## 论基于架构的软件设计方法（ABSD）及应用

基于架构的软件设计（Architecture-Based Software Design, ABSD）方法以构成软件架构的商业、质量和功能需求等要素来驱动整个软件开发过程。ABSD是一个自顶向下，递归细化的软件开发方法，它以软件系统功能的分解为基础，通过选择架构风格实现质量和商业需求，并强调在架构设计过程中使用软件架构模板。采用ABSD方法，设计活动可以从项目总体功能框架明确后就开始，因此该方法特别适用于开发一些不能预先决定所有需求的软件系统，如软件产品线系统或长生命周期系统等，也可为需求不能在短时间内明确的软件项目提供指导。

请围绕“基于架构的软件开发方法及应用”论题，依次从以下三个方面进行论述。

1. 概要叙述你参与开发的、采用ABSD方法的软件项目以及你在其中所承担的主要工作。

2. 结合项目实际，详细说明采用ABSD方法进行软件开发时，需要经历哪些开发阶段？每个阶段包括哪些主要活动？

3. 阐述你在软件开发的过程中都遇到了哪些实际问题及解决方法。

**范例**

摘要部分：

本文以某银行统一收单平台项目为例，主要论述了ABSD方法在该项目中的具体应用。2020年1月，我参与了某银行新一代统一收单平台的研发，该系统主要包含商户管理、风险控制、数据分析等功能模块，我在该项目中担任系统架构师，主要负责整体架构方案设计。系统整体采用了基于架构的软件设计方法。在架构需求阶段，通过需求调研形成需求库，通过生成类图、分组打包的方式对构件进行标识，为后续设计打下基础；在架构设计阶段，基于已识别出的构件需求，分析构件在架构设计中的映射以及各个构件间的相互作用，确定了构件清单并完成设计评审；在架构实现阶段，对每个构件的实现方式进行具体分析，完成构件的开发工作并组装以满足系统功能。通过以上技术应用，项目最终顺利上线，获得了银行方以及其收单商户的一致好评。

正文部分：

近年来，我国线上支付业务呈现爆发式的增长，各个存在小额高频交易场景的行业都需要支持微信/支付宝等支付方式。作为提供支付能力的某银行机构，其业务系统也亟需升级为可支撑更多业务场景的综合性收单系统。

2020年1月，我参加了某银行新一代统一收单平台的项目工作，该项目耗时9个月，共投入20余人参与项目实施，配备了42台服务器以实现双中心高可用，我在项目中担任系统架构师一职，主要负责平台整体架构设计。该平台是一个集商户管理、订单管理、支付通道管理、支付风险控制、数据分析等功能于一体的综合性收单平台，为银行服务的各类商户提供可灵活配置的支付方式，包括微信/支付宝扫码支付、APP/公众号/小程序支付等，并对支付业务配套所需的信息进行统一管理。其中订单管理模块主要包括订单预下单、发起支付、撤单与退货等接口；支付通道管理模块主要为各商户进行支付通道、手续费率等参数的配置；数据分析模块主要为业务人员提供各收单商户的存贷款和流水情况统计，以便制定符合商户经营情况的优惠政策与营销方案。因平台功能点多，需求分析周期长，我们使用基于架构的软件设计方法有效提高了设计的效率与质量。

基于架构的软件设计方法将软件开发过程划分为架构需求、架构设计、架构文档化、架构复审、架构实现、架构演化6个子过程,很好地支持了软件重用。架构需求阶段，需要从需求库中提取需求并标识出系统所需构件，形成架构需求进行评审；架构设计阶段，根据架构需求对构件进行映射与关联关系分析，形成架构设计方案并评审；架构文档化阶段，主要输出架构规格说明书以及质量属性测试手册；架构复审需要组织技术专家对架构设计进行再一次审核；架构实现阶段通过构件的开发形成构件库，以进一步实现架构方案；系统上线后，基于业务需求的变更以及构件的不断更新迭代，对架构设计方案进一步演化迭代升级。下面着重阐述架构需求、架构设计、架构实现三个阶段在统一收单平台项目实践过程中的具体情况。

一、架构需求阶段

为提高系统设计效率，项目初期我们对收单归口业务部门及原收单平台进行了大量的需求调研及系统分析，获取关键用户对目标系统功能、行为、性能以及设计约束等方面的期望，形成了业务需求库，并基于此需求库开展构件标识工作。首先我们根据需求描述，在构件库中寻找能够满足需求的可重用构件，如统一收单平台商户登录页面需要使用到密码键盘，该构件对键盘组件安全性要求较高，我们直接复用了银行网银页面成熟的密码键盘组件，有效提高了开发效率；其次我们将无法复用已有构件的需求进行分组打包，如订单通知需求，我们将微信渠道的通知与支付宝渠道的通知功能分为一组，将其共性的模块提取出来并进行构件化，通过这两方法将构件清晰地标识了出来。获取构件清单后，我们组织项目成员、银行技术部门专家对系统架构需求以专家经验判断以及头脑风暴的方式进行了评审，对需求的优先级以及重要程度进行了初步排序，基于全行系统架构的角度排除了部分非本系统功能范围的需求内容，最终形成了系统需求基线，为后续项目工作打下了坚定的基础。

二、架构设计阶段

标识出构件清单并完成评审后，我们对系统进行了架构设计分析并初步决定采用三层B/S架构来满足业务需求。为验证此架构方案的可行性，我们将已标识的构件与架构模型进行映射匹配，确定各个构件在模型中的位置以及他们之间的关联关系。如上文提到的密码键盘组件，该组件直接面向客户并与之交互，我们将其部署在表示层，而且所使用到的加密算法，需要通过逻辑层加密模块与银行加密机进行交互，两个模块之间存在通讯关系，在架构设计中必须考虑其跨层通讯的可行性以及性能问题，同时也需要考虑不同层次之间交互可能产生的安全性问题。为此我们在表示层与逻辑层之间通过防火墙进行权限控制，并将其通讯协议通过SSL进行加密保护以提高安全性；同时我们对加密模块进行了高可用多节点部署，并对多节点进行了负载均衡配置，提高了可用性以及性能。完成架构方案设计后，我们组织技术专家通过ATAM方法，基于场景对系统架构的安全性、可用性、可修改性等关键质量属性进行了评估，均满足预期，最终架构方案设计通过了评审。

三、架构实现阶段

完成架构设计后，我们又开展架构文档化以及架构复审工作，输出了架构规格说明书等文档，项目组据此进行架构实现。我们根据架构设计从构件库提取已有构件或开发新构件来制作构件库。在构建开发方面，基于平台采用JAVA开发的实际情况，我们采用了EJB技术对构件进行开发与组装，该技术使构件可以很好地与平台整体运行环境相融合，减少构件组装失配的风险。我们将不同的实体、功能制作成相应的Java Bean，当开发者在处理不同的任务时根据场景提取使用即可。完成了构件的组装，最后我们对系统开展了测试工作，通过LoadRunner在客户下单的场景收集线性测试用例，对系统发起高并发请求，并使用nmon等工具监控服务器运行状态；通过nessus对服务器、数据库以及各中间件进行基线扫描，查找安全漏洞；通过应急演练、数据中心切换来测试系统架构的可用性，统计RTO以及RPO指标；通过增加服务器配置、节点数量来验证架构的可拓展性。统一收单平台的架构方案在此阶段得到了很好的实现，同时通过一系列测试工作，也证明了该架构的先进性。

统一收单平台项目历时9个月，并最终于2020年10月上线运行，这与基于架构的软件设计方法密不可分。随后平台经历了国庆、双十一、春节期间超高频交易的重大考验，取得了很好的效果，银行方以及商户对平台的功能、性能均给予了高度的评价。

通过这次项目的实施，我也深刻认识到好的软件开发方法对于整个软件开发生命周期至关重要。统一收单平台也正是借助了ABSD方法，才能够在短期内以先进的架构方案来实现庞大的业务需求。作为架构师我在项目中采用了多种方法来推动架构落地，比如在架构设计阶段，项目组需要对架构方案进行评估。我推动项目组使用ATAM架构评估方法，该方法可以通过场景对架构性能、实用性、安全性进行有效地评估，因此我们也初步论证了架构设计方案可行。同时也采取了一些无效的手段，比如在架构需求阶段，我们采用了面对面访谈的方式来获取需求，然而因为收单平台所面临的商户数量庞大，类型丰富多样，业务部门产品经理岗位也经历过多人轮换，通过访谈的方式不仅工作量大，并且很难获取到完整的系统需求，同时也影响了整体的架构设计。无论是ATAM架构评估方法，还是面对面需求访谈，都是优秀的项目管理手段，运用在正确的场景下能够高效的推动项目实施，是作为架构师必须掌握的技能，未来我将根据不同的系统情况来选取最为合理的方式应用到项目过程中。

## 论SOA在企业集成架构设计中的应用

企业应用集成（Enterprise Application Integration，EAI）是每个企业都必须要面对的实际问题。面向服务的企业应用集成是一种基于面向服务体系结构（Service-Oriented Architecture，SOA）的新型企业应用集成技术，强调将企业和组织内部的资源和业务功能暴露为服务，实现资源共享和系统之间的互操作性，并支持快速地将新的应用以服务的形式加入到已有的集成环境中，增强企业IT环境的灵活性。

