# 1.3 环境层

## 1.3.1 通用环境

### 1.3.1.1 已使用技术

#### 1.3.1.1.1 CentOS

##### 简介

CentOS（Community Enterprise Operating System，中文意思是社区企业操作系统）是Linux发行版之一，它是来自于Red Hat Enterprise Linux依照开放源代码规定释出的源代码所编译而成。由于出自同样的源代码，因此有些要求高度稳定性的服务器以CentOS替代商业版的Red Hat Enterprise Linux使用。两者的不同，在于CentOS完全开源。

新版本的 CentOS 大约每两年发行一次，而每个版本的 CentOS 会定期（大概每六个月）更新一次，以便支持新的硬件。这样，建立一个安全、低维护、稳定、高预测性、高重复性的 Linux 环境。

##### 特点

* 可以被理解为Red Hat AS系列，它完全是对Red Hat AS进行改进后发布的，各种操作、使用和RED HAT没有区别。
* 完全免费，不存在RED HAT AS4需要序列号的问题。
* 独有的yum命令支持在线升级，可以即时更新系统。
* CentOS修正了许多RHEL的BUG。
* CentOS版本说明：CentOS3.1 等同于 RED HAT AS3 Update1 CentOS3.4 等同于 RED HAT AS3 Update4 CentOS4.0 等同于 RED HAT AS4。

#### 1.3.1.1.2 Ubuntu

##### 背景

Ubuntu由马克·舍特尔沃斯亦译为沙特尔沃斯(Mark Shuttleworth)创立，其首个版本Ubuntu4.10以Debian为开发蓝本发布于2004年10月20日。Ubuntu的开发目的是为了使个人电脑变得简单易用，同时也提供针对企业应用的服务器版本。

Ubuntu的运作主要依赖Canonical有限公司的支持，同时亦有来自Linux社区的热心人士提供协助。Ubuntu的开发人员多称马克·舍特尔沃斯为SABDFL（是self-appointedbenevolentdictatorforlife的缩写，即自封终生开源码大佬）。在2005年7月8日，马克·舍特尔沃斯与Canonical有限公司宣布成立Ubuntu基金会，并提供1千万美元作为启始营运资金。成立基金会的目的是为了确保将来Ubuntu得以持续开发与获得支持，但直至2006年，此基金会仍未投入运作。马克·舍特尔沃斯形容此基金会是在Canonical有限公司出现财务危机时的紧急营运资金。

##### 简介

Ubuntu（又称乌班图）是一个以桌面应用为主的开源GNU/Linux操作系统，Ubuntu 是基于Debian GNU/Linux，支持x86、amd64（即x64）、ARM和ppc架构，由全球化的专业开发团队（Canonical Ltd）打造的。

其名称来自非洲南部祖鲁语或豪萨语的“ubuntu”一词，类似儒家“仁爱”的思想，意思是“人性”、“我的存在是因为大家的存在”，是非洲传统的一种价值观。

Ubuntu基于Debian发行版和GNOME桌面环境，而从11.04版起，Ubuntu发行版放弃了Gnome桌面环境，改为Unity，与Debian的不同在于它每6个月会发布一个新版本。Ubuntu的目标在于为一般用户提供一个最新的、同时又相当稳定的主要由自由软件构建而成的操作系统。Ubuntu具有庞大的社区力量，用户可以方便地从社区获得帮助。 Ubuntu对GNU/Linux的普及特别是桌面普及作出了巨大贡献，由此使更多人共享开源的成果与精彩。

2013年1月3日，Ubuntu正式发布面向智能手机的移动操作系统。

ubuntu基于linux的免费开源桌面PC操作系统，十分契合英特尔的超极本定位，支持x86、64位和ppc架构。

##### 特点

* 安装简单。
* Unity 3D图形界面，比较华丽（因人而异）。
* 对一些专有驱动支持比较好，例如显卡驱动。
* 社区比较活跃，几乎遇到的问题都可以找到答案。
* 版本更新较快，基本半年一个版本。

#### 1.3.1.1.3 RancherOS

##### 简介

在生产中运行Docker，最小、最简单的方法之一是使用RancherOS。

RancherOS中的所有组件都是由Docker管理的容器运行，包括系统服务，如udev和syslog。因为RancherOS只包含运行Docker所需的服务，所以RancherOS比大多数传统操作系统小得多。通过删除不必要的库和服务，可以大大减少对安全补丁和其他维护的要求。因为使用Docker，用户通常会将所有必需的库打包到其容器中，所以宿主机系统不一定需要这些库文件。

RancherOS专为运行Docker而设计的另一种表现是它始终运行最新版本的Docker，允许用户使用最新的Docker功能和修复错误。

与其他极简主义的Linux发行版一样，RancherOS的启动速度非常快，通常在5-10秒内完成。启动Docker容器几乎是即时的，类似于启动任何其他进程。这种速度非常适合采用微服务和自动扩展的组织架构。

Docker是一个专为开发人员、系统管理员和DevOps设计的开源平台，它用于构建，传输和运行容器，使用简单而强大的CLI（命令行界面），您可以从Docker用户指南开始使用Docker 。

RancherOS中运行着两个Docker守护进程。第一个名为System Docker， RancherOS在这里运行ntpd和syslog等系统服务。您可以使用system-docker命令来控制System Docker守护进程。

系统上运行的另一个Docker守护进程是Docker，可以使用普通的docker命令访问它。

当您第一次启动RancherOS时，Docker守护进程中没有运行容器。但是，如果对系统Docker运行相同的命令，您将看到RancherOS附带的许多系统服务。

##### 特点

* 可以与 Docker 的开发速度相匹配，提供最新版本的 Docker；
* 不再需要复杂的初始化系统，使用一个简单的配置文件，管理人员很容易就可以将系统服务配置成 Docker 容器；
* 容易扩展，用户很容易通过配置使 RancherOS 启动一个自定义的控制台容器，提供 Ubuntu、CentOS 或者 Fedora 发行版的体验；
* 资源占用小，启动速度快，容易移植，安全性更好；
* 升级、回滚简单；
* 可以使用像 Rancher 这样的容器管理平台，容易维护。

#### 1.3.1.1.4 Git

##### 背景

Linus在1991年创建了开源的Linux，从此，Linux系统不断发展，已经成为最大的服务器系统软件了。

Linus虽然创建了Linux，这么多人在世界各地为Linux编写代码，那Linux的代码是如何管理的呢？

事实是，在2002年以前，世界各地的志愿者把源代码文件通过diff的方式发给Linus，然后由Linus本人通过手工方式合并代码。

Linus坚定地反对CVS和SVN，这些集中式的版本控制系统不但速度慢，而且必须联网才能使用。有一些商用的版本控制系统，虽然比CVS、SVN好用，但那是付费的，和Linux的开源精神不符。

到了2002年，Linux系统已经发展了十年了，代码库之大让Linus很难继续通过手工方式管理了，社区的用户也对这种方式十分苦恼，于是Linus选择了一个商业的版本控制系统BitKeeper，BitKeeper所在的公司BitMover，授权Linux社区免费使用这个版本控制系统。

2005年，BitMover公司收回Linux社区的免费使用权。

2005年，Linus用C写了一个分布式版本控制系统，这就是Git。一个月之内，Linux系统的源码由Git管理。

2008年，Git迅速成为最流行的分布式版本控制系统，GitHub网站上线，它为开源项目免费提供Git存储，无数开源项目开始迁移至GitHub，包括jQuery，PHP，Ruby等等。

##### 简介

Git是一个开源的分布式版本控制系统，可以有效、高速地处理从很小到非常大的项目版本管理。Git 是 Linus Torvalds 为了帮助管理 Linux 内核开发而开发的一个开放源码的版本控制软件。

##### 特点

* 从服务器上克隆完整的Git仓库（包括代码和版本信息）到单机上。
* 在自己的机器上根据不同的开发目的，创建分支，修改代码。
* 在单机上自己创建的分支上提交代码。
* 在单机上合并分支。
* 把服务器上最新版的代码fetch下来，然后跟自己的主分支合并。
* 生成补丁（patch），把补丁发送给主开发者。

##### 功能

* 查看邮件或者通过其它方式查看一般开发者的提交状态。
* 打上补丁，解决冲突（可以自己解决，也可以要求开发者之间解决以后再重新提交，如果是开源项目，还要决定哪些补丁有用，哪些不用）。
* 向公共服务器提交结果，然后通知所有开发人员。

#### 1.3.1.1.5 Gitlab

##### 背景

GitLab是由GitLabInc.开发，使用MIT许可证的基于网络的Git仓库管理工具，且具有wiki和issue跟踪功能。使用Git作为代码管理工具，并在此基础上搭建起来的web服务。

GitLab由乌克兰程序员DmitriyZaporozhets和ValerySizov开发，它使用Ruby语言写成。后来，一些部分用Go语言重写。截止2018年5月，该公司约有290名团队成员，以及2000多名开源贡献者。GitLab被IBM，Sony，JülichResearchCenter，NASA，Alibaba，Invincea，O’ReillyMedia，Leibniz-Rechenzentrum(LRZ)，CERN，SpaceX等组织使用。

##### 简介

GitLab 是一个用于仓库管理系统的开源项目，使用Git作为代码管理工具，并在此基础上搭建起来的web服务。安装方法是参考GitLab在GitHub上的Wiki页面。

##### 特点

* Web框架使用RubyonRails。
* 基于MIT代码发布协议。
* 需要gitolite协同工作。

#### 1.3.1.1.6 Swagger

##### 简介

Swagger是一款RESTFUL接口的文档在线自动生成+功能测试功能软件。Swagger是一个规范和完整的框架，用于生成、描述、调用和可视化RESTfu风格的web服务。目标是使客户端和文件系统作为服务器一同样的速度来更新文件的方法，参数和模型紧密集成到服务器。这个解释简单点来讲就是说，swagger是一款可以根据restful风格生成的接口开发文档，并且支持做测试的一款中间软件。

