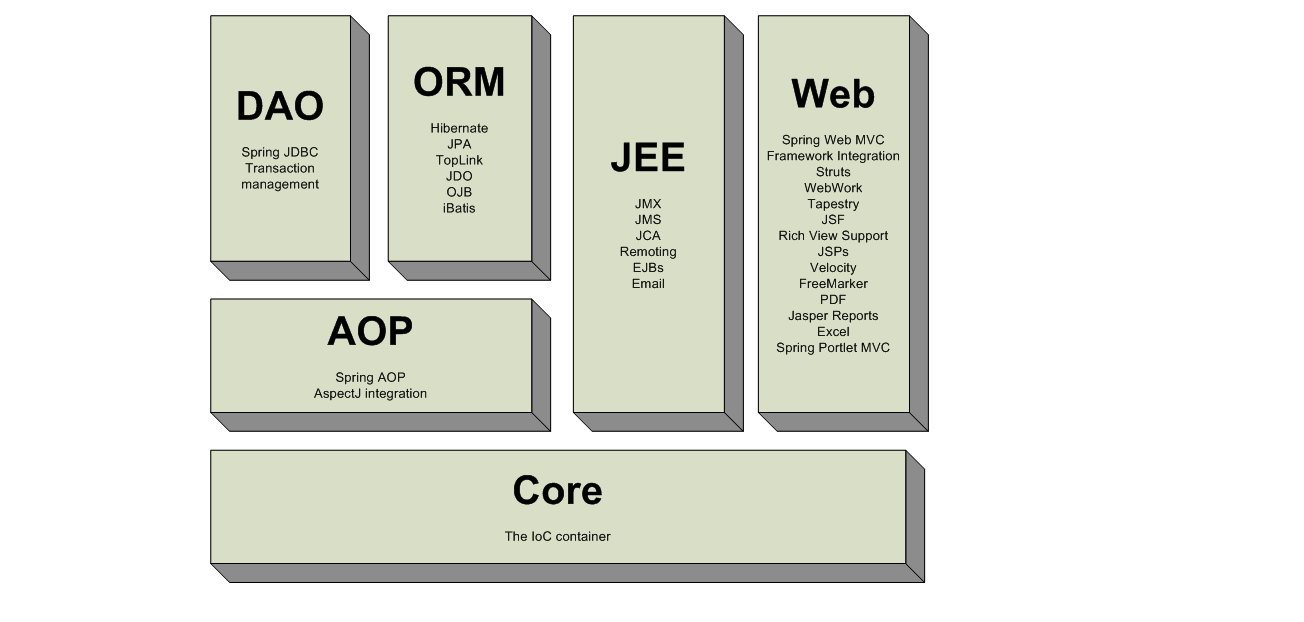
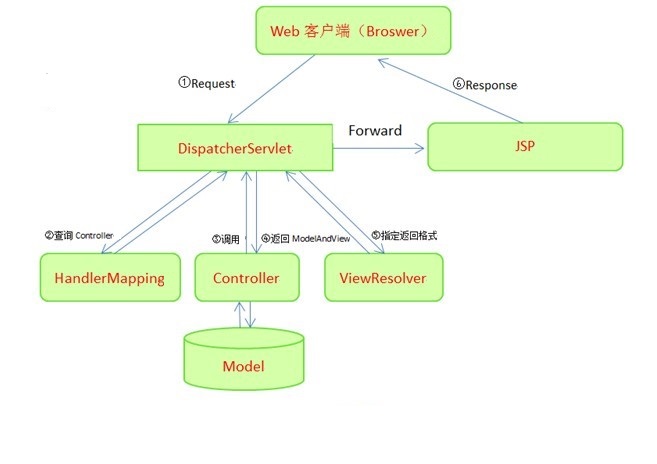
Spring MVC框架简介

