软件工程课程设计

《4D GIS标记系统项目》

软件设计说明书

设计组学生:

姓名 姚翛潇（组长） 学号1120161821

姓名 朱婧婧 学号1120161826

姓名 谢蜜雪 学号1120161761

姓名 杨冰琪 学号1120161762

姓名 杨 俊 学号1120161820

姓名 霍萱甫 学号1120162099

姓名 刘楠彬 学号1120162132

指导教师: 赵刚

北京理工大学计算机学院

2019年10月

4D GIS标记系统

4D GISMS (4D GIS Marking System)

《软件设计说明书》

目录

[目录 2](#_Toc22591975)

[1、引言 4](#_Toc22591976)

[1.1编写目的 4](#_Toc22591977)

[1.2项目背景 4](#_Toc22591978)

[1.3设计目标 4](#_Toc22591979)

[1.4运行环境 5](#_Toc22591980)

[2、技术路线 6](#_Toc22591981)

[3、数据结构设计 6](#_Toc22591982)

[3.1数据关系建模 6](#_Toc22591983)

[3.2 数据结构建模 6](#_Toc22591984)

[3.2.1 数据库表设计 6](#_Toc22591985)

[3.3 数据库建模 7](#_Toc22591986)

[4、功能设计 7](#_Toc22591987)

[4.1 数据流图 7](#_Toc22591988)

[4.1.1 第0层 7](#_Toc22591989)

[4.1.2 第1层 7](#_Toc22591990)

[4.1.2 第2层 7](#_Toc22591991)

[4.2 功能实现 7](#_Toc22591992)

[4.2.1登录注册 8](#_Toc22591993)

[4.2.2 视图切换 8](#_Toc22591994)

[4.2.3 即时查询 8](#_Toc22591995)

[4.2.4 离线数据同步 8](#_Toc22591996)

[4.2.5 标记信息分享 11](#_Toc22591997)

[4.2.6 信息标记 13](#_Toc22591998)

[4.2.7 图层查看 17](#_Toc22591999)

[4.2.8 信息发布 20](#_Toc22592000)

[5、界面设计 20](#_Toc22592001)

[5.1 登录/注册和密码找回界面 20](#_Toc22592002)

[5.2主界面默认展示 22](#_Toc22592003)

[5.3 即时查询界面 22](#_Toc22592004)

[5.4 数据上传界面 24](#_Toc22592005)

[5.5 分享界面 25](#_Toc22592006)

[5.6 添加标记/路线界面 26](#_Toc22592007)

[5.7 数据库管理界面 28](#_Toc22592008)

[5.8 个人信息修改界面 31](#_Toc22592009)

[6、程序设计 32](#_Toc22592010)

[6.1登录注册 32](#_Toc22592011)

[6.2 视图切换 32](#_Toc22592012)

[6.3 即时查询 32](#_Toc22592013)

[6.4 离线数据同步 32](#_Toc22592014)

[6.5 标记信息分享 32](#_Toc22592015)

[6.6 图层选择 32](#_Toc22592016)

[6.7 信息标记 32](#_Toc22592017)

[6.8 信息发布 32](#_Toc22592018)

[7、重要技术难点及解决方案 32](#_Toc22592019)

# 1、引言

## 1.1编写目的

该文档负责说明软件的详细设计，为编码和测试工作提供依据。根据软件需求分析的结 果，对项目的各个子系统和功能进行模块划分，并对其界面、模块、系统结构间的关系进行了设计与描述。这个阶段的任务解决了软件系统总体结构设计的问题，包括整个软件系统的结构、模块划分、模块功能和模块间的联系等。还要解决 如何实现各个模块的内部功能，即模块设计。模块的设计会为各子系统设计算法，是整个软件开发过程中所有后续工作的指导。

本文档面向多种读者对象：

（1）软件编写人员：了解数据结构，数据形式以及需实现的功能，据此进行软件编写。

（2）软件测试人员：根据此文档编写测试用例，对软件产品进行功能性测试。

## 1.2项目背景

项目名称：4D GIS标记系统

近来，行业普遍认为GIS将朝着普适化、智能化方向发展。在未来，任何人都可以在任何地方，拿着任意终端访问GIS服务，而不局限于特定用户和专业的终端，希望使普通用户也能够通过多媒介进行访问。随着云计算技术、移动终端、物联网等技术的快速发展，这一想法是极有可能实现的。因此，我们希望创造一个GIS环境，把大家的知识和经验用地图的方式来表达，让用户非常方便的获得地图数据，满足他们对于出行、旅游、测绘、科研等方面的需求。

## 1.3设计目标

本系统的设计说明书主要分为四个部分来完成：功能设计、数据结构设计、界面设计及程序伪代码的实现。功能设计部分将完成本系统的功能建模，包括软件的类图、数据流图等；数据结构设计部分讲完成软件的数据建模，其中包括数据的ER图、数据结构定义和数据库表的设计；界面设计部分将完成简要的界面设计和接口模块的设计。后续的软件开发编码人员和测试人员可根据此设计说明书进行编写。

## 1.4运行环境

* **客户端程序：**

客户端程序可以运行在装有 Android 操作系统或IOS的智能手机、平板电脑上，web网页以及个人笔记本电脑上，需要满足的具体要求如下：

Android系统：

|  |  |
| --- | --- |
| 操作系统 | Android 4.1  以上 |
| 处理器 | 高通骁龙  200 以上 |
| 运行RAM | 1G 以上 |
| 存储空间 | 50MB 以上 |
| 其他要求 | 支持 GPS 导航 |

IOS系统：

|  |  |
| --- | --- |
| 操作系统 | IOS 7.0  以上 |
| 处理器 | Apple A7 以上 |
| 运行RAM | 1G 以上 |
| 存储空间 | **50MB** 以上 |
| 其他要求 | 支持 GPS 导航 |

Web及PC端：

|  |  |
| --- | --- |
| 操作系统 | Windows |
| 运行RAM | 4G 以上 |
| 存储空间 | 100MB 以上 |

* **服务器程序：**

服务器程序运行在安装有Windows 操作系统的服务器上，需要满足的具体要求如下：

|  |  |
| --- | --- |
| 操作系统 | Windows 7以上 |
| 数据库 | MySQL |
| 处理器 | Intel |
| RAM | 8GB以上 |
| 数据存储空间 | 512GB以上 |
| 网络 | 支持TCP/IP |

