教育部人才培养模式改革和开放教育试点教材

计算机应用专业系列教材

软件开发工具与环境

陈明 编

中央广播电视大学出版社

计算机应用专业系列教材

软件开发工具与环境

陈　明　编

中央广播电视大学出版社

图书在版编目（CIP）数据

软件开发工具与环境/陈明编. -北京：中央广播电视大学出版社，2001.8
计算机应用专业系列教材
ISBN 7-304-02051-2

Ⅰ.软... Ⅱ.陈... Ⅲ.①软件工具-电视大学-教材 ②软件开发环境-电视大学-教材 Ⅳ.TP311.52

中国版本图书馆 CIP 数据核字（2001）第 053121 号

计算机应用专业系列教材
软件开发工具与环境
陈 明 编

出版·发行/中央广播电视大学出版社
经销/新华书店北京发行所
印刷/北京云浩印刷厂
开本/787×1092 1/16 **印张**/15.75 **字数**/357 千字

版本/2001 年 7 月第 1 版 2002 年 5 月第 2 次印刷
印数/15001～26000

社址/北京市复兴门内大街 160 号 **邮编**/100031
电话/66419791 68519502 **（本书如有缺页或倒装，本社负责退换）**

书号：ISBN 7-304-02051-2/TP·167
定价：22.00 元

计算机应用专业系列教材

软件开发工具与环境

策　划　钱辉镜

设　计　沈雅芬　徐孝凯　何晓新

顾　问　许卓群　任为民

课程建设指导小组(按姓氏笔画排序)

陈　明（石油大学　教授）

郑纪蛟（浙江大学　教授）

侯炳辉（清华大学　教授）

高金源（北京航空航天大学　教授）

内容简介

本书主要包括四部分，即四章。第一章主要介绍有关软件开发工具与软件开发环境概念方面的内容；第二章主要介绍较流行的 Power Builder 软件开发工具；第三章介绍先进的、流行的图形开发工具 Power Designer；第四章介绍三个典型的实验。

全书在内容上呈积木结构，方法准确而简明，工具先进而流行，实验典型而实用。

本书可作为大专院校教材，也作为从事软件开发工作的工程技术人员的参考书。

前　言

软件工程的目标是研究一套科学的工程方法，以及与此相应的方便的工具系统，用来指导软件的开发研究工作。

软件开发工具是指支持软件生存期中某一阶段的任务实现而使用的计算机程序。软件开发环境是一组相关的软件工具的集合，它们组织在一起支持某种软件开发方法或与某种软件开发模型相适应。软件开发工具和软件开发环境是软件工程的重要组成部分，对于提高软件生产率，改进软件质量有着越来越大的作用。

本书主要包括四部分，即四章。第一章主要介绍有关软件开发工具与软件开发环境概念方面的内容；第二章主要介绍较流行的Power Builder软件开发工具；第三章介绍先进的、流行的图形开发工具Power Designer；第四章介绍三个典型的实验。

全书在内容上呈积木结构，方法准确而简明，工具先进而流行，实验典型而实用。

本书由陈永义教授、殷人昆教授、高全泉研究员审阅，他们提出了许多有益意见，在此表示真诚的谢意。

在编写过程中，中央广播电视大学理工部副主任沈雅芬副教授、徐孝凯副教授、何晓新老师和袁薇老师提出了许多指导性意见，在此一并表示真诚感谢。

由于作者水平有限，书中不足之处在所难免，敬请读者批评指正。

陈　明

2001年5月于北京

目　录

第一章　软件开发工具与环境概述

【学习目标】

1. 理解：软件开发工具与环境的概念及其相互关系，各种软件开发工具的分类和特点。
2. 了解：软件开发工具的发展过程，软件开发的基本过程。

【学习要点】

软件开发工具的功能、性能和常用软件开发环境。

1.1　软件工具概述

软件工具是一种软件，它是辅助和支援其它软件研制和维护的工具，研制软件开发工具的主要目的是为了提高软件生产率和改进软件的质量。

1. **软件工具的范围**　它既包括操作系统、编译程序、解释程序和汇编程序等成熟的传统工具，又包括支持需求分析、设计、编码、测试、维护等软件生存周期各阶段的开发工具和管理工具。

