**[企业应谨慎对待微服务架构](http://blog.sina.com.cn/s/blog_493a84550102wkbe.html" \o "" \t "_blank)**

微服务架构本质是**单个业务系统彻底的组件化（前端，逻辑层，数据库）解耦，同时相互之间通过轻量的服务接口和协议进行协同**。这和很早就谈到的组件化架构思想是一致的，实现微服务架构后，你会看到没有传统业务系统的概念了，有的只是微服务模块或小应用。  
  
微服务架构最近又炒的相当活，很多人会说SOA过时了，ESB过时了，甚至还有人用微服务架构去彻底的否定SOA和ESB，这些都是相当危险的信号。在我12，13年写企业私有云PaaS平台的一系列文章的时候，已经提出了业务能力组件化，组件服务化的微服务架构思想，但是实际应用实施效果并不太理想。  
  
**一个企业所涉及到的IT开发和架构能力以及企业本身的IT治理管控成熟度都将直接影响到微服务架构能否实施成功，要知道引入微服务架构后集成和后续运维等的复杂度都会成指数级增长。**  
  
简单举例来说，一个企业已经实施了5个业务系统，业务系统之间有10个接口。如果全部微服务化则可能设计到50个微服务模块，上100个接口和集成点。可想而知，在彻底实施微服务后，我们前期架构设计，后期集成和管控的复杂度增加10倍以上。  
  
这种集成难度会远超大多数人想象，如果拿真实做的项目来说，如果谈业务系统只有3个，而到微服务模块级别则有接近60个，而实际涉及到的集成接口上1000个。我们做任何一个复杂端到端业务的联调基本就需要花2，3周甚至更长的时间。  
  
互联网企业为何适合做微服务架构，其重要的一个原因就是互联网企业如电商平台，在进行了微服务化后各个模块之间耦合性很低，并不会有太多的集成和协同点。或者简单来说，**各个微服务模块更多的是向上面的PC端或APP端提供服务能力，而模块横向间接口协同很少。**  
  
正是在这种低耦合情况下，协同和集成相对来说容易。而转回到企业内部你会发现，在微服务模块化后，各个模块之间的集成点相当多，特别是业务系统拆分到模块或组件这一个级别后，很多原有内部的集成和依赖点全部暴露出来了，你都需要去很好的管理。由于这里面有大量的横向集成和协同点，因此导致的就是**微服务模块间的横向集成异常复杂，远超很多互联网应用，这是实际你会面临的问题。**  
  
**企业信息化管理或业务系统的建设，根据企业自身IT管控能力和成熟度的发展必须要能够容忍灰度**。就像一个企业招标建设了3个业务系统一样，刚开始一定关心的是三个系统能够运转起来支撑业务，同时三个业务间数据能够集成。你期望3个系统都安装统一的标准，框架，开发语言和组件化架构进行设计和交付，但是实际上却不可能，各个厂家标准和方法都不一样，厂家宣称的组件化设计而实际上内部模块仍然是强耦合在一起无法拆分。这些都是现实情况，需要容忍，即业务系统只要功能正常能和外面集成起来，其内部混乱一点我不太关心。  
  
这和管理有时候是一样的道理，我们需要抓目标和过程，但是团队一大最上层领导就不可能管理到每个人。他更加关心的是下达给各个部门经理的目标能否很好完成，部门经理间协同上是否有问题。而各个部门经理管理的方法和流程容忍有差异，当企业的管理和治理水平不断成熟后我们再考虑如何更加精细化管理。  
  
**分而治之，一个事物能够分解到哪层？适合分解到哪层？是和你个人的能力和成熟度密切相关的。**  
  
一开始分解的粗，比如只分解到业务系统层面，刚开始就好管控，但是容忍了业务系统内部的一些杂乱无序。那么带来的问题就是当业务系统出现细粒度管控要求和能力扩展要求的时候，你会发现业务系统再要拆分为松耦合的组件并作到彻底的独立管理已经相当困难。这也是为何现在在企业内部也提微服务架构的原因。  
  
