分类号 TP391 学校代码 10590
UDC 620 密 级 公开

深圳大学硕士学位论文

面向移动应用开发的数据交互服务框架的研究

蔡 笋

硕 士 类 别 工程硕士

领 域 名 称 软件工程

学院（系、所）计算机与软件学院

指 导 教 师 朱映映副教授

深圳大学学位论文原创性声明和使用授权说明

原创性声明

本人郑重声明： 所呈交的学位论文面向移动应用开发的数据交互服务框架的研发，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品或成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本声明的法律结果由本人承担。

论文作者签名：

日期：　　年　　月　　日

学位论文使用授权说明

本人完全了解深圳大学关于收集、保存、使用学位论文的规定，即：

- 按照学校要求提交学位论文的印刷本和电子版本；
- 学校有权保存学位论文的印刷本和电子版，并在校园网内提供检索与全文阅览服务；
- 学校可以采用影印、缩印、数字化或其它复制手段保存论文。

（保密论文在解密后遵守此规定）

论文作者签名：　　导师签名：

日期：　　年　　月　　日　　日期：　　年　　月　　日

摘　要

21 世纪后，互联网企业如雨后春笋般的出现创造着形形色色的互联网神话。我国也紧追时代的步伐，提出互联网+的概念。互联网企业出现更是呈现指数级爆发。与此同时，如何在较短的时间内开发一个移动应用并展现其核心价值观和商业模式已经变得越来越重要，并成为一个移动应用能够跟随网络潮流热点开发出相关移动应用的关键要素。

本文从以下几个方面入手，结合理论方法与实践经验，设计了一个适用于各种移动平台的数据交互服务的解决方案的框架，通过实现本框架验证其可行性，并结合应用开发实例，阐述了本框架在现实领域中的应用能力和应用前景。

本论文首先分析总结了当前时期，移动应用开发者在开发一个应用时最常遇到的问题，即需要在某个话题还有热度和新鲜度的时候，就能够结合自身独特的创意和方向，制作相关的移动应用程序提供给相关的用户。

本论文第二部分提出一个数据交互服务的框架。此框架分为两个部分，首先是内容管理系统，该系统方便用户录入和导入数据，开发者只需要利用辅助管理工具，设计基于 XML 模型的数据业务结构，即可自动生成内容管理系统。另一部分则是通过内容管理系统自动生成面向各个移动开发平台的开发包，直接让移动开发者在自己的 APP 内调用相关的数据接口，实现数据交互功能。

本文基于数据交互服务框架实现了一个典型的 Android 新闻应用，验证数据交互服务框架可以为开发提供快捷和便利的开发过程，以及简易的二次开发扩展，让开发者和用户只需要写少量的代码即可完成系统设计，使得开发者可以专注于核心开发任务。

通过一系列的分析研究与实践，本文提出的面向移动应用开发的数据交互服务框架是现实可行的解决方案之一。其实现过程简单，易于定制和优化，容易学习和掌握，可以帮助开发人员提高开发效率，并简化代码管理和维护。本开发框架具有可扩展性，可接入几乎所有的移动平台，实现起来相对简单，维护成本较低，为移动应用开发者提供了一个可行的数据交互解决方案。

关键词：数据交互　互联网+　移动应用开发　集成式开发

Abstract

Since 21th century, the appearance of internet enterprises that have sprung up like mushrooms has created various internet myths. China also follows the pace of the times and put forward the concept of internet+. The arising of internet enterprises has grown exponentially. Meanwhile, it becomes more and more important to develop a mobile application and display its core value and commercial mode in the shortest possible time. Mobile application is a critical element to develop relevant mobile application following internet fashion trends.

Combined with theoretical method and practical experience, a solution framework of data exchange service applied to various mobile platform data is designed in this article in terms of the following aspects. Through realizing this framework and verifying its feasibility, and combination of practical examples of application and development, this article illustrates the applied ability and promising application of this framework in realistic realms.

Analysis and conclusion are made firstly in this paper that mobile application developers have encountered the most common problems when they develop an application at present, that is, combined with their unique creation and direction, they should make some relevant mobile application program for relative users when some topic is still hot and fresh.

Secondly, a framework of data exchange service is put forward in this paper to solve this problem. It is divided into two parts. The first one is content managing system. When users input or import data, developers enable to make content managing system generate automatically only by utilizing assistant managing tool and designing business data structure based on XML model. The second one is about producing development package automatically facing to various mobile development platform through content managing system, and the developers could transfer relevant data interface in their own APP to realize the function of data exchange.

Based on this framework, a typical Android news application is realized in this article. Through development process, it is verified that this framework could provide convenience

and quickness, and simply secondary expansion for development. This application is to allow developer and reader experience the completion of system design only by writing a little code, thus users could focus on relative tasks upon development.

By a series of analysis and test, it is concluded that data exchange service framework facing to mobile application and development is one of feasible solutions in reality in this article. With simple process, easy customizing and optimizing, easy learning and mastering, this framework helps developers improve development efficiency and simplify code management and maintenance. Besides, this development framework could be expanded to switch in almost all mobile platforms, with relatively simple operation and low maintaining cost. It provides mobile application developer with a superior solution of data exchange.

Key word： Information Management System, Internet Plus, Automated Build, Application interface

目　　录

第1章 绪论

随着互联网产业越来越发达和移动互联网的崛起，互联网产业再次迎来一次井喷。据统计，世界上在不同形式中使用互联网的人已经超过百分之八十，并且这个数字还是持续性增长的。在我国，互联网用户已经达到了 8 亿以上。随着互联网产业的现象，我国提出互联网+的概念，并且基于此概念推出“大众创业，万众创新”的观点。因此，越来越多的人加入互联网+产业的大军进行创业。

1.1 本论文的研究意义

进入 21 世纪后，互联网对现实生活产生了一系列的革命，把人们将生活带入到互联网中。在 21 世纪的前 10 年，人们已经习惯于坐在个人电脑前进行各种各样的生活，例如交友，工作和购物等[1]。在第一波的互联网浪潮过去后，移动互联网出现了，它更是像一个巨大的浪潮般将人们的生活完全吞噬，把人们在日常生活中的种种活动和所需的带进了移动互联网中，利用网页，移动终端或者微服务等[2]。人们的日常生活已经离不开互联网。与此同时，我国紧追着时代的步伐，与时俱进地提出互联网+的概念，互联网+是把互联网产品和人们生活结合而出现的一个新的产业，在此基础上更是提出“万众创新”的重要概念，为互联网+的创业者们提供巨大的支持[3]。

那么，对于每一个互联网产品，其重中之重的一个首要任务就是构建其核心的信息管理服务系统，并且基于此系统，为各个终端例如网站，移动终端或其他微服务等提供数据和交互。那么，如何快速地把这套核心基础的信息管理构建出来就成为了争分夺秒的关键了。快速简单的为系统构建基础数据的数据交互接口平台将会为新兴的互联网+企业创业者带来速度上的优势，并且结合其自动数据接口服务功能，更是为其把相关的移动互联网终端紧密连接，最终快速，简洁和高效的为互联网产品提供基础服务，节省了在基础数据的数据交互接口平台和其相关的数据接口的管理时间[4]。

1.2 主要研究内容

如何为移动应用创业者提供快速，简洁和高效的互联网产品核心的数据交互服务框架是本论文的主要关键，尤其是针对于以产品理念为驱动的移动应用和开发水平相对普通的开发者。本论文将基于面向移动开发应用的数据交互的核心问题，提出一个为数据

关系和数据业务模型的解决方案以及针对移动应用快速可以与数据业务模型和数据交互的解决方案的一整套框架。

本论文的主要设计任务分为两方面，首先本论文会提供一个可继承性并且面向二次开发的数据交互接口平台，在此系统中，用户只需要利用 UI 工具来定义基于 XML 模型相关的数据业务对象，则自动生成一个基于增删查改的数据交互接口平台，并且基于此系统开放 SOAP 协议的数据交互服务提供给相关开发的移动应用使用。相关的开发技术是基于 Apache 的 PHP 开发技术。

第二部分则是需要为移动开发者提供可以调用数据交互服务的类库，开发者通过引用相关的核心代码类库，并且添加相关的继承核心代码类库的业务数据对象（Business Data Object）和数据访问对象（Data Access Object），即可以直接访问远端的数据并且展示在移动应用中。本论文前端移动开发技术支持 Android 和 Cordova 技术，所以需要进行基于 Android 的 JAR（Java Archive，Java 归档文件）包以及基于 JavaScript 的类库的开发。

本人在此系统的中承担所有角色的任务，包括系统的设计，开发和测试等所有工作，并撰写本论文对研究内容进行阐述。

首先，在可行性分析方面，由于有众多客户的支持和长久以来为客户提供服务的基础，原来的一套从无到有来构建一套应用系统的方式已经跟不上服务态度与服务质量。开发这一套新型并且快速构建修改的系统已经成为了迫切的需求。由于有需要为客户提供业务服务和为新客户提供最快速的技术支持，本人连同老客户已经新客户加上长期的业务系统开发经验，总结业和技术难题进行可行性分析。然后根据可行性分析、客户调研以及技术分析的总结，获得需求说明书用于对整个系统的业务功能和业务规划的一个指导和说明。然后总结技术，把相关的为客户的使用的代码抽取出公用和通用的类库，用于对数据交互接口平台的支持和让其可以快速为周边的移动客户端提供服务。

在此过程中，本人将承担概要设计，详细设计，实现以及测试的一系列工作和任务。并且最终把相关的产品规划及实现过程在本文中得以体现。通过概要设计，把产品的重要轮廓描述和设计各个模块。然后通过详细设计优化产品的各个模块的功能和接口，最终通过编码来实现本论文所阐述的系统。

然后本论文将会基于本论文所开发的系统来开发一个简单新闻管理系统，用以验证实用本产品系统开发一个互联网产品的快捷性和便利性以及实用性。

1.3 研究现状

在移动应用软件高速发展的今天，已经有各种各样的服务和框架为移动应用的开发做准备。在开发方面，各种跨平台的开发框架应运而生。在移动开发跨平台的开发语言中，比较突出的有 Apache 基金会的基于 HTML5 开源框架 Cordova 和微软旗下基于 C# 的 Xamarin。开发者可以通过这两个框架轻松开发相应的移动平台的应用软件。其他方面如整合通用支付渠道的 Ping++，以大数据为卖点的聚合数据平台和以提供消息推送的极光推送接口等都轻松为移动开发一同各种各样的解决方案。然而，在现有的各种服务和框架中，尚未有对移动应用数据进行管理和交互的成熟框架推出。各个移动应用开发企业基本上是以企业内部开发和构建一个管理系统和数据交互接口移动同时开发移动应用调用数据进行交互的方式对移动应用数据进行管理和交互。本论文则是基于此现状，结合作者本人的工作经验和技术，整合现今移动应用开发中遇到的问题，提供一个可行的数据交互服务框架，以供开发者有一个更好的选择。如上文描述，本系统分为三个主要部分，数据管理中心，数据交互接口以及移动应用开发包。本论文基于现有主流的开发技术对以上三部分进行整合，通过以业务数据模型驱动的方式，来形成一个框架服务。

在数据管理系统方面，大多数开发者通常会利用成熟的 MVC (Model View Controller) 模型作为开发基础，来完成一个基础的数据管理系统。可选择的开发技术通常有微软的 MVC4.0 框架，PHP 的各种流行框架如 ThinkPHP 或者 Ruby on Rail 框架等。这些框架都有成熟开发模型和强大的企业技术做支持，是现今主流的开发技术之一。然而，尽管是非常简单易用的模型，开发者要开发一个数据管理系统依然需要耗费大量的代码在各种 MVC 的代码中进行编码。作者本人曾经在金蝶软件有限公司工作，并且服务于其中 K3Cloud 平台组，将其中基于业务模型驱动的 ERP 管理系统的业务模型驱动的概念提取出来，变成以业务模型驱动的方式，来设计数据管理系统并且自动生成各个业务模型对象的字段相关的行为。最终，本论文提出了开发者无需再编写一个程序代码，只需要通过设计基于 XML 的业务模型，则自动生成各个业务对象模块，以此快速的来对数据管理系统进行开发。在现在的开发软件中，有基于 JAVA 的 JETOOL 与本论文的数据管理系统较为相似，但是 JETOOL 框架主要是为了在数据管理系统开发方面节省开发时间，而本论文的框架则是针对于移动应用开发方面进行整合处理，比 JETOOL 框架有更好的拓展和针对性。

在数据交互接口方面，大多数开发者依然是通过调用 WebSerice 或者 SOAP 等方式，编写各个相关的数据交互接口，尤其是需要编写大量数据库调用语句来获取数据交互接口返回以 Json 或者 Xml 格式的数据到移动应用端进行数据交互。本论文的框架则依然是以业务模型的方式生成相关的增删查改的数据交互接口，用户可以在除非特殊需求的接口，才进行二次开发。

最后，本论文依然基于业务模型来生成移动平台的开发包，移动平台开发包包含三个方面，远程数据接口访问对象，数据对象以及本地数据库访问对象。通过业务模型对象，本框架能够根据定义来来自动生成远程数据接口访问对象的各个接口和相关数据，以及生成数据对象让移动开发平台进行数据访问和本地化存储与使用等。开发者可以不需要编写数据对象相关的代码则已经自动完成了远程数据交互，本地数据库调用和数据使用等代码。在代码生成软件方面，现今流行的 CodeSmith（CodeSmith 是一种基于模板的代码生成工具，它使用类似于 ASP.NET 的语法来生成任意类型的代码或文本）需要编写模板来完成各个业务模块代码的生成。尽管最终能够自动生成代码，但是在编写模板的过程中，依然要花费不少的时间，依然占有一定的工作量。

所以，本论文提供的是面向移动应用开发的数据交互服务的框架。他解决了数据管理系统开发，数据交互接口的开发和移动应用代码的开发的重复劳动问题，整合主流开发技术的思想和设计，来完成面向移动应用的数据交互服务。开发者可以在只设计业务对象上，得到一个可行的数据交互框架。

