今天谈下云平台下的多租户架构，不论是在公有云还是私有云平台，是设计一个面向最终组织或用户的SaaS应用还是面向业务系统的PaaS平台，多租户都是前期架构设计的一个关键内容，因此有必要对里面的一些核心要点进一步说明。

多租户架构概述

首先还是看下百度百科对多租户的一些关键说明如下：

> 多租户技术（英语：multi-tenancy technology）或称多重租赁技术，是一种软件架构技术，它是在探讨与实现如何于多用户的环境下共用相同的系统或程序组件，并且仍可确保各用户间数据的隔离性。多租户简单来说是指一个单独的实例可以为多个组织服务。多租户技术为共用的数据中心内如何以单一系统架构与服务提供多数客户端相同甚至可定制化的服务，并且仍然可以保障客户的数据隔离。一个支持多租户技术的系统需要在设计上对它的数据和配置进行虚拟分区，从而使系统的每个租户或称组织都能够使用一个单独的系统实例，并且每个租户都可以根据自己的需求对租用的系统实例进行个性化配置。

多租户技术可以实现多个租户之间共享系统实例，同时又可以实现租户的系统实例的个性化定制。通过使用多租户技术可以保证系统共性的部分被共享，个性的部分被单独隔离。通过在多个租户之间的资源复用，运营管理维护资源，有效节省开发应用的成本。而且，在租户之间共享应用程序的单个实例，可以实现当应用程序升级时，所有租户都可以同时升级。同时，因为多个租户共享一份系统的核心代码，因此当系统升级时，只需要升级相同的核心代码即可。

这段描述可能理解起来比较啰嗦，我们还是从简单的场景来进行说明。

比如我们开发一个SaaS云服务的CRM系统。这个系统部署在公有云端可以开放给多个企业客户使用。那么我
们就遇到了一个关键问题。
即是否当新入驻一个新的企业客户的时候，我们都需要重新在部署一套应用给这个客户使用？

如果是这样，那么当新客户入驻的时候，将带来具体的人工投入和资源投入成本。

因此实际的情况是我们希望新增加客户的时候，仍然还是已有的那套应用系统。但是对于最终的入驻客户来说，我们又希望客户完全感知不到这点，就像是单独给他们部署了一套系统一样。
虽然很多客户使用同一套应用，但是能够很好地做到资源和数据的隔离。

而这正好就是多租户架构的一个关键点。

多租户，多组织，用户区别

接着谈下一些常见概念的关键区别。

多租户和多组织

实际上在云计算和多租户这些概念出来前，就已经有多组织的概念。

比如常说的类似Oracle ,SAP等ERP系统都是支持多组织架构。多组织架构简单来说就是对于一个大的集团性质企业，企业本身涉及到子公司或分公司，子公司可能涉及到独立法人也可能涉及到需要独立输出财务报表，或者相关公司还在海外涉及到不同的财务和会计准则。

基于以上各种场景持续了多组织架构。

一个多组织架构支撑集团所有的企业都上同一套ERP系统，里面通过法人，财务账簿，OU等设置进行了多组织的支撑。而不是单独为一个子公司再去部署一套独立的应用系统。

从这个概念来看多组织和多租户相当类似。

那么两者的关键区别点在哪里？

简单总结来说多组织架构重点考虑的是数据层面的隔离，但是对于多租户架构更多的还需要考虑资源层面的隔离。多组织架构一般不会考虑类似云平台中的计费和计量管理，数据隔离更多是为了后续财务和数据安全管控要求，而多租户架构则需要考虑计费和计量管理。多组织架构下一般资源全共享，而多租户架构下资源是否共享和资源安全管控要求相关。

租户和用户

租户和用户实际是不同的两个概念，
租户更多的是为了资源管理和计费计量使用，而用户更多的是为了业务功能和授权使用。

租户和用户有时候也是一一对应的关系，比如你开发一个面向个人用户的在线邮箱SaaS应用，那么这个时候租户和用户本身是对应的，租户即用户。

但是如果你开发的是一个面向企业的SaaS应用系统，那么这个时候租户对应的是组织这个层面，即入驻的企业是租户，对应企业入驻后，SaaS应用会先给企业分配一个管理员账号，这个时候管理员再去详细的录入企业里面的具体用户账号。

也就是说租户是第一层，而下面的组织架构和用户是第二层。

SaaS应用和PaaS平台的多租户

注意对于SaaS应用和PaaS平台本身都有多租户的概念。

对于SaaS应用来说，比如一个toB的SaaS应用服务。最终面对的是企业和最终用户，因此每一个入驻的企业组织就是租户。

而对于PaaS平台来说，比如我们在企业内部建设一个公共流程平台，这个流程平台

而这个平台的租
户实际是各个需要使用流程引擎
能力的业务系统。
对于类似容器云PaaS平台，消息，缓存各种PaaS技术服务，都可以看到实际上各个业务系统就是最终的租户。

