

**《软件项目管理》个人作业**



**个人作业（1）**

**专 业 软件工程**

**姓 名 高楚劼**

**学 号 2021141240067**

**指导老师 毌攀良**

**成绩分数**

2024 年 12 月 31 日

**一、Serverless架构的定义**

Serverless架构，即无服务器架构，是一种云计算模型，开发者无需管理底层服务器等基础设施，而专注于应用程序的开发和业务逻辑的实现。在这种模型中，云服务提供商负责动态分配、管理和扩展计算资源，确保应用程序的高可用性和可扩展性。



图1 Serverless架构应用运行原理

Serverless架构主要包括以下两种服务形式：

①函数即服务（FaaS，Function as a Service）： 开发者将应用逻辑划分为独立的函数，这些函数在特定事件触发时执行。云服务提供商负责函数的运行环境、资源分配和自动扩展。开发者仅需关注函数的实现，无需关心服务器的配置和管理。常见的FaaS平台包括AWS Lambda、Azure Functions和Google Cloud Functions等。

②后端即服务（BaaS，Backend as a Service）： 开发者通过调用云服务提供商提供的后端服务，如数据库、身份验证、存储等，来构建应用程序。这些服务由云提供商全权管理，开发者无需处理服务器端的实现和维护。例如，Firebase提供了实时数据库、身份验证和托管等BaaS服务。

**二、Serverless架构的应用场景**

Serverless架构在现代软件开发中展现出广泛的应用潜力，特别适用于以下场景：

**2.1 事件驱动的应用程序**：

Serverless架构非常适合处理事件驱动的任务，即在特定事件发生时自动触发相应的函数执行。

·数据库变更响应： 当数据库中的数据发生插入、更新或删除操作时，Serverless函数可以被触发以执行相应的业务逻辑。例如，在用户上传新文件至存储桶时，自动生成缩略图或提取文件元数据。

·物联网（IoT）数据处理： 物联网设备通常会产生大量实时数据。Serverless架构可以实时处理这些数据，例如对传感器数据进行分析、过滤和存储，以便后续分析和监控。

·实时流数据处理： 在需要对实时数据流进行分析或转换的场景中，Serverless函数可以用于处理来自消息队列或数据流平台的数据，实现实时监控、报警或数据清洗等功能。

**2.2 突发或不可预测的服务使用量：**

对于流量模式不可预测或存在突发高峰的应用，Serverless架构提供了自动弹性伸缩能力。

·高并发请求处理： 在电商促销活动或抢购等场景下，系统需要在短时间内处理大量用户请求。Serverless架构能够根据实际负载自动扩展计算资源，确保服务的稳定性和响应速度。

