社交网络系统

软件构架文档

版本 <1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 2015.3.26 | 1.0 | 迭代一 | 张子辰 |
| 2015.4.22 | 2.0 | 迭代二 | 张子辰 |
|  |  |  |  |
|  |  |  |  |

目录

1. 简介 4

1.1 目的 4

1.2 功能 4

1.3 开发平台和环境 4

2. 架构风格 4

3. 逻辑视图 5

3.1 类图 5

3.2 包图 6

4. 进程视图 7

5. 部署视图 8

6. 开发视图 8

7. 监控与恢复 8

8. 可靠性 9

9. 安全 9

10. 可修改性 9

11. 易用性 9

12. 可测试性 9

13. 设计模式 10

软件构架文档

# 简介

## 目的

本文档将从构架方面对社交网络系统（即SNS）进行综合概述，其中会使用4+1视图来描述系统的各个方面。接着，它会从安全、故障监控与备份、易用性、可靠性来详细描述该系统的设计。

## 功能

该社交网络系统具有如下功能：发微博、写博客和即时聊天。

## 1.3 开发平台和环境

开发语言：Java。

组件或框架：servlet+jsp。

建模工具：PowerDesign。

数据库：mysql。

# 架构风格

该网站共使用了以下架构：

（1）浏览器和服务端间用了B/S架构。

