建模工具 Rose 的学习

作者：天雨

在随着面向对象的程序设计的广泛应用，可视化编程遍地开花的今天，编程工作人员的地位一再被动摇，早以不再作为开发中的主流，而软件工程的应用以作为软件开发的质量和效率的重要见证已越来越被重视，针对早期的结构化建模工具的明显不足，ROSE 吸取众多建模工具的优点，排除其不足，采用面向对象的成熟技术和双向工程的技巧，为提高软件开发的效率，保证软件开发的质量和可维护性作出了巨大的贡献。

软件工程概述

软件开发是一套关于软件开发各阶段的定义、任务、作用的建立在理论上的一门工程学科。它为解决软件危机，指导人们利用科学、有效的方法来开发软件，提高及保证软件开发的效率和质量取到了一定的作用。

软件开发过程：需求分析（开始阶段）--概要设计（静态结构）--详细设计（动态结构） --编码-测试-维护

结构化模型设计方法

A．E-R 图（实体关系图）

实体：客观存在并可区分的事物。

属性：实体所具有的某种特性，一个实体可以有多个属性。

关系：实体之间的对应关系，可分为 1：1 联系、1：n 联系、m：n 联系

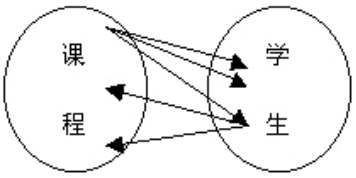


图 1.1 学生-课程关系图

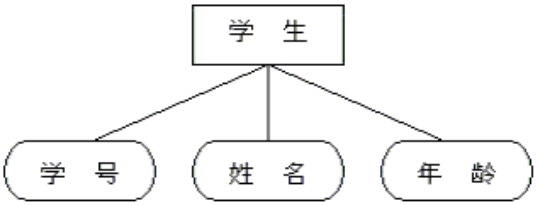


图 1.2..学生**属性**图

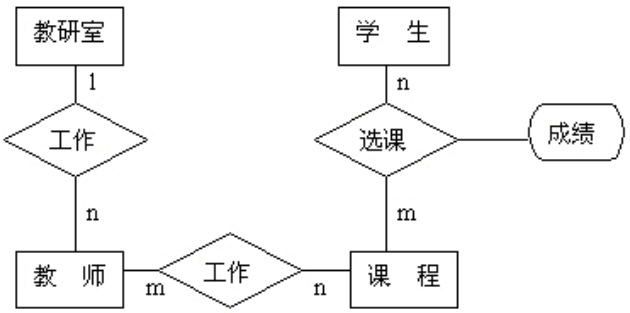
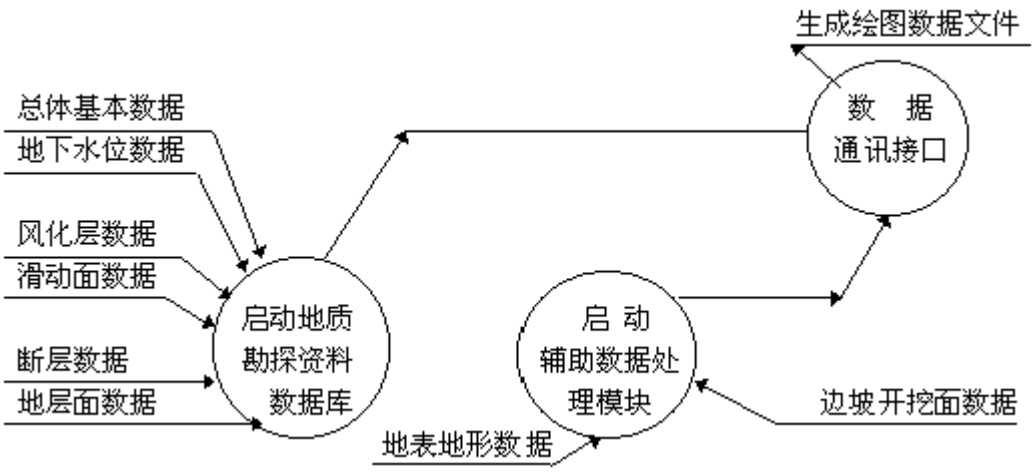
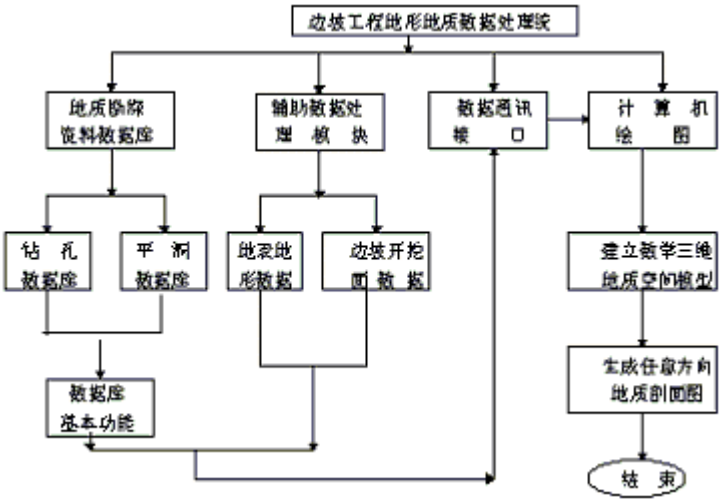


