第1章 绪论

随着互联网产业越来越发达，并且移动互联网的崛起，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 80，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，并且基于次概念推出“大众创业，万众创新”的观点。越来越多的人加入互联网+产业的大军进行创业。

1.1 课题背景

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。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，移动终端或者其它微服务等。人们的日常生活已经离不开互联网。于此同时，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。

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或交互。那么，如何快速地把这套核心并基础的信息管理构建出来就成为了争分多秒的关键了。快速简单的为系统构建基础数据的信息管理系统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信息管理系统和其相关的数据接口的管理时间。

1.2 课题任务

如何为创业者提供提供快速，简洁和高效的互联网产品核心的信息管理系统是本课题的主要关键。从一个想法到实现，其中结合面向对象的设计，生成出一个一个对象，并将其实体化，是本课题研究的重点。

1.2.1 课题内容

本课题的主要任务是研发一个基于 XML 模型的自动化构建的信息管理系统，并且利用此系统提供数据接口服务以便于其它相关终端进行数据的交互。并且针对不同的技术终端提供数据交互的插件，做到用最简单的方式构建一整套的信息管理系统和其相关的客户终端。最后，利用本文提供的技术，生成简单的安卓端新闻系统。

本论文则主要阐述其相关的开发过程包括需求分析，概要设计，详细设计和实现等，最终完成系统和事例程序，并对该套核心系统进行展望。

1.2.2 本人承担任务

本人在此系统的中承担所有角色的任务，包括系统的研发，需求设计和设计等所有工作，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，用于对信息管理系统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。

然后我们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 论文结构

首先，本论文首先阐述为什么要研发此系统的课题背景和介绍。

第二章则简单介绍相关使用的技术。

然后从三到七章分别通过需求分析，概要设计，详细设计，代码实现和测试来描述本文对基于 XML 模型的信息管理系统的研发过程。在此过程中，我们将详细的通过需求分析来分析本文开发的信息管理系统将会提供什么业务和什么服务，最终让读者知悉本系统的功能和要求。其次，通过概要设计来从总体上规划整个信息管理系统的蓝图，并通过详细设计对每一个子系统进行详细的功能规划，然后进行代码实现得到目标系统，最终通过测试来最后验证本文所开发的系统是否完成预期规划和可用性。

最后一章则对此系统进行未来的展望和总结系统的不足。

第 2 章 相关技术

介绍论文中用到的所有重要技术。对其原理及使用方法做简单介绍

2.1 XML

XML（Extensible Markup Language），可扩展标记语言。在 1998 年 2 月正式被 W3C 批准为一种标准的数据传输格式。用户可以利用其规定的格式结构生成相关的数据文件，其优势是便于传输和存储，并且基本上能否符合对象的各种表现形式，是当今最常用的数据传输格式之一。其优势如下

1） 兼容现有格式，其便于利用 HTTP 协议进行传输。

2） 标准的数据形式，便于不同应用系统间数据的共享和互通，在各种语言和平台中均提供的标准的读写的支持库代码。

3） 完整的格式，使其简单方便的判断数据传输的完整性，不会因为数据传输断续带来各种数据破坏或者错误的可能。

本文主要利用 XML 模型来定义信息管理系统中的数据模型，通过定义模型的字段，属性等，来快速生成信息管理系统中各个信息管理数据的模块的功能。系统通过读取相应的 XML 模型，自动生成页面上的不同形式的字段或者在结果 API 中显示相应的字段。同时，也利用 XML 格式来传输数据给各种移动终端包括 Web 端，Android，IOS 端等转化为需要的数据在自己相应的应用中展示出来。

2.2 PHP

PHP（Hypertext Preprocessor），超文本预处理器。他是一种常见于生成网页脚本的语言。其特点是开源并且高度融合其它语言的特点。速度快，代码架构简洁快速，比起其它语言，能够更加快速生成 Web 页面。结合常用的模板技术形成简便易用的 MVC 结构。同时，因为 PHP 作为一种开源代码，所以其有丰富多彩的开源库和开源可用，其直接带来的好处便是其支持多样的数据库和各种其他相关的业务模式例如数据交换等可选择 Webservice，XMLRPC，或者 Soap 等。

本课题使用 PHP 作为信息管理系统的核心代码语言，并且也用于利用其生成 API 数据接口。

2.3 数据库技术

数据库是一种在 60 年前出现的用于存储数据的仓库。在上世纪 90 年代以后，数据库不仅仅只是存储和管理数据，并且已经转成成为数据库的管理者和拥有者的角色，其已经和对应的业务流程结合。

本文的课题的核心是信息管理系统，其最终就是通过简便的方式使得用户可以简单创建一个系统并且自动提供增删查改的方式对业务数据进行管理。所以，本课题的研究是适用于各种数据库包括 MSSQL,MYSQL,SYBASE,ORACLE,SQLITE 等各种主流数据库。同时，安全性也是本课题的研究重点之一。

2.3 数据库技术

数据库是一种在 60 年前出现的用于存储数据的仓库。在上世纪 90 年代以后，数据库不仅仅只是存储和管理数据，并且已经转成成为数据库的管理者和拥有者的角色，其已经和对应的业务流程结合。

本文的课题的核心是信息管理系统，其最终就是通过简便的方式使得用户可以简单创建一个系统并且自动提供增删查改的方式对业务数据进行管理。所以，本课题的研究是适用于各种数据库包括 MSSQL,MYSQL,SYBASE,ORACLE,SQLITE 等各种主流数据库。同时，安全性也是本课题的研究重点之一。

2.4 Android 平台开发

Android 是一种基于 Linux 的开放源代码的操作系统。现在已经成为主流的移动终端操作系统之一。其市场份额已经达到全球移动终端操作系统的 8 成。其优秀的的 UI 及显示，资源管理和进程管理等操作系统必要因素都一流使得其快速被应用。尤其是其基于 Java 的安卓开发平台，更是被广大 Java 程序员接受。

本论文中，我们会开发安卓开发相关的 Jar 包，使利用我们自动业务系统的用户能够利用系统的核心资源类快速进行开发。其包括数据访问对象（DAO，Data Access Object），数据对象，远程数据接口数据读取类，远程数据解析类等。使得开发者直接通过配置即可简单获取自动业务系统提供的数据。然后根据自己的需要把数据展现给用户使用。

2.5 SOAP 协议

SOAP 是一种基于 XML 的简单协议，主要用于方便用户在 HTTP 协议的基础上进行数据的传送。因为其基于 XML，所以其有稳定的格式和高效的可读性，并且因为其完整的闭合性格式，所以开发者在使用 SOAP 协议进行数据传输的过程中不需要对其数据完整性进行检查或者说如果非闭合性的完整结构则是不可读的。同时，SOAP 比起 WebService，XMLRPC 等同类型基于 XML 的协议来说，其更加高效和简洁，传输的数据更少。是经常在互联网中被传输的数据载体。

在本课题中，我们使用 SOAP 协议进行安卓客户终端对信息管理系统的数据的读取。通过解析其协议的特殊业务格式然后将其分解为特性的数据对象并让安卓客户端进行数据展示。

2.5 HTML5+JQUERY+Boostrap

现今，HTML5 已经成为大多数浏览器的共同协议，其实 HTML 协议的第五次重大的修改，增加许多的新功能以及核心概念。其增加了许多新的特增，例如本地存储，设备强兼容性，连接特性（更强大的页面远程访问功能，例如 Server-Sent 和 WebSockets，在大大提高 AJAX 效率的情况下更加了更多的远程连接服务特性），网页多媒体特性，更强大的三维解释特性。最大的特性是支持 CSS3，使得其页面的应用展现更加丰富多姿。

基于 HTML5，其相关的支持的第三方插件或者工具应运而生，其中，我们使用了 JQUERY 脚本框架来作为我们的页面辅助开发语言，在 PHP 中通过模板技术把相关数据输出到页面中，然后通过 JQUERY 的简易语法结合使得页面出现各种各样的操作。另外，我们添加由 Twitter 设计基于 CSS3 的前段开发的开源工具包。其简洁灵活，使得 Web 开发更加快捷，并且不在需要开发中在开发过程中关注 UI 的问题，因为其标记了一系列的 CSS 格式，开发者非常简单就可以使用并且有十分简单但是优雅的 UI。

本课题的信息管理系统是一个 Web 系统，用户将利用 Web 界面来管理和操作用户的业务数据。那么，Web 系统的界面的外观以及性能则成为本系统是否能否被开发这广泛接受的一个关键。本系统结合 JQUERY 和 BOOSTRAP 在 HTML5 的基础上进行界面上的开发。

第 3 章 需求分析

需求分析在软件开发模型中的地位非常重要，它的任务重中之重，就是为了全面地理解用户的各项需求，同时能够准确地对用户所提出的需求进行表达。

事实上，需求分析的首要任务就是问“做什么”，通过一系列调研和客户沟通来弄明白这个问题。需求分时其实指的就是理解用户的需求，参考客户的需求分析出最终软件将会实现何种功能而满足用户。

本章将详细介绍该快速集成系统的需求分析阶段的工作。

3.1 总体需求

在本课题中，我们面向的用户为将会在互联网+产业中创建新产品的客户，其客户可能将会创建 O2O 平台，P2P 平台等产品。在本论文的设计中，本论文面向的用户将有一定的技术背景，至少需要如何进行各个系统的开发，本课题的研究成果主要是为了能否解决开发用户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业务数据管理系统。对于这些客户，我们要为他们解决以下几个问题

