Amazon MWS Research

MWS 提供了哪些功能，在亚马逊MWS 文档里很清楚提到：

MWS 提供以下功能：

* 库存管理 — 您可以执行库存批量上传、添加商品、检查库存数量、查看定价信息及其他库存管理任务。
* 订单管理 — 您可以下载订单信息、获取付款数据、确认订单以及安排报告时间。
* 报告管理 — 您可以请求多种报告、查询报告状态并下载报告。

利用亚马逊 MWS 执行以下操作：

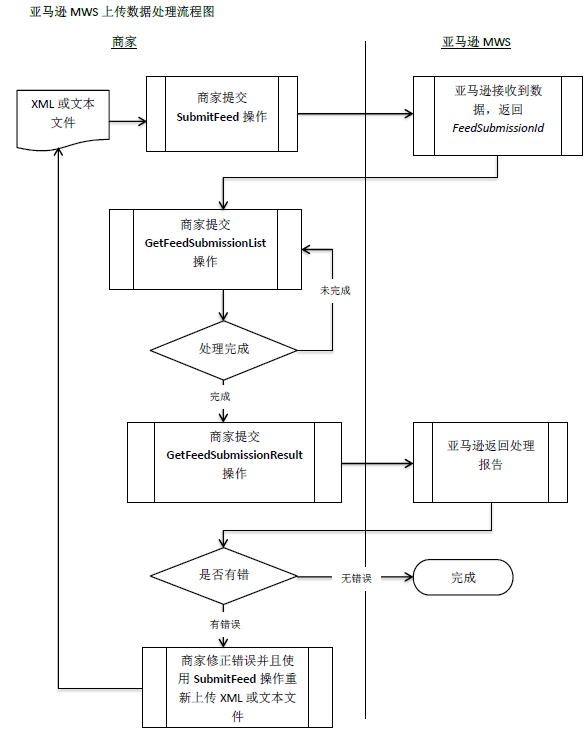
* 创建发往亚马逊运营中心的入库货件 — 您可以自动完成该流程，为发往亚马逊运营中心的商品创建标签。
* 查看入库货件的状态 — 您可以查看货件是否已到达运营中心。如已到达，是否已得到处理。
* 提交配送订单 — 将您的系统与亚马逊 MWS 进行整合后，您的买家即可随时提交多渠道配送订单。当您处理或批量处理订单时，也不会产生时间延迟。
* 追踪并管理出库货件请求 - 订单离开亚马逊运营中心后，您可以追踪货件，并帮助买家时刻了解送达时间。

William 总结： 这里面从程序员的角度看其实可以分为两部分：

1. 本地系统的数据上传到 MWS 服务， 即把本地系统的数据更新到亚马逊网上商城。 这种上传操作无非是涉及：对商品数据，库存数据，定价数据，订单数据等的增加， 删除，更新操作。
2. 从MWS 服务获取数据。把MWS的数据下载并更新到本地系统。

挑战： 本地系统与MWS 系统数据结构和业务逻辑不相同， 此时通常需要构建数据装换层作为数据交换的桥梁。

1. 先看看第一部分： 数据上传到MWS
   1. MWS数据上传的原理: 如图所示



* 1. 个人分析:
  2. 所有上传的数据都是采用 XML 的格式作为信息内容。现在的数据格式普遍使用JSON格式，因为JSON 格式具有很多优越性，最大的优越性直接被 Javascript使用。这点 XML 格式，对系统的数据转化工作会使用编程量和难度增加不少。
  3. MWS 的服务是异步服务， 上传的数据需要经过MWS排队处理，所以我们上传个数据，不要指望马上看到更新或者成败。 我们的系统必须要设计专门的模块或者机制来跟踪成功与否， 失败的要列出错误出在哪里，修改后，继续上传， 继续跟踪，重复此过程，直到成功为止。这个需要有很强的异步编程思维（尤其是经验）。
  4. 在MWS的文档里，只是提到了上传的原理和框架，但是并没有列举上传 XML 信息的内容, 如： 商品的，库存的，运输的，XML 应该是怎样的内容。由于我没有MWS 的账号，我估计应该是登陆MWS以后，到各个模块里应该有规范。等拿到这些细节定义， 按照MWS的规范生成XML 内容上传即可。问题不大

1. 在看看第二部分： 从MWS 获取数据
   1. 获取数据部分就相对简单许多而且也提供了很详细的文档

个人点评： 获取数据不需要构造复杂的 XML 信息内容， 都是通过URL 的方式来获取，当然为了数据安全，URL 里必须要包含有必须的验证码：如access key, token 之类的。

