



软件开发环境

Software Development Environment

主讲教师 刘凡
fanliu@hhu.edu.cn



- ▲ 软件开发环境概述
- ▲ **Web**开发技术概述
- ▲ **Web**软件开发--JSP编程
- ▲ **Web**软件开发--框架与开发模式

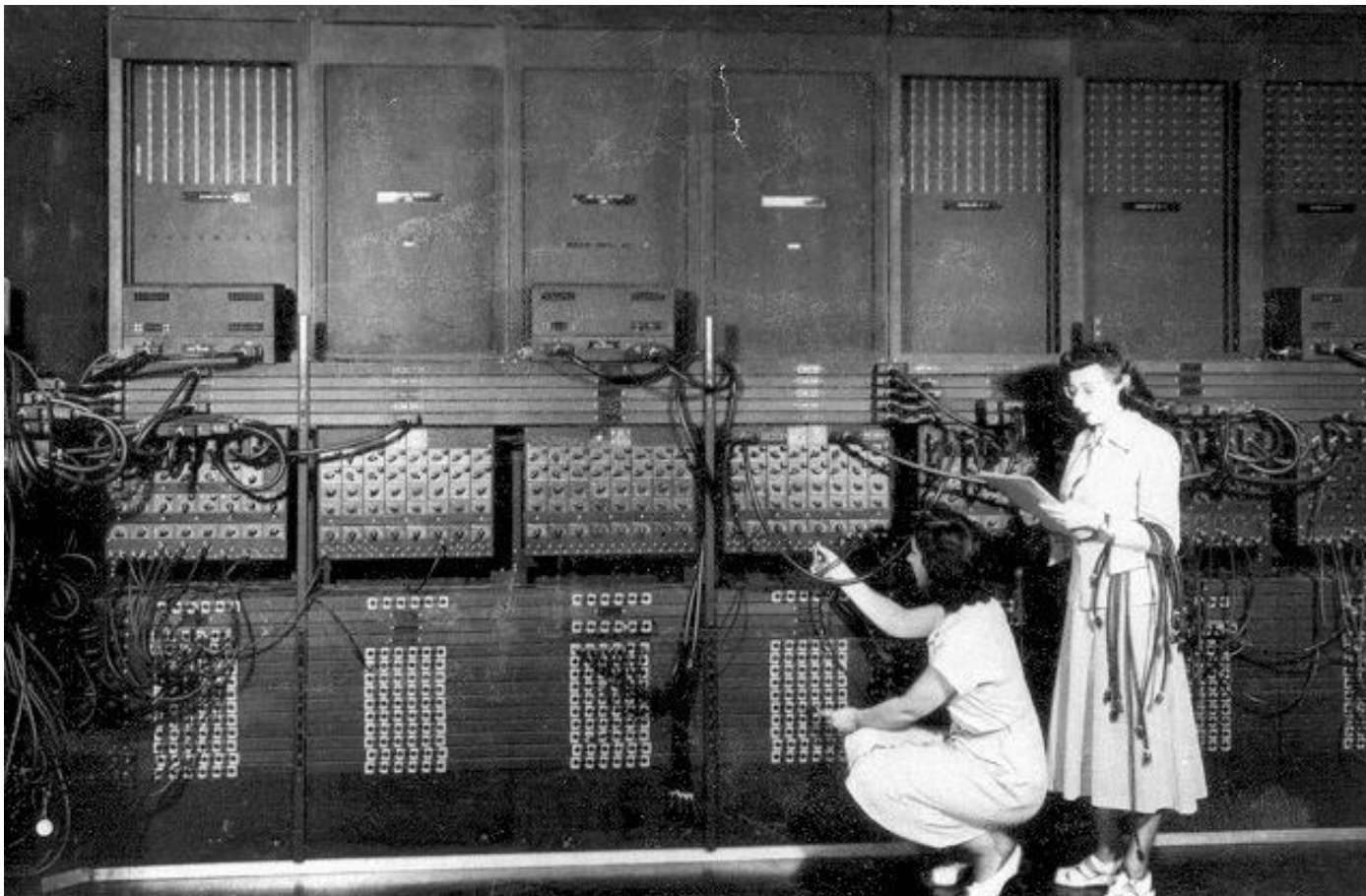


第一章

软件开发环境概述



软件开发环境的背景



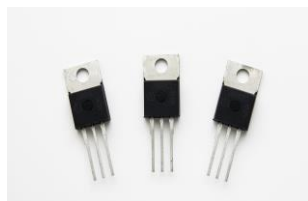
ENIAC：电子数字积分计算机（1943）



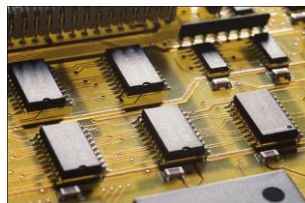
计算机硬件发展史



第一代
电子管计算机
1946-1958



第二代
晶体管计算机
1958-1964



第三代
集成电路计算机
19564-1971



第四代
大规模集成电路
计算机
1971年以后



第五代
智能计算机
1981年以后

1946年至今





软件开发环境的背景

- ❏ 需要计算机辅助大型程序的组装、调试、修改。
- ❏ 需要计算机参与文档管理。
- ❏ 早期的软件工具彼此独立，为使用一个工具必须从另一工具退出。事实上，软件开发过程的各阶段紧密联系。
- ❏ 软件开发环境将所有工具有机地联系起来，实现各工具有统一的接口和内部格式，前阶段工具产生的信息能被后继阶段的工具利用。特别是开发环境中的软件工具提供的统一的友好的用户界面，非常便于使用者从使用一个工具转换到使用另一个工具。



软件开发环境是一个综合性的概念，
从软件工程的角度看它还涵概了软件
生命周期中软件开发与设计方法



一、软件开发环境的概念

1、软件开发环境的定义

SDE: Software Development Environment

- ❏ 广义：是围绕着软件开发的一定目标而组织在一起的一组相关软件工具的有机集合。
- ❏ IEEE和ACM支持的国际工作小组提出的关于“软件开发环境”的定义：“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”。



