INTERNATIONAL SOFTWARE QUALITY ASSURANCE SERIES

ISO 9001国际标准和软件质量保证

● [英] Darrel Ince 著
● 李月芳 汪玲玲 等译
● 刘德贵 审校

ISO 9001 and Software Quality Assurance

McGraw-Hill

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

ISO 9001
and Software Quality Assurance

国际软件质量保证丛书

ISO 9001 国际标准 和 软件质量保证

[英] Darrel Ince 著
李月芳 汪玲玲 等译
刘德贵 审校

電子工業出版社

内 容 简 介

本书共分 21 章。根据 ISO 9001 国际标准外部质量标准 20 章节内容的要求，对软件质量、管理职责、开发设计、文档编制、系统采购、测试验收和培训服务等有关软件质量保证的各个方面均作了透彻的分析。是广大软件开发人员和管理人员必读之书，也值得公司高层领导，广大计算机用户和大专院校师生参考阅读。

ISO 9001 and Software Quality Assurance

[英] Darrel Ince 著

McGraw-Hill International (UK) Limited 1994 出版

*

ISO 9001 国际标准和软件质量保证

李月芳 汪玲玲 等译

刘德贵 审 校

责任编辑 路 石

*

电子工业出版社出版(北京市万寿路)

电子工业出版社发行 各地新华书店经销

北京大中印刷厂印刷

*

开本：787×1092 毫米 1/16 印张：7 字数：179 千字

1996 年 8 月第一版 1996 年 8 月第一次印刷

印数：3000 册 定价：15.00 元

ISBN 7-5053-3500-6/TP.1402

著作权合同登记号图字：01-1995-159

译者序言

我国在1992年等同采用了ISO 9000系列国际标准，并以双编号形式GB/T1 9000-ISO 9000命名这套标准系列，这标志着我国质量管理标准化工作已经同国际接轨。我国等同采用了ISO 9000-3作为计算机软件质量管理和质量保证的标准，其编号为GB/T1 9000.3-ISO 9000-3。

为了宣传和贯彻软件质量保证，根据我们同美国McGraw-Hill著名出版公司签约，翻译出版ISO 9001 and Software Quality Assurance：《ISO 9001国际标准和软件质量保证》一书。

全书共分21章，根据ISO 9001外部质量标准20章节内容的要求，对软件质量、管理职责、开发设计、文档编制、系统采购、测试验收和培训服务等有关软件质量保证的各个方面均作了透彻的分析。

作者Darre Ince教授是国际知名软件质量研究专家，著有多部学术著作。

该书内容丰富、题材广泛、联系实际、通俗易懂，适合于广大软件开发人员和管理人员，以及有关大专院校的师生参考使用。

参加本书翻译工作的还有白小燕、李珍华、孙燕谋。

质 量 法 庭

质量法庭很高兴与 McGraw-Hill 公司共同出版这本软件质量保证方面的书。

本机构的宗旨是：通过会员与会员以及会员与其他类似机构间的信息交换，帮助会员提高其产品和服务的质量。

质量法庭已有 200 多个会员，这些组织来自各工业及商业部门，有当地政府的，也有国家的。这些组织主要是英国的，但也有从其他欧洲国家吸收的会员。

这套丛书旨在为作者提供一个出版既实用又能反映当前技术水平的著作的机会，以便于质量法庭的会员及其他组织通过交换信息，提出可促进信息技术领域内质量事业发展的新观点而从中受益。

质量法庭出版这套丛书的目的是促进软件团体的讨论，以使各工业部门的产品质量和服务质量都有所提高。尽管并不是每本书的所有观点都得到了认同，但能被收入该丛书还是很荣幸的。

如果您想进一步了解质量法庭，请与下述地址联系：

Quality Forum
17 St Catherine's Rd
Ruislip
Middlesex HA4 7RX
UK
Tel：+44(0)895 635222
Fax：+44(0)895 679178

目　录

第一章　软件质量和质量保证

1.1　引言

本书论述了 ISO 9001 质量标准及其在软件开发方面的应用。五年来，世界上采用该标准的软件开发人员数量迅速增长。本书各章将分述该标准中的二十个质量体系要求。本章将阐述“质量”一词的含义及其在软件产品中的应用。这是很关键的一章：它阐述了如何把软件质量分为若干个**质量要素**(quality factor)(有时被称为**质量属性**(quality attribute))以及质量体系是如何按这些要素进行分级的。

