SIP保留事务状态代理中基于 UDP协议的状态共享机制

罗仕漳, 廖建新, 朱晓民
(北京邮电大学网络与交换技术国家重点实验室, 北京 100876)

摘 要: 已有的 IMS(IP multimedia subsystem)容错系统仅仅实现了保留呼叫状态代理下的状态共享机制. 提出并实现了保留事务状态代理中基于 UDP(User Datagram Protocol)协议的状态共享机制. 在大话务量测试的基础上, 给出了可靠性的测量值曲线图. 通过数学分析和理论推导, 提出并证明了超时参数 T_{out}与可靠性 R之间的关系公式. 实验表明, 理论分析的结果能够反映出系统的真实性能, 推导出的 T_{out}断裂点值与实际值是一致的.

关键词: IP多媒体子系统; 下一代网络; 状态共享; 容错; 会话初始协议; 可靠性
中图分类号: TN915 文献标识码: A 文章编号: 0372-2112 (2006) 07-1311-05

A UDP-Based State-Sharing Mechanism of SIP Transaction Stateful Proxy

LUO Shi-zhang, LIAO Jian-xin, ZHU Xiao-min
(State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876)

Abstract Current IMS (IP multimedia subsystem) fault-tolerant systems only implemented state-sharing mechanism of SIP call stateful proxy. We proposed and implemented a UDP-based state-sharing mechanism of SIP transaction stateful proxy. Based on heavy stress testing, experimental reliability figure is shown. By mathematical analysis and derivation, we proved the proposed relationship equation of T_{out} and reliability R. Experiments showed that the results of the theoretical analysis can reflect the real performance of the system, and the theoretical and practical T_{out} breakpoint value is consistent.

Key words IP multimedia subsystem; next generation network; state-sharing; fault-tolerance; session initiation protocol; reliability

1 引言

IMS(IP multimedia subsystem)[1]中, 为了对 SIP(Session Initiation Protocol)会话提供高可靠性, 特定的 CSCF(Call Session Control Function)必须提供容错系统来增加业务的可靠性. 一种方法是特定的几个 CSCF服务器组成一个服务器池. 在这个服务器池中, 每个对等体服务器都拥有完全相同的呼叫/事务状态.

SIP定义了三种代理服务器: 保留呼叫状态代理(call stateful proxy), 保留事务状态代理(transaction stateful proxy)和不保留状态代理(stateless proxy)[2]. 他们广泛的存在于 NGN(Next Generation Network)网络中. 典型情况下, 保留呼叫状态代理作为 P-CSCF(Proxy-CSCF)存在, 保留事务状态代理作为 I-CSCF(Interrogation-CSCF)存在; 为了进行信息的简单快速转发, 不保留状态代理处于网络的核心.

M. Bozinovski在研究过程中, 通过借鉴分布系统中的各种复制技术[3]实现了保留呼叫状态代理的容错系统. 在文献[4~6]中, 他提出了一种新的、作为容错系统组件之一的状态共享(state-sharing)机制, 并通过理论和实验得到了 FTP(File Transfer Protocol)和 TFTP(Trivial File Transfer Protocol)下的断裂点值(breakpoint), 并进一步得出 TFTP优于 FTP的结论. 文献[7, 8]继承了之前的成果, 在容错系统中增加了并发指派协议, 保证了状态传递过程中的一致性. 文献[9]详细阐述了失败接管中的服务器选择策略, 实现了错误发生时对接管服务器的动态选择.

M. Bozinovski很好地实现了保留呼叫状态代理的容错系统, 但是, 他没有触及保留事务状态代理下的容错问

收稿日期: 2005-09-09; 修回日期: 2006-04-26
基金项目: 高等学校博士学科点专项科研基金(No. 20030013006); 国家移动通信产品研究开发专项基金; 电子信息产业发展基金重点项目; 电子信息产业发展基金项目; 国家高技术产业化信息化装备专项项目

题.保留事务状态代理的一个典型的例子是派生代理.为了知道所有尝试过的地址都返回了最终响应,派生代理必须存储有关 INVITE事务的状态信息.它关心事务级的状态,而保留呼叫状态关注呼叫级状态,它们在理论上和实现中都相差很大,所以研究事务级的容错系统有着十分重要的现实意义.

