

**本科毕业设计（论文）开题报告**



**题 目 基于TCP的远程调用RPC框架研究**

|  |  |  |  |
| --- | --- | --- | --- |
| **学生姓名** | **郑剑** | **学　号** | **202031070390** |
| **教学院系** | **计算机科学学院** | | |
| **专业年级** | **软件工程2020级** | | |
| **指导教师** | **梅广超** | **职　称** |  |
| **单　　位** | **计算机科学学院** | | |

|  |
| --- |
| **设计（论文）的选题意义及国内外研究现状：**  **选题目的：**  随着中国互联网行业的迅速发展，分布式系统在各个领域得到了广泛应用，无论是大型电商系统还是社交系统，网络节点之间的高效通信和远程方法调用变得尤为重要。在这一背景下，RPC作为一种在分布式系统中应用广泛的通信协议，发挥着至关重要的作用。本论文旨在深入研究并设计一种基于TCP的RPC框架，以更好地满足中国互联网环境下的需求和挑战。  在中国互联网生态中，我面临着不断增长的用户规模、多样化的应用场景以及对通信效率和服务可靠性的高要求。因此，本研究将专注于设计一个基于TCP的RPC系统，旨在提供高效的分布式通信、实现简单且可扩展的服务调用、确保高可靠性和容错性，并推动RPC技术在中国互联网行业的广泛应用。  通过实际实现基于TCP的RPC系统，我期望能够为中国互联网应用提供更优越的解决方案，提升分布式系统的通信效率、可靠性和可扩展性。这将有助于促进中国互联网技术的进步，推动数据驱动的科学研究、促进商业应用的创新，为社会发展提供更强有力的支持。  **选题意义：**  推动分布式系统技术的进步： 通过深入研究并实现基于TCP的RPC框架，我们将为分布式系统技术的发展做出贡献。提升通信效率、服务可靠性和可扩展性将推动分布式系统的发展，并在学术界引领相关领域的研究。  满足中国互联网环境下的需求： 中国互联网用户规模巨大，应用场景丰富多样。我们的研究旨在设计适应中国互联网环境的基于TCP的RPC系统，以满足不断增长的用户需求、提高服务质量，并促进互联网产业的创新与发展。  促进数据驱动的科学研究： 通过提高分布式系统通信效率，我们有望为数据驱动的科学研究提供更强有力的支持。高效的RPC框架将有助于更快地处理大规模数据，促进科学家们在各领域的研究成果。  服务社会发展的需要： 高可靠性和容错性是分布式系统的重要特征，对于社会发展至关重要。我们的研究将有助于构建更稳定、可靠的互联网基础设施，支持社会各个领域的数字化转型。  **国内外研究现状：**  1. 国外研究现状：  远程过程调用（Remote Procedure Call，RPC）的发展历史可以追溯到20世纪80年代。  在1984年，Bruce Jay Nelson在他的博士论文《The Design and Implementation of a High-Performance Transport Protocol》中首次提出了远程过程调用（Remote Procedure Call，RPC）的概念[1]中将远程过程调用（RPC）定义为一种协议，它允许一个程序能够导致另一台主机上的程序执行过程，而执行过程的环境看起来就像是本地过程调用。RPC抽象了底层的网络通信细节，使得开发分布式系统时，远程通信可以像本地函数调用一样直观。在论文中，Nelson详细讨论了RPC的设计目标，它旨在提供一种简单、高效、通用的远程交互机制。他强调了RPC在分布式计算中的重要性，尤其是在提高网络服务效率和简化复杂网络协议栈时的作用。Nelson还探讨了RPC协议的各种实现细节，包括参数传递、错误处理、传输协议选择等方面。  1985年：Andrew Birrell和Bruce Jay Nelson在Xerox PARC（帕克研究中心）开发了一种名为“Courier”的RPC系统。