**微景观小程序**

**系统架构设计说明书**

**2018年 7 月**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 文档标识： | | WJGXCX\_RW\_01 | | | 当前版本： | | **1.0** |
| 当前状态： | | 草稿 | |  | 发布日期： | | 2018/07/07 |
| 发布 | | √ |
| **修改历史** | | | | | | | |
| **日期** | **版本** | | **作者** | **修改内容** | | **备注** | |
| 2018/07/10 | 1.0 | | 陈煦 | 初步编写SRS | | 还需进行审核 | |
|  |  | |  |  | |  | |
|  |  | |  |  | |  | |
|  |  | |  |  | |  | |
|  |  | |  |  | |  | |

[1 前言 4](#_Toc17759)

[1.1 编写目的 4](#_Toc335)

[1.2 文档范围 4](#_Toc32167)

[1.3 读者对象 5](#_Toc18201)

[1.4 参考文档 5](#_Toc642)

[2 软件系统架构设计 5](#_Toc20345)

[2.1 背景 5](#_Toc22442)

[2.2 系统分层架构视图 6](#_Toc27590)

[2.3 关键功能性需求 8](#_Toc31638)

[2.4 非功能性需求及解决方案 9](#_Toc27150)

[3 关键技术设计 11](#_Toc11060)

[3.1 接口设计 11](#_Toc10325)

[3.2 数据架构设计 12](#_Toc13868)

[3.3 安全架构设计 13](#_Toc1050)

[3.4 UI 架构设计 16](#_Toc11762)

[3.5 运维架构设计 18](#_Toc18991)

# 前言

## 编写目的

本说明书的编写目的是描述系统的架构设计方案，包括系统的软件总体架构设计及使用的框架说明，以及基于该架构的开发流程，并作为指导开发人员、测试人员进行系统开发及测试的依据。

## 文档范围

本系统架构设计文档用于开发微景观小程序，用以明确整体开发架构以及相关流程。

## 读者对象

|  |  |
| --- | --- |
| **读者对象** | **阅读重点与建议** |
| 系统详细设计人员 | 仔细阅读全部内容 |
| 系统开发人员 | 仔细阅读全部内容 |
| 系统测试人员 | 仔细阅读全部内容 |

## 1.4 参考文档

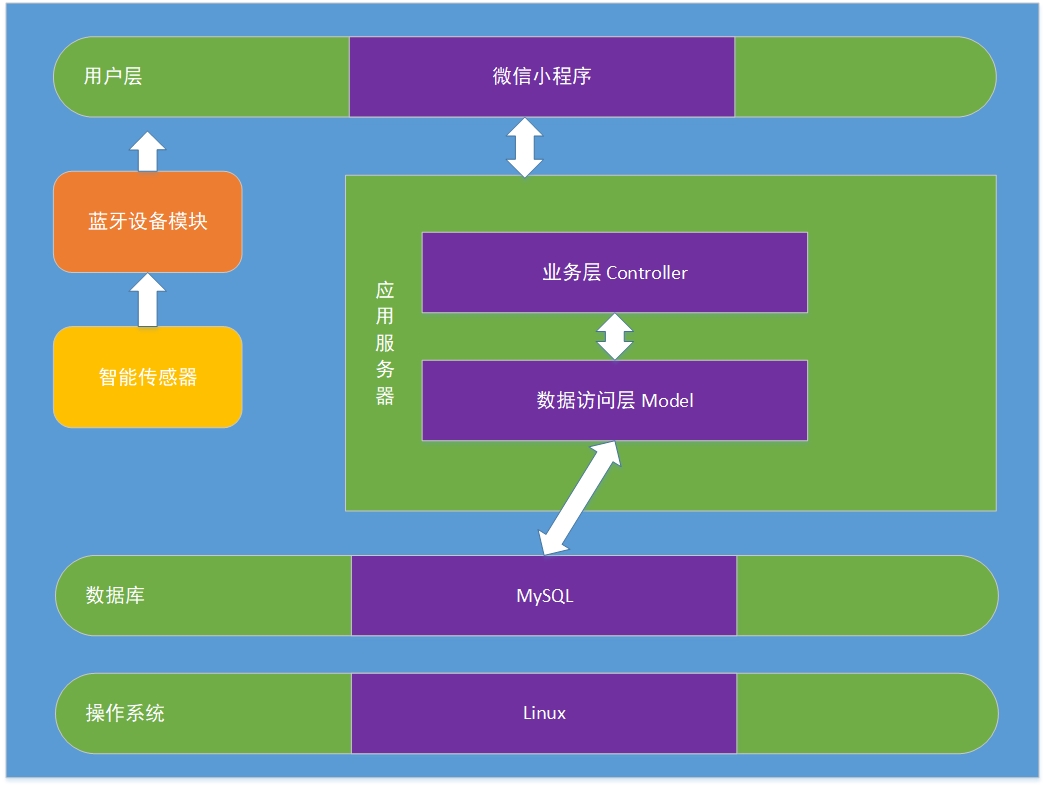
|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 文档、资料名称 | 来源 | 备注 |
| 1 | 《架构设计文档模板》 | 百度文库 | Liwenhui118上传 |
| 2 | 《软件工程基础》 | 赵一丁 | 北京邮电大学出版社 |