## 2、技术路线

本系统在设计时采用C/S的架构进行设计。产品设计分层按服务器端和客户端之间的不同来进行。其中服务器端使用apache服务器采用微服务器的架构进行设计，数据库使用SQL Server 2012数据库，编程语言使用Java。客户端使用多平台独立开发App的策略，在ios端和mac os端使用swift进行开发，在android端使用kotlin进行开发，windows端使用.Net开发框架进行开发。外部的数据接口我们采用两种策略，当显示的部分不要求相当精细时，使用百度地图的外部接口进行；当内容要求精细时，尤其是要显示地图的3D信息、地理信息等时，使用Arc GIS。

# 3、数据结构设计

## 3.1数据关系建模

(ER图及说明)

## 3.2 数据结构建模

### 3.2.1 数据库表设计

用户表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段说明** | **名称** | **类型** | **长度** | **属性(主键/外键)** |

标记表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段说明** | **名称** | **类型** | **长度** | **属性** |

线路表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段说明** | **名称** | **类型** | **长度** | **属性** |

图层表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段说明** | **名称** | **类型** | **长度** | **属性** |

离线标记表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段说明** | **名称** | **类型** | **长度** | **属性** |

标记分享表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段说明** | **名称** | **类型** | **长度** | **属性** |

线路分享表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段说明** | **名称** | **类型** | **长度** | **属性** |

## 3.3 数据库建模

（建模图及说明）

# 4、功能设计

## 4.1 数据流图

### 4.1.1 第0层

### 4.1.2 第1层

### 4.1.2 第2层

## 4.2 功能实现

（每部分包括详细介绍、流程图、类图（新增）、函数接口）

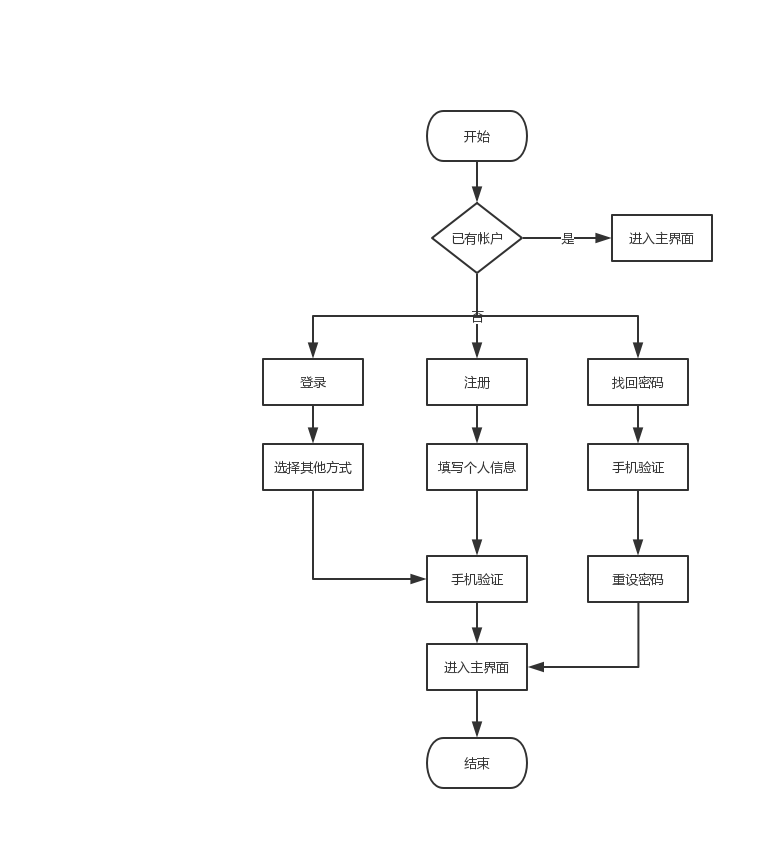
### 4.2.1登录注册

登录注册模块主要包括登录、注册和找回密码三个功能。用户第一次打开软件时，默认界面为登录界面，此时用户可以选择使用其他端口账号登录，如微信、QQ、微博等。当选择以这些账号登录时，会获取权限，并将该账号的头像，昵称等基本资料复制到本软件的账号上，此时用户只需要输入手机号、设置密码、填入发送过去的验证码即可，手机号即为用户账号。若用户进入注册界面，此时用户需要填写部分个人信息，再以手机号和密码进行注册。

当用户再次打开软件时，会直接以上一次登录的账号直接登录，可以退出账号重新以其他账号进行登录。

当用户忘记密码时可以选择找回密码，通过手机号的短信验证后即可重新设置密码。

登录注册的流程图如下图所示：



登陆注册模块共有一个用户信息类：

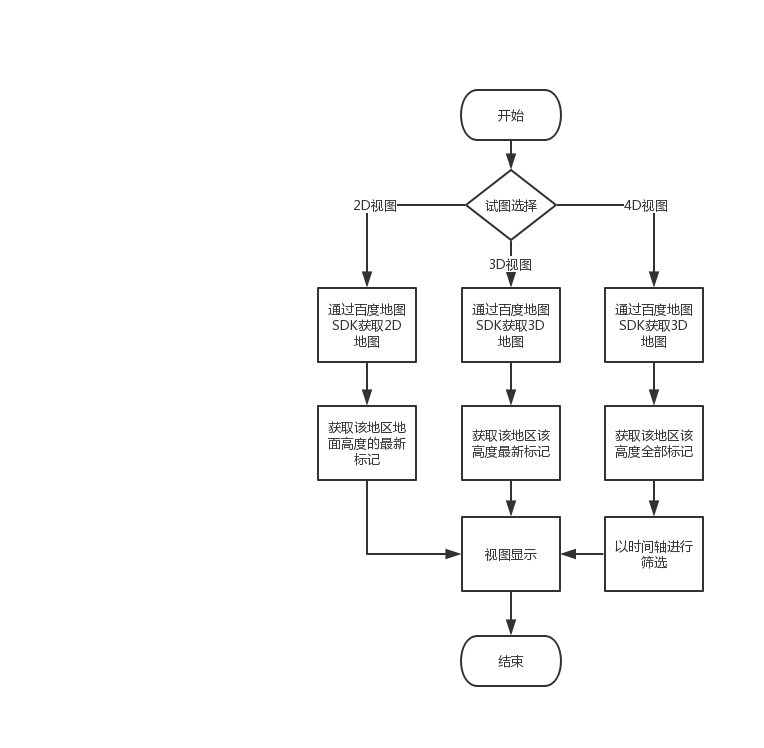


### 4.2.2 视图切换

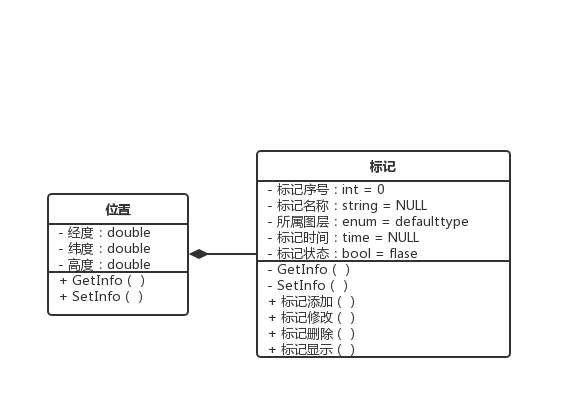
视图切换共包括3个功能，分别是切换2D视图、切换3D视图、切换4D视图。其显示时只有2D和3D地图可显示，4D需要加一个坐标轴来实现。