1.4 论文结构

本论文首先分析和结合了国家提出互联网+概念下，移动应用开发者如何能够在市场中快速的利用自己的创意和商业模式为用户提供移动平台应用服务，对此本论文提出一个快速，高扩展性，低成本的数据交互服务框架来解决开发者的移动应用开发过程中遇到的问题，以此来节约开发中繁冗的基础数据服务器的设计时间。

第二部分本论文提出一个数据交互服务的框架来解决这个问题。并且本论文将会详细地阐述相关的使用技术和原因，然后详细地把框架的开发过程为读者进行描述。总的来说，此框架分为两个部分，首先是内容管理系统，开发者只需要利用辅助管理工具自动构建其设计的业务模型对象的内容管理系统，用户则可以直接对此设计的业务数据进行数据录入或导入。第二部分则是通过内容管理系统自动生成面向各个移动开发平台的数据接口调用开发包，直接让移动开发者在自己的 APP 内即可调用相关的数据接口来

实现数据交互功能。

实现了此框架后，本论文则基于此框架实现一个典型的 Android 新闻应用，通过开发过程来验证本框架为开发提供的快捷性和便利性以及简易的二次开发扩展。让开发者和读者感受到只需要写少量的代码则已经完成了系统的设计，用户可以专心地把开发工作放在相关的核心价值上。

最后，本论文根据框架开发过程中研究的结果和应用开发过程中的体验，进行本论文研究成果的总结，让开发者和读者体验到实现过程简单，易于定制和优化，容易学习和掌握，帮助开发人员提高开发效率并简化代码管理和维护。同时，本论文对本框架存在的一些问题进行分析以及未来的发展情况做展望。

第 2 章 数据交互接口后台的设计与实现

在本论文中，本论文面向的框架使用者主要是将会在互联网+产业中创建新产品的开发者或者设计师，他们可能会创建各种移动互联网的平台、服务或者工具等产品。在本论文的设计中，面向的用户将有一定的技术背景，他们至少需要能够对进行各个移动开发平台的部分有一定的熟悉，本论文的研究成果主要是为了能够解决移动互联网产品开发者在系统开发过程中经常遇到的问题，针对这些问题为他们提供简便的方式以自动生成核心的数据交互服务管理系统[7]。对于这些客户，本论文要为他们解决以下几个问题：

1)快速构建一个数据交互服务管理系统，其业务数据应具有简洁高雅的用户界面设计，快速的页面加载速度和方便简易的操作方式，高通用性和易拓展能够简易进行二次开发。

2)基于数据交互服务管理系统的自动数据生成接口，在此接口中，可以直接通过调用业务系统中相同的链接，基于不同的参数，获得相关 SOAP 协议返回的 XML 数据。

3)基于业务数据接口，本论文需要在各种终端中进行客户端的开发，例如 Android 端和 Web 端等。本论文将会基于 Android 平台提供核心的 Jar 包供用户快捷简单的获取接口数据，做异步数据交互读取的新闻系统的开发。

4)配套的辅助生产工具，例如辅助生产 XML 驱动模型，自动连接数据库进行数据库表的构建等一系列通用功能，更敏捷的辅助用户进行开发。

2.1 数据交互服务管理系统功能需求

本论文中涉及的数据交互服务管理系统主要是为了能够方便开发者能够快速完成系统设计和开发并提供数据接口服务。换言之，在本部分，主要是为开发者提供一个数据管理平台和接口。因此，我们将有以下几方面的设计，来让开发者能够快速生成数据管理系统和数据接口交互服务。

1)根据 XML 格式的业务模型自动生成相应的数据业务管理模块。在本论文的框架设计中，开发者需要打破常规的从设计开发流程。众所周知，尽管开发模型分为几个形态，如瀑布模型或迭代模型等，但是始终离不开从需求设计、概要设计、详细设计到开发等几个阶段。从编程人员的角度则可以看作是从数据库设计，前端界面操作界面设计，后

端数据操作响应以完成用户的增删查改的功能需求。本论文设计的框架以业务模型为核心，则为了打破此种传统模式，让开发者只需要设计 XML 格式的业务模型，自动来完成数据库设计，前端界面和后端响应服务。换个角度说，以往开发者开发一个数据管理系统，尽管用了使得代码结构更清晰的 MVC 框架，依然留下许多的工作需要开发者完成。以 MVC 框架为例，用户依然需要自己进行数据库表设计，通过表设计来编写模型（Model）的代码，编写控制器（Controller）来控制用户的行为，再编写界面和用户交互。尽管好的代码能够减少许多的重复调用，但是实际需要开发的工作量依然不少。本框架则为了打破此方式，让开发者仅仅把开发工作放到业务模型的设计上，其他的数据库表设计，前端界面和后端增删查改服务。

2)业务模型加载模块。虽然本论文设计的数据管理系统已经能够通过业务模型自动生成各个业务管理模块，但是依然免不了开发者需要扩展其相关的功能。举个例子，如果我们是研发一个新闻管理系统，那么当新闻编辑人员录入一条新闻之后，我们则需要自动更新发布时间，而不需要用户录入。这种具体的实际需求则需要开发者编写继承业务模型加载模块的基类。此基类完成一般 MVC 开发框架中控制器和模型的代码。因此，在开发者重写某些特殊方法时，比如开发者重写保存方法，则可以把发布时间字段在代码中设置为当前系统时间，而不需要编辑人员录入。因为业务模型加载模块是结合了控制器和模型的，因此，我们可以直接得到用户请求的业务、数据模型、数据实体和数据库行为，开发者可以在任何必要的时候，通过重写方法的方式，把自己需要的二次开发行为进行编码。

3)基于 Boostrap3.0 设计的前端操作界面。一般来说，开发者同样可能有自己的特殊业务逻辑需要干涉前端编辑人员的录入行为。举个简单例子，当用户输入地址，开发者可以根据地址通过 AJAX 去地址搜索接口获取相关的位置坐标。因此，本框架同样允许开发者以编写 JavaScript 插件的方式，来对用户行为进行操作。比如数据删除后动作，删除前数据过滤，列表页面加载，搜索行为，搜索行为返回结果后，数据保存前，数据保存校验，数据保存，数据保存后以及数据导入后等。开发者可以重写相关的方法实时对用户操作进行控制，以达到自己的业务逻辑完成。

4）数据交互接口。在此数据管理系统作为一个数据录入的行为，那么数据交互接口则是作为数据输出的一个行为。任何的数据录入都是为了能够使数据输出做准备。因此，我们依然会为开发提供相关的业务模型的增删查改接口功能，增删查改接口的权限依然是在业务模型中进行设计。但是，往往仅有基础的单个模型的数据业务模型的表的交互

式不够的。按照开发经验，一般的接口数据交互往往都涉及了多个表的联合查询。而在提交方面则会设计多个表的数据插入和更新。因此，本框架允许自由添加各种行为的数据交互接口，然后通过本框架把放回的数据转换为固定二位格式的 JSON 或者 XML 格式输出到接口中。开发者只需要专注于编写相关的业务逻辑就可以了。尽管在此接口开发中依然需要开发者进行一定的代码编辑工作，但是在普通流程上的开发这一个步骤也是不可避免，那么本论文框架则提供良好的数据输出格式，帮助开发者可以不考虑输出格式等问题，提高开发速度。

2.2 辅助工具功能需求

为了能够让用户更少的输入代码和更快捷的使用本系统构建对应的自动化数据交互服务管理系统，本论文中将会同时提供相关的辅助工具，主要包括以下几方面：

1) XML 模型生成工具，通过有用户界面的 UI 工具，用户可以创建一个 XML 模型，并且利用该辅助工具编辑模型的相关字段以及字段的相关功能需求等。最后生成到系统中直接使用。
2) 数据库表生成工具，用户可以通过导入模型的方式，根据其已经创建的字段和字段类型，自动在数据库中更新或者新增相关的表和字段，减少用户的工作量，协助用户更快速度的生成一套业务系统。
3) 为了能够让开发者减少代码开发工作并提高标准代码统一接口，辅助工具同时需要能够为前端数据读取组件生成相关的核心代码以及业务数据模型对象和相关的业务模型访问接口。

以下图 1 为框架功能模型。

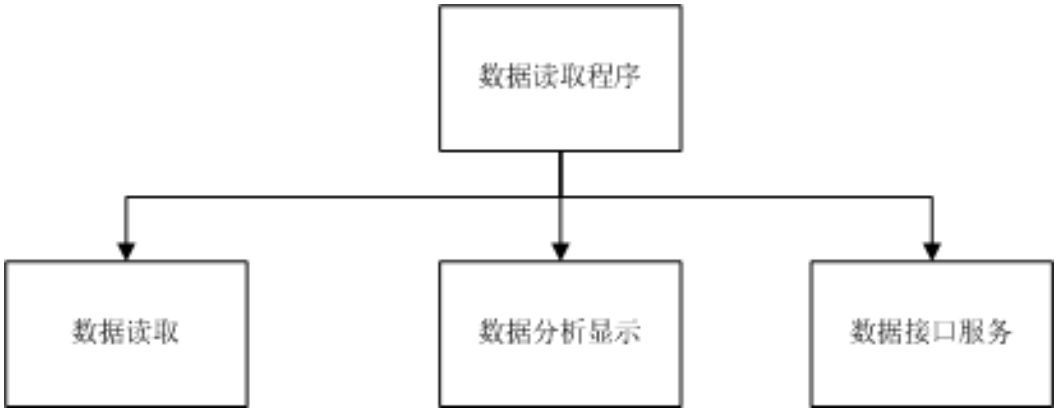

图 1 框架功能模型

2.3 系统模块功能设计

这个是整个业务系统的核心模块，其通过读取开发者定制化的 XML 业务模型，为其用户自动生成一套数据交互服务管理系统，并最终构建整个业务系统和为其用户提供对业务数据的增删查改功能[9]。表 1 至表 5 列出了几个重要模块的模块以及描述：

表 1 XML 模型读取模块功能表

模块名称	XML 模型读取模块
功能描述	本模块主要通过自动读取 XML 业务模型，自动生成业务模型对象，后续系统可以通过解析业务模型对象把相关的业务操作界面展示给用户使用
接口与属性	接口需指定当前的业务对象需要读取哪个 XML。

表 2 业务模型模块功能表

模块名称	业务模型模块
功能描述	本模块主要通过加载已经生成的业务模型对象，然后根据用户提供的业务请求，展示不同的增删查改
接口与属性	用户通过传入业务模型对象以及相应的请求动作，页面自动返回相应的处理动作，至少包括：列表及可搜索字段，单条业务数据新增和编辑保存，批量业务数据删除，字段视图列表

表 3 数据交互接口模块功能表

模块名称	数据交互接口模块
功能描述	本模块主要通过加载已经生成的业务模型对象，或者用户自己生成的业务数据，自动生成 SOAP 协议的 XML 文件流返回到页面中让不同业务中断读取
接口与属性	用户通过传入业务模型对象以及相应的请求动作以及必须的请求参数，开发者可以返回增删查改形式的基本 XML 数据，也可以根据用自己组织业务数据，自动返回用户特殊的业务数据请求

表 4 XML 模型管理功能表

模块名称	XML 模型管理模块
功能描述	提供一个 UI 工具，是的用户可以快速根据自己的需求新增或者编辑 XML 模型，并自动更新数据库的表对象
接口与属性	指定 XML 文件名字名，并且添加模型相应的业务字段

表 5 移动终端数据交互模块功能表

模块名称	移动终端数据交互模块
功能描述	主要功能是指向相应的业务数据对象接口，然后返回有内容的业务数据对象或者返回调用接口的结果（例如保存数据，返回成功）
接口与属性	指定需要请求的 API，设计业务数据对象参数，则自动返回业务数据对象让用户在不同的移动终端上显示和操作数据

2.4 数据库总体设计

本论文设计的系统不涉及数据库设计，因为主要是通过模型驱动的方式为用户把他们实际的数据存入数据库。但是必须遵循本系统定义的数据设计。如果用管理工具辅助生成数据库表，则自动生成基础表结构的内容以及用户自定义的字段。

2.5 系统部署架构

由于本系统基于 PHP 开发，所以首选部署于 Linux 操作系统的 Apache 服务器中。必须开通的模块包括 MBSTRING，SOAP，SimpleXML，XMLREADER 等模块。

由于本系统支持多系统，所以只要 APACHA 中开通相应的数据库模块即可。

推荐部署架构如图 2：

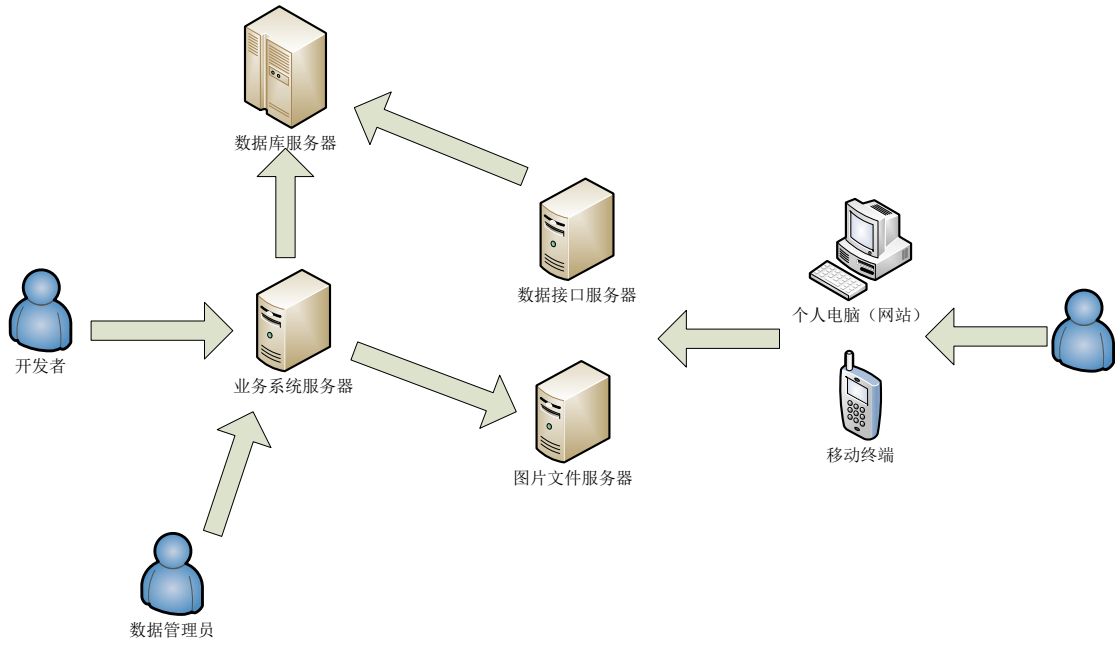

图 2 系统环境部署架构模型

2.6 数据交互接口平台总体设计

这个模块主要用于读取用户已经生成的 XML 并将其转化为相关的业务模型对象，便于以后针对不同页面进行请求以展示不同的结果。在系统的实现中，本论文首先遵循 MVP 的设计模型。众所周知，M 为模型（Model），V 为视图（View），P 为控制器（Presenter）。作为一种新的模式，MVP 与 MVC 有着一个重大的区别：在 MVP 中 View 并不直接使用 Model，它们之间的通信是通过 Presenter (MVC 中的 Controller)来进行的，所有的交互都发生在 Presenter 内部，而在 MVC 中 View 会直接从 Model 中读取数据而不是通过 Controller。本论文这个数据交互接口平台在业务管理模块端则一样是基于 MVP 模型的设计规范，首先本论文通过用户设计一个业务模型对象，此业务模型对象通过读取用户 XML 文件中对 XML 模型的定义，最终生成模型，然后通过控制器加载相关的业务模型对象到业务视图中展现到用户面前并操作进行增删查改[10]。