一、软件开发环境的概念

2. 软件开发环境的组成

软件开发环境基本组成部分：

工具集、交互系统、环境数据库

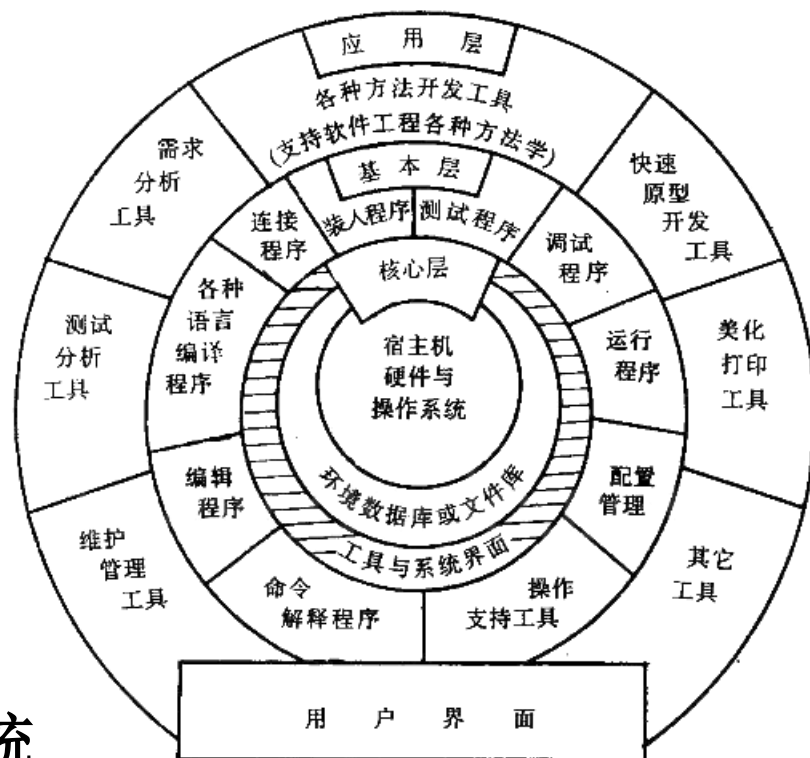
软件开发环境可分为4层：

宿主层：包括宿主硬件和操作系统

核心层：包括工具组、环境数据库和系统界面

基本层：包括最少限度的一组工具，如编译、编辑、调试等

应用层：以特定的基本层为基础，但可包括一些补充工具





一、软件开发环境的概念

3.软件开发环境的发展

👉 软件开发环境数据库：较初级→较完整→智能化

👉 软件开发与设计方法：

70年代的结构化方法学→80年代中后期的面向对象方法学→90年代对于系统集成方法和集成系统的研究



一、软件开发环境的概念

4.软件开发环境的分类

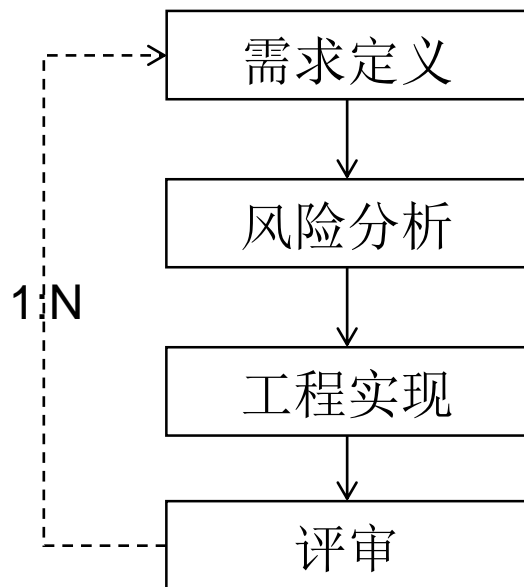
- 按软件开发模型及开发方式分类：瀑布、螺旋、喷泉、原型及结构化方法、面向对象方法等





一、软件开发环境的概念

4. 软件开发环境的分类

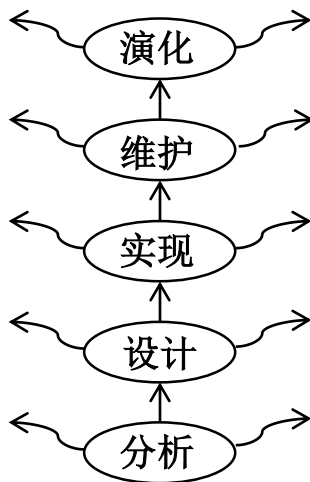


👉 **螺旋模型**基本做法是在瀑布模型的每一个开发阶段前引入一个非常严格的风险识别、风险分析和风险控制，它把软件项目分解成一个个小项目。每个小项目都标识一个或多个主要风险，直到所有的主要风险因素都被确定。



一、软件开发环境的概念

4. 软件开发环境的分类




👉 **喷泉模型**各个阶段没有明显的界限，开发人员可以同步进行开发，其优点是可以提高软件项目开发效率，节省开发时间。缺点是不利于项目的管理，这种模型要求严格管理文档，使得审核的难度加大，尤其是面对可能随时加入各种信息、需求与资料的情况。



一、软件开发环境的概念

4.软件开发环境的分类

 **原型化模型：** 第一步是建造一个快速原型，实现客户或未来的用户与系统的交互，经过和用户针对原型的讨论和交流，弄清需求以便真正把握用户需要的软件产品是什么样子的。充分了解后，再在原型基础上开发出用户满意的产品。


 **适用于：**

- (1) 用户需求不清，管理及业务不稳定，需求经常变化
- (2) 规模小，不太复杂



一、软件开发环境的概念

4.软件开发环境的分类

 **结构化方法：**结构化系统开发方法基本思想在系统建立之前信息就能被充分理解。它要求严格划分开发阶段，用规范的方法与图表工具有步骤地来完成各阶段的工作，每个阶段都以规范的文档资料作为其成果，最终得到满足用户需要的系统。

 **适用于：**
组织相对稳定、业务处理过程规范、需求明确且在一定时期内不会发生大的变化的大型复杂系统的开发



一、软件开发环境的概念

4.软件开发环境的分类

👉 **面向对象法：**利用面向对象的信息建模概念，如实体、关系、属性等，同时运用封装、继承、多态等机制来构造和模拟现实系统的开发方法。



一、软件开发环境的概念

4.软件开发环境的分类

- 👉 按应用范围分类：通用型和专用型
- 👉 按开发阶段分类：前端开发、后端开发
- 👉 按集成化程度分类：操作系统之上；具有真正的数据库，而不是文件库；建立在知识库系统之上，出现集成化工具集



二、软件工具的概念

1. 什么是软件工具

软件工具指为支持计算机软件的开发、维护、模拟、移植或管理而研制的程序系统。所以软件工具是一个程序系统。

2. 软件工具的分类

Reifer和Trattner将软件工具分为6类：模拟工具、开发工具、测试和评估工具、运行和维护工具、性能测量工具和程序设计支持工具。



三、计算机辅助软件工程CASE

1、CASE概念

计算机辅助软件工程：**Computer-Aided Software Engineering**，缩写为**CASE**。

CASE是一组工具和方法集合，可以辅助软件开发生命周期各个阶段进行软件开发。



三、计算机辅助软件工程CASE

2、CASE技术的发展史

第一代CASE工具(70年代早期): 一般是基于文件的







第二代CASE工具(80年代早期): 不仅支持使用图形的结构化方法, 还能通过工程字典的方式使开发信息在不同的CASE工具中共享, 但局限于同一制造商的工具; 80年代晚期, 出现基于数据库的CASE产品。