例如：获取订单列表：

只要提交构建好URL 即可（当然必须要研究好各个参数的意思）：

https://mws.amazonservices.jp/Orders/2013-09-01

?AWSAccessKeyId=0PB842EXAMPLE7N4ZTR2

&Action=ListOrders

&MWSAuthToken=amzn.mws.4ea38b7b-f563-7709-4bae-87aeaEXAMPLE

&MarketplaceId.Id.1=A1VC38T7YXB528

&FulfillmentChannel.Channel.1=MFN

&PaymentMethod.1=COD

&PaymentMethod.2=Other

&OrderStatus.Status.1=Unshipped

&OrderStatus.Status.2=PendingAvailability

&SellerId=A2NEXAMPLETF53

&Signature=ZQLpf8vEXAMPLE0iC265pf18n0%3D

&SignatureVersion=2

&SignatureMethod=HmacSHA256

&LastUpdatedAfter=2013-08-01T18%3A12%3A21

&Timestamp=2013-09-05T18%3A12%3A21.687Z

&Version=2013-09-01

MWS 收到请求以后，会返回以下XML格式的信息：

<?xml version="1.0" encoding="utf-8"?>

<ListOrdersResponse xmlns="https://mws.amazonservices.com/

Orders/2013-09-01">

<ListOrdersResult>

<NextToken>2YgYW55IGNhcm5hbCBwbGVhc3VyZS4=</NextToken>

<LastUpdatedBefore>2013-09-25T18%3A10%3A21.687Z</LastUpdatedBefore>

<Orders>

<Order>

<ShipmentServiceLevelCategory>Standard

</ShipmentServiceLevelCategory>

<ShipServiceLevel>Std JP Kanto8</ShipServiceLevel>

<EarliestShipDate>2013-08-20T19:51:16Z</EarliestShipDate>

<LatestShipDate>2013-08-25T19:49:35Z</LatestShipDate>

<MarketplaceId>A1VC38T7YXB528</MarketplaceId>

<SalesChannel>Amazon.com</SalesChannel>

<OrderType>Preorder</OrderType>

<BuyerEmail>5vlhEXAMPLEh9h5@marketplace.amazon.com</BuyerEmail>

<FulfillmentChannel>MFN</FulfillmentChannel>

<OrderStatus>Pending</OrderStatus>

<BuyerName>John Jones</BuyerName>

<LastUpdateDate>2013-08-20T19:49:35Z</LastUpdateDate>

<PurchaseDate>2013-08-20T19:49:35Z</PurchaseDate>

<NumberOfItemsShipped>0</NumberOfItemsShipped>

<NumberOfItemsUnshipped>0</NumberOfItemsUnshipped>

<AmazonOrderId>902-3159896-1390916</AmazonOrderId>

<PaymentMethod>Other</PaymentMethod>

</Order>

<Order>

<AmazonOrderId>058-1233752-8214740</AmazonOrderId>

<PurchaseDate>2013-09-05T00%3A06%3A07.000Z</PurchaseDate>

<LastUpdateDate>2013-09-07T12%3A43%3A16.000Z</LastUpdateDate>

<OrderStatus>Unshipped</OrderStatus>

<OrderType>StandardOrder</OrderType>

<ShipServiceLevel>Std JP Kanto8</ShipServiceLevel>

<FulfillmentChannel>MFN</FulfillmentChannel>

<OrderTotal>

<CurrencyCode>JPY</CurrencyCode>

<Amount>1507.00</Amount>

</OrderTotal>

<ShippingAddress>

<Name>Jane Smith</Name>

<AddressLine1>1-2-10 Akasaka</AddressLine1>

<City>Tokyo</City>

<PostalCode>107-0053</PostalCode>

<Country>JP</Country>

</ShippingAddress>

<NumberOfItemsShipped>0</NumberOfItemsShipped>

<NumberOfItemsUnshipped>1</NumberOfItemsUnshipped>

<PaymentExecutionDetail>

<PaymentExecutionDetailItem>

<Payment>

<Amount>10.00</Amount>

<CurrencyCode>JPY</CurrencyCode>

</Payment>

<PaymentMethod>PointsAccount</PaymentMethod>

</PaymentExecutionDetailItem>

<PaymentExecutionDetailItem>

<Payment>

<Amount>317.00</Amount>

<CurrencyCode>JPY</CurrencyCode>

</Payment>

<PaymentMethod>GC</PaymentMethod>

</PaymentExecutionDetailItem>

<PaymentExecutionDetailItem>

<Payment>

<Amount>1180.00</Amount>

<CurrencyCode>JPY</CurrencyCode>

</Payment>

<PaymentMethod>COD</PaymentMethod>

</PaymentExecutionDetailItem>

</PaymentExecutionDetail>

<PaymentMethod>COD</PaymentMethod>

<MarketplaceId>ATVPDKIKX0DER</MarketplaceId>

<BuyerName>Jane Smith</BuyerName>

<BuyerEmail>5vlhEXAMPLEh9h5@marketplace.amazon.com</BuyerEmail>

<ShipmentServiceLevelCategory>Standard

</ShipmentServiceLevelCategory>

</Order>

</Orders>

</ListOrdersResult>

<ResponseMetadata>

<RequestId>88faca76-b600-46d2-b53c-0c8c4533e43a</RequestId>

</ResponseMetadata>

</ListOrdersResponse>

* 1. 个人分析：

返回的数据是以XML 的格式返回， 如果上所说与JSON 格式相比，XML 的缺点， 我们必须要对XML 进行转化成我们需要的数据模型。 处理会相对JSON 格式复杂。 而且对数据的转换必须是在本地服务器端完成。因为用Javascript 转换XML 不那么容易。

1. 关于 MWS API 与 其他API 的比较：
   1. 从技术的角度来比较： 显然如果MWS 可以根据我们提出的数据格式要求来返回我们所需要的格式，将很完美。因为我接触到很多第三方的API，他们可以根据请求的格式返回。

例如： Google Alert:

content-type: application/json - 则返回JSON 格式

content-type: application/xml - 则返回 XML格式

* 1. 技术文档： MWS 关于上传修改数据的文档，只详细介绍了上传的规范与原理， 却没有对具体的操作文档化。而且google 也很少见到代码例子。而获取数据的API 文档却写得很详细。令人费解。
  2. 从业务的角度比较： 不同的API 完成不同的使命，没有可比性。

个人最后总结： MWS API 和其他API ，从程序设计来说， 主要就是数据交换，没有什么技术难度。 关键的难度就是对现有系统的认知和业务逻辑的整合，以及管理层对现有系统改造或者重新设计的魄力。