# 第四章架构设计

首先介绍了整个运维系统的整体架构以及物理架构，并对本模块的关键用例的实现进行了分析。最后，介绍了本模块的静态结构分析与数据库设计。

## 系统架构及其原理

### Clip系统概述

本系统设计了五个模块，通过服务器部署、服务器管理、应用管理、监控管理、日志管理。加入docker的概念简化服务器的部署，节省资源的消耗，大大提高应用部署的效率与灵活性。为了更加清晰地理解本模块首先对整个运维系统的整体架构进行简单的介绍，具体如表4- 1所示：

表4-1 Clip系统模块划分表

|  |  |  |  |
| --- | --- | --- | --- |
| **模块** | **解决的需求** | **模块的功能** | **实现原理** |
| **服务器部署**  **模块** | 服务器批量部署困难，难以管理 | 完成对服务器的注册、电源控制以及操作系统的远程部署 | Cobbler部署软件针对服务器进行系统配置，IPMI控制电源启动  服务器上应用难以统一管理 |
| **服务器管理**  **模块** | 服务器日常管理 | 完成远程登陆服务器、备份文件、网络的相关配置 | Ssh实现服务器访问；saltstack实现文件备份；IMPI，实现网卡设置 |
| **应用管理模块** | 服务器上应用难以统一管理 | 完成对服务器远程配置应用的功能，并进行一系列的监控 | Saltstack实现docker应用的部署与配置 |
| **监控管理模块** | 服务器的监控 | 监控服务器的硬件、网络信息，增加报警功能 | Python中的包完成监控数据的获取与统计分析等 |
| **日志管理模块** | 日志采集、查看、分析、统计等 | 对服务器应用、管理系统的日志进行整理归纳 | Elasticsearch日志信息的操作；logstash完成日志信息的获取；kibana日志信息统计与展现 |

总体框架对主要的五个功能模块进行整合，提供一个交互界面传递信息，管理人员可以在界面上对服务器及上面的应用进行各种操作处理，高效管理服务器集群，快捷部署Openstack服务。

### Clip系统架构

整体架构图如图4-1所示：

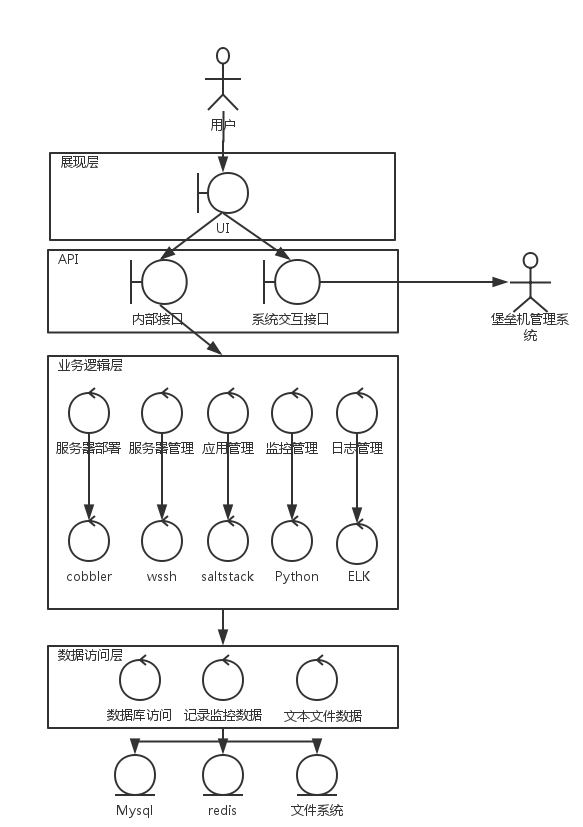


图4-1整体架构图

本系统架构清晰，功能模块众多，主要分为三层：展现层、API层、业务逻辑层、数据访问层。

* 展现层

Clip运维平台主要的接入方式是，主要是通过浏览器，浏览器接入直观与可展现性强大，并且兼容的平台会比较多，支持Windows、iOS、Linux以及Android平台。

