**摘要:** 【问题】 软件设计模式（Software Design Pattern）是一套被反复使用的、多数人知晓的、经过分类编目的、代码设计经验的总结。使用设计模式是为了重用代码以提高编码效率、增加代码的可理解性、保证代码的可靠性。软件设计模式是软件开发中的最佳实践之一，它经常被软件开发人员在面向对象软件开发过程中所采用。项目中合理地运用设计模式可以完美地解决很多问题，每种模式在实际应用中都有相应的原型与之相对，每种模式都描述了一个在软件开发中不断重复发生的问题，以及对应该原型问题的核心解决方案。 请围绕“论软件设计模式及其应用”论题，依次从以下三个方面进行论述。 1．概要叙述你参与分析和开发的软件系统，以及你在项目中所担任的主要工作。 2．说明常用的软件设计模式有哪几类？阐述每种类型特点及其所包含的设计模式。 3．详细说明你所参与的软件系统开发项目中，采用了哪些软件设计模式，具体实施效果如何。 【摘要】 2019年10月，本人所在保险公司启动了超级销售APP项目，该项目通过运用先进的销售工具、客户管理、营销活动管理等功能以达到提升销售人员的效能，加大业务驱动的目标。在该项目中我担任系统架构师，负责系统的架构设计工作。本文以该项目为例，主要论述了软件设计模式在开发中的具体应用。通过抽象工厂模式，实现出单流程车+人联合销售模块间的对象解耦，符合设计模式迪米特法则；通过外观模式，实现保费支付实名认证生物识别、手机号校验隐藏系统的复杂性，系统中的接口提供一致的界面；通过策略模式，实现客户营销基于大数据客户画像，达到不同活动方案可以自由切换。设计模式提高系统的可复用性、扩展性和开放性。基于以上技术应用，项目成功上线，获得用户一致好评。

本人所在的保险公司分支机构遍布全国，已设立分公司36家，机构总数超过2100家，营业机构覆盖全国各个省份，系统员工人数超6万人。因保险生态体系的变革，各保险公司都在积极科技转型，公司基于新业态发展通过“线上化、数字化、智能化”加速推进“三新三聚焦”的战略转型。故启动了超级销售APP项目建设，本项目旨在建设业界领先的面向营销员的、具有前瞻性和可扩展性的，符合主流技术的保险销售一体化平台，聚焦核心作业功能，体现支持、服务、提效和赋能。系统主要实现功能车险、非车险出单、业绩管理、客户管理、营销活动、商业计划书、续保管理等。通过两个视角挖掘，营销员视角，集获客、展业、服务、个人成长为一体，作业辅导始终伴随的创新工作模式、挖掘潜在销售机会，提高工作效率，促进职业能力发展；管理视角，综合管理招募、培训、业绩、活动、提供营销指导和线索及客户服务锦囊，降低消息传递成本、提升营销员团队整体产能和绩效。  
该项目于2019年10月正式启动，我担任系统架构师角色，负责系统总体架构设计工作。在系统实现过程中，通过面向对象的分析和设计方案，达到了系统的可复用性和扩展性。  
设计模式代表了最佳的实践，是开发人员在软件开发过程中面临的一般问题的解决方案。这些解决方案经过相当长的一段时间的试验和错误总结出来的。在面向对象的软件设计过程中，为了使系统体系架构更加精巧灵活，针对特定问题采用的简洁而易于理解的解决方案。经典的软件设计模式按目的不同可以分为三大类：创建型模式、结构型模式和行为型模式。第一类创建型模式，包含工厂方法、抽象工厂、构建器、原型和单例五种模式，提供在创建对象的同时隐藏创建逻辑的方式。第二类结构型模式包含适配器、桥接、组合、装饰、外观、享元和代理七种模式。解决类和对象的组合，从而获得更大的结构。第三类行为型模式是包含解释器、模板方法、职责链、命令、迭代器、中介者、备忘录、观察者、状态、策略和访问者十一个模式，解决对象之间的通信职责分配。  
在超级销售APP项目的开发过程中，我们综合使用了多种设计模式，本文着重对抽象工厂模式、外观模式、策略模式等三种设计模式在该项目中的具体应用进行介绍。  