全局设计图 3 如下：

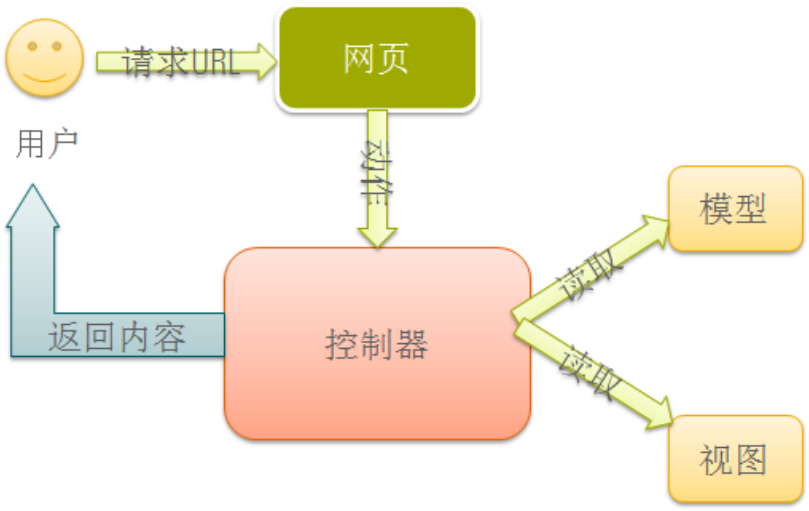

图 3 内容管理系统模型流程

2.7 XML 模型读取模块详细设计

首先必须是系统能够读取 XML。在本系统中，将会设计一个类，类专门读取 XML 文件，然后把其数据加载到相关的属性上，最终生成业务模型对象。通过读取此类，本论文可以获悉相关的数据源（通常为表或者视图）以及相关的字段类型和字段属性，最终

通过处理，本论文这个业务模型对象可以被业务系统生成器使用，最终生成相关的业务。本论文尽可能通过查看 XML 模型和修改 XML 模型则可以了解和修改网页上不同的动作请求生成不同的页面字段。类设计如图 4：

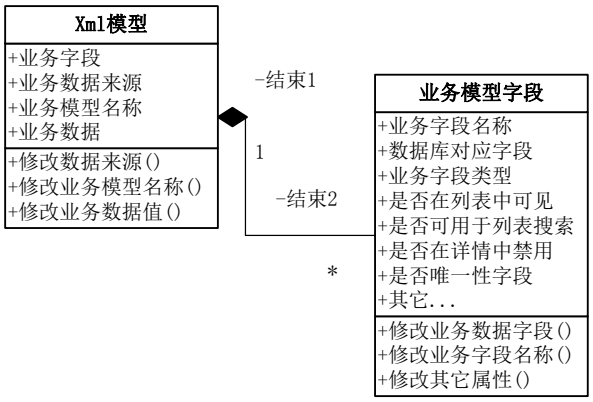

图 4 XML 模型接口设计

同时，某个业务字段可能本身就是要去读取业务数据或者根据特殊业务需求动态修改显示的数据，显示的字段或者其他业务需求等。在此，本论文可以添加一个接口，让用户直接修改业务模型对象及其相关数据。用户可以实现编辑接口的相关方法然后根据业务需求实时修改业务模型对象的内容和其相关的搜索值。接口设计如图 5。

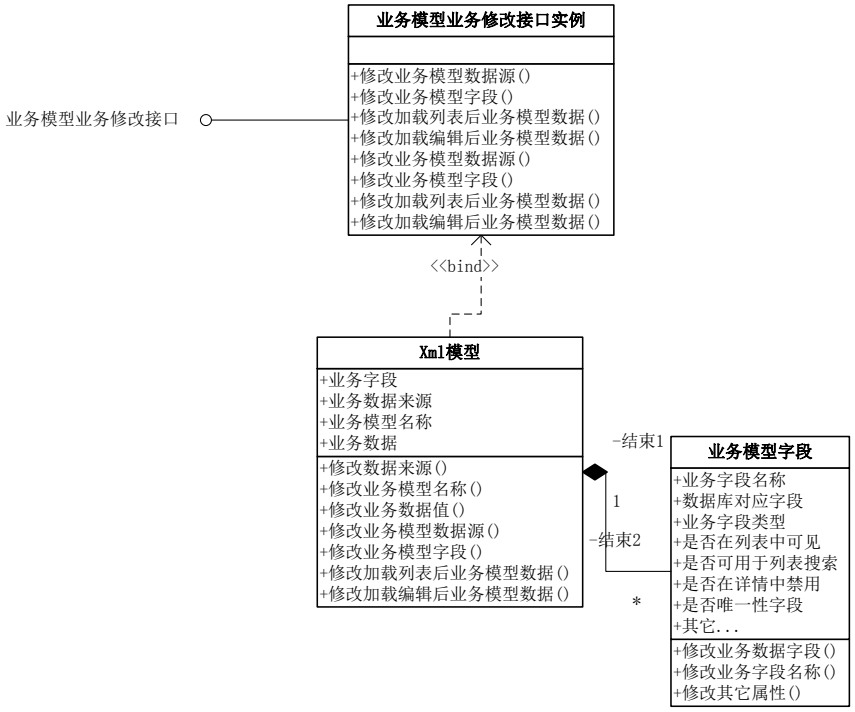

图 5 业务模型接口设计

2.8 业务模型模块详细设计

业务模型模块主要是为 MVP 设计模型的控制器和视图部分，简单的来说，用户通过调用页面并提交相关的动作请求，例如增、删、查和改等动作给控制器，然后控制器则主要负责控制需要如何加载业务模型对象到哪个相关的请求视图中去，然后把获得的页面源代码返回给用户使得用户能够进行增删查改的操作。根据这个动作，本论文可以设计出控制器类如图 6：

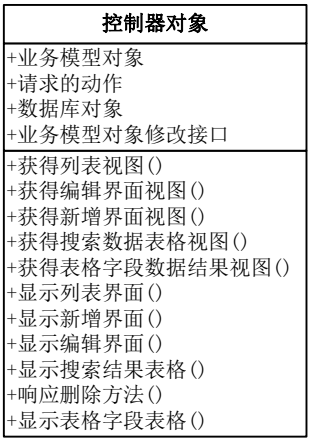

图 6 控制器对象接口设计

在此对象中，主要是获得传入 action 参数，然后加载相应的视图模型，最后业务模型对象加载到对应的视图模型上，实现给对象，简单流程图如图 7：

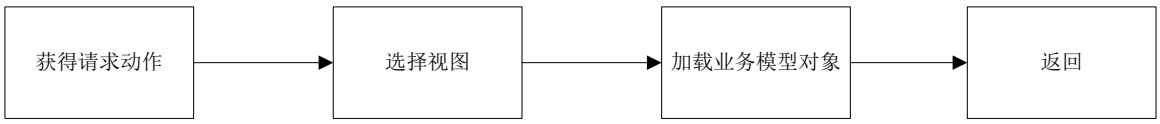

图 7 业务模型加载流程

此控制器对象的方法可以方便被重载，以方便用户进行二次开发。

2.9 数据交互接口访问模块详细设计

业务模型数据接口访问模块主要功能是把用户的输入，把读取视图显示 UI 的方式转换为直接返回 XML 页面数据给用户，以便于开发着进行其它终端例如 PC 端，PC Web 端或者安卓端等 APP 或其他应用进行数据的使用。此对象继承于控制器对象，但是主要有

显示列表搜索数据，显示单条数据详情，响应保存方法，响应删除方法，另外就是响应自定义用户请求[11]。

其请求过程相对数据交互接口平台模块更加简单，主要是获得用户的请求动作，加载业务模型对象，请求数据源数据然后返回结果。通常以特定的 XML 格式返回，以便于相关终端能够以响应的 XML 格式进行读取。接口控制器接口设计如图 8。

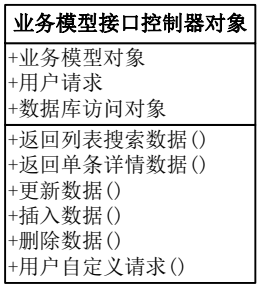

图 8 业务模型控制器接口设计

2.10 XML 模型管理模块详细设计

本系统中为了辅助开发者更加快速的生成数据交互接口平台，需要为用户提供一个用户管理 XML 模型的管理工具，通过使用此工具，加大用户管理的效率。

首先是设置工作空间，由于本系统设计的 XML 模型为视图的 XML 文件对象，所以讲所有的 XML 文件都集中存放于系统的固定目录。那么本系统首先设定工作空间，则可以自动加载工作空间下面有效的所有 XML 文件并且转换为 XML 模型。

当本论文得到所有 XML 模型之后，本论文就可以为开发者提供增查改的功能了。同样的，本论文需要为用户提供搜索页面，用于对当前的所有 Xml 模型进行搜索，然后可以对搜索的数据进行编辑修改或者新增相关对象。

同时，本论文需要可以通过加载相关的 XML 模型，直接自动的生成数据库表对象。通过自动生成数据库对象，本论文可以更加快速的开发和生成数据交互接口平台。首先本论文读取 XML 模型，首先生成基础表字段，主要是 id ,created_date, created_user, updated_date, updated_user，如果是多语言的业务模型对象，则同时会生成多语言表的相关字段，例如 OID 和 Lang 字段。然后再根据用户定义的字段，生成相关的字段，例如用户定义字段类型为文本，则插入或者修改某个字段为 varchar 字段，如果设置为日期，则使用 Date 字段等等。

最后，本论文将根据用户定义的业务模型对象，自动生成第三方移动终端平台开发产品的代码，例如业务数据对象的相关字段以及其 getter 和 setter 访问器。生成数据库访问对象，自动生成用户需要保存到第三方移动终端平台的本地数据库对象的增删查改方法。生成的业务代码方法用户直接服务到相关的项目中。

图 9 为基本的功能设计：

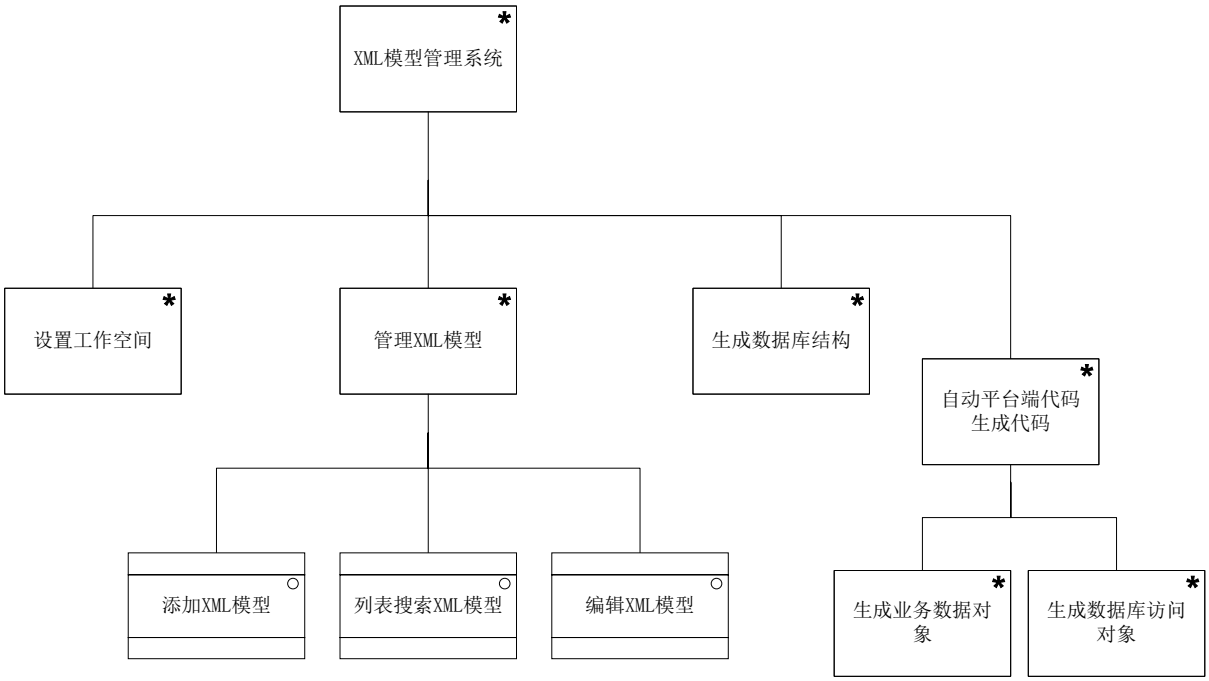

图 9 后台数据管理系统功能模型

2.11 XML 模型读取模块编码实现

这个模块主要用于读取用户已经生成的 XML 并将其转化为相关的业务模型对象，便于以后针对不同页面进行请求以展示不同的结果。首先本论文读取用户表示的 XML 模型名称读取 model 目录下面的 XML 模型文件，然后检查其是否符合相关定义的格式，接着本论文首先读取器数据来源字段和业务数据名称字段，再来读取相关的业务字段，最后读取定义的业务模型功能。此业务模型对象提供完整的数据格式对应数据库表，并且提供接口让开发者可以再次根据自己的实际业务进行二次开发，预留相关的接口。相关的二次开发接口分为两方面，主要是前端界面和 PHP 数据业务处理后端代码的开发。用户可能会根据自己需求在前端界面工作时对用户操作进行约束或者引导。也可能在后端用户数据保存时进行数据计算的约束等。业务模型数据接口访问模块主要功能是把用户的

