ICS 33.040.40
M 33

中华人民共和国国家标准

GB/T 21645.4—2010

自动交换光网络(ASON)技术要求 第4部分:信令技术

Technical requirements for automatically switched optical network—Part 4:Technical specification of signalling

2010-12-01 发布　　2011-04-01 实施

中华人民共和国国家质量监督检验检疫总局
中国国家标准化管理委员会　发布

目　次

前　言

GB/T 21645《自动交换光网络(ASON)技术要求》标准的结构预计如下：

——第1部分：体系结构与总体要求；

——第2部分：术语和定义；

——第3部分：数据通信网(DCN)；

——第4部分：信令技术；

——第5部分：用户—网络接口(UNI)；

——第6部分：管理平面；

——第7部分：自动发现。

本部分是GB/T 21645的第4部分。

本部分与ITU-T G.7713:2006《分布式呼叫和连接管理(DCM)》和ITU-T G.7713.2:2003《基于GMPLS RSVP-TE的DCM信令机制》的一致性程度为非等效。

本部分的以下章节在技术内容上与ITU-T G.7713:2006和ITU-T G.7713.2:2003协调一致：

——第4章对应于G.7713:2006的第5章，并增加了信令协议的选择；

——第5章对应于G.7713:2006的第6章；

——第6章对应于G.7713:2006的第6.2节；

——第7、8、9、10章分别对应于G.7713:2006的第7章、第8章、第9章、第10章；

——第11章对应于G.7713.2:2003的第7、8章以及附录I。

此外，本部分还参考了OIF、IETF等国际标准化组织有关自动交换光网络的建议和草案。

本部分的附录A、附录B和附录C为资料性附录。

本部分由中华人民共和国工业和信息化部提出。

本部分由中国通信标准化协会归口。

本部分起草单位：信息产业部电信研究院、华为技术有限公司、中兴通讯股份有限公司、上海贝尔阿尔卡特股份有限公司。

本部分主要起草人：徐云斌、张海懿、高建华、柯明、许宗幸、王郁、李伟。

自动交换光网络(ASON)技术要求 第4部分:信令技术

1 范围

GB/T 21645的本部分规定了自动交换光网络(ASON)的信令技术要求,包括ASON信令功能要求、分布式呼叫和连接管理(DCM)操作过程、信令控制器的可靠性、DCM属性列表、DCM消息集和状态图、基于通用多协议标签交换的流量工程资源预留协议(GMPLS RSVP-TE)的信令流程等内容。

本部分适用于ITU-T G.803定义的同步数字体系(SDH)传送网络和ITU-T G.872定义的光传送网络(OTN)。

2 规范性引用文件

下列文件中的条款通过GB/T 21645的本部分的引用而成为本部分的条款。凡是注日期的引用文件,其随后所有的修改单(不包括勘误的内容)或修订版均不适用于本部分,然而,鼓励根据本部分达成协议的各方研究是否可使用这些文件的最新版本。凡是不注日期的引用文件,其最新版适用于本部分。

GB/T 21645.1—2008 自动交换光网络(ASON)技术要求 第1部分:体系结构与总体要求

ITU-T G.7713 分布式呼叫和连接管理

ITU-T G.7713.1 基于PNNI的分布式呼叫和连接管理(DCM)信令

ITU-T G.7713.2 采用GMPLS RSVP-TE的DCM信令

ITU-T G.7713.3 采用GMPLS CR-LDP的DCM信令

IETF RFC2205 RSVP功能规范(版本1)

IETF RFC2961 RSVP信令减少刷新开销扩展

IETF RFC3209 RSVP-TE流量工程扩展

3 术语、定义和缩略语

3.1 术语和定义

下列术语和定义适用于GB/T 21645的本部分。

3.1.1

呼叫控制器 call controller

呼叫受呼叫控制器控制,呼叫控制器元件有两种,主叫/被叫方呼叫控制器和网络呼叫控制器。

3.1.2

主叫方/被叫方呼叫控制器 calling/called party call controller

该控制器与呼叫的一终端关联,它可以位于一终端系统,也可以位于其他远端,作为一个代理执行终端系统的功能。该控制器可以承担一个角色或同时承担两个角色:支持主叫方,支持被叫方。

3.1.3

网络呼叫控制器 network call controller

网络呼叫控制器担任两种角色,一种支持主叫方,另一种支持被叫方。主叫方呼叫控制器和被叫方呼叫控制器通过一个或多个中间网络呼叫控制器进行交互。

3.1.4

连接允许控制 connection admission control

连接允许控制用于决定是否有足够的资源来接纳一个连接(或者在呼叫过程中重新协商资源)。

3.1.5

连接控制器 connection controller

连接控制器是 ASON 控制平面中的元件。连接控制器负责协调链路资源管理器、路由控制器以及对等和从属的连接控制器，以管理和监控连接的建立、释放和修改已有连接的参数等操作。

3.1.6

回溯 crank-back

当连接建立请求未成功并从故障点返回建立失败的信息时，回溯机制允许发起新的连接建立请求，尝试重新建立连接以避免资源的阻塞。回溯机制也可以用于连接的恢复机制。

3.1.7

分配 allocate

建立一条 LC 或创建一个子网连接。

3.1.8

撤销 de-allocate

删除一条 LC 连接或释放一个子网连接。

3.1.9

下游 downstream

根据使用的路由模式，下游定义为层次中的下一级组件(层次路由模式)或下一跳组件(源路由或逐跳路由模式)。

3.1.10

信令控制器 signalling controller

信令控制器包含连接控制和(或)呼叫控制的功能。

3.1.11

链路连接建立 LC establish

通过获取或选择一条存在的可用的链路连接来满足连接请求，如从可获得的链路连接列表中获取。

3.1.12

链路连接释放 LC free

将一条链路连接归还到可用的链路连接列表中去。

3.1.13

子网连接创建 SNC created

在子网内部的两个 SNP 之间创建一条连接。

3.1.14

子网连接释放 SNC released

在子网内部的两个 SNP 之间释放一条连接。

3.1.15

上游 upstream

根据使用的路由模式，上游定义为层次中的上层组件(层次路由模式)或前一跳组件(源路由或逐跳路由模式)。

3.2 缩略语

下列缩略语适用于本部分。

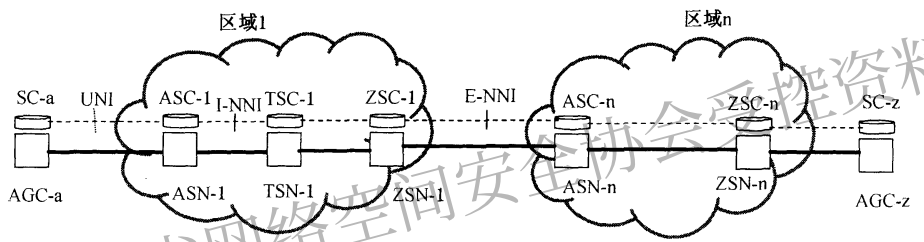
ACC-n	A-end CC at domain n	域 n 中的 A 端连接控制器
AGC	Access Group Container	接入组容器
AGC-a	A-end AGC	A 端接入组容器
AGC-z	Z-end AGC	Z 端接入组容器

ARC-n	Alarm Reporting Control	在域 n 中的告警上报控制
ASC-n	A-end signalling controller in domain n	在域 n 中的 A 端信令控制器
ASN-n	A-end SN in domain n	在域 n 中的 A 端子网
ASON	Automatically Switched Optical Network	自动交换光网络
CallC	Call Controller	呼叫控制器
CAC	Call Admission Control	呼叫许可控制器
CC	Connection Controller	连接控制器
CC-a	A-end Connection Controller	A 端连接控制器
CC-z	Z-end Connection Controller	Z 端连接控制器
CCC	Calling/Called Party Call Controller	主叫方或被叫方呼叫控制器
CCC-a	A-end CCC	A 端主叫方呼叫控制器
CCC-z	Z-end CCC	Z 端呼叫控制器
CR-LDP	Constraint-based Routed Label Distribution Protocol	基于约束路由的标签分发协议
CoS	Class of Service	服务分类
DCM	Distributed Call and Connection Management	分布式呼叫和连接管理
DTL	Designated Transit List	指定传送列表
E-NNI	Exterior NNI	外部网络—网络接口
GoS	Grade of Service	服务等级
GMPLS	Generalized Multi-Protocol Label Switching	通用多协议标签交换
I-NNI	Interior NNI	内部网络—网络接口
LC	Link Connection	链路连接
LCI	Link Connection Identifier	链路连接标识符
LRM	Link Resource Manager	链路资源管理器
MI	Management Information	管理信息
MPLS	Multi-Protocol Label Switching	多协议标签交换
NCC	Network Call Controller	网络呼叫控制器
NCC-n	NCC in domain n	在域 n 中的网络呼叫控制器
NNI	Network Node Interface	网络—网络接口
PC	Protocol Controller	协议控制器
PNNI	Private Network-Network Interface	私有网络—网络接口
RC	Route Controller	路由控制器
RSVP-TE	Resource Reservation Protocol-Traffic Engineering	流量工程扩展的资源预留协议
SC	Switched Connection	交换连接
SC-a	A-end user signalling controller	A 端用户信令控制器
SC-z	Z-end user signalling controller	Z 端用户信令控制器
SPC	Soft permanent connection	软永久连接
SN	SubNetwork	子网
SNC	SubNetwork Connection	子网连接
SNP	Sub-network Point	子网点
SNPP	Sub-network Point Pool	子网点池
SNCr	SubNetwork Controller	子网控制器
TCC-n	Transit CC in domain n	域 n 中的中间节点连接控制器
TCP	Termination Connection Point	终端连接点

TDM	Time Division Multiplex	时分复用
TNA	Transport Network Assigned Adress	传送网络分配地址
TSN-n	Transit SN in domain n	在域 n 中的中间传送子网
TSC-n	Transit signalling controller in domain n	域 n 中的中间节点信令控制器
TTL	Time To Live	生存时间
ZCC-n	Z-end CC at domain n	域 n 中的 Z 端连接控制器
ZSN-n	Z-end SN in domain n	域 n 中的 Z 端子网
ZSC-n	Z-end signalling controller at domain n	域 n 中的 Z 端信令控制器
UNI	User Network Interface	用户网络接口

4 分布式呼叫和连接管理元件定义

在分布式呼叫和连接管理环境下，根据不同的代理在信令流中所处的不同位置，为其分配一定的角色。图 1 定义了这些参考点。



AGC-a	A端接入组容器
AGC-z	Z端接入组容器
ASN-1	区域1A端子网络
TSN-1	区域1中间子网络
ZSN-1	区域1中Z 端子网络
ASN-n	区域n中A 端子网络
ZSN-n	区域n中Z端子网络

SC-a	A端用户信令控制器(CCC和CC)
SC-z	Z端用户信令控制器(CCC和CC)
ASC-1	区域1A端信令控制器(NCC and CC)
TSC-1	区域1中间信令控制器1(CC)
ZSC-1	区域n中的Z端信令控制器(NCC和CC)
ASC-n	区域n中的A端信令控制器(NCC和CC)
ZAC-n	区域n中的Z端信令控制器(NCC和CC)

图 1 分布式呼叫和连接管理参考配置图

图 1 中的传送平面元件为多种子网以及接入组容器。它们定义了和控制平面功能相关的具体位置。如图 1 中标明的 AGC-a、ASN-1、TSN-1、ZSN-1、ASN-n、ZSN-n 和 AGC-z。

分布式呼叫和连接管理也称为信令。终端用户与呼叫相关的功能称为主叫或被叫方呼叫控制器，即 CCC。源端 CCC 称为 CCC-a，目的端 CCC 称为 CCC-z。和子网相关的呼叫控制器称为网络呼叫控制器 NCC，在特定域 n 中称为 NCC-n。

终端用户的连接控制器表示为 CC-a 和 CC-z。在域 n 中，A 端、中间以及 Z 端连接控制器表示为 ACC-n、TCC-n 和 ZCC-n。

信令控制器包括连接控制和(或)呼叫控制功能。终端用户可以通过 SC-a 和 SC-z 来表示。域 n 中 A 端、中间以及 Z 端信令控制器表示为 ASC-n、TSC-n 和 ZSC-n。如图 1 所示，TSC 通常没有呼叫控制功能。

每个信令控制器分配一个信令地址，协议控制器使用信令地址在呼叫或连接控制器之间交换信息。信令控制器的地址为控制地址，信令通道通过两个相邻的信令控制器名字来标识。信令控制通道由信令通信网(DCN)提供。

5 分布式呼叫和连接管理操作过程

5.1 分布式呼叫和连接管理需求

在呼叫建立之前，服务提供方和服务请求方之间要建立合约，合约中可以指定如下内容：

——合约 ID；

——服务等级协议以及服务等级规范；

——允许请求进行策略控制所需要的信息。如用来提供鉴权和一致性验证相关的信息。

ASON 网络中呼叫请求的建立性能与不同的参数有关。影响信令性能的参数包括：

——用来传送信令消息的数据通信网络的带宽容量；

——传送网的规模(通常以节点或链路数目来衡量)；

——每个时间周期内总的呼叫请求次数，包括新的呼叫请求次数、保护事件次数、恢复事件次数等；

——消息包的平均尺寸；

——混和的连接类型数；

——完成呼叫请求的时间；

——网络接收到的请求后，因为连接建立不成功需要重新尝试连接建立而发出的请求占总请求数的比例；

——为实现可靠的消息传送机制所增加的额外带宽的要求(如超时重传)；

——同步和异步消息传送。

为了满足信令基本需求，以及将来的扩展能力，DCM 机制必须具备基本能力和可扩展能力，以满足多个应用需求。基本能力要提供建立和释放连接的必要机制。

呼叫和连接管理操作发生在控制区域内部，在此范围内可以实现完整的呼叫和连接动作。呼叫和连接操作在一个区域内部执行时，此区域可以包括更小的区域。这种划分关系同路由区域的层次划分相对应。路由区域边界的信令元件使用此区域内部的路由元件，获取穿越此区域的路由。

5.2 呼叫和连接管理功能

呼叫控制器(CallC)、连接控制器(CC)、以及链路资源管理器(LRM)能够完成对呼叫和连接请求的监控和管理功能，包括连接建立、连接修改和连接释放等基本操作。为完成一个具体的操作，CallC、CC 和 LRM 通过和下面的组件交互来建立或删除一条连接：

a) 路由控制器；

b) 呼叫允许控制(CAC)功能；

c) 呼叫控制器(CallC)；

d) 连接控制器(CC)；

e) 链路资源管理器(LRM)。

主叫方呼叫控制器和被叫方呼叫控制器通过一个或多个中间网络呼叫控制器(NCC)进行交互。NCC 功能由网络边缘节点提供(如 UNI 参考点)，也可以由域间的网关来提供(如 E-NNI 参考点)。网络边缘 NCC 执行的功能由用户和网络之间相关的策略来定义，域边界的 NCC 执行的功能由域间相关的策略来定义。当呼叫跨过多个域时，一个端到端的呼叫由多个呼叫段组成。每个呼叫段可以对应一条或多条连接(LC 或 SNC)。这样就可以在不同的域中灵活地选取信令、保护和恢复机制。

同样一个端到端的呼叫，不同的呼叫分段对应的连接数目可以不同。在图 2 中，UNI 呼叫分段对应一个链路连接(LC)，控制域 1 的子网内的呼叫分段对应 2 条连接，网络在不同的域中拥有不同的策略。图 2 中的所有传送资源都在一个完整的区域中，此区域包含区域 1 和区域 n。为了实现两个客户之间的呼叫建立，在整个区域内部必须提供路由功能，同时能够获得穿越区域 1 和区域 n 的路由。

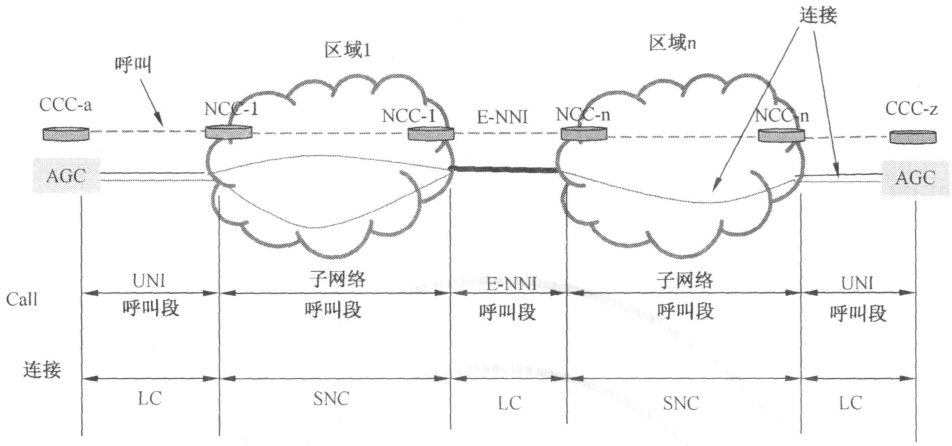


图 2　呼叫分段和连接

呼叫和连接都可以跨经运营商内部的 E-NNI 参考点。呼叫和连接分离以及呼叫段可以满足下面的应用：

a）　基于域的保护。子网连接在网络内部具有保护属性，在不同的域中子网连接(SNC)数目可以不同；

b）　基于域的恢复。子网连接在网络内部具有重路由功能，子网连接 SNC 故障不会引发链路连接 LC 失效，在故障的子网中可以通过重路由来完成对子网内故障 SNC 连接的恢复。

域边界的 NCC 也允许每个域拥有独立的功能。例如，一个域中能提供 1+1 保护能力，而另外一个域中没有这种保护能力。

NCC 和 CC 在网边缘和边界执行不同的功能。

呼叫控制器执行如下的功能：

a）　NCC 将一个 SNC 连接关联到一个呼叫；

b）　和网络边缘的主叫或被叫方呼叫控制器工作的 NCC 将一个 LC 关联到一个呼叫；

c）　和域边界的对等 NCC 工作的 NCC 将一个 LC 关联到一个呼叫；

d）　NCC 将 LC 和 SNC 关联到一个同一个呼叫；

e）　CC 建立和每个呼叫分段对应的连接。

由于控制器之间的通信作为一个外部接口来定义，本部分中所定义的消息用于完成信息的交换。端到端呼叫建立涉及呼叫请求、连接请求以及使用不同类型的资源来创建连接。图 3 中描述了为支持呼叫而完成的连接建立的过程。

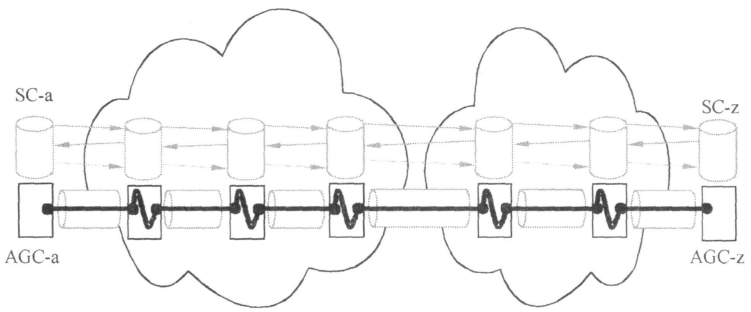


图 3　呼叫请求相关的 LC 和 SNC 连接的建立过程

呼叫的建立要使用下面的资源：

——子网点(SNP)；

——SNP 池(SNPP)；

——链路连接(LC)。

通过分配 SNP 来完成 LC 连接的建立,这需要在 LRM 之间进行协商。然后才允许 CC 创建一条 SNC 连接。对 SNP 的分配可以看成是 SNP 状态的改变(如从可用变为被占用)。当 SNP 的状态为潜在的(Potential)或忙(Busy)时,它不能够在连接建立时使用。

图 4 中描述了 LRM 建立 LC 的过程。

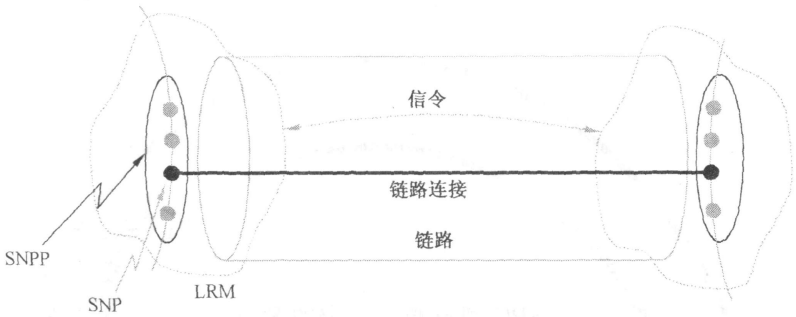


图 4　通过分配 SNP 来建立 LC 连接的过程

当建立一条 LC 连接时,两端需要进行协商。如对于用户—网络信令,用户可以指定一条 LC,而网络可以选择另外一条 LC 来使用。

在建立一条 SNC 连接之前,SNP 必须已经存在,并且通过 LRM 标识绑定这些 SNP 来创建一条 SNC 连接。LRM 要和上游 LRM 以及下游 LRM 协商所用到的 SNP(可以代表一条 LC 连接)。这些输入和输出的 LC 连接以及同它们相关联的输入和输出 SNP 可以用来创建 SNC 连接。

为一条连接选择资源时并不意味着要分配这些资源。资源分配可以发生在信令过程的任何阶段。例如,可以在发起请求或响应请求的期间来分配资源。另外,在资源被分配之前可以先预约这些资源。在呼叫建立前后过程中,通过预约标识出这些资源可以被连接所用,但只有对资源的分配动作完成后连接才能真正使用这些资源。使用预约可以防止其他请求来标识相同的资源。所有这些资源预约和资源分配动作实际上是通过设置 SNP 的状态并和 LRM 组件交互来完成。

SNC 连接的建立是由子网内部处理完成,并由 CC 控制。当确定了入口和出口连接点的 SNP 后才创建一条 SNC 连接。入口和出口 SNP 的标识作为建立 LC 连接的一部分(通过 LRM 来完成)。如图 5 中描述,SNC 连接的建立通过由相关的连接控制器以及用来建立 SNC 连接的 SNP 对来完成。

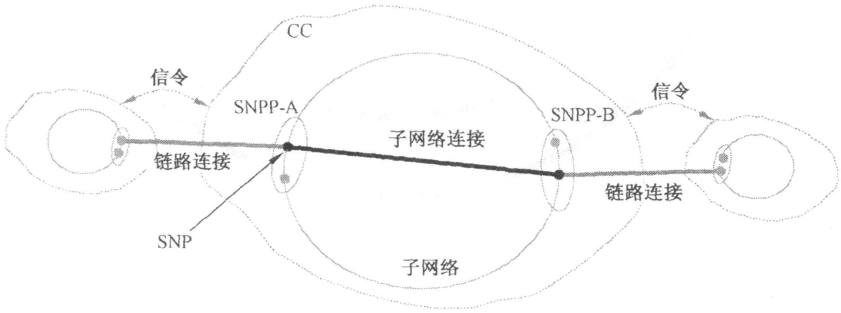


图 5　LC 连接建立后创建 SNC 连接过程

5.3　呼叫处理过程

5.3.1　呼叫请求处理

对交换连接(SC),呼叫请求是由 A 端用户请求代理(CCC-a)向网络呼叫控制器发送“呼叫建立请求”消息来发起。呼叫请求消息中指定与用户请求的呼叫相关的信息。呼叫请求相关的信息可以包含在与服务相关的信息以及与策略相关的信息中。请求信息被 ASC 内部的 CallC 接收到后,CallC 处理

呼叫请求并和 ASC 内部的其他组件交互来完成呼叫请求。

对软永久连接服务(SPC),客户呼叫控制器由管理平面处理,向 NCC 发送管理请求建立呼叫。此呼叫终端点的传送平面位置为 SPC 终端点,采用 UNI 传送资源标识符同这些资源关联。SNPP 具有公共的传送网络地址,SPC 终端点并不需要 UNI 传送资源标识符。

为处理出错情况并预防异常信令传送情况发生,需要超时机制。超时机制由呼叫请求的上游用户发起。具体的超时相关的异常处理见 5.5。

5.3.2 SPC 和 SC 的互操作

同 NCC 关联的 SPC 端点支持 SPC 业务的发起和终止。同样,同 NCC 关联的 SC 端点支持 SC 业务的发起和终止。每一种类型的端点都使用 UNI 传送资源标识符,用于标识在客户和网络之间 UNI 参考点的传送资源。

SPC 端点可以向 SC 端点发送呼叫,并使用此 SC 端点的 UNI 传送资源标识符。同样,SC 端点也可向 SPC 端点发送呼叫,使用此 SPC 端点的 UNI 传送资源标识符。SPC 或 SC 端点都可以释放呼叫。

在呼叫和被叫的情况下,都可以为 SPC 端点指定特定的 SNP。如果没有指定,同 SPC 端点相关的 CC 是空闲的,可以分配 SNP。

5.3.3 呼叫建立

图 6 描述了呼叫的建立以及对应组件之间的相关信令流程。

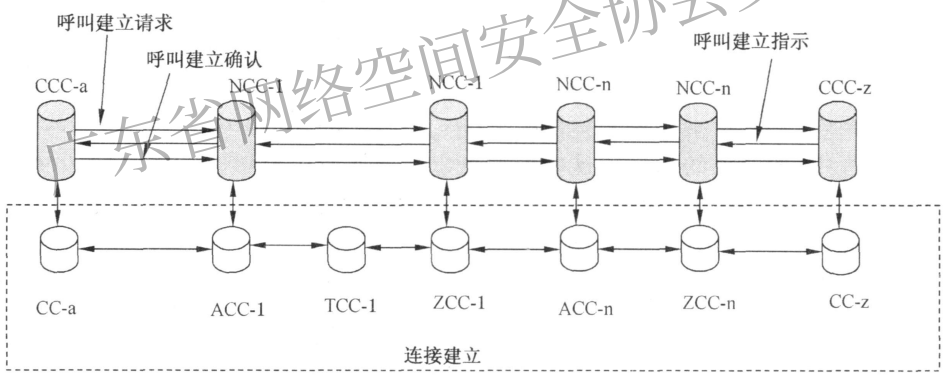


图 6 呼叫建立请求过程

呼叫建立步骤如下:

——主叫方呼叫控制器(CCC-a)请求呼叫建立。入口的 NCC-1 验证呼叫请求的有效性,包括授权和一致性验证以及验证是否满足策略相关的约束条件。随后请求信息发送到中间网络呼叫控制器。出口 NCC 验证呼叫请求是否能够接受。

——当验证成功后,CCC-a 通过向 CC 发起连接建立请求来继续进行呼叫。连接建立请求的处理描述见 5.4.2。对不同的协议,连接建立请求的发起可以和图 6 中所描述的顺序不一样。在图 6 中网络连接的建立在呼叫完成之前完成。

——当连接建立请求处理成功完成后,整个呼叫建立请求成功完成,并可以开始传送用户业务信号。

如果连接建立请求处理不成功,要向用户发送呼叫被拒绝的通知消息。

5.3.4 呼叫释放

图 7 描述了呼叫释放以及相关组件之间的信令流。

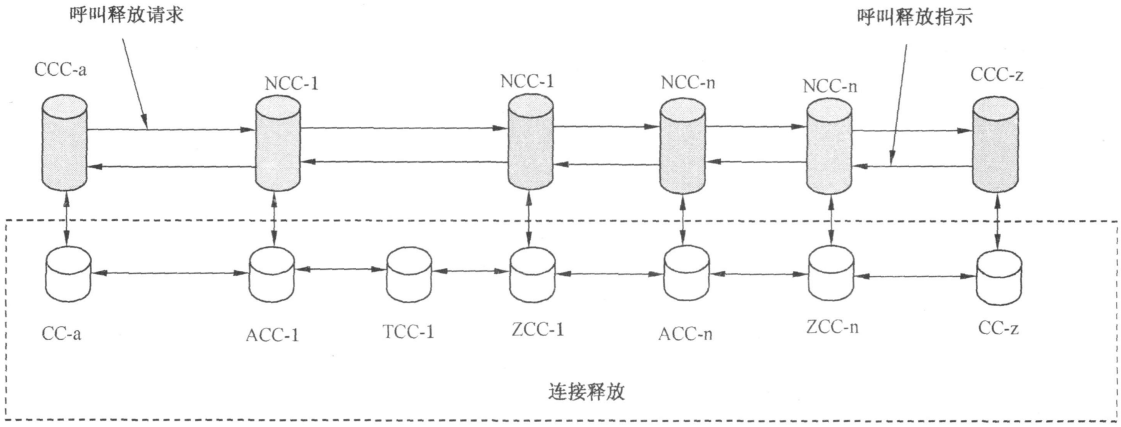


图 7　呼叫释放请求处理过程

任何呼叫控制器都可以发起呼叫释放请求。已通过验证的呼叫释放请求必须总能够成功释放呼叫。在呼叫释放过程中的任何故障情况都要上报给管理系统，包括报告部分连接没有释放的信息，后续的处理主要防止对没有被成功释放连接的再次使用。从主叫方控制器发起释放请求的描述见图 7：

——在入口 NCC-1 验证呼叫释放请求，包括授权和一致性验证以及验证是否满足策略相关的约束条件。

——当验证成功后，随后发起连接释放请求。连接释放请求的处理过程见 5.4.3。对不同的协议，连接释放请求的发起可以和图 7 中所描述的顺序不一样。图 7 中描述的情况是网络连接的释放在呼叫释放完成之前完成。如果呼叫对应多个连接，将会释放所有对应的连接。

——当收到连接释放成功完成的通告消息后，呼叫释放成功完成。

连接释放被拒绝（由于不能释放资源、释放 SNC 连接或释放 LC 连接）会产生对管理平面（MP）的通知。呼叫释放指示应该能够指示成功释放一个呼叫。上面情况是假设在连接释放发起之前呼叫释放请求就已经验证成功。

根据传送网络的特征，呼叫释放请求和连接释放请求之间可能会发生竞争。在 CCC-a 到 CCC-z 之间的信令处理以及 AGC-a 到 AGC-z 之间的信号传送处理这种竞争条件下，下游子网可能会引发某些告警。为支持上述情况，需要提供监控和踪迹能力，这种和呼叫相关的监控踪迹功能在连接释放之前就应该打开。

5.4　连接处理过程

5.4.1　连接请求处理流程

呼叫请求的处理会导致发起连接请求。连接请求执行相关的连接建立和连接释放操作，并为对应的连接分配或释放资源。

图 8 描述了信令和连接建立请求的端到端处理过程，其最终结果是为连接分配资源并完成呼叫。

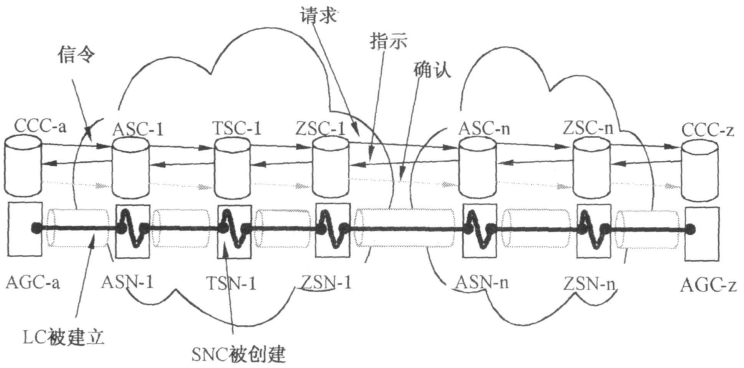


图 8　连接建立请求时的 LC 连接建立和 SNC 连接创建

图 9 描述了信令和连接释放请求的端到端处理过程，其最终结果是释放连接资源并完成呼叫的释放。连接释放的过程可以有不同的顺序。如按 SNC-LC-SNC 的顺序来完成，或先释放所有的 LC 连接再释放所有的 SNC 连接。

