Concurrent Development Processes of Software

软件并行开发过程

李彤 孔兵 金钊 王黎霞 著

科学出版社
www.sciencep.com

软件并行开发过程

李 彤 孔 兵 金 钊 王黎霞 著

云南省自然科学基金项目(项目编号:98F023M,98F005G,2001F0006M)
云南省中青年学术和技术带头人培养基金项目(项目编号:1998-37)

科学出版社
北 京

内 容 简 介

软件过程是软件产品开发成功与否的关键性因素。本书借鉴了制造业并行工程的思想，将其引入到计算机软件工程中，以软件开发过程中的并行性为研究对象，通过尽量使软件开发并行进行来达到提高软件生产率的目的；通过优化改善软件开发过程来达到提高软件质量的目的。

本书对可并行的软件过程及其模型、并行成分划分、并行性挖掘与延拓、并行控制、测试与集成技术、计算机辅助软件并行开发、软件开发过程的自动化等方面进行了研究，系统地讨论了相关的概念、原理、方法、技术和工具，讨论了软件并行开发对 CMMI 的支持。

本书可以作为计算机专业研究生和高年级本科生的教材和教学参考书，也可供从事软件工程的科技人员使用和参考。

图书在版编目(CIP)数据

软件并行开发过程/李彤等著. —北京：科学出版社，2003
ISBN 7-03-012007-8

Ⅰ. 软… Ⅱ. 李… Ⅲ. 软件开发-高等学校-教材 Ⅳ. TP311.52

中国版本图书馆 CIP 数据核字(2003)第 066602 号

策划编辑：鞠丽娜/责任编辑：韩 洁
责任印制：吕春珉/封面设计：王 浩

科学出版社 出版
北京东黄城根北街16号
邮政编码：100717
http://www.sciencep.com
新蕾印刷厂 印刷
科学出版社发行 各地新华书店经销
*
2003 年 8 月第 一 版 开本：B5(720×1000)
2003 年 8 月第一次印刷 印张：12 1/4
印数：1—4 000 字数：235 000
定价：22.00 元
(如有印装质量问题，我社负责调换(环伟))

序

《软件并行开发过程》是一部软件工程领域的学术专著。作者李彤教授及其他几位学者在多项基金的支持下，从1995年起，一直从事软件并行开发的研究工作，本书是这些丰富的研究成果的总结和集成。本书借鉴了制造业并行工程的思想，并将其引入到计算机软件工程中，以软件开发过程中的并行性为研究对象，通过尽量使软件开发并行进行来达到提高软件生产率的目的；通过优化改善软件开发过程来达到提高软件质量的目的。

本人认为本书所讨论的内容在以下几个方面独具特色：

1. 软件过程及软件过程模型

软件过程是软件生存周期中所涉及的一系列相关过程，是软件过程模型的动态执行过程。软件过程模型是软件过程的静态描述。软件并行开发涉及到软件生存周期中的各种软件过程，通过过程模型描述各种不同类型与粒度的并行性。在过程模型执行时(表现为各种软件过程)就体现出相应的并行行为，以达到软件并行开发的目标。

在本书中，分析了存在于软件生存周期中的不同层次的并行性，提出了基于Petri网的支持软件并行开发的软件过程模型SDDM，并给出了一个支持软件并行开发的形式化过程建模语言SDDML和基于SDDML的过程建模方法。SDDML具有面向对象的特征，可表示不同抽象级的过程模型，支持逐步求精的过程建模方法，为软件并行开发中软件过程的控制、分析、评估和优化奠定了基础。

2. 并行成分划分

将一个完整的软件开发过程划分为若干个可以并行进行的成分是软件并行开发的基础。本书讨论了并行开发的特征，以降低各模块之间的耦合度为目的，并结合面向对象的开发策略，提出了基于Petri网系统模型的动态划分方法和基于脚本的系统划分方法两种并行模块划分方法。