##### 特点

* 不用再手写Wiki接口拼大量参数，避免手写错误。
* 对代码侵入性低，采用全注解的方式，开发简单。
* 方法参数名修改、新增、减少参数都可以直接生效，不用手动维护。
* 联调方便，直接测试接口，实时检查参数和返回值，可以快速定位问题。
* 对于测试来讲，操作简单，不用了解具体代码就可以操作。

##### 框架说明

**Swagger Codegen:** 通过Codegen 可以将描述文件生成html格式和cwiki形式的接口文档，同时也能生成多钟语言的服务端和客户端的代码。支持通过jar包，docker，node等方式在本地化执行生成。也可以在后面的Swagger Editor中在线生成。

**Swagger UI:**提供了一个可视化的UI页面展示描述文件。接口的调用方、测试、项目经理等都可以在该页面中对相关接口进行查阅和做一些简单的接口请求。该项目支持在线导入描述文件和本地部署UI项目。

**Swagger Editor:** 类似于markendown编辑器的编辑Swagger描述文件的编辑器，该编辑支持实时预览描述文件的更新效果。也提供了在线编辑器和本地部署编辑器两种方式。

**Swagger Inspector:** 感觉和postman差不多，是一个可以对接口进行测试的在线版的postman。比在Swagger UI里面做接口请求，会返回更多的信息，也会保存你请求的实际请求参数等数据。

**Swagger Hub：**集成了上面所有项目的各个功能，你可以以项目和版本为单位，将你的描述文件上传到Swagger Hub中。在Swagger Hub中可以完成上面项目的所有工作，需要注册账号，分免费版和收费版。

#### 1.3.1.1.7 Nexus

##### 背景

私服是指私有服务器,是假设在局域网的一种特殊的远程仓库,目的是代理远程仓库及部署第三方构建.有了私服之后,当maven需要下载构件时,直接请求私服,私服上存在则下载到本地仓库;否则,私服请求外部的远程仓库,将构件下载到私服,在提供给本地仓库下载。

##### 简介

nexus是一个强大的仓库管理器,它极大的简化了本地内部仓库的维护和外部仓库的访问，是一套开箱即用的系统不需要数据库,使用文件系统加Lucene来组织数据

nexus使用ExtJS来开发界面,利用Restlet来提供完整的REST APIs,通过IDEA和Eclipse集成使用，支持webDAV与LDAP安全身份认证.

nexus提供了强大的仓库管理功能,构件搜索功能,它基于REST,友好的UI是一个extjs的REST客户端,占用较少的内存,基于简单文件系统而非数据库。

##### 特点

* 加速构建仓库，节省带宽。
* 稳定（应付一旦中央服务器出问题的情况）。
* 控制和审计。
* 能够部署第三方构件。
* 可以建立本地内部仓库。
* 可以建立公共仓库。

#### 1.3.1.1.8 Selenium

##### 背景

Selenium 诞生于 2004 年，当在 ThoughtWorks 工作的 Jason Huggins 在测试一个内部应用时。作为一个聪明的家伙，他意识到相对于每次改动都需要手工进行测试，他的时间应该用得更有价值。他开发了一个可以驱动页面进行交互的 Javascript 库，能让多浏览器自动返回测试结果。那个库最终变成了 Selenium 的核心，它是 Selenium RC（远程控制）和 Selenium IDE 所有功能的基础。Selenium RC 是开拓性的，因为没有其他产品能让你使用自己喜欢的语言来控制浏览器。

Selenium 是一个庞大的工具，所以它也有自己的缺点。由于它使用了基于 Javascript 的自动化引擎，而浏览器对 Javascript 又有很多安全限制，有些事情就难以实现。更糟糕的是，网站应用正变得越来越强大，它们使用了新浏览器提供的各种特性，都使得这些限制让人痛苦不堪。

在 2006 年，一名 Google 的工程师， Simon Stewart 开始基于这个项目进行开发，这个项目被命名为 WebDriver。此时，Google 早已是 Selenium 的重度用户，但是测试工程师们不得不绕过它的限制进行工具。Simon 需要一款能通过浏览器和操作系统的本地方法直接和浏览器进行通话的测试工具，来解决Javascript 环境沙箱的问题。WebDriver 项目的目标就是要解决 Selenium 的痛点。

2008 年，Selenium 和WebDriver 的合并。Selenium 有着丰富的社区和商业支持，但 WebDriver 显然代表着未来的趋势。两者的合并为所有用户提供了一组通用功能，并且借鉴了一些测试自动化领域最闪光的思想。或许，关于两者合并的最好解释，是由 WebDriver 的开发者，在 2009 年 8 月 6 日发出的一封给社区的联合邮件中提到的：一部分是因为 WebDriver 弥补了 Selenium 的一些短处（例如提供了一组很棒的 API，绕开浏览器的限制），一部分是因为 Selenium 弥补了 WebDriver 的一些短处（例如对浏览器更广泛的支持），还有一部分是因为 Selenium 的主要贡献者和我都认为这样能为用户提供最优秀的框架。

##### 简介

Selenium 是一组软件工具集,每一个都有不同的方法来支持测试自动化。大多数使用 Selenium 的QA工程师只关注一两个最能满足他们的项目需求的工具上。然而，学习所有的工具你将有更多选择来解决不同类型的测试自动化问题。这一整套工具具备丰富的测试功能，很好的契合了测试各种类型的网站应用的需要。这些操作非常灵活，有多种选择来定位 UI 元素，同时将预期的测试结果和实际的行为进行比较。Selenium 一个最关键的特性是支持在多浏览器平台上进行测试。

##### 特点

* 支持跨浏览器的自动化测试。
* 支持跨操作系统的自动化测试。
* 支持多中编程语言编写脚本。
* 支持分布式测试分发和管理。
* 拥有一个支持录制的FF插件。

##### Selenium组件

Selenium由三个主要的工具组成。每一个在帮助网络应用测试自动化开发上都有独特的作用。

###### Selenium-IDE

Selenium-IDE是构建Selenium测试用例的集成开发环境。 它是一个Firefox插件，为开发和运行单一测试用例或者完整测试套件提供了易用的界面。 Selenium-IDE具有录制功能，能记录用户执行的操作并保存成可复用的脚本用来回放。 它也有一个上下文菜单（右键单击）集成到Firefox浏览器，允许用户在断言和验证列表中选择。 Selenium-IDE还为了测试用例更加精确和便于控制提供了充分的编辑功能。

###### Selenium-RC (Remote Control)

Selenium-RC允许测试自动化开发人员使用编程语言以最大的灵活性和可扩展性开发测试逻辑。 例如，如果被测程序返回一个结果集，并且如果自动化测试程序需要在结果集的每个元素上运行测试， 编程语言的迭代支持可以用来遍历结果集，在结果集的每一项上调用Selenium命令运行测试。

###### Selenium-Grid

Selenium-Grid允许把Selenium-RC的解决方案应用到大型测试套件或者需要多环境运行的测试套件。 通过Selenium-Grid，多个Selenium-RC实例可以运行在不同的操作系统和浏览器配置中， 在运行的时候每个实例都会注册到一个中心。 当测试到达这个中心，他们会重定向到可用的Selenium-RC，然后启动浏览器运行测试。 这就可以并行的运行测试，理论上整个测试套件的运行时间就是那个运行时间最长的单个测试的时间。

#### 1.3.1.1.9 Easy-Mock

##### 简介

Mock 方法是单元测试中常见的一种技术，它的主要作用是模拟一些在应用中不容易构造或者比较复杂的对象，为测试提供模拟数据,比如你可以模拟HttpServletRequest。从而把测试与测试边界以外的对象隔离开。同时也可以当调用别人的模块，而该模块又没有实现时（只提供接口），我们可以在独立的环境中测试自己的模块逻辑。

EasyMock 是一套用于通过简单的方法对于给定的接口生成 Mock 对象的类库。它提供对接口的模拟，能够通过录制、回放、检查三步来完成大体的测试过程，可以验证方法的调用种类、次数、顺序，可以令 Mock 对象返回指定的值或抛出指定异常。通过 EasyMock，我们可以方便的构造 Mock 对象从而使单元测试顺利进行。

##### 特点

* 支持接口代理
* 支持快捷键操作
* 支持协同编辑
* 支持团队项目
* 支持 RESTful
* 支持 Swagger | OpenAPI Specification (1.2 & 2.0 & 3.0)
* 基于 Swagger 快速创建项目
* 支持显示接口入参与返回值
* 支持显示实体类
* 支持灵活性与扩展性更高的响应式数据开发
* 支持自定义响应配置（例：status/headers/cookies）
* 支持 Mock.js 语法
* 支持 restc 方式的接口预览

#### 1.3.1.1.10 Fiddler

##### 简介

Fiddler是位于客户端和服务器端的HTTP代理，也是目前最常用的http抓包工具之一 。 它能够记录客户端和服务器之间的所有 HTTP请求，可以针对特定的HTTP请求，分析请求数据、设置断点、调试web应用、修改请求的数据，甚至可以修改服务器返回的数据，功能非常强大，是web调试的利器。

##### 使用

Fiddler支持断点调试技术，当你在软件的菜单—rules—automatic breakpoints选项选择before request,或者当这些请求或响应属性能够跟目标的标准相匹配，Fiddler就能够暂停Http通讯，并且允许修改请求和响应。这种功能对于安全测试是非常有用的，当然也可以用来做一般的功能测试，因为所有的代码路径都可以用来演习。

通过显示所有的Http通讯，Fiddler可以轻松地演示哪些用来生成一个页面，通过统计页面（就是Fiddler左边的那个大框）用户可以很轻松地使用多选，来得到一个WEB页面的“总重量”（页面文件以及相关js,css等）你也可以很轻松地看到你请求的某个页面，总共被请求了多少次，以及多少字节被转化了。

用户可以加入一个Inspector插件对象，来使用.net下的任何语言来编写Fiddler扩展。RequestInspectors 和 ResponseInspectors提供一个格式规范的，或者是被指定的（用户自定义）Http请求和响应视图。

