**Personal Software Process 读书笔记**

**第一章 软件工程师的任务**

一个软件工程师的任务就是要在预定的时间和进度下交付高质量的软件产品。

要进行有效的软件工程的工作，要从以下三个方面进行：

开发出高质量的软件产品；

在预期的费用内进行工作；

在预定的进度下完成任务。

要想使所做的工作富有成效，需要进行下列工作：

制订工作计划；

按照此计划进行工作；

尽最大努力生产出高质量的产品。

小结:软件工程师应该计划要做的工作，然后按照这个计划来工作。这就能够在规定的预算和时间内开发出高质量的产品。个体软件过程（PSP）就是为工程师提供一个如何更好估计和计划工作而设计的框架。

**第二章 时间管理**

为了管理好时间，首先制订时间分配计划，然后按照计划去做。

按照预先制订的计划进行工作会有两点好处：

了解计划中还存在哪些错误，有助于更好地计划下一个项目;

按照计划好的方式完成工作。

有了计划，就不用浪费时间去考虑下一步要干什么，它会帮助你把精力集中在所做的事情上，很少分心，从而提高了工作效率。

小结:为了对时间进行有效的管理，必须对时间进行计划并按照计划来工作。计划和管理时间的第一步是要弄清楚现在是如何利用时间的。为了做到这一点，需要把活动分成几个主要的类。而后，用一种标准的方法把每项活动所花费的时间记录下来，为了方便记录和查阅数据，采用了工程记事本。

**第三章 时间跟踪**

要想改进工作质量，首先要搞清楚你当前要做的事情。这意味着必须了解要完成哪些任务、如何完成以及将要得到怎样的结果。

一旦决定进行时间跟踪，用分钟作为测量单位将比用小时更容易。

提高记录时间数据的一致性和准确性的方法：

随时准备好工程记事本；

当偶尔忘了记录开始时间、结束时间或中断的持续时间时，凭记忆尽早作出估计；

用秒表来跟踪中断的时间；

及时总结记录的时间数据。

**第四章 阶段计划与产品计划**

阶段计划是基于时间段的，时间段即日历上的一段时间，如一天、一周、一月或一年。阶段计划是关于在这段时期内对时间的安排。

产品计划是基于活动的，例如开发一个程序或写一份报告，这种计划称为产品计划。

产品计划在业务运作中很重要，因为市场部要根据产品计划确定什么时候新产品可以卖给用户，财务部要根据产品计划计算开发和制造所需要的费用，决定产品的价格，并向公司管理部提供准确的阶段计划。因此，工程师懂得怎样制订好的阶段计划和产品计划是非常重要的。

**第五章 产品计划**

制订产品计划的第一步就是要有一份待开发产品的清晰定义。只有在知道要做什么之后，才应该开始思考如何去做，此时就是开始制订计划的时候。

一份合适的产品计划包括以下三个方面：

待开发产品的规模及其重要特性;

完成工作所需时间的估计;

项目进度计划。

**第六章 产品规模**

由于各种任务的规模的复杂程度的差异很大，所以掌握一种比较它们规模的方法是很有帮助的当估计编写某个程序所用的时间时，根据以前编写类似程序所用的时间进行估计。

测量程序规模的方法是统计源程序有多少文本行。约定不统计空行和注释行。

无论使用什么测量方法，主要目的是为了估计开发的工作量。基于这一点，需要使用与产品的工作量相关的规模测量方法。

小结:产品计划的第一步是要估计产品的规模。为了作出准确的估计，需要用到以前的规模数据。这样，就可以估计出新程序中各类代码会有多少行。随着所积累的数据越来越多，作出的估计就会越来越准确。

**第七章 管理好时间**

管理时间的步骤：

决定如何使用时间;

制订时间安排表;

对照制订的安排表跟踪使用时间的方式;

决定应该改变什么以使自己的行动达到所作安排的要求。

为了对时间进行管理，必须把重点放在那些占用大部分时间的少数几项活动。明确哪些活动是最重要的，并且考虑是否给这些活动分配了足够的时间。

**第八章 契约的管理**

对于带有合同性质的契约，在它生效之前，两个或是更多的人必须在预期的行动上达成一致。

可以遵照如下的步骤来确保契约是认真制订并且是管理得很好:

在同意契约之前要对所做的工作进行分析;

用一个计划支持契约;

把协议文档化;

