，软件版本，实用程序等。用户还需要：

- 记录系统性能，质量，错误处理，启动，关闭，安全等要求;
- 识别任何与安全相关的功能或特征，例如传感器，报警，互锁，逻辑处理步骤或命令序列;和
- 确定确定可接受绩效的客观标准。

验证必须按照文件化的协议进行，验证结果也必须记录在案。（参见21CFR§820.70（i）。）应记录测试用例，使系统根据预先确定的标准对其性能提出质疑，特别是对于其最关键的参数。测试用例应解决错误和报警条件，启动，关闭，所有适用的用户功能和操作员控制，潜在的操作员错误，允许值的最大和最小范围，以及适用于设备预期用途的压力条件。应该执行测试用例并记录和评估结果，以确定结果是否支持软件验证其预期用途的结论。

设备制造商可以使用他们自己的人员进行验证，或者可以依赖于第三方，例如设备/软件供应商或顾问。无论如何，设备制造商保留了确保生产和质量系统软件的最终责任：

- 根据特定用途的书面程序进行验证;和
- 将在所选应用程序中按预期执行。

设备制造商应该有文档，包括：

- 定义用户要求;
- 使用验证协议;
- 验收标准;
- 测试用例和结果;和

- 验证摘要

客观地证实该软件已经过验证，符合其预期用途。

6.3. 验证现成的软件和自动化设备

设备制造商使用的大多数自动化设备和系统由第三方供应商提供，并且是现成的（OTS）购买的。设备制造商负责确保OTS软件开发人员使用的产品开发方法适用于设备制造商对该OTS软件的预期用途。对于OTS软件和设备，设备制造商可能会或可能不会访问供应商的软件验证文档。如果供应商可以提供有关其系统要求，软件要求，验证过程及其验证结果的信息，则医疗设备制造商可以将该信息用作其所需验证文档的起点。供应商的生命周期文档（例如测试协议和结果，源代码，设计规范和需求规范）可用于确定软件已经过验证。但是，商业设备供应商通常无法提供此类文档，或者供应商可能拒绝共享其专有信息。

在可能的情况下，根据所涉及的设备风险，设备制造商应考虑审核供应商在OTS软件构建中使用的设计和开发方法，并应评估为OTS软件生成的开发和验证文档。此类审核可由设备制造商或合格的第三方进行。审核应证明供应商执行OTS软件的验证和验证活动的程序和结果对于使用该软件生产的医疗设备的安全性和有效性要求是适当和充分的。

一些不习惯在受监管环境中运行的供应商可能没有可记录的生命周期过程，可以支持设备制造商的验证要求。其他供应商可能不允许进行审核。如果供应商无法获得必要的验证信息，设备制造商将需要执行足够的系统级“黑盒”测试，以确定软件满足其“用户需求和预期用途”。对于许多应用程序，单独的黑盒测试不是足够。根据所生产设备的风险，OTS软件在流程中的作用，审核供应商的能力以及供应商提供的信息的充分性，OTS软件或设备的使用可能适合也可能不适用，尤其是如果有合适的替代品可供选择。如果供应商终止其支持，设备制造商还应考虑对OTS软件的持续维护和支持的影响（如果有的话）。

对于一些现成的软件开发工具，例如软件编译器，链接器，编辑器和操作系统，设备制造商进行的详尽的黑盒测试可能是不切实际的。
如果没有这样的测试 - 验证工作的关键要素 - 可能无法验证这些软件工具。但是，通过其他方式可以令人满意地推断出它们的正常操作。例如，编译器经常通过独立的第三方测试进行认证，商业软件产品可能具有“错误列表”，系统要求和供应商提供的其他操作信息，可以与设备制造商的预期用途进行比较，以帮助集中精力“黑箱“测试工作。现成的操作系统无需作为单独的程序进行验证。但是，应用程序软件的系统级验证测试应该解决所有使用的操作系统服务，包括最大加载条件，文件操作，系统错误处理

