基于生产线方式的软件开发

屈庆明　赵昊翔　潘金贵

（南京大学软件新技术国家重点实验室　南京大学计算机科学与技术系　南京210093）

摘　要　计算机在不同行业的广泛使用，需要大量的软件作为支撑。越来越多的软件企业发现按照传统的方式即一次开发一个软件的方式已经不能够满足需求，它们需要提高软件开发的效率，而基于生产线方式的软件开发可以较好地解决这个问题。本文介绍了基于生产线方式开发软件的基本概念及其优点，软件生产线与软件成熟度模型集成的关系，以及实施软件生产线过程中可能用到的度量标准，最后指出实施软件生产线过程中可能引入的风险。

关键词　软件生产线，软件成熟度模型集成，度量标准

Product Line-Based Software Development

QU Qing-Ming　ZHAO Hao-Xiang　PAN Jin-Gui

(National Key Laboratory of New Software Technology, Department of Computer Science , Nanjing University, Nanjing 210093)

Abstract　While more and more software products are needed with the popularity of computers, an increasing number of organizations are realizing that they can no longer afford to develop multiple software products one product at a time. They should improve their productivity, and they find that product line-based software development is a good choice. This paper introduces the concept, advantages and measures of product line-based software development, demonstrates the relationship between Software Product Line and CMMI. In the end we indicate the risks when adopting Software Product Line.

Keywords　Software product line, Capability maturity model integration (CMMI), Software measurement

1　引言

提高软件开发的质量和效率一直是软件企业追求的目标，但从对软件产品的各种调查来看，效果并不理想。美国政府软件开发项目统计数据显示软件真正得到成功应用的比例并不高，相当大比例的软件没有能够如期交付使用，这对于很多时效性要求非常强的软件（譬如手机软件等）来说也就意味着失败。同时，随着计算机在各个行业中的深入应用，需要越来越多的软件提供支持，软件企业发现如果再用传统的软件开发方式即同一时间开发一种软件已经不能够满足这种需求。软件工程的学者们从传统工业界汲取营养：如果能够像传统工业界中的生产流水线生产工业产品一样来开发软件，无疑会大大提高软件开发的效率和质量，从而更好地满足日益增长的软件需求。正是基于这一观察，软件生产线的概念应运而生。目前有很多国外的大型软件企业已经在软件生产线方面进行初步尝试并且取得了丰厚的回报[1]。譬如瑞典海军的软件开发商 CelsiusTech 使用了一个名为 Ship System 2000 的软件生产线使得其生产过程中的软硬件消耗比从65：35变为20：60，大大降低了软件开发的费用以及时间[2]。

2　软件生产线相关概念

2.1　基本概念

一条软件生产线（SPL：Software Product Line）是指一组具有下列特征的软件密集型系统[3]：

（1）它们共享一组满足特定任务或特定市场需求的公共可管理的特征；

（2）它们以某种预先指定的方式从一组公共核心资产（Core Assets）中被开发出来。

软件生产线的定义与传统工业的生产线定义是一致的，但是又增加了一些新的东西，表现在它对软件生产线中开发软件的方式进行了限制。显然以预先指定的方式从一组公共资产中生产软件比从头开始或者以随意的方式开发软件会获得更大的效益（无论从时间还是从金钱方面来衡量），这也正是软件生产线能够提高软件开发效率的原因所在。在实施软件生产线的企业中，开发一种新的软件更像传统工业的组装和生产而非传统软件产业的创新。初次接触软件生产线概念的人可能会把它同别的概念混淆起来，我们通过下面的表格来进一步理解这个概念。

表1　SPL 与其他概念的比较

易混淆的概念	软件生产线与它们的区别
偶然的小粒度复用	SPL 中的复用是有计划的，全面的，可以提高利润的
单个系统开发中的复用	SPL 中的复用资产是设计良好的；SPL 中的软件是整体的
基于组件的开发	SPL 中的组件以预先指定的方式搭配
可重新配置的体系结构	可重新配置的体系结构只是 SPL 的一个核心资产
一种产品的多个版本	SPL 中多种产品同时存在，旧版产品仍然可以是资产
一组工业标准	标准只是 SPL 的一种输入