在本文中,我们实现了保留事务状态代理下基于UDP协议的容错系统.详细解释了保留事务状态代理的错误侦测和失败接管机制.通过理论分析和实验,我们给出了可靠性的测量值曲线图.最后,我们提出并证明了T_{out}与可靠性R之间的关系公式,并通过此公式,证明了T_{out}断裂点的理论值与实际值是一致的.

2 保留事务状态代理的状态共享机制

2.1 状态共享机制的功能模型

作者构建的状态共享机制结构如图1所示.其中服务器1和服务器2运行相同的状态共享机制.我们假定每个服务器池只包含两个服务器,当然,此机制也可应用于更大的服务器池中.

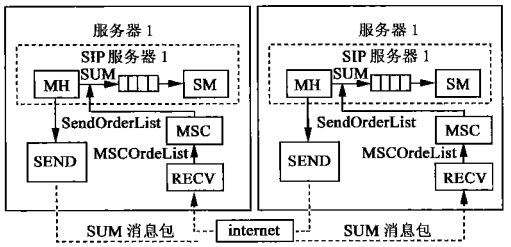

图1 通用状态共享机制结构

状态共享机制包含三个逻辑部分:SEND,MSC(Message-to-SUM Converter)和RECV.其中,SEND负责将SUM(StateUpdateMessage)为元素的向量转换为消息包格式并发送给对等体服务器.SUM被定义为一个数据结构,它包含了一些事务状态.SUM强制包含三个呼叫状态信息:主叫SIP地址,被叫SIP地址,和呼叫ID.依据SIP服务器状态机所处的状态不同,其他事务状态可以加入作为SUM的内容.RECV负责接收对等体服务器传来的消息包,并转化为以SUM为元素的向量.MSC则将此向量的元素取出并分别放入到SM(StateManager)中.

在此状态共享机制中,我们把一个SUM的所有事务状态元素封装入单个消息包中,形成一个字符串类型数据,发送给对等体服务器.经过UDP协议的传送,形成服务器间呼叫状态的相同镜像.

此机制的运行过程如下:

1.MH(MessageHandler)产生SUM,并把它加入到SM入队列中.MH产生以SUM为元素的向量SendOrderList,SEND从中取出每个SUM,并将它转化字符串类型,并发送这个消息包至对等体服务器.

2.RECV侦听对等体服务器.当有消息进入时,它打开接收到的消息包并将它转化为以SUM为元素的MSCOrdeList向量中.

3.MSC将MSCOrdeList中的SUM取出并加入到SM的入队列中.

2.2 状态共享机制的实现

作者构建的共享状态机制实现如图2所示.其中,MSCOrdeList存储了RECV线程的输出结果.它是vector类型的数据,元素为SUM.它同时又作为MSC线程的输入数据而存在.MHOrdeList也是vector类型的数据,从SIP消息中提取的状态信息组成SUM保存在其中作为元素.它同时也作为MH线程的输入数据.SendOrdeList是SEND线程的输入数据.Vector类型并以Message对象作为其元素.它作为发送到对等体服务器的数据源存在.SM存放所有的状态信息,它是list类型数据.MSCOrdeList,MHOrdeList,SendOrdeList使消息的接收发送和机制内部的数据转换分开,避免大业务量时直接放入SM带来的数据拥塞和数据丢失.

