

[19] 中华人民共和国国家知识产权局

[51] Int. Cl⁷

H04L 29/06

G06F 9/46



[12] 发明专利说明书

[21] ZL 专利号 00817108.4

[45] 授权公告日 2005 年 1 月 19 日

[11] 授权公告号 CN 1185841C

[22] 申请日 2000.11.18 [21] 申请号 00817108.4

[30] 优先权

[32] 1999.12.13 [33] EP [31] 99124727.1

[86] 国际申请 PCT/EP2000/011503 2000.11.18

[87] 国际公布 WO2001/045353 英 2001.6.21

[85] 进入国家阶段日期 2002.6.13

[71] 专利权人 艾利森电话股份有限公司

地址 瑞典斯德哥尔摩

[72] 发明人 S·鲍克 M·兹穆达

审查员 程 东

[74] 专利代理机构 中国专利代理(香港)有限公司

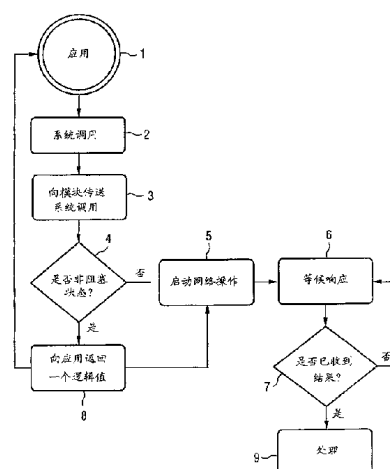
代理人 程天正 梁 永

权利要求书 2 页 说明书 11 页 附图 4 页

[54] 发明名称 执行网络操作的方法和设备

[57] 摘要

本发明涉及一种方法和设备，能够通过一个处理应用层系统调用并启动网络层网络操作的模块、经过高时延网络来快速执行网络操作。在所述模块中区分阻塞和非阻塞的实现模式。非阻塞实现模式意味着所考虑的系统调用向应用返回一个逻辑值作为结果，它用信号通知是否成功执行该系统调用。在此情况下，本发明提供在调用非阻塞的系统调用时向应用直接发送逻辑值，而不必等候在通信伙伴实例中执行并且对应于该系统调用的操作的实际结果。稍后进行对实际执行的操作结果的处理。



1. 通过一个用于处理应用层的系统调用并启动网络层的网络操作的模块来经过具有高时延的网络而快速执行网络操作的方法，包括下面的步骤：

- 5 - 将系统调用传送到此模块，
 - 通过区分阻塞和非阻塞执行模式来确定系统调用的执行模式，
 - 在非阻塞执行模式的情况下，直接向应用层返回一个逻辑值，并启动一个网络操作。

2. 根据权利要求 1 的方法，其中网络操作被传送到一个与启动网络操作的单元进行通信的伙伴实例。

3. 根据权利要求 2 的方法，其中在伙伴实例中所接收的网络操作被转换成要执行的操作，而且其中该操作的结果被返回到启动网络操作的单元。

4. 根据权利要求 1-3 中任意一个的方法，其中在此模块中实现对
15 所接收操作结果的处理。

5. 根据权利要求 1-3 中任意一个的方法，其中在启动网络操作之后，一个非阻塞的系统调用就被转换成一种状态：其中等候在伙伴实例中所执行系统调用的实际结果而不阻塞调用应用的执行。

6. 根据权利要求 3 的方法，其中所接收的结果涉及非阻塞状态并且具有逻辑值，或者它们是在伙伴实例中所执行的阻塞系统调用的结果。

7. 根据权利要求 3 的方法，其中对具有非阻塞执行模式的所接收结果进行缓冲。

8. 根据权利要求 1-3 中任意一个的方法，其中此逻辑值具有逻辑
25 肯定或者逻辑否定的建议值。

9. 根据权利要求 8 的方法，其中通过后续系统调用的执行以逻辑否定返回值的形式向应用报告逻辑否定结果。

10. 根据权利要求 3 的方法，其中对于非阻塞系统调用，在先前呼叫的未决否定结果的情况下，总是向应用返回一个逻辑肯定值。

11. 根据权利要求 1-3 的方法，其中为了保证返回先前所执行操作的结果报告，连接的最后的系统调用被设置为阻塞状态。

12. 根据权利要求 1 的方法，其中阻塞系统调用是通过等候在伙伴

实例中所执行系统调用的结果而以一种熟知的方式实现的。

13. 根据权利要求 1-3 中任意一个的方法，其中系统调用是套接字系统调用。

5 14. 根据权利要求 13 的方法，其中套接字系统调用形成了一个对于操作系统的编程接口。

15. 根据权利要求 1 的方法，其中此模块是一个流水线模块。

16. 通过一个用于处理应用层的系统调用并启动网络层的网络操作的模块来经过具有高时延的网络而快速执行网络操作的设备，具有

10 - 一个用于通过区分阻塞和非阻塞执行模式而确定系统调用执行模式的确定单元，

- 用于把系统调用转换成网络操作的转换装置，

- 一个用于发送网络操作的发送器，

- 一个用于接收执行的网络操作结果的接收器，

- 一个用于存储所接收结果的存储器，

15 - 一个用于处理所接收结果的处理单元。

17. 根据权利要求 16 的设备，具有用于实现一种状态的单元，其中一旦启动网络操作就等候在伙伴实例中所执行系统调用的结果，而不阻塞调用应用的执行。

20 18. 根据权利要求 16 或 17 的设备，其中用于处理所接收结果的装置区分肯定和否定值。

执行网络操作的方法和设备

技术领域

- 5 本发明涉及一种经过特征为高时延的网络来快速执行网络操作的方法和设备。

背景技术

- 高时延网络的一个例子是 GSM (全球移动通信系统) 网络。GSM 网络是每个信道可用数据速率为 9.6kBit/s 的数字移动通信网。与目前
10 诸如数据速率高达 64kBit/s 的 ISDN (综合业务数字网) 等固定网络相比, GSM 的数据速率较小。因此, GSM 连接严重影响在异类网络连接中 (即在延续至少两种不同网络的连接中) 整个连接的绝对吞吐量。在 GSM 用户为了进行所谓的网络冲浪或/和从互联网上下载数据、而已经建立了一个到互联网服务提供商 (ISP) 的连接时, 这尤其会引发问题。
15 在所述场景中, GSM 连接构成关于有效传输速率的关键部分连接。