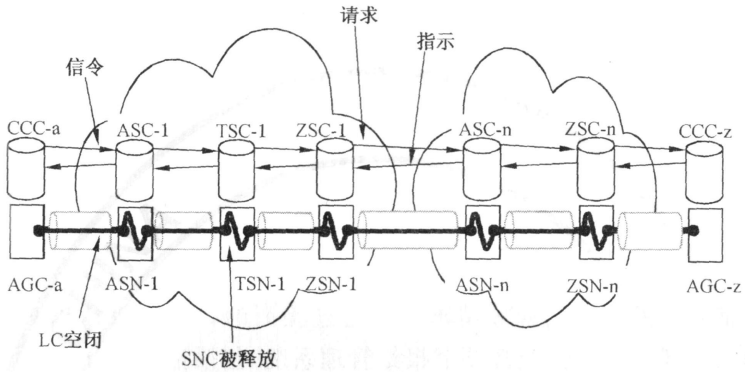


图 9 连接释放请求时的 LC 连接释放和 SNC 连接释放

5.4.2 连接建立处理

连接建立需要执行如下的处理：

——根据 5.3.3，呼叫建立请求通过验证后才允许处理连接建立请求。

——根据呼叫请求，CCC-a 的 LRM 指定 LC 链路，LC 链路是通过在 AGC-a 和 ASN-1 之间协商后建立的。这样就可以形成 AGC-a 的出口 SNP ID。

——在 ASC-1 的 CC，入口 SNP 通过入口 SNP ID 来标识，入口 SNP ID 通过映射 AGC-a 的出口 SNP 到 ASN-1 的入口 SNP 来获得。ASC-1 的 LRM 通过和 TSC-1 的 LRM 协商来建立 ASN-1 到 TSN-1 之间的 LC 连接(TSN-1 由 RC 提供的路由信息或由从上游 CC 接收到的信息来决定)。当 LC 连接建立成功后，出口 SNP 被 LRM 标识出来。SNC 连接的创建用来连接入口 SNP 和出口 SNP，然后此 SNP 对的状态更新为“占用”。连接控制器 CC 通过和下游 CC 的通信继续连接建立的处理过程。

——在 TSC-1 的 CC，入口 SNP 通过入口 SNP ID 来标识。TSC-1 的 LRM 和 ZSC-1 的 LRM 的协商来建立 TSN-1 到 ZSN-1 的 LC 连接(TSN-1 由 RC 提供的路由信息或由从上游 CC 接收到的信息来决定)。当 LC 连接建立成功后，出口 SNP 被 LRM 标识出来。SNC 连接的创建用来连接入口 SNP 和出口 SNP，然后此 SNP 对的状态更新为“占用”。连接控制器 CC 通过和下游 CC 的通信继续连接建立的处理过程。

——连接请求的处理过程一进行到 CCC-z。

——在 CCC-z 的连接控制器 CC，入口 SNP 通过入口 SNP ID 来标识。当 CC 处理连接请求后，发送响应消息来指示连接请求已经被处理。

——当 CCC-a 的 CC 收到指示信号后，CC 向下游节点发送连接确认消息，这为可选项。

如果路由不能建立，子网使用连接拒绝消息来响应连接请求。

应该有能力指定双向连接，在两个方向上 SNP 索引值可以是相同的。例如，在一个传送网络单元上的一个端口上在两个方向采用相同的时隙编号。

5.4.3 连接释放处理

连接释放的处理和连接建立的处理相反。呼叫释放请求先进行处理。连接释放时执行下面的处理：

——根据5.3.4,呼叫释放请求验证通过后才允许进一步处理连接释放请求。当收到连接释放的指示后,ASC-1的CC发起连接释放处理。

——从呼叫请求中,发起释放呼叫请求的代理标识呼叫即将释放。

——在ASC-1的连接控制器CC,SNC被释放,包括释放SNP。ASC-1的LRM通过信令消息通知TSC-1的LRM来释放TSN-1使用的LC链路连接。SNP对的状态更新为"可用"。CC和下游的CC通过信令通信继续进行连接释放处理。

——在TSC-1的CC,SNC被释放。TSC-1的LRM通过和ZSC-1的LRM进行信令交互来释放ZSN-1使用的LC。SNP对的状态更新为"可用"。CC和下游的CC通过信令通信继续进行连接释放处理。

——连接释放请求的处理一直延续到目的CCC-z的CC。

——在目的CCC-z的CC,呼叫使用的链路连接LC被释放。当CC处理完连接释放请求后,向上游节点发送连接释放处理完成的指示信息。

5.5 恢复信令流程

5.5.1 连接恢复处理机制

呼叫的恢复会引起连接的恢复,由连接的重路由实现并使用新的网络资源。此动作在重路由区域内实现,并且由每一个呼叫的策略决定。重路由操作主要在两个呼叫控制器之间建立两个分离的连接。一旦新的重路由连接建立,此连接就会替代现存的连接,这两个连接属于相同的呼叫段。注意,呼叫在连接重路由的过程中保持不变。

网络中存在两种类型的恢复,软重路由和硬重路由。软重路由服务是用于管理维护目的的呼叫重路由机制(如路由优化、网络维护、工程规划等)。当软重路由操作被触发(一般经由管理平面发起的请求),并发送给重路由元件,重路由元件首先建立一个重路由连接,然后使用此连接并将最初的连接删除。这称作先建后拆(make-before-break)。

硬重路由机制提供了呼叫连接失效时的恢复机制,并能响应失效事件。对于一个已激活硬重路由的呼叫连接,源节点阻止呼叫释放,并试图在重路由域边界建立一个到宿节点的替代连接段,即重路由连接。在重路由域边界的宿节点同样阻止呼叫连接的释放,并等待重路由域边界的源节点建立一个重路由连接。对于硬重路由,初始的连接段可以在重路由连接建立之前或者之后被释放,即采用"先拆后建"或"先建后拆"的方式,对于采用返回方式的硬重路由机制,可以采用"只建不拆"方式。

软重路由和硬重路由机制可以用于交换连接和软永久连接。在网络呼叫控制器之间应用,UNI参考点没有此功能。

两个呼叫控制器之间的重路由操作可能同一个呼叫段相关,也可以在几个连续的呼叫段的之间发生,在这种情况下,呼叫段可能会因为重路由连接路由的改变而发生变化,但是呼叫参数以及呼叫名称不会发生变化。

5.5.2 单个重路由域的硬重路由

如果一个呼叫在一个区域内部配置了硬重路由服务,连接的故障导致了到达区域边界NCC的信令故障。此故障不会扩散到区域外部,并且区域内会发起恢复操作。例如在图8中,区域1的连接发生了故障,硬重路由操作过程如图10所示。当故障通知到达ASC-1后,呼叫没有释放,但是新的呼叫和连接请求向ZSC-1发起。注意ASC-1和ZSC-1都包含呼叫和连接控制器。当域1的新的连接建立以后,呼叫就可以使用它。ASC-1和ZSC-1之间的呼叫段在重路由前后保持不变。

如果硬重路由操作是不可返回的,ASC-1发起故障连接的释放。此操作可以在新的连接建立之前或之后进行。在图10中,释放操作在连接建立完成以后执行。

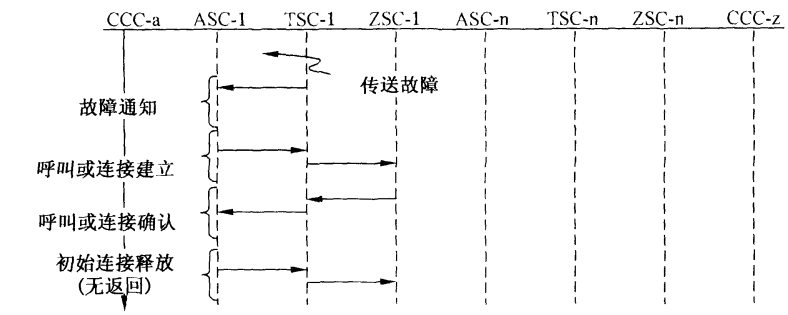


图 10　单个域的硬重路由

5.5.3　多个重路由域的硬重路由

当呼叫穿越多个域时，这些域又包含在一个更大的域中，此域能够实现在它包含的域中的路由。这是一种层次路由的概念，硬重路由可以由最高层次的域来完成，重路由操作可以在包含的域中实施。如果释放故障的连接(非返回式)，并且新的连接调用新的呼叫控制器，使用新的呼叫段，那么必须释放故障连接中间呼叫控制器中的呼叫状态。

在图 11 中存在 3 个域，其中 2 个域通过 E-NNI 链路互联(域 1 和域 2)，它们都属于一个更大的重路由器域——域 0。图中显示了一个呼叫通过 E-NNI-1 传输，假设它的连接经过 AGC-a、ASN-1、TSN-1、ZSN-1、ASN-2、ZSN-2 和 AGC-z。如果在 E-NNI-1 的链路上发生了故障，那么就会通知 ASC-1 进行重路由操作，此操作由域 0 来执行。注意 ZSC-1 不能实现重路由操作，因为它和域 2 之间只有一条链路连接。同样，域 1 也不能执行重路由操作，因为 E-NNI-2 不属于它管辖的范围。

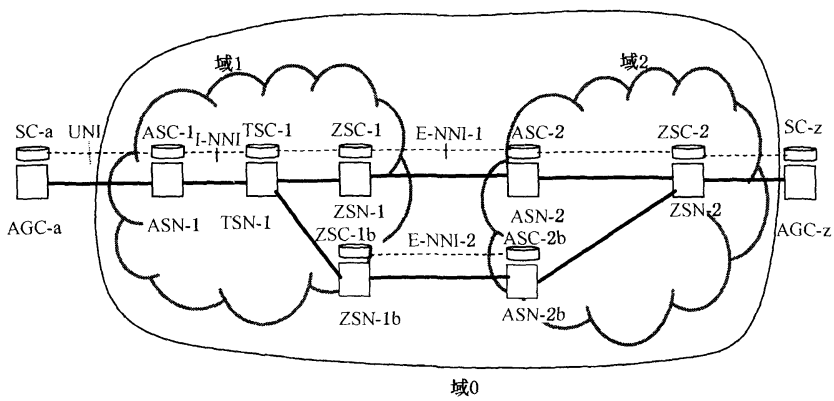


图 11　多个域的硬重路由

在 ASC-1，呼叫控制器配置为执行硬重路由操作，而且业务不返回。在域 0，为新的连接选择路径为域 1 经由 E-NNI-2 到达域 2。为了到达 E-NNI-2，在域 1 进行重路由操作，同样在域 2 执行从 E-NNI-2 到达 ZSC-2 的重路由操作，那么就建立一条通过 ASN-1、TSN-1、ZSN-1b、ASN-2b 和 ZSN-2 的路径。为新的连接建立了三个呼叫段，ASC-1 到 ZSC-1b、ZSC-1b 到 ASC-2b 和 ASC-2b 到 ZSC-2。重路由过程如图 12。

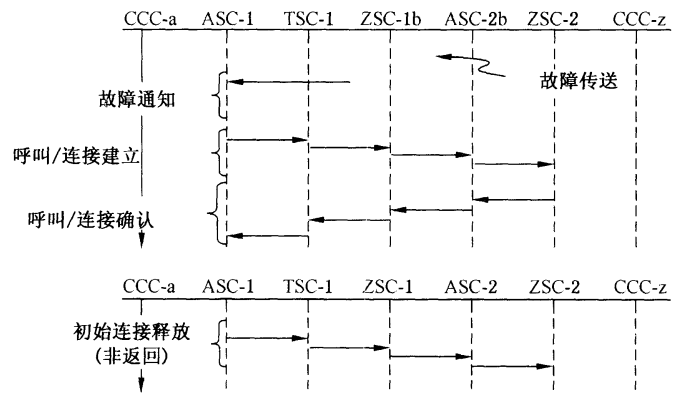


图 12　多个域的硬重路由流程

如果操作是不可返回的，那么 ASC-1 的呼叫控制器会发起故障连接的释放操作，相应的呼叫段不再使用，并在 ZSC-1 和 ASC-2 移除呼叫状态。此操作可以在新的连接建立以前进行，也可以在之后进行。在图 11 中，此操作在新的连接建立以后实现。

5.6　信令异常处理流程

5.6.1　信令异常处理机制

网络中会发生不同等级的异常事件，异常事件会同时影响传送平面和控制平面。异常事件通常包括：信令通信网络故障、连接控制器故障、连接控制器行为异常。

——通信链路中断会导致信令通信网故障；

——构成控制器的不同代理的故障会导致连接控制器失效，如连接建立代理故障；

——消息的解析出错会导致连接控制器行为异常。

故障信息在呼叫控制器(CallC)和管理平面之间以及 CC 和管理平面之间传递，故障信息中包括详细的故障原因。图 6 描述了 CallC 呼叫请求处理的示意图。使用这种网络模型时，下面的章节中提供了不同的情形。如图 13～图 27 所示。还描述了建立和释放呼叫的流程以及处理流程中的不同故障情形。

5.6.2　连接建立异常处理

5.6.2.1　CCC-a UNI 故障(请求消息)

下面的故障导致信令动作：

——无法识别的信令信息；

——RC 故障(如查找到 AGC-z 的路由时发生故障)；

——CAC 故障(如验证策略信息时发生故障)；

——AGC-a 和 ANSN-1 之间的链路连接故障；

——ASN-1 内部子网连接的故障；

——LRM 故障(如将所请求的带宽映射到传送网络资源时发生故障)；

——ASC-1 的 CC 定时机制超时；

——CCC-a 的 CC 定时机制超时。

图 13 和图 14 显示了两个建立拒绝的示例。图 13 中，ASC-1 中的故障导致拒绝呼叫建立请求。图 14 显示了在收到响应之前定时器已经超时的故障情况。在这种情况下，用户撤销请求。为清除所有的状态，并防止网络在随后的一段时间内建立呼叫请求(因为请求同步出错)，使用请求删除命令来删除上一次的建立请求。

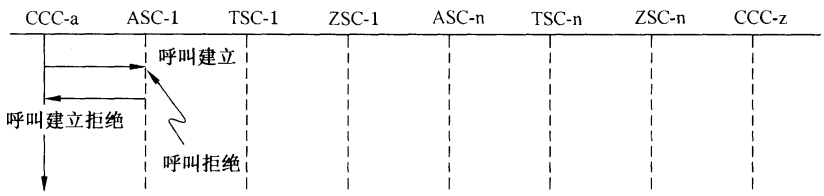


图 13　建立—CCC-a UNI 故障(呼叫被拒绝)

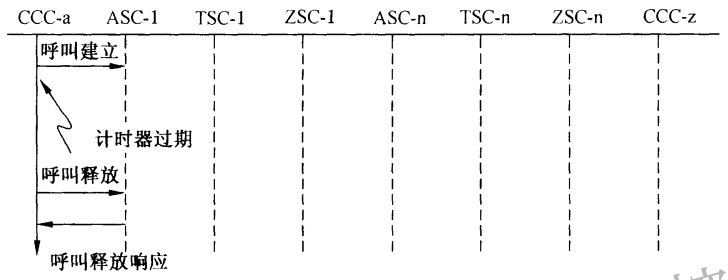


图 14　建立—CCC-a UNI 故障(定时器超时)

5.6.2.2　**CCC-a UNI 故障(响应消息)**

下面的原因代码导致信令动作：

——响应消息没有到达用户请求代理；

——CC 对所建立的连接没有应答；

图 15～图 18 描述了基于故障响应的信令流程。第一个示例中，从 CCC-a 来看和超时处理流程相似，在这种情况下，下游的连接已经建立，如资源已经预留或已经分配。根据上面所描述的情况，用户的释放请求随后会解除预留或释放与连接相关的任何资源。

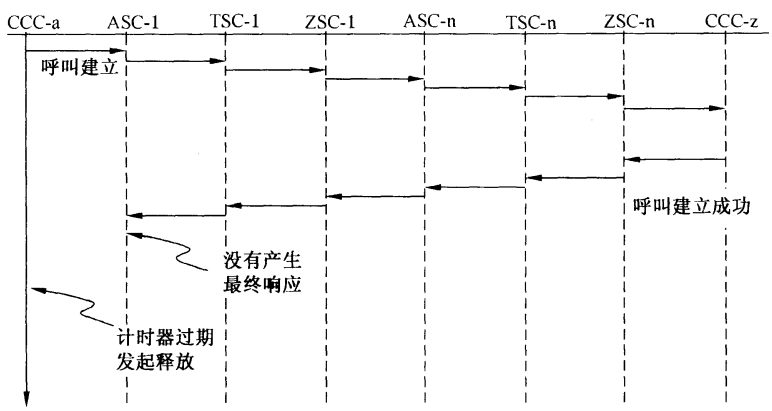


图 15　建立—信令响应消息故障(没有产生最终的响应消息)

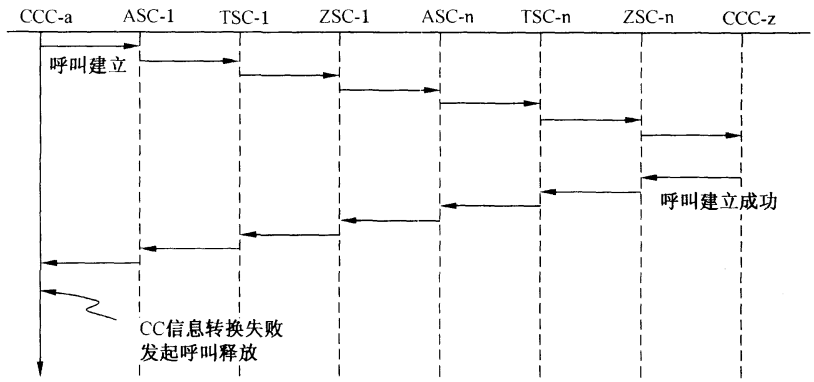


图 16　建立—信令响应消息故障(CCC-a 信息转换失败)

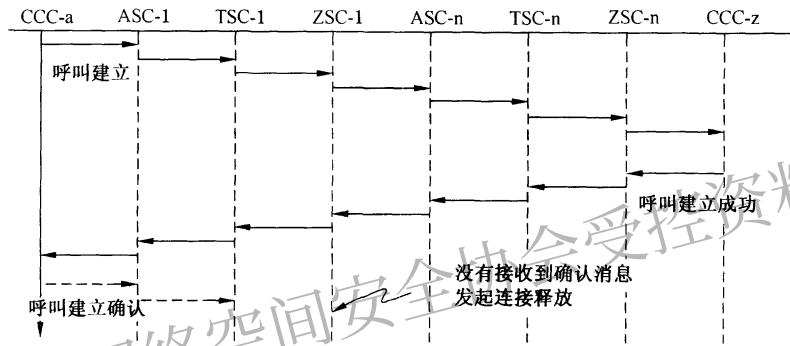


图 17　建立—信令确认消息故障(传送过程中确认消息丢失)

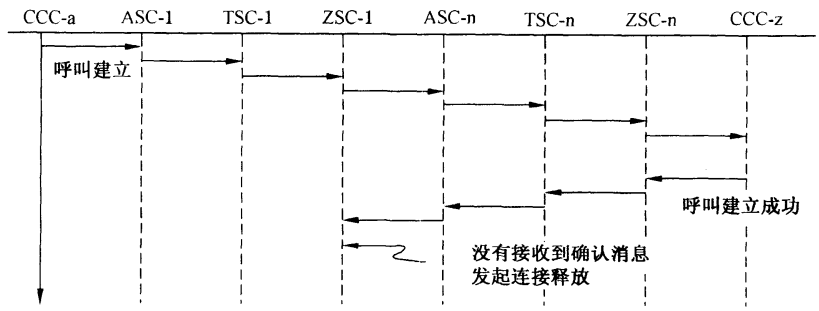


图 18　建立—信令确认消息故障(确认消息一直没有产生)

5.6.2.3　域内和域间的故障

下面的故障原因编码会导致信令动作:

——RC 故障,如查找到 AGC-z 的路由时发生故障;

——CAC 故障,如验证策略信息时发生故障;

——子网或域之间的链路连接故障,如 ASN-1 和 TSN-1 之间连接故障或 ZSN-1 和 ASN-n 之间的连接故障;

——子网内部的子网连接的故障,如 TSN-1 子网连接故障;

——LRM 故障,如将所请求的带宽映射到传送网络资源时发生故障;

——CC 定时器超时。

在图 19 和图 20 的例子中，策略的验证已经通过，但网络资源不足导致请求被拒绝。服务被拒绝的结果就是子网会解除预留或释放已经预留或分配的网络资源。

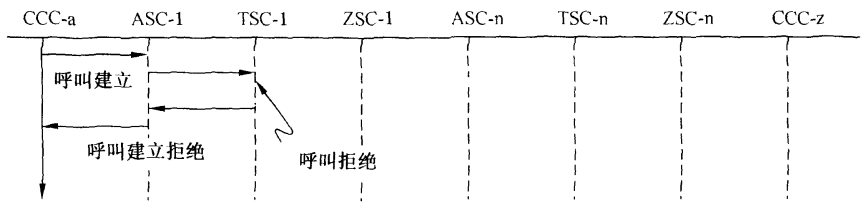


图 19　信令流：建立—域内故障（呼叫被拒绝）

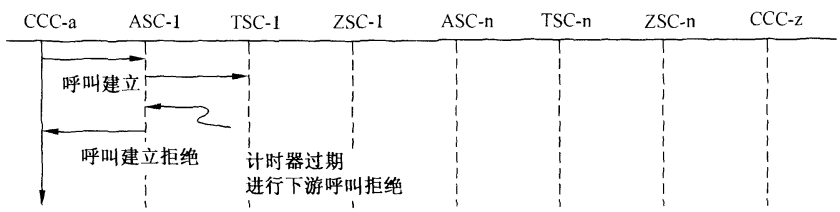


图 20　信令流：建立—域内故障（定时器超时）

定时器超时时，建立请求的拒绝消息会向上游发送，同时会释放已经分配的资源。另外，释放请求也可能会向下游发送来防止下游节点企图处理建立请求。

5.6.2.4　CCC-z 的 UNI 故障

下面的故障原因代码引发信令动作：

——CAC 故障，如验证策略信息；

——域间的链路连接故障，如 ZSN-n 和 AGC-z 之间的链路连接故障；

——子网内部的子网连接故障，如 ZSN-n 内部的子网连接故障；

——LRM 故障，如将请求带宽映射到传送网络资源；

——ZSC-n 的 CC 定时器超时；

——CCC-z 的 CC 定时器超时。

在图 21 和图 22 的故障情况下，CCC-z 拒绝请求，这可能是因为：

——CCC-a 对建立从 AGC-a 到 AGC-z 的连接的许可验证失败（如在呼叫建立过程中）；

——AGC-z 没有资源（如在连接建立过程中）。

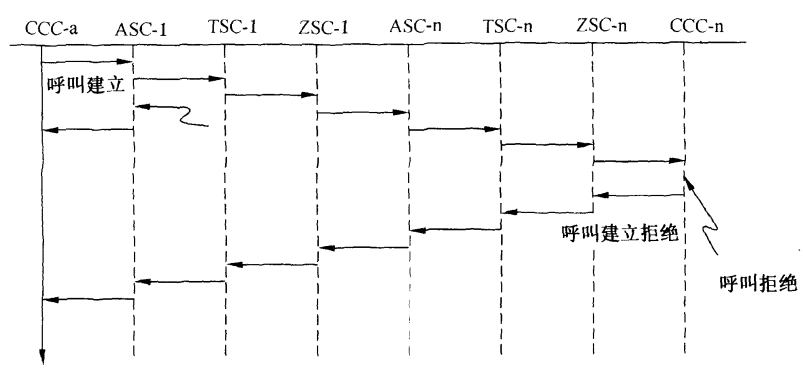


图 21　信令流：建立—CCC-z UNI 故障（呼叫被拒绝）

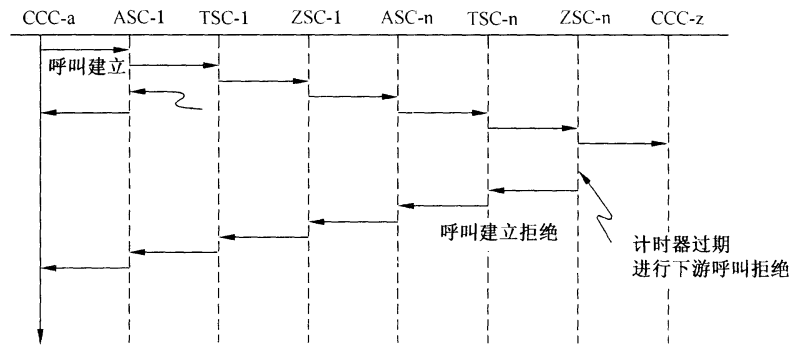


图 22　信令流：建立—CCC-z UNI 故障（定时器超时）

当拒绝响应消息向上游节点传输时，释放子网内部已经预留或已经分配的资源。

在定时器超时的情况下，连接建立请求被拒绝的消息向上游节点传播。并释放对应的连接。另外，释放请求也要发送到 CCC-z 来阻止 CCC-z 企图处理连接请求。

5.6.3　已建立呼叫的异常处理

5.6.3.1　异常情况下的呼叫处理机制

呼叫一旦建立，各种不同的故障会影响存在的呼叫。这些故障可能发生在任何传送网络连接或信令通道上。信令网络的故障和误动作也会影响业务。

传送平面资源发送故障时，根据呼叫请求的类型需要执行两个可能的动作：

——对于建立软永久连接（SPC）的呼叫，控制平面或传送平面通过恢复或保护来恢复故障连接（假设对呼叫提供了保护或恢复功能）。如果在一定的时间周期内连接不能恢复或保护，需要向管理平面发送通知消息。呼叫仍然保持活动状态。

——对于建立交换连接（SC）的呼叫，在硬件恢复不可用或者不成功的情况下，呼叫要被释放（通过释放各个对应的连接）。

在双向故障的情况下，如果需要进行保护恢复动作，两个 CC 都会动作。对于同一个呼叫，在两个 CC 动作时，会产生竞争。为解决竞争问题，标识符大的 CC 的请求覆盖标识符小的 CC 的请求。

5.6.3.2　传送网络连接故障（交换连接）

下面的故障原因代码引发信令动作：

——链路连接故障：链路连接故障发生可能是因为 SNP 故障或因为关联故障；

——子网连接故障：子网连接故障发送可能是因为 SNP 故障或因为关联故障。

图 23 描述了传送网络资源故障的情形。在这种情况下，假定呼叫会因为故障而释放（如连接没有实现分集、保护或恢复功能）。

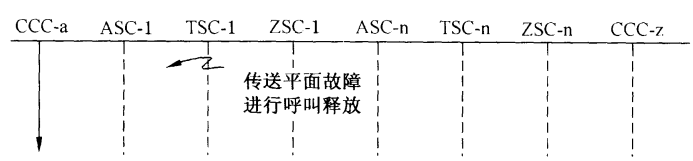


图 23　信令流：存在的呼叫—用户传送网络连接故障

在请求释放的过程中，也可能会发生超时（因为没有收到接收端的响应）。这种情况单独处理，作为释放请求的分异常处理的一部分。但是，对用户的计费是根据呼叫状态的“UP”或“DOWN”来进行的，用户发起的释放命令应该收到对应的释放应答。由于在释放操作过程中的故障可能会产生部分残留的连接，CC 应该通知管理平面异常的情况，通告信息中包含没有释放的残留连接相关信息。

5.6.3.3 传送网络连接故障(软永久连接)

下面的故障原因会导致信令行为:

——链路连接故障:链路或链路连接故障可能是由 SNP 发生故障或关联故障引起;

——子网连接故障:子网连接故障发送可能是因为 SNP 故障或因为关联故障。

图 24 描述了网络内部连接故障发现过程。故障的发生可能会引发相关的恢复机制来对连接进行恢复或保护。这要依靠呼叫的属性(呼叫指定的 CoS/GoS 和路由类型)。如果恢复或保护不成功,应该向管理平面发送通知消息。在没有收到强制呼叫释放命令之前,呼叫会一直保留。

假定如下,如果连接不能恢复,作为故障结果呼叫将会继续维持活动状态。

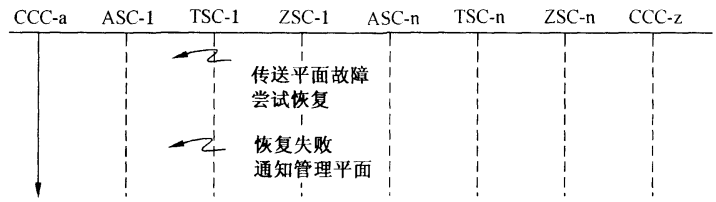


图 24 信令流:存在的呼叫—传送网络连接故障

5.6.4 呼叫释放的异常处理

5.6.4.1 呼叫释放异常处理机制

对于异常情况,网络内部的子网控制器也可能会发起释放请求。通过校验的释放请求必须向用户返回成功的释放应答。任何与释放请求相关的故障要上报给管理系统,包括任何没有释放的残留连接的信息。对于没有成功释放的连接,应该防止对该连接的访问或使用。

5.6.4.2 CCC-a 或 CCC-z 发起的呼叫释放发生故障(请求消息)

下面的故障原因代码会导致信令行为:

——CAC 故障,如验证策略信息发生故障;

——LRM 故障,如释放连接资源发生故障;

——CC 定时器机制超时;

——在 ASC 处理消息发生拥塞。

CCC-a 或 CCC-z 定时器超时,如图 25 所示。用户需要通过交互方式来释放呼叫,如通过手动处理来释放。在网络定时器超时的情况下,网络向用户发送释放确认响应,如图 26 所示。另外,故障信息将通知到管理系统。这样允许对残留连接进行任意清除。

注意:在连接部分释放时,应该有机制防止 AGC-a 或 AGC-z 访问该呼叫。

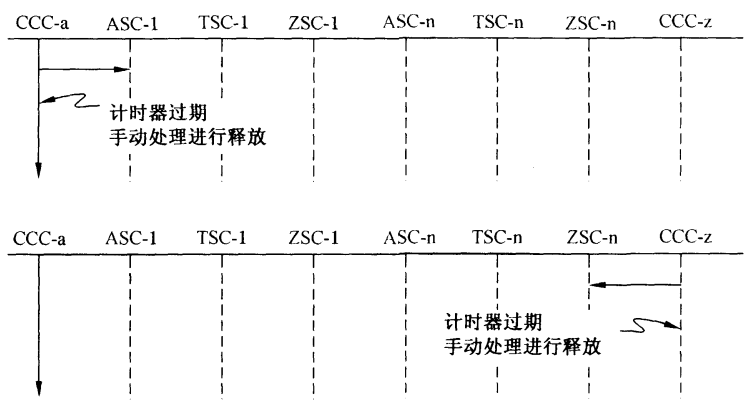


图 25 信令流:呼叫释放—用户发起释放(用户定时器超时)

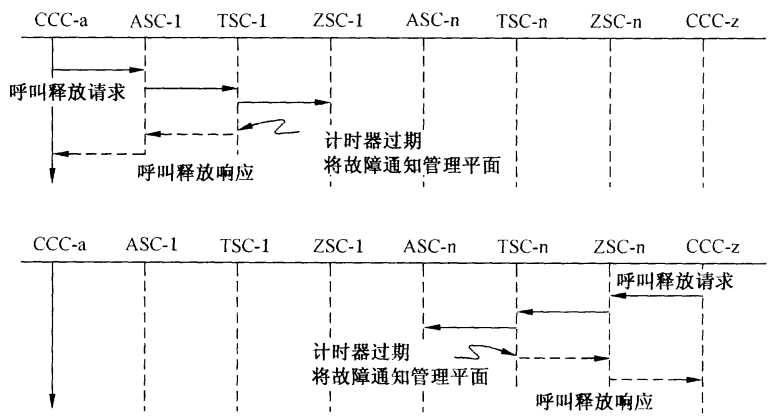


图 26　信令流:呼叫释放—用户发起释放(网络定时器超时)

5.6.4.3　CCC-a 或 CCC-z 发起呼叫释放(响应消息)

下面的出错原因会产生信令行为:

——响应消息没有到达用户请求代理;

——CC 没有对释放应对进行响应。

图 27 故障信令流程。在第一种情况下,从 CCC-a 的角度看,和超时情况类似。这种情况下,下游连接已经被释放。这种情况要引入额外的功能让管理系统来标识连接的状态。如 5.6.4.2 中的超时情况,连接一直为 UP 状态,但这个例子中连接已经部分删除。