屈庆明、赵昊翔　硕士研究生，研究方向为软件工程、软件过程改进等。潘金贵　教授，博士生导师，从事中间件、Agent 技术及多媒体远程教学系统等的研究。

从表1我们可以看到软件生产线不同于其他的六个概念，它是一种全新的软件开发的方式。

2.2 基于生产线方式的软件开发

一般来说，使用软件生产线方式进行软件开发包括三个关键活动：

(1)核心资产开发(Core Assets Development)

(2)软件产品开发(Software Product Development)

(3)支持上面两个部分的技术和组织管理。

整个过程可以用图1来表示：

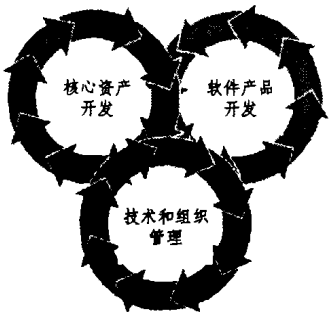

图1 软件生产线开发示意图

正如图1中所描述的，每个环分别表示软件生产线中的关键活动，三个关键活动之间存在紧密的互动关系：核心资产可以用来开发软件产品，而在产品开发的过程中也会更新甚至增加新的核心资产。下面对这三种关键活动分别做一个比较详细的介绍。

2.2.1 *核心资产开发* 核心资产是软件生产线能够生产软件的基础，一般来讲，一条软件生产线的核心资产包括以下内容：软件生产线中软件的体系结构，用以复用的软件组件，以及所有用来开发软件的相关设施，包括语言，工具等等。我们可以把核心资产开发看成一个黑盒子，它的输入为：

(1)对软件产品的约束：主要是软件产品的公共属性以及各自特征等；

(2)对软件生产的约束：软件开发时间约束，对软件工程师的要求等；

(3)软件生产策略：软件生产如何管理等；

(4)已存在资产列表。

核心资产开发的输出包括：

(1)产线的适用范畴：指出哪些产品可以被该生产线开发；

(2)核心资产：每种核心资产都附带一个处理过程用来指出怎么用这个核心资产来开发软件，这是利用核心资产开发软件的前提；

(3)生产计划：指明软件产品怎样用核心资产开发出来。

需要指出的是，核心资产的开发是一个循环的过程，正如上面所提到的，新开发的产品可能会对是已有的核心资产库造成影响：更新已有核心资产或者增加新的核心资产。另外核心资产的开发是在软件生产线的第三个关键活动即技术和组织管理的支持下进行的。

2.2.2 *软件产品开发* 软件产品开发活动是利用软件生产线进行软件开发的核心，它的输入是：

(1)特定产品的需求

(2)软件生产线的适用范畴

(3)软件生产线的核心资产

(4)软件生产计划

后面三个输入恰好是核心资产开发的输出，软件产品开发的输出就是软件产品。同核心资产开发一样，软件产品开发也是在技术和组织管理的支持下进行的。

2.2.3 *技术和组织管理* 基于生产线方式的软件开发中的各种活动都需要相应的资源以及各种支持，这是技术和组织管理所提供的。技术管理的职责在于能够随时发现核心资产开发和产品开发中的问题，以使两者沿着软件生产线的目标前进。而组织管理的职责在于建立合适的软件生产组织结构并确保各个组织单元能够得到所需的资源。

上述三者都是软件生产线中极其关键的活动，一般来说，将三者结合起来的方式有两种：前摄式和反应式，前者指企业直接从核心资产开发出发来构建软件生产线，后者则指企业从以后的产品来开发核心资产，这些是在上面所说的软件生产策略中指定的。

