

计算机软件工程环境 和软件工具

董士海 编著

科学出版社

1988

内 容 简 介

目 录

前言

第一编 软件工程环境

第一章 软件工程环境综述.....	1
第一节 什么是软件工程环境	1
第二节 软件工程环境的类型、构成成分和特性	4
第三节 软件工程环境的发展前景	10
第二章 工具箱——一种常见及灵活的软件工程环境...	14
第一节 工具箱类型的软件环境	14
第二节 UNIX 程序设计环境.....	17
第三章 面向语言的整体化程序设计环境.....	41
第一节 整体化程序设计环境的特点	41
第二节 INTERLISP 程序设计环境	43
第三节 Ada 程序设计支撑环境——APSE	54
第四节 GANDALF 系统.....	62
第四章 支持整个软件生存周期的软件工程环境.....	70
第一节 软件生存周期与方法论	70
第二节 C 软件工程环境	74
第三节 SOFTING	78
第四节 ARGUS	82
第五节 PRISM.....	86
第五章 交互式软件工程环境.....	90
第一节 交互系统中用户工程原理	90
第二节 个人工作站一例: SMALLTALK-80	96
第三节 用户软件工程 USE 系统	102

第六章 支持形式化方法的软件工程环境.....	108
第一节 引言	108
第二节 以时态逻辑为基础的 XYZ 系统	109
第三节 支持 VDM 方法的工具系统	112
第七章 软件工程环境的数据库支持.....	119
第一节 软件信息库及它在软件工程中的地位	119
第二节 PSL 及 PSA	121
第三节 PMDB 与其它	132
第四节 一个软件信息库 SIB 的设计.....	141
第八章 智能化软件工程环境.....	147
第一节 软件工程环境的重要发展方向——智能化	147
第二节 智能软件开发工具或环境的一些实例	151
 第二编 软件工具 	
第九章 软件工具综述.....	157
第一节 软件工具的历史与现状	157
第二节 软件工具与软件设计方法论	162
第三节 软件工具的分类	166
第十章 编辑程序.....	176
第一节 编辑程序综述	176
第二节 编辑程序的构成及实现	180
第三节 结构编辑程序与语法制导编辑程序	198
第十一章 软件开发与图形工具.....	212
第一节 软件的图形表示	212
第二节 图形化软件工具模型和组成	215
第三节 图形化软件工具的举例	220
第十二章 调试工具和测试工具.....	232
第一节 调试工具	232
第二节 测试工具	243
第三节 软件的质量保证	254

第十三章	软件维护与版本管理工具	258
第一节	软件的维护	258
第二节	软件维护工具	260
第三节	软件版本管理工具	266
第十四章	软件移植工具	277
第一节	软件的可移植性	277
第二节	中间语言抽象机	280
第三节	交叉编译及代码生成程序例子	285
第十五章	事务处理的软件工具	290
第一节	第四代语言及工具	290
第二节	电子表格软件和组合软件	297
第三节	应用程序生成器	305
第四节	操作环境	312
总参考书目		317
各章参考文献		317

第一编 软件工程环境

第一章 软件工程环境综述

第一节 什么是软件工程环境

一、软件工程

当前计算机已广泛应用于工业、商业、军事,及社会生活的各个领域。计算机的硬件由于大规模集成电路的飞速发展而使其成本大大降低,可靠性大大提高。计算机软件由于其本身的特点——“智力密集型”的生产,而使其成本十分昂贵,质量也因复杂性增高而明显下降。为解决上述“软件危机”问题,从60年代末到70年代中,计算机科学家从各个角度提出了解决问题的方法。“软件工程”的概念就是这期间提出的,它要求人们采用“工程”的方法来开发、维护和管理软件。

软件工程一般要研究以下几方面的内容:

1. 将软件的生产和维护的全过程(也称软件生存周期)划分成阶段,根据各阶段的不同性质和特点,对软件生产的产品、人员和质量进行管理。这些阶段一般为:计划和定义阶段,需求分析阶段,设计阶段(有时再细分为概要设计和细节设计),编码阶段(过去人们常把程序设计当作软件生产的全部),测试阶段及维护阶段等。