这是RPC的第一个实现，它使用了一种基于消息传递的通信协议。  1991年：Sun Microsystems发布了NFS（Network File System），这是一种基于RPC的分布式文件系统。NFS的成功推动了RPC的发展，并促使其他公司和组织开始关注RPC技术。  1995年：Microsoft发布了DCOM（Distributed Component Object Model），这是一种基于RPC的分布式对象通信技术。DCOM在Windows平台上广泛应用，为RPC的发展做出了重要贡献。  2000年：Google发布了GFS（Google File System），这是一种基于RPC的分布式文件系统。GFS的成功启发了Google开发更多基于RPC的分布式系统，如MapReduce和Bigtable。  2006年：Facebook发布了Thrift，这是一种开源的RPC框架。Thrift支持多种编程语言，并提供了强大的代码生成工具，使得开发者可以轻松地定义和使用RPC接口。  2010年：Google发布了gRPC，这是一种高性能的RPC框架。gRPC使用了基于HTTP/2的协议，并支持多种编程语言。它在性能和可扩展性方面具有显著优势，被广泛应用于云计算和微服务架构。  上述的thrift和gRPC也是现在行业内使用的最多的远程调用框架。国内的很多研发的RPC框架也是学习国外优秀开源框架的思想，结合内部业务需求进行开发的。  综上所述，自从RPC思想提出到如今RPC的身影无处不在，期间经历了20多年的发展，国外顶尖互联网公司和专家非常重视RPC这项技术， RPC技术在互联网领域的应用和发展已经取得了显著的成果，并且在未来仍有巨大的发展潜力。无论是对于构建高性能、低延迟的互联网应用，还是对于构建复杂、多样化的互联网应用，RPC都将继续发挥着重要的作用。  2. 国内研究现状：  随着互联网产业的快速发展，中国的互联网公司面临着越来越复杂的业务需求和技术挑战。为了解决这些问题，许多公司开始研究和应用RPC技术，以实现服务化架构和微服务架构。这种趋势推动了RPC技术的研究和应用在中国的快速发展。  2008年：阿里巴巴开始开发Dubbo框架，这是中国最早的开源RPC框架之一。Dubbo引入了服务治理的概念，并提供了丰富的服务治理功能。  2010年：Dubbo框架开源，成为中国最受欢迎的RPC框架之一。许多互联网公司开始使用Dubbo构建自己的服务化架构。  Dubbo对中国互联网的发展产生了深远的影响，推动了服务化架构和微服务架构的发展，提高了系统的性能和可靠性，并推动了开源社区的发展。未来，随着技术的不断进步和应用场景的不断扩展，Dubbo在中国的互联网产业中的影响将更加广泛。  随着这么一个现象级的产品出现，国内各个公司都开始在开源框架的思想上，开发复合自己公司业务的rpc框架，2012年：Motan框架开源，提供了更加灵活的服务治理和负载均衡功能。Motan在中国的互联网公司中得到了广泛的应用。  2014年：SofaRPC框架开源，提供了更加高效的RPC通信和服务治理功能。SofaRPC在中国的互联网公司中得到了广泛的应用。  综上所述，国内的RPC框架以阿里的dubbo为一个节点，后续各种rpc框架运行而生，有服务于自己公司业务的rpc框架，也有服务于大众开发程序员的开源框架，这些框架都为国内RPC技术做出了发展。 |
| **主要研究内容：**  本研究的主要目标是设计、实现和优化一种基于TCP的RPC框架，以满足中国互联网环境下分布式系统的通信需求。为了实现这一目标，我们将重点关注以下几个方面的研究内容：  协议设计与优化： 我们将深入研究RPC通信协议的设计，重点考虑在TCP协议基础上的优化策略，以提高通信效率。这包括数据序列化、压缩算法的选择以及协议头的优化等方面。  服务调用的简化与可扩展性： 我们将探索设计简单且易于使用的服务调用机制，以降低开发者使用RPC框架的门槛。同时，我们将考虑如何实现框架的良好可扩展性，使其能够适应不断变化的业务需求。  服务注册与发现机制：实现自动的服务端注册机制与客户端服务发现机制，并利于负载均衡选择合理的机器进行调用。  错误日志监控：当调用数据流转过程出现错误时，通过健康检查机制和埋点快速定位错误。  实际系统实现与评估： 我们将基于研究内容实际实现一个基于TCP的RPC框架原型，并通过大规模的实际测试和评估来验证其性能和可用性。这将涵盖在实际应用场景下的测试，以确保框架的实际适用性。 |