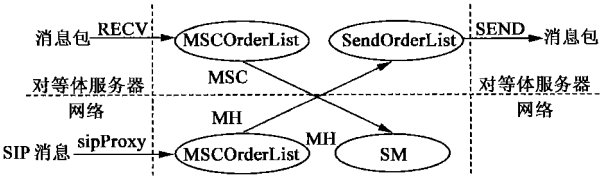

图2 共享状态机制的实现

实现中,状态共享机制由7个线程实现.其中RECV线程从对等体服务器中收取消息包,并解析出SUM,作为元素放入MSCOrdeList中.MSC线程提取MSCOrdeList中的数据,并转换为SUM放入SM中(或删除SM中的SUM).在容错系统的实现中,这被称为远程SUM的输入.SipProxy线程收取网络中SIP消息,从中提取出状态信息,形成SUM,放入MHOrdeList中.MH线程将MHOrdeList转化为以Message对象为元素的SendOrdeList中,将它作为SEND线程的输入数据.SEND线程提取出SendOrdeList结构中的Message对象,把Message对象转化为字符串,形成状态消息包,并通过UDP协议发送给对等体服务器.同时,MH线程还从MHOrdeList中提取出SUM,查询并放入新的状态至SM中,这被称为本地SUM的输入.远程SUM和本地SUM组成了SM的SUM来源.除此之外,我们还实现了MONITOR线程,它使用TELNET方式开启关闭各线程的跟踪信息,在终端实现了对各线程的监控功能.系统还有一个主进程,它用来读取配制文件,生成启动各个进程.状态共享机制是在UNIX环境下使用HP OpenCall SIP Stack v121开发实现的.各线程并发执行,并使用互斥锁,条件变量通知等待等技术实现了数据间同步[10].调用了HP OpenCall SIP协议栈的Transaction级函数,实现了SIP消息的回调函数[11].

2.3 错误侦测和失败接管

SIP UAS(User Agent Server)使用定时系统来进行错

误侦测. 当最终响应的定时器超时 (T_{out})时, 错误被侦测到. 超时值 T_{out}可以改变. 错误侦测触发失败接管机制. 采用服务器选择策略 (SSP, Server Selection Policy), 可以选择下一服务器来处理新的请求[9].

根据搜索的类型是并行还是串行, 错误出现在搜索过程中还是出现在搜索成功后, 派生代理的失败接管分为四种情况. 我们实现了这四种失败接管算法.

下面以串行情况下搜索过程中出现错误为例, 给出串行失败接管的算法.

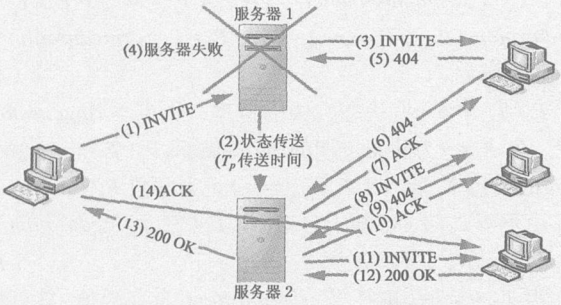

图 3　串行搜索过程中的错误侦测及失败接管的典型情况

图 3中, 如果 3, 8, 11发生错误, 则客户端直接重发 INVITE, 所以它们的错误处理不在此算法范围内. 如果 5和 9发生错误, 将重传 404(如果最终响应为 404), T_{out}时间内未收到 ACK 则错误发生, SSP 计算后将最终响应重发至服务器 2

代码如下:

```
if ( statuscode> 400) and( timeinterval> T_out)
  { //搜索过程中错误发生
SERVERN = SSP( ); //静态和动态的方法
Send( theFinalResponse, SERVERN); //重发此最终响应
  }
if ( statuscode= 200) and( timeinterval> T_out)
  { //搜索到正确的 UAS时错误发生
  SERVERN= SSP( );
  Send( 200OK, SERVERN); //重发 200OK
  }
```

3　实验

3.1　实验床及实验条件

为了评价保留事务状态代理在 UDP协议下状态共享机制的性能, 我们以派生代理在并行搜索下搜寻 UAS成功时为例, 给出图 4所示实验床.

