文章
Qiao Peng · 十二月 18, 2021 阅读大约需 12 分钟

精华文章--从软件架构发展谈业务集成技术演进与展望

应用集成技术是市场上被广泛使用的，也是充斥着术语和概念的一个技术领域。集成平台、消息引擎、消息中间件、集成引擎、集成中间件、企业服务总线(ESB)、API网关、API管理…
很多概念与名词。到底它们是什么意思？有什么区别？哪种技术适合解决哪种集成问题？

业务集成的需求和技术的演进是紧随业务系统的软件架构发展而发展的。通过小结软件架构的发展，我们更容易梳理业务集成技术的演进、更容易看清楚各种集成架构的优势和未来发展方向。

软件架构发展简史

大型机架构
：上世纪60-70年代。大型机负责数据存储、业务逻辑处理、数据展现等所有工作；客户端只是一个终端，基本上就是键盘加上显示器，负责输入、输出。大型机架构的优势是简单，但显然可扩展性很差。

在大型机时代，几乎没有集成的需求。

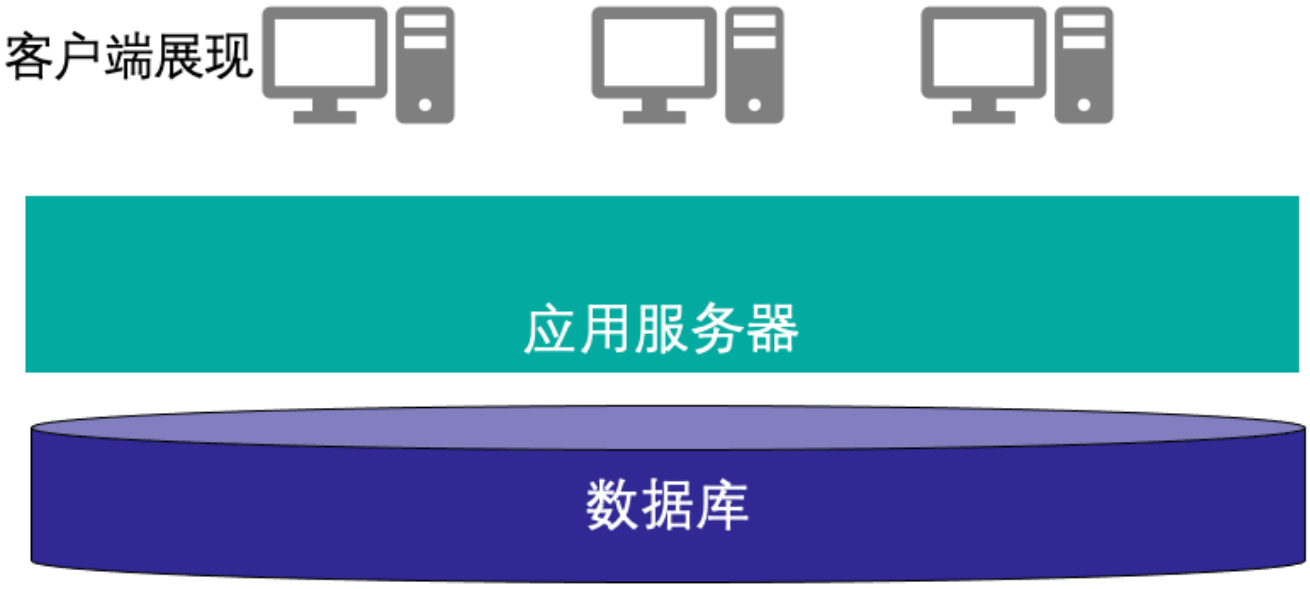
客户端/服务器（C/S）架构

：上世纪80年代到90年代。服务器负责数据存储和处理，以及服务器端端业务逻辑处理；客户端（PC）负责客户端逻辑处理、数据验证、数据展现等工作。

客户端（PC）分担了很多工作，且数量众多，因此它是一个分布式架构，可扩展性提升。但维护客户端应用，例如升级等带来了很大的管理维护成本。现在很多企业核心应用，例如ERP、HIS都是当时在这个架构下开发出来的，基本都是单体架构应用，业务逻辑间耦合度非常高。应用之间需要做业务的共享交换，主要通过点对点方式集成，也催生了集成技术，尤其是以消息交换为基础的、以消息队列技术为载体的集成方式。在这个时期，HL7组织开发了HL7 V2.x的医疗消息交换标准。

三层架构

：上世纪90年代中后期至今。三层架构是展现层（用户界面）、应用层（业务逻辑）和数据层（数据）三层架构。



这里的三层架构不仅指软件功能的层次，而且指软件运行的基础架构的层次。由于在软件功能和基础架构上的分层和分布式设计，三层架构应用通常有很好的可扩展性和可维护性，虽然仍是单体架构应用，但它降低了业务逻辑的耦合度和基础架构层次间的耦合度。另一方面，三层架构让整体复杂程度上升了。分层架构和不断涌现的技术实现，如.net、java、EJB…使这些应用间的互相调用变得越来越复杂了，接口引擎或集成引擎应运而生，并在这个时期发展壮大起来。它们要解决这么多技术平台的连接问题，适配器是主要手段。