| **拟采用的研究思路（方法、技术路线、可行性分析论证等）:**   1. **方法**   文献综述与案例分析： 我们将深入挖掘相关文献，系统性地梳理国内外基于TCP的RPC框架的设计和优化方法。通过对各种案例的详细分析，我们将汲取先前研究的经验教训，理解已有方案的优缺点。  实证研究： 为了更好地理解协议设计在实际应用中的表现，我们计划进行原型实验。通过构建一个基于TCP的RPC系统原型，我们将模拟不同网络环境、负载和使用场景，以验证协议设计的性能和可用性。   1. **技术路线**： 2. RPC总体框架：   服务调用方：服务调用方是需要使用某个服务的客户端。当服务调用方需要使用某个服务时，它会向注册中心查询服务提供方的地址，并通过RPC调用服务提供方提供的服务。服务调用方通常会将请求数据传递给服务提供方，并接收服务提供方返回的响应数据。  服务提供方：服务提供方是提供服务的服务器。服务提供方会向注册中心注册自己提供的服务，并监听RPC调用请求。当服务提供方收到服务调用方的请求时，它会处理请求数据，并返回响应数据给服务调用方。  注册中心：注册中心是一个中心化的服务发现和注册系统。服务提供方会向注册中心注册自己提供的服务，并定期更新自己的状态信息。服务调用方在需要使用某个服务时，会向注册中心查询服务提供方的地址，并获取服务提供方的地址信息。注册中心通常会提供一个API，供服务提供方和服务调用方进行注册和查询操作。   1. 客户端和服务端架构     调用方客户端调用方法就像处理本地方法一样，但是底层原理实际上是通过动态代理的技术，框架生成一个代理类，在代理类里面会处理对应的序列化、压缩、协议封装等逻辑，最终通过sockets连接传输到对应的服务端。  服务提供方这边会有一个监听服务监听sockets通道，当有数据到达时，socket将对应的二进制数据传输到server slub，在其中进行反序列化、协议解析，并通过反射技术调用服务端对应的方法进行执行，完成后将结果通过原路径返回给服务调用方即可。  客户端用例包括 方法调用、请求重试、服务发现、负载均衡和日志监控。    其中，方法调用的用例包括调用远程方法并得到对应的返回值；请求重试涉及到当方法调用失败后，客户端如何进行的重试策略；服务发现用例指的是客户端需要感知到自己调用服务方的服务列表即ip地址，并需要提供负载均衡策略调用具体的一台机器服务；日志监控用例包括error日志监控和调用次数的监控。  服务端用例包括服务发布、请求处理、并发处理和日志监控    其中，服务发布包括将自己的服务方法签名和机器标识注册到注册中心，请求处理包括处理来自不同客户端发来的方法执行请求，并发处理用于处理同时接收多个请求时的并发安全处理，日志监控包括error日志的处理和使用本服务的客户端列表查询，  时序图：    RPC框架的核心是客户端成功调用服务端的服务。根据UML活动图（图 3.3）的不同职责划分为server、server slub、注册中心、client和client slub五个泳道。一次rpc的调用成功，需要依次经历以下步骤：   1. client调用接口方法，client slub生成代理对象 2. client slub根据配置从注册中心获取调用目标机器列表。 3. client slub使用负载均衡功能获取到具体的机器。 4. client slub 封装请求体，并进行序列化、压缩、协议encode等后进行TCP网络传输。 5. server slub接受请求后，对请求参数进行协议decode、解压、反序列化等操作后，封装对应的rpc请求体。 