服务器的行为可以抽象成一个 ON /OFF过程模型, 参数 TTF(time-to-failure)和 TTR(time-to-repair)服从指数分布. 它们的期望分别记为 MTTF和 MTTR. UAS振铃时长 ringduration也服从指数分布, 期望为 1/μ

UAC(User Agent Client)在前一呼叫终止后立即产生新的呼叫请求. 由于我们只关注 INVITE事务, 并不关心随后的通话过程和通话拆除, 所以实验中仅仅把通话时长设

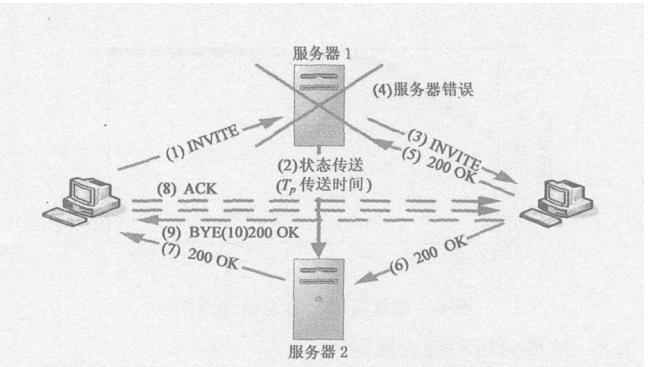

图 4　并行搜索下搜寻 UAS成功时的实验床

为 0为了简化的目的, 图中没有标出 UAS发出的临时呼叫.

如 2.2节所述, SipProxy线程收取网络中 SIP消息, 并将状态放入 MHOrderList中. 在实现中, 考虑到业务量可能很大, 我们采用了分批处理的方法. 即当 SipProxy收集到一定数目的 SIP消息后才进行状态的提取. 然而, 为了评测性能的需要, 我们将每批处理的个数设为最小值 1.

我们仅仅分析了双服务器下的可靠性. 单服务器时, 由于 SIP协议的逐跳处理机制[13]保证了下一跳传递任务的完成, 所以, 当它正常运行时, 会话将保证完成, 可靠性为 100%. 当它发生故障时, SIP消息将不能到达目的地.

双服务器下并行搜索 UAS成功时的实验条件见表 1.

表 1　并行搜索 UAS成功时的实验条件

服务器配置
CPU 主频: 800MHz　内存容量: 4Gbytes　磁盘容量: 2* 72Gbyte
网络适配器速率: 100Mbyte/s
操作系统版本: HPUX11i
SIP栈开发平台: HP OpenCall SIP Stack v121
服务器型号　RP3410　CPU 数: 2
固定系统参数
通话时长: 0s　MTTR: 50s　MTTF/MTBF: 50s
请求每批到达个数:　1
UAS振铃时长期望:　1/μ= 0.5s

3.2　试验结果

大话务量测试后得到服务器间消息传递时间 T_p, 截取其中一段如图 5所示. 计算后的可靠性数据见表 2.将数据绘制成的曲线图如图 6所示.

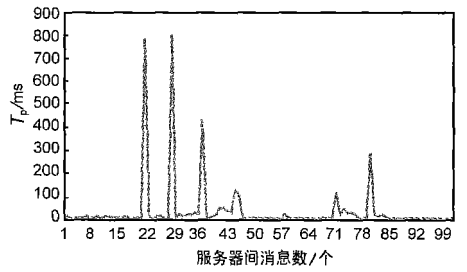

图 5　服务器间消息传递时间曲线图

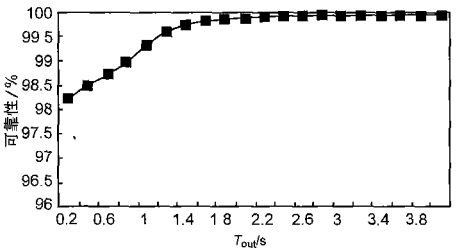

图 6 系统可靠性测量值曲线图

3.3 结果分析和理论推导

我们考察的最重要的性能指标是会话的可靠性. 可靠性被定义为会话成功完成的概率.