1)快速构建一个业务数据管理系统，其业务数据应具有简洁高雅的用户界面设计，快速方便的页面加载速度和操作方式，高通用性和可拓展能够简易进行二次开发。

2)基于业务数据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，我们需要在各种终端中进行客户端的开发，例如 Android 端和 Web 端等。本课题将会基于 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，自动链接数据库构建表等一系列通用功能，更敏捷的辅助用户进行开发。

3.2 业务数据管理系统功能需求

为了能否服务于互联网+产品的开发者，本课题将会研发一套让开发这快速就能使用的业务数据管理系统。此业务系统可以让用户只需要配置相关的 XML，用 XML 来标记其

产品系统中的业务实例，然后系统中设置对应的请求页面，使请求的页面知道应该去读取哪个相关的 XML 数据业务模型。之后当用户用网页打开此配置好服务器的 PHP 页面后，则自动能够提供对该业务数据对象出现增删查改的功能。以下是功能简单说明：

1)用户定义相关的符合定义的 XML。通过 XML 数据模型的定义，例如是否在列表搜索中显示某个字段或者某个字段是否可以作为搜索字段被搜索。又例如定义字段类型等，让用户在单条数据页面编辑页面输入相应的数据，例如文本，数值，图片亦或是下拉列表或者多选框等。

2)针对用户定义的 XML，然后通过 PHP 语言的解析，然后在页面中显示列表页面，编辑页面，添加页面，删除功能等相关功能。相关增删查改等功能应该是能否在 XML 定义并直接表现在相关页面中。

3)本系统应该具有高可扩展性，应该从几个方面可让用户拓展。首先，在前端页面显示中应该是可拓展的，用户可以自己添加 JS 或者 CSS 从而增加 UI 上操作的通用逻辑。其次，在后端数据处理代码，也就是 PHP 代码中，也应该是可拓展的，用户可以根据自己的实际情况判断某个字段是否允许被录入，或者数据的正确性或者其他相关的业务流程等。

4)本系统支持多种主流数据库，包括 MSSQL，MYSQL，SYBASE,ORACLE,SQLITE 等。由于各个系统支持的语法略有分别，所以将采用简单工厂模式的方法来生成需要的数据库以方便用户可以根据自己系统的情况选择相应的服务器。

5)本系统支持各种前端 UI 插件的拓展，例如拓展添加日期控件，扩展添加 FullCalendar 等插件。也支持后端各种插件的拓展，例如添加发送短信功能，读取 Excel 功能等。

6)系统将会首先预定义用户管理的增删查改的定义，以便用户学习并且可以快速进入系统。

7)系统将在增删查改的基础上，提供数据接口交互服务，用户可以通过该定义的 PHP 页面请求数据以 XML 的方式利用 SOAP 协议返回数据，供各个终端业务系统返回相应的数据。

8)系统支持多语言的，换句话说，就是相关的文本或者长文本字段支持用户提交多语言。在系统中将以 1 对 N 的表结构映射到多语言表中。

3.3 终端系统数据交互核心代码类库需求

在本课题中，我们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。

本课题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。

本课题讲主要针对安卓客户端进行 JAR 代码继承包的编写，使用户可以简单继承。

3.4 辅助工具功能需求

为了能否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业务数据管理系统，本课题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，根据其以及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。

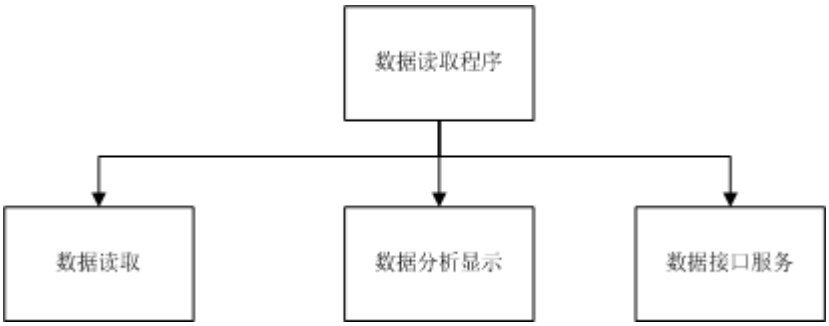


图 6

3.5 性能需求

因为本系统是为了让用户最终可以在移动终端上让数据让用户读取，所以我们需要保证尽可能多的移动终端请求数据并且或者数据正确性。所以性能需求在数据接口的请求中需要被测试并且提供相应的数据，证明本系统对数据接口的访问性能能否满足用户的业务需求。本文将会用 JMeter 工具来进行批量的请求测试来测试请求峰值和均值。

自动化业务数据管理系统则无性能需求。

3.6 安全性需求

本课题的研发结果需要注意安全性的需求，首先是业务数据管理系统上的需求，必须通过用户登录之后进行操作，已经相应的角色权限管理，防止各种安全性注入等安全性需求。

在数据接口方面，则需要建立密匙和公钥的机制，移动终端拥有密匙，数据接口则拥有公匙，当公匙密匙相匹配才会进行数据的交互。

第 4 章 概要设计

概要设计的主要任务是把软件系统划分为各个模块并且确定其模块层次结构。根据数据流向和数据对象的需求分析，设计数据库形成数据字典和 E-R 图。笼统地来说，就是为了从一个大的方向把握将要开发的软件系统将要实现哪些模块以及用怎样的方法来完成该软件系统。这个过程通常以生产概要设计说明文档为标志，其包含一些子文档例如数据字典、E-R 图、软件开发模型、概要设计说明书等。

概要设计内容主要包括以下几个方向：（1）确定设计方针和方法，说的是选用何种设计方式来完成整个软件开发过程，一般有瀑布型或者迭代性等类型。（2）将所得的需求分析说明书分解为具体的项目需求和设计为项目的实体，通过一系列详细的文档例如数据字典、E-R 图、软件开发模型、概要设计说明书、模块流程图和接口设计等文档表现。

4.1 系统整体结构

本课题研发的系统主要是一个自动化的业务数据管理系统，用户通过定制 XML 格式的 XML 模型来自动生成目标业务数据管理系统。然后这个系统提供外部接口让各个业务终端系统进行数据的读写和业务交互。从而满足用户快速的定义一个业务数据管理系统和简单让其为自己的业务终端进行业务数据的交互操作。

本课题研究的自动化业务数据管理系统主要分为几个大模块，然后几个模块基本上可以实现以下功能。

4.2 业务数据管理系统模块设计

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业务数据管理系统，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。主要有以下几个模块：

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。

模块名称	业务模型业务实现模块
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

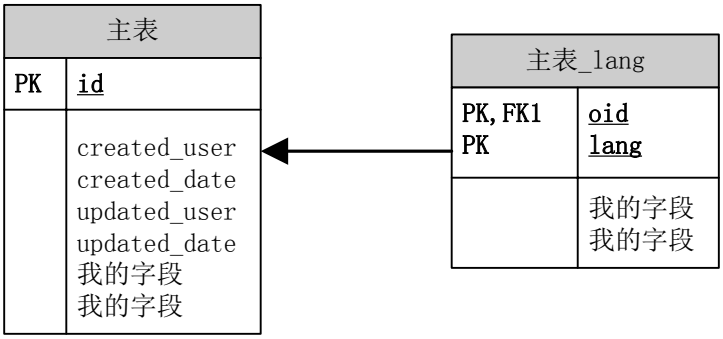
模块名称	业务模型数据接口访问模块
功能描述	本模块主要通过加载已经生成的业务模型对象，或者用户自己生成的业务数据，自动生成 SOAP 协议的 XML 文件流返回到页面中让不同业务中断读取
接口与属性	用户通过传入业务模型对象以及相应的请求动作以及必须的请求参数，开发者可以返回增删查改形式的基本 XML 数据，也可以根据用自己组织业务数据，自动返回用户特殊的业务数据请求

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

模块名称	移动终端业务交流模块
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

4.3 数据库总体设计

本课题研发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，则自动生成以下基础表机构的内容以及用户自定义的字段。

