水利模型库管理系统的设计与实现

学　　院　　软件学院

专　　业　　软件工程

年　　级　　2004 级

姓　　名　　黄海峰

指导教师　　张坤龙

2008 年　6 月　15 日

摘　　要

水利模型库管理系统是水利决策支持系统开发运行平台的一个子项目，它为水利决策支持系统开发运行平台中所有与模型相关的操作提供服务，是水利决策支持系统开发运行平台运转不可或缺的一部分。

论文在认真分析对水利模型库管理系统的需求的基础上，设计了水利模型库管理系统的功能模块和数据文件。在功能模块方面，除了模型的查询、增加、删除和修改，还包括了模型的调试和组合。在数据文件方面，定义了单一模块文件、组合模块文件、和用户分类信息文件。

水利模型库管理系统的实现主要采用了 XML 的相关技术。例如，模型被封装成 XML 文件，组合模型通过 XML 文件生成，模型运行脚本也是 XML 格式的等等。水利模型库管理系统的实现除了提供窗口界面支持最终用户对水利模型进行一系列管理操作之外，还提供函数界面供水利决策支持系统开发运行平台调用模型并进行数据处理。

关键词：决策支持系统；模型库管理系统；C#；XML；

ABSTRACT

Water Resources Model Base Management System is a sub-project of Water Resources Decision Support System developing and running Platform, it provides all kinds of services that related to model operations, it's an indispensable part for WR-DSSRP's running.

Basing on conscientious requirement analysis, this paper designed the function module and data files of the WR-MBMS. On function module, besides searching, adding, deleting and modifying models, it also support debugging models and combining models. On data files, it defined single model file, combined model file and user classing info file.

XML-related technologies are often used in implementing the system. For examples, packaging the model as XML documents; generating combined models through XML documents; model running scripts in XML format and so on. Water Resources Model Base Management System provided a Windows Form which let the users can do a series of management work, also it can be invoked by the WR-DSSRP to transact data.

Key words：Decision Support System；Model Base Management System；C#；XML

目　　录

第一章　　绪论 ……1
1.1　开发背景 ……1
1.2　相关概念 ……3
1.3　模型库管理系统的现状与发展 ……5
1.4　论文的组织结构 ……6
第二章　　水利模型库管理系统的需求分析 ……7
2.1　用例需求 ……7
2.2　功能需求 ……8
2.3　使用场景需求 ……8
2.4　软件硬件需求 ……10
第三章　　水利模型库管理系统的设计 ……11
3.1　接口设计 ……11
3.2　功能模块的设计 ……12
3.3　数据文件的设计 ……13
3.4　类的设计 ……19
第四章　　水利模型库管理系统的实现 ……22
4.1　模型编辑模块的实现 ……22
4.2　模型查找模块的实现 ……24
4.3　模型组合模块的实现 ……24
4.4　模型管理模块的实现 ……27

4.5　模型调用模块的实现 …… 27
第五章　　总结与展望 …… 30
参考文献 …… 32
外文资料
中文译文
致谢

第一章　　绪论

1.1　开发背景

水利模型库管理系统是水利决策支持系统开发运行平台（WR_DSSDRP, Water Resources Decision Support System developing and running Platform)的一个组成部分，它的设计与实现都以 WR_DSSDRP 为依据，并为之服务。从整个 WR_DSSDRP 的外部来讲，水利模型库管理系统是 WR_DSSDRP 的一部分用户界面，用户通过模型管理界面进行模型库的相关操作；从整个 WR_DSSDRP 的内部来讲，模型库管理系统是 WR_DSSDRP 的一个服务函数库，WR_DSSDRP 系统通过调用模型库管理系统所提供的服务函数，进行相关模型处理操作。

1.1.1　水利决策支持系统开发运行平台

决策支持系统（DSS, Decision Support System）有两种主要的开发策略：利用决策支持系统生成器（DSSG, Decision Support System Generator）开发，或者利用决策支持系统工具（DSST, Decision Support Tools）开发。

决策支持系统生成器可以被看作是一个通用决策支持系统。实际上，由于决策的复杂性，人们不可能去建造一个通用的决策支持系统，而只可能建造一个生成器，然后利用这个生成器来生成各种专用的决策支持系统。利用决策支持系统生成器，决策支持系统研制者可以根据决策者的要求、环境和任务，迅速构造一个专用的决策支持系统，从而大大缩短决策支持系统的研制周期，并降低决策支持系统的研制费用[1]。

决策支持系统工具是用来构成决策支持系统或者决策支持系统生成器的硬件或者软件单元，例如某种程序设计语言、某种操作系统、某种数据库管理系统等等。决策支持系统工具处于决策支持系统技术的最低层次。因此，虽然它具有很大的灵活性和适应变化的特征，但是用它来直接开发决策支持系统还是存在一些困难。

决策支持系统工具既可用于决策支持系统生成器的开发，也可用于决策支持系统的开发。决策系统支持工具、决策支持系统生成器和决策支持系统的关系如图 1-1 所示。

WR-DSSRP 是一个水利决策支持系统生成器，命名为水利决策支持系统开发运行平台（WR-DSSRP)。

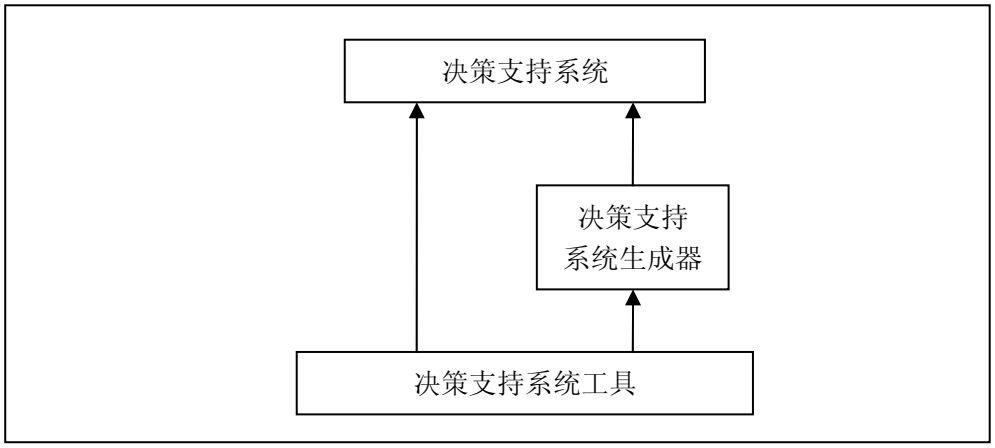

图 1-1　　DSST、DSSG 和 DSS 的关系

水利决策支持系统开发运行平台首先是一个水利决策支持系统的开发平台。作为一个水利决策支持系统生成器，水利决策支持系统开发运行平台在概括各专用水利决策支持系统共性的基础上，向水利决策支持系统开发人员提供丰富的二次开发接口。通过平台提供的二次开发接口，水利决策支持系统开发人员可以定制自己的用户接口、自己的数据库、和自己的模型，象搭积木一样形成一个既具有共同行业特征又具有具体系统个性的专用水利决策支持系统。

其次，水利决策支持系统开发运行平台又是水利决策支持系统的运行平台。水利决策支持系统开发运行平台的用户除了水利决策支持系统开发人员以外，还包括水利决策人员。对于水利决策人员，水利决策支持系统开发运行平台执行水利决策支持系统开发人员编写的代码，表现得像一个专用水利决策支持系统。水利决策支持系统开发运行平台将建立功能强大的用户接口控件库、平台独立的数据库存取服务、和符合标准的业务模型库。除此以外，该平台还要考虑如何解决新老应用系统的平稳过渡问题，以便能够充分保护用户已有的投资。

1.1.2　水利模型库管理系统在 WR_DSSDRP 中的作用

如图 1-2 给出了整个 WR_DSSDRP 的体系结构，从中可以看出水利模型库管理系统（WR-MBMS，Water Resource-Model Base Management System）是水利决策支持系统开发运行平台不可缺少的组成部分。在进行水利决策时，平台需要根据用户的要求调出系统中已有的基本模块和基本模型，生成一个针对某一特定问题的求解过程，即专用模型[2]。模型管理应当具有存储和动态建模的功能。

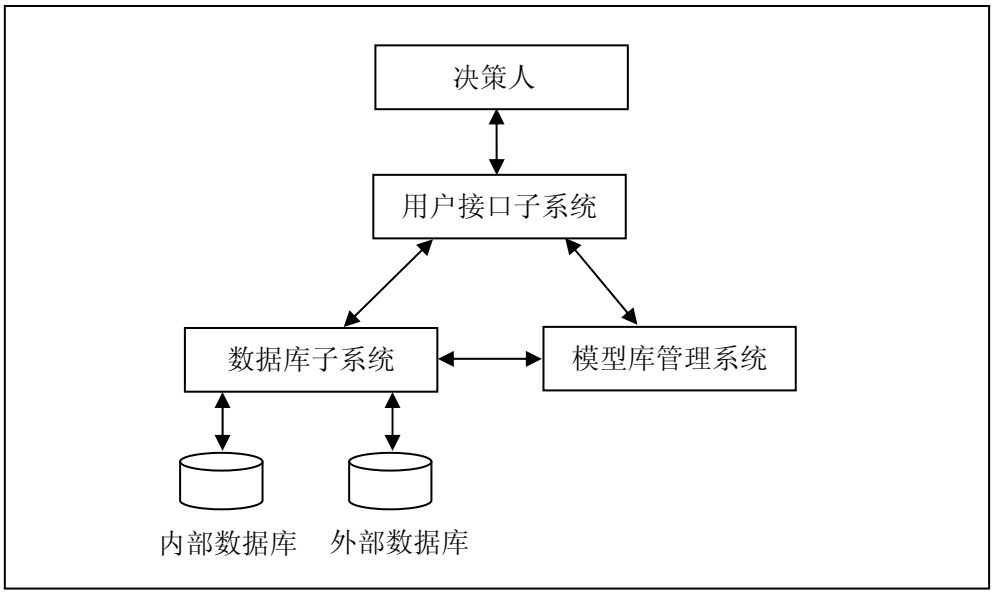

图 1-2　　WR_DSSDRP 的体系结构

模型库子系统在水利决策支持系统中占有重要地位。水利决策人员使用水利决策支持系统不是直接依靠数据库中的数据进行决策，而是依靠模型库中的模型进行决策。数据库是为决策提供数据能力或资料能力，而模型库则是给决策提供分析能力的部件，它给决策者以通过推理、比较、选择来分析解答整个问题的能力因此，可以认为，水利决策支持系统是由“模型驱动的”。