1.2　质量的含义

如果读者已经看过许多质量保证方面的书籍，就会发现它们通常对质量的含义持有一致的看法。通常它们都使用了**“适合目的**(fitness for purpose)”一词。也即，优质产品(比如一辆汽车、一台冰箱、一个吹风机或任何其它制品)就是能满足用户期望的产品。“适合目的”是一个重要概念，它构成了质量保证的核心原则。在本章的后文中读者会发现它并不是高品质产品的唯一特性。然而，在研究全部问题之前，应研究一下“适合目的”的特定含义。

“适合目的”的含义是，应在某处对制品预期完成的目的加以说明。对于很简单的制品，该说明一般就包含在制品的名称中。例如，“吹风机”一词就基本上表明了它是一种用于吹干湿头发的装置。对于有些制品，其用途的说明可能包含在**用户手册**(user manual)中。除了详细说明如何使用该制品外，用户手册中可能还包括一些技术指标。对于较大的制品(如软件或硬件系统)，其用途一般是包含在一个通常称为**需求规格说明**(requirement specification)——有时称为**系统规格说明**(sysytem specification)的文档中。在阅读本章时，对于需求规格说明只需要了解具体系统的用途说明，以及象响应时间这样的与软件系统约束有关的说明即可。需求规格说明是软件项目中最重要的文档，质量体系就是围绕着这个文档展开的。

现代质量保证观点要比“适合目的”这一观点复杂得多。一个高质量的产品与多个质量要素有关。有的质量要素在需求规格说明中有明确的规定；有的质量要素是**约定俗成**的(cultural)，是否写入需求规格说明往往取决于人们对其使用的熟知程度和用户的共同经验；还有一些质量要素开发者认为重要但用户并不认为重要，因而需求规格说明中不予规定。

不妨为这三类质量要素各举一个例子。第一类是包含在需求规格说明中的质量要素，如**可移植性**(portability)。软件系统的用户可能要求系统能在各种硬件结构上运行。因此应在需求规格说明中对这些硬件结构加以描述。

第二类是约定俗成的质量要素，如**可用性**(usability)。因为用户可能用过比较易于和计算机进行通信的系统，所以在需求规格说明中可能就不详细规定接口如何实现。关于如何使用**WIMP**(Wait Interface Message Processer)接口或**逐条命令**(line by line command)接口，可能会有一个简要的说明，但由于开发者可能用过比较易于和计算机进行通信的系统，需求规格说明中通常将接口的详细情况略去。

第三类是开发者可能有利可图但用户并不直接有利的质量要素，如**复用性**(reusability)。复用性是指把为一个软件系统构造的模块转换到另一软件系统中去的能力。例如，软件开发者目前可能正在为一用户构造软件系统，该软件系统与打算为另一用户构造的软件系统有些类似。如果第二个系统正在向多个开发者招标，则第一个系统的生产厂家由于能保证第一个系统具有很高的复用性，在投标的过程中将占有较大的商业优势。当然这个优势将通过很低的出价获得，因为基本上不需要重新编程。这类质量要素对用户来说顶多是间接受益，因此不必写在需求规格说明中。然而，在考虑复用性的情况下，尽管在需求规格说明中可能并未要求复用性，用户可能会在收取复用性软件的许可证费用方面有受益。

需要指出的是，这三类质量要素并非一成不变的。质量要素到底应属于哪一类取决于用户、用户环境、应用领域和开发者环境。例如，人人都希望系统是可用的，因此本书将可用性归为约定俗成的质量要素。通常对多数软件系统来说都是如此。然而，对于有些用于**安全性应用**(safety-Critical application)中的软件系统来说，可用性非常重要，必须写入需求规格说明中。例如，在规格说明中要规定由系统操作员造成错误命令的不可接受频度的数量的度量标准。

尽管这三类质量要素并非一成不变，质量体系要认识到，确实存在三类质量要素，并确保恰好在项目的刚开始，由项目经理在确定项目所需的质量控制之前，仔细考虑一下所有可能需要的质量要素是非常重要的。项目经理不应认为需求规格说明中含有上述实例中说明的“适合目的”的要求就足够了。

