微软公司软件开发模式简介

Smalllamb

北京大学出版社 96 年底所出的《微软的秘密》一书是目前我所见到的对微软公司软件产品开发过程介绍的最专业、最深入的一本书。通过本书，我们可以看到微软公司是如何对科学地对软件产品开发进行有效地管理，我想这些经验对于中国的广大软件开发人员，尤其是关心中国软件产业发展的各位朋友是大有益处的。所以特将此书中涉及软件产品开发的部分内容摘录出来（第四章"产品定义与开发过程"），加上我在微软中国工作的实际经验总结出这篇文章，希望与大家共同分享。本文作为摘录，自然是挂一漏万，所以建议大家若有时间还是找来原书一读。

在微软的产品定义与开发过程中，微软软件开发遵循着一种可称之为"靠改进特性（Feature）与固定资源（Resource）来激发创造力"的战略。该战略可分为五个原则：

1、将大项目分成若干里程碑式（Milestone）的重要阶段，各阶段之间有缓冲时间，但不进行单独的产品维护。

2、运用想象描述和对特性的概要说明（Program Specification）指导项目。

3、根据用户行为（User Behavior）和有关用户的资料确定产品特性及其优先顺序。

4、建立模块化的和水平式的设计结构，并使项目结构反映产品结构的特点。

5、靠个人负责和固定项目资源实施控制。

原则一：将大项目分成若干里程碑式的重要阶段，各阶段之间有缓冲时间，但不进行单独的产品维护。

1、项目进度安排与里程碑

微软通常采用"同步－稳定产品开发法"。典型项目的生命周期包括三个阶段：

- 计划阶段：完成功能的说明和进度表的最后制定
- 开发阶段：写出完整的的源代码
- 稳定化阶段：完成产品，使之能够批量生产（Roll Out）

这三个大阶段以及阶段间内在的循环方法与传统的"瀑布"（Water Fall）式开发方式很不相同，后者是由需求、详尽设计、模块化的代码设计与测试、集成测试以及系统测试组成的。而微软的三个阶段更像是风险驱动的、渐进的"螺旋"式的生命周期模型。

计划阶段的产品是想象性描述与说明文件，用来解释项目将做什么和怎么做。在管理人员拟定进度表、开发员写出代码之前，这些东西都促进了人们对设计问题的思考与讨论。开发阶段围绕三次主要的内部产品发布来进行；稳定化阶段集中于广泛的内部与外部测试。在整个产品生产周期中，微软都使用了缓冲时间的概念。缓冲时间使开发组能够对付意外的困难和影响到时间进度的变故，它也提供了一种手段，可以缓和及时发货与试图精确估计发货时间之间的矛盾。

在开发和稳定化阶段的所有时间中，一个项目通常会将 2/3 的时间用于开发，1/3 的时间用于稳定化。（Office 部门副总裁曾这样概述通常的进度："一般说来，在总的进度表中，用一半的时间写出产品，留下另一半的时间调试或应付意外事故。这样，如果我有一个两年的项目，我会用一年来完成事先想好的东西……如果事情有点麻烦，我便去掉我认为不太重要的特性。"）。这种里程碑式的工作过程使微软的经理们可以清楚地了解产品开发过程进行到了哪一步，也使他们在开发阶段的后期有能力灵活地删去一些产品特性以满足发货时期的要求。

- 计划阶段

计划阶段是在一个项目的生命周期中，所有于开发前进行的计划所占用的时间。计划阶段产生出想象性描述、市场营销计划、设计目标、一份最初的产品说明、为集成其他组开发的构件而规定的接口标准、最初的测试计划、一个文档策划（印刷品和联机帮助形式的）以及一份可用性问题清单（Usability List）。计划阶段从想象性描述开始。想象性描述来自产品经理以及各产品单位的程序经理；它是对规划产品的市场营销设想，包括了对竞争对手产品的分析以及对未来版本的规划。想象性描述也可能讨论在前一次版本中发现面必须解决的问题以及应添加的主要功能。所有这些都基于对顾客和市场的分析以及从产品支持服务组处得到的资料。