综上所述，为了对模型库进行集中控制和管理，WR_DSSDRP 必须有一个强有力的模型库管理系统来进行各项管理。模型库管理系统是为生成模型和管理模型提供一个用户友好环境的计算机软件系统。用户可以利用模型库管理系统灵活地访问、更新、生成和运行模型，从而更加充分，方便的运行 WR_DSSDRP。

1.2　相关概念

下面简单介绍一下水利模型库管理系统中所涉及的一些相关概念。这有助于读者对模型库管理系统能建立一个初步的印象。

1.2.1　模型的概念

什么是模型？尽管没有统一的定义，但是我们可以通过以下两种描述来了解模型的概念：

a) 如果我们把现实世界中的某些事物叫做一个现实原型，那么模型就是对这种现实原型的一种抽象或模拟，这种抽象或模拟不是简单的“复制”，而是强调反映原型的本质，略去原型中的次要因素。因此，模型既反映原型，又不等于原型，或者说它是原型的一种近似。

b) 模型是以某种形式（如数学表达式、工作流程）对一个系统的本质属性

的描述，以揭示系统的功能、行为及其变化规律。一般来说，模型就是将一组数据通过一个或一系列方法转换为另一组数据的一个系统。由于从软件实现的角度来看，一个方法就是一个程序模块，因此模型由许多程序模块组成。为了方便起见，这些程序模块也称为基本模块。

水利模型可以分为两个层次：基本模型和专用模型。基本模型通过组合基本模块来构成，通常为支持频繁的操作而设立。专用模型通过组合基本模块和基本模型来构成，通常用于为一个水利决策问题提供一个决策方案，一个水利决策支持系统有多个专用模型。基本模块、基本模型、专用模型的关系如图 1-3 所示。

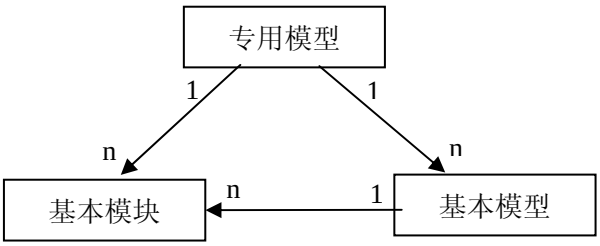

图 1-3　　基本模块、基本模型和专用模型

水利决策支持系统开发运行平台中的模型都使用程序来表示，包括输入、输出和算法在内的完整程序就表示一个模型。

1.2.2　模型库的概念

模型库是提供模型存储和表示模式的计算机系统[3]。在这个系统中，还包括可以对模型进行提取、访问、更新和合成等操作的软件系统，这个软件系统称为模型管理系统。模型库并不追求模型本身的完美和复杂，也不认为模型可以解决一切决策问题，它的目标是在决策者需要的时候按它的要求构造模型。模型库所构造的模型不一定很复杂，但一定要符合实际，建模时特别注意推理能力和决策者的干预。由于模型的生成、修改、更新、删除、连接是经常性的操作，所以模型库的管理成为非常重要的工作。

模型库是决策支持系统的共享资源，它有一些具有支持不同层次的决策活动的基本模型，其中有一些为支持频繁操作的单一模型；还有一些用于生成新模型和基本要素。这样模型库就是一个“产生”模型的基地，而不是预先建立的模型集合，通过模块的组合，可以使模型灵活地变更。因此，动态性是模型库的一个基本特征，也是研究模型库生成技术的前提。

1.2.3　模型库管理系统的概念

决策支持系统的应用领域十分泛，各个不同的领域对模型库管理系统的要求也不一样。目前被广泛接受的对模型库管理系统的定义为：设 M={M1,M2...Mn}

为模型的集合，P 为施于 M 上的操作的集合，则<M, P>为一模型库管理系统。也可以说模型库管理系统是一个支持模型生成、存储、维护和应用的软件系统。

决策支持系统概念中的模型管理管理系统通常意味着一个模型的集合和模型管理工具程序的集合。模型管理系统的主要功能包括：辅助用户建立起与特定问题相关联的模型；提供模型库管理功能，使用户可以方便地使用模型库中的模型；辅助用户对模型库进行增减、存取等操作；提供模型库与其它子系统的界面，使用户可以方便地将模型与相关的数据联系起来:辅助用户进行模型的创建、检索、选择、修改等工作。

1.3 模型库管理系统的现状与发展

1.3.1 模型库管理系统的现状

模型管理是决策支持系统的核心部分，也是近年来 DSS 研究中十分活跃的领域[4]。对模型的管理不可能只停留在模型的存储、显示、修改和输出上，而应能支持模型创建、链接、复合与重用。

- ✧ 传统模型管理系统的不足
  - (1) 没有采用统一开发平台。（使参数传递困难）
  - (2) 模型与数据不独立。(数据结构一变，模型要跟着变，输入复杂，如输入表、向量、数组，不得不编写大量专用程序)。
  - (3) 不支持模型的组合与重用。
  - (4) 不支持领域知识推理，模型操纵困难。
- ✧ 需要研究的问题有
  - (1)模型的表示
  - (2)模型的自动生成
  - (3)模型的复合
  - (4)模型的自动选择
  - (5) 模型的重用等。

1.3.2 模型库管理系统的发展

随着计算机科学技术的不断发展，模型库管理技术的发展经过了由易到难的三个阶段[5]：

1) 模型库管理技术的初期阶段。比较成熟的模型以子程序的形式汇编成册或存放在计算机内，用户可以根据自己的需要调用它们。在这种情况下用户同时也是程序员，他的任务是编写子程序。模型可以多层嵌套，一个解决小问题的模型在调试成功后，可以存放在模型库内，作为解决更大问题的子模型。
2) 模型软件包。软件包阶段对大量的较通用的模型由专职人员预先编制好

模型程序，模型统一挂在交互菜单下。用户就不用自己再编制模型程序，而是在人机交互菜单的驱动下，调用所需要的模型。模型之间是相对独立的，模型使用的数据仍放在各自的数据文件中。这比上一阶段前进了一步，它建立了大量模型的简单组织机构。这才真正进入了模型库管理阶段，因为模型软件包降低对用户的要求，但缺乏灵活性，这就是模型库管理系统出现的背景和客观要求。

3) 模型库管理系统。这一阶段，建立了大量模型的有效组织机构。模型不简单的挂在菜单下面，而是在管理系统下，有效的对模型进行存储、修改、查询、调用。模型之间不仅可以相互独立，而且可以相互结合。模型库管理系统是随决策支持系统的需要而发展起来的，它使模型管理技术提高到一个新水平上。

1.4 论文的组织结构

本文从水利决策支持系统开发运行平台的背景出发，逐步讨论水利模型库管理系统的设计与实现。

第二章叙述了对水利模型库管理系统的需求，从不同方面阐述了用户需要的是怎样的一个系统。

第三章论述了水利模型库管理系统的设计方案。

第四章讨论了水利模型库管理系统的具体实现，和运行效果。

第五章是对已完成工作的总结和对未来工作的展望。

第二章　　水利模型库管理系统的需求分析

本章阐述了水利模型库管理系统（WR-MBMS）的用户需求，分别从用例分析、功能需求、使用场景、运行环境等方面描述系统应满足的要求。通过本章，读者可以对 WR-MBMS 有一个初步的认识，并可以清楚 WR-MBMS 是一个什么样的系统。

2.1　用例需求

用例是系统提供的功能块，换句话来说用例演示了人们如何使用系统。通过用例观察系统，能够将系统实现与系统目标分开，有助于了解最重要的部分——满足用户要求和期望，而不会沉浸于实现细节。通过用例，用户可以看到系统提供的功能，先确定系统范围再深入开展项目工作。所以根据用户的需求我们判断出水利模型库管理系统(WR-MBMS）所面向的用户应分为三大类：