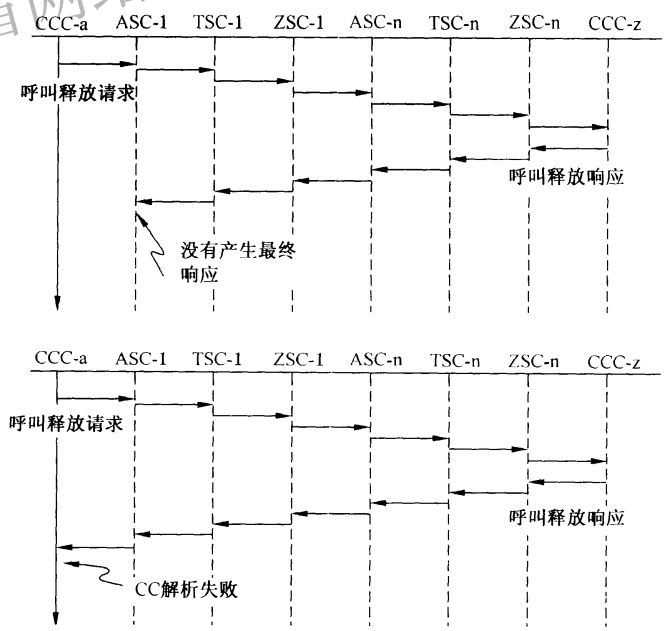


图 27　信令流:呼叫释放—信令响应消息故障

5.6.5　连接资源释放

如果网络不能建立一条新的呼叫的所有连接,任何建立成功或者部分建立成功的连接都需要被删除,同时呼叫请求会被拒绝。这里假设恢复重试操作(如果策略允许)没有成功。SC 和 SPC 的连接释放如图 28 和图 29 所示。

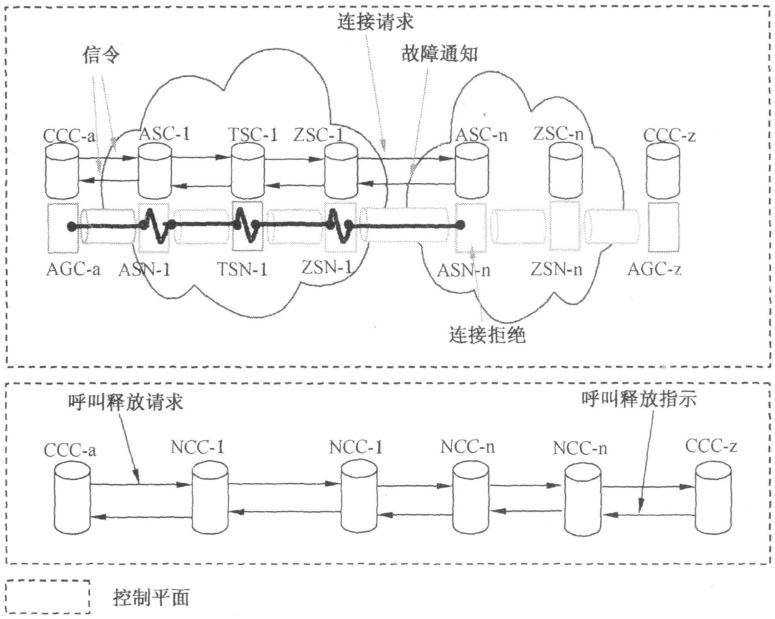


图 28　SC 连接建立故障后的呼叫释放

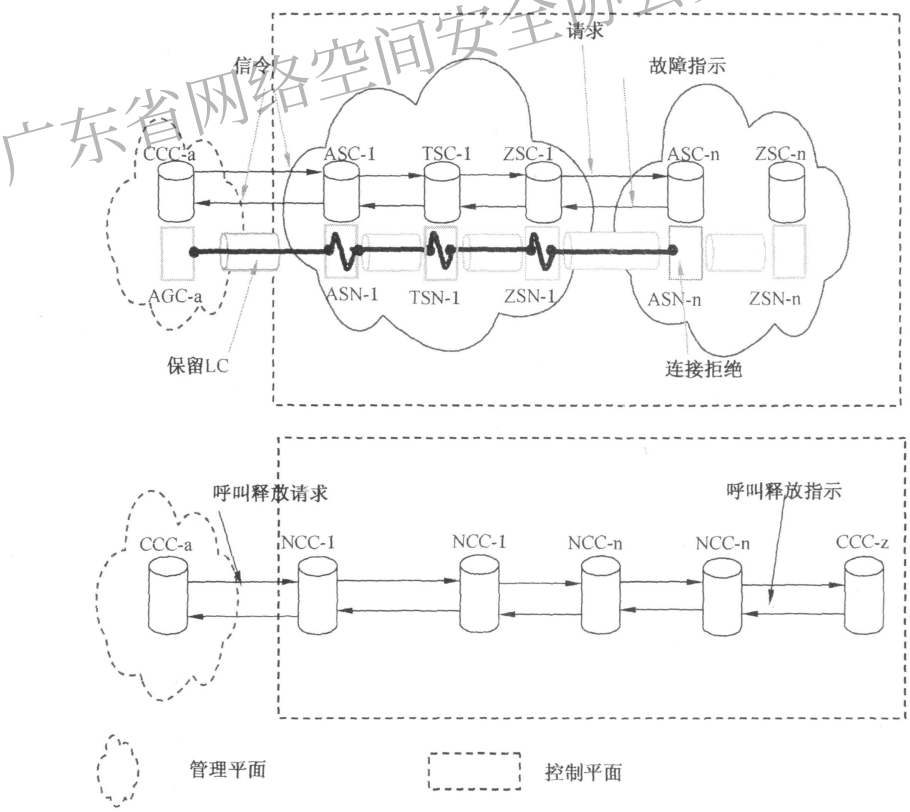


图 29　SPC 连接建立故障后的呼叫释放

在图 28 和图 29 所示的情形中，呼叫的连接在 ASC-n 被拒绝。故障通知指示向 CCC-a 发送，同时释放相应的连接。在图 28 中，CCC-a 位于用户设备，在图 29 中，CCC-a 位于管理平面。在两种情况中，呼叫和连接处理在信令层面都进行完全的分离，并且假定呼叫在被叫放实体进行处理。在图 29 中，到达网络边界的连接都会释放，但是到达用户的 LC 仍旧保留。这是因为 LC 是由管理平面配置的。

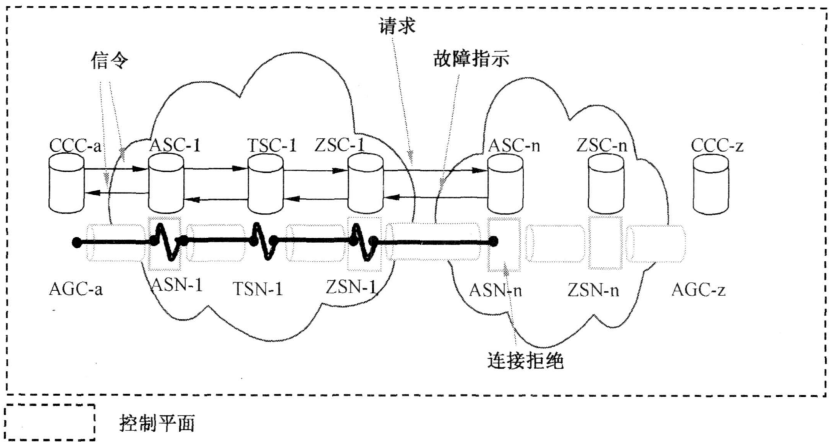


图 30　SC 连接建立故障后呼叫状态不改变

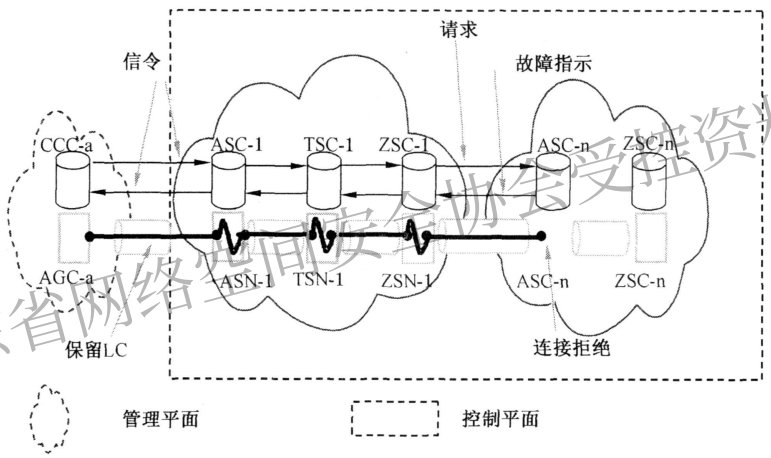


图 31　SPC 连接建立故障后呼叫状态不改变

对于呼叫修改操作，如果网络不能修改连接，那么就认为呼叫修改操作是失败的。任何已修改或者部分修改的连接都应该被释放，对现有的呼叫不做任何改变。图 30 和图 31 分别显示了 SC 和 SPC 的情况。

在图 30 和图 31 所示的情形中，对呼叫的修改操作造成新的连接请求。当此连接请求在 ASC-n 被拒绝时，故障通知向 CCC-a 发送，并且连接被释放。CCC-a 在接收到故障通知以后，CCC-a 发送连接释放请求，释放故障的连接并且保持呼叫的状态不变。

6　信令的弹性

6.1　信令控制器的弹性

6.1.1　信令控制器的弹性定义

信令控制器的弹性指信令控制器在故障情况下继续工作的能力。信令控制器位于 UNI、I-NNI 以及 E-NNI，用于支持呼叫以及连接的建立和拆除功能。它的操作依赖于数据通信网络(DCN)、传送平面、管理平面以及控制平面本身的内部元件。

6.1.2　信令控制器的故障检测和指示

信令控制器与传送平面的通信中断，或者信令控制器与其相邻信令控制器之间的通信中断，则认为信令控制器出现了故障。这些故障可能是由于 DCN 的故障或者其他的原因(例如相邻信令控制器的故障)引起的。

控制平面必须能够探测与传送平面之间的通信故障，并且能够将这些故障通知信令控制器。控制平面和传送平面之间的通信故障不应引起信令通道的故障。

信令通道的维护必须由信令协议实现。信令通道的故障可能是由 DCN 的故障或者是相邻信令控制器故障引起。信令协议的故障探测机制应该在以下几种情况下都能执行。

当发生故障时：

——在故障以及故障恢复的过程中，现存完整的呼叫及其连接不能发生改变；

——信令通道的故障必须产生告警，此故障应通知其他的信令控制器。当故障持久存在或者需要管理者参与时，应将故障通知管理平面；

——不接受或者处理信令消息。如果控制平面和传送平面之间出现了故障，而信令控制器仍旧是可达的，新的呼叫建立或拆除请求以及新的连接建立或拆除请求将会失败，同时发出适当的错误指示。

如果故障是由于信令通道引起的，新的呼叫建立或拆除请求将会在 UNI、E-NNI 或 I-NNI 丢失。新的连接建立或者拆除请求将会在 I-NNI 丢失。

6.1.3 信令控制器与传送平面的同步

当控制平面和传送平面之间的通信可用时，信令控制器必须重建呼叫和连接的状态，以同步传送平面的连接信息。其中一种可能的同步过程是：

——链路资源管理器同步传送平面的 NE 状态信息、保护 NE 的交叉连接信息以及端口信息；

——连接控制器同链路资源管理器进行同步，恢复连接的状态；

——呼叫控制器（如果应用）同连接控制器进行同步，恢复呼叫状态。

在这个过程中，信令协议控制器依赖于其他错误通知（如零带宽）或者维护合适的状态，如垂直恢复状态。在这种状态下，所有的信令信息都是可接受的，但是不进行处理。另外必须返回适当的消息，指示错误。

6.1.4 信令控制器与邻接信令控制器的同步

当信令控制器之间的通信可用时，信令控制器必须同它的邻居同步呼叫和连接的状态。信令控制器必须具备正确的状态，如水平恢复状态。在这种状态下，应该拒绝用于建立或者拆除现存呼叫以及连接的消息，另外信令控制器必须检查它的连接信息是否同它相邻控制器的连接信息保持一致。检查的内容包括：

——呼叫和连接 ID：如果呼叫或连接的 ID 只在一端信令控制器存在，那么此呼叫或连接是无效的；

——对相同连接的资源分配：如果在两端对一条连接的资源分配不一致，那么此连接无效。

当标识出无效的呼叫或者连接时，信令协议控制器必须发送相应的消息删除这些呼叫和连接，这些不完整的呼叫或连接将会被删除。

6.2 信令网络的弹性

6.2.1 信令网络的弹性定义

信令网络对呼叫控制器之间和连接控制器之间的信令消息提供可靠的传送功能。信令通道是信令网中提供信令实体之间消息传送的连接。信令通道失效后应该执行如下的处理：

——检测到信令通路失效后，信令网络尝试恢复信令通信。如可以通过冗余信令通道实现信令网故障恢复。在信令网络恢复期间不接受处理信令消息，不受故障影响的信令通道上的消息仍然正常处理，受故障影响的链路上的消息不能进行处理。已经存在并受故障影响的呼叫维持现状。

——如果故障不可恢复，如呼叫控制器（CallC）或 CC 不能和其他的 CallC 或 CC 通信，需要向管理系统发送通告信息。已经存在的呼叫依然维持现状。可以通过其他的手段来释放这些受影响

的呼叫，如通过手动处理。

——故障恢复成功后，CallC 或 CC 可能需要同步已经存在的呼叫和连接状态数据。

如果信令网络和信令通道都支持恢复，单个信令网络的故障可能会同时导致 DCN 和控制平面都开始恢复。这些恢复机制之间需要统一协调以避免对同一个故障进行共同恢复。

另外，不同组件的故障（CallC、CC、LRM 发生故障）会导致控制平面故障。如，有些故障是不可恢复的。但是，应该向管理平面上报相关的故障信息。

6.2.2 用户信令故障

根据下面的故障原因代码，应该执行对应的信令动作：

——CCC-a 和 ASC-1 之间的信令通道中断；

——ZSC-n 和 CCC-z 之间的信令通道中断。

其他目前还没有考虑到的故障原因代码，也会影响故障。当额外的故障原因编码会导致故障并被标识出来后，应该添加故障原因编码。

图 32 中描述了 CCC-a 和 ASC-1 之间的故障处理。ZSC-N 和 CCC-z 之间的故障处理类似。如 CCC-z 或 ZSC-n 发起信令通道的恢复。

图 32 UNI 信令故障

上图中，从 CCC-a 到 ASC-1 的信令连接故障检测。由于 CCC-a 不发起呼叫释放请求，需要指定其他的方法。不影响已经存在的呼叫。

6.2.3 网络信令故障

下面的故障原因导致信令动作：

——域内的 CC 之间信令通道失效；

——域间的 CC 之间信令通道失效。

其他的现在还没有考虑到的故障原因代码，也会影响故障。当额外的故障原因编码会导致故障并被标识出来后，应该添加故障原因编码。

图 33 中描述了 ASC-1 和 TSC-1 之间域内故障情况。

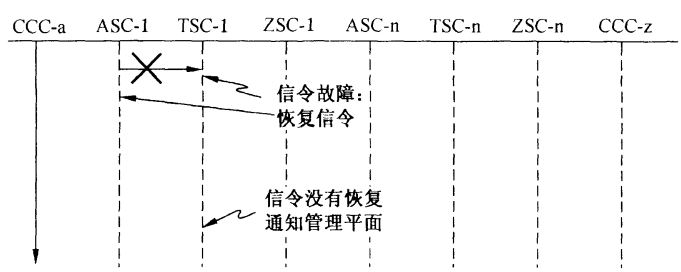


图 33 网络信令故障

信令通道故障不会导致呼叫的释放。网络要尽量尝试恢复信令通道。当信令通道故障恢复后，控制平面继续处理新的请求。如果故障的信令通道不能恢复，CC 通知管理平面相关的故障信息。

7 分布式连接管理(DCM)属性列表

7.1 分布式连接管理属性类型

DCM 属性列表可以分为呼叫属性和连接属性。表 1、表 2、表 3 总结了 UNI、I-NNI 和 E-NNI 信令处理相关的属性列表。

——UNI 信令处理的内容包括呼叫属性，也包括从用户到网络之间链路连接建立相关的连接属性。

——NNI 信令处理包括连接属性。呼叫属性必须在呼叫控制器之间交互(如图 1 的 ASC-n 和 ZSC-n 之间)。所用到的许多交互机制不是该结构中的部分。I-NNI 信令可以通过将呼叫属性信息附加在连接相关的信息上来交换呼叫属性信息。但是，这样处理不会形成 I-NNI 处理的一部分。

——E-NNI 信令处理的内容包括呼叫属性，也包括从网络之间链路连接建立相关的连接属性。

属性信息为逻辑信息，这些逻辑信息在支持 CCC 或 NCC、CC 和 LRM 的接口上进行交换。不同的协议设计可能会对一些逻辑信息进行合成或分割。但应该体现属性信息所支持的功能。

表 1 UNI 属性列表

	属性	作用范围	消息类型
标识属性	主叫方 UNI 传送资源标识符	端到端	呼叫
	被叫方 UNI 传送资源标识符	端到端	呼叫
	源 CC 名称	本地	连接
	源呼叫控制器(CallC)名称	本地	呼叫
	目的 CC 名称	本地	连接
	目的 CallC 名称	本地	呼叫
	连接名称	本地	连接
	呼叫名称	端到端	呼叫
业务属性	主叫方 AGC SNP ID	本地	连接
	主叫方 AGC SNPP ID	本地	连接
	被叫方 AGC SNP ID	远端本地	连接
	被叫方 AGC SNPP ID	远端本地	连接
	方向	本地	呼叫或连接
策略属性	CoS	端到端[a]	呼叫
	GoS	端到端[a]	呼叫
	安全	本地	呼叫或连接
[a] 尽管 CoS 和 GoS 为端到端有效，但它们的值在跨域时可能会改变。但要满足和请求业务相关的策略。			

表 2　I-NNI 属性列表

	属性	作用范围	消息类型
标识属性	主叫方 UNI 传送资源标识符	透明传输	呼叫
	被叫方 UNI 传送资源标识符	透明传输	呼叫
	源呼叫控制器(CallC)名称	本地	呼叫
	目的 CC 名称	本地	连接
	目的 CallC 名称	本地	呼叫
	连接名称	本地	连接
	连接名称	域内唯一	连接
	呼叫名称	端到端	呼叫
业务属性	SNP ID	本地 Local	连接
	SNPP ID	本地 Local	连接
	主叫方 AGC SNP ID	透明传输	连接
	被叫方 AGC SNPP ID	透明传输	连接
	方向	域内唯一	呼叫或连接
策略属性	CoS	透明传输	呼叫
	GoS	透明传输	呼叫
	连接 CoS	域内唯一	连接
	连接 GoS	域内唯一	连接
	显式资源列表	域内唯一	连接
	恢复	域内唯一	连接

表 3　E-NNI 属性列表

	属性	作用范围	消息类型
标识属性	主叫方 UNI 传送资源标识符	端到端或透明传输	呼叫
	被叫方 UNI 传送资源标识符	端到端或透明传输	呼叫
	源 CallC 名称	本地	呼叫
	目的 CC 名称	本地	连接
	目的 CallC 名称	本地	呼叫
	连接名称	本地	连接
	连接名称	本地	连接
	呼叫名称	端到端	呼叫
业务属性	SNP ID	本地	连接
	SNPP ID	本地	连接
	主叫方 AGC SNP ID	透明传输	连接
	被叫方 AGC SNPP ID	透明传输	连接
	方向	本地	呼叫或连接
策略属性	CoS	端到端	呼叫

表 3（续）

	属性	作用范围	消息类型
策略属性	GoS	端到端	呼叫
	安全	本地	呼叫或连接
	显式资源列表	本地	连接
	恢复	本地	连接

7.2 UNI 属性列表

7.2.1 标识属性

7.2.1.1 呼叫方 UNI 传送资源标识符

该属性用于标识到达 A 端呼叫控制器的 UNI 传送资源标识符。此属性的值是全局唯一的，由服务提供商分配。例如，用户名称可以由网络分配一个 IPv4 地址，其他另外的用户名称可以由网络分配一个 IPv6 地址。由于用户名称为用户的全网唯一标识，因此不同的编址格式可以共存。

7.2.1.2 被叫方 UNI 传送资源标识符

该属性用于标识到达 Z 端呼叫控制器的 UNI 传送资源地址。具体的特征和主叫方 UNI 传送资源标识符相同。

7.2.1.3 源端 CC 和呼叫控制器(CallC)名称

这个属性指定了和 CC 和 CallC 相关的名称，该 CC 或 CallC 发起显示信令消息。

7.2.1.4 目的端 CC 和呼叫控制器(CallC)名称

这个属性指定了和 CC 和 CallC 相关的名称，该 CC 或 CallC 终结显示信令消息。

7.2.1.5 连接名称

该属性唯一标识一条链路连接，该值为本地唯一，既可以由用户分配也可以由网络分配。

7.2.1.6 呼叫名称

该属性唯一标识一个请求的呼叫。该值为全局唯一并由网络分配。

7.2.2 业务属性

7.2.2.1 SNP ID

SNP ID 代表在连接请求消息中用来建立一条链路连接的子网点。SNP ID 也可以用来创建 SNC 连接的 SNP。对于具体的连接请求，SNP ID 从一组在指定 SNPP 内部的 SNP 中选择。SNP ID 值本地唯一，可以自动发现或手动提供，并在层网络之间独立。对于请求双向连接的呼叫，或请求多条连接的呼叫，SNP ID 包含多个值，并按下游 ID 在前上游 ID 在后的顺序排列。SNP ID 由源节点或终节点 LRM 来指定。为防止竞争，名字值更大的 LRM 可以优先选择 SNP ID。

信令消息的 SNP ID 包括：

——主叫方 AGC SNP ID：该 ID 用来建立从主叫方 AGC 到网元之间的 LC 链路；

——被叫方 AGC SNP ID：该 ID 用来建立从被叫方 AGC 到网元之间的 LC 链路；

——SNP ID：该 ID 用来建立网元与网元之间的 LC 连接链路。

7.2.2.2 SNPP ID

SNPP ID 用来标识子网点池，用于请求建立一条连接。SNPP ID 唯一标识一组子网间用来请求连接的 SNP，在不同的网络层次之间 SNPP ID 是彼此独立的。子网之间可以存在多个 SNPP。对于请求双向连接或请求多条连接的呼叫来说，该属性可以包含多个属性值并且按下游 ID 在前上游 ID 在后的顺序排列。SNPP ID 可以由发起点 LRM 来指定，由终结点 LRM 在 SNPP 中选择 SNP 来建立链路连接。为防止竞争，具有较高名字值的 LRM 优先选择 SNPP ID，以满足基于约束的呼叫请求。

7.2.2.3 方向性

方向属性指定了连接的方向。方向属性支持单向、对称双向以及不对称双向的连接请求。在不对

称双向请求时，要在属性附加信息中指定上游和下游方向请求连接的数目。

7.2.3 策略属性

7.2.3.1 服务分类(CoS)和连接 CoS

服务分类属性指定呼叫请求的 CoS。该属性值在每个域中唯一，由网络分配并且每个用户网络之间的 CoS 值都可以不同(如，用户 1 使用的 CoS 和用户 2 使用的 CoS 可以不同)。跨域的情况下，为支持端到端的 CoS 请求，网络需要提供转换功能，将一个网络的 CoS 值转换成另外一个网络的 CoS 值。呼叫 CoS 属性为主叫方 SLA 指定 CoS 信息的部分属性。网络应该提供将呼叫 CoS 转换成域中具体的 CoS(连接 CoS)功能。

域与域之间的连接 CoS 可能不同，但每个域中的连接 CoS 要和 SLA 对应以支持端到端的 CoS 请求。CoS 中的信息列表包括：服务分类的列举。

7.2.3.2 服务等级(GoS)和连接 GoS

服务等级属性指定呼叫请求的 GoS 并被用来更详细的指定和每个 CoS 请求相关的 GoS。服务等级属性值在每个域中唯一并由网络分配，每个用户网络关系可以有不同的 GoS。跨域的情况下，为支持端到端的 GoS 请求，网络需要提供转换功能，将一个网络的 GoS 值转换成另外一个网络的 GoS 值。呼叫 GoS 属性为主叫方 SLA 指定 GoS 信息的部分属性。网络应该提供将呼叫 GoS 转换成域中具体的 GoS(连接 GoS)功能。

域与域之间的连接 GoS 可能不同，但每个域中的连接 GoS 要和 SLA 对应以支持端到端的 GoS 请求。GoS 中的信息列表包括：

a) 分集信息；

b) 子网控制器名字列表、SNPP、SNP、避开或必经的 SNPP 和 SNP。

7.2.3.3 安全性

安全属性指定需要允许验证呼叫请求的相关信息，这里包括的信息可以对呼叫请求进行授权认证，或者检查呼叫请求的完整性。该属性值应具有本地唯一性。

7.2.4 状态属性

7.2.4.1 呼叫响应编码

呼叫建立—成功

呼叫建立—失败：被叫方忙

呼叫建立—失败：主叫方忙

呼叫建立—失败：网络忙

呼叫建立—失败：消息出错

呼叫建立—失败：标识错(A 端用户名无效)

呼叫建立—失败：标识错(Z 端用户名无效)

呼叫建立—失败：标识错(连接名无效)

呼叫建立—失败：业务错(SNP ID 无效)

呼叫建立—失败：业务错(SNP ID 不可用)

呼叫建立—失败：业务错(无效的 SNPP ID)

呼叫建立—失败：业务错(SNPP ID 不可用)

呼叫建立—失败：策略错(无效的 CoS)

呼叫建立—失败：策略错(CoS 不可用)

呼叫建立—失败：策略错(GoS 无效)

呼叫建立—失败：策略错(GoS 不可用)

呼叫建立—失败：策略错(安全验证失败)

呼叫释放—成功

呼叫释放—失败:消息出错

呼叫释放—失败:标识错(呼叫名无效)

呼叫释放—失败:策略错(安全验证失败)

7.2.4.2 通知编码

呼叫出错—不影响业务

呼叫出错—影响业务

7.3 I-NNI 属性列表

7.3.1 I-NNI 接口属性定义

表 2 定义的属性和 UNI 相应属性定义相同,下列属性除外。

7.3.2 策略属性

7.3.2.1 连接名称

此属性唯一的标识一条子网络连接,属性值在域内是唯一的。

7.3.2.2 显式资源列表

显式资源列表属性指定了用来建立连接的显式资源。列表有序排列,包括零个或多个 CC 名字实例、路由控制域内的 SNPP ID 和(或)SNP ID,如路由域。

7.3.2.3 恢复

恢复属性指定了连接所使用的恢复算法。例如,恢复属性信息包括的内容如下:

a) 连接类型指示(工作或保护连接);

b) 恢复类型(1+1、1∶1、自动重路由);

c) 恢复行为(返回式、非返回式)。

7.3.2.4 服务分类(CoS)和连接 CoS

见 GB/T 21645.1—2008 的 3.1.19。

7.3.2.5 服务等级(GoS)和连接 GoS

见 GB/T 21645.1—2008 的 3.1.28。

7.3.3 状态属性

7.3.3.1 连接响应编码

连接建立—成功

连接建立—失败:被叫方忙

连接建立—失败:主叫方忙

连接建立—失败:超时

连接建立—失败:标识错(连接名无效)

连接建立—失败:业务错(SNP ID 无效)

连接建立—失败:业务错(SNP ID 不可用)

连接建立—失败:业务错(无效的 SNPP ID)

连接建立—失败:业务错(SNPP ID 不可用)

连接建立—失败:策略错(无效的显式资源列表)

连接建立—失败:策略错(恢复无效)

连接建立—失败:连接错误(创建 SNC 连接失败)

连接建立—失败:连接错误(创建 LC 连接失败)

连接释放—成功

连接释放—失败:消息出错

连接释放—失败:超时

连接释放—失败:标识错(呼叫名无效)

连接释放—失败:连接错误(释放 SNC 连接失败)

连接释放—失败:连接错误(释放 LC 连接失败)

7.3.3.2 通知编码

连接出错—不影响业务

连接出错—影响业务

连接出错—异常呼叫释放

7.4 E-NNI 属性列表

7.4.1 E-NNI 属性定义

表 3 定义的属性和 UNI 相应属性定义相同,下列属性除外。

7.4.2 状态属性

7.4.2.1 连接响应编码

连接建立—成功

连接建立—失败:消息出错

连接建立—失败:被叫方忙

连接建立—失败:主叫方忙

连接建立—失败:超时

连接建立—失败:标识错(A 端用户名无效)

连接建立—失败:标识错(Z 端用户名无效)

连接建立—失败:标识错(连接名无效)

连接建立—失败:业务错(SNP ID 无效)

连接建立—失败:业务错(SNP ID 不可用)

连接建立—失败:业务错(无效的 SNPP ID)

连接建立—失败:业务错(SNPP ID 不可用)

连接建立—失败:策略错(CoS 无效)

连接建立—失败:策略错(CoS 不可用)

连接建立—失败:策略错(GoS 无效)

连接建立—失败:策略错(GoS 不可用)

连接建立—失败:策略错(安全验证失败)

连接建立—失败:策略错(显式资源列表无效)

连接建立—失败:策略错(恢复无效)

连接建立—失败:连接错误(创建 SNC 连接失败)

连接建立—失败:连接错误(创建 LC 连接失败)

连接释放—成功

连接释放—失败:消息出错

连接释放—失败:超时

连接释放—失败:标识出错(呼叫名无效)

连接释放—失败:策略出错(安全验证失败)

连接释放—失败:连接错误(释放 SNC 连接失败)

连接释放—失败:连接错误(释放 LC 连接失败)

7.4.2.2 通知编码

连接出错—不影响业务

连接出错—影响业务

连接出错—异常呼叫释放

8 分布式连接管理(DCM)消息集

8.1 分布式连接管理消息类型

DCM 消息可以根据 UNI 连接操作以及 NNI(包括 I-NNI 和 E-NNI)连接操作来进行分类。表 4～表 6 总结了信令过程中可能要包含的消息列表。这些消息为逻辑消息,在各自的接口上支持呼叫控制器(CallC)、CC 和 LRM。注意,不同的协议设计可能会对一些逻辑消息进行合并或分拆。但应该体现消息所支持的功能。

属性修改相关的操作待研究,暂不作为消息集的一部分。

表 4 UNI 消息

	UNI 消息
呼叫建立消息	呼叫建立请求
	呼叫建立指示
	呼叫建立确认
呼叫释放消息	呼叫释放请求
	呼叫释放指示
呼叫查询消息	呼叫查询请求
	呼叫查询指示
呼叫通知消息	呼叫通知

表 5 I-NNI 消息

	I-NNI 消息
连接建立消息	连接建立请求
	连接建立指示
	连接建立确认
连接释放消息	连接释放请求
	连接释放指示
连接查询消息	连接查询请求
	连接查询指示
连接通知消息	连接通知

表 6 E-NNI 消息

	E-NNI 消息
连接建立消息	连接建立请求
	连接建立指示
	连接建立确认
连接释放消息	连接释放请求
	连接释放指示
连接查询消息	连接查询请求
	连接查询指示
连接通知消息	连接通知

8.2 UNI 消息

8.2.1 呼叫建立

8.2.1.1 呼叫建立机制

呼叫建立定义了两个步骤(和可选的第三个步骤)。网络创建呼叫名称并发送给用户作为客户呼叫请求的回应。连接名称由发起呼叫请求者为所请求的连接创建。

8.2.1.2 呼叫建立请求

定义呼叫建立请求(callSetupRequest)消息用来建立呼叫。相关的呼叫建立请求属性如下表 7 所示。

表 7 UNI 呼叫建立请求消息

用户发送消息属性	网络发送消息属性
主叫方 UNI 传送资源标识符	主叫方 UNI 传送资源标识符
被叫方 UNI 传送资源标识符	被叫方 UNI 传送资源标识符
源呼叫控制器(CallC)名称	源 CallC 名称
目的 CallC 名称	目的 CallC 名称
主叫方 AGC SNP ID	主叫方 AGC SNP ID
主叫方 AGC SNPP ID	主叫方 AGC SNPP ID
被叫方 AGC SNP ID	被叫方 AGC SNP ID
被叫方 AGC SNPP ID	被叫方 AGC SNPP ID
方向	方向
CoS	CoS
GoS	GoS
安全	安全
连接名称	连接名称
	呼叫名称