·按需资源分配： 对于使用频率不固定的应用，Serverless架构按需分配资源，避免了传统服务器架构中资源闲置或不足的问题，提高了资源利用率。

**2.3 短暂、无状态的应用：**

Serverless架构特别适合执行短暂且无状态的任务，这些任务通常在短时间内完成，不需要维护持久的会话状态。

·定时任务调度： 通过设置定时触发器，Serverless函数可以在预定时间执行任务，例如每日的数据备份、日志清理或定期报告生成。

·批处理作业： 对于需要处理大量数据的批处理任务，Serverless架构可以并行执行多个函数实例，加速任务完成。例如，批量图像处理、视频转码或数据格式转换等。

**2.4 需要快速开发、迭代的业务：**

在竞争激烈的市场环境中，快速开发和迭代新功能对于业务成功至关重要。Serverless架构通过简化基础设施管理，加速开发流程。

·Web应用后端： 开发者可以利用Serverless架构快速构建Web应用的后端服务，处理用户请求、与数据库交互，并实现业务逻辑，而无需管理服务器。

·移动应用后端： 移动应用通常需要一个灵活且可扩展的后端来支持各种功能。Serverless架构允许开发者专注于功能实现，快速响应市场需求，缩短产品上市时间。

**三、Serverless架构的优缺点分析**

**3.1 优点**

成本效益：采用按需付费模式，用户仅需为实际使用的计算资源和执行时间付费，减少了空闲成本。

自动扩展：Serverless架构可以根据需求自动扩展，无需手动配置，确保应用在面对突发流量或长期增长时都能保持高性能和稳定性。

开发效率：开发者可以专注于代码，而无需管理服务器和运行时环境，提高开发效率。

快速迭代：支持快速部署新功能，加速产品上市时间。

**3.2 缺点**

冷启动问题： 在无服务器环境中，当一个函数在一段时间内没有被调用，云服务提供商可能会将其从内存中移除，以节省资源。当下一次请求到达时，这个函数需要重新加载到内存中，初始化执行环境，这一过程会产生额外的延迟，即“冷启动延迟”。

资源限制： 云服务提供商通常会为无服务器函数设置最大内存使用量和最长执行时间限制。超过这些限制会导致函数执行失败，这对处理复杂或长时间运行的任务构成挑战。

监控与调试复杂性： 在Serverless环境中调试和监控应用程序可能比传统的服务器环境更复杂。由于函数的短暂运行特性，开发者难以复现问题环境进行调试，尤其是冷启动相关的问题。

1. **Serverless架构所需的技术栈**

在采用Serverless架构时，以下技术组件是构建和运行无服务器应用的关键：

**4.1 编程语言**

Serverless平台通常支持多种主流编程语言，开发者可以根据项目需求选择最适合的语言进行开发。常用的语言包括：JavaScript/Node.js、Python、Go、Java、C#

**4.2 函数即服务（FaaS）平台：**

FaaS平台允许开发者部署和执行独立的函数，这些函数在特定事件触发时运行。常见的FaaS平台包括：

·AWS Lambda：支持多种编程语言，能够与AWS生态系统中的其他服务深度集成。

·Google Cloud Functions：提供快速部署和自动扩展功能，适用于构建轻量级、事件驱动的应用。

·Azure Functions：支持多种编程语言和开发框架，适合企业级应用和微服务架构。

**4.3 API网关：**

API网关充当客户端与后端服务之间的中介，负责请求路由、协议转换和安全管理。主要的API网关服务有：

·Amazon API Gateway：用于创建、发布、维护、监控和保护RESTful或WebSocket APIs。

·Azure API Management：提供全面的API管理功能，包括安全、分析和监控。

**4.4 事件源服务：**

事件源服务用于触发无服务器函数执行，常见的事件源包括：

·对象存储：如Amazon S3、Azure Blob Storage，当有新文件上传时触发函数处理。

·消息队列：如Amazon SQS、Azure Service Bus，用于解耦和协调不同服务之间的通信。

**4.5 身份验证和授权：**

确保应用的安全性，管理对资源的访问控制，主要服务有：

·AWS Identity and Access Management (IAM)：用于控制对AWS服务和资源的访问。

·Azure Active Directory：提供身份验证、应用访问管理和安全性功能。

**4.6 监控和日志服务：**

用于监控函数性能、记录日志和诊断问题，常用的服务包括：

·Amazon CloudWatch：提供监控、日志记录和警报功能，帮助追踪应用的运行状况。

·Azure Monitor：用于收集、分析和响应云环境中的遥测数据。

**4.7 开发工具和框架**

选择合适的开发工具和框架对提高开发效率和保证代码质量至关重要。在Serverless架构中，以下工具被广泛使用：

·Serverless Framework： 提供了一种简单、统一的方式来定义、部署和管理Serverless应用，支持多个云平台，能够大大简化跨云部署和管理的复杂性。

·AWS Serverless Application Model (SAM)： AWS提供的一种Serverless应用开发和部署工具，它简化了在AWS上创建、管理和部署Serverless应用的流程。

·Cloud Development Kit (CDK)： 一种基于软件开发的云资源管理方式，它允许开发者使用熟悉的编程语言来定义云资源和Serverless应用，提高了开发的灵活性和效率。

**4.8 数据存储**

在Serverless架构中，选择合适的数据库和存储解决方案至关重要。无服务器数据库服务，提供了高可扩展、高性能、全托管的数据库服务，非常适合Serverless应用，例如：

