附加一点说明

一定要懂得灵活变通，需求分析文档本质，其实就是我们之前需求评审，只是看到了PM写出来的一份产品设计文档。你看到的那份需求文档，实际上是我写的，我不是专业的PM，我很多时候还是技术人员的角度和思维，去写的那份需求文档。

PM是一点技术都不懂的，跟我写的应该是不太一样的，一般来说PM写的产品需求文档，很多细节都是没有的，很多细节是比较模糊的，他不懂底层系统的一些运行流程什么的，他就是站在一个纯小白，纯用户的角度去考虑这个系统产品应该如何设计。

那么如果说是上面说的那样的一种情况，那么需求分析文档，是必写。站在架构师的角度去分析那份需求文档，从需求文档中抽取出来对应的核心需求，考虑清楚我们的系统为了要实现所有的需求，我们要实现哪些功能。

**需求文档的本质，就是考虑清楚系统有哪些功能。**这就是本质。

另外说一点点变通，如果是项目进度很紧张，如果那个产品需求文档写的很清晰，碰到了一个很专业的很好的PM，写出来的产品需求文档就是很清晰，条理清晰，细节重试，流程清晰，那么可以考虑说，这个需求分析文档，不写也可以的。

# 1、引言

## 1.1 编写目的

表明这份需求分析文档的核心目的是什么，比如说是用来定义系统的核心功能的，指导后续的系统设计的

## 1.2 面向对象

该文档编写出来之后，是要给谁看的？其实主要是给RD看的，让工程师明白系统要做哪些东西

## 1.3 参考文档

（1）《电商系统v1.0需求文档》

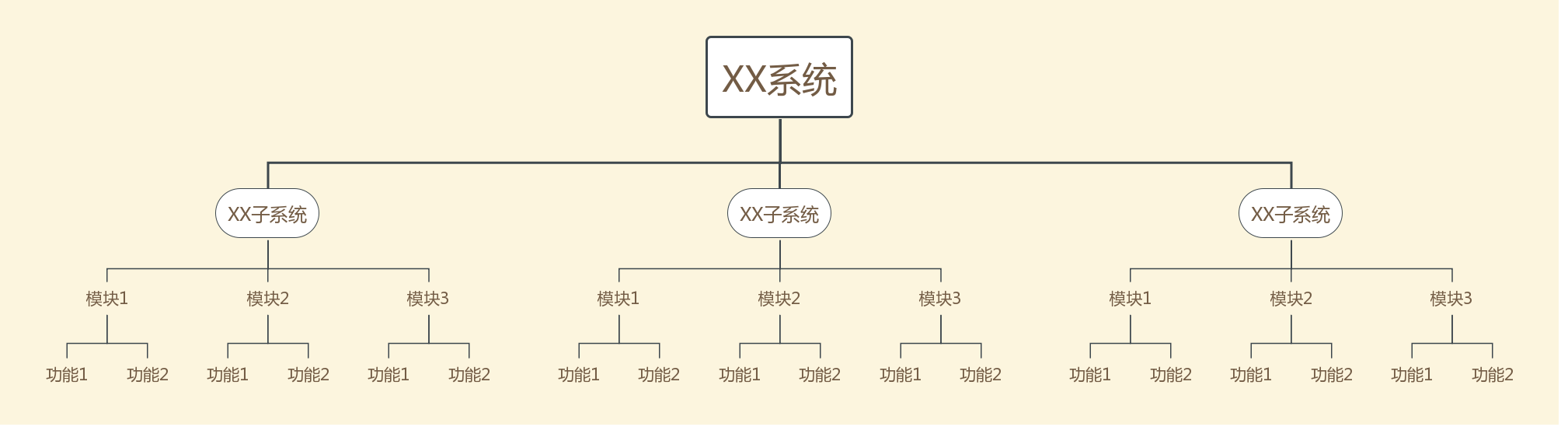
# 2、概述

## 2.1 开发意图

描述清楚咱们这个系统开发出来是用来干嘛的，主要的作用是什么

## 2.2 功能结构图

功能结构图，主要是给人一目了然的看到这个系统需要包含哪些子系统，每个子系统包含哪些模块，每个模块包含哪些功能，同时对每个子系统以及其模块进行简单的介绍。功能结构图只是一个静态的这么一个东西。



## 2.3 功能简介

### 2.3.1 XX子系统

XX子系统主要是用来干什么的

#### 2.3.1.1 模块1

模块1主要是用来干什么的

#### 2.3.1.2 模块2

模块2是用来干什么的

### 2.3.2 XX子系统

# 3、用例图

用例图其实也是用来描述一个系统的这么一个功能的，但是他的意义不太一样，我们从另外一个角度来看一下这个系统的功能。他是每个子系统画一张用例图，这张用例图可能会很大。这个用例图最关键的一点是，强调出来系统的功能，但是同时要强调出来，是哪些参与者跟这些功能在交互。