2. **软件工具提高了工作效率**　好的软件工具应该为软件人员所乐意使用。有时候并不在于功能如何齐全，而在于能减轻开发人员的劳动，提高效率和质量，方便用户，工作可靠。

软件工具的发展极大地推动了软件生产率的提高，缩短了开发周期。例如有关多窗口环境的工具、用于软件测试的工具等，都给软件生产率的提高带来了很大的影响。软件工具提高了软件的可移植性和标准化程度，便于相互借鉴和推广。用它可以开发方便的图形用户界面，使用户喜欢、爱用。用它可以降低学习计算机软件的难度，便于更多不懂计算机的人员使用计算机。软件工具是推广应用、开发应用的重要手段。

3. **软件工具的特点**　使用户易于操作，功能强大，覆盖面大，可靠性高，其本身可修

改、可扩充等。

4. 软件工具的评价　目前，用客观、定量的方法来统计软件工具对软件生产的效果还比较困难，一般只采用对若干软件项目在使用或不使用工具时，比较两者效率的方法。

软件工具的广泛使用，也促进了软件理论方法的发展，这是因为人们在大量的软件实践中，积累了经验，找出了问题，促进了理论和方法的发展。

1.2　软件开发工具的功能

在软件开发过程中，有许多工作需要用软件开发工具去支持或帮助。可以把软件开发工具应提供的各类支持工作归纳成为以下五个主要方面。

1. 认识与描述客观系统

这主要使用在软件开发工作的第一个阶段——需求分析阶段。由于需求分析在软件开发中的地位越来越重要，人们迫切需要在明确需求、形成软件功能说明书方面得到工具的支持。与具体的编程相比，这方面的不确定程度更高，更需要经验，更难形成规范化，因为这是一种对复杂系统的认识与理解的工作。俗话说："隔行如隔山"。每一个应用领域都有各自特殊的情况与规律，在这个领域中工作的人常常是通过几十年的实践才深刻领会它，而设计软件的人员要在尽可能短的时间内了解它，并在此基础上抽象出信息需求与信息流程，这无疑是十分困难的。这也正是人们希望软件开发工具提供帮助的一个重要原因。

2. 存储与管理开发过程中的信息

在软件开发的各阶段都要产生与使用许多信息。例如需求分析阶段要收集大量客观系统的信息，在此基础上形成系统功能说明书。而这些信息到了测试阶段还要用来对已经完成的软件进行评价。在总体设计阶段形成的对各模块的要求也要在模块验收时使用。当项目规模比较大时，这些信息量就会大大增加，当项目持续时间比较长的时候，信息的一致性就成为一个十分重要、十分困难的问题。如果再涉及软件的长期发展和版本更新，则有关的信息保存与管理问题就显得更为突出了。

3. 代码的编写与生成

在整个软件开发工作过程中，程序编写工作占了较多的人力、物力和时间，提高代码的编制速度与效率显然是改进软件工作的一个重要方面。根据目前以第三代语言编程为主的实际情况，这方面的改进主要是从代码自动生成和软件模块重用两个方面去考虑。代码的自动生成对于某些比较固定类型的软件模块来说，可以通过总结一般规律，制作一定的框架或模块，利用某些参数控制等方法，在一定程度上加以实现。这正是许多软件开发工具所要完成的任务。至于软件重用，则需要从更为根本的方面，即对软件开发的方法、标准进行改进，在此基础上形成不同范围的软件构件库（通用的、行业专用的、企业专用的等），这是一个远大而困难的目标。

4. 文档的编制或生成

文档编写是软件开发中十分重要的一项工作，不但费时费力，而且很难保持一致。在这方面，计算机辅助的作用可以得到充分的发挥。在各种文字处理软件的基础上，已有不少专用的软件开发工具提供了这方面的支持与帮助，如文档自动生成系统等。这里的困难往往在于保持文档的一致性，而且最后归结于信息管理方面的要求，即前面第 2 点阐述的内容。