请围绕“SOA在企业集成架构设计中的应用”论题，依次从以下3个方面进行论述。

1、概要叙述你参与管理和实施的企业应用集成项目及你在其中所担任的主要工作。

2、具体论述SOA架构的内容、特点，以及你熟悉的工具和环境对SOA的支持，在应用中重点解决了哪些问题。

3、通过你的切身实践详细论述SOA在企业应用集成中发挥的作用和优势。

**范例**

摘要部分：

2016年8月，我参与了胶凝砂砾石坝施工质量监控系统的开发工作，该系统旨在帮助水利工程建设法人单位、施工企业、监理机构及相关政府部门解决水利工程建设施工质量监控和工程项目管理等问题。我在该项目中担任该项目的技术负责人，主要负责该系统的技术方案制定及技术指导。本文以胶凝砂砾石坝施工质量监控系统为例，主要论述了SOA在企业集成架构设计中的具体应用。服务提供者主要完成服务的设计、描述、定义和发布等相关工作；服务注册中心保证该系统各个模块、服务的相互独立性与松耦合；服务请求者通过WebService技术调用服务。实践证明，通过以上技术的应用有效实现了资源共享和系统间的互操作性，提高了系统的灵活性，最终系统顺利上线，获得用户一致好评。

正文部分：

胶凝砂砾石坝是在面板坝和碾压混凝土重力坝基础上发展起来的一种新坝型，其特点是采用胶凝砂砾石材料筑坝，使用高效率的土石方运输机械和压实机械施工。与常规坝型相比，胶凝砂砾石坝在适用性和经济性方面具有独特的优势，可以就地、就近取材，不需设置集料筛分，施工进度快，施工工序简单高效，因而要求施工过程紧凑，高峰期筑坝效率要求高，这给施工质量控制带来了一定的困难和风险，需要综合考虑影响施工质量的各方面因素，尽量采用自动化监控手段，加强实时质量监控力度，这使胶凝砂砾石坝施工质量监控系统应运而生。

2016年8月，我参与了胶凝砂砾石坝施工质量监控系统的开发工作，担任该项目的技术负责人，主要负责该系统的技术方案制定及技术指导。该系统的主要功能模块包括采料监控、运料监控、拌合监控、碾压监控和温湿度监控等。旨在帮助水利工程建设法人单位、施工企业、监理机构及相关政府部门，解决水利工程建设施工质量监控和工程项目管理等问题，通过信息技术和施工信息现场采集、实时传输、统一存储、科学分析和在线处理，及时生成质量监控报表和发布质量预警信息，提高水利工程建设管理科学化、现代化和信息化，落实法人负责、监理控制、施工保证、政府监督等各项职能。因此，要满足该系统的需求，选择一种合适的架构技术至关重要。

SOA是一种应用程序架构，在这种架构中，所有功能都定义为独立的服务，服务之间通过交互和协调完成业务的整体逻辑。SOA指定了一组实体，包括服务提供者、服务消费者、服务注册表、服务条款、服务代理和服务契约，这些实体详细说明了如何提供和消费服务。服务提供者提供符合契约的服务，并将他们发布到服务代理。这些服务是自我包含的、无状态的实体，可以由多个组件组成。服务代理者作为存储库、目录库或票据交换所，产生由服务提供者发布的事先定义的标准化接口，使得服务可以提供给在任何异构平台和任何用户接口使用。这种松散耦合和跨技术实现，使各服务在交互过程中无需考虑双方的内部实现细节、实现技术、以及部署在什么平台上，服务消费者只需要提出服务请求，就可以发现并调用其他的软件服务得到答案。SOA作为一种粗粒度、松耦合的架构，具有松散耦合、粗粒度服务、标准化的接口、位置和传输协议透明、服务的封装和重用、服务的互操作等几个特点。

该系统要求开发周期短，系统灵活性高等，结合SOA的特点，我们最终采用了面向服务的、基于SOA的企业应用集成。下面具体论述其应用过程。

1、服务提供者

服务提供者主要完成服务的设计、描述、定义和发布等相关工作。经过对水利行业施工工程及施工工艺的深入研究，通过查阅《胶凝砂砾石坝施工指南》等相关资料，根据企业应用集成的要求，对胶凝砂砾石坝施工质量监控系统的业务流程进行梳理；综合考虑服务粗粒度、松耦合、自包含和模块化等特点进行服务的设计。为了避免服务通信期间，信息量过大，服务之间交互过于频繁，尽量的减少了服务的数量。同时，为了保证服务自身功能的完整性，尽可能的减少服务与系统之间的通信，在胶凝砂砾石坝施工质量监控系统的分析与开发过程中，先行设计，提取出了两个必要，急需的服务便于日后集成使用，其中包括拌合监控中标准拌合比对比服务和碾压监控中的碾压轨迹生成服务。

在标准拌合比对比服务中主要实现针对现有拌合配比与标准拌合比的对比，以判断现有拌合配比是否符合标准的工作。由于胶凝砂砾石坝就地、就近取材的特性，因此在不同的水利施工工地所使用的采料也不尽相同，标准拌合比对比服务预留了标准拌合比的输入接口，以适应不同的需求。在碾压轨迹生成服务中主要实现读取定位信息绘制碾压轨迹，以监控是否存在漏碾和欠碾的情况。由于受到胶凝砂砾石坝选址和机密程度的限制，定位信息可以选择GPS或者超宽带技术，但是两种定位的方式的数据格式并不相同，因此碾压轨迹生成服务的开发中预留了两种数据格式的接口来读取定位信息。待完成服务设计之后，服务提供者采用WSDL进行服务描述，而后再利用UDDI技术将这些服务信息发布至服务注册中心，公布查找和定位服务的方法。

2、服务注册中心和服务请求者

在胶凝砂砾石坝施工质量监控系统采用了服务注册中心。服务注册中心比不是一个必选角色，但是为了保证该系统各个模块、服务的相互独立性与松耦合，在该系统中依然保留了服务注册中心。同时，服务注册中心的存在也使得服务请求者与服务提供者之间进一步解耦。在服务注册中心包含有已发布的标准拌合比对比服务与碾压监控中的碾压轨迹生成服务的描述信息，其描述信息主要包括服务功能描述、参数描述、接口定义、信息传递等相关信息。

服务请求者通过WebService技术调用服务。当服务完成发布，在服务请求者要使用已发布的服务。利用Web Service技术在拌合监控阶段，通过服务注册中心获取拌合监控中标准拌合比对比服务的相关功能，接口，参数及其返回值等相关服务信息；之后使用Web Service技术传递服务所需的标准拌合比等相关参数，进而调用该服务相关的运算、处理和分析。利用Web Service技术在拌合监控阶段，通过服务注册中心获取实时施工数据采集处理服务的服务定义和功能，接口，参数及其返回值等相关服务信息；之后根据施工工地的具体情况选择不同的定位方式，传递服务所需相关参数，最后实时施工数据采集处理服务运行结束返回的绘制碾压轨迹坐标点同样是利用Web Service技术传递至服务请求者。服务请求者接收到碾压轨迹的坐标点后最终完成碾压轨迹的绘制工作并在界面中将其呈现出来。在这期间服务请求者无需了解服务是如何对数据进行处理和分析的。

整个项目历时10个月开发，于2017年6月完成交付，到目前运行稳定。通过在水利施工工地等恶劣环境下的一段时间的使用，用户普遍反馈良好。总体来讲，选用SOA有如下优势：1、系统更易维护。当需求发生变化时，不需要修改提供业务服务接口，只需要调整业务服务流程或者修改操作即可。2、更高的可用性。该特点是在于服务提供者和服务请求者的松散耦合关系上得以发挥与体现。这种没有绑定在特定实现上、具有中立的接口定义的特征称为服务之间的松耦合。松耦合有两个明显的优势，一是它的灵活性，其独立于实现服务的硬件平台、操作系统和编程语言；二是当组成整个应用程序的每个服务的内部结构和实现逐渐地发生改变时，它能够继续存在。3、更好的伸缩性，依靠业务服务设计、开发和部署等所采用的架构模型实现伸缩性。使得服务提供者可以互相彼此独立地进行调整，以满足新的服务需求。

实践证明，SOA技术的使用大大提高了系统开发效率，节省了开发和维护成本，使得系统具有更好的开放性、易扩展性和可维护性，从项目完工后的使用效果来看，达到了预期目的。

## 论软件系统架构风格

系统架构风格（System Architecture Style）是描述某一特定应用领域中系统组织方式的惯用模式。架构风格定义了一个词汇表和一组约束，词汇表中包含一些构件和连接件类型，而这组约束指出系统是如何将这些构件和连接件组合起来的。软件系统架构风格反映了领域中众多软件系统所共有的结构和语义特性，并指导如何将各个模块和子系统有效地组织成一个完整的系统。软件系统架构风格的共有部分可以使得不同系统共享同一个实现代码，系统能够按照常用的、规范化的方式来组织，便于不同设计者很容易地理解系统架构。

请以“软件系统架构风格”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开发项目以及你所担任的主要工作。

2．分析软件系统开发中常用的软件系统架构风格有哪些？详细阐述每种风格的具体含义。

3．详细说明在你所参与的软件系统开发项目中，采用了哪种软件系统架构风格，具体实施效果如何。

**范例**

摘要部分：

本人于2016年1月参与浙江省某市公交集团“公交车联网一体化”项目，该系统为新能源营运车辆补贴监管、安全监控等方面提供全方位的软件支撑，在该项目组中我担任系统架构师岗位，主要负责整体架构设计与中间件选型。本文以该车联网项目为例，主要讨论了软件架构风格在该项目中的具体应用。底层架构风格我们采用了虚拟机风格中的解释器，因该公交共有几十种不同的数据协议，使用解释器风格可以满足整车数据协议兼容性需求；中间层关于应用层的数据流转我们采用了独立构件风格中的隐式调用，这种风格主要用于减低系统间耦合度、简化软件架构，提高可修改性方面的架构属性；应用系统层我们采用了B/S的架构风格，统一解决公交行业性难题“实施推广难、维护难”问题。最终项目成功上线，获得用户一致好评。

