应用软件开发中的若干重要问题

孟章荣　（航天局北京计算机应用和仿真技术研究所）

摘要：本文说明过去在软件产品开发中被忽视的几个方面。有些是几乎被一般的软件开发人员完全忽视；有些虽然没被忽视，但没有被足够重视，切实贯以实施。根据我们的实践体会必须强调这些开发要领，使得开发的应用软件产品真正成为用户的得力助手。

一、引言

随着计算机技术的迅速发展，各种各样的计算机应用对软件的需求越来越多，对软件的开发要求也越来越高。过去软件开发、使用、维护主要是软件人员的工作，但随着应用的深入发展不仅需要软件人员开发，也需要领域专家的参与。至于各领域的计算机使用和维护就主要由各领域使用人员承担。由于软件难于予测和掌握的特性使软件开发及日后维护均成为使用者的沉重包袱，为此除了对软件要有通常的功能和性能以外，对介面质量、全面品质、安全性、可靠性应有更加突出的要求，要求软件系统友好，可用性高的呼声特别高涨。软件人员开发软件时必须遵循一定的原则，最后经过测试、不断修正、维护，市场检验后的软件才能成为相应各领域所需的软件产品。目前有些人认为软件人员开发出来的软件就是产品的看法是非常错误的。实际上，没有人购买及使用或很难使用、很少有人购买的软件就不能算是软件产品，开发单位最后交给用户的软件系统应是对用户有用的介决具体问题的 Solution，而不应是一堆 Problems。

二、软件产品常被忽略的几个方面

软件是由人装到计算机上进行运行，在运行过程中同时还有大量交互的活动。因此首先要考虑容易使用、维护、另外如直观、生动，提高人的工作效率等，也就是软件产品的设计必须充分考虑人因素（Human factors）问题，我们在下面列出某些过去通常被忽略的方面，严重影响了软件产品具有上述那些重要的品质。

1.软件系统的出错讯息

人在计算机使用过程和编程中一样总是会出错的、关键是软件系统如何对待人的错误。当用户使用和程序中有错误时用户就会得知他所使用的系统是好帮手，还是难于驾驭的魔鬼。一个友好的系统将会给出有丰富信息量的反馈，为用户修正错误提供充分支持，可是曾有人作过一个简单实验发现，现有的软件产品远离这个原则。就是软件人员常用的较为成熟的编译程序这类软件，对 15 个编译程序做的实验表明，最好的也只是能获得部分有用出错讯息，而最差的则出现一些令人可笑的东西。像编译程序这类软件通常的软件人员熟悉其基本用法和概念；可各领域的使用人员不熟悉。目前许多应用系统在提供给相应领域人员使用时也需要编译程序，不像早期提供给应用人员的一般就是目标码程序。因此应用人员在使用编译程序时由于不熟悉基本概念和用法就容易出错。，但许多编译程序出错讯息很差，用得不多，其它类型的软件系统可能会更糟糕。实际上有许多介决方法已知，例如，这些方法是：

(1)予想各种可以出现的错误，从用户角度给出其可能的解决办法。现在有不少软件人员在编制程序时只想到操作人员应该做的动作，而不考虑万一操作人员做错动作计算机如何处理。所以常有些软件操作一出错误就使计算机无所适从，只能进入死机状态！

(2)要考虑出错讯息的级联关系。刚编制的程序上机时常会出现许多出错讯息，有些很可能是在别的错误恢复时由某些徒劳的企图所引起的。

(3)出错讯息应使用单一模式。实验表明，从词泯错误到运行错误几乎极少编译程序呈现错误类型有统一的方式。代之，常按照错误探测时间人为地对错误分类，这

是一种错误的方法。因为像编译程序等的内部工作对用户来说是不关心的。

近年来集成化的程序设计环境和带窗口的高分辨率显示的发展和应用为软件系统出错讯息提供了许多新的介决方法。大部分这种环境消除了编辑、编译和运行之间的传统的严格分离情况，因而对错误就有一个统一的解法。高分辨率显示器显示有错误的几行源程序区域，用箭头指明一行内的集团，绘出行号。近代系统充分发挥显示的能力，用颜色来标识错误的符号；让用户看到前后几行源码，允许增加上下文视窗的大小；提供不同的视觉强度表明错误出现的地方（如用闪烁等）；与出错讯息一起显示用户手册有关部分，来帮助用户修正。

2.软件系统的人机接口

近年来随着微机和工作站图形功能的大大加强，计算机接口的人因素（Human factors）问题成为当前研究的一个热点。人们发问计算机接口设计为什么不能更科学些？设计接口的人员为什么不能像工程师那样？总之，我们认为接口设计要基于下列一些准则。

（1）生动、直观、发挥操作人员想像力、创造力，提高工作效率。随着软件直观化技术的重大进展，软件系统的人机接口采用高分辨率显示、图符（icon）、多窗口等功能是当今软件开发技术中的基本特征之一。采用领域专家习惯使用的示意性图表可大大缩小用户和机器之间的隔阂，使人们觉察到一个真实的物理世界。图形信息自然，容易产生形象思维，信息量高，信息处理速度大大快于语言和文字信息。同时不同国家之间还存在的因语言、文字而造成的信息交流的障碍；而图形信息则完全排除这些不利之处。