用户可以通过选择“2D”、“3D”和“4D”标识来切换视图。

视图切换的流程图如下图所示：



视图切换模块共有两个类，分别是标记类、位置类。各类的实现及它们之间的关系如下：



视图切换需要通过百度地图SDK获取地图位置信息，并筛选标记进行显示。该部分对应的接口如下表所示：

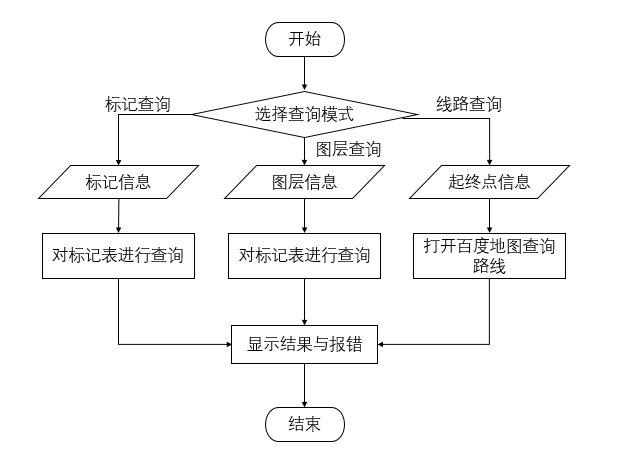
|  |  |  |  |
| --- | --- | --- | --- |
| **接口名称** | **参数** | **返回值** | **说明** |
| get\_baidu\_map\_info() | Null | MapView | 根据当前位置进行选择，返回该部分地图 |
| select\_tag() | enum pos | enum tag | 根据当前位置的坐标进行选择，返回tag列表 |

### 4.2.3 即时查询

即时查询可根据三种输入参数进行查询，输入标记信息及图层信息将返回从数据库中查询的标记，输入起终点为位置信息将在客户端以高德地图查询地铁路线的格式，显示线路。

当用户想对所需信息进行查询时，首先进入查询页面，再选择三种查询模式中的一种，输入查询信息进行搜索及结果显示。

图片 1 即时查询流程图



表格 1 即时查询接口

|  |  |  |  |
| --- | --- | --- | --- |
| **接口名称** | **参数** | **返回值** | **说明** |
| query\_tag\_send() | string query\_tag\_info | bool | 输入标记的信息（经度，维度，高度，时间）或标记的名字。  对标记表执行查询操作，将查询到的标记信息进行格式转换以及维度扩充 ,在客户端地图上展示从数据库中查询到的标记。返回是否查询成功。 |
| query\_layer\_send() | string query\_layer\_info | bool | 输入图层名称。对标记表执行查询操作，将查询到的标记信息进行格式转换以及维度扩充，在地图上显示。返回是否查询成功。 |
| Query\_liner\_send() | string query\_begin\_place,query\_end\_place | bool | 输入起点及终点信息。对线路表执行查询操作，并按照路程进行升序排序，将查询到的线路进行格式转换，在客户端显示。返回是否查询成功。 |

### 4.2.4 离线数据同步

离线数据同步共包括4个功能，分别是离线标记、标记逐条上传、在线集中上传和本地数据同步。用户在添加标记后，可以选择“提交”或“提交并上传”功能，若选择“提交”，则仅将标记保存在本地数据库，该 标记为离线标记，显示“未上传”按钮标识。