正文部分：

随着国家十三五计划中-能源战略的深入和推广，该市公交集团自2016年1月起全面停止采购燃油机公交车，规划到2020年纯电公交车采购占比必须在70%以上，同时配套将车联网方面的系统建设被列为工作重点。不管从新能源营运车辆补贴监管、安全监控或者公交公司自身的营运和机护需求，都要求有新的车联网系统对他们进行全方位的支持，而我司是该公交的主要仪表与can模块产品的主要供应商，全市4000多台车中有3000多辆是我司的产品，我司不仅掌握熟悉该公交整车数据而且在车联网底层can数据有非常明显的领域知识优势，因此2016年1月我司被该市公交集团委托建设公交集团车联网一体化项目。本项目组全体成员共有27人（不含业主方），我在项目中担任系统架构师职务，架构小组共4人，我主要职责负责整体架构设计与中间件选型，4月份完成架构工作，整个项目共耗时了7个月，2016年8月顺利通过验收。

在架构工作开始阶段，我们便意识到，架构风格是一组设计原则，是能够提供抽象框架模式，可以为我们的项目提供通用解决方案的，这种能够极大提高软件设计的重用的方法加快我们的建设进程，因此在我司总工程师的建议下，我们使用了虚拟机风格、独立构件风格以及B/S架构风格这三种较常用风格。虚拟机风格中的解释器架构风格能够提供灵活的解析引擎，这类风格非常适用于复杂流程的处理。独立构件风格包括进程通讯风格与隐式调用风格，我们为了简化架构复杂度采用了隐式调用风格，通过消息订阅和发布控制系统间信息交互，不仅能减低系统耦合度，而且还提高架构的可修改性。B/S架构风格是基于浏览器和服务器的软件架构，它主要使用http协议进行通信和交互，简化客户端的工作，最终减低了系统推广和维护的难度，以下正文将重点描述架构风格的实施过程和效果。

底层架构我们使用解释器风格来满足整车数据协议兼容性需求。解释器风格是虚拟机风格中的一种，具备良好的灵活性，在本项目中我们的架构设计需要兼容好86种不同can数据协议，一般来说这种软件编写难度非常高，代码维护难度压力也很大，因此这个解释器的设计任务便很明确了，软件设计需要高度抽象、协议的适配由配置文件来承担。具体的做法如下，我们对各个车厂的can数据结构进行了高度抽象，由于can数据由很多数据帧组成，每个数据帧容量固定并且标识和数据有明确规定，因此我们将can协议中的ID和数据进行关系建模，将整体协议标识作为一个根节点，以canid作为根节点下的叶子节点，使用XML的数据结构映射成了有整个协议链-数据帧-数据字节-数据位这4层的数据结构，核心的代码采用jdom.jar与java的反射机制动态生成java对象，搭建一套可以基于可变模板的解释器，协议模板的产生可以由公交公司提供的excel协议文档进行转换得到，解释器支持协议模板热部署，这种可以将二进制数据直接映射成java的可序列化对象，将数据协议的复杂度简化，后期数据协议更改不会对软件产生影响，仅仅更改协议模板文件即可，最终我们使用了86个协议描述文件便兼容了这些复杂的can数据协议，规避了can数据巨大差异带来的技术风险。

中间层我们使用独立构件风格中的隐式调用来简化构件间的交互复杂度，降低系统耦合度。主要的实现手段是，我们采用了一个开源的消息中间件作为连接构件，这个构件是apache基金会下的核心开源项目activemq，它是一款消息服务器，其性能和稳定性久经考验。由上文提到的解释器解析出对象化数据经过activemq分发到各个订阅此消息的应用系统，这些应用系统包括运营指挥调度、自动化机护、新能源电池安全监控等，这种多web应用的情况非常适合采用消息发布与消息订阅的机制，能够有效解决耦合问题，我们在编码的过程中发现只要采用这种风格的web应用，整个迭代过程效率极高，错误率降低，而且我们使用的spring框架，消息队列的管理完全基于配置，清晰简单，维护性良好，例如整车安全主题、运营调度主题、机护维修主题等消息队列分类清晰，可以随时修改其结构也能够随时增加其他主题的消息队列，不同的web系统监听的队列也可以随时变换组合，基于消息中间件的架构设计能够让系统的构件化思路得到良好实施，总体来说这种架构风格带来了非常清晰的数据流转架构，简化了编码难度，减低本项目的二次开发的难度。

应用系统层我们主要采用B/S的架构风格，主要用于解决公交推广难、维护难的问题。公交行业有一个明显的特点，公交子公司分布在全市各个地区，路途很远，且都是内网通讯，车联网络也是走的APN专网，一般是无法远程支持的，这给我们的系统推广以及后期维护带来了很大难题，我们可以想象如果使用C/S架构，更新客户端一旦遇到问题很可能需要全市各个站点跑一遍。这让我们在系统推广和维护方面面临较大压力。我们采用的B/S架构风格能够解决这个难题，并充分考量可现在的相关技术成熟度，例如现在的html5完全能够实现以前客户端的功能，项目中我们使用了大量的前端缓存技术与websocket技术，能够满足公交用户实时性交互等需求。这种风格中页面和逻辑处理存储在web服务器上，维护和软件升级只要更新服务器端即可，及时生效，用户体验较好，例如界面上需要优化，改一下Javascript脚本或者CSS文件就可以马上看到效果了。

项目于2016年8月完成验收，这 1年内共经历了2次大批量新购公交车辆接入，这几次接入过程平稳顺利，其中协议解释器软件性能没有出现过问题，消息中间件的性能经过多次调优吞吐量也接近了硬盘IO极限，满足当前的消息交互总量，另外由于我们的项目多次紧急状态下能够快速适应can协议变动，得到过业主的邮件表扬。除了业主机房几次突发性的网络故障外，项目至今还未有重大的生产事故，项目组现在留1个开发人员和1个售后在维护，系统的维护量是可控的，系统运行也比较稳定。

不足之处有两个方面，第一在架构设计的过程中我们忽略PC配置，个别PC因为需要兼容老的应用软件不允许系统升级，这些电脑系统老旧，其浏览器不支持html5，导致了系统推广障碍。第二在系统容灾方面还有待改善。针对第一种问题，我们通过技术研讨会说服业主新购PC，采用两台机器同时使用方式解决。针对第二种问题我方采用了服务器冗余和心跳监测等策略，在一台服务暂停的情况下，另外一台服务接管，以增加可用性。

## 论数据访问层设计技术及其应用

在信息系统的开发与建设中，分层设计是一种常见的架构设计方法，区分层次的目的是为了实现“高内聚低耦合”的思想。分层设计能有效简化系统复杂性，使设计结构清晰，便于提高复用能力和产品维护能力。一种常见的层次划分模型是将信息系统分为表现层、业务逻辑层和数据访问层。信息系统一般以数据为中心，数据访问层的设计是系统设计中的重要内容。数据访问层需要针对需求，提供对数据源读写的访问接口；在保障性能的前提下，数据访问层应具有良好的封装性、可移植性，以及数据库无关性。

请围绕“论数据访问层设计技术及其应用”论题，依次从以下三个方面进行论述。

1．概要叙述你参与管理和开发的与数据访问层设计有关的软件项目，以及你在其中所担任的主要工作。

2．详细论述常见的数据访问层设计技术及其所包含的主要内容。

3．结合你参与管理和开发的实际项目，具体说明采用了哪种数据访问层设计技术，并叙述具体实施过程以及应用效果。

**范例**

摘要部分：

2015年8月，我司承接了华为公司ERP体系下的仓储拣料系统的项目，该系统主要实现将上游系统的拣料任务令转化成具体的拣料任务及拣料路径并下发给下游系统进行拣料工作，主要包括非标、加工、零星、转库和电子流等功能模块。我在该项目中担任技术负责人，主要负责该项目技术方案的制定及技术指导工作。本文以仓储拣料系统为例，主要论述了数据访问层技术在该项目的具体应用。首先在技术选型上，综合需求层面及技术特点的考虑，确定了Mybatis技术；接着通过导入资源包、配置核心文件及配置映射文件三大步骤来具体进行数据访问层的开发；最后针对业务执行层面、一二级缓存的配置及数据量大的查询等方面进行性能调优。通过以上技术应用，项目最终顺利上线，达成客户期望，获得用户一致认可。

正文部分：

华为公司作为世界500强企业之一，每年在制造业的产值数以亿计，其高效产能的根本原因便在于其制造流程的规范化。而支持华为制造业规范化的，正是华为公司主导研发的企业资源计划管理系统（ERP）。ERP体系将制造业务中的规划、拣料、运输、生产等相关工作进行了严谨精准的管控。