1.最终用户，即水利决策支持系统的使用者。

2.模型库编写人员，即水利计算方面的专业人员。

3.水利决策支持系统中的其他模块。

其具体用例分析如图 2-1 所示。

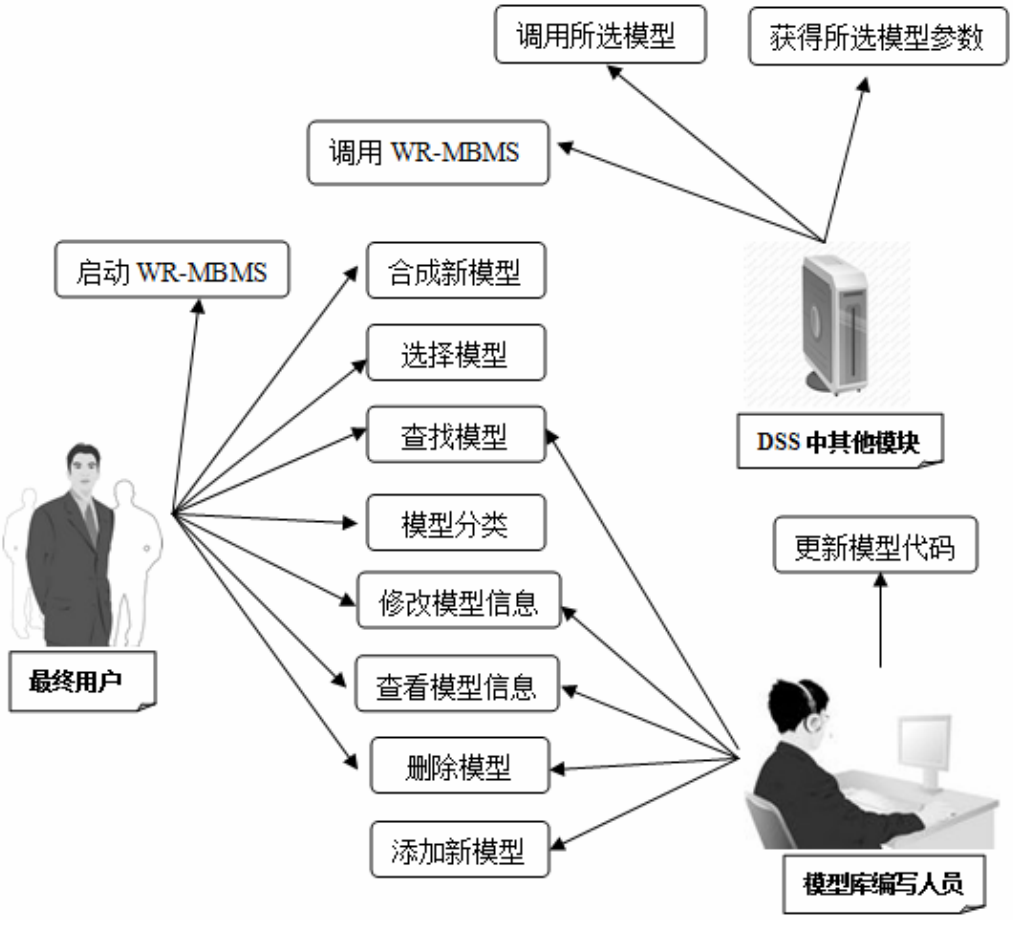

图 2-1　　水利模型库管理系统 WR-MBMS 用例图

2.2 功能需求

水利模型库管理系统需要实现的主要功能描述如下：

1) 添加新模型：模型库编写人员可将最新编写出的模型文件添加到模型库中，并按提示输入模型的相关必要信息，诸如函数接口，功能描述等。
2) 删除模型：对于已经不符合计算要求或者存在 bug 的模型文件，可以将其从模型库中删除
3) 查看模型信息：显示模型的基本信息和属性，帮助用户了解模型的特性，从而能选择适合的模型进行下一步操作。
4) 修改模型信息：用户可以修改模型的描述信息，使之更为详尽具体。用户也可以为模型添加或修改备注，便于自己的使用。
5) 模型分类：用户可以根据自己的需要，建立模型分类，将具有类似特点的模型存储于同一分类中，使模型库的层次更加清晰，便于管理和使用。
6) 查找模型：用户可以按照模型名称，创建时间，属性值等条件在模型库中查询到所需的模型。
7) 选择模型：用户选择最终要使用的模型。
8) 调试模型：对于已加入到 WR-MBMS 中的模型，为确定其正确性，系统应提供模型的调试功能。
9) 合成新模型：用户可以通过对现有进行拼装组合，来生成一个新模型。
10) 启动模型库管理系统：用户在使用水里决策支持系统的时候，可以启动 WR-MBMS，来进行有关模型的操作。
11) 调用模型库管理系统（WR-MBMS）：水利决策支持系统中的其他模块可以利用 WR-MBMS 提供的函数接口，获取需要的模型相关信息。
12) 调用所选模型：其他模块可以调用已选择的模型，通过这个模型进行水利方面的一些计算和数据处理。
13) 获得所选模型参数：其他模块可以获得所选模型的一些参数，用于信息显示，数据分析等方面的操作。
14) 更新模型代码：用新版本的模型替换旧版本模型。

2.3 使用场景需求

水利模型库管理系统(Water Resource-Model Base Management System—WR-MBMS）完成后所满足的使用场景应如图 2-2 所示

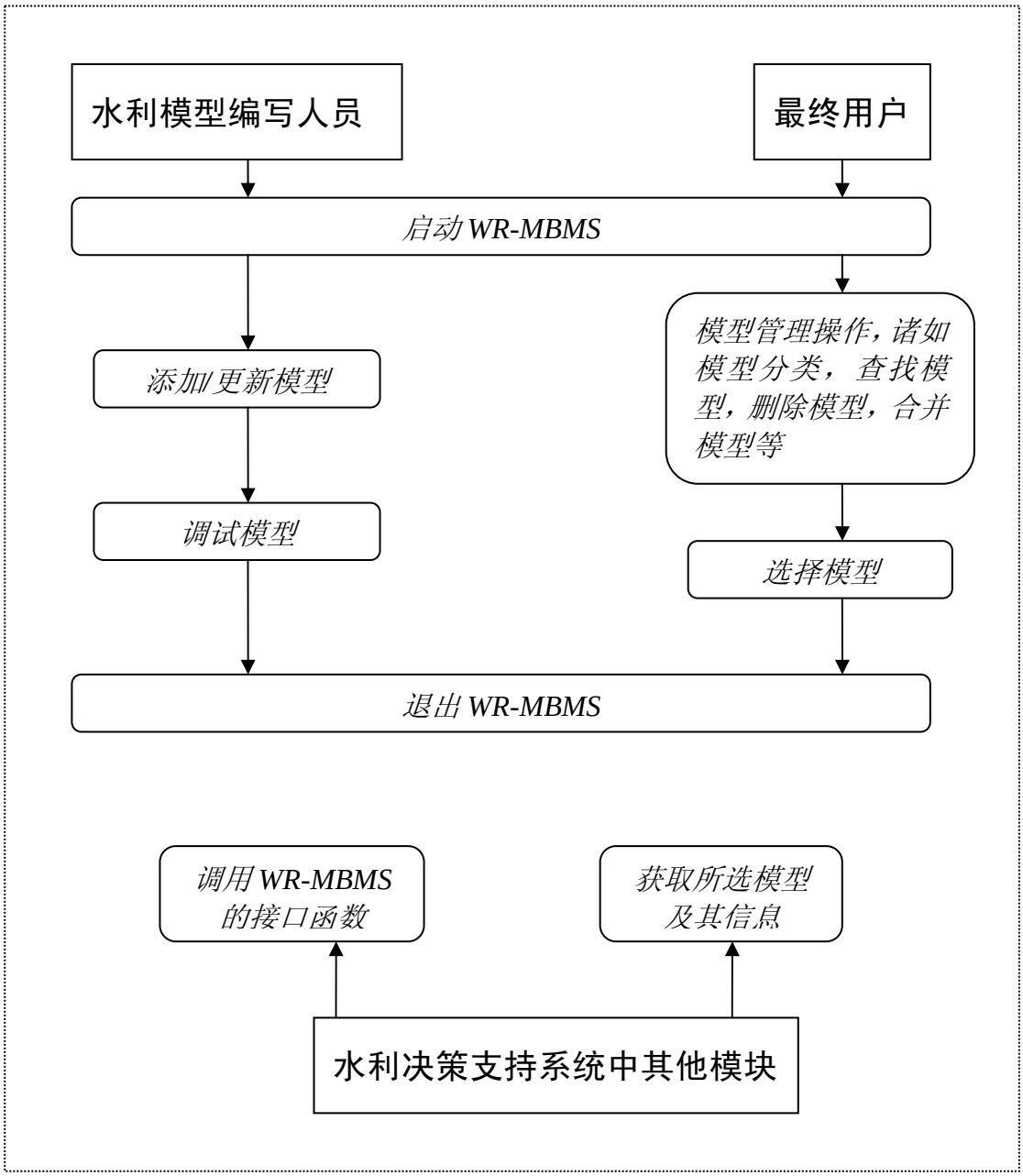

图 2-2　　水利模型管理系统使用场景图

使用场景应满足三类用户的需求，这三类分别是水利模型编写人员，最终用户以及 DSS 中的其他模块。

1)　水利模型编写人员：

水利编写人员是水利模型库中所有模型的编写者。他们的主要的工作是将自己编号的模型加载到 WR-MBMS 中，所以水利编写人员的使用步骤按照需求应如下：

i.　首先启动模型库管理系统。

ii.　添加新的模型文件或者更新原有的模型文件。

iii.　调试模型，确定其正确性（可选操作）。

iv. 退出模型库管理系统。

2) 最终用户：

最终用户是指水利模型管理系统的最终使用人员。他们的主要工作是使用 WR-MBMS 来进行模型管理，从而计算相关水利数据。因此最终用户的使用步骤按照需求应如下：

i. 首先启动模型库管理系统。

ii. 然后进行一些模型库管理操作，主要有模型分类、模型查找、删除模型、合成新模型、修改模型信息、查看模型信息等。

iii. 选择要调用的模型，为下一步的数据处理和计算做准备。（可选操作）。

iv. 退出模型库管理系统。

3) 水利决策支持系统中其他模块：

其他模块可以调用模型库管理系统所提供的接口函数，通过接口函数进行相应操作。同时，其他模块也可以直接调用模型库管理系统所选择的模型，直接进行数据处理和计算工作。

2.4 软件硬件需求

根据需求分析，水利模型库管理系统采用的软硬件设施如下。

运行系统: 安装了.Net Framework3.5（.Net Framework 是微软的几个开发团队一起努力发展的成果，是一个可以用来快速开发、部署网站服务及应用程序的开发平台。这个架构是两个项目的结果，第一个项目的目的是用来改善 windows 作业平台上的程序开发；第二个项目则是制作一个以发展服务软件为目标的开发平台[6]。）的 WINDOWS 系统。

运行硬件环境：普通 PC 机

开发系统：WINDOWS XP/ VISTA

开发环境：Microsoft Visual Studio 2008

开发语言：C#

开发模式：Windows 窗体项目

第三章　　水利模型库管理系统的设计

3.1　接口设计

WR-MBMS 的未来扩展其实在很大程度上取决于接口如何设计。我们知道，在软件工程范畴内，“接口”有两种众所周知的含义：其一则指的是人与软件之间的交互界面。我们把这种人和软件之间的接口称作“用户界面”,也就是“UI”。 其二是指软件本身的狭义“接口”，比如各种软件开发 API 等。