企业IT成熟度不够而推行微服务架构，带来诸多的问题点，在下面一篇将详细分析常见问题。第3篇则讲企业微服务架构最好的切入点在哪里？

[**企业应谨慎对待微服务架构**](http://blog.sina.com.cn/s/blog_493a84550102wkbe.html)**(2)**

下面实际谈谈引入微服务架构的难点，以下谈到的都是企业引入和实施微服务架构可能遇到的困难和阻力点，而实际实施难度可能远高于我下面的描述。  
  
**引入的软件开发商本身的水平和意愿**  
  
如果一个企业本身IT部门规模小，软件以外购为主，那么势必在对ERP等各类软件的选型评估后引入不同的软件产品提供商或软件开发商。那么软件商本身都有了成熟的产品或架构，其产品内部的模块是否符合组件化和微服务架构的要求，我们不得而知。  
  
即使招标要求写明软件提供商提供产品需要基于SOA或微服务参考架构，但是实际上由于企业本身的IT能力和水平往往也无法验证，而**对于软件厂商来说一定希望是卖现有产品，减少改造和定制实现利润的最大化。**  
  
对于软件开发厂商来说对已有的软件产品是没有微服务架构改造的动力的。那在这种情况下要推动微服务架构实施落地必须的就是**企业本身有很强的架构管控能力和甲方话语权。**  
  
在曾经实施的案例里面可以看到，甲方在有较强的IT规划和架构设计能力情况下，才可能一开始就划分好微服务模块并且设计好微服务模块间的接口，在进行招标和选型。同时甲方话语权强的情况下，可以完全要求软件供应商按照自己定义好的标准，规范，架构进行微服务模块的开发。  
  
简单点来说顶层架构分解和接口设计能力不在单个微服务模块开发商手里面，而是在甲方手里，或者在甲方请的专门负责规划架构设计的技术咨询团队手里。  
  
在这种模式下，**技术咨询团队应该对整体模块划分和后续集成负责**，技术咨询团队就需要有业务和技术两方面的能力，同时有类似领域的规划设计经验，系统开发建设经验等。这些本身就对技术咨询团队提出了相当高的要求，可以来讲很少有技术咨询团队达到这个水平，包括埃森哲或德勤等也难。  
  
在微服务架构下，我们希望的是一个业务系统如果由三个微服务模块组成，在我们进行了前期的架构和接口设计后，我们完全可以将三个模块发标给不同的软件开发商建设和实施，然后在根据预先定义的服务接口进行集成。这个从理论上是行得通的，但是实际上出现两个问题。  
  
**其一是刚开始的模块划分或接口设计不合理，在后面开发过程中才发现又很难再大变更。  
其二是微服务模块间的接口服务太多，导致了模块间的集成和联调异常复杂。**  
  
从上面也看到引入微服务架构后，企业本身可以削弱单个软件供应商对企业本身的约束，防止被单一厂商绑定。因此企业没有特色要求，从软件厂商来说没有任何动力和意愿推微服务架构。  
  
**企业自由开发团队实践微服务架构**  
  
如果企业本身的IT成熟度没有达到一定阶段，显然是不可能推行实施微服务架构的。这个道理前面已经谈到过，在企业IT建设中，**如果连粗粒度的业务系统以及它们之间的集成都管理不好，那么更没有能力管理细粒度的微服务模块**。那么如果企业IT成熟度达到一定水平，在推广微服务架构还存在的难点如下：  
  
**首先是架构设计能力的显性化**，即架构设计这个工作的输入，输出和过程需要更加的显性化出来形成团队都认同的标准工件。一个业务系统没有拆分开时候，虽然有架构设计和组件划分，但是这个工作是属于团队内部的事情，即使架构设计不合理，在后期集成也可以通过诸多变通方式解决掉。而现在是不同的微服务模块可能分派到两个独立的团队开发，原来属于自己内部黑盒的问题变为团队间问题。  
  
简单来说你原来藏着或没做规范的东西太多，而现在这些不能再藏着掖着了，当真要把这些东西拿出来的时候，你才会发现你原来架构能力是有欠缺的。正如我们理解了一个东西，那么要让我们清楚的讲出来困难，那么我们的理解有欠缺。对于我们能讲清楚的东西，要系统的写下来有困难，那么说明我们讲的结构和条理有欠缺。  
  
**其次管控要求和规范体系的建立**，对于管控要求可以看到如果两个微服务模块分给同一个团队开发，如何才能保证开发的团队保持两个模块的完全独立和解耦，两个模块间不会出现相互交叉的数据库直接调用，也不会存在直接绕开Service接口的其它耦合调用？这些如果没有完整的管控和检查体系我们很难约束。  
  