通过暴露HTTP头，用户可以看见哪些页面被允许在客户端或者是代理端进行缓存。如果要是一个响应没有包含Cache-Control 头，那么他就不会被缓存在客户端。

## 1.3.2 开发工具

### 1.3.2.1 已使用技术

#### 1.3.2.1.1 VS Code

##### 简介

Microsoft在2015年4月30日Build 开发者大会上正式宣布了 Visual Studio Code 项目：一个运行于 Mac OS X、Windows和 Linux 之上的，针对于编写现代 Web 和云应用的跨平台源代码编辑器。

##### 特点

* 轻量级编辑器
* 丰富的插件系统
* 代码跟踪功能

#### 1.3.2.1.2 IDEA

##### 简介

IDEA 全称 IntelliJ IDEA，是java编程语言开发的集成环境。IntelliJ在业界被公认为最好的java开发工具之一，尤其在智能代码助手、代码自动提示、重构、J2EE支持、各类版本工具(git、svn等)、JUnit、CVS整合、代码分析、 创新的GUI设计等方面的功能可以说是超常的。IDEA是JetBrains公司的产品，这家公司总部位于捷克共和国的首都布拉格，开发人员以严谨著称的东欧程序员为主。它的旗舰版本还支持HTML，CSS，PHP，MySQL，Python等。免费版只支持Python等少数语言。

##### 特点

* 智能的选取
* 丰富的导航模式
* 历史记录功能
* JUnit的完美支持
* 对重构的优越支持
* 编码辅助
* 灵活的排版功能
* XML的完美支持
* 动态语法检测
* 代码检查
* 对JSP的完全支持
* 智能编辑
* EJB支持
* 列编辑模式
* 预置模板
* 完美的自动代码完成
* 版本控制完美支持
* 不使用代码的检查
* 智能代码
* 正则表达式的查找和替换功能
* JavaDoc预览支持
* 程序员意图支持

#### 1.3.2.1.3 Eclipse

##### 简介

Eclipse 是一个开放源代码的、基于Java的可扩展开发平台。就其本身而言，它只是一个框架和一组服务，用于通过插件组件构建开发环境。幸运的是，Eclipse 附带了一个标准的插件集，包括Java开发工具（Java Development Kit，JDK）。

##### 特点

* 适合JAVA语言开发
* 丰富的插件系统
* 开源

#### 1.3.2.1.4 PyCharm

##### 简介

PyCharm是一种Python IDE，带有一整套可以帮助用户在使用Python语言开发时提高其效率的工具，比如调试、语法高亮、Project管理、代码跳转、智能提示、自动完成、单元测试、版本控制。此外，该IDE提供了一些高级功能，以用于支持Django框架下的专业Web开发。

##### 特点

* 编码协助
* 项目代码导航
* 代码分析
* Python重构
* 支持Django
* 支持Google App引擎
* 集成版本控制
* 图形页面调试器
* 集成的单元测试
* 可自定义&可扩展

#### 1.3.2.1.5 Android Studio

##### 简介

Android Studio 是谷歌推出的一个Android集成开发工具，基于IntelliJ IDEA. 类似 Eclipse ADT，Android Studio 提供了集成的 Android 开发工具用于开发和调试。

##### 主要功能

Android Studio是谷歌推出了新的Android开发环境，开发者可以在编写程序的同时看到自己的应用在不同尺寸屏幕中的样子。

谷歌对开发者控制台进行了改进，增加了五个新的功能，包括优化小贴士、应用翻译服务、推荐跟踪、营收曲线图、用版测试和阶段性展示。

#### 1.3.2.1.6 npm

##### 简介

npm全称为Node Package Manager，是一个基于Node.js的包管理器，也是整个Node.js社区最流行、支持的第三方模块最多的包管理器。

npm的初衷：JavaScript开发人员更容易分享和重用代码。

##### 特点

* 使软件包适应您的应用程序
* 下载后可以立即使用的独立工具
* 运行软件包而不使用npx下载
* 与任何npm用户在任何地方共享代码
* 将代码限制为特定的开发人员
* 查找正在解决类似问题的其他开发人员
* 管理多个版本的代码和代码依赖

#### 1.3.2.1.7 Maven

##### 简介

Maven项目对象模型(POM)，可以通过一小段描述信息来管理项目的构建，报告和文档的项目管理工具软件。

Maven 除了以程序构建能力为特色之外，还提供高级项目管理工具。由于 Maven 的缺省构建规则有较高的可重用性，所以常常用两三行 Maven 构建脚本就可以构建简单的项目。由于 Maven 的面向项目的方法，许多 Apache Jakarta 项目发文时使用 Maven，而且公司项目采用 Maven 的比例在持续增长。

Maven这个单词来自于意第绪语（犹太语），意为知识的积累，最初在Jakata Turbine项目中用来简化构建过程。当时有一些项目（有各自Ant build文件），仅有细微的差别，而JAR文件都由CVS来维护。于是希望有一种标准化的方式构建项目，一个清晰的方式定义项目的组成，一个容易的方式发布项目的信息，以及一种简单的方式在多个项目中共享JARs。

##### 特点

* 标准构建文件
* 特性处理顺序
* 构建规则
* 扩展语言
* 构建规则可扩展性

#### 1.3.2.1.8 pip

##### 简介

pip 是一个现代的，通用的 Python 包管理工具。提供了对 Python 包的查找、下载、安装、卸载的功能。

#### 1.3.2.1.9 Conda

##### 简介

Conda 是一个开源的软件包管理系统和环境管理系统，用于安装多个版本的软件包及其依赖关系，并在它们之间轻松切换。

Conda 是为 Python 程序创建的，适用于 Linux，OS X 和W indows，也可以打包和分发其他软件。

目前最流行的 Python 环境管理工具。

## 1.3.3 云原生

### 1.3.3.1 已使用技术

#### 1.3.3.1.1 Kubernetes

##### 简介

kubernetes，简称K8s，是用8代替8个字符“ubernete”而成的缩写。是一个开源的，用于管理云平台中多个主机上的容器化的应用，Kubernetes的目标是让部署容器化的应用简单并且高效（powerful）,Kubernetes提供了应用部署，规划，更新，维护的一种机制。

传统的应用部署方式是通过插件或脚本来安装应用。这样做的缺点是应用的运行、配置、管理、所有生存周期将与当前操作系统绑定，这样做并不利于应用的升级更新/回滚等操作，当然也可以通过创建虚拟机的方式来实现某些功能，但是虚拟机非常重，并不利于可移植性。

新的方式是通过部署容器方式实现，每个容器之间互相隔离，每个容器有自己的文件系统 ，容器之间进程不会相互影响，能区分计算资源。相对于虚拟机，容器能快速部署，由于容器与底层设施、机器文件系统解耦的，所以它能在不同云、不同版本操作系统间进行迁移。

容器占用资源少、部署快，每个应用可以被打包成一个容器镜像，每个应用与容器间成一对一关系也使容器有更大优势，使用容器可以在build或release 的阶段，为应用创建容器镜像，因为每个应用不需要与其余的应用堆栈组合，也不依赖于生产环境基础结构，这使得从研发到测试、生产能提供一致环境。类似地，容器比虚拟机轻量、更“透明”，这更便于监控和管理。

##### 特点

* 可移植: 支持公有云，私有云，混合云，多重云（multi-cloud）
* 可扩展: 模块化，插件化，可挂载，可组合
* 自动化: 自动部署，自动重启，自动复制，自动伸缩/扩展

#### 1.3.3.1.2 Helm

##### 简介

Helm 是 Kubernetes 的包管理器。Helm 由客户端组件 helm 和服务端组件 Tiller 组成, 能够将一组K8S资源打包统一管理, 是查找、共享和使用为Kubernetes构建的软件的最佳方式。

自 2016 年 2 月起，Helm 作为 Kubernetes CNCF 托管项目的子项目存在。该项目由 Deis（于 2017 年被微软收购）发起并与谷歌的 Kubernetes 部署管理器合并成为 Kubernetes Helm。 CNCF 在博文中宣布将 Helm 作为独立项目托管，并表示，“随着Kubernetes 将重点转移到自己的核心，以更好地管理增长，CNCF 是Helm 的理想家园，可以让开发人员和运营商继续更容易地简化Kubernetes 部署”。

从概念上讲，Helm 与 Debian 的 APT、Red Hat 的 Yum 和 MacOS 的 Homebrew 等操作系统的软件包管理器类似，它将文件复制到文件系统适当的位置，适当地设置权限，并配置应用程序应该如何初始化和运行。 Helm 的打包格式（称为 chart ）是一组描述相关 Kubernetes 资源的文件集合。chart 为带有特定目录树中结构的文件，可以被打包成带有版本的压缩包进行部署。

##### 特点

* 统一管理、配置和更新这些分散的 k8s 的应用资源文件
* 分发和复用一套应用模板
* 将应用的一系列资源当做一个软件包管理

#### 1.3.3.1.3 Rancher

##### 简介

Rancher是开源软件，它结合了组织需要采用的所有组件，并在生产环境中运行容器。Rancher基于Kubernetes构建，使DevOps团队可以轻松测试、部署和管理他们的应用程序。运维团队使用Rancher来部署、管理和维护各个Kubernetes集群，而不需关心它运行在何处。

Rancher由以下四个部分组成：

基础设施编。Rancher可以使用任何公有云或者私有云的Linux主机资源。Linux主机可以是虚拟机，也可以是物理机。Rancher仅需要主机有CPU，内存，本地磁盘和网络资源。从Rancher的角度来说，一台云厂商提供的云主机和一台自己的物理机是一样的。Rancher为运行容器化的应用实现了一层灵活的基础设施服务。Rancher的基础设施服务包括网络， 存储， 负载均衡， DNS和安全模块。Rancher的基础设施服务也是通过容器部署的，所以同样Rancher的基础设施服务可以运行在任何Linux主机上。

容器编排与调度。很多用户都会选择使用容器编排调度框架来运行容器化应用。Rancher包含了当前全部主流的编排调度引擎，例如Docker Swarm， Kubernetes， 和Mesos。同一个用户可以创建Swarm或者Kubernetes集群。并且可以使用原生的Swarm或者Kubernetes工具管理应用。除了Swarm，Kubernetes和Mesos之外，Rancher还支持自己的Cattle容器编排调度引擎。Cattle被广泛用于编排Rancher自己的基础设施服务以及用于Swarm集群，Kubernetes集群和Mesos集群的配置，管理与升级。

