

现有系统流程

1. 获取数据服务请求 amazon mws .
2. 获取数据服务对MWS返回的数据进行处理后,存储进中间件数据库.

3. 输出数据服务被动等待SAP 的请求服务

因无法知晓SAP需要哪些数据，需将符合条件的所有数据全部输出，再交由SAP排查。

webService 返回数据效率低，速度慢 , 请求webservice 耗时，一次只能返回100个订单，SAP解析XML耗时.

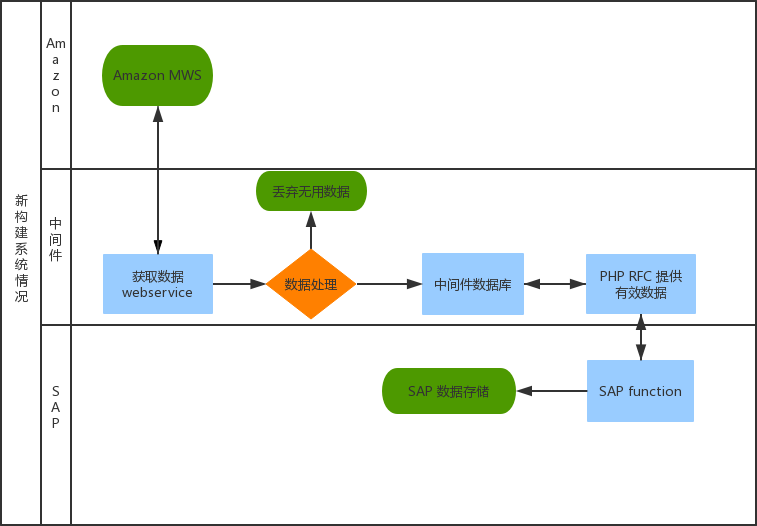
无法知晓数据是否已经全部输出给SAP，需要人工排查，历史数据清理会有风险。

以销售订单为例，SAP请求过去7天的订单，那么每次请求 输出数据服务都会返回过去7天的订单，无论SAP是否已经生成该订单。

1. SAP对输出数据进行处理后，进行存储

数据存储量巨大，排查耗时。 请求服务频繁，造成不停的对返回数据进行排查。

以SAP请求过去7天订单，每2小时请求一次为例，假设过去7天订单有10000单，每两小时新订单增量为500单。SAP在第一次请求后其实就已经保存了所有订单，那么在接下来的1天时间里，SAP每次请求数据都要做9500的重复数据排查，1天12次 已经超过10W多次。SAP 历史订单的数据不会被清理，体积越大，排查速度就会越慢，最后效率越低。



新构建系统情况

1. 获取数据服务请求 amazon mws .

2. 获取数据服务对MWS返回的数据进行处理后,存储进中间件数据库.

3. PHP RFC主动输出数据到SAP function，SAP function 返回告知是否成功，RFC标记已存储到SAP状态

RFC只将新数据提供给SAP，在极端情况下，SAP也只需做简单排查，实测插入数据1000条，耗时0.2秒。

有标记告知是否已经导入SAP，历史数据过大时，可以清理，无风险。

因相比现有系统 中间件做了回写操作，相对只输出，对中间件，数据库及系统的要求 需按数据量相应的提高，以保证服务质量。

4. SAP对RFC数据进行处理后，进行存储

SAP function 接收数据，进行存储

总结：

可以看到区别主要在第三，四点改进上，

现有系统webservice 返回的数据量小，请求耗时，无法回写状态等造成SAP请求webservice 这一步耗时耗力；SAP 不断的对重复数据进行处理，不断的对重复XML进行解析，SAP数据处理端这一步耗时耗力。

新系统则优化了这些方面，用RFC来主动写入数据给SAP，传输数据量大，并自行标记状态，只输出有用的数据给SAP。虽然因为回写会消耗中间件的负载，但是在数据可以安全清理的情况下是完全可以接受的。SAP function 接收过来的数据可以直接用于存储，不用再去不断的请求, 不断的解析XML，不断的排查这个过程。

甘总，早上好， facebook 基本注册需要资料如下：

1 可登录验证的邮箱

2 可验证的手机号 （注册开发者账号和 账号验证需要用到）

3 需要一张头像图片