CASE系统集成: 90年代, CASE工具发展为CASE环境



三、计算机辅助软件工程CASE

2. 常见的CASE工具

-  画图工具
-  屏幕显示和报告生成工具
-  数据字典
-  规格说明检查工具
-  代码生成工具
-  文档自动生成工具



图稿绘制工具Visio



提供绝大多数框图的绘画功能（包括信息领域的各种原理图，设计图），同时提供部分信息领域的实物图。使用方便，与**Word**集成良好。

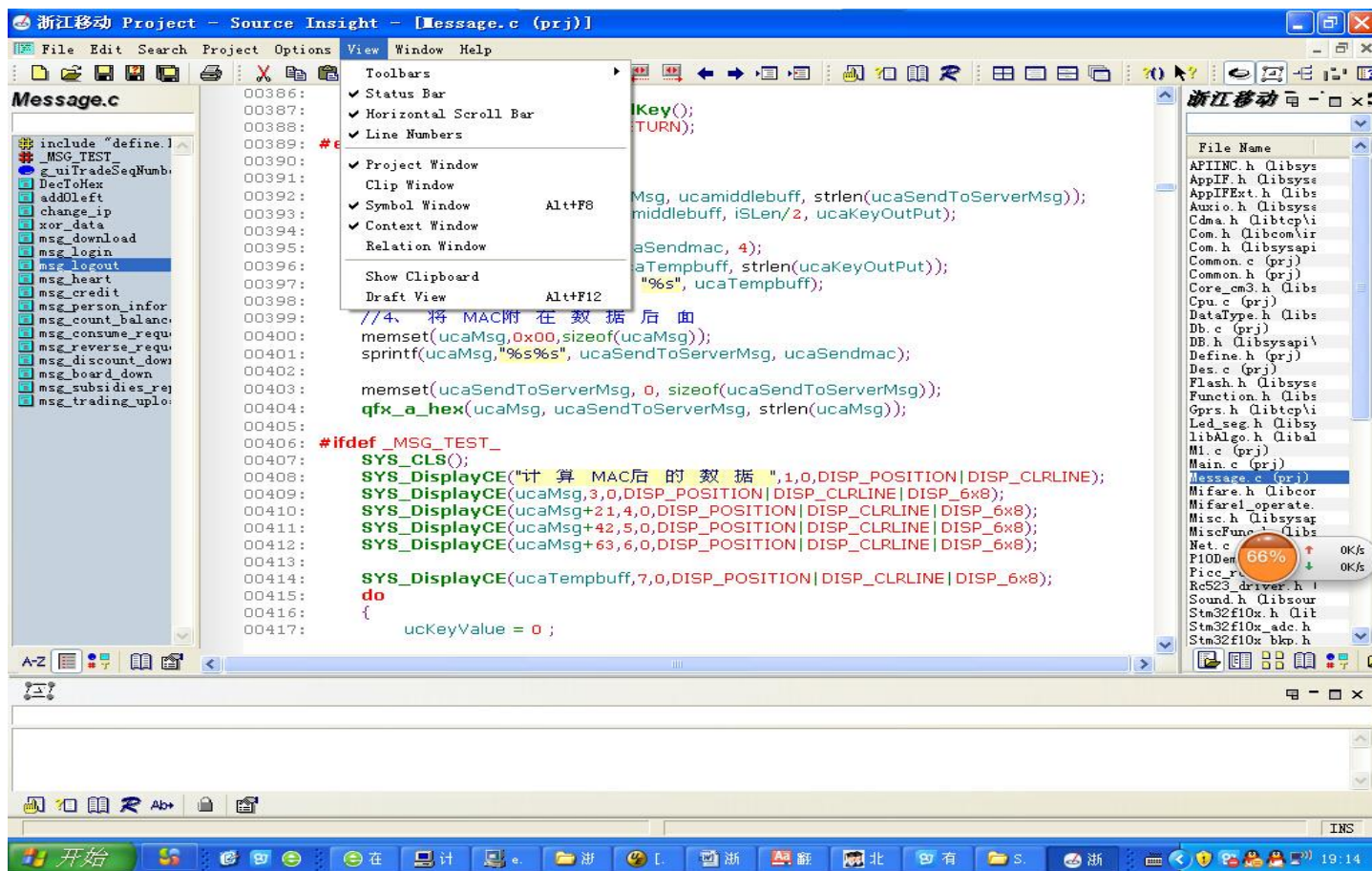




源码浏览工具 Source Insight



以工程的方法管理原码，提供非常适合再工程的浏览手段。提供函数交叉调用的分析，并以树状形式显示调用关系。

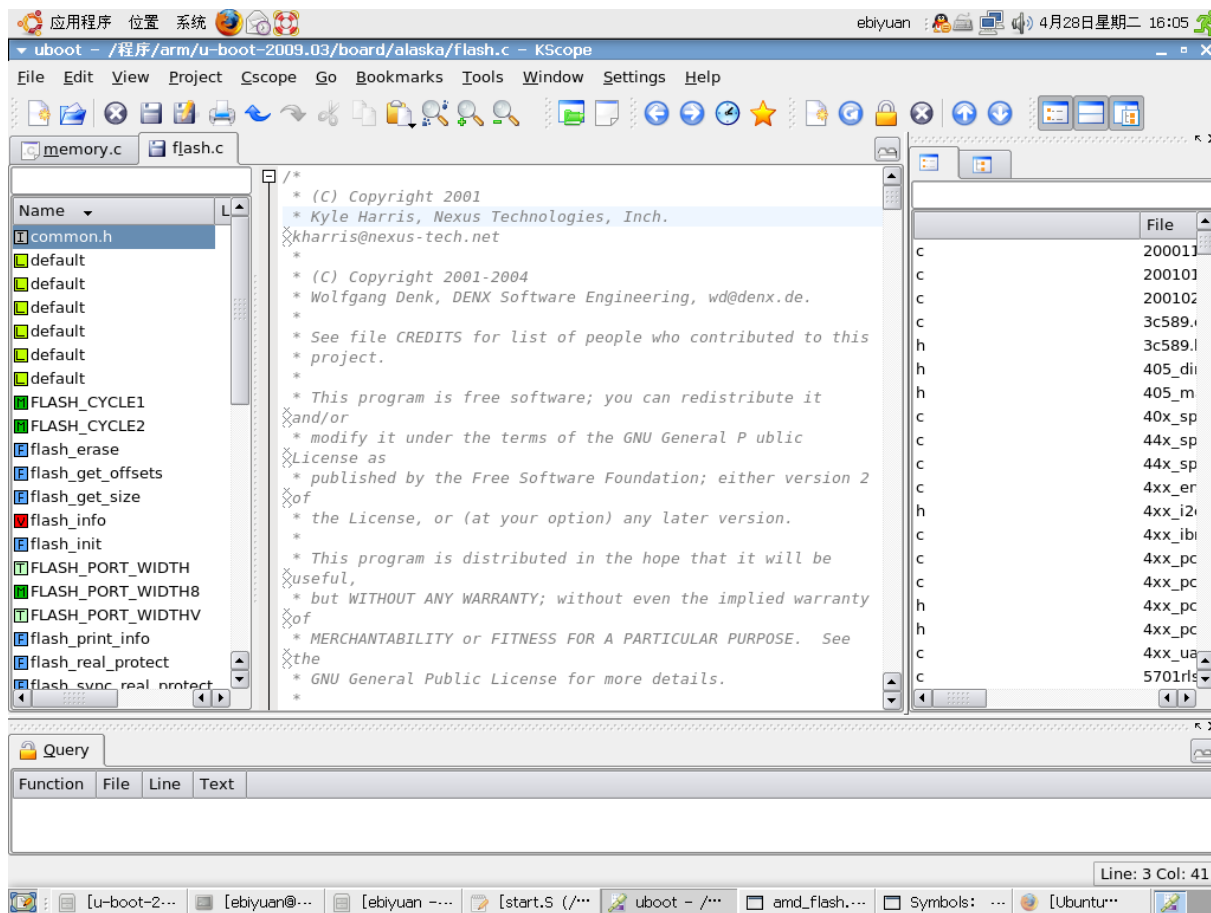




源码浏览工具Source Navigator