面向服务架构（SOA）：上世纪90年代末期至今。前面的所有架构，基本都是单体架构模式 – 也就是孤岛模式，业务耦合度高而难以拆分，代码复用性低。不同应用中有大量功能重复的业务逻辑代码和数据，例如医疗行业的患者管理，几乎每个科室系统都有。这不但造成了需要同步的数据越来越多，更造成了数据互相冲突、运行效率降低、运行成本增加。 代码跨业务、跨语言、跨平台复用成为快速满足业务发展的需求与趋势。以服务的方式封装和复用软件组件，然后可以敏捷地重新组织这些服务快速形成新的应用，SOA带来了软件架构上的革命。

每个服务都封装有数据和逻辑（代码），以提供独立的功能，服务之间高度松耦合。而采用XML作为数据格式、以WSDL标准描述服务、以SOAP协议作为交互方式，面向服务架构使不同技术栈的差异对服务透明，使服务在任何技术平台上都可以使用。

随着SOA架构的逐步采纳，企业环境里有了越来越多的服务，企业服务总线（ESB）发展起来。它是一个中心化的软件组件平台，将企业环境中的服务注册在总线上，并协同这些服务。ESB通常借助消息引擎、业务流程引擎、数据转换引擎，执行服务路由、服务协同和服务间的数据结构转换。

更近一步，先进的ESB还融合了接口引擎的技术，有能力将非SOA架构的应用接口封装为服务并注册到总线，使其可以和其它服务一样被调用和协同。这些恰恰都是集成的目的，因此ESB成为一个重要的集成技术。

HL7 在此期间，也推出了基于参考信息模型（RIM）的HL7

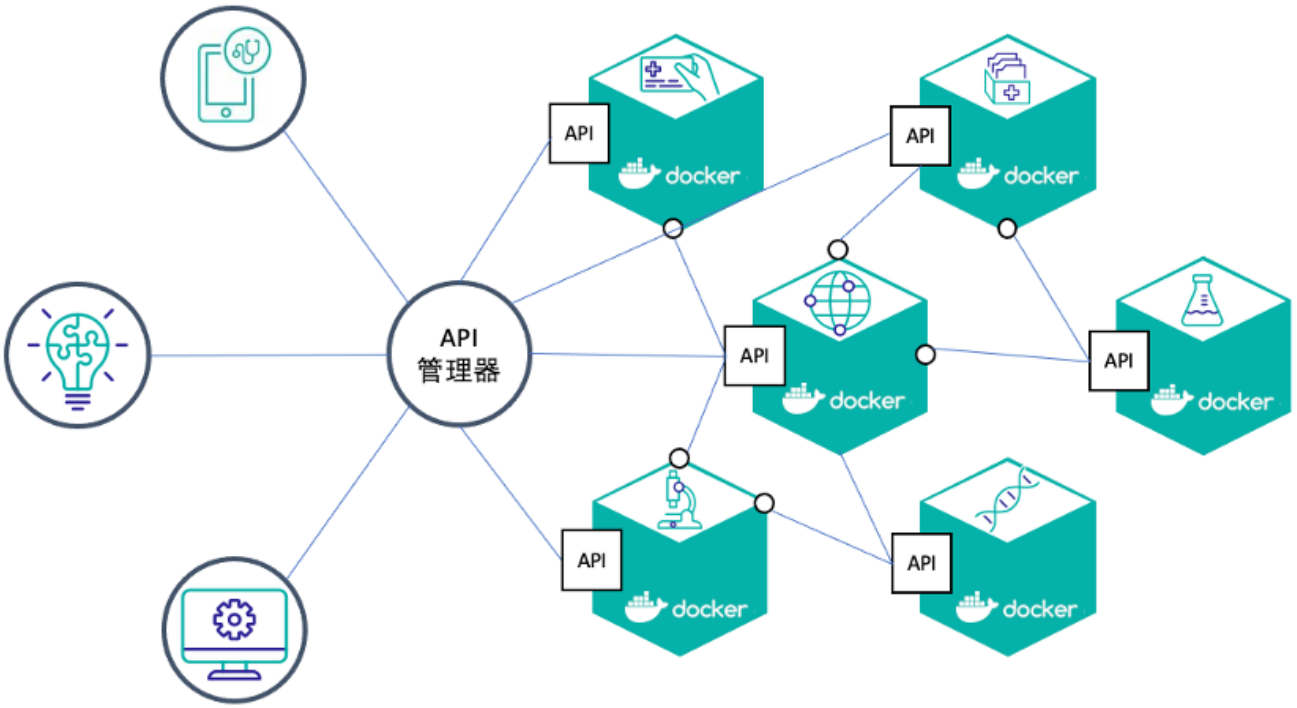
CDA临床文档标准用于***文档交换***和HL7 V3
消息标准用于***消息交换***
，它们都是基于XML的。IHE也推出了基于SOA架构的***业务协同服务***标准。