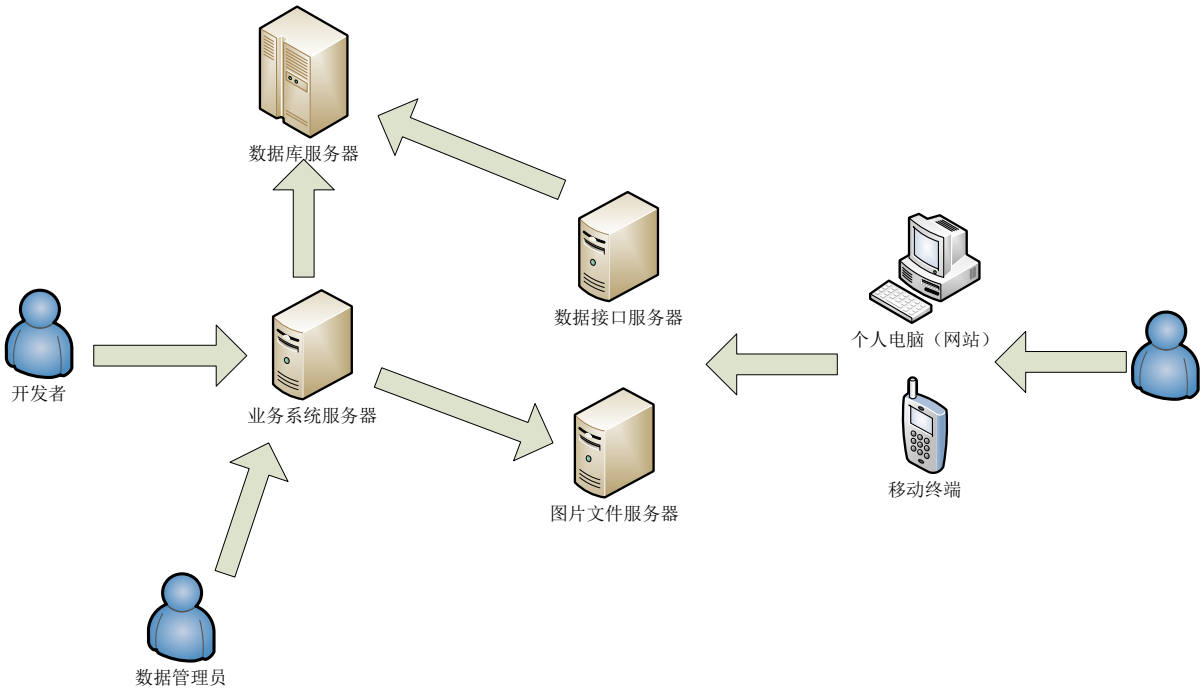


4.4 系统部署架构

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。

推荐部署架构如下：



第 5 章 详细设计

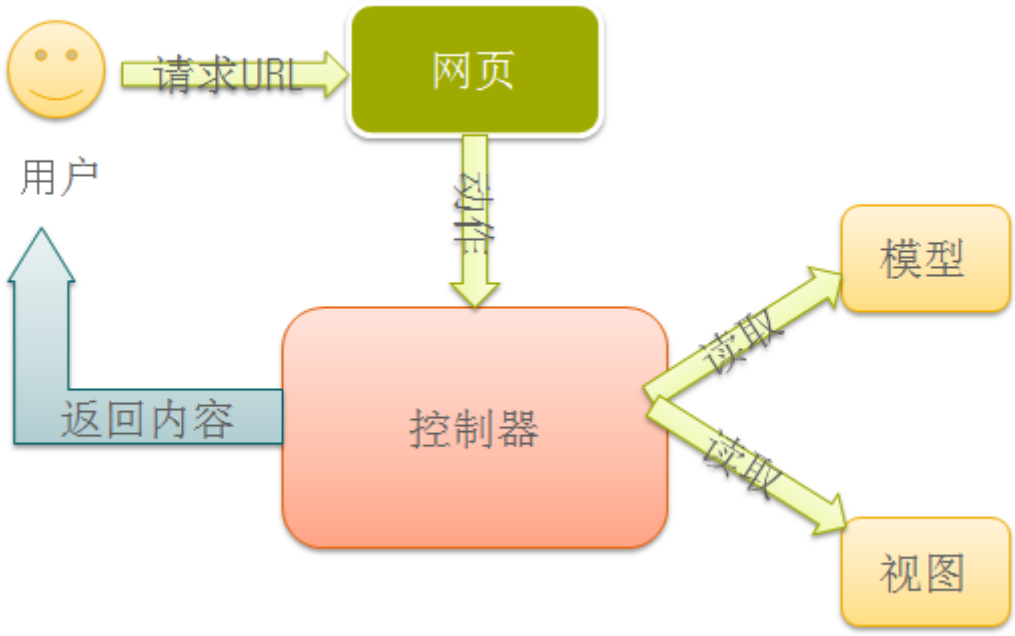
详细设计则是要详细的把系统的每个模块的控制流程、内部算法和数据结构等的设计，基本上在这个过程以及把整个系统所有用到的方法和对象全部列出并能够列出所有方法从开始到结束的一系列调用等。良好且完整地做好详细设计的工作，则可以为将来系统实现步骤实现起来更简单。

详细设计内容主要包括以下几个方向：（1）将所得的概要设计过程中的说明书分解为若干个子系统并确定各个模块的目标例如输入输出结果等。（2）模块功能流程确定，基本上可以描述出该模块的输入输出，并且给定相应的方法来简单阐述如何实现该模块。

再第 4 章的概要设计中，我们已经设计出本信息管理系统会有以下几部分：XML 模型读取模块，业务模型业务实现模块，业务模型数据接口访问模块，XML 模型管理模块，移动终端业务交流模块以及数据库设计。在本章中，我们将进一步分析各个模块应该有那些功能和接口以及其实现方式。

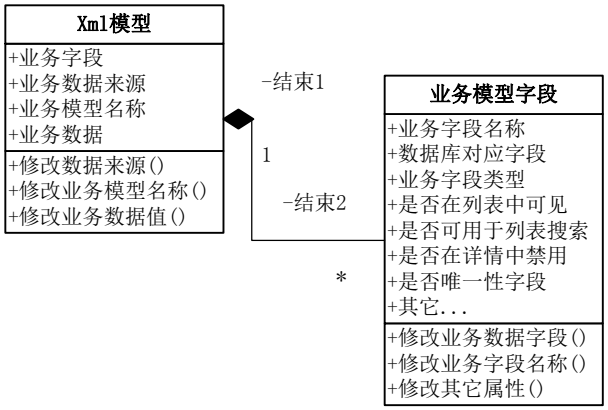
5.1 信息管理系统总体设计

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，我们首先遵循 MVC 的设计模型。众所周知，M 为模型（Model），V 为视图（View），C 为控制器（controller）。我们这个信息管理系统在业务管理模块端则一样是基于 MVC 模型的设计规范，首先我们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。全局设计图如下：



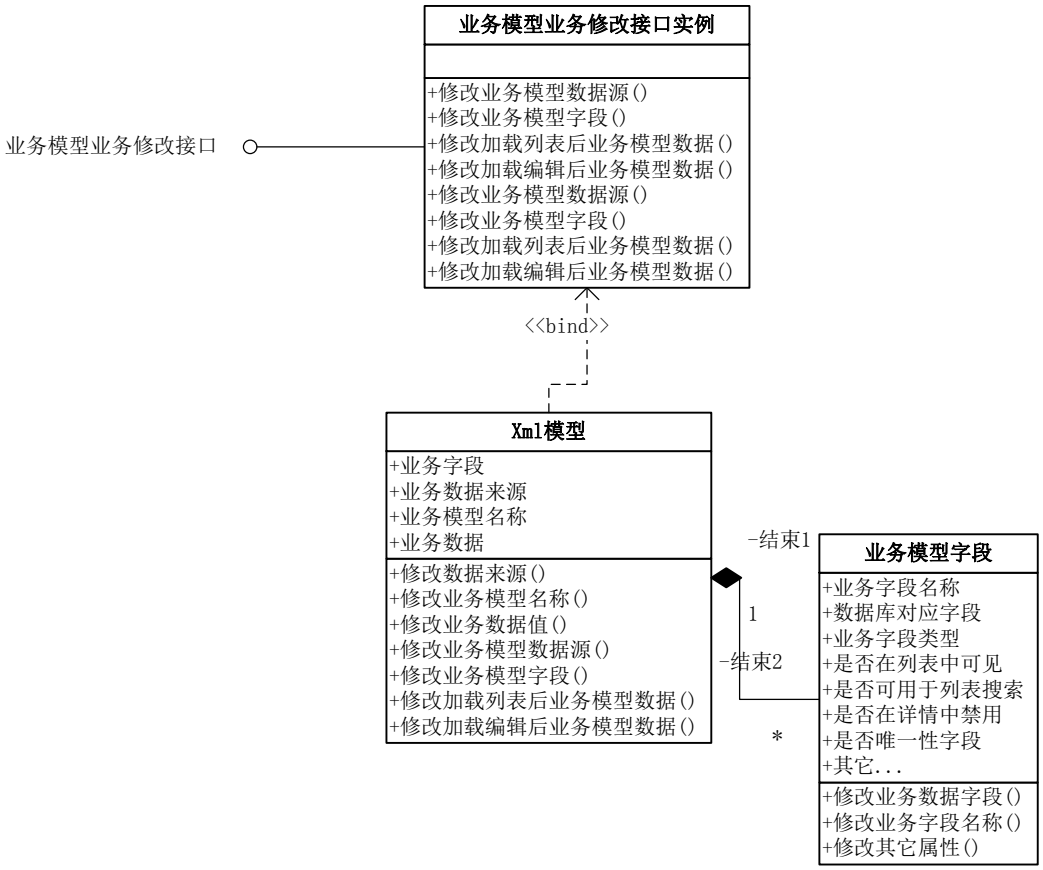
5.2 XML 模型读取模块详细设计

那么，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，我们可以获悉他相关的数据源（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，我们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。我们尽可能通过查看 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。类设计如下：



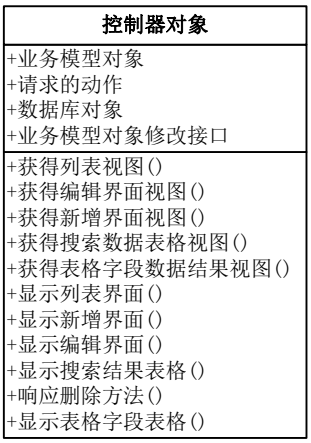
同时，某个业务字段可能本身就是要去读取业务数据或者根据特殊业务需求动态修改显示的数据，显示的字段或者其他业务需求等。在此，我们可以添加一个接口，让用

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。

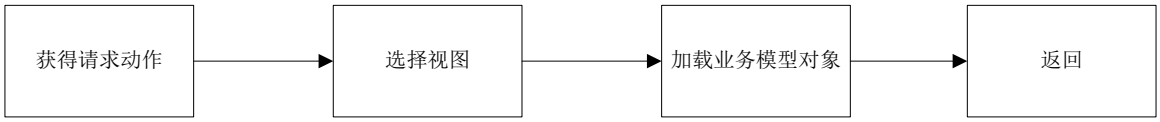


5.3 业务模型业务实现模块详细设计

业务模型业务实现模块主要是为 MVC 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，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，我们可以设计出控制器类如下：



