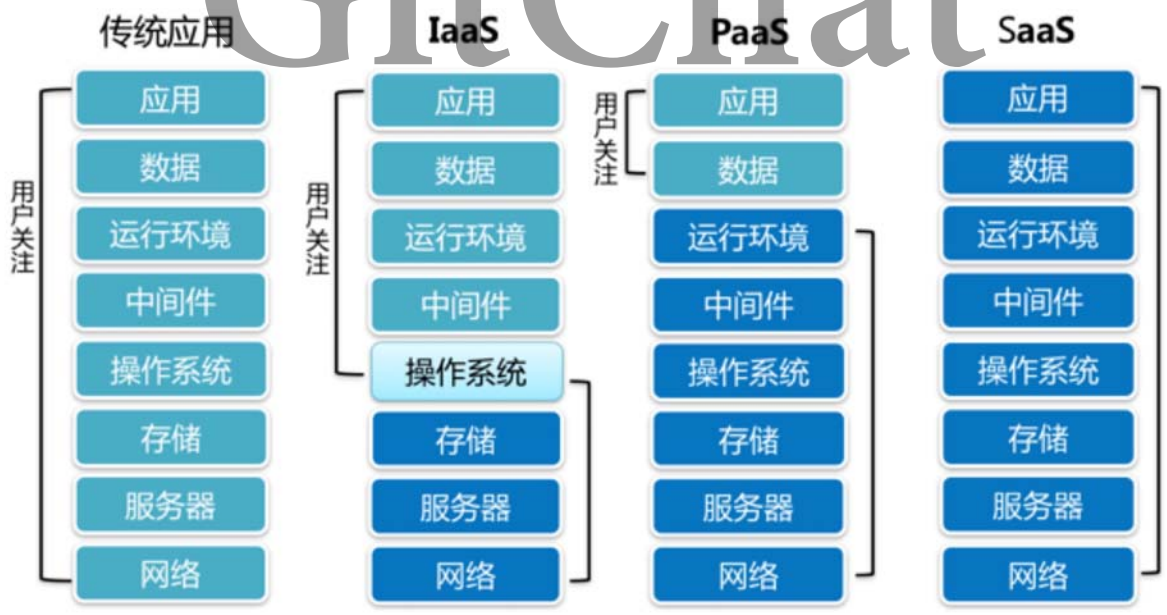
PaaS 平台的架构、现状及未来

说起云计算平台，大家可能都知道有IaaS、PaaS和SaaS。IaaS和SaaS的概念大部分人都能很清晰的认知。说到IaaS大多会讲：存储、计算和网络这三大基础资源，说到SaaS大家会想到各种类型的应用，但是说到PaaS就没有一个非常明确的共识。做大数据平台的厂商会数自己的大数据平台是PaaS，做容器云的厂商会数自己的容器平台是PaaS，甚至传统的IaaS厂商会数自己的平台也是PaaS。那么PaaS究竟是什么呢？

PaaS的定义

云计算相关概念

我们来说PaaS的定义时就要先理解什么是云计算。云计算是指基于互联网等网络，通过虚拟化方式共享IT资源的新型计算模式。其核心思想是通过网络统一管理和调度计算、存储、网络、软件等资源，实现资源整合与配置优化，以服务方式满足不同用户随时获取并扩展、按需使用并付费，最大限度地降低成本等各类需求。



目前云计算提供的服务模式主要包含三大类：基础设施即服务（IaaS）、平台即服务（PaaS）、软件即服务（SaaS）。

基础设施即服务（**IaaS**）

云计算服务商提供虚拟的硬件资源，如虚拟的主机、存储、网络、安全等资源，用户无需购买服务器、网络设备和存储设备，只需通过网络租赁即可搭建自己的应用系统。IaaS定位于底层，向用户提供可快速部署、按需分配、按需付费的高安全与高可靠的计