输入，从读取视图的方式，转换为不读取视图直接返回 XML 页面数据给用户，以便于开发着进行第三方平台例如安卓、苹果、微软等 APP 应用进行数据，此对象继承于控制器对象，但是主要有显示列表搜索数据，显示单条数据详情，响应保存方法，响应删除方法。同时，基于此模型，同样地返回数据访问接口的数据，增删查改能够得到的数据是相同的[12]。程序流程图如图 10：

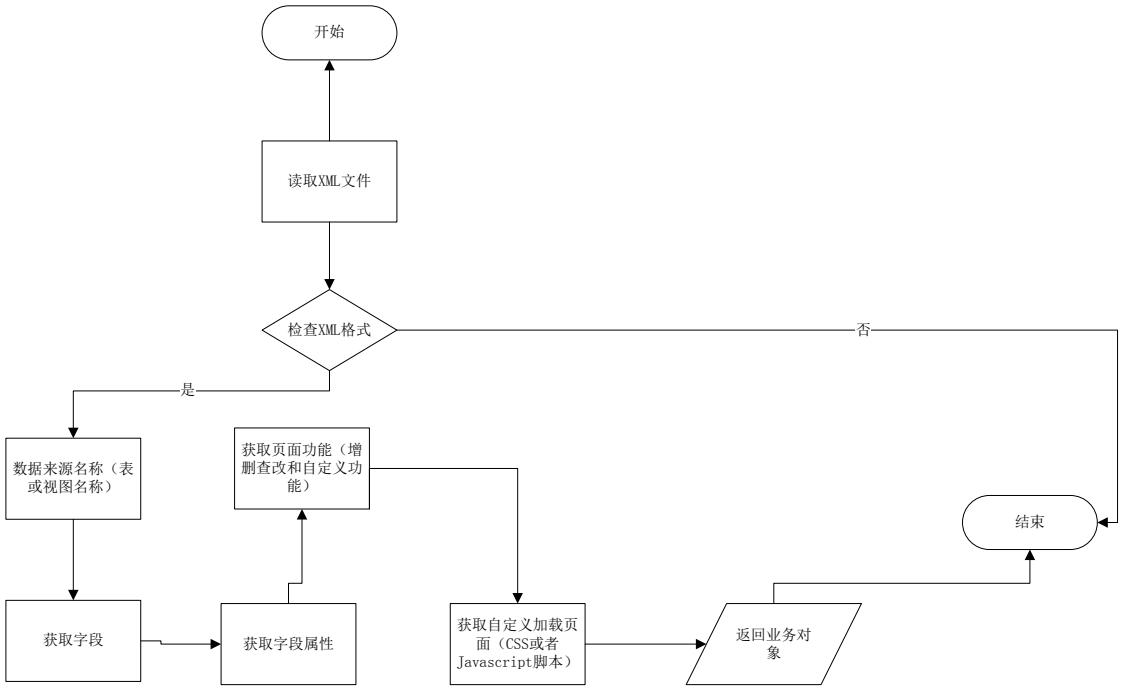

图 10 业务系统模块运行流程

2.12 业务模型编码实现

本模块的编码实现中，主要本论文在控制器中基本流程代码的实现，以流程图方式简单概括，其相关的接口保留方法则不在本节中展现。读者如果感兴趣可以自定阅读相关的源代码。

首先本论文接受用户请求的动作$action 参数，判断其是否常规的定义，例如“”，默认为显示列表界面，“search”搜索数据返回到列表页面中，“grid”搜索数据返回到表格字段中，“new”显示新增业务对象的业务数据新增界面，“edit”，加载相关数据并显示业务数据编辑界面，“delete”删除用户数据或者其它进入用户自定义方法的业务逻辑中。然后加载这个控制器相关的业务模型对象，如果需要加载数据，则根据是请求

生成相应的数据对象，然后把业务模型对象和数据对象一起加载到相应的视图对象中，利用视图中本身签入的业务逻辑代码返回相应的 html 源代码，返回到用户界面中。以流程图方式简单概括，其相关的接口保留方法则不在本节中展现[14]。

首先是整体的程序流程如图 11。

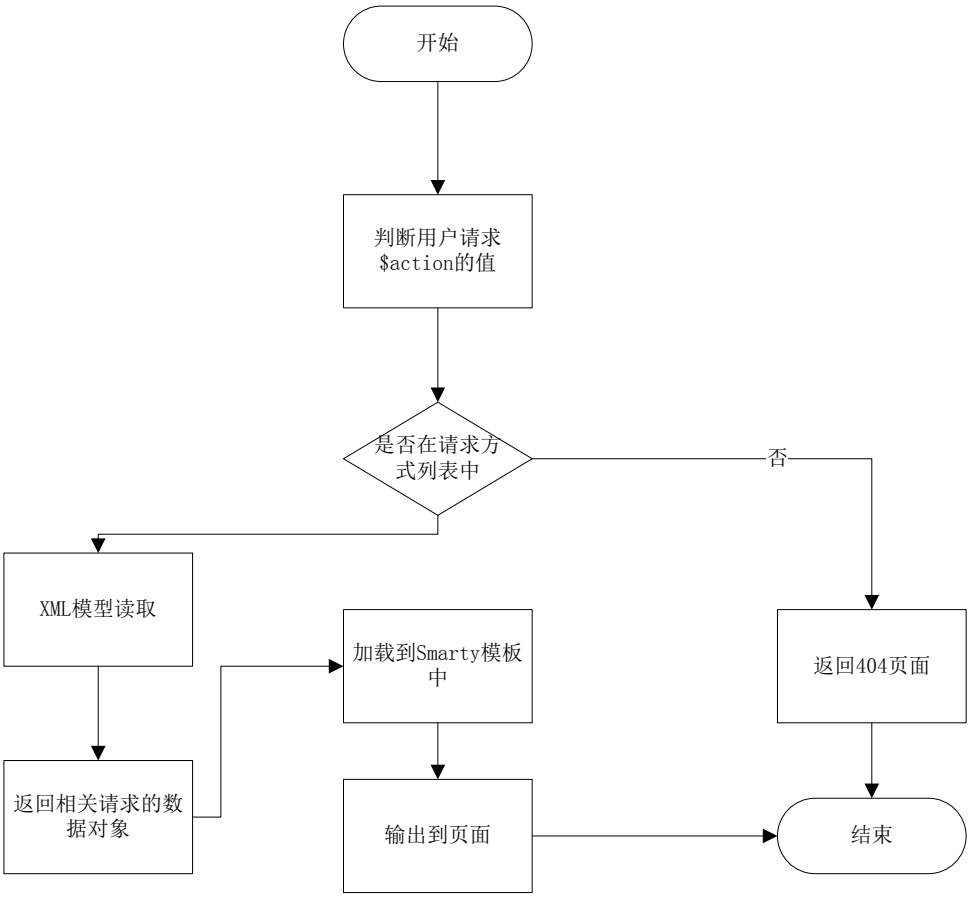

图 11 用户请求到页面输出程序流程

接下来本论文详细的分解响应列表显示界面的程序流程，首先本论文获得业务模型对象，然后判断其相关的外键字段，在本系统中，本论文需要定义集中外键字段，例如下拉列表的数据源可能是另一个表的数据而不是预定于 XML 模型的预定义简单选择字段。多选组合字段也同样可能来源于其它表的数据。所以，首先本论文需要加载相关的外键字段的数据，然后在把当前业务模型对象传入业务模型修改接口实例进行自业务的业务逻辑修改加工，然后则开始把业务模型输出到列表视图中。

在列表视图中，搜索本论文读取业务模型对象加载业务模型管理功能，包括自定义功能和预定义增删改功能，然后加载业务模型定义了 search 为 1 的字段显示到界面中，最后完成其他相关默认的 JS 脚本的编写。例如在这个页面中，本论文需要响应用户点

击搜索按钮后的代码，点击删除后的代码，点击新增或者编辑后的页面 JavaScript 代码。如果用户定义了自定义的通用功能和页面相关业务逻辑，本论文则需要用户自定义一个 js 脚本文件设置到业务模型对象中，然后开发者就可以通过书写相关的 JS 代码来结合自己业务逻辑的流程实现更加丰富多彩的业务管理需求。程序流程图如图 12。

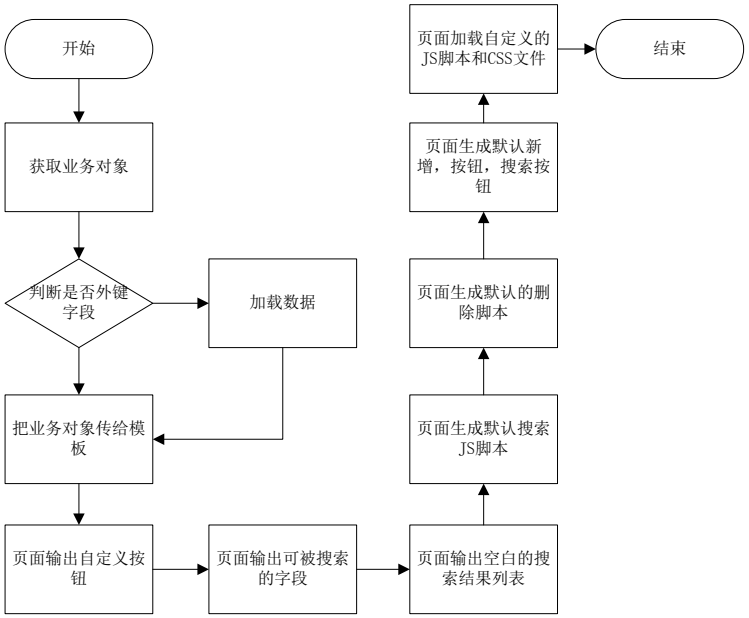

图 12 前端页面加载流程

响应用户点击搜索按钮的程序流程图如图 13。

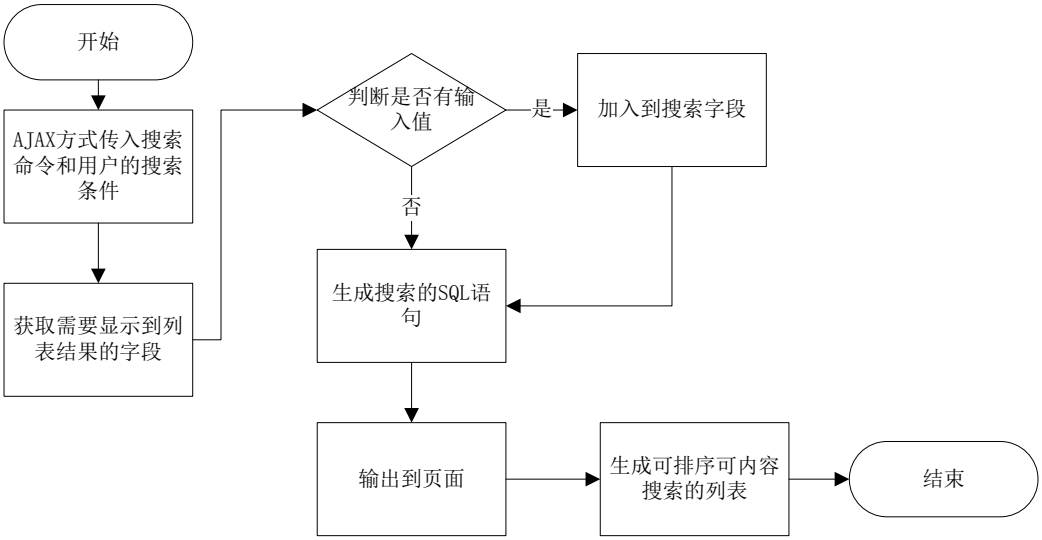

图 13 用户搜索事件响应流程

同样的，响应新增和编辑的整体流程方法与加载列表数据类似，唯一不同则是当用

户显示了编辑页面之后，用户点击保存后，本论文则生成 JSON 数据，提交到相关的后台响应控制器的 save 方法，插入或者更新数据，并返回是否成功的结果给用户。开发者也可以继承控制器对象，来自定义业务逻辑的约束或者数据保存方式的修改。以下提供流程图，程序流程图如图 14。