还是拿上面的例子来说。

如果企业内部的公共流程平台提供给多个业务系统开发商使用，类似用友在该企业本身开发了CRM和SRM两个业务系统。

那么实际管理方式可以是CRM和SRM单独进行租户申请和注册。也可以是两层结构，即还是先进行组织申请，组织作为第一层租户。但是组织接入后还需要维护需

要接入的业务系统，业务系统作为
第二层租户。第一层的组织实际只是一个抽象的租户集的概念。

而实际的资源管理，计量计费等可以细粒度地管理到业务系统这个层级。

多租户架构设计和资源隔离

在多租户和云结合的情况下，IaaS基础资源层的共享已经会变化为最基本的要求。那么在Iaas层之上来谈主要则包括两个方面的内容，即应用是一套还是多套？数据库是一套还是多套？最彻底的多租户即上图中的第6种share everything的模式，在这种模式下数据库和应用都为一套。

多租户我们首先考虑隔离，在多租户下的隔离包括了几个方面的内容。

一个是系统本身元数据和基础主数据的隔离（用户，角色，权限，数据字典，流程模板），一个是系统运行过程中产生的动态数据的隔离，一个是业务系统底层所涉及到的计算资源和存储资源的隔离。

在应用一套，数据库多套或多schema分离情况，我们比较容易实现计算资源和存储资源的单独分配，但是在完全share everthing的情况下，对于计算和存储资源的隔离则需要我们的PaaS应用本身去考虑。

> 在当前云原生和容器下，整个动态部署和持续交付都更加容易，那么为了更好地进行资源隔离，我们完全可以为单独的大租户动态的扩展一套独立的容器集群为该租户服务，即实现该租户能够单独使用一组容器资源池而非共享。

在私有云下的多租户，往往隔离又不是绝对的，在能够完全隔离的情况下又需要支撑跨租户或组织的数据共享，可以看到如果存在这种需求，在Share everthing的情况下是比较容易满足的。

多租户除了隔离外，另外一个重点就是
能够为各个租户按需要
实时地提供各种计算资源和存储资源

，而且有清楚定义的数据采集和计费模型。由于资源池是共享的，我们必须要能够准确地采集到各个租户对实际资源的使用情况，以方便进行多租户的计费。

共享资源时候的资源隔离

当在IaaS云平台的时候，一台物理机可以虚拟化为多台虚拟云主机提供给不同的租户使用，虚拟机可以做到在计算，网络，存储等方面的资源逻辑隔离。也就是说一个租户本身导致的虚拟机使用异常或性能问题，并不会影响到其它租户使用的虚拟机。

到了SaaS层多租户，实际上仍然需要考虑租户下面的资源管理，特别是在多个租户共享一套底层资源的情况下。

> 比如当前有A，B，C，D四个租户在使用SaaS版本的CRM系统，那么我们就需要考虑是不是会出现由于A租户出现的大并发和大数据量访问而导致了剩余的三个租户无法正常使用系统。要做到这点，我们就必须做到面向租户的服务容量控制，服务限流等能力。

多租户下的资源计费

如果是一个IaaS平台的多租户，可以看到对于弹性计算和弹性存储资源都是单独申请的，资源本身也是逻辑隔离，这个时候计费相对简单。

但是对于SaaS应用来说，要做到按资源使用情况计费就比较复杂。因此一般的SaaS应用会简单地根据用户注册数，并发数或存储容量分配来进行组合计费。当然如果是非共享资源模式的多租户架构，相当来说就更加容易按资源使用来进行计费。

多租户下的分域和分组

即使是资源完全共享下的多租户架构，仍然不建议采用一个大集群来为所有租户提供服务，而是应该对大集群进行分域或分组，或者多大的集群资源进行分区或分片处理。让不同的租户分配到不同的集群组或分片上面。

这样做的好处可以避免单个大集群无限扩展导致的性能问题和管理难度，同时也提升了整个应用对外的容错能力，比如A集群全部故障，还可以快速的将A集群流量切换到B集群。

多租户下的数据库扩展

在公有云下的多租户，如果采用完全共享的模式，还必须考虑数据库的可扩展性，多租户架构服务下的数据库可以是独立数据库，共享数据库但是Schema独立，完全共享数据库几种模式。

独立数据库模式为每个租户分配一个独立的数据库，其在SID层就是完全独立的。而对于共享数据库但是Schema独立这种模式下，SID只有一个，但是当入驻新的租户的时候会单独新生成一个独立的Schema。

最后一种模式就是完全共享数据库，SID和Schema都只有一套，在这种模式下核心是
所有数据库表都需要增
加租户ID字段对数据进行多租户隔离
，以保障某一个租户登录系统只能够看到自己租户下的相关信息。如果是一个完整的多租户应用，还需要考虑第二层按用户，组织，角色群组等进行第二级的数据隔离，以满足业务系统的使用需求。

可以看到独立数据库模式资源利用率低，但是数据隔离性最好；而完全共享模式下资源利用率高，但是数据隔离性最弱。因此具体采用哪种模式仍然需要根据实际租户的需求来进行灵活创建和配置，一个灵活的SaaS应用实际需要同时灵活支撑上面三种模式。