提供原码高亮显示和编辑，提供头文件的包含关系分析，提供类的层次关系，其最大的特点是原码始终和文件联系在一起，提供到文件的导航。

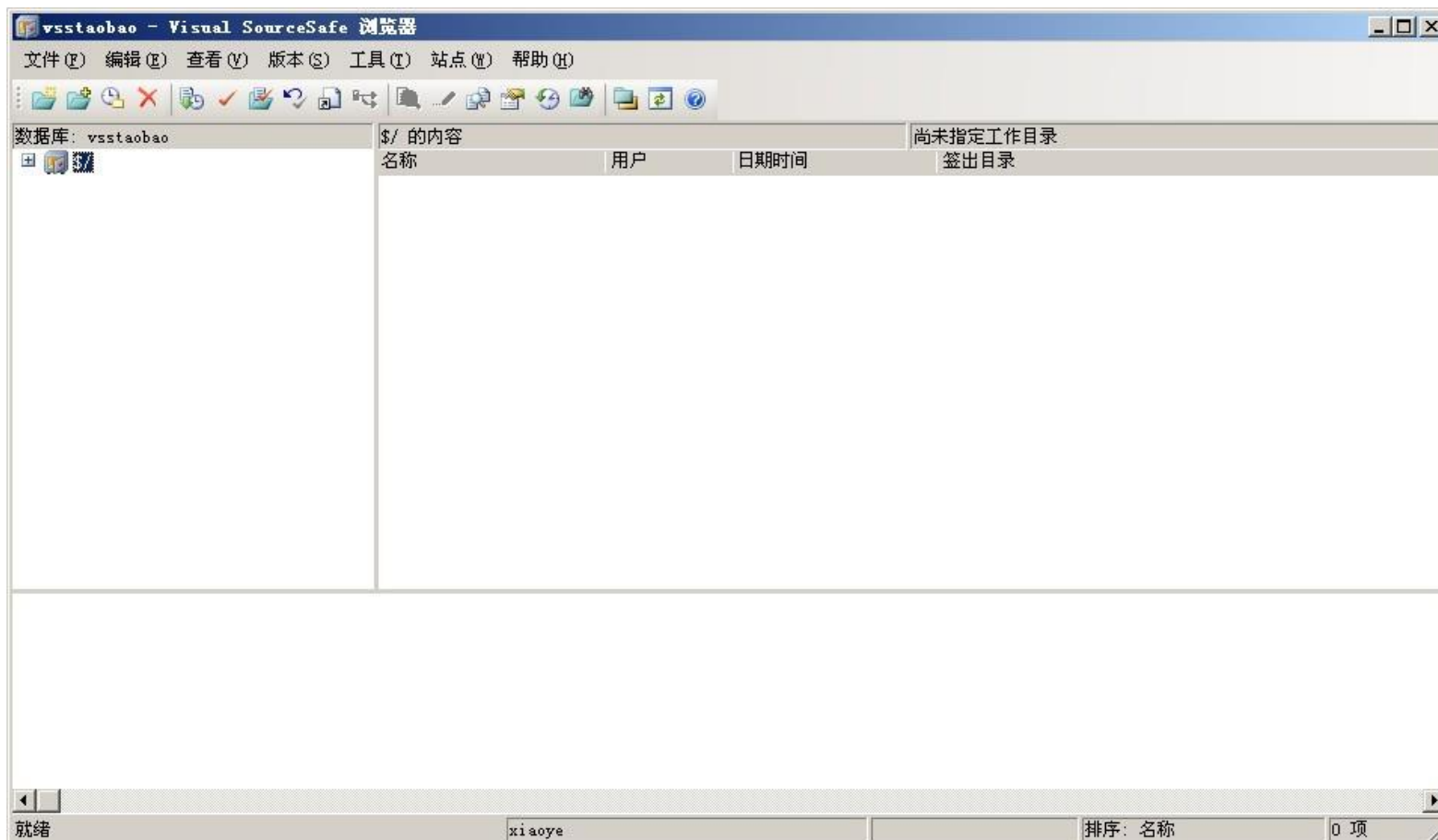




配置管理工具 **Visual Source Safe**



VSS是微软的**Studio**企业版包含的版本管理工具，提供了基本的认证安全和版本控制机制，能够对几乎任何类型的文件进行控制，提供版本对比。

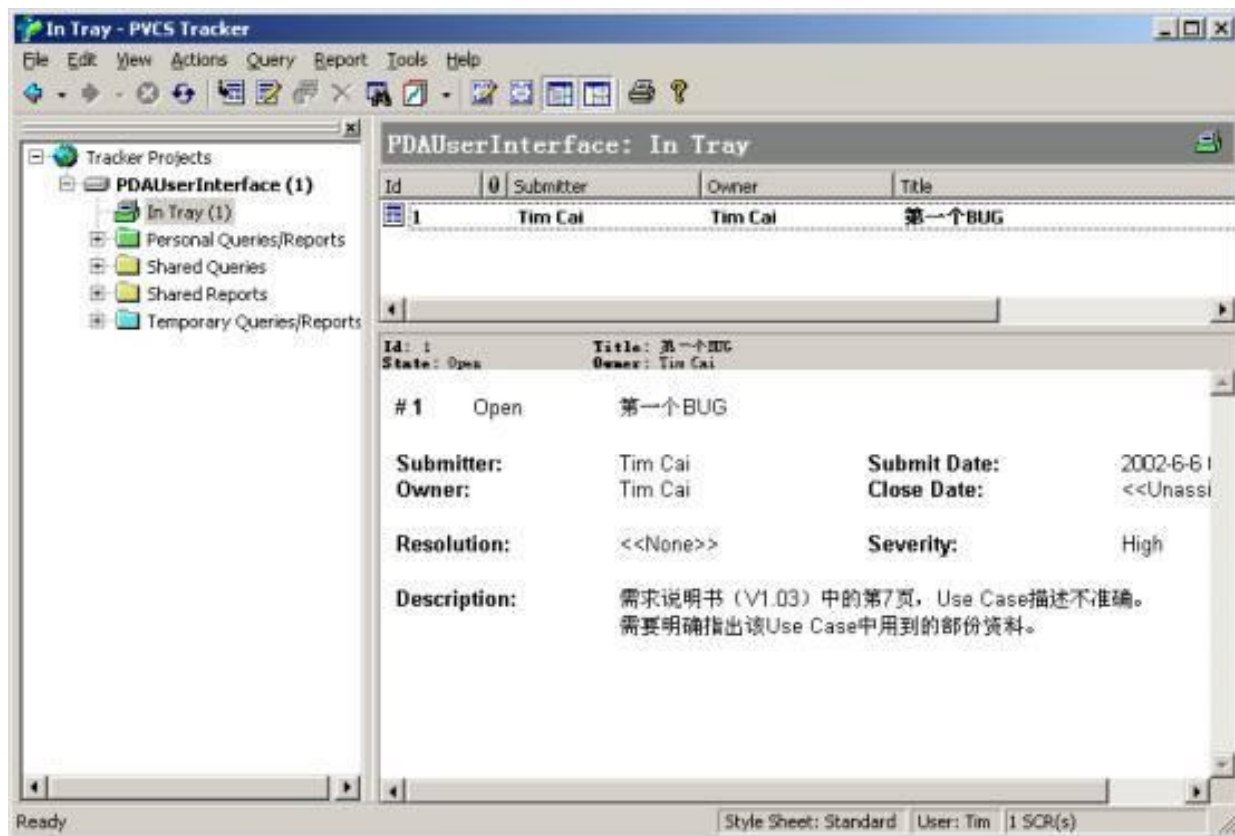




配置管理工具PVCS



PVCSVersionManager记录开发过程中出现的变更和修改并使修订版本自动升级；PVCSTracker、PVCSNotify自动追踪上述变更和修改；PVCSRequisitePro提供独特的MicrosoftWord界面和需求数据库，使开发机构实时、直观地对来自于最终用户的项目需求及需求变更进行追踪和管理。