2. 对软件生存周期各阶段所采用的规格说明、分析设计方法和形式化描述技术等进行研究。

3. 提供各种提高软件生产率、改进软件质量的实用程

序——即各种软件工具。

4. 研究保证软件质量的方法和措施。

5. 研究软件项目的管理、人员组织、成本估算及各种度量方法。

二、软件工程环境

(一) 定义

早期的软件工具只是一个个孤立的实用程序，多数也只是支持软件的编程和调试。随着对软件开发全过程的研究和各种设计方法的提出，许多研究单位、计算机公司陆续研制了用来辅助软件开发的大型程序系统。这是一组软件工具的有机集合，这些软件工具是按照一定的软件开发方法或遵循一定的软件生产和维护模型组织起来、为一定的领域所使用的。这样的计算机辅助软件开发的程序系统，被称为“软件工程环境”。在 1985 年 8 月伦敦举行的第八届国际软件工程会议上，一个由 IEEE（美国电气及电子工程师协会）和 ACM 的 SIGSOFT（美国计算机协会的软件工程专业委员会）支持的国际工作小组提出了对 Software Process, Model, Method, Methodology, Tools, Development Environment 等的定义^[1]。其中关于“软件开发环境”一项是这样定义的：

软件开发环境是相关的一组软件工具集合，它支持一定的软件开发方法或按照一定的软件开发模型组织而成。

在美国国防部的 STARS（Software Technology for Adaptable Reliable Systems）计划中，对“软件工程环境”一词是这样定义的：

一个软件工程环境（SEE）是一组方法、过程及计算机程序（计算机化的工具）的整体化构件，它支持从需求定义、程序生成直到维护的整个软件生存周期。

虽然上面这些定义不完全相同，但它们在下面几点上是一致的：

1. 软件工程环境是一组软件工具的集合；
2. 这些工具是按照一定方法或模型组织起来的；
3. 这些工具支撑整个软件生存周期的各个阶段或部分阶段。

有关“软件工程环境”(software engineering environment)一词的别名是很多的，如：

software development environment		软件开发环境
software support environment		软件支撑环境
programming support environment		程序设计支撑环境
project support environment		项目支撑环境
automated development environment	...	自动开发环境
integrated programming environment	...	整体化程序设计环境
toolbox		工具箱
toolkit		工具箱

这些名称多数含义是一样的，有些则大同小异。如“程序设计环境”也是一组软件工具的有机集合，而这些工具主要是支持软件的编程和调试等工作。在本书中我们一般均称为“软件工程环境”，或简称“软件环境”。

(二) 软件工程环境的发展

大量系统软件和应用软件的开发促进了“软件工程”这门学科的发展，许多新的开发模型、设计方法和技术不断出现，许多高质量的“软件工程环境”相继产生。而“软件工程环境”的使用又大大提高了软件的生产率，降低了软件的成本，改善了软件的质量。但是在 70 年代，“软件工程环境”的发展还很缓慢，其主要原因是：

1. 开发一个“软件环境”周期长，成本高，并不是所有单位都愿下决心去冒险开发。

2. 任务一紧，就没有足够时间研制“软件环境”，缺乏商业刺激。

3. 当时的硬件环境不适合一个统一的软件环境。

近年来由于提高软件生产率、降低软件成本的迫切性，进一步促进了“软件环境”的发展。一些典型的“软件环境”的研制成功也给人以启示。尤其是硬件的性能/价格比的飞速提高，为“软件环境”的发展创造了物质条件，如大规模集成电路的性能/价格比提高，大容量磁盘存储器的使用，高分辨率的光栅扫描图形终端的出现等等。因而在国际上对各种软件开发方法、工具和环境的研究十分活跃，人们从各种途径加以探索，一些商品化的“软件环境”也已推出。我国近年来微型计算机大量推广，各行各业计算机广泛应用，正处于一个新的发展阶段。我国软件的开发与应用也应从低水平的重复，有步骤地向高水平发展。因而认真分析各种“软件环境”的技术特点和应用领域，了解新的研究动向和掌握这方面的技术是十分重要的。