下面将从这两个方面来设计 WR-MBMS 的接口，即人机交互接口（即上面提到的用户界面）以及 WR-MBMS 接口函数（及上面所说的 API）。

3.1.1　人机交互接口设计

首先应设计出易于用户操作并且简洁美观的用户界面，以此作为人机交互的接口。用户界面设计在漫长的软件发展中一直没有被重视起来。其实软件界面设计就像工业产品中的工业造型设计一样，是产品的重要买点。一个友好美观的界面会给人带来舒适的视觉享受，拉近人与电脑的距离。界面设计不是单纯的美术绘画，它需要定位使用者、使用环境、使用方式并且为最终用户而设计，是纯粹的科学性的艺术设计。检验一个界面的唯一标准就是最终用户的感受。所以界面设计要和用户研究紧密结合，是一个不断为最终用户设计满意视觉效果的过程。

经过多方面考虑，最终决定 WR-MBMS 的用户界面风格采取普通 windows 程序样式。它包含一个主窗体以及若干个弹出对话框。模型的主要操作都在主窗体中完成，主窗体的大体布局设计如图 3-1 所示。

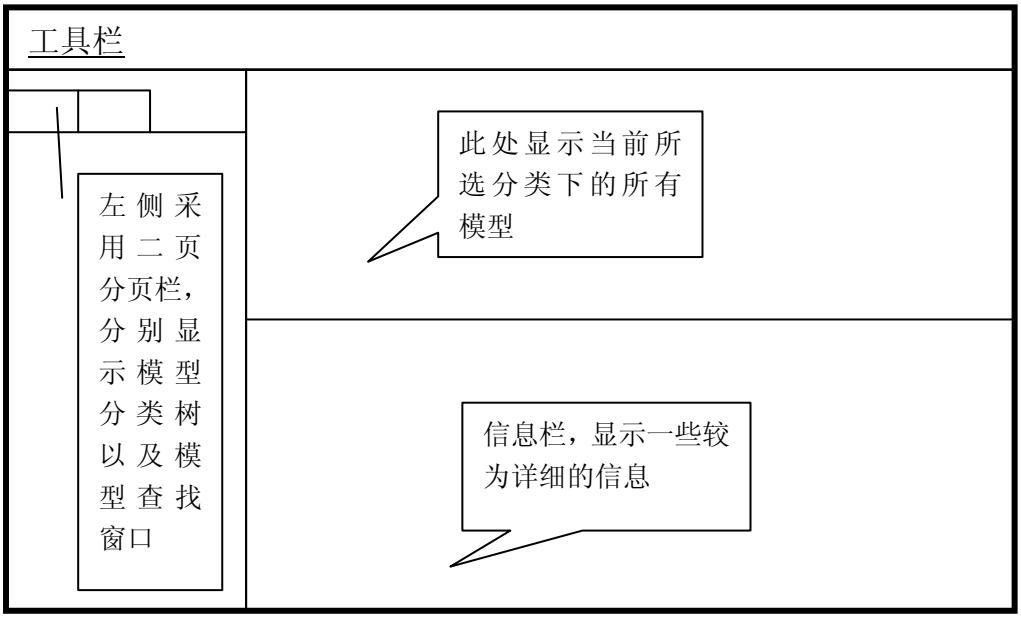

图 3-1　　水利模型管理系统主窗体设计图

采用此种设计方案的主要原因有两个：

1) 符合用户的操作习惯。由于 WR-MBMS 是一款运行在 windows 系统上的软件，所以应保持与 windows 系统相同的操作风格，这样用户的使用适应期可以将到最低。如图 3-1 所示，左侧显示模型分类树，类型之间以树的形式表示它们的层次关系。点击某个类型（即模型分类树中的一个节点）后，在右侧会显示此类型下的所有模型文件。这种设计风格与 windows 系统中的资源管理器如出一辙，用户使用起来不会感觉陌生。
2) 界面简洁，便于用户使用。不论是模型信息显示还是模型组合操作，所有需要显示信息的部分都由右侧的信息栏负责，这样避免了大量窗口的弹出，给用户带来麻烦。

3.1.2 接口函数设计

WR-MBMS 除了提供用户界面之外，还应提供一些接口函数，为水利决策支持系统其他模块以及 WR-MBMS 本身提供服务。接口函数的设计采用单一静态类的方法，即将所有的接口函数封装到一个类中，并且是一个静态类，命名为 ModelService。采用这种设计方案的好处有以下几点：

1) 采用静态类，在调用接口函数时就不用进行实例化，这样不但减少了编码的数量，而且提高了代码的安全性。
2) 封装到一个类中，这样可以将所有的接口函数写在一个.CS 文件中，不仅便于管理，而且方便日后编译成.dll 文件。
3) 可以声明静态变量，这些静态变量可以作为公用参数，被其他代码访问。如果不是在静态类中，每次要想获取这些变量的值，就必须先实例化这个类。静态变量可以节省内存，因为它是所有对象所公有的。因此，对多个对象来说，静态变量成员只存储一处，供所有对象共用。静态变量成员的值对每个对象都是一样，但它的值是可以更新的。只要对静态变量成员的值更新一次，保证所有对象存取更新后的相同的值，这样可以提高时间效率。

接口函数还应考虑到可扩展性的问题，也就是说，接口函数之间要做到彼此独立，互不相干。这样日后如果某个接口函数更新了，也不会影响到其他的函数。

3.2 功能模块的设计

WR-MBMS 按照各自所负责的工作差异，分为以下几个模块：模型编辑，模型组合，模型查找，模型调用以及模型分类。下面将进一步设计每一个模块具体负责的操作。

✧ 模型编辑模块

应实现的功能包括：打开已有模型，显示其具体信息；编辑已有模型，修改其中的某些模型信息；删除模型，将不需要的模型从系统中删除；添加模型，用

户可以通过添加向导，将新的模型添加到系统中；更新模型相关文件，用户可以更新模型所对应的计算模型文件。

✧ 模型组合模块

应实现的功能包括：添加模型间关系，确定哪两个模型之间的哪对变量有输入输出关系；生成组合模型文件，获取所有的模型间关系，生成组合模型文件；检查组合模型，通过拓扑排序，以及其他一些限制条件来判断组合模型是否符合逻辑条件。

✧ 模型查找模块

应实现的功能包括：条件查询，通过一组或一个查询条件来选出符合要求的模型；查询功能界面，提供一个方便用户输入查询条件的查询窗口。

✧ 模型调用模块

应实现的功能包括：根据模型文件获取调用模型的相关信息；提供模型调用的接口函数；运行模型。

✧ 模型管理模块

应实现的功能包括：对模型进行分类管理；模型分类树的实现，分类树节点的添加，删除，拖放；点击某一分类节点，显示此节点下的所有模型；改变模型的所属分类；分类信息的保存和导入。

3.3 数据文件的设计

数据文件是整个 WR-MBMS 的核心内容，主要的数据文件包括用户分类信息文件，单一模型文件，组合模型文件。它们都采用 XML 技术来组织。扩展标记语言 XML 是一种简单的数据存储语言，使用一系列简单的标记描述数据，而这些标记可以用方便的方式建立。之所以水利模型库管理系统要采用 XML 进行数据存储，主要是考虑到了它有以下几个优点[7]：

a) XML 通过使用一个可扩展标记集提供文档内容的更准确描述。

b) XML 可用标准化语法来验证文档内容。

c) XML 使用户与应用程序之间文件交换更容易。

d) XML 支持高级搜索。

e) XML 将文档结构与显示内容分开。

f) XML 改进用户响应、网络负载和服务器负载。

g) XML 支持 Unicode。

h) XML 支持文档间的高级链接。

下面将阐述每一种数据文件的具体格式设计，并给出相应的 XML 文件举例，使读者能清楚的了解每一个文件在 WR-MBMS 中的作用。

3.3.1 单一模型文件

单一模型文件（.mod 文件）是对一个计算模型中某一函数的封装，除了提供一个由水利专业人员所编写的计算模型（一般为.dll 文件或.exe 文件）外，还包含对此计算模型中某一函数的信息描述，例如它的名称、编写时间、编写人、功能描述、所包含的参数、方法等。图 3-2 是.mod 文件的结构设计举例。

```
<model>
    <m_id>0</m_id>
    <m_name>模型名称</m_name>
    <filename>***.dll</filename>
    <uiname> UI 文件</uiname>
    <m_description>模型功能描述</m_description>
    <createdate>2008/5/24 0:00:00</createdate>
    <editdate>2008/5/24 0:00:00</editdate>
    <author>张三</author>
    <class_name>CLASS1</class_name>
    <method_name>METHOD1</method_name>
    <method_type>1</method_type>
    <Parameters>
        <parameter>
            <type>string</type>
            <p_description>参数描述</p_description>
            <p_name>p1</p_name>
            <pid>0</pid>
        </parameter>
    </Parameters>
    <Variables>
        <Variable type="string" name="V1" text="参数一" des=""
            constrain="约束表达式" default="默认值" InOut="In" />
    </Variables>
</model>
```

图 3-2　　.mod 文件的结构设计举例

其中<model>节点代表一个模型文件，其子结点所表示的含义如表 3-1 所示。

表 3-1　　<model>的所有子结点的涵义

节点名称	节点含义	节点名称	节点含义
<method_type>	模型类型	<m_name>	模型名称
<filename>	模型文件名称	<uiname>	模型 UI 文件名称
<m_description>	模型功能描述	<editdate>	模型修改时间
<createdate>	模型创建时间	<author>	模型作者
<class_name>	所调用类的名称	<method_name>	所调用方法的名称