6. server slub 根据rpc请求体中的请求签名，利用反射技术，拿到server的反射对象，并调用具体方法的执行，最终拿到执行结果 7. server slub根据执行结果，封装rpc响应，并进行序列化、压缩、协议encode，最后通过TCP返回给client slub。 8. client 通过协议decode、解压、反序列化等操作后，返回数据给client。   以上对于用户使用client来说，屏蔽了client slub细节，使得调用远程方法像调用本地方法一样方便。  涉及技术：  Java、Spring、SPI、netty、序列化、压缩、传输协议、zookeeper、负载均衡 可行性分析1.技术可行性 目前，上述技术已经由开源社区进行了维护，并且在工业界也得到了广泛的认可和应用。这表明这些技术已经相对成熟，具备了实施项目的技术基础。由于采用的技术是开源免费的，这意味着不需要支付昂贵的许可费用，从而降低了项目的成本。同时，开源技术也意味着有一个庞大的社区支持和更新，可以获得更多的技术支持和帮助。由于技术是开源免费的，我们不需要担心侵权的风险。这为项目的实施提供了法律保障，避免了潜在的法律纠纷。在限制条件下，我们需要确保必要的软件和硬件资源能够支持系统的开发工作。这包括硬件设备、操作系统、开发工具等。我们需要评估这些资源的可用性和可靠性，以确保项目的顺利实施。 2.操作可行性 从操作层面来讲，对于服务调用方，只需要引入rpc jar包就可以调用远程服务，不需要关注rpc本身的网络通信、反射、序列化等实现逻辑，对用户透明。  对于服务提供方来讲，用户只需要制定相应的接口，引入jar包，后续调用远程方法只需要像调用本地方法一样简单。 3.经济可行性 由于上述工具已经开源，我们不需要支付昂贵的许可费用。这意味着我们可以节约大量的成本，从而降低项目的总体成本。本系统能够按需部署，这意味着我们可以根据实际需求进行灵活的部署和扩展。这不仅可以降低项目的运维成本，还可以提高生产效率，从而带来更高的经济效益。若系统正式部署上线，它将为我们带来可观的经济效益。这可能包括提高生产效率、降低运维成本、增加收入等方面。因此，从长远来看，这个项目具有较好的经济价值。我们还需要考虑项目的投资回报率。通过评估项目的收益和成本，我们可以计算出项目的投资回报率。如果投资回报率较高，那么这个项目在经济上是可行的。 |
| **设计（论文）的预期结果（成果）:**   1. 完成《基于TCP的远程调用RPC框架研究》论文。 2. 搭建可以验收题目研究工作的系统及开发环境。 3. 完成《基于TCP的远程调用RPC框架研究》的代码开发。 |
| **设计（论文）的工作进度安排:**  2023年12月27日至2024年1月16日 文献调研  2024年1月16日至2024年1月20日 撰写开题报告  2024年1月21日至2024年1月29日 设计方案验证  2024年2月1日2024年3月1日 数据库设计，框架搭建，需求分析，概要设计  2024年3月1日2024年4月15日 详细设计，系统开发与测试  2024年4月16日2024年5月15日 撰写毕业论文  2024年5月16日至2024年5月25日 修改论文，查重，提交等任务  2024年5月25日至2024年6月1日 答辩  **参考文献**  [1] Kevin R. Fall W. Richard Stevens.TCP/IP详解 卷1：协议 .机械工业出版社,2016  [2] James F. Kurose / Keith W. Ross.计算机网络. 机械工业出版社,2018  [3] 华钟明. 深入理解RPC框架原理与实现. 电子工业出版社,2021  [4] Norman Maurer,Netty实战. 人民邮电出版社，2017 |
| **指导教师意见**  课题符合专业培养要求，研究内容明确，技术方案合理，同意开题。  **指导教师（签名）：** 2024年 1月 20日 |