云平台架构概述：

首先应明白建立云平台的目的，与传统的服务器相比，云平台可以将物理资源虚拟化为虚拟机资源池，灵活调用软硬件资源，实现对用户的按需访问。而且在运行过程中根据用户并发量不同，实时迁移虚拟机资源，一方面保证提供高质量服务，另一方面最小化资源成本，提高CPU、内存等利用率。

该架构主要分为4层，从底层到上层分别是资源层、虚拟层、中间件层、应用层。以下从底到上分别说明各层的构造和作用。

资源层：由服务器集群组成。传统服务器要想提供高质量服务，需要性能特别好的服务器（内存高，CPU快，磁盘空间大等），价格昂贵。而服务器集群可以使用以前性能不太好的服务器，利用分布式处理技术，依然可以提供可靠服务，节省费用。

虚拟层：有了物理机集群后，我们需要在物理机上建立虚拟机。建立虚拟机的目的是为了最小化资源成本（最大化资源利用率）。试想一下某台物理机有16G内存，当某段时间连续有小任务量的应用需要处理时，物理机的内存利用率会很低，所以为最大化资源利用率，可以在物理机上独立开辟几个虚拟机，每台虚拟机相当于一个小型服务器，依然可以处理应用请求。我们采用KVM来给每一台虚拟机分配适量的内存、CPU、网络带宽和磁盘，形成虚拟机池。(KVM就是虚拟机监控器hypervisor，可以给虚拟机分配资源，当然也可以开关虚拟机。同样还有XEN和OVM)。

中间件层：这层应该是云平台的核心层，主要功能为：对虚拟机池资源状态进行监测、预警、优化决策。①资源监测：实时监测当前各台虚拟机CPU、内存等使用情况，当然也监测用户应用请求，以便根据应用规模大小进行决策。②预警：防患于未然，根据当前虚拟机资源使用情况预测下一秒用户请求量，以便做出相应资源调整，防止宕机。比如CPU使用率上限为70%，所以当预测下一秒达到该触发点时，应有相应响应。当然，触发阈值应该有更科学的设定。③优化决策：预警之后，虚拟机要进行资源调度(迁移或伸缩)，采用何种调度策略，才能保证服务和资源利用率是研究重点。由于该层需要对应用进行响应处理，所以需要在虚拟机上搭建操作系统，文件存储系统，以及服务器，当然最应该有负载均衡系统Nginx，其实现中间件层功能，相当于网络中的路由器不处理数据，只进行数据转发，数据处理交有虚拟机上的tomcat服务器执行。（也相当于hadoop中的Namenode，其他虚拟机相当于datanode）。

应用层：给用户提供可视化界面，应用若为存储：比如百度云会给用户提供交互界面，建立文件夹，进行数据存储，在线播放视频等界面，供用户选择操作。应用若为租用服务器：界面应该有租用的服务器资源状态。

补充：中间件层是核心，还有一个功能是初始化分配。就是说整个云平台第一次运行，第一次接收用户应用，考虑用户的应用规模很大，虚拟机处理完一个任务后还要继续处理下一个任务，所以要找到一个恰当的分配策略，使得满足高质量服务和资源成本最小。类比于数学建模中的论文—教师匹配问题，论文全部批阅完作为目标，如何给教师分配任务，使得他们工作量、工作时间接近是我们需要考虑的问题。这里面若教师处理速度不同就等价于异构多核处理器任务调度问题，若教师处理速度相同就等价于同构多核处理器任务调度问题。

另外，该平台是面向服务的体系架构(SOA)，给用户的应用提供的是服务。这里面有个概念为组件服务。组件是已经编译好，并且可以使用的二进制代码，组件之间的组合可以组成一个应用程序。比如用户的应用为在线播放视频，那么云平台提供的组件应有在线播放组件及相关联组件，给用户提供的服务就是在线播放视频。而不同的组件组合，需要不同大小的虚拟机资源，所以为使资源利用率最大，需要合理进行组合。