ESB中心化的部署模式意味着低维护成本和高一致性，但敏捷性不够。

微服务架构（MSA）
：2010年代开始至今。微服务架构类似SOA，强调业务封装和复用、业务解耦，往往和SOA发生混淆。但今天的业务环境已经和10多年前不可同日而语了：

- 业务迭代进化速度上，越来越快；
- 业务范围上，传统的企业内部服务的围墙被突破了，越来越多的业务需要直接和上下游供应链打通，服务延伸到企业之外，直接面向客户。例如以前患者只能去医疗机构看病，医疗机构也只有患者的院中数据，现在不但有互联网医院，医院也需要院前、院后的患者数据和例如基因测序公司来的组学数据来支撑精准医学；
- 业务量上，随着传统业务围墙的打破，业务量越来越大，例如互联网医院和医疗物联网带来了大量的业务和数据。传统可扩展性已经难于满足，需要提供更灵活的、更细颗粒度的业务弹性；
- 部署模式上，云部署、容器化部署越来越主流。

而SOA架构技术上很重：复杂啰嗦的XML、沉重的SOAP协议、中心化的ESB架构，在当今追求敏捷性的今天显得力不从心，甚至从“赋能者”变成了“拖累者”。



微服务架构站在SOA的肩膀上，而且：

- 轻巧的JSON、轻量化的Restful API*帮助微服务架构获得更好的性能；
- 不同于严格分工、各司其职的的需求分析-设计-开发-测试-部署的瀑布模式（Waterfall），微服务架构允许团队合作的持续开发、持续部署模式（CI/CD），可以更快的响应业务需求，具有更好的敏捷性；
- 而云原生、容器化的特性又帮助微服务架构获得了去中心、高度的伸缩性和适应互联网时代的快速迭代能力；
- 重新组合微服务可以快速构建新的应用、满足新的业务；
- 适合打造软件即服务(SaaS)类型的应用。

因此越来越多的企业采用微服务架构战略支撑他们业务创新和数字化转型。HL7 在2014年推出了采用微服务架构的FHIR标准，并参考持续开发持续部署的方式，以成熟度模型为依据不断产生、优化FHIR资源和API。

注：API不等于微服务， API有可能是单体架构的API。

微服务架构构建了一个跨业务域的、跨企业的、去中心的或多中心的架构，其中的API提供分子级的服务独立性， 快速增加、快速迭代、分布部署。不过，这让API的发现、访问、版本管理、安全管理、流量管理、测试等变得很复杂，例如应用访问这么多不同服务端点的微服务，如果后台微服务的端点改了怎么办？如果微服务升级怎么办？因此API网关应运而生。

API网关为API提供全生命周期的管理：

- API的创建、发现与测试
- API版本管理
- API安全管理
- 协议和数据转换管理
- API流量管理
- API统计分析

API管理器在API网关基础上提供开发、管理、运维门户， 进而成为基于API集成的集成架构。它不需要中心化的消息引擎或服务总线，本质上是一个去中心化、应用于更紧密业务集成度的互操作方式。

业务集成范式和特点

至此，我们现在在市场上常见的集成相关术语都全了。虽然各个厂商使用不同的名词，但可以大致梳理一下：

消息队列=消息引擎=消息中间件

API网关=API管理器

接口引擎=集成引擎=集成中间件

而集成平台是对于使用各种架构作为应用集成产品的统称。

同时，从上面的分析也可以看到常见的集成范式(套路)：

- 点对点：

点对点集成不需要任何特定的技术和数据规范，由被集成的双方确定接口方式并开发实现。

- 消息交换：

消息交换基于消息引擎，应用在低业务集成度的跨数据管理域的业务系统间。在医疗行业通常消息是基于临床事件的，描述临床事件的上下文，并在临床事件发生时通过消息引擎路由给消息接收方。典型的医疗行业消息交换标准就是HL7 的V2和V3消息。

- 文档交换：

在医疗行业，有别于基于临床事件的消息，文档是阶段性、小结性的完整医疗信息汇总。它也是应用在低业务集成度的跨数据管理域的业务环境中的，不过通常是不同的医疗机构间。

文档交换的可以通过消息引擎，也可以使用其它方式。

文档的标准常见的有HL7 CDA、C-CDA。

- 服务：

服务是封装好并暴露出的一组内聚的应用功能。所以基于服务交互的互操作，需要双方规范互操作的业务流程和角色。服务交互通常基于面向服务架构，通过服务总线交互。也应用在低业务集成度、跨数据管理域的业务环境中。

服务基于规范的业务流程和角色设定，而并不是所有的医疗流程都已经或能够规范，因此服务交互的适用范围有限。

最常见的国际服务标准就是IHE。

- API：

有别于消息和文档，API可以只传输必要的信息，而不是完整的上下文。与服务类似，但它不需要中心化的消息引擎或服务总线，本质上是一个去中心化，应用于更紧密业务集成度的互操作方式。