8.2.1.3 建立呼叫指示

定义呼叫建立指示(callSetupIndication)消息作为呼叫建立请求消息的响应。指示消息可以由 Z 端用户或网络来发送。与呼叫建立指示相关的消息如表 8 所示。

表 8 UNI 呼叫建立指示消息

用户发送属性	网络发送属性
连接名称	连接名称
呼叫名称	呼叫名称
状态	状态

8.2.1.4 呼叫建立确认

定义呼叫建立确认消息(callSetupConfirm)来响应呼叫建立指示消息。呼叫建立确认消息可以由 A 端用户或网络来发送。与呼叫建立确认相关的属性如表 9 所示。

表 9 UNI 呼叫建立确认消息

用户发送属性	网络发送属性
连接名称	连接名称
呼叫名称	呼叫名称
状态	状态

8.2.2 呼叫释放

8.2.2.1 呼叫释放请求

定义呼叫释放请求消息(callReleaseRequest)来释放一个呼叫。呼叫释放消息可以由用户或网络来发送。呼叫释放消息中包含的属性如表10所示。

表10 UNI链路连接释放请求消息

用户发送属性	网络发送属性
呼叫名称	呼叫名称
安全	安全

8.2.2.2 呼叫释放指示

定义呼叫释放指示(callReleaseIndication)消息作为呼叫释放请求消息的响应。呼叫释放指示消息可以由用户或网络来发送。呼叫释放指示消息中包含的属性信息如表11所示。

表11 UNI呼叫释放指示消息

用户发送属性	网络发送属性
呼叫名称	呼叫名称
状态	状态

8.2.3 呼叫查询

8.2.3.1 呼叫查询请求—查询每个呼叫特征

定义呼叫查询请求消息(callQueryRequest)用来查询存在的呼叫。呼叫查询消息由用户或网络来发送。呼叫查询请求消息中包含的属性如表12所示。

表12 UNI呼叫查询请求消息

用户发送属性	网络发送属性
呼叫名称	呼叫名称
安全	安全

8.2.3.2 呼叫查询相应—查询每个呼叫各种

定义呼叫查询响应消息(callQueryResponse)用来响应呼叫查询请求。呼叫查询响应消息由用户或网络来发送。呼叫查询响应消息中包含的属性如表13所示。

表13 UNI呼叫查询响应消息

用户发送属性	网络发送属性
呼叫名称	呼叫名称
CoS	CoS
GoS	GoS
每条连接名称 SNP ID SNPP ID	每条连接名称 SNP ID SNPP ID
状态	状态

8.2.3.3 呼叫查询请求—查询所有呼叫

定义查询所有呼叫的请求消息(callQueryAllRequest)用来查询与特定的信令代理或网络呼叫控制器名称相关的所有现存的呼叫。此信息可以由其他的用户或网络发生。查询所有呼叫的请求消息中包含的属性如表14所示。

表 14 UNI 所有呼叫查询请求消息

用户发送属性	网络发送属性
CC 或 NCC 名称	CC 或 NCC 名称
安全	安全

8.2.3.4 呼叫请求响应—所有呼叫查询响应

定义查询所有呼叫的响应消息(callQueryAllResponse)用来响应 callQueryAllRequest 消息,该响应消息由用户或网络来发送。查询所有呼叫的响应消息中包含的属性如表 15 所示。

表 15 UNI 所有呼叫查询响应消息

用户发送属性	网络发送属性
CC 或 NCC 名称	CC 或 NCC 名称
呼叫名称列表	呼叫名称列表
状态	状态

8.2.4 通知消息

使用通知消息允许交换连接相关的状态信息。通知消息的发送用来通知呼叫或连接的状态。与通知消息相关的属性信息如下表 16 所示。

表 16 UNI 通知消息

用户发送属性	网络发送属性
呼叫名称	呼叫名称
连接名称	连接名称
出错编码	出错编码
安全	安全

通知消息一步完成。

8.3 I-NNI 消息

8.3.1 连接建立

8.3.1.1 连接建立请求

连接建立分两个步骤完成(第 3 个步骤可选)。为所请求的连接创建一个连接名字。

连接建立请求消息(connSetupRequest)用于建立一条连接。连接建立请求消息中包含的属性如表 17 所示。

表 17 I-NNI 连接建立请求消息

属　　性
主叫方 UNI 传送资源标识符
被叫方 UNI 传送资源标识符
源 CC 名称
目的 CC 名称
连接名称
呼叫名称
本地 SNP ID
本地 SNPP ID

表 17（续）

属　　性
被叫方 AGC SNP ID
被叫方 AGC SNPP ID
方向性
CoS 服务分类
GoS 服务等级
显式资源列表
恢复

8.3.1.2 连接建立指示

连接建立指示消息(connSetupIndication)用于响应连接建立请求消息。与连接建立指示相关的属性信息如表 18 所示。

表 18 I-NNI 连接建立指示消息

属　　性
连接名称
呼叫名称
状态

8.3.1.3 连接建立确认

可选的连接建立确认消息(connSetupConfirm)是为了对连接建立指示进行响应。连接建立确认属性如表 19 所示。

表 19 I-NNI 连接建立确认消息

属　　性
连接名称
呼叫名称
状态

8.3.2 连接释放

8.3.2.1 连接释放请求

连接释放请求消息(connReleaseRequest)用来释放一条连接。释放请求消息属性如表 20 所示。

表 20 I-NNI 连接释放请求消息

属　　性
呼叫名称
连接名称

8.3.2.2 连接释放指示

连接释放指示消息(connReleaseIndication)是对连接释放请求进行响应。连接释放指示消息包含的属性信息如表 21 所示。

表 21 I-NNI 连接释放指示消息

属　　性
呼叫名称
连接名称
状态

8.3.3 连接查询

下面给出了连接查询的有效结果：连接激活、连接不存在、连接不可用、连接挂起。

8.3.3.1 连接查询请求—查询每条连接

连接查询请求消息(connQueryRequest)用来查询一条存在的连接。连接查询请求相关的属性信息如表22所示。

表22 I-NNI连接查询请求消息

属　　性
呼叫名称
连接名称

8.3.3.2 连接查询响应—查询每条连接的特征

定义连接查询响应消息connQueryResponse用来响应连接查询请求。连接查询响应消息中的属性如表23所示。

表23 I-NNI连接查询响应消息

属　　性
呼叫名称
连接名称 SNP ID SNPP ID
方向
显式资源列表
恢复
状态

8.3.3.3 连接查询请求—查询所有连接

查询所有连接的请求消息(connQueryAllRequest)用来查询与连接控制器名称对应的所有存在的连接。查询请求消息的属性如表24所示。

表24 I-NNI所有连接查询请求消息

名　　称
CC名称

8.3.3.4 连接查询响应—查询所有连接

查询所有连接的响应消息(connQueryAllResponse)用来响应所有连接的查询请求。响应消息可以由用户或网络来发送。响应消息中的属性信息如表25所示。

表25 I-NNI所有连接查询响应消息

属　　性
CC名称
连接名称列表及其相应呼叫名称
状态

8.3.4 通知消息

定义通知消息(notification)用来允许对连接状态相关的信息进行交换。发送通知消息用来对呼叫或连接进行通知。通知消息相关的属性如表26所示。

表 26 I-NNI 通知消息

属　　性
呼叫名称
连接名称
出错编码

通知消息只需要一个步骤。

8.4 E-NNI 消息

8.4.1 连接建立

8.4.1.1 连接建立请求

连接建立分两个步骤完成(第 3 个步骤可选)。为所请求的连接创建一个连接名字。

连接建立请求消息(connSetupRequest)用于建立一条连接。连接建立请求消息中包含的属性如表 27 所示。

表 27 E-NNI 连接建立请求消息

属　　性
主叫方 UNI 传送资源标识符
被叫方 UNI 传送资源标识符
源 CC 或 NCC 名称
目的 CC 或 NCC 名称
连接名称
呼叫名称
本地 SNP ID
本地 SNPP ID
被叫方 AGC SNP ID
被叫方 AGC SNPP ID
方向性
CoS 服务分类
GoS 服务等级
显式资源列表
恢复

8.4.1.2 连接建立指示

连接建立指示消息(connSetupIndication)用来响应连接建立请求消息。与连接建立指示相关的属性信息如表 28 所示。

表 28 E-NNI 连接建立指示消息

属　　性
连接名称
呼叫名称
状态

8.4.1.3 连接建立确认

可选的连接建立确认消息(connSetupConfirm)是为了对连接建立指示进行响应。连接建立确认属性如表29所示。

表29 E-NNI连接建立确认消息

属　　性
连接名称
呼叫名称
状态

8.4.2 连接释放

8.4.2.1 连接释放请求

连接释放请求消息(connReleaseRequest)用来释放一条连接。释放请求消息属性如表30所示。

表30 E-NNI连接释放请求消息

属　　性
呼叫名称
连接名称
安全

8.4.2.2 连接释放指示

连接释放指示消息(connReleaseIndication)是对连接释放请求进行响应。连接释放指示消息包含的属性信息如表31所示。

表31 E-NNI连接释放指示消息

属　　性
呼叫名称
连接名称
状态

8.4.3 连接查询

8.4.3.1 连接查询请求—查询每条连接

连接查询请求消息(connQueryRequest)用来查询一条存在的连接。连接查询请求相关的属性信息如表32所示。

表32 E-NNI连接查询请求消息

属　　性
呼叫名称
连接名称
安全

8.4.3.2 连接查询响应—查询每条连接

连接查询响应消息(connQueryResponse)用来响应连接查询请求。连接查询响应消息中的属性如表33所示。

表 33 E-NNI 连接查询响应消息

属　　性
呼叫名称
连接名称 SNP ID SNPP ID
方向
显式资源列表
恢复
状态

8.4.3.3 连接查询请求—查询所有连接

查询所有连接的请求消息(connQueryAllRequest)用来查询与网络呼叫控制器名称对应的所有存在的连接。查询请求消息的属性如表 34 所示。

表 34 E-NNI 所有连接查询请求消息

名　　称
NCC 名称
安全

8.4.3.4 连接查询响应—查询所有连接

查询所有连接的响应消息(connQueryAllResponse)用来响应所有连接的查询请求。响应消息可以由用户或网络来发送。响应消息中的属性信息如表 35 所示。

表 35 E-NNI 所有连接查询响应消息

属　　性
NCC 名称
每个呼叫的名称： 相关连接名称列表
状态

8.4.4 通知消息

通知消息(notification)用来允许对连接状态相关的信息进行交换。发送通知消息用来对呼叫或连接进行通知。通知消息相关的属性如表 36 所示。

表 36 E-NNI 通知消息

属　　性
呼叫名称
连接名称
出错编码
安全

通知消息只需要一个步骤。

9 分布式连接管理(DCM)状态图

9.1 分布式连接管理状态定义

本部分详细介绍了连接控制器实体基于状态转移事件的状态转移图。本部分所规范的状态图应用于每个CC,另外还描述了每条连接的状态迁移过程,以及每个通信实体从一个状态迁移到另外一个状态时所发生的事件和动作。

状态转换为下列实体指定:

——发起方用户:发起某个操作的用户。连接建立操作时为A端用户(CCC-a)。释放操作时可以为A端或Z端用户。

——终结方用户:终结一个呼叫的用户。呼叫建立操作时为Z端用户(CCC-z)。呼叫释放操作时为Z端或A端用户。

——呼叫控制器。

——连接控制器。

呼叫操作相关的状态如下:

——空闲状态(S_i):该状态为缺省状态。在这种状态下,信令控制实体可以接受呼叫建立请求并进行对应的动作。

——验证呼叫建立请求状态(S_{svreq}):该状态下,呼叫控制器(CallC)验证呼叫建立请求(包括安全和策略验证)。

——呼叫建立请求发起状态(S_{sreq}):该状态下,CallC产生并发送呼叫建立请求消息,并等待建立请求死的响应。

——连接建立状态(S_{sconn}):该状态下,CC执行连接的建立以支持所接受的呼叫。

——呼叫建立接受状态(S_{sacpt}):该状态下,CallC产生并发送呼叫建立指示消息来响应呼叫建立请求,并等待呼叫最终建立的确认消息。

——验证呼叫状态(S_{svcall}):该状态下,CallC做呼叫是否已经成功建立的验证处理。

——激活状态(S_a):该状态下,呼叫建立过程已经完成,与呼叫对应的各个连接也已经建立并准备开始传送用户数据。

——验证呼叫释放请求状态(S_{rvreq}):该状态下,CallC对呼叫释放请求进行验证。验证通过后允许进行呼叫释放处理,验证不通过拒绝呼叫释放请求。验证包括权限验证、完整性验证,也可能包含策略验证。

——呼叫释放请求发起状态(S_{rreq}):该状态下,CallC产生并发送呼叫释放请求,并等待呼叫释放请求响应。

——释放连接状态(S_{rconn}):该状态下,CC执行连接释放来支持呼叫释放动作。

——信令故障(S_{sigerr}):该状态下,信令通信通道中断。

连接操作的相关状态操作如下,并描述了上述呼叫状态相关的连接状态的详细情况:

——空闲状态(S_i):该状态为缺省状态,在空闲状态下,信令通信实体可以接收连接建立请求并执行相应的动作。

——验证连接建立请求状态(S_{svreq}):该状态下,CC验证连接建立请求(如可能包括安全和策略验证)。

——连接建立请求发起状态(S_{sreq}):该状态下,CC产生并发送连接建立请求,并等待建立请求响应。

——连接建立接受状态(S_{sacpt}):该状态下,CC已经产生并发送了连接建立指示状态作为连接请求的响应,并等待连接最终建立的确认消息。

——验证连接状态(S_{svconn}):该状态下,CC验证连接是否已经正确建立。

——连接激活状态(S_a):该状态下,连接建立请求完成并向 CallC 发送通知消息指示连接建立已经完成。

——验证连接释放请求状态(S_{rvreq}):该状态下,CC 验证连接释放请求。验证通过后允许释放连接,否则拒绝连接释放请求。验证包括权限验证、完整性验证,也可以包括策略验证。

——连接释放请求发起状态(S_{rreq}):该状态下,CC 产生并发送连接释放请求,并等待连接释放请求响应。

——信令故障状态(S_{sigerr}):该状态下,信令通信通道中断。

可能导致状态转移的事件包括:

——外部触发事件,如用户决定请求一个呼叫,流量工程管理发起一个新的呼叫,连接故障导致请求释放一个呼叫等。这些事件可以通过用户接口或应用接口来触发。

——接收到消息的事件。

——验证结果事件。

——超时事件。

异常的或不可知的消息不会导致状态转移。这些消息会被忽略或向该消息的发送者回应一个出错指示的消息。

9.2 呼叫状态

9.2.1 呼叫状态事件

下面的事件和用户呼叫控制器(CallC)的呼叫状态有关,如表 37 所示。

表 37 用户 CallC 呼叫状态事件

事 件	事件描述
Unk	接收到无法识别或异常的消息
SetReq	用户呼叫控制器 CallC 接收到一个建立呼叫的请求
SetVer	用户呼叫控制器 CallC 验证呼叫建立请求成功
SetNVer	用户呼叫控制器 CallC 验证呼叫建立请求不成功
SetSuc	用户呼叫控制器 CallC 接收到呼叫已经成功建立的响应
SetNSuc	用户呼叫控制器 CallC 接收到呼叫建立不成功的响应
SetExp	呼叫建立定时器超时
RelReq	用户呼叫控制器 CallC 接收到释放呼叫的请求
RelVer	用户呼叫控制器 CallC 验证呼叫释放请求成功
RelNVer	用户呼叫控制器 CallC 验证呼叫释放请求失败
RelSuc	用户呼叫控制器 CallC 接收到呼叫释放成功的响应
RelExp	呼叫释放定时器超时
SigErr	检测到信令通信通道故障
SigNErr	信令通信通道故障修复

下面的事件和网络 CallC(主叫方或被叫方)呼叫状态有关,如表 38 所示。

表 38 网络 CallC 呼叫状态事件

事 件	事件描述
Unk	接收到无法识别或异常的消息
SetReq	用户呼叫控制器 CallC 接收到一个建立呼叫的请求
SetVer	用户呼叫控制器 CallC 验证呼叫建立请求成功
SetNVer	用户呼叫控制器 CallC 验证呼叫建立请求不成功

表 38（续）

事　件	事件描述
SetAcp	网络呼叫控制器接收到呼叫建立请求被接受的响应
SetNAcp	网络呼叫控制器接收到呼叫建立请求被拒绝的响应
SetCon	网络呼叫控制器接收到支持呼叫的连接建立成功的响应
SetNCon	网络呼叫控制器接收到支持呼叫的连接建立失败的响应
SetCallVer	网络呼叫控制器对已经建立的呼叫验证成功
SetCallNVer	网络呼叫控制器对已经建立的呼叫验证失败
SetExp	呼叫建立定时器超时
RelReq	网络呼叫控制器接收到呼叫释放的请求
RelVer	网络呼叫控制器对呼叫释放请求验证成功
RelNVer	网络呼叫控制器对呼叫释放请求验证失败
RelCon	网络呼叫控制器接收到支持呼叫的连接已经成功释放的响应
RelNCon	网络呼叫控制器接收到支持呼叫的连接没有成功释放的响应
RelExp	呼叫释放定时器超时
SigErr	检测到信令通信通道故障
SigNErr	信令通信通道故障修复

9.2.2　发起方用户呼叫控制器(CallC)呼叫状态

9.2.2.1　用于呼叫建立的发起方用户呼叫状态

下面的状态转移图应用于呼叫建立的发起用户的 CallC，如表 39 和图 34 所示。

表 39　源用户呼叫控制器呼叫建立状态转移

当前状态	事件	行　为	下一个状态
*	Unk	向消息发送者发送通知消息报告错误信息，状态维持不变	*
S_i	SetReq	验证请求	S_{svreq}
S_{svreq}	SetVer	发送呼叫建立请求消息 启动呼叫建立定时器(T_{call_setup})	S_{sreq}
S_{svreq}	SetNVer	通知呼叫发起者呼叫建立请求已经被拒绝	S_i
S_{sreq}	SetSuc	删除呼叫建立定时器(T_{call_setup}) 发送呼叫建立请求确认消息(可选)	S_a
S_{sreq}	SetNSuc	删除呼叫建立定时器(T_{call_setup})	S_i
S_{sreq}	SetExp	通知呼叫发起者呼叫建立超时 发送呼叫释放请求消息 启动呼叫释放定时器($T_{call\_release}$)	S_{rreq}
S_a	SigErr	无	S_{sigerr}
S_{sigerr}	SigNErr	无	S_a

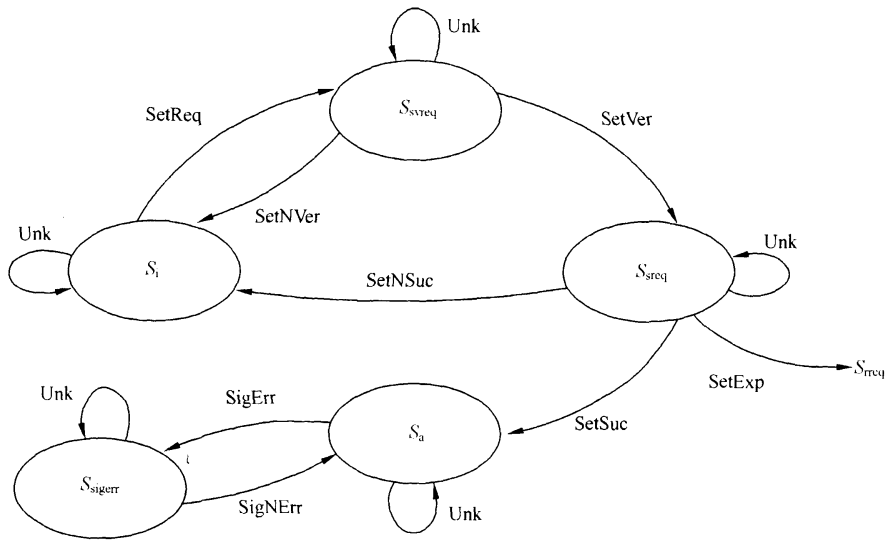


图 34　源用户呼叫控制器呼叫建立状态转移图

9.2.2.2　用于呼叫释放的发起端用户呼叫状态

下面的状态转移图应用于发起端用户 CallC 的呼叫释放情况，如表 40 和图 35 所示。

表 40　源用户呼叫控制器呼叫释放状态转移表

当前状态	事件	行　为	下一个状态
*	Unk	向消息发送者发送通知消息报告错误信息，状态维持不变	*
S_a	RelReq	验证请求	S_{rvreq}
S_{rvreq}	RelVer	发送呼叫释放请求消息 启动呼叫释放定时器($T_{call\_release}$)	S_{rreq}
S_{rvreq}	RelNVer	通知呼叫发起者呼叫释放请求被拒绝	S_a
S_{rreq}	RelSuc	删除呼叫释放定时器($T_{call\_release}$)	S_i
S_{rreq}	RelExp	通知呼叫发起者呼叫释放请求定时器超时	S_i

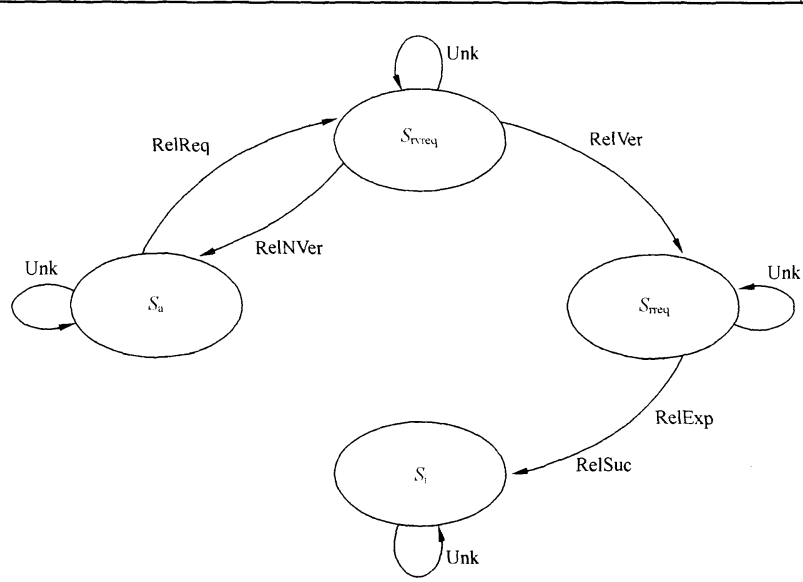


图 35　源呼叫控制器呼叫释放状态转移图

9.2.3 终结方用户呼叫控制器(CallC)呼叫状态

9.2.3.1 用于呼叫建立的终结方用户呼叫状态

下面的状态转移应用于终结端用户 CallC 的呼叫建立时的状态。如表 41 和图 36 所示。

表 41 目的用户呼叫控制器呼叫建立状态转移表

当前状态	事件	行 为	下一个状态
*	Unk	向消息发送者发送通知消息报告错误信息,状态维持不变	*
S_i	SetReq	验证请求	S_{svreq}
S_{svreq}	SetVer	发送呼叫已经被接受的消息 启动呼叫建立定时器(T_{call_setup})	S_{sacpt}
S_{svreq}	SetNVer	通知呼叫发起者呼叫建立请求被拒绝	S_i
S_{sacpt}	SetSuc	删除呼叫建立定时器(T_{call_setup})	S_a
S_{sacpt}	SetNSuc	删除呼叫建立定时器(T_{call_setup})	S_i
S_{sacpt}	SetExp	通知呼叫发起者呼叫建立请求定时器超时 发送呼叫释放请求消息 启动呼叫释放定时器($T_{call\_release}$)	S_{rreq}
S_a	SigErr	无	S_{sigerr}
S_{sigerr}	SigNErr	无	S_a

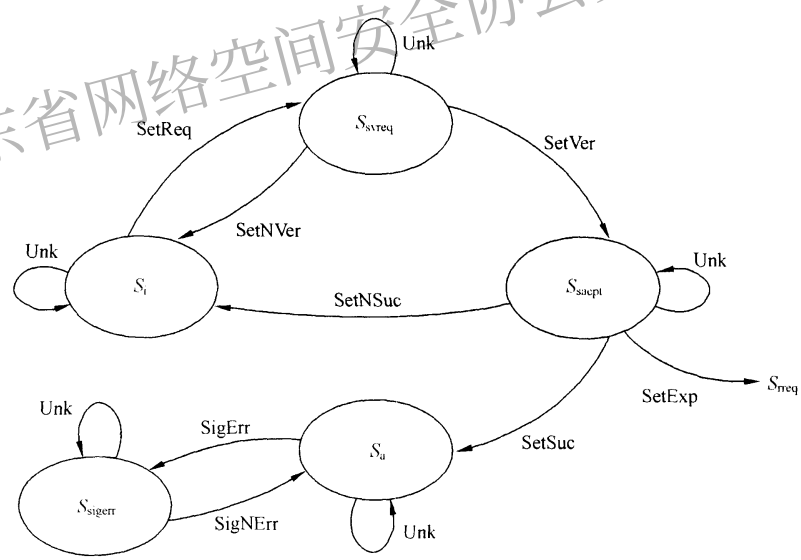


图 36 目的用户呼叫控制器呼叫建立状态转移图

9.2.3.2 用于呼叫释放的终结方用户呼叫状态

下面描述的状态转移应用于终端用户的 CallC 呼叫释放情况,如表 42 和图 37 所示。

表 42 目的用户呼叫控制器呼叫释放状态转移表

当前状态	事件	行 为	下一个状态
*	Unk	向消息发送者发送通知消息报告错误信息,状态维持不变	*
S_a	RelReq	无	S_{rreq}
S_{rreq}	RelSuc	无	S_i

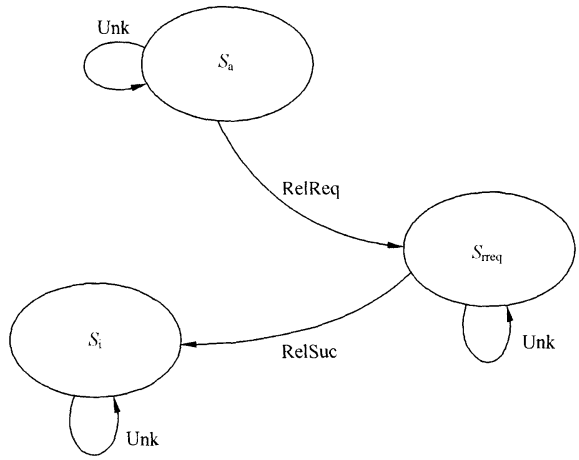


图 37　目的用户呼叫控制器呼叫释放状态转移图

9.2.4　网络呼叫控制器(CallC)呼叫状态

9.2.4.1　用于呼叫建立的网络呼叫控制器(CallC)呼叫状态

下面的状态转移应用于网络呼叫控制器 CallC 的呼叫建立，如表 43 和图 38 所示。

表 43　网络呼叫控制器呼叫建立状态转移表

当前状态	事件	行　为	下一状态
*	Unk	向消息发送者发送通知消息报告错误信息，状态维持不变	*
S_i	SetReq	验证请求	S_{svreq}
S_{svreq}	SetVer	发送呼叫建立请求消息 启动呼叫建立定时器(T_{call_setup})	S_{sreq}
S_{svreq}	SetNVer	通知呼叫发起者呼叫建立请求被拒绝	S_i
S_{sreq}	SetAcp	删除呼叫建立定时器(T_{call_setup}) 发送呼叫建立已经接受的消息 发起连接建立处理(对主叫方呼叫控制器) 启动连接建立定时器(T_{conn_setup})	S_{sconn}
S_{sreq}	SetNAcp	删除呼叫建立定时器(T_{call_setup}) 通知呼叫发起者呼叫建立请求被拒绝	S_i
S_{sreq}	SetExp	通知呼叫发起者呼叫建立请求被拒绝	S_i
S_{sconn}	SetCon	删除连接建立定时器(T_{conn_setup})	S_{svcall}
S_{sconn}	SetNCon	通知呼叫发起者呼叫建立请求被拒绝	S_i
S_{sconn}	SetExp	通知呼叫发起者呼叫建立请求被拒绝 发起连接释放处理(对主叫方呼叫控制器) 启动连接释放定时器($T_{conn\_release}$)	S_{rconn}
S_{svcall}	SetCallVer	发送呼叫已经建立的消息	S_a
S_{svcall}	SetCallNVer	通知呼叫发起者呼叫建立请求被拒绝 发起连接释放处理(对主叫方呼叫控制器) 启动连接释放定时器($T_{conn\_release}$)	S_{rconn}
S_a	SigErr	无	S_{sigerr}
S_{sigerr}	SigNErr	无	S_a

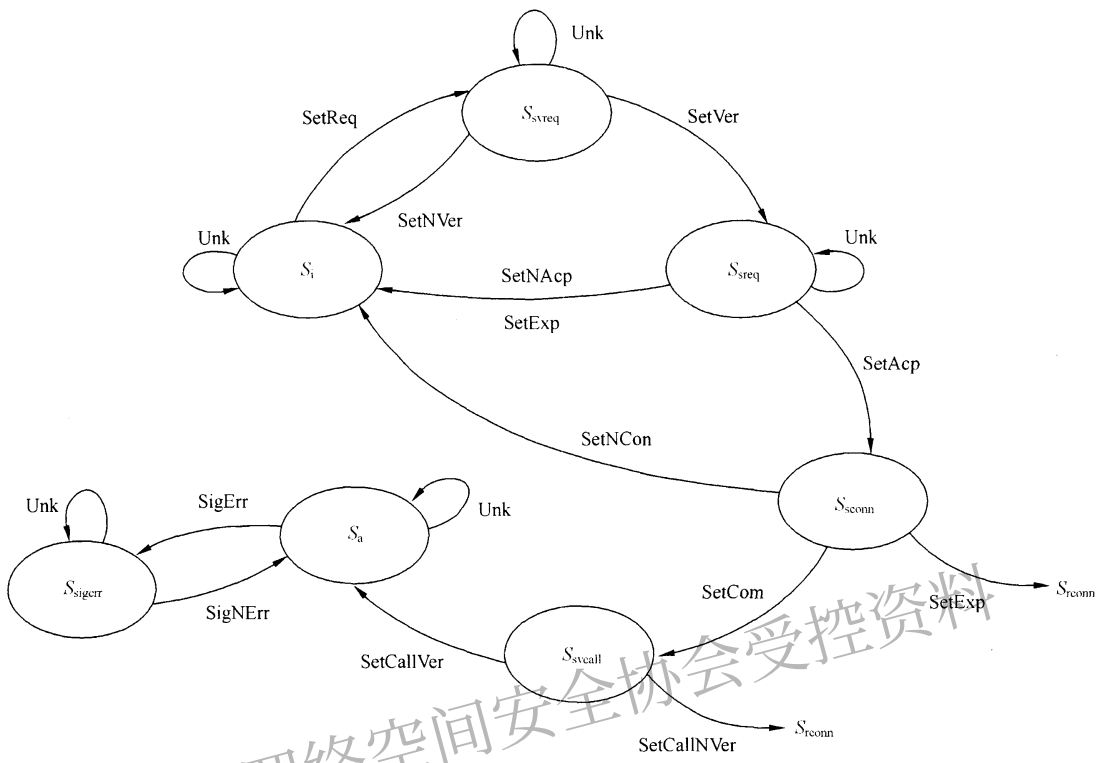


图 38　网络呼叫控制器呼叫建立状态转移图

9.2.4.2　用于呼叫释放的网络呼叫控制器(CallC)呼叫状态

下面的状态转移情况适用于 CallC 的呼叫释放时的状态变化情况，如表 44 和图 39 所示。

表 44　网络呼叫控制器呼叫释放状态转移表

当前状态	事件	行　为	下一状态
*	Unk	向消息发送者发送通知消息报告错误信息，状态维持不变	*
S_a	RelReq	验证请求	S_{rvreq}
S_{rvreq}	RelVer	发送呼叫释放消息 发起连接释放处理(对主叫方呼叫控制器) 启动连接释放定时器($T_{conn\_release}$)	S_{rconn}
S_{rvreq}	RelNVer	通知呼叫发起者呼叫释放请求被拒绝	S_a
S_{rconn}	RelCon	删除呼叫释放定时器($T_{call\_release}$) 通知呼叫控制器呼叫已经被释放	S_i
S_{rconn}	RelNCon，RelExp	通知呼叫发起者呼叫释放请求定时器超时	S_i