条件和可能适用于应用程序的预期用途的内存约束。

有关更多详细信息，请参阅附录A中的生产和过程软件参考。

附录a - 参考文献

食品药品管理局参考文献

医疗器械制造商的设计控制指南， 食品和药物管理局器械和放射卫生中心，1997年3月。

通过设计，医疗器械中的人为因素介绍， 食品和药物管理局器械和放射卫生中心，1997年3月。

电子记录;电子签名最终规则，62联邦登记册13430（1997年3月20日）。

*计算机系统和软件开发术语表，*1995年8月，区域业务办公室，监管事务办公室，食品药品管理局，实地调查司。

医疗保健软件上市前提交内容指南 设备， 设备评估办公室，设备和放射卫生中心，食品和药物管理局，1998年5月。

*行业指南，FDA审查员和现有软件使用合规性 医疗设备，*设备评估办公室，设备和放射卫生中心，食品和药物管理局，1999年9月。

*过程验证一般原则指南，*1987年5月，药物和生物制品中心，设备和放射卫生中心，食品和药物管理局。

医疗设备;现行良好生产规范（CGMP）最终规则;质量体系法规，61联邦登记册52602（1996年10月7日）。

1997年1月，血液培养计算机软件，生物制品评估和研究中心，食品和药物管理局上市前通知的审查员指南

学生手册1，课程INV545，计算机系统验证，人力资源开发司，法规事务办公室，食品和药物管理局，1997年。

技术报告，软件开发活动，实地调查司，区域业务办公室，法规事务办公室，食品药品管理局，1987年7月。

其他政府参考资料

W. Richards Adrion，Martha A. Branstad，John C. Cherniavsky。国家统计局特刊500-75，计算机软件验证，验证和测试，计算机科学与技术中心，计算机科学与技术研究所，国家标准局，美国商务部，1981年2月。

Martha A. Branstad，John C Cherniavsky，W。Richards Adrion，NBS特刊500-56，个人程序员验证，验证和测试，计算机科学与技术中心，国家标准局计算机科学与技术研究所，美国商务部，1980年2月。

JL Bryant，NP Wilburn，适用于核工业的软件质量保证技术手册，NUREG / CR-4640，美国核管理委员会，1987年。

H.Hecht，et.al.，High Integrity Systems的验证和验证指南。NUREG / CR-6293。1995年为美国核管理委员会做准备。

H.*Hecht等，“核电厂安全系统使用软件语言评论指南”，最终报告。NUREG / CR-6463。1996年为美国核管理委员会做准备。*

JD Lawrence，WL人员，生产高度可靠软件的行业方法调查，NUREG / CR-6278，美国核管理委员会，1994年。

JD Lawrence，GG Preckshot，安全关键软件的设计因素，NUREG / CR-6294，美国核管理委员会，1994年。

编辑Patricia B. Powell。国家统计局特刊500-98，计划软件验证，验证和测试，计算机科学与技术中心，计算机科学与技术研究所，国家标准局，美国商务部，1982年11月。

编辑Patricia B. Powell。国家统计局特刊500-93，软件验证，验证和测试技术与工具参考指南，编程科学与技术中心，计算机科学与技术研究所，国家标准局，美国商务部，1982年9月。

Delores R. Wallace，Roger U. Fujii，NIST特刊500-165，软件验证和验证：它在计算机保证中的作用及其与软件项目管理标准的关系，国家计算机系统实验室，美国国家标准与技术研

究所商业，1995年9月。

Delores R. Wallace，Laura M. Ippolito，D。Richard Kuhn，NIST特刊500-204，高完整性软件，标准和指南，计算机系统实验室，国家研究所

标准与技术，美国商务部，1992年9月。

Delores R. Wallace，et.al。NIST特刊500-234，软件验证和验证过程的参考信息。美国商业部国家标准与技术研究所计算机系统实验室，1996年3月。

