

农业墒情预警信息实时推送系统的设计与实现

郑九锋,姚凯学

(贵州大学 计算机科学与技术学院,贵州 贵阳 550025)

摘 要:农业墒情预警信息实时推送系统将农业墒情监测站传过来的墒情数据,包括土壤湿度、土壤温度、大气温湿度等实时数据与预警规则相绑定,并且每分钟将解析一次预警规则。如果满足预警条件,系统就会自动将预警信息及时发送到安装有与系统配套的 App 的 Android 手机终端上,同时也支持在 Web 页面手动编辑信息进行推送,用户点击手机通知栏即可查看消息详情,方便用户及时采取相应的应对措施。文中介绍了一种基于 XMPP 协议的 AndroidPn 开源框架的实现模型,通过详细的系统设计,在此基础上运用 SpringMVC 框架,采用 Spring 定时任务技术、MySQL 数据库技术、QLEExpress 规则引擎技术、Android 开发技术等实现了该系统,包括 Web 端与客户端。测试结果表明,该系统满足功能要求并且运行稳定。

关键词:墒情预警;实时推送;安卓;可扩展通信协议;Web;AndroidPn

中图分类号:TP31 **文献标识码:**A **文章编号:**1673-629X(2018)06-0137-05
doi:10.3969/j.issn.1673-629X.2018.06.031

Design and Realization of Real-time Push System for Agricultural Moisture Early Warning Information

ZHENG Jiu-feng, YAO Kai-xue

(School of Computer Science & Technology, Guizhou University, Guiyang 550025, China)

Abstract:The system of agricultural moisture alert information real-time push bounds the soil moisture data, which includes soil moisture, soil temperature, atmospheric temperature, humidity and so on, with warning rules parsed every minute. If the warning conditions is met, the system will automatically send early warning information to the Android phone terminal installed the App. And it also supports the Web page to manually edit the information to push. The users can click on the phone notification bar to view the message details, which is convenient for them to take corresponding measures in time. We introduce an implementation model of AndroidPn open source framework based on extensible messageing and presence protocol (XMPP). With the detailed system design, it is implemented by Spring-MVC framework, Spring timing task technology, MySQL database technology, QLEExpress rule engine technology, Android development technology and so on, including the Web side and the Android client. The test shows that the system meets the functional requirements and runs stably.

Key words:moisture situation warning; real-time push; Android; XMPP; Web; AndroidPn

0 引 言

农业墒情监测站集成多种气象传感器,采集了包括土壤湿度、土壤温度、光照强度、光合有效辐射、风向、风速、雨量、雨速、气温、空气相对湿度、气压、海拔、土壤 PH 值、PM2.5 等实时数据^[1],并将这些数据经过下位机以及无线通信技术实时传输至后台服务端,后台将这些数据通过制定的协议解析,将合法的数据封装后存入数据库相应的表中。

运用 QLEExpress 规则引擎技术设定预警规则,预

警规则与实时数据绑定,规则内容包括大气温度报警、大气湿度报警、墒情站离线报警等。推送系统将使用 Spring 定时任务每分钟解析规则一次,满足预警条件后就触发相应的推送方法,将预警信息及时地发送到用户的手机终端上,同时推送系统也支持在 Web 页面手动编辑信息进行推送。用户点击手机通知栏就能查看消息详情,在用户获取预警信息后,就可以采取相应的应对措施,避免因环境气候因素导致农作物减产,带来不必要的经济损失,因此该系统在农业生产应用中

意义重大。

1 关键技术

1.1 基于 XMPP 协议的长连接推送技术

XMPP(extensible messaging and presence protocol,可扩展通讯和表示协议)是一种基于可扩展标记语言(XML)的近端串流式即时通讯协议^[2-3]。它是开源的,利用它可以实现简单的推送功能,开发者可以修改其源代码以适应自己的应用程序。

AndroidPn 就是一个基于 XMPP 协议实现的开源消息推送服务,包含了完整的客户端和服务端,其服务端是在另外一个开源项目 openfire 的基础上修改实现的,其客户端需要用到开源 XMPP 协议包 asmack^[4],这个包同样是基于 openfire 下的另外一个开源项目 smack。基于 XMPP 实现推送的优点就是简单,开发者还可以根据自己的实际使用需求来扩展 XMPP 协议,实现消息的定制功能。

基于以上分析,系统采用基于 XMPP 协议的 AndroidPn 框架进行设计实现。通过集成开源的 JAR 包,修改开源项目,缩短开发成本,提高系统应用的稳定性和成熟性^[5]。