（2）服务端的程序整体上用了MVC三层架构。

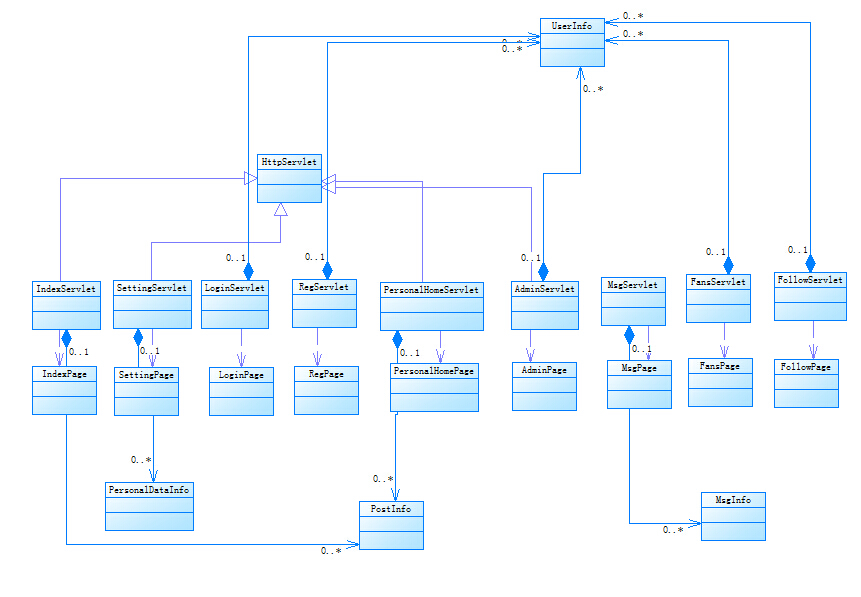
（3）控制层和数据层间用了黑板报模型。

（4）控制层的输入校检处用了管道过滤器模型。

在web开发方面，基本都是套现有的框架或者中间件，框架是哪种模型就用哪种模型，没有太多选择的余地。

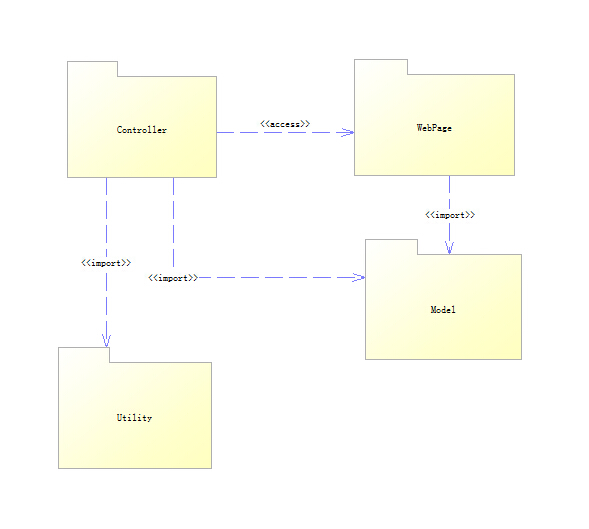
# 逻辑视图

## 类图



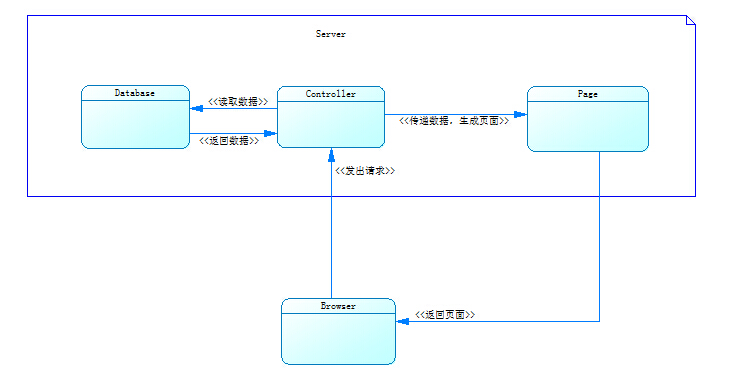
首先，按照功能划分为如下几个页面：主页、登录、注册、个人页、设置、粉丝、关注、消息、后台管理。每个页面配有一个Servelet类和一个Page。Info后缀的是存放数据实体的类。

## 包图

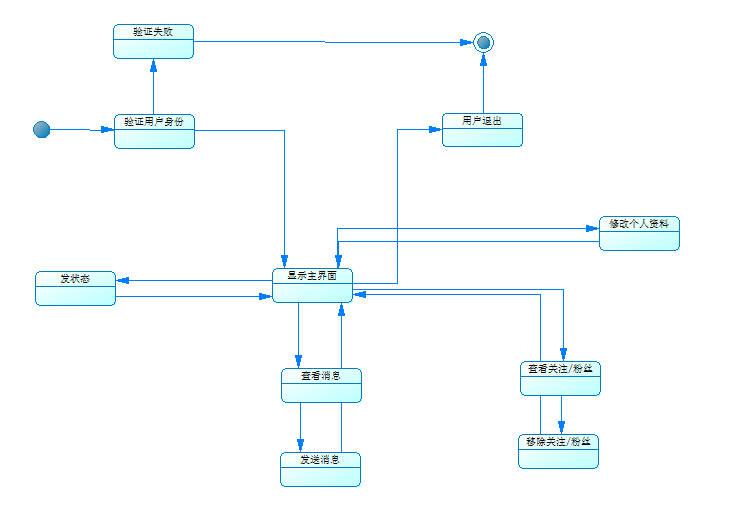


其中，Sevlet存放于Controller中，Page存放于WebPage中，Info存放于Model中。Utility中是一些功能类，比如编码解码。

# 进程视图

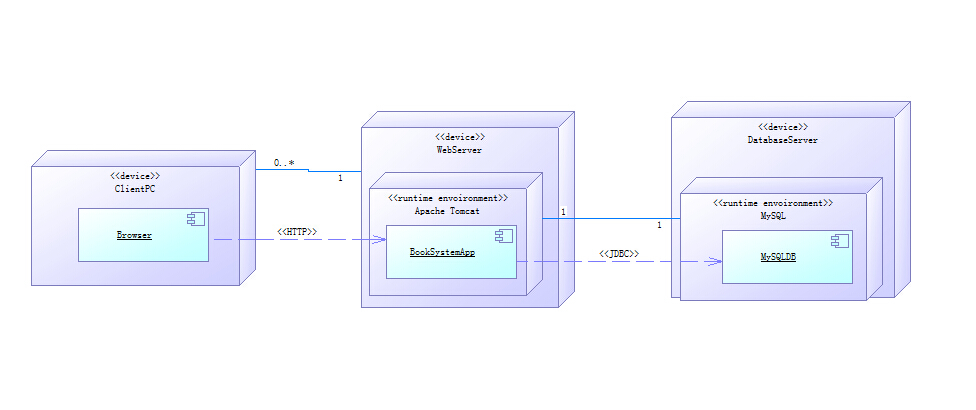


这是在一次请求之内的视图。当用户访问一个页面时，服务器会创建相应Controller类的对象，这个对象从数据库读取数据后，封装为一些数据结构，通过request传递给相应的Page。Page生成页面后返回给浏览器。



