

(19) 中华人民共和国国家知识产权局



(12) 发明专利申请

(10) 申请公布号 CN 104320404 A

(43) 申请公布日 2015.01.28

(21) 申请号 201410619737.1

(22) 申请日 2014.11.05

(71) 申请人 中国科学技术大学

地址 230026 安徽省合肥市包河区金寨路
96 号

(72) 发明人 陆世亮 朱明

(74) 专利代理机构 北京凯特来知识产权代理有限公司 11260

代理人 郑立明 郑哲

(51) Int. Cl.

H04L 29/06(2006.01)

H04L 29/08(2006.01)

权利要求书2页 说明书6页 附图6页

(54) 发明名称

一种多线程高性能 http 代理实现方法及系统

(57) 摘要

本发明公开了一种多线程高性能 http 代理实现方法及系统，本发明的方案综合使用了 epoll 事件通知机制和多线程技术，通过合理的建模和设计，极大的提高了 http 代理的并发处理能力，有效地解决了现有 http 代理实现方案无法有效处理特大并发请求这一问题。

当服务器接收到来自客户端的连接请求，在成功连接后，服务器将请求中携带的操作码进行判定当客户端字符串为“CONNECT”时，且该客户端接

当服务器接收到客户端发来的http请求，并将该http请求从待处理队列中解析，若解析失败，则不处理该请求并向上层业务发送错误消息。在成功接收到数据后，半成品一端的服务器将接收到的数据存储在缓冲区中，且将该服务端数据接下及其EPOLLIN事件注册到epoll中

从服务器中定位该客户端连接，并触发服务端的EPOLLIN事件，通知客户端将数据输出给该客户端的http请求，通过该服务端将数据输出给客户端，且将该服务端接下及其EPOLLIN事件注册到epoll中

当服务端接收到客户端的EPOLLIN事件时，通过该服务端从哈希表中定位对应的客户端连接，且将服务端读取消息字符串写入客户端的http请求的响应消息并存储，且将该服务端接下及其EPOLLOUT事件注册到epoll中

通过该服务端从哈希表中定位客户端连接，并触发EPOLLOUT事件，通过该服务端将接下及其EPOLLOUT事件注册到epoll中

1. 一种多线程高性能 http 代理实现方法,其特征在于,该方法包括:

通过指定的地址接收来自客户端的连接请求,在成功连接后,根据该连接请求中携带的描述符创建客户端套接字并存储在哈希表中,且将该客户端套接字及其 EPOLLIN 事件注册到 epoll1 中;

当触发客户端的 EPOLLIN 事件时,通过所述客户端套接字读取来自客户端的 http 请求,并将该 http 请求存储至本地后进行解析,若解析成功,但本地缓存未命中时向服务器发起连接,在成功连接服务器后,生成一对对应的服务器套接字并存储在哈希表中,且将该服务器套接字及其 EPOLLOUT 事件注册到 epoll1 中;

从哈希表中定位该客户端套接字,并触发服务器的 EPOLLOUT 事件,通过该客户端套接字取出存储在本地的 http 请求后通过所述服务器套接字转发给对应的服务器,且将该服务器套接字及其 EPOLLIN 事件注册到 epoll1 中;

当触发服务器的 EPOLLIN 事件时,通过该服务器从哈希表中定位对应的客户端套接字,从服务器中读取该客户端套接字对应客户端 http 请求的响应消息并存储,且将该客户端套接字及其 EPOLLOUT 事件注册到 epoll1 中;

通过该服务器从哈希表中定位该客户端套接字,并触发 EPOLLOUT 事件,通过该客户端套接字取出存储在本地的响应消息后通过所述客户端套接字发送至对应的客户端。

2. 根据权利要求 1 所述的方法,其特征在于,所述通过该客户端套接字取出存储在本地的响应消息后通过所述客户端套接字发送至对应的客户端之后包括:

判断是否与该客户端继续保持连接;

若是,则将该客户端套接字重新注册到 epoll1 中;否则,在哈希表删除该客户端的相关信息并关闭连接。

3. 一种多线程高性能 http 代理实现系统,其特征在于,该系统包括:

接收连接模块,用于通过指定的地址接收来自客户端的连接请求,在成功连接后,根据该连接请求中携带的描述符创建客户端套接字并存储在哈希表中,且将该客户端套接字及其 EPOLLIN 事件注册到 epoll1 中;

读取请求模块,用于当触发客户端的 EPOLLIN 事件时,通过所述客户端套接字读取来自客户端的 http 请求,并将该 http 请求存储至本地后进行解析,若解析成功,但本地缓存未命中时向服务器发起连接,在成功连接服务器后,生成一对对应的服务器套接字并存储在哈希表中,且将该服务器套接字及其 EPOLLOUT 事件注册到 epoll1 中;

转发请求模块,用于从哈希表中定位该客户端套接字,并触发服务器的 EPOLLOUT 事件,通过该客户端套接字取出存储在本地的 http 请求后通过所述服务器套接字转发给对应的服务器,且将该服务器套接字及其 EPOLLIN 事件注册到 epoll1 中;

读取响应模块,用于当触发服务器的 EPOLLIN 事件时,通过该服务器从哈希表中定位对应的客户端套接字,从服务器中读取该客户端套接字对应客户端 http 请求的响应消息并存储,且将该客户端套接字及其 EPOLLOUT 事件注册到 epoll1 中;

响应转发模块,用于通过该服务器从哈希表中定位该客户端套接字,并触发 EPOLLOUT 事件,通过该客户端套接字取出存储在本地的响应消息后通过所述客户端套接字发送至对应的客户端。