* API层

主要有两种类型的接口方式，一种是内部API，接口已封装实现功能的函数，

调用API可以使代码逻辑更加简洁；另一种是外部API，即系统交互接口，主要是连接堡垒机管理系统，与本系统的配合使用能使数据的管理更加安全。

* 业务逻辑层

主要实现五个模块的业务功能，分别是服务器部署模块、服务器管理模块、

应用模块、监控模块、日志模块。是整个运维系统核心层。利用cobbler工具将服务器加入到管理系统中来进行集中部署操作系统，控制服务器的启动与关闭。使用wssh可以远程登陆服务器，同时进行服务器文件的管理、同步与备份，通过网络管理可以建立虚拟局域网，并对在线服务器进行网络流量限制。使用saltstack完成在服务器上进行应用部署的工作，主要针对docker，同时对容器进行监控和错误报警的配置。监控模块则使用python的部分模块对服务器状态进行持续监测，并针对监测结果提供一定的系统控制操作。日志管理主要是采用ELK模式完成。

* 数据访问层

数据库访问层主要存放的数据有数据库访问数据、记录监控数据与文本文件，它们分别存于Mysql、Redis、文件系统中，用户调用的数据均能在数据访问层找到。

总的来说，本运维平台具有良好的架构，同时可根据实际需求对该系统进行功能的扩展。

全局地来看看本系统的基础设施，它反映了本系统中间件、数据库、操作系统、服务器、存储、网络、运维终端等之间的关系，如图4-2 物理架构图所示：

用户通过个人PC中的浏览器连网访问后端服务器，后端服务器中划分接口层、业务层、数据访问层，接口层通过Ssh连接网管服务器中的堡垒机，堡垒机再与服务器集群中的服务器相连，后端服务器的数据访问层通过Sqlalchemy与数据库服务器连接，数据访问层同时与文件服务器连接，可获取文件服务器中的日志、监控、备份文件等文件。