现在考虑一些重要的质量要素。第一个质量要素是**正确性**(correctness)。也即，软件系统确实符合其需求规格说明，当然任何系统中都有该质量要素。

下一个质量要素是**可维护性**(maintainability)或**可修改性**(modifiability)。它是指软件系统可以更改的难易程度。最初是因用户发现了错误才对软件系统进行更改。然而，近十年来，确实出现了许多要求其它类型性的更改。尽管人们期望随着技术的进步这类变动的程度会轻些，但因出错而引起的更改总是在所难免的。还有由于需求的易变性引起的更改。似乎近于绝妙的是，能表明软件开发者已开发出优质软件系统的标志之一，是他们能不厌其烦地接受用户因需求的改动而提出的修改要求。结果通常是用实现新功能来解决问题。

开发者还将接受用户因外部环境的改动而提出的更改要求。例如一个已经交付使用的**财会软件包**(accounting package)，虽然它满足了用户原来的要求，但因税法修改了，所以也必须对该软件包进行修改。由此可见，由于用户对成功系统的要求越来越高和外部环境的改变，需求的更改可能都属于高层应用的修改。还有一种更改就是：不改变系统功能而对系统进行某种改进的更改。为缩短响应时间对系统进行调整性修改，或为适应新的输出设备而重写**设备处理程序**(device handler)就是这种更改的实例。

第一种更改是为了纠正错误而进行的修改，叫作**纠错性更改**(corrective change)；第二种是开发者为反映要求的改变而进行的修改，叫作**适应性更改**(adaptive change)；第三种更改叫作**完善性更改**(perfective change)，旨在对系统进行改进。

重要的不是更改的分类，而是许多软件系统因这些更改**维护**(maintenance)工作量很大，而其中绝大多数更改是适应性的修改。况且，许多更改不仅在维护时出现，对于开发时间较长的项目，还会在项目的开发期内出现。正因为如此，可维护性这一质量要素在每个软件体系中都应占据重要的位置。本书认为可维护性的重要性仅次于正确性。

可维护性是用户通常只能间接受益的质量要素之一。很少有用户真的在给软件开发者的指导书中含有可维护性方面的要求，当然要撇开那些由开发者自己对系统进行维护的情况。然

而，值得指出的是：如果软件开发者在开发系统时将可维护性考虑在内，则在规定的时间和预算内，交付一个正确系统的可能性就要大得多，开发人员因而将间接受益。例如，如果系统的更改需要的时间较长，则很可能发生这样的情况：如果发现了许多错误，则项目的开发时间就会增加并超过预计的软件交付时间。

另一个前文已经提到的质量要素是**可移植性**。它是将系统从一个硬件平台转换到另一硬件平台所需做的工作。一般地说，可移植性应在需求规格说明中加以详细说明。然而，由于商业原因，有时它是一个对开发者有好处而用户只是间接受益的质量要素；如前文中的那个例子所述，在系统中考虑可移植性只是为了在投标另一个系统的过程中具有竞争优势。

另一个质量要素为**可测试性**(testability)，它对开发者直接有利，极少有用户直接规定该质量要素。可测试性描述了系统或系统的一部分易被测试的程度。例如，有一系统，其需求规格说明写得很差，而且很多内容都是模棱两可的或是陈词滥调的，则对该系统就很难进行测试，因为负责系统测试的人员在确定需求规格检查测试时将会遇到较大困难。

另一个重要的质量要素是**可用性**。这是一个学习、使用和中断一个功能操作由系统做的工作。可用性常常是系统的主要问题：许多软件开发者在项目的开始都倾向于只考虑系统的功能，而在项目的开发将要结束时选了一个不合适的接口。如果要证明对质量要素的执着要求和关于"'适合目的'要求并不是质量保证完全充分的依据"这一观点是正确的，可举这样一个例子：有一系统，运行后满足需求规格说明中的所有功能，但是由于接口太差，致使该系统无法使用。

另一个重要的质量要素是**可靠性**(reliability)。可靠性描述的是软件系统持续运行而其功能基本上不中断的能力。在某些安全性系统中，一般期望该质量要素要有完整的表达，通常是用**平均故障间隔时间**(mean time between failure)等表示，并且用公制计量单位规定。对于其他系统，该质量要素将属于第二类质量要素，通常不在需求规格说明中加以规定，但用户将假设系统是高可靠性的。