4. 根据权利要求 3 所述的系统,其特征在于,所述通过该客户端套接字取出存储在本

地的响应消息后通过所述客户端套接字发送至对应的客户端之后包括：

判断是否与该客户端继续保持连接；

若是，则将该客户端套接字重新注册到 epoll 中；否则，在哈希表删除该客户端的相关信息并关闭连接。

一种多线程高性能 http 代理实现方法及系统

技术领域

[0001] 本发明涉及互联网信息传输领域,尤其涉及一种多线程高性能 http 代理实现方法及系统。

背景技术

[0002] 近十年来,互联网飞速发展,互联网的使用人数也在急剧膨胀,这就对传统 C/S(客户机 / 服务器)模式下的 http(超文本传送协议)代理服务器提出了很大的挑战,代理服务器需要有足够大的并发处理能力才能去满足日常的用户需求,随着互联网使用率的不断增大,这样的并发处理能力要求也在不断的增加,而传统的 http 代理服务器在并发处理能力上则没有跟上步伐。

[0003] 关于如何提升 http 代理服务器的并发服务能力,以满足大并发情景下用户的需求,方法有多种,既可以在现有 http 代理服务器的基础上优化配置,也可以使用最新的技术重新设计 http 代理的架构,传统的 http 代理实现方案包括 apache http 代理项目以及 haproxy 项目等。

[0004] 下面将分别介绍 apache http 代理项目以及 haproxy 项目这两种现有的 http 代理实现方案。

[0005] apache http 代理项目是目前互联网上最流行的 http 代理实现方案。

[0006] 它提供多进程多线程的代理解决方案,用户可以通过配置文件的方式来控制代理的运行模式,运行级别,以及其他的信息。其稳定的性能能够为互联网上绝大多数的网站提供良好的对外服务。

[0007] apache http 代理项目也存在着不足之处:首先配置文件过于庞大,这是因为 apache http 代理项目的目标是通用性,就难免要兼顾到各种情形,从这种意义上讲,它已经不是单纯的 http 代理,再者就是其经过公测的并发处理能力并不能满足特定情境下的用户需求,使得使用该项目的网站在特大并发请求的冲击下容易出现瘫痪。

[0008] haproxy 项目是提供高可用性,负载均衡和基于 TCP 和 HTTP 应用的代理。

[0009] haproxy 项目的实现技术是事件驱动,单一进程,同样具有稳定的并发处理能力,并且易于整合,也易于部署。但是,haproxy 项目也存在这不足之处,仅仅采用单进程,单线程技术,并没有能够充分的利用系统的资源,这一定程度上限制了其并发处理能力的进一步提高,使得在特大并发情境下,采用该项目的网站仍然容易出现瘫痪。

发明内容

[0010] 本发明的目的是提供一种多线程高性能 http 代理实现方法及系统,有效地解决了现有 http 代理实现方案无法有效处理特大并发请求这一问题。

[0011] 本发明的目的是通过以下技术方案实现的:

[0012] 一种多线程高性能 http 代理实现方法,该方法包括:

[0013] 通过指定的地址接收来自客户端的连接请求,在成功连接后,根据该连接请求中

携带的描述符创建客户端套接字并存储在哈希表中,且将该客户端套接字及其 EPOLLIN 事件注册到 epoll 中;

[0014] 当触发客户端的 EPOLLIN 事件时,通过所述客户端套接字读取来自客户端的 http 请求,并将该 http 请求存储至本地后进行解析,若解析成功,但本地缓存未命中时向服务器发起连接,在成功连接服务器后,生成一对对应的服务器套接字并存储在哈希表中,且将该服务器套接字及其 EPOLLOUT 事件注册到 epoll 中;

[0015] 从哈希表中定位该客户端套接字,并触发服务器的 EPOLLOUT 事件,通过该客户端套接字取出存储在本地的 http 请求后通过所述服务器套接字转发给对应的服务器,且将该服务器套接字及其 EPOLLIN 事件注册到 epoll 中;

[0016] 当触发服务器的 EPOLLIN 事件时,通过该服务器从哈希表中定位对应的客户端套接字,从服务器中读取该客户端套接字对应客户端 http 请求的响应消息并存储,且将该客户端套接字及其 EPOLLOUT 事件注册到 epoll 中;

[0017] 通过该服务器从哈希表中定位该客户端套接字,并触发 EPOLLOUT 事件,通过该客户端套接字取出存储在本地的响应消息后通过所述客户端套接字发送至对应的客户端。

[0018] 进一步的,所述通过该客户端套接字取出存储在本地的响应消息后通过所述客户端套接字发送至对应的客户端之后包括:

[0019] 判断是否与该客户端继续保持连接;

[0020] 若是,则将该客户端套接字重新注册到 epoll 中;否则,在哈希表删除该客户端的相关信息并关闭连接。

[0021] 一种多线程高性能 http 代理实现系统,该系统包括:

[0022] 接收连接模块,用于通过指定的地址接收来自客户端的连接请求,在成功连接后,根据该连接请求中携带的描述符创建客户端套接字并存储在哈希表中,且将该客户端套接字及其 EPOLLIN 事件注册到 epoll 中;

[0023] 读取请求模块,用于当触发客户端的 EPOLLIN 事件时,通过所述客户端套接字读取来自客户端的 http 请求,并将该 http 请求存储至本地后进行解析,若解析成功,但本地缓存未命中时向服务器发起连接,在成功连接服务器后,生成一对对应的服务器套接字并存储在哈希表中,且将该服务器套接字及其 EPOLLOUT 事件注册到 epoll 中;

[0024] 转发请求模块,用于从哈希表中定位该客户端套接字,并触发服务器的 EPOLLOUT 事件,通过该客户端套接字取出存储在本地的 http 请求后通过所述服务器套接字转发给对应的服务器,且将该服务器套接字及其 EPOLLIN 事件注册到 epoll 中;

[0025] 读取响应模块,用于当触发服务器的 EPOLLIN 事件时,通过该服务器从哈希表中定位对应的客户端套接字,从服务器中读取该客户端套接字对应客户端 http 请求的响应消息并存储,且将该客户端套接字及其 EPOLLOUT 事件注册到 epoll 中;

[0026] 响应转发模块,用于通过该服务器从哈希表中定位该客户端套接字,并触发 EPOLLOUT 事件,通过该客户端套接字取出存储在本地的响应消息后通过所述客户端套接字发送至对应的客户端。