应用商店。Rancher的用户可以在应用商店里一键部署由多个容器组成的应用。用户可以管理这个部署的应用，并且可以在这个应用有新的可用版本时进行自动化的升级。Rancher提供了一个由Rancher社区维护的应用商店，其中包括了一系列的流行应用。Rancher的用户也可以创建自己的私有应用商店。

企业级权限管理。Rancher支持灵活的插件式的用户认证。支持Active Directory，LDAP， Github等 认证方式。 Rancher支持在环境级别的基于角色的访问控制 (RBAC)，可以通过角色来配置某个用户或者用户组对开发环境或者生产环境的访问权限。

#### 1.3.3.1.4 Prometheus

##### 简介

Prometheus是由SoundCloud开发的开源监控系统的开源版本。2016年，由Google发起的Linux基金会(Cloud Native Computing Foundation,CNCF)将Prometheus纳入其第二大开源项目。Prometheus在开源社区也十分活跃。

Prometheus由Go语言编写而成，采用Pull方式获取监控信息，并提供了多维度的数据模型和灵活的查询接口。Prometheus不仅可以通过静态文件配置监控对象，还支持自动发现机制，能通过Kubernetes、Consl、DNS等多种方式动态获取监控对象。在数据采集方面，借助Go语音的高并发特性，单机Prometheus可以采取数百个节点的监控数据；在数据存储方面，随着本地时序数据库的不断优化，单机Prometheus每秒可以采集一千万个指标，如果需要存储大量的历史监控数据，则还支持远程存储。

##### 特点

* 具有由metric名称和键值对标示的时间序列数据的多位数据模型
* 有一个灵活的查询语言promQL
* 不依赖分布式存储，只和本地磁盘有关
* 通过HTTP的服务拉取时间序列数据
* 支持推送的方式来添加时间序列数据
* 支持通过服务发现和静态配置发现目标
* 多种图形和仪表盘支持

#### 1.3.3.1.5 FluentD

##### 简介

Fluentd是一个用于统一日志层的开源数据收集器。Fluentd允许您统一数据收集和使用，以便更好地使用和理解数据。Fluentd是云端原生计算基金会(CNCF)的成员项目之一，遵循Apache 2 License协议。

##### 特点

* 使用JSON进行统一日志记录：Fluentd尝试尽可能地将数据结构化为JSON：这允许Fluentd 统一处理日志数据的所有方面：收集，过滤，缓冲和跨多个源和目标（统一日志层）输出日志。使用JSON可以更轻松地进行下游数据处理，因为它具有足够的结构，可以在保留灵活模式的同时进行访问。
* 可插拔架构：Fluentd拥有灵活的插件系统，允许社区扩展其功能。我们的500多个社区贡献插件连接了数十个数据源和数据输出。通过利用插件，您可以立即开始更好地使用日志。
* 所需的资源较少： Fluentd是用C语言和Ruby组合编写的，只需要很少的系统资源。vanilla实例运行30-40MB内存，可处理13,000个事件/秒/核心。
* 内置可靠性：Fluentd支持基于内存和文件的缓冲，以防止节点间数据丢失。Fluentd还支持强大的故障转移功能，可以设置为高可用性。

#### 1.3.3.1.6 Istio

##### 简介

Istio 提供一种简单的方式来为已部署的服务建立网络，该网络具有负载均衡、服务间认证、监控等功能，而不需要对服务的代码做任何改动。

##### 特点

* 连接（Connect）：智能控制服务之间的调用流量，能够实现灰度升级、AB 测试和红黑部署等功能
* 安全加固（Secure）：自动为服务之间的调用提供认证、授权和加密。
* 控制（Control）：应用用户定义的 policy，保证资源在消费者中公平分配。
* 观察（Observe）：查看服务运行期间的各种数据，比如日志、监控和 tracing，了解服务的运行情况。

#### 1.3.3.1.7 Docker CE

##### 简介

Docker 是一个开源的应用容器引擎，让开发者可以打包他们的应用以及依赖包到一个可移植的容器中,然后发布到任何流行的Linux机器上,也可以实现虚拟化,容器是完全使用沙箱机制,相互之间不会有任何接口。

##### 特性

在docker的网站上提到了docker的典型场景：

Automating the packaging and deployment of applications（使应用的打包与部署自动化）

Creation of lightweight, private PAAS environments（创建轻量、私密的PAAS环境）

Automated testing and continuous integration/deployment（实现自动化测试和持续的集成/部署）

Deploying and scaling web apps, databases and backend services（部署与扩展webapp、数据库和后台服务）

由于其基于LXC的轻量级虚拟化的特点，docker相比KVM之类最明显的特点就是启动快，资源占用小。因此对于构建隔离的标准化的运行环境，轻量级的PaaS(如dokku), 构建自动化测试和持续集成环境，以及一切可以横向扩展的应用(尤其是需要快速启停来应对峰谷的web应用)。

1. 构建标准化的运行环境，现有的方案大多是在一个baseOS上运行一套puppet/chef，或者一个image文件，其缺点是前者需要base OS许多前提条件，后者几乎不可以修改(因为copy on write 的文件格式在运行时rootfs是read only的)。并且后者文件体积大，环境管理和版本控制本身也是一个问题。
2. PaaS环境是不言而喻的，其设计之初和dotcloud的案例都是将其作为PaaS产品的环境基础
3. 因为其标准化构建方法(buildfile)和良好的REST API，自动化测试和持续集成/部署能够很好的集成进来
4. 因为LXC轻量级的特点，其启动快，而且docker能够只加载每个container变化的部分，这样资源占用小，能够在单机环境下与KVM之类的虚拟化方案相比能够更加快速和占用更少资源

#### 1.3.3.1.8 DockerMachine

##### 简介

Docker Machine是一种工具，可让您在虚拟主机上安装Docker Engine，并使用docker-machine命令管理主机。您可以使用Machine在本地Mac或Windows盒，公司网络，数据中心或Azure，AWS或DigitalOcean等云提供商上创建Docker主机。

使用docker-machine命令，您可以启动，检查，停止和重新启动托管主机，升级Docker客户端和守护程序，以及配置Docker客户端以与您的主机进行通信。

将计算机CLI指向正在运行的托管主机，您可以docker 直接在该主机上运行命令。例如，运行docker-machine env default指向一个名为的主机default，按照屏幕上的说明完成 env安装，然后运行docker ps，docker run hello-world等等。

##### 特点

* Docker Machine是当前docker运行在mac或者windows上的唯一方式。

### 1.3.3.2 未使用技术

#### 1.3.3.2.1 Compose

##### 简介

Compose项目来源于之前的fig项目，使用python语言编写,与docker/swarm配合度很高。

Compose 是 Docker 容器进行编排的工具，定义和运行多容器的应用，可以一条命令启动多个容器，使用Docker Compose不再需要使用shell脚本来启动容器。

Compose 通过一个配置文件来管理多个Docker容器，在配置文件中，所有的容器通过services来定义，然后使用docker-compose脚本来启动，停止和重启应用，和应用中的服务以及所有依赖服务的容器，非常适合组合使用多个容器进行开发的场景。

docker-compose默认的模板文件是 docker-compose.yml，其中定义的每个服务都必须通过 image 指令指定镜像或 build 指令（需要 Dockerfile）来自动构建。

其它大部分指令都跟 docker run 中的类似。

#### 1.3.3.2.2 Docker Swarm

##### 简介

Swarm 是 Docker 官方提供的一款集群管理工具，其主要作用是把若干台 Docker 主机抽象为一个整体，并且通过一个入口统一管理这些 Docker 主机上的各种 Docker 资源。

Swarm 和 Kubernetes 比较类似，但是更加轻，具有的功能也较 kubernetes 更少一些。

Docker Swarm 包含两方面：一个企业级的 Docker 安全集群，以及一个微服务应用编排引擎。

集群方面，Swarm 将一个或多个 Docker 节点组织起来，使得用户能够以集群方式管理它们。

Swarm 默认内置有加密的分布式集群存储（encrypted distributed cluster store）、加密网络（Encrypted Network）、公用TLS（Mutual TLS）、安全集群接入令牌 Secure Cluster Join Token）以及一套简化数字证书管理的 PKI（Public Key Infrastructure）。我们可以自如地添加或删除节点。

编排方面，Swarm 提供了一套丰富的 API 使得部署和管理复杂的微服务应用变得易如反掌。通过将应用定义在声明式配置文件中，就可以使用原生的 Docker 命令完成部署。

此外，甚至还可以执行滚动升级、回滚以及扩缩容操作，同样基于简单的命令即可完成。

以往，Docker Swarm 是一个基于 Docker 引擎之上的独立产品。自 Docker 1.12 版本之后，它已经完全集成在 Docker 引擎中，执行一条命令即可启用。到 2018 年，除了原生 Swarm 应用，它还可以部署和管理 Kubernetes 应用。

#### 1.3.3.2.3 DockerRegistry

##### 简介

Docker私有镜像仓库。

#### 1.3.3.2.4 Kong

##### 简介

Kong 是在客户端和（微）服务间转发API通信的API网关，通过插件扩展功能。Kong 有两个主要组件：

1、Kong Server ：基于nginx的服务器，用来接收 API 请求。

2、Apache Cassandra：用来存储操作数据。

你可以通过增加更多 Kong Server 机器对 Kong 服务进行水平扩展，通过前置的负载均衡器向这些机器分发请求。根据文档描述，两个Cassandra节点就足以支撑绝大多数情况，但如果网络非常拥挤，可以考虑适当增加更多节点。

对于开源社区来说，Kong 中最诱人的一个特性是可以通过插件扩展已有功能，这些插件在 API 请求响应循环的生命周期中被执行。插件使用 Lua 编写，而且Kong还有如下几个基础功能：HTTP 基本认证、密钥认证、CORS（ Cross-origin Resource Sharing，跨域资源共享）、TCP、UDP、文件日志、API 请求限流、请求转发以及 nginx 监控。