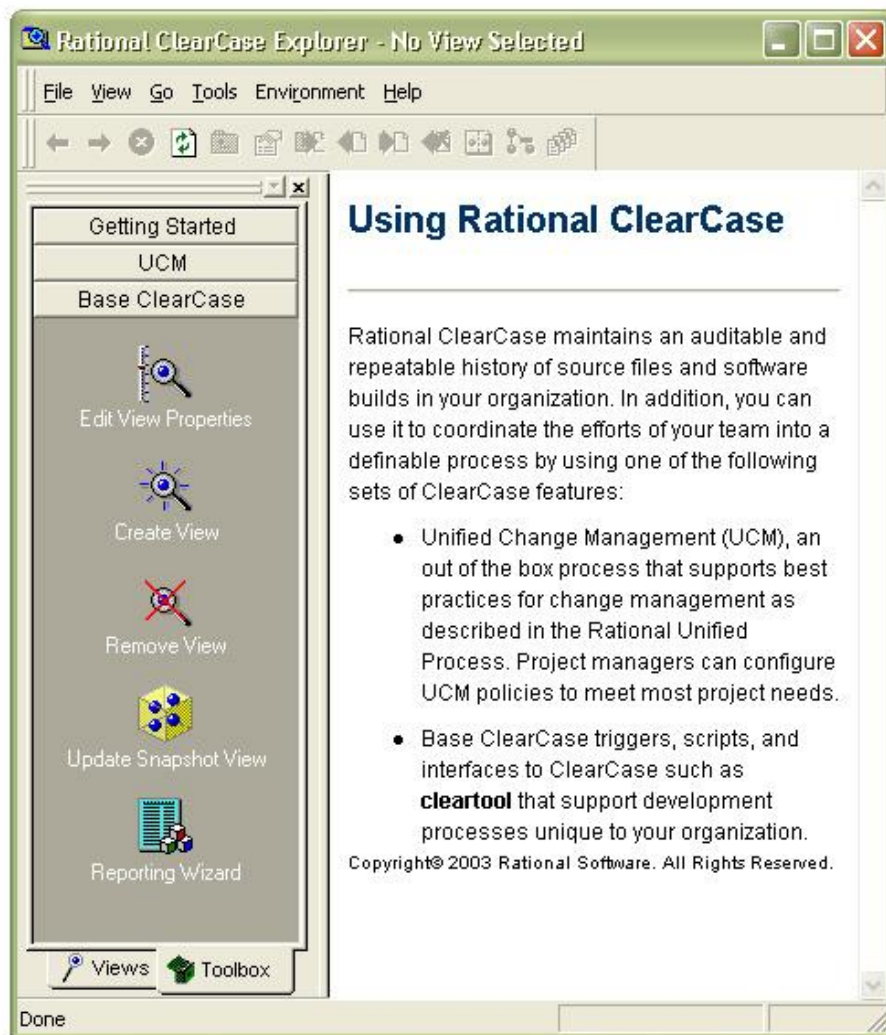




配置管理工具ClearCase



提供了**VOB**（**Versioned Object Base**）的概念进行配置管理，所有的版本存储在**VOB**中。

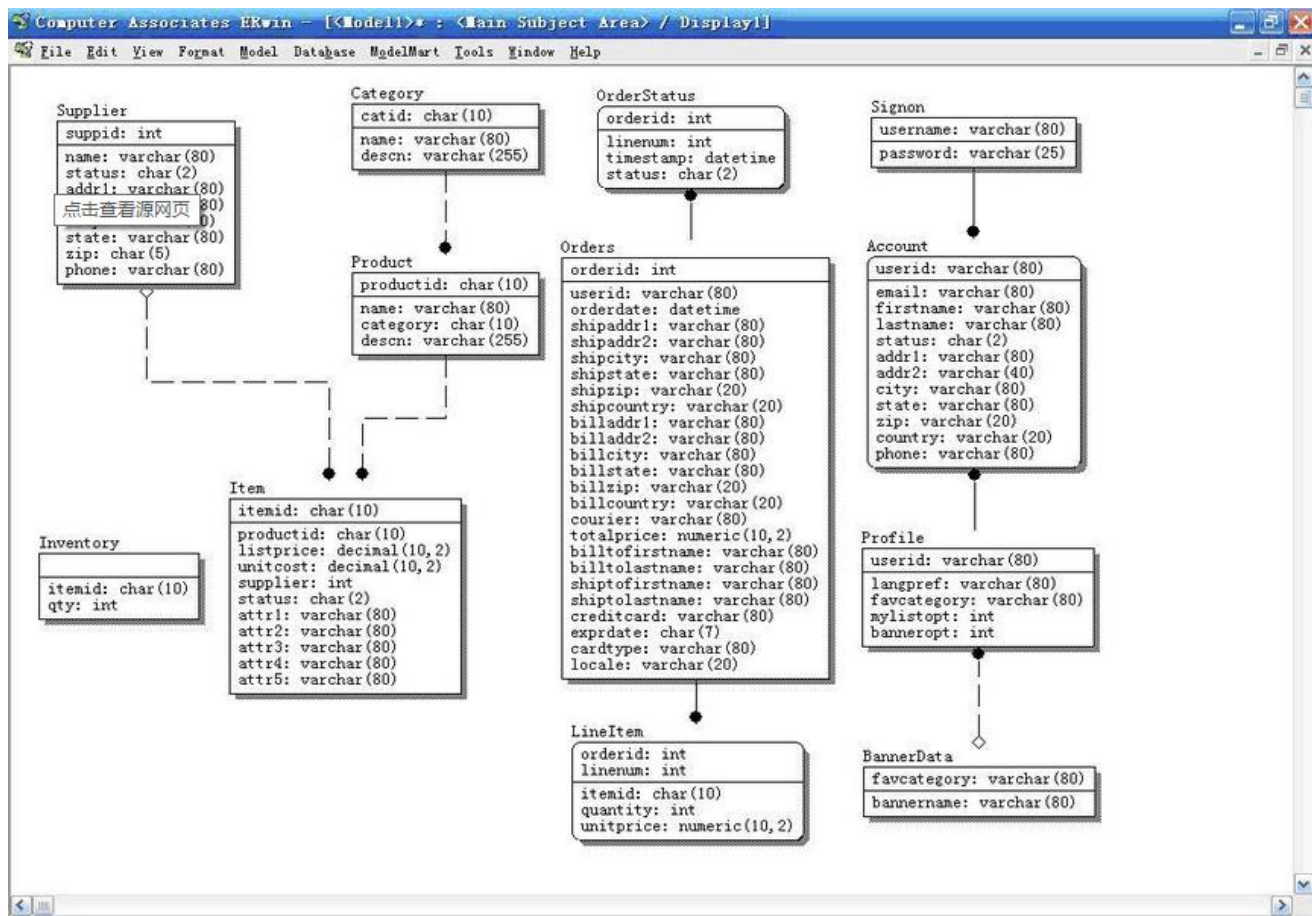




数据库建模工具ERWin



CA公司出品的强大的数据库建模工具，采用ER模型，不同的属性类型采用可定制的图标显示，实体间的关系一目了然。