举个例子，比如说商品中心，商品部门的运营专员，他会负责去使用这个新增商品这个用例，其实这里要体现出来，就是时候每个参与者跟每个用例的关系，每个用例就是对应着一个功能。同时这个参与者可能不是一个人，可能是外部的其他系统，比如说订单中心的更新订单状态的这么一个用例，可能是WMS系统是一个参与者，调用了更新订单状态的这么一个用例。

用例图，是在功能结构图基础之上， 再从另外一个角度来描述这个系统的功能，也就是说描述清楚哪些参与者跟哪些用例有关系。

## 3.1 XX子系统

每个子系统是一张用例图，一张用例图中包含了很多个用例

### 3.1.1 功能概述

说一下XX子系统的功能大概是干嘛的，这里可以说一下，这个子系统是干嘛的，里面包含了哪些模块

### 3.1.2 用例图

对于这个画用例图，我从实践的角度给大家一点建议，首先就是，一定要记住一点，就是对你要做的任何事情看透背后的本质，明白你做这个事情是要干嘛，产生什么价值和作用的？

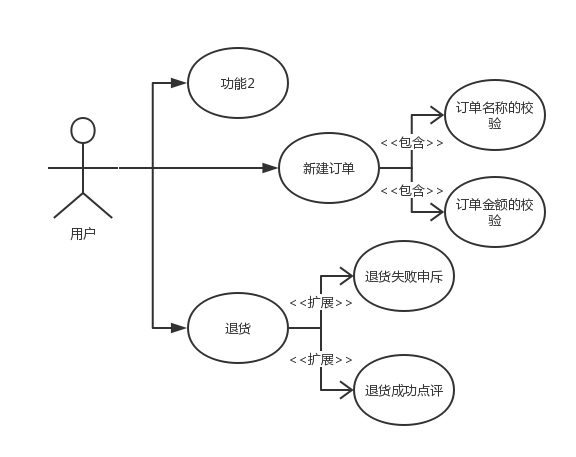
其次，在明白了你要干的事情本质的基础之上，你可以对做这个事情的方法，做一些变通。

一般情况下，我们正常而言，一个用例就是一个单独的一个功能。但是，我们发现自己的系统中，有很多是CRUD类的用例，对某个数据列表、新增、删除、修改、启用/停用，基本就是这么些东西，或者是加上一些提交审核、审核。然后他的话呢就是有很多的重复性的东西。特别的琐碎。

再加上我们的系统比较庞大的，包含了16个中心，几十个模块，几百个功能。如果你每个中心去画一个用例图，每个中心可能都是几十个功能，那个用例图看起来会特别的庞杂无序。

这里我们来思考一下，我们画用例图的本质是为了什么？为了体现出系统中的一些功能跟参与者的一个关系，就可以了。

所以说，如果是这样子的话，我们可以考虑在画用例图的时候，可以考虑将一个模块作为一个用例，也是可以的，看透背后的本质，不要拘泥于固定的形式，一定要懂得变通。画出来这个子系统中，参与者跟模块用例之间的一个关系即可。如果以后觉得有时间了，一定要做一些扩展的话，可以考虑采用包含关系，去画出来一个模块用例里面包含的一些功能用例。



一般情况下，为了简化起见，可以就只是画关联关系就可以了，表明哪些参与者会去执行哪些用例

包含关闭、扩展关系，会相对增加复杂性，可以将那些太细的一些用例不画在里面了

### 3.1.3 用例描述

用例描述这块是这样，如果是系统很大，时间比较着急，同时的话呢某个用例功能对应的流程不是特别的复杂，而且呢，在需求文档里，如果对这个功能的使用流程有了较为详细的说明，我们可以简化，不在这里对每个功能，每个用例，做过于详细的描述

不然的话，相当于是再把需求文档里的东西给全部重新写了一遍，那样的话就比较坑爹了

但是这里需要写明，可以参考需求文档中的功能描述以及功能运行流程

什么情况下，需要对这个用例进行详细的描述呢？如果某个用例，他压根儿就不是产品需求文档里提到的，比如说你自己想到的系统应该做的某个事情，采购入库单的校验，采购入库单备份看，有些用例可能是会在需求分析的过程中，自己站在架构师的角度，初步加入了一些额外的系统层面的功能。那么这种你肯定就是要详细描述用例的。