高级版本和虚拟主机

都是提供一个虚拟计算环境的。不同的是普通虚拟主机是一台实体主机上提供多个站点的虚拟环境，云平台是不计其数的实体主机提供不计其数的站点的虚拟环境。

都会提供一种或几种语言的支持。普通虚拟主机有的只支持一种语言，有的则支持多种语言。而云计算平台也是如此。

都会为了安全对支持的语言都做某种程度的限制。普通虚拟主机通常会限制本地文件操作的各种API，而云计算平台则会有更多的限制。

都提供特殊的API服务。既然有些API被限制不能使用，那被限制的API就会有一些安全的替代品来代替。另外，为了将用户绑定于自己的平台，提供一些特别的API服务，也会让用户的站点无法轻易转移。

都会提供数据存储服务。不同的是普通虚拟主机通常提供的是SQL数据存储，而云计算平台则提供更利于分布式计算的NOSQL数据存储。

都会提供特殊的管理平台。虚拟主机通常会提供一些Web管理接口，或者FTP等管理方式来管理。而云计算平台通常会提供特殊的数据上传同步工具。

都是通过计时和计量来收费的。不同的是，云计算平台的计费和计量更加细化，会精确到多少个CPU时间和使用了多少M的存储。

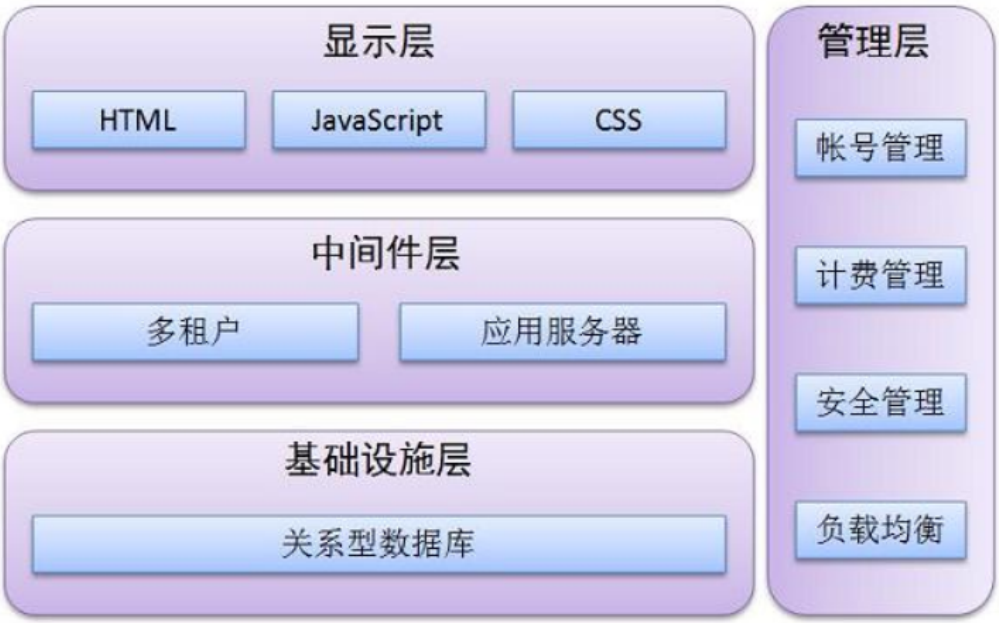
开放性。云平台通常是一个开放的平台，理论上可以满足客户潜在的各种需求，然而，由于云平台是共享的，客户的个性化需求要么通过云平台的服务层（SaaS）来满足，要么客户自己叠加上层功能。譬如，酒店的场景化需求在通用的IoT云平台难以满足，可在酒店管理系统或者单独的应用系统中实现。
操作系统也是开放的，它的抽象程度更高。客户的个性化需求通过上层应用或者额外的系统服务来满足。通用的操作系统加上行业定制的组件（服务或应用）可以形成行业版本的操作系统，譬如从物联网操作系统发展成智慧酒店操作系统。尽管云平台也可以通过SaaS服务来满足智慧酒店的应用需求，但是智慧酒店操作系统是一个更为直接的行业方案。
云平台的开放性在于能通过对一个共享实例的扩展来服务各种客户的需求；操作系统的开放性在于，不同实例可以使用不同的扩展形式和内容，从而服务于该实例所在的场景。