Delores R. Wallace，编辑。NIST特刊500-235，结构化测试：使用Cyclomatic Complexity Metric的测试方法。美国商业部国家标准与技术研究所计算机系统实验室，1996年8月。

国际和国家共识标准

A*NSI / ANS-10.4-1987，核工业科学和工程计算机程序的验证和验证指南，美国国家标准协会，1987年。*

ANSI / ASQC标准D1160-1995，正式设计评论，美国质量控制协会，1995年。

A*NSI / UL 1998：1998，可编程组件软件安全标准，*
Underwriters Laboratories，Inc.，1998。

AS 3563.1-1991，软件质量管理体系，第1部分：要求。由澳大利亚标准协会出版[澳大利亚标准协会]，1 The Crescent，Homebush，NSW 2140。

AS 3563.2-1991，软件质量管理体系，第2部分：实施指南。由澳大利亚标准协会出版[澳大利亚标准协会]，1 The Crescent，Homebush，NSW 2140。

I*EC 60601-1-4：1996，医用电气设备，第1部分：安全通用要求，4。并列标准：可编程电气医疗系统。国际电工委员会，1996年。*

I*EC 61506：1997，工业过程测量和控制 - 应用软件的文档。国际电工委员会，1997年。*

I*EC 61508：1998，电气/电子/可编程电子安全相关系统的功能安全。国际电工委员会，1998年。*

IEEE Std 1012-1986，软件验证和验证计划，电气和电子工程师协会，1986年。

/EEE标准集，软件工程，电气和电子工程师协会，1994。ISBN 1-55937-442-X。

ISO 8402：1994，质量管理和质量保证 - 词汇。国际标准化组织，1994年。

I*SO 9000-3：1997，质量管理和质量保证标准 - 第3部分：ISO 9001：1994应用于计算机软件开发，供应，安装和维护的指南。国际标准化组织，1997年。*

I*SO 9001：1994，质量体系 - 设计，开发，生产，安装和服务质量保证模型。国际标准化组织，1994年。*

I*SO 13485：1996，质量体系 - 医疗器械 - ISO 9001的应用的特殊要求。国际标准化组织，1996年。*

ISO / IEC 12119：1994，信息技术 - 软件包 - 质量要求和测试，联合技术委员会ISO / IEC JTC 1，国际标准化组织和国际电工委员会，1994。

ISO / IEC 12207：1995，信息技术 - 软件生命周期过程，联合技术委员会ISO / IEC JTC 1，分技术委员会SC 7，国际标准化组织和国际电工委员会，1995年。

ISO / IEC 14598：1999，信息技术 - 软件产品评估，联合技术委员会ISO / IEC JTC 1，分技术委员会SC 7，国际标准化组织和国际电工委员会，1999年。

I*SO 14971-1：1998，医疗器械 - 风险管理 - 第1部分：风险分析的应用。*
国际标准化组织，1998年。

*机*载系统和设备认证中的软件考虑因素。RTCA特别委员会167。华盛顿特区的RTCA公司电话：202-833-9339。文件号RTCA / DO-178B，1992年12月。

生产过程软件参考

GLP原理在计算机系统中的应用，环境专着
*#*116，经济合作与发展组织（经合组织），1995年。

George J. Grigonis，Jr。，Edward J. Subak，Jr。和Michael Wyrick，“用于受监管操作的计

算机系统的验证关键实践”，制药技术，1997年6月。

药品加工，参考材料和计算机化系统检验指南

为调查员提供培训援助，药品质量合规司，药品办公室，药品办公室，国家药品和生物制品中心和实地调查司，外勤支助副主任，区域业务执行主任，食品和药品管理局，1983年2月。

Daniel P. Olivier，“验证过程软件”，FDA调查员课程：医疗器械过程验证，食品和药物管理。

GAMP制药生产自动化系统验证指南，版本V3.0，良好自动化制造规范（GAMP）论坛，1998年3月：

第1卷，第1部分：用户指南

第2部分：供应商指南

第2卷：用户和供应商的最佳实践。

*技*术报告第18号，计算机相关系统的验证。PDA计算机相关系统验证委员会。PDA Journal of Pharmaceutical Science and Technology，Volume 49，Number 1，January-February 1995 Supplement。