# 软件系统架构设计

## 2.1 背景

整个软件架构方案采用分层、分布式的部署结构，明确地分离了表现层和业务逻辑，能够保证应用服务逻辑的一致性和稳定性、结构的开放性、功能的可扩展性和可维护性、开发的可并行性，同时采用一些开源的框架，兼顾了经济性。框架是一种特殊的软件，为软件开发带来了高度的重用性，是无数软件开发人员的多年项目开发经验的总结。在一个优秀的框架上开发应用，而不是从零开发，可以大量缩短项目的开发周期、降低开发风险、增强应用系统的稳定性。

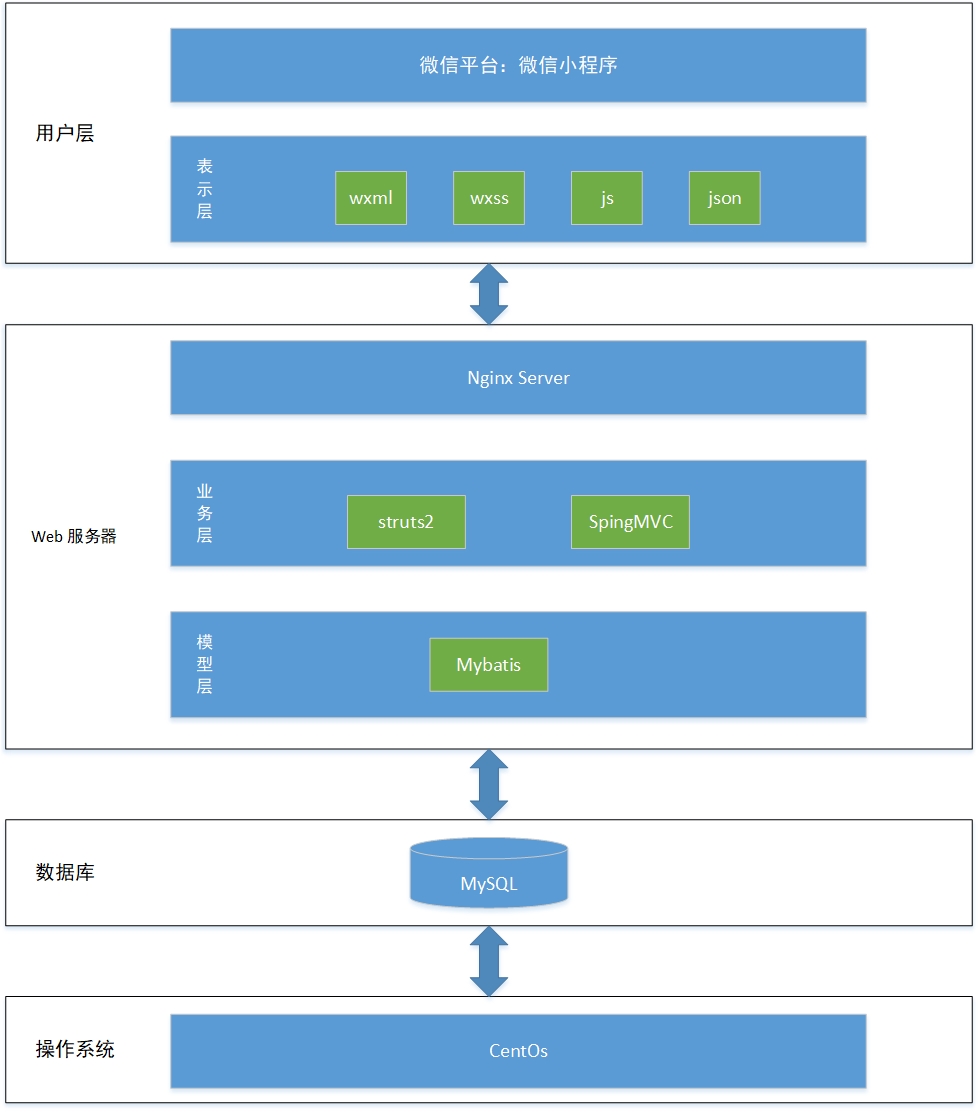
## 2.2 系统分层架构视图



系统根据功能特点和业务需求，采用基于 web 的 B/S 架构模式。开发过程实现前后端分离，因此表示层与用户层合并，服务器后台仅需处理业务层以及数据访问层。

**【数据流向说明】：**

1. 智能传感器将数据通过蓝牙协议传递给手机的蓝牙设备中。
2. 微信小程序调用蓝牙模块获取到蓝牙信息，从而读取智能传感器传递的植物数据信息。
3. 微信小程序调用 wx.request() API 向应用服务器发起请求。应用服务器接收到请求后调用请求对应的 Controller，并将前端传递的数据分发给指定 Model。