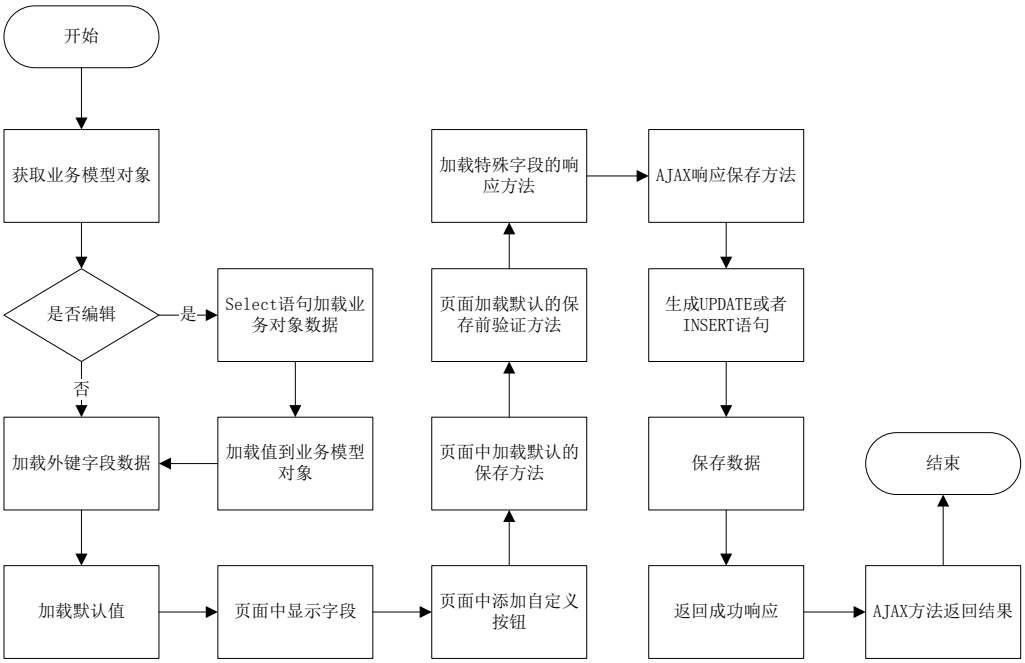

图 14 响应用户数据新增保存流程

响应删除的程序流程图如图 15。

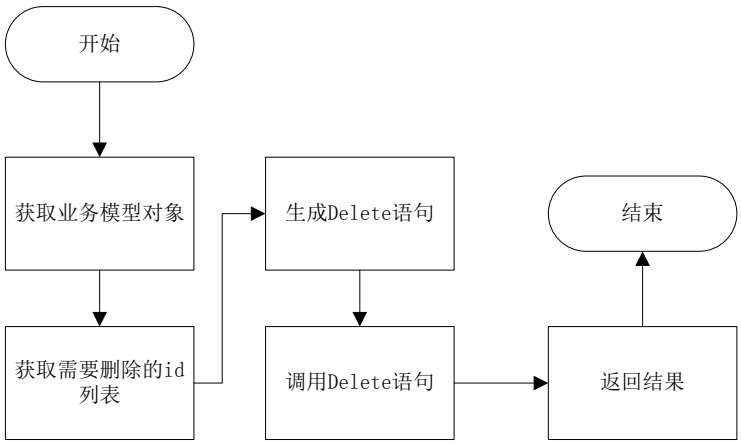

图 15 响应用户数据删除流程

2.13 数据交互接口访问编码实现

当用户通过移动终端通过 URL 利用 GET 或者 POST 方式请求业务模型数据接口事，本论文有以下流程，首先依然是解析请求类型：

“”默认为请求列表数据，请求列表数据与在数据交互接口平台中点击搜索按钮类似，其不同结果为不需要在最终加载到视图中，而是直接把在数据源中获得的数据直接转换为 XML 数据格式，返回到页面中，返回给请求方。

“one”则会获取某一条业务数据的详细信息，根据业务模型的定义，本论文会显示相应的所有字段，其同时要求数据的 id。

“save”则是请求新增或者编辑的业务相应，如果没有附带 id，则默认认为是请求新增，最终将会在数据库中插入一条数据。当然，本论文在此会判断业务的准确性。返回结果为固定的格式，一行加三个字段，分别是 id，result，return。Id 则是操作标示码，通常 0 位成功，其他均为失败，result 为返回结果，通常作为对 id 的解析。Return 则是返回结果，在 save 的请求中，通常返回其相关的业务数据 id。

“delete”则为删除单条或者多条数据，其结果依然返回结果类型的固定格式。

其他请求，用户可以读取 url 请求中的其他类型，自己编写 SQL 语句，最后直接以二维固定格式的方式返到 XML 数据中。程序流程图如图 16。

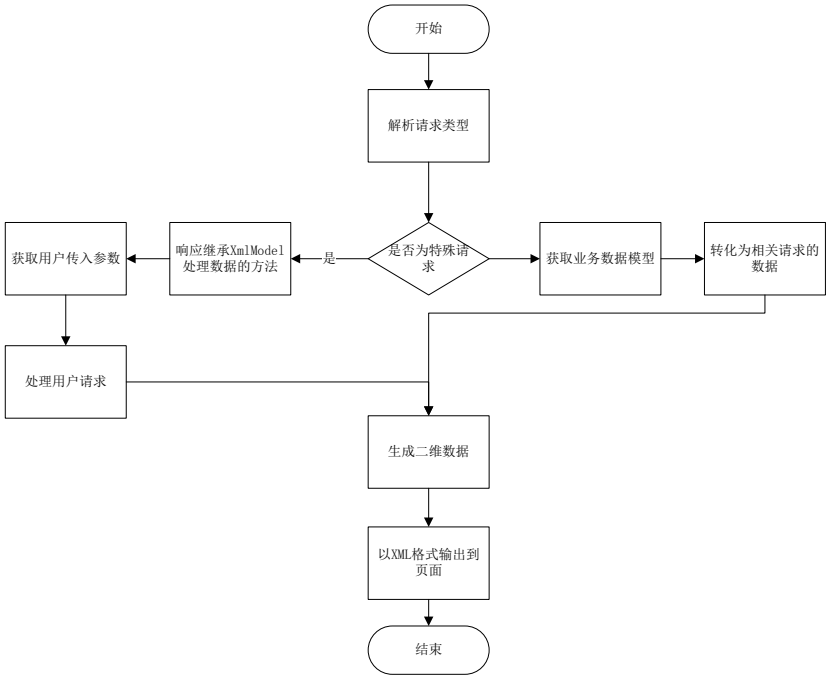

图 16 数据应用程序接口业务流程图

2.14 XML 模型管理模块编码实现

本系统中将会提供 XML 模型管理模块，方便用户用可视化的界面最直接的方法创建相关的业务对象模型，使得快速完成业务模型的设计，同时方便开发者进行管理。程序流程图如图 17。

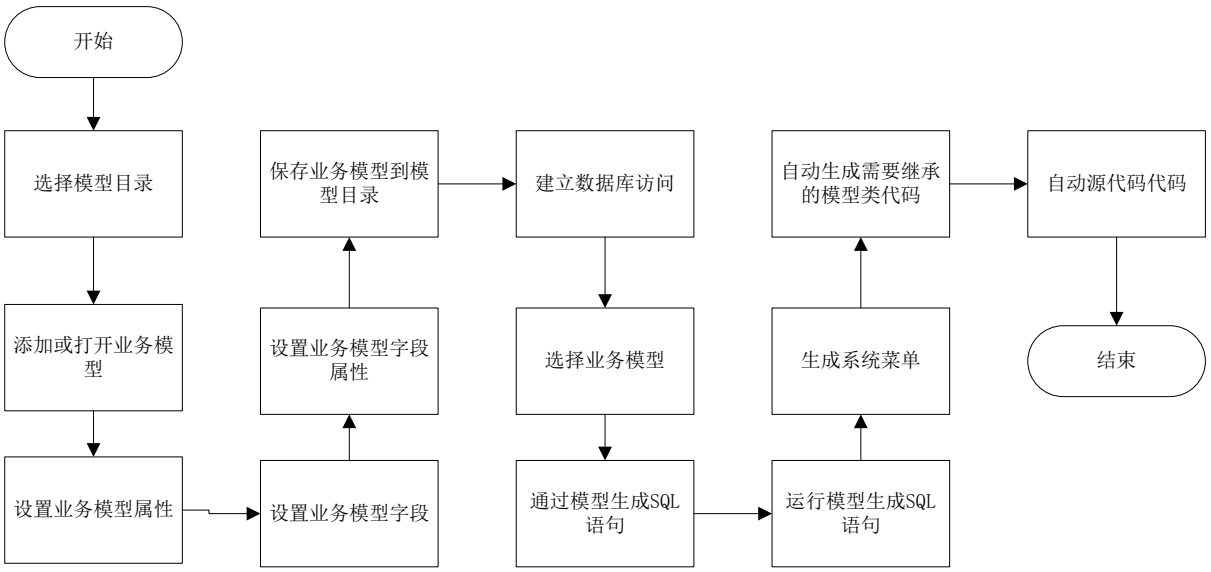

图 17 系统自动架构程序流程

2.15 小结

以上的过程就是设计整个数据交互接口后台管理系统的过程，总的来说，用户通过定义一个丰富内容的 XML 模型，然后本论文代码中的 XML 模型读取模块则自动根据 XML 模型的定义自动来生成相应的前端界面，前端界面数据，后台数据操作等功能。用户只需要根据自己的实际业务模型来定义 XML 模型的结构，就可以自动生成一个数据管理后台和数据交互接口。数据管理后台直接服务于数据管理人员录入数据和管理数据。而根据录入的数据生成数据交互的数据交互接口数据供移动端进行数据交互读取。

整个过程的主要设计就是为了减少用户在反反复复的后台数据管理系统中进行重复编码的工作。如读者所知，正常的任何一个不同形式的应用，实际上都是对输入输出的数据进行交互，除非读者开发的是一个纯单机应用，否则，一般都会花时间和代价开发一个后台数据管理系统供开发者和管理员对用户数据进行管理。本系统的第一部分设计则是希望在最大程度上免去开发者在这方面的重复性工作，开发者只需要定义数据业务模型生成 XML 模型，则可以自动生成整个后台数据交互服务管理系统[15]。

同时，因为现在移动互联网的流行，大部分的移动互联网应用都会需要跟服务器进行数据交互以进行业务流程的推进和完成。本系统则是把后台管理系统生成的 XML 模型进行二次利用，直接生成一个可以被移动端调用的数据交互接口。开发者同样不需要进行任何代码编写就已经可以通过调用数据交互接口来完成移动应用端对后台进行数据较。当然，根据用户的自己的需求，可以定义特殊自定义相关请求接口，更加人性化和合理化方便开发者提供给移动应用的服务[16]。

因此，本论文开发一个易于定制和优化，容易学习和掌握，可以帮助开发人员提高开发效率并简化代码管理和维护。本开发框架具有可扩展性，可接入几乎所有的移动平台，实现起来相对简单，维护成本较低，为移动应用开发者提供了一个优秀的数据交互解决方案。

第 3 章 移动端数据交互组件的设计与实现

在本论文中，本论文将为各种主流的数据终端提供数据交互的核心代码。相关的代码包括对远程数据接口的数据读取，对远程数据接口进行数据提交，将远程数据接口的数据转化为相应的业务数据对象，以及提供相应的数据访问对象接口可以实时对本地数据进行增删查改，使有选择性的让本地用户数据与服务器上定义的业务数据保持同步，最终使用户的想法和业务成功展现在移动终端用户面前。本论文中的提供的代码应该是高拓展高继承性高集成性的，让用户只需定义对应数据请求接口和数据返回对象，就立即可以得到业务数据对象。目的是尽可能的节省用户的编码，使其可以集中精力在自己相关的业务代码的编写上[17]。

本论文讲主要针对安卓客户端和进行 JAR 代码继承包的编写和基于 Cordova 的 JS 包，满足现有的 APP 主流开发平台。

3.1 移动端数据交互组件功能需求

在本论文的数据交互服务框架中，主要的设计理念就是为了让移动应用的开发者能够在最短时间内解决移动应用开发过程中的数据问题。在数据交互接口后台的设计与实现后，基本上解决了数据来源的问题。开发者可以让将数据进行录入并通过数据应用接口返回。那么，在移动端的开发方面，主要的功能需求就是如何为移动应用开发者提供数据处理的解决方案。在本框架中，我们将需要实现以下几个需求，以满足移动应用开发者在开发过程中面临的问题。

1、本框架需要提供开发代码包的自动生成。在接口后台的架构中，我们得知本框架主要是围绕着业务模型自动对系统的数据管理系统和数据应用接口进行部署。那么，同样的，通过业务模型，我们能够自动生成相关的代码。开发者能够通过引用相关的代码包，来进行与数据交互接口的交互，以完成开发者的核心数据从数据管理后台输入到移动应用端输出的闭环。为了实现移动应用端的数据接收和处理，本框架主要提供三种类对数据进行接收，主要是数据加载对象，数据对象以及数据库访问对象。

2、数据对象（Data Object）。数据对象主要的作用是将数据接口中返回的数据，以持久化的方式加载到数据对象的访问字段中。在业务模型中，我们已经知道了业务模型的名称以及相关的字段和属性。那么，根据业务模型的名称和相关字段和属性，我们已

经足够生成一个类，并拥有相关的字段。在移动开发端中，开发者将利用返回的数据对象，将其相关的字段返回到用户界面中与用户进行交互。也可以选择利用数据对象进行数据的本地数据库保存。也可以在本地数据库中将数据读取到数据对象中，展示都用户界面中进行离线数据的交互等。总而言之，数据对象就是业务模型的一个直接映射，开发者基于数据对象对数据进行获取和使用，以达到数据使用的稳定性和确定性。

3、数据加载对象（Web Loader）。数据加载对象主要的任务就是为了和远程的数据交互接口进行交互。通过访问不同的接口或者提交数据到不行的接口，进行数据的交互，然后装载到数据对象中，让开发者进行访问。那么，在数据加载对象中，我们一般开发一个远程数据交互获取的代码中，通常要用许多的代码解决许多的问题。如果是资深的开发者，他会考虑更多细节和安全性的上问题等，这在一般的移动应用开发者业务考虑不到，因此我们生成的数据加载对象则需要提供相关的方法或者属性，并且解决相关的以下问题：

- 以 SOAP 或者 WebService 方式进行远程数据获取。
- 接获取到的数据进行解析，加载为数据对象。应用程序接口返回的数据也许是 JSON 或者 XML。
- 解决异步加载和同步加载的问题。在异步加载完成后，我们需要进行回调以让应用知道返回的结果。通过将结果封装为数据对象。
- 判断是否仅在 WIFI 情况下加载数据。通过对象属性进行设置。
- 每次至更新对比与上一次更新过的数据。比比较更新日期的方式，至获取对比与上一次更新的数据，通常来列表数据获取中使用，这样可以解决用户的流量和提高数据获取速度。
- 定时器和定时循环。很多情况下，开发者需要在设置定时循环后台访问数据接口对数据进行更新。同时也需要延时加载，设置一个 timeout 时间后才去获取数据。
- 流量控制。本框架可以记录用户提交和返回的过程中，分别使用了多少数据，以方便开发者进行流量控制和流量统计。

