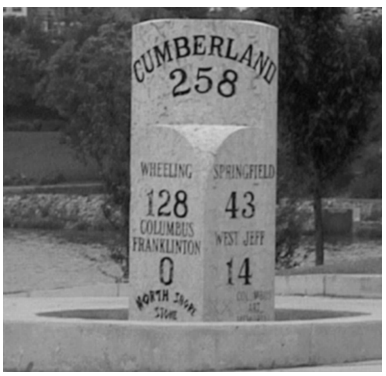
第3章

软件工程目标和要求



里程碑的选择只有一个原则，那就是，里程碑必须是具体的、特定的、可度量的事件，能够被清晰地定义。

——《人月神话》

软件工程学是为软件企业服务的，帮助企业更好地实现其业务目标。软件工程学着重帮助软件企业实现其工程目标，也就是帮助企业理顺软件开发和维护的流程，提供先进的、有效的方法，提高软件团队的能力，最终达到软件开发的目标。

业务是企业之本，企业的工作都是为了改善业务、提高竞争力和赢利水平，在这过程中，“成本、进度和质量”是影响企业业务目标的主要因素，企业的管理工作也必须围绕着“成本、进度和质量”来进行。软件工程也不例外，其目标就是提高软件开发生产率、降低企业的软件开发成本，准确预估和控制进度，不断提高软件产品和服务的质量。在开展软件工程活动的任何时候，始终不要忘记这些基本目标，这样软件工程会不断给软件组织带来效益，而不是仅仅停留在软件工程的理论上。

3.1 软件工程的基本目标

做任何事情，都要清楚其目标才能把事情做好。软件工程旨在提高软件开发的效率和软件产品的质量，这是最基本的两项目标。在任何时刻、做任何事情，“质量”和“生产力”都是工作的核心，也就是时时刻刻要问自己这样一个问题——如何以低成本、高效率开发出更优秀的软件。用通俗的语言来概括软件工程的基本目标，那就是“多、快、好、省”四个字。

- 多——更多地实现客户所需要的功能，产品的功能特性越强，越能满足用户更多的需求。

- 快——开发效率高、开发周期被缩短，项目在预期内完成或提前完成。
- 好——所开发出来的产品质量高，产品性能稳定、实用性强、可扩展性高，能切切实实满足客户的需求。
- 省——开发的成本低，用最小的代价开发出特定的功能。

在这 4 个目标中，“好”是最重要的，也就是说，质量是核心，以质量为中心，在“多、快和省”上面获得最佳平衡，可以用图 3-1 表示。软件工程的实施不是虚的，是实实在在的，其结果就是使软件企业能够开发出品质好的软件或提供优质的软件服务，而且开发速度快、成本低、维护容易，这也是企业所期望的。如果达到这些基本目标，就说明软件工程在企业应用获得成功。

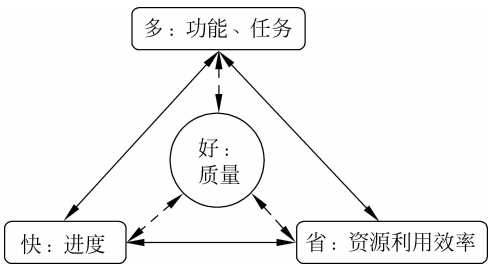


图 3-1　四项基本目标“多快好省”之间的关系

3.2　软件工程的影响要素

软件工程学科的设定和研究，就是利用有限的时间和人力资源达到上述基本目标。要达到这些目标，要充分分析软件工程的影响因素，这些因素直接关系到目标的实现。在建筑过程中，人们也是要考虑气候、交通、材料等各方面的因素影响，恶劣的气候、交通运输不畅和钢铁水泥价格上涨都会影响建筑工厂的进度和成本，甚至影响建筑工程的质量。

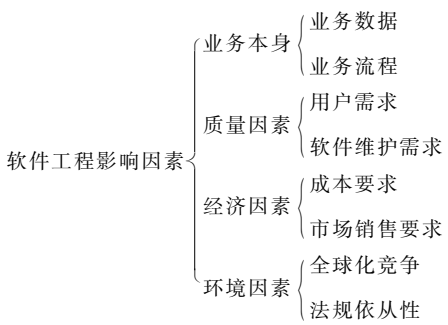


图 3-2　软件工程影响要素的分类

在软件工程中，虽处在一个相对比较独立而灵活的软环境，但其影响因素还是很多。例如，需求的不确定性主要是受业务因素的影响，例如业务数据是否容易归纳、业务流程是否清晰等，都会直接影响到需求定义的结果。从大的方面看，软件工程的影响因素可以归纳为待处理的业务自身因素、质量因素、经济因素和环境因素等 4 大类，如图 3-2 所示。这些内容将在后面逐步展开讨论。