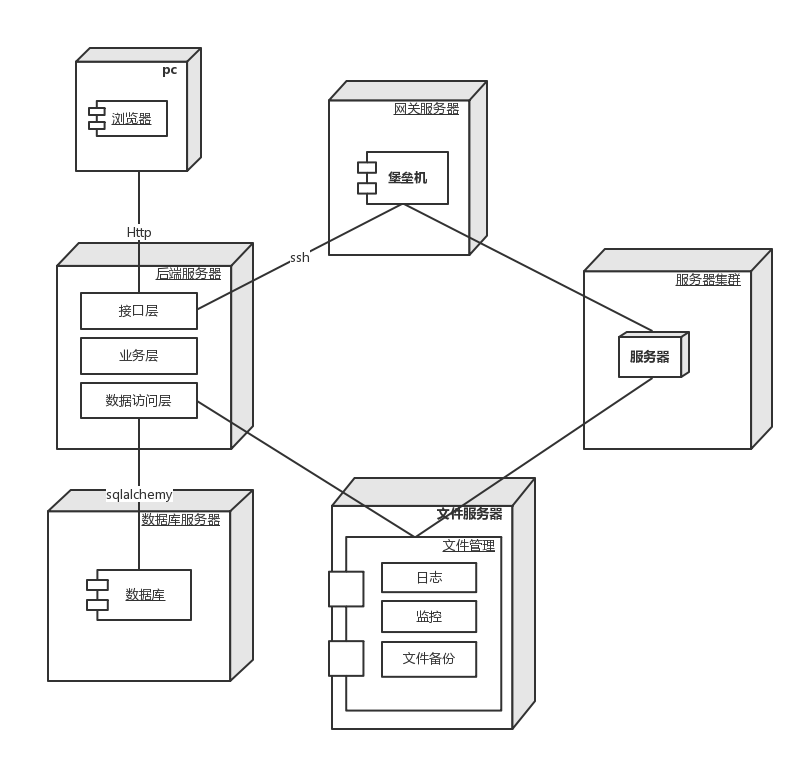


图4-2 物理架构图

## 关键用例实现

下面对本系统中监控模块的关键用例的实现进行描述：

* 监控数据管理

监控数据是监控系统中重要的一环，它主要是在实时显示的监控信息的基础上做一系列操作，需要与其他的模块协调使用。监控数据管理的接口表如表4-2所示：

表4-2 监控数据管理接口表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 名称 | 监控数据管理 | | | 子系统  名称 | 监控管理 | 系统名称 | Clip |
| 接口说明 | 输入 | 输入监控结果显示的限制条件（如只显示某些监控点、服务器，只显示某些符合条件的监控结果） | | | | | |
| 输出 | 输出符合限制条件的监控数据组成的列表 | | | | | |
| 功能说明 | 输入用户期望的监控结果显示的限制条件，对监控结果进行查询，返回符合限制条件的所有监控信息组成的列表。 | | | | | | |
| 运行环境 | 所有支持sql、redis和python的操作平台都可以运行此程序。 | | | | | | |
| 调用关系 | 调用模块 | | 服务器列表获取 | | | | |
| 被调用  模块 | | 无 | | | | |
| 数据库 | 服务器列表、收集的服务器监控信息列表 | | | | | | |

监控数据具体的实现过程如图4-3时序图：

维护人员登录运维系统后在浏览页面中点击“监控管理”按钮请求监控页面，浏览器返回监控页面，接下来用户输入监控条件，比如所有服务器的CPU、或某服务器的所有监控点，浏览器根据输入的监控条件调用筛选监控操作的API，API再向监控器调用筛选监控操作，监控器向服务器管理模块调用服务器列表获取操作，服务器管理模块向数据库查询服务器列表，数据库将请求的服务器列表返回给服务器管理，最后将返回的监控信息一步步返回到浏览器页面，完成并呈现用户想要的监控信息。

