系统设计中的端到端参数

JH SALTZER，DP REED和DD CLARK
麻省理工学院计算机科学实验室

本文提出了一种设计原则，有助于指导功能在以下模块中的放置
分布式计算机系统。该原理称为端到端参数，表明功能
与系统的成本相比，放置在较低级别的系统可能是多余的或价值不大的
在较低的水平上提供它们。本文讨论的示例包括误码恢复，安全性
使用加密，重复消息抑制，从系统崩溃中恢复以及传递确认-
饰物。支持这些功能的低级机制只有在性能增强的情况下才是合理的，
。

CR类别和主题描述符：C.0 [General]计算机系统*组织-系统
建筑； C.2.2 [计算机通信网络]：网络*协议-协议*
tecture; C.2.4 [计算机通信网络]：分布式系统；D.4.7 **[运行
系统]：**组织和*设计-分布式系统*

通用术语：设计

其他关键字和短语：数据通信，协议设计，设计原则

1.引言

在功能之间选择适当的边界也许是主要活动
系统设计人员。在此提供指导的设计原则
选择功能放置是系统最重要的工具之一
设计师。本文讨论了一类函数放置参数，
已经使用了很多年，既没有明显的认可，也没有太多的信念，
tion。但是，数据通信网络作为计算机的出现
系统组件通过以下方式使函数放置参数这一行更加清晰
使其适用的情况和原因更加明显。
本文明确阐述了该论点，以检验其性质和
看看它到底有多普遍。该论点符合申请要求
并提供了在分层系统中向上移动功能的理由
到使用该功能的应用程序。我们首先考虑一下
nication网络版本的参数。

这**是** JH在《系统设计中的端到端参数》中改编的论文**的**修订版
第二届分布式系统国际会议上的Saltzer，DP Reed和DD Clark
（法国巴黎，4月8日至10日），第509-512页。©IEEE 1981
这项研究得到了美国高级研究计划局的部分支持。
国防部，由海军研究办公室根据合同N00014-75-进行监控
C-0661。
作者地址：麻省理工学院计算机科学实验室，JH Saltzer和DD Clark，545
技术广场，剑桥，马萨诸塞州02139。DP Reed，软件艺术公司，韦尔斯利市Mica Lane 27号，
MA O2181。

ACM **Transactions on** Computer Systems，第一卷。2，第4号，1984年11月，第277-288页。

第2页

在一个包含通信的系统中，通常会绘制一个模块
通信子系统周围的边界并定义一个牢固的接口
在它和系统的其余部分之间。这样做时，很明显
有一个功能列表，每个功能都可以通过以下几种方式实现
方式：由通信子系统，由其客户，作为合资企业，或
也许是多余的，每个人都有自己的版本。在考虑这个选择时，
应用程序的要求为以下类别的应用提供了基础
参数：

*只有使用以下方法才能完全正确地实现所讨论的功能
站在应用程序端点的知识和帮助
通讯系统。因此，将该功能作为功能提供
通信系统本身是不可能的。（有时不完整
通信系统提供的功能的版本可能会有用
性能增强。）*

我们将这种针对底层函数实现的推理称为
端到端参数。以下各节研究端到端参数
详细地讲，首先以使用它的典型示例为例进行研究-
所讨论的功能是可靠的数据传输-然后通过展示
可以应用相同参数的函数范围。对于
数据通信系统，此范围包括加密，重复邮件-
贤者检测，消息排序，保证消息传递，检测主机
崩溃和交货收据。在更广泛的背景下，该论点似乎适用
计算机操作系统的许多其他功能，包括其文件系统。
但是，如果我们先考虑一下，这种更广泛的背景下的检查会更容易
更具体的数据通信上下文。