当用户想上传某条标记时，点击“未上传”按钮，进行上传，上传时会根据网络状态判断上传是否成功并返回给用户。

用户可以长按标记进入选择模式，可选择多条或全选标记，进行集中上传。

用户下拉界面，更新信息，自动进行本地数据与服务端数据同步（参考QQ消息刷新功能）。

离线数据同步的流程图如下图所示：

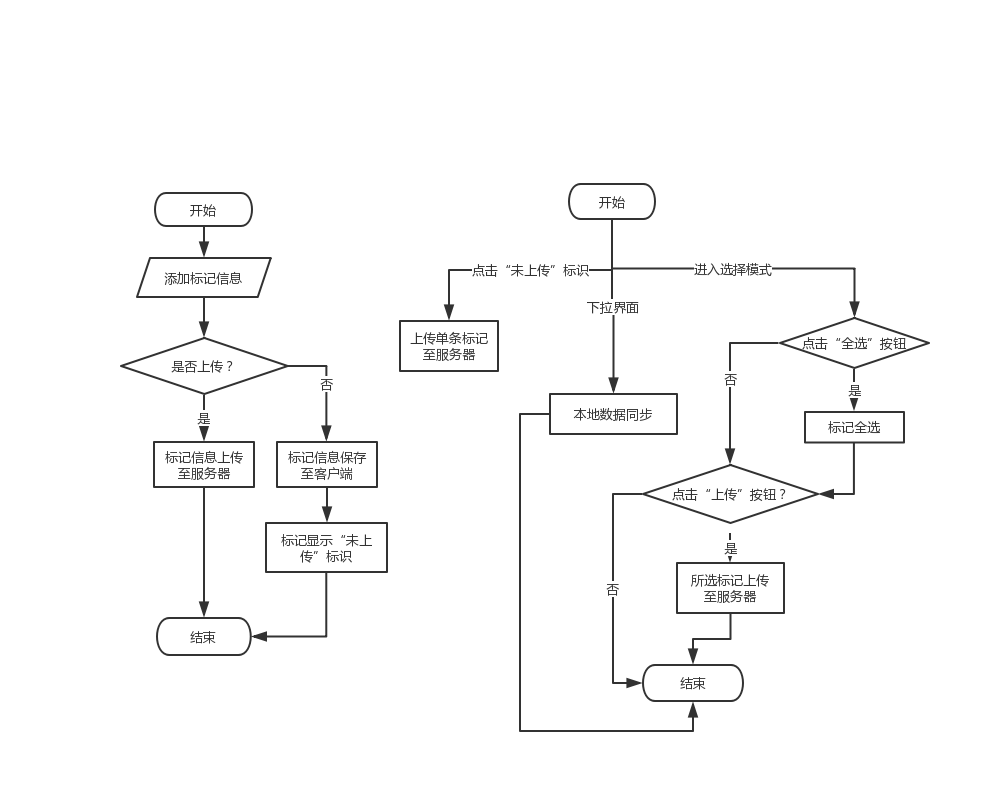


图 1 离线数据同步流程图

离线数据同步模块共有4个类，分别是离线标记类、标记类、位置类、上传信息类。各类的实现及它们之间的关系如下：

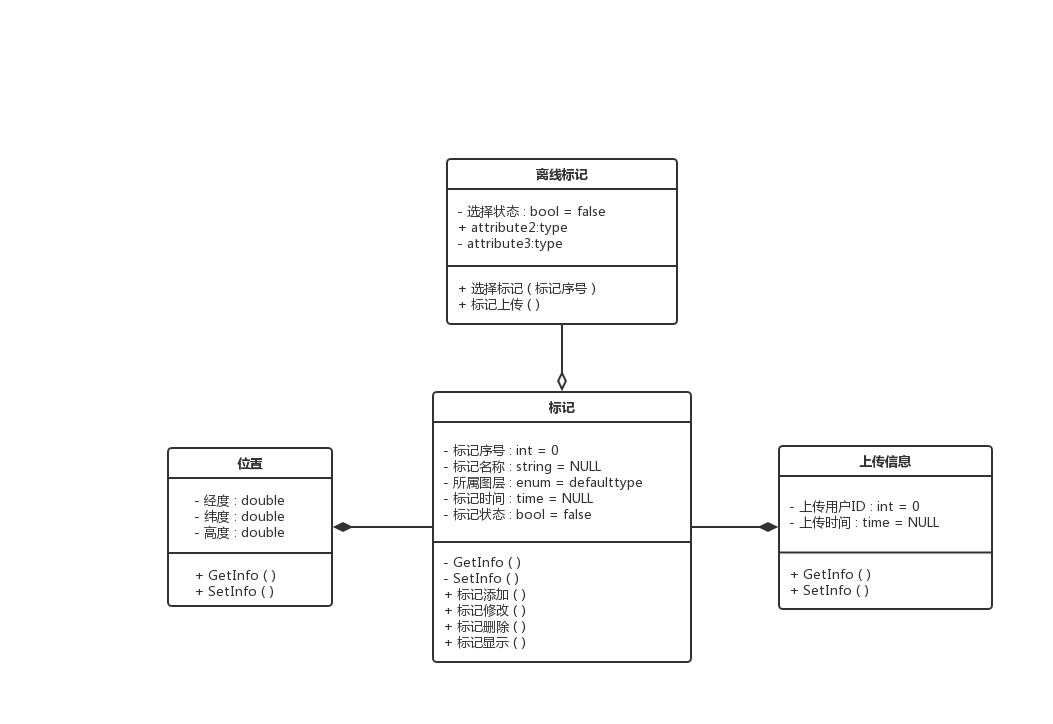


图 2 离线数据同步类图

离线数据同步需要显示离线标记信息，并对选择上传的标记进行上传操作。该部分对应的接口如下表所示：

表格 1 离线数据同步接口

|  |  |  |  |
| --- | --- | --- | --- |
| **接口名称** | **参数** | **返回值** | **说明** |
| show\_offline\_tag() | int offline\_tag\_id | bool | 根据离线标记的id显示离线比较（包括“未上传”标识按钮），返回是否显示成功 |
| upload\_offline\_tag() | int select\_tag\_id | bool | 根据选择标记的id将标记上传至服务器 |

### 4.2.5 标记信息分享

标记信息分析包括标记分享和线路分享两个部分。当用户想向第三方平台分享标记时，长按标记进入选择模式，选择一条或多条标记，点击“分享”按钮，选择想要分享的第三方平台，添加分享文字信息，分享标记；线路同理。