3.3　软件工程的业务需求

软件工程的要求，首先来源于其面向的业务，或者说业务需求是第一的，软件工程要立足于业务需求，最大程度地满足业务需求。业务是根本，业务决定了客户群、客户需求、产品投放市场的时间、软件服务的模式等。

业务需求主要从软件产品所处理的具体业务来分析，包括业务要面对的市场或市场机遇、市场环境等因素的影响，可以从以下 7 个方面来确定软件的业务需求。

(1) 业务机遇。描述现存的市场机遇或正在解决的业务问题，包括业务的竞争程度和

业务所处的周围环境。首先，要对现存的其他软件产品有一个全面的评估，了解其优点和缺点，从而确定新产品的业务解决方案，以及新产品的特点、所具有的竞争优势。

(2) 业务目标。试图用一个定量(可度量)的合理方法来评估软件新产品可能带来的商业价值，重点在业务的价值上。软件能服务于业务的核心目标之上，例如电信行业也需要信息处理、人事和财务管理等系统，但真正为电信行业带来业务价值、为业务目标服务的软件还是信号处理、客户管理、通信计费等业务软件。

(3) 业务模式。业务处理的方式是在线的还是非在线的、分布式的还是集中式的、实时的还是非实时的？这些业务需求对后期软件系统的设计也是至关重要的。

(4) 业务市场，包括市场的定位和划分。对不同客户群的需求支持，包括一些典型客户(如大客户)的需求支持以及现有市场上的产品还没有满足的需求(潜在的需求)。阐述如何在新产品中避免出现客户目前所遇到的问题、如何更好地支持客户或市场新的需求。

(5) 业务增值。提供给客户的价值，确定产品所能带给客户的价值，并说明产品是如何满足客户的需要，更重要的是，如何帮助客户提升业务、帮助客户获得更大的成功。

(6) 周期性业务，会影响或决定产品投放时间。不同的业务对时间有不同的需求，特别是对一些周期性比较强的业务，投放时间会显示出更强的敏感性。例如，教学软件上市时间在寒暑假比较好；财务软件不宜在季度末、年末等关键时间被开始投入使用，应在财务处理相对比较宽松的时间内开始使用。

(7) 业务风险。识别出所开发产品有关的主要业务风险，例如市场竞争、时间问题、用户的接受能力、实现的问题或对业务可能带来的消极影响。评估风险产生的可能性和严重性，设法找出所能采取的风险防范措施。

3.4 软件工程的质量要求

所开发的软件产品都希望拥有更多的用户，其实质就是所开发出的软件产品或服务能更好地满足客户的不同的需求。满足客户的需求是软件开发的首要目标，是软件质量的诉求。软件质量与传统意义上的质量概念并无本质差别，软件质量和一般质量概念的共性是明显的，软件质量也在于软件固有特性满足客户要求的程度。提高软件产品质量就是提高客户的满意度，也就是提高产品满足客户需求的程度。

3.4.1 质量的含义

质量是一个大家都非常熟悉的词汇，在人们日常生活中可以说无处不在。似乎每个人在讨论质量的时候，都明白其含义，人们对质量的理解有时非常简单，即“好”与“坏”的区别，或“好”的程度。但实际上，“好”的程度是很含糊的，许多人并不能真正理解质量的含义。也就是说质量不是一个简单的概念，质量是一个相对客户而存在的、富有内涵的概念。而且给“质量”所下的定义必须是可控制的和可衡量的，这样才可以帮助软件企业控制质量、管理质量，从而最终提供高质量的软件产品或服务。

传统的质量概念基本是指产品性能是否符合技术规范，也就是将产品的质量特性与技术规范(包括性能指标、设计图纸、验收技术条件等)相比较。如果质量特性处于规范值的容

差范围内，即为合格产品或质量高的产品；超出容差范围，即为不合格产品或次品，这就是所谓的“门柱法”(goalpost)——符合性质量控制。门柱法一直是质量管理的基本方法，在制造业界被普遍采用，并发挥了重要作用。