说明文件从一个大纲开始，然后定义出新的或增加的产品特性，并对其赋以不同的优先级。说明文件只是产品特性的一个预备性概览；从开始开发到项目完成它要增加或变化 20% - 30%。虽然在生命周期的后期说明变化一般较小，但越到后期，开发员就越是必须具充分的理由来作改变。

通常程序经理使用 VB 创建项目原型。他们也开展设计可行性研究以了解设计中的取舍情况，尽快做出涉及产品说明的决定。对于重要产品的说明需由公司高层领导进行复审。对于不太重要的产品，则由部分经理去完成。

- 开发阶段

开发阶段的计划对三四个主要的里程碑版本都逐个分配一组特性，规定出特性的细节和技术上的相关性，记录下单个开发员的任务以及对进度的估计。在开发阶段中，开发员在功能性说明的指导下写源代码，测试员写出测试项目组以检查产品的特性与工作范围是否正常，用户教育人员（User Education）则编写出文档草案。

当测试员发现错误时，开发员并不是留待以后处理，而是马上改正，并在整个开发阶段内使测试不断地、自动地进行。这就改善了产品的稳定性并且使版本发布日期更易估计。当达到项目中的一定阶段点后（40%时），开发员就试图“锁定”产品的主要功能要求或特性，从此只允许小范围的改动。如果在此点之后开发员想作大的改动，他们必须与程序经理以及开发经理进行讨论协商，也许还要征求产品部门经理的意见。

一个项目是围绕着 3 或 4 个主要的内部版本，或“里程碑子项目”来组织开发阶段的。一般用 2 至 4 个月来开发每一个主要的里程碑版本。每个版本都包括其自身的编码、优化、测试以及调试活动。项目为意外事故保留总开发 1/3 的时间，即“缓冲时间”（Padding Time）。（苹果公司的小组是割裂的、独立的，各自开发各自的东西。在还有 3 个月就要发货时，才会将所有的东西集成起来；Borland 公司以一种渐近的方式进行开发，即把工作分成许多小的部分，并且总是让开发的东西能够运转。看起来似乎这种渐进的方法费时，但实际上几乎没有用过很长时间，因为这使你总是能掌握住事情真实的情况。）

当对最后一个主要的里程碑版本做了测试与稳定化之后，产品就要进行“外观固定”（UI Freeze），即确定产品的主要用户界面，如菜单、对话框以及文件窗口等。此后有关用户界面将不再进行大的改动，以免引进同步修改相应文档的困难。