2.正确的文件传输

2.1端到端护理

考虑*仔细的文件传输*问题。文件由文件系统存储在
计算机A的磁盘存储。计算机A通过数据通信链接
与计算机B进行网络连接，计算机B也具有文件系统和磁盘存储。的
目的是将文件从计算机A的存储移动到计算机B的存储
在没有损坏的情况下，请记住，故障可能发生在沿线的各个点
方式。在这种情况下，应用程序是文件传输程序，它是
其中的一部分在主机A上运行，另一部分在主机B上运行。
在此事务中对文件完整性的威胁，让我们假设
涉及的具体步骤：

（I）在主机A上，文件传输程序调用文件系统以读取文件

从磁盘时留在多个磁道上。文件系统通过
将其块固定大小的块（选择为磁盘格式）传输到文件传输程序
独立。

（2）同样在主机A上，文件传输程序询问数据通信系统
使用涉及拆分的某些通信协议传输文件
数据打包成数据包。数据包大小通常与文件不同
块大小和磁盘轨道大小。

ACM Transactions on Computer Systems，第一卷。2，第4号，1984年11月。

第3页

（3）数据通信网络将数据包从计算机A移动到
电脑B.

（4）在主机B上，数据通信程序从
数据通信协议，并将包含的数据移交给第二部分
在主机B中运行的文件传输应用程序的功能。

（5）在主机B上，文件传输程序要求文件系统写入接收到的信息。
主机B磁盘上的数据。

使用此模型所涉及的步骤，以下是一些潜在的威胁
细心的设计师可能会关心的事务：

（1）该文件虽然最初正确地写入了主机A的磁盘（如果已读取）
现在可能包含不正确的数据，可能是由于
磁盘存储系统。

（2）文件系统的软件，文件传输程序或数据
通信系统在缓冲和复制
主机A或主机B上文件的数据。

（3）硬件处理器或其本地内存可能有暂时性错误
在主机A或主机B上进行缓冲和复制时。

（4）通信系统可能会丢弃或更改数据包中的位或
多次发送一个数据包。

（5）在发生交易后，其中任一主机可能会在交易过程中部分崩溃
执行未知数量（可能全部）的交易。

那么仔细的文件传输应用程序将如何处理此威胁列表？
一种方法可能是加强使用过程中的每个步骤
复制副本，超时并重试，仔细定位冗余以防止错误
检测，崩溃恢复等。目标是减少
每个威胁的价值都可以接受。不幸，
系统地应对威胁（2）需要编写正确的程序，即
非常困难。另外，并非所有必须正确的程序都由
文件传输应用程序程序员。如果我们进一步假设所有这些威胁
的概率相对较低-足够使系统无法进行有用的工作
要实现-暴力对策，例如做三件事
时代，显得不合算。

替代方法可能称为*端到端检查和重试*。假设
与每个文件一起存储的用于应对威胁（1）的校验和
具有足够的冗余以减少文件中未检测到错误的机会
达到可以接受的微不足道的价值。应用程序遵循简单
将文件从A传输到B的上述步骤。然后，作为最后的附加步骤，
主机B中的文件传输应用程序部分读取已传输的文件
文件从其磁盘存储系统复制回其自己的内存，重新计算
校验和，然后将此值发送回主机A，与主机A进行比较

原始的校验和。只有两个校验和一致，文件传输
应用程序声明已提交的事务。如果比较失败，则说明
错误，可能会尝试从头开始重试。

如果失败很少发生，则此技术通常会在第一次尝试时起作用；
有时可能需要第二次甚至第三次尝试。一个可能
将同一文件传输尝试中的两次或更多次失败视为表明
该系统的某些部分需要维修。

ACM Transactions on Computer Systems，第一卷。2，第4号，1984年11月。

第4页

现在，让我们考虑一项共同提案的有用性，即
通信系统在内部提供可靠的数据传输保证
sion。它可以通过在
数据包校验和，序列号检查和内部重试的形式
机制。充分注意，未检测到位的可能性
错误可以减少到任何期望的水平。问题是这是否
尝试对通信系统有所帮助对
仔细的文件传输应用程序。