数据保护。对于商业或者工业场景，数据保护是一个关键事项。云平台获取到客户的设备的运行数据，必须尽责保护好这些数据，不得泄漏或者滥用。

部署形式。云平台提供商在云端部署其服务，客户的设备或者应用系统连接到云平台，客户自有的应用系统可以灵活选择部署在本地或者云上。大多数云平台只提供了一个部署地来提供服务，但也有云平台提供多个部署地为不同区域的客户提供服务。例如，西门子的MindSphere云平台分别部署在Amazon的AWS、Microsoft的Azure以及Alibaba的公有云上，服务于各个地区的工业企业。

业务可靠性和性能。在可靠性方面，云平台的优势是有专业的团队来维护，劣势是中间的依赖链路较长，有物理上的不可控因素。另外，云平台自身的逻辑复杂性也潜在地影响客户业务的可靠性。受链路长的影响，性能和延迟是客户在选择云平台来承载其业务时的一个重要考虑因素。

迭代发展。无论是云平台还是操作系统，都存在迭代演进的需求和过程。云平台的迭代相对而言，影响面大，一个功能的迭代有可能影响到很多正在使用的客户，回退机制复杂，要全面考虑数据一致性问题。对于客户而言，涉及到云平台内部实现逻辑的功能调试比较复杂，甚至依赖于云平台的响应。



显示层：基于 HTML、JavaScript 和 CSS 这对黄金组合。
中间件层：在此层，Salesforce 引入了多租户内核和为支撑此内核运行而经过定制的应用服务器。
基础设施层：虽然在后端还是使用在企业环境中很常见的 Oracle 数据库，但是其为了支撑上层的多租户内核做了很多的优化。
管理层：在安全管理方面，Salesforce 提供了多层保护，并支持 SSL 加密等技术，除此之外，其还在帐号管理、计费管理和负载均衡这三方面有不错地支持。

显示层

这层主要是用于以友好的方式展现用户所需的内容，并会利用到下面中间件层提供的多种服务，主要有五种技术：

HTML：标准的 Web 页面技术，现在主要以 HTML4 为主，但是将要推出的 HTML5 会在很多方面推动 Web 页面的发展，比如视频和本地存储等方面。
JavaScript：一种用于 Web 页面的动态语言，通过 JavaScript，能够极大地丰富 Web 页面的功能，最流行的 JS 框架有 jQuery 和 Prototype。
CSS：主要用于控制 Web 页面的外观，而且能使页面的内容与其表现形式之间进行优雅地分离。
Flash：业界最常用的 RIA（Rich Internet Applications）技术，能够在现阶段提供 HTML 等技术所无法提供的基于 Web 的富应用，而且在用户体验方面，非常不错。

Silverlight：来自业界巨擎微软的 RIA 技术，虽然其现在市场占有率稍逊于 Flash，但由于其可以使用 C#来进行编程，所以对开发者非常友好。
在显示层，大多数云计算产品都比较倾向 HTML、JavaScript 和 CSS 这对黄金组合，但是 Flash 和 Silverlight 等 RIA 技术也有一定的用武之地，比如 VMware vCloud 就采用了基于 Flash 的 Flex 技术，而微软的云计算产品肯定会在今后使用到 Silverlight。

中间件层

这层是承上启下的，它在下面的基础设施层所提供资源的基础上提供了多种服务，比如缓存服务和 REST 服务等，而且这些服务即可用于支撑显示层，也可以直接让用户调用，并主要有五种技术：