3.3.1 T_p 值结果分析

如图 5所示, T_p 值绝大时间位于 100ms以下, 在某些时候将产生峰值. 峰值是正常值的几十倍. 实验中, 81.9% 的 T_p 值小于 100ms, 超出 200ms的峰值只占 6.01%. 这部分 T_p 由于数值过大, 是造成状态误读的主要原因之一.

实验证明, 出现峰值的主要原因为多线程环境下操作系统的调度和各线程对资源的等待 (wait).

服务器 1和服务器 2在消息的处理过程中产生了峰值, 消息包的发送时间稳定在 (5~50) ms中, 均值为 15ms. T_p 由服务器 1的处理时间 D_{proxy1}, 服务器 2的处理时间 D_{proxy2}和消息发送时间 T_{send}叠加而成.

3.3.2 可靠性结果分析

可靠性随着 T_{out}的增加而单调递增, 当 $T_{out}\in[0.2, 1.8]$时可靠性增长很快. $T_{out}\in[1.8, 3.2]$时增长很缓慢, $T_{out}=3.6$时达到常数值 99.935%.

消息包在服务器间传递时, 当传输层为 UDP协议, 会产生包的丢失. 这被称为状态的丢失. 状态误读由状态丢失和状态过期两部分构成. 随着 T_{out}的上升, 状态过期的呼叫迅速减小, 当到达一定点之后, 误读则主要来自于状态的丢失. 当状态过期数为 0时, 可靠性就稳定在一个常数值.

3.3.3 状态过期方程

当 UAS发出最终响应后 T_{out}时间仍没有收到 ACK时, 则触发失败接管, 最终响应将被重传至服务器池中另一可用服务器. 如果在失败接管时, 同时满足如下方程, 则产生事务状态的误读:

$$T_p > T_{out} + \text{ringduration} \tag{1}$$

其中 T_{out}是 UAS发送 200OK和重发 200OK的时间间隔, 它代表了超时时间; ringduration是 UAS收到 INVITE消息和发送 200OK的时间间隔, 它代表了振铃的时长 (图 4中省略了临时响应); T_p 是 SUM 消息包从服务器 1传送到服务器 2的时间间隔.

由不等式 (1)可以得到:

①当 $T_p=0$时, 不存在误读, 有最大可靠性 1.

②当 $T_{out}\downarrow$, ringduration$\downarrow$ 或 $T_p\uparrow$ 时, 误读率上升.

③给定 T_{out}和 ringduration, 当 $T_p\downarrow$ 时, 有较大可靠性. 这说明状态共享机制的实现效率越高, 硬件条件越好, SUM 消息包传送时间就越短, 可靠性就越高.

3.3.4 T_{out}与可靠性 R 之间的关系

T_{out}与可靠性 R 之间的关系可以概括为: $T_{out}\uparrow$, $R\uparrow$; 当 T_{out}到达一定值时, R 为常数. 证明如下:

$$P_r(\text{statenotfound}) = P_r(T_p=\infty) + P_r(T_p > \text{ringduration} + T_{out}, T_p \neq \infty) \tag{2}$$

$$\therefore R = 1 - P_r(\text{statenotfound}) = 1 - P_r(T_p=\infty) - P_r(T_p > \text{ringduration} + T_{out}, T_p \neq \infty) = P - P_r(T_p > \text{ringduration} + T_{out}, T_p \neq \infty) \tag{3}$$

$P_r(T_p=\infty)$为状态丢失的概率, $P_r(T_p > \text{ringduration} + T_{out}, T_p\neq\infty)$为状态过期的概率. 显然, $P_r(T_p=\infty)$就是 UDP 的丢包率. 所以 $1-P_r(T_p=\infty)$记为常数 P.