2015年8月，我司承接了华为ERP项目体系中的仓储拣料系统（以下简称SPS）的建设。SPS系统作为ERP体系的子系统，包括非标、加工、零星、电子流和转库等十几种任务管理模块。主要实现从上游系统（如APA系统、DMS系统、IMALL系统等）下载对应的拣料任务令，根据拣料任务的类型、交付对象等相关属性信息对拣料任务令进行汇总打包、拆分、计算出最佳交付路径后生成拣料任务并发送给下游的仓储系统，最后根据下游系统反馈的拣料任务的完成情况，对ERP中的物料信息进行回写核销，以保持上下游系统数据的一致性。SPS系统拥有高达8万的用户量，且存储着数以千万计的大量数据。因SPS系统本身所处的位置，其数据的操作处理速度直接影响整个ERP体系的运转效率，因此如何使用数据访问层技术来满足本项目的功能及性能需求成了我们考虑的技术问题之一。

数据访问层技术也称持久层技术，主要负责将系统数据通过调用数据库的SQL语句对数据进行读取和存储。java语言的JDK中自带一种数据访问层技术——JDBC。市面上主流的数据库均实现了JDBC中所定义的对数据库访问的各类接口，因此JDBC能够较为高效的完成对各类数据库的数据存取操作。但是，单纯的使用JDBC进行开发，效率低下的同时会增加人力成本的支出。基于这种情况，Hibernate和Mybatis技术应运而生。Hibernate是一个对JDBC进行轻量级封装的全自动框架，能够实现数据库道POJO的完整映射，开发人员无需编写复杂的SQL，仅需对POJO进行操作便可同步处理数据库中的数据。而Mybatis相对Hibernate而言则是一个半自动的框架，可以支持自定义的SQL语句的操作，相对于复杂的数据查询操作支持性更佳。

针对SPS项目，如何应用数据访问层技术，是SPS项目组所关注的问题。总体来讲，我们将这个问题分成了三个阶段来处理，分别是技术选型、技术应用以及性能调优。

技术选型

从SPS的整体层次上来讲，为了保障SPS系统的处理拣料数据的效率，我们将九成以上的事务处理放到了数据库层面进行执行，也就是通过数据库层面的存储过程来完成事务的处理，因为数据库本身的事务处理效率比起Spring框架中的事务机制要更加高效，且彼此间的网络开销会更低。而作为上层的JavaWeb工程，主要是为用户提供对应的查询检索机制，主要包括两大块：其一，将每一种拣料任务的完成进度为用户提供查询的接口；其二，为用户提供有关拣料任务的各种类型的复杂报表，报表的关联性查询较多。Hibernate本身的优势在于对数据的更新处理可通过对POJO的处理直接映射到对应的数据表/视图中，由于SPS系统的事务处理并不在Java层面，显然在这个系统中作用不大；另外Hibernate是一个全自动框架，对自定义的SQL查询支持性并不好；而Mybatis相对能够很好的支持自定义SQL的开发，对于SPS系统而言，不仅能够很好的支持各种简单对象的自定义查询，在复杂报表的开发支持上较Hibernate更加强大，能够显著的提升开发效率。因此，从SPS系统的需求层面及两种数据访问技术的特点上考虑，Mybatis更适合SPS系统。

技术应用

SPS系统的开发根据拣料指令业务类型，一共分成了十多个JavaWeb工程，每一个Web工程专门负责处理一种拣料指令。所以，在应用Mybatis进行数据访问层开发时，所有的工程都将引用Mybatis的资源jar包。在应用过程中，主要包括三大步骤。首先将Mybatis的jar包资源，导入到一个公共目录中，并将该路径通过IDE开发集成环境工具设置到公共资源库中为各个工程包所引用。其次，为所有的工程包配置Mybatis的一个名为sqlMapConfig.xml的核心配置文件，主要包括整个系统运行时的公共配置信息，例如数据库的基本配置信息（数据库连接路径、驱动、账号及密码）。然后，针对每个模块的开发为其实体类单独配置Java对象；例如SPS系统用户要查询来自APA系统中的计划指令的信息，其实体类为PlanOrderVO，对应的数据访问接口文件为PlanOrderMapper.java，该接口文件内定义了一系列的查询计划指令的方法，则在开发过程中同样需要配置一个名为PlanOrderMapper.xml的映射文件；映射文件中定义的每一个查询语句皆与接口文件的方法保持一致，且通过<resultMap >标签来定义Java对象及其属性与数据库对象及其字段之间的映射关系，同时将自定义的SQL语句在该配置文件中完成。

性能调优

从整个系统的角度上讲，性能涉及到方方面面，但从数据访问层技术的应用上看，调优主要包括以下几个方面。首先是业务执行层面的确定。众所周知，事务级操作在数据库层面比在Java层面要快得多。因此，对于复杂的事务处理，乃至复杂的报表查询，都定义到了数据库层面（存储过程、视图），Java层面只需要负责传参和接受结果即可。其次，对于Mybatis里面的相关设置的应用，也决定了系统处理数据的效率，最典型的，就是一二级缓存的配置。对Mybatis而言，如果配置了缓存，系统会优先在一二级缓存中寻找目标数据，当在一二级缓存中找不到数据时，才会到数据库中进行查询。但是缓存的配置因为缓存的有限容量也存在使用上的讲究。根据二八定律，我们将最常用的上月数据统计报表结果会放入缓存中，避免系统在数据库层面对百万级的数据进行多次的复杂关联查询。最后，对于数据量太大的查询，SQL语句的写法本身也是决定着查询效率的因素。例如，数据库执行SQL语句的过滤条件时，是从后往前执行，因此，如果某个条件能够最大限度的降低数据范围，那么这个条件在SQL语句中应该放在后面。

通过以上技术的应用，项目历经10个月周期，SPS系统最终于2016年6月顺利上线，目前系统运行稳定，获得了用户一致认可。无论是从业务执行层面，还是在面临数据量大的查询，系统的稳定性都达到了预期。

但在该系统实施过程中，其实也发现了Mybatis技术相对于Hibernate技术的不足。如Hibernate拥有可通过配置方言属性（Dialect）屏蔽数据库类型的特点，让用户不用考虑不同数据库的操作方式，但Mybatis因为要支持灵活的SQL语句，导致无法支持方言。当某天系统数据库需要进行切换时，对于Mybatis而言是一个重大问题，届时需要将所有涉及到与数据库的对接写法全部维护，无疑增加了维护的工作量。要真正解决这个问题，恐怕需要从更高的层面，建立统一的数据库标准后才能实现。当然，这也是我们在下一代技术应用中需要去完善的地方。

## 论软件系统架构评估

对于软件系统，尤其是大规模的复杂软件系统来说，软件的系统架构对于确保最终系统的质量具有十分重要的意义，不恰当的系统架构将给项目开发带来高昂的代价和难以避免的灾难。对一个系统架构进行评估，是为了：分析现有架构存在的潜在风险，检验设计中提出的质量需求，在系统被构建之前分析现有系统架构对于系统质量的影响，提出系统架构的改进方案。架构评估是软件开发过程中的重要环节。

请围绕“论软件系统架构评估”论题，依次从以下三个方面进行论述。

1.概要叙述你所参与架构评估的软件系统，以及在评估过程中所担任的主要工作。

2.分析软件系统架构评估中所普遍关注的质量属性有哪些？详细阐述每种质量属性的具体含义。

3.详细说明你所参与的软件系统架构评估中，采用了哪种评估方法，具体实施过程和效果如何。

**范例**

摘要部分：

2018年4月，我参与某市大型车企的车联网开发项目，该项目主要为该车企的用户提供车联网服务，功能包含车控、维保预约、智能家居、音乐电台等。我在该项目担任系统架构师角色，主要负责项目整体的架构设计。本文以该项目为例，介绍软件架构评估在项目中的实际运用。通过对场景和需求收集，包括功能和非功能需求，确立了项目所需要关注的场景和需求。通过对架构视图和场景实现，展现架构的整体思想，分析场景和架构的匹配度，对场景做优先级划分。通过属性模型构造和分析，对单一属性进行分析，对安全性和性能等属性冲突之处进行权衡分析。通过以上实施过程，我们顺利对项目的软件架构进行了评估，为提升项目质量提供了坚实保证。最终项目顺利上线，获得预期目标，得到客户一致好评。

正文部分：

随着汽车行业的快速扩张，以及互联网行业的快速发展，汽车不再仅仅是个移步工具，人们更多赋予了汽车互联网的功能，让车变得更加智能，各大车企车联网的开发也如火如荼进行着。为了提升用户黏性与品牌忠诚度，提升汽车行业市场份额，同时收集用户埋点数据，为后面的决策提供重要数据，某市某知名车企自2018年4月至今实施了车联网开发项目。项目主要开发车联网APP、车机系统，功能包含三大块：灰色服务，即车控、超速报警、电子围栏、健康报告、行程统计等；彩色服务，即智能家居、音乐电台、停车缴费、违章代缴、维保预约等；基础服务，即品牌资讯发布、绑车入会、商城、在线客服等；同时做好线上功能的埋点，实时采集用户的数据，定期对绑车率、激活率等功能做报表分析。项目前后端开发、测试、产品经理等角色合计50余人，集成了斑马、喜马拉雅、酷我、京东、美团、高德地图、中国移动等多家服务提供商，开发周期约一年半。我在项目中担任架构师，负责架构组整体架构需求分析、架构设计、架构评估等工作。

对于大型的系统来说，良好的系统架构至关重要，将直接关系到系统的质量。在软件架构评估中，人们普遍关注的质量属性包括性能、可用性、可修改性、安全性等。性能是指系统的响应能力，即要经过多长时间才能对某个事物做出响应，或者在某段时间内系统所能处理的事件个数。可用性，即系统能够正常运行的时间比例，经常用两次故障之间的时间长度或在故障出现时系统能够恢复正常的速度来表示。安全性，是指系统在向合法用户提供服务的同时能够阻止非授权用户使用的企图或拒绝服务的能力。可修改性，是指能够快速以较高的性价比对系统进行变更的能力。