算能力以及存储能力租用服务，并可为应用提供开放的云基础设施服务接口，用户可以根据业务需求灵活定制租用相应的基础设施资源。在这种服务模式下，用户无需考虑对琐碎的基础设施进行管理与维护，用户可直接在基础设施上面方便地加载应用。**IaaS**服务对应的用户是系统管理员。

平台即服务（**PaaS**）

PaaS提供商提供应用服务引擎，将软件研发测试和运维的平台作为一种服务提供，如应用程序接口（API）服务或应用运行时服务，用户基于这些服务构建业务应用。从用户角度来说，这意味着他们无需自行搭建开发，测试和运维平台，也不会在不同平台兼容性方面遇到困扰。**PaaS**服务对应的用户是应用的开发者和运维人员。

软件即服务（**SaaS**）

用户通过标准的 Web 浏览器来使用网络上的软件。从用户角度来说，这意味着前期无需在服务器或软件许可证授权上进行投资；从供应商角度来看，与常规的软件服务模式相比，维护一个应用软件的成本要相对低廉。SaaS供应商通常是按照客户所租用的软件模块来进行收费的，因此用户可以根据需求按需订购软件应用服务，而且SaaS的供应商会负责系统的部署、升级和维护。**SaaS**提供商对应的用户是应用软件使用的终端用户。

PaaS

平台即服务（PaaS）与基础设施即服务（IaaS）是不同的，PaaS并不是IaaS的一个扩展特性，对于基础设施即服务（IaaS）来说，基础单元就是资源，这里的资源是指服务器，磁盘，网络等，IaaS所做的一切就是按照需要提供这些资源。例如，亚马逊（amazon）的EC2服务，所有的工具都以资源为中心，所有的文档都是关于资源的，所有的开发都是专注于资源，同时人们也因为需要这些资源而使用它。

对于平台即服务（PaaS）来说，基础单元就是应用。那么什么是应用？就是一个系统，就是代码以及所有那些在任何时候都与这些代码通信的服务。这不仅仅是资源，事实上，一个应用是由很多单独的资源绑定在一起组成的。将所有这些资源连接在一起所需要付出的工作量通常被低估了。从一个单一的运行Apache和Mysql的服务器转移到一个拥有单独的负载均衡服务器，缓存服务器，应用服务器，数据库服务器以及冗余的失效恢复的系统架构需要大量的工作，包括前期投入以及后期维护。一个成熟的PaaS平台可以为用户提供上述的所有功能，在减少用户大量工作的前提下，大幅度提升用户应用的开发速度，运行稳定性，可靠性，极大的降低了用户的开发，测试及运维的成本。

利用PaaS可以做的另外一件事，就是从应用的角度来管理IaaS。通常情况下，使用者对于IaaS的资源需求实质上是来源于运行在IaaS之上的应用，如何根据应用的需求动态的使用IaaS资源又成为摆在云使用者面前的一个难题，PaaS作为SaaS与IaaS的沟通者，可以根据SaaS的需求动态的协调IaaS资源，使IaaS按需分配资源的理念变得更智能，更有实际意义。

IaaS为云使用者提供了按需分配的能力，用户可以按照自己的需求定制计算资源，存储资源，网络资源，并且利用云端的海量资源随时快速的开启资源，并在工作完成时，随

时释放资源，在享受云带来的高可靠性的同时，也最大化的降低了使用成本，提升了资源利用率。

但是IaaS为云使用者带来的便利只局限在资源这个层面上，云使用者可以快速，稳定，海量的使用资源，但是一旦获取到资源后，云使用者依然要为运行在资源之上的应用搭建各种适配环境（部署），解决应用的各种依赖，安装应用要使用的各种服务，维护应用的运行生命周期。这些问题IaaS都没有解决，或者说，这些问题本质上也不是IaaS需要解决的问题，而是PaaS需要解决的问题。

综上，IaaS关注与硬件的自动化管理，目标是人与机器的解耦合，提升了效率和性能，而PaaS关注与应用的自动化管理，目标是应用与操作系统的解耦合，提升了弹性和控制。