- 稳定化阶段

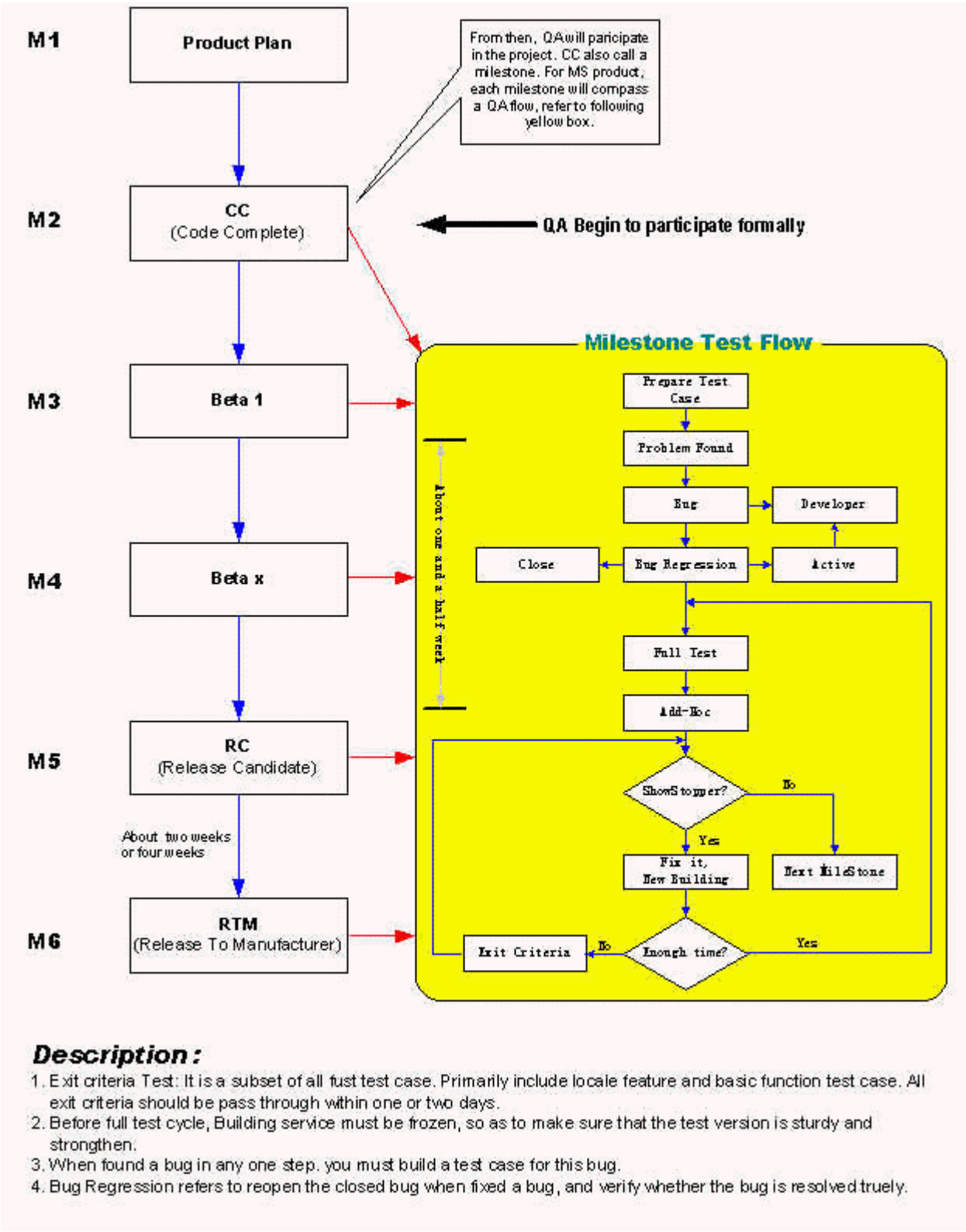
稳定化阶段着重于对产品的测试与调试。项目在此阶段尽量不再增加新的功能，除非是竞争产品或者市场发生了变化。稳定化阶段也包括了缓冲时间，以应付不可预见的问题或者延迟。下面我将 Micosoft 开发软件的模式用以下这张简图加以描述：(这张图对微软的测试进行了比较详细的描述，我个人认为微软的测试是 Microsoft 软件产品开发中一个十分重要也是十分有特色的分工。这是通过在微软将近一年的观察和与国内同类企业的分析，我才得出这样的结论。大家都很明白，国内的软件开发商在这方面做得很不够，尤其不重视软件的内部测试，在他们的思想中，可能有一个误区：认为测试应该完全去由用户去负责，其实不然，在软件的开发流程中，软件的测试与开发是一种“矛与盾”的关系，互为补充，缺一不可。在微软，可能这种关系发挥到了极至：有时开发部门与测试部门互相较着劲，开发经理和测试经理的地位是相同的，有时甚至测