结合项目实际，我们的软件架构评估方法主要采用了基于场景的评估方法—架构权衡分析法（ATAM），过程主要包含场景与需求收集，架构视图与场景实现，属性模型构造分析，折中四个方面。下面将从重点从前面三个方面介绍ATAM方法在项目中的实际运用。

1、场景和需求收集

由于项目庞大，涉及的系统多，对项目的质量有很高要求，若不提前确认项目的场景和需求，将无法为架构评估提供依据。鉴于此，在架构评估的第一步，我们进行了场景和需求收集。我们召集了项目的关键干系人成立评估小组，包含架构人员、设计人员、管理人员和客户代表，在和他们宣贯软件评估过程和思想以后，收集他们对于场景的设想与需求。场景包括功能性和非功能性需求，功能需求主要按照项目的功能要求确定，非功能需求我们按照项目需求确立了可用性、性能、安全性、可修改性、互操作性、功能性、可变性等，并对非功能需求需要进行细化的描述，包括需求、约束和环境，例如对于性能的要求，需要细化到用户点击某一个页面，在正常网络情况下页面响应时间需要控制在2s之内。在场景收集中，不同角色关注的视角不同，因此我们需要尽可能覆盖到不同类别的人员，以保证场景不会遗漏。通过这个环节，我们收集了全面的场景和需求，为软件架构评估提供了关键的依据。

2、架构视图和场景实现

在场景和需求收集以后，若不与项目的架构结合起来做匹配，也无法针对场景对架构做评估。因此，我们第二步进行了架构视图和场景实现。我们向评估小组展现了项目的整体架构与思想，对不同的场景介绍了架构设计方案。例如对于用户查看行程统计报表的场景，需求是每天24:00更新报表，因此架构设计方案为数据通过跑定时任务进行批处理，而非实时处理。对于用户关键的个人信息，例如手机号、身份证号，场景需求是保证用户敏感数据安全，因此我们的架构设计为用HTTPS对数据通道进行加密，同时APP前端对敏感数据做脱敏处理。通过这样的方式将架构设计与具体场景匹配起来。同时，也建立了质量属性效用树，将涉及的质量属性进行图形化表示，以更加直观展示。评估项目这个环节还需要对场景进行划分，我们召集了评估小组成员对场景的优先级进行投票，对场景进行排序。通过这个方式，使评估小组成员可以对架构和场景的匹配有了很好的认识，也对场景进行了分级处理，为后续的评估工作提供了重要参考。

3、属性模型构造和分析

在获得项目的需求和场景之后，若不对质量属性以及质量属性之间的关系进行分析，也无法对软件架构进行评估。因此，我们进行了属性模型构造和分析。首先我们对单一的属性抽取出来做详细分析，对属性确立了参数化指标。例如我们在通过场景分析后，确立了可用性的指标，对于接口的成功率需要达到99.8%，页面响应时间需要控制在2s以内。我们还对质量属性之间的关系做了分析，很多属性之间会有对冲关系，鱼翅熊掌不可兼得，在这个时候我们往往要基于项目实际做权衡与取舍。例如，性能和安全这两个指标往往会有冲突，我们在APP登录的时候由于要做鉴权处理，会导致登录时间有几秒的延迟，会对性能有影响，这个时候就需要依据项目需求进行权衡。经过权衡利弊分析，我们选择了安全性而牺牲了一部分性能，确保用户的敏感信息不受到泄露。通过这个过程，我们对每个质量属性都进行了详细的分析，细化了指标，也分析了属性之间关系，使得评估小组成员对质量属性有了更深入的认识。

通过以上过程实施，我们成功完成了软件架构的评估，并搭建了项目质量属性评价机制。通过对项目的非功能需求评价，我们发现了软件架构需要改善的地方以及质量属性需要提升的地方，为后续的架构优化提供了重要的依据。同时，对于系统性能指标的梳理，也作为考核系统以及团队成员绩效的依据。对于质量属性的逐级评价，也便于我们对系统的不足点进行逐步优化，逐步提升系统的健壮性。系统已经2019年12月成功上线，且上线以后表现稳定，得到了客户高层领导的大力好评。

尽管如此，这个过程还是有需要改进的地方。部分团队成员对于质量属性的分析理解还是不够深入，分析过程掌握不够熟练，对于多个场景综合分析或者同时涉及几个对冲质量属性的场景分析，分析不够到位。后期我们将加强对质量属性评估方法的研究和应用，提升综合场景下质量属性方法灵活使用的能力，更好地发挥软件架构评估的作用。

## 论基于构件的软件开发

软件系统的复杂性不断增长、软件人员的频繁流动和软件行业的激烈竞争迫使软件企业提高软件质量、积累和固化知识财富，并尽可能地缩短软件产品的开发周期。

集软件复用、分布式对象计算、企业级应用开发等技术为一体的“基于构件的软件开发”应运而生，这种技术以软件架构为组装蓝图，以可复用软件构件为组装模块，支持组装式软件的复用，大大提高了软件生产效率和软件质量。

请围绕“基于构件的软件开发”论题，依次从以下三个方面进行论述。

1.简述你所参与开发的运用了构件技术的项目，以及你所担任的工作；

2.论述你在项目中如何运用构件技术来进行软件开发；

3.分析并讨论各种构件技术的优点、缺点，并展望构件技术的发展趋势。

**范例**

摘要部分：

2019年3月，我单位联合某高校研发了《程序在线评测比赛考试系统》。系统以程序代码在线提交自动评测功能为核心，主要分为题库模块、评测机、实验作业模块、考试模块、比赛模块、抄袭判定模块、用户管理模块等，支持对接教务平台。在项目中我担任系统架构师，主要负责架构设计工作。本文以该平台为例，论述了基于构件的软件开发技术在本项目中的具体应用。系统基于Spring Cloud微服务框架来进行构件化开发，在构件获取阶段通过对接现有系统、使用构件库、集成第三方软件来实现需求，在构件开发阶段使用了多种设计模式来保证构件的可重用性，在构件组装阶段使用了同步消息、异步消息、工作流方式来实现不同业务类型下构件的组合。最终项目顺利上线并稳定运行，获得用户一致好评。

正文部分：

我在一个专为高校建设计算机专业智能教学一体化平台的单位任职，过往成果有《计算机组成原理仿真实验系统》等。2019年3月，我单位联合某大学研发了《程序在线评测比赛考试系统》项目（以下简称为“OJ系统”），以取代原有传统的编程上机考试平台。

系统以程序代码的在线提交自动评测功能为核心，主要分为题库模块、评测机模块、实验作业模块、考试模块、比赛模块、抄袭判定模块、用户管理模块等。题库模块主要负责试题和测试用例的管理，用户根据试题要求编写程序代码提交到系统，系统将测试用例与程序代码发送给评测机模块，由评测机自动编译、执行、判分，并将结果发送给其他相关模块进行统计；实验作业模块用于在线布置作业，从题库中选取试题，设置截止日期等要求；考试模块用于学生在线考试，按教师预先设置的参数自动从题库随机抽题生成试卷，以及向教务平台上传考试成绩；比赛模块主要用于ACM竞赛的培训；抄袭判定模块用于鉴定代码与他人代码雷同率；用户管理模块负责用户信息的管理。在这个项目中，我担任了系统架构师的职务，主要负责系统的架构设计相关工作。

构件技术是指通过组装一系列可复用的软件构件来构造软件系统的软件技术。通过运用构件技术，开发人员可以有效地进行软件复用，减少重复开发，缩短开发时间，降低软件的开发成本。主流的构件技术有三种：CORBA、EJB和COM。CORBA分为对象请示代理、公共对象服务和公共设施三个层次，优点是大而全，互操作性和开放性好，缺点是庞大且复杂，技术标准更新缓慢；EJB基于Java语言，支持跨平台，提供了远程访问、安全、持久化和生命周期等机制，支持分布式计算，缺点是服务治理能力差，逐渐被Spring Cloud等轻量级框架取代；COM基于Windows平台，功能强大、效率高，有一系列开发工具支持，缺点是跨平台性差。基于构件的软件开发过程主要分为模块划分、构件标识、构件获取、构件组装与测试、构件管理等步骤。

OJ系统采用了基于Spring Cloud的微服务架构开发。这里重点从构件获取、构件开发、构件组装三个阶段说明本系统采用构件化开发的过程。

1.构件获取

在OJ系统中，可复用的功能需求和非功能需求从实现方式上分为三类。第一类需要对接现有系统实现，比如需要将学校在用的OA、教务管理两套系统中的学生信息与课程信息同步到OJ系统中，我们通过访问OA、教务管理的开发商网站，联系开发商，取得了相关接口调用的SDK。第二类是常见信息系统中共同具备的用户管理、角色权限管理、日志记录、内容维护、消息中心等基本功能，我们通过取用单位过往项目开发中所积累的构件库中的相应构件（例如RBAC权限管理框架）来实现。第三类需要集成第三方的软件来实现，例如微服务注册中心、API消息路由网关、负载均衡机制、程序评测时调用的编译器、代码抄袭判定的文本比对功能等，无需另行开发，我们使用了Spring Cloud中的Eureka框架来作为微服务注册中心，Nginx反向代理来作为负载均衡机制，GCC、JRE等作为编译器，针对文本比对功能，我们使用了GitHub上的开源程序。此外我们还根据OJ系统的使用场景，重新开发了一些功能构件，以支撑本系统特定的用户需求。