需要注意的是在 WR-MBMS 中所指的模型，其实上就是一个功能函数的封装，封装后的结果就是对应的.mod 文件。因此每一个.mod 文件被看作是一个单独的数据处理模块，它只含有一项特定功能。这样设计的好处是保持了每个功能函数的独立性，不仅有利于.mod 文件的注册，也方便于模型的组合，以及模型信息的描述。

模型、.mod 文件、功能函数以及计算模型（一般为.dll 文件或.exe 文件）几个概念之间的关系如图 3-3 所示

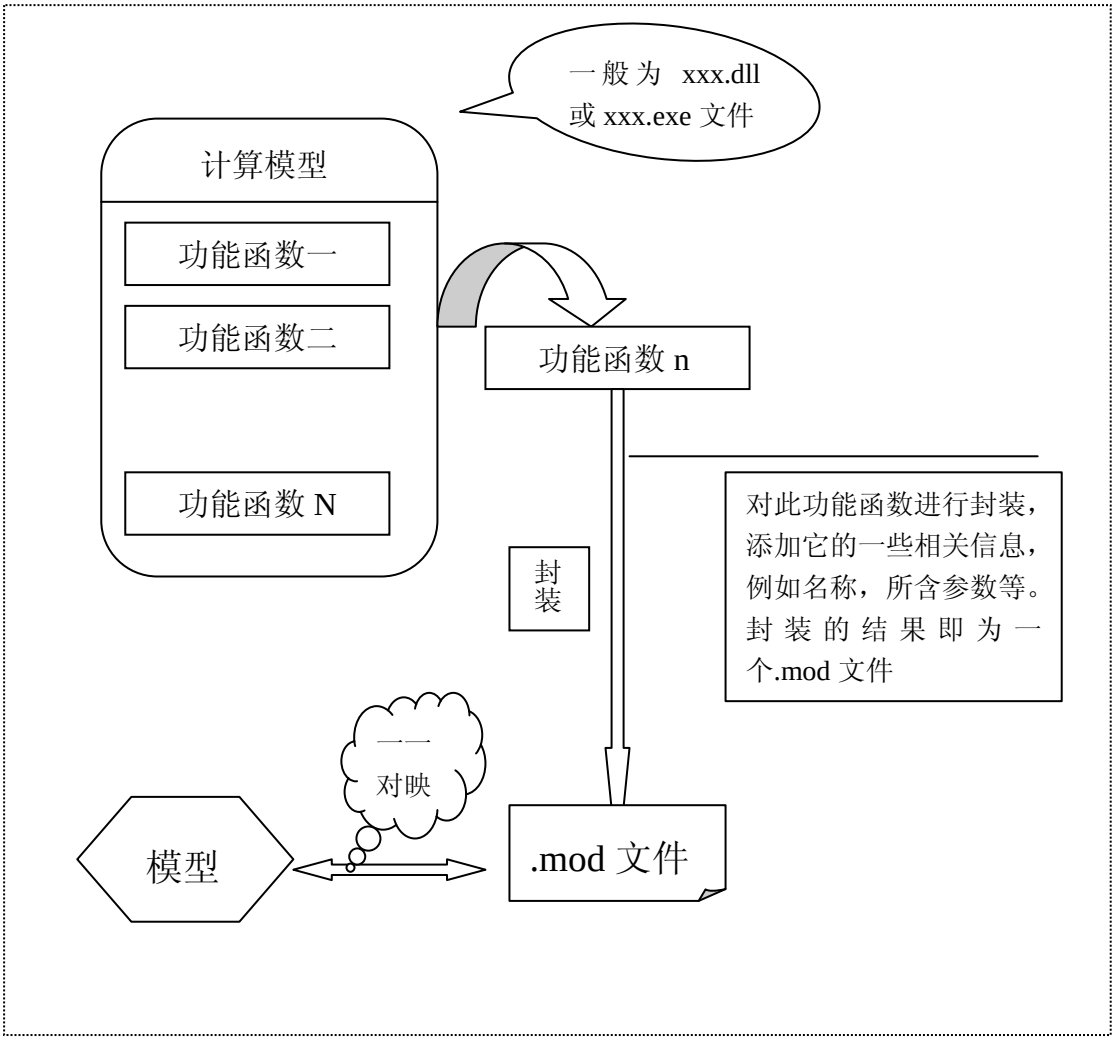

图 3-3　　模型，计算模型，功能函数之间的关系

由图3-3可以看出，一个水利人员所编写的计算模型可以包含多个功能函数。而本文中所指的模型（以.mod 文件的形式存在）实际上就是对其中某个功能函数的封装。也就是说，一个计算模型有可能被封装成多个 WR-MBMS 中的模型。

.mod 文件的结构设计也考虑到了模型的调用问题。.mod 文件中涉及到模型调用的节点如表 3-2 所示。

表 3-2　.mod 文件中涉及模型调用的节点说明

节点名称	用途说明
<parameter>	调用参数，代表一个在模型调用过程中必须赋值的参数
<pid>	调用参数编号，代表此参数在参数列表中的位置
<Variables>	变量，调用过程中不强制要求，但是也可以改变计算过程的参数
<method_type>	模型调用类型。
<class_name>	所调用功能函数所属的类
<method_name>	所调用的功能函数的名称

关于.mod 文件的其他具体细节，将在下一章结合实现过程进行进一步描述。

3.3.2　组合模型文件

组合模型文件（.cmod 文件）是用来描述一个组合模型信息的数据文件。之所以要设计这样一种文件，主要是考虑到了以下两个因素：

1) 用户每次按照步骤组装完成一个模型后，除了会立即利用此组合模型进行计算，也需要将当前的这种组合关系保存起来，从而方便日后再次使用。
2) 由于 WR-MBMS 的模型组合是采用逻辑组合方式（最终的组合模型是利用 XML 语言描述内部组合关系的，实质上就是一个脚本文件）而不是物理组织方式（将组合模型所涉及到的所有子模型代码拼装起来，生成一个真实存在的最终模型），所以必须设计出一种能够描述子模型之间关系的数据文件，即.cmod 文件。

此外还需要特别澄清 WR-MBMS 中“模型间关系”这一概念。一般来说模型之间可以存在许多种关系，调用关系、并行关系、判别关系等。但在 WR-MBMS 中只支持一种模型间关系，即数据传递关系。如图 3-2 所示，组合模型 I 由模型 A,模型 B,模型 C 组成，它内部包含了两个关系，由 a 到 c，和 d 到 e 这个条线表示。模型 A 计算出的 a 值传递给模型 B，作为 c 的初始值。类似的，模型 B 计算出的 d 值传递给模型 C，作为 e 的初始值。这种关系就是数据传递关系。

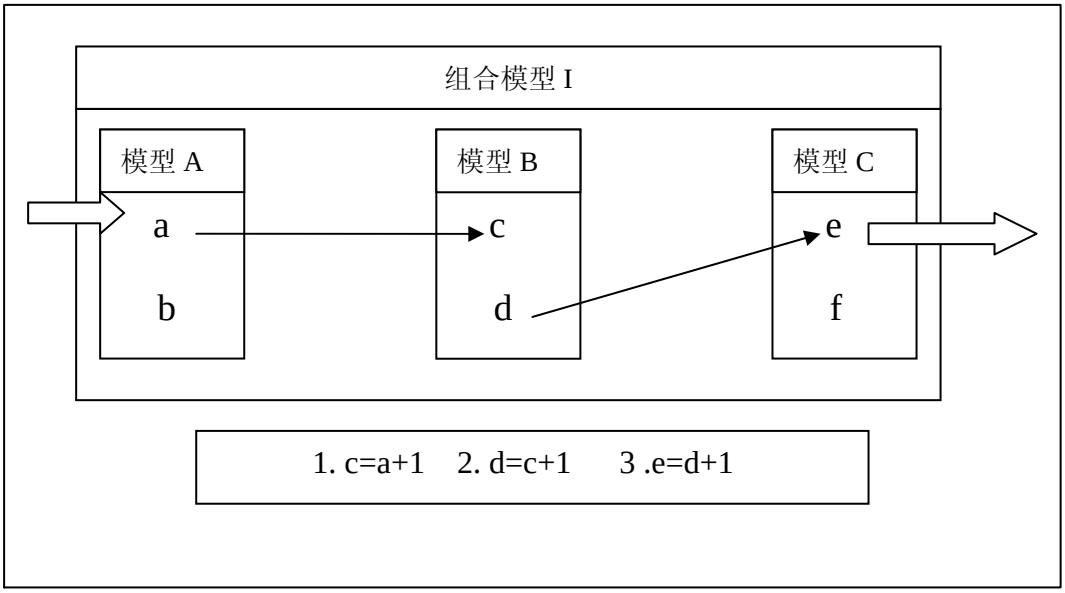

图 3-4 组合模型举例

如果不存在图 3-4 所示的组合模型 I，那么用户最少需要进行三次模型调用，才能完成这一系列操作。但只要有了组合模型 I，用户只需最初给定一个 a 值，组合模型便能只经一次调用就返回最后结果。可见，这种数据传递关系对于实现模型组合来说是必不可少的。

如图 3-5 是.cmod 文件的一个简单实例：

```
<con-model>
    <Relations>
        <Relation InMid="0" InVal="inputfile" OutMid="1"
                  OutVal="inputfile" Exsit="false" />
    </Relations>
    <InitModels>
        <InitModel mid="0" />
    </InitModels>
    <Sequence>
        <Step mid="0" no="0" />
        <Step mid="1" no="1" />
    </Sequence>
</con-model>
```

图 3-5 .cmod 文件举例

其中<Relations>节点表示组合模型中的所有关系，它的每个子结点<Relation>代表一组模型间的关系，<Relations>节点每个属性的含义如表 3-3 所

示。

表 3-3　<Relation>节点各属性的含义

属性名称	含义
InMid	关系中提供数据源的模型编号
InVal	关系中提供数据源的变量名称
OutMid	关系中接受数据源的模型编号
OutVal	关系中接受数据源的变量名称
Exsit	关系标志位