5. 软件项目的管理

这一功能是为软件项目管理人员提供支持的。一般来说，项目管理包括进度管理，资源与费用管理，质量管理三个基本内容。在项目管理方面已有不少成功的经验、方法与软件工具。对于软件项目来说，还有两个比较特殊的问题。首先是测试工作方面的支持。由于软件的质量比较难于测定，所以不仅需要根据设计任务书提出测试方案，而且还需要提供相应的测试环境与测试数据。人们很自然地希望软件开发工具能够在这些方面提供帮助。另一个是版本管理问题。当软件规模比较大的时候，版本的更新，各模块之间以及模块与使用说明之间的一致性，向外提供的版本的控制等，都带来一系列十分复杂的管理问题。如果软件开发工具能够在这些方面给与支持或帮助，无疑将有利于软件开发工作的进行。

上述的功能就是人们对软件开发工具所寄予的希望与要求。

1.3 软件开发工具的特性

任何软件都有一定的性能指标。由于上面列举的功能范围十分广泛，各种功能在性能上的要求也不尽相同，很难有统一的规定。

所谓软件功能是指软件能做什么事，而性能则是指事情做到什么样的程度。换句话说，前者是定性地说明能不能做的问题，告诉我们它能在软件开发工程中提供哪些帮助；后者则尽可能定量地说明软件开发工具能做到什么样的程度，说明这些支持或帮助的程度如何。当然，作为一般的软件来说，效率、响应速度等也都是必须考虑的。但是，对于软件开发的工具来说，以下五项应当是特别重要的。

1. 表达能力或描述能力

因为软件项目的情况千变万化，软件开发工具要能够使用某些软件项目，就要能适应软件项目的种种不同的情况，否则就不可能对软件开发提供有效的、实际的帮助。比如在代码生成类型的软件开发工具中，常常是根据使用者的若干参数来生成特定的代码段。这些参数的多少与选择是否合理就决定了这个工具的能力大小。如果选择合理、参数详尽，则使用者就可以通过选择适当的参数，充分地规定自己所需要代码段的各种特征，从而成为自己真正需要的代码段。反之，如果工具只提供很少几个参数，用户没有什么选择的余地，那么生成的代码段就会十分死板，很难符合具体的应用软件的要求。类似的情况在需求分析，文档生成，项目管理中也经常遇到。我们将其统称之为描述能力或表达能力。在选择与比较软件开

发工具的时候，这一点应当是首先要考虑的。

2. 保持信息一致性的能力

软件开发者在管理开发过程中涉及大量的信息。这项工作中一致性的检验与控制是十分关键的。随着软件系统规模的增大，单靠人的头脑来保证这些信息的一致性，几乎是不可能的。所以实际工作要求软件开发工具不但要为人们存储大量的有关信息，而且要有条不紊地管理这些信息，而管理的主要内容就是保持它的一致性。至少在出现不一致的情况时要能够给出警告与提示。这方面的要求现在越来越高：各部分之间的一致，代码与文档的一致，功能与结构的一致都要求软件开发工具提供有效的支持与帮助。

3. 使用的方便程度

既然是工具，当然就应当尽量方便用户，而不能使用户因为用工具而增添麻烦。在计算机技术中，人机界面已经发展成为一个重要的分支。软件开发工具无疑应当充分利用这些技术成果。使其成为用户与硬件之间的桥梁。软件的开发应当与用户有充分的交流，其中涉及的表达方法，人机界面应当尽量通俗易懂，以便吸引使用者参与开发过程。因此，对于软件开发工具来说，是否易用是一项重要的性能指标。

4. 工具的可靠性

软件开发工具应当具有足够的可靠性，即在各种各样干扰条件下仍能保持正常工作，而不致丢失或弄错信息。软件开发工具涉及的都是软件开发过程中的重要信息，绝对不能丢失或弄错，因此，可靠性特别要紧。而且，使用软件开发工具的目的就是要防止出现不一致的情况。

5. 对硬件和软件环境的要求

如果软件开发工具对硬件、软件的环境要求太高，也会影响它的使用范围。一般来说，软件开发环境对环境的要求不应当超出它所支持的应用软件的环境要求，有时甚至还应当低于应用软件的环境要求。例如，项目管理的一些工具就可以在便携机上运行，尽管它支持的项目也许是小型机、以至中型机上运行的应用系统。当然，对于综合的、集成化的软件开发工具来说，环境的要求总会比单项的工具要求高，但随着硬件、软件技术的迅速发展，这方面的限制会减少。总之，软件开发工具的环境要求尽量低，以有利于广泛使用。