标记信息分享的流程图如下图所示：

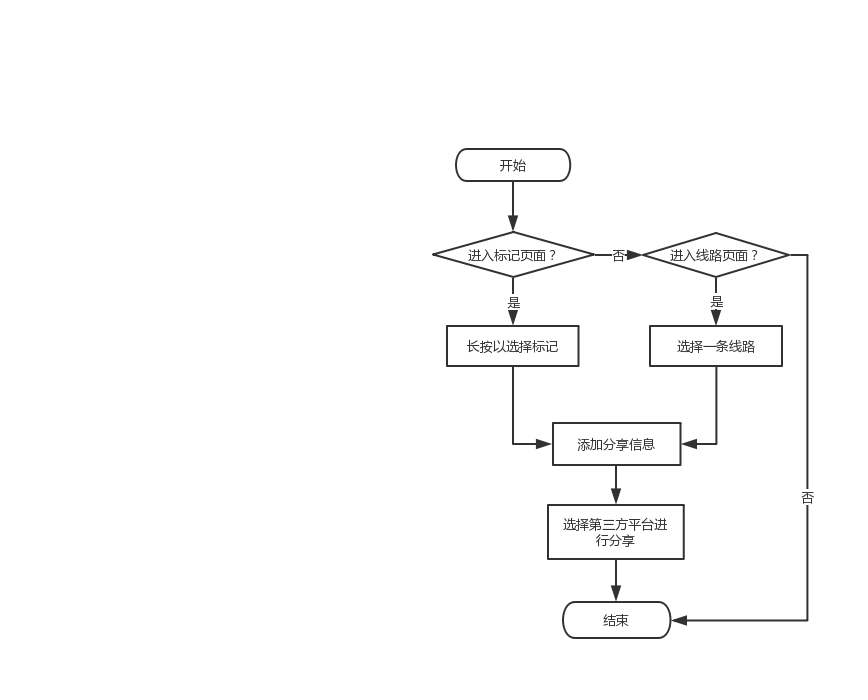


图 3 标记信息分享流程图

标记信息分享模块共有6个类，分别是分享类、分享标记类、分享线路类、链路类、位置类、上传信息类。各类的实现及它们之间的关系如下：

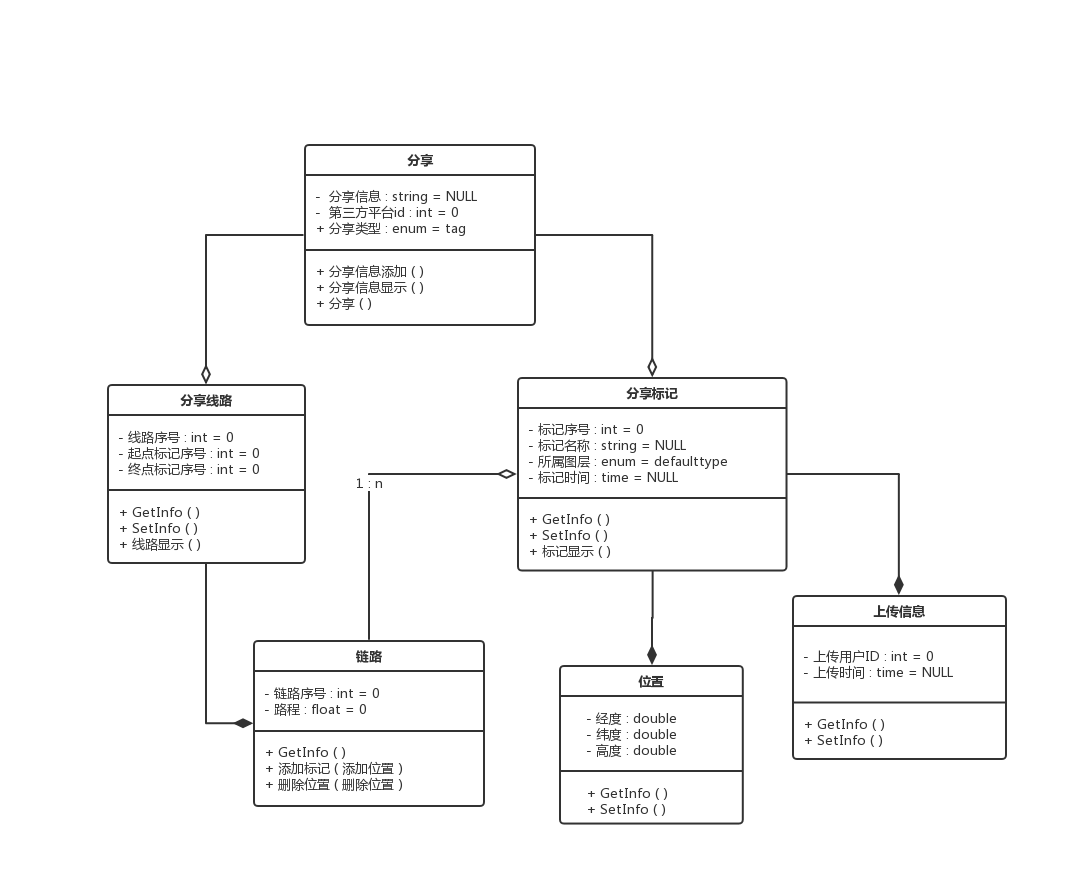


图 4 标记分享类图

标记信息分享需要选择要分享的标记或路线，添加分享信息，选择第三方平台并分享。该部分对应的接口如下表所示：

表格 2 标记分享接口

|  |  |  |  |
| --- | --- | --- | --- |
| **接口名称** | **参数** | **返回值** | **说明** |
| select\_share\_tag() | int select\_tag\_id | int | 根据选择标记的id进行选择，返回其id |
| select\_share\_line() | int select\_line\_id | int | 根据选择线路的id进行选择，返回其id |
| add\_share\_info() | string share\_info | bool | 添加分享内容，返回是否添加成功 |
| select\_platform() | int platform\_id | int | 根据第三方平台的id进行选择，返回其id |
| share() | int share\_platform\_id | bool | 将分享类中的实例化内容分享至所选平台，返回是否成功 |

### 4.2.6 信息标记

用户可以在地图的指定位置进行信息标记，并可以在标记处添加具体标记内容。之后可以将标记保存至相应图层，也可以将不同位置标记连接起来，保存至路径标记。

位置标记的流程图如下图所示：



图1 位置标记流程图

线路标记的流程图如下图：



图2 路线标记流程图

保存图层的流程图如下图：



图3 图层保存流程图

信息标记需要选择选择指定位置，添加位置标记和内容标记，并将其保存至相应路线，也可将其保存至相应图层，其接口信息如下表所示：

表格1 信息标记模块接口

|  |  |  |  |
| --- | --- | --- | --- |
| **接口名称** | **参数** | **返回值** | **说明** |
| select\_location() | double& longitude,  double& latitude,  double& height | void | 将该位置的位置信息保存在引用类型的数据中 |
| add\_location\_tag() | void | int | 在该位置新建一个location\_tag对象，并添加位置标记信息，返回该标记序号 |
| add\_content\_tag() | void | int | 在该位置新建一个content\_tag对象，并添加内容标记信息，返回该标记序号 |
| save\_layer() | int\* tags, int layer | int | 将标记内容保存至指定图层，并返回图层编号，失败则返回-1 |
| save\_routine() | int\* tags, int start, int routine | int | 将标记点保存至指定路线，并记录起点位置，返回路线编号，失败则返回-1 |