- 在通信伙伴实例之间, 在此情况下是在 GSM 终端和互联网服务提供商 (ISP) 之间, 交换信息是基于一个为互联网应用而标准化的协议栈, 也就是所谓的 TCP/IP (传输控制协议/互联网协议) 协议栈。所述协议栈包括五个协议层, 此后也称之为层。最高层为应用层, 包括用户
20 所直接使用的应用, 例如互联网上冲浪就是一种用户可能使用的应用。与应用层紧连的下层为传输层, 包括诸如 TCP (传输控制协议) 或 UDP (用户数据报协议) 等实现所述层任务的传输协议。网络层由例如互联网协议 (IP) 来组成。最下两层 (链路层和物理层) 可以概括为术语面向网络的层, 因为它们是根据底层的网络特别定义的。

- 25 一个层的任务尤其是要处理所接收的数据或要传输的数据。在这方面, 一个层的通信只与其直接相邻的层发生。提供用于传输的数据从一个层传送到紧连的下层, 由此对数据进行格式化。应用层的数据经常超出了可经物理连接传输的分组大小。为此把一个消息划分成一些在一次传输中连续排列的较小数据分组。数据的划分是格式化的一部分。在每
30 个协议层都进行数据的格式化。格式化数据特别地包括添加每个协议层所特有的控制数据。在多数情况下, 控制数据以所谓头标的形式加在数据分组的开始、或/和以所谓尾标的形式加在分组的结尾。实际的数据

包含在数据字段中。所述控制数据对于每层进行特别地定义和标准化，因为只有用这种方式才能保证双方实例的统一标准通信。关于本发明，TCP 的具体控制数据非常重要。

传输协议 TCP 为两个通信伙伴之间的字节流提供可靠的传输服务。在此方面可靠性是指避免错误、维持顺序并保护防止数据丢失和重复。要传输的字节流从应用层向 TCP 层传送，它被 TCP 细分为分段，而这些分段会作为 IP 数据报来传输。IP 数据报指定一个根据 IP 协议规则进行格式化的数据分组。数据报的性质在于使用数据报来实现的数据交换并不可靠。于是，IP 不能保证一个分组确实被传输到接收方。
而且，IP 数据报可能以错误的顺序或者重复的到达接收侧。但是在所述概念中，检测有错误的传输并去除已出现的错误是 TCP 的任务。由于分层原则，IP 数据报又被传送到其紧连的下层。

为了实现这些任务已向 TCP 中集成了多种复杂的机制。这里不会详细描述各种机制，如窗口机制、累积确认方案，所谓的慢启动算法或用于避免拥塞的算法，所谓的拥塞避免算法。所述方法的精确描述可以参考 W. Richard Stevens 的《TCP/IP Illustrated, Volume 1 (TCP/IP 说明，第一卷)》。

一种会在这里进行更为仔细处理的重要机制涉及用 TCP 建立连接。建立连接时，TCP 使用所谓的三次握手来确保交换固定定义的信令信息。在图 2 中可见这些内容。通信双方被称为客户机和服务器，其中客户机是通信的启动单元，而服务器是通信的接受单元。总体而言，客户机和服务器之间的连接建立需要三次消息交换。通过第一消息的发送通告了通信伙伴之间的初始序号。所交换的数据分组由准确的号码来标识，即所谓的序号。为此 TCP 数据分组头标提供了一个用于连接建立时同步序号的字段即所谓的 SYN 标志，在图 2 中表示为 SYN。如果服务器准备好接受连接，则会应答一个消息，其中包含设置的 SYN 标志以及对客户机初始序号的确认 ACK(SYN)。客户机最后发送的消息含有对于服务器 SYN 消息的确认 ACK(SYN)。三次消息已交换时，连接建立阶段完成。在下一步中传输数据，这是通过连接建立而完成的实际通信。连接启动要求交换四个消息。首先，从客户机向服务器发送一个包括设置的 FIN 标志(FIN)的消息，此标志实现了发送方向上连接的断开。由此客户机启动了连接。服务器用 ACK(FIN)确认此消

息的接收,并同样启动连接断开阶段。为此服务器向客户机发送带有设置的 FIN 标志 (FIN) 的消息,而客户机用 ACK (FIN) 应答。一旦交换此消息就完成了连接启动。

5 总体而言,连接建立和启动需要最少七个消息或三个半所谓的 RTT (往返行程时间)。RTT 描述了一个在发送请求消息和接收相关应答消息之间的时间容限。所述时间相应的随底层网络的传输速率而有所改变。

10 但是许多应用基于通信伙伴实例之间的简单交互。简单交互的一个例子是所谓的事务处理,由此客户机可以向服务器发送一个请求消息,而接收方在响应消息中向发送方发送所请求的信息。这种数据交换可能出现在例如互联网中,并且其特征为万维网协议的面向事务处理的流量模式。互联网中的事务处理主要由于所用的 HTTP 协议而引起。