另一个重要的质量要素是**有效性**(efficency)。它用于描述在应用中对计算机资源(即文件空间、内存及处理时间)的使用程度。这是一个难于分类的质量要素：许多需求规格说明中都详述了可用硬件的数量和所需的响应时间。然而，本书认为有效性具有第二类质量要素的特性，因为用户都希望(虽然并未明确表示)软件开发者充分利用所能得到的计算机资源。

完整性(integrity)是另一个重要的质量要素。该术语用于描述系统及其数据免受未授权用户存取的程度。这也是一个无法精确分类的质量要素。有些系统(如商用系统)的需求规格说明就描述了系统允许存取的程度。然而，通常有一个不成文的看法：交付的系统应能免受未授权用户或入侵者窜改。

复用性也是一个日趋重要的质量要素。它描述一个系统中大部分软件可移至另一系统的难易程度。它通常是一个用户只能间接受益的质量要素。

最后还要说明的质量要素是**互操作性**(interoperability)。它是一个软件系统与另一个系统互连操作的能力，例如**电子数据表软件**(spreadsheet)。通常这是一个在需求规格说明中规定的质量要素，用户只能间接受益。

这里只是简要地讨论一下什么是质量要素。许多读者可能已经觉得有点枯燥无味；对质量要素长篇大论一番的确是一种枯燥无味的工作。

然而，为何本书如此早地讨论质量要素呢？有一个很实际的原因就是与质量体系的灵活性有关系。通常，质量体系体现在一个叫作**质量手册**(quality manual)的文档中。在软件项目的开

始，为确保将要交付的软件的质量，项目经理应确定要用到质量手册中的哪些要素。为此，质量经理首先应该检查质量要素包含的内容；确定质量体系中应该用于强化所需质量要素水平的那些部分；确定质量体系中不应该使用的部分；甚至开发一些质量手册中没有的新的质量控制内容。对将要增添到质量体系中的**质量要素**，以及它们在质量体系中所处的级别的考虑，通常都体现在检查**质量要素清单**(quality factor checklist)的内容中。在项目开始时，项目经理对这个检查清单内容的考虑将会推动质量要素的选择进程。

显然每个软件项目中都使用了质量体系的某些部分，例如需求规格说明标准总是需要的。然而，项目经理将根据用户、应用及所使用的软件技术决定使用、加强或省略质量体系中的哪些部分。为了理解质量要素和质量体系之间的这种联系，不妨看一看下面的几个例子。

第一个例子是一个开发人员决定开发一个已**确认**(valitation)为具有较高复用性的系统。通常这种确认是由项目经理在察看质量要素清单的过程中作出的，清单中包含与某一要素有关的一系列问题。当项目经理遇到复用性质量要素内容时，会有下述问题驱使项目经理考虑该要素：

- 我们将来要开发一个类似的软件系统吗？
- 我公司是否有开发**复用库**(reusable library)的方针？
- 我们目前正在投标一个与将要开发的系统相类似的软件系统吗？

不妨假设第三个问题的回答为“是”，而且项目经理决定使用像C++这样的具有高度复用性的**程序设计语言**(programming language)。但可能出现这样的情况：质量体系中没有C++的**程序设计标准**(programming standard)，需要制订一个程序设计标准或对现有的C语言程序设计标准进行修改，因为C++是以它为基础的。

第二个例子，开发一个**安全性系统**(safety-critical system)。这种系统必须具有极高的正确性和可靠性。在启动这样的项目之前，项目经理一读质量手册就会决定：因为该质量要素的要求很高，所以要将质量手册中能保证正确性的每一项技术都用上。这些合法的技术手段可能比项目经理通常在正确性要求较低的项目中(如一个用于文书应用的系统)所采用的那些手段高得多。

第三个例子是为使用键盘操作员的用户开发**基于事务处理的系统**(transaction-based system)，该系统将在键盘操作员短缺的地区使用。在阅读了由顾客提供的文档或与顾客的会谈之后，项目经理可能会决定该系统需要很高的可用性。这就意味着在软件项目的后期要进行一些**可用性研究**(usability study)并将**可用性测试**(usability testing)列入计划。要查询质量手册，任何与可用性有关的章节都要用到项目上，如果没有，则应进行开发编写。