### 4.2.7 图层查看

用户可以查看本地保存的“交通图层”、“旅游图层”和“美食图层”等，以及查看其他用户通过网络分享的图层。

图层查看的流程图如下图：



图4 图层查看流程图

信息标记和图层查看模块包括“标记”、“位置”、“位置标记”、“内容标记”、“图层”和“路线标记”六个类，其具体关系如下图所示：

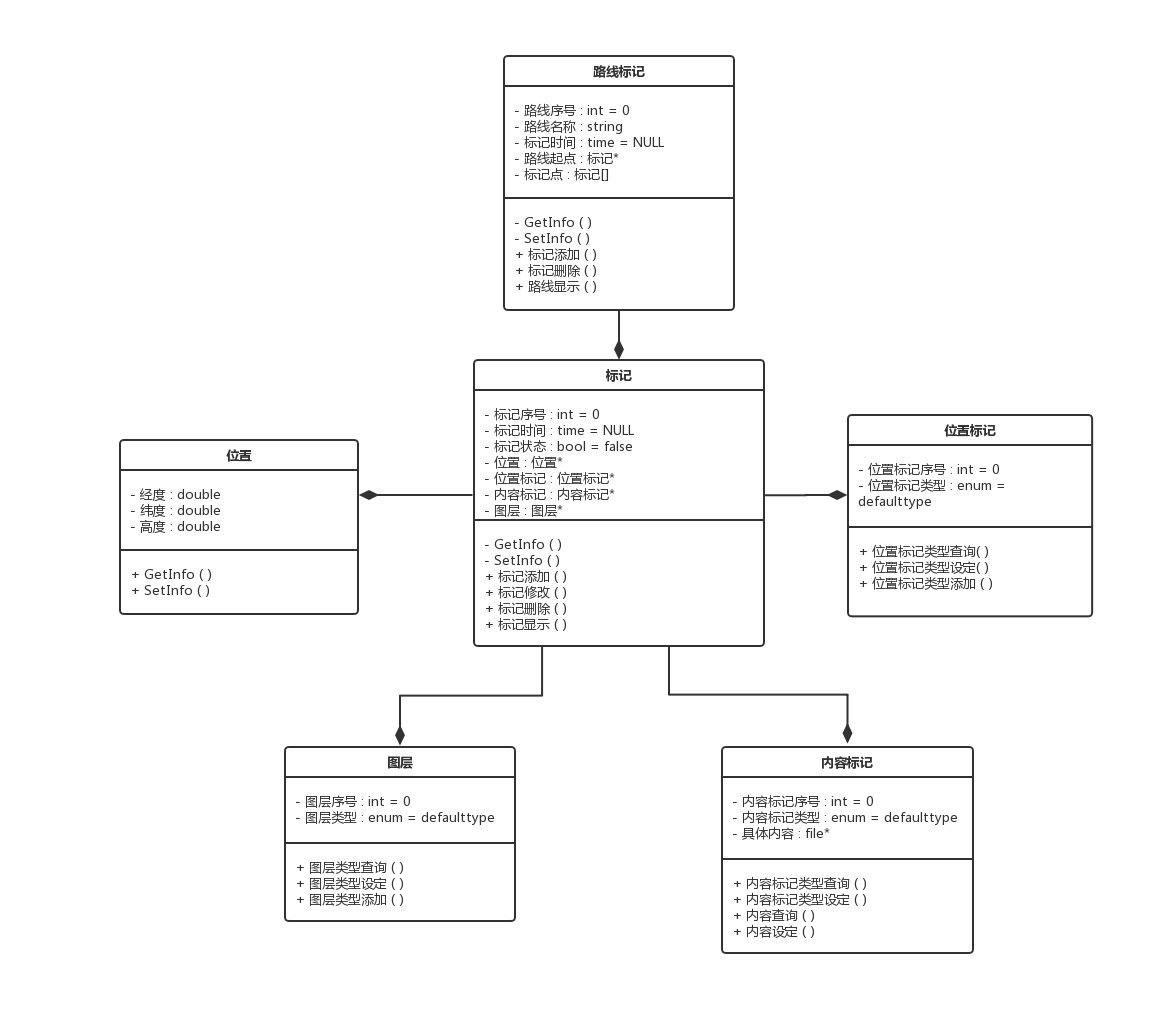


图4 信息标记和图层查看类图

图层查看包括从本地导入图层和从网络导入图层，该部分的接口设计如下表所示：

表格2 图层查看模块接口

|  |  |  |  |
| --- | --- | --- | --- |
| **接口名称** | **参数** | **返回值** | **说明** |
| import\_local() | int select\_local\_layer\_id | int | 从本地文件夹选择相应图层，并返回其id，失败则返回-1 |
| import\_net() | int select\_net\_layer\_id | int | 从服务器选择相应图层，并返回其id，失败则返回-1 |

### 4.2.8 信息发布

在信息发布部分用户可以发布文字、图片或者链接类型的标记，共分两种情况。

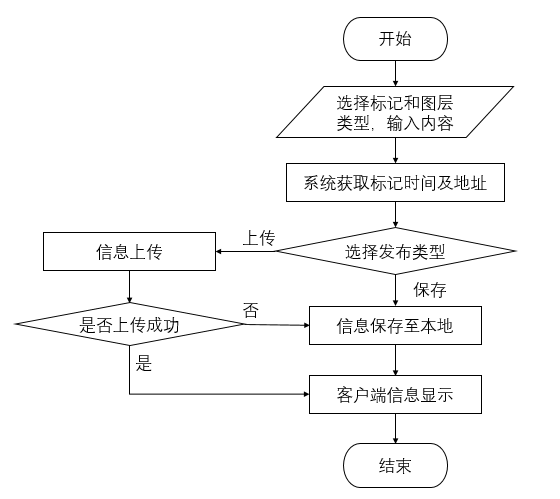
上传标记时，用户首先选择标记类型和图层类型，输入内容，选择保存或者上传按钮，系统自动获取点击按钮时的的时间和用户所在地点。如点击上传并且上传成功，将获取的时间作为上传时间，将标记上传至服务器端，客户端显示信息“标记已成功上传”。

若用户选择上传且并未上传成功，则跳至第一种情况，自动保存。

以获取的时间作为标记时间，系统将用户编辑的内容、标记时间和地点保存至本地。客户端显示信息“标记未上传成功，已将其保存至本地”。

若用户选择保存按钮，以获取的时间作为标记时间，系统将用户编辑的内容、标记时间和地点保存至本地。客户端显示信息“标记已成功保存至本地”。

图一 信息发布流程图



# 5、界面设计

## 5.1 登录/注册和密码找回界面

登录 注册 密码找回

登录/注册和密码找回界面接口：

① void GetUserLoginInfo(Shell Login, char\* username, char\* password);

功能：从登录界面获取登录用户的用户名和密码信息，并放入参数username和password所指向的内存区域中；

参数1：名称：LoginOrRegieter；类型：Shell；说明：登录窗口；

参数2：名称： username；类型：char\* ；说明：所指向区域用于存放用户名；

参数3：名称：password；类型：char\* ；说明：所指向区域用于存放密码；

返回值：无

② void GetMobileInfo(Shell RegisterOrFindPassword, char\* mobile);

功能：从注册界面或找回密码界面获取手机号信息，并放入参数mobile所指向的内存区域中，初始化与发送验证码函数相关的手机号参数；

参数1：名称：RegisterOrFindPassword；类型：Shell；说明：注册窗口或找回密码窗口；

参数2：名称：mobile；类型：char\* ；说明：所指向区域用于存放手机号；

返回值：无

③ void GetRegisterUserInfo(Shell Register, char\* username, char\* password, char\* verify\_code);

功能：从注册界面获取注册用户的用户名、密码和验证码信息，并放入参数username、password、mobile和verify\_code所指向的内存区域中；

参数1：名称：Register；类型：Shell；说明：注册窗口；

参数2：名称： username；类型：char\* ；说明：所指向区域用于存放用户名；

参数3：名称：password；类型：char\* ；说明：所指向区域用于存放密码；

参数4：名称：verify\_code；类型：char\* ；说明：所指向区域用于存放验证码；

返回值：无

④ void GetFindPasswordInfo(Shell FindPassword, char\* new\_password, char\* verify\_code);