上面对使用生产线方式进行软件开发做了简要的介绍，为了能够具体进行软件生产线的实践，SEI 又定义了很多实践域(Practical Area)，实践域指定软件企业如何具体地来操作上面的三个关键活动。

3 软件生产线和软件过程改进

3.1 CMMI 简介

为了提高软件质量，软件工程研究者向传统工业界借鉴经验，从工程的角度来看待软件开发过程，提出了软件过程改进的概念。卡内基·梅隆大学(CMU)的软件工程研究所(SEI)提出的软件能力成熟度模型(SW-CMM)为软件过程改进提供了一个很好的范本，它在许多软件企业的成功应用大大提高了软件开发的质量，目前已经成为一种事实标准[4]。SW-CMM 的成功使得其它领域(譬如系统工程领域)的类似模型不断涌现(譬如 SE-CMM)，同时应用多个模型虽然会给软件企业带来好处，但是也会使复杂度上升，不易操作，SEI 的 CMMI 项目组就是为解决这个问题而成立的。经过努力，这个小组提出了软件成熟度集成的概念，包括下面几个模型：

(1)CMMI-SE/SW

(2)CMMI-SE/SW/IPPD

(3)CMMI-SE/SW/IPPD/SS

上述三种模型中(2)是在(1)的基础上提出的，而(3)是在(1)和(2)的基础上提出的。CMMI 最主要的组成元素是过程域(Process Area)，过程域描述了为改进软件过程应该做些什么而不是如何做。根据 SEI 的规划，CMMI 最终会完全取代 SW-CMM，所以我们看软件生产线和软件过程改进的关系主要是看它和 CMMI 的关系。

3.2 软件生产线和 CMMI

Clements 和 Northrop 在文[6]中曾经这样描述软件生产线和软件过程改进之间的关系：

"……如果软件工程是一群人在一起合作来解决问题，那么基于生产线的软件工程必然也需要合作。……实际上，一个没有很强的(软件)过程文化的组织想配置一条成功的软件生产线是一件非常危险的事情。"

从上面的引文中可以看出软件生产线和过程改进的紧密联系，而这种联系也在处于不同领域中的公司的软件生产线实践中被不断证实。图2显示了它们的这种互补关系[7]。

事实上，从许多公司的实践来看，如果一个机构在需求管理、项目计划、配置管理和需求开发等过程域达到了 CMMI 二级的话，将为实践软件生产线打下一个很好的基础。软件生

产线实践和 CMMI 的结合会使一个公司大大提高软件生产的效率和质量。

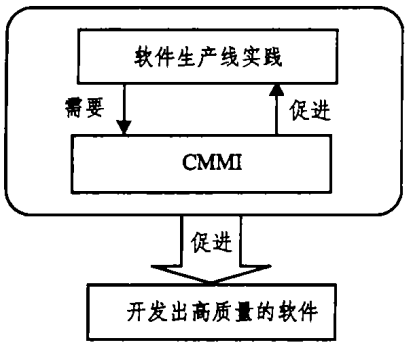

图2 软件生产线时间和 CMMI 的关系

4 软件生产线中的度量标准

度量是自然科学的一个传统，正像 Fred S. Roberts 在文[10]中描述的那样，一门科学的成熟程度跟这门科学中的度量成熟度成正比。从软件工程诞生之日起，如何度量其中的变量就吸引了很多人的兴趣[9]，特别是当人们在软件开发过程中采取了各种优化手段后，总是想知道新方法比老方法好了多少，这就是度量问题。同样的在软件生产线的实施过程中，学者和业界也提出了很多度量标准，以便于软件生产线的实施和管理。