1.2 QLEXPRESS 规则引擎技术

QLEXPRESS 是一个开源的 JAVA 规则引擎,它一般作为一个嵌入式规则引擎在业务系统中使用^[6-7]。该规则引擎技术不仅支持标准的 JAVA 语法,而且还支持自定义操作符号、操作符号重载、函数定义、宏定义、数据延迟加载等,所以使业务规则定义简便并且非常灵活。

该规则的原理主要是定义规则、解析规则、执行规则几个步骤,编译的过程类似 java class 文件的编译过程,会经过词法分解、词法分析、语法分析、规则执行等几个步骤。系统按照 QLEXPRESS 语法定义预警规则, Spring 定时任务会每隔一分钟解析一次预警规则,解析预警规则仍是使用 QLEXPRESS。

1.3 Spring 定时任务

Spring task 是 Spring3.0 以后自身提供的一种定时任务实现工具,该工具使用起来非常简单,可以将它看作是一个轻量级的 Quartz^[8]。使用该工具时除了导入 Spring 相关的包外不需要再导入其他额外的包,而且支持注解和配置文件两种形式,在 Spring 的配置文件中配置定时器开关<task:annotation-driven />,在定时器的实现代码中配置@Scheduled 注解,@Scheduled(cron="0 0/15 9-23 * * ?")表示每分钟执行一次。该系统使用 Spring task 定义预警规则解析任务,每分钟解析一次预警规则,若满足预警条件则将预警信息推送到用户手机端。

2 系统设计

2.1 服务端设计

2.1.1 服务端架构

服务器端采用 B/S 架构,主要是在一个基于 XMPP 协议的开源工程 openfire 和高性能并发框架 MINA 基础上修改实现,运用 SpringMVC 和 Hibernate 框架^[9-10],结合性能高效的 MySQL 数据库,服务器端软件架构如图 1 所示。服务器采用了分层的架构设计,共分为四层,分别为前端表现层、业务逻辑层、数据访问层、数据库管理层。

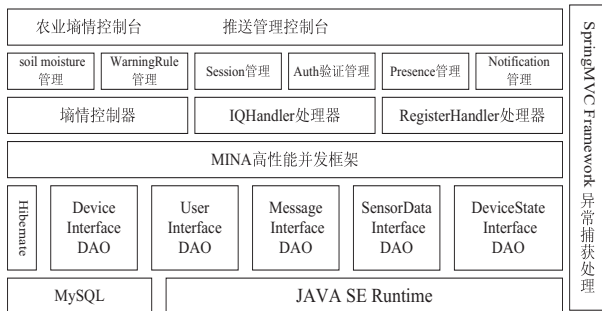


图 1 服务端架构

(1) 前端表现层。

前端表现层包括两个部分,一部分是农业墒情控制台,主要是以图表形式展示墒情监测站发送过来的墒情数据,预警规则管理界面主要用来进行预警规则的添加、修改、删除、查询等;另一部分是推送管理控制台,主要是消息推送管理界面。在表现层,系统前端运用了 jQueryEasyUI 框架, JSP2.0、CSS3、HTML 以及 Ajax 等技术。

(2) 业务逻辑层。

业务逻辑层主要是将表现层所涉及到的前端服务进行相关的逻辑实现。该系统业务逻辑层主要基于 SpringMVC 框架开发, SoilmoistureManager 负责墒情数据的管理, WarningRuleManager 负责预警规则的管理;客户端与服务器端之间的通信使用了 AndroidPn,它是一个实现了 XMPP 协议的开源项目,其中 SessionManager 负责管理客户端 App 与服务器端之间的会话, AuthManager 负责 App 用户认证管理, PresenceManager 负责管理 App 用户的登录状态, NotificationManager 负责实现服务器向客户端 App 推送消息。MINA 框架将网络通信与应用程序隔离开来,开发者只需关心传输的数据以及业务逻辑即可,做到了高并发 Socket 访问的有序管理。

(3) 数据访问层。

数据访问层对数据库进行一些相关业务的操作处理,例如数据的添加、删除、修改、查询,将业务的一系列数据操作提交到数据库中。该系统后台服务器采用 Hibernate 与 SpringMVC 框架开发, Spring 框架本身提