[0027] 进一步的,所述通过该客户端套接字取出存储在本地的响应消息后通过所述客户端套接字发送至对应的客户端之后包括:

[0028] 判断是否与该客户端继续保持连接;

[0029] 若是，则将该客户端套接字重新注册到 epoll 中；否则，在哈希表删除该客户端的相关信息并关闭连接。

[0030] 由上述本发明提供的技术方案可以看出，本发明所提供的方案综合使用了 epoll 事件通知机制和多线程技术，通过合理的建模和设计，极大的提高了 http 代理的并发处理能力，有效地解决了现有 http 代理实现方案无法有效处理特大并发请求这一问题。

附图说明

[0031] 为了更清楚地说明本发明实施例的技术方案，下面将对实施例描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图仅仅是本发明的一些实施例，对于本领域的普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其他附图。

[0032] 图 1 为本发明实施例一提供的一种多线程高性能 http 代理实现方法的流程图；

[0033] 图 2 为本发明实施例一提供的接收连接模块流程图；

[0034] 图 3 为本发明实施例一提供的读取请求模块流程图；

[0035] 图 4 为本发明实施例一提供的转发请求模块流程图；

[0036] 图 5 为本发明实施例一提供的读取响应模块流程图；

[0037] 图 6 为本发明实施例一提供的转发响应模块流程图；

[0038] 图 7 为本发明实施例二提供的一种多线程高性能 http 代理实现系统的示意图。

具体实施方式

[0039] 下面结合本发明实施例中的附图，对本发明实施例中的技术方案进行清楚、完整地描述，显然，所描述的实施例仅仅是本发明一部分实施例，而不是全部的实施例。基于本发明的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都属于本发明的保护范围。

[0040] 实施例一

[0041] 图 1 为本发明实施例一提供的一种多线程高性能 http 代理实现方法的流程图。如图 1 所示，该方法主要包括如下步骤：

[0042] 步骤 11、通过指定的地址接收来自客户端的连接请求，在成功连接后，根据该连接请求中携带的描述符创建客户端套接字并存储在哈希表中，且将该客户端套接字及其 EPOLLIN 事件注册到 epoll 中。

[0043] 步骤 12、当触发客户端的 EPOLLIN 事件时，通过所述客户端套接字读取来自客户端的 http 请求，并将该 http 请求存储至本地后进行解析，若解析成功，但本地缓存未命中时向服务器发起连接，在成功连接服务器后，生成一对对应的服务器套接字并存储在哈希表中，且将该服务器套接字及其 EPOLLOUT 事件注册到 epoll 中。

[0044] 步骤 13、从哈希表中定位该客户端套接字，并触发服务器的 EPOLLOUT 事件，通过该客户端套接字取出存储在本地的 http 请求后通过所述服务器套接字转发给对应的服务器，且将该服务器套接字及其 EPOLLIN 事件注册到 epoll 中。

[0045] 步骤 14、当触发服务器的 EPOLLIN 事件时，通过该服务器从哈希表中定位对应的客户端套接字，从服务器中读取该客户端套接字对应客户端 http 请求的响应消息并存储，

且将该客户端套接字及其 EPOLLOUT 事件注册到 epoll1 中。

[0046] 步骤 15、通过该服务器从哈希表中定位该客户端套接字，并触发 EPOLLOUT 事件，通过该客户端套接字取出存储在本地的响应消息后通过所述客户端套接字发送至对应的客户端。

[0047] 其中，所述通过该服务器套接字取出响应消息后通过所述客户端套接字发送至对应的客户端之后包括：判断是否与该客户端继续保持连接；若是，则将该客户端套接字重新注册到 epoll1 中；否则，在哈希表删除该客户端的相关信息并关闭连接。

[0048] 本发明实施例中，套接字及其事件的管理使用了如下的 epoll1 机制：(1) 初始化阶段完成 epoll1 机制的准备工作，完成 epoll1 的初始化；(2) 在注册套接字及其事件的时候声明使用 EPOLLONESHOT 属性；(3) 在修改套接字事件的时候使用 MOD 操作。

[0049] 同时，本发明实施例中大并发连接的处理使用了如下的多线程机制：上述 5 个步骤都有对应的线程进行实施；且使用互斥锁和条件变量来进行线程同步。

[0050] 另外，本发明实施例中对于线程的管理使用了如下的线程池方法：(1) 全部线程的整体状态通过线程池中的变量群来标识，并且变量群的维护在每一个线程中自主完成；(2) 线程池中的线程数量由专门的动态管理算法来决定。(3) 动态管理算法的运行有专门的动态管理线程维护。

[0051] 为了便于理解，下面结合附图 2-6 对上述 5 个步骤做详细的说明。

[0052] 步骤 11 为接收连接的过程，其工作流程如图 2 所示。首先，创建需要的对象，进行相应的初始化，然后在一个无限循环中，不断的接收来自客户端的连接请求，如果失败，则记录在日志文件中，对于接受成功的连接，将相关的信息（套接字及其描述符）存储在哈希表中，并且将该套接字及 EPOLLIN 事件注册到 epoll1 当中。

[0053] 当成功接收连接后操作系统会为这条连接分配一个新的描述符来创建一个新的套接字，用于双方的通信，且每个客户端的套接字都不一样，服务器则是通过套接字和客户端通信的，客户端和套接字是一一对应的。

[0054] 本发明实施例中引入了 epoll1 事件，epoll1 事件包括 EPOLLIN 事件与 EPOLLOUT 事件。其中，EPOLLIN 事件指的是当中套接字有数据进来，epoll1 机制就会触发 EPOLLIN 事件，然后由服务器读取数据；EPOLLOUT 事件则是指若该套接字有数据要发送，则 epoll1 机制触发 EPOLLOUT 事件，然后由服务器通过套接字把数据发送出去。

[0055] 本步骤中将 EPOLLIN 事件注册到 epoll1 当中是为了采用 epoll1 的事件通知机制，即用户无需一直等待客户端发送请求，而是可以去处理其他事务，比如说处理其它客户端套接字中的数据；当客户端通过该套接字发送请求时 epoll1 机制则会通知用户。