Kong可运行在某些 Linux 发行版、Mac OS X 和 Docker 中，无论是本地机还是云端服务器皆可运行。

除了免费的开源版本，Mashape 还提供了付费的企业版，其中包括技术支持、使用培训服务以及 API 分析插件。

## 1.3.4 中间件

### 1.3.4.1 已使用技术

#### 1.3.4.1.1 Nginx

##### 简介

Nginx (engine x) 是一个高性能的HTTP和反向代理web服务器，同时也提供了IMAP/POP3/SMTP服务。Nginx是由伊戈尔·赛索耶夫为俄罗斯访问量第二的Rambler.ru站点（俄文：Рамблер）开发的，第一个公开版本0.1.0发布于2004年10月4日。

其将源代码以类BSD许可证的形式发布，因它的稳定性、丰富的功能集、示例配置文件和低系统资源的消耗而闻名。2011年6月1日，nginx 1.0.4发布。

Nginx是一款轻量级的Web 服务器/反向代理服务器及电子邮件（IMAP/POP3）代理服务器，在BSD-like 协议下发行。其特点是占有内存少，并发能力强，事实上nginx的并发能力确实在同类型的网页服务器中表现较好，中国大陆使用nginx网站用户有：百度、京东、新浪、网易、腾讯、淘宝等。

##### 优点

服务器：

Nginx作为负载均衡服务：Nginx 既可以在内部直接支持 Rails 和 PHP 程序对外进行服务，也可以支持作为 HTTP代理服务对外进行服务。Nginx采用C进行编写，不论是系统资源开销还是CPU使用效率都比 Perlbal 要好很多。

处理静态文件，索引文件以及自动索引;打开文件描述符缓冲。

无缓存的反向代理加速，简单的负载均衡和容错。

FastCGI，简单的负载均衡和容错。

模块化的结构。包括 gzipping, byte ranges, chunked responses,以及 SSI-filter 等 filter。如果由 FastCG或其它代理服务器处理单页中存在的多个 SSI，则这项处理可以并行运行，而不需要相互等待。

支持 SSL 和 TLSSNI。

代码：

Nginx代码完全用C语言从头写成，已经移植到许多体系结构和操作系统，包括：Linux、FreeBSD、Solaris、Mac OS X、AIX以及Microsoft Windows。Nginx有自己的函数库，并且除了zlib、PCRE和OpenSSL之外，标准模块只使用系统C库函数。而且，如果不需要或者考虑到潜在的授权冲突，可以不使用这些第三方库。

代理服务器：

作为邮件代理服务：Nginx 同时也是一个非常优秀的邮件代理服务（最早开发这个产品的目的之一也是作为邮件代理服务器），Last.fm 描述了成功并且美妙的使用经验。

Nginx 是一个安装非常的简单、配置文件非常简洁（还能够支持perl语法）、Bug非常少的服务。Nginx 启动特别容易，并且几乎可以做到7\*24不间断运行，即使运行数个月也不需要重新启动。你还能够不间断服务的情况下进行软件版本的升级。

##### 功能

支持操作系统：

* FreeBSD 3— 10 / i386; FreeBSD 5— 10 / amd64;
* Linux 2.2— 4 / i386; Linux 2.6— 4 / amd64; Linux 3— 4 / armv6l, armv7l, aarch64;
* Solaris 9 / i386, sun4u; Solaris 10 / i386, amd64, sun4v;
* AIX 7.1 / powerpc;
* HP-UX 11.31 / ia64;
* Mac OS X / ppc, i386;
* Windows XP, Windows Server 2003,Windows 10

结构与扩展：

* 最小化的数据拷贝操作。
* 其他HTTP功能。
* 基于IP 和名称的虚拟主机服务。
* Memcached 的 GET 接口。
* 支持 keep-alive 和管道连接。
* 灵活简单的配置。
* 重新配置和在线升级而无须中断客户的工作进程。
* 可定制的访问日志，日志写入缓存，以及快捷的日志回卷。
* 4xx-5xx错误代码重定向。
* 基于 PCRE 的 rewrite 重写模块。
* 基于客户端IP 地址和 HTTP 基本认证的访问控制。
* PUT,DELETE，和 MKCOL 方法。
* 支持 FLV （Flash 视频）。
* 带宽限制。

实验特性：

* 内嵌的 perl。
* 通过 aio\_read()/aio\_write() 的套接字工作的实验模块，仅在 FreeBSD 下。
* 对线程的实验化支持，FreeBSD 4.x 的实现基于 rfork()。
* 英语文档草稿由 Aleksandar Lazic 完成。

HTTP基础功能：

* 处理静态文件，索引文件以及自动索引。
* 反向代理加速（无缓存），简单的负载均衡和容错。
* FastCGI，简单的负载均衡和容错。
* 模块化的结构。过滤器包括gzipping,byte ranges,chunked responses，以及 SSI-filter。在SSI过滤器中，到同一个 proxy 或者 FastCGI 的多个子请求并发处理。
* SSL 和 TLS SNI 支持。
* IMAP/POP3代理服务功能：
* 使用外部 HTTP 认证服务器重定向用户到 IMAP/POP3 后端。
* 使用外部 HTTP 认证服务器认证用户后连接重定向到内部的 SMTP 后端。

#### 1.3.4.1.2 Tomcat

##### 简介

Tomcat是Apache 软件基金会（Apache Software Foundation）的Jakarta 项目中的一个核心项目，由Apache、Sun 和其他一些公司及个人共同开发而成。由于有了Sun 的参与和支持，最新的Servlet 和JSP 规范总是能在Tomcat 中得到体现，Tomcat 5支持最新的Servlet 2.4 和JSP 2.0 规范。因为Tomcat 技术先进、性能稳定，而且免费，因而深受Java 爱好者的喜爱并得到了部分软件开发商的认可，成为目前比较流行的Web 应用服务器。

Tomcat 服务器是一个免费的开放源代码的Web 应用服务器，属于轻量级应用服务器，在中小型系统和并发访问用户不是很多的场合下被普遍使用，是开发和调试JSP 程序的首选。对于一个初学者来说，可以这样认为，当在一台机器上配置好Apache 服务器，可利用它响应HTML（标准通用标记语言下的一个应用）页面的访问请求。实际上Tomcat是Apache 服务器的扩展，但运行时它是独立运行的，所以当你运行tomcat 时，它实际上作为一个与Apache 独立的进程单独运行的。

诀窍是，当配置正确时，Apache 为HTML页面服务，而Tomcat 实际上运行JSP 页面和Servlet。另外，Tomcat和IIS等Web服务器一样，具有处理HTML页面的功能，另外它还是一个Servlet和JSP容器，独立的Servlet容器是Tomcat的默认模式。不过，Tomcat处理静态HTML的能力不如Apache服务器。目前Tomcat最新版本为9.0.27。

##### 版本差异

Apache Tomcat 7.x

是目前的开发焦点。它在汲取了Tomcat 6.0.x优点的基础上，实现了对于Servlet 3.0、JSP 2.2和EL 2.2等特性的支持。除此以外的改进列表如下：

* Web应用内存溢出侦测和预防。
* 增强了管理程序和服务器管理程序的安全性。
* 一般 CSRF保护。
* 支持web应用中的外部内容的直接引用。
* 重构 (connectors, lifecycle)及很多核心代码的全面梳理。

Apache Tomcat 6.x

在汲取 Tomcat 5.5.x优点的基础上，实现了Servlet 2.5和JSP 2.1等特性的支持。除此以外的改进列表如下：

* 内存使用优化。
* 更大的IO容量。
* 重构聚类。

Apache Tomcat 5.x

Apache Tomcat 5.5.x 和Apache Tomcat 5.0.x 对于Servlet和JSP的支持是一样的。大量底层代码里的重大修改，带来性能的提升、稳定性的提升及整体成本。详请参照Apache Tomcat 5.5的更新日志。

Apache Tomcat 5.0.x在Apache Tomcat 4.1的基础上做了很多改动，包括：

性能优化和减少垃圾回收动作

重构程序部署，通过一个可选的独立部署程序，允许在将一个web应用放进产品前验证和编译它

基于JMX的服务器全面监视及web程序管理

提高Taglibs的支撑能力，包括改进的数据池和tag插件

改进平台集成性，包括Windows和Unix

基于JMX的嵌入

增强的安全管理支撑

集成session集群

文档扩充

#### 1.3.4.1.3 Zookeeper

##### 简介

ZooKeeper是一个分布式的，开放源码的分布式应用程序协调服务，是Google的Chubby一个开源的实现，是Hadoop和Hbase的重要组件。它是一个为分布式应用提供一致性服务的软件，提供的功能包括：配置维护、域名服务、分布式同步、组服务等。

ZooKeeper的目标就是封装好复杂易出错的关键服务，将简单易用的接口和性能高效、功能稳定的系统提供给用户。

ZooKeeper包含一个简单的原语集，提供Java和C的接口。

ZooKeeper代码版本中，提供了分布式独享锁、选举、队列的接口，代码在$zookeeper\_home\src\recipes。其中分布锁和队列有Java和C两个版本，选举只有Java版本。

##### 原理

ZooKeeper是以Fast Paxos算法为基础的，Paxos 算法存在活锁的问题，即当有多个proposer交错提交时，有可能互相排斥导致没有一个proposer能提交成功，而Fast Paxos作了一些优化，通过选举产生一个leader (领导者)，只有leader才能提交proposer，具体算法可见Fast Paxos。因此，要想弄懂ZooKeeper首先得对Fast Paxos有所了解。

ZooKeeper的基本运转流程：

1、选举Leader。

2、同步数据。

3、选举Leader过程中算法有很多，但要达到的选举标准是一致的。

4、Leader要具有最高的执行ID，类似root权限。

5、集群中大多数的机器得到响应并接受选出的Leader。

##### 特点

在Zookeeper中，znode是一个跟Unix文件系统路径相似的节点，可以往这个节点存储或获取数据。如果在创建znode时Flag设置为EPHEMERAL，那么当创建这个znode的节点和Zookeeper失去连接后，这个znode将不再存在在Zookeeper里，Zookeeper使用Watcher察觉事件信息。当客户端接收到事件信息，比如连接超时、节点数据改变、子节点改变，可以调用相应的行为来处理数据。Zookeeper的Wiki页面展示了如何使用Zookeeper来处理事件通知，队列，优先队列，锁，共享锁，可撤销的共享锁，两阶段提交。

