深入浅出Streaming API

数据推送在项目集成中是很常见的场景，在Salesforce中，Streaming API是数据推送的一种方案。

Streaming Event Features | Streaming API Developer Guide | Salesforce Developers

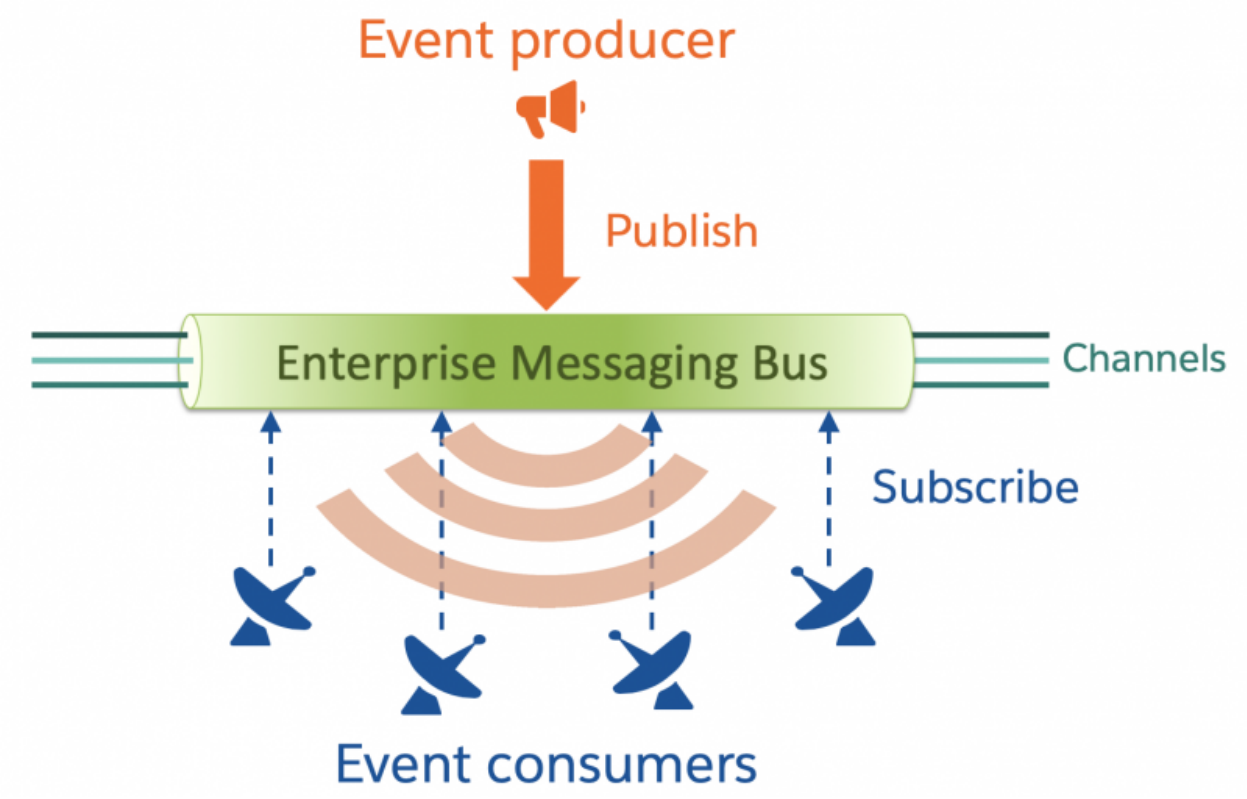
Receive notifications in a secure and scalable way with API Streaming. To receive notifications of changes to Salesforce data that match a SOQL query, use PushTopic Streaming. To deliver events with a payload of your choosing, use Generic Streaming.

https://developer.salesforce.com/docs/atlas.en-us.api_streaming.meta/api_streaming/event_comparison.htm

Streaming Api是一种数据监控，一共有四种模式：generic, push topic, platform event和change data capture

Streaming Api分两代，第一代是generic和push topic，随着技术的发展又推出了platform event和change data capture。

