# 第二章技术与原理

本章主要是对本文所用到的核心技术进行介绍，首先阐述了监控系统的核心开发语言：Python语言。然后介绍了监控服务器应用对象：Docker。最后详细阐述了监控系统开发所用到的Web框架Django，对Django的整体架构进行了讲解。



## Python

Python语言是一种简单易学、功能强大的编程，它有高效率的高层数据结构，简单而有效地实现面向对象编程。Python简洁的语法和对动态输入的支持，再加上解释性语言的本质，使得它在大多数平台上的许多领域都是一个理想的脚本语言，特别适用于快速的应用程序开发。

主要表现在几个特点：

简单：它是一种代表简单主义思想的语言，阅读一个良好的python程序就感觉像是在读英语一样，Python的这种伪代码本质是它最大的优点之一，它能使你能够专注于解决问题而不是去搞明白语言本身。

易学：它有极其简单的语法，极其容易上手，网上有非常多的爱好社区和开源代码，能自由地发布这个软件的拷贝、阅读它的源代码、对它做改动、把它的一部分用于新的自由软件当中。

高层语言：当使用Python语言编写程序的时候，无需考虑诸如如何管理程序使用的内存一类的底层细节。

可移植性：由于其开源的本质，python程序无需修改件就可以在很多平台上使用，如Linux、Windows、FreeBSD、Solaris、OS/2、pocketPC等。

解释性：Python语言写的程序不需要编译成二进制代码。可以直接从源代码运行程序，在计算机内部，Python解释器把源代码转换成字节码中间的形式，然后把它翻译成计算机使用的机器语言并运行。事实上，由于不再需要担心如何编译程序，如何确保连接转载正确的库等，所有这一切使得使用Python更加简单。只需要把Python程序拷贝到另外一台计算机上，它就可以工作了，这也使得Python程序更易移植。

面向对象：它既支持面向过程的编程也支持面向向对象的编程，在面向过程的语言中，程序是由过程或仅仅是可重用代码的函数构建起来的。在面向对象的语言中，程序是由数据和功能组合而成的对象构建起来的。与其他主要的语言如C++和Java相比，Python以一种非常强大又简单的方式实现面向对象编程。

丰富的库：Python标准库很庞大，可以帮助处理各种工作，包括正则表达式、文档生成、单元测试、线程、数据库、网络浏览器HTML、密码系统、GUI和其他与系统有关的操作。只要安装了Python，所有这些功能都用，这被称作Python的“功能齐全”理念。除了标准库以外还有很多高质量的库，如wxPython、Twisted和Python图像库等。

### 2.2 Docker

由于Docker所基于的LXC的轻量虚拟化特点，相较于传统的虚拟化，Docker可以做到启动快且占用资源少，因此Docker是良好的构建PaaS产品的基础环境。

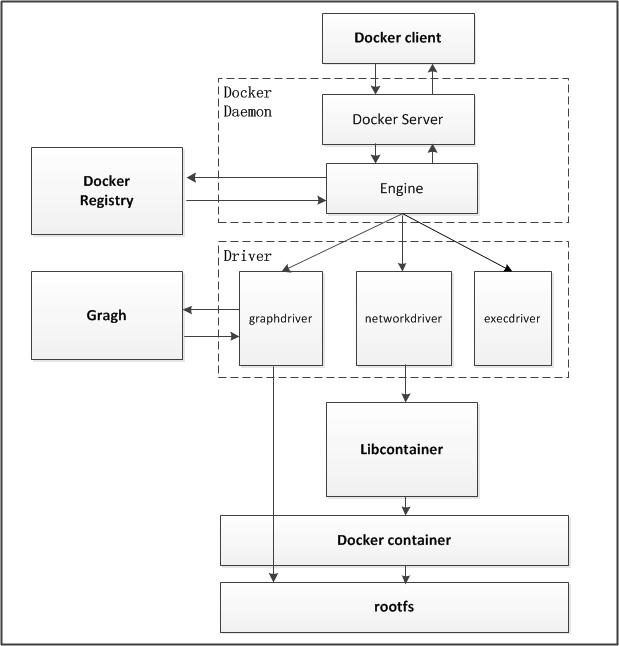
由于Docker使用了基于操作系统的虚拟化技术，因此Docker与传统的基于虚拟机的虚拟化相比，使用了不同的技术，Docker使用了命名空间技术区分不同的容器，当运行一个容器时，容器内部的进程、网络接口、进程间通信资源、挂在点以及系统内核等都在该容器的命名空间内部。Docker使用Contro Group来保障可以将硬件资源公平的分享给容器，并设置相关限制，如CPU核心，优先级以及容器的最大内存等。同时它还使用了UnionFS来为容器提供块级设备，UnionFS是一个层次化的轻量级和快速堆栈式的文件系统，可以把多个目录内容合并在一起，而目录的物理分布却是离散的。

当执行一个容器时，Docker的运行流程如下：

首先去Docker register下载指定Docker镜像，然后创建一个执行容器，为容器分配文件系统，并挂载一个读写层，然后分配相关网络接口并设置IP地址，容器准备就绪后便会在容器中执行指定的应用程序，并且捕捉应用程序的输出，并将其打印输出给用户

是一个客户-服务器架构的应用程序，其客户端和服务端既可以运行在同一个系统之上，也可以通过socket或者RESTful API实现网络通信。Docker的主要组建包括Docker Client 、Docker Daemon 、Driver、Docker Registry、Gragh、

Libcontainer、Docker container、rootfs等组件。

图2‑1 Docker结构图

## 2.3 Django

Django是一个由python编写的开源Web应用框架，采用MVC软件设计模式，即模型M，视图V，控制器C[30]。它源自一个在线新闻Web站点，于2005年以开源的形式被释放出来。Django框架核心组件主要有[31]：

1. 映射器：用作数据模型和关联性数据库间的媒介
2. URL分离器：基于正则表达式进行URL的匹配，用于关联URL与业务请求处理函数。
3. 视图系统：用于处理请求，所有的业务请求处理函数都应在此定义
4. 模板系统：用于解析模板文件（html文件，包含特定的模板语法），相当于视图层。

除此之外，Django的核心框架还包括一个轻量级的独立的Web服务器（用于开发和测试）、一个表单序列化及验证系统（用于html表单的处理）、一个缓存框架、一个可扩展的认证系统、防止CSRF（跨站请求伪造）攻击的工具等。



图2‑2 Django架构图

它的主要优势有：

1. 高级WEB框架：免去了构件动态web站点的痛苦，它把常见的Web开发的问题抽象出来并提供了对频繁编程任务的捷径。
2. 提供快速开发：可以极大程度提高开发速度，由于采用python语言，以为几天、几星期、甚至几年的站点，用DJANGO几个小时就可以做好。
3. 干净时效的设计：django严谨的维护、干净的设计，它所有代码让开发的web程序遵循最佳实践，它的哲学是：DJANGO很轻松地做事情。