度量是为了对某一活动中的变量进行量化，软件生产线中的度量需要关注以下三个方面的内容：

(1)核心资产的开发及其使用情况

(2)产品开发的性能特征

(3)整个软件生产线的性能

软件生产线中的度量标准可以分成三类：

(1)性能相关：主要是指基于生产线的软件开发相对于传统软件开发方式节约了多少天，质量提高了多少等，比如软件生产率，整体开发成本。

(2)一致性相关：主要是指基于生产线的软件开发对核心资产的复用程度等，比如体系结构的一致性，过程的一致性。

(3)效率相关：主要是指基于生产线的软件开发相对于传统软件开发方式提高的效率等，比如市场满意度、顾客满意度等。

需要指出的是对于每种度量，大都需要有对应的计算公式来进行计算从而获得量化结果。譬如对于整体软件开发成本来讲，它指的是利用软件生产线开发一种新的软件产品所需要的成本，一般来说需要包括以下两个部分的成本：①直接利用软件生产线开发该软件的成本；②开发核心资产的均摊成本。我们可以采用下面的公式来计算②中的成本[8]：

$$\text{均摊成本}=\frac{\text{年度资产开发成本}}{\text{年度开发软件数目}}$$

公式中的变量都是可量化的，从而可以求出最后的整体软件开发成本，进而比较采用软件生产线开发软件前后的成本大小来得出结论。

软件生产线乃至软件工程的度量一直都在研究当中，随着人们对于软件开发的进一步认识，会有越来越多的度量标准被提出来以供参考。

5 采用软件生产线开发软件的风险及其规避策略

基于生产线方式的软件生产会大幅度提高软件开发的效率和质量，但同时也存在风险。之所以把风险放在最后一节是想引起企业界的重视，从而可以在软件生产线实践中有效地规避风险。总体来讲采用软件生产线主要有以下两种风险[3]：

(1)生产线方式生产软件是一种全新的技术策略，存在技术风险；

(2)组织和管理风险，这些在实际操作中更易于被忽略。

有效规避这两类风险需要富有经验的技术人员和组织管理人员高效地配合，同时需要对软件开发中的各种问题进行及时的解决。

结束语 软件企业最终目的是为了高质量、高效率、低投入地开发软件。借鉴传统工业界流水线的技术，以生产线方式开发软件就是为了达到这一目的而提出的一种全新的软件开发方式。充分挖掘软件的共性组成生产线的核心资产以便在将来的软件开发中能够大粒度地复用，结合 CMMI 进行软件过程改进，利用已有的度量标准来管理软件生产，在引入生产线的过程中有效地规避风险，所有这些都大大提高了软件开发的效率和质量。

参 考 文 献

1 Bergey J, et al. Fourth DoD Product Line Practice Workshop Report: [Technical Report]. CMU/SEI-2001-TR-017

2 Brownsword L, Clements P. A Case Study in Successful Product Line Development, CMU/SEI-96-TR-016, ADA315802

3 A Framework for Software Product Line Practice Version 4.2, May 2004. URL: http://www.sei.cmu.edu/plp/frame_report/what.is.a.PL.htm

4 Paulk M, et al. The Capability Maturity Model: Guidelines for Improving the Software Process. Reading, MA: Addison-Wesley, 1995

5 Phillips D. CMMI Version 1.1: What Has Changed?, Cross talk 15, 2 Feb. 2002. 4～6

6 Clements P, Northrop L. Software Product Lines: Practices and Patterns. Reading, MA: Addison-Wesley, 2002

7 Jones L G, Soule A L. Software Process Improvement and Product Line Practice: CMMI and the Framework for Software Product Line Practice, Technical Note, CMU/SEI-2002-TN-012

8 Zubrow D, Chastek G. Measures for Software Product Lines, Technical Note, CMU/SEI-2003-TN-031

9 Ing. Horst Zuse, History of Software Measurement, URL: http://irb.cs.tu-berlin.de/～zuse/metrics/3-his-t.html

10 Roberts, Fred S. Measurement Theory with Applications to Decisionmaking, Utility, and the Social Sciences. Encyclopedia of Mathematics and its Applications, Addison Wesley Publishing Company, 1979