Spring MVC是SpringFrameWork的后续产品，已经融合在Spring Web Flow里面。Spring 框架提供了构建 Web 应用程序的全功能 MVC 模块。使用 Spring 可插入的 MVC 架构，可以选择是使用内置的 Spring Web 框架还可以是 Struts 这样的 Web 框架。通过相应的策略配置，Spring MVC 是高度可配置的，而且支持多种视图技术，例如 Java Server Pages（JSP）技术、Velocity、Tiles、iText 和POI。Spring MVC内部不关心具体上层使用的是何种视图技术，这样用户就可以完全自主选择视图。Spring MVC 分离了模型对象（M）、视图（V）、分派及处理对象（C），这种分离更符合用户的需求，使它们能够更好的按需配置。Spring web MVC和Struts2都属于表现层的框架,它是Spring框架的一部分。相对SSH2（Spring Struts2 Hibernate）,Spring MVC 更接近原生的Servlet，控制器的单例模式，减少了大量的过滤器，这样以来，Spring MVC灵活性、性能等方面更好，我们可以从Spring的整体结构中看得出来:



Spring MVC框架是有一个MVC框架，通过实现Model-View-Controller模式来很好地将数据、业务与展现进行分离。从这样一个角度来说，Spring MVC和Struts、Struts2非常类似。Spring MVC的核心是分发器（DispatcherServlet），DispatcherServlet负责将请求派发到特定的handler。通过可配置的处理器映射（HandlerMappings）、视图解析（ViewResolution）、本地化（Locale以及主题解析（ThemeResolution）来处理请求并且转到对应的视图



Docker

* 1. Docker 背景

2013年初，Docker在dotCloud —— 一个平台即服务的、以云计算为中心的公司，以一个开源项目的形式诞生。Docker是该公司已经开发的用来在数千台服务器上运行云业务的一个自然扩展技术。它是使用Go语言编写的，快速发展了6到9个月，加入了Linux基金会，将公司名改为Docker，并且宣布将工作重心转移到Docker容器及其生态系统的开发[1]。随着Docker公司发布了第一个版本容器的产品部署以及非常活跃的社区助推，图1显示Google Trends 近年来的关注趋势，截至本文2016年6月初，Github中Docker.git star 高达31000+，fork 9000+，相信未来Docker的发展会持续高歌猛进。



图1.近年来Docker在被搜索的趋势线

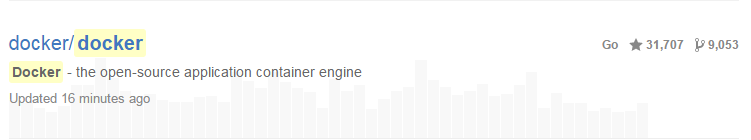


图2.Docker 在GitHub中的关注度

* 1. 什么是Docker

Docker 是开源的容器虚拟化平台，是 PaaS（Platform as a Service）提供商 dotCloud 开源的一个基于 [LXC](http://baike.baidu.com/view/6572152.htm) 的高级容器引擎，源代码托管在GitHub上, 基于go语言并遵从Apache2.0协议开源。

容器化技术并不是全新的技术，早在1982年，Unix内建的chroot机制就是一种容器化技术，Docker采用了UFS(Another/Alternative/Advance Union File System 把不同物理位置的目录合并mount到同一个目录中)文件系统来设计一个可以层层堆叠的容器镜像文件，将容器内的所有程序都打包进Docker镜像中，同时支持使用一个Dockerfile配置文件来记录建立容器过程。只要在任何支持Docker平台的环境中，就可以从这个镜像来建立出一个完全一样的容器环境来执行同一个应用程序。如此一来，就可以通过Docker镜像，或甚至只需要Dockerfile，就能将应用程序以及它的执行环境带走，迁移到任何支持Docker的环境中。通过Docker公司发布的API，使用相关指令来管理容器，这样以来就等同于将容器运用标准化了。

Docker不止是一个轻量级的容器，Docker是一个程序运行、测试、交付的开放平台，使用Docker能够使你快速地交付应用。在Docker中，可以将应用程序分为不同的基础部分，对于每一个基础部分都可以当作一个应用程序来管理。Docker能够帮助你快速地测试、快速地编码、快速地交付，并且缩短你从编码到运行应用的周期。Docker允许软件开发者将应用以及所有的依赖打包到一个标准化单元中，这个标准化的单元就是Docker 容器。

* 1. Docker架构简介
     1. Docker架构

Docker是Client/Server的架构，如图3所示，Docker客户端与服务端的Docker daemon通过RESTful API进行socket进行交互，daemon负责构建、运行和发布Docker容器。Client和Server可以运行在同一个系统中，也可以部分在不同系统中，连接远程的daemon。