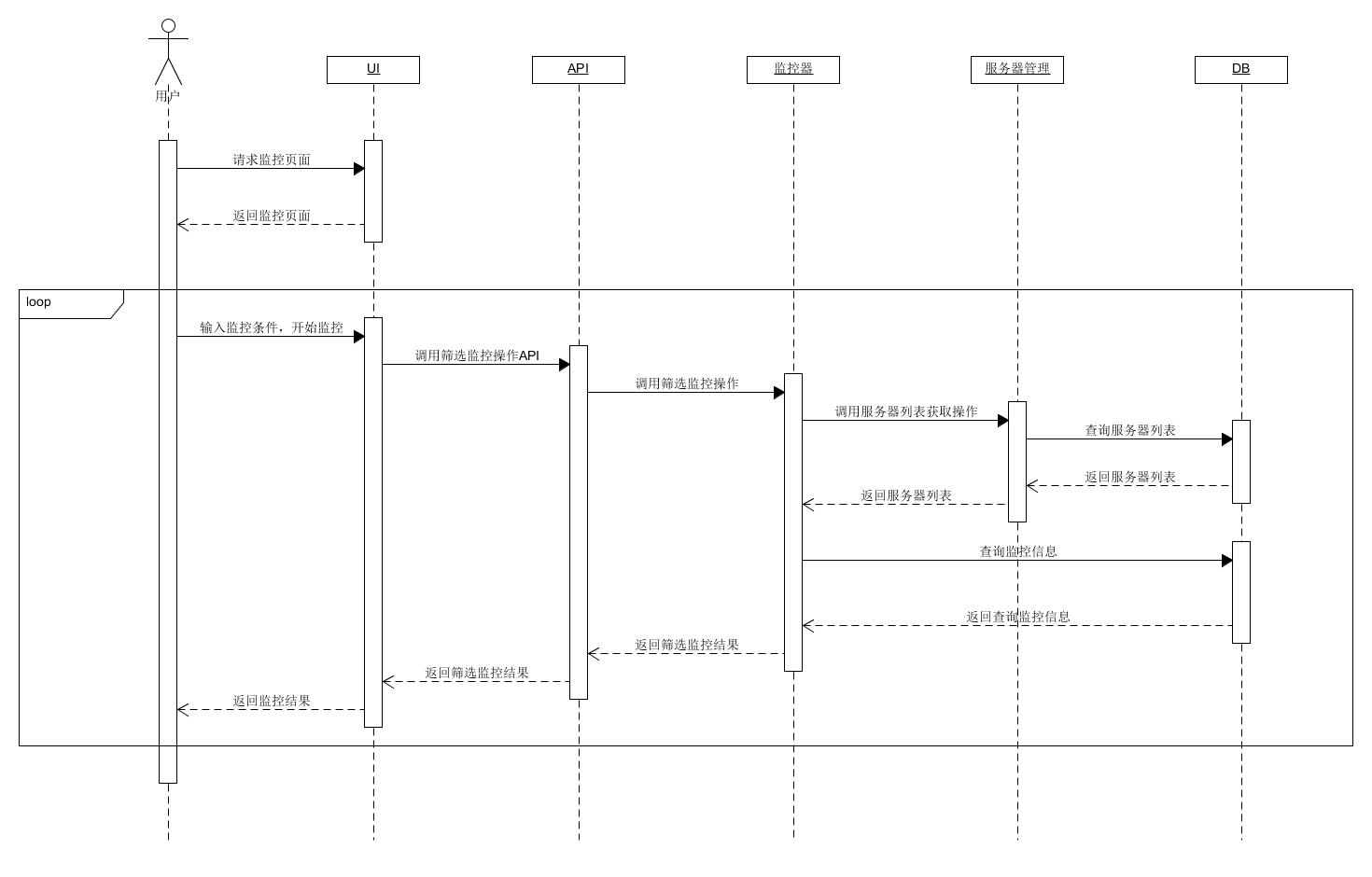
****

图4-3 监控数据管理时序图

* 监控报警设置

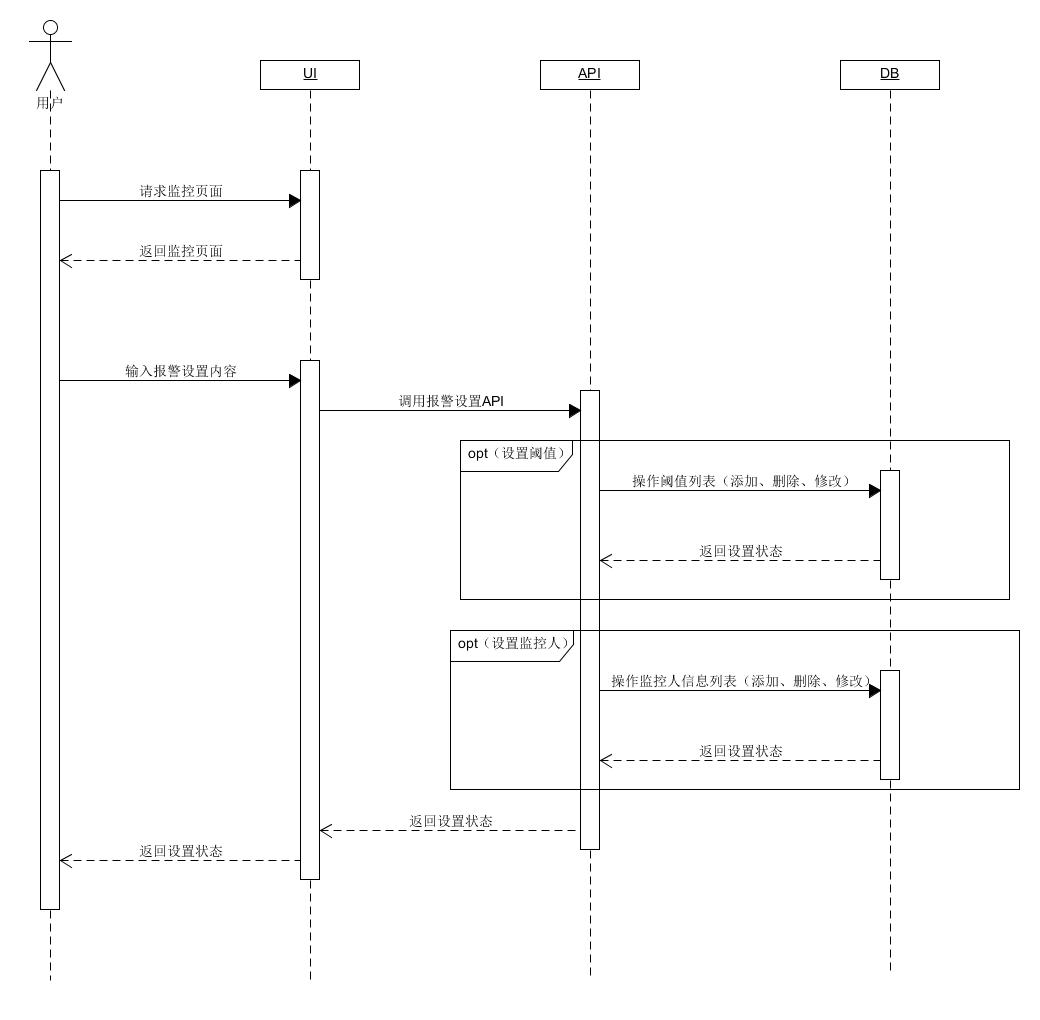
监控警报设置提供了对监控阈值和用户报警联系方式的设置功能，用户可对监控点对应的监控阈值、警报相关的设置信息（监控人、监控邮箱）进行设置如添加、删除、更改等操作。请参考表4-3：