在工业发展的初期，产品技术含量低、结构简单，符合性质量控制可以发挥其重要的质量把关的作用，但对于高科技和大型复杂的产品，符合性质量控制已不能满足质量管理的要求。于是世界著名的美国质量管理大师朱兰(Joseph M. Juran)提出了产品的质量就是适用性(fitness for use)的观点，所谓适用性就是产品在使用过程中满足客户要求的程度。

这正如国际标准 ISO 8492 给质量下的定义“质量是产品或服务所满足明示或暗示需求能力的特性和特征的集合”，这种特性和特征的集合正是客户所期望得到的，其中有些是明示的，其中有些是暗示的。

(1) 固有特性。特性可以是固有的或赋予的。固有特性是指某事物中本来就有的，尤其是那种永久的特性，例如，木材的硬度、桌子的高度、声音的频率和螺栓的直径等技术特性。赋予特性不是某事物中本来就有的，而是完成产品后因不同的要求而对产品所增加的特性，如产品的价格、硬件产品的供货时间、售后服务要求和运输方式等。同时，固有特性与赋予特性是相对的，某些产品的赋予特性可能是另一些产品的固有特性，例如：供货时间及运输方式对硬件产品而言，属于赋予特性；但对运输服务而言，就属于固有特性。

(2) 明示或暗示的特性、需求。明示的特性，可以理解为是规定的要求，一般在国家标准、行业规范、产品说明书或产品设计规格说明书中进行描述或客户明确提出的要求，如计算机的尺寸、重量、内存和接口等，用户可以查看。对于暗示或隐含的用户需求，一般没有文档说明，而是由社会习俗约定、行为惯例所要求的一种潜规则，所考虑的需求或期望是不言而喻的。一般情况下，客户或相关方的文件中不会对这类要求给出明确的规定，组织应根据自身产品的用途和特性进行识别，并做出规定。如台式计算机可以使用本地区的、约定的家用电压，在我国就是 220V，在北美和欧洲就是 110V。但对笔记本计算机，由于是移动式设备，可以随身携带，所以应该支持国际范围的宽幅电压 (110～240V)，这样笔记本计算机就可以在世界各地使用。如果某种笔记本计算机不支持这样的宽幅电压，这种笔记本计算机就存在质量上的缺陷。比如一张四条腿的餐桌，只要告诉一条腿的高度就可以了，暗示着另外三条腿必须是相同高度，而且桌面要光滑，不能刮破用户的手脚，这些都是产品需要满足的隐含要求。

(3) 必须履行的是指法律法规要求的或有强制性标准要求的。如食品卫生安全法、GB/T 18220“手持式个人信息处理设备通用规范”等，组织在产品的实现过程中必须执行这类标准。

(4) 要求可以是多方面的。当需要特指时，可以采用修饰词表示，如产品要求、质量管理要求、客户要求等。所以要求可以由不同的相关方提出，不同的相关方对同一产品的要求可能是不相同的。例如对汽车来说，客户要求美观、舒适、轻便、省油，但社会要求对环境不产生污染。组织在确定产品要求时，应兼顾客户及相关方的要求。

3.4.2　客户是质量的焦点

质量是相对客户存在的，没有客户，就没有质量，所以说客户是质量的焦点，质量是客户

的满意度。我们知道，不同的客户可能对同一产品的功能提出不同的需求；也可能对同一产品的同一功能提出不同的需求；需求不同，质量要求也就不同，只有满足需求的产品才会被认为是质量好的产品。这种相对性要求我们对质量的优劣要在同一等级基础上做比较，不能与等级混淆。等级是指对功能用途相同但质量要求不同的产品、过程或体系所做的分类或分级。

强调客户是质量的焦点，就是要求软件产品开发时始终服从于客户的需求，即一切从客户需求出发，从客户的角度思考问题，想客户所想。

- 以客户为关注焦点，以提高客户满意度为目的，确保客户的要求得到确定并予以满足。
- 软件所提供的所有功能是客户所需要的，绝不要开发那些客户不需要的功能。
- 经常拜访客户和客户调查，认真、仔细地听取客户的意见。例如，通过调查，可以发现客户最关心的不是功能，而是易用性。
- 提供满足客户和适用的法律法规要求的产品，内容健康，有益于生活和工作。
- 用户的数据比软件系统本身更重要，应当受到格外关注、保护，使用户数据具有非常高的安全性和兼容性。
- 理解并满足现有及潜在客户和最终使用者的当前和未来的需求和期望，以及理解和考虑其他相关方的当前和未来的需求和期望。
- 强调持续改进，客户能及时得到新的产品或得到更完美的软件服务。