图 1.3. 实体关系图

B．数据流图



C．功能模块图



结构化模型的不足

传统的结构化模型的设计所建立的模型不能反应源代码，与程序设计脱节。模型与代码几乎没什么关系。这样的模型不能生成代码，代码更不能生成模型，模型大多是画给领导看或拿出作掩盖。所以不能保证软件的质量，更不易软件的维护，没什么约束力也没有检测的标准，它的弊端是显于易见的。

面向对象的模型设计方法

定义：利用面向对象方法，把应用程序分成许多小块（对象），这些对象是独立的，然后组合这些对象，建立程序。

特点：包装、继承、多态。

常用的建模工具：PlayCase, Rational ROSE, Computer Association BPWin , Computer Association ERWin, Oracle Designer/2000, Sybase PowerDesigner

UML 语言概述

定义：一种面向对象的统一建模语言。

作用：帮助我们对软件系统进行面向对象的和建模。

核心：类，类之间的关系。

建模：通过将用户的业务震需求映射为代码，保证代码满足这些需求，代码能方便地回溯需求，这个过程叫建模。

ROSE **建模工具**

定义：是一种分析和设计面向对象的建模工具。

作用 利用 ROSE **这个工具,我们可以建立用** UML **的软件系统的模型,面目可以自动生成和维护** C++、

JAVA、VB、PB、ORACLE 等语言和系统的代码。

核心：七大框图

1. Use-Case Diagrams（用例框图）
2. sequence diagram （顺序框图）
3. Collaboration diagram （协作框图）
4. Class diagram （类框图）
5. State Transition diagram （状态框图）
6. Component diagram （组件框图）
7. Deployment diagram （扩展框图）

下面结合软件工程知识、利用 ROSE 建模工具，本人在开发电力部们的"110KV 标准设计图文管理系统"中所用到的 ROSE 模型及对 ROSE 在开发实践中的剖析

