# 论设计模式在图书馆数据中台的实践

## 一、摘要

2019年11月，本人所在信息技术部承接了馆内数据中台的建设，该项目为期半年。该项目一期于 2020 年 5 月正式上线， 2020 年 6 月通过最终验收。通过数据中台，实现了我馆数据的分层与水平解耦，沉淀了公共基础数据。通过数据建模实现跨域数据整合和知识沉淀，通过数据服务实现数据的封装和开放，通过数据开发工具满足个性化数据需求，为全市文旅数据融合工作提供了有力的数据支撑。该项目时间紧任务重，涉及人员组织多，涉及馆内 10 余个部门，外部配合协作 10 多个厂商团队。本人在此项目中担任系统架构设计师，主要完成技术方案评估与实现，项目立项论证等工作。本文以馆内大数据基础平台为例，讨论设计模式在该项目中的具体应用，包括设计模式的概念、组成以及在实际场景中遇到的问题。

## 二、正文

近年来，随着我馆服务质量的不断提升，大量软硬件系统在我馆得到部署。硬件设备包括自助借还机、交互机器人、电子书借阅机、人脸识别门禁、人流量计数器，软件系统包括图书管理系统、微信小程序、移动端APP。软硬件系统的增加带来了大量用户数据的流入，如读者借阅数据、读者信息、阅读偏好、入馆行为等海量数据，但是这些数据缺乏统一的接入和分析。因此，需要通过数据技术，对海量数据进行采集、计算、存储、加工，同时统一标准和口径，并在此基础之上，对数据进行归纳分析，对外提供统一的、科学的、精准的馆情数据服务。

笔者带领团队于2019年11月承接馆内数据中台的建设，并担任系统架构设计师。主要负责项目整体技术方案评估实现、立项论证以及项目管理工作。该项目的架构工作于同年12月完成，整个项目耗时6个月，于2020年5月上线测试。于2020年6月完成验收。

在实际开发阶段，笔者意识到设计模式代表了最佳的实践，通常被有经验的面向对象的软件开发人员所采用。设计模式是软件开发人员在软件开发过程中面临的一般问题的解决方案。这些解决方案是众多软件开发人员经过相当长的一段时间的试验和错误总结出来的。由于传统的结构化的软件设计方法不符合面向对象的设计原则，无法很好的实现高内聚和低耦合的要求。模块之间过于紧密，给软件扩展和维护带来很多困难。在这种情况下，设计模式的出现和广泛应用给问题的解决提供了一种有效方法。通过利用设计模式，可以帮助开发者复用已有的设计方法，设计出结构合理、易于复用和可维护的软件，当用户需要发生改变时，可以通过修改少量代码或不修改原有代码即可满足新的需求，增强了系统的可修改性和稳定性，降低系统开发成本。

设计模式依据其目的可以分为创建型、结构型、行为型三种类型。创建型模式，主要负责对象的创建工作，程序在确定需要创建对象时，可以获得更大的灵活性。常用的创建型设计模式有：单例模式、工厂方法、原型、构造器、抽象工厂等5种模式。结构型模式，负责处理类或对象之间的关系，用于构件结构更加复杂庞大的系统。常用的结构型设计模式有适配器、桥接模式、享元模式、组合模式、外观模式、代理模式等7种模式。行为型模式，主要任务是对类或对象如何交互以及为类和对象分配具体职责进行描述。常用的行为型模式有观察者、状态、策略模式、备忘录、命令、责任链、中介者等11种模式。这些设计方法都是经过反复使用的成熟方法，对优化软件结构，提高软件质量具有重要的指导意义。针对数据中台的应用场景，为尽可能达到软件复用与业务水平扩展的目的，笔者采用层次化+微服务的风格来进行系统架构。按照业务层次的不同，将系统分成数据接入层、数据模型层、数据服务层，数据开发层。本文也按照此为脉络讲解设计模式在各个层次上的应用。

在数据接入方面，数据接入是中台的基础业务，通过对接各个软硬件厂商，针对不同类型软硬件系统的特点，提出不同的数据需求,联合各个厂商一起讨论制定标准化的上报接口。由于业务是一直在演化的，可能随着硬件和产品的不断迭代，数据处理的方式可能也发生演化，为了确保向下兼容和代码的复用性，笔者决定采用策略模式来处理这一部分的逻辑。在策略模式中，一个类的行为或其算法可以在运行时更改。这种类型的设计模式属于行为型模式。在策略模式中，笔者将不同的数据处理方法封装在一个个独立的策略类中，开发者可以根据自己的需求从不同策略中进行选择，有效的避免了使用条件转移语句带来的不易维护的缺点。而且策略模式利用组合代替继承，将数据处理的实现与选择分离开来，降低模块之间的耦合度，增强了代码的可扩展性和可维护性。