因此，质量要素(更准确地说是对质量要素的一些考虑)构成了软件项目质量控制的基础。这些质量控制是各不相同的，可利用标准，也可利用工具。下一节对它们进行略微详细的一些说明。

1.3 质量和质量体系

质量体系应该是灵活的想法自然导致了对质量体系应如何进行工作的考虑。本节旨在阐述质量体系与软件开发之间的关系，为了保证读者正确地理解本书的后续章节，本节提供了一些研究原则。在考虑质量保证的核心问题时，有一个叫做**质量体系**(quality system)的内容，随

着人们对它的逐渐了解，也称之为**质量管理体系**(quality management system)。它是由能确保项目所生产的软件产品具有所要求的质量要素的管理结构、责任、工作、能力和资源组成，用户和开发者都将研究考虑它们。这意味着质量体系包括下述工作：

- 为确保遵循质量控制而进行的项目**审核**(auditing)。
- 为改进质量体系而进行的质量体系评审。
- 在质量保证领域所用人员的培训。
- 为确保执行质量保证工作人员正常发挥作用，解决所用资源。
- 对开发工作提供改进意见；例如需求规格说明采用新设想。
- 标准、步骤和准则的制订。
- 撰写说明目前的质量管理体系有效性的高级管理报告。
- 撰写能使管理人员对质量体系进行改进的高级质量管理报告。

这里列举了几种通常与质量管理体系有关的一些工作。

质量管理体系的具体情况将写入**质量手册**——有时将之误称为质量体系。该手册通常包括质量标准、有关项目开发工作以及对质量控制的详细说明。国际标准、质量管理体系和各个项目之间的关系如图 1.1 所示。

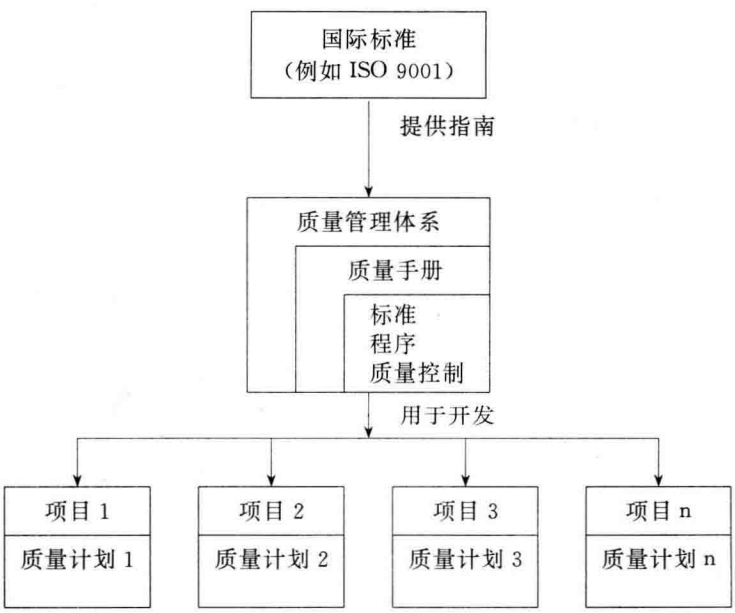

图 1.1　外部标准、质量管理体系和质量计划

象 ISO 9001 这样的国际标准，为公司如何组织其质量管理体系提供了指南。质量手册是质量管理体系的一个组成部分，阐明了可用于项目的各种标准和质量控制。在项目的形成阶段(通常是在策划中)，项目经理将确定重要的质量要素，并从质量手册中摘录必要的标准和步骤，以确保将要开发的软件产品含有这些质量要素。同时项目经理还将选定所需的质量控制。这些都写在一个叫作**质量计划**(quality plan)的文档之中，质量计划是软件项目中全部项目计划的一部分。质量计划将用大量的篇幅阐明如何执行质量计划、谁于何时要用这些质量控制以及其它信息(如是否需要专用的工具)。

因此，象 ISO 9001 这样的国际标准往往会(但并不总是)促进质量管理体系的产生。这就为项目经理制定质量计划提供了方便。由软件开发者承担的每个项目都各有一个质量计划。