如果不能达到契约的要求，及时告诉对方并努力把对对方的影响减小到最小的程度。

小结:本章定义了契约，解释了为什么要履行契约规定的义务以及如何管理契约。对于制订恰当的契约，应有满足契约的合理的保证条件，并要制订一个计划来支持其实现。此外，计划和契约都要存档。

**第九章 进度管理**

制订作业进度的目的是为了完成委托。随着要从事的项目越来越大，精心地安排时间就会越来越重要。要使这些工作无一例外地运转起来，就需要使用一份个人进度表。

Gantt图是展现项目中各个任务进展状况的一种有用的工具。这种图表对于协调多种活动特别有用。

小结:工作顺序按计划进行时，用Gantt图跟踪工作进度；工作顺序与原计划不同时，使用积分跟踪工作进度。

**第十章 项目计划**

项目计划定义要完成的工作和如何做这项工作。项目计划也是一个强大的学习工具。当把这个计划严格地写成文档时，它就是一个与实际性能比较的基准。

小结:本章讲述了如何制订项目计划。对项目计划文档化，这样可以将程序的实际开发时间和规模与估计的数据进行比较，从而学会制订更好的计划。

**第十一章 软件开发过程**

软件工程师的过程质量在很大程度上决定了他们的工作质量和效率，所以过程的设计和管理在软件工程中是非常重要的。

PSP过程的各个阶段：

计划阶段

设计阶段

编码阶段

编译阶段

测试阶段

后置处理阶段

小结:本章定义过程，描述了基本的PSP过程，并且说明已定义过程如何帮助你制订更好的计划。扩展了项目计划总结表，使它包括项目阶段时间，计算每个阶段的累计时间和所花时间的百分比。

**第十二章 缺陷**

软件质量体现在许多方面，但首先要面对的而且必须解决的方面是软件缺陷。

缺陷是指程序中存在的错误，例如语法错误、拼写错误、标点符号错误或者是一个不正确的程序语句。事实上，缺陷是任何影响到程序完整而有效地满足用户要求的东西。因此，一个缺陷是客观的事物，是可以标识、描述和统计的。

减少缺陷对于每个软件工程师都非常重要，这不仅是因为缺陷不仅影响用户的使用，而且一般的软件组织要用一半以上的精力来查找和修复缺陷。由于测试时间难以预计，产品缺陷常常是产生超支和延期的主要原因。

小结:软件缺陷是软件产品中的不正确的东西。缺陷是由人为的错误引起的。因为查找和修复缺陷的代价很大，所以，最有效的方法是工程师能及时发现和修复引入的缺陷。

**第十三章 缺陷查找技术**

发现程序中缺陷的方法包括以下几个步骤：

标示缺陷征兆;

从征兆推断出缺陷的位置;

缺点程序中的错误;

决定如何修复缺陷;

修复缺陷;

验证这个修复是否已经解决了这个问题。

工程师最常用的发现和修复缺陷的工具是编译器。

发现错误的的第二种方法是测试。

最后，最有效的发现和修复缺陷的方法是个人复查源程序清单，虽然这种方法似乎很难彻底清除程序中的缺陷，但事实证明，这是最快而且是最有效的方法。

小结:在PSP中主要的缺陷排除方法是个人代码复查。因为人们往往容易犯同样的错误，所以缺陷数据有助于在第一次编译和测试前找到全部或绝大部分程序的缺陷。

**第十四章 代码复查检查表**

检查表包括一系列规程式的步骤，并要求你精确地遵循这些步骤去做。如果想发现和改正程序的每一个缺陷，就必须遵照一个精确的规程。检查表包括了个人的经验。通过不断地使用和改进个人检查表，可以更好地发现在程序中的缺陷。

使用检查表要注意的问题是，通常只能找到那些已知可能的缺陷。但是，有些严重的问题经常是难以预料的。因此，至少要对程序进行一次全面的审查，以查找那些未曾预料到的问题。

小结:本章介绍了代码复查检查表以及如何设计和使用自己的某种检查表。检查表不但能帮你找到更多的缺陷，而且还能帮你更快地发现缺陷。在构造代码复查检查表时，要按照所使用的编程语言进行裁减，要根据缺陷数据进行设计，而且要随着技能和经验的提高，对原有的代码复查检查表作适当的调整。