*验*证合规性1995年，国际验证论坛，公司

一般软件质量参考

B*oris Beizer，Black Box Testing，软件和系统功能测试技术，John Wiley & Sons，1995。ISBN 0-471-12094-4。*

Boris Beizer，软件系统测试和质量保证，国际汤姆森计算机出版社，1996.ISBN 1-85032-821-8。

Boris Beizer，软件测试技术，第二版，Van Nostrand Reinhold，1990.ISBN 0-442-20672-0。

Richard Bender，Writing Testable Requirements，Version 1.0，Bender & Associates，Inc.，Larkspur，CA 94777,1996。

Frederick P. Brooks，Jr。，The Mythical Man-Month，Essays on Software Engineering，

AddisonWesley Longman，Anniversary Edition，1995。ISBN 0-201-83595-9。

Silvana Castano，et.al.，Database Security，ACM Press，Addison-Wesley Publishing Company，1995.ISBN 0-201-59375-0。

用于非临床安全性评估的计算机化数据系统，当前概念和质量保证，药物信息协会，宾夕法尼亚州Maple Glen，1988年9月。

M*S Deutsch，软件验证和验证，现实项目方法，Prentice Hall，1982。*

Robert H.Dunn和Richard S.Ullman，TQM for Computer Software，Second Edition，McGraw-Hill，Inc.，1994。ISBN 0-07-018314-7。

Elfriede Dustin，Jeff Rashka和John Paul，Automated Software Testing - Introduction，Management and Performance，Addison Wesley Longman，Inc.，1999。ISBN 0-201-43287-0。

Robert G. Ebenau和Susan H. Strauss，Software Inspection Process，McGraw-Hill，1994。ISBN 0-07-062166-7。

Richard E.Fairley，Software Engineering Concepts，McGraw-Hill Publishing Company，1985.ISBN 0-07-019902-7。

Michael A.Friedman和Jeffrey M.Voas，软件评估 - 可靠性，安全性，可测试性，Wiley-Interscience，John Wiley & Sons Inc.，1995.ISBN 0-471-01009-X。

Tom Gilb，Dorothy Graham，Software Inspection，Addison-Wesley Publishing Company，1993.ISBN 0-201-63181-4。

R*obert B. Grady，项目管理和过程改进的实用软件度量标准，PTR Prentice-Hall Inc.，1992.ISBN 0-13-720384-5。*

L*es Hatton，更安全C：为高完整性和安全关键系统开发软件，*McGraw-Hill Book Company，1994。ISBN 0-07-707640-0。

Janis V. Halvorsen，医疗器械行业软件需求规范文档模型，会议论文集IEEE SOUTHEASTCON '93，Banking on Technology，1993年4月4日至7日，北卡罗来纳州夏洛特市。

D*ebra S. Herrmann，软件安全性和可靠性：关键工业部门的技术，方法和标准，IEEE计算机学会，1999年.ISBN 0-7695-0299-7。*

Bill Hetzel，软件测试完全指南，第二版，Wiley-QED出版物，John Wiley & Sons，Inc.，1988。ISBN 0-471-56567-9。

Watts S. Humphrey，软件工程学科。Addison-Wesley Longman，1995.ISBN 0-201-54610-8。

Watts S. Humphrey，管理软件过程，Addison-Wesley出版公司，1989年。国际标准书号0-201-18095-2。

Capers Jones，软件质量，分析和成功指南，国际汤姆森计算机出版社，1997年.ISBN 1-85032-867-6。

JM Juran，Frank M. Gryna，Quality Planning and Analysis，Third Edition，McGraw-Hill，1993。ISBN 0-07-033183-9。