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，简单流程图如下：



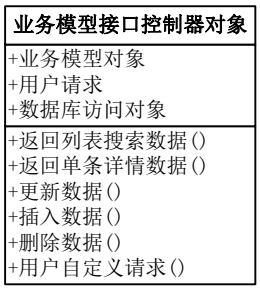
此控制器对象的方法可以方便被重载，以方便用户进行二次开发。

5.4 业务模型数据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，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，另外就是响应自定义用户请求。

类对象设计如下：

其请求过程相对信息管理系统模块更加简单，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，以便于第三方平台开发的软件能够以响应的 XML 格式进行读取。



5.5 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信息管理系统，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。

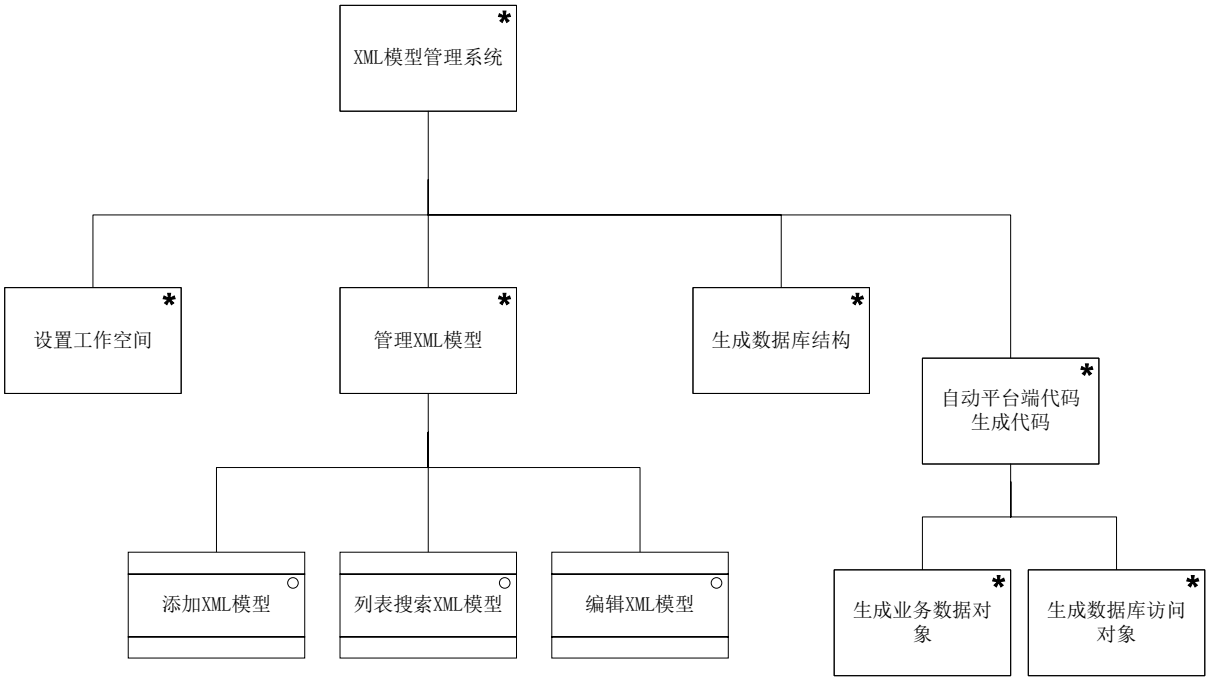
首先是设置工作空间，由于本系统设计的 XML 模型为视图的 XML 文件对象，所以讲所有的 XML 文件都集中存放于系统的固定目录。那么本系统首先设定工作空间，则可以自动加载工作空间下面有效的所有 XML 文件并且转换为 XML 模型。

当我们得到所有 XML 模型之后，我们就可以为开发者提供增查改的功能了。同样的，我们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，我们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，我们可以更加快速的开发和生成信息管理系统。首先我们读取 XML 模型，首先生成基础表字段，基础表字段的定义在第 4 章第 5 节中有提起，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，我们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。

下图为基本的功能设计：



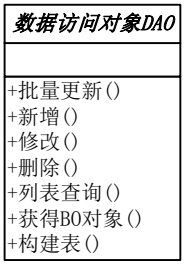
5.6 移动终端业务交互模块详细设计

移动终端实际上是最终的目标开发程序，当今互联网产品的开发，离不开开发移动互联网端的产品提供给用户对于网页意外的体验。而且，之所以开发一个快捷的信息管理系统，目的就是为了可以快速为移动终端平台或者网站平台提供业务数据的支持以及提供一个数据访问接口方便开发者开发移动终端业务模块时，对远程数据服务器的访问。

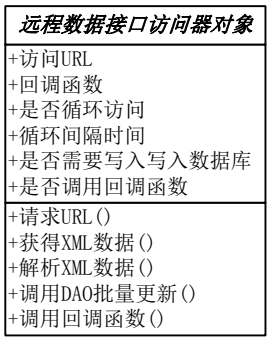
本论文将会以安卓平台为目标，开发一个可以被用户使用的 JAR 包，用户可以把这个 JAR 包添加到开发项目中，然后继承一些业务对象，例如抽象的业务数据对象，抽象的数据访问对象和网络交互对象等，重载其指定的相关的抽象方法，来指定移动终端对象的用户开发模块应该去读取数据访问接口中的相关数据。

首先我们会定义一个抽象业务数据对象，Business Object 对象。此对象主要用户装载业务数据，以方便开发者在开发过程中对业务数据对象的处理和传输，但是其需要负责两个主要的功能，解析 SOAP 传送的单行数据到当前对象的相关业务数据字段，和解析数据表游标的数据到当前对象的相关业务字段，对象定义如下：

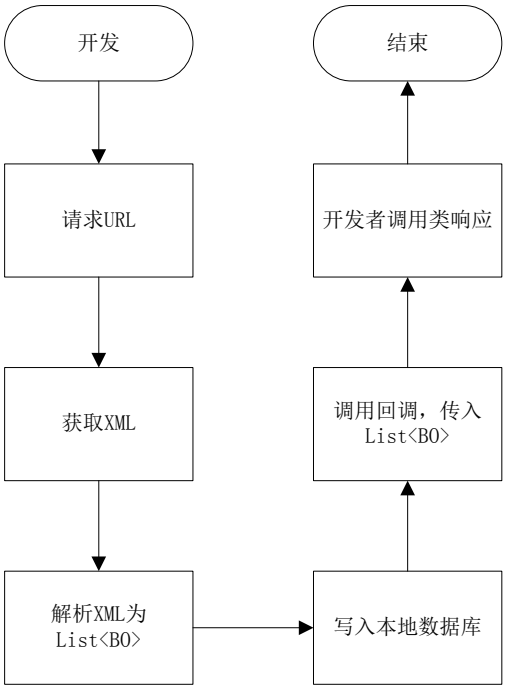
首先我们会定义一个抽象的数据访问对象，Data Access Object 对象。此对象主要提供对某个业务数据的增删查改功能。对象定义如下：



然后我们需要定义一个远程数据接口访问器对象，此对象主要有几个抽象方法等待用户实现，首先需要定义相关调用的数据访问接口的超链接，只有定义了访问相关的超链接，我们才可以准确的去响应的 URL 中获取 SOAP 协议的 XML 对象返回到本地。然后利用 BO 对象的解析 XML 功能，将其加载为一个 BO 对象，然后根据需要，我们需要设定其是否需要更新到本地数据库，此处也是一个抽象方法，用户需要返回是否需要调用本地。最后则是加载数据或者写入本地数据库成功后，我们需要定义一个抽象方法让用户实现是否需要调用回调函数让用户立即去处理用户相关数据。对象定义如下：



最后，我们设计一个简单流程用以描述从请求到结束返回的过程：


开发
请求URL
获取XML
解析XML为
List<BO>
写入本地数据库
调用回调，传入
List<BO>
开发者调用类响应
结束

第 6 章 编码实现

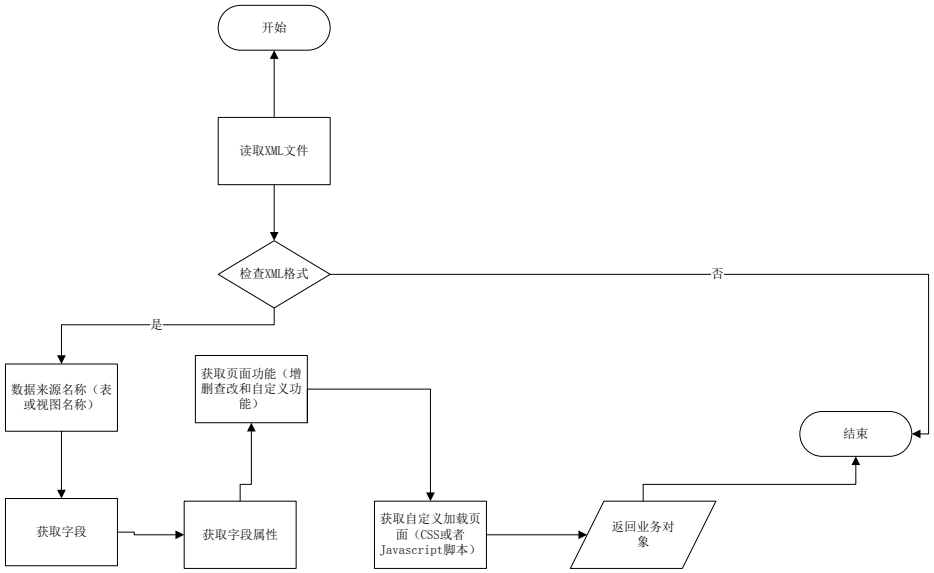
所谓实现，就是利用已经获得的需求和设计和算法等，利用工具编写程序代码，最终将代码转化为目标软件结果的过程。在本课题中，将设计几种语言的编码，其中，我们将选择相对重要的代码以简单的介绍开发过程。