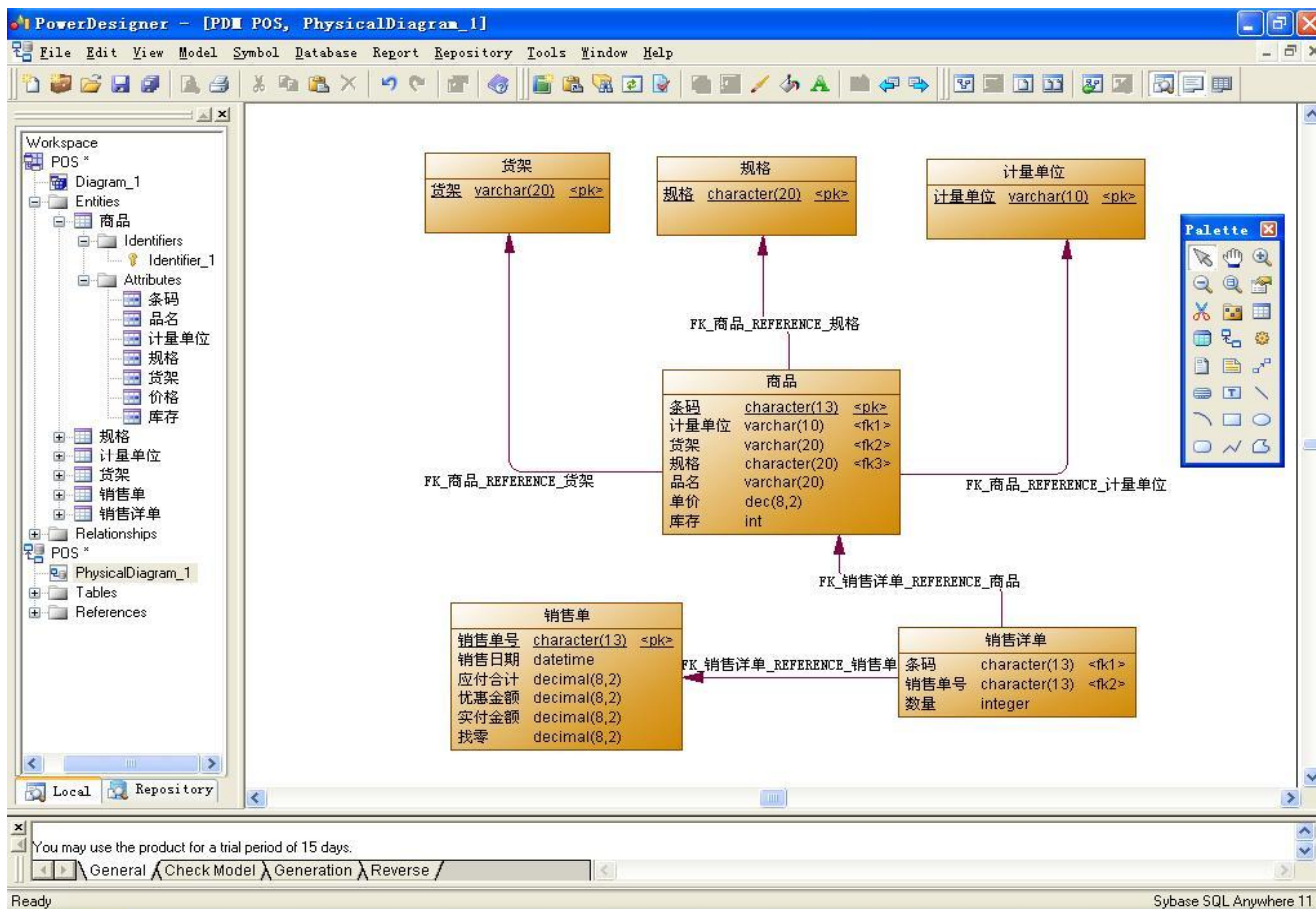




数据库建模工具PowerDesigner

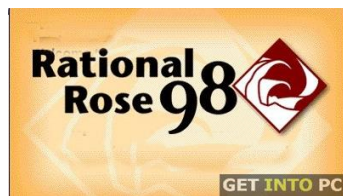


Sybase推出的数据库设计工具，采用**E-R**模型，从概念数据模型和物理数据模型两个层次对数据库进行设计，概念模型描述的是独立于**DBMS**的实体和实体关系定义，物理模型是在概念模型的基础上针对目标数据库管理系统的具体化。