答案是威胁（4）可能已经消除，但谨慎的做法
转移应用程序仍必须应对其余威胁；所以它应该仍然
根据文件的端到端校验和提供自己的重试。如果有的话
在通信系统中花费的额外努力以提供保证
可靠的数据传输只会减少文件重试的频率
转让申请；它对结果的必然性或正确性没有影响，
由于端到端校验和可确保正确的文件传输并重试
数据传输系统是否特别可靠。

因此，参数：为了实现仔细的文件传输，应用程序
执行传输的程序必须提供特定于文件传输的端到端
终端可靠性保证-在这种情况下，是用于检测故障的校验和，然后重试-
提交计划。为了使数据通信系统不至于成为现实
非常可靠，不会减轻应用程序的负担
确保可靠性。

2.2太真实的例子

一个有趣的例子，一个人可能遇到的陷阱最近出现了
在麻省理工学院。一个涉及网络的系统
网关连接的几个局域网在每个局域网上使用数据包校验和
假设存在以下主要威胁，则从一个网关跳到下一网关
正确的通信是传输过程中的位损坏。应用
知道此校验和的程序员认为网络正在提供
可靠的传输，而没有意识到传输的数据不正确
当存储在每个网关中时进行检测。一台网关计算机产生了瞬态
错误：将数据从输入复制到输出缓冲区时，一个字节对是
互换的频率，大约每百万分之一这样的互换
传递的字节数。在一段时间内，一个操作的许多源文件
系统通过有缺陷的网关反复传输。其中一些
字节交换破坏了源文件，它们的所有者被迫
最终的端到端错误检查：手动比较和更正
从旧列表。

2.3性能方面

但是，过于简单地得出结论，较低的级别应该发挥作用
没有获得可靠性的任何部分。考虑一个不太可靠的网络，
每发送一百条消息，仅丢弃一条消息。简单策略
如上所述，传输文件，然后检查文件是否具有
正确到达时，随着文件长度的增加，性能会更差。
文件的所有数据包正确到达的概率呈指数下降

ACM Transactions on Computer Systems，第一卷。2，第4号，1984年11月。

第5页

与文件长度有关，因此传输文件的预期时间会增加
与文件长度成指数关系。显然，需要在较低层次上进行一些改进
网络可靠性可能会对应用程序性能产生重大影响。但
这里的关键思想是较低级别不需要提供“完美”的可靠性。

因此，在数据内投入可靠性度量的工作量
通信系统被认为是基于性能的工程折衷，
而不是对正确性的要求。注意性能有
这里有几个方面。如果通信系统太不可靠，该文件
传输应用程序的性能将因频繁重试而受到影响
端到端校验和失败。如果通讯系统得到加强
如果采用内部可靠性措施，这些措施也会产生性能成本，
以带宽丢失给冗余数据的形式，增加了等待的延迟
在传送数据之前完成内部一致性检查。有
当人们认为*最终结果-*
无论如何，仍必须执行文件传输应用程序的端到端检查
*通信系统变得多么可靠。*在*适当的*权衡，需要
仔细思考。例如，可以从设计交流开始
系统仅提供成本低，工程少的可靠性
努力，然后评估剩余错误级别以确保其一致
在文件传输级别具有可接受的重试频率。可能不是
在低于应用程序的任何位置上争取可忽略的错误率很重要
水平。

必须使用性能证明将功能放置在低级子系统中的合理性
要小心做。有时，通过彻底检查问题，同样
或更高级别的性能增强。表演
如果可以执行低级别的功能，则该功能可能会更有效
最低限度地干扰了已经包含在底层的机械
子系统。但是可能会发生相反的情况-即执行功能
较低的价格可能会花费更高-有两个原因。一，自下而上
子系统对于许多应用程序来说是通用的，而那些不需要的应用程序
该功能将为它付费。其次，底层子系统可能不会
拥有与较高级别一样多的信息，因此它不能像
有效率的。