一、抽象工厂模式  
在抽象工厂模式中，接口是负责创建一个相关对象的工厂，不需要显式指定它们的类。每个生成的工厂都能按照工厂模式提供对象。在车险出单功能联合销售方案模块，通过车维度信息、客户维度信息等标签信息测算推荐最佳组合销售方案，联合销售的非车方案相对固定，按照设计模式迪米特法则，实体应当尽量少地与其他实体之间发生相互作用，使得系统功能模块相对独立。为了解决这个问题，我采用的方法是定义一个抽象的服务类，在其中声明了一个用来创建每一类组合销售方案的接口，而每一类的具体销售方案都继承一个抽象的销售方案类，通过具体的子类来实现不同的销售方案具体定义，这就是一个典型的抽象工厂模式。首先抽象工厂封装了创建销售方案行为和过程，具体不同的销售方案细节由不同的类实现，出单过程中通过调用工厂类操作实例，解决接口选择的问题。其次具体的工厂类只在初始化时出现一次，使得改变这个销售方案的具体工厂变得容易，把对象的实例化和初始化都封装起来，这样做其实保证了对象解耦。  
二、外观模式  
外观模式是为子系统中的一组接口提供一个一致的界面，外观模式定义了一个高层接口，这个接口使得这一子系统更加容易使用。对于出单后的保费支付根据各地监管要求不同，根据各地区政策实名验证要求提供如活体检测认证、手机验证码、公安联网身份认证等多种方式，为了解决访问复杂系统的内部子系统时的复杂度，简化客户端与之的接口，提供相对一致的界面。综合业务场景分析采用外观模式解决。在系统中定义每个具体验证方法的实现类，通过facade对象，将这些具体的功能组件对象进行一次性封装，达到对用户进行了屏蔽。用户只需要访问Facade类，即可以将请求发送给对应的子系统类，完成逻辑处理并把结果返回给用户，最终实现了子系统内部各个组件与用户之间的松耦合关系。用户无需关心组件内部具体的实现逻辑，在未来组件的功能发生变化时也不会对用户产生影响。通过外观模式有效解决了系统相互依赖、提高灵活性、安全性。  
三、策略模式  
在策略模式中，我们创建表示各种策略的对象和一个行为随着策略对象改变而改变。客户管理的增值服务，基于大数据的客户画像分析千人千面，针对不同的客户喜好送不同权益、或参与抽奖等。如果传统模式在有多种算法相似的情况下，使用IF…else实现将带来复杂和难以维护的代价，同时活动在越来越多情况下，开发人员对代码的掌控难道也会增大，可能修改其中一点会影响到其他的人活动正常使用。基于上述问题采用策略模式，创建服务接口类，权益及各种活动基于服务接口类进行具体实现细节，Context类构造函数初始化基于接口的对象类提供对外访问，客户的增值服务功能模块通过Context类达到服务的自有切换。如果业务的活动发生变化对于代码的调整不需要改变原有代码接口只需要实现服务类接口即可达到服务的新增。通过策略模式的活动方案可以自由切换，避免使用多重条件判断，极大的增加了系统的扩展性。  
整个项目历时9个月的实施，于2020年7月完成验收并顺利上线，日均出单保费规模达到千万级别，赢得了良好的用户口碑也在业界内树立了标杆。  
在整个系统的分析设计过程中，我始终秉承面向对象的设计理念，对不同的用例进行全面的分析和考虑，结合自身的工作经验，采用了多种设计模式来支撑系统的实施工作。实践证明软件设计模式在面向对象的设计过程中发挥了不可或缺的作用，取得了满足业务要求的项目成果。同时我也积累了新的经验教训，例如在使用外观模式时，每个具体子类都对接口方法进行了重载不符合开闭原则，如果要修改流程很麻烦，进而带来了维护的更多困难。为此解决问题和提高代码的质量是我们需要在设计模式和反设计模式之间找到平衡点，满足项目实际要求。谨以本文浅显地讨论了软件设计模式及在项目中的实际应用，希望能不断分析总结，提高面向对象软件设计水平和能力。