一、 需求分析阶段

任务：建立用户需求和功能模块，确定系统中的角色和使用案例。利用 ROSE，生成角色，使用案例和生成用例图

所用到的框图：

1.Use-Case Diagrams：**显示使用案例（表示系统功能）与角色（人或系统）间的交互。如下图：**

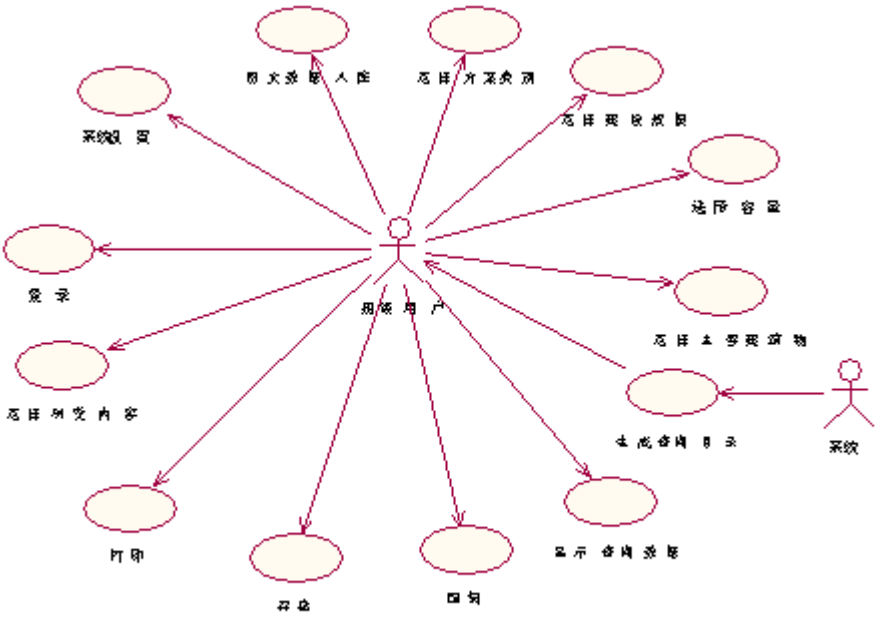


图 1.1 超级用户主用例图

Use Case（**用例）：在不展现一个系统或系统内部结构的情况下，对系统或系统的连贯的功能单元的定义和描述。**

角色：使用软件的人或外部系统本身。

2. sequence diagram

按时间先后顺序，从上到下分析使用案例，确定案例的处理流程。如下图：

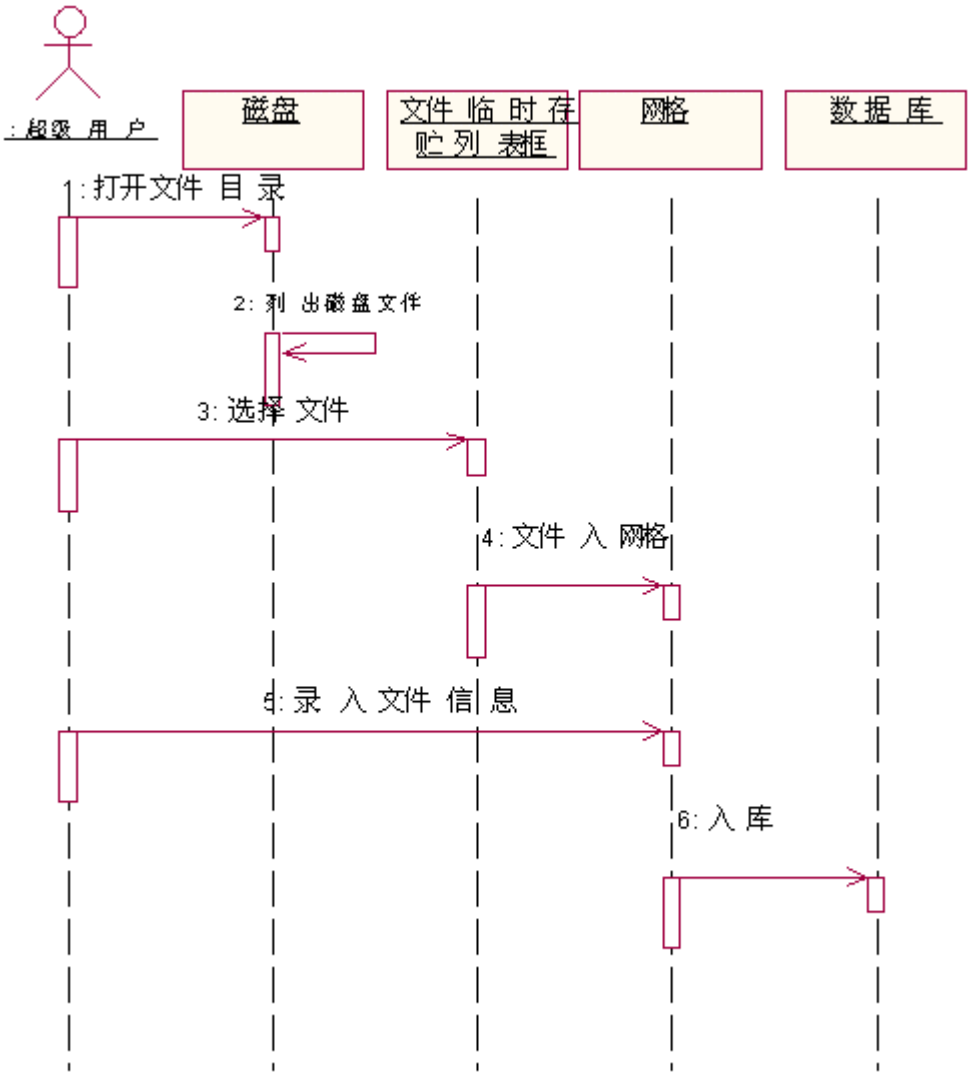


图2.2.1 入 库 顺序 图

3 Collaboration diagram ：

确定对象之间的关系的处理过程的分析流程。如下图：

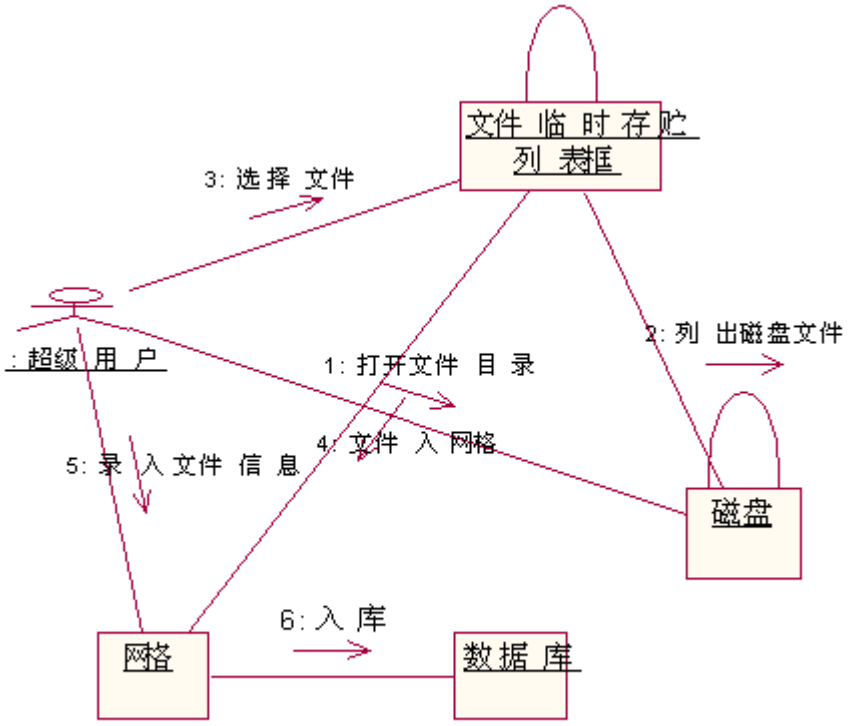