3. 并行性挖掘和延拓

本书从软件过程的角度来探讨并行性的挖掘，通过活动间相关性分析，寻找软件过程中可并行化的因素，挖掘出可并行进行的活动，构造出Petri网表示的并行化的软件过程模型。

另外，通过延拓并行性的方法，可以使并行性由局部延拓到全局，从而提高软件过程的并行度，使软件开发活动在全局并行进行，以达到进一步提高软件生产率的目的。

4. 并行控制

软件并行开发虽然以缩短软件开发周期为目标，但却使系统开发的复杂度大

大增加，人员间接口也更加复杂，从而使得并行控制这个问题更加突出。

为了解决相关问题，本书中提出了两个并行控制模型：即基于 Petri 网的用于并行过程控制的模型 CCM(concurrent control model)，描述开发过程中的对象的 C-P/T(control-place/transition)系统。

5. 集成技术

软件并行开发由于其并行性强、交互频繁，有其不同于传统集成技术的特点和要求。本书提出了一种新的集成测试策略——基于 Petri 网系统模型的集成测试，利用 Petri 网构造的系统模型，分事件测试和网际测试两个层次测试面向对象软件的质量。

6. 计算机辅助软件并行开发

本书介绍了作者设计的一个支持软件并行开发的 CASE 系统，它支持与软件并行开发有关的各种模型的建立；提供可视化的并行控制手段，用户可交互式地使用该系统实施控制，保证开发目标的正确性。

7. 软件开发过程的自动化

本书基于公理语义的形式化需求规约语言，讨论了形式化功能分解、代码生成技术，讨论了实现软件开发过程自动化的途径。

8. 软件并行开发对 CMMI 的支持

软件并行开发技术中所讨论的软件模型(如 SDDM、CMM 等)可以支持 CMMI中的一些关键过程域的实施和改进。本书对软件过程高度重视，其方法和技术值得实施 CMMI 的组织借鉴。

我们所熟知的软件工程包括软件开发技术和软件项目管理。前者涉及软件开发方法学、软件工具和软件工程环境等内容；后者涉及软件质量、项目估算、进度控制、人员组织、配置管理、项目计划等内容。这些方面国内外的学术界、工程界都做了大量的学术研究和工程实现，展现在大家面前的学术专著、论文和有关文献资料如烟波浩淼的大海，而相应的产品也琳琅满目，让人目不暇接。据笔者陋见，本书所专注的是一个比较新的研究领域，相关的研究工作尚不多见，应该说作者的工作是有创新性和开拓性的，具有一定的理论意义，其方法和技术值得我辈学习和参考。

本书系统地讨论了软件并行开发的概念、原理、方法、技术和工具，以及软件并行开发对 CMMI 的支持。全书行文严谨流畅，结合内容给出了一些例子，便于读者阅读，具有较高的学术价值。当然软件并行开发作为一个新的研究领域，书中的一些观点也当请所有读者评判斧正，许多工作当有待进一步完善。

据本人所知，作者从科研实践到草成书稿，五易其稿，历时三载，倾注了大量心血。相信本书的出版有助于推动这一领域研究和教学工作的进一步深入和发展。

承蒙李彤教授的错爱，代为作序，不胜惶恐，写上几句，权作“抛砖”。

孙玉芳

2003 年 5 月于北京

前　言

软件过程是软件产品开发成功与否的关键性因素。软件过程能力的成熟度如何已成为衡量一个软件企业整体有效性的关键性尺度。本书以软件开发过程中的并行性为研究对象，期望通过并行地开发软件来达到提高软件生产率的目的；通过优化改善软件过程来达到提高软件质量的目的。

软件工程界长期存在软件生产率低下、软件质量欠佳两大难题。究其原因，同软件过程有很大的关系。本书所讨论的软件并行开发借鉴了制造业并行工程的思想，让软件开发过程中具备并行条件的各种粒度不同的成分（包括过程、阶段、活动、任务等）并行进行，以达到加快软件开发速度的目的。正如硬件运算能力的显著提高是由串行转向并行所引发的，我们认为软件并行开发是提高软件生产率最具有潜力的途径之一。软件并行开发要求在早期阶段考虑下游的阶段与过程，支持软件需求分析、设计与下游各阶段并行交叉进行，强调各并行成分的相对独立，支持各并行成分之间的经常交互，从而改善软件过程。尽早考虑下游因素，有利于减少反复次数、提高上游结果的质量，达到优化分析和设计的目的。因此，软件并行开发对于提高软件质量也是有希望的突破口。