Stephen H. Kan，软件质量工程中的度量和模型，Addison-Wesley出版公司，1995.ISBN 0-201-63339-6。

Cem Kaner，Jack Falk，Hung Quoc Nguyen，Testing Computer Software，Second Edition，Vsn Nostrand Reinhold，1993。ISBN 0-442-01361-2。

Craig Kaplan，Ralph Clark，Victor Tang，软件质量的秘密，40来自IBM的创新，McGraw-Hill，1995。国际标准书号0-07-911795-3。

Edward Kit，Real World中的软件测试，Addison-Wesley Longman，1995。ISBN 0-201-87756-2。

A*lan Kusinitz，“软件验证”，医疗器械质量系统的当前问题，*
医学仪器促进协会，1997年.ISBN 1-57020-075-0。

Nancy G. Leveson，Safeware，System Safety and Computers，Addison-Wesley Publishing Company，1995.ISBN 0-201-11972-2。

Michael R. Lyu，编辑，软件可靠性工程手册，IEEE计算机学会出版社，McGraw-Hill，1996。国际标准书号0-07-039400-8。

S*teven R. Mallory，医疗保健制造业软件开发和质量保证，Interpharm Press，Inc.，1994。ISBN 0-935184-58-9。*

Brian Marick，The Craft of Software Testing，Prentice Hall PTR，1995。ISBN 0-13-177411-5。Steve McConnell，Rapid Development，Microsoft Press，1996。ISBN 1-55615-900-5。

Glenford J. Myers，The Art of Software Testing，John Wiley & Sons，1979。ISBN 0-471-04328-1。

Peter G. Neumann，Computer Related Risks，ACM Press / Addison-Wesley Publishing Co.，1995.ISBN 0-201-55805-X。

D*aniel Olivier，执行软件审计，符合FDA要求的审计软件，计算机应用专家，加利福尼亚州圣地亚哥，1994年。*

William Perry，软件测试的有效方法，John Wiley & Sons，Inc。1995.ISBN 0- 471-06097-6。

W*illiam E. Perry，Randall W. Rice，生存软件测试的十大挑战，Dorset*

House Publishing，1997。国际标准书号0-932633-38-2。

Roger S.Pressman，Software Engineering，A Practitioner's Approach，Third Edition，McGraw-Hill Inc.，1992.ISBN 0-07-050814-3。

Roger S.Pressman，软件工程经理指南，McGraw-Hill Inc.，1993 ISBN 0-07-050820-8。

AP Sage，JD Palmer，软件系统工程，John Wiley & Sons，1990。

Joc Sanders，Eugene Curran，Software Quality，Addison-Wesley Publishing Co.，1994。ISBN 0-201-63198-9。

K*en Shumate，Marilyn Keller，软件规范和设计，实时系统的纪律方法，John Wiley & Sons，1992。ISBN 0-471-53296-7。*

Dennis D. Smith，Designing Maintainable Software，Springer-Verlag，1999。ISBN 0-387-98783-5。

Ian Sommerville，Software Engineering，Third Edition，Addison Wesley Publishing Co.，1989。ISBN 0-201-17568-1。

Karl E. Wiegers，创建软件工程文化，Dorset House Publishing，1996。ISBN 0-932633-33-1。

K*arl E. Wiegers，软件检查，软件检查提高质量，软件开发，1995年4月，第55-64页。*

Karl E. Wiegers，Software Requirements，Microsoft Press，1999。ISBN 0-7356-0631-5。

附录b - 开发团队

器械与放射卫生中心

合规办公室	斯图尔特克鲁普尔
设备评估办公室	James Cheng，Donna-Bea
Tillman健康与工业计划办公室	Bryan Benesch，Dick Sawyer
科技办公室	约翰默里
监督和生物识别办公室	霍华德出版社

药物评估和研究中心

医疗政策办公室	查尔斯斯奈普斯

生物制品评估与研究中心

合规与生物制品质量办公室	爱丽丝Godziemski

法规事务办公室

区域业务办公室	大卫伯格森，Joan Loreng