**微服务架构下导致的开发复杂度增加**  
  
只谈微服务架构的松耦合和高扩展性，而不谈开发和集成复杂度的都是耍流氓。  
  
实际上当前很多企业对微服务架构并没有如此迫切，互联网很多企业推行微服务架构更多的还是考虑到巨大的业务并发量下的系统弹性扩展能力，而实际大多数企业内应用往往并没有如此海量并发。其次，即使在并发量增加的情况下通过进行代码本身的优化，数据库调优或者升级硬件服务器资源都可以较好的解决性能问题。而做这些事情投入的成本远远小于微服务架构带来的开发复杂度增加成本，后期的运维管控成本。  
  
要做到完全微服务模块独立，**微服务架构下最大的一个变化就是数据库也拆分开了**，原来的一个业务系统如果分为5个微服务模块，那理论上就是5个独立的后台数据库，而且数据库间还不能随便相互连接和访问。只有这样微服务模块才能做到独立部署和管理。  
  
由于数据库拆分带来两个问题，其一是我们原来很**简单的一个跨表查询操作现在无法做了**，我们必须调用两个微服务模块提供的服务，查询到数据后再到逻辑层进行组合。其次最大的问题就是如果一个业务操作需要同时更新两个微服务模块的数据，由于服务本身无状态，导致了这种**分布式事务问题很难解决**。  
 **企业内业务系统很大一个特点就是业务逻辑和规则相对互联网更加复杂，而且有更高的事务一致性要求**。正是由于这个原因，无法解决好分布式事务的问题都将直接导致后续数据不一致和业务错误。  
  
原来通过调用项目内一个API方法就能解决的问题，现在要调用远程WS接口才能解决，这本身就增加了开发和调试的复杂度。一个微服务模块与外部其它模块的集成和协同越少，你会发现该微服务模块和传统业务系统开发没太大区别，但是当其涉及到完成任何一个功能都需要调用外部微服务模块的服务接口时候，其开发模式和效率上就会带来巨大的变化。  
  
**微服务架构下导致的集成复杂度增加**  
  
任何一个微服务模块在外部协同上都存在两个点，一个是模块本身要消费和调用其它微服务模块提供的服务接口，另外一个是模块本身又需要将其业务能力暴露为服务接口给其它微服务模块使用。  
  
如果一个微服务模块要同时支撑PC端和APP端，可以看到微服务模块暴露的服务还需要统一提供给前端的展示用。那么可以看到一个微服务模块需要完成自身组件层和展现层间的集成，同时又需要完成多个微服务模块组件间的横向服务集成。  
  
如果我们将消息，日志，流程，4A等能力下层到平台层微服务模块，那么一个组件要跑起来还涉及到和平台层的多个技术类微服务模块集成。在如此复杂的集成场景下，**要将复杂的跨多个微服务模块的横向端到端业务跑通，其涉及到的模块数，接口数都远超原有单一系统的模式**。  
  
一个业务系统如果没有拆分为微服务模块，那么其它内的模块间集成和集成测试是系统内部的事情，但是一旦拆分为多个微服务模块，那么这种集成就变成外部第三方的事情，或者必须要显性的事情。  
  
对于一个微服务模块来说，当其需要集成的外部微服务模块和接口都变多的时候带来什么问题呢？这个问题大家容易理解，即**该模块究竟是否稳定已经不是模块本身的事情了，而是涉及到诸多外部依赖模块是否稳定**。更简单说本来原来我自己可以确认稳定的事情，现在我无法确认了。即使每个模块的稳定度都90%，但是你会发现一集成起来90%\*90%\*90%，那么稳定性就下降的很厉害。正是由于这个原因，我们要求在一个大体系里面，每个微服务模块的开发质量都要得到保证，这已经不是单个模块自己的事情，而是直接影响到大系统的质量。  
  
**是否要实施和Docker集成的问题**  
  
实际上，一个企业开始实施微服务架构，并不一定马上要和docker集成，毕竟企业内部的的业务系统并发量和性能拓展不需要做到完全的自动化。先把组件化和服务化做好，确保各个微服务模块是可拆分的，等真正实施上去了再来考虑如何自动化部署和动态扩展的问题。  
  
**微服务架构下的运维问题**  
  
在实施了微服务架构后，运维的复杂度也是成倍增加，任何一个微服务模块出问题都可能影响到整个业务应用的功能使用。我们在运维时候不仅仅要健康单个微服务模块，还需要健康所有的接口服务监控状态。  
  