2.构件开发

构件的优势体现在其粗粒度的重用性，因此为最大限度保证构件的可重用性和重用力度，保持其高内聚、低耦合的特性，我们在开发中用到了一些经典的设计模式，例如装饰器模式、策略模式、工厂模式等。在题库构件中，存放了全部的试题数据，包括题目介绍、测试用例等公共内容，同一试题在自由练习、实验作业、考试、比赛等不同使用场景下，需要增加不同的扩展功能。我们使用了装饰器模式，来给试题对象动态添加不同职责。在评测机构件中，需要同时支持对C语言、C++、Java、PHP、Python等多种常用编程语言程序的评测，不同语言对应的编译与执行方法存在较大差异。我们使用了策略模式，将不同语言的编译和执行方法封装起来，并使它们可以相互替换。在数据库连接构件中，因不同的业务需求，需要同时支持不同类型的数据库，包括MySQL、Oracle、SQL Server三种。我们使用了工厂方法模式，有效解决了不同数据库类型对软件程序的影响，具有很好的可扩展性。这样的设计有效地体现了构件的优势。

3.构件组装

OJ系统中不同的业务类型，需要采用不同的构件组装方式。在本系统的开发过程中，我们用到了以下三种方式。首先是同步消息方式，以考试功能为例，用户进入考试时，考试构件需要核验考生信息，然后调取题库中的试题信息组成试卷，在此过程中，考试构件使用同步方式，依次向考生构件、题库构件请求数据，等待返回结果后再加工、组合，提供给调用者。其次是异步消息方式，以程序代码的在线提交评测功能为例，评测机构件对代码的编译、执行、判分过程，相比于其他构件的数据处理过程，属于耗时操作，这时如果采取同步方式，将引起调用者阻塞，严重影响了用户体验，甚至出现雪崩效应。因此采用了异步消息队列，代码提交后写入消息队列立即返回原程序执行，待评测机评测完成后再异步显示评测结果。最后是基于工作流的方式，能够通过图形化的界面，动态编排系统构件之间的交互和依赖关系，灵活改变多个构件之间协同工作的顺序，通过简单构件的组合，以实现复杂功能的定制。

系统自2019年10月正式上线已运行一年有余，在学校的日常教学考试和竞赛培训中投入使用，截至目前已有3000以上的学生用户、评测了70000条以上的程序代码，获得了单位同事领导和学校教师们的一致好评。

实践证明，OJ系统项目能够顺利上线，并且稳定运行，与系统采用了基于构件的开发方式密不可分。经过这次构件化开发的方法和实施的效果后，我体会到了软件元素重用对开发过程的重要价值。从软件开发的方式看，由机器语言、汇编语言，到面向过程开发、面向对象开发，再到现在基于构件、面向服务的软件开发，软件元素在两个维度上呈现进化趋势：内部功能越来越强大、全面，对外的接口越来越简单、标准。最终各领域软件可在一个统一标准下无缝组装，届时面向协作的软件开发、基于职能的软件开发等新技术都将出现，上层应用功能的实现也将变得异常简单，计算机软件将会无所不在，数字化生活、智能地球等现在还处于概念阶段的事物，将会走进现实。这个目标值得我们每一位软件从业人员为之努力奋斗。

## 高可靠性系统中软件容错技术的应用

容错技术是当前计算机领域研究的热点之一，是提高整个系统可靠性的有效途径，许多重要行业（如航空、航天、电力、银行等）对计算机系统提出了高可靠、高可用、高安全的要求，用于保障系统的连续工作，当硬件或软件发生故障后，计算机系统能快速完成故障的定位与处理，确保系统正常工作。

对于可靠性要求高的系统，在系统设计中应充分考虑系统的容错能力，通常，在硬件配置上，采用了冗余备份的方法，以便在资源上保证系统的可靠性。在软件设计上，主要考虑对错误（故障）的过滤、定位和处理，软件的容错算法是软件系统需要解决的关键技术，也是充分发挥硬件资源效率，提高系统可靠性的关键。

请围绕“高可靠性系统中软件容错技术的应用”论题，依次从以下三个方面进行论述。

1．简述你参与设计和开发的、与容错相关的软件项目以及你所承担的主要工作。

2．具体论述你在设计软件时，如何考虑容错问题，采用了哪几种容错技术和方法。

3．分析你所采用的容错方法是否达到系统的可靠性和实时性要求。

**范例**

摘要部分：

2018年3月，本人在公司参加了收单交易系统的开发。该项目主要功能是做传统pos机、智能pos机的收单业务，商户信息进件以及维护，交易对账以及清算业务。本人在该项目中担任了系统架构师职位，负责该系统的架构设计工作。本文以该项目为例，主要论述了高可靠性系统中软件容错技术的应用。为了防止硬件和软件的单节点故障问题，我们采用了集群部署的方式来提高软硬件服务的可靠性；数据库是所有业务服务都依赖的基础服务，我们采用了主备部署的方式提高数据库的可靠性；为了提高单个业务服务软件健壮性与可靠性，我们采用了防卫式程序设计、降低复杂度设计和错误日志记录的方案来提高服务的可靠性。通过上述方案的使用，保障了服务的高可靠性，最终项目顺利上线，持续稳定运行。

正文部分：

2018年3月，本人在公司参加了收单交易系统的开发。该系统的核心业务场景是帮助各个商户进行收款服务，下个工作日对账完成后，将商户的收款资金再清算给各个商户。该系统的主要功能模块有交易系统、风控系统、商户终端管理系统、清结算系统、内部管理平台等。交易系统为收单业务的核心服务，帮助商户进行有卡和无卡的消费、撤销、退货、预授权等业务支持，因为业务场景是支持各大商户的实际收款业务，所以该服务必须保证7\*24小时的高可用性；风控系统主要是给收单交易系统提供风控业务支持，避免一些套现、洗钱等非法金融类交易；商户终端管理系统是用来管理维护商户信息和商户购买的终端信息，进行信息的新增、修改和查看业务；清结算系统是进行交易对账和清算业务的，对账完成后将商户的交易款清算给各个商户；内部管理平台是一个综合性的内部管理平台，为内部工作人员提供一些管理工具，比如交易数据查看、各种交易类统计报表、清算明细数据查看、商户信息查看等。本人在该系统中担任系统架构师职位，负责该系统的架构设计工作。

系统可靠性包括硬件可靠性和软件可靠性，两个方面都需要我们考虑到，软件的复杂度比硬件高，所以软件系统发生故障的概率也会高于硬件系统。硬件在使用时间过长的情况下，物理退化的现象加重，故障发生概率就会变高，通常我们会采用主动冗余、监控检错配合报警的方案来解决。软件系统包含各种的业务逻辑，复杂度较高，故障概率高，通常我们采用防卫式程序设计、N版本程序设计、降低复杂度设计等方案来提高单个服务的自身可靠性；单个服务的可靠性不可能达到百分百，总归避免不了特殊场景下导致服务崩溃无法提供服务支持的情况，针对这一问题通常采用集群部署配合监控报警的方案来解决，即使一台服务崩溃，还有其他服务提供者进行业务支持。合理的采用容错技术方案，可以大大提高服务的可靠性。

在收单交易系统中，我们采用了多种容错技术来提高系统的可靠性，下文着重讨论应用系统集群部署、数据库主备部署和程序设计方面在该项目中容错方面具体的应用和效果。

一、系统集群部署容错

单个硬件设备或者说单个软件服务进程，都不可避免的会发生故障导致服务中断。如果单个硬件设备上部署一个或多个软件服务进程，当硬件发生故障，所有软件服务都将不可正常服务；如果一个软件服务就部署了一个进程在提供服务，一旦这个软件进程发生故障，服务也将不可正常运行。针对上述问题，集群部署方案可以很好的解决。同一个软件服务，在不同的多台物理设备上做集群部署，即使一台物理设备发生故障或者一个软件服务进程发生故障，都有其他的多台物理设备和软件进程在提供服务，达到服务永不中断的目的。集群部署就要考虑到请求分发和状态的问题，因为我们的业务是pos机交易，走的是TCP/IP协议，采用了硬件F5做负载分发业务；每个业务进程都尽量做成无状态服务，实在需要处理状态的，我们采用了redis集群来保存状态数据的方案。发生问题的物理设备和软件服务要尽快被发现然后修复，我们采用了zabbix分布式服务监控系统来实现，一旦监控发现问题，立马发送邮件给相关维护人员进行处理。通过集群和监控报警的配合，大大提高了整个系统的可靠性和实时性。

二、数据库主备部署容错

数据库服务是整个收单交易系统的基础服务，每个业务模块都要使用到，所以一旦数据库出现问题，整个服务将中断，影响重大。单个数据库服务不可能说永远不可能出现故障，我们要做到的首先是提升单个数据库服务的可靠性，其次是如果单个数据库服务发生故障，立刻快速使用备用数据库继续支持服务。针对上述问题我们采用了数据库主备部署方案，主备都同时提供服务，每台服务的压力就会减少，提高单台数据库服务的可靠性。主库服务于全部的写操作和少量的读操作，比如说交易业务；备库服务于实时性要求不高的读取操作，比如说内部管理平台查看流水、构建各种报表业务。一旦主库发生故障无法提供服务，备库可以快速的升级为主库，继续提供服务，保证了整个数据库系统的整体可靠性。在数据库的物理文件保存方面，通过磁盘阵列技术做到磁盘容错，防止数据丢失。通过数据库主备部署和磁盘阵列方案，大大提升了数据库服务的可靠性。

