

摘要:

本人于 2018 年 1 月参与了中石化 XX 油田 XX 采油厂“用电管理系统”的项目建设，该系统建设目标是实现分单位、分线路、分系统评价、优化、考核，全面提升采油厂用电管理水平。在该项目组中我担任系统架构师一职，主要负责系统整体架构设计。本文以该项目为例，讨论软件设计模型的应用，包括问题的分析，以及如何选择和应用设计模式。整个系统采用了 MVC 设计模式作为主体设计模式，以单例模式、抽象工厂模式、迭代器模式、策略模式等多种模式作为辅助模式，设计模式的应用，使该系统具备良好的稳定性、可扩展性、可修改性，最终系统开发取得圆满成功。系统开发历时 10 个月，目前已上线稳定运行一年多，获得用户的一致好评，该项目也成功获得了当年度分公司科技进步创新一等奖。

正文:

“用电管理系统”项目是采油厂能源管控中心系统的一个子系统。能源管控中心是中石化集团公司十三五规划中的“能效倍增”计划在胜利油田分公司的示范应用项目，该示范项目能够在实现企业节能目标管理、能源计量统计、节能潜力识别、能效分析优化的同时，有效支撑企业实施节能技术改造、促进企业用能水平不断提升。“用电管理系统”的建设目标是建立覆盖厂、区两级用电管理一体化体系，实现分单位、分线路、分系统评价、优化、考核，达到电网运行质量实时监控、异常情况精准管控、能耗总量全面受控，按照运行产量的方式运行电量，全面提升采油厂用电管理水平。该项目功能设计参考 PDCA 闭环管理的理念，共设计包括用电计划、用电分析、用电优化、用电考核、设备管理等五大功能模块。

我作为单位技术骨干之一，主持并参与了项目计划制定、需求分析、整体架构设计与技术选型、底层设计、部分编码等多项工作。下面，我将首先介绍软件系统中常用的软件设计模式、每种类型特点及其所包含的设计模式具体含义，然后详细介绍“用电管理系统”的分析和设计过程中所采用的设计模式及具体实施效果。

设计模式 (Design Pattern) 是前辈们对代码开发经验的总结，是解决特定问题的一系列套路。它不是语法规则，而是一套用来提高代码可复用性、可维护性、可读性、稳健性以及安全性的解决方案。设计模式根据目的可以分为创建型模式、结构型模式和行为型模式 3 种，其中创建型模式用于描述“怎样创建对象”，它的主要特点是“将对象的创建与使用分离”。GoF 中提供了单例、原型、工厂方法、抽象工厂、建造者等 5 种。结构型模式用于描述如何将类或对象按某种布局组成更大的结构，GoF 中提供了代理、适配器、桥接、装饰、外观、享元、组合等 7 种结构型模式。行为型模式用于描述类或对象之间怎样相互协作共同完成单个对象都无法单独完成的任务，以及怎样分配职责。GoF 中提供了模板方法、策略、命令、职责链、状态、观察者、中介者、迭代器、访问

者、备忘录、解释器等 11 种行为型模式。设计模式还可以根据作用范围分为类模式和对象模式，其中类模式用于处理类与子类之间的关系，这些关系通过继承来建立，是静态的，在编译时刻便确定下来了。GoF 中的工厂方法、（类）适配器、模板方法、解释器属于该模式。对象模式用于处理对象之间的关系，这些关系可以通过组合或聚合来实现，在运行时刻是可以变化的，更具动态性。GoF 中除了以上 4 种，其他的都是对象模式。

项目启动后，由于油田系统对安全性、可靠性、可用性和扩展性要求很高，在架构设计工作的开始阶段，在公司技术顾问的建议下，我们决定采用三层分层架构作为该系统的软件架构模式，将系统分为应用层，中间层，数据层。在三层结构中的具体设计和开发过程中选择了许多经典的设计模式，以提高代码可复用性、可维护性、可读性、稳健性以及安全性。其中应用层负责具体业务和视图展示，如系统首页的电网拓扑展示、单位用电、线路用电等，其又分为视图层与业务逻辑层，主要使用了 MVC 设计模式实现数据、显示和处理分离，中间层除负责为应用层提供通用服务支持如系统管理服务，session 管理服务等之外，还提供预报警模型、能效优化模型、优化方案推送模型等中间件支持，使用了单例、迭代器、策略、模板方法等多种设计模式，数据层负责提供数据存储访问服务，如数据库服务，缓存服务，文件服务，搜索服务等，使用了单例、抽象工厂、策略等多种设计模式。接下来，我将分层次详细对几种主要设计模式的设计过程进行论述。

首先是应用层。应用层采用了 SpringMVC 这一基于 J2EE 平台的 MVC 框架，主要通过 Vue+Bootstrap 技术实现。SpringMVC 是一种实现了 Web MVC 设计模式的请求驱动类型的轻量级 Web 框架。通过 MVC 设计模式的使用，使开发人员可以专注于表示逻辑和业务逻辑的开发工作，实现了数据、显示和处理的分离，缩短了开发周期，减少了开发费用和维护费用，提高了开发的成功率。另外我们将系统根据应用进行水平划分，这有助于代码管理与维护。我们将系统分为用电分析，用电优化，用电考核，用电计划等十余个子系统，这里不再赘述。

其次是中间层。中间层采用了 Spring、Shiro 等服务框架实现。Spring 框架本身就使用了大量的设计模式，例如工厂方法、单例、适配器、装饰器、代理、观察者、策略、模板方法等。在实际编码过程中，我们也采用了较多的设计模式，如单例、代理、策略等。如站库信息查询等需要统计在线人数的功能中，我们实现了一个计数器功能来统计模块的使用人数，在此类对象需要被共享的场合，采用单例模式设计的话只允许创建一个对象，通过共享该对象可以节省内存，并加快对象访问速度。通过 Spring 框架与合理设计模式的组合使用，缩短了开发周期，减少了开发费用和维护费用，提高了开发的成功率。

最后是数据层。由于系统需要访问 Oracle、SqlServer、Mysql 等多种异构数据库平台，同时还需要满足复杂的数据管理需求，针对这种情况，我们选择了抽象工厂设计模式，抽象工厂模式提供一个接口，可以创建一系列相关或相互依赖的对象，可以针对 Oracle、MySQL、SQLServer 分别建立抽象工厂，如指定当前工厂为 Oracle 工厂，则创建出来的数据库连接，数据集等一系列的对象都是符合 Oracle 操作要求的，这样便于数据库之间的切换。另外我们选择了 MyBatis 作为持久层框架，实现了业务逻辑和数据访问逻辑分离，使系统的设计更清晰，更易维护，更易单元测试，极大提高了系统的可维护性。我们还选择 Redis 缓存提高了系统性能和访问效率。限于篇幅，不再赘述。

最终项目成功上线，正常运行了近一年，收到各方好评。尤其是标杆井推荐与耗电量推算功能，为采油厂的节能工作提供了良好的技术和数据支持。该项目也成功获得了当年度分公司科技进步创新一等奖。在系统的架构设计中，我们引入了层次架构的设计思想，有效地降低了维护成本，提高了系统的开放性，可扩展性，可重用性以及可移植性。当然还是存在一些问题的。如系统采用 http 协议，易被非法劫持，可以将协议修改为 https 来解决。还有就是旧系统在数据库层面存在很多无效的存储过程与定时任务，对数据库性能造成了影响。对这种高水平高价值的遗留系统问题，我们采用了重构整合的方式进行了改造，保证了系统的延续性。这些都是我在今后的系统架构设计工作中需要注意与改进的地方，也是日后我应该努力的方向。