表4‑3 监控警报设置接口表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 名称 | 监控警报设置 | | | 子系统名称 | 监控管理 | 系统名称 | Clip |
| 接口说明 | 输入 | 监控点对应的监控阈值、警报相关的设置信息（监控人、监控邮箱） | | | | | |
| 输出 | 数据库中的监控阈值列表、监控人信息列表更新 | | | | | |
| 功能说明 | 用户可对监控点对应的监控阈值、警报相关的设置信息（监控人、监控邮箱）进行设置（添加、删除、更改） | | | | | | |
| 运行环境 | 所有支持sql、redis和python的操作平台都可以运行此程序。 | | | | | | |
| 调用关系 | 调用模块 | | 无 | | | | |
| 被调用模块 | | 无 | | | | |
| 数据库 | 监控阈值列表、监控人信息列表 | | | | | | |

监控警报设置具体的实现过程如图4-4时序图：

维护人员（用户）点击“监控管理”按钮，向浏览器请求监控页面，浏览器返回监控页面，用户在返回的监控页面中选择报警设置内容，接着页面调用报警设置API，如果用户是设置监控阈值，API对数据库中的阈值列表进行操作，如添加、删除或修改操作，然后数据库向API返回设置状态，设置状态通过页面最后返回到用户，完成阈值的设置；如果用户是设置监控人信息，API对数据库中的监控人信息列表进行操作，如添加、删除或修改操作，然后数据库向API返回设置状态，设置状态通过页面最后返回到用户，完成监控人信息的设置。

****图4-4 警报邮件发送时序图

* 警报邮件发送

警报邮件发送主要是完成对服务器的监控结果进行监测，在检测到异常数据时，发送报警邮件到指定邮箱，将异常监控信息通知给维护人员。此模块需和其他模块配合完成，主要是服务器列表获取、监控数据管理、监控报警设置等模块，具体请参考表4-4：

表4‑4警报邮件发送接口表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 名称 | 警报邮件发送 | | 子系统名称 | 监控管理 | 系统名称 | Clip |
| 接口  说明 | 输入 | 需要在超出一定值时触发报警操作的监控点对应的监控阈值、警报相关的监控邮箱、服务器的监控信息 | | | | |
| 输出 | 在监测到异常数据时发送报警邮件到指定邮箱 | | | | |
| 功能  说明 | 对服务器的监控结果进行监测，在检测到异常数据时，发送报警邮件到指定邮箱，通知异常 | | | | | |
| 运行  环境 | 所有支持sql、redis和python的操作平台都可以运行此程序。 | | | | | |
| 调用  关系 | 调用  模块 | 服务器列表获取、监控数据管理、监控报警设置 | | | | |
| 被调用模块 | 无 | | | | |
| 数据库 | 服务器列表、收集的服务器监控信息列表、监控阈值列表、监控人信息列表 | | | | | |