在数据模型方面，笔者这里概括为三层。分别是基础模型、融合模型和挖掘模型。其中基础模型一般是关系建模，主要实现数据的标准化，这也叫作“书同文、车同轨”。通过设计好的数据标准将接入的数据进行入库管理。其次是融合模型。融合模型一般是维度建模，主要实现跨域数据的整合，整合的形式可以是汇总、关联，也包括解析。这些融合模型表是通过各个业务部门梳理而得到的常用的宽表，比如读者终端使用行为表、读者阅读偏好表等，目前这类已经梳理出来的模型（宽表）已有20余个。最后是挖掘模型。常规理解，挖掘模型其实是偏应用的，但是在本馆的馆情分析中，例如读者常驻区域模型这类常用的挖掘模型，也把它规整到中台模型，以开放给其它人使用。在数据模型层这里所有的任务全部由定时调度器完成。为了在周期内生成报表之后可以通知相关人员，采用观察者模式来设计此部分。观察者模式可以很好的处理类似这样的一堆多的关系。通过设定报表 Subject 和关注人员 Observer 的关系，在 Observer 中自定义通知方法，根据用户设定调用短信或者邮件模块进行通知。

在数据服务方面，笔者将数据模型按照应用要求做服务封装，就构成了数据服务。这个跟业务中台中的服务概念是完全相同的，只是数据封装比一般的功能封装要难一点，毕竟OLTP功能的变化有限，而数据分析受业务因素的影响很大，变化更快。对于这部分开放的服务,笔者非常重视日志的搜集。所以在在日志生成部分，笔者引入责任链模式进行设计。顾名思义，责任链模式为请求创建了一个接收者对象的链。这种模式给予请求的类型，对请求的发送者和接收者进行解耦。在这种模式中，通常每个接收者都包含对另一个接收者的引用。如果一个对象不能处理该请求，那么它会把相同的请求传给下一个接收者，依此类推。在实践中，笔者创建了抽象类 AbstractLogger，并设定三种级别，分别是 INFO,WARNING和ERROR，它们都扩展了AbstractLogger。在日志生成时，每个记录器消息的级别是否属于自己的级别，如果是则相应地打印出来，否则将不打印并把消息传给下一个记录器。

但有数据模型和数据服务还是远远不够的，因为再好的现成数据和服务也往往无法满足前端个性化的要求。这时候就得授人以鱼。数据中台的最后一层就是数据开发，其按照开发难度也分为三个层次，最简单的是提供标签库（DMP），用户可以基于标签的组装快速形成分析报表，一般面向普通馆员。这里的标签库实际使用非常简单，馆员以拖拽的形式选择感兴趣的标签字段，比如性别、常驻区域、阅读偏好（分类号），通过添加逻辑运算符组合，并点击确定就可以快速生成目标读者群体。然后可以通过电话、短信、APP等推送方式对读者进行针对性的书籍、活动等推送服务。其次是提供数据开发平台，用户可以基于该平台访问到所有的数据并进行可视化开发，一般面向技术部SQL开发人员。最后就是提供应用环境和组件，让技术人员可以自主打造个性化数据产品，以上层层递进，满足不同层次人员的要求。在这个层次上的设计，由于时前后端分离下的单页应用SPA，因此引入了许多在前端中应用的较多的设计模式。一个是前端组件缓存的设计。为了缓存前端异步获取的数据，且在页面存货周期内不丢失数据，笔者建议团队采用单例模式确保缓存服务，来确保的全局唯一性。在文本输入框框验证场景，笔者通过封装一个验证类库，并引入策略模式减少多余条件代码的编写。这样在使用时，开发者就可以之间根据场景传入不同的验证算法实现灵活的验证实现。当然其他设计模式还由很多，比如使用工厂模式实现用户权限类的生成，使用代理模式是实现懒加载以及请求合并等。在这里限于篇幅不在展开。

通过引入设计模式来进行软件开发，提升了数据中台的设计的质量，满足了前端馆员多变的业务需求，同时极大减少了软件复用的成本。但是在实践中，笔者也发现，团队中不少开发人员在设计过程中，有时还是习惯于原有的设计方法，对模式的使用有些抵触，作为架构设计者，在于他们交流时，在管理和沟通上需要花很多时间。再者，在设计模式在应用过程中，需要对多个模式进行综合运用。这方面，笔者还缺少相关的经验。所以，在项目二期建设中，笔者将将继续推进各种设计模式的应用，做到融会贯通，确保项目二期能在软件质量上取得新的突破。