PaaS的分类

通过上面的介绍，大致可以清晰IaaS、PaaS和SaaS的关系以及面向的用户。在继续介绍PaaS的技术细节前，我们需要了解一下PaaS平台自身的分类。Gartner把它们分为两类，一类是应用部署和运行平台APaaS（Application Platform As a Service），另一类是集成平台IPaaS（Integration Platform As a Service）。

- APaaS是一种面向IT企业和机构的云计算应用开发与部署平台。APaaS主要为应用提供运行环境和数据存储，能够将本地部署的传统应用直接部署到APaaS上。
- IPaaS是用于集成和协同的PaaS平台，不仅可以支持与现有云服务间的连接性，而且可以以安全的方式提供企业应用的访问能力。IPaaS主要用于集成和构建复合应用。

大数据厂商的PaaS实际上是属于IPaaS，而容器厂商和IaaS厂商的PaaS大致为APaaS。

APaaS的一般特性

大规模分布式系统：

- 完全模块化的分布式系统，保证云平台可靠性；
- 每个模块单独存在和运行，通过消息总线进行通讯；
- 系统耦合度低，便于弹性动态扩展；

弹性伸缩框架：

- 平台自身组件支持实时横向扩展;
- 根据应用的负载情况，动态加载应用实例；
- 应用实例支持实时水平扩展；

运维自动化：

- 日常运维操作简化;

- 故障自动恢复;

应用部署简单化：

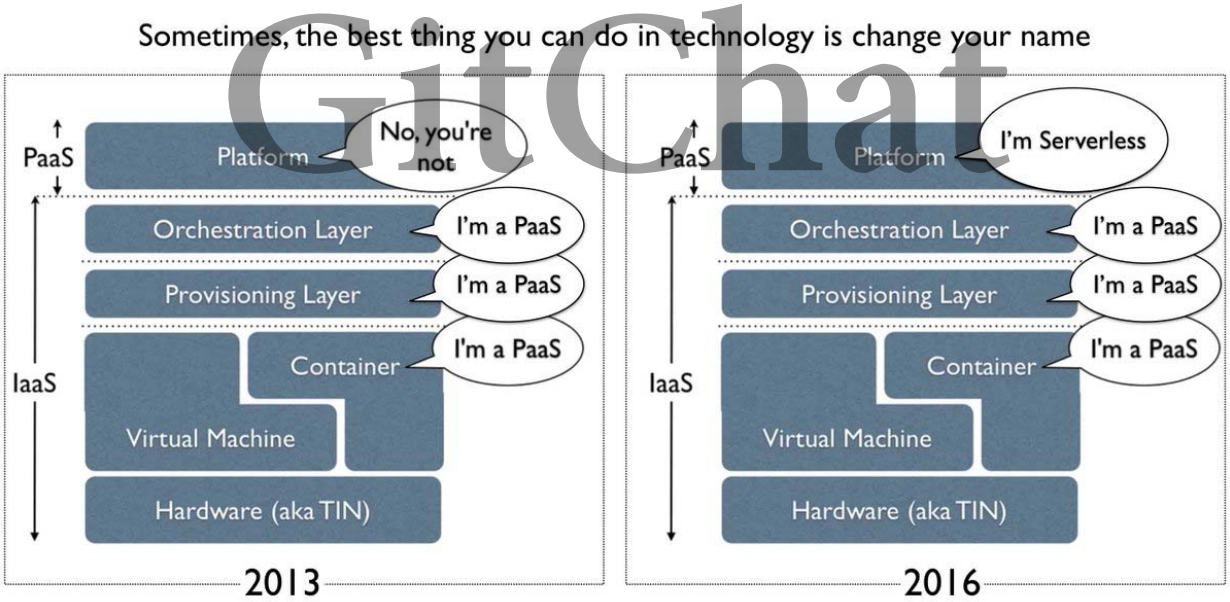
- 一键式应用快速部署;
- 支持多种应用开发框架，包括Spring、.NET、Ruby on Rails，Node.js等;
- 通过buildpack扩展运行不同语言应用的能力;