试经理的地位甚至凌驾于开发经理之上，但他们之间没有根本的利益冲突，只有一个共同的目标：将产品的质量提高。）

补充一点：（对微软的测试流程加以简要的描述一下）微软内部，专门有一个小组负责为微软的工程师们提供日常工作和管理的工具软件，他们是非盈利机构，其主要任务是开发微软内部所需要的工具软件：例如：

- SLM(Source Library Tree),源代码管理工具，负责管理软件开发过程中各个程序员的源码，各个程序员负责写自己的模块，每天将完成的代码 Check-in 到一个中央服务器的 SLM 树中，这个 SLM 树由预先定义好的脚本在固定的时间开始编译，通常这个过程需要好几个小时，所以微软内部根据各个项目组的情况有各自的规定：比如开发员必须在下班前（比如下午 6：00）之前将当天修改的代码 Check-in 进去，这样 SLM 才开始编译。
- 第二天，QA 组的各个测试员从服务器上下载前一天的一个 Build 开始测试，将测试的情况及时的反映到另外一个工具软件中：RAID（Raid is a tool for entering, tracking, analyzing, and reporting product defects during development and maintenance.）.这个工具负责管理产品的 BUG 情况，每个 BUG 包含很多属性：比如状态（活动的、解决的、关闭的）、严重级、优先级、哪个区域、哪个版本出现的、发现者、要将这个 BUG 赋给哪一个开发员等等一系列属性。还可以根据这个工具查询哪个开发员当天的 BUG 活动的、解决的数量，哪个测试员的 BUG 质量数目等等一些基本的产品质量情况，这样项目经理可以很容易的掌握该项目的具体进展情况。如果在项目的开发中期，发现的 BUG 数目比解决的 BUG 数目持续的多（意味着该产品的活着的 BUG 越来越多），可能意味着这个项目出现了问题，决策者可以迅速的作出相应的决策，及时的纠正产品开发中出现的失误（微软曾经有很多产品因为这样的因素被 Cancel 了）。还有项目经理可以根据这个工具，及时的掌握、了解每个测试员和开发员的工作状态，这一点很重要。有很多人曾经说过：Microsoft 凭借着 SLM 和 RAID 打败了无数的竞争对手，通过我在微软的经历，我看这话一点也不假。这两个工具确实是非常杰出的工具，微软将它们使用到了十分艺术的程度上，对微软的成功起着非常重要的作用。更难能可贵的是，目前这些工具在功能上还在不断的进行改进、升级，使得微软的工程师们工作起来更加如虎添翼、虎虎生风，这样的企业哪有不成功的道理?
- 在测试过程中，也不是随便的对软件产品毫无目的的瞎使用、乱使用，微软也有一套十分先进的方法和工具支撑着测试的每个方面：比如 ATCM(Access Test Case Management)，一种基于 Test Case（测试用例）的测试管理工具就承担着这方面的工作。

微软也许正是靠着“程序员的聪明和测试员的勤奋”构建起软件帝国的大厦、谱写着软件事业的辉煌。



Product Developing Process in Microsoft

上图中：QA 是微软大的产品部门下设的一个比较专业的测试部门（Quality Assurance Dept）

1、项目进度表中的缓冲时间（Padding Time）

微软使用缓冲计划，以在最高的效率与较好地对未来作预计之间求得平衡。这种应付突发事件的时间在开发和稳定化过程中是每一个主要里程碑的一部分。缓冲时间主要用于弥补由于对

特性（Feature）的不完全理解，或者是技术困难或是由于疏忽而忘记把任务写入进度，或者是未料到的难题而形成的漏洞。缓冲时间有助于一个项目适应意料之外的事件。

原则二：运用想象性描述和对特性的概要说明指导项目

为了给出足够的开发框架以使工作能持续进行，并且能容纳开发过程中出现的变化并保持足够的灵活性，微软采用想象性描述和概要的说明来指导项目开发，而不是在一开始就努力写出一份完整和详细的说明。所谓想象性描述是由程序经理和来自市场营销组的产品计划人员共同编写的一份非常短的文件，在其中主要是定义产品开发的目标（不涉及产品的具体细节 ）。通常对一个全新的产品，想象性描述一般会相对较详细，在其中还含有一份粗略的说明文件。总的来说，微软对于想象性描述的要求是：