那么Zookeeper能做什么事情呢，简单的例子：假设我们有20个搜索引擎的服务器(每个负责总索引中的一部分的搜索任务)和一个总服务器(负责向这20个搜索引擎的服务器发出搜索请求并合并结果集)，一个备用的总服务器(负责当总服务器宕机时替换总服务器)，一个web的cgi(向总服务器发出搜索请求)。搜索引擎的服务器中的15个服务器提供搜索服务，5个服务器正在生成索引。这20个搜索引擎的服务器经常要让正在提供搜索服务的服务器停止提供服务开始生成索引，或生成索引的服务器已经把索引生成完成可以提供搜索服务了。使用Zookeeper可以保证总服务器自动感知有多少提供搜索引擎的服务器并向这些服务器发出搜索请求，当总服务器宕机时自动启用备用的总服务器。

### 1.3.4.2 未使用技术

#### 1.3.4.2.1 Kafka

##### 简介

Kafka是由Apache软件基金会开发的一个开源流处理平台，由Scala和Java编写。Kafka是一种高吞吐量的分布式发布订阅消息系统，它可以处理消费者在网站中的所有动作流数据。 这种动作（网页浏览，搜索和其他用户的行动）是在现代网络上的许多社会功能的一个关键因素。 这些数据通常是由于吞吐量的要求而通过处理日志和日志聚合来解决。 对于像Hadoop一样的日志数据和离线分析系统，但又要求实时处理的限制，这是一个可行的解决方案。Kafka的目的是通过Hadoop的并行加载机制来统一线上和离线的消息处理，也是为了通过集群来提供实时的消息。

## 1.3.5 数据库

### 1.3.5.1 已使用技术

#### 1.3.5.1.1 MySQL

##### 简介

MySQL是一个关系型数据库管理系统，由瑞典MySQL AB 公司开发，目前属于 Oracle 旗下产品。MySQL 是最流行的关系型数据库管理系统之一，在 WEB 应用方面，MySQL是最好的 RDBMS (Relational Database Management System，关系数据库管理系统) 应用软件之一。

MySQL是一种关系数据库管理系统，关系数据库将数据保存在不同的表中，而不是将所有数据放在一个大仓库内，这样就增加了速度并提高了灵活性。

MySQL所使用的 SQL 语言是用于访问数据库的最常用标准化语言。MySQL 软件采用了双授权政策，分为社区版和商业版，由于其体积小、速度快、总体拥有成本低，尤其是开放源码这一特点，一般中小型网站的开发都选择 MySQL 作为网站数据库。

由于其社区版的性能卓越，搭配 PHP 和 Apache 可组成良好的开发环境。

##### 特点

* MySQL使用 C和 C++编写，并使用了多种编译器进行测试，保证了源代码的可移植性。
* 支持 AIX、FreeBSD、HP-UX、Linux、Mac OS、NovellNetware、OpenBSD、OS/2 Wrap、Solaris、Windows等多种操作系统。
* 为多种编程语言提供了 API。这些编程语言包括 C、C++、Python、Java、Perl、PHP、Eiffel、Ruby,.NET和 Tcl 等。
* 支持多线程，充分利用 CPU 资源。
* 优化的 SQL查询算法，有效地提高查询速度。
* 既能够作为一个单独的应用程序应用在客户端服务器网络环境中，也能够作为一个库而嵌入到其他的软件中。
* 提供多语言支持，常见的编码如中文的 GB 2312、BIG5，日文的 Shift\_JIS等都可以用作数据表名和数据列名。
* 提供 TCP/IP、ODBC 和 JDBC等多种数据库连接途径。
* 提供用于管理、检查、优化数据库操作的管理工具。
* 支持大型的数据库。可以处理拥有上千万条记录的大型数据库。
* 支持多种存储引擎。
* MySQL 是开源的，所以你不需要支付额外的费用。
* MySQL 使用标准的 SQL数据语言形式。
* MySQL 对 PHP 有很好的支持，PHP是比较流行的 Web 开发语言。
* MySQL是可以定制的，采用了 GPL协议，你可以修改源码来开发自己的 MySQL 系统。
* 在线 DDL/更改功能，数据架构支持动态应用程序和开发人员灵活性（5.6新增）
* 复制全局事务标识，可支持自我修复式集群（5.6新增）
* 复制无崩溃从机，可提高可用性（5.6新增）
* 复制多线程从机，可提高性能（5.6新增）
* 3倍更快的性能（5.7新增）
* 新的优化器（5.7新增）
* 原生JSON支持（5.7新增）
* 多源复制（5.7新增）
* GIS的空间扩展（5.7新增）

#### 1.3.5.1.2 MySQL Galera

##### 简介

Galera-Cluster是集成了Galera插件的MySQL集群，是一种新型的，数据不共享的，高度冗余的高可用方案，目前Galera Cluster有两个版本，分别是Percona Xtradb Cluster及MariaDB Cluster，都是基于Galera的，所以这里都统称为Galera Cluster了，因为Galera本身是具有多主特性的，所以Galera Cluster也就是multi-master的集群架构。

##### 特点

* 同步复制
* Active-active的多主拓扑结构
* 真正的multi-master，即所有节点可以同时读写数据库
* 自动成员资格控制，失败节点从群集中删除
* 新节点加入数据自动复制
* 真正的并行复制行级数据

#### 1.3.5.1.3 Redis

##### 简介

Redis是一个key-value存储系统。和Memcached类似，它支持存储的value类型相对更多，包括string(字符串)、list(链表)、set(集合)、zset(sorted set --有序集合)和hash（哈希类型）。这些数据类型都支持push/pop、add/remove及取交集并集和差集及更丰富的操作，而且这些操作都是原子性的。在此基础上，redis支持各种不同方式的排序。与memcached一样，为了保证效率，数据都是缓存在内存中。区别的是redis会周期性的把更新的数据写入磁盘或者把修改操作写入追加的记录文件，并且在此基础上实现了master-slave(主从)同步。

Redis 是一个高性能的key-value数据库。 redis的出现，很大程度补偿了memcached这类key/value存储的不足，在部 分场合可以对关系数据库起到很好的补充作用。它提供了Java，C/C++，C#，PHP，JavaScript，Perl，Object-C，Python，Ruby，Erlang等客户端，使用很方便。

Redis支持主从同步。数据可以从主服务器向任意数量的从服务器上同步，从服务器可以是关联其他从服务器的主服务器。这使得Redis可执行单层树复制。存盘可以有意无意的对数据进行写操作。由于完全实现了发布/订阅机制，使得从数据库在任何地方同步树时，可订阅一个频道并接收主服务器完整的消息发布记录。同步对读取操作的可扩展性和数据冗余很有帮助。

##### 数据模型

Redis的外围由一个键、值映射的字典构成。与其他非关系型数据库主要不同在于：Redis中值的类型不仅限于字符串，还支持如下抽象数据类型：

* 字符串列表
* 无序不重复的字符串集合
* 有序不重复的字符串集合
* 键、值都为字符串的哈希表

值的类型决定了值本身支持的操作。Redis支持不同无序、有序的列表，无序、有序的集合间的交集、并集等高级服务器端原子操作。

#### 1.3.5.1.4 FastDFS

##### 简介

FastDFS是一个开源的轻量级分布式文件系统，它对文件进行管理，功能包括：文件存储、文件同步、文件访问（文件上传、文件下载）等，解决了大容量存储和负载均衡的问题。特别适合以文件为载体的在线服务，如相册网站、视频网站等等。

FastDFS为互联网量身定制，充分考虑了冗余备份、负载均衡、线性扩容等机制，并注重高可用、高性能等指标，使用FastDFS很容易搭建一套高性能的文件服务器集群提供文件上传、下载等服务。

##### 特点

FastDFS服务端有两个角色：跟踪器（tracker）和存储节点（storage）。跟踪器主要做调度工作，在访问上起负载均衡的作用。

存储节点存储文件，完成文件管理的所有功能：就是这样的存储、同步和提供存取接口，FastDFS同时对文件的metadata进行管理。所谓文件的meta data就是文件的相关属性，以键值对（key value）方式表示，如：width=1024，其中的key为width，value为1024。文件metadata是文件属性列表，可以包含多个键值对。

跟踪器和存储节点都可以由一台或多台服务器构成。跟踪器和存储节点中的服务器均可以随时增加或下线而不会影响线上服务。其中跟踪器中的所有服务器都是对等的，可以根据服务器的压力情况随时增加或减少。

为了支持大容量，存储节点（服务器）采用了分卷（或分组）的组织方式。存储系统由一个或多个卷组成，卷与卷之间的文件是相互独立的，所有卷的文件容量累加就是整个存储系统中的文件容量。一个卷可以由一台或多台存储服务器组成，一个卷下的存储服务器中的文件都是相同的，卷中的多台存储服务器起到了冗余备份和负载均衡的作用。

在卷中增加服务器时，同步已有的文件由系统自动完成，同步完成后，系统自动将新增服务器切换到线上提供服务。

当存储空间不足或即将耗尽时，可以动态添加卷。只需要增加一台或多台服务器，并将它们配置为一个新的卷，这样就扩大了存储系统的容量。

FastDFS中的文件标识分为两个部分：卷名和文件名，二者缺一不可。

#### 1.3.5.1.5 YARN

##### 简介

Apache Hadoop YARN （Yet Another Resource Negotiator，另一种资源协调者）是一种新的 Hadoop 资源管理器，它是一个通用资源管理系统，可为上层应用提供统一的资源管理和调度，它的引入为集群在利用率、资源统一管理和数据共享等方面带来了巨大好处。

##### 优点

大大减小了 JobTracker（也就是现在的 ResourceManager）的资源消耗，并且让监测每一个 Job 子任务 (tasks) 状态的程序分布式化了，更安全、更优美。