4. 数据访问层根据 Controller 传递的信息执行相应的 SQL 语句，并将命令发送到数据库。
5. 数据库执行 SQL 语句后将成功或失败的信息传回给 Model。
6. Model 将信息回馈给 Controller，而 Controller 进一步将信息包装成用户层所需的指定数据格式，并返回。
7. 用户层把应用服务器最终返回的数据呈现在页面。



分层架构图说明：

本项目基于 JAVA 代码实现，运用 Spring MVC和struts2 搭建服务器代码框架，Mybatis 实现与 MySQL 数据库的数据交互。服务器建立在 CentOs 操作系统的基础上。

用户层：微信平台——微信小程序（智能手机）。

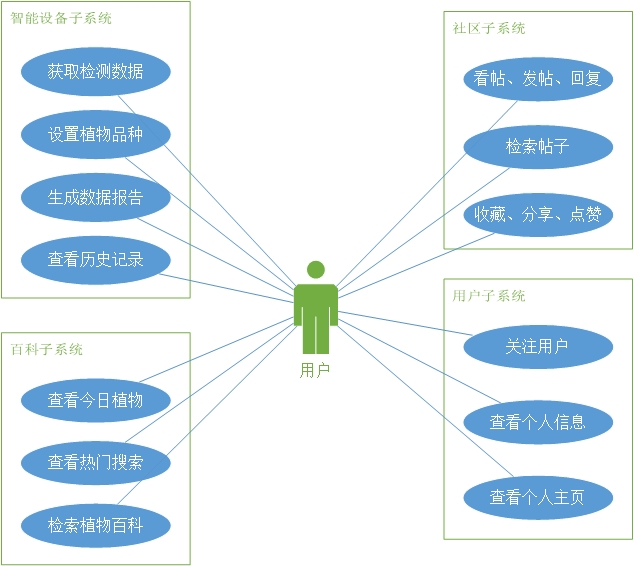
表示层：借助微信小程序特有的 wxml、wxss以及js、json语言来实现微信小程序的制作。

业务层：业务层处理应用的核心业务逻辑。业务逻辑对象把业务、规则、约束、活动和数据结合在一起。Controller 负责对这些业务对象进行管理，并运用SpirngMVC等框架，实现整个微信小程序的业务逻辑。