<InitModels>节点表示那些不需要其他模型提供数据源的模型，这些模型在组合模型被调用时最先运行。<Sequence>节点代表组合模型中各个子模型的执行顺序，它的子节点<Step>决定每一步执行哪个子模型。

3.3.3　用户分类信息文件

用户分类信息文件的扩展名为".cate"，它用于存储用户对模型文件分类的信息。在模型库中因为存在着数目庞大的模型文件，所以如果不加以分类，将使得用户在管理过程中缺乏条理性。此外，不是每个用户都需要所有的模型，所以用户专属的.cate 文件可以列出他所需的模型，而不是数以百计的所有模型。

.cate 文件的组织形式采取类似于 windows 中文件夹的组织形式，每个用户有一个属于自己的主文件夹，在这个主文件夹用包含了他自己的一些分类文件夹。其具体设计结构如图 3-6 所示

```
<root>
    <class>
        <id>0</id>
        <name>根节点</name>
        <parid>0</parid>
        <del>True</del>
    </class>
    <class>
        <id>1</id>
        <name>父分类一</name>
        <parid>0</parid>
        <del>False</del>
    </class>
    <class>
        <id>2</id>
        <name>子分类 1</name>
        <parid>1</parid>
        <del>False</del>
    </class>
    <class>
        <id>3</id>
        <name>父分类二</name>
        <parid>0</parid>
        <del>False</del>
    </class>
</root>
```

图 3-6　　用户分类信息文件举例

Class 元素代表一个分类，其中的 id 是分类编号，name 是分类名称，parid 是该分类所属的父分类编号，del 代表该分类是否可删除。对于用户主文件夹元素来说，它是不可以被删除的，因为他是所有其他分类元素的父分类。因此图 3-6 的显示效果应和图 3-7 一样。

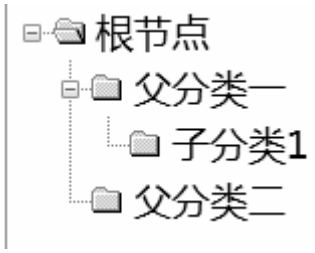

图 3-7　　用户分类文件的效果图

3.4　类的设计

为了利于编码的实现，在设计类的时候一定要做到结构合理、条理清晰。比

如说，应将类所包含的方法设计成简要的功能性单元，用它描述和实现一个不连续的类接口部分。理想情况下，方法应简明扼要。若长度很大，可考虑通过某种方式将其分割成较短的几个方法。这样做也便于类内代码的重复使用。

在设计类的过程中也要遵循以下几个原则[8]：

a) 单一职责原则：对一个类而言，应该仅有一个引起它变化的原因
b) 开放，封闭原则：软件实体（类，模块，函数等等）应该是可以扩展的，但是不可修改的。
c) 依赖倒置原则：高层模块不应该依赖于低层模块，二者都应该依赖于抽象；抽象不应该依赖于细节，细节应该依赖于抽象。
d) 接口隔离原则：不应该强迫客户依赖于他们不用的方法。

因此，在 WR-MBMS 中，鉴于功能的划分以及 C#编码的特点，设计出以下几个比较关键的类。

- ModelService 类：此类所包含的方法主要是一些公用的函数，例如添加模型函数，获取模型函数，查找模型函数等。此外 ModelService 类中还应包含一些参数，这些参数是全局变量，WR-MBMS 中的所有代码都可以获取它们的值（例如要声明一个变量 DataPath，此变量的值代表 WR-MBMS 数据的根目录，其他代码便可据此找到相应的数据）。ModelService 类以及它所包含的所有方法和变量都是静态型的，这样每次调用的时候无需再进行实例化。
- ModelType 类：这个类是用来描述模型信息的，换句话说，这个类的用途是将.mod 文件信息转化为可识别的 C#代码。如果将.mod 文件转化为 ModelTyp 类，在编码过程中，就可以使用 C#语言对模型进行诸如添加，删除，修改等操作。举例来说，如果想新建一个模型，只需在程序中编写 “ModelType MyModel=new ModelType()”。
- ModelType 所包含的一些类：为了更详尽的描述一个模型，就应该对 ModelType 类进行细化，所以它还应包括以下几个类的实例。分别是 Method 类，Variable 类，MethodCollection 类以及 VariableCollection 类。这些是用来描述模型的变量及方法的。
- CategoryManager 类：这个类主要负责模型的分类管理，前面已经提到 WR-MBSM 支持模型分类树，模型根据分类信息被组织成一个树形的结构。因此就需要 CategoryManager 提供下列功能：
  a) 添加节点——当用户添加一个新的分类时，在模型分类树上一个生成一个新的节点。同时更新树的层次关系以及用户分类信息文件。
  b) 修改节点——修改某个节点（分类）的名称。
  c) 删除节点——删除一个节点，同时删除此节点的所有子结点。

d) 收起/展开节点——将所选节点的所有已展开/收起的子结点收起/展开。

e) 移动节点——当某个节点被拖放到一个新的位置时，判断它现处层次，更新整棵分类树。如果此节点还包含子结点，则连同它的子结点一起随它移动。

f) 导出当前树的信息——将当前的模型分类树转换成用户分类文件，导出保存。

g) 导入一个树——根据已有的用户模型分类文件，建立一颗新的树。

✧ CDocManager 类：这个类主要是负责模型文件的显示。当用户点击某个模型分类树的节点时，CDocManager 类即时获取此节点所代表的类包含的所有模型，并将这些模型显示到窗口。用户打开某个模型时，CDocManager 类首先判断出被选择的是哪个模型，随后找到它的.mod 文件，并最终根据.mod 文件的信息将模型显示出来。

第四章　水利模型库管理系统的实现

4.1　模型编辑模块的实现

4.1.1　添加模型

在前面已经提到，WR-MBMS 中的模型是以.mod 文件的形式存在的，所以添加一个模型也就等价于生成一个对应的.mod 文件。

在 WR-MBMS 中，首先通过 PropertyGrid 控件来获取.mod 文件相关参数。如图 4-1 所示是 WR-MBMS 的添加新模型窗口。

图 4-1　添加新模型

PropertyGrid 列出了一个.mod 文件所需的所有信息，用户只需按照描述将信息逐一添入便可完成添加模型操作。而 PropertyGrid 应该列出哪些信息是由它所选定的对象决定的。这也是选择在此处使用 PropertyGrid 的主要因素，设想如果采用其他的方法，若模型有 N 组信息，那么界面上就要添加 N 个文本框，N 个文字标签，……。并且一但模型有新的信息项加入，这个界面就需要重新设计。而采用 PropertyGrid 就不会有这些麻烦。

在 WR-MBMS 中，PropertyGrid 所选定的对象是一个 ModelType 类，ModelType 类包含了所有的模型相关信息，所以当 PropertyGrid 选定 ModelType 后，它就会将这些信息显示出来，以供用户输入。

当用户填写好所有信息后，点击“确认添加”，此时 WR-MBMS 所要执行的操作就是将 PropertyGrid 所选定的对象转换为.mod 文件。而.mod 文件采用的是 XML 编码方式，所以这一步的转换操作实质上就是利用 C#语言生成 XML 文件。我们采用的方式是通过 XmlTextWriter 类，利用 XmlTextWriter 生成 XML 文件的

步骤如下[9]：

1) 声明一个 XmlTextWriter 类 wr，并指定 XML 文件的输出路径以及编码方式。

   XmlTextWriter wr = new XmlTextWriter(输出路径, Encoding.Unicode)。

2) 声明 XML 文件的头部信息。wr.WriteStartDocument()。

3) 通过 wr.WriteStartElement()书写 XML 文件中的节点；通过 wr.WriteString()书写节点中的文本；通过 wr.WriteAttributeString()书写节点的属性。

4) 最后保存这个 XML 文件。

至此，整个添加模型操作已经顺利完成。

4.1.2 显示模型与修改模型

显示模型与修改模型在操作上有着许多相同的环节。首先，无论是显示模型还是修改模型，都要先选定一个指定模型 M。随后 WR-MBMS 应在模型库中找到 M 的对应.mod 文件，并将其信息显示出来。这些操作的实现方法是：每个模型都有自己的一个 ID，此 ID 是独一无二的，因此当用户选中某个模型时，系统会得到它的 ID，并将 ID 传给 GetModelByMid 函数。随后 GetModelByMid 返回一个类型为 ModelType 的返回值。这个返回值包含了 M 的所有信息。最后，将这个返回值赋给 PropertyGrid 的选择对象，完成信息显示。

到此显示模型的操作已经完成。如果用户执行的是修改模型操作，则可在 PropertyGrid 上直接修改信息，全部修改结束后点击保存。系统会将最新的 PropertyGrid 上的值反馈给对应模型的.mod 文件，完成模型修改操作。图 4-2 是显示模型与修改模型的过程示意。