图2.2.2 入 库协 作图

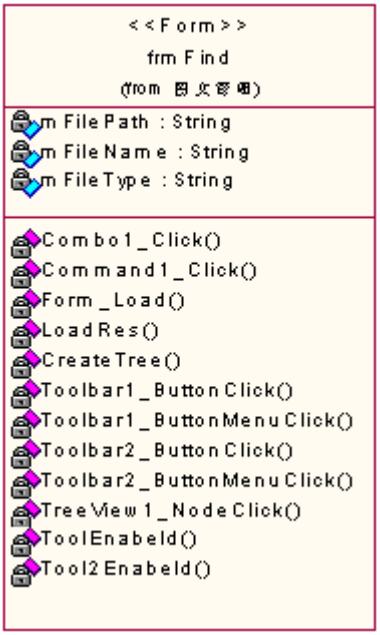
二、 概要设计阶段

任务：通过分析 Use-Case Diagrams **，得到所用到的类，分析这些类的属性、操作和它们之间的关系。**

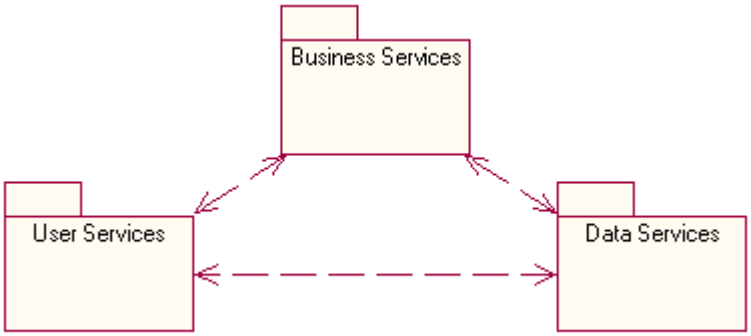
所用到的框图：

1.Class Diagrams.

显示系统中类与类之间的交互。



2.**包：具有一些共性的类组合在一起的图。**



三、 详细设计阶段

任务：细化和个性 Use-Case **的描述 ，如类的操作和对象之间的消息相对应，填充参数及复杂的类的设计。**

所用到的框图：

1.Class Diagrams

2.State Diagrams:**显示一个对象从生成到删除的生命周期。**

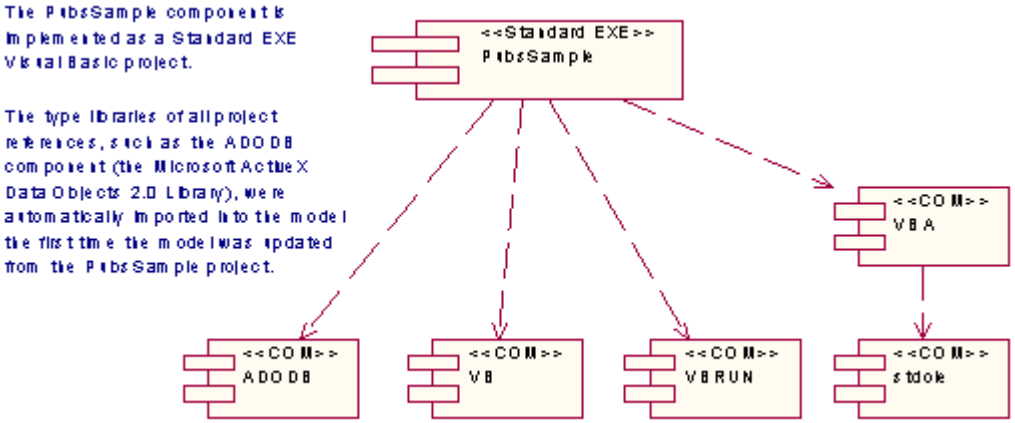
四、 编码和测试阶段

任务：进行软件的开发和测试，生成组件框图。

组件：表示代码的物理模块。

组件框图：表示系统中的组件及相互依赖性。

Delpoyment Diagrams:**显示网络中的物理布局和各种组件的位置。**



双向工程

1.**生成代码：根据选择开发应用程序的语言生成对应的程序的代码。**

步骤：检查模型-生成组件-将类映射组件-设置代码生成属性-选择类、组件和包-生成代码

2.**逆向转出工程：根据选择开发应用程序的语言生成对应的程序的代码。**

步骤：检查模型-生成组件-将类映射组件-设置代码生成属性-选择类、组件和包-生成代码

建模工具-ROSE 的介绍

在这个面向对象应用程序开发不断变化的时代，在合理时间内开发和管理高质量应用程序变得越来越困难。为了面对这种挑战，制定出每个公司都能使用的通用对象模型语言，统一建模语言（UML**）。**UML **是信息技术行业的蓝图，是详细描述系统结构的方法。利用这个蓝图，我们越来越容易建立和维护系统，保证系统能适应需求的改变。一个系统的模型建得好，就为满足用户需求、保证系统的稳定性和质量、提高系统的扩展性打下了良好的基础。**ROSE **是用** UML **快速开发应用程序的工具之一，它是一个面向对象的建模工具。**