越短越好，尽量说明"产品不做什么"(而不是"产品要做什么"！)。

- 运用想象性描述，程序经理开始编写功能说明文件，该文件解释产品的特性是什么以及这些特性如何与其他特性及产品发生关系。最初它只是一个概要性的说明文件，随着项目的进展，程序经理会随时向其中添加更多的细节，最终的说明文件将变得象用户手册一样。完整的说明不只起着对产品最新功能的描述作用，而且它还是在产品投产与发货之前进行测试与评估的主要依据。
- 想象性描述有助于决定删除哪些特性。

微软内的各个开发组采用想象性描述帮助细化产品版本的规定主题，然后以此主题来决定是否需要增加产品各个可能的特性。通常不要轻易改变所确定的主题，否则可能造成产品开发上的混乱。

- 编写说明文件

说明文件在产品小组的所有成员之间，产品小组之间以及产品小组与管理部门之间起着传递产品的设想与要求的作用。在说明文件中必须清楚地描述产品特性（描述每个特性如何工作，外观如何以及从用户的角度出发如何与用户交互。如果特性有一个界面，还应包括一张示意图，以显示出界面的效果），并赋于其相应的优先级。程序经理据此建立起项目的开发进度表。此外在其中还应包括以下各项内容：用一句话表示的项目开发目的，关于产品是什么与不是什么的清单，对顾客的定义，对竞争产品的定义，产品对系统的要求（包括操作系统版本、最小内存要求、硬盘空间、处理器速度以及显示器分辩率），对第三方（如打印机驱动程序、组件）的任何依赖性。程序经理负责协调并"写下"说明

程序经理（Program Manager）应考虑以下问题：

- 这项特性的要点是什么？
- 用户如何使用该特性？
- 这项特性有意义吗？
- 该产品中或微软的其他产品中有类似的特性吗？
- 有哪些问题被遗漏了？
- 组内的交流令人满意吗？

最终程序经理通过与组内开发人员的共同讨论决定有关特性的内容，并将其写下来。

1、构造原型

构造原型是程序经理具体说明一件新产品或一个新版本的最好方法，这从许多方面来说都使开发前测试成为可能，尤其在可用性方面，并且有助于对与用户交互情况作出好的理解，它也能使产品说明更紧凑。

微软的开发人员通常采用 VB 构造用户界面原型，但是对于构造计算机屏幕模型之类的工作，画笔（Paint brush）也是一个很好用的工具。死板的说明变成有生命的文件，说明不应过于详细以至限制了发明创造。在项目开发过程中，说明文件的早期版本会有相当大的增加与改变。

由于说明的变动可能会导致相应开发工作的极大变动，所以微软通常是将精力首先集中于那些没有什么用户界面的特性上，因为在完成开发前不必去了解用户对它们有何反应，也就是说这些特性不大可能改变。然后再面对其它特性。但是当产品开发到一定程序后，例如 40%之后，程序经理必须严格控制对特性的修改（主要是指增加新的特性），否则不光会造成开发延迟，而且会压缩可用的测试时间。

原则三：根据用户行为和有关用户的资料确定产品特牲及其优先顺序

对于一个开发项目而言，如何确定最终产品中应包含什么特性通常是比较困难的一件事。为此微软采用了一个称之为“基于行为制定计划”的方式来进行特性选择与优先级安排。

基于行为制定计划法从对用户行为，诸如写信或做预算，做系统研究开始。然后，根据某一特性在支持重要的或者是经常的用户行为上的程序对其进行评价。这样做的优点是对特性取舍更具理性：讨论对顾客想要做什么加以更好的安排，对某个给定特性是否方便了特定任务的更集中的辩论，可读性更强的说明，以及在市场营销、用户教育和产品开发中更好地同步。

