

提高软件质量的有效工具 QFD

专家们认为,提高软件质量是核心问题,因为软件质量越高需要的维护越少,从而为新的设计解放人力资源。这就是说,经费要重新分配,把资金投放在开发过程的前端,而在测试阶段可大大节省。一个程序的每次变化一般都使其更加复杂。因此,程序变化越少,所保持的结构越好。

质量功能调配(QFD)是用于提高过程质量的一种工具,它既适用于制造业和服务业,也能用于提高软件质量。

质量的概念

现在还没有一个被普遍接受的软件质量定义。国际标准化组织(ISO)为建立普遍接受的质量定义作了努力,ISO/IEC DEC9126 标准给的定义是:“一种软件产品的全部指标和特性都是为满足确定的或蕴含的功能需要。”这个定义作为我们以下讲座的出发点。定义中包含软件质量的几种观点:用户的观点,开发者的观点和管理人员的观点。这就是说,按照 ISO 标准,一种软件产品可以按照几种不同的观点,有几种质量测定。

用户以质量评价产品是不言而喻的。可是全面质量管理(TQM)的基本原则是,质量必须在开发过程中建立。如果这个过程不加控制,产品质量是“随意的”,问题将集中到事后测试上。如果开发过程加以控制,就可能保证最终的产品质量。完全相同的原则也适用于信息系统的研制。

就质量来说,从用户需求和各种规范到进度量的转换肯定不是件易事。本文将用 QFD 完成这个转换过程和质量因素的进一步分析。

质量功能调配(QFD)

QFD 是 1972 年在日本提出,1983 年进入美国的。这项技术在美国企业中获得极大普及。它的目的是允许企业的所有职工参加新产品的的设计。这是与日本全公司质量控制(CWQC)一致的。

在日本,QFD 首先用于造船业,而当前规模最大的用户是汽车工业。日本制造商也把 QFD 用于电子技术,家用设备、服务行业、集成电路和农业机械领域。在美国,QFD 的最大用户是汽车工业,现在计算机产业对这项技术也表现出浓厚的兴趣。

QFD 由不同图表和代表不同控制过程的矩阵表组

成。基本的概念是把用户需求转换成适合产品开发的每个阶段的技术要求。矩阵的分级结构有助于使逐级过程从用户相当含糊和抽象的要求变成详细和准确的信息。

用于信息系统开发的 QFD

专家们认为,QFD 对信息系统开发也是有益的技术。它可用于调配包括软件开发在内的全公司的所有过程。

信息系统开发是个复杂的过程。出发点是把用户需求转换成规范。系统设计一般遵循自顶向下或分级方法和逐步的过程,从相当抽象的要求到详细的专用信息。这个过程涉及不同部门的人员和用户。他们都有不同的利益、背景知识和交流与合作的能力。在质量要求上也可能有不同的利益冲突。这些冲突必须得到解决,实现均衡。

将 QFD 嵌入软件生产的全过程质量控制是有益的。它不增加生产成本,只是在前期投资增加,而在测试,实现和维护阶段投资节省。

例证:METPATH 公司

我们选择了把 QFD 用于实际的 Metpath 公司。Metpath 是个临床实验室,处理医生代理病人提供的样品。医生通过邮件或信使把样品送到实验室,期望得到准确而及时的分析。临床实验室是竞争激烈的行业,在竞争对手之间与服务业几乎没有什么差别。

在用户中建立信誉,Metpath 加强了用户服务,在医生办公室安装计算机终端,并把这些终端同实验室的计算机连接起来,医生在办公室就能得到刚刚完成的实验结果。今后,METPATH 将扩大业务范围,提供其它服务,如医疗诊断(使用专家系统),病人分析资料和药物作用研究。

在 METPATH 的 QFD

QFD 过程是通过由 Metpath 职员和用户机构代表参加的一系列会议实行的。

第 1 次会议: 确定用户需求, 把这些需求列入优先地位。这次会议的目的是确定新信息系统的主要要求。

这次会议讨论了 3 个主要信息系统应用: 实验结果, 病人分析, 药物作用研究。实验结果系统的功能是记录、跟踪和报告在 Metpath 实验室进行的临床实验结果。病人分析系统是维护描述病人和 Metpath 进行的临床实验结果的数据库。药物作用系统的目的是模拟医生提出的药物治疗、病人分析、实验结果和其它药剂的效果。

这些系统分别是事务处理系统、信息检索系统和决策支持系统。

这些系统的用户的主要要求见图 1 的左边部分。对实验结果系统的要求是: 便于使用、准确、安全、适应能力和响应时间。需要的实验结果指的是此系统必须跟踪和报告的实验种类和应该提供的信息的格式。其它项目(如准确)是医药中心用户对此系统的要求。

对病号剖析系统的要求也见图 1(为了简便起见, 只有一小部分表现在这个矩阵中)。

除了规定系统的要求之外, 医药中心用户提出对每项要求按其重要程度定为 1-5 级, 5 为最重要。



图 1 软件指数与用户需求

第 2 次会议: 确定软件指标, 把用户的要求变换成软件指标。结果归纳在图 1 的矩阵上。

文献上有多种标识质量特性的质量模型, 如 Boehm 模型和 McCall 模型等。Metpath 小组选择了 ISO 模型作为标准 (ISO / IEC DIS9126)。这个标准规定了描述任何软件产品质量的 5 个指标即: 可用性、效率、可维护