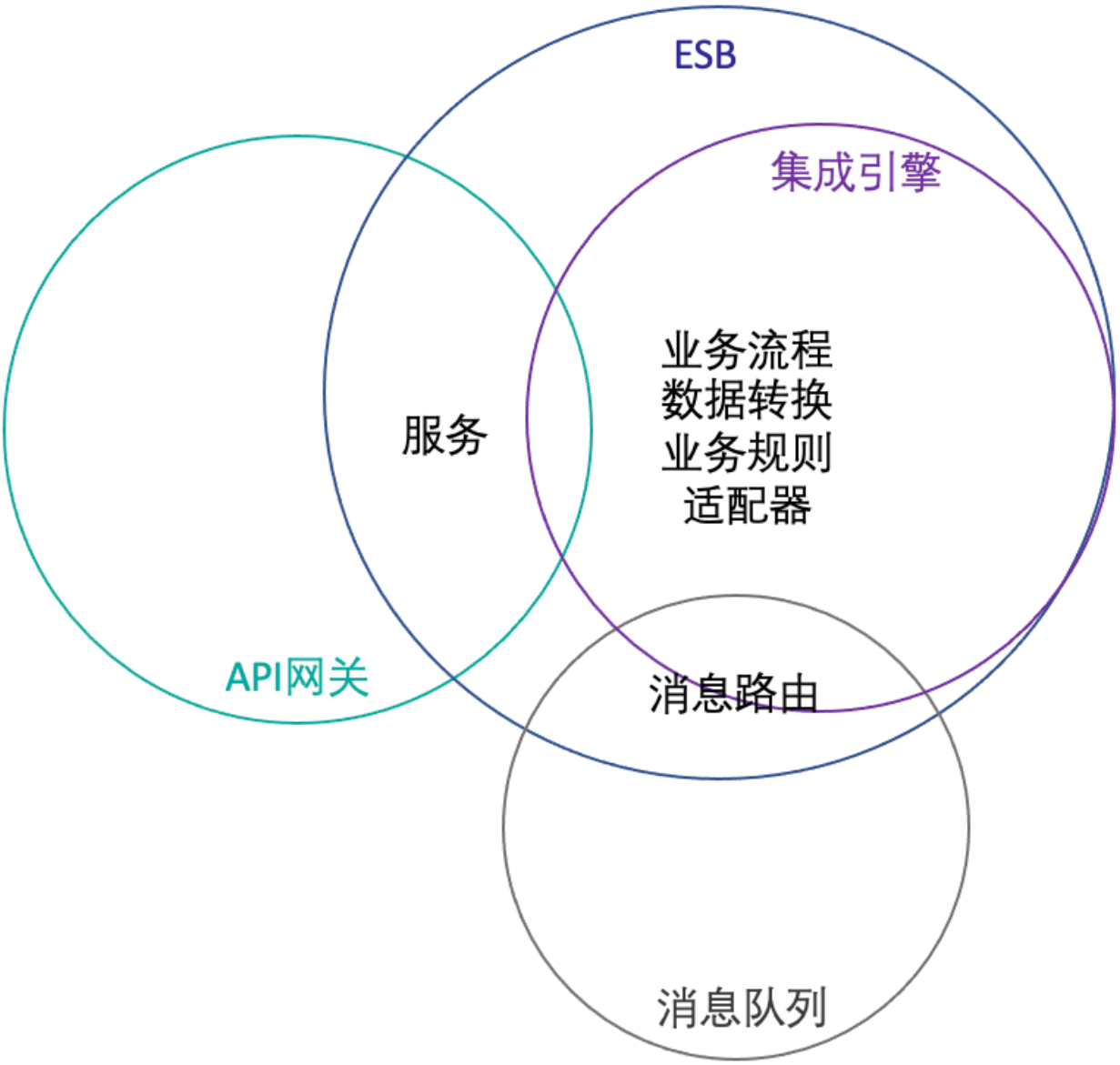
业务集成架构和特点

有哪些主要的应用集成架构呢？大致有这么几种：

- 点对点集成
- 基于消息队列（消息中间件）的集成

- 基于集成引擎的集成
- 基于ESB的集成
- 基于API网关的集成

下面这张图显示各种架构（除了点对点）的能力和它们之间的技术重叠情况。



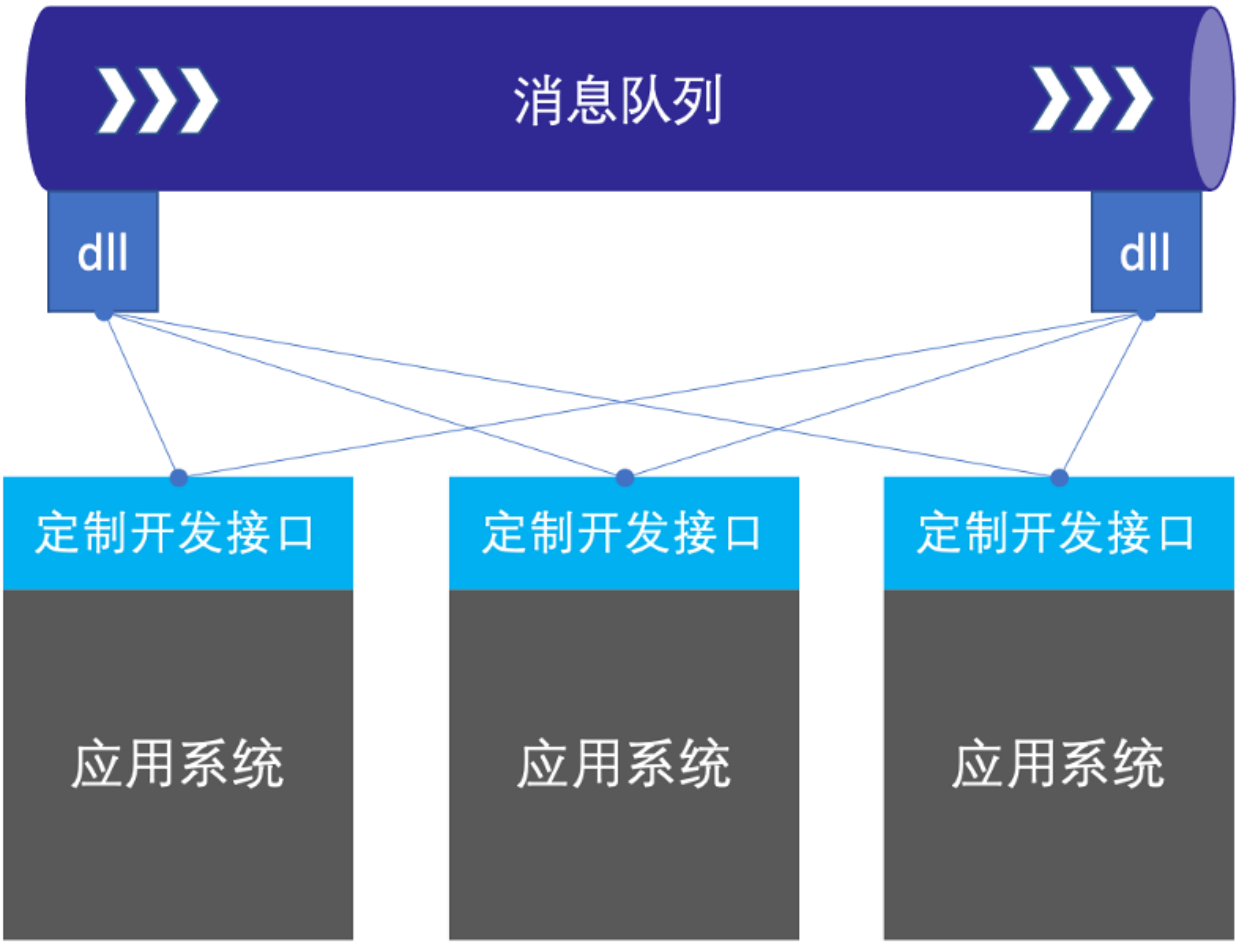
基于消息队列/消息引擎/消息中间件的集成方案：

消息引擎是非常有价值的基础技术架构，很多应用都需要基于消息队列处理，例如保证先进先出、异步处理、处理长时间运行的任务、削峰填谷等。因此很多集成场景也需要基于消息队列和消息路由。这类方案适用于消息交换和文档交换范式。

但消息引擎仅仅是集成需要的一个底层基础技术架构，完整的集成方案应该整合好其它技术，例如解决连接问题的适配器、流程建模、数据转换等，也就是说基于消息引擎的方案需要做很好的二次开发和封装才能开发出一个适用的集成平台或集成方案。

同时，我们看到市场上不少基于消息引擎的“ 半成品 ”集成方案。这类方案的特点是仅有消息引擎，通过消息路由规则来替代流程建模；同时要求各被集成的应用直接连接消息引擎来收发消息。这意味着：

- 高昂的开发成本：不论各业务系统本身是否已经提供有接口，它们都需要现场使用消息引擎提供的连接技术，如dll，开发出一个模块用来连接消息引擎并收发消息、再和自己的应用做对接。
- 性能和稳定性风险：对于开发本身，要求来自不同技术栈的开发人员按消息引擎提供的技术栈进行开发，面临很高的性能和稳定性风险。
- 能力缺失：显然，没有数据校验能力、数据丰富与转换能力、复杂的流程建模能力…