供了对 DAO 层的支持,可以提高开发者的效率,保证数据库操作的准确性和安全性;Hibernate 是一种 ORM 框架,用于将数据持久化。

(4)数据库层。

该项目采用开源的关系型 MySQL 数据库系统。

MySQL 用于记录设备信息、用户信息、待推送或已推送的消息内容、预警规则、墒情信息等数据。

农业墒情预警信息实时推送系统由墒情管理、预警规则管理、用户状态管理、会话管理、通知管理、消息记录管理 6 个模块构成。功能结构如图 2 所示。

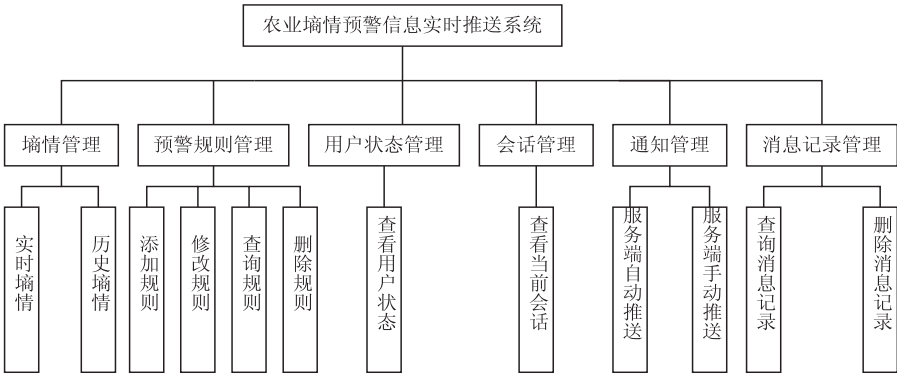


图 2 服务端功能结构

2.1.2 数据库设计

系统采用的是 MySQL 数据库,定义了用户信息表 (tb_user)、客户表 (tb_appuser)、信息表 (tb_message)、预警规则表 (tb_rule)、离线信息表 (tb_offmessage)、墒情数据表 (tb_moisture)。为了使用这些表,在 Spring 框架中集成 Hibernate 框架,在 Spring 配置文件中配置数据源、SessionFactory,配置实体类等实现表的建立以及数据库的连接。

2.2 客户端设计

Android 客户端使用基于 JAVA 的开源 XMPP 协议包 asmack,该包是基于 openfire 下的另外一个开源项目 smack,在开发客户端时,直接将 asmack.jar 包导入 Android studio 下的工程项目中即可使用。App 利用 asmack 中提供的 XMPPConnection 类与服务器端建立起持久连接,并通过该连接进行用户注册和登录认证,该连接也用于服务器端发送的消息通知^[11]。客户端程序中 ServiceManager 负责管理消息服务以及加载相关的配置,ConnectivityReceiver 负责处理网络状态的广播,NotificationReceiver 负责处理服务端发送的推送消息,NotificationService 负责在后台服务响应服务端的消息,PersistentConnectionListener 负责监控连接关闭以及重连事件的监听,PhoneStateChangeListener 负责监听手机状态的事件,ReconnectionThread 负责重连线程类,Notifier 负责客户端发送通知,Notification-IQ 负责处理消息的数据包^[12-13]。

3 系统实现

3.1 墒情管理模块实现

墒情监测站将各传感器采集到的实时数据通过下位机无线传输至推送系统服务端,服务端通过制定的

通信协议将数据解析封装后存入数据库。为了更直观地显示这些数据,Web 前端视图层将以图表形式显示这些墒情数据,用户可以查看实时墒情数据,并且可以根据条件查询历史墒情数据。

3.2 预警规则管理模块实现

预警规则管理模块主要负责预警规则的添加、查看、修改、删除。规则内容中有的是回调系统方法,例如“获取传感器值 (realdata,“大气温度”)、判断是否再次触发 (rule,生产基地 id,180)”等”。这些规则运用了 QLExpress 规则引擎,它是阿里内部的一个开源的 java 规则引擎,主要是定义规则、解析规则、执行规则几个步骤,编译的过程类似 java class 文件的编译过程,首先进行词法分解、词法分析、语法分析、规则执行等步骤。

例如,规则内容可以定义如下:

```
double 温度值=获取传感器值 (realdata,“大气温度”);
String 生产基地 id=获取生产基地 id (realdata);
String 生产基地名称=获取生产基地名称 (realdata);
如果 (温度值大于 30)
{
    boolean 再次触发=判断是否再次触发 (rule,生产基地 id,180);
    if (再次触发 ==true) {String 触发规则 id=触发规则 (rule,生产基地 id,“墒情警报”);
        发 APP 消息 (触发规则 id,生产基地 id,“接收墒情警报”,“温度超限报警”,生产基地名称+realdata.getGathertime().toString()+“传感器温度为”+温度值.toString()+“超过上限了”);
    }
}
```