性、可靠性和功能性。

ISO 认为需要把这些指标分解成更具体的子指标。如果需要, 这些子指标还能分解成详细的第 3 级。

完成第 1 个质量矩阵的最后步骤是变换用户需要与软件指标和子指标之间的关系。这些关系在图 1 中用 3

种关系符号来表示: 坚固关系($\times$), 某些关系($\circ$), 否定关系($\triangle$)。

在图 1 中, 实验结果的准确性和可靠性的 3 个子指标(成熟、容错、可复性)有坚固关系。在软件设计和实现过程中必须认真注意这 3 个子指标。以便保证实验结果子系统的准确性。

用户的某些要求很可能显示出与软件指数的否定关系。例如图 1 中的安全性与可操作性和时间行为(响应和处理时间以及解题速度)就是否定关系。当否定关系出现在矩阵中时, 它表示需要综合分析才能在关系因素中达到可接受的平衡。

第 3 次会议: 标识产品特性, 并把软件指数变换成产品特性。此次会议的目的是进一步推敲上次会议制定的软件定义。

在第 3 次会议上, Metpath 小组选择了一组普遍熟悉的产品特性, 并把图 1 上的软件指数变换成这些产品特性。结果见图 2。

图 2 上表示的主要产品特性有: 系统复杂性、系统结构、系统模块化、源代码和文件的可靠性、源代码和文件的规模。其它产品特性是系统的可靠性、培训时间、响应时间和解题能力、图象用户接口、登记或注销简便。这些产品特性(用 * 号表示的)能直接测量(它能作为软件度量标准)。

产品特性

× 坚固关系

○ 某些关系

△ 否定关系

软件指数		产品特性												
		系 统			可 靠 性		规 模		其 它					
		复杂性	结 构	模块化	源代码	文件	源代码	文件	可用性**	培训时间**	响应时间**	解题能力**	图象用户接口**	登记或注销简便**
可用性	可理解性	△	×		○	×		△					×	
	可学习性	△	○	○	○	×	△			△			×	○
	可操作性	△	×	×		×		○	×		△	○	×	×
效 率	时间行为	○	○	○			×				×	×	○	
	资源行为						×				△	×	○	
可维护性	模块化	△	×	×	×									
	可变性	△	×	×	×	○	△							
	可实验性	△	×	×			△							
可靠性	容 错								×					
	...													
...	...													
...	...													

图 2 产品特性与软件指数

第 4 次会议: 选择产品度量标准, 把产品特性变换成产品度量标准。

为了实现软件质量程序, 需要给软件的关键特性标识度量标准(定量测量)。为此, Metpath 小组中包括软件开发管理人员, 质量保证管理人员, 高级系统分析员和

程序员。这次会议的结果见图 3 上的矩阵。同每个产品特性有关的产品度量标准用简单记号表示, 而不用符号表示。图 3 上的一些产品特性有 1 个以上的度量标准; 例如, 复杂性的度量标准是 McCabe 的 $V(G)$ 和

HALSTEAD 的 E。另一方面,1 个度量标准(如代码和文件的可读性)。Gunning 的模糊索引)可用于 1 个以上的产品特性(源

		产品度量标准											
		技 术											
产品特性		Yau和Collofello设计结构	Henry和Kafaro信息流	Gunning Fog索引	LOC	Halstead容量和长度	Halstead文件规模	McCabe V(G)	Halstead E	Keutgen U	Littlewood可用性与测量	...	...
系 统	复杂性							X	X				
	结 构	X	X										
	模块化		X							X			
可 读 性	源代码			X									
	文 件			X			X						
规 模	源代码				X	X							
	文 件						X						
其 它	可用性									X			
	...												

图 3 产品度量与产品特性

在这次会议上, Metpath 小组检查了为 QFD 的一致性和准确性过程制定的图 1 至 3 上的所有度量标准。在制定这些度量标准中, Metpath 小组的出发点是用户需求最好理解, 并通过几个详细的层次把它们变成产品度量标准。这些度量标准均可用作整个开发过程的指南, 也可用作对产品主要特性的质量检查工具。

第 5 次会议: 最后这次会议的目的是同用户一起检查 QFD 过程的关键部分以及项目日程和执行细节。

结 论

QFD 是推动在用户需求与系统设计人员之间尽早结合在一起的系统。它证明在软件开发中同样也是有效的。应用 QFD 的主要好处如下:

1. QFD 使我们集中注意用户需求, 并把这些需求放在优先的位置。用户往往并不了解开发人员视为系统质量的技术因素, 如模块化和内聚性。另一方面, 开发人员

往往并不注意用户认为是重要的诸如使用简便、公认的准确性和快速响应等问题。

2. QFD 促使我们按照预防故障的要求考虑问题。在设计阶段预防故障(如不适当的规定需求)的起因比在后期阶段校正故障所花的成本少得多。

3. QFD 是模拟用户、软件设计人员和其他参加者之间提出讨论的问题的有力工具。

4. QFD 使我们便于把用户的主要需求描划到软件指标, 产品特性和产品度量标准上。

5. QFD 为接受这个过程的后果提供了机会。

总之, 这种技术为在开发过程提高软件质量提供了使人乐观的前景。今后, 专家们准备在两个方面扩大这项研究。第 1, 进行实践研究, 评价 QFD 用于支持软件开发的效果。第 2, 除了软件产品之外, 还要在 QFD 用于软件开发过程上增加投资。(柯 新 编译)