15 关于当前所使用的协议版本 HTTP/1.0 (Berners-Lee, R. Fielding 和 H. Frystyk 1996 年 5 月的 RFC 1945 “Hypertext/Transfer Protocol-HTTP/1.0(超文本传输协议—HTTP/1.0”)), 首先为 WWW 网页中每个所请求的对象建立 TCP 连接,例如文本块或者图像。只有此时才能执行一个事务处理,即发送方向接收方发送一个请求消息,然后接收方在应答消息中发送所请求的数据。此后再断开 TCP 连接。如果必须为每个对象建立明确的连接,而此对象仅是一个小图像对象,且此连接基
20 于三次握手并再次要求至少三个建立消息和四个启动消息,则所述情况会严重影响会话的总体性能。

25 TCP 必须完成的任务已经表明, TCP 被设计为适用于特征为快速数据传输和少量传输失败的网络环境,这是对于有线连接的数据网络而发生的。但是这些假设没能在 GSM 中实现,这是造成在这种网络中 TCP 的非最优功能性的原因。许多文档表示了为小传输容量的网络解决 TCP 优化的概念。此后会介绍一个解决方案的例子。

30 R. Braden 的 RFC 1644 “T/TCP — TCP Extensions for Transactions Functional Specification(T/TCP—事务处理功能规范的 TCP 扩展)”介绍了一种特别优化连接建立阶段的概念。为此缓冲有关到具体服务器的连接信息。于是,只通过与服务器的第一次连接建立而执行完整的三次握手,并缓冲在所述连接建立期间所通告的有关连接配置的信息。在对同一服务器的后续访问中通过从所存储的信息取回配

置，从而回避了完整的连接建立阶段。

但是这种解决方案的不利之处在于对于应用不存在透明性。实现此概念需要修改应用中的系统调用。实现更快的连接建立可以通过例如用系统调用“sendto（发送到）”取代“connect（连接）”和“send（发
5 送）”来获得。所述系统调用是所谓的套接字编程接口的一部分，其更为精确的功能性可以参考下面的描述。

发明内容

因此，本发明的一个目的是提供一种经过特征为高时延的传输链路来实现两个通信单元之间事务处理的有效执行的方法和设备。

10 根据本发明提供了通过一个用于处理应用层的系统调用并启动网络层的网络操作的模块来经过具有高时延的网络而快速执行网络操作的方法，包括下面的步骤：将系统调用传送到此模块，通过区分阻塞和非阻塞执行模式来确定系统调用的执行模式，在非阻塞执行模式的情况下，直接向应用层返回一个逻辑值，并启动一个网络操作。

15 根据本发明还提供了通过一个用于处理应用层的系统调用并启动网络层的网络操作的模块来经过具有高时延的网络而快速执行网络操作的设备，具有：一个用于通过区分阻塞和非阻塞执行模式而确定系统调用执行模式的确定单元，用于把系统调用转换成网络操作的转换装置，一个用于发送网络操作的发送器，一个用于接收执行的网络操作结
20 果的接收器，一个用于存储所接收结果的存储器，一个用于处理所接收结果的处理单元。

由此看出有利之处在于，优化事务处理的执行可以直接影响应用的效率，因为事务处理的快速执行保证了对时延敏感的应用更为稳定的工作方式。

25 另一有利之处在于本发明的实现独立于应用层以及传输层，因为它保证了本发明的使用对两层都保持透明。

本发明其他有利的实施方案包括：在上面提及的快速执行网络操作的方法中，网络操作被传送到一个与启动网络操作的单元进行通信的伙伴实例。在伙伴实例中所接收的网络操作被转换成要执行的操作，而且
30 其中该操作的结果被返回到启动网络操作的单元。在此模块中实现对所接收操作结果的处理。在启动网络操作之后，一个非阻塞的系统调用就被转换成一种状态：其中等候在伙伴实例中所执行系统调用的实际结果

而不阻塞调用应用的执行。所接收的结果涉及非阻塞状态并且具有逻辑值，或者它们是在伙伴实例中所执行的阻塞系统调用的结果。存储具有非阻塞执行模式的所接收结果。此逻辑值具有逻辑肯定或者逻辑否定的建议值。通过后续系统调用的执行以逻辑否定返回值的形式向应用报告逻辑否定结果。对于非阻塞系统调用，在先前呼叫的未决否定结果的情况下，总是向应用返回一个逻辑肯定值。为了保证返回先前所执行操作的结果报告，连接的最后的系统调用被设置为阻塞状态。阻塞系统调用是通过等候在伙伴实例中所执行系统调用的结果而以一种熟知的方式实现的。系统调用是套接字系统调用。套接字系统调用形成了一个对于操作系统的编程接口。此模块是一个流水线模块。

而在上面提及的快速执行网络操作的设备中还具有：用于实现一种状态的单元，其中一旦启动网络操作就等候在伙伴实例中所执行系统调用的结果，而不阻塞调用应用的执行。用于处理所接收结果的装置区分肯定和否定值。

附图说明

下面通过实施方案和示图更为详细的对本发明进行解释，其中图 1 表示了根据本发明方法的流程图，图 2 表示了三次握手的示意图，图 3 表示了根据现有技术（3a）和根据本发明（3b）建立连接的图解，以及