·Amazon DynamoDB： 一种完全托管的NoSQL数据库服务，支持键值和文档数据模型。它可以无缝扩展，适合处理大规模数据的应用场景。

·Google Firestore： 提供了实时数据库同步功能，非常适合需要实时数据更新的应用。

·Azure Cosmos DB： 提供全球分布式、多模型数据库服务，支持多种数据模型和API，适用于需要低延迟和高可用性的应用。

**4.9 持续集成与交付（CI/CD）**

在Serverless架构中，自动化的持续集成与交付流程有助于提高开发效率和代码质量。常用的CI/CD工具包括：

·Jenkins： 开源的自动化服务器，可用于构建、部署和自动化各类项目。

·GitLab CI/CD： GitLab提供的内置CI/CD功能，支持从代码提交到部署的全流程自动化。

·AWS CodePipeline： AWS提供的持续交付服务，可实现代码的自动化构建、测试

1. **使用Serverless架构的系统/软件**

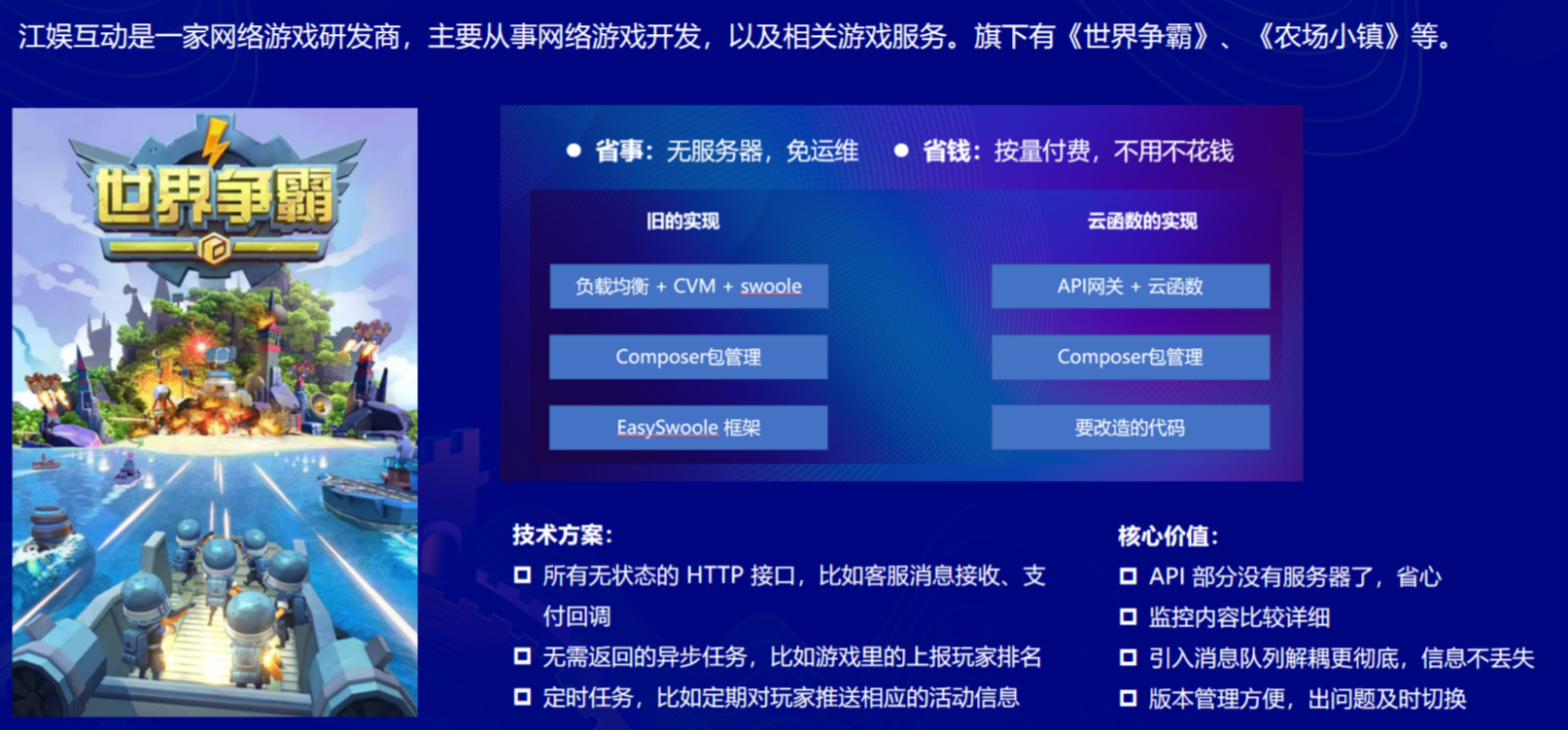
**5.1 猎豹移动：**



猎豹移动前端团队在其技术框架Koot.js中集成了Serverless架构，利用Serverless Framework封装，实现了服务端渲染（SSR）项目的高效部署。

通过Serverless架构，猎豹移动有效降低了前端团队对后端和运维的依赖，简化了部署流程，提高了开发效率和项目的可维护性。

**5.2 江娱互动：**



江娱互动将其聊天服务的API部分迁移至云函数平台，利用Serverless架构的弹性伸缩特性，成功应对了突发的高并发请求，提升了系统的稳定性和成本效益。

采用Serverless架构后，江娱互动的聊天服务在面对用户量激增时，能够自动扩展资源，确保服务的高可用性，同时降低了服务器维护成本。

**5.3 新东方：**



新东方在其在线教育平台中，采用Serverless架构进行音视频转码处理。通过云函数与对象存储的结合，实现了自动化的音视频处理流程，满足了高峰期的业务需求，降低了运维成本。