笔者日前曾到印度考察，参观了通过 CMM 五级认证的软件公司，亲眼看到印度软件企业对软件过程的高度重视。严格定义的软件过程模型及规范执行的软件过程令人惊叹不已，这也正是印度软件企业的特色所在，其经验值得我们借鉴。作为软件工程的研究者和实践者，我们切身感受到我国学术界和产业界均不同程度地存在对软件过程重要性认识不够的问题。我国软件产业要迅速发展，必须高度重视软件过程在软件工程中的地位和作用，必须加强对软件过程的研究和应用。

我们在云南省自然科学基金项目“软件并行开发技术及其 CASE 系统研究”、“基于公理语义的面向对象并行软件自动化研究”、“基于构件的快速软件并行开发技术”和云南省中青年学术和技术带头人培养基金的支持下，对可并行的软件过程及其模型、并行成分划分、并行性挖掘与延拓、并行控制、测试与集成技术、计算机辅助软件并行开发、软件开发过程的自动化、软件并行开发对 CMMI 的支持等方面进行了研究，本书就是上述研究工作的总结。除本书作者外，参加研究工作的还有柳青、郝林、梁彬、和伟民、刘剑、赵辉、卫玲、刘勇、倪志凌、张屿等同一个课题组的同仁。回顾八年研究工作和课题组一起度过的日日夜夜，取得突破后的欢欣鼓舞，遇到困难时的刻苦攻关、相互鼓励，甘于寂寞对计算机科学的不懈追求，至今历历在目。笔者愚钝，但对计算机科学的深情挚爱与执着追求却矢志不渝，斗胆携其他几位作者将研究心得撰写成书，该书今日得以出版，感慨不已。成果来之不易，课

题组诸君的努力、信任与理解，笔者深感欣慰，并将永远铭记在心。

本书承蒙中国科学院软件研究所孙玉芳研究员认真审阅，并欣然作序。孙老师严谨的学风、渊博的学识、深刻睿智的学术思想给了笔者极大的启发、教导与鞭策；他在本书几易其稿过程中，对书稿内容提出了大量宝贵修改意见和建议，从而对本书质量的提高起到了重要作用。本书还得到科学出版社鞠丽娜副编审的关心、鼓励和精心审校。鞠老师严谨认真的工作作风给笔者留下了深刻印象。在本书付梓之际，谨向他们表示衷心的感谢。

最后，还要衷心感谢曾经对本书的研究工作给予过关心、支持与帮助的教授们，他们是：北京大学计算机系杨芙清院士，中国科学院软件研究所仲萃豪研究员，中国银行纽约分行丁茂顺研究员，汕头大学计算机系龚世生教授，云南大学计算机系刘惟一教授，云南师范大学计算机系林毓材教授，昆明理工大学计算机重点实验室丁志强教授。

由于水平和学识所限，书中肯定存在不少疏漏、欠妥与谬误之处，真诚希望读者、专家和同行不吝赐教。

李　彤

2003 年夏于云南大学英华园

目　录

第1章 绪 论

1.1 软件并行开发的提出

1.1.1 传统软件工程面临的问题

20世纪60年代后期，为克服“软件危机”诞生了软件工程学，为软件的开发与维护注入了生机和活力。传统的软件工程学通常认为软件开发是一个串行的过程，从时间的延续上看需要顺序经历一些阶段才能完成软件系统的开发。正是基于这样的认识，软件生存周期模型(software life cycle model)把软件开发划分为若干分离的阶段，认为软件开发可以采用人类解决问题时最普遍采用的策略：分而治之、各个击破，并且认为能够像数学推导那样完美地保证前一步的正确结果能够导致后一步的正确结果，最终得到期望的结果。

