

# 1.系统结构说明图

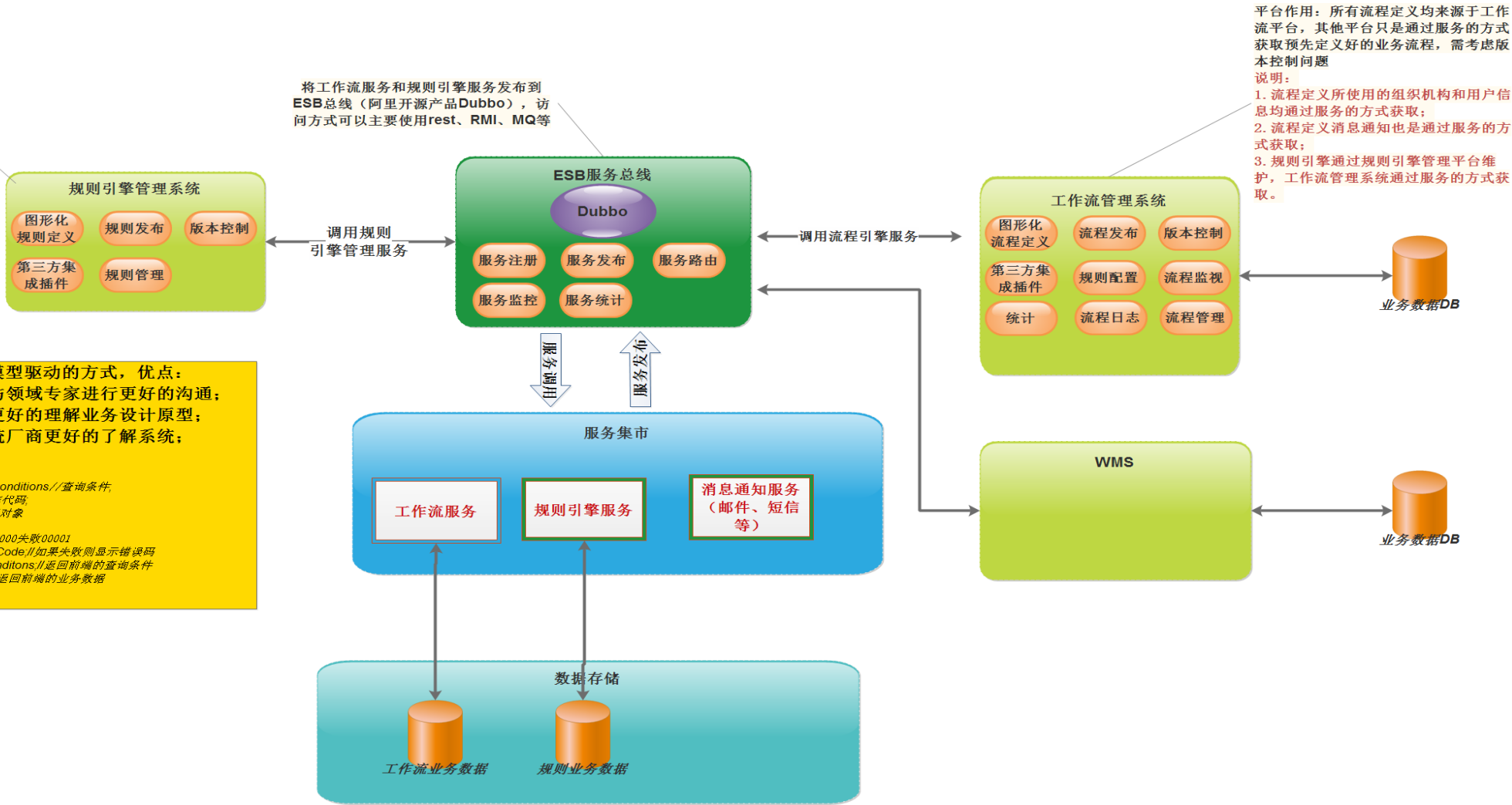
- 1. 规则应用于流程或者直接应用于业务；
- 2. 如何做好版本控制

整个系统设计采用模型驱动的方式，优点：

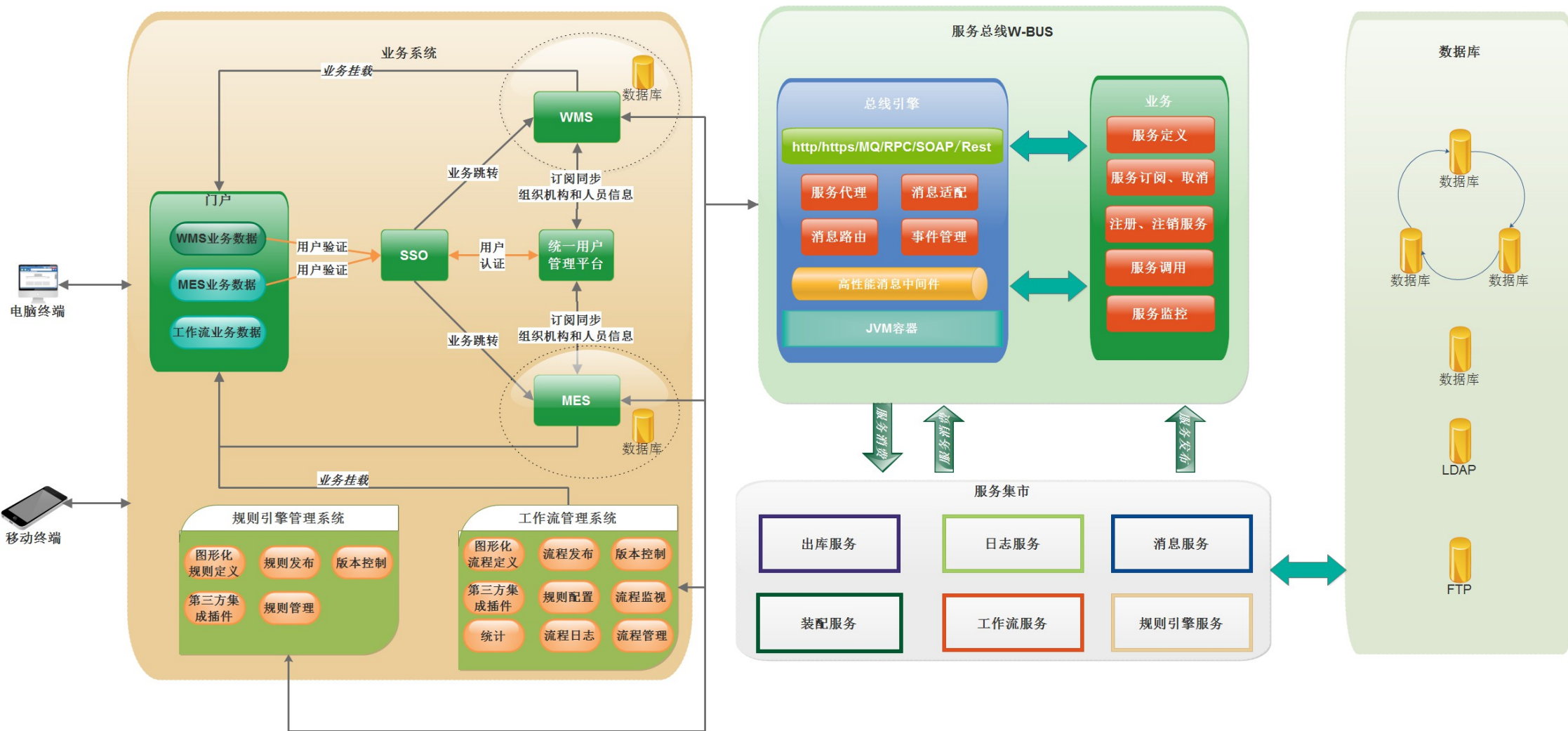
- 1. 有利于软件专家与领域专家进行更好的沟通；
- 2. 有利于开发人员更好的理解业务设计原型；
- 3. 有利于第三方系统厂商更好的了解系统；

例子：

```
ReqMessage:
private Map<String,String> conditions;//查询条件;
private String opeCode//操作代码;
private List<Object> list//前端对象
RespMessage:
private String success//成功00000失败00001
private ArrayList<String> errorCode//如果失败则显示错误码
private Map<string,String> conditions//返回前端的查询条件
private List<Object> dataList//返回前端的业务数据
```



2. 系统总貌图



### 3. 说明

(1) 整个系统设计采用面向服务的方式，服务调用时有可能采用异步的方式调用，系统需支持同步和异步两种方式（**MQ**）；

(2) 系统说明

整个系统采用服务的架构，服务总线使用京东扩展淘宝的 **Dubbo**， workflow管理平台、规则引擎管理平台和 **WMS** 均使用 **JEDEC** 平台开发；

将 workflow平台封装成微服务，可供第三方调用； workflow在设计时所需外部交互的数据，均采用服务的方式获取；

将规则引擎封装成服务，通过规则引擎管理平台进行图形化的设计，同时支持版本控制。

(3) 服务采用微服务的方式, **Spring Boot**;

(4) 服务入口消息采用模型的方式提供，图上有事例；

(5) 需做一个简单的 **Demo**

具体事例：

假如我们有多个库房，现在有客户要下订单（订单中有货物详情），当填写完订单提交后，系统会根据订单中货物的类型，分配至不同的库房办理，也就是会自动生成出库单，相应的库管员登录系统后即可办理出库，出库完成后会回填数据至订单。