模型层：数据访问对象把底层的数据访问操作和上层的业务逻辑分离。

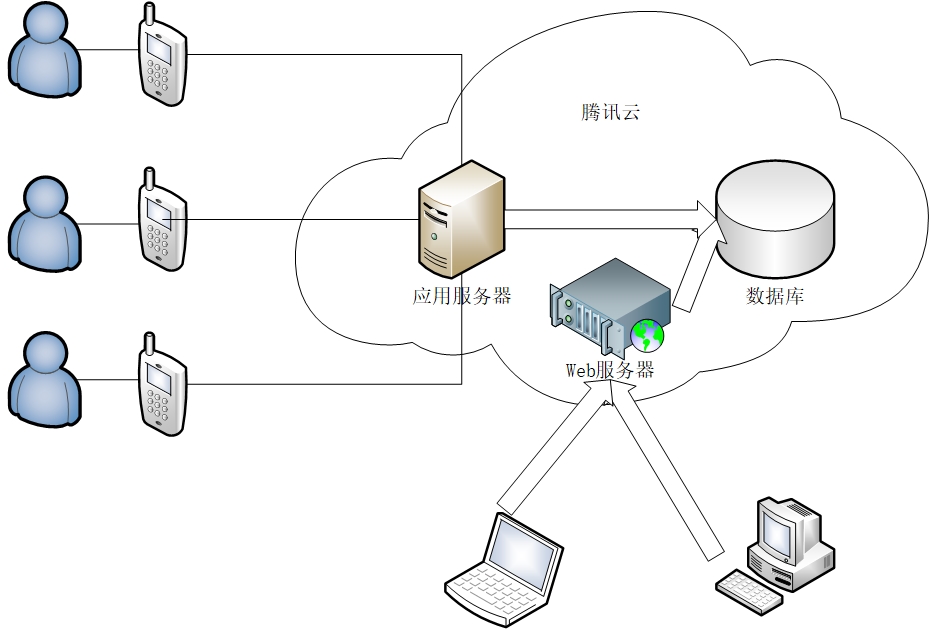
## 2.3 关键功能性需求

### 2.3.1 用例视图



### 2.3.2 逻辑视图

### 2.3.3 部署视图



## 2.4 非功能性需求及解决方案

整理和引述对于软件系统架构总体设计来说关键性的非功能需求与约束，以及针对这些非功能性需求与约束，在架构设计中考虑的解决方案。这些非功能性需求与约束来自但不仅仅限于软件需求过程产生的非功能性需求。例如安全性、保密性等。

### 2.4.1 性能需求及解决方案

1. 采用前后端分离的设计思路，将业务逻辑前移至客户端，以减少服务器端的压力。

2. 挑选性能强劲的服务器以满足所需的需求要求。

3. 后台做好并发处理（使用 Nginx 作为服务器软件）。

4. 前后端代码在完成后都需要进行优化，以提高程序的响应速度。

### 2.4.2 可靠性需求及解决方案

1. 后台代码必须保持代码的强壮性，不能因为某项操作报错而导致整个后台程序停止运行。因此在可能会出现错误的代码块处必须加上 try-catch。

2. 后台代码必须要记录下错误日志，并保存到服务器的指定文件夹下，方便后续维护以及调试工作。

### 2.4.3 安全性需求及解决方案

请参考4关键技术设计中的4.3安全架构设计。

### 2.4.4 可维护性需求及解决方案

1. 前端和后台人员在开发过程中必须完成相应的开发文档，并在代码合适处添加注释，注释应简单易懂。

2. 当代码发生变动时，应立即将变动添加到开发文档中。

【提高程序维护性的方案】：

1. 将配置数据从代码中分离出来：单独创建文件，将代码中的数据放置到该文件中。今后需要变更配置数据时，可以直接到该文件中进行修改，而不需要到代码中寻找并修改。
2. 方法名和变量名应更具语义化，能直接通过名称便知道该方法和变量的作用和意义。
3. 抽取代码中重复使用的代码块，并将其封装成构件（公用的类）。
4. 遵循结构、样式、行为分离的原则，表示层、控制层、模型层分离的指导思想。

### 2.4.5 可扩展性需求及解决方案

程序开发过程中需时刻牢记程序的扩展性，在设计接口以及方法时遵守以下规则：

1. 颗粒度小：每个接口以及方法实现一个基本操作，复杂的业务逻辑则由基本操作组合实现。这么做的好处在于——程序扩展时，不需要重新编写已有相似的代码，可以直接沿用以前编写的方法（设计的越简单、解耦度越大越好），节省开发成本。
2. 耦合度低：接口以及公有的方法不应与特定的情景耦合，应设计成在任何场合都能够使用，保持足够的客观性。