[0056] 步骤 12 为读取 http 请求的过程，其工作流程如图 3 所示。首先，通过客户端套接字读取客户端的 http 请求（即此处已经触发步骤 11 中注册的 EPOLLIN 事件），对于暂时不可读的套接字，则再次注册到 epoll1 当中，稍后再读；对于可以读取的，则将读取到的 http 请求存储到本地，并且解析该请求，解析失败则返回错误给客户端，解析成功则在本地缓存未命中的情况下向服务器发起连接，并且将成功的服务器套接字与 EPOLLOUT 事件注册到 epoll1 当中。

[0057] 本发明实施例所述的本地缓存未命中表示本地没有对应于该 http 的响应。具体来说，任何一个 http 代理都会有缓存的功能，也就是说本地的 http 代理会缓存一部分后台

服务器发来的响应,当下次有新的 http 请求到来的时候,首先在本地查找,看看有没有对应的响应,如果说有的话就直接把响应发给客户端了,这样可以分担一部分后台服务器的压力;但是如果本地的 http 代理没有对应的响应,则应该转发给服务器,此时有一个专有名词,称为缓存未命中。

[0058] 步骤 13 为转发请求的过程,其工作流程如图 4 所示。首先,从哈希表中定位客户端套接字(即此处已经触发步骤 12 中注册的 EPOLLOUT 事件),如果定位失败,则进行错误处理;如果定位成功,则从本地取出与该客户端套接字相对应的 http 请求通过所述服务器套接字转发给对应的服务器,并且将服务器套接字及 EPOLLIN 事件注册到 epoll1 当中。

[0059] 步骤 14 为读取响应的过程,其工作流程如图 5 所示。首先,通过服务器定位到哈希表中的对应客户端套接字(即此处已经触发步骤 13 中注册的 EPOLLIN 事件),如果定位失败,进行错误处理;如果定位成功,则从服务器中读取该客户端套接字对应的 http 请求的响应消息并存储在本地,且将该客户端套接字及 EPOLLOUT 事件注册到 epoll1 中。

[0060] 步骤 15 为转发响应的过程,其工作流程如图 6 所示。首先,通过该服务器从哈希表中再次定位该客户端套接字,(即此处已经触发步骤 14 中注册的 EPOLLOUT 事件),定位失败,则进行错误处理;如果定位成功,则通过该客户端套接字取出存储在本地的响应消息后通过所述客户端套接字发送至对应的客户端,再判断是否保持连接,如果保持,则重新注册到 epoll1 当中,如果不保持,则删除哈希节点,并且关闭连接。

[0061] 本发明实施例的方案综合使用了 epoll 事件通知机制和多线程技术,通过合理的建模和设计,极大的提高了 http 代理的并发处理能力,有效地解决了现有 http 代理实现方案无法有效处理特大并发请求这一问题。

[0062] 实施例二

[0063] 本发明实施例二提供一种多线程高性能 http 代理实现系统,该系统可以部署在连接于客户端与服务器池之间的代理服务器中。如图 7 所示,该系统主要包括:

[0064] 接收连接模块 71,用于通过指定的地址接收来自客户端的连接请求,在成功连接后,根据该连接请求中携带的描述符创建客户端套接字并存储在哈希表中,且将该客户端套接字及其 EPOLLIN 事件注册到 epoll1 中;

[0065] 读取请求模块 72,用于当触发客户端的 EPOLLIN 事件时,通过所述客户端套接字读取来自客户端的 http 请求,并将该 http 请求存储至本地后进行解析,若解析成功,但本地缓存未命中时向服务器发起连接,在成功连接服务器后,生成一对应的服务器套接字并存储在哈希表中,且将该服务器套接字及其 EPOLLOUT 事件注册到 epoll1 中;

[0066] 转发请求模块 73,用于从哈希表中定位该客户端套接字,并触发服务器的 EPOLLOUT 事件,通过该客户端套接字取出存储在本地的 http 请求后通过所述服务器套接字转发给对应的服务器,且将该服务器套接字及其 EPOLLIN 事件注册到 epoll1 中;

[0067] 读取响应模块 74,用于当触发服务器的 EPOLLIN 事件时,通过该服务器从哈希表中定位对应的客户端套接字,从服务器中读取该客户端套接字对应客户端 http 请求的响应消息并存储,且将该客户端套接字及其 EPOLLOUT 事件注册到 epoll1 中;

[0068] 响应转发模块 75,用于通过该服务器从哈希表中定位该客户端套接字,并触发 EPOLLOUT 事件,通过该客户端套接字取出存储在本地的响应消息后通过所述客户端套接字发送至对应的客户端。

[0069] 其中,所述通过该客户端套接字取出存储在本地的响应消息后通过所述客户端套接字发送至对应的客户端之后包括:判断是否与该客户端继续保持连接;若是,则将该客户端套接字重新注册到 epoll 中;否则,在哈希表删除该客户端的相关信息并关闭连接。

[0070] 本发明实施例所提供的系统可以部署在连接于客户端与服务器池之间的代理服务器中。

[0071] 需要说明的是,上述系统中包含的各个功能模块所实现的功能的具体实现方式在前面的各个实施例中已经有详细描述,故在这里不再赘述。

[0072] 所属领域的技术人员可以清楚地了解到,为描述的方便和简洁,仅以上述各功能模块的划分进行举例说明,实际应用中,可以根据需要而将上述功能分配由不同的功能模块完成,即将系统的内部结构划分成不同的功能模块,以完成以上描述的全部或者部分功能。

[0073] 通过以上的实施方式的描述,本领域的技术人员可以清楚地了解到上述实施例可以通过软件实现,也可以借助软件加必要的通用硬件平台的方式来实现。基于这样的理解,上述实施例的技术方案可以以软件产品的形式体现出来,该软件产品可以存储在一个非易失性存储介质(可以是 CD-ROM, U 盘, 移动硬盘等)中,包括若干指令用以使得一台计算机设备(可以是个人计算机, 服务器, 或者网络设备等)执行本发明各个实施例所述的方法。