REST：通过 REST 技术，能够非常方便和优雅地将中间件层所支撑的部分服务提供给调用者。
多租户：就是能让一个单独的应用实例可以为多个组织服务，而且保持良好的隔离性和安全性，并且通过这种技术，能有效地降低应用的购置和维护成本。
并行处理：为了处理海量的数据，需要利用庞大的 X86 集群进行规模巨大的并行处理，Google 的 MapReduce 是这方面的代表之作。
应用服务器：在原有的应用服务器的基础上为云计算做了一定程度的优化，比如用于 Google App Engine 的 Jetty 应用服务器。
分布式缓存：通过分布式缓存技术，不仅能有效地降低对后台服务器的压力，而且还能加快相应的反应速度，最著名的分布式缓存例子莫过于 Memcached。
对于很多 PaaS 平台，比如用于部署 Ruby 应用的 Heroku 云平台，应用服务器和分布式缓存都是必备的，同时 REST 技术也常用于对外的接口，多租户技术则主要用于 SaaS 应用的后台，比如用于支撑 Salesforce 的 Sales Cloud 等应用的 Force.com 多租户内核，而并行处理技术常被作为单独的服务推出，比如 Amazon 的 Elastic MapReduce。

基础设施层

这层作用是为给上面的中间件层或者用户准备其所需的计算和存储等资源，主要有四种技术：

虚拟化：也可以理解它为基础设施层的“多租户”，因为通过虚拟化技术，能够在一个物理服务器上生成多个虚拟机，并且能在这些虚拟机之间能实现全面的隔离，这样

不仅能减低服务器的购置成本，而且还能同时降低服务器的运维成本，成熟的 X86 虚拟化技术有 VMware 的 ESX 和开源的 Xen。
分布式存储：为了承载海量的数据，同时也要保证这些数据的可管理性，所以需要一整套分布式的存储系统，在这方面，Google 的 GFS 是典范之作。
关系型数据库：基本是在原有的关系型数据库的基础上做了扩展和管理等方面的优化，使其在云中更适应。
NoSQL：为了满足一些关系数据库所无法满足的目标，比如支撑海量的数据等，一些公司特地设计一批不是基于关系模型的数据库，比如 Google 的 BigTable 和 Facebook 的 Cassandra 等。
现在大多数的 IaaS 服务都是基于 Xen 的，比如 Amazon 的 EC2 等，但 VMware 也推出了基于 ESX 技术的 vCloud，同时业界也有几个基于关系型数据库的云服务，比如 Amazon 的 RDS（Relational Database Service）和 Windows Azure SDS（SQL Data Services）等。关于分布式存储和 NoSQL，它们已经被广泛用于云平台的后端，比如 Google App Engine 的 Datastore 就是基于 BigTable 和 GFS 这两个技术之上的，而 Amazon 则推出基于 NoSQL 技术的 Simple DB。

管理层

这层是为横向的三层服务的，并给这三层提供多种管理和维护等方面的技术，主要有下面这六个方面：

帐号管理：通过良好的帐号管理技术，能够在安全的条件下方便用户地登录，并方便管理员对帐号的管理。
SLA监控：对各个层次运行的虚拟机，服务和应用等进行性能方面的监控，以使它们都能在满足预先设定的 SLA（Service Level Agreement）的情况下运行。
计费管理：也就是对每个用户所消耗的资源等进行统计，来准确地向用户索取费用。
安全管理：对数据，应用和帐号等 IT 资源采取全面地保护，使其免受犯罪分子和恶意程序的侵害。
负载均衡：通过将流量分发给一个应用或者服务的多个实例来应对突发情况，
运维管理：主要是使运维操作尽可能地专业和自动化，从而降低云计算中心的运维成本。
现在的云计算产品在帐号管理，计费管理和负载均衡这三个方面大都表现地不错，在这方面最突出的例子就是 Amazon 的 EC2，但可惜的是，大多数产品在 SLA 监控，安全管理和运维管理等方面还有所欠缺。