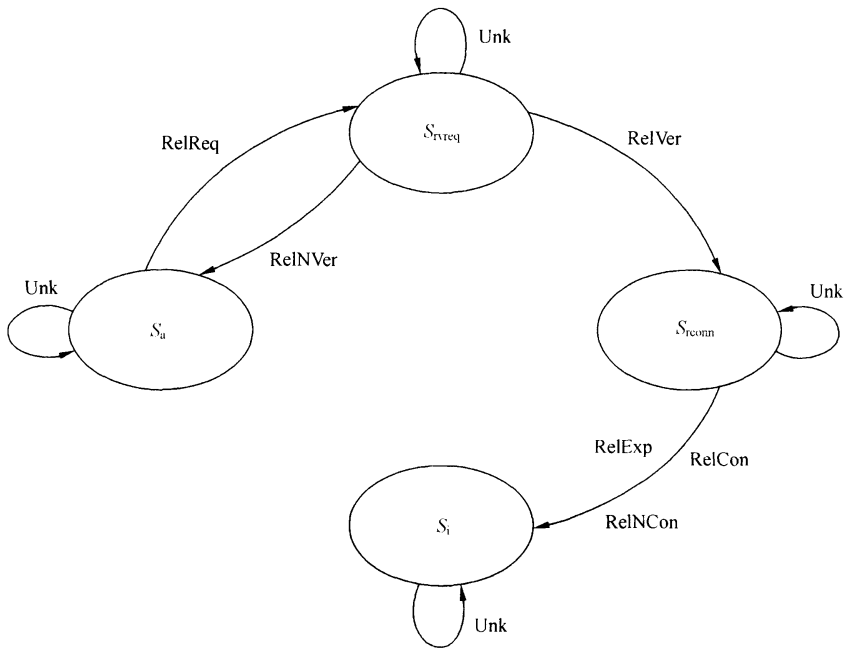


图 39　网络呼叫控制器呼叫释放状态转移图

9.3　连接状态

9.3.1　连接控制器状态事件

下面的事件与用户连接控制器 CC 的连接状态相关，如表 45 所示。

表 45　用户连接控制器状态事件

事件	事件描述
Unk	接收到无法识别或异常的消息
SetReq	用户连接控制器 CC 接收到一个建立连接的请求
SetVer	用户连接控制器验证连接建立请求成功
SetNVer	用户连接控制器验证连接建立请求不成功
SetInd	用户连接控制器接收到连接建立请求已经成功处理的指示消息
SetNInd	用户连接控制器接收到连接建立请求没有成功处理的指示消息
SetCnfm	用户连接控制器接收到连接已经成功建立的确认消息
SetNCnfm	用户连接控制器接收到连接没有建立的确认消息
SetExp	连接建立定时器超时
RelReq	用户连接控制器接收到连接释放请求
RelVer	用户连接控制器验证连接释放请求成功
RelNVer	用户连接控制器验证连接释放请求不成功
RelInd	用户连接控制器接收到连接已经释放的指示信息
RelExp	呼叫释放定时器超时
SigErr	检测到信令通信通道故障
SigNErr	信令通信通道故障修复

9.3.2 源用户连接控制器或源网络连接控制器的连接状态

9.3.2.1 用于连接建立的源端连接控制器的连接状态

对于 SC 业务，下面的状态转移表适用于源用户的连接控制器。对于 SPC 业务，下面的状态转移表适用于源网络连接控制器，如表 46 和图 40 所示。

表 46 源连接控制器连接建立状态表

当前状态	事件	行 为	下一状态
*	Unk	向消息发送者发送通知消息报告错误信息，状态维持不变	*
S_i	SetReq	验证请求	S_{svreq}
S_{svreq}	SetVer	发送连接建立请求消息 启动连接建立定时器(T_{conn_setup})	S_{sreq}
S_{svreq}	SetNVer	通知连接发起者连接建立请求被拒绝	S_i
S_{sreq}	SetInd	删除连接建立定时器(T_{conn_setup}) 发送连接确认消息(可选)	S_a
S_{sreq}	SetNInd	删除连接建立定时器(T_{conn_setup})	S_i
S_{sreq}	SetExp	通知连接发起者连接建立请求超时 发送连接释放请求消息 启动连接释放定时器($T_{conn\_release}$)	S_{rreq}
S_a	SigErr	无	S_{sigerr}
S_{sigerr}	SigNErr	无	S_a

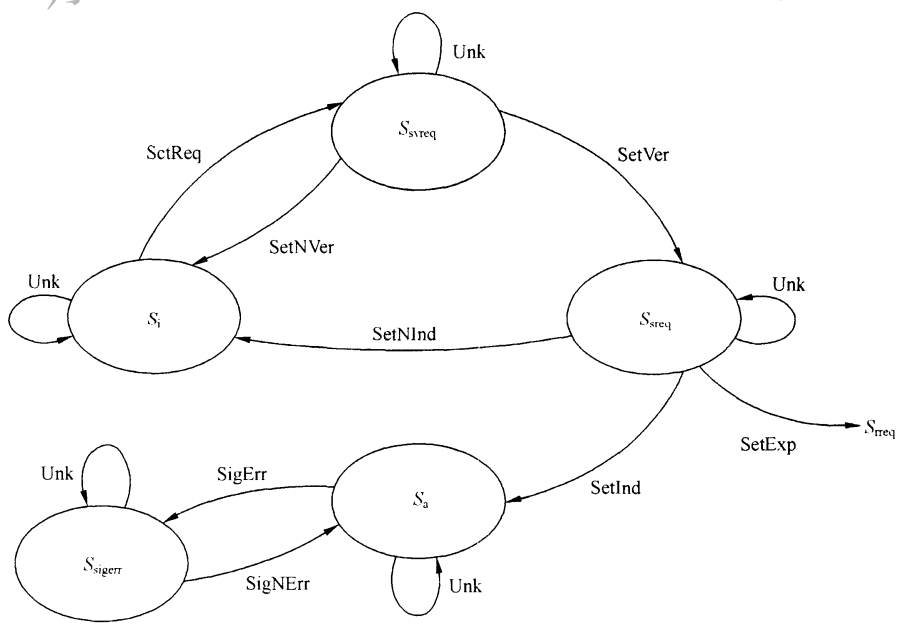


图 40 源连接控制器连接建立状态转移图

9.3.2.2 用于释放的源端连接控制器连接状态

下面所描述的状态转移图适用于源端连接控制器(CC)的连接释放时的状态变化，如表 47 和图 41 所示。

表 47　源连接控制器连接释放状态表

当前状态	事件	行　为	下一状态
*	Unk	向消息发送者发送通知消息报告错误信息，状态维持不变	*
S_a	RelReq	验证请求	S_{rvreq}
S_{rvreq}	RelVer	发送连接释放请求消息 启动连接释放定时器($T_{conn\_release}$)	S_{rreq}
S_{rvreq}	RelNVer	通知连接发起者连接释放请求被拒绝	S_i
S_{rreq}	RelInd	删除连接释放定时器($T_{conn\_release}$)	S
S_{rreq}	RelExp	通知连接发起者连接释放请求超时	S

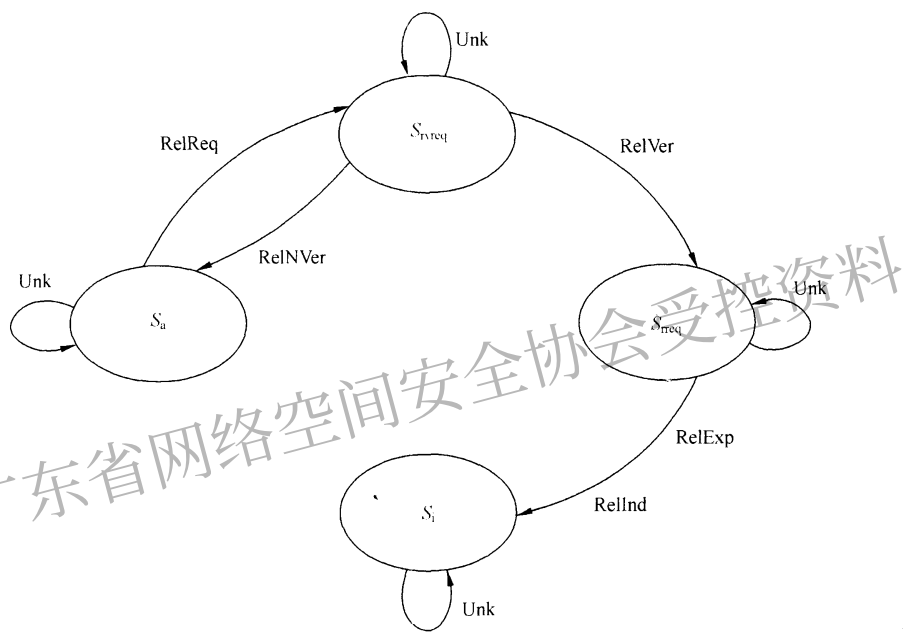


图 41　源连接控制器连接释放状态转移图

9.3.3　目的端用户连接控制器或目的端网络连接控制器连接状态

9.3.3.1　用于连接建立的目的端连接控制器连接状态

对于 SC 业务，下面的状态转移表适用于目的端用户 CC。对于 SPC 业务，下面的状态转移表适用于目的端网络 CC。

下面描述的状态转移情况适用于目的端连接控制器的连接建立过程时的状态转移情况，如表 48 和图 42 所示。

表 48　目的连接控制器连接建立状态表

当前状态	事件	行　为	下一状态
*	Unk	向消息发送者发送通知消息报告错误信息，状态维持不变	*
S_i	SetReq	验证请求	S_{svreq}
S_{svreq}	SetVer	发送连接指示消息 启动连接建立定时器(T_{conn_setup})	S_{sacpt}
S_{svreq}	SetNVer	通知连接发起者连接建立请求被拒绝	S_i
S_{sacpt}	SetCnfm	删除连接建立定时器(T_{conn_setup})	S_a

表 48（续）

当前状态	事件	行　为	下一状态
S_{sacpt}	SetNCnfm	删除连接建立定时器(T_{conn_setup})	S_i
S_{sacpt}	SetExp	通知连接发起者连接建立请求超时 发送连接释放请求消息 启动连接释放定时器($T_{conn\_release}$)	S_{rreq}
S_a	SigErr	无	S_{sigerr}
S_{sigerr}	SigNErr	无	S_a

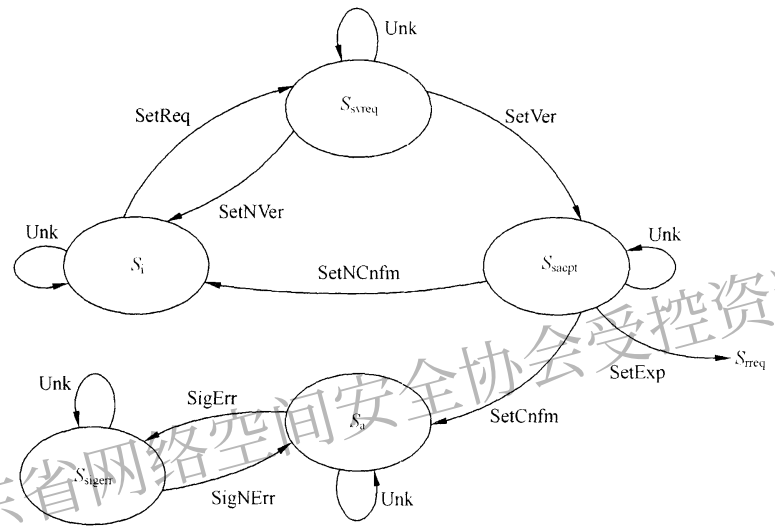


图 42　目的连接控制器连接建立状态转移图

9.3.3.2　用于连接释放的目的端连接控制器连接状态

下面描述的状态转移情况适用于目的端连接控制器的连接释放情况，如表 49 和图 43 所示。

表 49　目的连接控制器连接释放状态表

当前状态	事件	行　为	下一状态
*	Unk	向消息发送者发送通知消息报告错误信息，状态维持不变	*
S_a	RelReq	发连接指示消息	S_i

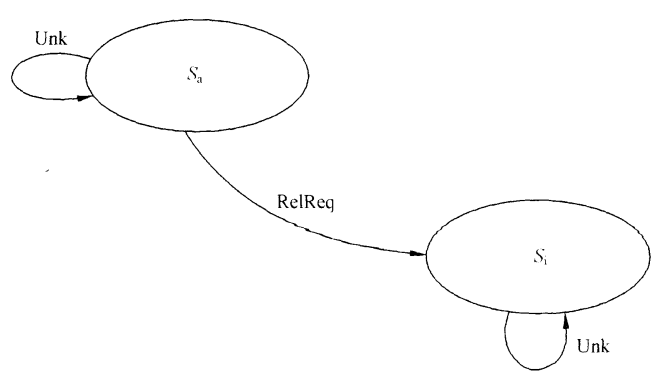


图 43　目的连接控制器连接释放状态转移图

9.3.4 中间节点连接控制器的连接状态

9.3.4.1 用于连接建立的中间节点网络连接控制器连接状态

对于 SC 业务，下面的状态转移表适用于网络连接控制器。对于 SPC 业务，下面的状态转移表适用于中间网络连接控制器。

下面所描述的状态转移适用于中间节点 CC 连接建立过程时的状态变化情况，如表 50 和图 44 所示。

表 50 中间网络连接控制器连接建立状态表

当前状态	事件	行 为	下一状态
*	Unk	向消息发送者发送通知消息报告错误信息，状态维持不变	*
S_i	SetReq	验证请求	S_{svreq}
S_{svreq}	SetVer	发送连接建立请求消息 启动连接建立定时器(T_{conn_setup})	S_{sreq}
S_{svreq}	SetNVer	通知连接发起者连接建立请求被拒绝	S_i
S_{sreq}	SetInd	删除连接建立定时器(T_{conn_setup}) 发送连接指示消息 启动连接建立定时器(T_{conn_setup})	S_{sacpt}
S_{sreq}	SetNInd	删除连接建立定时器(T_{conn_setup}) 通知连接发起者连接建立请求被拒绝	S_i
S_{sreq}	SetExp	通知连接发起者连接建立请求被拒绝 发送连接释放请求消息 启动连接释放定时器($T_{conn\_release}$)	S_{rrec}
S_{sacpt}	SetCnfm	删除连接建立定时器(T_{conn_setup})	S_{svconn}
S_{sacpt}	SetNCnfm	删除连接建立定时器(T_{conn_setup}) 通知连接发起者连接建立请求被拒绝	S_i
S_{sacpt}	SetExp	通知发起者连接建立请求被拒绝 发送连接释放请求消息 启动连接释放定时器($T_{conn\_release}$)	S_{rreq}
S_{svconn}	SetConnVer	发送连接确认消息	S_a
S_{svconn}	SetConnNVer	通知连接发起者连接建立请求被拒绝 发连接释放请求消息 启动连接释放定时器($T_{conn\_release}$)	S_{rreq}
S_a	SigErr	无	S_{sigerr}
S_{sigerr}	SigNErr	无	S_a

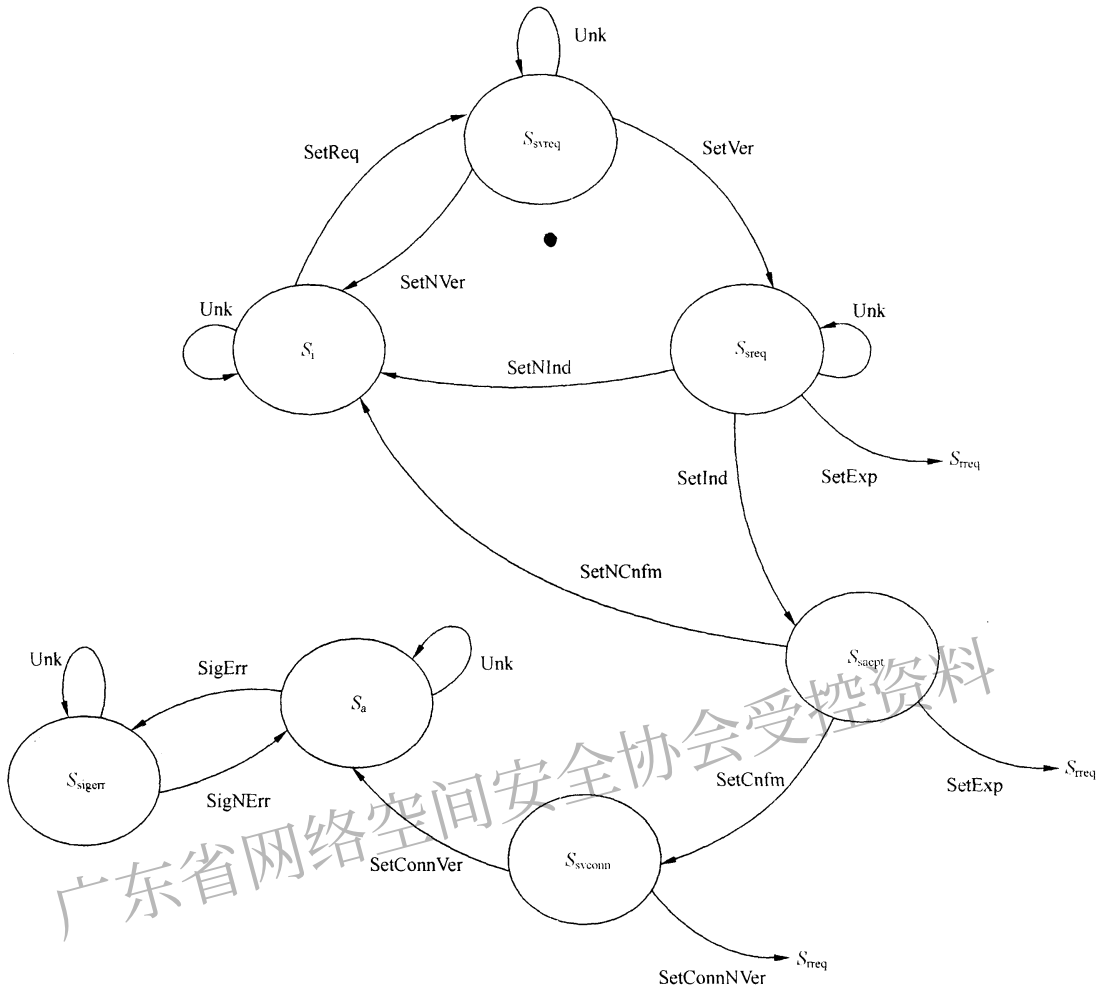


图 44　中间连接控制器连接建立状态转移图

9.3.4.2　用于连接释放的中间节点连接控制器连接状态

下面描述的状态转移适用于中间节点连接控制器连接释放时的状态变化情况，如表 51 和图 45 所示。

表 51　中间网络连接控制器连接释放状态表

当前状态	事件	行　为	下一状态
*	Unk	向消息发送者发送通知消息报告错误信息，状态维持不变	*
S_a	RelReq	验证请求	S_{rvreq}
S_{rvreq}	RelVer	发送连接释放请求消息 启动连接释放请求定时器($T_{conn\_release}$)	S_{rreq}
S_{rvreq}	RelNVer	通知连接发起者连接释放请求被拒绝	S_a
S_{rreq}	RelInd	删除连接释放定时器($T_{conn\_release}$) 发送连接指示消息	S_i
S_{rreq}	RelExp	通知连接发起者连接释放请求超时	S_i

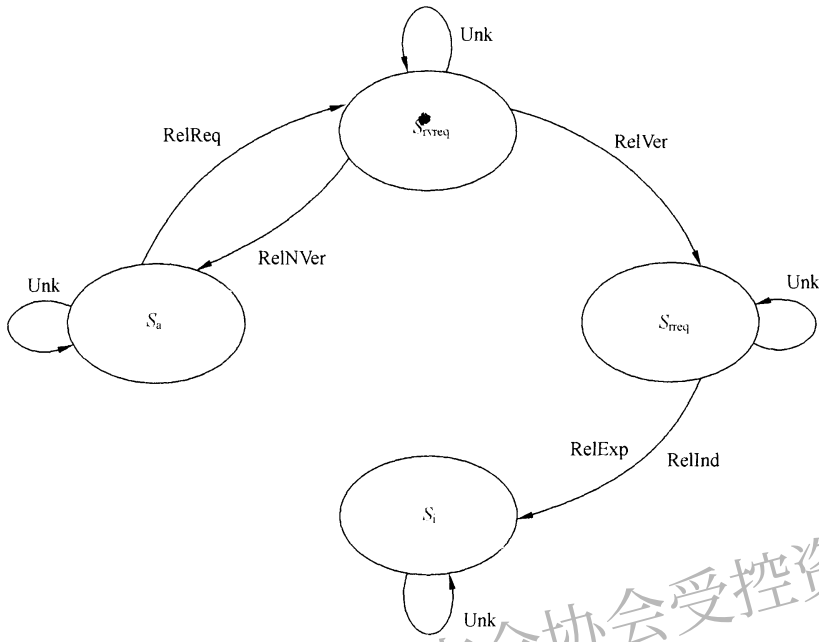


图 45　中间网络连接控制器连接释放状态转移图

10　呼叫和连接控制器的功能管理

10.1　呼叫和连接控制管理功能定义

本部分定义了以下功能：

——管理平面的功能应支持故障管理、配置管理(包括资源分配和去分配)、性能管理、安全管理和计费管理能力；

——传送平面支持净荷传送、性能监视、故障检测和保护倒换功能；

——控制平面支持动态通道计算、动态分布呼叫和连接的建立或释放、动态保护和恢复的指配或分配、恢复功能。

呼叫控制器、连接控制器和链路资源管理器元件提供了对由分布式连接建立和释放功能建立和释放的呼叫进行监视和管理的能力。作为对这些请求进行监视和管理的一部分，呼叫控制器(CallC)、连接控制器(CC)、链路资源管理器(LRM)以及管理平面之间需要发生通信。元件与管理平面(MP)之间的通信包括元件配置和确保元件健全的特定信息交换。为了能够支持管理信息信号在元件与 MP 之间交换信息，要包括元件的各种行为。图 46 说明了元件与 MP 之间的接口，这里描绘的 MI 信息仅用于 DCM 功能。

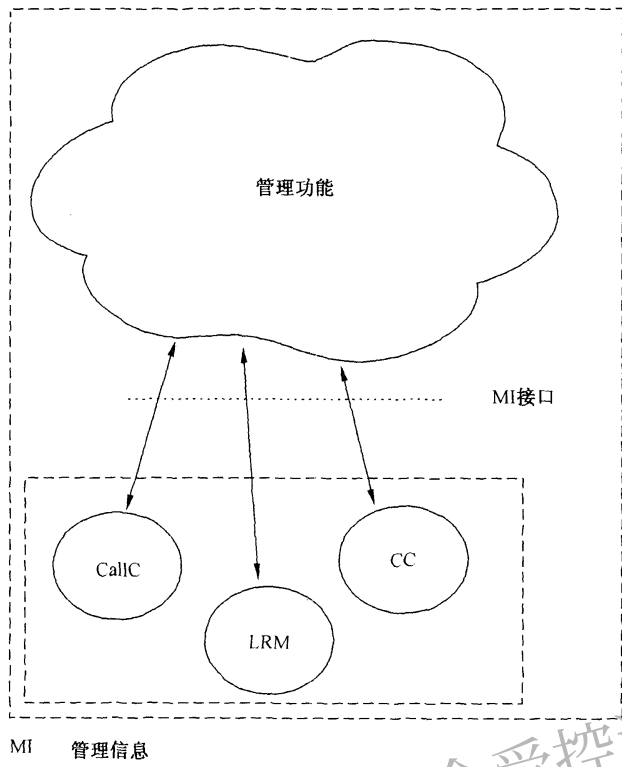


图 46 控制平面元件与管理平面之间的接口

管理系统能够设置有关元件行为的特定信息，并且能将这些信息传递给元件。元件还会报告其控制信息。这些由元件报告的信息会被分类并由元件自动传递给管理系统，或者通过管理系统查询元件获得该信息。在这种通信方式下，可以控制自动报告功能是使能还是禁止。要注意的是 MP 应能始终获得状态信息，甚至在状态报告功能被禁止的情况下也应如此。以下是由管理功能控制的可供选择的 MI 信号：

——元件状态：呼叫控制器(CallC)状态报告、CC 状态报告和 LRM 状态报告允许管理平面为特定的元件指定状态报告功能为使能；而 CallC 操作状态、CC 操作状态和 LRM 操作状态独立提供从元件到管理平面有关元件健全性的信息(例如“使能”为执行该功能，“禁止”为禁止执行该功能)。CallC 问题列表、CC 问题列表和 LRM 问题列表独立提供有关元件健全性和相应接口的详细信息(如在元件接口上或元件内部检测出的问题列表)。如果管理平面不具备状态报告信号，则默认为“禁止”状态。

——应用的信令机制：当信令协议控制器的接口支持多信令机制时(例如：RSVP-TE、CR-LDP、PNNI)，MI 信号 DCM 信令模式可以让管理平面决定信令协议控制器应用哪种信令机制。协议控制器还可以指定特定的信令机制，但管理平面不考虑协议控制器所指定的结果。

以下 MI 信号由元件来控制：

——元件性能的跟踪：CallC 和 CC 提供呼叫和连接请求的监视和管理功能。同样的，这些元件会跟踪各种不同的参数，包括业务利用率信息和呼叫状态信息。其他类型信息例如连接利用率还没有加入这些元件中，因此这里没有描述。如果元件提供连接请求的跟踪功能，那么 MI 信号应支持控制平面的管理查询。以下定义了 CallC、CC 和 LRM 元件的 MI 信号。

- MI CallC 呼叫周期：提供源于 CallC 的呼叫持续期的跟踪。
- MI CallC 呼叫状态：提供指定信号状态信息。
- MI CC 连接状态：提供指定连接的状态信息。
- MI LRM 连接状态：提供指定链路连接的状态信息。

- MI CallC 呼叫尝试：提供 CallC 接收的呼叫建立请求的数目信息。
- MI CallC 呼叫阻塞：提供 CallC 接收的呼叫建立请求被阻塞的数目信息。

——增强的呼叫监视功能：《ASON 标准第一部分》定义的连接没有任何监视能力，而增强的连接具有连接的监视能力。CC 的监视能力和影响效果待研究。

除了以上这些 MI 信号之外，也支持在控制平面和管理平面之间的其他类型的通信。这些类型包括在建立和释放连接期间的通信。以下部分主要描述了与建立和释放相关的处理过程，和在不同连接操作阶段应用的管理通信。

表 52 总结了本部分定义的可供选择的 MI 信息。

表 52 可供选择的 MI 信息

MI 信息	MI 信息(英文)	信息属性
MI CallC 状态报告 MI CC 状态报告 MI LRM 状态报告	MI_CallCstatusReporting MI_CCstatusReporting MI_LRMstatusReporting	使能或禁止
MI CallC 操作状态 MI CC 操作状态 MI LRM 操作状态	MI_CallCoperationalState MI_CcoperationalState MI_LRMoperationalState	上或下
MI CallC 问题列表 MI CC 问题列表 MI LRM 问题列表	MI_CallCproblemList MI_CCproblemList MI_LRMproblemList	问题例如“对等元件通信失败”、“过量差错说明消息”
MI DCM 信令模式	MI_DCMsigMode	如 RSVP-TE、CR-LDP、PNNI
MI CallC 呼叫周期	MI_CallCcallDuration	时间域、开始时间和日期、终止时间和日期
MI CallC 呼叫状态	MI_CallCcallState	参见第 9 节定义的状态
MI CC 连接状态	MI_CCconnectionState	参见第 9 节定义的状态
MI LRM 连接状态	MI_LRMconnectionState	参见《ASON 标准第一部分》中定义的状态
MI CallC 呼叫尝试	MI_CallCcallAttempt	每个时间周期呼叫建立请求的总数
MI CallC 阻塞呼叫	MI_CallCcallBlocked	呼叫请求阻塞的总数
MI 呼叫建立请求	MI_CallSetupRequest	属性与呼叫建立请求消息属性相同
MI 呼叫建立指示	MI_CallSetupIndication	属性与呼叫建立指示消息属性相同
MI 请求建立新路由	MI_requestSetupNewRoute	呼叫名称 连接名称 A 端用户名称 Z 端用户名称 SNP ID SNPP ID 方向性 CoS GoS

表 52（续）

MI 信息	MI 信息(英文)	信息属性
MI 新路由建立响应	MI_responseSetupNewRoute	呼叫名称 连接名称 SNP ID SNPP ID 方向性 显式资源列表 恢复
MI 认证 CAC	MI_verifyCAC	呼叫名称 连接名称 A 端用户名称 Z 端用户名称 SNP ID SNPP ID CoS GoS 显式资源列表 恢复
MI 响应 CAC	MI_responseCAC	接受或拒绝
MI 资源预留建立响应	MI_responseSetupResourcesReserved	预留或不预留
MI 资源分离建立响应	MI_responseSetupResourcesAllocated	分配或未分配
MI 呼叫资源失效响应	MI_responseCallResourceFail	失效的资源
MI 呼叫释放请求	MI_CallReleaseRequest	属性与呼叫释放请求消息属性相同
MI 呼叫释放指示	MI_CallReleaseIndication	属性与呼叫释放指示消息属性相同
MI 现存路由请求	MI_requestExistingRoute	呼叫名称 连接名称
MI 现存路由响应	MI_responseExistingRoute	呼叫名称 连接名称 SNP ID SNPP ID 方向 显式资源列表 恢复
MI 释放资源去分配响应	MI_responseReleaseResourcesDeallocated	去分配
MI 释放错误	MI_releaseError	连接释放的状态编码与 7.4.2.1 中定义的相同

10.2 建立连接

以下 MI 信号列表描述了管理功能与控制平面功能之间建立连接进行的交互过程。

——支持两种类型的连接：交换连接(SC)和软永久连接(SPC)。在 SPC 方式下，由管理平面发出连接建立请求，MI 呼叫建立请求消息会通过 MI 接口。

——在接收到建立请求后，会经过不同方式完成内部处理来创建相应的资源。它包括请求优先级的处理、最终路由信息的处理、资源或路由的验证信息处理和资源的分配处理（如果业务需要，资源的分配处理还会包括保护或恢复的分配）。要注意的是有些处理过程不必要求，因为管理系统可以完成这些处理。当 CC 请求一个 RC（路由控制器）的出口路由时（RC 分布在管理平面中），路由信息的通信还需要一些附加的消息，如 MI 请求建立新路由和 MI 建立新路由响应信号。要注意的是为了支持恢复和多样性约束条件，还要在信号中传送多个路由，也就是说，显式的资源列表属性会包含保护或不同连接的多个路由。

当管理平面具有 CAC 功能时，需要传送有关该功能的一些附加消息，如 MI 认证 CAC 和 MI 响应 CAC。

——需要完成相关请求的验证、资源的预留等处理过程。需要传送 CC 有关资源已被预留的消息 MI 资源预留建立响应。

——需要完成相关请求的验证、资源分配等处理过程。需要传送 CC 有关资源已被分配的消息 MI 资源分配建立响应。

——一旦 CC 完成处理以后，连接请求继续到下游的 CC。下游的 CC 由 RC 建立的路由来决定。完成请求之后，下游的 CC 收到一个请求状态的响应（例如确认或拒绝）。

- 如果接收的响应为被拒绝，所有通向入口的节点（例如呼叫发起节点）需要完成取消分配资源的附加处理。

——在确定了连接操作状态之后会向 CC 发送一个响应消息，发送方可能是上游 CC、网络 CallC，对于 SPC 来说也可能是管理平面。在 SPC 方式下，连接建立响应发送给管理平面。MI 呼叫建立指示信号需要通过 MI 接口传送。

- 作为 MI 呼叫建立指示消息的一部分，如果连接没有被成功建立或出现导致局部建立的错误时，详细的错误信息会发送给管理平面。错误信息包括错误原因和已建立的但不能由控制平面释放的所有局部链路（或链路连接）。

——在建立呼叫连接之后，CallC 通过 CC 和 LRM 监视连接。如果连接失败必须通知 CallC，以便采取相应措施，例如对该呼叫进行恢复或者中断使用该呼叫（也就是呼叫释放事件）。需要将这种 MI 呼叫资源失效响应消息传送给 CallC。