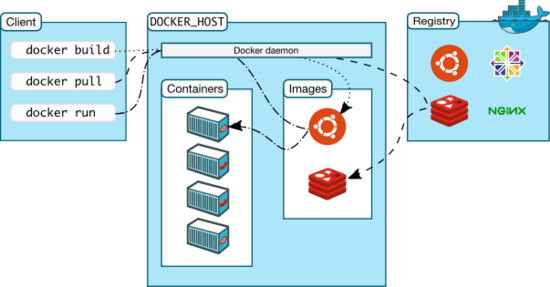


图3. Docker C/S结构图

与传统虚拟化技术相比，Docker并没有Hypervisor层。因为Docker的虚拟化技术是基于内核的CGroup和Namespace等技术，处理逻辑与内核深度融合， 所以在很多方面， 它的性能与物理机非常接近，而且可以快速地创建、开启、停止或删除。在通信上， Docker并不会直接与内核交互， 它是通过一个更底层的工具Libcontainer（自Docker1.9之后，runC 代替了Libcontainer）与内核交互的[2]。

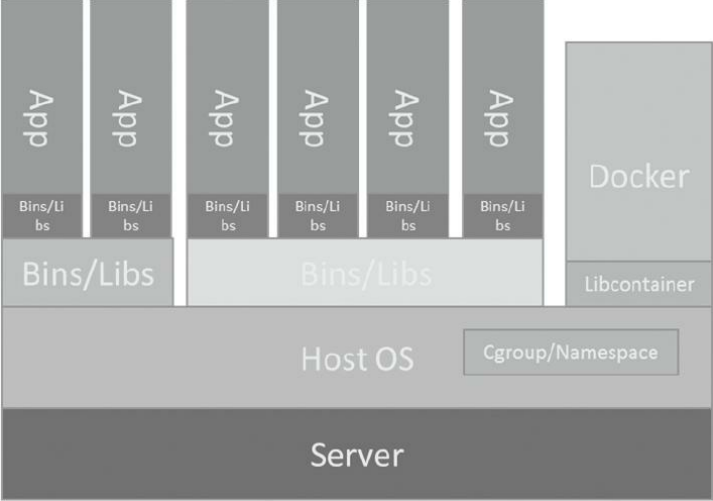


图4.Docker 架构图

* + 1. Docker内部结构

在Docker内部主要包含以下三个部分

* Docker镜像
* Docker仓库
* Docker容器

1. Docker 镜像是 Docker 容器运行时的只读模板，每一个镜像由一系列的层 (layers) 组成，如图5所示。Docker 使用 AUFS 来将这些层联合到单独的镜像中。AUFS允许独立文件系统中的文件和文件夹(称之为分支)被透明覆盖，形成一个单独连贯的文件系统。正因为有了这些层的存在，给Docker在性能上有了极大的提高。当你改变了一个 Docker 镜像，比如升级到某个程序到新的版本，就会有一个新的层会被创建。因此，不用替换整个原先的镜像或者重新建立，只是一个新的层被添加或升级了。现在你不用重新发布整个镜像，只需要升级，然后将本次升级commit，层模式使得分发 Docker 镜像变得简单和快速。

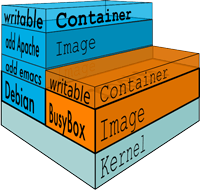


图5.Docker的层级结构

1. Docker 仓库用来保存镜像，可以理解为代码控制中的代码仓库,类似GitHub，与之相似，Docker 仓库也有公有和私有的概念
2. Docker 容器，它是Docker的核心，和文件夹很类似，一个Docker容器包含了所有的某个应用的运行环境。每一个容器都是从 Docker 镜像创建而来的。Docker 容器可以被运行、开始、停止、移动和删除。
   * 1. Docker 与VM的区别