图 4 表示了一个实施示例的图解。

具体实施方式

此后会通过图 1 来解释本发明。

根据图 1，应用 1 调用一个系统调用 2，此调用在步骤 3 被传送到一个模块，所谓的流水线模块。流水线模块的首要任务在于确定系统调用 4 的执行模式，即是阻塞系统调用还是非阻塞系统调用。如果流水线模块确认系统调用为阻塞调用，则会用熟知的方式进行处理，也就是启动网络操作 5 而仍阻塞调用应用 6，直至返回所启动网络操作的结果 7。在图 1 中识别应用的阻塞状态可以通过：没有箭头从阻塞执行模式返回到应用，而只是启动网络操作 5。在非阻塞系统调用的情况下直接向应用返回一个逻辑值 8。此步使应用没有保持阻塞而是继续其后续程序步骤，以便等候在伙伴实例中所实际执行的系统调用的结果成

为可能。术语“直接”在这里是指在启动网络操作的单元中确定逻辑值所需的时间。由于这在所述单元中本地完成，故所述时间远远少于网络操作经网络传输、系统调用在伙伴实例中执行以及发送和接收结果所需的时间。

5 在非阻塞执行模式的情况下直接返回逻辑值同样引起网络操作的启动 5。这发生在逻辑值返回之前不久或者紧随其后。对于非阻塞系统调用，在流水线模块中同样等候所启动网络操作的结果 6，但这独立于应用运行而发生。一旦接收到结果，则根据执行模式和返回的值检查并处理所述结果 9。

10 下面会说明一个处理所接收结果的示例。

首先，检查所接收结果的执行模式。如果结果源自非阻塞系统调用，则很明显应用此时正在运行，并在非阻塞状态中等候一个结果。为此需要存储所接收的结果。在下一步中，检查此结果的值，即它是否否定还是肯定。如果结果是否定的，则例如通过特定的注释来标记它，
15 以便为应用标记错误信息的必要性。否定结果表示出现了错误，例如连接的断开，而且必须向正在运行的应用报告此情况。如果尽可能早地作用于错误消息，则可保证考虑到错误消息的实现本发明的最安全方式。为此最好首先检查对于每个系统调用，是否存储了否定结果以及至今还未报告的先前调用的结果。如果所述检查认识到存在未决的
20 否定结果，则用逻辑否定值的形式向应用返回所述结果，作为当前正执行系统调用的返回值。

发生一个有关返回值的类似过程，其结果包含对正处于阻塞状态的应用的响应。在此情况下，同样首先检查是否存储了至今还未报告的否定结果。如果通过所述检查认识到存在未决的否定结果，则用逻辑
25 否定值的形式向应用返回所述结果，作为返回值。但是如果不存在否定结果，则向应用返回当前所接收的值。

理论上要谈及为了例如建立连接而发送的数据分组形式的消息。实际上可以用例如一种编程接口来实现它，这种编程接口实现了从应用层编程命令到网络层网络操作的转换。所述编程命令可以实现为使用
30 用户能访问网络服务的系统调用。

网络服务形式上的特征为多个服务单元或网络操作，由此用户能够使用服务。这些服务单元关心例如，服务执行某种活动、或用某种

方式响应伙伴实例的活动。在一个诸如 TCP/IP 或 OSI(开放系统互连)的协议栈中,不同服务单元相互区别。所述服务单元关心:一个服务执行某种活动。例如,通过服务单元“请求(request)”以及活动“CONNECT(连接)”来启动连接建立。在 OSI 格式中,相应消息

5 具有格式 CONNET.request(连接请求)。如果已满足所述请求,则伙伴实例经过一个指示来接收一个通告活动的信令。例如,定址的伙伴实例在 CONNET.request(连接请求)之后接收一个表示需要建立连接的 CONNET.indication(连接指示)。然后所述实例为了传达对于连接请求它想要接受还是拒绝而使用一个响应消息,在此情况下为

10 CONNET.response(连接响应)。发送最初 CONNET.request(连接请求)的伙伴实例通过一个确认,即所谓的 CONNET.confirm(连接确认)来获悉伙伴实例的就绪。所述例子涉及一个由 CONNECT(连接)启动的连接建立。还为诸如数据发送(SEND)等其他操作定义了可比较的服务单元,目前不会对它们进行更为详细的描述。但是所述

15 示例表示网络操作的规范非常抽象。例如,对于操作系统 Unix 4.3BSD,为了与系统无关地访问网络服务而引入了统一编程接口即所谓的套接字接口。这保证了服务单元更为明确的规范。因而消息的启动具体涉及应用层的编程命令规范。然后套接字接口实现了把此后被称为系统调用或者调用的编程命令转换成服务单元。为了解释本发

20 明,下面会使用所述的编程接口作为示例。

套接字编程接口包括一组系统调用,每个都涉及一个所谓的套接字。至于这个概念,一个套接字构成一个通信终结点,一侧连接网络连接,而另一侧连接应用。此后会更为详细的规定最重要的系统调用。用系统调用套接字生成新套接字,从而把地址格式、套接字类型和协议指定为参数。例如,对于 TCP 连接,选择了互联网地址格式、面向连接的套接字类型和 TCP/IP 协议。通过系统调用 connect(连接),一个应用请求到网络中远程套接字的连接。由此包括了本地套接字和远程套接字的地址作为参数。因此每个连接明确的特征在于一对套接字。数据的实际发送由调用 send(发送)来实现而所接收数据的读取

25 由调用 recv 来实现。除了这里所提及的示例外,还有更多系统调用,但是为了简洁不进行详细的描述。这些系统调用通过编程接口被转换成网络操作。这意味着例如调用 connect(连接)的执行引起三次握手

30

5 的启动。用户对网络操作的执行没有直接影响。但是用户通过系统调用可以控制对网络操作的访问。为了避免修改当前应用，本发明的目的也相应的是在系统调用和网络操作之间集成一个附加层，这使得可透明转换两种类型的操作却不改变应用的实现。此后被称为流水线模块的所述附加层的任务是对从应用所接收的系统调用进行翻译和相应处理。