# 3 关键技术设计

针对软件系统中关键的技术性问题的解决方案给出说明。主要包括系统接口设计、数据架构设计、系统安全架构设计、UI 架构设计、运维架构设计等内容。

## 3.1 接口设计

接口设计必须符合 RESTful API 接口规范。

理解RESTful架构：[http://www.ruanyifeng.com/blog/2011/09/restful.html](http://www.ruanyifeng.com/blog/2011/09/restful.html" \t "https://blog.csdn.net/wangyanchao000/article/details/_blank)

RESTful API 设计指南：[http://www.ruanyifeng.com/blog/2014/05/restful\_api.html](http://www.ruanyifeng.com/blog/2014/05/restful_api.html" \t "https://blog.csdn.net/wangyanchao000/article/details/_blank)

【HTTP 动词】：

- GET：从服务器取出资源。

- POST：在服务器新建一个资源。

- PUT：在服务器更新资源（客户端提供改变后的完整资源）。

- PATCH：在服务器更新资源（客户端提供改变的属性）。

- DELETE：在服务器删除资源。

【HTTP动词-示例】：

获取资源：GET

Request URL: https:*//example.com/getUserInfo?limit=5*

Request **Method**: GET

新建资源：POST

Request URL: https:*//example.com/addUserInfo*

Request **Method**: POST

更新资源：PUT/PATCH

Request URL: https:*//example.com/modifyUserInfo*

Request **Method**: PUT

删除资源：DELETE

Request URL: https:*//example.com/deleteUserInfo*

Request **Method**: DELETE

【响应体格式】：前后端交互格式为 JSON。

{

"code": "1",

"msg": "返回的消息"

"data": {

// ... 数据主体

}

}

- code：响应代码，1 表示操作执行成功，0 表示操作执行失败。

- msg：响应消息，返回的消息内容。

- data：响应数据，用于存放返回给前端的所有数据内容。

【响应体格式-示例】：

{

"code": "1",

"msg": "获取用户信息成功",

"data": {

"nickname": "游学者",.

"age": 23,

"gender": "male"

}

}

## 3.2 数据架构设计

阐述与数据架构特定相关的问题，包括数据库模型、数据访问层设计。

### 3.2.1 数据库模型

### 3.2.2 数据访问层设计

【规则】：

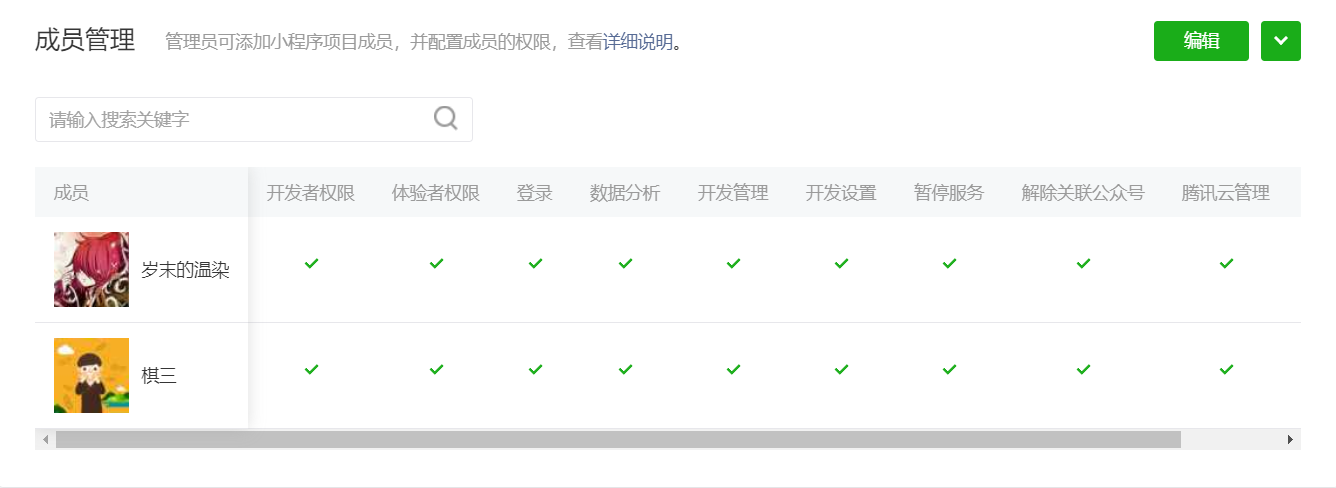
1. 涉及到多个操作时，请必须做好事务处理，保证SQL语句执行的原子性。
2. 当一个接口需要多次访问数据库时，请执行存储过程或者函数，以减少网络通讯所造成的开销。