对这个过程的一部分可以举一个例子：如项目经理认为产品的可移植性是该软件项目主要的质量要素。经理将在质量手册中查阅阐述可移植性的部分并将它们结合到的项目的质量控制中。质量手册提供了多种确保可移植性的方法，例如下述四个标准：

- 为项目选择能确保程序设计人员不使用非标准特性的程序设计语言。
- 对计算机和**操作系统**(operating system)进行大范围的**可移植性测试**(portability testing)。
- 检查所有处理程序代码并检测其不可移植特性的专用工具软件的输出。
- 可获得较高可移植性的开发技术(如**信息隐蔽**(information hiding))。

质量计划将包括检查系统中存在的某些特殊质量要素的**质量控制**(quality control)。质量控制通常与表示质量要素的文档依据有关。下面是几个质量控制的例子：

- **验收测试**(acceptance test)，它检查某一功能是否已正确地执行。文档依据为测试记录。该质量控制是用来检查正确性的。
- **代码评审**(code review)，是工作议程的一部分，要求进行该审查以检查与可移植性方面有关的问题。签署后的代码评审记录将作为文档依据。该质量控制是检查可移植性的。
- 用工具来检查程序代码是否是按照程序设计标准编写的。用工具进行**测试**(testing)出现违背标准的那些结果也将作为文档依据。由于开发程序设计标准旨在减少阅读代码时出现的错误，因而提高了可维护性和正确性。
- 由高级程序员对**测试报告**(test report)进行检查和签字，测试报告应由测试过系统中一些模块的程序员撰写完成。

1.4 标准和步骤

为了写完本章并提供一个术语表，本节简述质量手册中提供的主要工具：标准、步骤和准则。

首先需要提醒的是，对于质量保证这样如此严谨的学科，有些术语还似是而非且不太严密。这里给出的定义已被相当多的软件开发人员所接受，但并不是说大多数软件开发人员都已接受。本书把**标准**(standard)定义为：说明如何将文档写在纸上或显示在计算机屏幕上的指导书。例如，**需求规格说明标准**(requirements specification standard)要规定出需求规格说明中应有哪些章节，每节是如何构成的。

本书把**步骤**(procedure)定义为：阐述某一特定软件工作如何完成的文本。例如，程序设计的步骤将阐明应采用哪些标准、程序或模块的源代码和目标代码、将存在何处、如何完成某类测试、当编程和测试过程完成时需要填写哪些文件。步骤和标准的要点是：一旦某一项目采用了标准和步骤，就必须遵循这些标准和步骤。

准则(guideline)是为一项工作所提建议的文本。它与标准和步骤不同，不是必须执行的。例如，一个公司可能有一个控制项目**进度会议**(progress meeting)举行的准则，它规定了通常有哪些人参加这些会议。尽管准则只是提出了一些建议，但是多数时候还是希望项目本身遵守这些准则。步骤和准则往往是写在一起的，步骤主要含强制性说明，但也提出一些建议，例如某一

个会议应有多少人员参加。这种写在一起的文件的要点是:要明确哪些是强制性的,哪些是建议性的。

本小节解释了一些术语。在结束本小节之前,须指出的是:公司有时将标准和步骤统称为标准,有时也称准则。

1.5 技术工作

本小节的主要目的是提供技术术语词汇。在本书的其余各章,将不断地谈到一些技术工作,简要地说明一下它们是怎样组成的。

需求分析(requirement analysis)是一个发现用户对系统的要求的过程。要求可能是功能性的,描述系统的用途;也可能是非功能性的,限制系统的某些方面(如系统的**响应时间**(response time))。

编写需求规格说明(requirement specification)是详细说明需求分析所发现的系统特性的过程。这些特性写在一个常被称作**系统规格说明**或**需求规格说明**的文档之中。该文档将作为以后所有开发工作的依据。

系统设计(system design)是一个详细说明系统的总体结构软件(本书称之为**模块**)的过程。这些大块程序是第三代语言(如 FORTRAN 和 COBOL)的**子程序**(subroutine)或用第四代语言(fouth-genaration language)编写的程序。系统设计必须实现系统的功能特性,同时也应考虑文件大小等非功能性特性。这些特性可以在需求规格说明中找到。