主要涉及几部分代码，业务模型系统的代码实现，数据访问接口的代码实现，安卓端核心 JAR 代码实现。

在本论文中，我们主要以程序流程图的方式来分解各个模块的模块是如何实现，以便于了解在本系统生成的过程中想法和难题。

6.1XML 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。首先我们读取用户表示的 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，然后我们首先读取器数据来源字段和业务数据名称字段，然后读取相关的业务字段，然后读取定义的业务模型功能，例如是否有允许新增，允许修改，允许删除等，最后返回业务模型对象。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，在第 5 章中已经有说明。以下是其程序流程图的：

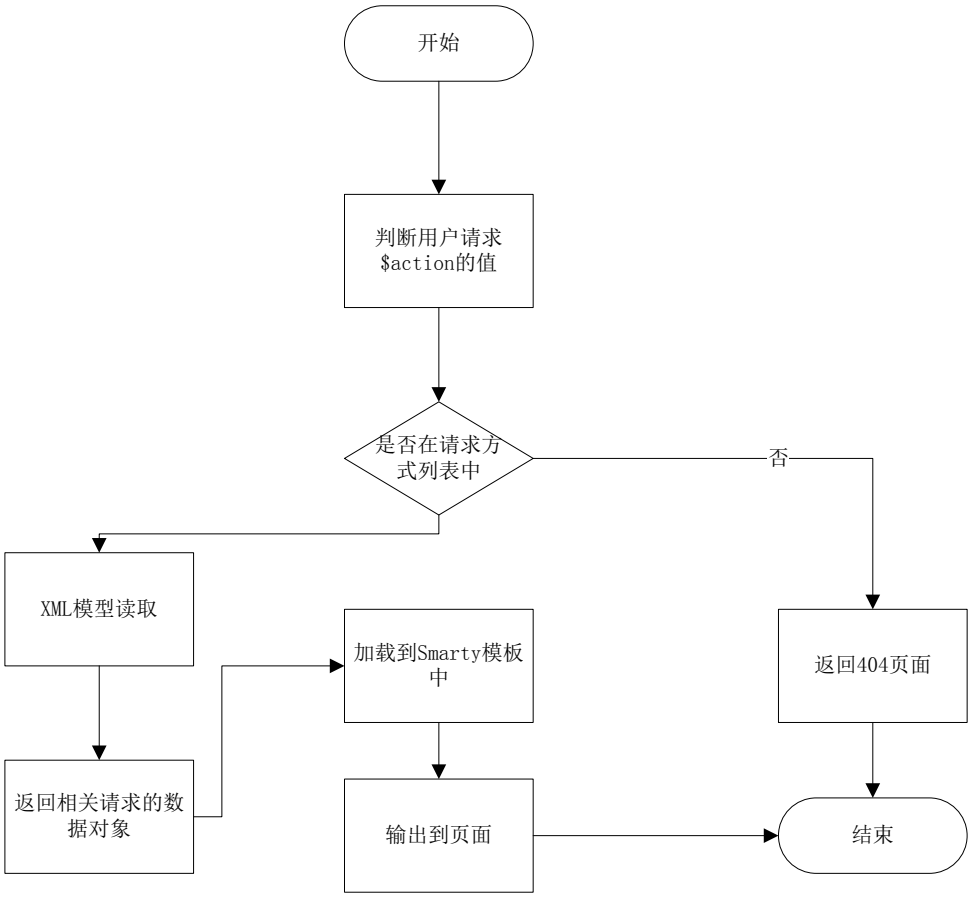


6.2 业务模型业务实现编码实现

本模块的编码实现中，主要我们暂时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：

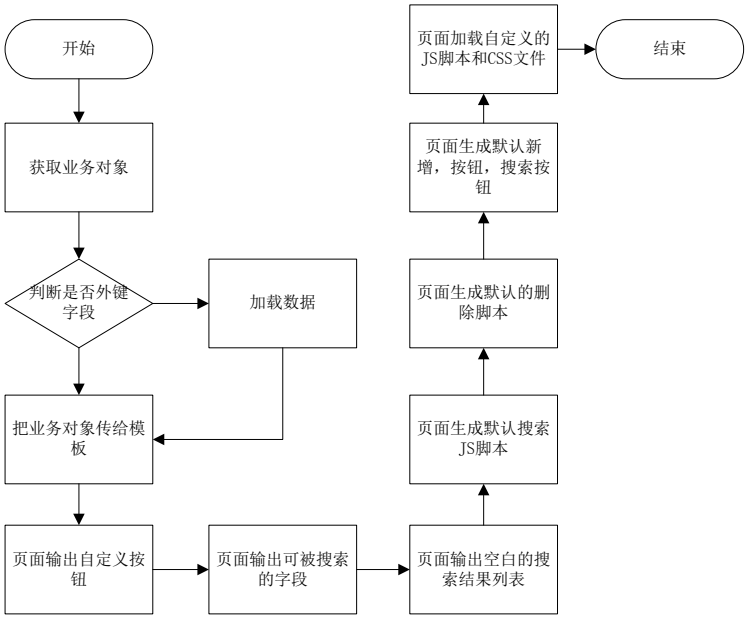
首先我们接受用户请求的动作$action 参数，判断其是否常规的定义，例如“”，默认为显示列表界面，“search”搜索数据返回到列表页面中，“grid”搜索数据返回到表格字段中，“new”显示新增业务对象的业务数据新增界面，“edit”，加载相关数据并显示业务数据编辑界面，“delete”删除用户数据，或者其它，进入用户自定义方法的业务逻辑中。然后加载这个控制器相关的业务模型对象，如果需要加载数据，则根据是请求生成相应的数据对象，然后把业务模型对象和数据对象一起加载到相应的视图对象中，利用视图中本身签入的业务逻辑代码返回相应的 html 源代码，返回到用户界面中。

首先是整体的程序流程

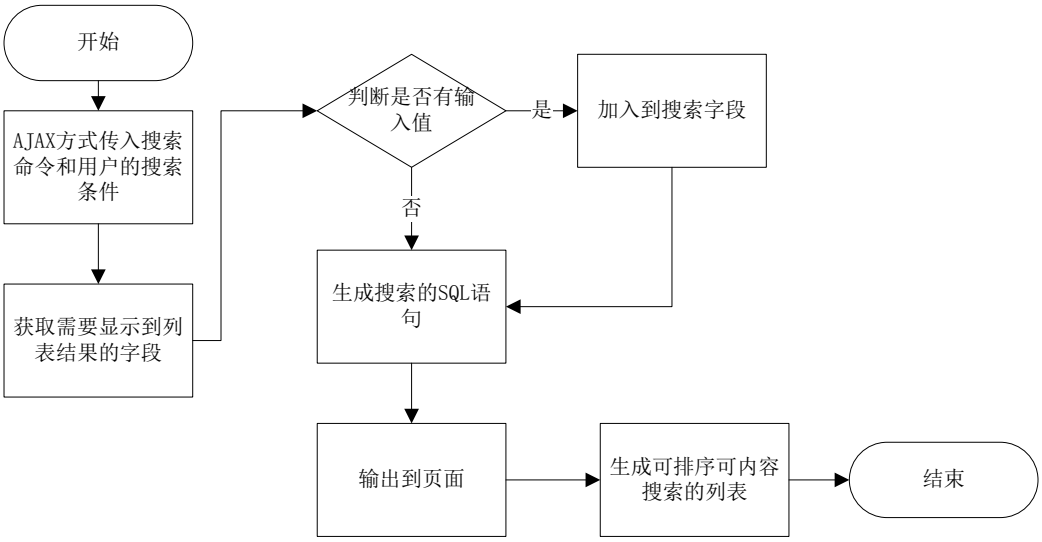


更加详细的分解响应列表显示界面的程序流程，首先我们获得业务模型对象，然后判断其相关的外建字段，在本系统中，我们需要定义集中外键字段，例如下拉列表的数据源可能是另一个表的数据而不是预定于的简单选择字段。多选组合字段也同样可能来源于其它表的数据。所以，首先我们需要加载相关的外键字段的数据，然后在把当前业务模型对象传入业务模型修改接口实例进行自业务的业务逻辑修改加工，然后则开始把业务模型输出到列表视图中。