上面提到Docker没有Hypervisor层。另外VM强依赖Guest OS，也就是在使用VM之前必须要安装Guest OS，如图6所示。

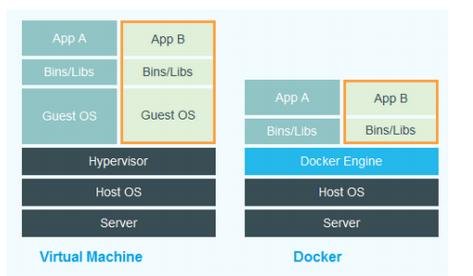


图6 . Docker与VM的区别

Docker与VM深层次的对比，如图7所示，

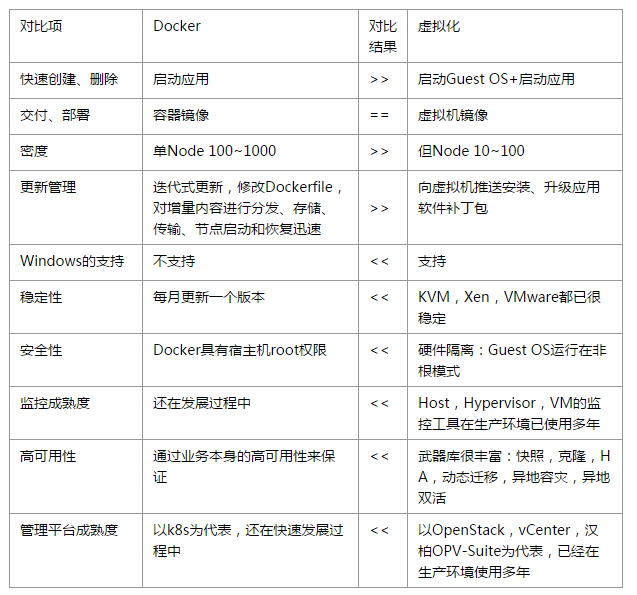


图7.Docker与VM深层次对比[3]

* 1. Docker解决的问题
     1. 现今应用部署的挑战

随着云技术的发展，服务器应用的部署已经变得越来越复杂了，以往那种执行脚本文件安装服务应用的方式日益突出其存在的弊端，例如对已安装软件、正在运行的服务、特定的操作系统的依赖以及特别是对系统资源的使用等方面，在软件开发过程中，使用到开发环境、测试环境、预发环境、线上环境等，环境的不一致导致问题排查困难，产品上线推迟等问题，显然这种方式已经不能跟上日益庞大的服务器应用的需求。特别是应用的隔离性、安全性、资源控制、应用迁移升级备份等方面都是对传统应用部署管理的重大挑战。

* + 1. Docker的解决方案

Docker从文件系统以及网络级别使用CGroup、Namespace、rootfs对应用进行了深层次的资源控制、访问隔离以及文件系统隔离，再加上Docker容器引擎加以对容器生命周期的控制。Docker镜像允许用户自定义方式或则使用Dockerfile的方式对Docker容器进行变更，发生变更之后，可以提交到原始镜像或者生成新的镜像。在多种环境的应用程序开发过程中，可以直接将配置好的镜像在不同环境之间传输或者使用Dockerfile来重新构建完全一致的镜像，这样以来就保证了开发环境的一致性，在环境发生改变的时候，只需要重新构建Dockerfile文件。

* + 1. Docker的局限性

Docker毕竟是最近发展起来的开源的容器虚拟化平台，势必存在一些不足：

* Docker是基于Linux 64bit的，无法在32bit的Linux/Windows/Mac环境下使用。
* Docker 在安全性方面存在些问题，例如从容器中获取Host 主机的root权限等，但这方面的问题Docker团队已经确定，并采取了一些控制方案。
* Docker Hub访问问题，在国内访问Docker Hub网站会存在一些稳定性问题。但是，我们可以采用国内的一些镜像，例如阿里云、灵雀云等。
* 容器技术的资源管理和运维方面，容器技术本身更适于解决大规模应用场景，所以通常都是在集群基础上的部署、运维，但是目前对这一系列任务的自动化处理尚无统一的或者标准的框架[4]。现如今发展较好的是Kubernetes有望将来制定统一标准。

# [1]. Docker —— 用于统一开发和部署的轻量级 Linux 容器

<http://www.linuxjournal.com/content/docker-lightweight-linux-containers-consistent-development-and-deployment>

[2].华为Docker实践小组 Docker进阶与实战--容器技术系列

[3].徐安 [Docker浅见](http://blog.csdn.net/cxboyee/article/details/50698398) [http://blog.csdn.net/cxboyee/article/details/50698398]