由不等式 (1)可知: 当 $T_{out}\uparrow$, $P_r(T_p > \text{ringduration} + T_{out}, T_p\neq\infty)\downarrow$, 所以 $R\uparrow$.

当 $T_p\neq\infty$时, 由于 T_p-ringduration 为有限值, 总可以找到一个足够大的 T_{out}, 使 $T_{out} > T_p$-ringduration 不等式总是成立, 即 $P_r(T_p > \text{ringduration} + T_{out}, T_p\neq\infty) = 0$, 代入方程 (3), 则 R = P.

3.3.5 计算 T_{out}的断裂点

R 达到常数值时最初的 T_{out}值被称为 T_{out}的断裂点 (breakpoint).

$$\begin{aligned} R &= P_r(\text{statefound}) = 1 - P_r(T_p > \text{ringduration} + T_{out}) \\ &= P_r(\text{ringduration} > T_p - T_{out}) \\ &= \begin{cases} 1, & T_p \leqslant T_{out} \\ \exp[-(T_p - T_{out})\mu], & T_p \geqslant T_{out} \end{cases} \end{aligned} \tag{4}$$

由式 (4)可知: 当 $T_p = T_{out}$时, R 为最大值; 随着 $T_{out}\downarrow$, 可靠性单调递减. 当 $T_{out}\in[T_p, \infty]$时, 可靠性为常数 P.

根据 T_p 值的组成情况, 我们构造出它在峰值时的表达式:

$$T_p = D_{proxy1} + \frac{a_{message}}{B} + D_{proxy2} = na_1 + a_2 \tag{5}$$

其中, D_{proxy1}表示 INVITE消息在服务器 1的处理时延; 常数 $a_{message}$为 SUM 消息包的比特数; B 为服务器之间设定的链路带宽; D_{proxy2}为 SUM 消息包在服务器 2的处理时间. n 为状态共享机制的线程数, a_1 为峰值时线程调度和等待的平均延时, a_2 为正常情况下的传输延时.

通过改变程序的物理部署, 例如, 将 UAC 和 UAS都放在服务器 1中运行, 来改变方程 (5)中的 n值. 联立两个方程, 就可以解出 $a_1 = 238.6$ms, $a_2 = 88.7$ms. 当 n = 14时,

表 2 系统可靠性测量值

T_{out}(s)	0.2	0.4	0.6	0.8	1.0	1.2	1.4	1.6	1.8	2.0
可靠性(%)	98.2	98.51	98.73	98.98	99.32	99.62	99.79	99.86	99.89	99.9
T_{out}(s)	2.2	2.4	2.6	2.8	3.0	3.2	3.4	3.6	3.8	4.0
可靠性(%)	99.92	99.925	99.925	99.93	99.93	99.93	99.93	99.935	99.935	99.935

可以求出$T_p = T_{out} = 3.43s$

实际中, 我们并不将断裂值作为 UAS的超时参数, 而是取T_{out}为 1.8s这是因为此时系统的可靠性已经达到了接近常数的 99.89%, 而可靠性达到常数值 99.935%, 还要经过 1.8s的时间.

4 结论

我们提出并实现了保留事务状态代理中基于 UDP协议的状态共享机制的整套解决方案. 对原型系统进行了大话务量测试, 给出了可靠性测量值曲线图. 对原型系统进行了数学分析和理论推导, 结果表明, 理论值和实际值相一致.

本文的主要贡献包括: (1)提出并实现了基于多线程的状态共享机制的整套解决方案, 包括: 定义了 SUM 消息包的格式, 给出了错误侦测和失败接管算法, 定义了状态的读取 /更改原则等. (2)对原型系统进行了大话务量实验, 为理论分析提供了真实的数据基础. (3)对理论计算结果和实验结果进行了深入分析, 发现了T_{out}与可靠性 R之间的关系公式, 在理论上推导出T_{out}的断裂点值 3.43秒. 这个理论值与实际值 3.6秒相吻合.