## 3.3 安全架构设计

阐述软件系统架构中与安全相关的问题与策略，包括权限管理、身份认证、安全服务与部署和针对数据的安全规则。

### 3.3.1 权限管理

- 微信小程序：根据职责划分权限范围。



- 腾讯云服务器：根据职责划分权限范围。

### 3.3.2 身份认证



【说明】：

1. 身份认证采用微信小程序规定的方式。
2. 前端通过 wx.login() 获取 code，并将该值传递给服务器后台。
3. 服务器后台将 appid、appsecret以及code 传递给微信接口服务，以获取到 openid、session\_key，并将这些值保存下来，返回与 openid、session\_key 相关联的自定义登录态。
4. 登录成功后，前端的所有请求（wx.request()）都必须携带自定义登录态。服务器后台根据请求携带的自定义登录态去检索是否有相关联的 openid 和 session\_key。若存在，则执行相应的业务操作，并返回业务数据。否则，则返回拒绝信息。

### 3.3.3 安全服务与部署

可直接借助腾讯云自身带有的云服务器保护措施，或通过设置安全组来进一步设置服务器的安全策略。



### 3.3.4 针对数据的安全规则

1. 前端请求的参数及参数本身进行加密。

【示例】：

jegcs: jnuashdnfb *// account: 123456*

【说明】：在不知道接口文档的情况，入侵者无法直接通过参数名得知该参数的具体含义。

1. 请求携带自定义登录态：自定义登录态存放于 header 头部，而非请求体内，所以后台在获取自定义登录态时需注意这一点。

【前端请求示例】：

wx.request({

url: '地址',

method: '请求方法',

data: {

jenck: 'sdanuhcu',

neruh: 'bcbeyyga'

},

header: {

'content-type': 'application/json',

'自定义登录态': '具体的值'

},

success(res) {

*// ...*

},

error(res) {

*// ...*

}

})

【后台处理流程】：

1. 从前端请求的 header 头部取出自定义登录态，并根据自定义登录态去查找关联的 openid 以及 session\_key 是否存在。
2. 若不存在，则返回拒绝消息；否则，继续执行接口对应的业务逻辑。

## 3.4 UI 架构设计

微景观小程序的 UI 设计必须符合《微信小程序设计指南》的要求。

详情可参考以下网址：https://developers.weixin.qq.com/miniprogram/design

### 3.4.1 友好礼貌

为避免用户在微信中使用小程序服务时，注意力被周围复杂环境干扰，小程序在设计时应该注意减少无关的设计元素对用户目标的干扰，礼貌地向用户展示程序提供的服务，友好地引导用户进行操作。

#### 重点突出

每个页面都应有明确的重点，以便于用户每进入一个新页面的时候都能快速地理解页面内容。在确定了重点的前提下，应尽量避免页面上出现其他与用户的决策和操作无关的干扰因素。

流程明确

为了让用户顺畅地使用页面，在用户进行某一个操作流程时，应避免出现用户目标流程之外的内容而打断用户。

### 3.4.2 清晰明确

一旦用户进入我们的小程序页面，我们就有责任和义务清晰明确地告知用户身在何处、又可以往何处去，确保用户在页面中游刃有余地穿梭而不迷路，这样才能为用户提供安全且愉悦的使用体验。

### 导航明确，来去自如

导航是确保用户在网页中浏览跳转时不迷路的最关键因素。导航需要告诉用户，当前在哪，可以去哪，如何回去等问题。微信在小程序内现不提供统一导航栏样式，开发者可根据需要自行设计小程序首页和次级页面界面导航。建议所有的次级页面左上角提供返回上一级页面操作。此外，微信iOS用户还可通过界面边缘向右滑动操作，返回上一级小程序或微信页面。安卓用户可通过物理返回键达到同样目的。