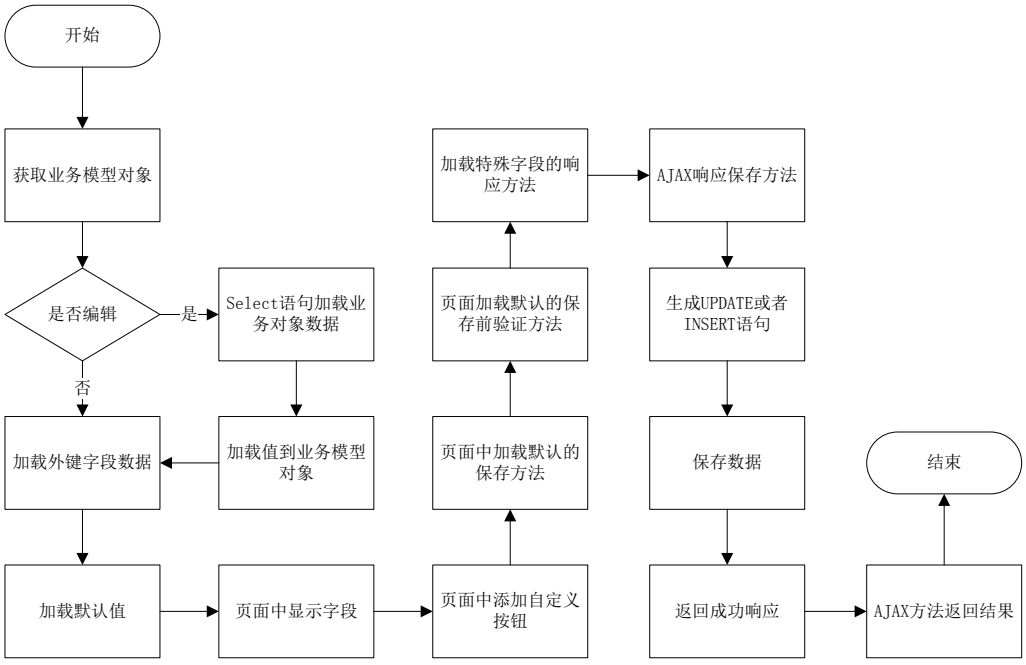
在列表视图中，搜索我们读取业务模型对象加载业务模型管理功能，包括自定义功能和预定义增删改功能。然后加载业务模型定义了 search 为 1 的字段显示到界面中。最后完成其他相关默认的 JS 脚本的编写，例如在这个页面中，我们需要响应用户点击搜索按钮后的代码，点击删除后的代码，点击新增或者编辑后的页面 javascript 代码。如果用户定义了自定义的通用功能和页面相关业务逻辑,我们则需要用户自定义一个 js 文件设置到业务模型对象中，然后开发者就可以通过书写相关的 JS 代码来结合自己业务逻辑的流程实现更加丰富多彩的业务管理需求。



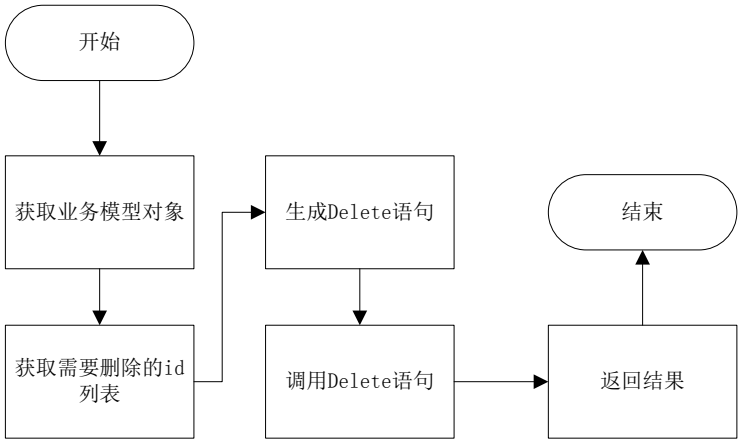
响应用户点击搜索按钮的程序流程图



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，我们则生成 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



响应删除的程序流程图



6.3 业务模型数据接口访问编码实现

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,我们有以下流程，首先依然是解析请求类型：

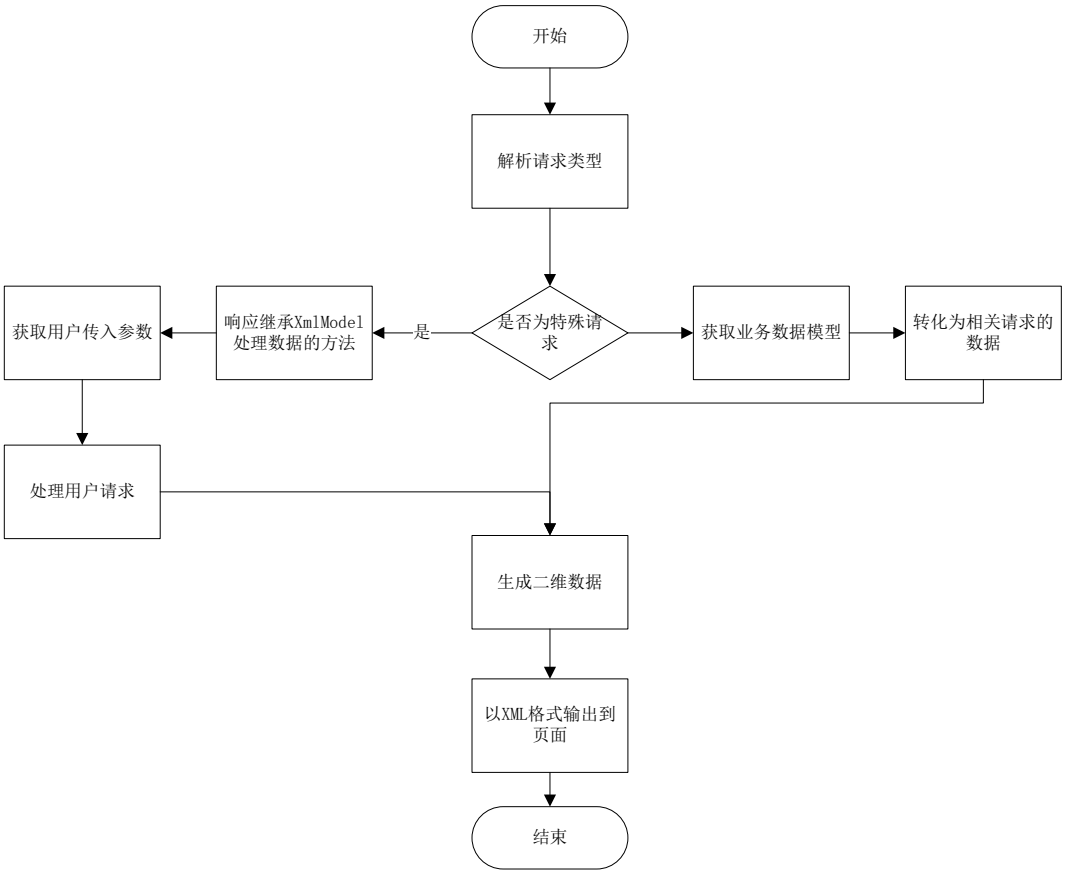
“”默认为请求列表数据，请求列表数据与在信息管理系统中点击搜索按钮类似，其不同结果为不需要在最终加载到视图中，而是直接把在数据源中获得的数据直接转换为 XML 数据格式，返回到页面中，返回给请求方。

“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，我们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，我们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。