UML 统一建模语言

UML，Unified Modeling Language，统一建模语言，是一种面向对象的建模语言，它的主要作用是帮助我们对软件系统进行面向对象的描述和建模，它可以描述这个软件开发过程从需求分析直到实现和测试的全过程。UML 通过建立各种类、类之间的关联、类/对象怎样相互配合实现系统的动态行为等成分（这些都称为模型元素）来组建整个模型，刻画客观世界。UML 提供了各种图形，比如 Use Case 图、类图、顺序图、协作图、状态图等，来把这些模型元素及其关系可视化，让人们可以清楚容易的理解模型。我们可以从多个视角来考察模型，从而更加全面的了解模型，这样同一个模型元素可能会出现在多个图中，对应多个图形元素。

由视图 view，图 diagram，模型元素 model element 和通用机制 general mechanism 等几个部分组成 .视图是表达系统的某一方面特征的 UML 建模元素的子集，由多个图构成，是在某一个抽象层上，对系统的抽象表示.图是模型元素集的图形表示，通常为弧（关系）和顶点（其他模型元素）相互连接构成的.模型元素代表面向对象中的类、对象、消息和关系等概念，是构成图的最基本的常用概念.通用机制用于表示其它信息，比如注释、模型元素的语义等。另外，它还提供扩展机制，使 UML 语言能够适应一个特殊的方法（或过程），或扩充至一个组织或用户.

UML 是用来描述模型的，用模型来描述系统的结构或静态特征，以及行为或动态特征。从不同的视角为系统的构架建模，形成系统的不同视图（VIEW）。

用例视图(use case view)，强调从用户的角度看到的或需要的系统功能，是被称为参与者的外部用户所能观察到的系统功能的模型图；

逻辑视图(logical view)，展现系统的静态或结构组成及特征，也称为结构模型视图(structural model view)或静态视图（static view）；

并发视图(concurrent view)，体现了系统的动态或行为特征，也称为行为模型视图（behavioral model view）、动态视图(dynamic view)；

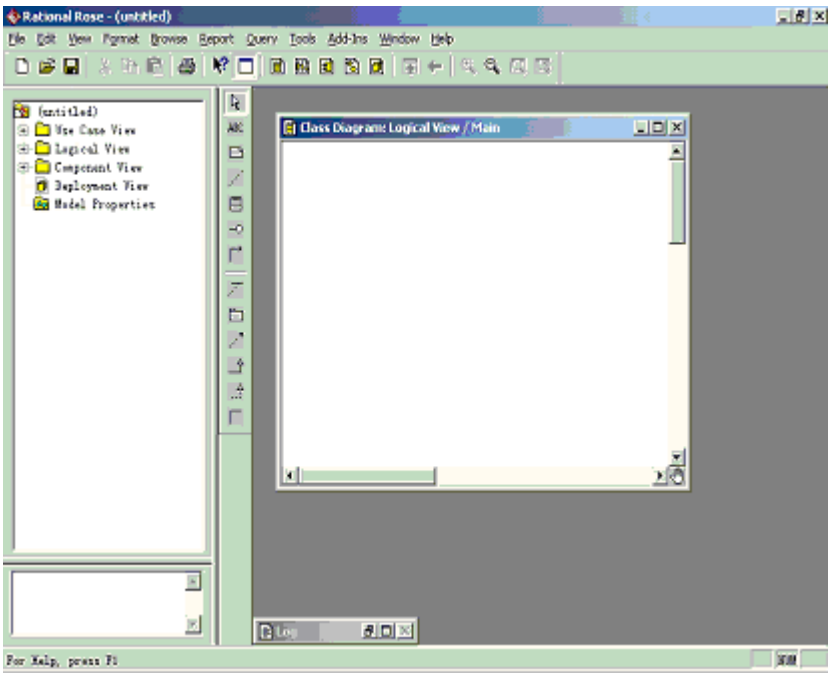
组件视图(component view)，体现了系统实现的结构和行为特征，也称为实现模型视图(implementation model view) ；

配置视图(deployment view)，体现了系统实现环境的结构和行为特征，也称为环境模型视图（environment model view）或物理视图(physical view)。

