



# 基于过程控制的 个体软件质量管理：PSP

杨丽娜 (四川大学人口研究所 610064)

**摘要：**软件组织的过程管理既依赖于组织过程的改进，也依赖于组织中个体过程的优化。如何提高个体的工作技能，改进个体的工作方法是软件组织和个人面临的一个重要问题，PSP（个体软件过程）的研究在这个问题上做出了卓有成效的探索。本文试从PSP的发源和概念上进行阐述，讨论了PSP的过程框架和进化框架。

**关键词：**个体软件过程 过程框架 进化框架

## 1 引言

软件的质量取决于软件过程的质量。过程质量既依靠软件组织对软件过程的宏观管理，又依靠个体对软件过程的微观优化。在宏观管理上，通过实施CMM（软件能力成熟度模型）来指导软件过程的改进已被普遍接受；在微观优化方面，必须得到软件工程师们对过程改进的理解和支持，这就需要一套系统的方法从个体角度来保障过程改进的实施。PSP（个体软件过程）为软件工程师们提供了发展个人技能的结构化框架和必须掌握的方法，它为个体软件过程提供了文档化管理的工具，它使个体在软件开发过程中摒弃了习惯性的工作方式，使个体过程朝着更为规范化、可控制性的方向发展。

CMM只为软件企业提供了组织过程改进的框架，并未涉及实现关键过程域所需要的具体知识和技能，而PSP为个体提供了这种方法和技能，它的实施是软件企业CMM升级的有力保证。本文试从PSP的发源和概念上进行阐述，主要介绍PSP的过程框架和进化框架。

## 2 PSP的提出及其概念

在产品质量管理发展历程中有这样具有重大意义的一步，20世纪80年代，戴明(W. Edwards Deming)和朱兰(J. M. Juran)博士提出：通过改进人们的工作方法来提高产品的质量。1987年卡内基-梅隆大学软件工程研究所(CMU/SEI)的亨弗雷(Watts S. Humphrey)等人提出软件能力成熟度模型CMM (Capability Maturity Model)，它致力于软件开发组织的软件过程能力成熟度的提高。在这

之后他又从事如何改进软件组织中个体的过程意识和过程能力的研究，并发现Deming和Juran的过程管理原则非常适用于软件工程师们的工作，经过三年多的研究提出了PSP，并于1993年开始在卡内基梅隆大学教授PSP的课程。至今，美国、欧洲、澳大利亚等国已先后有20多所大学开设了PSP课程，我国也于1997年开始在北航软件工程研究所组织了PSP的应用实验。

根据CMU/SEI实施PSP的经验表明：在接受PSP的培训后工程师们比培训前少引入58%的错误；在开发时间估计上的错误率由39.4%下降到10.4%；排除了重复劳动从而减少循环开发的时间；它教给软件工程师不断地进行软件过程度量、管理和改进的方法使其成为过程的控制者；PSP的实施还使软件产品的生产率提高了20.8%。

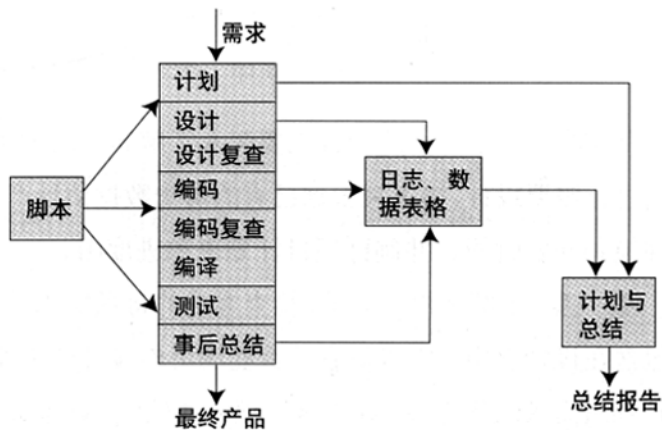


图1 PSP的过程流程

PSP (Personal Software Process) ——个体软件过程, 是一种可用于控制、管理和改进个人工作方式的自我改善过程, 它包括了软件开发表格、指南和规程的结构化框架。PSP能有效地帮助工程师们持续地使用正确的工程方法, 指导他们如何计划工作, 对工作进行追踪记录, 通过对工作结果的分析改进下一计划中的个体行为过程。它使工程师们从自己及他人的实际工作中总结并找到最适合特定任务及自己能力的工作方法, 提高了个体的过程意识和过程能力。

### 3 PSP 的过程框架

PSP 的过程框架通过图 1 表示:

在 PSP 的过程流程中有一个核心的框架:

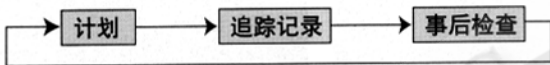


图 2 PSP 的核心过程框架

在过程流程中, PSP 包含了计划、追踪记录、事后检查这一不断循环的过程, 经过经验累积、方法的改进、技术的提高, 个体工作过程得到持续改进。

#### 3.1 计划

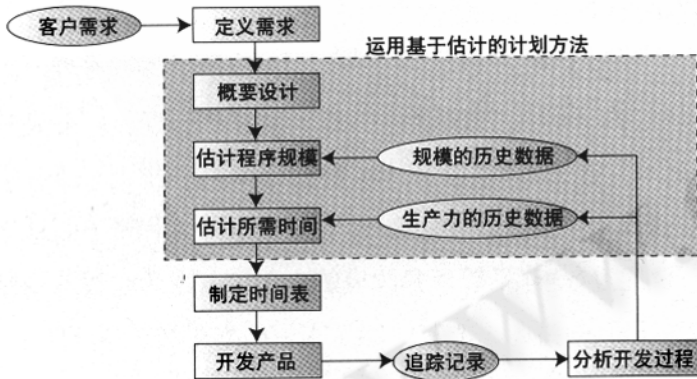


图 3 计划过程

计划阶段有三项任务, 首先要深入了解客户的需求, 然后进行概要设计, 接下来根据已有的历史数据估计产品的规模和开发时间, 并制订项目计划表和进度图。

第一步, 了解客户的需求。检查客户的每项需求是否都包含在设计之中, 需求信息是否能满足你设计的需要, 并且判断需求的优先级。如果这三项无法确定, 就难以避免在设计阶段引入缺陷。

第二步, 概要设计。在需求明确后, 便进行产品的概要设计, 虽然它还不是一个完整的产品设计, 但基于这

一概要性设计, 可以对开发规模和所需资源做出估计。

第三步, 也是设计阶段最重要的一步, 估计程序规模、开发时间和开发所需资源(包括软件资源、硬件资源、人件资源)。PSP 提供的方法使工程师通过收集以前不同任务所用时间的数据, 引入基于估计的计划方法 PROBE (PROxy Based Estimating), 根据历史数据来预测新程序的大小和所需的开发时间, 并在每次总结时将计划与实际比较, 不断地提高其估计的能力。

对程序规模进行估计, 采取的方法是把计划要做的工作与以前做过的相似工作相比较, 根据至少前三个程序的历史数据估计出此次开发实际要编写的总代码行(只计算新编写的和更改的代码, 而不应将重用的代码计算在内), 同时估算出不同功能程序的大、中、小代码尺寸。

得到程序规模的估计值后, 工程师根据自己的开发效率(分钟/LOC)可得到其负责模块所需开发时间的大、中、小值, 加总各模块的开发估计时间便得到总的时间估计值。之后, 工程师们参考最近一个软件中各个阶段所花时间的累计百分比, 分派开发各阶段的时间, 并填写项目计划表。然后将每阶段的工作细分, 确定各阶段的任务组成及工作量, 制订出工作进度表。

笔者认为对开发时间的估计不能只考虑程序规模和开发效率, 程序的创新性和开发者对已有资源的掌握程度均影响着开发时间(见图 4), 因此应做综合性的考虑。

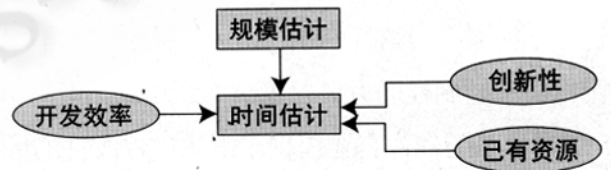


图 4 影响开发时间的四个因素

#### 3.2 跟踪记录(也可称为数据收集)

在PSP中, 工程师们跟踪自己的工作, 收集在各过程中产生的时间数据、规模数据以及产品质量数据, 在这一过程中, 日志和表格是重要的工具。

时间度量 人们通常是按照过去使用时间的方式和经验对今后的时间进行管理和计划, 因此, 对开发中时间的跟踪和记录是必要的。时间记录日志(见表 1)可以提供一份完整而准确的有关时间花费的详细记录, 它记录了每项工作的开始时间、中断时间、结束时间, 通过计算和总结, 工程师可以了解花费在具体开发工作上的有效时间,

这既帮助其对时间进行管理,又为其在下一个软件开发项目中提供了时间计划所需的数据依据。

规模度量 在开发中,工程师们对产品规模进行度量和记录(见表2),目的是为了计算总工作量,这为下一项目的规模估计提供了历史数据,使其准确性提高。产品规模包括了程序的规模、文档的大小及其他使用辅助工具生成的程序要素,但不同要素使用的规模度量方法有差异。文档的大小以页为单位进行度量。利用辅助工具生成的程序要素,只统计自己编写的代码行 [3]。

表 1 时间记录日志

日期	开始时间	结束时间	中断时间	净时间	活动	备注	完成与否

表 2 程序规模记录日志记录日志

程序	开发时间(分钟)	逻辑代码行	分钟/逻辑代码行	功能

表 3 缺陷记录日志

日期	编号	类型	引入阶段	排除阶段	更改时间	相关缺陷
描述:						

度量程序规模的方法是使用代码行 LOC (lines of code), 不统计空行和注释行。在循环开发中,代码行可分为不同的种类:基本的、增加的、更改的、删除的及重用的。计算工作量时,我们应只考虑新增的和更改的代码。