**第十五章 缺陷预测**

准确地估计程序中缺陷的个数是很重要的。通过把当前的项目数据和以前的数据相比较，就能大概知道正在开发的程序的质量情况。在PSP中，基本的缺陷测量是以每千行代码的缺陷数（Defects/KLOC）来测量的。称为缺陷密度（Dd），其测量单位是Defects/KLOC。当为一个新的程序作计划时，先估计出程序可能有多少新的和修改的代码行。然后计算以前开发程序的每千行源代码的平均缺陷数。

小结:制订缺陷预测计划时，你需要以前程序的缺陷和程序规模的数据以及新程序的规模的估计值。

**第十六章 缺陷排除的经济效益**

为了了解和控制缺陷带来的费用，很有必要测量缺陷排除的效果。有一种测量方法是计算每小时排除缺陷的个数；另一种是计算缺陷排除效益，即测量通过某一排除方法所发现的缺陷的百分比。当知道每种缺陷排除方法的排除效益和排除率之后，就知道如何更好地发现和修复缺陷。

小结:缺陷是引起软件问题的主要原因。修复缺陷需要花钱，也会延迟项目进度。由于缺陷修复的问题难以预测，因此它会降低计划的准确性。因此，缺陷管理可以被看成是一个经济学问题。

**第十七章 设计缺陷**

事实上，在编码阶段引入的缺陷是设计阶段引入的缺陷的一倍半。测试阶段发现的缺陷大部分是设计类型的缺陷，但它们多数是在编码阶段引入的。任何与程序结构和实现有关的内容都涉及到设计；这些内容包括程序流程、语言构造的结构和组成以及主源代码的标点符号等。在进行设计时，有经验的工程师经常在各个设计层次之间动态地进行切换[Curis]。原因是他们要处理很多抽象功能。有时当系统设计建立在一个看似简单但却包含有定义笨拙的抽象上时，最终会发现整个设计方案是不可行的。

设计缺陷产生的几种原因：

设计错误（即想法错了）;

知道应该做什么样的设计，但犯了一个简单的错误;

误解了要做的内容;

不但理解局部设计和系统设计，甚至还理解需求，只是不明白系统的语境。

**第十八章 产品质量**

两种方法：

了解编译阶段、测试阶段和用户使用阶段发现的缺陷数据之间的关系。;

收集关于自己工作情况的数据，并且看一看这种策略产生的效果。

对于大型复杂的程序，发现和修复每个缺陷的花费很大，能发现的缺陷的百分比在下降。

遵照规范化的方法，仔细复查和测试每个小程序模块，这比让任何测试小组在你的程序中发现缺陷的效果要好。如果你最终能肯定程序模块中已经没有缺陷，将大大减少以后清除程序缺陷的时间和花费。

小结:开发高质量的大程序的策略是首先要排除组成大程序的模块中的所有的缺陷。缺陷排除是一个过滤过程：把每个缺陷排除阶段看成是一个过滤器。放入过滤器的缺陷越多，将要发现的缺陷就越多。同样，放入过滤器的缺陷越多，将要漏掉的缺陷也就越多。在过程中的每个阶段，仔细工作都是有回报的，而且也节省时间。

**第十九章 过程质量**

程序的质量取决于过程的质量，而过程的质量又取决于人们的工作方式。最基本的过程测量包括所生产产品的产量、产品的质量以及完成工作需要的时间和资源。

一个软件工程师需要全面地权衡所花费的时间和所生产的产品质量等问题。质量成本（COQ）包括三个主要元素：过失成本、质检成本和预防成本。

小结:PSP使用简化的质量成本计算方法。质检成本用代码复查时间除以总开发时间再乘100；这个计算忽略了代码复查阶段的缺陷修复时间。过失成本用编译和测试时间之和除以总开发时间乘以100；这个计算忽略了无缺陷的编译和测试时间。

**第二十章 个人对质量的承诺**

作为软件工程师，你所生产的程序的质量对你的雇主和客户都至关重要。重要的是所有的软件工程师都要认识到，他们的工作对千百万人的健康、安全和幸福将有至关重要的影响。

当软件缺陷打破约定时，将会出现不可预测的后果。因此，为了拥有真正的安全系统，必须努力生产出无缺陷的软件。

当工程师对质量作出承诺时，他们就会在乎缺陷。当他们对此关心时，他们在工作中就会更加精心。当他们更加精心时，他们就会生产出更好的产品。

SE2018春-G11

陈铭阳