1、特性选择和优先级安排中的基于行为制定计划

基于行为制定计划法中的关键点在于按用户行为、产品特性以及行为和特性之间的内部联系来分析产品。程序经理和产品计划者把产品试图支持的用户任务或方案分成大约 20 个“行为”，然后他们努力把行为（以及任何子行为）映射入微软的现行特性和竞争对手产品的特性中去。他们也把行为映射到不同的顾客形象或不同的市场部分中去。

当说明产品的新版本时，基于行为制定计划法帮助程序经理和开发员集中他们的精力与创造力。象 Excel 之类的项目，争取在每个新版本中加入的主要行为不超过四个。绝大多数特性直接映射入这些行为之中。该做法使项目可以按特性对用户的价值来进行分级。通过分级，促使程序经理和开发人员都行动起来，使他们的特性支持尽可能多的行为。这种良性竞争对于用户有益，同时也利于提高生产率。

2、为顾客行为而非产品特性准备资料

基于行为制定计划进度，项目在计划阶段首先集中于行为，其次才是特性。程序经理和市场营销人员并不去思考和排除他们喜爱的特性，再围绕它们搞出想象性描述的草案。他们真正做的是列出一份顾客都做些什么的清单，然后把想象性描述集中于支持那些行为的特性上。

3、以行为为中心对产品进行全面考虑

由于基于行为制定计划法是从整个产品的观点着眼，因此有助于在不同职能上工作的项目成员理解产品做什么，以及其他产品的相应特性如何可能支持那些需要或不需要其他应用软件产品的行为。

4、做市场营销研究以支持基于行为制定计划法

为支持基于行为制定计划法，从市场营销组来的产品经理与程序经理、开发人员一起开展一些联合的研究，如指导对用户的研究工作。然而，一般来说是产品经理做大多数的研究，并可使其更明确地影响微软产品的演进。

原则四：建立模块化的和水平式的设计结构，并使项目结构反映产品结构的特点

微软产品设计中的一个关键概念是产品的基础结构（Infrastructure），尤其是生命周期短的应用软件，应随项目的进展变得更加单一（而不是错综复杂）。当开发组构造产品的第一版时，他们更多地使用分级式结构，好为产品设计规定出一个最初的架构。随着时间推移，他们向单一的结构迈进，以使项目能集中于特性开发。微软越来越强调不同产品间的特性共享。共享有助于使不同产品的“性能与感觉”（Look and Feel）都统一协调起来；它也方便了需要不只一个应用软

件的用户，减少了代码的重复书写，缩小了单独一个应用软件的规模。

微软用特性小组组织产品开发，这种方法使得每个人都容易明白小组是如何与整个产品相关联的。项目从规定概要说明开始。概要说明的形式是一份已确定了优先级安排的内容清单，涉及产品下一版本将要开发的相对独立的特性，以便由分开的特性小组加以开发。

程序经理和开发员把项目分成特性子集，再将之分配给每个特性小组，让他们在 3 到 4 个主要的内部项目里程碑中进行生产。这种产品组织与开发方法使微软能靠简单地增加开发员和创建一个大的小组来渐进地增加产品的功能。

1、把特性(与函数)作为开发单位

微软软件产品的特性是用户最终可见的相对独立的功能单位，就如建筑材料一般，对应用软件产品更是如此。系统软件产品，如 NT 或者 95 的特性，对最终用户通常不直接可见。微软和其他公司有时简单地称这些不直接可见的特性为“函数”。

程序经理承担开发一组特性或函数，实现从说明经测试、文档化直到最后完成的过程。他们必须与开发员合作，后者负责估计进度表与完善每个特性。开发员还要在一台联网开发计算机上存储一到几个文件，用以保存特性的程序源代码。大多数特性的开发与改进只要一名开发员，而有的大型特性则要一个小的小组。