支持多种服务：

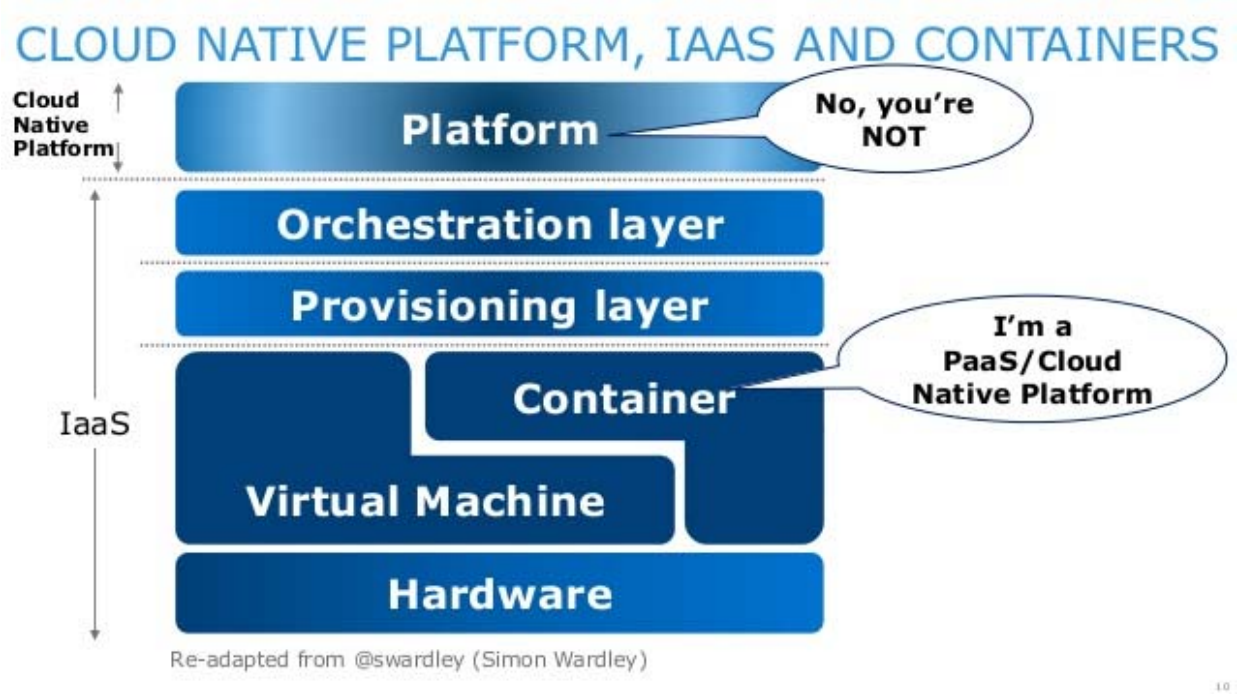
- 支持多种数据服务，包括MySQL、mongodb、PostgreSQL等;
- 通过service broker组件扩展多种应用服务能力，包括数据库、中间件、缓存、云存储等。

