**个人软件过程**

**(Personal Software Process，PSP)**

PSP是由美国Carnegie Mellon大学软件工程研究所(CMU/SEI)的Watts s. Humphrey领导开发的，于1995年推出，在软件工程界引起了极大的轰动，可以说是由定向软件工程走向定量软件工程的一个标志。

PSP是一种可用于控制、管理和改进个人工作方式的自我持续改进过程，是一个包括软件开发表格、指南和规程的结构化框架。PSP与具体的技术（程序设计语言、工具或者设计方法）相对独立，其原则能够应用到几乎任何的软件工程任务之中。

PSP为基于个体和小型群组软件过程的优化提供了具体而有效的途径，例如如何制订计划，如何控制质量，如何与其他人相互协作等等。

**PSP能够干什么？**

1.说明个体软件过程的原则；

2.帮助软件工程师作出准确的计划；

3.确定软件工程师为改善产品质量要采取的步骤；

4.建立度量个体软件过程改善的基准；

5.确定过程的改变对软件工程师能力的影响；

**PSP的原则和思想**

PSP的一个基本原则是：每个人都是不同的，对于某个工程师有效的方法不一定适合另一个，PSP帮助工程师测量和跟踪他们自己的工作，使得他们能够找到最适合自己的方法。

PSP内容丰富，具有良好的实践性，包括个人时间管理、时间跟踪、任务估计和阶段性工作计划等内容。基于PSP，通过采用一些表格、脚本和标准，可帮助软件工程师估算和计划其个人的任务，改善个人软件过程及测量，从而最终获得高生产率的回报，能够在规定的预算和时间内开发出高质量的产品。

其目的是为了改善软件工程师的开发性能，而提高性能在于早期对项目有一个比较准确的把握。这不仅帮助软件工程师提高编码水准，如何管理所开发的程序质量；还指导软件工程师更好的进行需求或过程定义、评审、测试和文档编写等。

**PSP的作用**

1.使用自底向上的方法来改进过程，向每个软件工程师表明过程改进的原则，使他们能够明白如何有效地生产出高质量的软件。

2.为基于个体和小型群组软件过程的优化提供了具体而有效的途径。其研究与实践填补了CMM的空白。

3.帮助软件工程师在个人的基础上运用过程的原则，借助于PSP提供的一些度量和分析工具，了解自己的技能水平，控制和管理自己的工作方式，使自己日常工作的评估、计划和预测更加准确、更加有效，进而改进个人的工作表现，提高个人的工作质量和产量，积极而有效地参与高级管理人员和过程人员推动的组织范围的软件工程过程改进。

**PSP0和PSP0.1（个体度量过程）**

PSP0的目的是**建立个体过程基线**，通过这一步，学会使用PSP的各种表格采集过程的有关数据，此时执行的是该软件开发单位的当前过程，通常包括计划、开发（包括设计、编码、编译和测试）以及后置处理三个阶段，并要作一些必要的试题，用作为测量在PSP的过程中进步的基准。

PSP0.1增加了编码标准、程序规模度量和过程改善建议等三个关键过程域，其中过程改善建议表格用于随时记录过程中存在的问题、解决问题的措施以及改进过程的方法，以提高软件开发人员的质量意识和过程意识。

**PSP1和PSP1.1（个体规划过程）**

PSP1的重点是**个体计划**，引入了基于估计的计划方法PROBE（PROxy BASed EStimating），用自己的历史数据来预测新程序的大小和需要的开发时间，并使用线性回归方法计算估计参数，确定置信区间以评价预测的可信程度。在PSP1阶段应该学会编制项目开发计划，这不仅对承担大型软件的开发十分重要，即使是开发小型软件也必不可少。只有对自己的能力有客观的评价，才能作出更加准确的计划，才能实事求是地接受和完成客户（顾客）委托的任务。

PSP1.1增加了对任务和进度的规划。

**PSP2和PSP2.1（个体质量管理过程）**

PSP2的重点是个体质量管理，根据程序的缺陷善建立检测表，按照检测表进行设计复查和代码复查（有时也称"代码走查"），以便及早发现缺陷，使修复缺陷的代价最小。随着个人经验和技术的积累，还应学会怎样改进检测表以适应自己的要求。

PSP2.1则论述设计过程和设计模板，介绍设计方法，并提供了设计模板、但PSP并不强调选用什么设计方法，而强调设计完备性准则和设计验证技术。

**PSP3（个体循环过程）**

PSP3的目标是把个体开发小程序所能达到的生产效率和生产质量，延伸到大型程序；其方法是采用螺旋式上升过程，即**迭代增量式**开发方法，首先把大型程序分解成小的模块，然后对每个模块按照PSP2.1所描述的过程进行开发，最后把这些模块逐步集成为完整的软件产品。

**时间管理**

管理时间，第一步就是要先做计划，然后把该计划写下来，以便今后与实际数据相比较。首先制定时间分配计划，然后按照计划去做。如果要制定出更准确的计划，就需要知道以前的计划中存在哪些错误，哪些地方可以进行改进。

按照计划进行工作有三点好处：

1. 了解计划存在哪些问题，有助于更好的计划下一个项目。
2. 按照好的计划完成工作，从而避免一些错误。
3. 它实际上在改变你的工作方式，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中的事情上，很少分心，从而提高了工作效 率。

可以按照如下步骤管理时间：

1. 分析自己使用时间的历史记录；

2. 制定时间安排表，决定如何使用时间；

3. 对照制定的安排表跟踪使用时间的方式；

4. 决定应该改变什么意思自己的行动达到所作安排的要求。

**制订计划**

阶段计划是关于这段时间内对时间的安排，产品计划是关于制作产品活动期间的时间安排。

制定阶段计划，必须清楚时间的使用情况。制定这种计划最简单的方法就是假设将要使用的时间与过去平均使用的时间相同。一种较为精确的方法就是首先考虑下周将要做的工作内容，然后根据以前的最长和最短时间来估计出一个合适的时间。

制定产品计划是可以通过事件加以提高的一种技能。从现在开始对每个产品制订计划，产品可以是一个可制定的程序、一个程序设计方案或是一个测试计划，并在以后的项目中继续这样做下去。

**缺陷管理**

缺陷是指程序中存在的错误，例如语法错误、标点符号错误或者是一个不正确的程序语句，是任何影响程序完整而有效的满足用户要求的东西，是可以表示、描述和统计的客观事物。

缺陷并不像无足轻重的BUG，更像是定时炸弹。很小一部分看起来无关紧要的缺陷都可能引起严重问题。

引入缺陷是人类的正常现象，所有的工程师都会引入缺陷。因此所有的工程师都应该了解自己引入缺陷的类型和数据。

为了减小缺陷，就必须进行缺陷管理，研究已经引入的缺陷，确定引起这些缺陷的原因，并学会在将来如何避免重复同样的错误。

代码复查是发现缺陷的一种办法，代码复查就是研究源代码，并从中发现错误。但是复查非常耗时，而且很难恰当的进行，这是复查的一个缺点。

如果在代码复查方面花了足够的时间，你的过程会迅速稳定下来。一旦你的过程相当稳定，缺陷也将容易预测。如果能够很好地适应这一切。代码也将变得更规范更易维护。