另外一个，对于一些核心的，重要的功能，要求必须是得写详细的用例描述。常见于某个功能有较为复杂的流程，比如说我们后面的那个调度销售出库，这个用例，他的流程就较为复杂，要跟其他的系统做一些交互，可能光靠需求文档中的描述，不一定特别的清晰。需要我们架构师去对核心重要的功能，包含了复杂的业务流程，去要做详细的描述，画清楚他的泳道图。

但是对于一些CRUD类的一些功能，可能就不需要了。一定要懂得变通，不要认死理，不要认死法。

#### 3.1.3.1 用例1描述

（1）简要说明

这个用例对应的功能是干嘛的

比如说，新建订单用例主要是用户在购物车中，可以对要下单的商品进行选择，同时选择要使用的优惠，在订单计算出来金额之后，确定提交订单

（2）事件流

在执行这个用例的时候，需要执行哪些操作和步骤

步骤1：选择要订购的商品

步骤2：对商品的库存进行校验

步骤3：选择这个订单要使用的优惠是什么

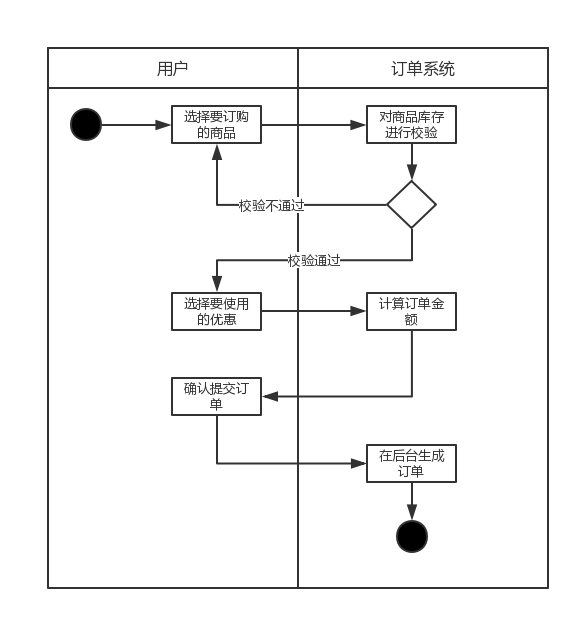
步骤4：计算出来订单的最终金额

步骤5：点击确认提交订单

步骤6：生成订单

（3）事件流图（processon画泳道图）

将上面写好的事件流画成事件流图，一般用泳道图来表示



#### 3.1.3.2 用例2描述

#### 3.1.3.3 用例3描述

# 4、领域类图

领域类图，一般就是从系统间的调用和流转，使用的核心数据类

订单类，就是一个典型的领域类

一般来说，领域类如何抽象，其实就是将需求文档中比较核心的那些数据抽象成领域类，比如订单，库存调拨单，出库单，入库单

这边是可以将一个较为复杂的领域类图拆分为多个领域类，比如说订单主信息，订单项信息，订单优惠信息

还是这个从实践中来，如果说希望加快项目的进度，不要在写文档上耗费过多的时间，领域类图有一个实践，就是只要将核心的领域类与领域类之间的关系画出来就可以了，不需要给出这个领域类的字段。因为可能需求文档中已经将领域类的字段基本都给出来了。不需要再这个环节去确认这个字段。只是确认系统里核心的一些领域类。具体的字段可以参考需求文档。

在这个环节，要不要做的很仔细，考虑如何将一个领域类，比如说，商品领域类，拆分成商品基础信息，商品价格信息，一般建议不需要，因为这个环节里，主要还是面向需求的。

领域类图将每个子系统中，核心的领域数据，描述出来，让我们看一眼大概知道即可

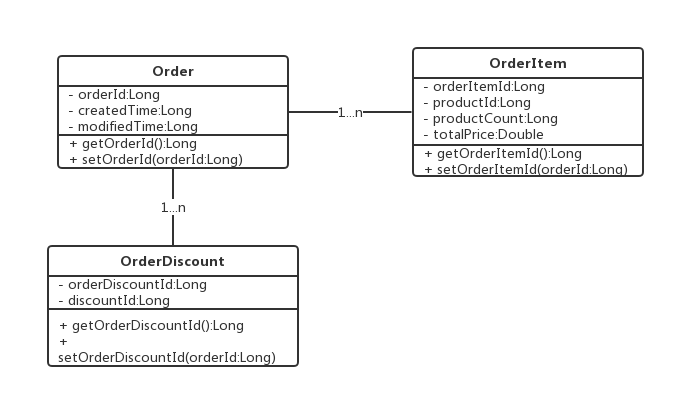
## 4.1 XX子系统

public：+

private：-

protected：#

package：~

853769620