3.4.3 软件质量的特性

虽然软件质量具有质量的一些基本属性或特性，但是软件质量的具体内涵是不同的，仅仅在功能上满足客户的要求是不够的，还要在易用性、性能、兼容性、安全性等很多方面来满足客户的需求，如表 3-1 所示。

表 3-1 用户要求与软件质量特性

用户要求	要求质量的定义	质 量 特 性
功能	• 能否在有一定错误的情况下也不停止运行	完整性(integrity)
	• 软件故障发生的频率如何	可靠性(reliability)
	• 故障期间的系统可以保存吗	生存性(survivability)
	• 使用方便吗	可用性(availability)
性能	• 需要多少资源	效率性(efficiency)
	• 是否符合需求规格	正确性(correctness)
	• 能否回避异常状况	安全性(safety)
	• 是否容易与其他系统连接	互操作性(inter-operability)
修改 变更	• 发现软件差错后是否容易修改	可维护性(maintainability)
	• 功能扩充是否简单	可扩充性(expandability)
	• 能否容易地变更使用中的软件	灵活性(flexibility)
	• 移植到其他系统中是否正确运行	可移植性(migratability)
	• 可否在其他系统里再利用	再利用性(reusability)
管理	• 检验性能是否简单	可测量性(testability)
	• 软件管理是否容易	可管理性(manageability)

(1) 功能性　软件所实现的功能符合预先定义的设计规范和满足用户需求的程度。

(2) 易用性　对于一个软件,用户学习、操作、准备输入和理解输出所付出努力的大小,如安装简单方便、容易使用、界面友好,并能适用于不同特点的用户,包括对残疾人能提供产品使用的有效途径或手段。

(3) 性能　用来衡量系统占用系统资源(CPU 时间、内存)和系统响应、表现的状态。如果系统用完了所有可用的资源,那么系统就会出现性能下降。系统响应时间或操作性能不仅受到系统资源的影响,也受到系统内部算法、外部负载等方面的影响。

(4) 容量　系统的接受力、容纳或吸收的能力、某项功能的最大限度。有时需要确定系统的特定需求情况下所能承受的最大负载。如 Web 系统能承受多少并发用户访问、一次加入会议系统的最多人数等。

(5) 安全性　系统和数据的安全程度,包括功能使用范围、数据存取权限等受保护和受控制的能力。数据和系统的分离、系统权限和数据权限分别设置等都可以提高系统的安全性,而且系统稳定性和可靠性对系统安全性有很大的影响。

(6) 可靠性　在规定的时间和条件下,软件所能维持其正常的功能操作、性能水平的程度。通过防故障能力、资源利用率、代码完整性以及技术兼容性等可以提高软件的可靠性。健壮性和有效性有时可看成是可靠性的一部分。有些软件系统要求特别高的可靠性,具有很强的容错能力,能保证系统长时间稳定运行,例如银行交易系统、铁路交通管制系统、导弹防御系统等。

(7) 可测量性　系统某些特性可以通过一些量化的数据指标来描述其当前状态或理想状态。

(8) 可维护性　在一个运行软件中,当环境改变或软件发生错误时,进行相应修改所付出的工作量或难易程度。可维护性取决于理解软件、更改软件和测试软件的难易程度,可维护性与灵活性密切相关。高可维护性对于那些经历周期性更改的产品或快速开发的产品很重要。

(9) 兼容性　软件从一个计算机系统或环境移植到另一个系统或环境的难易程度,或者是一个系统和外部条件共同工作的难易程度。兼容性表现在多个方面,如系统的软件和硬件的兼容性、软件的不同版本的系统、数据的兼容性。

(10) 可扩展性　指将来功能增加、系统扩充的难易程度或能力,如简单的模块结构、模块间低耦合性、多层分布体系架构等。

软件系统的可靠性和性能是相互影响的,高可靠性可能降低性能,比如数据的复制备份、重复计算等可以提高软件系统的可靠性,但在一定程度上降低了系统的性能。又如,一些协同工作的关键流程要求快速处理,达到高性能,而这些关键流程可能会减少一些检验、保护或容错的环节,降低了可靠性。