第二节 软件工程环境的类型、构成成分和特性

一、类型

软件工程环境已经在国外的许多公司和研究单位研制使用。这些软件工程环境都是针对一定的需要，按照它们所支持的软件开发方法进行研制的。至今，尚无一种公认的分类方法。对于软件工具，美国国家标准局在 1982 年 3 月的第 500-88 号专门报告上，对他们数据库中 362 个软件工具进行了分类（详见第九章）。

为进一步分析各种软件环境的特点，下面我们从不同方面来讨论软件工程环境的类型。

(一) 按软件生产的不同方面、不同阶段分类

1. 软件开发环境。这类环境目前数量最多，它支持软件的需求分析、设计、编码或测试等。这类环境中有的主要支持编程，也称为程序设计环境。如 UNIX 程序设计环境，Ada 程序设计支撑环境 (APSE) 等。有的专门支持软件的需求分析，如密执安大学的 PSL/PSA 及 SDS 等。

2. 软件项目管理环境。这类环境支持软件的项目管理，诸如计划、人员组织、成本估算等，SDEM 属于这一类。

3. 软件质量保证的环境。这类软件环境包括程序正确性证明、验证及软件测试的功能。早期的 ASSET 和以后的 GYPSY, AFFIRM, SPECIAL 可属这一类。

4. 软件维护环境。

(二) 按是否与软件开发方法有关分类

1. 不依赖于软件开发方法的环境。工具箱是这类环境的重要一种。它并不依赖于某种软件开发方法，而是将最常用的软件工具组成一个软件包，供用户使用。UNIX 程序设计环境就是属于这类工具箱，它在 UNIX 操作系统之上进行不断扩充。许多大、中型计算机的软件开发系统也属于工具箱这一类。

面向高级语言的软件环境也属于不依赖于软件开发方法的这一类。如用于开发 LISP 程序的程序设计环境 INTERLISP, 用于开发 Ada 的程序设计支撑环境 APSE, 美国卡内奇-梅隆大学的 Gandalf 项目也是一个 APSE 系统。这类环境比工具箱具有更高的“整体性”。

2. 专门支持某软件开发方法的软件环境。近 15 年来,对软件开发方法已进行了大量的研究。为使软件开发建立在更加严谨的科学基础之上,许多计算机科学家在形式方法上作了大量研究。维也纳开发方法(VDM)是由欧洲计算机科学家 1973 年左右提出的,至今已经被应用到程序设计语言、数据库、操作系统、办公室自动化系统及其它应用系统的领域中。中国科学院软件研究所研究的 XYZ 系统以表示能力非常强、语义非常简单、便于验证和推理的 XYZ/E 时序逻辑语言为基础,开发了适应于多种程序设计方式的语言和工具,支持形式方法的软件开发环境往往包括语言处理器和正确性验证器等。斯坦福大学的 AFFIRM 和爱丁堡大学的 LCF 都属于这一类软件环境。

在软件开发方法中还有一类偏重于技术的非形式方法。这类方法在工业界、商业及一些应用部门使用较多。其中有 Yourdon 结构化分析方法(SA),结构化设计方法(SD),SADT 及有限状态机等方法。例如软件环境 AIDES 支持结构化设计(SD)方法,波音公司的 ARGUS 支持结构化分析(SA)方法。

(三) 按工作方式分类

1. 交互式软件环境。当前多数软件工程环境都有一定交互功能,如允许用户从终端输入信息,并输出反馈信息等。但近年来已出现一批以高分辨率图形显示终端为中心的、被称为工作站(workstation)的交互式程序设计环境,如 XEROX 公司的 INTERLISP-D、SMALLTALK-80 等。这种软件工程环境尤其适合探索性的软件开发,它的产生已经对软件工程、人工智能及程序设计语言等计算机科学有关领域产生重大影响。它所采用的一系列交互技术、窗口及菜单管理和鼠