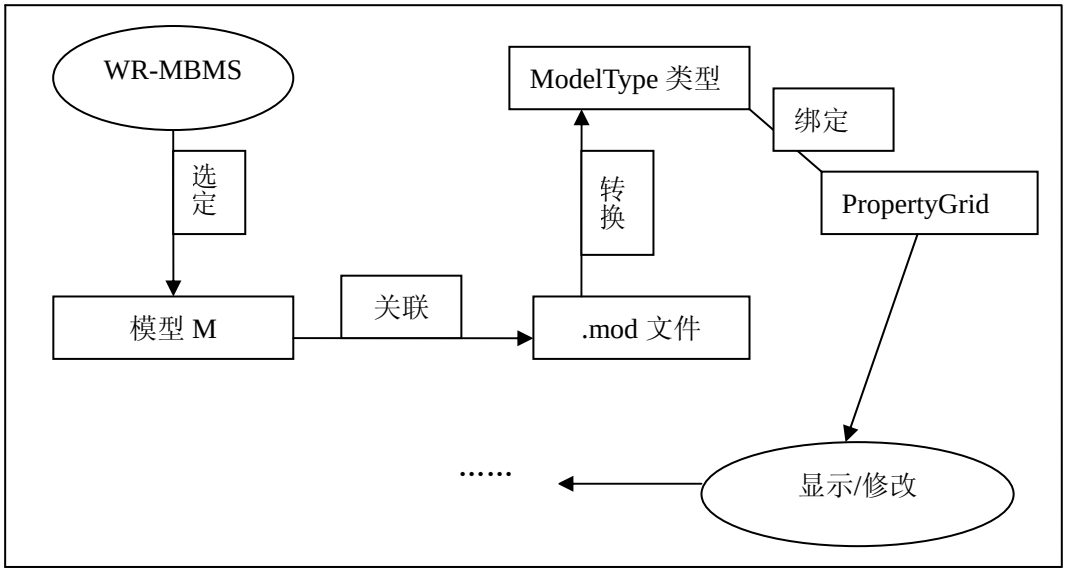

图 4-2　显示模型与修改模型的过程示意

4.2 模型查找模块的实现

如图 4-3 所示是 WR-MBMS 的模型查找窗口。查询模型的条件可以是模型名称、编写者、创建时间以及修改时间中的任意一项。当然也可以是几项的组合，只要某项被勾起，它就将作为条件之一传给查询函数 SearchResult。

图 4-3　　模型查找

SearchResult 函数首先在模型库中根据限定条件筛选出查询结果。随后将它们添加到“查询结果”窗口，此时窗口中的每一项都已经与相应的模型绑定，用户可以直接对他们进行操作。

4.3 模型组合模块的实现

模型合并模块主要包括添加模型间关系和生成组合模型文件这两个基本服务。现在来说明它们的具体实现方法。

4.3.1 添加模型间关系

WR-MBMS 中的“模型关系”这一概念已经在 3.3.2 节进行了详细的描述，这里不再重复。现在让我们来看添加模型间关系这一操作是如何实现的。如图 4-4 和图 4-5 所示，是 WR-MBMS 添加模型关系的界面。

图 4-4　　已添加的关系

图 4-4 显示的是已经添加的模型间关系。每一个关系包含 6 项信息，分别是：关系源模型 ID、关系源模型名称、关系源变量、关系终端模型 ID、关系终端模型名称和关系终端变量。这些模型间关系是随后生成组合模型的必备信息。对于已添加的关系，如果用户发现它已不再被需要，则可以右键选择删除。

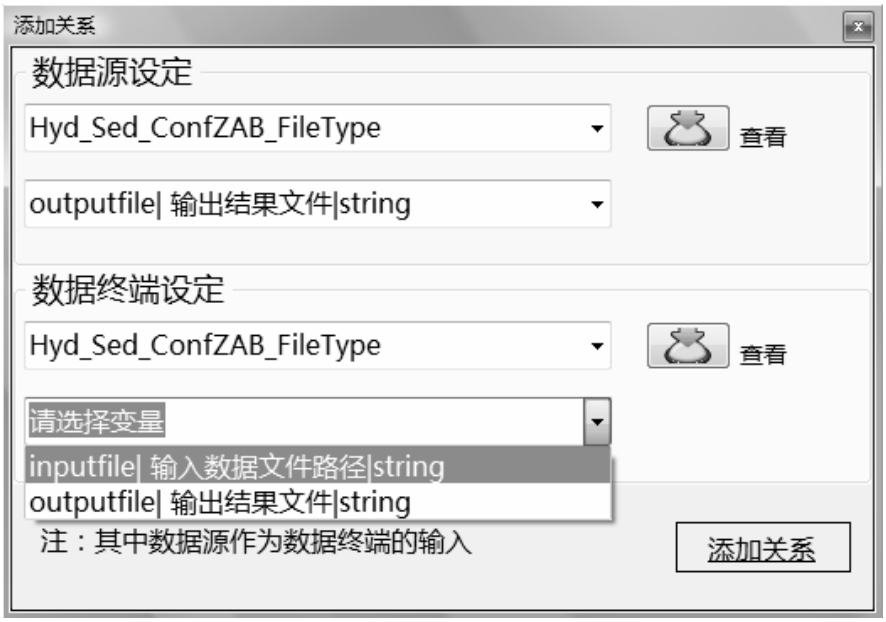

图 4-5　　添加一组模型关系

图 4-5 是添加模型关系的界面，添加一组关系需要输入 4 项信息：数据源模型，数据源变量，数据终端模型和数据终端变量。这 4 项信息都通过 comboBox 来选择。

当界面初始化的时候，用于选择模型的 comboBox 会自动与模型库进行绑定。也就是说，如果用户点击 comboBox，下拉列表会显示出所有可选择的模型。随后，若用户选定了一个模型，此时点击右侧的“查看”按钮便可以查看这个模型的所有信息。当用于模型选择的 comboBox 值不为空时，变量选择 comboBox 就会与选定模型的变量列表进行绑定，下拉列表显示此模型的所有变量。

当 4 项信息都选定以后，用户要点击“添加关系”完成操作。此时 WR-MBMS

会进行逻辑检查。首先，要确保 4 项信息的完整性，每一项都应该有值，不能空缺，否则系统会发出错误提示。其次，数据源模型与数据终端模型不能是同一个，因为这样会造成死锁现象。若关系符合所有逻辑条件，则添加成功，窗口会自动关闭。

4.3.2 生成组合模型文件

当所有的关系都添加完成后，可以将它们保存成组合模型文件。这项服务分成以下两个环节实现：

1. 拓扑排序检查。

因为模型间的关系是数据传递，即数据由数据源传递到数据终端，因此可以将一组关系模拟成一条有向边。这样多组关系加在一起就可以模拟成一个有向图。所以，若是一个符合逻辑结构的组合模型，它就应该满足拓扑排序[10]。

拓扑排序是由某个集合上的一个偏序得到该集合上的一个全序,这个操作称之拓扑排序。用顶点表示活动，用弧表示活动间的优先关系的有向图称为顶点表示活动的网（activity on vertex network），简称 AOV-网。若从顶点 i 到顶点 j 有一条有向路径，则 i 是 j 的前驱；j 是 i 的后继。下面是拓扑排序的方法及举例：

（1）在有向图中选一个没有前驱的顶点且输出之。

（2）从图中删除该顶点和所有以它为尾的弧。

（3）重复上述两步，直至全部顶点均已输出，或者当前图中不存在无前驱的顶点为止。

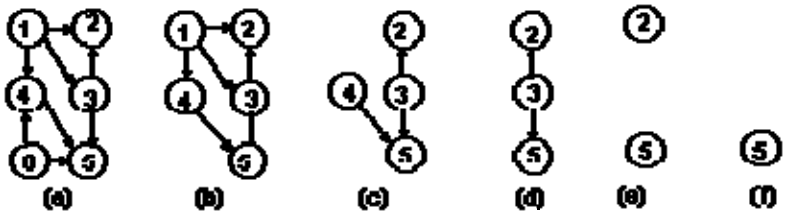

图 4-6　　拓扑排序举例

图 4-6 的 AOV-网及拓扑有序序列产生过程如下：（a）AOV-网（b）输出 V6 之后（c）输出 V1 之后（d）输出 V4 之后（e）输出 V3 之后（f）输出 V2 之后。

在 WR-MBMS 中采用的就是这种拓扑算法，若最终所有的模型都包含在拓扑序列中，则此组合模型能够正常运转，并且得到的拓扑序列就是最终子模型的执行顺序。

2. 生成组合模型文件

在第 3.3.2 节中我们已经介绍了组合模型文件的大体结构。如果用户选定了所有关系，便可以生成这种文件。实现的方法如下：首先，通过上一步的拓扑排序检查,程序获已得了一个拓扑序列,即组合模型中个子模型的执行顺序序列(以

数组的形式储存)；接下来，程序从前往后地将执行顺序序列由数组转换成<Step>节点（参见 3.3.2），每个节点对应一个子模型；同时，程序也要把每一组关系转换为<Relation>节点（参见 3.3.2），这样组合模型文件在执行的时候才能知道数据传递的路径；最后，将整个组合模型文件按照 XML 的编码方式保存[11]。

4.4 模型管理模块的实现

模型管理的主要操作就是模型分类树的维护。实现过程中采用的主要方法是，将模型分类树与 XML 文件进行绑定。这个 XML 文件就是 3.3.3 节提到的用户分类信息文件。当用户修改分类信息的时候，就要对当前使用的用户分类信息文件（一个 XML 文件）进行更新操作。主要的更新操作实现如下：

1) 添加新的分类：在用户分类信息文件中加入一个新的<class>节点，同时分配给它一个 id，并且根据模型分类树的层次关系获取其父分类的 id，将这些信息都更新到 XML 文件中。
2) 删除一个分类：首先，判断出删除的是模型分类树中的哪个节点，获取其 id；随后，找出此 id 对应节点的所有子结点，并将它们从 XML 文件中一一删除；接着，删除此节点本身；最后，更新 XML 文件并自动保存。
3) 重命名一个分类：只需通过分类 id 找出此分类对应的<class>节点，并将<class>节点下的<name>节点值更新即可。
4) 移动一个分类：每个节点都有一个父 id 值，代表其父节点的 id。所以移动一个分类，只需将它的父 id 值更新即可。

由上述文字可以看出，在模型的管理过程中，有很多处都涉及到了 XML 节点查询操作。WR-MBMS 是通过 X-Path 技术来实现这些操作的。

XPath 是给 XSL 转换和 XPointer 的共享功能提供一个共用的句法及语义的结果[12]。XPath 的主要的目是用于对 XML 文档元件寻址。在支持这个主要目的的同时，它也为字符串，数字和布尔的操作提供了基本手段。XPath 使用简明的、非 XML 句法以便于在 URIs 和 XML 属性值以内使用 XPath，XPath 对 XML 文档抽象的、逻辑的结构而非它的表面句法进行操作，XPath 的名字源于它是在 URL 中用作对 XML 文档层次结构进行导航的一个路径标志[13]。举例来说“//Variable[@mid='1' and @name='a']”表示 mid 为 1 并且 name 为 a 的< Variable>节点[14]。

