

分层技术在计算机软件开发中的应用策略研究

胡文华

浙江艺高文化创意有限公司 浙江杭州 310000

摘要：本文主要聚焦分层技术在计算机软件开发里的应用策略开展研究，首先对分层技术进行全面概述并介绍其定义、基本原理以及常见架构模型，接着分析分层技术在企业级应用、Web应用程序、移动应用和嵌入式系统开发等不同场景中的应用特点与需求，在此基础上深入探讨分层技术的应用策略，涵盖依据软件功能与性能等因素合理划分层次、遵循设计原则定义清晰接口、结合项目需求选择合适的分层架构模型，还有注重层次间通信机制、数据格式和错误处理等交互细节，研究目的是为计算机软件开发有效运用分层技术提供理论支持与实践指导以提升软件开发的质量与效率。

关键词：分层技术；计算机软件开发；应用

在如今的数字化浪潮席卷之下计算机软件开发行业发展得很是蓬勃，不过同时也面临着各种各样十分严峻的挑战。从一方面来讲，用户的需求变得日益多样化而且复杂多变，软件需要去满足的功能点也在不断增加。从另一方面来说，软件的规模在持续不断地扩大，代码量呈现出几何级数一样的增长态势。这样的情况让软件开发的过程变得越来越复杂，后期的维护成本急剧上升，并且软件的质量和可扩展性难以得到保障。在这样的背景条件之下，分层技术就应运而生了。它通过把软件系统进行合理的划分使其成为不同的层次，实现了功能模块的解耦，为解决上述所提到的问题提供了一个有效的途径，对于提升软件开发的效率、保障软件的质量、增强软件的可维护性和可扩展性有着不可忽视的重要意义。

一、分层技术概述

分层技术会把软件系统划分成多个层次，每层都具备特定功能和相应职责，各层之间通过接口进行通信交互。其基本原理包括保证每个层次的独立性以降低耦合度，明确各层次之间的依赖关系并遵循特定规则，规范接口定义从而确保信息准确传递。在常见的分层架构模型当中，三层架构包含表示层、业务逻辑层和数据访问层，在Web开发领域得到广泛应用，有利于实现职责分离和方便维护，MVC架构由模型、视图和控制器共同协作，能够实现业务逻辑和用户界面的分离，可有效提升代码的可维护性，除此之外还有N层架构、微服务架构等，它们适用于不同规模和不同需求的软件系统^[1]。

二、分层技术在计算机软件开发中的应用场景分析

1. 企业级应用开发

企业级应用一般有着业务逻辑复杂、数据量庞大、用户并发访问高等特性，这就要求系统必须具备高可用性、可扩展性以及安全性，为了应对这些挑战，分层技术得到广泛采用，像借助多层架构（例如表现层、业务逻辑层、数据访问层）来实现业务逻辑和数据存储的分离，从而降低耦合度并提升可维护性，中间件技术（例如消息队列、分布式缓存）的应用进一步提高了系统的可扩展性和可靠性，能够支持高并发场景下的稳定运行，微服务架构和容器化技术也渐渐成为企业级开发的主流选择，以此实现更灵活的部署和资源管理^[2]。

表1 企业级应用分层技术对比

分层	功能	关键技术
表现层	用户交互与界面展示	React、Angular、Vue.js
业务逻辑层	处理核心业务规则与流程	Spring、EJB、gRPC
数据访问层	数据库操作与数据持久化	JPA、Hibernate、MyBatis
中间件层	提供分布式支持与系统集成	Rabbit、Kafka、Redis、Dubbo

2. Web应用程序开发

Web应用程序的架构特点主要体现于前后端分离、微服务化以及云原生技术应用上，前后端分离采用RESTful API + SPA架构提升了开发效率让前端React、Vue.js和后端Spring Boot、Django能够独立演进，微服

务化通过把单体应用拆分为独立服务如用户服务、订单服务再结合容器化 Docker 和编排工具 Kubernetes 提高系统可扩展性和可维护性，分层技术在 Web 开发当中被广泛应用前端框架负责表示层交互后端框架如 Spring、Express 处理业务逻辑和数据访问而 API 网关如 Kong、Spring Cloud Gateway 统一管理服务间通信保障安全性和负载均衡。

表 2 Web 应用程序分层架构及关键技术

分层	功能	关键技术
表示层	用户界面渲染与交互	React、Vue.js、Angular
业务逻辑层	核心业务处理与流程控制	Spring Boot、Express、Flask
数据访问层	数据库操作与持久化管理	JPA、MyBatis、Hibernate、Sequelize
API 网关层	路由、认证、负载均衡	Kong、Nginx、Spring Cloud Gateway

3. 移动应用开发

移动应用有着跨平台兼容性要求高、需极致优化用户体验、数据安全防护压力大等明显特点与开发需求，跨平台兼容性要求应用适配多种操作系统与不同设备型号，用户体验优化涵盖界面流畅性和操作便捷性等多个方面，数据安全涉及用户隐私保护以及数据传输加密等关键环节，分层技术在移动应用开发里起着重要作用，通过采用分层架构可将代码按功能进行模块化设计，以此降低耦合度并便于后续维护与扩展，利用本地存储技术如 SQLite 数据库、文件存储等能提高数据本地访问效率，同时结合云服务可实现数据的云端同步与共享，方便用户在不同设备间实现无缝切换，还能借助云端强大的计算与存储能力为移动应用提供更丰富功能支持^[3]。