通常，性能折衷是相当复杂的。再考虑一下
在不可靠的网络上进行仔细的文件传输。常用的增加技巧
数据包可靠性是某种带有重试协议的每数据包错误检查。
该机制可以在通信子系统中实现，也可以在通信子系统中实现。
在仔细的文件传输应用程序中。例如，接收者在小心
文件传输可以定期计算文件部分的校验和，从而
远接收并将其发送回发送者。然后发送者可以重新启动
通过重新传输已到达错误的任何部分。

端到端的论点并没有告诉我们在哪里进行早期检查，因为
任何一层都可以执行此性能增强工作。尽早重试
文件传输应用程序中的协议简化了通信系统，但是
由于通信系统由其他人共享，可能会增加总体成本
应用程序，每个应用程序现在必须提供自己的可靠性增强功能，
精神。在通信系统中放置早期重试协议可能会更多

ACM Transactions on Computer Systems，第一卷。2，第4号，1984年11月。

6页

高效，因为它可以在网络内部逐跳执行，
减少纠正故障所涉及的延迟。同时可能
是一些发现增强功能的成本不值得的应用程序
结果，但现在别无选择。1有关的大量信息
需要系统实施才能智能地做出此选择。

3.端到端参数的其他示例

3.1交货保证

基本论点是，较低级别的子系统支持分布式
应用程序可能在浪费其精力，以提供一种本质上必须具有的功能，
可以在应用程序级别上实现，无论如何都可以应用于
除了可靠的数据传输外，还具有其他功能。也许是最古老和最古老的
争论的广为人知的形式是关于交付的确认。一种
数据通信网络可以轻松地向发送方返回确认
传递给收件人的每封邮件。例如，ARPANET返回
每当它传递时被称为*请求下一个消息*（RFNM）[1] 的数据包
一个消息。尽管此确认在网络中可能是有用的
一种拥塞控制（原本是ARPANET拒绝接受另一种
消息发送到同一目标，直到先前的RFNM返回为止）
被发现对使用ARPANET的应用程序非常有帮助。原因是
不能肯定地知道邮件已传递到目标主机
很重要。应用程序要知道的是目标是否
主持人对消息采取了行动；消息传出后，各种灾难可能已经袭来
发送，但在完成消息请求的操作之前。的
真正需要的确认是端到端的确认，可以是
仅源于目标应用程序-“我做了”或“我没有”。

获得即时确认的另一种策略是使
目标主机足够复杂，当它接受邮件传递时
还承担保证消息由以下人员采取行动的责任
目标应用程序。这种方法可以消除对端到端的需求
在某些（但不是全部）应用程序中确认。端到端确认-
对于目标请求动作的应用程序仍然需要
仅当成功请求其他主机的类似操作时，才应执行host。
这种应用需要两阶段提交协议[5、10、15]，
是一种复杂的端到端确认。另外，如果目标应用
要么失败，要么拒绝执行所请求的操作，因此，否定确认-
边缘化是可能的结果，端到端的确认仍然可能是
需求。

3.2数据的**安全传输**

可以应用端到端参数的另一个领域是数据领域
加密。这里的论点是三重的。一，如果数据传输系统
执行加密和解密，必须信任它才能安全地管理
必需的加密密钥。其次，数据将是明文数据，因此容易受到攻击

1例如，语音的实时传输对消息延迟的约束比对位延迟的约束要严格。
错误率。大多数重试方案会大大增加延迟的可变性。

ACM Transactions on Computer Systems，第一卷。2，第4号，1984年11月。

第7页

当它们传递到目标节点并扇形展开到目标应用程序时。
第三，消息的*真实性*仍必须由应用程序检查。
如果应用程序执行端到端加密，则它将获取其所需的身份验证。
抽签检查并可以处理密钥管理使其满意，并且数据
永远不会暴露在应用程序外部。