在新的 Yarn 中，ApplicationMaster 是一个可变更的部分，用户可以对不同的编程模型写自己的 AppMst，让更多类型的编程模型能够跑在 Hadoop 集群中，可以参考 hadoop Yarn 官方配置模板中的 mapred-site.xml 配置。

对于资源的表示以内存为单位 ( 在目前版本的 Yarn 中，没有考虑 cpu 的占用 )，比之前以剩余 slot 数目更合理。

老的框架中，JobTracker 一个很大的负担就是监控 job 下的 tasks 的运行状况，现在，这个部分就扔给 ApplicationMaster 做了，而 ResourceManager 中有一个模块叫做 ApplicationsMasters( 注意不是 ApplicationMaster)，它是监测 ApplicationMaster 的运行状况，如果出问题，会将其在其他机器上重启。

Container 是 Yarn 为了将来作资源隔离而提出的一个框架。这一点应该借鉴了 Mesos 的工作，目前是一个框架，仅仅提供 java 虚拟机内存的隔离,hadoop 团队的设计思路应该后续能支持更多的资源调度和控制 , 既然资源表示成内存量，那就没有了之前的 map slot/reduce slot 分开造成集群资源闲置的尴尬情况。

##### 主要架构

ResourceManager（RM）：RM是一个全局的资源管理器，负责整个系统的资源管理和分配。它主要由两个组件构成：调度器（Scheduler）和应用程序管理器（Applications Manager，ASM）。

调度器 调度器根据容量、队列等限制条件（如每个队列分配一定的资源，最多执行一定数量的作业等），将系统中的资源分配给各个正在运行的应用程序。需要注意的是，该调度器是一个“纯调度器”，它不再从事任何与具体应用程序相关的工作，比如不负责监控或者跟踪应用的执行状态等，也不负责重新启动因应用执行失败或者硬件故障而产生的失败任务，这些均交由应用程序相关的ApplicationMaster完成。调度器仅根据各个应用程序的资源需求进行资源分配，而资源分配单位用一个抽象概念“资源容器”（Resource Container，简称Container）表示，Container是一个动态资源分配单位，它将内存、CPU、磁盘、网络等资源封装在一起，从而限定每个任务使用的资源量。此外，该调度器是一个可插拔的组件，用户可根据自己的需要设计新的调度器，YARN提供了多种直接可用的调度器，比如Fair Scheduler和Capacity Scheduler等。

应用程序管理器负责管理整个系统中所有应用程序，包括应用程序提交、与调度器协商资源以启动ApplicationMaster、监控ApplicationMaster运行状态并在失败时重新启动它等。

ApplicationMaster（AM）：用户提交的每个应用程序均包含一个AM，主要功能包括：

与RM调度器协商以获取资源（用Container表示）；

将得到的任务进一步分配给内部的任务(资源的二次分配)；

与NM通信以启动/停止任务；

监控所有任务运行状态，并在任务运行失败时重新为任务申请资源以重启任务。

当前YARN自带了两个AM实现，一个是用于演示AM编写方法的实例程序distributedshell，它可以申请一定数目的Container以并行运行一个Shell命令或者Shell脚本；另一个是运行MapReduce应用程序的AM—MRAppMaster。

注：RM只负责监控AM，在AM运行失败时候启动它，RM并不负责AM内部任务的容错，这由AM来完成。

NodeManager（NM）：NM是每个节点上的资源和任务管理器，一方面，它会定时地向RM汇报本节点上的资源使用情况和各个Container的运行状态；另一方面，它接收并处理来自AM的Container启动/停止等各种请求。

Container：Container是YARN中的资源抽象，它封装了某个节点上的多维度资源，如内存、CPU、磁盘、网络等，当AM向RM申请资源时，RM为AM返回的资源便是用Container表示。YARN会为每个任务分配一个Container，且该任务只能使用该Container中描述的资源。

注：1. Container不同于MRv1中的slot，它是一个动态资源划分单位，是根据应用程序的需求动态生成的。

2. 现在YARN仅支持CPU和内存两种资源，且使用了轻量级资源隔离机制Cgroups进行资源隔离。

YARN的资源管理和执行框架都是按主/从范例实现的——Slave ---节点管理器（NM）运行、监控每个节点，并向集群的Master---资源管理器(RM)报告资源的可用性状态，资源管理器最终为系统里所有应用分配资源。

特定应用的执行由ApplicationMaster控制，ApplicationMaster负责将一个应用分割成多个任务，并和资源管理器协调执行所需的资源，资源一旦分配好，ApplicationMaster就和节点管理器一起安排、执行、监控独立的应用任务。

需要说明的是， YARN不同服务组件的通信方式采用了事件驱动的异步并发机制，这样可以简化系统的设计。

#### 1.3.5.1.6 MapReduce

##### 简介

MapReduce是一种编程模型，用于大规模数据集（大于1TB）的并行运算。概念"Map（映射）"和"Reduce（归约）"，是它们的主要思想，都是从函数式编程语言里借来的，还有从矢量编程语言里借来的特性。它极大地方便了编程人员在不会分布式并行编程的情况下，将自己的程序运行在分布式系统上。 当前的软件实现是指定一个Map（映射）函数，用来把一组键值对映射成一组新的键值对，指定并发的Reduce（归约）函数，用来保证所有映射的键值对中的每一个共享相同的键组。

##### 定义

MapReduce是面向大数据并行处理的计算模型、框架和平台，它隐含了以下三层含义：

1）MapReduce是一个基于集群的高性能并行计算平台（Cluster Infrastructure）。它允许用市场上普通的商用服务器构成一个包含数十、数百至数千个节点的分布和并行计算集群。

2）MapReduce是一个并行计算与运行软件框架（Software Framework）。它提供了一个庞大但设计精良的并行计算软件框架，能自动完成计算任务的并行化处理，自动划分计算数据和计算任务，在集群节点上自动分配和执行任务以及收集计算结果，将数据分布存储、数据通信、容错处理等并行计算涉及到的很多系统底层的复杂细节交由系统负责处理，大大减少了软件开发人员的负担。

3）MapReduce是一个并行程序设计模型与方法（Programming Model & Methodology）。它借助于函数式程序设计语言Lisp的设计思想，提供了一种简便的并行程序设计方法，用Map和Reduce两个函数编程实现基本的并行计算任务，提供了抽象的操作和并行编程接口，以简单方便地完成大规模数据的编程和计算处理。

##### 功能

MapReduce提供了以下的主要功能：

1）数据划分和计算任务调度：

系统自动将一个作业（Job）待处理的大数据划分为很多个数据块，每个数据块对应于一个计算任务（Task），并自动 调度计算节点来处理相应的数据块。作业和任务调度功能主要负责分配和调度计算节点（Map节点或Reduce节点），同时负责监控这些节点的执行状态，并 负责Map节点执行的同步控制。

2）数据/代码互定位：

为了减少数据通信，一个基本原则是本地化数据处理，即一个计算节点尽可能处理其本地磁盘上所分布存储的数据，这实现了代码向 数据的迁移；当无法进行这种本地化数据处理时，再寻找其他可用节点并将数据从网络上传送给该节点（数据向代码迁移），但将尽可能从数据所在的本地机架上寻 找可用节点以减少通信延迟。

3）系统优化：

为了减少数据通信开销，中间结果数据进入Reduce节点前会进行一定的合并处理；一个Reduce节点所处理的数据可能会来自多个 Map节点，为了避免Reduce计算阶段发生数据相关性，Map节点输出的中间结果需使用一定的策略进行适当的划分处理，保证相关性数据发送到同一个 Reduce节点；此外，系统还进行一些计算性能优化处理，如对最慢的计算任务采用多备份执行、选最快完成者作为结果。

4）出错检测和恢复：

以低端商用服务器构成的大规模MapReduce计算集群中，节点硬件（主机、磁盘、内存等）出错和软件出错是常态，因此 MapReduce需要能检测并隔离出错节点，并调度分配新的节点接管出错节点的计算任务。同时，系统还将维护数据存储的可靠性，用多备份冗余存储机制提 高数据存储的可靠性，并能及时检测和恢复出错的数据。

#### 1.3.5.1.7 HDFS

##### 简介

Hadoop分布式文件系统(HDFS)被设计成适合运行在通用硬件(commodity hardware)上的分布式文件系统。它和现有的分布式文件系统有很多共同点。但同时，它和其他的分布式文件系统的区别也是很明显的。HDFS是一个高度容错性的系统，适合部署在廉价的机器上。HDFS能提供高吞吐量的数据访问，非常适合大规模数据集上的应用。HDFS放宽了一部分POSIX约束，来实现流式读取文件系统数据的目的。HDFS在最开始是作为Apache Nutch搜索引擎项目的基础架构而开发的。HDFS是Apache Hadoop Core项目的一部分。

HDFS有着高容错性（fault-tolerant）的特点，并且设计用来部署在低廉的（low-cost）硬件上。而且它提供高吞吐量（high throughput）来访问应用程序的数据，适合那些有着超大数据集（large data set）的应用程序。HDFS放宽了（relax）POSIX的要求（requirements）这样可以实现流的形式访问（streaming access）文件系统中的数据。

##### 特点和目标

硬件故障

硬件故障是常态，而不是异常。整个HDFS系统将由数百或数千个存储着文件数据片段的服务器组成。实际上它里面有非常巨大的组成部分，每一个组成部分都很可能出现故障，这就意味着HDFS里的总是有一些部件是失效的，因此，故障的检测和自动快速恢复是HDFS一个很核心的设计目标。

数据访问

运行在HDFS之上的应用程序必须流式地访问它们的数据集，它不是运行在普通文件系统之上的普通程序。HDFS被设计成适合批量处理的，而不是用户交互式的。重点是在数据吞吐量，而不是数据访问的反应时间，POSIX的很多硬性需求对于HDFS应用都是非必须的，去掉POSIX一小部分关键语义可以获得更好的数据吞吐率。

大数据集

运行在HDFS之上的程序有很大量的数据集。典型的HDFS文件大小是GB到TB的级别。所以，HDFS被调整成支持大文件。它应该提供很高的聚合数据带宽，一个集群中支持数百个节点，一个集群中还应该支持千万级别的文件。