4. 嵌入式系统开发

嵌入式系统存在资源受限情况像存储空间小计算能力弱、实时性要求高得在规定时间内响应并完成任务、硬件相关性强紧密依赖特定硬件平台等特点，这给开发带来很多难点比如需精细优化资源使用、精准把控任务执行时间、深度适配硬件。分层技术在嵌入式系统开发当中应用广泛而且有效，采用分层架构能够将软件与硬件进行分离，降低开发复杂度同时提高软件可移植性，利用实时操作系统可合理调度任务确保系统实时响应，通过中间件技术能为不同模块提供标准化接口，实现模块间高效通信与协同工作，减少模块耦合进而提升开发效率与系统稳定性，助力嵌入式系统更好满足复杂应用

需求。

三、分层技术在计算机软件开发中的应用策略

1. 合理划分层次

合理进行层次划分是分层技术应用当中的关键所在，依据软件功能划分时要深入剖析功能需求，把关联紧密的功能归为同一层次让各层职责清晰、功能边界明确，例如将数据处理、业务逻辑处理分别置于不同的层次，在平衡性能和可扩展性方面要结合软件规模和预期发展，避免层次过多增加通信开销从而降低性能，也需要防止层次过少限制软件功能扩展的可能性，同时层次划分应该遵循开闭原则进行设计时预留扩展点，当添加新功能的时候仅需在扩展点处新增代码而无需修改现有层次，这样既能够保证系统的稳定性又能快速响应需求变化提升开发效率^[4]。

2. 定义清晰的接口

在软件开发里应用分层技术时定义清晰的接口极为重要，在接口设计原则方面要遵循单一职责原则，让一个接口只专注负责一项功能以避免功能冗余与耦合，同时还需要遵循接口隔离原则，把庞大又臃肿的接口拆分成多个小接口使功能明确简洁且易用，在接口文档规范上详细准确的文档是必不可少的，应该涵盖接口功能描述输入输出参数及调用方式等内容来促进开发人员高效沟通协作，接口版本管理也是不能忽视的，需要制定合理策略像采用版本号标识渐进式更新等方式，以此确保接口变更时不会引发系统兼容性问题保障软件系统稳定运行。

3. 选择合适的分层架构模型

选择适合的分层架构模型对软件项目成功来说十分关键，在项目需求分析阶段要综合考量软件规模、复杂度以及性能要求等因素，小型简单项目可以选择三层架构这类简洁的模型，大型复杂项目微服务架构或许更为合适，它能够实现功能模块解耦从而提升开发效率，在进行架构模型评估的时候要从开发效率、可维护性、可扩展性以及性能等多个维度考量，对比不同模型的优缺点并以此为依据做出选择，选定架构模型之后在实际项目当中还需要进行优化调整，比如针对特殊性能需求优化数据传输机制，或者根据业务变化调整层次间的交互方式，确保架构模型能够充分满足项目特殊需求以保障软件项目顺利推进。

4. 注重层次间的通信与交互

注重分层架构软件系统层次间的通信与交互对其稳

定运行非常重要，在通信机制选择方面，同步通信适用于对实时性要求高且数据量小的场景能保证数据及时处理，异步通信则适合处理耗时操作可避免阻塞并提高系统并发能力，消息队列可实现解耦与削峰填谷常用于分布式系统，同时定义统一的数据格式规范是不可或缺的能确保不同层次准确传输和解析数据避免因格式不一致引发错误，此外要建立有效的错误处理与容错机制在层次间通信出现异常时能及时捕获并处理错误保证系统稳定运行增强系统的健壮性。

结论

对分层技术在计算机软件开发里的应用策略和成果进行深入探讨，在应用策略方面合理划分层次且明确定义接口、科学选择架构模型并注重层次间通信交互可有效发挥分层技术优势，分层技术在企业级开发、Web开

发、移动应用开发及嵌入式系统开发等场景中有着广泛应用，借助模块化设计实现功能解耦进而提升开发效率和代码质量，在实际项目当中应用分层技术明显提高软件的可维护性与可扩展性、降低开发复杂度和维护成本，有力证明分层技术是提升计算机软件开发水平的关键手段。

参考文献

- [1] 吴纯璐, 夏焯. 分层技术在计算机软件开发中的应用[J]. 信息记录材料, 2024, 25(11): 222-224.
- [2] 刘晓晓. 分层技术在计算机软件开发中的应用[J]. 信息记录材料, 2024, 25(02): 193-195.
- [3] 张晓, 李军丹, 张富瑞. 分层技术在计算机软件开发中的应用研究[J]. 软件, 2023, 44(10): 161-163.
- [4] 张贝. 分层技术在计算机软件开发中的应用[J]. 网络安全和信息化, 2023, (10): 106-108.