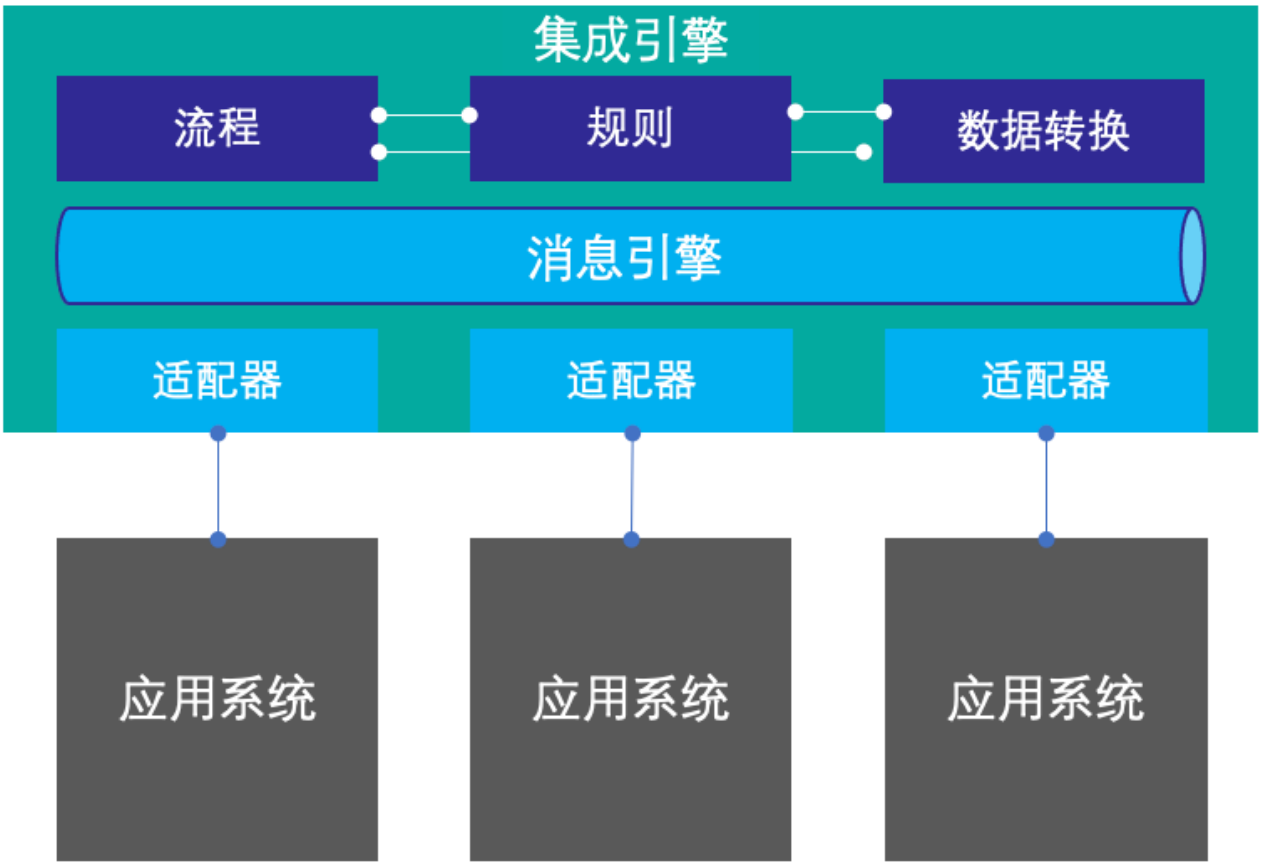


如果把集成平台/集成方案看作一台整车，那消息引擎只是一个零件。因此，这类“ 半成品 ”集成方案/产品是应该谨慎采用的。

基于集成引擎的集成方案：

这个架构通过适配器以其既有接口能力连接业务系统，因此对业务系统的改造要求很低。适配器有很多类型：

- 技术协议适配器可以连接各种技术协议，如TCP、HTTP、SOAP、REST、SQL
- 数据协议适配器支持各种数据协议，如HL7 V2、DICOM、X12
- 应用适配器可以连接各种应用，如SAP、IBM WebSphere MQ
- 语言适配器可以调用各种语言编写的代码，如Java、.net、python