但是流水线模块的主要任务在于确定系统调用的执行模式，即判定系统调用的结果是可以直接被传送到应用，还是应用必须等候服务器正执行操作的实际值。在第一种情况下涉及非阻塞系统调用，而在
10 第二种情况下涉及阻塞系统调用，在这种情况下应用期望所执行操作的实际值。这意味着流水线模块必须区别两种类型的系统调用，这可以以下面的方式来实现。如果返回值具有逻辑建议值，即在成功执行操作的情况下对于真返回 1，而在未成功执行操作的情况下对于假返回 0，则一个应用可以识别出有关一个非阻塞调用，而这隐含了一个给应用
15 的直接返回值。一旦接收到该值，应用就继续执行下一个操作。未决操作的状态（即已经发送给相连的服务器，却至今还未收到有关其的实际值的操作）在流水线模块中进行缓冲。状态的存储对于允许流水线模块一旦接收到实际值就向仍在未决状态的操作分配实际值是必要的。

20 在阻塞系统调用的情况下需要等候数据而非逻辑值来作为系统调用的返回值，故流水线模块并不向应用返回一个直接的值，而是等待直到收到实际数据。在此期间仍阻塞应用。除了那些不返回逻辑返回值而返回数据的系统调用，系统必须确保一个连接中最后的系统调用也被额外地设置成阻塞状态，因为只有用此方式才能保证返回先前所
25 执行操作的结果。根据套接字编程接口语义，在终止其使用时需要关闭套接字，其完成是通过使用系统调用 `close` 或 `shutdown`（关闭），即可以用阻塞模式实现的系统调用。

一个有关本发明保证的优化的问题在于非阻塞套接字调用的返回值的语义的改变，并不等候该调用的实际执行。为此必要的是，流水
30 线模块通过给应用的相应消息来确保控制所执行的系统调用。例如，如果由于网络故障造成了定址的 IP 地址不可达，而调用 `connect`（连接）失败，则在接收到相关返回值之后以及在应用认为所述调用已经

被成功终止之后，流水线模块仅获悉相应的错误。但是可以一旦接收到返回值就将其存储在流水线模块中并在下一系统调用中向应用进行报告。优选的实现方案规定通信直接发生在后面的系统调用中，保证了尽可能早地报告错误。检查未决错误报告的计算机时间少于在 GSM 中的传输时间。因此，存在于流水线模块中的返回值检查无需太多投入就可实现。

另一可能性在于在以阻塞状态执行的下一系统调用中报告所出现的错误。

这些实施方案取决于实现。但是在两种情况下必要的是在阻塞状态下实现最后系统调用的执行。

“延迟”的错误报告并未在已知的应用中带来任何问题，因为同样有关 TCP，这些应用已经准备好错误可以出现在任意时间。

此后会通过一个实施方案以及附图 3a 和 3b 来更为详细的解释本发明的实现。一方面，附图说明了应用所启动的、执行的系统调用，这在附图的左侧进行了解释。而附图的右侧表示了网络操作，由此启动了相应系统调用的执行。

根据图 3a 的实施方案表示了一个连接建立，它可以以现有技术即协议 HTTP 1.0 (RFC 1945) 实现。在这个用于调用 WWW 网页的协议中，为每个对象分别建立连接。根据图 3a 通过第一调用 connect (连接) 实现了一个连接。结果，诸如对象等数据是通过第一调用 send (发送) 和第二调用 recv 进行请求，并且传输，而此后连接是通过另一调用 close (关闭) 启动的。根据本发明，转换的系统调用并未经过传输装置来直接传输，而是首先传送到负责处理系统调用的流水线模块。根据图 3b 在第一步中启动第一系统调用 connect (连接)，这被流水线模块解释为一个非阻塞的系统调用。这里的解释是用这种方式：一方面流水线模块向服务器发送相应的网络操作，而同时直接向应用发送有关系统调用的结果而不等候服务器所执行操作的实际结果。结果的直接传输用直接返回箭头来标记。

此方法的作用在于应用不必在持续整个 RTT (往返行程时间) 期间都设置成阻塞状态，而是一旦接收到结果就直接继续下面步骤的执行。根据图 3b，是另一系统调用 send (发送)，它被使用与第一系统调用 connect (连接) 相同的方式处理。下一系统调用是 recv。所述系统

调用也首先被传输到流水线模块,此模块标识它为阻塞调用,因为期望数据作为返回值。为此流水线模块不能向应用返回一个直接的结果,而是等候在服务器中实际执行的系统调用的结果。由于计算机的操作时间相对于网络连接尤其是 GSM 连接的传输时间较短,故网络操作的发送直接依次进行。为此在现在的情况下节省了两个 RTT,因为并不等候系统调用 connect (连接) 和 send (发送) 的结果。所述结果随着系统调用 recv 的结果一起被返回。在图 3b 中通过较宽的箭头说明了直接依次发送网络操作,其中包括依照发送顺序的相应网络操作。

本发明的特征在于独立于传输平台,因为如果使用了不同于 T/TCP 的传输协议也确保了此功能性。在 T/TCP 的情况下,系统调用 connect (连接) 和 send (发送) 被解释为所谓的 sendto (发送到) 系统调用。但是使用所述其他协议在总体节省所需 RTT 的这方面(在所说明的示例中其数量从 3 降到 1)并未有任何改变。