因此，为了满足应用程序的要求，不需要
通信子系统提供对所有流量的自动加密。
通信子系统可以自动加密所有流量
但要求确保其他行为-行为不端的用户或
应用程序不会故意传输不应
被暴露。将所有数据放入网络后自动进行加密
是系统设计人员可以用来确保信息确实可以使用的另一种防火墙
不能逃离系统之外。但是请注意，这是一个不同的要求
从验证系统用户对数据特定部分的访问权限。
这种网络级加密可能非常简单-相同的密钥可以
由所有主机使用，且密钥经常更改。每个用户密钥都不复杂
关键管理问题。在应用程序级别的au-中使用加密
认证和保护是相辅相成的。两种机制都无法满足
完全符合这两个要求。

3.3重复消息抑制

可以将更复杂的参数应用于重复消息抑制。
一些通信网络设计的一个特性是消息或一部分
一条消息可能会传递两次，通常是由于超时触发
网络内运行的故障检测和重试机制。的
网络可以监视并抑制任何此类重复消息，也可以简单地
送他们。可能有人希望应用程序会发现它很麻烦
处理可能两次发送相同消息的网络；它的确是
麻烦。不幸的是，即使网络禁止重复，
复制本身可能会因自身故障而意外地发出重复请求/
重试程序。这些应用程序级重复看起来像不同的消息-
对通信系统的先知，因此它无法抑制它们；抑制
必须由应用程序本身完成并具有如何检测的知识
它自己的副本。

重复抑制的常见示例，必须高度重视
级别是当远程系统用户由于缺乏响应而感到困惑时，发起一个新的
登录到分时系统。另一个例子是，大多数沟通
应用程序涉及一项应对系统崩溃的规定
多站点事务：当系统崩溃时重新建立事务
再次上升。不幸的是，可靠地检测系统崩溃是有问题的：

问题可能只是丢失或延迟很长时间的确认。如果是这样，则重试
现在，该请求是重复的，只有应用程序才能发现。就这样
再次使用端到端参数：如果应用程序级别必须重复-
无论如何，这种抑制机制也可以抑制任何重复
在通信网络内部生成的目录；因此，该功能可以
从较低的层次上省略。相同的基本推理完全适用
省略的消息，以及重复的消息。

ACM Transactions on Computer Systems，第一卷。2，第4号，1984年11月。

第8页

3.4保证FIFO消息传递

确保消息以与接收消息相同的顺序到达接收者
发送是通常分配给通信子系统的另一个功能。
通常用于实现这种先进先出（FIFO）行为的机制
确保在同一虚拟电路上发送的消息之间的FIFO顺序。
沿着独立的虚拟电路或通过中间过程发送的消息
散布在通信子系统之外，可能到达的顺序与
订单已发送。一个节点可以在其中发起请求的分布式应用程序
在几个站点上启动操作的操作无法利用FIFO排序
属性，以确保请求的操作以正确的顺序发生。
相反，它是一个比通讯更高层次的独立机制
子系统必须控制动作的顺序。

3.5交易管理

现在，我们在构建
SWALLOW分布式数据存储系统[15]
减少开销。SWALLOW提供称为reposi-的数据存储服务器
可以远程用于存储和检索数据的Tory。存取资料
通过向存储库发送一条消息来指定要访问的对象来完成存储库，
版本和访问类型（读/写），以及如果要写入的值
访问是写操作。基础消息通信系统没有
禁止重复消息，因为（a）对象标识符加上版本
信息足以检测重复的写入，并且（b）重复的影响
读取请求消息仅生成重复的响应，这很容易
被创建者丢弃。因此，低级消息通信
协议已大大简化。

基础消息通信系统不提供传递
确认。确认写入的发起者
请求的需求是数据已安全存储。此确认可以是
仅由高级SWALLOW系统提供。对于读取请求，
传递确认是多余的，因为响应包含值
阅读就足够了。通过消除交货确认，
传输的消息数量减半。此消息减少可以有
对主机负载和网络负载均产生重大影响，从而提高了性能。
同样的推理方法也已用于实验的开发
远程访问磁盘记录的协议[6]。导致路径减少
在较低级别的协议中，长度对于保持良好的性能非常重要-
远程磁盘访问。