警报邮件发送具体的实现过程如图4-5时序图：

在服务器列表获取、监控数据管理、监控报警设置模块的基础上，警报邮件发送首先是服务器调用阈值判断，报警器从数据库中查询阈值信息，数据库返回阈值信息，接着判断监控数据与阈值条件，如果监控数据超出阈值，报警器向数据库请求查询报警人信息，数据库返回监控人信息邮箱和用户名等，收到监控人信息后，报警器根据监控人的联系方式自动发送邮件至监控人邮箱，邮件发送完毕后返回发送完毕的状态。

## 

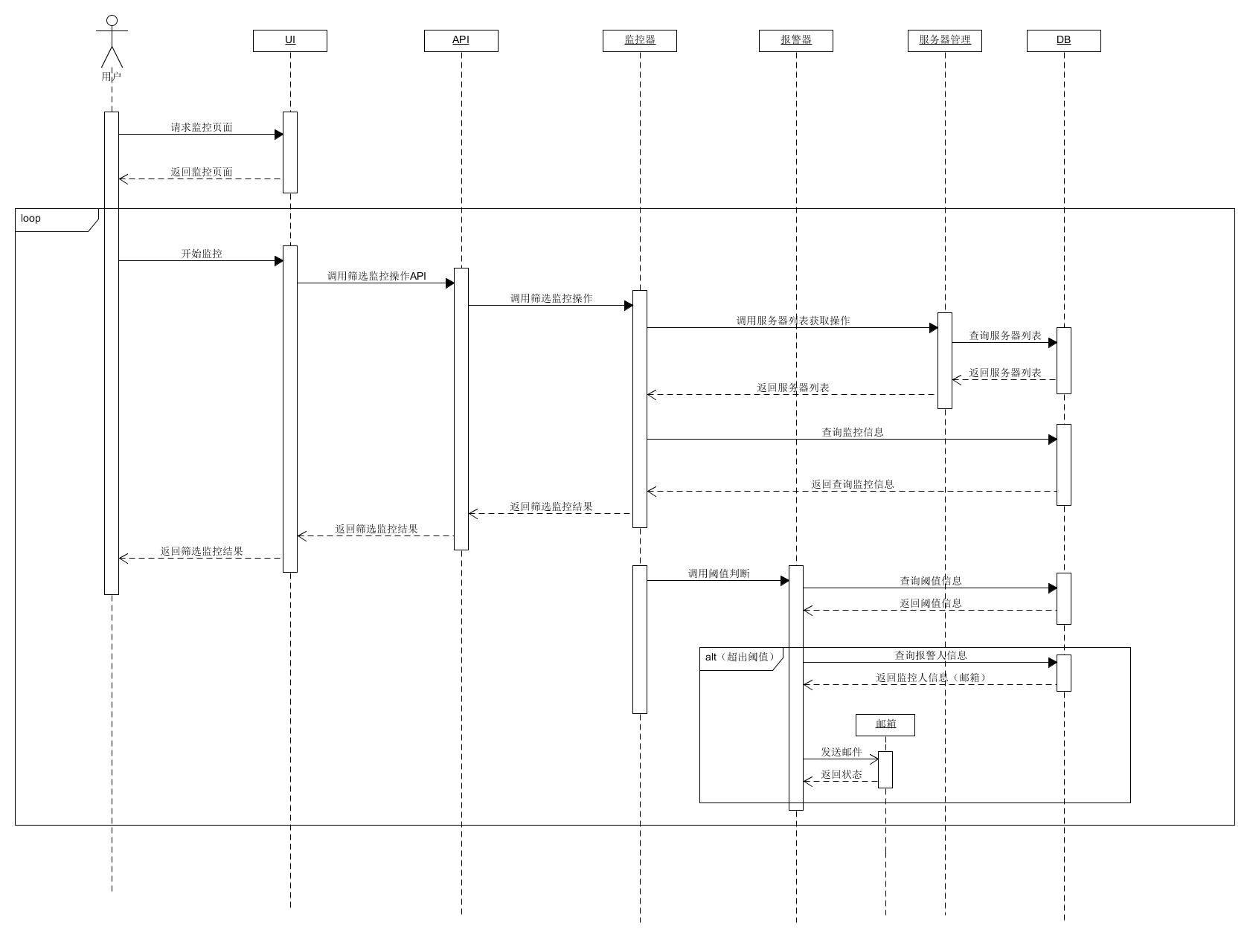
****

图4-5 警报邮件发送时序图

## 系统静态结构设计

本文的智能排课系统是基于Django框架开发的，在系统的代码结构设计上遵循Django框架的设计理念，如图4-6。

系统的代码主要有以下模块组成：

* models模块：系统所有的实体类都放在该模块中，包括Term、Course、Classroom、CourseType等，Django框架会为该模块中所有的类建立相应的数据表。
* views模块：该模块主要包含的是业务请求处理相关的函数，每一个请求都能在该模块中找到相应的处理函数。
* urls模块：负责请求URL与views函数的对应关系，所有的url都是通过该模块来查询相应的views函数的。



图4-6 智能排课系统代码结构

* forms模块：该模块包含的是一些表单校验类。当前台提交过来的表单需要进行校验的时候，可以定义相应的校验表单类来进行校验。除此之外，表单内还能根据表单中的某些参数映射出它在数据库中对应的记录。
* utils模块：包含一些工具类接口，主要提供给views模块使用。
* autoschedule模块：该模块包含的是智能排课模块的代码，主要包括scheduler、utils、managers三个模块。
* autoschedule.scheduler模块：该模块主要包含智能排课数据输入模块、课表计算模块、课表生成模块。
* autoschedule.utils模块：包含一些工具类函数，主要为autoschedule.scheduler和autoschedule.managers模块服务。
* autoschedule.managers模块：模块定义了一些参数管理类，每个管理类都对应着一类参数的管理。

## 数据库设计

**1.4.1 数据库数据说明**

根据上述的设计，本系统主要的实体有节次安排、学期、课室、课程、课程类型、课程时间安排、调课申请以及与智能排课参数相关的实体。每一个课程由教师或者管理员用户进行创建（属主），它对应着一个课程类型，并且与一个学期相对应。每一条课程时间安排记录对应着一个课程、一个上课的课室、上课日期、开始的节次和结束的节次。每个调课申请对应着一条课程时间安排记录，并且包含可调整的时间范围的信息。EER图如图4-7所示。