软件系统的安全性和可靠性,两者是一致的,安全性高的软件,其可靠性也要求相对高,因为任何一个失效,可能造成数据的不安全。安全性相关的关键组件,需要保证其可靠,即使出现错误或故障,也要保证代码、数据被储存在安全的地方,而不能被不适当地使用和分析。但软件的安全性和性能、适用性会有些冲突,如加密算法越复杂,其性能可能会越低。对数据的访问设置种种保护措施,包括用户登录、口令保护、身份验证、所有操作全程跟踪等等,在一定程度上降低了系统的性能和适用性。所以,不仅要考虑功能、性能和可靠性等的要求,而且在可靠性、安全性、性能、适用性等软件质量特性方面达到平衡也是非常重要的。

3.4.4 影响软件产品质量的因素

软件质量因素是影响软件质量特性的参数，如果用不同的视点去看，结果可能不一样。软件质量的内容由软件产品质量、软件开发过程质量和软件商业环境的质量所构成，所以在研究软件质量因素时，也可以根据这样的分类去分析，即：

- 从软件产品看，什么因素对产品运行、产品修改和产品移植有较大的影响？
- 从软件开发过程看，什么因素对计划过程、设计过程、实施过程和维护过程中的质量有较大的影响？
- 从软件商业环境看，什么因素对客户（体验、支持、培训）、市场（规划、机会）和销售（成本、效率、服务）等的质量有较大的影响？

1. 产品运行的质量因素

软件产品的质量因素是对软件产品的功能、可用性、可维护性和可移植性等的影响因素，可以按产品运行、产品修改和产品转移来划分，如产品运行的质量影响因素有以下几点：

- 处理流程：功能的每一步操作都实现了吗？操作合乎逻辑吗？
- 算法：选用正确的或优化过的数值算法、计算的精度满足要求吗？
- 界面可视化：图形界面是否形象生动、清晰、容易理解？
- 系统登录：系统的登录设置了不同权限的用户组和用户？密码有严格要求吗？
- 内存分配和释放：运行时占用最少内存资源吗？不用的内存都释放了吗？
- 异常或错误：系统能否判断出错并重新初始化或弹出提示框？

2. 产品修改的质量因素

软件产品，不仅要满足其运行时的用户需求，还要满足用户需求的变化，要对软件产品进行修改，修正不正确的功能、增强已有的功能、增加新的功能。在软件产品修改时，影响其质量的因素有以下几点。

- 程序可读性：程序命名符合规范吗？注释是否充分？
- 可理解性：程序中每个对象、组件是否设计合理而容易被理解？
- 文档性：所有的文档都具备吗？
- 模块的耦合性：每个模块自己是否比较独立、模块之间关系简单又少？
- 自定义性：功能的许多设置可以通过外部数据库、XML 等实现，程序中没有“死代码（Hard code）”吗？
- 可预见性：预先知道每个功能达到的预期结果，从而可以通过测试去验证吗？

3. 产品移植的质量因素

- 操作系统的独立性：产品不需要修改或做很小的修改，在不同系统上运行？
- 硬件的独立性：产品通过有效接口去实现和硬件的关系？
- 数据的独立性：数据和软件产品的有效分离从而使系统具有灵活的数据导入/导出功能？