[0074] 以上所述,仅为本发明较佳的具体实施方式,但本发明的保护范围并不局限于此,任何熟悉本技术领域的技术人员在本发明披露的技术范围内,可轻易想到的变化或替换,都应涵盖在本发明的保护范围之内。因此,本发明的保护范围应该以权利要求书的保护范围为准。

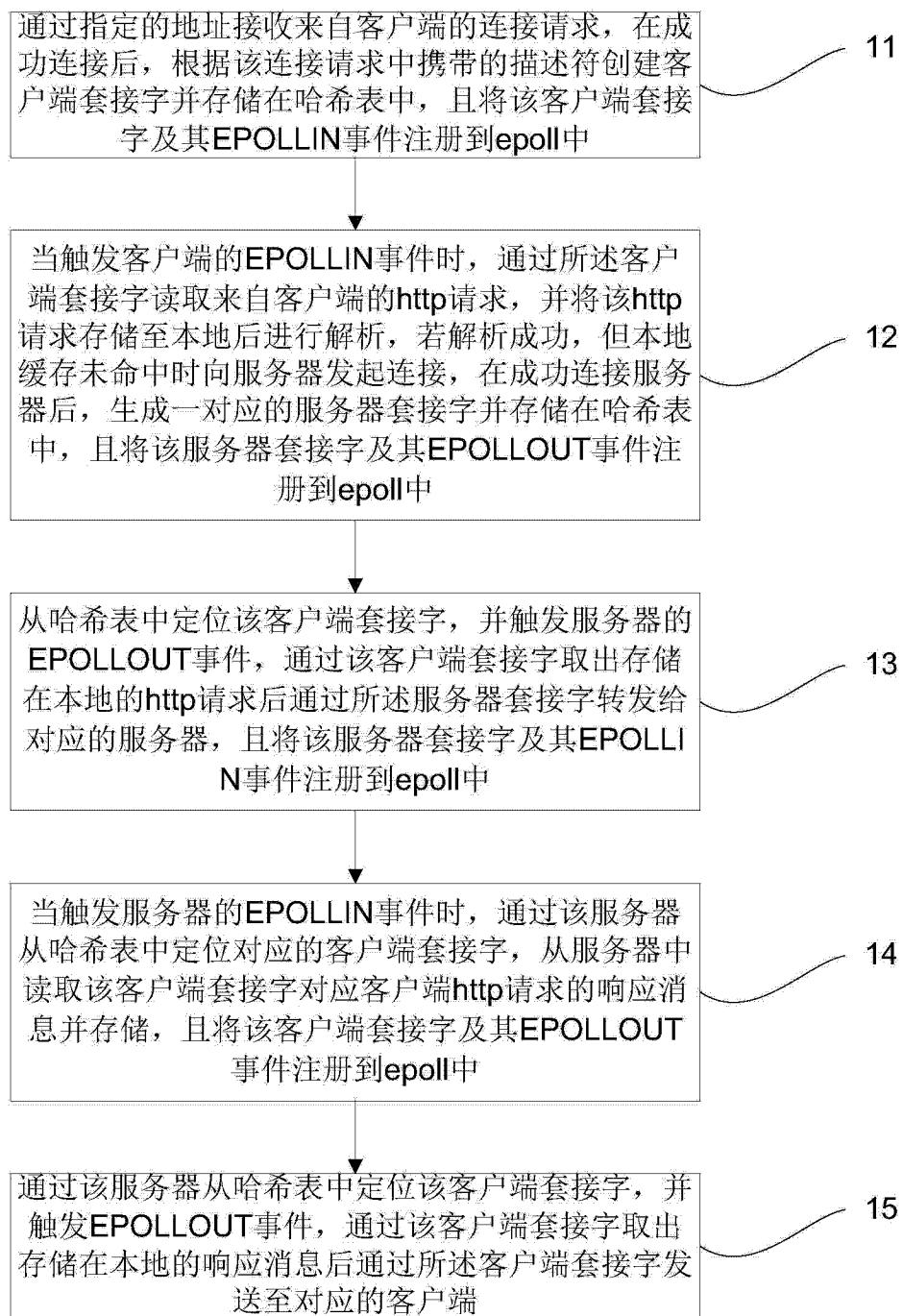


图 1

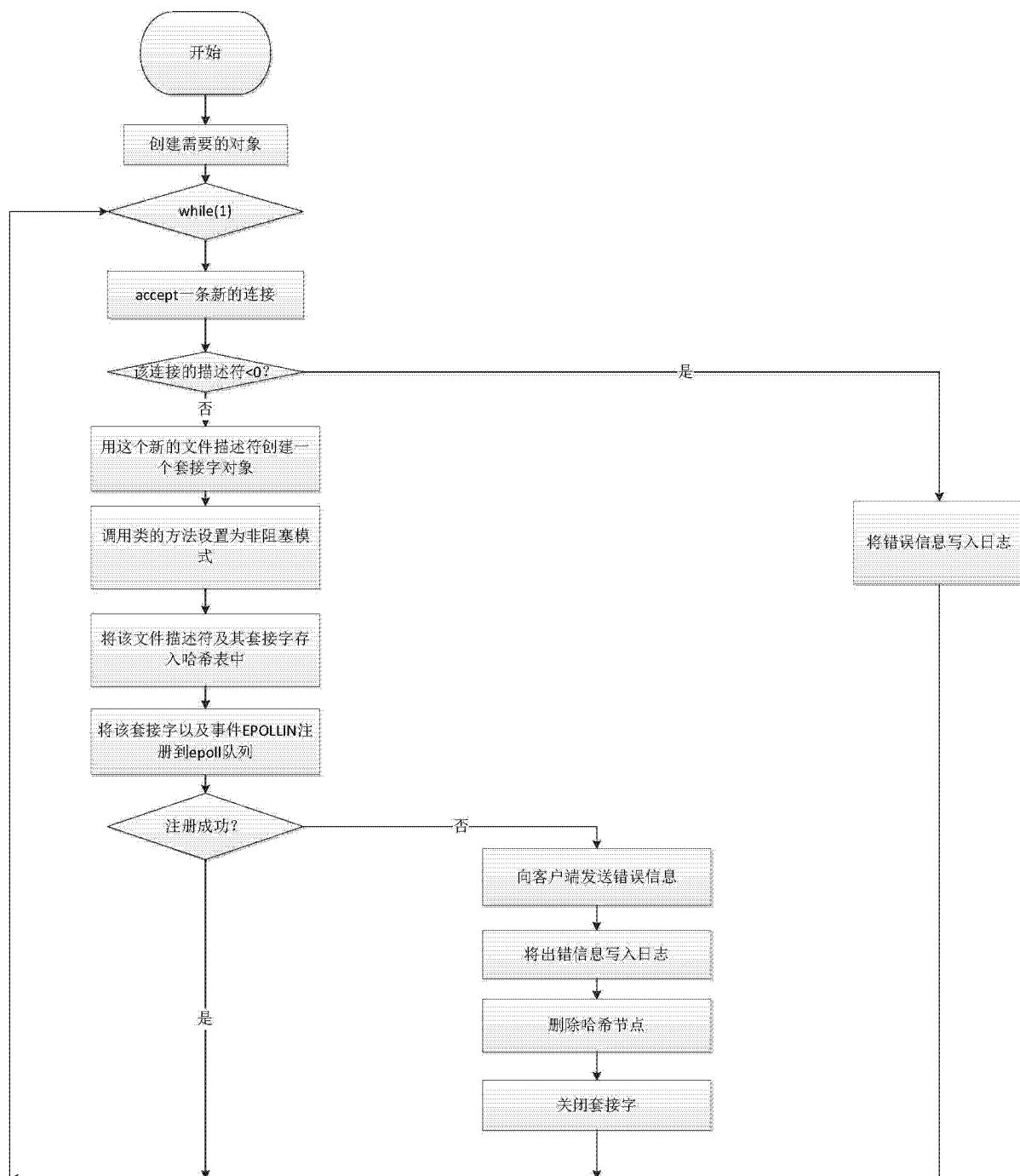


图 2