3.3 客户状态管理模块实现

客户状态管理模块以列表形式展现已注册客户端的在线/离线、用户名、姓名、邮件、创建日期等信息。如果客户处于“在线”状态,则“在线/离线”列的图片会变成蓝色;如果处于“离线”状态,则图片会变成灰

色。用户名是 XMPP 的地址 JabberID 中的 node identifier,它由 32 位字母和数字随机生成,用来唯一标识通信中的客户端^[14]。

3.4 会话管理模块实现

会话管理模块以列表形式展现当前所有客户端与服务端建立的会话连接,“在线/离线”列的图片全部显示蓝色,表示该列表的用户全部在线且可以收到消息,系统可以对该列表的用户群发消息,也可以选择某个用户单独发消息,会话列表记录了用户名、状态、客户端 IP 等信息。

3.5 通知管理模块实现

通知管理模块分为服务器自动推送和 Web 端手动推送,分别介绍如下:

3.5.1 服务器自动推送

使用 Spring 定时器,定义实时墒情数据规则处理任务类,该类的功能是处理实时数据报警规则,主要是运用 qlExpress 对规则表达式进行处理,解析规则后将实时数据与规则进行绑定,最后执行规则。执行规则过程会执行规则里面的推送方法,即规则中的“发送 APP 消息”方法,该方法实际上是调用 Notification-Manager 中的 sendNotificationToUser 方法。

该任务每分钟被触发并执行一次,对实时数据报警规则进行处理,如果实时墒情数据符合规则定义要求,系统则会发送预警信息到用户手机端上。

3.5.2 Web 端手动推送

Web 端手动推送是在推送页面编辑推送信息,然后提交信息即可完成推送,用户点击手机通知栏即可查看信息详情。在该页面选择推送的目标用户,可以选择所有用户进行推送,也可以指定用户进行推送,还可以只推送给当前在线用户。当选择对所有用户或者指定用户进行推送时,离线用户不会立即收到消息,服务端会把离线消息存入数据库的离线信息表中,当用户再次在线时,客户端会自动请求服务器,服务器会将离线信息表中最新的信息发送到用户手机端。标题和内容用户可以自己定义,URI 则是用户的地址,不是必填项,点击“提交”后请求执行的方法是 Notification-Manager 中的 sendBroadcast()、sendNotificationToUser()或者 sendNotificationToUserOnline(),当选择向所有用户发送信息时会执行 sendBroadcast()方法,当指定用户时会执行 sendNotificationToUser()方法,当选择推送给在线用户时会执行 sendNotificationToUserOnline()方法。

3.6 消息记录管理模块实现

消息记录管理模块是对所有的推送信息进行记录,每发送一条信息则会将该信息的相关信息存入数据库相应的表中,在该页面可以按消息类型、状态、开

始时间、结束时间等条件查询消息记录,其中消息类型分为服务器自动推送和 Web 手动推送。点击“查询”按钮,页面会以列表的形式显示符合该条件下的消息记录,用户在该页面选中某条信息记录双击可以查看消息详情,同时,该页面也支持消息记录的删除。

3.7 客户端的实现

基于 Android 平台的客户端运用 Android Studio 进行开发,开发过程中在项目目录下导入基于 XMPP 协议的 asmack.jar 包;客户端主要进行连接信息的管理,例如 XMPP 协议的端口、IP、用户登录等信息的管理,以及定时发送连接请求以保持连接处于有效状态。服务器端和客户端的通信是一个 session(会话)过程。会话开始时客户端首先会通过指定的服务器的端口号,把信息发送到服务器端,而服务器发送消息是通过 <stream>根节点的方式传递,在数据传输过程中,信息处理采用 XML 节的方式传递,其中 IQ(Info/Query)元素用来发送和获取实体之间的信息,<Message/>(消息)元素用来存储消息内容,<Presence/>(在线状态探测)元素用来传递用户的状态,当服务器和客户端关闭时会发送它的结束标记</stream>。客户端只需要通过 XMPP 协议接收消息,而其他所有的操作都由服务器完成,这样的处理机制大大减轻了客户端的负担^[15]。