• 系统的剪裁性：是否可以根据需要抽取系统的若干个部分组成一个新的系统？

概括起来，软件产品运行、修改和移植的质量影响因素，如图 3-3 所示。

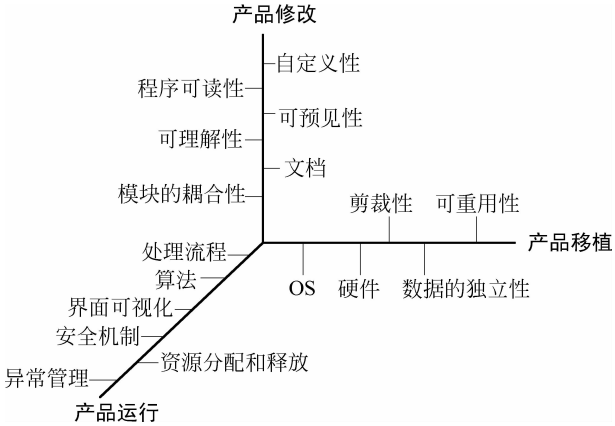


图 3-3　软件产品质量因素的三维特性

4. McCall 模型中软件质量影响因素

- 阐述性；
- 数据公开性；
- 连贯性；
- 容错性；
- 执行效率/储存效率；
- 存取控制/存取检查；
- 可训练；
- 沟通良好；
- 简单性；
- 易操作的工具；
- 自我操作性；
- 扩展性；
- 一般性；
- 模块性；
- 软件系统独立性；
- 机器独立性；
- 通信公开性；
- 正确性；
- 可操作性。

5. 质量影响因素的优先级

软件质量因素对软件质量影响的程度、深度是不一样的。例如，“正确性与精确性”应该排在这些质量因素的第一位，因为软件首先要能正常运行并输出正确的结果，才能为用户服

务，向用户提供正确的、有价值的服务；否则，软件就没有价值，甚至给用户造成损失，更谈不上性能、可靠性、兼容性等。

其次，对软件产品质量影响的因素是易用性、性能。在保证软件的正确性与精确性之后，就要保证用户能方便、快捷地使用产品。不仅产品要易用，而且要好用，系统响应时间短、操作速度快。产品易用、好用会赢得越来越多的用户青睐和使用，从而得到大量的用户反馈，及时知道用户的真正需求，不断改进软件产品。

对软件产品质量影响的因素是容错性和可靠性的设计。从某种意义上说，完全消灭软件中的缺陷几乎不可能，所以我们不得不假定系统存在着一定的不正确与不精确的因素，然后设计相应的措施，来保证在这些因素的影响下，系统依旧要保持可靠和安全，即容错性设计。

3.4.5 软件过程的质量因素

相对软件产品来说，影响软件产品过程的质量因素更多，也比较复杂，这里从软件开发生命周期的各个阶段——计划、设计、实施和维护等分别来进行介绍。

1. 计划过程的质量因素

在软件项目计划过程中，要完成需求分析、需求定义、产品规格说明书、项目计划书等工作。这些工作相对复杂，其影响因素相对较多，主要有：

- 和客户的沟通能力，与客户沟通是否充分决定了需求分析的结果。
- 软件产品特性定义的方法，分析的结果还需要通过有效、清晰的方式来描述。
- 项目计划策略会影响质量计划、测试计划等实施水平。
- 评审的流程、范围、方式和程度，包括对软件需求分析书、计划书的讨论和审查的力度。
- 协同工作流程。
- 合同和用户管理流程及方法。
- 文档编写、管理等的规范和流程。

2. 设计过程的质量因素

在软件项目设计过程中，需要完成系统的设计、程序设计、测试用例的设计等任务。设计可以说是一个纯技术工作，相对封闭，与客户的关系较小，其结果取决于软件产品质量指标的定义。例如，在系统构架、模块及其接口设计、可靠性设计、兼容性设计、界面设计等方面的要求是否得到一致的理解、正确的实现等，直接关系到这个过程的质量。除了计划阶段的共性因素（如协同工作流程、文档编写规范等）之外，设计阶段的质量影响因素有：

- 软件产品指标的定义和解释。
- 设计流程，包括知识交换、结果评审等流程。
- 设计标准改进流程。

3. 实施过程的质量因素

在软件项目实施过程中，需要完成系统的编程、测试脚本开发、单元测试、集成和系统测

试的执行等任务。如果计划、设计过程执行得很好，实施相对来说比较容易。如果计划被严格执行，并与良好的变更控制(change control)流程结合起来，实施的结果会更理想。这个过程特有的质量影响因素有以下几点：

- 变更控制流程及其系统。
- 执行过程跟踪方法、流程及其系统。
- 缺陷处理流程及其系统。

4. 维护过程的质量因素

软件维护是软件过程中很重要的一个过程，通过不断改进、增强来实现软件的成熟、稳定和可靠。这个过程的质量影响因素有以下几点：

- 变更控制流程及其系统。
- 用户反馈、相应处理机制。
- 回归测试流程。

软件商业环境也可以看作软件计划和维护过程的一个扩展，也可以看作软件产品运行方面的扩展。软件产品的一些特性会直接在商业环境中体现出来，如产品界面循序渐进的变化容易被客户接受，而跳跃式、突破式的用户界面变化不被接受。产品的定义还要考虑对市场、销售的支持。软件质量在这方面的质量影响因素有以下几点：