建模工具 Rose 之游

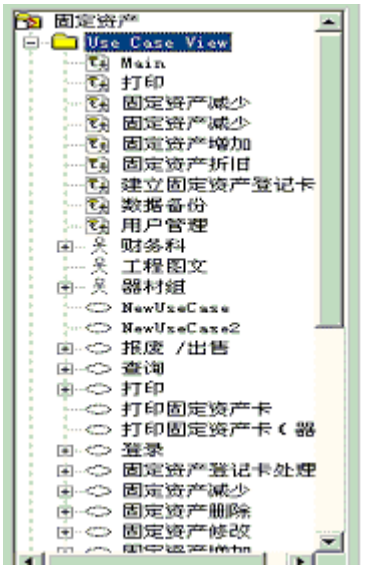
ROSE 是美国 Rational 公司的面向对象建模工具，利用这个工具，我们可以建立用 UML 描述的软件系统的模型，而且可以自动生成和维护 C++、Java、VB、Oracle 等语言和系统的代码。

ROSE 是个菜单驱动应用程序，用工具栏帮助使用常用特性。它的界面分为三个部分--Browser 窗口、Diagram 窗口和 Document 窗口。Browser 窗口用来浏览、创建、删除和修改模型中的模型元素；Diagram 窗口用来显示和创作模型的各种图；而 Document 窗口则是用来显示和书写各个模型元素的文档注释。



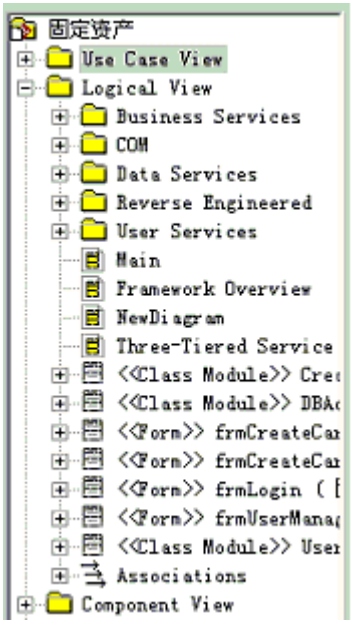
Rose 界面

Rose 模型的四个视图是 Use Case 视图 、Logical 视图、Component 视图和 Deployment 视图。每个视图针对不同对象，具有不同用途。Use Case 视图包括系统中的所有角色、案例和 Use Case 图，还包括一些 Sequence 图和 Collaboration 图。



Use Case 视图

Logical 视图关注系统如何实现使用案例中提到的功能。它提供系统的详细图形，描述组件间如何关联。除其它内容之外，Logical 视图还包括需要的特定类、Class 图和 State Transition 图。利用这些细节元素，开发人员可以构造系统的详细设计。



Logical 视图

Component 视图包括模型代码库、执行库和其它组件的信息。组件是代码的实际模块。Component 视图的主要用户是负责控制代码和编译部署应用程序的人。有些组件是代码库，有些组件是运行组件，如执行文件或动态链接库（DLL）文件。

Collaboration 图关注系统的部署，可能与系统的逻辑结构不同。整个小组都用 Collaboration 图了解系统部署，但用户是发布应用程序的人员。

Rose 的九种图

用例图 use case diagram，描述系统功能

类图 class diagram，描述系统的静态结构

对象图 object diagram，描述系统在某个时刻的静态结构

序列图 sequence diagram，**按时间顺序描述系统元素间的交互**

协作图 Collaboration diagram，**按照时间和空间顺序描述系统元素间的交互和它们之间的关系**

状态图 state diagram，**描述了系统元素的状态条件和响应**

活动图 activity diagram，**描述了系统元素的活动**

组件图 component diagram，**描述了实现系统的元素的组织**

配置图 deployment diagram，**描述了环境元素的配置，并把实现系统的元素映射到配置上**

根据它们在不同架构视图的应用，可以把 9 **种图分成：**

用户模型视图：用例图

结构模型视图：类图、对象图

行为模型视图：序列图、协作图、状态图、活动图（动态图）

实现模型视图：组件图

环境模型视图：配置图