4、数据访问对象（Data Access Object）。数据访问对象主要是与本地数据库进行交互，进行离线的数据读取和操作。在本框架中，我们一般将数据对象作为数据承载的对象对数据库数据进行交互。比如开发者需要将 List<DataObject>提交到 BatchUpdate 方法中，让数据访问对象将其批量保存到数据库当中。也可能会调用数据访问对象的

List 方法让其返回 List<DataObject>以便让开发者对列表数据进行使用。依然，为了减少开发者工作量和减少一般开发者和资深开发者的差距，本框架自动生成的代码中需要解决如下问题：

- 数据库读写。对于一个需要访问数据库的开发者来说，通常会有数据库的操作对象，让其可以连接数据库，创建数据库，提交 SQL 语句，创建事务等。在本框架中，开发者调用数据访问对象则不需要考虑此等情况，已经在相关的基础上封装让开发者直接使用。
- SQL 增删查改语句的编码。通常来说，我们需要讲一个数据对象在数据库操作，离不开为其编写相关的增删查改语句，而且往往都是重复的代码。因为我们是基于业务对象进行代码的生成，所以数据访问对象的增删查改方法一并生成，开发者调用相关的操作方法就可以将数据在数据库中进行操作。
- 离线数据读取。离线数据读取则是对远程数据在本地的持久化。基本上来说，如果我们使用数据加载对象发生错误无法加载数据，那么最直接的方法则是将本地数据库存储的数据加载到页面中。开发者可以不考虑数据是否加载成功的问题，而直接象数据加载对象进行数据获取，无论如何其也会返回数据。让移动应用有离线数据读取的功能。
- 数据分页读取。如果数据过多的话，开发者需要考虑对数据进行分页读取，以提高加载的速度。对于资深开发者来说通常会进行此部分的考虑与设计，本框架生成的代码也会提供相应的接口和请求，让其可以自动进行分页数据读取，方便开发者使用。

3.2 移动端数据对象设计

在本框架移动端数据对象的设计，我们要求开发者必须使用强类型的数据对象。所谓强类型的数据对象就是对象的没有访问属性都是明确的，而不是在使用的时候才定义。因此，首先创建一个抽象对象，AbstractObject，这个抽象类是所有的数据对象类的抽象基类。他一定会拥有几个相关字段，代码如下：

```
public abstract class AbstractObj {
    private int id=0;
    private String last_update_time=”” ;
    public int getId(){return id;}
```

```
    public void setId(int id){this.id=id}
    public int getLastUpdateTime(){return last_update_time;}
    public void setLastUpdateTime(String last_update_time)
            {this. last_update_time = last_update_time }
}
```

AbstractObject 对象拥有基础的主键和最后更新时间。因此，我们基于此抽象基类生成的业务模型的数据对象就可以只添加相关的设计字段。

同时，数据对象类将会做数据解析的工作。在本框架中，我们主要解析两种数据：解析从远程接口返回的数据，他可能是 XML 也可能是 JSON 格式的数据，但是，我们并不知道需要解释什么字段属性，和是否有特殊的解析逻辑业务。因此，我们只需要定义抽象方法：ParseData(String remotetext)即可。

第二种需要解析的数据就是从本地数据库读取的数据，因为我们可以从本地的数据库将数据加载到数据对象，然后将数据对象返回使用。因此，我们同样需要添加抽象方法：ParseCursor(Cursor cursor)。实现的数据对象就可以将搜索出来的数据游标加载到不同的属性。

最终，我们将通过设计的业务模型集合生成多个继承抽象基类 AbstractObject 的实体数据对象代码。在数据对象的代码中，就包含其相关的业务模型字段，以及相关的解析数据到某个字段的代码。例如解析游标的 Id 字段到数据对象的 id 字段，代码如下由系统自动生成：setId(cursor.getInt(cursor.getColumnIndex("id")));这种重复性的代码讲全部由框架自动生成，节约开发者的时间。

3.3 移动端数据加载对象设计

如 3.1 的功能需求说列，本框架的数据加载对象的主要功能是获取远程数据交互服务接口的数据并将其解析成数据对象。其中，为了减少开发者的工作量和解决常规性远程数据加载问题。首先，本框架依然设计一个 AbstractWebLoader 抽象基类。并实现其主要功能获取数据并返回：

```
Public abstract class AbstractWebLoader extent Thread{
    //构造函数，传入请求的链接以及参数
    Public AbstractWebLoader(string url,List<string> params){}
    private String url;
```

```
    private List<String> params;
    //多线程的运行函数，再次进行数据的获取
    public void run(){
        String res=getSoap(url,params);
          Object pData=processData(res);
          if(callBack!=null){
              callBack.CallBack(pData);
          }
    }

    //通过 SOAP 的方式获取远程的数据交互接口的数据
    Public void getSoap(String url,params){}
    //将获取的数据解析返回相应的数据对象，在子类具体实现
    Public abstract Object processData(String res);
    //定义回掉接口，需要的时候可以进行回掉
    Public IwebLoaderCallBack callBack;
}
```

通过以上代码，我们实现了通过 SOAP 方式获取数据，异步同步调用以及数据解析问题。我们将数据解析留给继承 AbstractWebLoader 的子类实现，子类中通过重写 processData 方法，可以自由选择应该使用相关的业务模型生成数据对象类的代码进行解析。子类代码依然是由框架自动生成。

接下来，我们添加一个参数设置，设置是否仅允许 wifi 情况下调用，并修改 run() 方法。

```
Public void run(){
    //利用 notusewifi 方法判断是否用户当前连接状态非 wifi
    if(onlyWifi){ if(notusewifi())return; }
    //原有代码
}
```

本框架为了节省用户的流量以及提高加载速度，本框架设计一个解决方法，对大数据搜索的列表数据，每次只获取最后更新时间的代码。原理是这样的，我们首先在本地

数据库中创建一个数据表 tb_api_require，他有两个字段，url 和 last_require 字段。其中 url 记录请求的 url，last_require 记录最后一次请求该 url 的时间。当我们进行 soap 请求的时候，我们将 last_require 作为参数传入到数据交互接口中，数据交互接口有相应的代码，仅返回更新时间比 last_requre 更晚时间的数据，这样就可以仅返回需要更新的数据。当然，返回的数据必须要实现本地化存储，否则获取的数据将是无用功。通常只应用在获取列表的批量数据中使用本接口。依然是设置参数 onlynew，默认值为 false。

对于定时器和定时循环的方法，我们添加两个函数，setInterval(int second)以及 setTimeout(int second)，并修改 run()方法，在 run()方法中分别判断是否需要延时启动和定时循环。简略代码如下：

```
Public run(){
   Thread.sleep(timeoutSecond*1000);
   while (isCircle) {
          Thread.sleep(intervalSecond*1000);
          //原有代码
        }
}
```

对于流量控制，我们则是在 getSoap 方法中添加相应的代码，记录下每次上传和下载的字节，添加到本地数据表 tb_byte_statistics，并且表中有字段，require_time(日期时间)，url(请求的 url)，uplength(上传字节),downlength(下载字节)和 type（请求类型，wifi，4G 等)。开发者在有需要的时候，可以通过列表查看。

通过以上代码，我们基本上完成了数据加载以及相关的控制等功能。那么，接下来，我们则可以利用自动生成的代码来实现远程数据获取的具体方法。如新闻列表的数据加载对象为例，代码如下：

```
Public class NewsListWebLoader extend AbstractWebLoader{
   Public NewsListWebLoader(){
      Super(“://news/list”,null);
      Onlynew=true; }
   //处理数据代码
   Public Object processData(String res){
```

```
        List<NewsObject> lst= new List<NewsObject>();
        //循环调用解析
        //调用 NewsDao.batchUpdate 批量更新到数据库
        Return lst;
    }
}
```

同样的，我们通过业务模型和相关的服务方法，我们可以生成很多个新闻请求相关的代码，以方便开发者根据不同的业务场景调用。

3.4 移动端数据访问对象设计

在 3.3 节最后自动生成的代码中，我们可以看到业务模型设置了当返回列表数据之后，会调用批量更新将最新的数据存储到本地数据库中。因此，本框架还会提供一个数据访问对象的自动生成的代码，作用就是将数据存储到本地。在数据访问对象的设计中，本框架依然是会生成一个抽象基类 AbstractDAO。其相关的简略代码如下，首先我们提供几个基础方法。

```
Public class AbstractDAO extend{
    protected DBUtil util;//数据库访问工具
    protected String TableName;
    public AbstractDao(String tablename){
        util=new DBUtil(ctx);
        util.open();
        this.TableName=tablename;
        tableCreate();
    }
    //对象结束后自动回收数据库访问工具
    protected void finalize(){util.close();}
    Public class checkExists(Abstract Obj){
        //判断数据已经在数据库表中，利用默认的 ID 字段，开发者可重写
    }
    public void batchUpdate(ArrayList<AbstractObj> lstObj) {
```

```
        //利用 checkExists 方法检查调用 insert 或 update 进行批量更新
    }
    public void deleteTable(){//删除整个数据表数据}
    public void deleteOne(AbstractObj obj){//删除某行数据表数据 }
    public void deleteNotInList(List<AbstractObj> lst){//删除不在 List 中
的数据表数据}
    public ArrayList<AbstractObj> getList(String cond){//获取列表数据}
    public AbstractObj getObj(AbstractObject obj){//获取单条数据}
}
```

从以上代码我们可以看出，基本上完成了数据获取和数据删除的功能。然而，依然是缺少了关键的和业务模型相关的代码，在本框架中，依然是自动生成相关的代码。我们首先在 AbstractDAO 中定义三个抽象接口：

```
Public void insert(AbstractObj obj);
Public void update(AbstractObj obj);
Public void tableCreate();
```

因此，在自动生成的相关业务对象的数据访问对象代码中，则是自动生成 insert、update 和创建表接口的 SQL 语句。

根据以上的代码，三个部分的代码的自动生成后，移动应用开发者已经可以直接调用数据加载对象获取远程的数据交互接口的数据，利用数据对象显示数据给用户前端和最终通过数据访问对象保存到本地数据库中进行离线使用。基本上解决了在移动应用中的比较重要的数据问题，减轻开发者的工作量和提高开发者开发的代码质量。

接下来，我们来详细介绍三部分代码是如何进行整个来完成整个交互流程的设计和实现。

3.5 移动端数据交互组件编码实现

在此模块中，本论文将针对安卓平台设计一个 JAR 包方便用户直接调用生成的 API 进行交互。分为两部分，首先本论文主要定义两个基础对象，BO 对象的 AbstractObjs 对象，其首先有一个 ID 字段，还有需要实现的两个方法，分别是解析 XML 行数据，本论文在 WebLoad 对象会首先解析为一个 List<Dictionary<string,string>>对象，其数据就是 XML 数据中的一行一行数据对应一个 Dictionary，然后将指点对象传入到 BO 对

象中，让其继承的对象把相关的字段数据解析到相关的字段属性上。用这个方法完成把远程数据访问接口的数据读取到本地。这时，在相应的运行时代码中，本论文已经获得了远程的服务器数据交互接口返回的数据，以新闻列表数据为例，这是，本论文通过继承了 WebLoad 抽象对象的 NewsWebLoad 对象进行远程数据的访问，在回调中本论文就获取了继承了 AbstractObjs 抽象对象的 NewsObj 对象，此对象以及拥有了相关的数据。当然，这里提到的 NewsWebLoad 和 NewsObj 都是自动生成的代码直接可以被用户调用已进行调用并直接显示到用户自定义的移动终端应用界面中[20]。

最终，根据其相关的 DAO 类中，本论文把改数据对象更新到客户端本地的数据库中进行离线数据访问。二次开发远程数据访问流程图如图 22 和图 23。

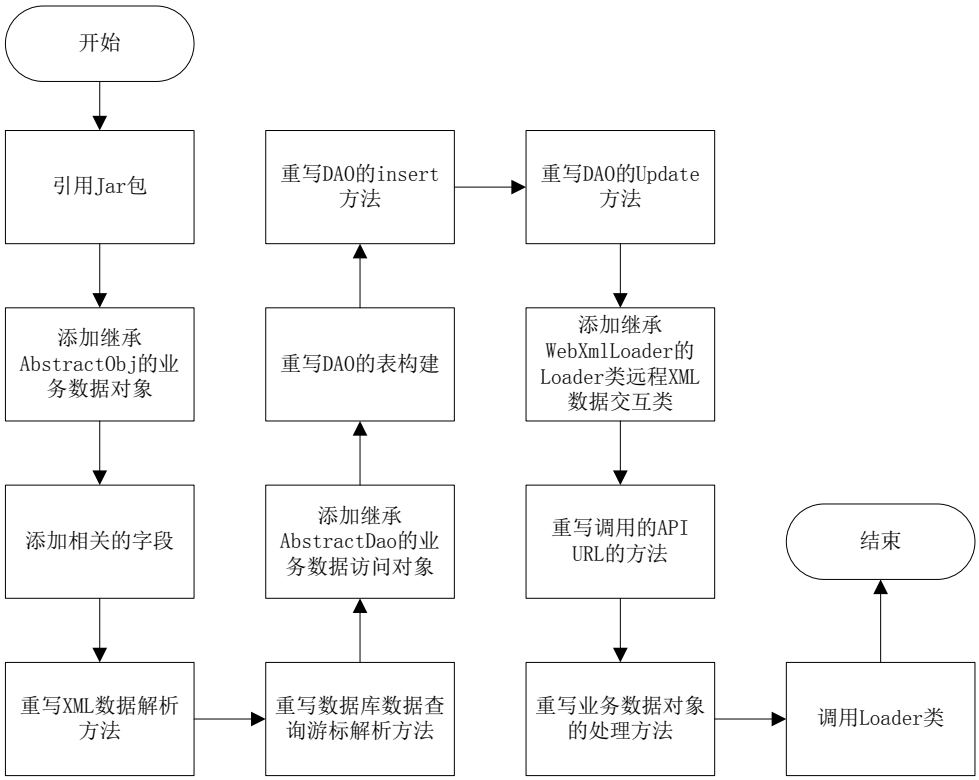

图 22 二次开发远程数据访问流程

3.6 小结

通过以上的设计和实现，我们基本知道了在移动应用开发方面，开发者能够得到一个基于业务模型来生成的代码集合的开发包。开发者基于这些代码，能够轻易的调用相关的方法从远程数据交互接口中获取和交互数据，并将其转换为可以被前端引用的，字段属性确定的强数据对象，同时还可以将数据对象的数据保存到本地数据库中进行离线