详细设计(detailed design)是一个明确系统中的每个模块的过程。详细设计通常是用一种有点象**程序设计语言**(program design language)或用流程图表示的程序设计语言实现的。许多软件开发者忽略了详细设计过程而直接转向编程。**编程**(programming)是采用系统设计或详细设计并将之转化为程序代码的过程。

有一些以检查系统实现为目的开发工作。第一个就是**验收测试**(acceptance testing)。这是系统的最终测试,由用户在系统最终安装的环境中完成。验收测试检查是否正确地实现了需求规格说明中的要求。该测试通常是在**系统测试**(system testing)之前进行,是以开发者对测试过程能成功有十分把握为前提的一系列测试。

模块测试(module testing),有时也叫**单元测试**(unit testing),是检查系统的每个模块的过程。通常选择系统设计中指明要模块做的测试数据。另一个重要的测试形式是**集成测试**(integration testing),遗憾的是,该测试往往被开发者省略。这是一个建立系统的过程,它通过不时地增加的模块进行测试看它们是否与系统中的其他模块正确地接口。

1.6 ISO 9000 系列标准

该标准很重要,正在日益成为用户判断软件开发者的竞争力的主要方式。它已经被 130 多个国家采用。ISO 9001 系列标准存在一个问题:它不是专门用于工业的标准,是用通用术语词汇表述的,可以由各种不同产品(如滚珠轴承、吹风机、摩托车、运动器械、电视机、还有软件)的开发者解释。已经出版了一些将该标准与软件产业联系起来的资料,但都不够详细。与软件产业相关的标准有:

- **ISO 9001《质量体系——设计、开发、生产、安装和服务的质量保证模式》**。这是一个阐述

用于包括设计在内的产品开发的质量体系标准。

· **ISO 9000-3《ISO 9001 在软件开发、供应和维护方面的应用指南》**。这是一本向软件开发人员解释 ISO 9001 的专用材料。

· **ISO 9004-2《质量管理和质量体系要素——第二部分》**。该文件为软件服务和设施(如用户支持)提供了指南。

ISO 9000 系列标准在每个国家都有其特定的标准号,例如在英国该标准被称为 BS5750(英国质量保证标准)。该标准的要求被分为 20 个标题:

管理职责	检验、测量和测试设备
质量体系	检验和测试状态
合同评审	不合格品的控制
设计控制	纠正措施
文档控制	搬运、储存、包装和交付
采购	质量记录
买方提供的产品	内部质量审核
产品标识和可追溯性	培训
过程控制	服务
检验和测试	统计技术

本书的每章都对应于上述的标题之一。

1.7 怎样取得 ISO 9001 认证

取得 **ISO 9001 认证**(accreditation)的过程在不同的国家将有所不同。本小节说明在英国将**英国标准协会**(British Standards Institute,BSI)作为认证机构,ISO 9001 认证是如何取得的。尽管本小节与英国的读者关系最为密切,但是当认证在许多其他国家是以类似的方式进行时,这一节对其他读者仍然是很有用的。

首先公司应明确公司需要通过认证。公司与 BSI 鉴定合同,BSI 给公司寄一份问卷调查表,询问开始认证过程所需要的一些基本情况。问卷调查表中的信息包括公司中雇员的数量、公司的性质以及公司中哪些部分未注册等问题。对于在质量保证方面不是很先进的公司,BSI 通常在这一阶段针对其会议、培训和从一些来源得到的咨询向公司提出建议。建议的一种形式是由 BSI 预审视察方案提供的。BSI 审核小组的注意力集中在公司的质量体系中,对公司付出的时间,考虑最大限度地使公司受益方面的那些部分。该小组有效地对现有质量体系进行非正式的审核,并拟写一份列举其不足之处的书面报告。

当公司认为已经做好了接受评审的准备时,就向 BSI 提出申请。BSI 指派一位雇员控制以后要进行的评审过程。然后,在英国标准局,由 BSI 检查公司的质量管理体系文件,并将检查结果写入一个报告中,该报告详细说明了为了符合 ISO 9001 质量体系所需要修改的地方。

下一阶段,BSI 指派一个评审小组去视察该公司。通常这次视察应在质量体系运行至少三个月之后,以获得足够的项目有效审查的证据。在评审视察的最后,评审组将做出下述三种建议的一种:

(1)不合格。