1.4 软件开发工具的分类

软件开发工具可以从若干不同的角度进行分类。以下是几种主要的分类方法。

1. 基于工作阶段划分的工具

基于各个阶段对信息的需求不同，软件开发工具大致可以分为三类：设计工具、分析工具、计划工具。

(1) 设计工具是最具体的，它是指在实现阶段对人们提供帮助的工具。例如各种代码生

成器、一般所说的第四代语言和帮助人们进行测试的工具（包括提供测试环境或测试数据）等，都属于设计工具之列。它是最直接地帮助人们编写与调试软件的工具。

（2）分析工具主要是指用于支持需求分析的工具，例如，帮助人们编写数据字典的、专用的数据字典管理系统，帮助人们绘制数据流程图的专用工具，帮助人们画系统结构图或ER图的工具等。他们不是直接帮助开发人员编写程序，而是帮助人们认识与表述信息需求与信息流程，从逻辑上明确软件的功能与要求。

（3）计划工具则是从更宏观的角度去看待软件开发。它不仅从项目管理的角度帮助人们组织与实施项目，把有关进度、资源、质量、验收情况等信息有条不紊地管理起来，而且考虑到了项目的反复循环、版本更新，实现了跨生命周期的信息管理与共享，为信息以及软件的复用创造了条件。

在实践中，设计工具是出现最早、数量最大的软件工具。原因是它们直接为软件开发过程中的编程、调试、文档编写工作提供帮助。即使是个人单独开发的软件，也可以从这种工具的使用中得益。分析工具则出现得较晚，数量也少一些，因为需求分析的复杂程度与项目的规模有直接的关系。对于功能单纯，规模较小的软件来说，需求分析是比较简单的，数据的结构与流程也比较容易弄清，不必用专门的工具就可以进行。然而对于功能强大、关系复杂、规模较大的软件系统，工具的应用就成为必不可少的了。正因为这样，除了少量专用的工具外，大多数的分析工具都是作为较大型专用系统出现的，例如上面提到的CDD和Dictionary/3000等。至于计划工具则完全是为项目主管人员服务的。这类软件开发工具保存与管理的信息，都是与整个项目有关的宏观信息。这样的工具当然只有在软件规模达到一定程度时才会需要与产生。所以它的出现与发展都较晚。但是从另一方面看，正是软件产业的规模越来越大，才使计划工具的出现成为必然的趋势。它的重要性将随着软件产业的发展而越来越为人们所认识，尽管现在它还不那么普遍。

2. 基于集成程度划分的工具

集成化程度是用户接口一致性和信息共享的程度，是一个新的发展阶段。集成化的软件开发工具要求人们对于软件开发过程有更深入的认识和了解。这并不是说集成化没有必要或者没有可能，而是说，目前我们还是应当充分利用各种专用软件开发工具。至于开发与应用集成化的软件开发工具是应当努力研究与探索的课题，而真正做到集成化地、统一地支持软件开发全过程的工具，还是相当困难的。集成化的软件开发工具也常常被称为软件工作环境。

3. 基于硬件、软件的关系划分的工具

按与硬件和软件的关系，软件开发工具可以分为两类：依赖于特定计算机或特定软件（如某种数据库管理系统）和独立于硬件与其它软件的软件开发工具。这与工具自身的情况有关。一般来说，设计工具多是依赖于特定软件的，因为它生成的代码或测试数据不是抽象的，而是具体的某一种语言的代码或该语言所要求的格式的数据。例如ORACLE的CASE所生成的ORACLE的代码，HP/9000机器上的4GL生成的就是在HP/9000上可运行的代码。而

分析工具与计划工具则往往是独立于机器与软件的，由于集成化的软件开发工具涉及较多的开发阶段，所以集成化的软件开发工具常常是依赖于机器与软件的。

软件开发工具是否依赖于特定的计算机硬件或软件系统，对于应用的效果与作用有直接影响。因此在研究与使用软件开发工具时必须注意。

4. 基于应用领域划分的工具

按照应用领域的不同，应用软件可以分为事务处理、实时应用、嵌入式应用软件等。随着个人计算机与人工智能的发展，与这两个方面相联系的应用软件，也取得较大的进展。

事务处理是范围最广的一类应用。从工资、仓库、会计等单项管理，到具有决策能力、拥有大型数据库的管理信息系统（Management Information System，简称 MIS），几乎无所不包。以批处理方式进行数据处理是这类软件的传统特征，近期又扩展了支持交互计算功能的应用方式，例如常用于收银处的各种计费软件，储蓄所使用的存款软件等，都可以归为此类。