4 运行结果

文中所研究的推送系统已经在试点基地运行,目前运行情况良好;基地管理员用户打开浏览器进入系统首页后要求输入用户名和密码,用户名和密码匹配成功后会进入系统功能页面。

墒情管理模块以图表形式直观地展示农业墒情监测站传过来的各种传感器的实时数据,在该页面能够查询历史数据、分析数据等。

预警规则管理模块可以进行预警规则的添加、删除、修改、查看详情等操作,但该页面需要专业人员去维护,否则系统可能会出现异常,因此对该页面进行了权限控制。

客户状态管理模块可以看到所有的已注册用户,在线的用户头像显示蓝色,离线的显示灰色。新用户用手机安装客户端并注册后,发现该页面会自动添加一行新客户信息,头像显示蓝色,当该用户退出客户端后,头像立即变为灰色。

会话管理模块完整地显示了当前可与服务端进行会话的客户信息,一旦用户离线,该用户的信息会在该页面消失,重新上线后又会显示。

通知管理模块可以进行服务器自动推送和 Web 端手动推送,经测试,服务端能够在满足条件的情况下

将信息自动推送给用户,手动推送则可以按照管理员用户的需求有针对性的推送。对于离线用户,经过测试,当对离线用户发送信息时,离线用户不会立即收到信息,而是当用户再次上线时会收到最新的推送信息。

消息记录管理模块记录了每一次的消息推送信息,管理员可以在该页面查看或删除消息记录。

消息推送成功后,用户手机端会接收到相关信息,手机通知栏显示信息如图 3 所示,点击通知栏显示信息详情如图 4 所示。



图 3 通知栏显示信息



图 4 显示信息详情

5 结束语

农业墒情预警信息实时推送系统包含完整的服务端与客户端,实现了墒情信息的服务端自动推送和

Web 端手动推送,能够及时地将墒情信息推送至用户的手机客户端上,用户点击手机通知栏即可查看消息详情。该系统提供准确及时的墒情预警信息给用户,方便用户及时了解农作物生长环境的突变情况,及时采取应对措施,避免因环境变化而造成农作物的损失,因而在农业生产上具有很高的推广价值,同时具有广阔的应用前景。

参考文献:

[1] 张 宇,张厚武,丁振磊,等. 农业小气候数据监测站的设计与实现[J]. 计算机工程与设计,2016,37(8):2072-2076.

[2] 杨 斌. XMPP 协议分析与应用探讨[J]. 微型机与应用,2005,24(8):32-34.

[3] LI Chengzhe,HSIEH M Y,HSU K H. An extended SOA for distributing workload to service providers using XMPP[J]. Frontiers in Artificial Intelligence & Applications,2015,274:1743-1752.

[4] WAGENER J, SPJUTH O, WILLIGHAGEN E L, et al. XMPP for cloud computing in bioinformatics supporting discovery and invocation of asynchronous web services[J]. BMC Bioinformatics,2009,10:279.

[5] 倪红军. 基于 Android 平台的消息推送研究与实现[J]. 实验室研究与探索,2014,33(5):96-100.

[6] 陶晓俊,朱 敏. 基于规则引擎的企业服务开发模式[J]. 计算机技术与发展,2008,18(2):115-118.

[7] 缴明洋,谭庆平. Java 规则引擎技术研究[J]. 计算机与信息技术,2006(3):41-43.

[8] 丁振凡,李馨梅. Spring 的任务定时调度方法的研究比较[J]. 智能计算机与应用,2012,2(4):55-56.

[9] 张 宇,王映辉,张翔南. 基于 Spring 的 MVC 框架设计与实现[J]. 计算机工程,2010,36(4):59-62.

[10] AHUJA S,YANG J L. Performance evaluation of java web services;a developer's perspective[J]. Communications and Network,2010,2(3):200-206.

[11] 汪海占,邸 萌,黄祥林. 基于 XMPP 协议的 Android 消息推送设计与实现[J]. 科技广场,2015(2):40-46.

[12] 黄明恩. 基于 Android 平台的云推送服务的设计与实现[D]. 北京:北京交通大学,2015.

[13] 张克建. 基于 JavaEE 与 Android 的消息推送系统的研究与实现[D]. 北京:华北电力大学,2015.

[14] JIN J H,LEE H C,LEE M J. Supporting collaborative workspaces over XMPP[M]. Berlin:Springer,2014.

[15] 殷 昊. 基于 Android 平台的消息推送能力的研究与实现[D]. 北京:北京邮电大学,2013.