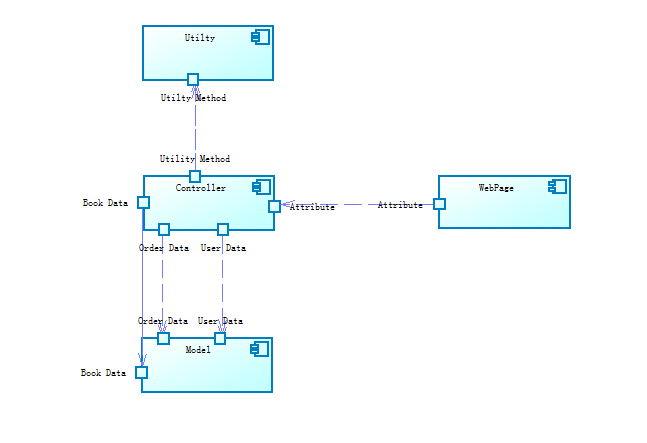
这是用户在线上的操作轨迹。由于http请求不是长连接，其中每一个操作包含一次完整的请求。

# 部署视图



其中，WebServer和DatabaseServe可以是一台服务器，也可以是一个集群，视用户数量而定。

# 开发视图



各个名字的含义和包图中相同。

# 监控与恢复

对于集群中的每台主机，采用心跳的方式进行监控。即被监测的服务器每隔一段时间，向监控服务器发心跳包，内容带上时间即可。如果没有收到某个服务器的消息，可以推算出大致是什么时候挂的。

监控一共有三种方式，除了心跳之外，有一种方案是被监控的主机在发生故障时向监控服务器发消息，但是故障可能很严重而导致消息并不能够发出。还有一种方案叫做ping或者echo，即监控用的主机定时向被监控者发请求，若收到请求则说明正常。但这种方案需要正在工作的服务器监听额外的请求，无疑增加了它的负载。综合各种因素，最后选择了心跳的方式。

该系统的数据采用被动冗余的方式进行备份。被动冗余指的是，服务器各节点之间不主动进行主备切换，而是由客户端来试，直到找到一个发送成功的节点，这个节点就自动成为主要节点。由于SNS从本质上来说可靠性要求不是很强，故采用这种策略。

# 可靠性

我们采用如下方式来提升它的可靠性：

（1）集群之间共享资源。

（2）限制单位时间请求次数，限制单位时间网络流量，限制单位时间内数据库连接次数，限制单位时间并发连接数。

（3）设置优先级。优先级相同的任务先创建先处理。

# 安全

经过我们在web开发方面的经验，采用如下方式来提升安全性：

（1）对所有具有格式的输入参数进行检验。

（2）对于所有可能带有html实体的输出参数，在page层进行转义。

（3）使用orm框架来防止sql注入。

（4）关键操作，比如添加删除等，设置并检验token来避免csrf。

（5）在所有页面的controller层，首先检验用户的权限。

（6）用户的密码用sha1或sha256加密后在数据库中存储。

（7）限制单ip24小时内的最大注册用户数。

# 可修改性

一般情况下，我们只是改动接口里面的逻辑，如果一定要改动接口，会采用增加适配器的方式来实现可修改性。适配器方式，实际上是在一个类上面实现两个接口，一新一旧，新的接口函数的实现去调用旧的。我们在开放新的接口同时，并不关闭旧的接口，以保证使用旧接口的客户端正常运行，当然旧接口只是没新接口强大罢了。

# 易用性

在易用性方面，我们会从一下几点来入手：

（1）页面使用合理的布局和颜色搭配。

（2）常使用的链接放置在突出位置，其他链接也不会放置过深。设置一个区域，用户可以自定义他们常使用的链接。

（3）顺应用户的使用习惯，通过参考其他网站，使用页面交互上的一些惯例。

此外，网站的易用性不仅仅是架构方面的事情。用户界面，包括布局和配色和交互方式，这种设计上的东西也影响者易用性。

# 可测试性

我们会考虑为测试人员开放特殊访问接口来增加可测试性。比如，重要操作前后都要记录日志，然后开放接口让测试人员可以按需访问日志的某个部分。也可以开放数据库的读权限给测试人员，便于检测数据的增删改查情况。

# 设计模式

该系统暂时采用以下设计模式：

1. 获取数据库连接时采用工厂模式。把读取配置，获取连接的细节部分放进一个factory中。
2. 在对外提供接口的扩展方面采用适配器模式，请参见10.可修改性。