2、产品结构是决定其长期结构完整性的基石

产品结构是产品内部的基干，它规定了重要的结构构件以及这些构件如何组装到一起。产品结构及用于组装结构的构件，提供了实现产品特性（即做详细设计与编码）的支柱。产品的结构对最终用户而言，通常并非直接可见。只有结构要实现的特性是可见的。产品结构也是决定产品长期结构完整性的基石。产品功能的任何改变都不应造成潜在的产品结构散架。

3、产品的层次结构

对于产品，也可以采用层次结构的方法加以分析。通常定义良好的层次结构有助于对产品特性进行灵活的增加、删除与改进。此外良好的层次结构有助于产品在不同平台上的移植。（例如 Excel 总共定义了五层，其中只有最底层的操作系统层是与平台相关的，其它各层均是通过调用其下层所提供的 API 接口加以实现的，所以其移植极其方便。而在 Windows 95 中通过“虚拟机”的概念实现了对 16 位、32 位以及 DOS 程序的支持。）

4、小的结构文档：源代码是唯一文件

除了 API 文档，微软不对其产品结构生成相应的文档，虽然有时高级开发员可能会写下高层结构。对复杂的特性，许多开发员在某些点记录并复查特定于他们所负责的结构细节，但此工作是可选的，并不强制执行。除了源代码文件与特性说明，为数不多的组为新程序员准务了描绘某层结构的文档（主要的数据结构，如何工作等等）。但是这些文件并不时常更新，经理们也不要求项目组生成此类内部文档。在有关的说明文件中，并不涉及实现问题。开发员应该知道如何去实现，或者能够去学会。记录的关于结构的文档如此之少是因为“一个开发员的工作是编写我们要卖的代码，而不是花时间写高水平的设计文件”，“设计文件不应与源代码分离”。分割代码与“保持事情的简单”。

5、特性小组和作为"内容专家"的小组领导

特性小组一般由一个领导和 3 至 8 名开发人员组成，工作于相关的特性领域。小组的规模常常视小组领导的经验和能力而定。特性小组领导向项目开发领导汇报并负责项目的全部开发工作；而项目开发领导则拥有对产品的更为全局性的观点，从而最有可能发现不互相关联的问题。在特性小组中的每个人均是此领域的“专家”，他们了解如何使用产品、了解竞争对手的产品、了解未来将向何处去。通常为便于交流，提高软件的组织结构（软件倾向于映射出构造它的组织的结构），应保持特性小组的小规模。

原则五：靠个人负责和固定项目资源实旋控制

对于软件项目而言，精确估计产品的开发与交付进度是很困难的。对此微软采取的方法是将进度安排和工作管理的责任推到最底层，即单个的开发人员和测试人员那儿去。这保证了每个人除了作为小组的一部分外，还负有个人的责任。单独的开发人员设立他们自己的进度表，程序经理把单独的进度表汇总起来，再加上缓冲时间，以制定出一个全面的项目进度表。顶层的总经理也固定人员与时间等基本资源，以确保项目集中并限制其努力与创造程序。

关键的目标，尤其对应用软件，是指明产品的目标出品日并争取尽可能长久地坚持它。程序经理和开发员从出品日回溯，规定中间的项目里程碑的日期。这个“固定的出品日法“的中心在开发员身上。以避免因为项目没有固定的结束点，导致在最终无用的设计、再设计和测试的循环中消耗一年或更多的时间。

1、开发人员做出他们自己的进度估计

“日期设定方法"。但是开发人员一般会做出较乐观的估计，因此开发经理还需对他们所提供的日期进行调整并加上缓冲时间以避免因因信息不完全而出现的问题。微软这种制定进度的方法的优点在于：它从人们那儿得到更多的合作，因为日期是自己定的，不是经理定的；进度总是富有进取性，因为开发人员不可避免地会低估他们真正需要的时间。