（2）接口设计要基于人错误分析的基础上。在使用计算机时人总有打算、意图（intention），一般意图本身中的错误称为出错（mistake），在执行意图中出现的一个错误称为疏忽（slip）。在对人错误分析中常发现可分成多种类型，这里的人错误分析主要对疏忽的分析，它也有多种类型。例如，意图说明的疏忽，即对当前状态搞混而出现的疏忽；描述时的疏忽，即含混或不完全的意图说明；由方案故障激活而导致的疏忽，即不是故意的激活等。这一系列疏忽建议应要求有更好的系统结构；屏幕显示和菜单系统应按功能组织；设计的命令语言和菜单标题应明显彼此不同；尤其应使那些会造成严重后果（或不能逆转动作的）操作不会很轻易执行。

（3）接口的一致性。接口缺乏一致性就容易引起错误。一方面要求显示反馈讯息的一改性，如不一致就要求用户搞清每一步时所处状态，这很难办到，容易搞错；另一方面交互操作的一致性，这样也就增加了用户在交互规则上的可予见性。一个更好设计的系统产生的一致性就能使用户更容易学习、记忆和使用。过份复杂和不实用的花俏是应该避免的。

（4）隐喻提示物的使用。在接口设计中应使用一些形象、直观的提示物。例如放大、缩小的比例变化可用滑轴的上下移动；删除某个文件和图符等可以拖向垃圾箱提示物。这样使人看了一目了然，很是直观、形象。

3.软件系统的 help 功能

help 功能是辅助计算机使用的功能，在较好的软件系统中都有一些，但还很不够。尤其是在开发应用软件系统，面对用户是对计算机使用不是很熟悉的新手时更为重要。通常应包括：

（1）范例（demo）演示。通过一个较为简单的例子说明整个使用过程。

（2）用户手册的说明。辅助学习和说明用户操作过程及命令使用。

（3）出错时的 help。与出错讯息一起尽量给出几种修正方法，用户只需选择或默认即可。不像过去传统的那样把整个出错讯息放在一起输出，让人看了无所适从，心中慌然。

（4）异常性的 help。外来异常的输入数据或事件出现，使原先的软件系统出现不适应时，或者显示出某些辅助方案；或者能通过人机交互搞清问题，找到解决办法。

（5）机器死机时的 help。提供重新起动的手段，通过交互搞清死机原因，选择恢复的方法。

4.软件系统程序的可读性和自动维护

在软件工程化要求中为了容易阅读、理解，源程序常规定要求注解行占 30%左右。这要求主要是为了容易维护的需要，因为通常软件维护成本占总成本的 60%左右。由于软件人员不了解领域应用人员的习惯术语、词汇工作方式，对具体应用领域不熟悉，在开发时常会作出某些不适当的假定，刚开发出来的软件必须在实践中由有经验领域应用人员不断评估、改进和维护，才能适应具体应用的需要。实际上上面说的情况和做法在软件产品

的发展历史上还只是属于软件开发时代，研制的成果是软件，包括为维护准备的内部说明书、外部说明书、维护手册等。而软件产品发展到现在已进入到真正的软件工程化时代，它的研制成果是软件产品，包括机器自动维护软件、高可靠性、联机对话手册等。我们目前开发的软件产品，机内自动维护这方面通常是被忽略了，即使有的话，功能也是很弱。应用人员常要承担起这种通常是很困难的重任。因此我们觉得必须减轻他们的负担，在开发软件产品时必须尽可能考虑机内自动维护的功能。

三、在软件产品开发时应强调的几个方面

上面我们谈了软件产品开发时通常被忽略的几个方面，当然只着重谈了某些方面，不可能概括全部。下面我们谈的几个方面，虽然在目前一般都注意到了，但我们在这里必须强调下列几个方面：

1.加强软件产品开发的工程化

这里我们不谈软件开发的各种模型和规范等问题，主要强调由 Edward Yourdon 及 Tom Demarco 等相继提出的软件结构化分析、设计方法，以便使软件开发过程和品质容易掌握，希望运用工程化方法解决软件危机。由于缺少计算机辅助，软件工程的推动一直未形成风潮。直至 80 年代初计算机辅助软件工程化(CASE)在欧洲兴起，到了 80 年代中期在欧美就很流行。因 CASE 本身所具有的特性，使软件开发自动化，软件品质和共生产力的提高，引起了欧美的软件革命。可是相对于其它软件技术，CASE 仍属一个年轻的技术，直至 90 年代人们估计在全球的使用率仍仅有 2%，国内的使用者更是不多，大多对它还很陌生。我们觉得各级部门应从战略上看到 CASE 的意义，不能满足目前国内开发软件的以往常规方式，应该居安思危。面对殷切需求更多更好的软件这个客观现实，我们必须加强培训和使用提供自动化所需的 CASE 环境。CASE 的总目标是以较少的代价得到较佳品质的软件。CASE 工具经过 10 多年发展已日趋成熟，且多样化，复盖软件开发全过程，供用户选用。一般而言 CASE 工具的背后均有软件方法学的支持，用户选用时要注意其所支持的方法是否符合自己的要求。类似的还有集成设计支撑环境(IPSE)。两种说法都在进化，逐渐界线变得不清，但仍保留着某些基本差别。