标器 (mouse) 的使用给计算机用户界面的设计带来了极大的影响。

2. 批处理软件开发环境。这是相对于交互处理而言的。早期的软件开发环境主要进行代码的转换(如编译)、代码的静态分析和动态分析,因而多数是批处理软件环境。

二、构成成分

软件工程环境应该包含哪些技术成分呢?这应根据它的使用者要求而确定,有的系统比较庞大,它的构成元素较多;而有的系统针对某一专门领域,它的构成就比较简单。下面我们讨论软件工程环境的一般构成元素。

(一) 软件信息数据库

软件信息数据库是软件工程环境的核心,一般它要将经过各软件工具加工后的软件产品和半成品(如各种文档、源代码及测试数据)存入库内,以便根据需要进行查询、修改或输出。在 Ada 程序设计支撑环境 APSE 中,数据库是其核心,它的基本数据实体 OBJECT(如一个程序单元)由一个标识用的名字,一组属性(如源代码是“类型”属性),和 OBJECT 的内容组成。密执安大学的问题描述分析器(PSA)也是以数据库为中心,对用户送入的问题描述(用 PSL 表达)进行分析,并输出各种类型的表格。由于整体化软件环境中数据库联系着各个工具,加上软件信息的复杂性,如有正文、源代码及图形等,因而它往往是软件环境的“瓶颈”部分,各国计算机软件专家正在这方面进行各种理论和实践上的探索。

(二) 交互式的人机界面

一个统一的、友好的计算机用户接口是高质量软件工程

环境的重要标志。当前,高分辨率的图形显示终端的出现为它的实现提供了新的物质基础。XEROX 公司的 INTFRLISP-D 和 SMALLTALK-80 之所以引起广泛兴趣,就是由于它具有多窗口屏幕显示、鼠标器控制及弹出型 (pop-up) 的菜单驱动等强有力的交互功能。

(三) 语言工具

程序设计往往是多数软件工程环境所支持的,不少软件工程环境主要是程序设计环境。它一般提供编译程序(或解释程序)、连接装配程序、调试程序、静态及动态分析程序等一系列语言支撑工具。也有的系统支持非过程语言的运行,如 SMALLTALK-80 则是一种专门的面向对象 (object-oriented) 语言的程序设计环境。而 SPECIAL 则是一种断言说明语言的支撑环境。词法分析和语法分析的生成程序(如 Lex 和 Yacc)也是重要的语言工具。

(四) 质量保证工具

软件的质量保证是一个十分重要的课题。目前大量使用的仍是静态、动态测试技术以及各种形式的评审 (Review) 技术。与形式方法相联系的程序证明和验证技术已为许多计算机科学家所重视,相应的证明或验证实验系统也已经研制。

(五) 需求分析及设计工具

这方面目前主要仍由正文编辑工具、图形工具及一致性检验工具等来支持。语法制导 (Syntax-Directed) 的编辑程序已为细节设计及编码提供了较好的工具,这也是当前十分重要的一种软件支撑工具。随着软件形式化技术的发展,软件设计工具也将会有相应的发展,这方面的研究正在受到重

视。

(六) 配置管理工具

对于不同用户或不同的硬件配置，一个系统往往需不同的软件配置；即使同一软件也有不同的版本；加上软件产品的修改需要进行严格的管理等原因，软件的配置管理已成为软件生产管理的重要课题。已经研制了许多配置管理工具，如 UNIX/PWB 的源代码控制系统（SCCS）和修改请求控制系统（MRCS）。

三、特性

到底采用哪种类型的软件工程环境好呢？这完全取决于环境所服务的对象。由哪些构成成分组成软件工程环境也要根据需求来决定。那么怎样衡量软件工程环境的性能呢？一般，在研制软件工程环境时应考虑以下几个特性。

(一) 通用性及适应性

软件环境应能在生存周期各阶段为用户提供尽可能多的支持，如软件规格说明、模块设计、程序准备、编译、连接、运行、调试、测试、报告产生及内部通讯等等。软件环境应极少依赖于硬件环境，也应能适应各种工具之间的联系。