随着时间的推移，以这种思想为主导的软件工程领域依然存在许多人们认为可以解决但却没有解决的问题，其中生产率低下和质量欠佳最为突出。如何提高软件生产率和软件质量是摆在软件工程界面前的重大课题。

1.1.2 来自制造业的启示

在制造业中，传统的产品开发模式是沿用“串行”、“顺序”和“试凑”的方法，开发过程是按顺序一步步完成的，即顺序地进行市场分析、产品设计、工艺设计、采购、加工和测试。每个部门分工明确，完成一项特定的工作，这种模式在制造业中称为串行工程[1,2]，如图1.1所示。

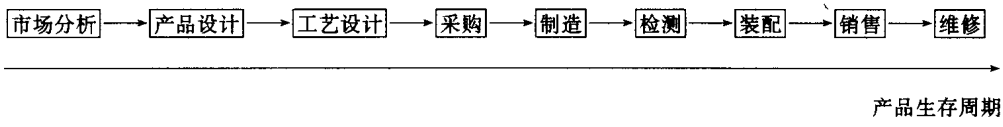

图1.1 制造业串行工程模式

在串行工程中，产品开发过程只是一个静态的、顺序的和互相分离的流程。除了在设计后期和制造阶段发现问题能提出工程修改外，设计的上下游工序之间不存在经常性的信息交换，各阶段间仿佛存在一堵墙，每一个阶段结束时完成部门将结果抛过墙交给下一阶段，出现问题则抛回上一阶段。

随着商品市场竞争的加剧，顾客对产品质量、成本和种类的要求越来越高，产品的生存周期要求越来越短。企业为了赢得市场，就不得不解决加速新产品开发、提高产品质量、降低成本和提供优质服务等一系列问题。此时串行工程的弊端日趋严重，因为这里追求的核心问题是时间和效率，这与串行工程模式格格不入。在此情况下，并行工程(concurrent engineering)应运而生。

在制造业领域中，早在第二次世界大战前，美国福特汽车公司就采用了跨部门的集成化多专业小组来进行产品设计；洛克希德航空制造公司也采用了被称为“先进项目开发方式”的工作流程进行产品设计，这些都是最早的并行工程雏形。

制造业并行工程真正由思想变为工程还是到了 20 世纪 80 年代。1982 年，美国国防部高级研究项目局开始研究如何在产品设计过程中提高各活动之间的并行度。1986 年夏天，美国防御分析研究所发表了著名的 R-338 报告，正式提出了并行工程的概念，从而标志着并行工程从思想变为工程科学[1]。