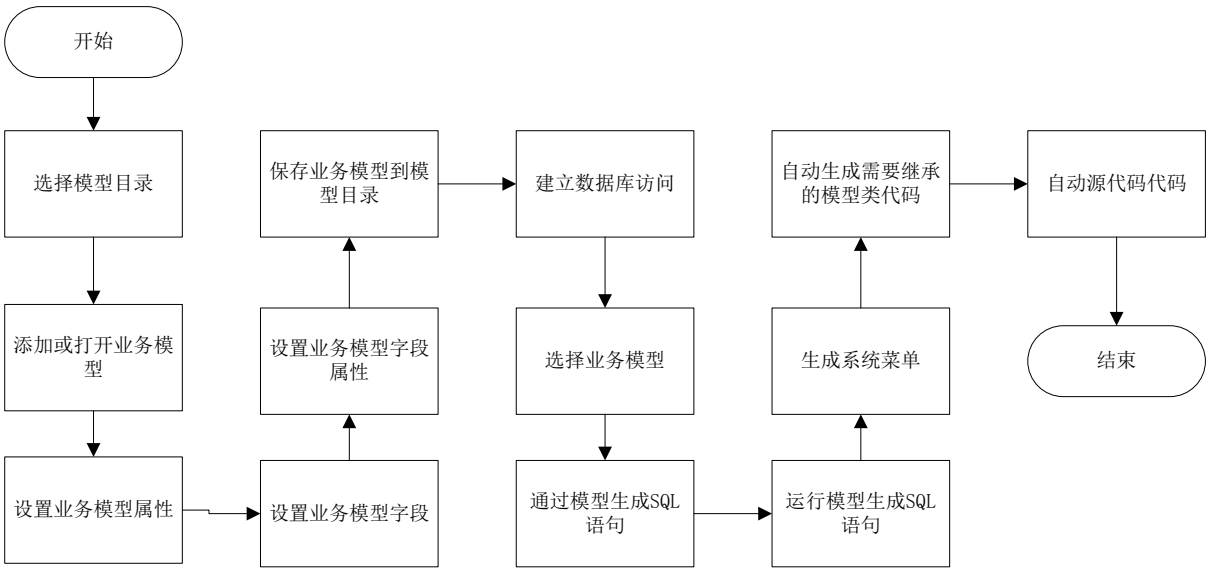
“delete”则为删除单条或者多条数据，其结果依然返回结果类型的固定格式。

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。



6.4 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，同事方便开发者进行管理。

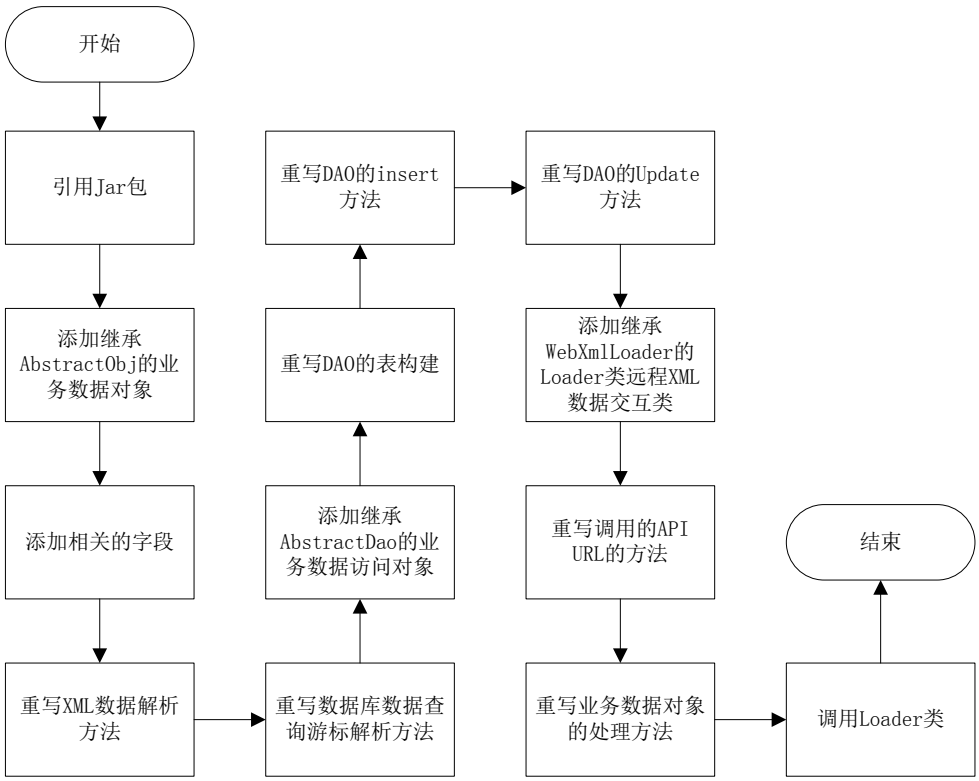


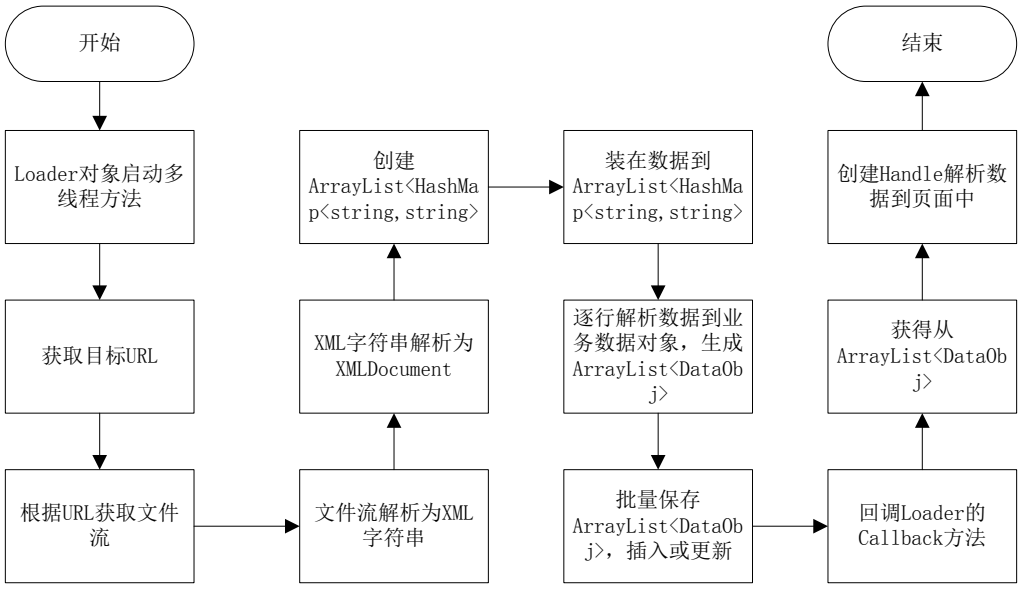
6.5 移动终端业务交互模块编码实现

在此模块中，我们将针对安卓平台设计一个 JAR 包方便用户直接调用 5.3 中生成的 API 进行交互。分为两部分，第一部分，当用户着手开发客户端时，应该首先需要做什么事情以完成开发。第二部分则描述程序流程。

首先我们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，我们在 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。

最终，根据其相关的 DAO 类中，我们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。




开始
Loader对象启动多线程方法
获取目标URL
根据URL获取文件流
文件流解析为XML字符串
XML字符串解析为XMLDocument
创建ArrayList<HashMap<string,string>
装在数据到ArrayList<HashMap<string,string>
逐行解析数据到业务数据对象，生成ArrayList<DataObj>
批量保存ArrayList<DataObj>，插入或更新
回调Loader的Callback方法
获得从ArrayList<DataObj>
创建Handle解析数据到页面中
结束

第 7 章 应用与测试

至此位置，基于 XML 模型的信息管理系统已经研发完成，接下来我们则需要验证我们所开发的系统如果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，我们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，我们将会同时验证本信息管理系统是否符合研发目的和完成所有计划的需求。

7.1 前提准备条件

首先，本信息管理系统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。

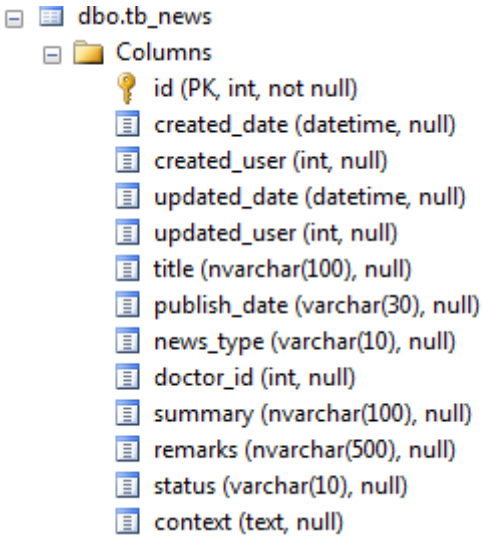
最后，配置 android 平台开发环境，并且引入 Android\helpfooter.jar 包。

至此，我们可以开始进行新闻客户端的开发了。

7.2 信息管理系统端的研发

如上设计和实现，我们只需要利用 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，首先我们进入\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，然后我们新增一个 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，我们在此添加标题，导读，内容，发布日期，状态等。并且添加新闻，删除功能。保存后，我们可以在 CMS\Model 文件夹下面多了一个 news.xml 文件，内容省略。

接下来我们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。



然后我们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
<?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，我们添加一个特殊的业务逻辑，设置发布日期必须为当天。我们继承 XmlModel 对象，然后命名为 NewsXmlModel 对象，重载其 Save 方法。代码如下：

```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);
    }

}

?>
```

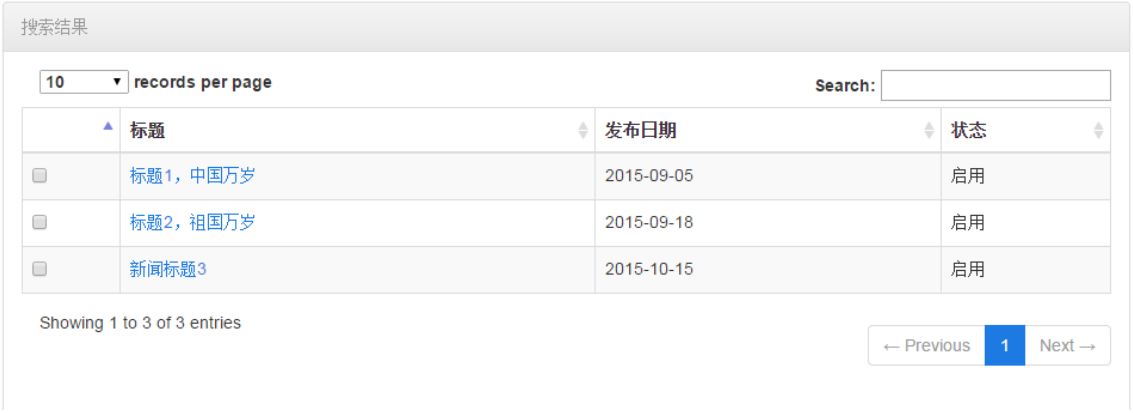
接下来，我们就可以验证是否已经完成新闻模块的管理的开发了。

首先我们使用浏览器进入 CMS\index.php，使用默认登录名登录系统。

然后见到主界面，是关于此系统的介绍。我们找到新闻管理的菜单，点击进入：

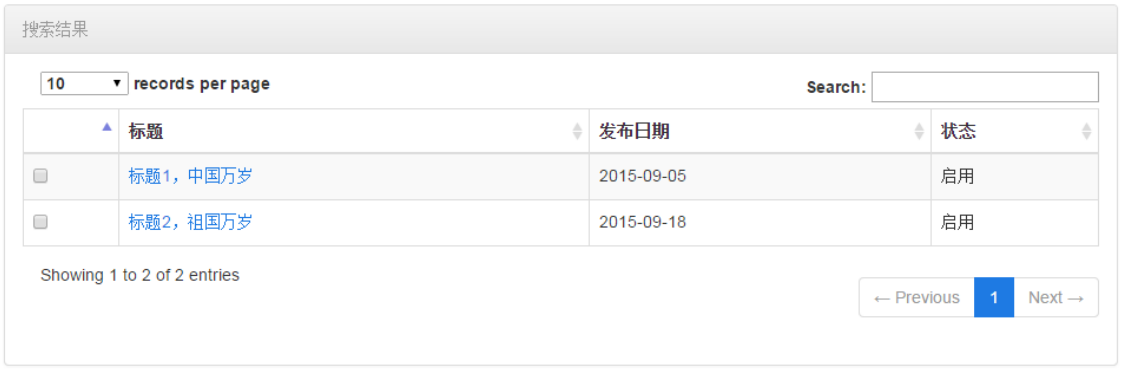
首先进入新闻列表，界面如下，发现左上角新增和删除按钮的功能已经添加：

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：



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我们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。



至此，我们已经验证，信息管理系统已经可以对新闻业务模块内容进行增删查改的实现，而我们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，我们通过 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，我们并没有书写任何代码就完成一个业务模块的管理。