4.确定目标

使用端到端参数有时需要对应用程序进行细微的分析。
阳离子要求。例如，考虑一个计算机通信网络
承载一些分组语音连接，即数字之间的对话
电话仪器。对于那些携带语音数据包的连接，
通常使用端到端参数的强版本：如果
通讯系统尝试完成位完美的通讯，他们将
可能会在数据包传送中引入不受控制的延迟，例如，通过重新

ACM Transactions on Computer Systems，第一卷。2，第4号，1984年11月。

第9页

要求重发损坏的数据包并阻止以后的数据包发送
直到更早的数据包已正确重传。这样的延误是
破坏语音应用程序，后者需要以恒定的速率馈送数据
给听众。最好接受稍有损坏的数据包，甚至
用静音，上一个数据包的副本或噪声突发替换它们。
语音的自然冗余以及高级纠错
一位参与者说“对不起，有人丢了一杯。
您能再说一遍吗？”
很少。

但是，端到端参数的这一强大版本是
特定的应用程序-两个人进行实时对话-而不是道具
一般说来，说话很不正确。相反，如果考虑使用语音消息系统，
其中语音数据包存储在文件中，供接收者以后收听，
这些论点突然改变了它们的性质。数据包传送中的短暂延迟
对存储介质的破坏不是特别严重，因此不再存在
反对可能引入延迟的低级可靠性措施，以便
实现可靠性。更重要的是，它实际上对这个应用程序有帮助
因为接收者在
收听录音的时间，将无法要求发送者
重复一句话。另一方面，使用存储系统作为
接收语音通信结束时，端到端参数确实适用于
数据包排序和重复抑制。因此，端到端参数不是
绝对规则，而是有助于应用程序和协议的指南
设计分析；人们必须格外小心地确定
参数应适用。

5.历史，以及在其他系统领域的应用

本文中引用的端到端论证的各个示例不是
原版的; 他们已经积累了多年。可疑的第一个例子
作者注意到的中间交付确认是“等待”
麻省理工学院兼容分时的讯息
系统，每当用户在用户终端上打印该系统
输入命令[3]。{该消息在
系统，当崩溃和通讯故障如此频繁以至于
调解确认提供了一些必要的保证，以确保一切都很好。）

关于加密的端到端争论首先被公开讨论
由Branstad在1973年发表的一篇论文[2]；大概是军事安全界
在此之前进行了分类讨论。迪菲和海尔曼[4]和肯特[8]

避此曾两次证论证。Needham和Schroeder [11]提出了
Gray [5]，Lampson和Sturgis的两阶段提交数据更新协议
[10]和里德[13]都使用一种端到端的论证形式来证明他们的存在。
它们是不依赖于可靠性的端到端协议，
FIFO排序或通信系统中的重复抑制，
因为所有这些问题也可能由其他系统组件引入
失败也是如此。里德在他的第二章中明确提出了这一论点。
博士 分散原子作用的论文[14]。

ACM Transactions on Computer Systems，第一卷。2，第4号，1984年11月。

第10页

端到端参数通常应用于错误控制和
应用系统。例如，银行系统通常提供高级
根据政策和法律要求的审核程序。那些高层
审核程序不仅会发现高级错误，例如执行
针对错误的帐户提款，但他们也会检测到低级别
基础数据管理系统中的错误，例如协调错误。
因此，绝对消除这种协调错误的昂贵算法
可能比一个成本更低的算法更不合适
这样的错误非常罕见。在航空公司预订系统中，可以依靠代理商
继续尝试解决系统崩溃和延迟，直到发生保留
确认或拒绝。较低级别的恢复程序，以确保
未经确认的保留请求将在系统崩溃后幸存下来
重要。在电话交换机中，可能导致单个呼叫丢失的故障是
被认为不值得为之提供明确的恢复，因为调用者可能会
如果重要，请更换电话[7]。所有这些设计方法都是
端到端参数应用于自动恢复。