R-338 报告对并行工程定义为[1]：并行工程是集成地、并行地设计产品及其相关的各种过程(包括制造过程和支持过程)的系统化方法。这种方法要求产品开发

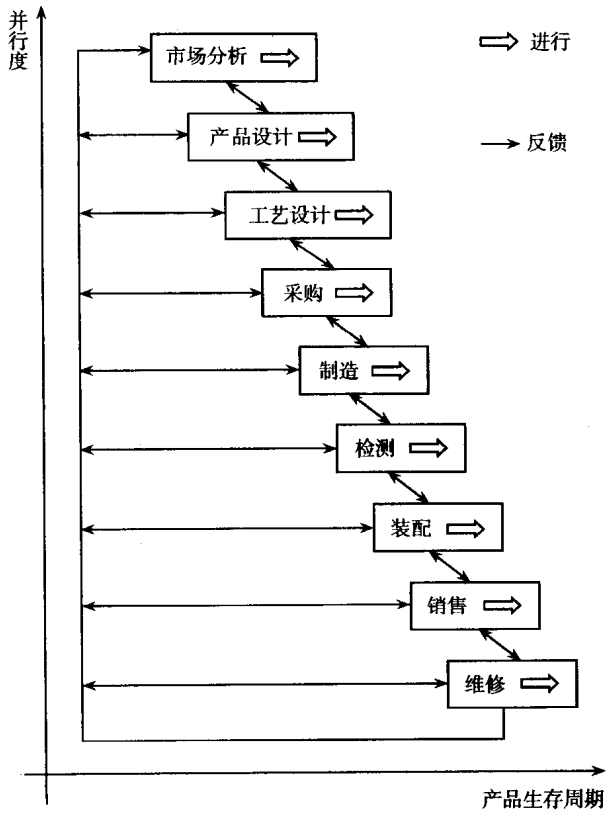

图 1.2　并行工程模式

人员从设计一开始就考虑产品整个生存周期中从概念形成到产品报废处理的所有因素，包括质量、成本、进度、计划和用户的要求。

并行工程改变传统的串行进行工程实施的模式，组织多个开发小组并行协同工作，对产品生存周期中的各方面、各阶段进行同时考虑和并行交叉设计，把串行产品开发流程转变成集成的、并行的产品开发过程，使工程尽可能并行进行（实际上是串并行同时存在），达到缩短产品开发周期、提高产品质量和降低成本的目的。并行工程模式如图 1.2 所示[1,2]。

并行工程目前已成为制造业研究与实践的热点领域。在国外已成功地应用于航空、航天、电子、汽车等领域[1~7]。福特汽车公司 20 世纪 70 年代后期财政处于极大的困难，此时，该公司重新采用并行工程方法，生产出了优良的汽车。从 1982～1987 年，福特 Escort 型汽车创世界汽车销售量之首[1]。波音公司 1991 年在开发波音 777 飞机时，组建了 200 多个开发小组，完全采用计算机辅助设计（无纸化设计），采用并行工程进行产品开发。过去大型客机的设计周期多则十几年，少则七八年，而波音 777 的开发过程实现了五年内从设计到一次试飞成功的目标。该公司的另外三个新产品开发成本降低了 16%（有的部门甚至降低了 50%），低于投标成本[1]。洛克希德公司采用并行工程方法缩短新型导弹的开发周期，于 1992 年接受了“战区高空领域防御”新型导弹的开发计划，开发经费为 6.88 亿美元，过去通常需要 5 年时间，结果通过改进产品开发流程，实现信息集成和共享，组织综合的产品开发队伍，为群组工作提供网络通信环境，支持异地的电子评审，利用产品数据管理系统辅助并行设计等并行工程方法与技术，在 24 个月内完成了开发任务[1]。日本丰田汽车公司采用并行工程技术，在 20 世纪 80 年代以后的美、日汽车贸易战中居于领先地位（包括前面提到的福特公司）。在丰田汽车公司里，各个阶段几乎都可以同步或交叉进行。如汽车车身在进行图纸设计时，制造部门就可以开始模具制造，因为模具师知道了新轿车的大致尺寸和覆盖件的概略数，所以可以提前订购模块，并在模块上进行粗加工，当覆盖件的最后一张设计图一发出，模具立即可以转到精加工去，这样显然大大加快了新产品开发的速度，但这需要小组成员间良好的信息沟通。

我国并行工程的研究始于 20 世纪 90 年代初，当时，国家科委在 863 计划中设立了一些课题支持并行工程的研究[2~4]。在 863 计划的推动下，目前我国已有少数几家企业开始实施并行工程，并取得了新产品开发样机周期缩短、产品综合技术经济指标好、符合市场需求、经济效益好等效益。

国内外大量事实表明，实施并行工程可使工程项目取得明显的效益。综合并行工程的定义与实践，可以看到制造业并行工程具有以下重要特征：

1）并行性：各种活动并行交叉进行。既包括各个部件开发的并行交叉，又包括各个阶段的并行交叉。

2）流程改善：通过改善与优化工作流程，提高产品质量。

3）尽早开始工作：在信息不充分的情况下开始工作，因此要有很强的应变能力。

4）集成化：强调全面优化，追求产品整体的竞争力和各小组的密切有机合作。

5）强有力的支撑环境：必须有良好的支撑多小组并行协同工作的网络与计算机平台。

6）强有力的管理：强调强有力的管理，保证各种并行活动协调统一地进行。

1.1.3　来自其他领域的启示

除来自制造业的启示外，以下几项来自其他领域的提高效率和质量方面的事例给了我们进一步的启示：

1. 并行处理(parallel processing)