下面将会更为详细的描述本发明。通过根据本发明经过长时延网络来快速执行网络操作的设备,借助一个确定单元来确定系统调用的执行模式。在这方面需要区分阻塞和非阻塞状态。系统调用通过转换装置被转换成网络操作。发送器向通信单元发送网络操作。在通信单元中所执行网络操作的结果由启动网络操作的单元通过接收器来接收。所接收的结果被存储在一个存储器中,并通过一个处理单元根据所包含的值来进行处理。区分肯定和否定的值。

在非阻塞系统调用的情况下,一旦已启动了相应的网络操作就等候在伙伴实例中所执行系统调用的结果,而不阻塞应用的执行。这是通过用于实现这种状态的单元来实现的。

本发明的实现基于在套接字编程接口下提供一个附加模块,上述流水线模块。为此本发明的实施必须通过考虑不同操作系统的不同配置而在操作系统层上实现。但是套接字编程接口的基本思想在于独立于协议,这就是为何在大多数操作系统中为了向套接字编程接口附加额外协议模块而提供内部软件接口,此内部软件接口便于本发明的实现。

此后基于示例的 Windows 操作系统和图 4 更为详细的解释本发明的实现。

图 4 描述了一个包括所涉及实例的协议栈。直接相邻的实例，应用、Winsock2、流水线模块和传输协议彼此间进行通信，这用箭头进行标记。相邻层的通信是通过一个已定义的软件接口进行，例如实例应用和实例 Winsock2 之间的通信是通过在 Windows 操作系统中称为 Winsock API 的套接字编程接口进行。软件接口 Winsock SPI（服务提供商接口）用于 Winsock2 层和底层实现的协议，在这种情况下是流水线模块，之间的通信，Winsock SPI 在微软 1997 年 8 月的“Windows Socket 2 Service Provider Interface(Windows Socket 2 服务提供商接口)” 2.2.2 版中有更为详细的解释。