- 软件改进的策略，如是否遵守持续不断、逐步改进的原则。
- 产品开发模式，如是否很好地采用增量模式，保证产品发布的最佳时机。
- 市场定位，如不同的软件产品在用户群定义上是否有区别、有没有冲突。
- 产品标准，如是否符合国际标准和业界标准、是否引导业界进步。
- 文档内容和形式，如是否通过 Flash、视频动画教学内容帮助客户的自我培训、是否通过在线帮助有助于客户支持。
- 软件的后续服务模式。

3.4.6　软件质量的指标

软件质量属性可以分为两部分，大部分特性在软件系统运行时可以识别，少部分特性在软件系统运行时不可识别。软件质量指标是衡量那些可识别的软件质量特性项，有助于软件质量进行度量，选择软件工程方法来达到特定的质量目标。在一个理想的范围中，每一个系统总是最大限度地展示所有这些属性的可能价值。系统将随时可用，决不会崩溃，可立即提供结果，并且易于使用。根据这些软件质量指标来定义用户和开发者的目标，从而软件的设计者可以作出合适的选择。

对于软件质量指标，可以按开发阶段和维护阶段来区分，对开发者具有重要意义的软件质量指标是使产品易于修改、扩充、测试和验证，并易于移植到新的平台上，从而可以间接地满足客户的需要。而对于维护阶段，具有重要意义的软件质量指标使软件系统易安装、强壮、性能好、稳定，更容易维护。

1. ANSI/IEEE 定义的质量指标

- 正确性(correctness)：实现的功能达到设计规范，并满足用户需求的程度。

- 可靠性(reliability)：规定的时间和条件下，仍能维持其性能水准的程度。
- 易用性(usability)：用户掌握软件操作所要付出的时间及努力程度。
- 效率(efficiency)：软件执行某项功能所需计算机资源(含时间)的有效程度。
- 可维护性(maintainability)：当环境改变或软件发生错误时，执行修改或恢复所做努力的程度。
- 可移植性(portability)：从一个系统/环境移到另一系统/环境的容易程度。

2. 功能性的质量指标

- 功能的正确性(correction)：系统功能和用户的实际需求、已定义的产品规范一致，没有出错，能正常运行。
- 功能的准确性(accuracy)：系统所产生的结果在精度允许的误差范围之内(如±0.001)。为了达到这个精度，函数变量、数值算法、数值转化等都要保留足够的数位，如长整型、双精度的浮点运算等。
- 软件功能的完整性(completeness)：所有功能及其定义应该清楚、可用，满足所需功能的每一个输入/输出数据项、功能、接口、文档等都已具备。

3. 可用性的质量指标

- 可操作性(operability)：容易使用和操作系统，包括易理解的用户界面、大多数场合使用很少的击键、适应一些特殊用户的可选项等。
- 通用性(commonality)：数据显示、网络通信接口和用户界面等都遵守已有的软件标准，包括采用单一模块来进行数据格式的处理、通信接口等。这一质量指标也和可维护性、扩展性相关。
- 一致性(consistency)：在软件开发整个生命周期中，建立和使用相同的标准，保证设计、实施的流程、方法等一致，保证全程变量、数据类型、出错处理等命名和使用的一致性。

4. 可靠性的质量指标

- 系统自我恢复能力(autonomy)：当系统的某个功能失效发生时，在当前环境下系统具有实现故障自动转移、重新自动配置、继续执行的能力。软件系统具有自我检测、容错、备份等机制，也尽量做到独立于硬件的编码、通信协议等。
- 健壮性(robust)：各种恶劣环境(如大数据量、大量用户同时访问等)下系统依旧能正常工作。
- 系统的分布性(distributivity)：软件系统的某些子系统或部分被定位于不同的处理主机、存储设备上，提高系统的可靠性和性能、容量、可维护性和可扩展性。分布性包括网络结构分布性、系统接口分布性、系统功能分布性和数据分布性、用户操作分布性等。在不同设备上运行相同的关键功能或对数据进行储存和复制，是解决单点失效问题的有效方法。

5. 性能的质量指标

- 有效性(efficiency)：系统在通信、处理、存储等方面占有很少资源或对所使用的资