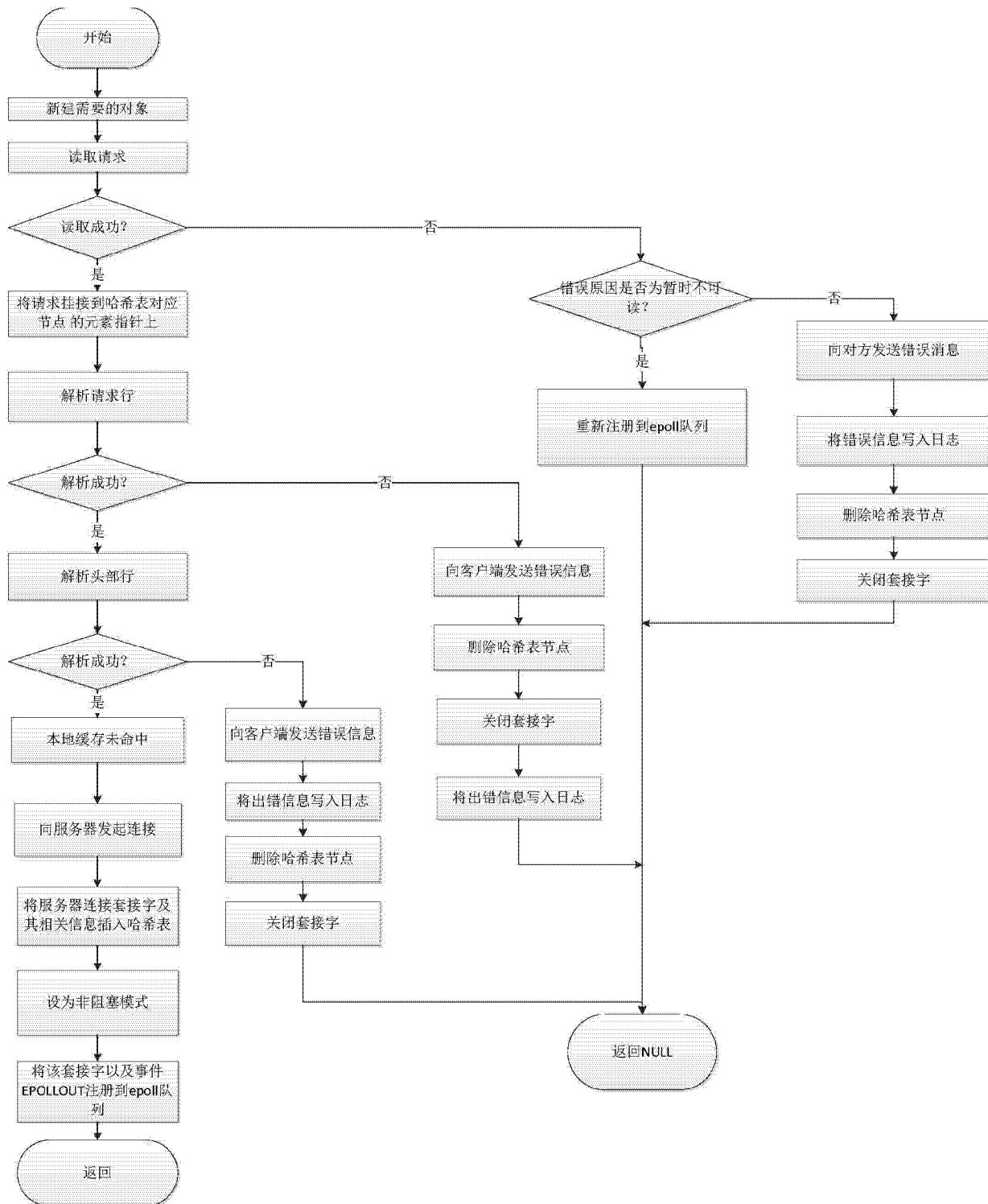


图 3

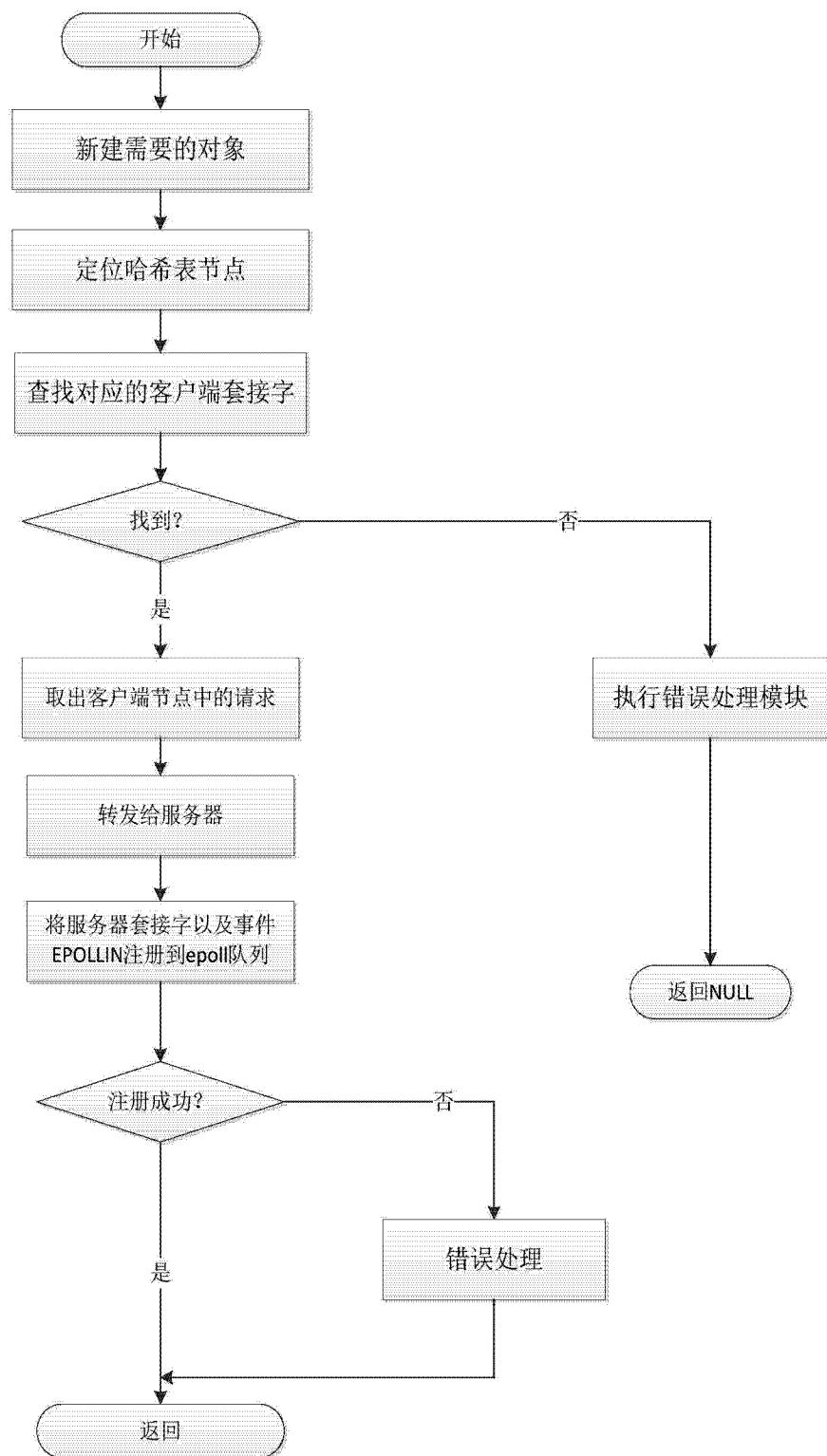


图 4

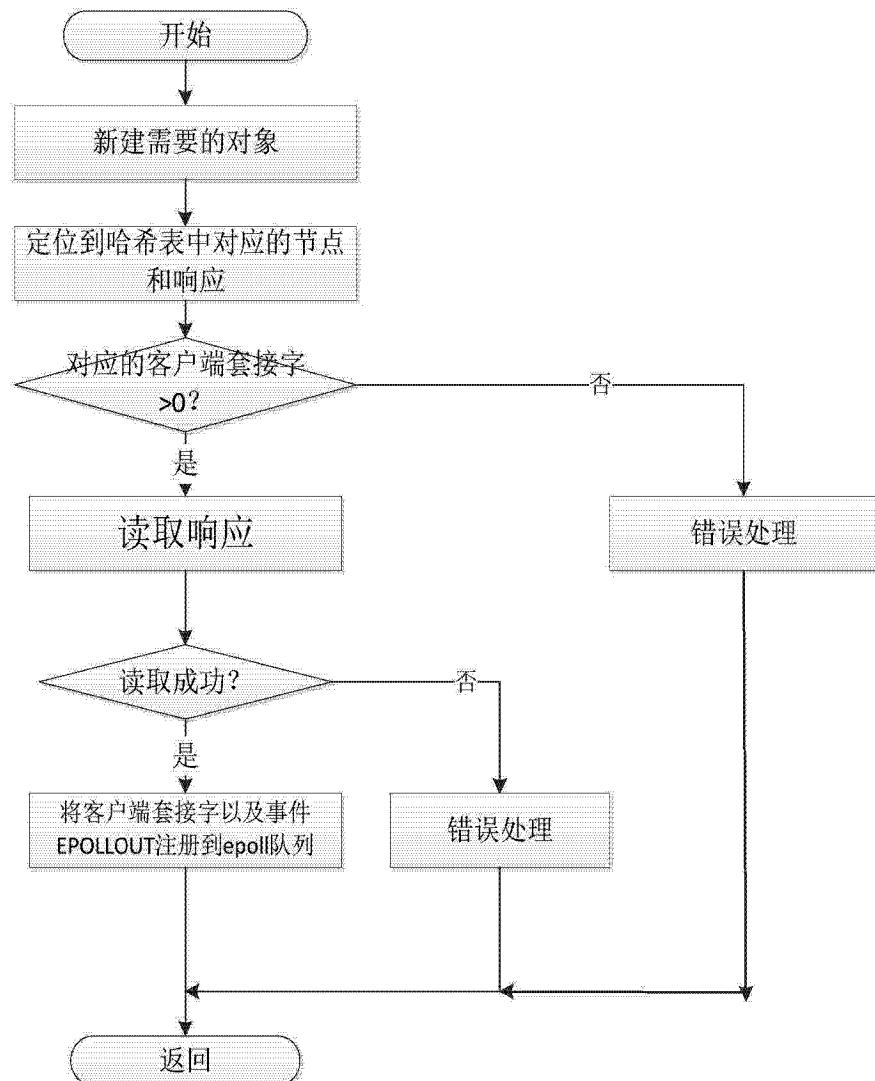