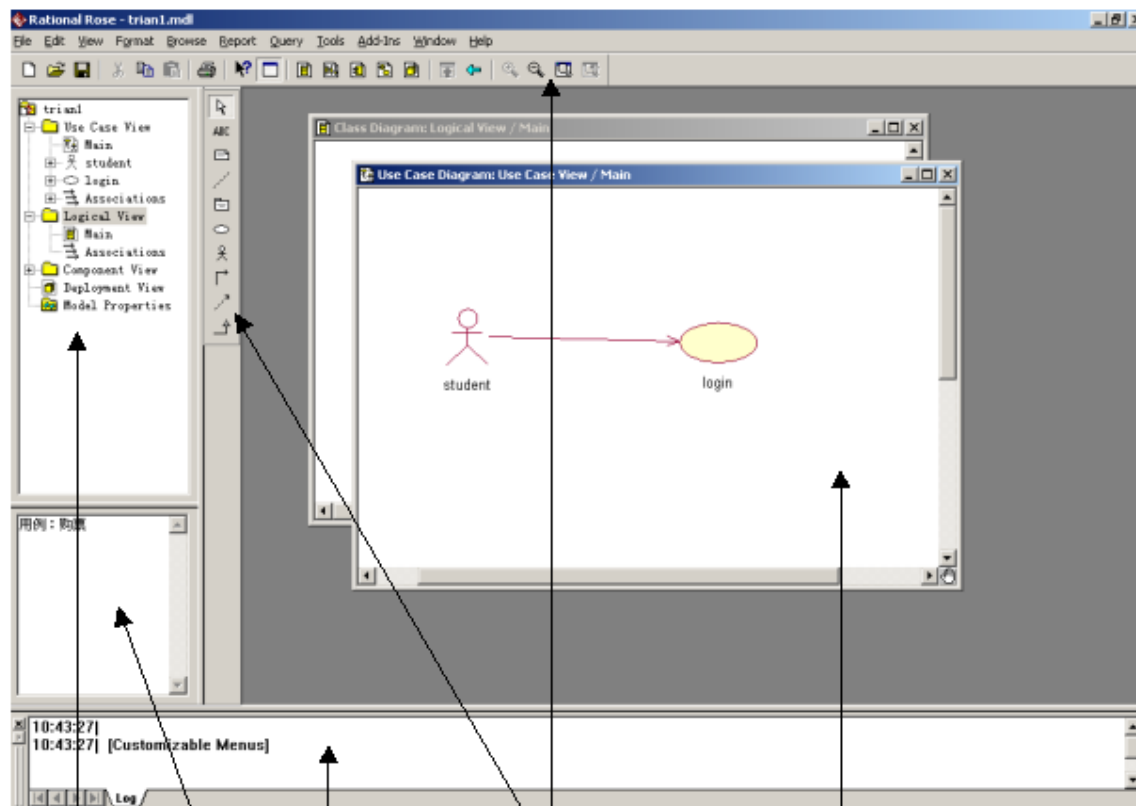




UML建模工具 Rational Rose



Rose体现出面向对象分析和设计的优势；**Rose**集中体现了**UML**的先进设计思想，通过一套统一的图形符号简洁有效地表达各种设计思想。



浏览器

文档窗口

日志

工具栏

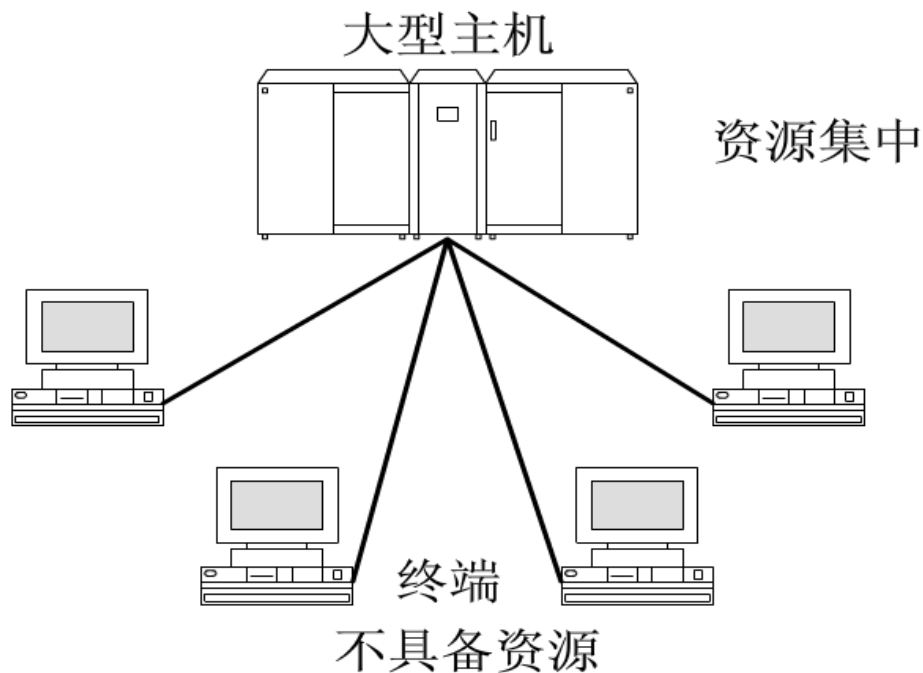
框图窗口



软件开发模式

1、集中式计算模式

在计算机诞生和应用初期，计算所需要的数据和程序集中在一台计算机上，用户通过终端登录到主机进行操作，通常被称为**主机/终端模式**，也叫集中式计算模式。





软件开发模式

1、集中式计算模式



计算能力和数据存储能力强
系统维护和管理费用较低
数据存储管理方便、安全性好

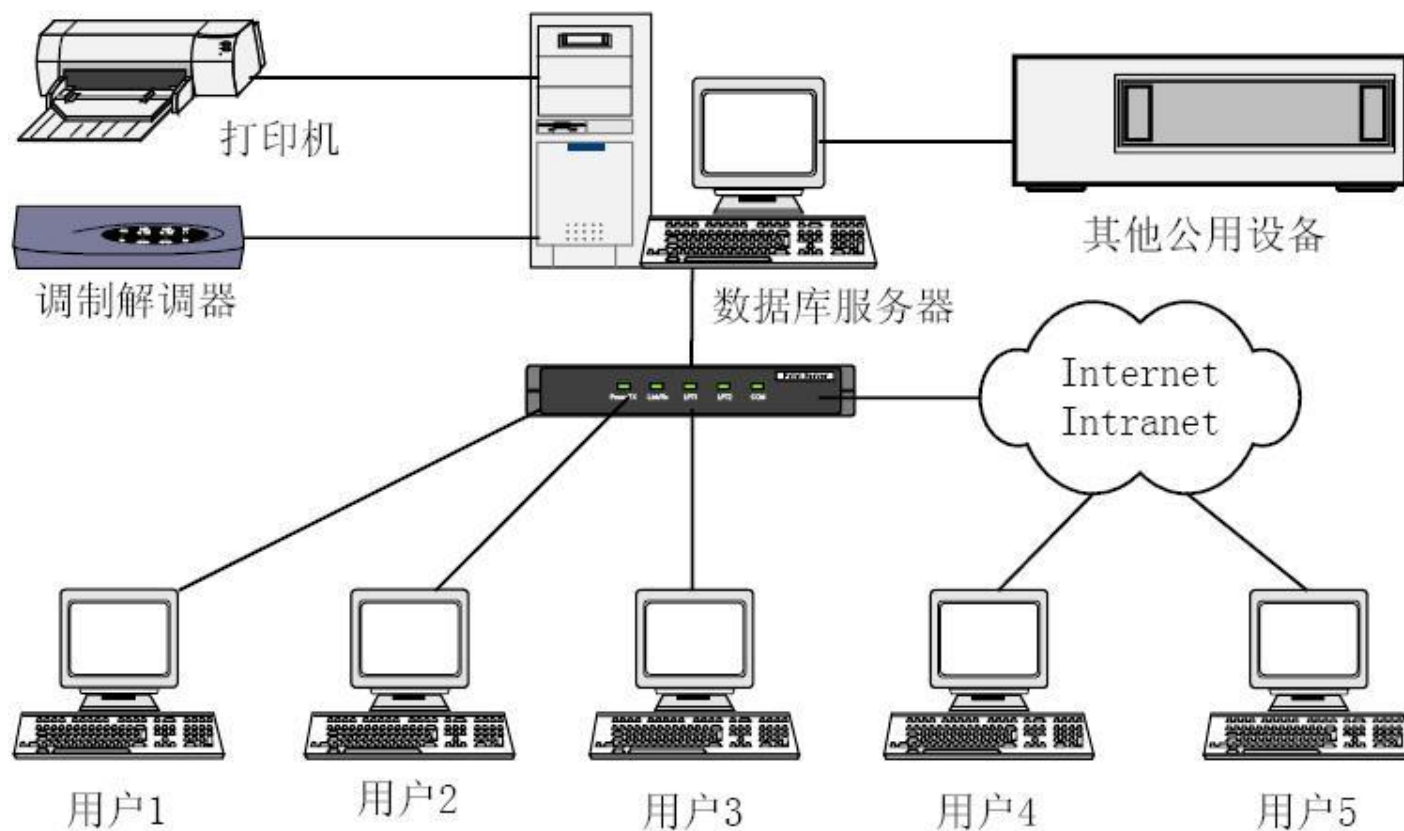
可移植性差
网络负载大
资源利用率低
大型机的初始投资较大



软件开发模式

2、客户/服务器（C/S）计算模式

C/S计算模式将应用一分为二：前端是客户机，一般使用微型机算机；后端是服务器，可以使用各种类型的主机。





2、客户/服务器（C/S）计算模式

优点：

充分利用两端硬件环境优势，将任务合理分配到客户端与服务端
充分发挥客户端PC的处理能力，服务器运行数据负荷较轻
通过异种平台集成，能够协调现有的各种IT基础结构
安全、稳定、速度快，且可脱机操作