近年来，高性能并行计算机的发展取得了巨大的进展，每秒十几万亿次计算能力的高端并行机已相继研制成功，使得以前许多无法解决的计算问题已成为可能。其基本手段是通过多处理机并行来实现比传统单处理机计算能力强得多的高性能计算。目前，并行处理已成为了提高计算机系统性能的最重要的手段。即使在单处理机环境下，多个进程并发(concurrent)执行也是公认的提高系统性能的有效手段。

2. 多个程序员同时为同一个系统编码

这是经常见到的一类现象，是加快编码速度的基本手段，并得到了普遍的运用。

3. 现实生活中的各类非自觉的、偶然的并行行为

并行是人类社会中广泛而普遍存在的一类现象，其本意是指两个或多个事件同时发生或在同一时间间隔内发生。人们都有这样的常识，多个人同时做一件事情、一个人分时轮流地在一段时间内做多件事情比一件一件串行地做这些事情效率要高得多，速度要快得多。这类非自觉的、偶然的同时行为就蕴含着朴素的并行思想。

以上事例有两个共同的特点：

1）通过资源(人或物)的冗余来实现同时工作。

2）可以极大地提高效率，加快工作的进度。

对上述事例进行总结，我们发现，它们均是通过付出空间(资源)代价来换取时间，于是我们得到启发，能否并行地开发软件呢?即能否尽量使软件开发并行进行，来达到提高软件生产率的目的，并进而通过改善软件过程来达到提高软件质量的目的呢?

1.2 软件并行开发研究的现状

事实上,并行地开发软件已逐渐引起了软件工程界的关注。Humphrey 和 Kellner 早在 1989 年就指出软件开发中任何一个阶段的活动都存在并行性[8]。Kellner 曾经使用状态图(statechart)来表达同特定事件相联系的诸活动间的并行关系,这些特定事件是指在后期开发时需求发生改变所引发的事件,但未能提出怎样捕捉存在于软件开发与管理活动中的并行性的手段[9]。

Davis 和 Sitaram 使用 Kellner 所提出的状态图作为工具,在文献[10]中提出了一种能有效地表达存在于软件活动中并行性的软件开发过程模型。该模型充分展现了软件开发过程中存在的并行性。通过挖掘传统串行开发过程中的并行性,建立了基于软件开发过程中不同角色(用户、软件开发人员、项目经理、复审人员)所并行进行的不同活动的并行模型。作者指出尽管软件开发模型经常被认为是由一系列分离的阶段组成,但许多开发者都发现了这些阶段是不可避免地相互纠缠(inevitable intertwining)在一起的,并且经常返回早期的阶段。作者认为进行初始化(没有指明怎样初始化)后软件开发的所有的活动都是可以并行开展的。软件开发过程中存在两方面的并行活动,并行不仅存在于开发人员开展的活动之间,也存在于用户、开发人员、项目经理以及复审人员开展的不同活动之间,还建立了由用户需求、软件开发、项目管理、软件计划四个并行活动所组成的并行开发模型。他们给出了一个极好的说明软件开发是一个并行过程的事实:

当问软件项目的经理"你们的项目进展如何?"