三、程序设计方面容错

程序设计方面我们采用了防卫式程序设计以及降低复杂度的策略。防卫式程序设计方面首先是报文字段的检测，报文中各个参数按照预定的规则先进行检测，检测通过后再进行后续的业务处理，比如说交易金额，必须是一个正整数，比如说商户编号必须为固定15位长的字符串，参数检测通过后会大大降低后续业务出现不可控制的错误几率；其次是异常处理，程序正常逻辑要处理，发生异常的情况下我们也要考虑如何处理，比如说查看商户法人证件照片，读取文件的过程中很可能会出现文件未找到的错误，一旦发生这个错误我们就要按照异常的流程处理，提示说照片文件未找到，不能进行处理；再次是进行错误信息日志记录，程序发生错误时，把相关的详细信息都记录到文件中，方便后续查找定位问题。整体的系统设计我们都采用了合理的架构模式，遵循高内聚、低耦合、单一职责、依赖倒置、接口隔离等基本的设计原则，这些都会降低整个系统的复杂性，增加系统的可靠性。

该项目于2019年6月完成测试并上线，通过上述容错方案的实施，整个系统持续稳定的运行，未出现过服务中断的故障，即使系统变更上线，也可以通过集群优势，逐台设备更新上线部署，达到服务不中断的效果。后续我们又完善了zabbix监控系统，丰富了监控的内容，包括硬件设备监控、网络流量监控、软件进程监控、软件业务数据监控等，实现了更加全面的监控策略。错误日志记录追踪发挥了良好的作用，定期我们会排查一遍错误日志，从中分析出系统存在的问题再进行修复，进一步完善系统，进一步提高系统的可靠性。

当然，在我们设计系统和后期运行维护过程中，也陆续发现了一些问题和不足，比如说集群部署导致运维成本增加，后期我们增加了jekins自动部署的功能来解决一些问题。另一个问题就是单个服务不可用时，还是需要人工来处理，不能及时的自动新增一个服务节点上去，目前我们正在研究使用容器技术来解决该问题，等成熟稳定后也会陆续投产使用。

## 论信息系统的安全性与保密性设计

在企业信息化推进的过程中，需要建设许多的信息系统，这些系统能够实现高效率、低成本的运行，为企业提升竞争力。但在设计和实现这些信息系统时，除了针对具体业务需求进行详细的分析，保证满足具体的业务需求之外，还要加强信息系统安全方面的考虑。因为如果一个系统的安全措施没有做好，那么系统功能越强大，系统出安全事故时的危害与损失也就越大。

请围绕“信息系统的安全性与保密性”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设计的信息系统及你所担任的主要工作。

2.深入讨论作者参与建设的信息系统中，面临的安全及保密性问题，以及解决该问题采用的技术方案。

3.经过系统运行实践，客观的评价你的技术方案，并指出不足，以及解决方案。

范例

摘要部分：

2018年3月，本人所在的公司承接了一套代付系统的建设。该项目主要业务是接收各个上游服务的代付请求，通过某个第三方代付通道进行实际出款。本人在该项目中担任了架构师的职位，负责该系统的架构设计。本文以该项目为例，主要论述了信息系统的安全性和保密性设计。在交互报文设计方面，为了防止报文信息泄露、篡改和抵赖问题，我们采用了数字信封的策略；在数据库存储方面，为了防止数据信息泄露以及篡改，采用敏感信息加密保存、记录摘要字段的策略；在登录认证方面，为了增加安全强度，我们采用了用户密码、图形验证码、手机验证码、Ukey四项策略来实现登录认证。通过以上技术方案的实现，系统的安全性和保密性达到了很好的效果，最终项目顺利上线，稳定运行。

正文部分：

2018年3月，本人所在的公司承接了一套代付系统的建设。该系统的核心业务点是将接收到的代付请求，审核通过后，通过第三方代付通道将钱实际发出去。该系统分为四大子系统：前置服务，用于接收上游系统发送过来的代付请求报文，验证通过后记录入库；代付通道服务，该服务内对接了多个第三方代付通道，风控清算等一系列部门审核通过后，该服务将钱实际代付出去；路由服务，该服务是根据不同的代付业务场景，以及每个代付通道的实际资金情况等，选择把每个代付流水分配到哪个第三方代付通道进行出款；管理平台服务，该服务的内容比较多，包括代付流水的查看、代付流水的审核、证书管理、第三方通道信息管理、报表服务等。因为该系统中存在着大量的敏感信息，比如银行卡号、银行卡户名、身份证号、手机号等，所以该项目的安全等级必须设计为最高等级。本人在该项目中担任系统架构师一职，负责该项目的架构设计工作。  
代付系统中，因为业务的敏感性，涉及到很多的敏感信息，比如说银行卡号、户名、身份证号、手机号等，这些敏感信息都需要在网络中进行传输，也会在数据库里进行保存，因此数据的安全性和保密性问题面临着巨大的挑战，网络传输的过程中有可能被非法采集，数据库被拖库的情况下也会导致信息泄露；业务场景是把钱真实的代付出去，涉及到上游系统或者机构的抵赖问题，每一笔交易报文都要让对方无法抵赖；接口报文在传输过程中绝对不能被篡改，否则实际出款数据就完全错了；报文接口也要考虑到重放攻击的可能，报文重放，签名也挡不住，会导致重复处理代付的问题；业务系统在处理的过程中，也存在着泄露信息的风险，比如说日志系统，如果说日志文件中记录了敏感信息的明文，就有可能存在日志文件被窃取导致信息泄露；管理平台服务的操作都是很敏感的，所以管理平台的登录认证功能也必须做到最高安全等级，一旦登录入口被攻破，后果难以想象。  
以上列出了部分系统中涉及到的安全性和保密性问题，针对这些问题，我们采用了对称和非对称算法的加解密、计算信息摘要、报文签名策略、日志脱敏打印、物理设备Ukey、防止重放攻击、https协议等一系列安全防控策略。下文着重讨论接口报文层面、数据库存储层面、登录认证层面所面临的问题以及我们采用的解决方案。  
一、接口报文层面  
代付项目中的报文，含有大量的敏感信息，包括银行卡号、户名、身份证号、手机号等，这些都是一级敏感数据，绝对不能在报文传输过程中被非法收集到。在实际业务场景方面，一笔报文请求的到来，意味着要把真金白银给发放出去，所以对上游系统的身份验证也必须做到最高的安全等级控制，防止对方抵赖不认可。针对以上问题我们决定采用数字信封技术进行解决。报文加密方面，因报文的数据量比较大，采用对称算法进行加密，每笔请求随机生成一个对称算法的秘钥加密报文，该对称秘钥采用对端的公钥进行加密传输，做到一笔一个随机加密秘钥，提高加密安全性。身份验证方面，将报文明文进行摘要签名，签名结果随报文一起上送，用于身份验证，签名串在数据库中也会进行存储，防止对方抵赖。加密和签名解决了，还有一个很重要的重放攻击需要防御，我们在报文内部加了唯一流水号以及时间戳信息，流水号必须全局唯一，在数据库层面做了唯一索引控制，业务系统上判断如果时间戳和当前时间相差超过30秒直接作为失败处理。  
二、数据库存储层面  
代付项目中的数据，都是极为敏感的，数据库中绝对不能明文存储，防止数据库被拖库后信息泄露，针对这一场景，我们决定对每个敏感字段进行加密存储。前置服务在收到代付请求后，把敏感字段信息加密后存储，后续的各个业务服务，都是只能使用密文来进行业务中转处理。加密防止信息泄露解决了，但是数据库还有一个大问题需要解决，防止数据篡改，如果说数据记录里的卡号、姓名、金额等信息被篡改，那这笔钱代付就完全错了，这个影响是不可接受的。针对这一问题，我们把每条数据记录额外计算一个摘要信息字段进行存储，数据在入库前，把关键性字段（卡号、姓名、金额等）拼接起来，再使用一个系统内部自定义的秘钥串作为盐，然后计算一个摘要结果一起入库保存，后续任何业务场景中读取出数据记录后，第一步就是重新计算摘要信息，如果发现和数据库里记录的不一致，拒绝处理。  
三、登录认证层面  
代付系统的管理平台服务，可以进行流水的查看、导出、秘钥证书管理等，都是极为敏感重要的操作，稍有不慎，后果就会很严重，因此该平台的首要入口登录认证就要严格控制。首先是用户名、登录密码、图形验证码基本三要素进行判断；其次是手机验证码控制，每次管理员登录时，系统都会发送手机验证码到管理员信息表里预设的手机号上，该手机号验证码验证通过才可以，防止管理员密码丢失导致他人恶意登录；再一层次是每个管理员都分配了一个单独的物理Ukey，该Ukey里保存着一个单独的私钥数据，管理员每次登陆的时候，系统都会调用Ukey生成一个签名数据送到服务端，服务端使用该管理员的公钥进行签名验证，该ukey为一个物理硬件，除非丢失，否则不可能被窃取，如果认为丢失了，可以在平台商直接禁用掉该管理员即可。通过手机号验证码和Ukey的引用，登录认证模块的安全性可以达到最高的层次。此处我们也加了防止重放攻击的措施，防止报文被截取到直接重放。  
该项目历时13个月，于2019年4月上线，一直持续稳定运行，未出现过安全事故，也未曾反应泄露过数据。前置服务接收代付请求模块中，后续经历了好多次的攻击，都是签名验证失败给挡掉了，目前未碰到重放攻击的，后续即使遭遇到重放攻击，系统也可以识别出进行防御。前置接口的攻击，虽然被签名验证给过滤了，但是大量的签名验证请求，严重消耗CPU资源，针对这一情况，我们在前置服务里加入了IP白名单的限制，如果是非IP白名单的报文请求，系统直接拒绝处理。关于证书的管理，考虑到相关人员定期有离职的情况，为了防止证书泄露，系统中使用到的证书和秘钥，我们都要求定期进行更换，虽然麻烦，但是必须做到安全第一。安全和性能一直就是对冲的属性，需要去平衡，但是针对该系统而言，安全必须做到最高，性能我们可以适当放弃，项目后期压力测试时，性能不是很满意，后续商讨决定，安全等级不能降，通过集群部署的方式来增加系统性能。  
系统安全是一个永久的话题，我们对系统的完善改进也是一个持续的过程，后续我们还会继续观察改善系统安全性方面的问题以及不足，使整个代付系统更加的安全可靠。