(二) 增量实现及可扩充性

软件工程环境应根据需要逐步扩充，先开发最急需及最基本的部分，同时也给用户提供可扩充的接口，包括用户可自行增加新的软件工具，或增加新的支撑硬件和输入输出设备。

(三) 工具间的整体性与一致性

由于软件环境是一组软件工具的有机集合,因此其整体性及一致性是十分重要的。统一的软件信息库、统一的用户界面及统一的输出报告产生器是其中的重要方面。由于工具之间的相互配合,能使小的工具构成强有力的工具,因此象 UNIX 程序设计环境中的管道线 pipe 及命令解释语言 shell 就是很重要的整体化工具。可重用的标准软件模块也是整体化讨论的另一侧面。INTERLISP 由于将编辑、编译、运行、调试溶为一体,并常驻在内存,而使其成为十分著名的整体化程序设计环境。更进一步如考虑技术上的整体性,将 60 年代发展起来的编译技术,70 年代发展起来的数据库技术和 80 年代发展起来的人工智能技术结合起来,将为软件工程环境提供更雄厚的技术基础。

(四) 效率

由于软件工程环境的开发成本比较昂贵,在研制时应充分考虑其可利用性。而效率是其重要因素,这包含两层意思,一是用户在使用软件环境时应是高效率的,不做浪费资源的多余工作;另一是软件环境本身应有效地利用其资源。

第三节 软件工程环境的发展前景

软件工程环境近年来发展很快,已经召开了各种专门的国际性或地区性的学术会议进行交流。近年来,在各届国际软件工程会议召开的同时,也举办了各种软件工具、环境的大型展览。许多商品化的“软件工程环境”已经进入市场。在技术上也出现了许多注目的成果,如多窗口交互图形技术、语法

制导编辑技术和形式化说明表示技术等。但总的来说，软件工程环境的发展还是初步的，从理论上、方法上、技术上还有待于进一步完善。由于软件产值的飞速增长，软件工程环境的重要性更突出了，随着计算机技术的进展，软件工程环境也将取得新的进展。

一、问题

目前存在的问题，大致如下：

1. 至今尚无完整的表示方法支持软件开发的各阶段，也还没有所有阶段可用的软件工程环境。
2. 软件信息数据库如何组织及构成还在摸索。
3. 如何兼顾软件工程环境的整体性和灵活性方面还有很多工作可做。例如 INTERLISP 是整体化很强的软件环境，但只是支持 LISP 语言；而 UNIX 程序设计环境有其很好的灵活性，各工具可灵活组合，但在支持软件开发各阶段的整体性方面却较差。
4. 软件工程环境在支持项目管理方面还较少，多数是支持软件开发的环境。
5. 软件工程环境的研制周期长，成本较高。

二、方法

为解决上面的问题，在技术上正在继续发展，在开发方法上也提出了以下一些策略。

（一）沿用现有成熟的工具

有的公司沿用现有成熟的软件工程环境，如 SDEM 沿用了 PSL/PSA 作为需求分析的工具。

（二）增量开发方法

软件工程环境的研制可以先生产一个子集，提交用户使用，然后再增加新的工具，不断扩充。

（三）快速开发原型

一些大型软件的需求分析和设计方案往往十分重要，为减少盲目性，明确整个软件系统的功能要求和设计方案，进行快速原型开发是有效的办法，它可以搜集到大量有用的信息，以便减少正规生产时的失误。软件工程环境的开发也可采用原型开发的方法。

三、展望

从国际上看，许多国家投入巨大的财力进行软件技术的开发与研究。美国国防部的 STARS 计划、英国的 ALVEY 计划、日本的 SIGMA 计划及欧洲共同体的 ESPRIT 计划，都强调软件开发方法、工具和环境研究的重要性。人们正从各种途径进行探索，以提高软件生产率。我国软件工作者也正在这个领域进行理论和实践方面的探索。下列几方面将是软件工程环境取得新进展的领域。

（一）智能加入工具

人工智能与软件工程的结合，将会产生一批智能化的工具或设计软件的“专家系统”。这不是人们通常采用的进化（evolutionary）方法，而是革命性变革（revolutionary）的手段。可以预计这一领域将会取得新的进展。

（二）形式方法的进展

软件生产自动化必然依赖于软件开发的形式化技术，它