2、对细致的任务的进度估计

微软的第二个进度安排方法是：对要完成的任务做非常详尽的考虑，在此基础上请开发人员给出他们对“实现”的估计，以此力图“促使”更加现实主义并避免过度低估。

通常微软把任务细化到 4 小时（半天）到 3 天之间。对于准确进度的安排，微软的经理是这样认识的：“任何任务只要超过一星期，那人们就一定没有充分地全盘考虑它。任何任务某人估计只用少于半天就可完成，则他对它考虑得太多了。他应该用更多的时间去编程，更少的时间来考虑”。对于类似类于 Windows NT 之类的操作系统而言，进度安排更加困难，对其一般以几天或者半周为工作单位进行进度估计。

3、安排开发人员与小组进度时的心理学

当项目变大时，微软把员工分成小组。然后经理把进度的责任和所有权尽可能地分发下去，直到小组和个人；这使二者都产生了一种拥用工作的感觉。它还在小组中，个人中，尤其是小组领导中造成强烈的跟上其它同事预计进度的压力，因为经理可能再平衡进度，从落后的小组或个人手中拿走工作。这样，同事间的压力使经理不需要太多的努力就可以对个人或单个小组的进程实施严格控制。

4、"固定的"出品日(　RTM: Release To Manufacture)

为了把创造力约束在时间限制之中，微软现在在新产品或者产品新版本开始前争取固定出品日，至少是有出品日的内部目标。这给人们施加砍去特性和集中在一个项目上的压力，逼迫他们去苦苦思考应将哪个新特性加入产品中。虽然最终产品的交付目标可能是由高级执行人员设定，但是开发人员与小组仍然设定他们自己的进度表。

微软一般根据预先的时间进度的大致估计出一个 RTM 日期，然后向前回溯相应的各个 Milestone 日期，如 RC、Beta、Tree Lock、UI Freeze、Feature Complete 以及 CC（Code Complete）等等各个 Milestone 的相应日期。制定出十分详尽的产品研究开发时间进度表，产品开发组的各个成员以这个进度表为目标统一协调工作。微软十分强调软件开发过程中的 Teamwork Spirits，这种理念贯穿在微软各个产品开发的各个阶段。这也是微软得以成功的一个十分重要的原因。

小结：同步－稳定开发法

- 计划阶段

定义产品的想象性描述、说明与进度

想象性描述 产品和程序管理部门运用广泛的顾客意见来确定和优化产品的特性。

说明文件 基于想象性描述，程序管理部门与开发组定义特性的功能，结构问题，以及各部

分间的相关性。

制订进度表与构造特性小组　其于说明文件，程序管理部门协调进度表，安排出特性小组，每个小组包括大约 1 名程序经理，3 - 8 个开发员，3 - 8 个测试员(以 1:1 比例与开发员平行工作。)

- 开发阶段

用 3 - 4 个顺序的子项目，每个产生一个里程碑式的产品发送，来完成特性的开发。程序经理协调开发过程。开发员设计、编码、调试。测试员与开发员配对，不断进行测试。

子项目 1　前 1/3 的特性：最重要的特性与共享的构件。

子项目 2 中间 1/3 的特性。

子项目 3 最后 1/3 的特性：最不重要的特性。

- 稳定化阶段

全面的内外部测试，最后的产品稳定化以及发货。程序经理协调 OEM 与 ISV，监督从顾客得到的信息反馈。开发员进行最后的调试与代码稳定化。测试员发现并清除错误。

内部测试　公司内部对整个产品做详尽的测试。

外部测试　公司外在的"β"测试点，象 OEM，ISV 以及最终用户处对整个产品做详尽的测试。

发货准备　为批量生产准备发布最后的“金盘”（Golden Disk）与文档，制作之前，还需要进行各种严格的检查：如政治敏感性术语检查、病毒检查、文件相关性检查等。