网络协议社区中有关虚拟数据报的许多争论
电路和无连接协议是有关端到端参数的争论。一种
模块化自变量奖励可靠的，FIFO顺序，重复抑制的
作为易于构建的系统组件的数据流，以及该参数
支持虚拟电路。端到端的论点声称，集中提供
对于某些应用程序，每个功能的版本都不完整，并且
这些应用程序将发现构建自己的功能版本更加容易
从数据报开始。

非通信应用程序中的端到端参数的版本是
由系统分析员于1950年代开发，其职责包括阅读
在大量磁带盘上写入文件。反复尝试
定义并实现一个*可靠的磁带子系统*，该*子系统*反复地被破坏
磁带驱动器，不可靠的系统操作员以及共同导致的系统崩溃
所有狭focused的可靠性指标。最终，它成为标准做法
为每个应用程序提供自己的与应用程序相关的检查和回复
策略，并假设最好使用较低级别的错误检测机制，
降低了较高级别检查失败的频率。举个例子，
Multics文件备份系统[17]，即使它是基于以下内容构建的：
磁带子系统格式，可提供非常强大的错误检测和
更正功能，以记录标签的形式提供自己的错误控制
以及每个文件的多个副本。

用于支持精简指令集计算机的参数

通过实施该系统的客户端到端参数的性能
正是原始工具所需的指令；计算机的任何尝试
设计人员可以预期客户对深奥功能的要求
可能会稍微错过目标，客户最终将重新实现该目标
反正功能。（我们感谢Satyanarayanan M.指出了这一点
例。）

兰普森（Lampson）在支持*开放操作系统的*论点中，[9]使用了
作为端对端参数，类似于端到端参数。兰普森认为

ACM Transactions on Computer Systems，第一卷。2，第4号，1984年11月。

第11页

禁止将任何功能永久固定在较低级别的模块上；的
功能可能由较低级别的模块提供，但应始终
可被应用程序的特殊功能版本取代。推理是
对于任何可以想到的功能，至少有些应用程序会发现
他们必须必须自己实现功能才能满足
正确地符合自己的要求。这条推理路线将兰普森引向
提出一个“开放”系统，其中整个操作系统包括
库中的可替换例程。这种方法直到最近才成为
在专用于单个应用程序的计算机的上下文中是可行的。可能是
大量的固定监督员功能通常是
规模操作系统只是造成经济压力的产物
要求对昂贵的硬件进行多路复用，因此需要一种受保护的超级
遮阳板。实际上，最近的系统“内核化”项目至少关注于
部分原因在于从低系统级别获得功能[12，16]。虽然这个功能
运动是受另一种正确性论证启发的，它具有侧面
产生对应用程序更灵活的操作系统的效果，
这正是端到端论证的主要重点。

6。结论

当涉及到选择时，端到端争论是一种“奥卡姆剃刀”
在通信子系统中提供的功能。因为com-
通讯子系统通常在使用
子系统是已知的，设计者可能会倾向于通过“帮助”用户
在不必要的功能上。了解端到端参数可以帮助您
减少这种诱惑。

这些天来谈论*分层*通信协议很流行，但是
没有为图层分配功能的明确定义标准。这样的分层
需要增强模块化。端到端参数可以被视为
组织这种分层系统的一组合理原则的一部分。我们希望
我们的讨论将有助于为有关“适当”的争论增加实质
分层。

致谢

许多人已经阅读并评论了本文的早期草案，包括
David Cheriton，FB Schneider和Liba Svobodova。该主题还
在ACM“分布式计算基础”研讨会上进行了讨论，
1980年12月，加利福尼亚州法尔布鲁克。那些评论和讨论相当
有助于阐明论点。