所有的Streaming Api都建立在相同的核心技术上**CometD**。此技术依赖于消息总线，该总线为事件(有时也称为消息)提供发布/订阅机制。



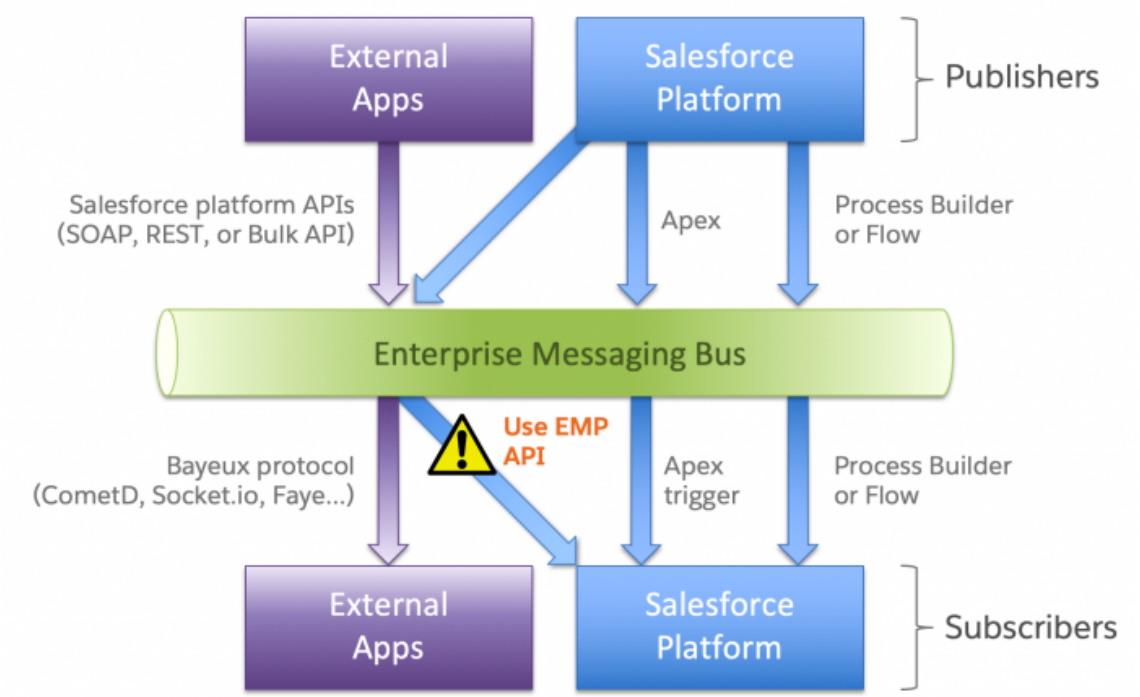
发布/订阅机制遵循以下场景：

1. 事件使用者连接到消息总线并订阅特定的通道(事件类型)
2. 事件生产者连接到总线，并在通道上发布事件
3. 总线将事件广播给所有注册的订阅者

这意味着消息生产者与使用者完全解耦。该模型允许大规模构建事件驱动的体系结构。典型的用例是系统集成和实时应用程序。

Streaming API的四种模式虽然是使用相同的底层结构**CometD**，但是他们还是有一些各自独立的特性，如下图：

会注意到，第二代api(Platform和CChange data capture)在发布和订阅方面提供了最大的灵活性。最值得注意的是，引入的新一代api具有强大的声明功能，因此，在可能的情况下应该优先使用它们。



当streaming api出现时，浏览器最初还没有可用的本地客户端。订阅自定义UI组件中的流事件将需要大量代码。随着lightning:empAPI（Aura）组件和lightning-emp-api（LWC）组件的引入，情况发生了变化。这个很棒的组件为您管理CometD连接(身份验证、多路连接等)，并极大地减少了需要编写的代码量。可以参考这期文章：

Building Real-Time Lightning Apps Gets Even Simpler - Salesforce Developers Blog

Building modern apps that react to changes in real-time used to require some implementation effort, but the latest update to our streaming technologies greatly simplifies things. In this post we will go through a brief refresher on the Streaming API and we will introduce you to a

https://developer.salesforce.com/blogs/2018/10/building-real-time-lightning-apps-gets-even-simpler.html

Platform Event

本质上，平台事件是在事务上下文之外发布的。换句话说，如果您有一些发布事件并在之后进行一些处理的Apex代码，那么事件将立即发布，而不等待Apex执行完成。这意味着Apex执行可能失败，数据库事务可能回滚，但事件已经发送，无法取消。

我之前的文章中也利用过Platform Event的这一特性做过一个日志的案例。

（预留一个链接位）

引入了发布行为设置后，您现在可以决定在Apex执行完成后延迟事件发布。相关文档。

Change Data Capture updates

Change Data Capture update (CDC)有两个主要更新:Apex触发器支持和自定义通道。

就像平台事件一样，您现在可以为CDC事件编写Apex触发器。与传统对象触发器相比，这带来了额外的灵活性，因为CDC触发器在首先**修改数据的数据库事务的范围之外执行**。

CDC的第二个显著更新是引入了具有自定义通道的流组成。在典型的业务场景中，您可能希望订阅多个事件，如帐户、联系人和订单记录更改。然而，以一致的方式处理单独的事件具有挑战性，因为您可能需要首先编写一些代码来聚合它们。使用自定义通道，您现在可以将这些事件重新组合到您可以处理的单个自定义事件中。这意味着您可以轻松地一次处理相关的更改。相关文档。

特别介绍工具：Streaming Monitor

由于流事件的短暂性，在没有适当工具的情况下使用流api实现应用程序可能会很困难。过去的事件不会保存到数据库中，因此没有内置的用户界面来查看日志。因此，为了调试，我们常常浪费时间编写一些可丢弃的代码来跟踪这些事件。

在AppExchange上有一个App：Streaming Monitor，这个工具可以让你监控四个Streaming Api，并方便地在一次点击中回放过去的事件。这会为你节省一些宝贵的时间。GitHub。

相关Trailhead：

- Change Data Capture Basics
- Platform Events Basics
- Build an Instant Notification App