功能：从找回密码界面获取用户的新密码和验证码信息，并放入参数new\_username和verify\_code所指向的内存区域中；

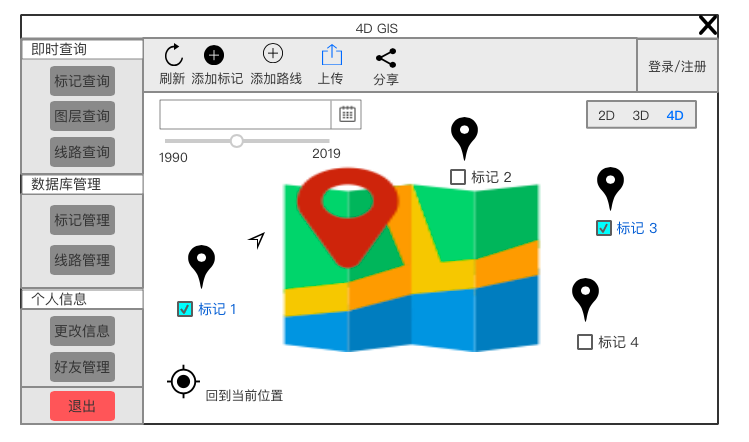
参数1：名称：FindPassword；类型：Shell；说明：找回密码窗口；

参数2：名称：new\_password；类型：char\* ；说明：所指向区域用于存放新密码；

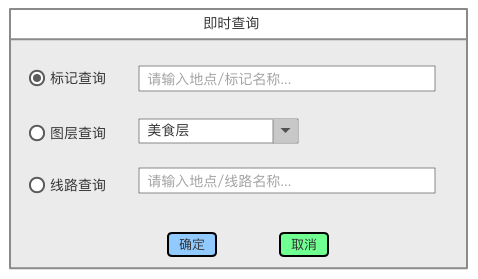
参数3：名称：verify\_code；类型：char\* ；说明：所指向区域用于存放验证码；

返回值：无

## 5.2主界面默认展示



## 5.3 即时查询界面



即时查询界面接口：

① 标记查询

void GeQueryTagInfo(Shell Query, float\* longitude, float\* latitude, float\* height, Time\* time);

功能：从即时查询界面获取待查询标记的信息，并放入参数longitude、latitude、height和time所指向的内存区域中；

参数1：名称：Query；类型：Shell；说明：即时查询窗口；

参数2：名称：longitude；类型：float \* ；说明：待查询标记的经度；

参数3：名称：latitude；类型：float\* ；说明：待查询标记的纬度；

参数4：名称：height；类型：float \*；说明：待查询标记的高度；

参数5：名称：time；类型：Time\* ；说明：待查询标记的时间；

返回值：无

② 图层查询

void GeQueryLayerInfo(Shell Query, int\* layer);

功能：从即时查询界面获取待查询图层的信息，并放入参数layer所指向的内存区域中；

参数1：名称：Query；类型：Shell；说明：即时查询窗口；

参数2：名称：layer；类型：int\* ；说明：待查询图层的ID；

返回值：无

③ 线路查询

void GeQueryLineInfo(Shell Query, int\* strat\_tag\_id, int\* end\_tag\_id);

功能：从即时查询界面获取待查询路线的信息，并放入参数strat\_tag\_id和end\_tag\_id所指向的内存区域中；

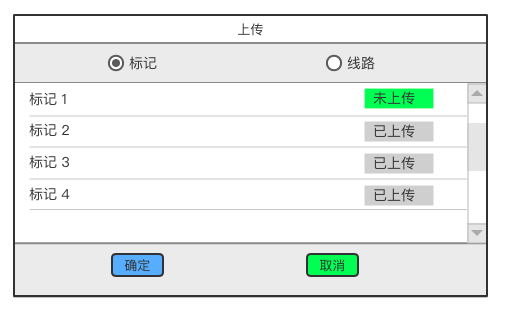
参数1：名称：Query；类型：Shell；说明：即时查询窗口；

参数2：名称：strat\_tag\_id；类型：int\* ；说明：待查询路线的起点标记ID；

参数3：名称：end\_tag\_id；类型：int\* ；说明：待查询路线的终点标记ID；

返回值：无

## 5.4 数据上传界面



数据上传界面接口：

① 标记上传

void GeUploadTagInfo(Shell Upload, Set\* tag\_id);

功能：从数据上传界面获取待上传标记的信息，并放入参数tag\_id所指向的内存区域中；

参数1：名称：Upload；类型：Shell；说明：数据上传窗口；

参数2：名称：tag\_id；类型：Set \* ；说明：待上传标记ID的集合；

返回值：无

② 线路上传

void GeUploadLineInfo(Shell Upload, List\* tag\_id);

功能：从数据上传界面获取待上传路线包含的标记的信息，并放入参数tag\_id所指向的内存区域中；

参数1：名称：Upload；类型：Shell；说明：数据上传窗口；

参数2：名称：tag\_id；类型：List \* ；说明：待上传路线包含的标记ID的链表；

返回值：无

## 5.5 分享界面



分享界面接口：

void GeShareInfo(Shell Share, List\* tag\_id, int\* platform\_id);

功能：从分享界面获取分享标记/线路及平台的信息，并放入参数tag\_id和platform\_id所指向的内存区域中；

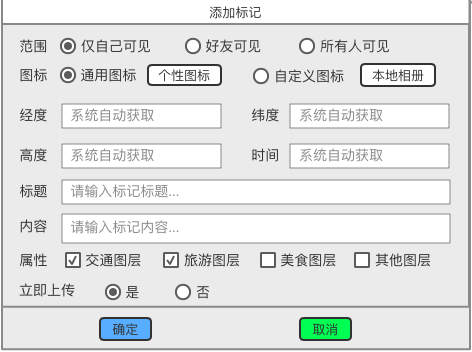
参数1：名称：Share；类型：Shell；说明：分享窗口；

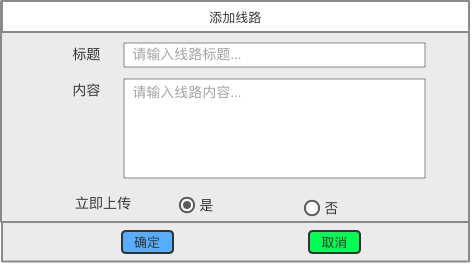
参数2：名称：tag\_id；类型：List \* ；说明：待分享标记ID的链表；