主流PaaS平台架构及对比

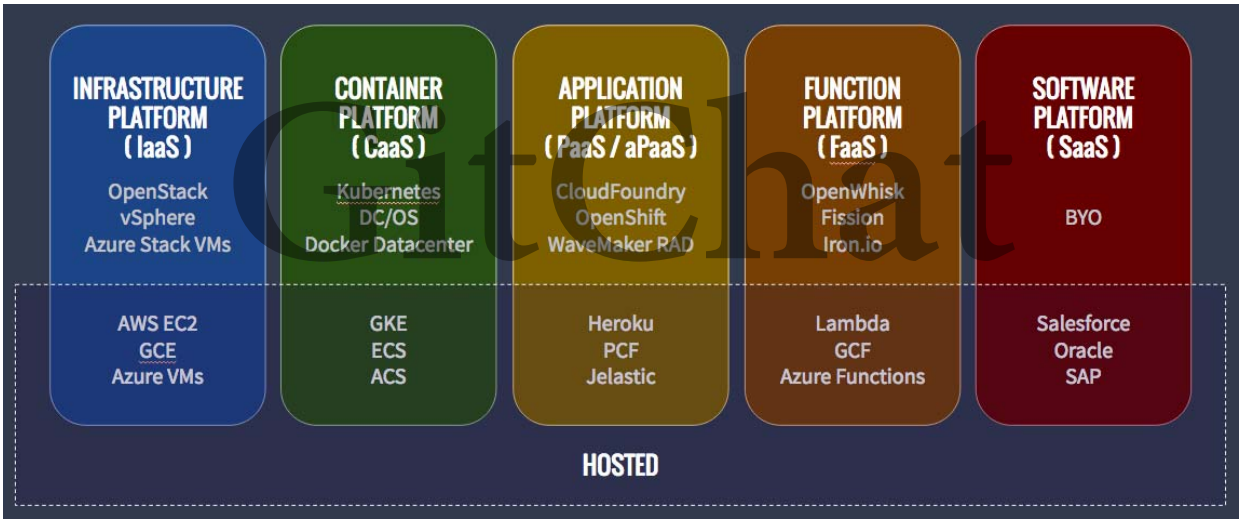
了解了PaaS的分类，我们再来看看PaaS的具体技术对比。由于IPaaS具有很强的业务属性，因此这里我们主要来看一下更通用的APaaS，也是目前被大家最多提起的。说到PaaS，相信很多人都会把他和容器、Docker关联起来。下面来看几张图：



上面这张图可以看出来从容器、编排、部署都自称为PaaS。



正如国内的容器厂商都自称为PaaS平台一样，目前大多数人提到PaaS都会想到容器。每过几年总会有新的概念出来。下面再来看一张图。



这张图很清晰的划分出了XaaS以及各种概念对应的平台。大家所熟知的容器平台在这里实际上是CaaS的一种。那么CaaS和PaaS有什么区别呢？我们接下来对比一下CaaS和PaaS里面最具代表性的两个平台：CaaS和CloudFoundry。

Docker/CaaS

Docker/CaaS 提供直接管理操作基础设施的性能，带来基础设施层面的灵活性。用户 通过直接操作容器可以更灵活的实现应用迁移、部署。但这个更加轻量的平台带来了用户学 习成本和使用复杂度的增加。

容器云平台的搭建只依托 Swarm/Mesos/K8s 等容器编排调度系统就可以实现，还需 要引入大量的第三方解决方案，例如日志、监控、网络等。这就意味着一定的试错成本，另外第三方系统的成熟度发展不一，组成一套统一的云平台后进入生产环境的应用需要经过一 定周期的论证和验证。

Cloud Foundry 平台

Cloud Foundry 隐藏了基础设施层面的复杂度，提供应用层的管理操作，简化基础设 施和应用的构建管理，平台也使用容器技术，但仅仅是平台架构中的实现细节。通过 CF 可以更加敏捷的实现应用开发、部署、业务实现等。

Cloud Foundry 的架构是一个相当完整的 PaaS 架构，模块丰富，平台自身可以提供 对于平台节点、应用的监控、管理，日志，自动化运维等完整的解决方案。每个模块都部署在一个或多个虚拟机上。这些虚拟机是自动创建的，由 PaaS 管理他的生命周期。它的应用 部署跟 Docker 镜像部署不太一样，它是把程序包直接部署。一个命令行或是一个点击就可 以部署

在经过长时间的应用，Cloud Foundry 已有很多在生产环境的案例。Cloud Foundry 率先采用 RunC 作为容器运行时，而且刚刚做了一个 25 万个容器集群的测试，验 证了 PaaS+RunC 的大规模集群的支持。