1.5　软件开发环境

软件开发环境（Software Development Environment，SDE）是一组相关的软件工具的集合，将它们组织在一起，支持某种软件开发方法，软件开发环境又称之为集成式项目支持环境（Integrated Project Support Environment，IPSE）。

1. 软件开发环境的特性

软件开发环境的具体组成可能千姿百态，但都包含交互系统、工具集和环境数据库，并具备下列特性：

（1）可用性

用户友好性、易学、对项目工作人员的实际支持等。

（2）自动化程度

在软件开发过程中，对用户所进行的频繁的、耗时的或困难的活动提供自动化的程度。

（3）公共性

公共性是指覆盖各种类型用户（如程序员、设计人员、项目经理和质量保证工作人员等）的程度。或者指覆盖软件开发过程中的各种活动（如体系结构设计、程序设计、测试和维护等）的程度。

（4）集成化程度

集成化程度是指用户接口一致性和信息共享的程度。

（5）适应性

适应性是指环境被定制、剪裁或扩展时符合用户要求的程度。对定制而言，是指环境符合项目的特性、过程或各个用户的爱好等的程度。对剪裁而言，是指提供有效能力的程度。对扩展而言，是指适合改变后的需求的程度。

(6) 价值

得益和成本的比率。得益是指生产率的增长，产品质量的提高、目标应用开发时间/成本的降低等。成本是指投资、开发所需的时间，培训使用人员到一定水平所需要的时间等。

2. 软件开发环境的结构

一般说来，软件开发环境都具有层次式的结构，可区分为四层：

(1) 宿主层：包括基本宿主硬件和基本宿主软件。

(2) 核心层：一般包括工具组、环境数据库和会话系统。

(3) 基本层：一般包括最少限度的一组工具，如编译工具、编辑程序、调试程序、连接程序和装配程序等。这些工具都是由核心层来支援的。

(4) 应用层：以特定的基本层为基础，但可包括一些补充工具，借以更好地支援各种应用软件的研制。

3. 软件开发工具与环境的关系

任何软件的开发工作都是处于某种环境中，软件开发环境的主要组成成分是软件工具。为了提高软件本身的质量和软件开发的生产率，人们开发了不少工具为软件开发服务。例如，最基本的文本编辑程序、编译程序、调试程序和连接程序；进一步还有数据流分析程序、测试覆盖分析程序和配置管理系统等自动化工具。面对众多的工具，开发人员会感到眼花缭乱，难于熟练地使用它们。针对这种情况，从用户的角度考虑，不仅需要有众多的工具来辅助软件的开发，还希望他们能有一个统一的界面，以便于掌握和使用，另外，从提高工具之间信息传递的角度来考虑，希望对共享的信息能有一个统一的内部结构，并且存放在一个信息库中，以便于各个工具去存取。因此，软件开发环境的基本组成有三个部分：交互系统、工具集和环境数据库。