简单一致性模型

大部分的HDFS程序对文件操作需要的是一次写多次读取的操作模式。一个文件一旦创建、写入、关闭之后就不需要修改了。这个假定简单化了数据一致的问题，并使高吞吐量的数据访问变得可能。一个Map-Reduce程序或者网络爬虫程序都可以完美地适合这个模型。

移动计算比移动数据更经济

在靠近计算数据所存储的位置来进行计算是最理想的状态，尤其是在数据集特别巨大的时候。这样消除了网络的拥堵，提高了系统的整体吞吐量。一个假定就是迁移计算到离数据更近的位置比将数据移动到程序运行更近的位置要更好。HDFS提供了接口，来让程序将自己移动到离数据存储更近的位置。

异构软硬件平台间的可移植性

HDFS被设计成可以简便地实现平台间的迁移，这将推动需要大数据集的应用更广泛地采用HDFS作为平台。

名字节点和数据节点

HDFS是一个主从结构，一个HDFS集群是由一个名字节点，它是一个管理文件命名空间和调节客户端访问文件的主服务器，当然还有一些数据节点，通常是一个节点一个机器，它来管理对应节点的存储。HDFS对外开放文件命名空间并允许用户数据以文件形式存储。

内部机制是将一个文件分割成一个或多个块，这些块被存储在一组数据节点中。名字节点用来操作文件命名空间的文件或目录操作，如打开，关闭，重命名等等。它同时确定块与数据节点的映射。数据节点负责来自文件系统客户的读写请求。数据节点同时还要执行块的创建，删除，和来自名字节点的块复制指令。

名字节点和数据节点都是运行在普通的机器之上的软件，机器典型的都是GNU/Linux，HDFS是用java编写的，任何支持java的机器都可以运行名字节点或数据节点，利用java语言的超轻便性，很容易将HDFS部署到大范围的机器上。典型的部署是由一个专门的机器来运行名字节点软件，集群中的其他每台机器运行一个数据节点实例。体系结构不排斥在一个机器上运行多个数据节点的实例，但是实际的部署不会有这种情况。

集群中只有一个名字节点极大地简单化了系统的体系结构。名字节点是仲裁者和所有HDFS元数据的仓库，用户的实际数据不经过名字节点。

#### 1.3.5.1.8 Hbase

##### 简介

HBase是一个分布式的、面向列的开源数据库，该技术来源于 Fay Chang 所撰写的Google论文“Bigtable：一个结构化数据的分布式存储系统”。就像Bigtable利用了Google文件系统（File System）所提供的分布式数据存储一样，HBase在Hadoop之上提供了类似于Bigtable的能力。HBase是Apache的Hadoop项目的子项目。HBase不同于一般的关系数据库，它是一个适合于非结构化数据存储的数据库。另一个不同的是HBase基于列的而不是基于行的模式。

##### 特点

* 高可用和海量数据以及很大的瞬间写入量。
* 适用于插入比查询操作更频繁的情况。

#### 1.3.5.1.9 Spark streaming

##### 简介

Hadoop的MapReduce及Spark SQL等只能进行离线计算，无法满足实时性要求较高的业务需求，例如实时推荐，实时网站性能分析等，流式计算可以解决这些问题，spark Streaming就是现在常用的流式计算框架。作为spark的五大核心组件之一，spark Streaming原生地支持多种数据源的接入，而且可以与Spark MLLib、Graphx结合起来使用，具有高吞吐量，容错机制，数据可以从Kafka、flume、Twitter、zeroMQ、K inesis或者TCP的端口，同时能够被类似于使用map、reduce、join和window这种高级函数的算法所处理，最终，被处理过的数据能够被推送到磁盘、数据库。简而言之，Spark Streaming的作用就是实时的将不同的数据源的数据经过处理之后将结果输出到外部文件系统。

#### 1.3.5.1.10 Spark SQL

##### 简介

SparkSQL的前身是Shark，给熟悉RDBMS但又不理解MapReduce的技术人员提供快速上手的工具，但是说到Shark不得不说一说Hive了。Hive是基于hadoop之上的一个开源的工具，他提供的就是类似于SQL的一种HQL语言，它的好处就是可以直接把你的SQL语句转换成mapreduce作业，然后提交在集群上运行，好处就是我们不需要基于mapreduce的api进行编程，你只需要写SQL语句就能完成大数据的统计和分析。

#### 1.3.5.1.11 Elasticsearch

##### 简介

ElasticSearch是一个基于Lucene的搜索服务器。它提供了一个分布式多用户能力的全文搜索引擎，基于RESTful web接口。Elasticsearch是用Java语言开发的，并作为Apache许可条款下的开放源码发布，是一种流行的企业级搜索引擎。ElasticSearch用于云计算中，能够达到实时搜索，稳定，可靠，快速，安装使用方便。官方客户端在Java、.NET（C#）、PHP、Python、Apache Groovy、Ruby和许多其他语言中都是可用的。根据DB-Engines的排名显示，Elasticsearch是最受欢迎的企业搜索引擎，其次是Apache Solr，也是基于Lucene。

ElasticSearch 是一个分布式、高扩展、高实时的搜索与数据分析引擎。它能很方便的使大量数据具有搜索、分析和探索的能力。充分利用ElasticSearch的水平伸缩性，能使数据在生产环境变得更有价值。ElasticSearch 的实现原理主要分为以下几个步骤，首先用户将数据提交到Elastic Search 数据库中，再通过分词控制器去将对应的语句分词，将其权重和分词结果一并存入数据，当用户搜索数据时候，再根据权重将结果排名，打分，再将返回结果呈现给用户。

Elasticsearch是与名为Logstash的数据收集和日志解析引擎以及名为Kibana的分析和可视化平台一起开发的。这三个产品被设计成一个集成解决方案，称为“Elastic Stack”（以前称为“ELK stack”）。

Elasticsearch可以用于搜索各种文档。它提供可扩展的搜索，具有接近实时的搜索，并支持多租户。”Elasticsearch是分布式的，这意味着索引可以被分成分片，每个分片可以有0个或多个副本。每个节点托管一个或多个分片，并充当协调器将操作委托给正确的分片。再平衡和路由是自动完成的。“相关数据通常存储在同一个索引中，该索引由一个或多个主分片和零个或多个复制分片组成。一旦创建了索引，就不能更改主分片的数量。

Elasticsearch使用Lucene，并试图通过JSON和Java API提供其所有特性。它支持facetting和percolating，如果新文档与注册查询匹配，这对于通知非常有用。另一个特性称为“网关”，处理索引的长期持久性；例如，在服务器崩溃的情况下，可以从网关恢复索引。Elasticsearch支持实时GET请求，适合作为NoSQL数据存储，但缺少分布式事务。

#### 1.3.5.1.12 Kibana

##### 简介

Kibana是一个开源的分析和可视化平台，设计用于和Elasticsearch一起工作。你用Kibana来搜索，查看，并和存储在Elasticsearch索引中的数据进行交互。你可以轻松地执行高级数据分析，并且以各种图标、表格和地图的形式可视化数据。Kibana使得理解大量数据变得很容易。它简单的、基于浏览器的界面使你能够快速创建和共享动态仪表板，实时显示Elasticsearch查询的变化。

#### 1.3.5.1.13 Logstash

##### 简介

logstash是一个具有实时管道功能的开源数据收集引擎，Logstash可以动态地将来自不同数据源的数据统一起来，并将数据规范化为你选择的目的地，清理和大众化你的所有数据，用于各种高级下游分析和可视化用例。

虽然Logstash最初推动了日志收集方面的创新，但是它的功能远远超出了这个用例，任何类型的事件都可以通过大量的输入、过滤器和输出插件来丰富和转换，使用许多原生编解码可以进一步简化摄取过程。Logstash通过利用大量和多种数据来提高你的洞察力。

### 1.3.5.2 未使用技术

#### 1.3.5.2.1 MongoDB

##### 简介

MongoDB 是由C++语言编写的，是一个基于分布式文件存储的开源数据库系统。

在高负载的情况下，添加更多的节点，可以保证服务器性能。

MongoDB 旨在为WEB应用提供可扩展的高性能数据存储解决方案。

MongoDB 将数据存储为一个文档，数据结构由键值(key=>value)对组成。MongoDB 文档类似于 JSON 对象。字段值可以包含其他文档，数组及文档数组。

#### 1.3.5.2.2 Neo4j

##### 简介

Neo4j是一个高性能的,NOSQL图形数据库，它将结构化数据存储在网络上而不是表中。它是一个嵌入式的、基于磁盘的、具备完全的事务特性的Java持久化引擎，但是它将结构化数据存储在网络(从数学角度叫做图)上而不是表中。Neo4j也可以被看作是一个高性能的图引擎，该引擎具有成熟数据库的所有特性。程序员工作在一个面向对象的、灵活的网络结构下而不是严格、静态的表中——但是他们可以享受到具备完全的事务特性、企业级的数据库的所有好处。

#### 1.3.5.2.3 Janusgarph

##### 简介

JanusGraph是一个可扩展的图数据库，可以把包含数千亿个顶点和边的图存储在多机集群上。它支持事务，支持数千用户实时、并发访问存储在其中的图。

JanusGraph最大的一个好处就是：可以扩展图数据的处理，能支持实时图遍历和分析查询。

因为JanusGraph是分布式的，可以自由的扩展集群节点的，因此，它可以利用很大的集群，也就可以存储很大的包含数千亿个节点和边的图。由于它又支持实时、数千用户并发遍历图和分析查询图的功能。所以这两个特点是它显著的优势。

#### 1.3.5.2.4 Spark MLib

##### 简介

MLlib是Spark的机器学习（ML）库。其目标是使实用的机器学习可扩展且容易。在较高级别，它提供了以下工具：

ML算法：常见的学习算法，例如分类，回归，聚类和协作过滤

特征化：特征提取，变换，降维和选择

管道：用于构建，评估和调整ML管道的工具

持久性：保存和加载算法，模型和管道

实用程序：线性代数，统计信息，数据处理等