下一步的研究计划: (1)实现基于 TFTP协议的保留事务状态代理, 并进行性能分析, 与 UDP下的共享机制进行性能对比. (2)实际中, 使用断裂值作为 UAS的超时参数是不现实的. 可以通过在读取状态前延时一段时间来降低误读率. 找出特定误读率下延时最小的算法, 是我们下一步需要考虑的问题.

参考文献:

[1] 3GPP. IP multimedia (IM) subsystem-stage 2[S]. 3GPP TS 23.228, 2004

[2] Camarillo G. SIP Demystified[M]. McGraw-Hill Companies, 2002

[3] HELAL A, HEDDAYA A, BHARGAVA B. Replication technique in distributed systems[M]. Kluwer Academic Publishers, 1996

[4] M Bozinovski, L Gavrilovska, R Prasad. Fault-tolerant SIP-based call control system [J]. IEE Electronics Letters, 2003, 39(2): 254- 256.

[5] M Bozinovski, L Gavrilovska, R Prasad A state-sharing mechanism for providing reliable SIP sessions[A]. In TELSIKS 2003 6th International Conference[C]. 2003, 1(1): 384- 387.

[6] M Bozinovski, L Gavrilovska, R Prasad, H-P Schwefel. Evaluation of a fault-tolerant call control system [J]. Facta Universitatis Series: Electronics and Energetics, 2004, 17(1): 33- 44.

[7] Bozinovski M, Renier T, Schwefel H-P, Prasad R. Transaction consistency in replicated SIP call control systems[A]. In Proc. 4th Int. Conf. Information, Communications and Signal Processing, 4th IEEE Pacific-Rim Conf. On Multimedia, (ICICS-PCM)[C]. 2003.

[8] M Bozinovski, H-P Schwefel, R Prasad Algorithm for controlling transaction consistency in SIP session control systems[J]. IEE Electronics Letters, 2004, 40(3): 209- 211.

[9] M Bozinovski, H-P Schwefel, R Prasad Maximum availability server selection policy for session control systems based on 3GPP SIP[A]. In Proc. Seventh International Symposium on Wireless Personal Multimedia Communications[C]. Padova, 2004.

[10] Gray J S. Interprocess Communications in UNIX, Second Edition[M]. Prentice-Hall, Inc, 1998

[11] HP OpenCall SIP协议栈开发文档, SIP stack Application Development Guide, 2003 HP Company. HP OpenCall SIP, SIP stack Application Development Guide[Z]. 2003.

[12] 中国移动通信集团公司, BOSS与 SMP接口规范总体要求 V1.0.0[S]. 2002
China Mobile Corporation. Interface Technical Specification between BOSS and SMP[S]. CMCC, 2002. (in Chinese)

[13] ROSENBER J, et al. SIP: session initiation protocol[S]. IETF RFC 3261, 2002.

[14] COULOURIS G, DOLLIMORE J, KINDBERG T. Distributed systems: concepts and design[M]. Addison Wesley, 2001.3.

[15] BOZINOVSKI M, GAVRILOVSKA L, PRASAD R. Performance evaluation of a SIP-based state-sharing mechanism [A]. In IEEE VTS 56th Vehicular Technology Conf. (VTC) 2002[C]. Vancouver, BC, Canada, 2002.

作者简介:

罗仕漳　男, 1978年出生于江西抚州, 北京邮电大学博士生, 主要研究方向为移动智能网、下一代网络. E-mail: luoshizhang@ ebupt.com

廖建新　男, 1965年出生于四川宜宾, 北京邮电大学教授、博士生导师, 主要研究方向为移动智能网、宽带智能网、下一代网络技术.

朱晓民　男, 1974年出生于浙江义乌, 博士, 北京邮电大学副研究员, 主要研究方向为智能网、下一代业务网络.