软件开发工具在软件开发环境中已不是各自封闭和分离的了，而是以综合、一致和整体连贯的形态来支持软件的开发，它们是与某种软件开发方法或者与某种软件加工模式相适应的。

4. 软件开发环境的分类

目前世界上已有近百个大小不同的程序设计环境系统在使用，这些环境系统相互之间的差别很大。根据各种软件环境的特点，软件开发环境的类型包括：

(1) 按研制目标分类

针对各个不同应用领域的程序设计环境，如开发环境、项目管理环境、质量保证环境和维护环境等。

(2) 按环境结构来分类

基于语言的环境，基于操作系统的环境和基于方法论的环境。

(3) 按工作模式分类

交互式软件环境、批处理软件开发环境和个人分布式的环境等。

1.6　软件开发过程

同任何事物一样，软件也有一个从孕育、诞生到生长、成熟最后再到衰亡的生存过程。一般，一个软件从定义到开发、使用和维护，直到最终被废弃，要经历一个漫长的时期，通常把软件经历的这个漫长的时期称为生存周期。软件生存周期就是从提出软件产品开始，直到该软件产品被淘汰的全过程。研究软件生存周期是为了更科学地、有效地组织和管理软件的生产，从而使软件产品更可靠、更经济。采用软件生存周期来划分软件的工程化开发，使软件开发分阶段依次进行。前一个阶段任务的完成是后一个阶段的前提和基础，而后一个阶段通常是将前一个阶段提出的方案进一步具体化。每一个阶段的开始与结束都有严格的标准。前一个阶段结束的标准就是与其相邻的后一个阶段开始的标准。每一个阶段结束之前都要接受严格的技术和管理评审。不能通过评审时，就要重复前一阶段的工作直至通过上述评审后才能结束。采用软件生存周期的划分方法，是每一阶段的任务相对独立，有利于简化整个问题且便于不同人员分工协作。而且其严格而科学的评审制度保证了软件的质量，提高了软件的可维护性，从而大大提高了软件开发的成功率和生产率。

软件生存周期一般可分为以下步骤：

S1：问题定义

S2：可行性研究

S3：需求分析

S4：概要设计

S5：详细设计

S6：编码

S7：测试

S8：运行与维护

在软件的研制和开发的过程中，首先，要了解和分析用户的需求，以及经济、技术和时间等方面是否可行。然后，将用户的需求规范化、形式化地写出来。编写成需求说明书及初步的系统用户手册，提交评审。第三，将软件需求设计为软件过程描述，即设计人员将已确定的各项需求转化成一个相应的体系结构。结构的每一组成部分都是意义明确的模块，每个模块都与某些需求相对应（概要设计）。然后对每个模块的具体任务进行具体的描述（详细设计）。第四，编写代码，就是把过程描述转换成机器可执行的代码。第五，测试，发现错误，进行改正。最后是维护，包括故障的排除以及为适应使用环境的变化和用户对软件提出新的要求所作的修改。

软件生存期也可以分为三个大的阶段：计划阶段、开发阶段和维护阶段。

1. 计划阶段

计划阶段又可分两步：软件计划和需求分析。第一步，因为软件是计算机系统中一个子系统，这样不但要从确定的软件子系统出发，确定工作域，即确定软件总的目标、功能等；开发这样的软件系统需要哪些资源（人力和设备）。作出成本估算，而且还要求作出可行性分析，即在现有资源条件下能否实现这样的目标；最后要提出进度安排，并写出软件计划文档。上述问题都要进行管理评审。第二步，在管理评审通过以后，要确定系统定义和有效性标准（软件验收标准），写出软件需求说明书。还要开发一个初步用户手册，这里要进行技术评审。技术评审通过以后。再进行一次对软件计划的评审，因为这时对问题有了进一步的了解。而计划制定时，数据较少，且经验不足，所以对制定的计划需要进行多次修改，以尽量满足各种要求，然后再进入到开发阶段。

2. 开发阶段

开发阶段要经三个步骤：设计、编码和测试。首先对软件进行结构设计，定义接口，建立数据结构，规定标记。接着对每个模块进行过程设计、编码和单元测试。最后进行组合测试和有效性测试，对每一个测试用例和结果都要进行评审。

3. 维护阶段

首先要做的工作，就是配置评审，检查软件文档和代码是否齐全，两者是否一致，是否可以维护等，下面要确定维护组织和职责，并定义表明系统错误和修改报告的格式。维护可分为改正性维护、完善性维护和适应性维护等。维护内容广泛，有人把维护看成是第二次开发，不只是纠错。要适应环境的变化，就要扩充和改进，但不是建立新系统。维护的内容应该通知用户，要得到用户的认可。然后则可进入修改，修改不只是代码修改，必须要有齐全的修改计划、详细过程以及测试等文档。

1.7 常用开发环境

目前，较流行的操作系统平台环境有 Windows、UNIX 和 Linix 等，下面给予较简单的介绍。

1.7.1 Windows98 开发环境

1.Windows 操作系统

（1）Windows 操作系统的特点：面向对象的图形用户界面，一致的用户接口，与设备无关的图形输出以及多任务。

（2）Windows 编程的四个特点：事件驱动、消息循环、图形输出、资源共享。

（3）Windows 的基本用户界面对象：包括窗口、标题栏、图标、光标、插入符号、对话框、控件等。