## 论系统安全架构设计及其应用

信息安全的特征是为了保证信息的机密性、完整性、可用性、可控性和不可抵赖性。信息系统的安全保障是以风险和策略为基础，在信息系统的整个生命周期中提供包括技术、管理、人员和工程过程的整体安全，在信息系统中保障信息的这些安全特征，并实现组织机构的使命。许多信息系统的用户需要提供一种方法和内容对信息系统的技术框架、工程过程能力和管理能力提出安全性要求，并进行可比性的评估、设计和实施。  
请围绕“论系统安全架构设计及其应用”论题，依次从以下三个方面进行论述。  
1、概要叙述你所参与管理或开发的软件项目，以及你在其中所承担的主要工作。  
2、详细论述安全架构设计中鉴别框架和访问控制框架设计的内容，并论述鉴别和访问控制所面临的主要威胁，并说明其危害。  
3、阐述你在软件开发的过程中都遇到了哪些实际问题及解决方法。

范例

摘要部分：

本人于2019年9月参与某运营商客户管理系统的架构演进项目，该系统是运营商传统架构面向互联网化分布式架构真正意义上的全面重构，前后端分离，全面的微服务化平台构建。在该项目组中我担任架构师，主要负责整体架构设计和中间件选型。本文以该演进项目为例，主要讨论系统安全架构设计及其应用在本项目中的具体应用。本项目通过采用面部识别和证件照片验证等生物特征技术解决客户因为证件丢失,被他人盗用证件办理业务后用于从事非法活动的问题;通过采用基于角色的访问控制来限制营业员的业务受理范围和用户的业务受理范围；通过采用访问控制列表技术实现该运营商的网上商城业务，拓展了新业务领域，增加了该运营商的收入。项目最终顺利上线，获得一线员工和用户的一致好评。

正文部分：

随着互联网行业的迅猛发展，需求的快速变化，传统的单体软件已无法支撑创新性新需求，无法吸引新客户，留住老客户。该运营商经过认真调研后决定采用平滑演进的方式来实现该客户管理2.0系统，因此面向2.0用户设计分布式云化架构，以提高系统安全性，提升服务体验，留住新老客户。

该客户管理2.0系统是该运营商营业厅受理业务的核心系统，包括了开户、变更、移机等几十个菜单。该系统实现了前后端分离，微服务化平台构件，被分成了12个中心，8个专题，我司以Spring boot主流微服务技术为核心，整合了msha、hystrix等容错技术；es、cache等缓存技术；mongDB、drds等数据库技术；MQ等消息中间件技术等。以最小的资源成本、全技术栈的独立实现了该系统的交付中心、产品中心、权限中心、报表中心和数据交换中心的建设，与甲方共建用户中心、客户中心、订单中心及其它多个专题建设。本项目组共68人（不含甲方），我在项目中担任系统架构师职位，架构小组共5人，我主要负责整体架构设计及中间件选型，项目于2020年2月成功启动。整个项目共耗时18个月，第一版于2021年10月顺利通过验收。同时也实现了跨域融合等重点业务。

系统的安全架构设计主要有鉴别服务、访问控制、数据完整性校验、数据保密性和抗抵赖等5个方面，鉴于该运营商的业务特点，我们着重考虑了鉴别服务的能力和访问控制能力。鉴别技术主要有用户名+口令模式、数字证书和生物特征识别等技术。鉴别技术面临的主要目的是验明用户或者信息的正身，面临的威胁主要有用户名、口令被盗；生物特征被窃取、数字证书重放冒用等。访问控制主要有自主访问控制、访问列表控制、强制访问控制、基于角色的访问控制和基于任务的访问控制等技术。访问控制监控哪些资源可以被访问和使用，以及在某种情况下最大限度的降低未经授权行为的风险。无论是终端用户因疏忽而导致的敏感信息泄漏，还是因公共web服务器软件漏洞而导致的敏感信息泄漏，都可能造成灾难性后果。

在该客户管理系统2.0系统中，我们着重采用了生物特征识别方式来验证客户身份的准确性，采用访问控制列表和基于角色的访问控制来实现系统的访问控制技术。下文着重讨论这些技术在系统中的具体应用和实现

一、生物特征识别

首先我们在受理业务的首页设置了客户认证菜单页，在客户认证通过的情况下才可以办理该客户下的业务，若认证未通过，则不允许受理任何业务。在用户受理业务时，我们首先要确保是实名办理业务，客户需要出示证件，系统调用第三方系统核实证件真伪，其次需要现场拍照，系统进行照片比对核对证件和证件持有人是否一致并将照片信息存储在后台hbase中，防止用户的业务被非法篡改，也防止别有用心的人利用账号从事非法活动，给国家和人民造成经济损失。只有当客户认证通过之后才可以正常办理业务。在受理业务之前通过生物特征识别校验用户的真伪性，在业务受理最后一个阶段需要采集用户的指纹或者在线签名，确保用户知道自己受理的具体业务内容，以及日后的防抵赖性。对于特殊类代办业务，系统需要验证代办人的证件资料，采集代办人的人脸信息，以及代办业务用户所收到的实时验证码，当验证码校验通过之后才运行受理该代办业务。

二、访问控制列表

访问控制列表主要是针对该运营商的网上商城业务而设计的，该运营商为了增加集团的收入，在手机app客户端和网上商城均开放了购物业务，用户可以使用话费购买商城的产品，比如京东卡、腾讯会员、生活用品等。由于账号和口令可能被暴力破解或者泄漏，所以如何在消费话费的时候确认该购买行为是该客户有意识的主动发起是我们遇到的最大的难题。因为客户可能是误操作，也可能账号是被盗取。还有可能是购买意愿不强烈，订单取消风险较大，但是某些商品如京东卡一旦售卖成功是不允许取消订单的，导致消费者的购买体验较差。经过慎重研究决定采用访问控制列表的方式来实现该购买行为。当购买行为进入支付阶段的时候系统会给用户发一条动态短信，若用户按照短信内容回复短信，则可以判断是用户的真实操作，才会进入真正的支付环节以及产品交付环节。该技术的应用成功拓展了业务领域，增加了集团的收入，同时也得到了用户的好评。

三、基于角色的访问控制

基于角色的访问控制，主要是针对营业厅营业员以及运维系统的运维人员，基于角色的访问控制可以限制营业员受理业务的范围，比如是否可以受理跨地市的业务，是否可以受理新上线的业务，是否可以给用户减免违约金，是否可以给予某种优惠等操作，一旦操作不当就有可能导致用户的投诉，给集团带来直接经济损失也会带来口碑上的下滑，不利于客户关系的维护。运维人员如果因误操作对用户数据造成不可恢复的损害，也将造成严重后果，所以需要实现“最小权限原则”和“权限分离原则”，将试图访问信息的用户只能访问执行自身角色职能所必须的那部分数据。实现基于角色的访问控制，还有一大好处就是便于对营业员和运维人员账号的维护，当员工需要某个权限的时候，只需要在该工号上配置上一条相应的角色就可以实现营业员账号权限的扩展，提供给工号管理系统的就是一个选项的按钮而已，可以以最快的时间实现营业员权限的扩展和收缩。

通过上述鉴别服务和访问控制方案的实施，该客户管理2.0系统于2021年10月完成第一个版本的顺利验收。上线后虽然出现了某些特殊用户到某个营业厅办理不了某些业务需要到指定的营业厅才能办理的场景，但是并没有发生角色越权行为，没有给集团带来经济损失。项目至今仍然保持着每月一次或者两次的版本迭代，顺利实现了1.0到2.0架构演进的平滑式过渡，得到了甲方的肯定，实践证明鉴别服务和访问控制方案适用于该项目安全架构的实现。

当然，在系统的后期运行维护过程中，也陆续发现了一些问题和不足，在服务调用链上没有做访问量控制，没有做调用链剥离，当推广的某一业务量突然上来后，会导致调用链崩溃而不可用，也会因为运维修复大量数据重跑某些业务，导致关键业务崩溃，波及实时业务。同时也导致了很多异常数据，给运维人员增加了工作量。在后续的改进方案中对重点调用链进行了剥离，并做了访问量控制。运维大批量的数据修复工作只能在晚上执行。运维人员对数据库的一切增删改查操作均需要有日志记录，并定期将该运维人员的操作日志发给该运维人员确认是否是本人操作，确保生产数据的安全性。

## 论软件设计模式及应用

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。

范例

摘要部分：

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。

正文部分：

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度也会增大，可能修改其中一点会影响到其他人活动的正常使用。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类达到服务的自由切换。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。