集成引擎可以满足多种互操作范式，包括消息交换和文档交换。通常都具有：

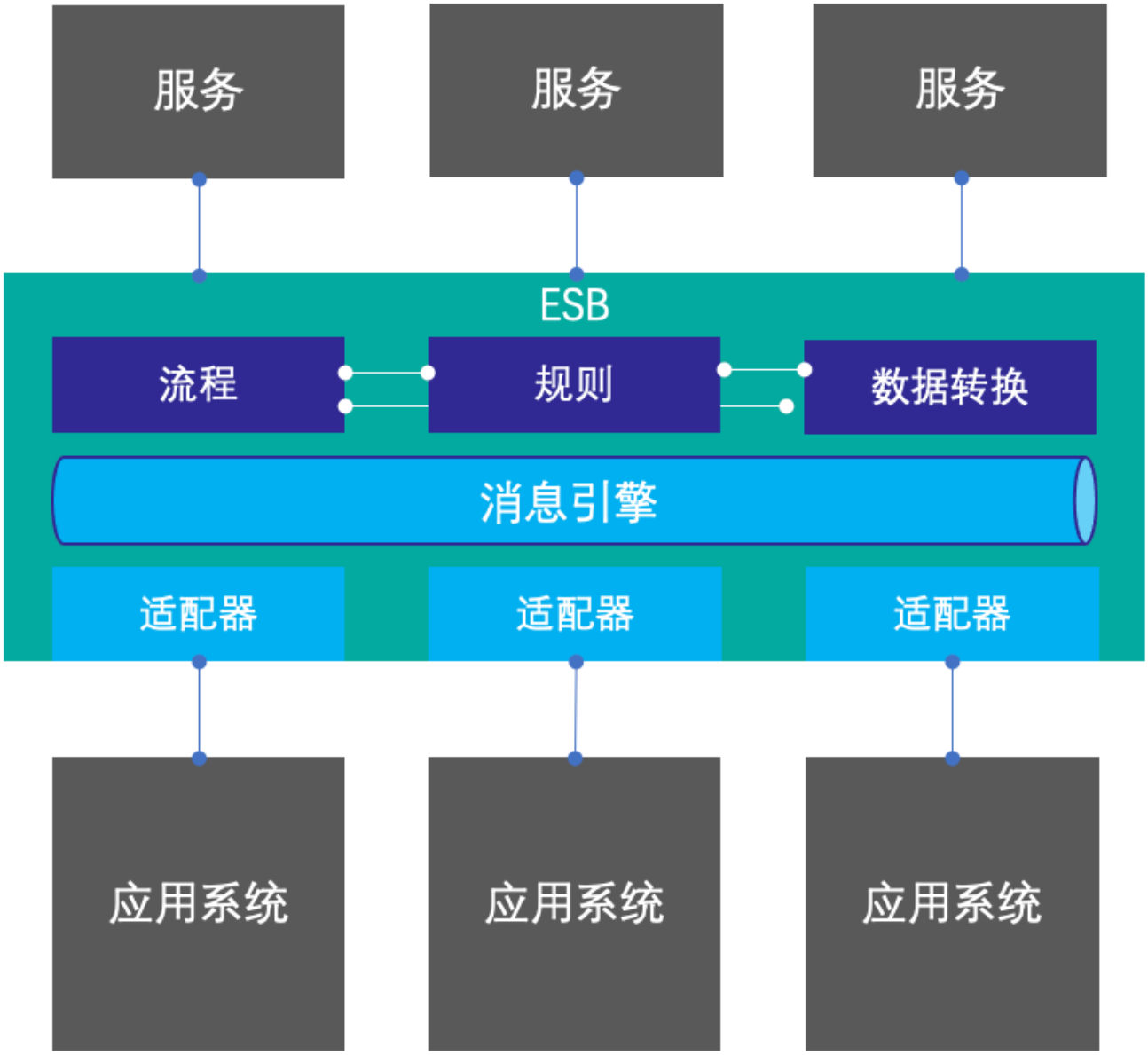
- 消息引擎：用于基于消息路由和业务流程的交换，保证先进先出、异步处理等。但一般是内部使用，而不会直接暴露给被集成系统直接使用。业务系统还是通过适配器连接。
- 业务流程引擎：创建和运行业务流程模型，实现业务流程自动化。相对路由规则，它提供更复杂流程处理的能力，例如循环处理、数据丰富（按流程从不同数据源补齐数据）。
- 规则引擎：建立并执行规则逻辑，给出逻辑输出。通常规则用于消息路由和业务流程。
- 数据转换引擎：建立数据在各种模型间的转换关系，并使用它们对数据进行转换。

集成引擎提供完整的工具集简化了集成工作，它也是目前市场采用最多的架构方案。但其中心化的架构影响了其弹性，同时为避免成为单点故障和性能瓶颈，需要其提供高可用能力和良好的性能。

基于ESB的集成方案：

企业服务总线是一个中心化的架构，通过封装、注册、协同调用和监控服务，以面向服务架构应用在业务集成领域，且多数用于企业内部的应用集成。

很多ESB产品都具有集成引擎的核心能力，例如适配器、流程建模、数据转换等。在集成架构上，基于ESB和基于集成引擎二者相似度非常高，甚至可能就是不同厂商叫的不同的名字，差异在于集成引擎不一定是基于SOA架构的。