10.3 释放连接

为了释放一个呼叫，在不同元件的内部和外部需要进行信息交换。

——支持两种连接类型：交换连接（SC）和软永久连接（SPC）。在 SPC 方式下，由管理平面发出连接释放请求。MI 呼叫释放请求需通过 NMI 接口传送。

——CC 接收到释放请求以后，CC 请求 RC 索取已存在连接的路由。在 RC 功能分配给管理平面的情况下，需要 MI 现存路由请求消息和 MI 现存路由响应消息来传送路由信息。

——在请求验证之后，就会对资源进行去预留和去分配处理。附加的消息 MI 释放资源去分配响应用来传送资源去分配信息。

——一旦 CC 完成处理之后，连接请求继续传送到下游的 CC。下游 CC 由 RC 指定的路由决定。请求完成的同时，会收到由下游 CC 发送的有关请求状态的响应（确认被拒绝）。

- 如果收到被拒绝的响应，需要经过附加处理过程通知管理平面释放操作被拒绝的原因。发送 MI 释放错误消息通知管理平面控制平面无法成功释放存在的呼叫。

——确定连接操作状态的同时，请求的元件会收到一个响应。该元件可能是上游 CC、网络呼叫控制器（CallC），对于 SPC 来说也可能是管理平面。在 SPC 方式下，会向管理平面发送一个呼叫释放响应。呼叫释放响应消息 MI 呼叫释放指示需通过 MI 接口传送。

- 作为 MI 呼叫释放指示信号的一部分，如果呼叫没有被释放，或者出现导致局部释放的错误，错误的详细信息会发送到管理平面。该信息包括错误原因和所有无法释放的局部链路（或链路连接）。

11 信令协议的选择

对于 ASON 网络内的所有控制域，域间信令协议应与域内信令协议的选择无关，UNI、I-NNI 和 E-NNI 接口的信令协议选择应相互独立。

a) UNI 接口涉及 ASON 网络与客户设备的互联互通，因此应采用统一的信令协议 RSVP-TE。UNI 接口的信令协议应符合《自动交换光网络(ASON)技术要求　第 5 部分：用户—网络(UNI)接口》的要求。

b) E-NNI 接口涉及 ASON 网络中不同域之间的互联互通，因此应采用统一的信令协议 RSVP-TE。E-NNI 信令协议应符合《自动交换光网络(ASON)技术要求　第 9 部分：外部网络—网络(E-NNI)接口》的要求。

c) I-NNI 接口为域内部接口，一般由单一供应商的设备组成。因此 I-NNI 接口信令协议，可以采用 PNNI、RSVP-TE、CR-LDP 三种协议的任何一种，应分别符合 ITU-T G.7713.1、G.7713.2、和 G.7713.3 的要求。

d) UNI、E-NNI 和 I-NNI 接口如果选择了不同的信令协议，应在边界节点的协议控制器上提供不同协议转换的网关功能。

12 基于 GMPLS RSVP-TE 信令流程

12.1 RSVP-TE 消息类型

RSVP 消息类型如表 53 所示。

表 53 RSVP 消息类型值

值	消息类型
1	路径消息(Path)
2	预留消息(Resv)
3	路径错误消息(PathErr)
4	预留错误消息(ResvErr)
5	路径拆除消息(PathTear)
7	预留确认消息(ResvConf)
13	响应消息(Ack)
15	刷新消息(Srefresh)
20	握手消息(Hello)
21	通知(Notify)

RSVP 消息在 UNI 接口和 DCM 消息的对象关系如表 54 所示。

表 54 GMPLS RSVP-TE 消息到 DCM UNI 消息的映射

消息类型	UNI 消息	GMPLS RSVP-TE 消息
呼叫建立消息	呼叫建立请求 CallSetupRequest	路径(Path)
	呼叫建立指示 CallSetupIndication	预留(Resv)，路径错误(PathErr)，预留错误(ResvErr)
	呼叫建立确认 CallSetupConfirm	预留确认(ResvConf)

表 54（续）

消息类型	UNI 消息	GMPLS RSVP-TE 消息
呼叫释放消息	呼叫释放请求 CallReleaseRequest	路径(Path)或预留(Resv)(w/D&R 比特)或路径(Path)或预留(Resv)(w/A&R bit)
	呼叫释放指示 CallReleaseIndication	路径错误(PathErr)(Path 状态移除标志)或路径拆除(PathTear)，预留错误(ResvErr)
呼叫查询消息	呼叫查询请求 CallQueryRequest	路径(Path)(在 RSVP-TE 中通过周期性的刷新实现)
	呼叫查询指示 CallQueryIndication	预留(Resv)(在 RSVP-TE 中通过周期性的刷新实现)
呼叫通知消息	呼叫通知 CallNotify	通知(Notify)，包括路径错误(PathErr)

RSVP 消息在 E-NNI 接口和 DCM 消息的对象关系如表 55 所示。

表 55 GMPLS RSVP-TE 消息到 DCM E-NNI 消息的映射

消息类型	E-NNI 消息	GMPLS RSVP-TE 消息
呼叫建立消息	连接建立请求 ConnectionSetupRequest 连接建立指示 ConnectionSetupIndication 连接建立确认 ConnectionSetupConfirm	路径(Path) 预留(Resv)，路径错误(PathErr)，预留错误(ResvErr) 预留确认(ResvConf)
呼叫释放消息	连接释放请求 ConnectionReleaseRequest 连接释放指示 ConnectionReleaseIndication	Path 或 Resv(w/D&R bit)或 Path 或 Resv(w/A&R bit) PathErr(Path 状态移除标志)或 PathTear，ResvErr
连接查询消息	连接查询请求 ConnectionQueryRequest 连接查询指示 ConnectionQueryIndication	Path(在 RSVP-TE 中通过周期性的刷新实现) Resv(在 RSVP-TE 中通过周期性的刷新实现)
连接通知消息	连接通知 ConnectionNotify	通知(Notify)，包括 PathErr，ResvErr

12.2 GMPLS RSVP-TE 功能结构

12.2.1 RSVP-TE 协议机制

资源预约协议(RSVP)为 IETF 定义的协议(RFC2205)，该协议为 IP 数据流建立网络资源。RSVP 协议的定义由基本的处理流程，以及 IP 网络内部的信令消息和对象格式构成。流量工程扩展的 RSVP-TE(RFC3209)用来在 MPLS 网络中建立具有约束路由的连接。RSVP-TE 定义在 RSVP 基础上新增的处理流程以及消息和对象格式。GMPLS 信令扩展是基于基本的 MPLS 信令处理和抽象消息来满足处理不同类型的交换应用，如 TDM 交换、端口交换、波长交换等等。图 47 描述了 GMPLS RSVP-TE 定义的相关消息流的方向。

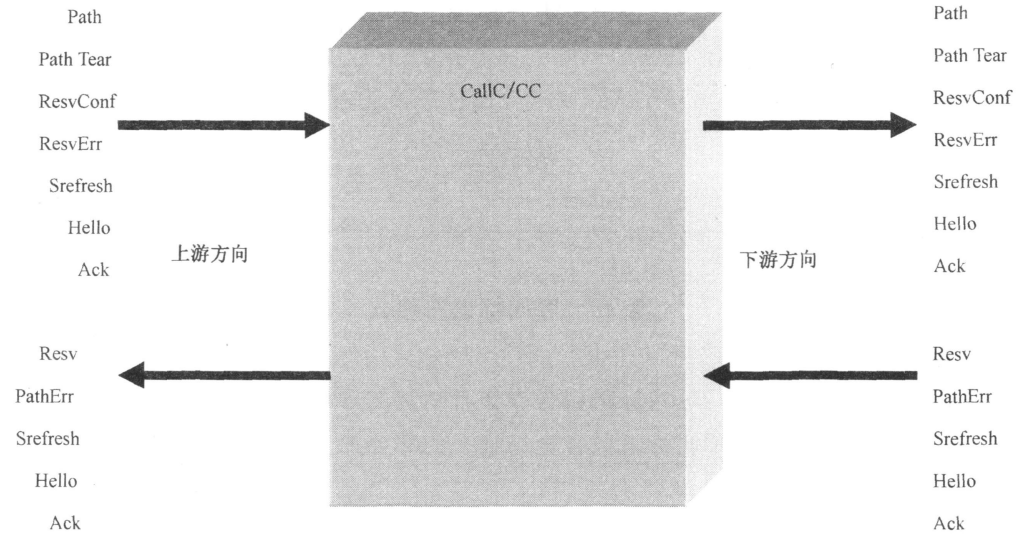


图 47　GMPLS RSVP-TE 消息流方向

GMPLS RSVP-TE 消息格式以 RFC2205 定义的基本结构为基础。RSVP 消息由通用消息头和与每种类型消息相关的一系列对象构成。通用消息头结构如图 48 所示。

0 1 2	7	8	15	16	23	24	31
版本	标志	消息类型		校验和			
发送 TTL		（保留）		RSVP 消息长度			

图 48　RSVP-TE 消息通用消息头结构

特定的消息类型扩展在 RFC2961 和 RFC3209 中有定义。各个域的定义如下：

——版本(4 比特)：协议版本编号，目前为版本号为 1。

——标志位(4 比特)：0x04-0x08(保留)，目前 4 个比特位没有定义。

——校验和：整个 RSVP 消息包的校验和，如果为 0，表示没有进行校验。

——发送 TTL：从源节点发出时，和对应的 IP TTL 值相同。经过 RSVP 处理节点处理后，该值减 1。RSVP 消息经过非 RSVP 网络时，IP TTL 值不断减少，但该值不发生变化。

——RSVP 消息长度：整个 RSVP 消息的长度(以字节为单位)。包括消息头长度和后面跟随的 RSVP 对象长度。

GMPLS RSVP-TE 的扩展定义了 GENERALIZED_UNI 对象来封装 A 端和 Z 端名字，以及在 UNI 接口支持业务请求所指定的 CoS 和 GoS。OIF UNI1.0 R2 定义 GENERALIZED_UNI 对象，并对该对象进行了扩展用来支持基本的呼叫概念并支持软永久连接(SPC)业务。

12.2.2　对呼叫标识符的支持

对 GMPLS RSVP-TE 进行扩展用来支持基本的呼叫模型。呼叫模型假定请求消息在同一个消息内同时处理一个呼叫以及该呼叫对应的连接，该消息在主叫呼叫控制器和网络呼叫控制器之间交互，同时也在网络呼叫控制器和被叫呼叫控制器之间交互。对一个呼叫添加或删除一条连接的处理由呼叫修改流程来完成。如对一条指定的连接属性进行修改。在呼叫修改的过程中，呼叫会话维持不变。可以通过呼叫标识对象(CALL_ID)来对一个已经建立的呼叫进行标识。呼叫标识对象的格式如图 49。

Class=230,C-Type=1

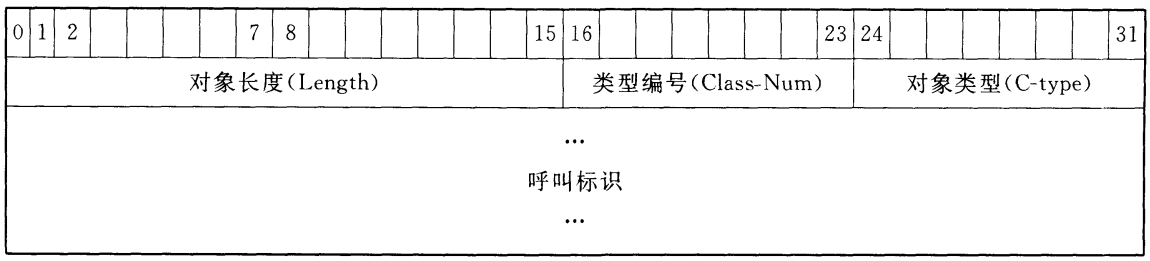


图 49　呼叫标识符格式

下面为 C-type 的定义：

——C-Type=1(操作者指定)：呼叫标识包含操作者指定的标识。

——C-Type=2(全局唯一)：呼叫标识包含全局唯一的部分加上操作者指定的标识部分。

下面定义了呼叫标识的结构：

——呼叫标识：长度×8-32 比特的标识符。呼叫标识的比特数必须为 32 比特的整数倍，最小为 32 比特。

全局唯一的呼叫标识结构是由全局唯一的固定 ID(由国家编码、运营商编码、唯一接入点编码构成)和操作者指定的 ID(操作者指定 ID 由源传送网元地址和本地标识构成)联合构成。

因此，通用全局唯一的呼叫 ID 包括全局 ID(国家编码＋运营商编码＋唯一接入点编码)和操作者指定 ID(源传送网元地址＋本地标识)。对于只需要操作者指定的呼叫 ID 只需要操作者指定 ID，但对于全局唯一的呼叫 ID 同时需要全局 ID 和操作者指定 ID。

全局 ID 由 3 字符国际分段码(国家编码)和 12 字符国家分段码(运营商编码＋唯一接入点编码)。这些字符根据 ITU-T T.50 建议进行编码。国际分段(IS)域提供 ISO 3166 中定义的 3 字符代表地理或政治国家编码。国家编码基于 3 个大写字母的国家编码(如 USA、FRA)。

国家分段(NS)域由两个子域构成：ITU 运营商编码和唯一接入点编码。ITU 运营商编码为分配给网络服务提供商的编码，由 ITU-T 维护。通信服务机构编码在 M.1400 中有定义。该编码由 1～6 位左起字母或字母加数字构成。唯一的接入点编码待分配国家编码和 ITU 运营商编码的组织进行统一分配，以保证编码的唯一性。该编码应该由 6～11 个字符构成，以字符 NULL 结束，来共同构成12 个字符的国家编码。

C-Type=1 时呼叫 ID 的格式如图 50 所示。

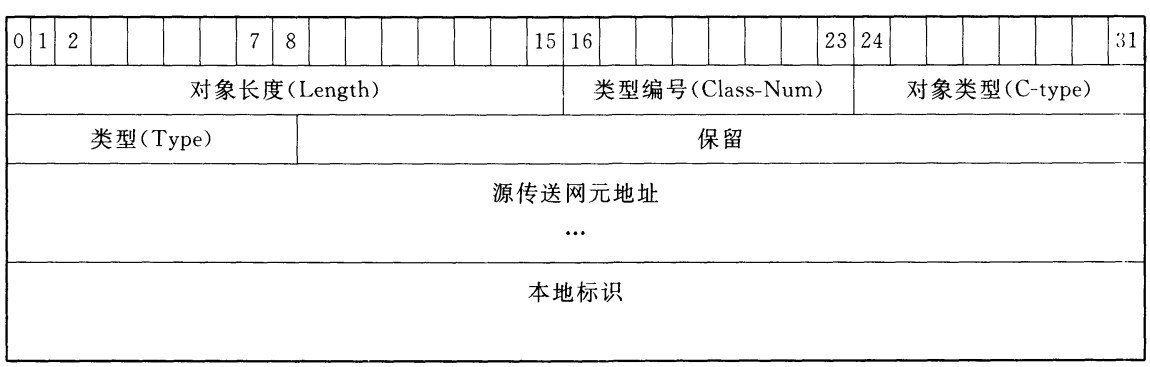


图 50　呼叫 ID 格式(C-Type=1)

C-Type=2 时呼叫 ID 的格式如图 51 所示。

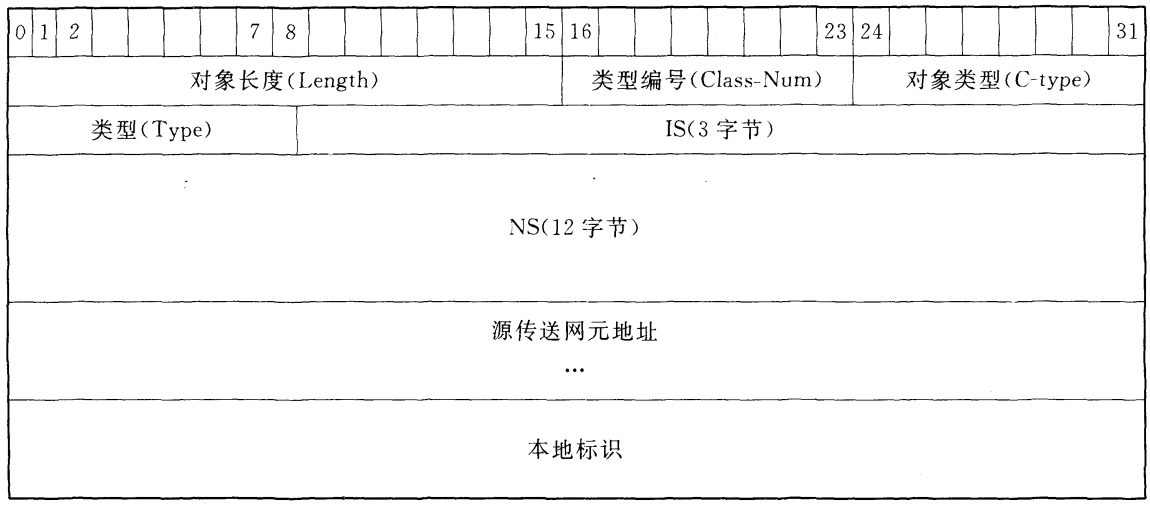


图 51　呼叫 ID 格式(C-Type=2)

——在两种情况下"类型"域用来指示源传送网元地址所使用的格式。此域中的不同值的意义如下：

类型=0x01，源传送网元地址为 4 字节；

类型=0x02，源传送网元地址为 16 字节；

类型=0x03，源传送网元地址为 20 字节；

类型=0x04，源传送网元地址为 6 字节；

类型=0x7f，源传送网元地址长度由运营商定义。

——源传送网元地址：由源网络控制的传送网元地址。

——本地标识：在呼叫的生命周期内保持不变的 64 比特标识。

如果源传送网元地址从全局唯一的地址空间中分配，操作者指定的呼叫 ID 也可以用来代替全局唯一的呼叫 ID。这个地址也可能从操作者指定的地址空间进行分配，因此不能保证全局唯一。

下面定义了对呼叫 ID 对象的处理规则：

——对于初始呼叫，主叫方或发起方呼叫控制器必须将呼叫 ID 的 C-Type 和呼叫标识值都置为 0。

——对于新的呼叫请求，源网络呼叫控制器设置适当的 C-Type 和呼叫 ID 值。

——对于存在的呼叫(呼叫 ID 值不为 0 的情况)，源网络呼叫控制器验证存在的呼叫。

——所有消息的呼叫 ID 对象从入口呼叫控制器发送到出口呼叫控制器，消息经过其他所有中间节时呼叫 ID 对象值不能改变。

——接收到请求的目的用户或客户使用呼叫 ID 值将源用户和自身之间的相关呼叫请求关联起来。关于该呼叫的后续动作使用该呼叫 ID 值来作为参考标识。

12.2.3　对软永久连接的支持

GMPLS RSVP-TE 扩展用来支持软永久连接(SPC)业务。SPC 业务假定用户网络分段连接在网络分段连接通过控制平面建立时就已经建立完成。当从外部(管理系统)收到初始请求后，已经假定控制平面有足够的信息来确定具体的要使用的目的链路连接(目的网络—用户之间链路)。通过 SPC 标签对象来支持 SCP 业务。

SPC 标签对象为通用 UNI 对象的子对象，并和通用 UNI 子对象出口标签(EGRESS_LABEL)有相同的格式。SPC 标签信息如下：

SPC 标签(类型=4、子类型=2)

为支持 SPC 的情况，使用了通用 UNI 对象。该对象用来支持 SPC 标记信息，并同时支持与 SPC 连接请求相关的服务等级和指定的分集属性。对于 SPC 请求，源和目的 TNA 地址包含由源和目的网

络呼叫控制器各自控制的传送网元地址。因此，源 TNA 地址包含由源网络呼叫控制器控制的传送网元地址，目的 TNA 地址包含由目的网络呼叫控制器控制的传送网元地址。

12.3 信令流程举例

12.3.1 SPC 信令流程

图 52 和图 53 显式了 SPC 请求的信令流程。对于 SPC 连接，假定用户—网络之间的链路连接已经提供，相关的信息已经通过链路连接标识提供给控制平面。建立 SPC 连接的交换端口和建立交换连接的情况相同。SPC 连接请求信令流程如图 52 所示。

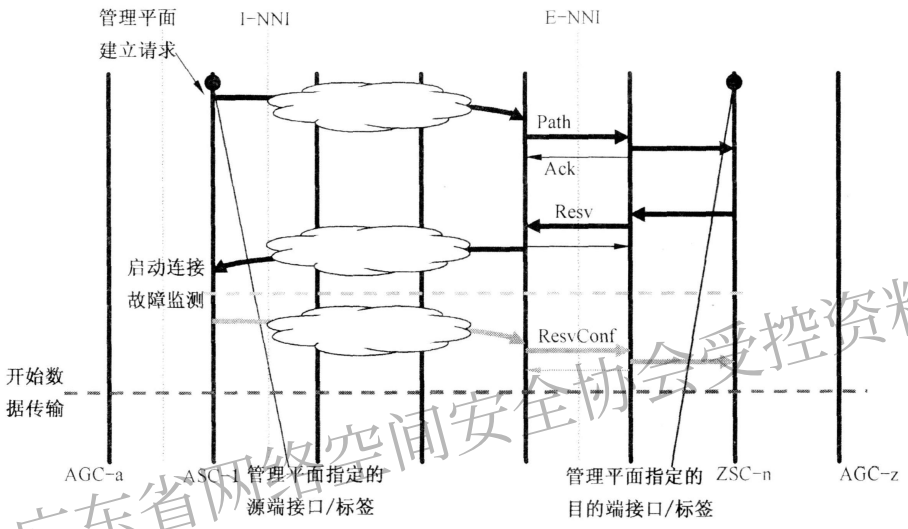


图 52 SPC 连接建立信令流

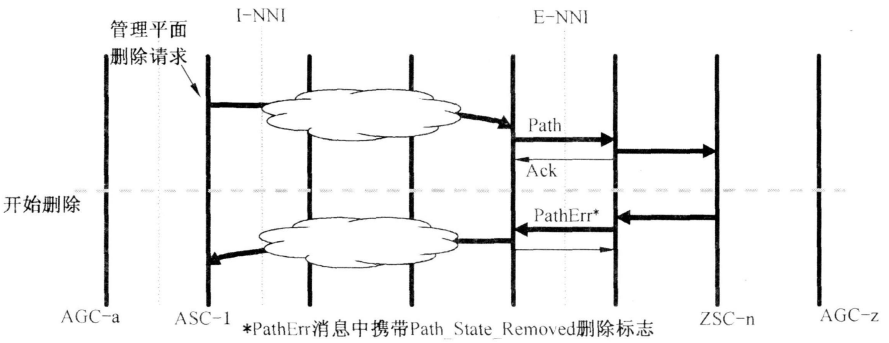


图 53 SPC 连接释放信令流

12.3.2 SC 信令流程

图 54 描述了 SC 连接建立的流程。为建立 SC 呼叫，用户通过 UNI 接口发起请求。呼叫请求通过网络传递到目的用户。当验证通过后接受请求，并向源用户发送肯定的指示信息。作为可选项，源用户也可以发送一个第三个阶段最终的连接确认消息。第三个阶段消息的引入用来支持向目的节点通知连接建立已经完成。

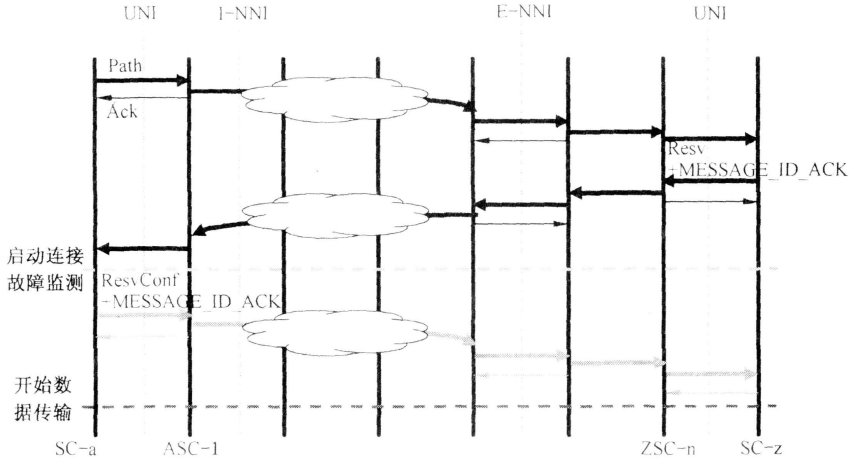


图 54　SC 连接建立信令流

SC 呼叫或连接释放时，释放请求可以由不同的控制器来发起。主叫或被叫呼叫控制器以及任何网络呼叫控制器等都可以发起呼叫释放动作。图 55 描述了主叫方发起释放请求的情况。图 56 描述了被叫方发起释放请求的情况。图 57～图 60 描述了网络呼叫控制器发起释放请求的情况。

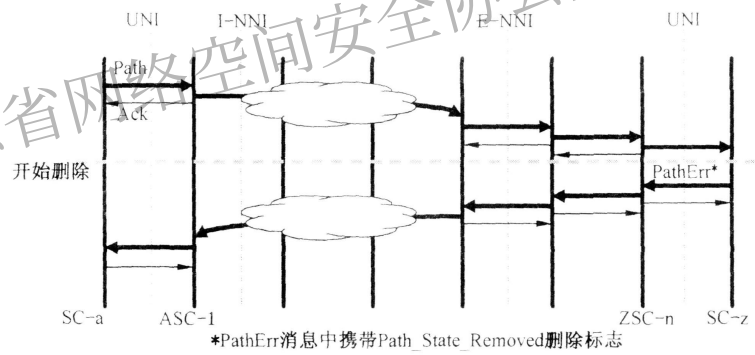


图 55　SC 连接释放信令流(SC-a 发起)

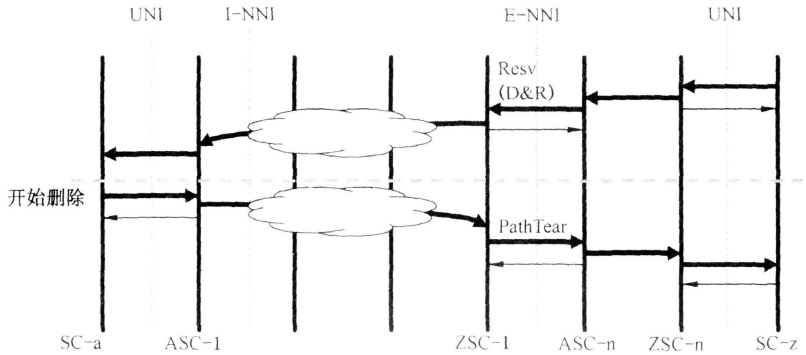


图 56　SC 连接释放信令流(SC-z 发起)

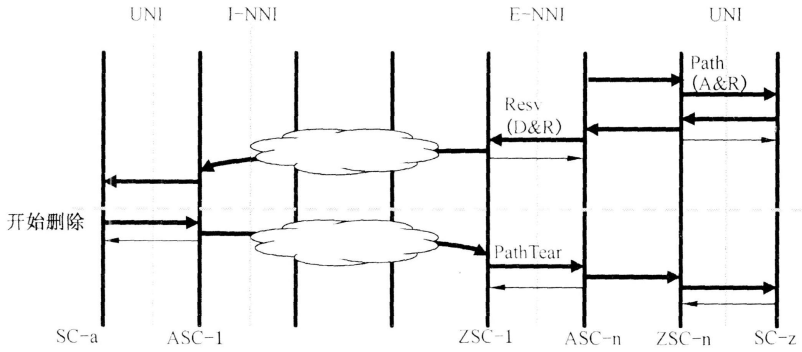


图 57　SC 连接释放信令流(中间控制器 ASC-n 发起,向下游 UNI 方向)

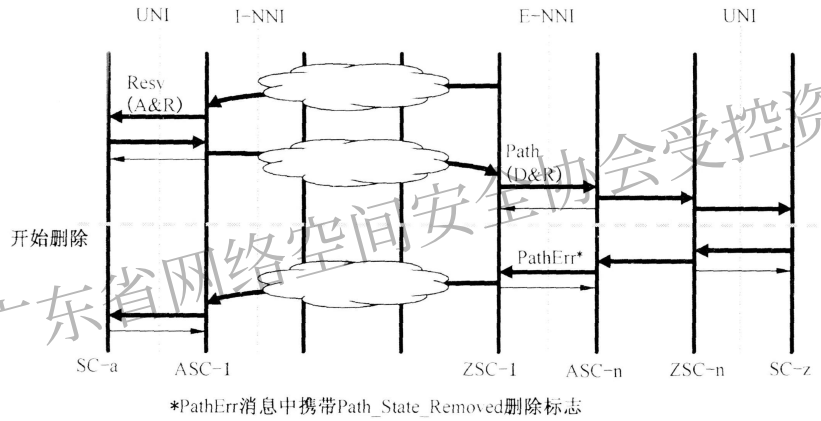


图 58　SC 连接释放信令流(中间控制器 ZSC-1 发起,向上游 UNI 方向)

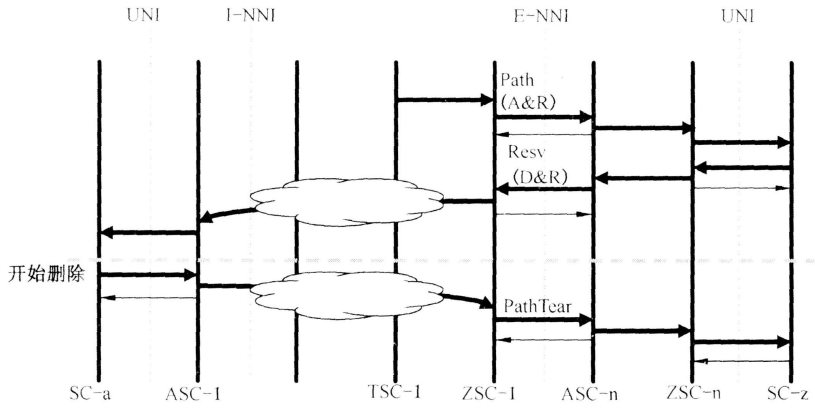


图 59　SC 连接释放信令流(中间控制器 TSC-1 发起,向下游 E-NNI 方向)

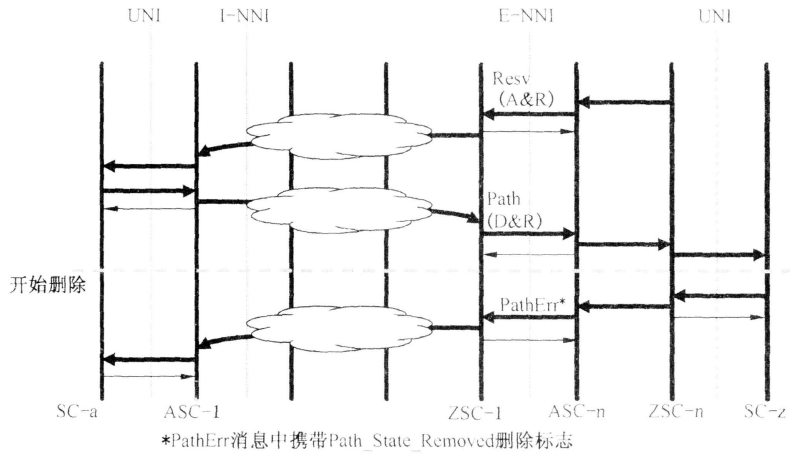


图 60　SC 连接释放信令流(中间控制器 ZSC-n 发起,向上游 E-NNI 方向)