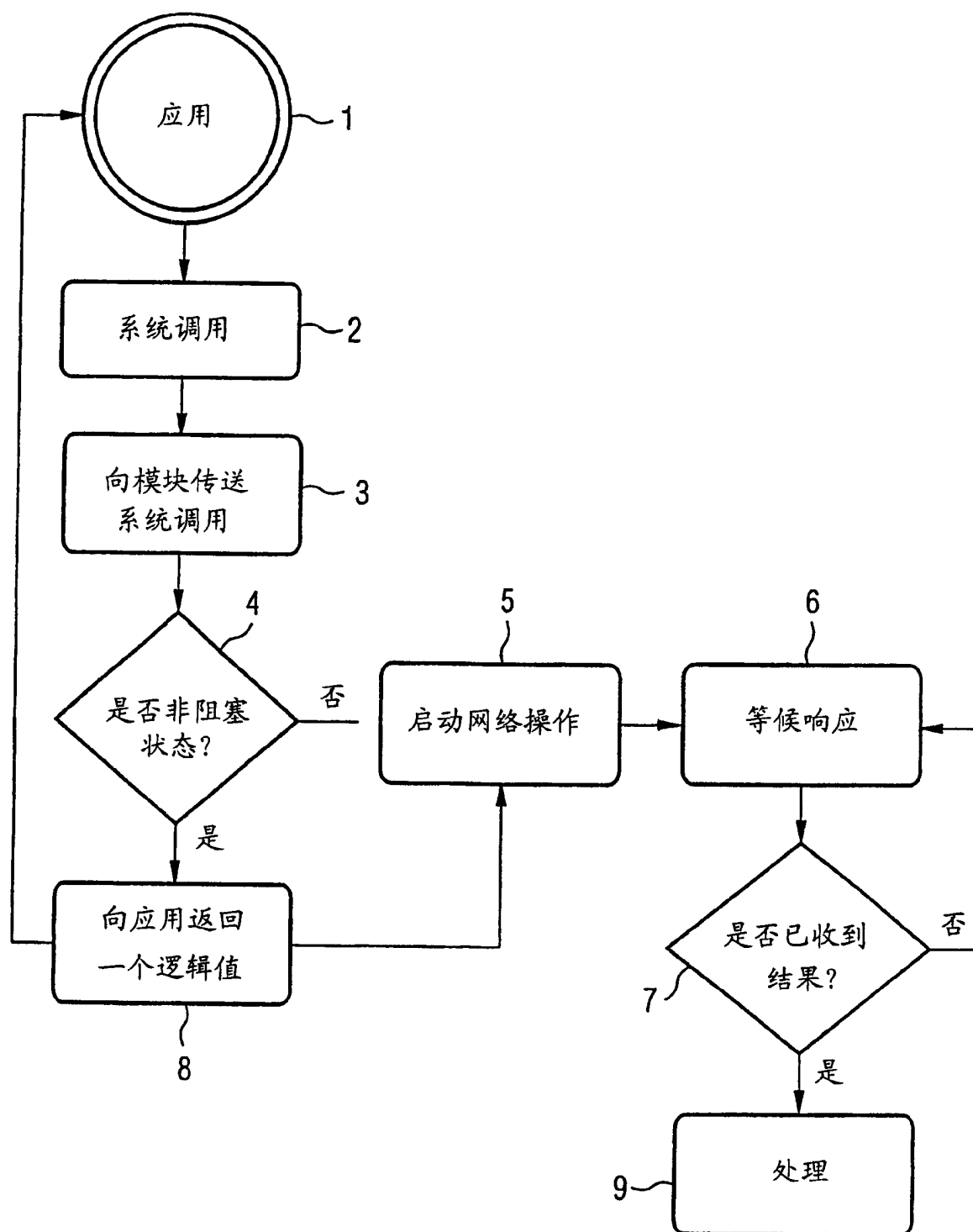


图 1

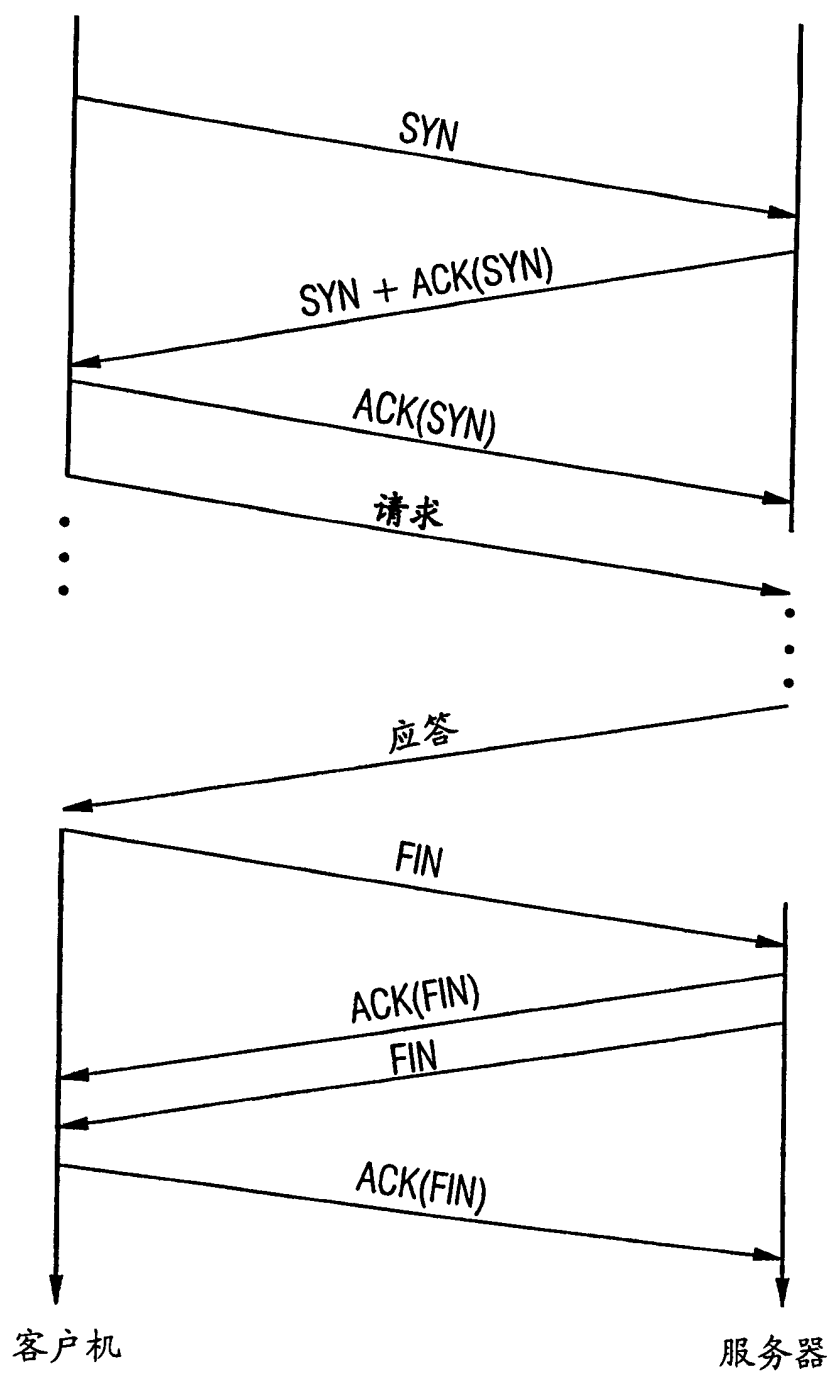


图 2

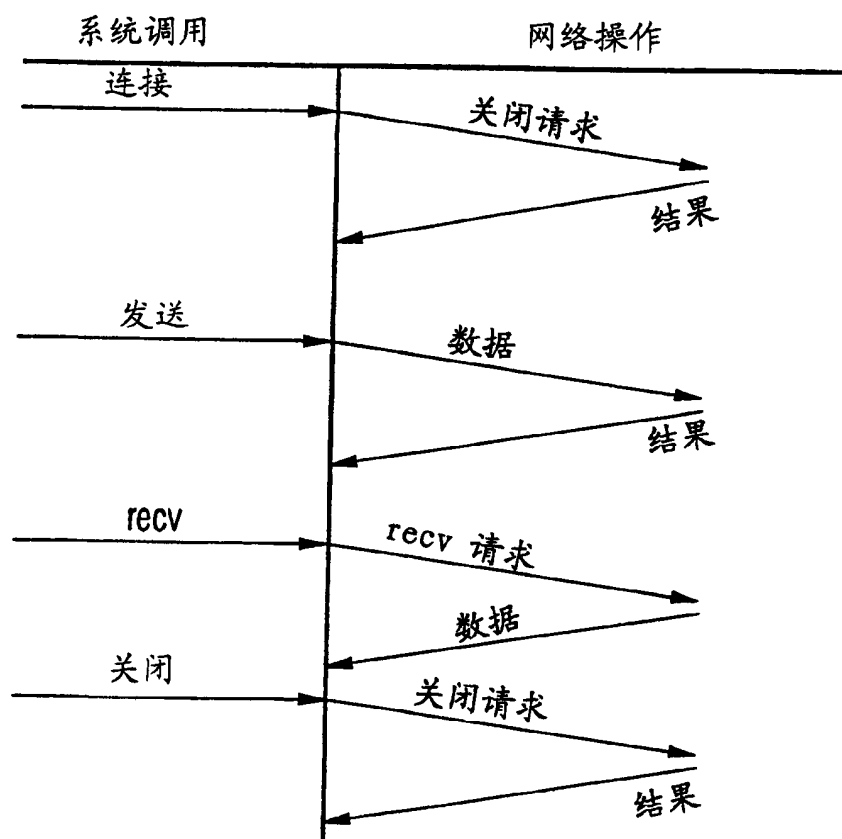