缺点：

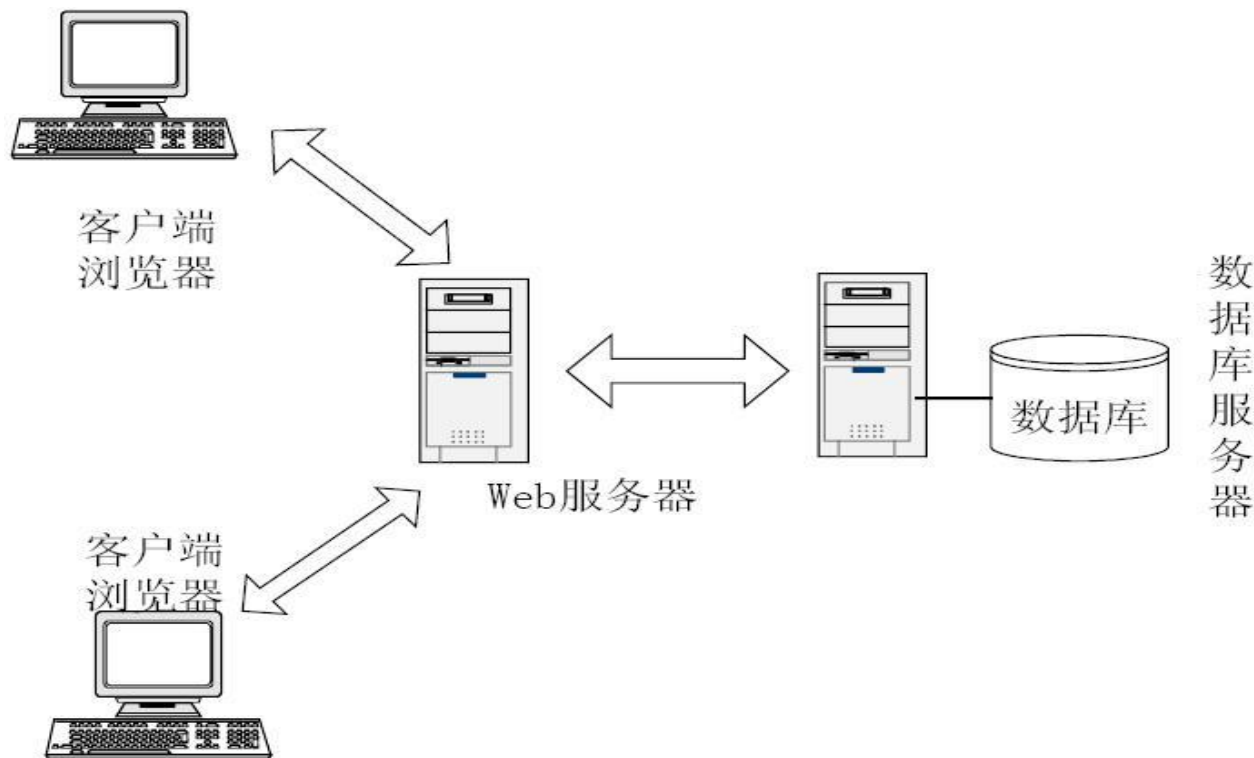
它必须在客户端安装大量的应用程序（客户端软件）
客户端安装、调试、维护、升级的成本高、任务量大
主要业务逻辑在客户端增加安全隐患
开发成本较高，移植困难



软件开发模式

3、浏览器/服务器（B/S）计算模式

B/S计算模式下，用户工作界面是通过浏览器来实现，极少部分事务逻辑在前端（**Browser**）实现，但是主要事务逻辑在服务器端（**Server**）实现。





3、浏览器/服务器（B/S）计算模式

优点：

单一的访问点，用户可以在任何地点使用系统，方便、快捷、高效

用户可以跨平台以相同的浏览器界面访问系统

减轻了系统维护与升级的成本和工作量

简化了客户端电脑载荷

降低用户总体成本

缺点：

无法进行脱机应用，需要稳定的网络连接和后台服务器正常运行

受制于 HTML ，无法使用丰富的效果展示数据，用户体验糟糕

应用服务器运行数据负荷重



软件开发模式

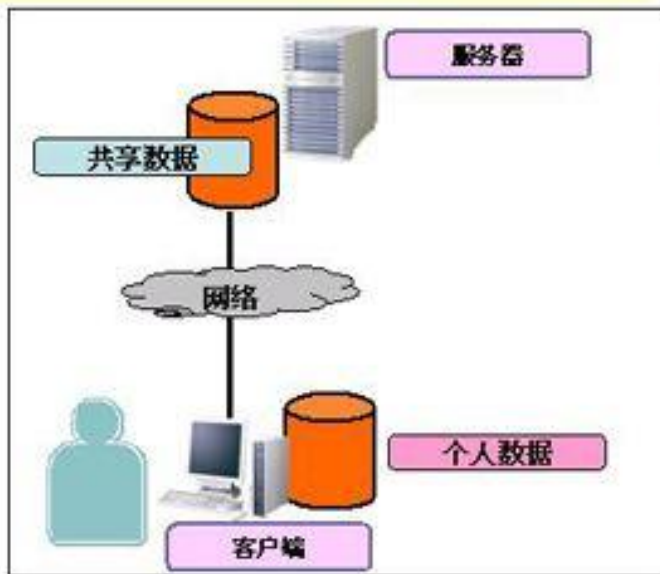
4、胖客户端模式与瘦客户端模式

C/S模式客户端集中了大部分的应用逻辑，被称为胖客户端模式；

B/S 模式客户端只有一个浏览器，应用逻辑集中在服务器上，被称为瘦客户端模式。

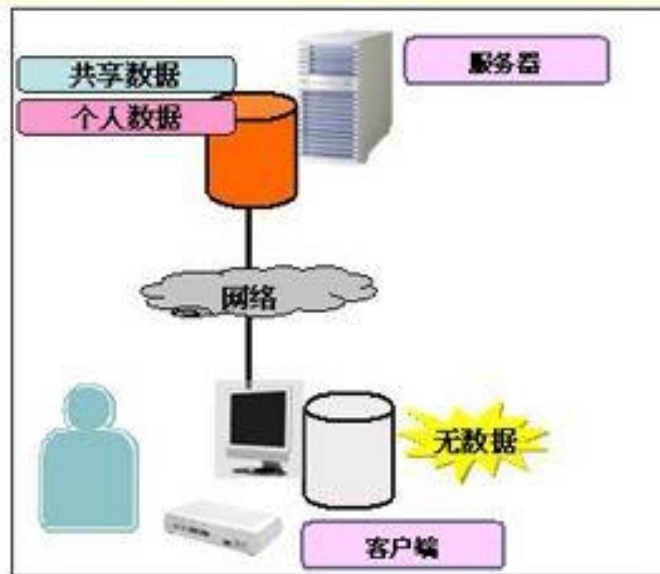
传统 PC

服务器和客户端都有数据存储



Thin client 瘦客户机系统

所有的数据都存储在服务器端






5、富客户端模式


- ✎ 从 C/S 到 B/S ，两者均受限于技术本身分别发展成为重客户端和重服务器端的模式，因此产生了富客户端模式（Rich Client ），结合了胖客户端和瘦客户端的各自优势并克服其固有缺点。
- ✎ 富客户端对应用程序提出新的要求-富因特网应用程序（Rich Internet Applications, RIA），利用富客户端技术RIA集成了桌面应用的交互性和传统Web应用的部署灵活性。
- ✎ 富客户端提供可承载已编译客户端应用程序（以文件形式，用 HTTP 传递）的运行环境，客户端应用程序使用异步C/S架构连接现有的后端应用服务器，这是一种安全、可升级、具有良好适应性的新的面向服务模型。



5、富客户端模式

富客户端是相对于胖客户端及瘦客户端而言的，区别在于“富”，而“富”的含义主要包含两方面的内容：

 **丰富的用户界面：**富客户端能够给用户带来丰富的界面体验。在富客户端模型中将界面分解成许多既可以和用户直接交互又可以和服务器进行通信的小单元模块。这将应用程序的设计从以一个个相对独立的页面为中心转移到以组件为中心，使客户层的设计提升到一个新的层次，并会使客户层变得更加灵活。

 **丰富的数据模型：**富客户端可接受或处理不同类型的数据，包括图像、语音、文本、视频等格式，异步**C/S**架构使用户使用的感觉就好像程序不需要和服务器进行通信或者只是偶尔才需要进行通信。



5、富客户端模式

富客户端技术仍在不断完善中，但并不意味着会取代HTML。相反它将进一步扩展浏览器功能，使之提供更加高效和友好的用户接口。许多RIA都在浏览器中运行，甚至它本身就是HTML的一部分，因此HTML将继续保持其原有的角色。另外，由于富客户端技术可以支持运动的图象、视频、音频、双向的数据通信和创建复杂的窗体，它为创建应用程序用户接口提供了一个高效而完善的开发环境。