12.3.3　建立拒绝信令流程

图 61 描述了连接建立请求被中间节点立即拒绝的情况。具体的拒绝原因可能有多种,如在请求时没有可用资源。

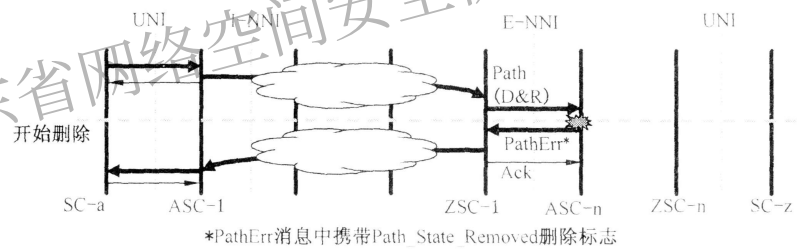


图 61　建立请求被中间节点 ASC-n 拒绝信令流

图 62 描述了连接建立请求被中间节点立即拒绝的情况。拒绝后由源节点发起删除动作(可选)。具体拒绝原因可能有多种,如在请求时没有可用资源。

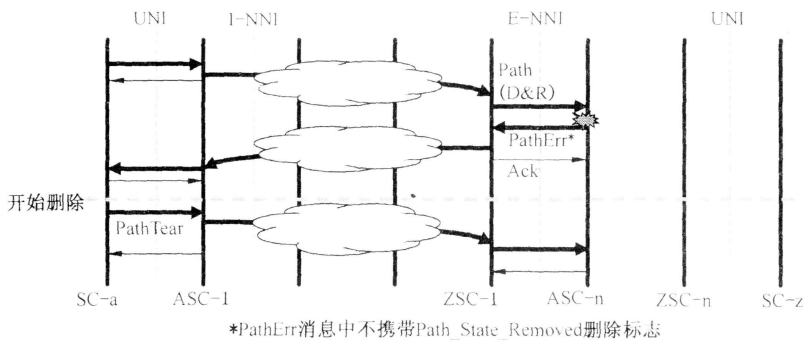


图 62　建立请求被中间节点 ASC-n 拒绝信令流(可选)

图 63 描述了请求建立过程中,当中间节点收到目的节点的指示信息后再拒绝连接的建立。例如,当中间节点传送平面出错而无法完成资源分配时对连接请求进行拒绝。

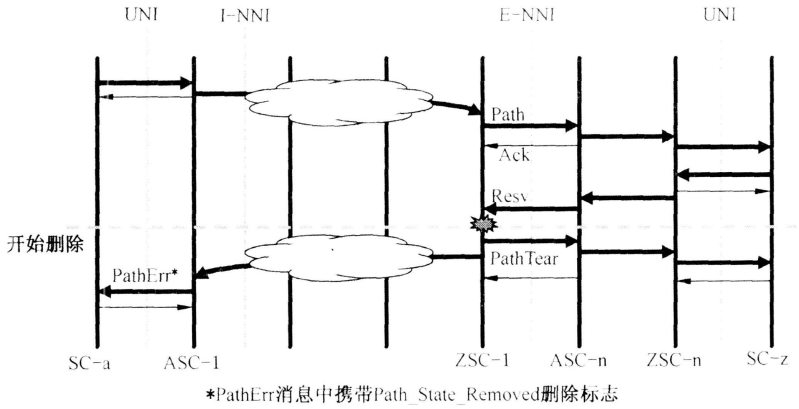


图 63　中间节点接收到指示消息后拒绝建立请求信令流

图 64 描述了在连接建立过程中，中间节点收到源节点的确认信息后对连接请求进行拒绝。如消息丢失（丢失 ResvConf 或与 ResvConf 对应的 Ack 消息）。这种情况下，连接已经建立，对连接的监控已经启动。因而，该故障构成了控制平面故障，不应该影响已有的业务。应该向管理系统报告控制平面故障的信息。

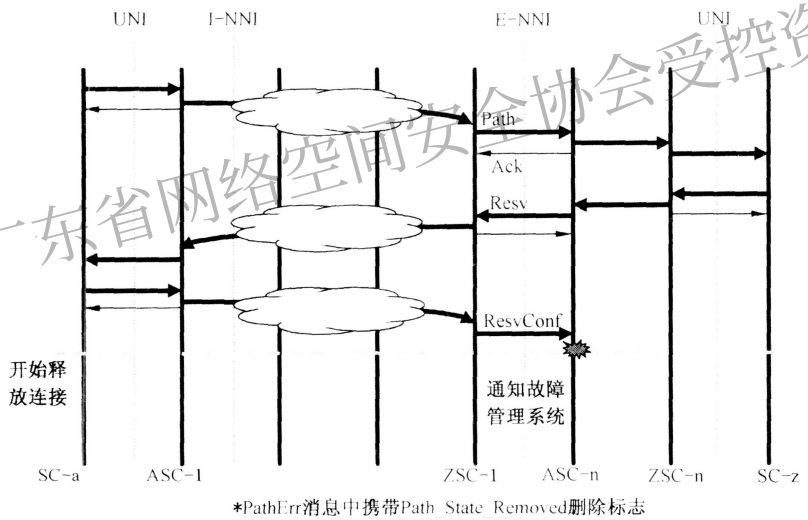


图 64　中间节点接收到确认消息后拒绝建立请求信令流

12.4　GMPLS RSVP-TE 的故障处理

12.4.1　GMPLS RSVP-TE 故障处理机制

不同类型的故障都有可能影响控制平面。这些故障包括从某一条信令通道失效到多个控制平面节点失效。控制平面需要提供适当的手段从这些故障中恢复，首先是基于本地控制平面机制进行恢复，然后借助控制平面与外部元件之间的交互进行恢复。故障处理的基本原则如下：

——控制平面失效将通知管理平面。管理平面将指导控制平面执行特定动作，包括清除部分连接和释放特定连接。

——控制平面节点提供相关信息的永久存储，例如对呼叫和连接状态信息，以及控制平面邻居信息进行永久存储。

——控制平面节点与外部元件通信，对由于控制平面失效引起丢失的信息进行同步。外部元件包括邻居控制平面节点和集中的永久存储元件（例如管理平面）。

——控制平面节点在不能恢复相关信息时，通知管理平面。管理平面可能采取以下措施：

- 释放受影响的连接；

- 保留受影响的连接。这时，连接与控制平面处在不同步状态，但连接仍然有效。

12.4.2 信令通道故障

在控制平面节点 A 和 B 之间的信令通道失效时，连接 1、4、6 将受到影响。由于 RSVP-TE 状态更新是点到点的，在节点 A 和 B 之间有 3 条通道更新消息被中断。根据上节所述，A 和 B 节点都将通知管理平面连接失效。图 65 表示失效过程。这里要特别强调的是，信令通道或控制协议实体故障一定不能导致对已经建立的连接进行删除。

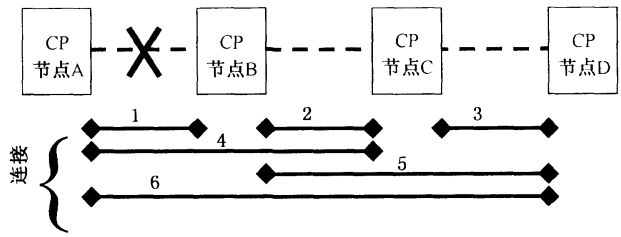


图 65 控制平面节点 A 和 B 间信令通道失效

信令通道修复后，节点 A 和 B 启动对受影响的连接和呼叫的状态同步(GMPLS RSVP-TE 重起机制)。如果同步过程发现连接状态是不同步的，将通知管理平面。

12.4.3 控制平面节点故障

传送网络提供了丰富的故障检测手段，特别是可以利用传送网络中的开销字节来检测传送层故障。检测到故障的节点可以尝试进行故障恢复。

在控制平面节点故障时，如图 66 所示节点 B 故障，邻节点 A 和 C 都会通知管理平面它们和节点 B 的通信发生故障。管理平面确定有哪些连接或呼叫受控制平面节点故障的影响。

在网络内部为允许不同的保护和恢复机制，当检测到故障后节点不应该删除连接，而是应该继续接收和发送 RSVP-TE 刷新消息直到接收到显式的拆除消息或连接状态超时。

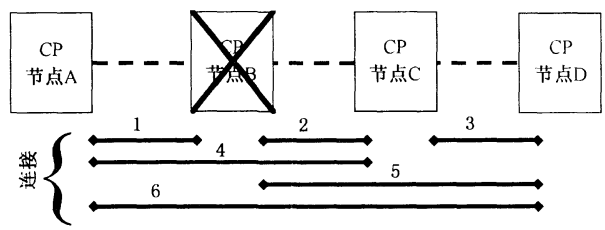


图 66 控制平面节点 B 故障

在自刷新情况下，当节点 B 的恢复时，节点 B 根据它本地永久保存的最近知道的连接状态来对连接进行恢复。节点 B 发起和节点 A 与 C 的恢复处理(注意，邻居节点同时会产生对应的处理动作)。因而，节点 B 的邻居可能会无法将信令通道故障和节点故障区别开。如果任何活动连接的状态已经不同步，管理平面需要根据上面定义的行为来提供信息对状态进行同步。

12.5 信令消息的可靠传递

为保证节点间信令消息的可靠传输，节点间消息的传送使用应答机制。具体的超时重传机制参考 RFC2961 中的定义。应答机制可以通过两种方式来实现：

——直接使用 ACK 消息，即每次发送信令消息后使用 ACK 消息来作为该消息的应答消息；

——间接使用响应消息中携带 MESSAGE_ID_ACK 对象来完成应答。即当所发送的消息有立即的响应消息时，将对应的 MESSAGE_ID_ACK 对象插入响应消息中完成应答功能。

RSVP-TE 还定义了 HELLO 消息和刷新(Srefresh)消息。这些消息只在相邻节点之间使用。

HELLO 消息机制提供相邻节点之间是否可达的监测机制。在节点发生故障时(包括节点故障和链路故障)通过 Hello 恢复重启机制来维持节点间的状态同步。

刷新消息用来定期刷新节点间的 RSVP 连接状态数据以保持节点间连接状态数据的同步。用来代替使用 Path 或 Resv 刷新消息以减少信令所需要的带宽。刷新消息携带一组消息 ID(MESSAGE_ID)对象,这些消息 ID 对象和所建立的连接状态相对应的触发消息 Path 或 Resv 一一对应。具体刷新机制见 IETF RFC2961。信令流程如图 67 所示。

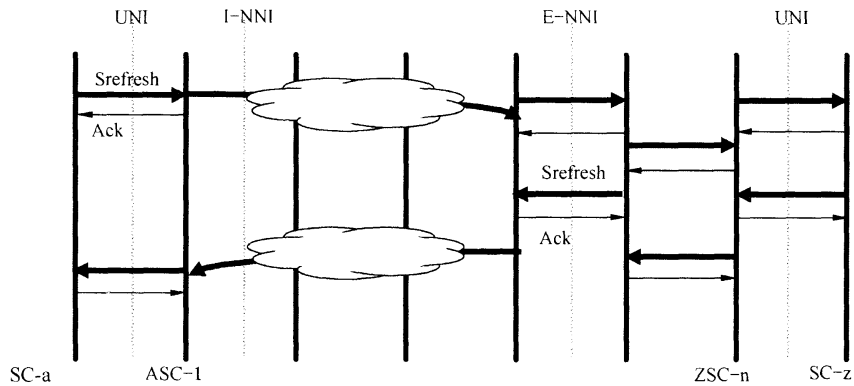


图 67　刷新机制信令流程

12.6　RSVP-TE 对象

GMPLS RSVP-TE 定义了如表 56 所述的对象来支持分布式的连接管理。

表 56　RSVP-TE 消息对象列表

编码	对象名称	对象格式(C-Type)
1	会话(SESSION)	7　LSP 管道 IPv4 8　LSP 管道 IPv6 11　UNI_IPv4 12　UNI_IPv6 15　ENNI_IPv4 16　ENNI_IPv6
3	RSVP 跳(RSVP_HOP)	1　IPv4 2　IPv6 3　IPv4 IF_ID 4　IPv6 IF_ID C-type 为 3、4,下面定义了子 TLV 类型值: 1　IPv4 2　IPv6 3　接口索引 4　COMPONENT_IF_DOWNSTREAM 5　COMPONENT_IF_UPSTREAM
4	完整型(INTEGRITY)	1　类型 1 完整性值
5	时间值(TIME_VALUES)	1　类型 1 时间值
6	错误说明(ERROR_SPEC)	1　IPv4 2　IPv6 3　IPv4 IF_ID 4　IPv6 IF_ID 相同子 TLV 如 RSVP_HOP

表 56（续）

编码	对象名称	对象格式(C-Type)
7	范围(SCOPE)	1 IPv4 2 IPv6
8	格式(STYLE)	1 类型 1 格式
9	流量描述符(FLOWSPEC)	2 Int-serv 流量描述符
10	过滤通配符(FILTER_SPEC)	7 LSP 管道 IPv4 8 LSP 管道 IPv6
11	发送方模版(SENDER_TEMPLATE)	7 LSP 管道 IPv4 8 LSP 管道 IPv6
12	发送方通配符(SENDER_TSPEC)	2 Int-serv
14	策略数据(POLICY_DATA)	1 1 策略数据
15	预留确认(RESV_CONFIRM)	1 IPv4 2 IPv6
16	预留标签(RSVP_LABEL)	1 1 号类型标签 2 通用标签 3 波段交换标签
19	标签请求(LABEL_REQUEST)	1 无标签范围 2 ATM 标签范围 3 帧中继标签范围 4 通用标签请求
20	显式路由(EXPLICIT_ROUTE)	1 类型 1 显式路由，子类型如下： 1 IPv4 前缀 2 IPv6 前缀 3 标签 4 无编号的接口 ID 32 自治系统
21	记录路由(RECORD_ROUTE)	1 类型 1 记录路由，子类型如下： 1 IPv4 地址 2 IPv6 地址 3 标签 4 无编号的接口 ID
22	HELLO	1 请求 2 应答
23	消息 ID(MESSAGE_ID)	1 类型 1 消息 ID
24	消息 ID 应答(MESSAGE_ID_ACK)	1 MESSAGE_ID_ACK 2 MESSAGE_ID_NACK
25	消息 ID 列表(MESSAGE_ID_LIST)	1 消息 ID 列表 2 IPv4 消息 ID 源列表 3 IPv6 消息 ID 源列表 4 IPv4 消息 ID 多播列表 5 IPv6 消息 ID 多播列表

表 56（续）

编码	对象名称	对象格式(C-Type)
34	恢复标签(RECOVERY_LABEL)	和 RSVP_LABEL 相同
35	上游标签(UPSTREAM_LABEL)	和 RSVP_LABEL 相同
36	标签系列(LABEL_SET)	1　类型 1
37	保护(PROTECTION)	1　类型 1
129	建议标签(SUGGESTED_LABEL)	和 RSVP_LABEL 相同
130	可接受标签系列 (ACCEPTABLE_LABEL_SET)	和 LABEL_SET 相同
131	重启动能力(RESTART_CAP)	1　类型 1
195	通知请求(NOTIFY_REQUEST)	1　IPv4 2　IPv6
196	管理状态(ADMIN_STATUS)	1　类型 1
207	会话属性(SESSION_ATTRIBUTE)	1　LSP_TUNNEL_RA 7　LSP 隧道
229	通用 UNI(GENERALIZED_UNI)	C-Type=1 时，类型、子类型定义如下： 1　源 TNA 地址 IPv4(子类型=1) IPv6(子类型=2) NSAP(子类型=3) 2　目的 TNA 地址 IPv4 子类型=1) IPv6(子类型=2) NSAP(子类型=3) 3　多样性 4　输出标签(子类型=1) SPC 标签(子类型=2) 5　服务等级
230	CALL_ID	1　操作者指定 2　全局唯一 Type 0x01　4 字节源传送网元地址 0x02　16 字节源传送网元地址 0x03　20 字节源传送网元地址 0x04　6 字节源传送网元地址 0x7F　设备供应商自定义的长度

对象的分类编号确定了当无法识别该对象时控制平面节点如何处理这些对象。

——Class-Num=0x0bbbbbbb

整个消息应该被拒绝并返回“不可知的对象类型”的错误消息。

——Class-Num=0x10bbbbbb

节点应该忽略该对象，不转发该消息也不发送相应的错误消息。

——Class-Num=0x11bbbbbb

节点应该忽略该对象，但不需要做任何检测和修改并在随后相关的消息中进行转发。

12.7 RSVP-TE 错误状态编码

GMPLS RSVP-TE 的错误/状态编码值如表 57 所示。

表 57 RSVP-TE 错误/状态编码

错误状态	编　码
连接建立—成功	Resv(或 ResvConf)消息
连接建立—失败：消息出错	错误编码(通用)
连接建立—失败：被叫方忙	错误编码 24/5
连接建立—失败：主叫方忙	错误编码 24/103
连接建立—失败：超时	错误编码 24/5 或 24/103[a]
连接建立—失败：标识错(A 端用户名无效)	错误编码 2/100
连接建立—失败：标识错(Z 端用户名无效)	错误编码 2/101
连接建立—失败：标识错(连接名无效)	错误编码 24/102
连接建立—失败：标识错(呼叫名无效)	错误编码 24/105
连接建立—失败：业务错(SNP ID 无效)	错误编码 24/6 或 24/11 或 24/12 或 24/14
连接建立—失败：业务错(SNP ID 不可用)	错误编码 24/6 或 24/11 或 24/12 或 24/14
连接建立—失败：业务错(无效的 SNPP ID)	错误编码 24/104
连接建立—失败：业务错(SNPP ID 不可用)	错误编码 24/104
连接建立—失败：标识错(SPC 标签无效)	错误编码 24/106
连接建立—失败：策略错(CoS 无效)	错误编码 24/101 也可以是其他值如 2/任意值
连接建立—失败：策略错(CoS 不可用)	错误编码 24/101 也可以是其他值如 2/任意值
连接建立—失败：策略错(GoS 无效)	错误编码 24/101 也可以是其他值如 2/任意值
连接建立—失败：策略错(GoS 不可用)	错误编码 24/101 也可以是其他值如 2/任意值
连接建立—失败：策略错(安全验证失败)	错误编码 2/100 或 2/101[b]
连接建立—失败：策略错(显式资源列表无效)	错误编码 24/1，24/2，24/3，或 24/7
连接建立—失败：策略错(恢复无效)	错误编码 24/15 也可以是 错误编码 24/100
连接建立—失败：连接错误(创建 SNC 连接失败)	错误编码 1/2
连接建立—失败：连接错误(创建 LC 连接失败)	错误编码 24/9
连接释放—成功	PathTear 或 PathErr (w/Path_State_Removed 标志置位)
连接释放—失败：消息出错	错误编码

表 57（续）

错误状态	编　码
连接释放—失败：超时	错误编码 24/5，24/103
连接释放—失败：标识出错（呼叫名无效）	错误编码 24/102
连接释放—失败：策略出错（安全验证失败）	（如果安全验证失败，GMPLS 拒绝请求）
连接释放—失败：连接错误（释放 SNC 连接失败）	错误编码 code＝21 时的错误值（通用）
连接释放—失败：连接错误（释放 LC 连接失败）	错误编码 code＝21 时的错误值（通用）
连接出错—不影响业务	错误编码（通用）
连接出错—影响业务	错误编码（通用）
连接出错—异常呼叫释放	错误编码 code＝21 时的错误值（通用）

[a] 超时是一个间隔事件。因而，报告的出错事件为下列之一：
——没有到达源节点的可用路由；
——没有到达目的节点的可用路由。

[b] 安全检验失败包括包括：
——源节点没有授权；
——目的节点没有授权。

除上面定义的错误编码和编码值之外，表 58 定义了协议指定的其他错误。

表 58　协议指定的其他错误编码值

错误编码	错误编码含义	错误编码值
0	证实	未定义
01	许可控制失败	比特位格式如：ssur cccc cccc cccc ss＝00：低 12 比特编码包含全局定义的子编码值（值见如下列表）。 ss＝10：低 12 比特编码包含组织指定的子编码。RSVP 不能将这些例外事件解释成某个具体的数值。 ss＝11：低 12 比特编码包含业务指定的子编码。RSVP 不能将这些例外事件解释成某个具体的数值。 由于流量控制机制可能替代了不同的业务，这些编码可能包含在其他正在使用的替代业务中。 u＝0：RSVP 拒绝消息后并不更新本地状态。 u＝1：RSVP 可以使用消息来更新本地状态并转发这些消息。这表示这些消息为带有通知性质的消息。 r：预留比特，应该置 0。 cccc cccc cccc：12 比特编码 当 ssur＝0000 时，低阶 12 比特全局定义的子编码定义如下： ——子编码＝1：边界延时要求不能满足。 ——子编码＝2：请求的带宽不可用。 ——子编码＝3：流规范中的 MTU 值必接口的 MTU 值大

表 58（续）

错误编码	错误编码含义	错误编码值
02	策略控制失败	（RFC 2750 中定义）： 0＝ERR_INFO：信息上报 1＝ERR_WARN：告警 2＝ERR_UNKNOWN：原因不明 3＝ERR_REJECT：通用策略拒绝 4＝ERR_EXCEED：配额或计费和合约不符 5＝ERR_PREEMPT：流被强占 6＝ERR_EXPIRED：以前定义的策略过期（未更新） 7＝ERR_REPLACED：以前定义的策略数据被替换并导致拒绝 8＝ERR_MERGE：策略不能合并（多播情况） 9＝ERR_PDP：PDP 状态失败或者功能不起左右 10＝ERR_SERVER：第三方服务不可用（如 Kerberos） 11＝ERR_PD_SYNTX：POLICY_DATA 对象语法错误 12＝ERR_PD_INTGR：POLICY_DATA 对象完整性校验失败 13＝ERR_PE_BAD：POLICY_ELEMENT 对象语法错误 14＝ERR_PD_MISS：缺少必需的 PE（PD 对象的 PE 为空） 15＝ERR_NO_RSC：PEP 没有资源来处理策略 16＝ERR_RSVP：PDP 遇到错误的 RSVP 对象或语法 17＝ERR_SERVICE：业务类型被拒绝 18＝ERR_STYLE：预约风格被拒绝 19＝ERR_FL_SPEC：流规范被拒绝（太大） 100＝发送者未授权 101＝接收者未授权 2^{15}～$2^{16}-1$ 之间的错误编码值留给站点或/和设备商使用
03	找不到和 Resv 消息对应的路径信息	未定义
04	找不到和 Resv 消息对应的路径信息发送者信息	未定义
05	预约风格冲突	错误值域包含低阶 16 比特的现有风格的可选向量，当发生冲突时使用该向量。此时不能转发该 Resv 消息
06	预约风格无法识别	未定义
07	目的端口冲突	未定义
08	发送端口冲突	未定义
09	（保留）	未定义
10	（保留）	未定义
11	（保留）	未定义
12	业务被抢占	比特格式如：ssur cccc cccc cccc 这里高阶比特 ssur 在错误编码＝01 时定义有效。全局定义的子编码由低阶 12 比特来定义，当 ssur＝0000 时有效，子编码具体值以后再定义
13	对象类型无法识别	错误值占 16 比特，由不明对象的类型值（Class-Num）和子类型值（C-Type）构成。该错误只有当 RSVP 准备拒绝消息时才发送，并且由类型编码（Class-Num）的高阶比特来决定

表 58（续）

错误编码	错误编码含义	错误编码值
14	对象子类型(C-Type)无法识别	错误值占 16 比特，由不明对象的类型值(Class-Num)和子类型值(C-Type)构成
15	(保留)	未定义
16	(保留)	未定义
17	(保留)	未定义
18	(保留)	未定义
19	(保留)	未定义
20	为 API 保留	错误值域包含一个 API 错误编码值，该 API 错误通过异步检测并必须上报
21	流量控制出错	比特格式如：ss00 cccc cccc cccc 高阶比特 ss 在错误编码值＝01 时有意义。当 ss＝0 时，下面全局定义的子错误编码值在低阶 12 比特中有效(cccc cccc cccc) ——子编码＝01：业务冲突 企图去合并两个不兼容的业务请求 ——子编码＝02：业务不支持 流量控制无法提供所请求的业务也没有相关的可接受的替换业务 ——子编码＝03：流规范值无效 残缺的或不合理的请求 ——子编码＝04：Tspec 值无效 残缺的或不合理的请求 ——子编码＝05：Adspec 值无效 残缺的或不合理的请求
22	流量控制系统出错	错误值包含系统指定的值，这些值提供了更详细的信息错误信息。RSVP 不要求能够解析这些编码值
23	RSVP 系统出错	错误值域将提供运行时的错误信息。RSVP 不要求能够解析这些编码值
24	路由问题	1 显式路由(EXPLICIT_ROUTE)对象错误 2 严格节点出错 3 松散节点出错 4 原始子对象出错 5 到目的节点无可用的路由 6 标签值不能接受 7 指示路由成环 8 MPLS 正在协商，但路径上有不支持 RSVP 能力的路由器 9 MPLS 标签分配失败 10 L3PID 不支持 11 标签集不支持 12 交换类型不支持 13 保留 14 编码值不支持 15 链路保护不支持 100 分集路由不可用 101 业务等级不可用 102 连接 ID 无效或不明 103 到源节点无可用的路由 104 接口 ID 不能接受 105 呼叫 ID 无效或不明 106 SPC 接口 ID 或标签无效

表 58（续）

错误编码	错误编码含义	错误编码值
25	通知出错	1 对 MTU 来说 RRO 太大 2 RRO 通知错误 3 隧道本地修复错误 4 控制通道激活状态错误 5 控制通道状态降级错误

附 录 A
（资料性附录）
PNNI 信令流程

A.1 概述

ATM 论坛的 PNNI 信令规范提供了可升级的网络—网络接口。PNNI 信令流程规范主要基于 ATM 论坛的 UNI 信令规范和 ITU-T 规范 Q.2931 所定义的信令协议规范和相关的扩展。本部分将信令协议进行了调整，以适应分布式的呼叫和连接管理以及自动交换光网络（ASON）。

本部分定义了新的连接标识符信息单元格式，用于传送网指定新的连接类型，包括子网点（SNP）和子网点池（SNPP）。定义了新的流量描述符信息单元，用于指定适合传送网络的流量描述符属性。

A.2 故障处理

多种类型的故障会影响到控制平面。这些故障可能是单个信令通道的故障，也可能是多个控制平面节点故障。控制平面需要能够从这些故障中恢复，本地的控制平面或者传送平面或者其他的外部元件进行交换以后，尝试发起恢复。

——控制平面的故障需要通知管理平面。管理平面可以指导控制平面执行针对故障的操作。这些操作包括清除或者释放部分的连接，以及执行特定的协议操作，实现状态的维护和恢复。

——控制平面节点可以对相关信息提供永久的存储，如呼叫和连接状态信息、配置信息以及控制平面邻居信息等。

——当一对呼叫和连接状态不能恢复时，控制平面接口可以同一个外部的元件进行通信，尝试进行状态信息的恢复。外部元件可以包括邻居控制平面节点或者由一个集中的元件（如管理平面）提供的永久存储的信息。

——控制平面节点通知管理平面不能恢复相关的信息（如不能同步连接的状态）。管理平面会响应如下的操作（默认的控制平面操作应该保留这些连接）：

- 释放受影响的连接；
- 保留受影响的连接，在这种情况下，此连接与控制平面所保留的连接信息不同步。但是这些连接应该保持有效。

——控制平面节点（在恢复节点故障后）不能从它本地的永久存储信息中获取邻居的连接状态，因而丢失了连接信息。在这种情况下，控制平面节点应该请求外部控制器（如管理系统）来获取信息恢复连接。类似的，呼叫状态不能恢复时也可以请求管理系统来解决。

因此，总的原则就是：

——控制平面故障不能导致释放已建立的连接。建立请求如果在处理过程中应该被移除（无论是在故障期间还是故障恢复以后），建立的连接如果收到释放请求，应该在故障恢复后释放。

——其余的针对没有连接的操作，控制平面应该根据其默认的配置来执行。

传送平面节点故障会导致释放已经建立的连接。这与连接的类型和服务级别相关。例如，对尽力而为的无保护连接，在传送平面故障后应该被释放掉，但是对于保护的连接应该能够得到恢复。在连接保护的过程中，最初的连接在新的连接建立后会被释放掉，这同样依赖于连接的保护类型。

A.3 信令流程举例

下面的例子显示了建立和释放连接的级别信令流程。

A.3.1 无监测的呼叫建立过程

图 A.1 显示了呼叫建立过程。T303 和 T310 是在 Q.2931 中描述的计时器。

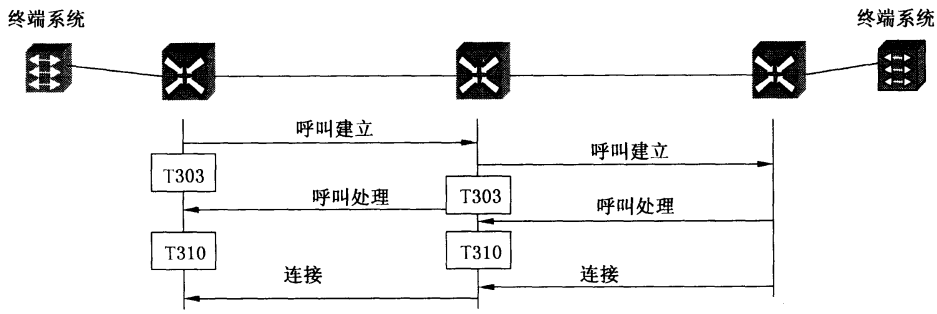


图 A.1 无监测的呼叫建立过程

A.3.2 无检测的呼叫释放过程

图 A.2 显示了呼叫建立过程。T308 是在 Q.2931 中描述的计时器。

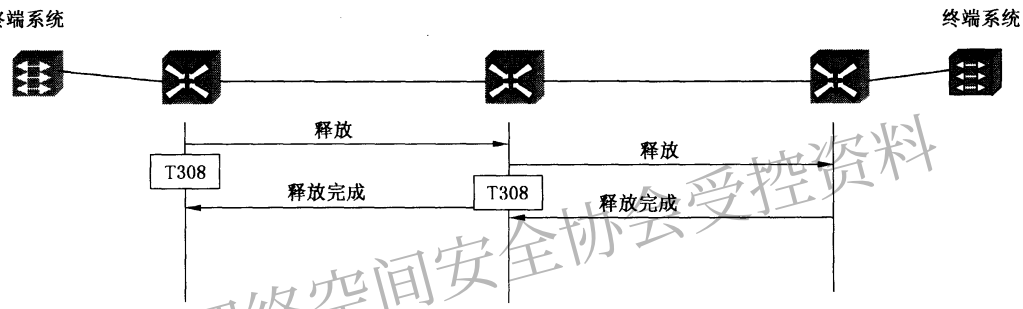


图 A.2 无监测的呼叫释放过程

A.4 信息功能定义

A.4.1 为分布式连接管理定义的信息

表 A.1 描述了用于点到点的呼叫和连接控制消息。

表 A.1 用于分布式连接管理的信息

消息类型	消息名称	消息名称(英文)
呼叫建立消息	呼叫处理	CALL PROCEEDING
	连接	CONNECT
	建立	SETUP
呼叫清除消息	释放	RELEASE
	释放完成	RELEASE COMPLETE
其他消息	连接可用	CONNECTION AVAILABLE
	调整确认	MODIFY ACKNOWLEDGE
	调整拒绝	MODIFY REJECT
	调整请求	MODIFY REQUEST
	通知	NOTIFY
	状态	STATUS
	状态查询	STATUS ENQUIRY
	踪迹连接	TRACE CONNECTION
	踪迹连接确认	TRACE CONNECTION ACKNOWLEDGE

A.4.2 呼叫处理消息

此消息用于指示呼叫建立已经发起，并且不再接受更多的呼叫建立。表 A.2 描述了呼叫处理消息的内容。

消息类型：呼叫处理

方向：接收方到发送方

范围：本地

表 A.2 呼叫处理消息

信息单元	类 型	长 度
协议辨别器	M	1
呼叫参考	M	4
消息类型	M	2
消息长度	M	2
连接标识符	M	5～*

A.4.3 连接消息

此消息由发送方向接收方传递，指示被叫用户接受了呼叫或者连接。表 A.3 描述了连接消息的内容。

消息类型：连接

方向：发送方到接收方

重要性：全局

表 A.3 连接消息

信息单元	类 型	长 度
协议辨别器	M	1
呼叫参考	M	4
消息类型	M	2
消息长度	M	2
带宽底层信息	O[a]	5～20
被叫方 SPC	O[b]	8～*
通知指示符	O[a c]	5～*
重路由	O[(1)]	11～80
重路由服务	O	8