4.5 模型调用模块的实现

模型调用主要分两大部分来实现，第一部分是单一模型的调用，另一部分是组合模型的调用。下面来分别说明它们的实现方法。

4.5.1 单一模型的调用

单一模型的调用主要是利用了.Net 的反射机制。反射（Reflection）是.Net 中的一种重要机制，通过反射，可以在运行时获得.Net 中每一个类型（包括类、结构、委托、接口和枚举等）的成员，包括方法、属性、事件，以及构造函数等。还可以获得每个成员的名称、限定符和参数等。有了反射，即可对每一个类型了如指掌。如果获得了构造函数的信息，即可直接创建对象，即使这个对象的类型在编译时还不知道。 反射提供了封装程序集、模块和类型的对象。可以使用反射动态创建类型的实例，将类型绑定到现有对象，或从现有对象获取类型并调用其方法或访问其字段和属性。如果代码中使用了属性，可以利用反射对它们进行访问。

在 WR-MBMS 中，单一模型利用反射机制，从引入到调用共经历了以下几个步骤：

1) 引入程序集：Assembly ass=Reflection.Assembly.LoadFile(模型文件路径)
2) 获取模型的数据类型：Type type = ass.GetType(模型方法的类名)
3) 建立模型的实例：object obj = ass.CreateInstance（模型方法的类名）
4) 获取模型的方法：MethodInfo method = type.GetMethod(方法名)
5) 调用模型：method.Invoke(obj, 输入参数列表)

由上面的说明可以看出，调用单一模型时所需要的信息包括：模型文件路径、模型方法的类名、模型方法名以及输入参数列表。前三项都可以通过.mod 文件查询得到，而最后一项（输入参数列表）的获得，WR-MBMS 通过三种不同的参数传递模式进行实现。

所谓的参数传递模式是指模型的输入参数是通过怎样的形式获得。第一种传递方式是普通传参型，即若模型需要 N 个输入参数，那么在调用时，就要将这 N 个参数一一给出；第二种传递方式是内存传参型，即所有参数都存放在一个叫做 dictionary<string,string>的数据结构中。这样模型通过 dictionary<string,string>变可以自行查寻每个参数的值（具体方法参见 C#中 dictionary<string,string>的使用）。第三种传递方式是文件传参型，即将所有的参数及其值存放到一个文件中，这样模型可以自行到文件中查询参数的值。这种存储参数的文件格式如图 4-7 所示。

```
<Variables>
        <Variable name="参数一" val="value1" />
        <Variable name="参数二" val="value2" />
        <Variable name="参数 N" val="valuen" />
</Variables>
```

图 4-7　　参数文件举例

由以上论述可以看出，单一模型调用的实现方法是一种动态的调用方法。所谓动态，就是说在 WR-MBMS 编译后、运行时，用户可以随便调用任何一个模

型，具体调用那些模型，在 WR-MBMS 运行之前是完全不知道的。

4.5.2 组合模型的调用

调用一个组合模型要比调用单一模型复杂的多，这项操作不可能一步完成，所以要划分成如下几步进行实现：

1) 获取初始参数：组合模型同单一模型一样，再调用之前一定要赋以初始参数。不同的是，组合模型中不一定每一个子模型都需要初始参数，有些子模型的初始参数是组合模型中其他子模型赋以的。因此，在调用的最开始，首先要为那些必须赋以初始参数的子模型传参（即.cmod 文件中<InitModels>节点所表示的那些子模型，详细参见第 3.3.2 节）。
2) 生成组合模型运行脚本：就是要通过.cmod 文件以及获取的初始参数，生成一个运行脚本文件。这个文件包含了组合模型运行的所有条件信息，如所有子模型的 id，初始参数值，子模型间的数据传输路线等。
3) 解析脚本文件，运行模型：最后一步，WR-MBMS 会自动解析组合模型运行脚本，按照脚本提供的信息一步步执行子模型，并且在执行的过程中，所有计算出的变量值被存放到一个公共的临时文件中。这是一个 XML 文件，其中每一个变量对应一个<variable>节点，节点有两个属性，一个是变量名属性，另一个是这个变量所属模型的 id。这样当某个子模型需要其他模型提供输入参数时，它就会自动去这个公共文件查找[15]。

第五章　　总结与展望

水利模型库管理系统（WR-MBMS）对于整个水利决策支持系统开发运行平台(WR-DSSRP)来说是非常重要的。WR-MBMS 的性能好坏很大程度上决定了 WR-DSSRP 的工作效率。

论文主要讨论了 WR-DSSRP 中与模型相关的内容，目的是使 WR-DSSRP 能够与外部模型达到无缝结合，即 WR-DSSRP 能无障碍地操作外部模型。

在编写论文的最初阶段确定了一些 WR-MBMS 必须提供并且需求明确的功能。例如，用户可以对模型进行分类，分类采取可视化的形式，即模型分类以一颗树的样式显示给用户，只需对树的节点进行添加、删除、移动等操作，就可改变模型的分类信息；用户还可以查看每一个模型的具体细节，就如同打开文件一样，双击一个模型便可查看它的信息；用户能对模型进行查找，只要给定一些限制条件，就可过滤出满足要求的模型。

此后，随着几次会议以及更多的需求分析又确定了模型调用的工作细节，包括：商定外部模型应该满足的格式；模型与功能函数的区别；模型的存储方式等。最后决定通过 C#的反射机制调用外部模型，这种方式使得水利模型编写人员能够用任何程序语言制作模型，大大提高了 WR-DSSRP 的应用范围。同时，随着模型相关细节的敲定，也确定了模型输入输出文件的存储方式，即采用 XML 技术存储。

在程序几次交付用户试用后，WR-MBMS 也改进并添加了一些功能。比如说 WR-MBMS 实现了对模型的组合操作，如果几个模型之间存在着数据传递关系，用户一定会觉得一个个地调用是非常麻烦的。这时他就可以将这些模型组合起来，生成一个单独的组合模型。

目前 WR-MBMS 已经完成了模型管理的基本操作并且提供了若干接口函数，能够很好的为 WR-DSSRP 提供服务。但是由于水利数据库目前还没有搭建起来，所以模型现阶段还不能以数据库中的数据作为参数进行计算。而且由于数据库的结构还没有确定，模型文件还不能存储在数据库当中，而是以结构文件夹的形式进行管理。另外还有一些有待完善的功能如下：

- ✧ 模型查找：目前的模型查找只支持用户输入单项的具体值。例如输入作者的值为“张三”。未来可以引入正则表达式和通配符，这样提高了搜索的智能性。
- ✧ 模型的组合：现在的组合方式是用户一次次的添加模型关系，有几个就添加几次，这样做虽然功能性上没有缺陷，但是缺乏直观性。在未来的系统中，可以添加模型间连线功能，即若两个模型之间存在着某种关系，则可在他们之间拖拽一根线，再添入关系的具体细节。这样做不但直观，而且方便。

- ✧ 模型的细化：现在虽然已经可以查看模型的相关信息，但是这些信息都是“外部的”,是模型的属性信息。以后的系统可以考虑支持查看模型的”内部“信息，比如说可以查看模型所包含的所有类，所有变量等。

相信，经过不断的改进与完善，WR-MBMS 一定会成为一个高性能，易于使用的模型库管理系统。

参考文献

[1] 唐锡晋，顾基发. 水资源管理决策支持系统的发展动态及特点[J]. 北京：管理科学学报, 1994. 1—5.

[2] Efraim Turban．Decision Support Systems and Intelligent Systems[J]．New York: Prentice Hall, 1998. 333—351.

[3] 曾珍香. 决策支持中心-DSS 的发展趋势[J]. 北京：机械工业出版社，2000. 13—19.

[4] 刘晓岩等．黄河水量调度决策支持系统的理论方法与实践[M]．北京：中国水利水电出版社，2003．113—150.

[5] 胡于进．决策支持系统的开发与应用[M]．北京：机械工业出版社，2006．33—45.

[6] 康博．C#高级编程[M]．北京：清华大学出版社，2002.568—635.

[7] 毛尧飞，崔伟．C# XML 入门经典[M]．北京：清华大学出版社，2004．25—226．

[8] 杨宏伟等．C#程序员开发手册[M]．北京：科学出版社，2005．113—244．

[9] 高洪．XML 技术内幕[M]．北京：清华大学出版社，2006．1—18.

[10] 王小科．数据结构之 C#描述[M]．长春：明日科技出版社，2006．101—118.

[11] 李勇平．.NET WINDOWS 应用开发教程[M]．北京：希望电子出版社，2004．221—268.

[12] W3 Schools. XML Tutorial [EB/OL]. http://www.w3schools.com/, December 1999.

[13] Zvon. XPath Tutorial [EB/OL]. http://www.zvon.org/xxl/XPathTutorial/General/examples.html, 2000.

[14] W3C Recommendation. Extensible Markup Language (XML) 1.0 (Third Edition) [EB/OL]. http://www.w3.org/TR/2004/REC-xml-20040204, 2004.

[15] 丁跃潮，张涛等．XML 实用教程[M]．北京：北京大学出版社，2006．44—65．

致　　谢

本论文的完成得益于很多人的指导，关心和帮助。

首先感谢张坤龙老师，从大三开始张老师就带领我开发这个与水科院联合的项目，给我提供了许多动手实践的机会。论文写作期间，张老师还细心指导我论文的大纲结构和内容构思，并为我提供了优越的工作环境。张老师认真严谨的教学态度也深深的影响了我，让我受益终身。

感谢 WR-DSSRP 项目同小组成员：和我一起讨论并给予我编程方面一些建议和讲解。

感谢何明民老师在水力学方面对我的帮助以及在北京工作期间热情的招待。

最后衷心感谢我的家人、室友和朋友，是他们对我的关心、支持和帮助才使我能够安心地做毕设，才能够顺利地完成本论文。