读取。开发者可以专心于在自己的移动应用中实现其核心业务和价值，将数据交互的问题交给本框架。

根据以上设计过程得知，我们通过三个基本的抽象类:数据加载对象抽象基类，数据对象抽象基类和数据访问对象抽象基类中的相关的代码来完成从数据获取，数据返回和数据存储的一系列过程。同时，基于业务模型生成的代码包中的自动生成的代码则是负责相关的业务代码，来得到相关的实际业务的数据流程。因此，如果开发者没有过多的实际需求，自动生成的代码包中的数据已经足够开发者进行移动应用的数据开发。同时，如果开发者有相关的特殊业务，也可以同样的自己编写基于三个抽象基类来实现数据交互。

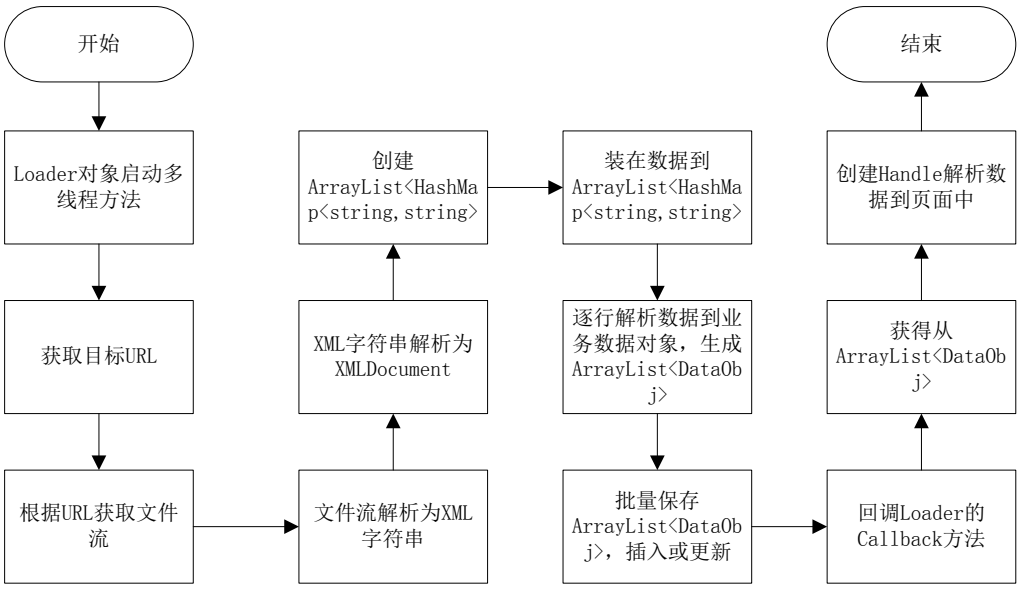

图 23 远程数据接口数据返回解析流程

第 4 章 数据交互服务框架的应用

面向移动应用开发的数据交互服务框架已经设计完成，接下来本论文则需要验证本论文所开发的系统如何应用到实际的开发过程中，以证明利用此系统开发互联网产品的快捷性和实用性。

在本章中，本论文将设计开发一个安卓平台的新闻移动客户端，充分利用本平台来开发的提供的所有工具来暂时开发的代码。并且在此过程中，本论文将会同时验证本数据交互接口平台是否符合设计目的和完成所有计划的需求。

4.1 研究环境安装与配置

首先，本数据交互接口平台的内容管理系统和数据访问接口部分都是部署在 Apacha+PHP 的环境中。数据库则不作要求，只要用户能够连上数据库即可。所以第一步请先配置到 APACHA+PHP+目标数据库环境，本论文中选择使用 MYSQL 环境。

第二步,把源代码中的 CMS 文件夹所有的文件部署到对应的 APACHA 源代码文件夹中，然后就是修改\CMS\Config\config.php 中的配置，例如修改部署路径，上传文件共享目录，数据库配置等。

第三步,把源代码中的 API 文件夹所有的文件部署到对应的 APACHA 源代码文件夹中，修改 API\Config\config.php 中的配置，主要是修改当前工作路径和数据库链接配置。

第四步，导入\DBBackup\init.sql 到数据库中，主要是关于用户管理模块基础数据。

最后，配置 android 平台开发环境，并且引入 Android\helpfooter.jar 包。

至此，本论文可以开始进行新闻客户端的开发了。

4.2 数据交互接口端的设计

如上设计和实现，本论文只需要利用 XML 模型管理系统生成 XML 模型和相关的访问菜单，就可以完成新闻管理端的配置，首先本论文进入\ModelCenter\Startup.exe，然后进入，选择工作目录，自动会显示所以在工作空间下面的所有 XML 模型对象，然后本论文新增一个 news 模型，名字为新闻，数据来源为 tb_news。然后为其添加需要的字段，本论文在此添加标题，导读，内容，发布日期，状态等。并且添加新闻，删除功能。保存后，本论文可以在 CMS\Model 文件夹下面多了一个 news.xml 文件，他就是本论文前

文所述的业务模型，本论文通过工具生成直接生成 Xml 的。主要的模块在为如下的几部分：

主要节点如下：

```
<root>
    <name>模块标题</name>
    <tablename>数据库表名称</tablename>
    <nolist>0</nolist>//是否没有列表页面，仅有编辑页面，0 为不是，可以省略
    <searchcondition> r_main.status&lt;&gt;'D' </searchcondition> 默认搜索条件，此例子中，Status 为 D 的是在数据库中已经删除的数据。
    <fields>相关字段</fields>
    <options>相关操作，例如新增，导入，自定义按钮的功能</options>
    <javascripts>加载自定义或者拓展的 JS 文件</javascripts>
    <csss>加载自定义的 CSS 文件</css>
</root>
```

然后本论文再看一下一些比较有代表性的字段：

比如新闻标题字段：

```
<field>
  <name>标题</name>
  <key>title</key>//数据库代表的字段
  <type>text</type>//字段类型，有 text,number,date 等
  <displayinlist>1</displayinlist>//是否在列表中作为搜索结果显示
  <gotodetail>1</gotodetail>//是否能够点击该字段跳转到详情页
  <search>1</search>//是否作为搜索字段
  <notnull>1</notnull>//是否非空
   <uniqe>1<uniqe>//是否唯一
</field>
<field>
```

当然，这是最基本的字段，本论文还是定制有相关的 type 为 select 的下拉字段，为 grid 的引用内容页面，为“content”的 Wordpress 字段，为“fkey”的外键字段，为“flist”的多选列表或者外表 1 对 N 的关联字段等，此处略。

接下来本论文点击生成数据表按钮，其就会立刻在数据库中添加一个表，并完成其表结构的设计。图 24 为表接口设计。

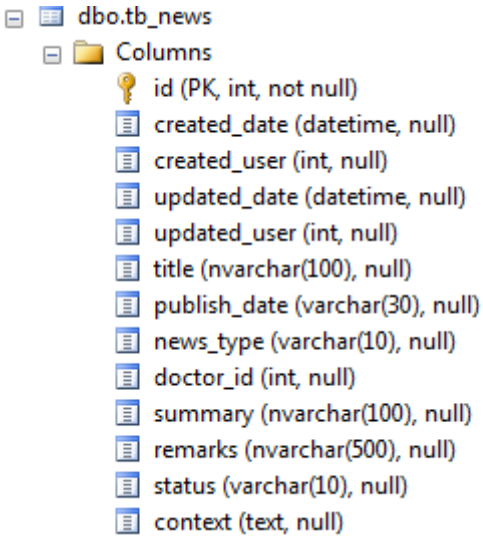

图 24 新闻业务数据库表

然后本论文需要打开编辑系统菜单的选项，添加一个网页和生成网页，然后在 CMS 系统中就会生成一个 CMS 文件夹中生成一个 news.php，并且里面已经有基本可调用的代码，代码如下：

```
<?php
require '../include/common.inc.php';
include ROOT.'/include/init.inc.php';

$action=$_REQUEST["action"];
$model=new XmlModel("news","news.php");
$smarty->assign("MyModule","website");
$model->DefaultShow($smarty,$dbmgr,$action,"news",$_REQUEST);

?>
```

如果开发者需要编写相关的业务逻辑代码，则可以继承相关的控制器对象或者业务模型对象修改接口实例，对数据进行修改，在本例子中，本论文添加一个特殊的业务逻辑，设置发布日期必须为当天。本论文继承 XmlModel 对象，然后命名为 NewsXmlModel 对象，重载其 Save 方法。代码如下图 25：

```php
<?php

class NewsXmlModel extends XmlModel{

    public function __construct($pagename){
        parent::__construct("banner",$pagename);
    }

    public function insert($request){

        //设置字段默认值，发布日期为当天
        setFieldValue("publish_date",Date('Y-m-d'));
        return parent::Save($dbMgr,$request,$sysuser);

    }

    public function update($request){

        //设置此字段不会被更新
        setFieldNoUpdate("publish_date");
        return parent::Save($dbMgr,$request,$sysuser);
    }

}

?>
```

图 25 NewsXmlModel 插入和更新代码重写

接下来，本论文就可以验证是否已经完成新闻模块的管理的开发了。首先本论文使用浏览器进入 CMS\index.php，使用默认登录名登录系统。登录界面如图 26。

图 26 登录页面

然后见到主界面，是关于此系统的介绍。本论文找到新闻管理的菜单，点击进入图 27 界面：

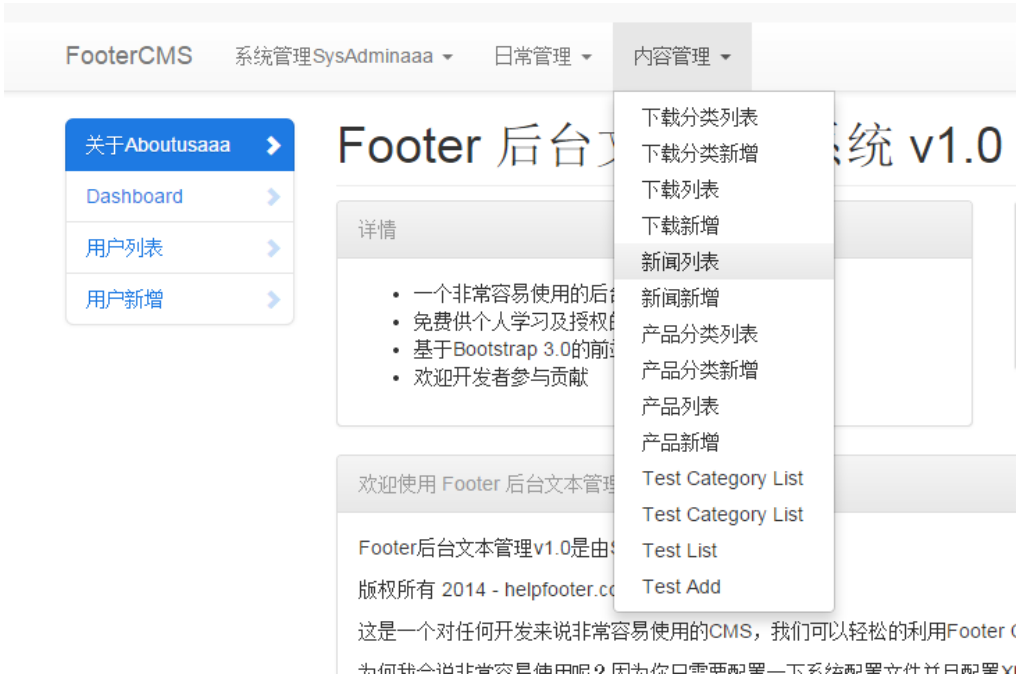

图 27 登录后控制面板

然后本论文点击新增添加一条新闻，输入内容，点击下方的保存按钮，如图 28。

图 28 新增界面

首先进入新闻列表，界面如下，发现左上角新增和删除按钮的功能已经添加如图 29：

图 29 新闻列表

点击搜索可以看到起通过 AJAX 局部刷新搜索结果栏，显示现有的新闻数据如图 30：

图 30 新闻列表搜索结果

点击“新闻标题 3”进入修改界面对数据进行修改，保存后在列表搜索界面中本论文可以验证数据已经被修改。

接下来验证删除功能，在搜索结果的列表中勾选需要删除的数据，然后点击右上方删除按钮，点击确认删除，则不会在搜索到相关数据。删除界面如图 31。

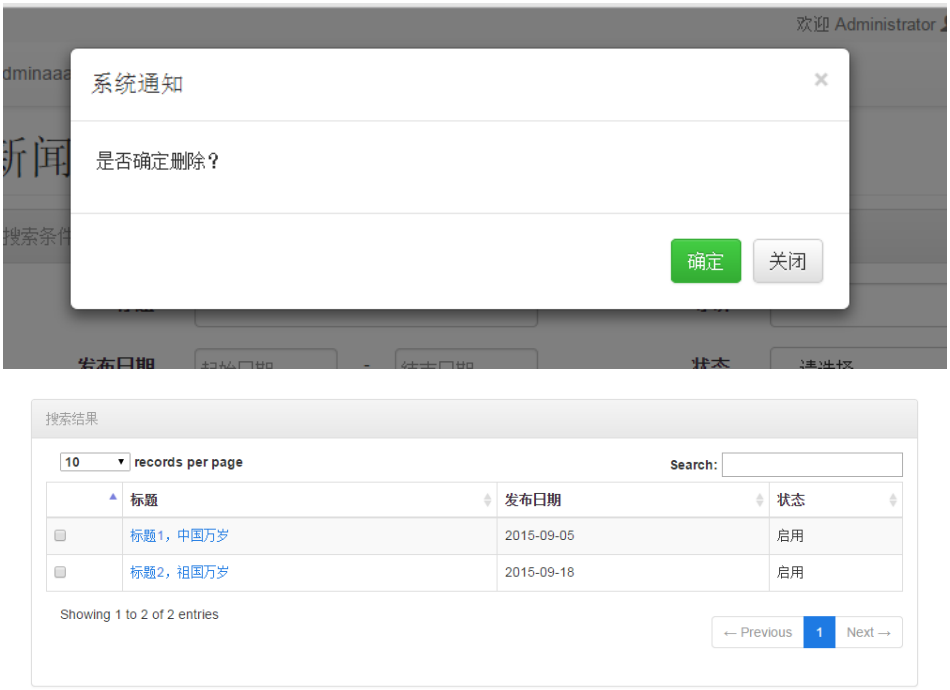

图 31 新闻列表

至此，本论文已经验证，数据交互接口平台已经可以对新闻业务模块内容进行增删查改的实现，而本论文添加的代码仅仅只是特殊需求对发布日期的强制写死录入。基本上，本论文通过 XML 模型管理工具已经完成了所以基本代码的生成，而在这个过程中，本论文并没有书写任何代码就完成一个业务模块的管理。