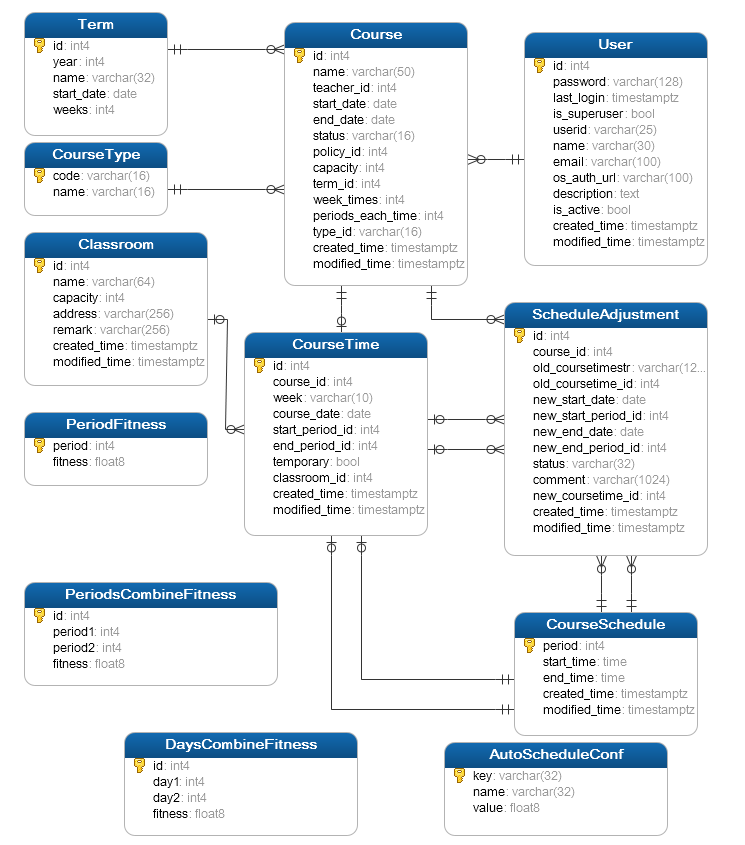


图4-7 智能排课系统EER图

下面对数据库中相关数据元素、来源、详细元素、元素名称及类型、作用性进行说明。

表4‑5数据库实体关键属性说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **数据元素** | **来源** | **详细元素** | **元素名称及类型** | **作用** |
| 监控点 | 来源于被监控的服务器本身数据 | Parameter类 | 系统时间[string]  用户信息[string]  CPU类[class]  Memory类[class]  Disk类[class]  Network类[class]  Process类[class] | 定义所有的监控点和监控项 |
| CPU类 | CPU使用率[int]  CPU空闲时间[int]  CPU数量[int] | 定义CPU监控项：CPU使用率、CPU空闲时间、CPU数量等 |
| Memory类 | 使用内存[int]  空闲内存[int]  内存使用率[int] | 定义内存监控项：使用内存、空闲内存、内存使用率等 |
| Disk类 | disk使用率[int]  disk分区[int]  IO统计[int] | 定义网络监控项：IO统计、网络连接情况等 |
| Network类 | IO统计[int]  网络连接情况[string] | 定义磁盘监控项：磁盘使用率、磁盘分区、IO统计等 |
| Process类 | 进程列表[list]  进程数量[int]  IO统计[int] | 定义进程监控项：进程列表、进程数量、IO统计等 |
| 阈值 | 来源于用户设置 | 阈值类 | 不同监控点设置各自对应的阈值[int/string] | 用户可以对每个监控点需要监控的内容设置或更改阈值，以便在达到阈值时报警 |
| 警报 | 来源于用户设置 | 警报类 | 警报邮箱[string]  收件人信息[string] | 在实时监控中，监控值超出阈值时发送报警邮件到指定邮箱 |

**1.4.2 数据库列表设计**

本监控模块需要涉及的数据库列表主要有四个数据库列表，分别是服务器列表、服务器监控监控项列表、服务器监控阈值列表、服务器监控人信息列表，详细信息如表4-6、4-7、4-8、4-9所示：

表4‑6服务器列表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **服务器ID** | **IMPI地址** | **用户名** | **密码** | **服务器状态** | **操作系统** |
| 服务器IP地址 | IPMI 用户名  IPMI密码 | 自定义  用户名 | 自定义  密码 | 开启、关闭、挂起 | Windows、lunix系统等 |

表4‑7服务器监控监控项列表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **服务器ID** | **CPU** | **Mem** | **Process** | **Disk** |
| 服务器IP地址 | CPU使用率  CPU空闲时间  CPU数量 | 使用内存  空闲内存  内存使用率 | 进程列表  进程数量  IO统计 | disk使用率  disk分区  IO统计 |

表4‑8服务器监控阈值列表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CPU** | **Mem** | **Process** | **Disk** | **Net** |
| CPU使用率  CPU空闲时间  CPU数量 | 使用内存  空闲内存  内存使用率 | 进程列表  进程数量  IO统计 | disk使用率  disk分区  IO统计 | IO统计  网络连接情况 |

表4‑9服务器监控人信息列表

|  |  |  |
| --- | --- | --- |
| **姓名** | **邮箱** | **联系方式（可选）** |
| 用户名 | 邮箱地址 | 手机号码或微信名 |