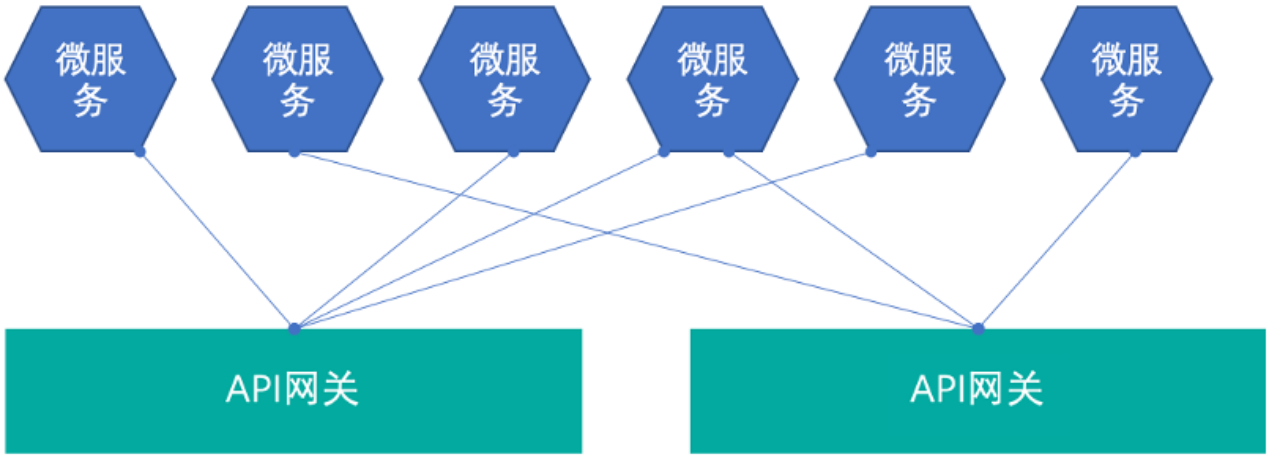
ESB的缺点在于其中心化的架构，一来要避免成为瓶颈，需要其有强大的性能；二来其架构较重，应对微服务多少有些力不从心。因此通常用于业务内部的应用集成。

基于API网关的集成方案

当前处在一个企业（包括医院）拆业务围墙的阶段。API和API网关更能帮助企业突破传统IT围墙，构建全新的去中心化或多中心化的业务IT环境。

在企业生态中的微服务只需要关注业务实现，不用考虑统一认证与授权、安全传输等等问题，从

而更从容应对跨业务边界的应用场景、 更容易容器化和集群化。API网关会处理这些问题，而且还可以做数据转换、路由、协议转换等。API网关可以灵活部署，例如为不同的应用部署不同的API网关，从而也降低了对API网关本身高性能的要求。



微服务是一个去中心化的架构，但API网关是中心化的，虽然可以部署多个API网关，但其仍有沦为小ESB的风险。另外API网关对于现有的非服务化的IT资产无能为力。

展望

相信随着软件架构的发展，集成技术也会不断发展。但立足于当前的技术，在集成架构和方案上，仍能看到一些趋势。

融合多种集成架构的集成平台

从上面分析可见，面对当前多样化的企业信息化环境，上述各种集成架构恐怕都无法单独满足所有集成需求。融合多种集成架构的集成平台产品能够更好地应对多种集成场景。

这样的集成平台可以梳理和集成现有业务，在保证业务稳定运行的前提下，逐步将现有不具备弹性的业务进行服务化或微服务化，从而安全、平稳地实现数字化转型。

数据与业务集成融合

数据的价值在于决策。随着数字化转型，企业IT架构需要更真实、更及时、更全面地表达现实世界(数据全貌)和更完整的流程（流程全貌），将开放性分析能力纳入数据和流程，提供基于实时数据的完整洞悉（insight），并将预测结果甚至是决策结果直接反馈回业务流程。

当今负责数据全貌的是数据中心，承载业务流程的是ESB，它们是不同的产品和架构，业务流程的产生数据要定期同步到数据中心，也造成数据中心的及时性成问题。

如果将数据中心比作数据存蓄的湖，集成平台比作数据流动的江，那我们需要江湖一体的架构，以提高数据实时性和流程决策闭环能力。在数据层面，要能处理、容纳各种模型的数据，支持数据编织(Data
Fabric)。同时这个架构下要有强大的分析、决策能力，包括BI、自然语言处理、机器学习等。

业务创新平台

业务创新依赖现有业务基础。使用ESB梳理现有企业信息、流程资产，拥抱新架构将数字资产微服务化，并通过创建新的服务、组合现有服务快速创新。融合行业互操作和数据标准，提供开箱即用的行业价值。拥有这些能力，将帮助现有架构演化为企业的业务创新平台。

当然，这些已经不仅是集成架构的未来，而是整个数字化架构的未来。

#InterSystems API管理器（IAM） #InterSystems 业务解决方案和架构 #业务流程 (BPL) #持续集成 #HealthShare #InterSystems IRIS #InterSystems IRIS for Health

---

源URL: https://cn.community.intersystems.com/post/%E7%B2%BE%E5%8D%8E%E6%96%87%E7%AB%A0-%E4%BB%8E%E8%BD%AF%E4%BB%B6%E6%9E%B6%E6%9E%84%E5%8F%91%E5%B1%95%E8%B0%88%E4%B8%9A%E5%8A%A1%E9%9B%86%E6%88%90%E6%8A%80%E6%9C%AF%E6%BC%94%E8%BF%9B%E4%B8%8E%E5%B1%95%E6%9C%9B