2.软件产品的测试和验收

在开发软件的各个阶段中，软件的综合测试和验收是非常重要的，但产品的质量不是靠最后测试得来的，而靠开发阶段的步步把关得到的。在开展软件的各个阶段都应加强检查、核实。过去传统的较注意编码的调试阶段；对软件开发前面阶段的软件需求、规格说明、概念设计阶段注意不够，一方面这些阶段的成果难于检查；另一方面缺乏相应的工具。随着 CASE 的发展及采用模块化结构设计方法，出现了一些有效工具，如 Teamwork 等有关工具可对前几个阶段的成果作出核实、检验。对软件系统的测试、验收要制订工作计划，包括：

(1)明确目的是证明系统符合需求规格说明，并符合工作要求。

(2)工作项目：点清文件，对软件系统中每一段的功能订定测试案例，最后系统综合测试。还要考虑验收测试时所需的基本资料、合同、需求和设计规格说明；系统建置、操作、测试及维护手册及所需设备等。

(3)进行方式要成立专案组，研订执行计划。重要的是要明确用户需求、验收需求，才能使验收成功进行。

通常验收的项目为功能、性能、介面质量、全面品质、安全性、可靠性。对可靠性、正确性、文件完备及满意程度等特点订立适度的量化指标。验收制度是软件产品开发中非常重要的一环。大型复杂软件系统通常不可能由一家全部承担，采用部分软件外包手段。这时验收制度不仅是开发中的一个环节，而且有利于活跃软件市场，有利于软件产业的形成。

3.软件产品的保护、加密与包装

计算机软件属于我国著作权法保护的作品之列，即可获得版权保护，也可申请专利权保护。知识产权保护与否要看对社会发展是否有利。保护知识产权已受发达国家重视，我们应做到什么程序才有助于现在和将来的发展。关于软件的版权、专利权保护世界各国都存在着争论。由于软件开发成本很高，而复制、拷贝成本很低，许多人认为完全拷贝一个软件是错误的，版权法应保护程序代码，在我国这方面的保护是很不得力的；至于像程序的功能、用户界面等软件表现形式、软件的设计技巧等属于社会的宝贵财富，不能属于版权、专利保护之列，因为这种保护是对社会发展不利的，实际上也是很难实现的。本文上面谈到的很多部分实际上是属于后一方面

的，决不能因为不属于版权、专利就不受重视。各级部门更不应该误导价值观念，使人们不愿意下苦功，急功近利伤害创作及研究的能力，尤其是创新的能力；更不应提倡短期效应，养成只求益，不求精的恶习。

如果软件产品要进入市场，就要考虑某些技术性问题。例如，安全性：预防利用计算机犯罪及受计算机病毒的侵害。在出售机器同时，安装相应软件产品，进行加密，限制任意人随意使用，能防止随意拷贝那就更好了。另外作为软件产品包装，正像前面已谈到的要在人机介面上很好加工，下点功夫，这不仅是表面好看的问题，而主要是为了用户更容易使用。

四、结束语

科技发展日新月异，资讯的快速传递，使全球的生产技术交流迅速异常。新产品的生命周期便愈来愈短，产业界更须时刻动脑筋，推陈出新，开发出新的产品。专家们指出，开发新产品殊非容易，但是只要把握一些关键原则就能达到事半功倍。对于软件产业界更是这样，软件产品的版本迅速不断更新。由于我们对软件的本质、软件开发过程的本质还认识得不是很清楚，在软件开发后不仅要进行修正性维护，还要进行适应性、完善性维护。同时各领域的应用软件还与应用领域有密切联系，与应用领域专家的思维方式有关，这方面的维护工作就更多，因此要开发出一个真正的好软件产品不是一件轻而易举的事情。只有迎着困难上，采取科学的态度。我们觉得在系统设计时应紧紧抓住以用户要求为核心，应用为导向的准则，因为开发出来的软件最后是要交给各领域人员使用的。属时各种应用软件产品将会别开生面，出现一个前所未有的欣欣向荣的景象。

参考文献：

〔1〕*Ann Janda, Human Factors in the Computing Systems, 1984.*

〔2〕*M. L. Schneider et. al. The Humanization of computer interfaces, CACM 4 vol.26 (1983)*

〔3〕*D. A. Norman, Design Rules Based on Analysis of Human error, CACM 4vol.26 (1983)*

〔4〕*A. W. Brown et. al. Learing from IPSE' s mistakes IEEE software, 3, 1992*

〔5〕林三容 *CASE 工具渐趋成熟 资讯与电脑 121 期 (1990.8)*

〔6〕徐家福 软件技术漫谈 计算机科学 *Vol.19 (1992) No.1*

〔7〕国外软件产业的形成和发展 计算机科学编辑部 *1983.4*