图 3a

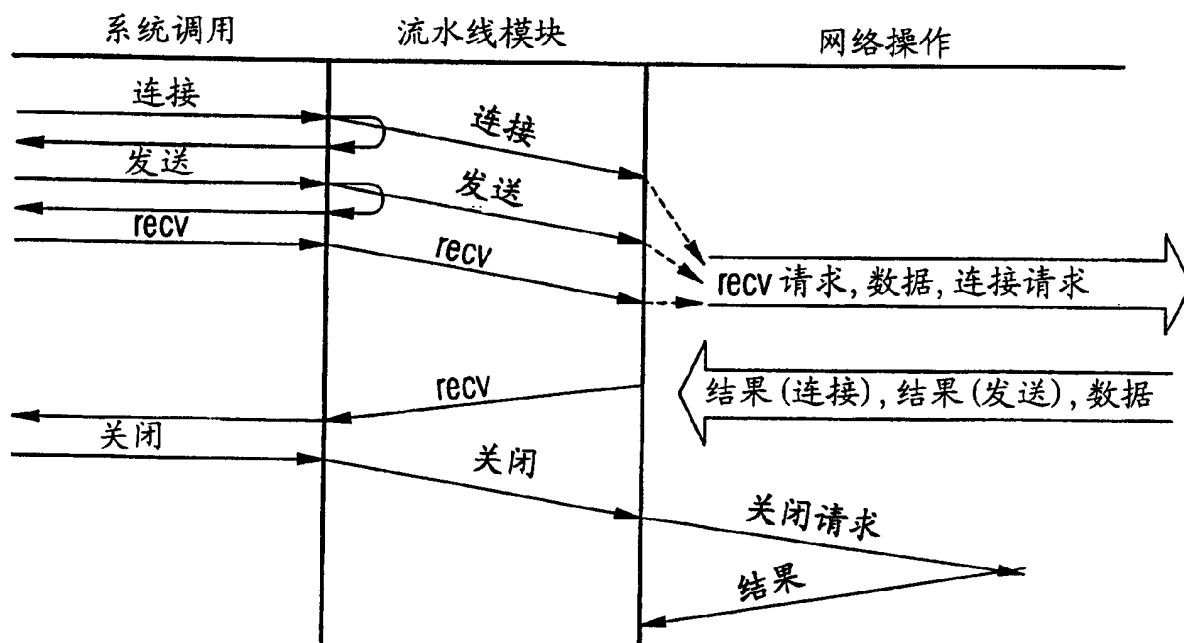


图 3b

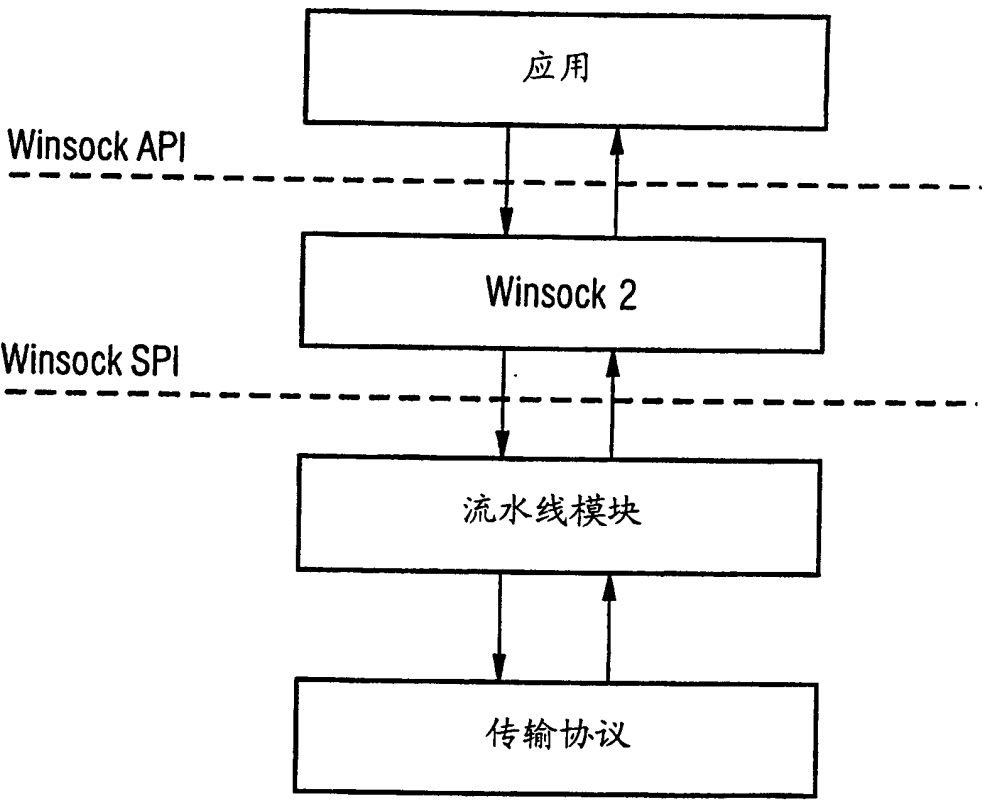


图 4