可以这样来说：容器技术是PaaS平台的底层技术，是PaaS平台不可缺少的部分。但是把容器平台说成PaaS，未免会有点以偏概全了。

特性	**Cloudfoundry**	**Docker / CaaS**
应用部署方式	直接部署程序Release包	需要制作Docker镜像
监控	完整解决方案	需要集成第三方工具
日志	完整解决方案	需要集成第三方工具
网络	完整解决方案	需要集成第三方工具
自动化运维	可以和IaaS联动	不支持IaaS联动

PaaS平台和SaaS应用市场的关系

大家有没有发现一个现象？10年以前，SaaS应用市场是非常少的。现在各大平台都会有自己的SaaS应用市场。出现这种现象的原因无外乎技术的进步，搭建SaaS应用市场的成本降低了。更确切的说：SaaS应用市场是PaaS平台的一种外延。在底层PaaS技术的支撑下，SaaS应用的开发、交付、运营的门槛大幅度的降低了。

基于PaaS平台构建的SaaS应用市场会逐步加快SaaS应用生态的发展。

PaaS平台下进行云原生应用开发

有了PaaS平台帮我们解决底层平台的问题，那么我们的应用开发者要怎么做才能开发出云原生应用呢？12-Factor 为构建如下的云原生应用提供了方法论：

- 使用标准化流程自动配置，从而使新的开发者花费最少的学习成本加入这个项目。
- 和操作系统之间尽可能的划清界限，在各个系统中提供最大的可移植性。
- 适合部署在现代的云计算平台，从而在服务器和系统管理方面节省资源。
- 将开发环境和生产环境的差异降至最低，并使用持续交付实施敏捷开发。
- 可以在工具、架构和开发流程不发生明显变化的前提下实现扩展。

这套理论适用于任意语言和后端服务（数据库、消息队列、缓存等）开发的应用程序。

下面我们一起来看看具体包含哪些要素：

- 基准代码：一份基准代码，多份部署
- 依赖：显式声明依赖关系
- 配置：在环境中存储配置
- 后端服务：把后端服务当作附加资源
- 构建，发布，运行：严格分离构建和运行
- 进程：以一个或多个无状态进程运行应用
- 端口绑定：通过端口绑定提供服务
- 并发：通过进程模型进行扩展
- 易处理：快速启动和优雅终止可最大化健壮性
- 开发环境与线上环境等价：尽可能的保持开发，预发布，线上环境相同
- 日志：把日志当作事件流
- 管理进程：后台管理任务当作一次性进程运行

PaaS的未来发展

先抛一个结论：**PaaS**是云的未来。

IaaS是一个资源转售的生意，PaaS代表了云计算的未来，PaaS解决的是平台架构的问题，同时也是真正做到按需付费。对于业务客户而言，底层技术不会给他们带来直接的业务价值，这也就决定了软件开发商更应该聚焦在业务层。高并发、高可靠、存储服务、应用自身维护等和业务不直接关联的平台服务都可以借助PaaS技术来完成。现在的创业团队可以借助各类PaaS技术快速的创建高并发、高可靠的应用，未来这种模式也会进一步普及。

- 更融合的调度
  物理机、虚拟机和容器各有优势，一个复杂的应用场景会用到各计算平台的优势，融合调度未来必然会成为主流框架。

- 更融合的编排
  说到调度，就离不开编排。当前阶段上云是大趋势，私有云加公有云的模式会长期持续。那么融合的编排框架也必然会成为解决成本问题的一个重要选择。

- 更细粒度的弹性
  IaaS解决了基础设施的弹性，但是还不够。技术的发展会进一步细化弹性的粒度，往更节约的方向发展。

- 更高的资源利用率
  当前的技术及架构下，计算资源很大一部分时间都是闲置的。以有限的资源来应对更高的数据处理要求，资源利用率会随着云技术的发展进一步提高。