小软件项目的经理回答是:"我们正在项目的编码阶段。"

而大规模复杂软件项目的经理会如此说:"我们正处于项目的编码阶段。目前的工作主要集中于产品的版本 1。版本 1 已经完成 25%的编码工作,已完成编码中一半的部件通过了单元测试。集成测试计划刚刚完成并且得到了项目经理的批准。一个开发小组已经开始集成通过单元测试的部件了。版本 1 准备在 1993 年的 7 月发布。我们同时工作在产品的版本 2,需求分析刚刚结束,正在修改版本 1 的概念设计以适合现在刚批准的版本 2 的需求分析,预计在 1993 年的 11 月发布版本 2。在过去的六个月里,来自用户新的需要已经促使我们又在修改需求分析了。版本 3 的软件需求规约安排在下周复审,版本 3 大约在 1994 年 4 月发布。"

Davis 的模型尽管给我们一种软件开发能够在大规模的层次上并行进行的印象,但并行的内容则过于粗略,没有明显地对并行开发的成分做出划分。

Aoyama 从硬件流水线组装技术受到启发,认为软件系统也是可以通过多个并行开发的系统部件组装而成[11,12]。他在文献[11]中,在瀑布模型的基础上构建了一个并行开发过程模型,日本富士通公司将其用于开发某大型通信系统,其目的是缩短开发周期,提高软件生产率。他指出将来软件开发将是分布式的并行开发

(distribute-concurrent development),认为软件开发的整个过程是可以并行的,并行是在粗粒度上进行的。作者认为一个系统由多个广义模块(enhancement)组成,而一个广义模块由多个模块组成,可以并行地开发多个广义模块,然后进行集成,完成一次发布(release)。经过多次发布完成系统的开发(怎样划分,集成没有提到),其本质是一种并行的进化式开发模型。该文主要表明了并行开发需要的管理技术以及怎样组织实现这些技术,但怎样划分系统的组成部件,在开发过程的哪一个阶段划分,怎样保持并行开发部件之间的交互以及最后的集成这一些实现并行开发的关键问题都没有阐明。

Go 和 Shiratori 在文献[13]中提出了这样一种并行开发:把总的需求规约分解为几个子需求规约模块,在不同的地点按照子规约并行地开发软件系统的部件(对应一个子规约模块),然后集成系统部件成为一个完整的系统。每一个部件的开发是一个完整的软件开发过程。其本质认为需求可以分解,分解得到的子需求可以并行开发,最后集成为系统。可以说这是一种比较理想的并行开发模式了,贯穿整个开发过程并且在最大的规模(粒度)上实现并行开发。

Norman、Raccon 和 Davis 在文献[14～16]中均指出:尽管开发过程中的活动是随时间按串行的方式开展执行的,但活动之间显著的重叠(overlap)和并行(concurrent and parallel)是完全可能的,其含义是不仅前一个活动的结尾和后一个活动的开始可以重叠,还包括前一个活动和后一个活动的完全重叠(只要活动的内容是不相同的),比如在做系统的人机界面的需求分析时同时开展系统功能的总体设计。在特定的项目开发中,根据实际的情况,开发过程中的活动是可以裁剪的(tailorable),根据实际需要(需求变化)回溯(revisiting)和循环(looping)到特定的活动是可以的。

文献[15]在论述 Coad 的面向对象方法时,曾明确提出开发活动的组织是自由的,它们可以按适当的顺序进行,在形势允许的条件下开发活动是可以并行的。同时他还指出组成系统的不同对象模型部件是可以并行开发的。Booch 则说明他的开发过程结构是模糊的(vagueness),目的是使用户能够根据开发形势需要做出裁剪和修改。在条件允许的情况下,用户可以根据需要安排开发活动的顺序[17]。被认为是面向对象方法集大成者的 UML[18](unified modeling language)和 UP[19](unified process)都继承了他们的观点,认为:开发活动是可以并行的;不同部件的开发是可以并行的。

很多主流的模型和方法从开发模型和开发方法的层次指出了在软件开发中开发活动是可以并行的。他们在强调开发过程是串行的同时都无法回避开发过程中存在并行性这样一个事实,因而都指明了软件开发是可以并行的。但并行应该怎样自觉地发生、进行以及最终实现目标系统都没有涉及到。如果说这些主流的模型和方法涉及到软件并行开发仅只是为了保证它们的完备性的话,那么对于一些研究者来说他们不只是满足于挖掘软件过程中的并行性,而是主动提出一些模型和方