#### 小程序菜单

小程序的所有页面，包括小程序内嵌网页和插件，微信都会在其右上角放置官方小程序菜单，样式如图。开发者不可对其内容自定义，但可选择深浅两种基本配色以适应页面设计风格。 官方小程序菜单将放置在界面固定位置，开发者在设计界面时请预留出该区域空间，若需要在此区域附近放置可交互元素，要特别注意交互事件是否会冲突，操作是否容易被使用

### 减少等待，反馈及时

页面的过长时间的等待会引起用户的不良情绪，使用微信小程序项目提供的技术已能很大程度缩短等待时间。即便如此，当不可避免的出现了加载和等待的时候，需要予以及时的反馈以舒缓用户等待的不良情绪。

### 3.4.3 便捷优雅

从PC时代的物理键盘鼠标到移动端时代手指，虽然输入设备极大精简，但是手指操作的准确性却大大不如键盘鼠标精确。为了适应这个变化，需要开发者在设计过程中充分利用手机特性，让用户便捷优雅的操控界面。

### 减少输入

由于手机键盘区域小且密集，输入困难的同时还易引起输入错误，因此在设计小程序页面时因尽量减少用户输入，利用现有接口或其他一些易于操作的选择控件来改善用户输入的体验。

避免误操作

因为在手机上我们通过手指触摸屏幕来操控界面，手指的点击精确度远不如鼠标，因此在设计页面上需点击的控件时，需要充分考虑到其热区面积，避免由于可点击区域过小或过于密集而造成误操作。当简单的将原本在电脑屏幕上使用的界面不做任何适配直接移植到手机上时，往往就容易出现这样的问题。由于手机屏幕分辨率各不相同，因此最适宜点击像素尺寸也不完全一致，但换算成物理尺寸后大致是在7mm-9mm之间。在微信提供的标准组件库中，各种控件元素均已考虑到了页面点击效果以及不同屏幕的适配，因此再次推荐使用或模仿标准控件尺寸进行设计。

### 3.4.4 统一稳定

除了以上所提到的种种原则，建议接入微信的小程序还应该时刻注意不同页面间的统一性和延续性，在不同的页面尽量使用一致的控件和交互方式。

统一的页面体验和有延续性的界面元素都将帮助用最少的学习成本达成使用目标，减轻页面跳动所造成的不适感。正因如此，小程序可根据需要使用微信提供的标准控件，以达到统一稳定的目的。

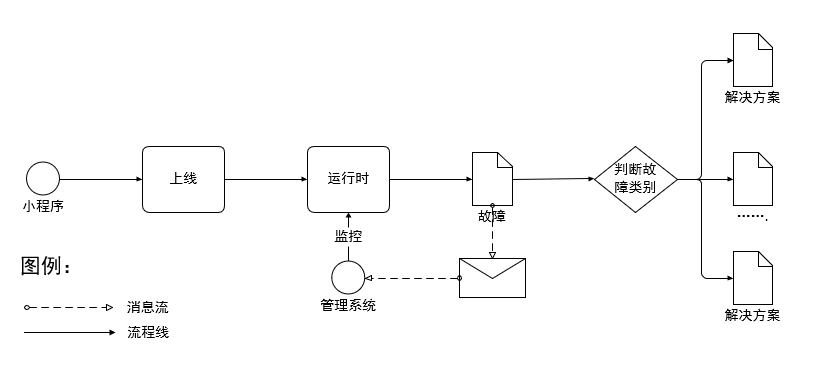
### 3.4.5 视觉规范

具体的规范请参考[《微信小程序设计指南之视觉规范》](https://developers.weixin.qq.com/miniprogram/design/" \l "%E8%A7%86%E8%A7%89%E8%A7%84%E8%8C%83)。

## 3.5 运维架构设计

阐述架构中与软件系统上线后运维相关的问题，包括故障检查与恢复、系统监控管理。

### 3.5.1 运维管理流程



流程说明：小程序成功上线后，后台管理系统需要时刻监控小程序运行时的各项指标。若小程序出现故障，则马上执行故障检查、判断故障类型，然后执行该故障类型所对应的解决方案。