4.3 远程数据数据访问端的设计

本论文已经可以基本上完成数据访问接口端的开发，当本论文调用 news.php 页面时，本论文获得一个 XML 格式的网页结果返回如图 32。

```
▼<table>
  ▼<row>
     <id>2</id>
     <title>新闻2标题例子</title>
     <published_date>2015-09-24</published_date>
     <status>启用</status>
   </row>
  ▼<row>
     <id>1</id>
     <title>新闻1标题例子</title>
     <published_date>2015-09-18</published_date>
     <status>启用</status>
   </row>
</table>
```

图 32 Xml 格式数据接口列表字段返回

当本论文需要获取某一条数据具体的所有数据时，本论文调用的 url 为 news.php?action=get&id=1，然后获得返回结果如图 33。

```
▼<table>
  ▼<row>
      <id>2</id>
      <title>新闻2标题例子</title>
      <summary>新闻2导读例子</summary>
      <published_date>2015-09-24</published_date>
      <content><h1>新闻2内容例子</h1></content>
      <viewcount>0</viewcount>
      <status>A</status>
      <created_date>2015-09-18 05:05:15</created_date>
      <created_user>1</created_user>
      <updated_date>2015-09-18 05:05:15</updated_date>
      <updated_user>1</updated_user>
    </row>
  </table>
```

图 33 Xml 格式的数据接口详情字段返回

在此基础上，本论文添加一个自定义需求，当用户在第三方客户端访问页面后，本论文需要 news 表中的访问数量字段。于是本论文基础修改 NewsXmlModel 代码如图 34：

```php
<?php

class NewsXmlModel extends XmlModel{

    public function __construct($pagename){
        parent::__construct("banner",$pagename);
    }

    public MyAPIAction($action,$request){
        if($action=""){
            $id=$request["id"];
            $sql="update tb_news set readcount=readcount+1 where id=".$id;
            $this->dbmgr->query($sql);
            return  outResult(0,"success",$id);
        }
        parent::MyAPIAction($action,$request);
    }

}

?>
```

图 34 NewsXmlModel 二次开发代码

当客户端调用 url news.php?action=read&id=1 之后，本论文则动态修改数据库中新闻访问数量的字段。提供开发者更贱简单直接建立一个机遇 SOAP 协议返回 XML 数据

的设计框架。

到这里为止，本论文已经可以看出，本论文并不需要为数据访问接口端书写太多的代码，本论文依然是想信息系统管理端一样自动生成业务数据以成功生成本系统，并且如果用户需要加入自定义的业务时，修改接口也是非常简单灵活的。

4.4 安卓平台新闻移动客户端的设计

开发一个简单的新闻移动客户端，我只需要在项目中添加两个 activity，一个是显示新闻列表，一个是显示新闻详情。本例子中，本论文为了减少客户存储过多的数据到本地数据库中，本论文将不会把新闻的详细内容加载到客户端中，只会简单的字段，例如新闻标题，新闻导读，发布日期等建短信息。

为此，本论文需要首先继承 AbstractObj 类，然后添加访问器和实现其定义的抽象方法。相关代码在框架中生成，开发者可以直接下载代码进行引用使用。继承的 NewsObj 对象主要拥有新闻的相关属性和解析远程数据字符串和解析游标的代码。

然后本论文需要添加一个继承与 AbstractDao 的 NewsDao，新闻数据访问类，并实现其相关的抽象方法。对于本论文要做的则是实现相关的构造函数方法，不过系统中也已经自动生成该代码，用户可以直接使用 NewDao.java 类，代码如下：

```
public class NewsDao extends AbstractDao {


    public NewsDao(Context ctx) {
        super(ctx,"tb_news");//指定表名
    }
    //自定义查询的接口方法
    public ArrayList<AbstractObj> getNewsList(){
        return super.getList(" status='A' order by publish_date,id desc  ");
    }


    @Override
    public void insertObj(AbstractObj abobj) {//代码自动生成}


    @Override
```

```
    public void updateObj(AbstractObj abobj) {//代码自动生成}

    @Override
    AbstractObj newRealObj() { return new NewsObj(); }
}
```

有了这两个对象，本论文的新闻数据就可以保存在内存中和本地数据库中。

然后本论文在定义一个继承 WebLoader 的 NewsListWebLoader 对象，通过这个代码本论文就可以自动去获取 URL 对象并将其保存到本地数据库中：

```
Public class NewsListLoader extends WebXmlLoader{
    @Override
    Public String getCallUrl(){
        Return “HOST\news.php”
    }
}
```

本论文在 NewsListActivity 中，可以如此调用此 NewsListLoader 对象：

```
Private void InitDate(){
    NewsListLoader loader = new NewsListLoader(this);
    //设置回调函数
    loader.setCallBack(this);
    //多线程启动
    loader.start();
}
Public void WebLoaderCallBack(ArrayList<AbstractObj> lstObjs){
    //用户根据返回的数据内容显示用户数据，此处显示新闻列表
    ShowNewsList(lstObjs);
}
```

最终，本论文可以在安卓客户端中看到新闻列表和详情如图 35 界面。

至此，本论文的安卓平台的新闻移动客户端已经开发完成，开发者可以快速的添加几个类，就可以自动和远程数据访问接口中的数据进行交互。可见，本论文除了对安卓界面的代码进行构建以外，本论文并不需要添加太多的代码就已经完成了新闻模块代码

的构建。

图 35 新闻列表和详情

4.5 小结

从以上的安卓平台新闻系统的设计过程可见，本论文已经不需要为书写太多的代码就已经可以完成整个系统开发，本论文主要要做的事情是利用面向对象的设计思想把相关的对象和模型定义好，然后本论文利用 XML 模型管理系统来服务生成数据交互接口平台端以及远端数据访问接口端。然后在移动客户端对业务模型对象进行数据对象，数据访问对象和数据交互对象的定义，就可以快速完成移动客户端的设计，中间不需要过多的关注系统业务逻辑。

第 5 章 总结与展望

5.1 研究工作总结

本论文提出的面向移动应用开发的数据交互服务框架具有开发快，可继承性高，配置化等特点，它能够帮助开发者为互联网+的产品快速提供核心数据交互服务管理系统，并且提供了相关的移动终端数据交互方位 Jar 代码包。通过这样的方法，开发者可以在最短时间内基于业务系统来完成业务数据系统的设计，可以较快实现开发者的设计思想。

本框架能解决大多数移动应用的数据接口访问问题，实现过程简单，易于定制和优化，容易理解和学习掌握，可帮助开发人员提高开发效率，并简化代码管理和维护。本开发框架具有可拓展性，可应用于跨平台的移动开发中，实现起来相对简单，维护成本较低，是一个可行的面向移动应用开发的数据交互服务框架。

实现本框架所用到的前后端技术，都是本论文书写阶段的流行而且成熟的应用。PHP 已经成为 Web 开发中最常用的技术之一，据统计也是开发人数最多的语言之一，本研究使用 PHP 就是面向 PHP 的学习成本低，开源且部署费用低等优点。所以对于使用本框架的开发者来说，要实现和优化本框架，只需要掌握主流开发技能则可，容易理解和掌握框架开发的核心思想。开发者可以根据自身项目需要，对框架本身做更多的优化与定制，从而提高应用程序开发的效率。

对于框架的使用者来说，只需要基础的移动应用开发技术即可，不需要掌握困难的经验，因为平台的框架和代码都已经解决了开发过程中的大部分问题。例如在后端数据管理系统中的繁冗的增删查改的代码的开发，在相关的增删查改的过程中，又涉及某些字段需要各种需求例如唯一性等，本研究均在后台提供的服务中给予解决，用户只需要集中精力关注自己的业务对象就可以，大大地降低了开发门槛，提高了开发效率。而在移动应用的部分，解决了网络，数据传输，数据解析等问题，用户直接调用远程数据加载接口或者本地数据库访问接口即可以直接获取数据直接显示，根据自己实际的需求选择是否要存到数据库中，或者是否必须要 WIFI 状态下可用等，同样也大大提高了开发效率并减少了代码量。

5.2 后续工作展望

本论文依然有较多的地方需要改善，例如安全性问题，因为本系统兼容多数据库，所以难以做到兼顾这个系统防止用户注入的问题。因为本论文设计的系统主要基于追求性能的快和开发速度的快和开放性上面，所以减少了用户请求的校验，这对于数据系统的安全性来说是一个需要解决的问题[23]。

在后续的工作中，本系统将会加入私有云的概念并成立为一个平台，根据用户的申请自动创建一个服务器和相关数据交互服务系统，用户可以直接在私有云中添加自己的业务驱动模型即可以在相关开发的移动终端产品中访问相关私有云自动化移动业务管理系统和其数据访问接口，使其成为更加高效、免于部署的自动化业务管理系统。

参 考 文 献

[1] Abel Avram. Adobe Is Distributing Cordova under the PhoneGapBrand. http://www.infoq.com/news/2012/03/Adobe-Cordova-PhoneGap[N],2014

[2] Reference implementation for building RESTful Web services[N] http://jsr311.dev.java.net/nonav/releases/l.l/index.html,2013

[3] Meier R.Professional Android 4 Application Development[R],2014,5-24

[4] Imielingski T,BadrinathB R.Mobile wireless computing: challenges in data management[R]. Communications of the ACM,2014:18-46

[5] Feng Y,Fen GJ.Incremental Updating Algorithms for Mining AssociationRules[R].Journalof Software,2012:20-34

[6] G H Forman,J Zahorjan.The Challenges of Mobile Computing[R].IEEE Computer,1994:1-6

[7] Zina Pozen.WMI and.NET. MSDN Magazine,2002:12-16

[8] 闫晓亮.基于 MVC 设计模式的轻量级 PHP 开发框架的研究与实现[D].长春工业大学,2016:4-48

[9] 汪盛.基于轻量级框架的企业应用快速开发平台的设计与实现[D].电子科技大学,2012:2-26

[10] 所修文.轻量级 JAVA EE 框架的研究和实现[D].天津大学,2012:4-12

[11] 杨迪.基于 Android 平台的美拓暴风系统的设计与实现[D].哈尔滨理工大学,2015:2-58

[12] 邹煜.企业级移动应用平台建设与安全保障体系探析[J].网络空间安全,2016:95-98

[13] 王荣海.基于 HybridApp 技术的企业移动应用系统构建研究[J].软件工程,2016:35-38

[14] 廖成玉.企业级移动应用平台的选择与建设[J].现代电视技术,2015:64-88

[15] 王壮志.关于软件过程模型构建策略的分析和思考[J].内蒙古民族大学学报(自然科学版), 2014:72-74

[16] 王璐.Web 模式下基于 XMPP 的即时通信系统的设计与实现[D].北京邮电大学,2014:6-38

[17] 赵哲.面向开放平台的移动应用构建与推送引擎的设计与实现[D].北京邮电大学,2013:5-46

[18] 高嘉泽,高强,吴国全,魏峻,黄涛.面向移动应用的后端服务平台[J].计算机系统应用,2014:16-17

[19] 陈皞.移动应用系统的设计与实现[D].北京邮电大学,2012:18-43

[20] 王欣.跨平台移动应用研究与实现[D].北京邮电大学,2013:5-41

[21] 刘海岩,梁建龙.基于中间件的分布式系统开发过程的研究[J],《计算机应用研究》,2014:21-23

[22] 设计模式-可复用面向对象软件的基础[M].背景.机械工业出版社,2012:2-89

[23] 朱华旻.AJAX 在 WEB 开发中的应用研究[D].哈尔滨.哈尔滨工程大学,2013:15-46

[24] W3C.HTML5 differences from HTML4[N].www.w3.org/TR/html5-diff

致　　谢

时光飞逝，在深圳大学共度软件工程硕士的日子也已经到了尾声。回想起这几年的工作与学习的生活，各种困难与快乐，仿佛还历历在目。即将临别之际，在论文进入尾声，以此致谢，表达我的感激之情。

首先，感谢我的指导老师朱映映副教授。在论文的撰写过程中，朱老师以其严谨的治学态度，诚恳育人的心，给予了我许多的帮助和指导。朱老师对本文所提的宝贵意见，敦促我不断精益求精，孜孜以求，使此篇论文得以更加严谨和完备，成为一篇合格的学术论文。朱老师不论在治学上还是为人处事上，都是我学习的榜样。

其次，感谢学园的各位老师和领导，感谢他们的帮助和鼓励，在上课的过程中，他们的课程给我流下了深刻的印象，我在其中获益良多，帮助我在工作中能够以更深刻技术积累，使得工作更进一步。

最后，在共度软件工程硕士的过程中，我结实了许多志同道合的同学们。他们来自五湖四海，从事不同的专业工作。在于同学们的交流中，我获取了别处获取不到的实战经验，从他们身上，学到了很多书本上学不到的知识和经验。我非常珍惜这段同学友情，他们是我人生难得的财富。走出校园，友情还将延续，人生也因而更加丰富灿烂。