质量度量 在PSP中质量管理的焦点是缺陷管理,运用缺陷记录日志(见表3),工程师们记录下开发各阶段发现的每一个缺陷,及其引入阶段、排除阶段、排除所花的时间,对这些数据进行分析,可得出缺陷密度(defect density)、缺陷比率(defect ratios)、检查速度(review rate)、开发时间比率(development time ratios)、缺陷产量(yield)、每小时缺陷量(defects per hour)、缺陷排除效果(defect removal leverage)、质检/过失比(A/FR)等指标。

### 3.3 事后检查

在软件开发工作结束后,项目组必须进行事后的分析总结(见表4)。利用项目计划总结表中的数据,可以做以下四方面工作:第一,通过将实际数据与计划数据进行比较和分析,检查实际工作与计划的差距;第二,更新有关程序规模、开发时间、开发效率、缺陷密度等历史数据,为以后项目的估计提供依据;第三,将该项目开发中的过

程数据与历史数据比较,追踪软件过程改善情况;第四,对于个体来说,通过不断总结,帮助他们在以后的工作中更有效地发现和改正曾出现的错误。

表 4 PSP项目计划总结表

总结	计划	实际	累计	
LOC/小时				
程序规模				
基本的				
删除的				
更改的				
增加的				
重用的				
新开发与更改的总数				
总的代码行				
开发时间(分钟)	计划	实际	累计	累计百分比
计划				
设计				
编码				
编译				
测试				
事后总结				
总计				
引入的缺陷	计划	实际	累计	累计百分比
计划				
设计				
编码				
编译				
测试				
总结				
排除缺陷	计划	实际	累计	累计百分比
计划				
设计				
编码				
编译				
测试				
总结				

## 4 PSP的进化框架

作为一种方法,PSP有一个不断提高的过程(见图5),这与CMM成熟度等级的递进相似,前者指示个体技能所达到的水平,后者指示软件企业软件能力的一个等级。

PSP中的第一步是建立个体过程基线,其要求包含在PSP0中,即如何度量,如何使用各种表格进行过程数据的采集。这一步为过程度量和过程改善提供了基础。

PSP1是在PSP0基础上演进的,它加入了个体计划的能力。在这一步,工程师们要学会如何用自己在开发中积累的历史数据来估计新程序的大小和所需的开发时间,并对任务和进度进行规划。这使工程师们具备了对自己做出客观评估的能力,有利于开发计划的准确性。

在PSP2中,其重点是个体质量管理,它在PSP1的

基础上增加了设计复查和代码复查,通过建立的检测表进行复查,使缺陷得以尽早发现,以最小代价进行修复。复查技术是一个合格的开发人员应必备的,它使缺陷的处理遵循着适当的软件过程,有利于软件质量的保障。在PSP2.1中增加了设计过程和设计模板技术,解决了设计完备性准则和设计验证技术的问题。

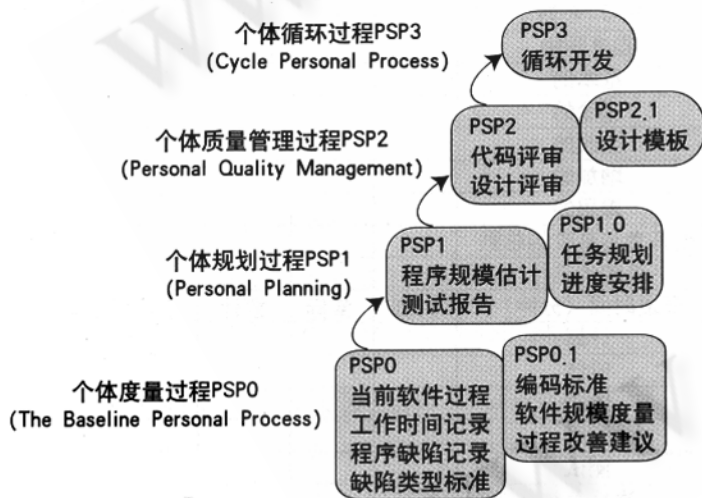


图5 PSP的进化框架

在PSP3中引入了增量式迭代开发方法,使个体在大程序开发中能应用PSP2中的方法,把个体开发小程序所能达到的生产效率和生产质量延伸到了大型程序开发中。

掌握PSP的技术和方法不是一蹴而就的,而是从对当前过程的记录开始,先通过开发中收集的数据对自己的能力有一个客观的评估,然后才是计划能力、个体质量管理能力、个体设计能力的获得和提高,只有对上一步技术的积累,个体软件过程才能不断改善。■

#### 参考文献

- 1 Watts S.Humphrey. *The Personal Software Process* [R].CMU/WEI-2000-TR-022.
- 2 何新贵等, 软件能力成熟度模型 [M].清华大学出版社,2001年。
- 3 Karl E.Wiegers. *Personal Process Improvement*[c].*Software Development magazine*, May 2000.
- 4 Watts S.Humphrey. *The Personal Process in Software Engineering* [R]. 1994.
- 5 Carnegie Mellon University. *Introducing Personal Software Process and Team Software Process Into an Organization*