如果跟Docker集成了，我们看到整个性能监控和问题分析都会变麻烦了，没有实施微服务架构前发现问题，我们直接可以看应用服务器上类似tomcat或jboss日志，而实施了微服务架构后，应用容器已经是自动部署和动态分配的，原有的故障诊断模式行不通，而需要PaaS平台本身提供完整的预警和日志分析能力。  
  
再次，如果发现了性能问题或故障，我们的解决方案是如何的？我们如何保证不影响到业务运行，不出现数据的丢失，或者在微服务模块扩展的时候不出现业务中断等。这些已经不是简单的部署架构层面的冗余能解决的问题，而涉及到我们在整个微服务架构中的消息策略，事务管理机制，持久化机制等问题。

[**企业微服务架构切入点**](http://blog.sina.com.cn/s/blog_493a84550102wkeu.html)

前面两篇文章我讲解了企业在自身IT成熟度还没有达到一定水平的时候，应该谨慎对待微服务架构，其核心原因就是**由于架构微服务化后会导致开发，集成，乃至后期的运维管控的复杂度呈指数级提升。即使企业本身有组件化和服务化的思想，但是也没有能够彻底构建微服务架构的能力。**  
  
正如很多企业连基础主数据都没有管理，也没有建设集成的研发，生产相关的PLM，MES，CIM等核心系统，就开始谈要一步迈入工业4.0和智能制造是一样的道理。任何事情都要考虑从简单到复杂，通过迭代的方式逐步演进。下面就简单分析下企业实施微服务架构可行的一些切入点。  
  
**1. 共性技术服务能力下沉建设**  
  
企业在刚开始规划建设，或者建设到一定阶段后，都会涉及到一下基础性的共性技术需求，类似消息管理，日志管理，文件存储，共性的小应用组件（论坛，通讯录，文档在线阅读）等。  
  
这些共性能力既可以是技术服务，也可以是共性小应用程序，其最大的特点就是这些组件本身横向交互相当少，而更多的是将自己的能力向上提供暴露和集成。因此这类场景采用微服务架构方式来独立构建并部署是合适的，这类模块的上线和集成可以最大限度的减少对已有横向业务的影响。  
  
要发现这类需求，企业应该有一个统一的需求受理和分析组，对各个业务部门或业务系统提交的需求同意进行分析，抽取出共性需求，然后再考虑是否通过微服务方式统一建设。  
  
**2. 基础平台层能力先行**  
  
企业在实施微服务架构的时候，一定要意识到对于4A+流程引擎这两个能力需要提取进行平台化和微服务模块化。因为这两个基础能力往往是任何一个业务微服务模块能够运转起来的基础。正是由于这两个基础能力的平台化，我们在构建新的微服务模块的时候，才能够将重心完全放在只关注业务实现上。  
  
**3. 新增模块移出**  
  
如果企业已经实施了采购系统而且已经上线运行多年，那么在对采购系统出现大的模块级需求的时候（例如需求在采购需求中增加招投标的功能），那么这种模块需求就可以考虑移出采购系统，通过微服务架构的方式独立构建，在构建完成后在和采购管理系统集成。  
  
对于一个新增模块是否能移出，重点还是要考虑该模块和已有的遗留业务系统间的耦合性和集成度。耦合度越小，越容易单独构建并后期集成。从这个角度来看对于哪些在原有业务系统中上游业务最适合移出，例如招投标模块构建只是需要将合格供应商和采购物料清单信息传递到采购系统，而并不需要从已有的采购系统返回任何信息。  
  
新增模块移出并进行微服务化往往是对遗留系统影响最小的方案。在微服务架构在企业内部逐步实施成熟后再考虑更多的模块或组件从已有系统中移出。  
  
**4. 大变更下模块移出**  
  
企业在接收到新的变更需求处理时，当已有业务系统的某一个模块出现重大变更时（比如变更内容和范围超过了模块本身30%-50%），在这种情况下可以考虑将变更模块移出并进行微服务架构的改造。  
  
要清楚在模块大变更情况下，即使按原有模块开发和处理，也会带来巨大的模块开发和集成，联调和实施工作量，还还不如和企业微服务架构演进策略一起处理。两次对业务的大影响变成一次影响，虽然增加了复杂度，但是实际上是降低了整体工作量和后期迁移难度。  
  
企业实施微服务架构不应该是将遗留系统彻底推翻并全新建设，而是应该采用3+4迭代进行的渐进式实施策略。