参数3：名称：platform\_id；类型：int \* ；说明：分享平台的ID；

返回值：无

## 5.6 添加标记/路线界面





添加标记/路线界面接口：

① 添加标记界面

void GetAddTagInfo(Shell AddTag, BufferedImage\* image, float\* longitude, float\* latitude, float\* height, Time\* time, char\* title, char\* content, set\* layer, boolean \* isUpload);

功能：从添加标记界面获取待添加标记的信息，并放入参数image 、longitude、latitude、height、time、title、content、layer和isUpload所指向的内存区域中；

参数1：名称：AddTag；类型：Shell；说明：添加标记窗口；

参数2：名称：image；类型：BufferedImage\*；说明：标记的图标；

参数3：名称：longitude；类型：float \* ；说明：标记的经度；

参数4：名称：latitude；类型：float\* ；说明：标记的纬度；

参数5：名称：height；类型：float \*；说明：标记的高度；

参数6：名称：time；类型：Time\* ；说明：标记的时间；

参数7：名称：title；类型：char\* ；说明：标记的标题；

参数8：名称：content；类型：char\* ；说明：标记的描述信息；

参数9：名称：layer；类型：set\* ；说明：标记所属图层的ID集合；

参数10：名称：isUpload；类型：boolean\* ；说明：是否立即上传标记；

返回值：无

② 添加线路界面

void GetAddLineInfo(Shell AddLine, int\* tag\_num, char\* line\_title, int\* strat\_tag\_id, int\* end\_tag\_id, List\* line);

功能：从添加线路界面获取待添加路线的信息，并放入参数tag\_num、strat\_tag\_id、end\_tag\_id、 line\_title和line所指向的内存区域中；

参数1：名称：AddLine；类型：Shell；说明：添加路线窗口；

参数2：名称：tag\_num；类型：int\* ；说明：待添加线路中含有的标记总个数；

参数3：名称：line\_title；类型：char\* ；说明：待添加线路的名称；

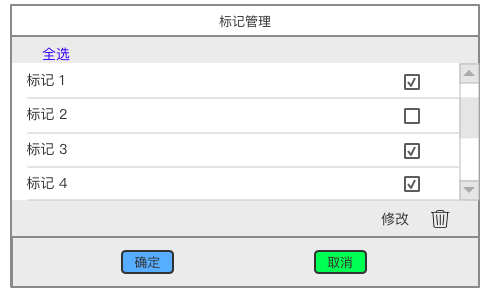
参数4：名称：strat\_tag\_id；类型：int\* ；说明：起点标记的ID；

参数5：名称：end\_tag\_id；类型：int\*；说明：终点标记的ID；

参数6：名称：line；类型：List\* ；说明：中间标记的ID链表；

返回值：无

## 5.7 数据库管理界面









数据库管理接口：

① 标记删除

void GetDeleteTagInfo(Shell DeleteTag, set\* tag\_id);

功能：从标记管理界面获取待删除的标记id，并放入参数tag\_id所指向的内存区域中；

参数1：名称：DeleteTag；类型：Shell；说明：标记管理窗口；

参数2：名称：tag\_id；类型：set\*；说明：待删除的标记ID的集合；

返回值：无

② 修改标记

void GetModifyTagInfo(Shell ModifyTag, BufferedImage\* image, float\* longitude, float\* latitude, float\* height, Time\* time, char\* title, char\* content, set\* layer, boolean \* isUpload);

功能：从修改标记界面获取修改后的标记信息，并放入参数image 、longitude、latitude、height、time、title、content、layer和isUpload所指向的内存区域中；

参数1：名称：ModifyTag；类型：Shell；说明：修改标记窗口；

参数2：名称：image；类型：BufferedImage\*；说明：标记的图标；

参数3：名称：longitude；类型：float \* ；说明：标记的经度；

参数4：名称：latitude；类型：float\* ；说明：标记的纬度；

参数5：名称：height；类型：float \*；说明：标记的高度；

参数6：名称：time；类型：Time\* ；说明：标记的时间；

参数7：名称：title；类型：char\* ；说明：标记的标题；

参数8：名称：content；类型：char\* ；说明：标记的描述信息；

参数9：名称：layer；类型：set\* ；说明：标记所属图层的ID集合；

参数10：名称：isUpload；类型：boolean\* ；说明：是否立即上传标记；

返回值：无

③ 修改线路

void GetModifyLineInfo(Shell ModifyLine, int\* tag\_num, char\* line\_title, int\* strat\_tag\_id, int\* end\_tag\_id, List\* line);

功能：从修改线路界面获取修改后的路线信息，并放入参数tag\_num、strat\_tag\_id、end\_tag\_id、 line\_title和line所指向的内存区域中；

参数1：名称：ModifyLine；类型：Shell；说明：修改路线窗口；

参数2：名称：tag\_num；类型：int\* ；说明：待添加线路中含有的标记总个数；

参数3：名称：line\_title；类型：char\* ；说明：待添加线路的名称；

参数4：名称：strat\_tag\_id；类型：int\* ；说明：起点标记的ID；

参数5：名称：end\_tag\_id；类型：int\*；说明：终点标记的ID；

参数6：名称：line；类型：List\* ；说明：中间标记的ID链表；

返回值：无

④ 路线删除

void GetDeleteLineInfo(Shell DeleteLine, set\* line\_id);

功能：从线路管理界面获取待删除的线路id，并放入参数line\_id所指向的内存区域中；

参数1：名称：DeleteLine；类型：Shell；说明：线路管理窗口；

参数2：名称：line\_id；类型：set\*；说明：待删除的线路ID的集合；

返回值：无

## 5.8 个人信息修改界面





个人信息修改界面接口：

void Get ModifyUserInfo(Shell ModifyUser, BufferedImage\* image,char\* username, char\* password);

功能：从个人信息修改界面获取用户个人修改的相关信息，并放入参数image 、username和password所指向的内存区域中；

参数1：名称：ModifyUser；类型：Shell；说明：个人信息修改窗口；

参数2：名称：image；类型：BufferedImage\*；说明：修改后的个人图像；

参数3：名称：username；类型：char\* ；说明：修改后的用户名；

参数4：名称：password；类型：char\* ；说明：修改后的密码；

返回值：无

# 6、程序设计

（伪代码）

## 6.1登录注册

## 6.2 视图切换

## 6.3 即时查询

## 6.4 离线数据同步

## 6.5 标记信息分享

## 6.6 图层选择

## 6.7 信息标记

## 6.8 信息发布

# 7、重要技术难点及解决方案