7.3 远程数据数据访问端的研发

首先，我们依然是使用 XML 模型管理系统，然后这次我们选择生成访问接口，那么，页面上自动就会在添加一个 news.php 到\API\目录下，比起在信息管理系统自动生成的代码,我们只需要修改模型为 API 模式即可，代码如下：

```php
<?php
require '../include/common.inc.php';
include ROOT.'/include/init.inc.php';
$action=$_REQUEST["action"];
$model=new XmlModel("news","news.php");
//Set API Model
$model->SetToAPIMode();
$smarty->assign("MyModule","content");
$model->DefaultShow($smarty,$dbmgr,$action,"news",$_REQUEST);
?>
```

至此，我们已经可以基本上完成数据访问接口端的开发，当我们调用\API\news.php 页面时，我们获得一个 XML 格式的网页结果返回：

```
▼<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>
```

当我们需要获取某一条数据具体的所有数据时，我们调用的 url 为 API\news.php?action=get&id=1，然后获得返回结果如下：

```
▼<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>
```

在此基础上，我们添加一个自定义需求，当用户在第三方客户端访问页面后，我们需要 news 表中的访问数量字段。于是我们基础修改 NewsXmlModel 代码如下：

```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);
    }

}

?>
```

当客户端调用 url API\news.php?action=read&id=1 之后，我们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，我们已经可以看出，我们并不需要为数据访问接口端书写太多的代码，我们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。

7.4 安卓平台新闻移动客户端的研发

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，我们为了减少客户存储过多的数据到本地数据库中，我们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，我们需要首先继承 AbstractObj 类，然后添加访问器和实现其定义的抽象方法。代码如下：

```
public class NewsObj extends  AbstractObj {

    String title;
    String publish_date;
    String summary;
    String status;
    String content;


    @Override
    public void parseXmlDataTable(HashMap<String, String> lstRowValue) {
        this.id=Integer.parseInt(lstRowValue.get("id"));
        this.title=lstRowValue.get("title");
        this.publish_date=lstRowValue.get("publish_date");
        this.news_type=lstRowValue.get("news_type");
        this.doctor_id=Integer.parseInt(lstRowValue.get("doctor_id"));
        this.summary=lstRowValue.get("summary");
        this.thumbnail=lstRowValue.get("thumbnail");
        this.status=lstRowValue.get("status");
        this.upvote=Integer.parseInt(lstRowValue.get("upvote"));
        this.content=lstRowValue.get("content");
        this.category=lstRowValue.get("category");
        this.photo=lstRowValue.get("photo");
    }


    @Override
    public void parseCursor(Cursor cursor) {
        setId(cursor.getInt(cursor.getColumnIndex("id")));
        this.title=cursor.getString(cursor.getColumnIndex("title"));
        this.publish_date=cursor.getString(cursor.getColumnIndex("publish_date"));
        this.news_type=cursor.getString(cursor.getColumnIndex("news_type"));
        this.doctor_id=cursor.getInt(cursor.getColumnIndex("doctor_id"));
        this.summary=cursor.getString(cursor.getColumnIndex("summary"));
        this.thumbnail=cursor.getString(cursor.getColumnIndex("thumbnail"));
        this.status=cursor.getString(cursor.getColumnIndex("status"));
        this.upvote=cursor.getInt(cursor.getColumnIndex("upvote"));
        try {
            this.content = cursor.getString(cursor.getColumnIndex("content"));
        }catch (Exception ex){

        }
        this.category=cursor.getString(cursor.getColumnIndex("category"));
```

然后我们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。

```
public class NewsDao extends AbstractDao {

    public NewsDao(Context ctx) {
        super(ctx,"tb_news");
        // TODO Auto-generated constructor stub
    }
}
```

首先实现表构建方法，当客户端代码第一次加载 NewsDao 时，我们回去判断是否新增表和构建相关的字段：

```
@Override
void gotoCreateTableSql() {
    util.open();
    StringBuffer sql = new StringBuffer();
    sql.append("create table IF NOT EXISTS  tb_news " +
            "(id int,title varchar ,publish_date varchar,news_type varchar," +
            "category varchar,doctor_id int,summary varchar," +
            "thumbnail varchar,photo varchar,status varchar,upvote int )");
    util.execSQL(sql.toString(), new Object[]{});

}
```

然后是相关的更新和插入处理语句的代码：

```
@Override
public void insertObj(AbstractObj abobj) {
    // TODO Auto-generated method stub
    util.open();

    NewsObj obj=(NewsObj)abobj;

    StringBuffer sql = new StringBuffer();
    sql.append("insert into tb_news (id,title  ,publish_date,news_type,category,doctor_id,summary,thumbnail,photo,status,upvote ) " +
            "values (?,?,?,?,?,?,?,?,?,?,?)");
    Object[] bindArgs = {obj.getId(),obj.getTitle(),obj.getPublish_date(),obj.getNews_type(),
            obj.getCategory(),obj.getDoctor_id(),obj.getSummary(),obj.getThumbnail(),obj.getPhoto(),obj.getStatus(),obj.getUpvote()};
    util.execSQL(sql.toString(), bindArgs);

    util.close();
}

@Override
public void updateObj(AbstractObj abobj) {
    // TODO Auto-generated method stub
    util.open();

    NewsObj obj=(NewsObj)abobj;

    StringBuffer sql = new StringBuffer();
    sql.append("update tb_news set title=?  ,publish_date=?,news_type=?,category=?,doctor_id=?," +
            "summary=?,thumbnail=?,photo=?,status=?,upvote=?" +
            " where id=? ");
    Object[] bindArgs = {obj.getTitle(),obj.getPublish_date(),obj.getNews_type(),obj.getCategory(),
            obj.getDoctor_id(),obj.getSummary(),obj.getThumbnail(),obj.getPhoto(),obj.getStatus(),obj.getUpvote(),obj.getId()};
    util.execSQL(sql.toString(),bindArgs);

    util.close();
}
```

有了这两个对象，我们的新闻数据就可以保存在内存中和本地数据库中。

然后我们在定义一个继承 WebLoader 的 NewsListWebLoader 对象，通过这个代码我们就可以自动去获取 URL 对象并将其保存到本地数据库中：

```
public class NewsListLoader extends WebXmlLoader {

    public NewsLoader(Context ctx) {
        super(ctx);
    }

    public String getCallUrl() {
        // TODO Auto-generated method stub
        return "API\news.php";
    }

}
```

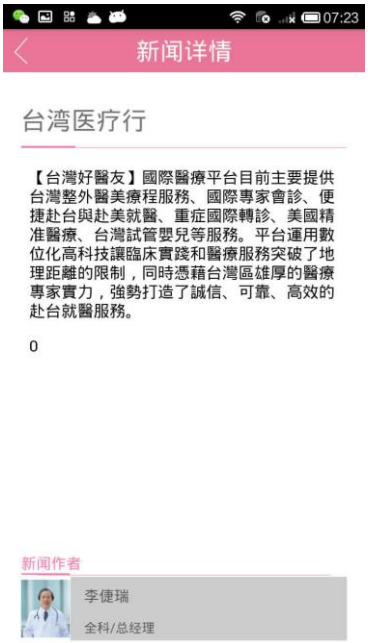
我们在 NewsListActivity 中，可以如此调用此 NewsListLoader 对象：

```
private void InitData() {
    NewsListLoader loader=new NewsListLoader(this);
    loader.setCallBack(this);
    loader.start();
}

public void WebLoaderCallBack(ArrayList<AbstractObj> lstObjs){
    ShowNewsList(lstObjs);
}
```

最终，我们可以在安卓客户端中看到如下界面：

点击获取新闻详情的代码类似，代码省略，我们可以得到的客户端界面截图如下：



至此，我们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，我们除了对安卓界面的代码进行构建以外，我们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。

7.5 总结

从以上的安卓平台新闻系统的研发过程可见，我们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，我们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，然后我们利用 XML 模型管理系统来服务生成信息管理系统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研发，中间不需要过多的关注系统业务逻辑。

第 8 章 总结与期望

本课题研发的基于 XML 模型的信息管理系统存在开发快，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的业务数据管理系统的研发，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的设计来完成业务数据系统的研发，可以立刻把自己的想法实现并且快速出现在市场中抢先一步。

但是，本课题依然有相当多的地方需要改善，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。因为本课题研发的系统主要在于追求性能的快和开发速度的快和开放性上面，则减少对于用户请求的校验，对于数据系统的安全性来说是一个问题。

在未来，此系统将会加入私有云的概念，会把本系统成立为一个平台，并且根据用户的申请自动创建一个服务器和相关的系统，用户可以直接在私有云中添加自己的业务驱动模型则可以在相关开发的移动终端产品中访问相关的私有云自动化移动业务管理系统和其数据访问接口，成为更加高效，免于部署的自动化业务管理系统。

致　　谢

在本次论文设计过程中，朱映映老师对该论文从选题，构思到最后定稿的各个环节给予细心指引与教导,使我得以最终完成毕业论文设计。在学习中,老师严谨的治学态度、丰富渊博的知识、敏锐的学术思维、精益求精的工作态度以及侮人不倦的师者风范是我终生学习的楷模，导师们的高深精湛的造诣与严谨求实的治学精神，将永远激励着我。这三年中还得到众多老师的关心支持和帮助。在此，谨向老师们致以衷心的感谢和崇高的敬意！同时感谢许多使用本系统的开发者提供各种宝贵意见和在开源库中为本系统添加和完善各种功能。

最后，我要向百忙之中抽时间对本文进行审阅，评议和参与本人论文答辩的各位老师表示感谢。