Serverless架构使新东方能够灵活应对在线教育业务的波峰波谷，按需分配计算资源，提高了资源利用率，减少了约20%的成本支出，并提升了系统的弹性和可靠性。

**5.4 丰图科技（深圳）有限公司：**

丰图科技通过华为云函数工作流FunctionGraph构建了质检推理系统，提升了业务的稳定性，资源利用率提高了30%，可靠性得到显著提升。

利用Serverless架构，丰图科技实现了质检推理系统的自动弹性扩展，按需使用计算资源，降低了运维复杂度和成本，提高了系统的稳定性和可靠性。

**5.5 阿里云函数计算控制台：**

阿里云在其函数计算控制台中，采用Serverless架构实现了全球用户的低延迟访问，降低了运维成本，提高了研发效率。

通过Serverless架构，阿里云函数计算控制台实现了全球范围内的快速响应，减少了运维工作量，提升了系统的可扩展性和用户体验。

**六、对Serverless架构的评价**

Serverless架构，即无服务器架构，作为云计算领域的创新模式，正逐渐改变着应用程序的开发和部署方式。它允许开发者将应用程序代码部署到云平台上，而无需关心服务器资源的配置和管理，从而降低了运营成本。此外，Serverless架构能够根据应用程序的实际负载自动调整资源规模，实现自动扩缩容功能，提高应用程序的性能和稳定性。开发者可以专注于业务逻辑的实现，而无需花费大量时间配置和管理服务器资源，加快了开发和部署的速度。

然而，Serverless架构也存在一些挑战。由于应用程序需要经过云平台的调度和分发，相对于直接在本地服务器上运行，可能会引入一定的延迟。此外，虽然云平台提供了自动化的管理和弹性伸缩功能，但在某些情况下，开发者可能需要对应用程序进行更精细的管理和调试，这可能需要更多的技术能力和经验。

因此，在选择是否采用Serverless架构时，需要根据具体的应用场景和需求进行权衡。对于那些对性能和稳定性要求较高、需要快速迭代和灵活部署的应用程序，Serverless架构可能是一个不错的选择；而对于需要自行管理服务器资源和高度定制化部署的应用程序，则可能不太适合使用Serverless架构。