### 3.5.2 故障检查与恢复

1. 服务器后台要有错误日志报告，将错误信息都记录到 error.txt 文件中。

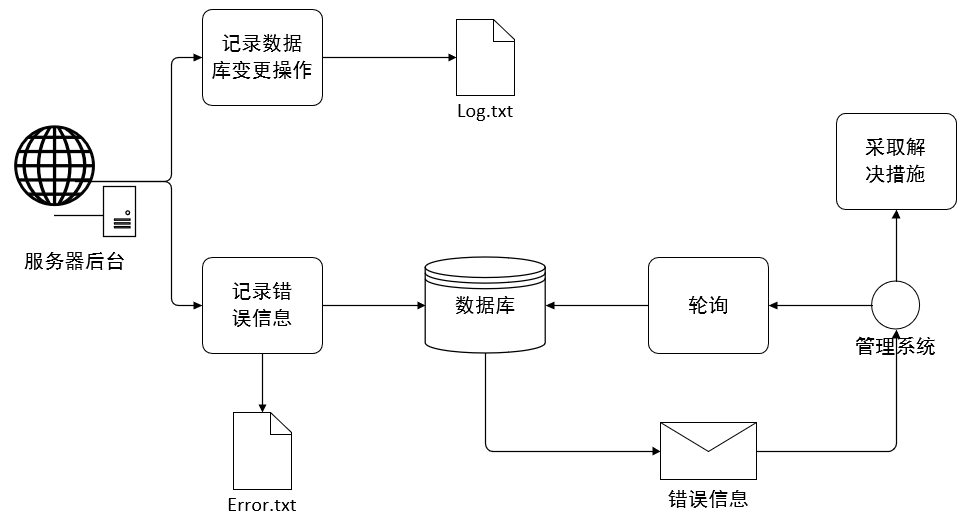
【示例】：格式 error：错误提示 操作时间。

error：java.lang.nullpointerexception 2018-7-11

1. 服务器后台涉及到数据库更改的操作都需要记录到 log.txt 文件中。

【示例】：格式 方法名 操作类型 SQL语句 操作时间。

writeArticle insert insert article values(...) 2018-7-11



【流程说明】：

log.txt 和 error.txt 文件存储在服务器指定文件夹下。若系统出现BUG，则由开发人员到服务器指定文件夹读取这两个文件，借助这两个文件的信息找出具体的BUG。

为什么需要 log.txt 文件？因为调试过程中语法错误是显而易见的，往往是逻辑错误难以发现，因此将每一次修改数据库的操作都记录在案，可以方便调试指定接口的业务逻辑是否正确。

轮询：管理系统每隔一段时间主动向服务器后台发起请求，以询问是否产生了新的错误信息（例如你在饭馆吃饭，因为人很多上菜比较慢，而你又是一个急性子，于是你时不时就去前台问服务员菜有没有做好）。若有错误信息，则立马提示管理员。

【可能存在的故障】：

1. 数据库数据有误。
2. 并发访问过多，服务器崩溃。
3. 其他错误。

【解决方案-数据库数据有误】：

1. 每天凌晨定时备份数据库，并保留每次版本更迭前后的数据库备份文件。
2. 若数据库数据有误，则立即停用小程序（在小程序的首页给出相应的提示，告知用户系统发生异常），并阅读 log.txt 和 error.txt 文件来找出造成数据库数据错误的原因。最后将最近备份的数据库替换现有的数据库。

【解决方案-并发访问服务器崩溃】：

1. 停用小程序（在小程序的首页给出相应的提示，告知用户系统发生异常），并阅读 error.txt 文件判断错误的原因。
2. 优化服务器代码，或搭建服务器集群来分担单服务器压力。

【解决方案-其他错误】：

1. 停用小程序（在小程序的首页给出相应的提示，告知用户系统发生异常），并阅读 error.txt 文件判断错误的原因。
2. 进行调试，修正错误，并重新开启小程序。

### 3.5.3 系统监控管理

后台管理系统会采用轮询（轮询概念请参考4.5.2 故障检查与恢复中轮询的说明）的方式从数据库中获取最新的错误信息，并将错误信息呈现在页面上。系统管理人员看到错误时，对照《错误应急手册》执行相应的操作。若涉及到编程性问题，则立马联系开发人员，交由开发人员来修复。

微信公众号平台以及腾讯云管理平台都提供了监控功能，具体操作请相关人员认真学习相应的操作手册。