图 5

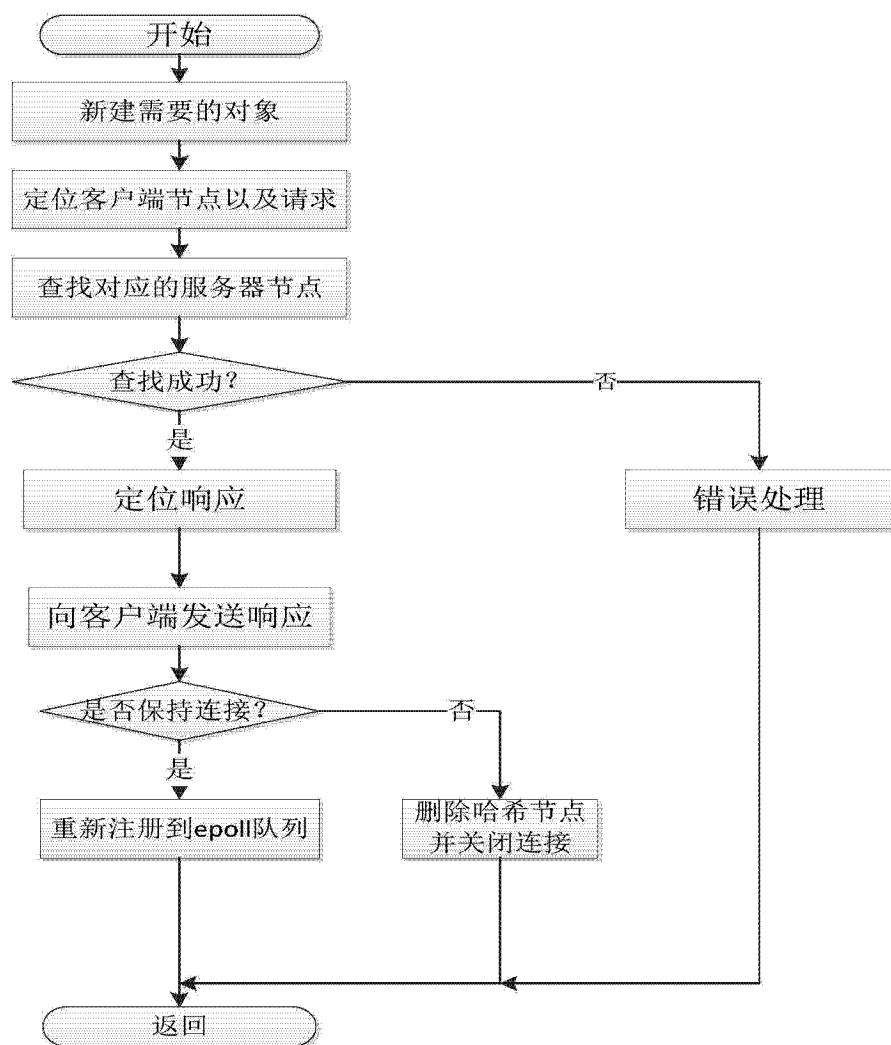


图 6

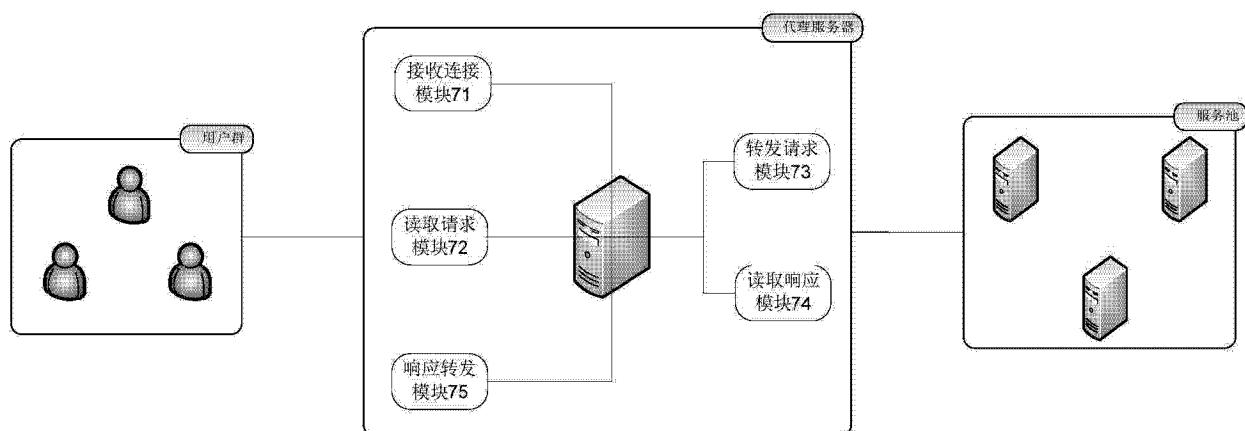


图 7