[a] 如果连接指示包含此信息则包括在内。

[b] 在永久连接建立的情况下包括在内。

[c] 可以出现 3 次。

A.4.4 释放消息

此消息由网络节点发送到相邻网络节点，指示连接的清除，并且等待释放呼叫。表 A.4 描述了释放消息的内容。

消息类型：释放

方向：双向

重要性：全局

表 A.4 释放消息

信息单元	类 型	长 度
协议辨别器	M	1
呼叫参考	M	4
消息类型	M	2
消息长度	M	2
原因	M[a]	6～34
CrankBack	O[b]	7～72
通知指示器	O	5～*
重路由原因	O	5

[a] 此信息单元在消息中可以出现两次。

[b] 指示 CrankBack。

A.4.5 释放完成消息

此消息由网络节点向相邻的网络节点发送，指示清除了内部的连接，并且释放相应的呼叫。表 A.5 描述了释放完成消息的内容。

消息类型：释放完成

方向：双向

重要性：本地。此消息具有本地意义。但是，作为第一个呼叫清除消息时，它也可以具备全局意义。

表 A.5 释放完全消息

信息单元	类 型	长 度
协议鉴别器	M	1
相关呼叫	M	4
消息类型	M	2
消息长度	M	2
原因	O[a]	6～34
CrankBack	O[b]	7～72
重路由原因	O	5

[a] 在第一个呼叫清除消息中必须。包括当释放完全消息在一系列错误条件下发送，此信息单元可以在消息中出现两次。

[b] 用于指示 CrankBack。

A.4.6 建立消息

此消息由发送方向接收方发送，用于指示呼叫或者连接的建立。表 A.6 描述了建立消息。

消息类型：建立

方向：发送方到接收方

重要性：全局

表 A.6 建立消息

信息单元	类 型	长 度
协议鉴别器	M	1
相关呼叫	M	4
消息类型	M	2
消息长度	M	2
业务描述符	M	13～19
带宽传输能力	M	6～7
带宽高层信息	O[a]	5～13
带宽低层信息	O[a]	5～20
带宽重复指示器	O[c]	5
带宽报告类型	O[a f]	5
被叫方编号	M	(2)
被叫方软永久连接	O	5～30
呼叫方编号	O[a]	6～26
呼叫方软永久连接	O[b]	6～29
连接指示符	O	5～*
指定的传送列表	M[d]	33～546
网络呼叫相关指示符	O[e]	33～73
通知指示符	O[a]	5～*
扩展的 QoS 参数	O	6
重路由服务	O	8
重路由	O	11～80
路径传送列表	O[T]	38～1 466

[a] 如果接收的建立指示包括此信息。

[b] 可以包括在软永久连接建立的情况，当呼叫方终端点希望指示目的网络接口的值时，用于呼叫末端的软永久连接段。

[c] 当带宽重复指示符信息单元在指定传送列表(DTL)信息单元之前，它指示 DTL 信息单元在 DTL 中占的位置。当带宽重复指示符信息单元在其他信息单元之前，如果接收建立指示包括此信息它就存在。

[d] 由源节点指示用于呼叫的层次源路由。此信息单元可以重复 10 次(因此连接可以穿越 10 个逻辑组)。

[e] 由源节点包括，唯一的标识一个穿越网络的呼叫。它也用于关联相关的多个连接，对应于相同的呼叫。

[f] 可以出现 2 次。

A.4.7 状态消息

此消息由双方发送，响应状态查询消息，或者在任何情况下响应某个错误条件。表 A.7 描述了状态消息的内容。

消息类型：状态

方向：双向

重要性：本地

表 A.7 状态消息

信息单元	类 型	长 度
协议鉴别器	M	1
相关呼叫	M	4
消息类型	M	2
消息长度	M	2
呼叫状态	M	5
原因	M	6～34
连接标识符	O	5～*

A.4.8 状态查询消息

状态查询消息可以由双方发送，在任何时刻向对等实体请求状态消息。发送状态消息响应状态查询消息是强制的。表 A.8 描述了状态查询消息的内容。

消息类型：状态查询

方向：双向

重要性：本地

表 A.8 状态查询消息

信息单元	类 型	长 度
协议鉴别器	M	1
相关呼叫	M	4
消息类型	M	2
消息长度	M	2
连接标识符	O	5～*

A.4.9 通知消息

通知消息发送指示与呼叫或者连接相关的信息。表 A.9 描述了通知消息的内容。

消息类型：通知

方向：双向

重要性：Access

表 A.9 通知消息

信息单元	类 型	长 度
协议鉴别器	M	1
相关呼叫	M	4
消息类型	M	2
消息长度	M	2
通知指示符	M	5～*

A.4.10 连接可用消息

此消息不由 PNNI 改变，从呼叫方用户到被叫方用户确认连接的可用性。表 A.10 描述了连接可用性消息的内容。

消息类型：连接可用

方向：发送方到接收方

重要性：全局

表 A.10 连接可用性消息

信息单元	类 型	长 度
协议描述符	M	1
相关呼叫	M	4
消息类型	M	2
消息长度	M	2
通知指示符	M	5～*
带宽报告类型	O[a]	5

[a] 可以出现两次。

A.4.11 全局呼叫使用的消息

表 A.11 描述了点到点呼叫和连接控制使用的消息。

表 A.11 全局呼叫使用的消息

消息名称	消息(英文)	Reference af-pnni-0055
重启动	RESTART	6.3.3.1
重启动确认	RESTART ACKNOWLEDGE	6.3.3.2
状态	STATUS	6.3.1.7
状态查询	STATUS ENQUIRY	18.3.1.2

A.4.12 重启动消息

此消息由双方发送,请求接受重启动(如,释放所有相关的资源),指示删除由信令通道控制的连接。表 A.12 描述了重启动消息的内容。

表 A.12 重启动消息内容

信息单元	类 型	长 度
协议标识符	M	1
相关呼叫	M	4
消息类型	M	2
消息长度	M	2
连接标识符	M	5～*
重启动指示符	M	5

A.4.13 重启动确认消息

此信息发送确认接收到重启动消息,指示请求的重启动完成。表 A.13 描述了重启动确认消息的内容。

消息类型:重启动确认

方向:双向

重要性:本地

表 A.13 重启动确认消息

信息单元	类 型	长 度
协议描述符	M	1
相关呼叫	M	4
消息类型	M	2
消息长度	M	2
连接标识符	M	5～*
重启动指示符	M	5

A.5 信令处理流程

A.5.1 链路连接标识符的分配和选择

A.5.1.1 链路连接标识符的分配和选择机制

存在两种情况:

1) 链路相关的信令:3 层的信令实体包括控制链路中的通道。

2) 链路不相关的信令:3 层的信令实体控制通道不在链路中。

当网络节点接收到连接标识符信息单元，其链路相关信令区域编号的值不由网络中的节点支持，呼叫应该被拒绝，产生原因“链路拒绝”。

下面的标识符必须以特定的顺序出现：

双向，对称的　　——链路 ID，LCI

双向，不对称的　　——链路 ID，LCI(前向)，LCI(后向)

单向，下游　　——链路 ID，LCI(前向)

单向，上游　　——链路 ID，LCI(后向)

A.5.1.2　链路相关的信令

对链路相关的信令，在链路中携带相关的信令请求。

在连接指示符信息单元中，链路相关的信令区域编码为“链路相关的信令”，选择下列值之一在首选或唯一区域内基于对称。

a)　排除链路 ID；任何 LCI；或

b)　排除(Exclusive)链路 ID；排除 LCI。

在情况 a)中，接收方选择任何可用的 LCI，用于链路携带信令通道。

在情况 b)中，如果指定的 LCI 在链路中携带的信令通道是可用的，接收方选择它用于呼叫。

如果选择的 LCI 值在第一个相应建立信息(如呼叫处理消息)而返回的第一个信息的连接标识符信息单元中给出。链路相关的信令区域编码为“链路相关的信令”。首选或排除区域编码为“排除链路 ID，排除 LCI”。

在情况 a)中，如果没有 LCI 是可用的，释放完全消息携带原因编号为 34，“没有电路或通道可用”，将向网络中发送。如果信令功能支持回溯(CrankBack)功能，应携带回溯信息单元，包括回溯的原因编号为 34。

在情况 b)中，如果指定的 LCI 不可用，释放完全消息携带编号为 44，“请求的电路或通道不可用”，向网络中发送。如果信令功能支持回溯功能，应携带回溯信息单元，包括回溯的原因编号为 44。

接口的双方同时发送建立消息，指示使用相同的链路 ID 和 LCI 时会产生呼叫冲突。对 PNNI 接口，为了避免呼叫冲突，具备较高节点标识符的一方应该分配链路连接标识符的值。用于此的节点标识符应该是最低层的节点标识符，用于接口的每一方。节点标识符是在节点 ID 区域中指示的区域(字节 11～32)，在 PNNI hello 消息中发送，并通过相同的接口接收。具备较高节点标识符的发送方应该在建立消息中包括连接标识符信息单元，带有选项(b)(排除链路 ID 和排除 LCI)。来自具备较低节点标识符的发送方的建立消息应该选择选项(a)。

A.5.1.3　非链路相关的信令

在建立消息中请求一个连接，发送方应该指示下列内容之一：

a)　排除链路 ID；任何 LCI；

b)　排除链路 ID；排除 LCI；或

c)　任何链路 ID；任何 LCI。

在情况 a)和 b)中，链路相关信令区域编码为“显式链路 ID”。

在情况 a)和 b)中，如果指定的链路 ID 是可用的，接收方选择它用于呼叫。在情况 b)中，如果指定的 LCI 在链路内是可用的，接收方选择它用于呼叫。在情况 c)中，接收方可以选择任何可用的链路 ID 和 LCI。

选择好的链路 ID 和 LCI 的值在第一个信息中的连接标识符信息单元中指定，此消息由接收方用于响应建立消息(如呼叫处理信息)。链路相关的信令区域编码为“设备的显式指示”。首选或排除的区域编码为“排除链路 ID；排除 LCI”。

在情况 a)和 b)中，如果指定的链路 ID 不可用，接收方发送释放完成消息，原因编号为 29，指示“链路 ID 拒绝”。如果信令通道支持回溯功能，在回溯原因编号 29 中会包括回溯信息单元。

在情况 a)中，如果没有 LCI 可用，接收方发送释放完成消息，原因编号为 34，指示“没有电路或通道可用”。如果信令通道支持回溯功能，在原因编号 34 中携带回溯信息单元。

在情况 b)中，如果指定链路中的 LCI 是不可用的，接收方发送“释放完成”消息，原因编号 34，指示“没有电路或通道可用”。如果信令通道支持回溯功能，在原因编号 34 中携带回溯信息单元。

在情况 c)中，如果没有能力分配链路中的 LCI，接收方发送“释放完成”消息，原因编号 44，指示“请求的电路或通道不可用”。如果信令通道支持回溯功能，在原因编号 44 中携带回溯信息单元。

当双方的接口同时发送建立消息，请求相同的链路 ID 和 LCI 时，就可能产生呼叫冲突。对 PNNI 接口，为了避免呼叫冲突，节点标识符大的一方可以分配链路连接标识符(链路 ID，LCI)的值。具有较高节点标识符的一方应该在建立消息中包括连接标识符信息单元，使用选项 b)(排除链路 ID 和排除 LCI)。来自较低节点标识符一端的建立消息应该使用选项 a)或者选项 c)。

A.5.1.4 使用链路 ID

链路 ID 为信令通道提供了控制连接或者多个链路的能力，存在两种可能的配置：

1) 链路相关的信令：3 层的信令实体控制链路中的通道，此链路中携带信令通道。

2) 非链路相关的信令：3 层的信令实体控制的通道可能不在携带信令通道的链路中。

链路 ID 是一个不透明的值，没有预先定义的格式。它是链路的逻辑指示器，指示用于建立连接的链路。

在上面的情况 1)中，链路 ID 是不重要的，编码为 0。

在上面的情况 2)中，链路 ID 包括指示用于建立连接的链路。

对于控制平面节点通传送节点物理上分离的情况与 2)相同。在这种情况下，链路 ID 包含用于建立连接的节点或链路的参考信息。

情况 2)中链路 ID 的默认值是两个节点 ID 的级联值，较低的值在签名，同时包括 SNP 标识符。

A.5.1.5 链路 ID 长度

链路 ID 默认的长度是 4 个 8 位比特，这是首选的长度，也可用支持其他可选的长度。如果接收方接收到的连接标识符信息单元中携带不支持的链路 ID 长度，那么它会认为指定的链路 ID 是不可用的。

对链路 ID 没有预留的值。

A.5.1.6 LCI 长度和值

此标识符使用的编码方案在 G.707 的第 7 部分中规定。较高顺位的有效载荷列使用 S。指定此列的位置的数值从 1 开始。如果值为 0 说明此编码没有使用。

A.5.1.7 服务分离、业务参数和 QoS 选择程序

QoS 应该在扩展的 QoS 的信息单元中指出。

如果网络能够提供所请求的 QoS 种类，网络应该为用户处理此呼叫。如果网络不能提供请求的 QoS 种类，网络应该拒绝此呼叫，返回一个“释放完成”消息，编号为 49，指示“服务质量不可用”。

业务描述符信息单元应该指示连接类型。

如果网络有能力提供请求的连接，网络应该为用户处理此呼叫。如果网络不能提供请求的连接，网络应该拒绝此呼叫，返回一个“释放完成”消息，指示下面的原因编码。

——编号 57：“请求的能力没有授权”；

——编号 58：“请求的能力目前不可用”；

——编号 65：“请求的服务没有应用”。

如果包括恢复或保护信息单元，且网络能够提供请求的服务，网络应该为用户处理呼叫。如果网络不能提供请求的服务，网络应该拒绝此呼叫，返回一个“释放完成”消息，编号为 63，指示“服务或操作不可用，未指定”。

A.5.2 非永久连接的清除

此程序只有在连接没有指定为永久连接的情况下才能应用。

此程序在 PNNI 1.1 的 6.5.3 中指定。

A.5.3 永久连接在故障情况下的清除

A.5.3.1 永久连接的清除

永久连接的清除程序只有在永久连接在控制平面故障的情况下使用，可以间接的通过服务分类或者通过配置实现。

如果接收方在 T308 计时器过期之前没有接收到“释放完成”消息，发送应该保持释放请求（N11）状态。发送方应该周期性的重试连接清楚程序。重试的频度与应用是独立的。

A.5.3.2 永久连接的信令 AAL 连接重置

无论何时，Q.2931 实体形成了自发的信令 AAL 重置，通过 AAL 建立指示消息，应用下面的程序：

a) 对于呼叫在清除阶段（状态 N11、N12），不做操作；

b) 对于呼叫在建立阶段（状态 N1、N3、N4、N6、N7、N8、N9）应该进行维护。应该调用状态查询程序；

c) 呼叫处于激活状态时应该维护，应该调用状态查询程序。

A.5.3.3 永久连接的信令 AAL 连接释放

当 Q.2931 实体通知进行信令 AAL 连接释放时，通过发送 AAL 释放指示，应该采用下面的程序：

a) 任何不再激活状态的呼叫应该进行本地清除；

b) 对于处于激活状态的呼叫，不做任何操作。

Q.2931 实体应该请求信令 ALL 重新建立，通过发送 ALL 建立请求信息。

当信令 AAL 重新建立通过使用 AAL 建立确认信息时，应该采用下面的程序：

执行状态查询操作。

A.5.3.4 状态查询程序

状态查询程序在 PNNI 1.1 的 6.5.6.11 中定义，状态查询程序应该同附加的信令状态恢复程序一起实现。

请求进行连接状态恢复时，应该在状态查询信息中包括连接标识符信息单元。链路 ID 和 LCI 应该设置成于指定的呼叫相关的值。链路相关的信令区域应该指出“显式指定的链路 ID”。首选或排除的区域应该指出“排除链路 ID，排除 LCI”。丢失状态信息的节点，但是检查在连接标识符中指定的内容存在，应该响应一个“状态”信息，状态应该指定为“空”，原因编号为 30，指示“响应状态查询”。

丢失状态信息的节点，不能认证指定连接标识符所指定的连接释放存在，应该返回状态信息，状态指示为“空”，原因编号 101，指示原因为“信息同呼叫状态不一致”。

丢失状态信息的节点，但是有能力执行本地状态查询，应该响应一个“状态信息”，状态指示为“激活”，原因编号为 30，指示“状态查询响应”。

A.5.3.5 接收一个状态信息

PNNI 1.1 的 6.5.6.12 定义了此程序。

对于处于激活状态的连接遇到控制通道故障的情况：

如果接收到的状态时释放请求，不做任何操作。

对于请求状态信息恢复的连接，接收到“状态”信息应该包括连接标识符信息单元，采用下面的程序：

1) 如果接收的状态为空，原因编号 30，指示“状态查询的响应”，应该调用状态恢复程序。

2) 如果接收的状态为空，原因编号 101，指示“信息同呼叫状态不一致”，应该调用连接清除程序。

A.5.3.6 增强的状态查询

在 PNNI 1.1 的附录 R 中定义的程序应该在下面的异常中使用：

此程序对点到多点的应用不支持。

A.5.3.7 永久连接信令状态的恢复

此程序用于恢复连接的状态，包括在“状态”和“状态查询”消息中使用连接标识符，需要进一步的研究。

A.5.3.8 控制通道故障情况下的信令处理

此程序用于连接在信令通道故障的情况下保持存在，此时控制平面通过 AAL 释放指示信息通知 Q.2931 实体故障发生。

此程序用于处理原本穿越故障控制通道的消息。除了“释放”和“释放完成”消息，其他的消息都应拒绝，并向消息源发送，原因编号 101，指示“消息同呼状态不一致”。

“释放”和“释放完成”消息应该被保留，直到控制通道恢复。在控制通道恢复以前，不能对这些信息做普通的处理。

A.5.4 传送平面故障

当接收到一个通知指示传送链路 ID 指示的传送通道出现故障时，执行下面的操作：

1) 如果连接在传送平面故障的情况支持持久存在，不需要做任何操作；

2) 否则，需要执行普通的连接清除程序。

A.5.5 接收到通知消息

通知消息提供连接的信息，同时并不影响连接的状态。接收到通知消息的节点应该执行下面的操作：

——如果节点是连接的终端，它应该发出一个通知指示，同时不做进一步的操作；

——如果节点是连接的中间节点，它应该发出一个通知指示，同时继续执行将通知消息向传送的方向发送。不做其他的操作。

附 录 B
（资料性附录）
基于 GMPLS CR-LDP 的信令流程

B.1 CR-LDP 概述

RFC 3036(LDP 规范)和 RFC 3212(采用 LDP 基于约束的 LSP 建立)定义了 GMPLS CR-LDP 所使用的属性。CR-LDP 信令的扩展应支持 GMPLS。

在 RFC 3036 中定义了四种类型的 LDP 消息：

- 发现消息：用于公布和保持存在的网元。
- 会话消息：用于在 CR-LDP 对等实体之间建立、保持和终止会话。
- 公告消息：用于建立、改变和删除标签映射(或连接)。
- 通知消息：用于提供咨询信息和指示出错信息。

LDP 和 CR-LDP 用于数据网络，因此不存在连接和呼叫的概念。采用 LDP 消息可支持连接控制。而呼叫控制还需要引入新的消息类型：

呼叫控制消息：用于呼叫控制进程。

呼叫控制消息是本部分引用的一个新的 LDP 消息类型。

发现消息描述了一种机制，就是通过周期性的发送 Hello 消息来让网元指示它在网络中的存在。该消息通过 UDP 传送给 CR-LDP 端口。子网中所有路由器的 IP 组播地址作为目的 IP 地址。当一个网元选择与另一个网元建立会话时(它的地址通过 Hello 消息被学习)，网元通过 TCP 进行 LDP 初始化。在成功完成初始化进程之后，两个网元成为 LDP 对等实体，并且开始交换公告消息。

LDP 采用 TCP 传送会话、公告和通知消息，也就是采用 TCP 传送除了基于 UDP 的发现机制之外的所有消息。采用 TCP 传送消息使 CR-LDP 保持一种硬状态特性，“硬状态”指的是实体的状态表现会一直保持不变，除非发生动作才会发生改变。CR-LDP 可以利用 TCP 提供的可靠传输和流控机制，因此无需在 LDP 层建立这些特性。

根据标签分布模式(独立的或顺序的)、标签保持方式(保守或自主)以及标签公告方式(按需发向下游或主动发向下游)使 LDP 具有多种方式。这些方式在 RFC 3036 中进行了定义。CR-LDP 的通用运行方式应是按需发向下游的顺序控制方式。

CR-LDP 支持确定路由和不确定路由，并且支持回溯(CrankBack)机制。

在缺少运营商级别的设备时，也就是在故障期间状态被丢失时 CR-LDP 采取一种完善的重启机制，在重启期间，故障节点利用其他节点学习到自己的状态信息来重建状态。

B.2 基本呼叫标识符的支持

为了支持 G.7713 中描述的呼叫模型，除连接之外，CR-LDP 还将包括呼叫控制。这可以实现由单呼叫建立多连接，改变现有连接，以及和计费相关的呼叫功能。支持穿越 UNI 和 E-NNI 的呼叫控制的 CR-LDP 扩展主要引入了呼叫 IDTLV。

B.3 支持 SPC 的 CR-LDP

配置源端和宿端的用户—网络连接部分，而网络连接部分由控制平面建立，这称之为 SPC 业务。例如：接收到一个来自外部系统(如来自管理系统)的初始请求，这意味着控制平面会根据相关的信息决定将使用的宿端(即：网络到用户)的链路连接。CR-LDP 中对 SPC 的支持在 OIF UNI1.0 定义的出口标签的使用中进行了描述。

B.4 故障处理

影响控制平面的故障有不同类型。这些故障包含了从简单信令通道故障到多控制平面节点故障。控制平面需支持相应的操作来克服这些故障，最初应尽量能克服的故障包括：基于本地控制平面机制的故障，与传送平面的本地互操作故障。在这之后应尽量克服基于控制平面与外部元件之间的互操作故障。

故障处理的指导方针包括：

- 控制平面故障被通报给管理平面。管理平面会根据故障指导控制平面采取一定的措施。这些措施可能包括：删除部分连接，释放一定的连接，或采取状态保持和恢复的其他专用操作协议。
- 控制平面节点会提供相关信息的永久存储，例如呼叫和连接的状态信息，配置信息，以及控制平面相邻节点信息。
- 如果连接或呼叫状态不能被恢复，在故障修复之后，控制平面节点会与外部元件进行通信以恢复状态信息。外部元件包括相邻控制节点或中央元件（如管理平面）提供的永久存储。
- 控制平面节点通知管理平面无法恢复相关信息子集（例如：无法同步连接状态）。管理平面会作出下述响应操作（缺省控制平面操作应保持连接）：
  ——释放受影响的连接；
  ——保持受影响的连接。在这种情况下，从控制平面的角度连接会保持非同步；但连接会保持有效。
- 控制平面节点（节点故障恢复之后）可能不会从它的本地永久存储中恢复相邻节点的连接状态，因此会丢失连接信息。在这种情况下，控制节点应请求外部控制器（例如：管理系统）恢复连接信息。同样的，呼叫状态可能也不会被恢复，采用请求管理系统干涉解决的方式。控制平面和管理平面之间的互操作规范已经超出了本部分的范围。

因此，作为一般规则：

- 控制平面故障不应导致已建立连接的释放。在连接完成后的过程中的建立请求会被删除（无论是故障期间还是故障恢复之后）。对已建立的连接提起的释放请求必须被释放（无论是故障期间还是故障恢复之后）。
- 控制平面的附加操作会由针对特殊连接类型配置的缺省动作决定。

但是，传送平面节点故障会导致已建立连接的释放。这是由连接类型和每条连接的服务等级决定的。例如：尽力而为的无保护连接在传送平面节点故障期间会被释放；而有保护的连接根据连接的业务等级规定必须被保存（或保持）。要注意的是即使对于有保护的连接来说，原始的连接可能会被释放，同时建立一条新的连接（这也是根据特定连接的保护类型决定的）。

可能发生的故障类型有三种。它们是信令通道故障，传输链路故障和节点（交叉连接）故障。故障恢复通常包括状态恢复和与邻接 NE 的再同步。

信令通道故障中断了两个以上节点之间的控制消息流。该故障条件应是没有影响已建立的连接，这些连接应继续处于无中断的激活状态。对于连接状态的恢复必须与邻接 NE 进行再同步。那些局部建立的连接必须被终止。

传输链路故障中断了数据流。数据平面故障需与控制平面进行通信以采取适当的措施。可能的操作包括：终止受链路故障影响的连接，通过其他链路或节点的连接重路由。每条连接的特定操作是由连接的保护必要条件决定的。

节点故障与链路故障相似。通过与故障节点通信的丢失，节点故障被间接的通知给控制平面。在这种情况下，必须终止受影响的连接或通过其他节点重新选路。

B.5 GMPLS CR-LDP 消息

所有 CR-LDP 消息采用类型长度值（TLV）编码方案，这样它们就具有了通用的结构，图 B.1 说明

了这种编码方案，图中同时给出了为每个域分配的比特数。TLV 的值部分(或简略的 TLV)会包含多个 TLV。长度域以字节为单位定义了值域的长度。U 和 F 比特的含义参见 IETF RFC 3036。

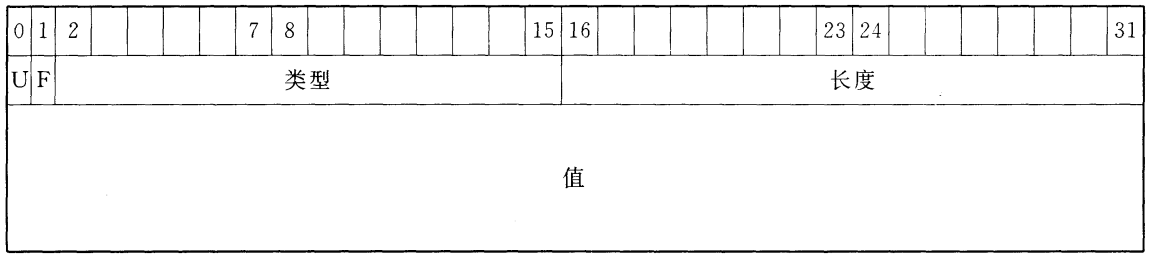


图 B.1 TLV 编码结构

表 B.1 给出了 CR-LDP 定义的消息集及其范围、功能和参考的源标准。

表 B.1 CR-LDP 消息

消息	消息(英文)	范围	参考点	功 能
标签请求	Label Request	端到端	全部	由呼叫方根据相关属性发出建立连接的请求
标签映射	Label Mapping	端到端	全部	由被叫方根据标签请求消息提供的属性发出建立连接的指示
初始化	Initialisation	本地	全部	用于在网元之间建立 LDP 对等实体
Hello	—	本地	全部	用于对等实体的发现
保持激活	KeepAlive	本地	全部	用于 LDP 会话的维护
标签释放	Label Release	端到端	全部	在下游方向发出连接拆除信号
标签撤消	Label Withdraw	端到端	全部	在上游方向发出连接拆除信号
标签中断	Label Abort	端到端	全部	中断一个特定的请求
通知	Notification	本地或端到端	全部	查询信息或错误信息的通知
查询	Query	端到端	I-NNI	收集连接信息
查询回复	Query Reply	端到端	I-NNI	将查询的信息编码生成查询回复消息
局部查询回复	Partial Query Reply	端到端	I-NNI	与查询回复相同。作为查询消息的相应，该查询消息不经过整个路由
状态查询	Status Enquiry	本地	UNI,E-NNI	请求相邻网元的状态
状态响应	Status Response	本地	UNI,E-NNI	包含相邻网元的状态
呼叫建立	Call Setup	本地	UNI,E-NNI	由呼叫方发出具有特定属性的呼叫建立请求
呼叫释放	Call Release	本地	UNI,E-NNI	由发起呼叫建立请求的呼叫方发出
注："全部"的含义包括 UNI,E-NNI 和 I-NNI。				

以下消息是 IANA 分配的 RFC 3036 LDP 名称空间编码：

0x0500＝呼叫建立

0x0501＝呼叫释放

表 B.2 总结了与呼叫和连接控制相关的不同 CR-LDP TLV。该表给出了不同 TLV 的用途和包含的消息。

表 B.2 呼叫和连接控制 TLV

TLV 名称	TLV 名称(英文)	功 能	消 息
通用的标签 TLV	Generalized Label TLV	标识分配给节点到特定连接的标签	标签请求、标签映射、查询、查询响应
建议的标签 TLV	Suggested Label TLV	上游节点建议下游节点使用的一组标签	标签请求
上游标签 TLV	Upstream Label TLV	对于双向连接,在上游方向上使用的标签	标签请求
可接受的标签集 TLV	Acceptable Label Set TLV	显示接受的标签值	通知
标签集 TLV	Label Set TLV	限制下游节点的标签选择	标签请求
通用标签请求 TL	Generalized Label Request TLV	传送被请求连接要求的特性	标签请求
波带交换 TLV	Waveband Switching TLV	波带交换时的标签值	标签映射
保护 TLV	Protection TLV	被请求连接的保护要求	标签请求
管理状态 TLV	Admin Status TLV	显示连接的管理状态	通知
ER TLV		说明明确的显式路由	
源 ID TLV	Source Id TLV	标识源端客户的 TNA 地址	标签请求、标签映射
宿 ID TLV	Dest Id TLV	标识宿端客户的 TNA 地址	标签请求、标签映射
本地连接 ID TLV	Local Connection Id TLV	标识通过 UNI 接口的本地连接	标签请求、标签映射、标签撤销、标签释放、通知
出口标签 TLV	Egress Label TLV	在通过 UNI 接口时使用,用以显示宿端客户使用的标签	标签请求、标签映射、标签释放、标签撤销、状态、状态请求、通知
分集 TLV	Diversity TLV	显示被请求连接的不同属性	标签请求、标签映射
合约 ID TLV	Contract Id TLV	由服务提供者确定的格式和含义	初始化
UNI 服务等级 TLV	UNI Service Level TLV	显示 UNI 的服务等级协议。该值由服务提供者分配	标签请求、标签映射
呼叫 ID TLV	Call Id TLV	标识通过单一运营商网络的呼叫	呼叫建立、标签请求、标签映射、标签释放、标签撤销
呼叫能力 TLV	Call Capability TLV	标识所请求呼叫的能力	呼叫建立
SDH 业务量参数 TLV	SDH Traffic Parameters TLV	所请求 SDH 连接的业务量参数	标签请求、标签映射
回溯 TLV	CrankBack TLV	携带有关连接建立失败的位置信息返回源节点	通知
反馈 TLV	Feedback TLVs	携带有关资源利用信息返回源节点	通知、标签映射、标签撤销

B.6 采用 CR-LDP 的呼叫和连接控制进程

B.6.1 CR-LDP 发现和对话的初始化

CR-LDP 发现是 CR-LDP 节点之间自动发现过程。自动发现不需要进行配置。CR-LDP 采用 Hello 机制以支持发现功能。使用 UDP 协议交换 Hello 消息(只有该 CR-LDP 消息不用 TCP 传送)。在 RFC3036 中定义的 Hello 进程(包括扩展的 Hello)可不用修改直接引用。

CR-LDP 会话的初始化通过交换初始化消息实现。初始化消息承载相关节点的特征信息,例如支持 FT 操作。RFC3036 中定义的初始化消息进程可不用修改直接引用。

B.6.2 采用 CR-LDP 建立呼叫

本部分对 CR-LDP 功能进行扩展(也就是新消息和新 TLV)以支持呼叫和连接分离功能。

可采用两种呼叫建立模式。第一种模式先进行呼叫建立请求,该请求不包括相应的连接。在这种情况下,信令消息只包括该呼叫的相关参数信息,例如源地址和目的地址。然后,该呼叫的对应连接通过 B.6.4 中描述的连接建立进程实现。

在第二种呼叫建立模式下,呼叫建立请求承载了连接建立的请求。这是加速连接建立过程的一个重要特性。随后的连接建立过程采用 B.6.4 中描述的连接建立进程实现。

CR-LDP 支持这两种呼叫建立模式。

在第一种呼叫建立模式下,CR-LDP 消息引入了一个新的消息——呼叫建立消息。呼叫建立消息由呼叫方发出,并且通过网络直到它到达目的地(被呼叫方)。然后,通知消息被发回呼叫方以指示呼叫建立是否成功。呼叫建立失败会有很多种原因,因此通知消息必须包含呼叫失败的原因。图 B.2 给出了采用第一种模式成功建立一个呼叫的过程。

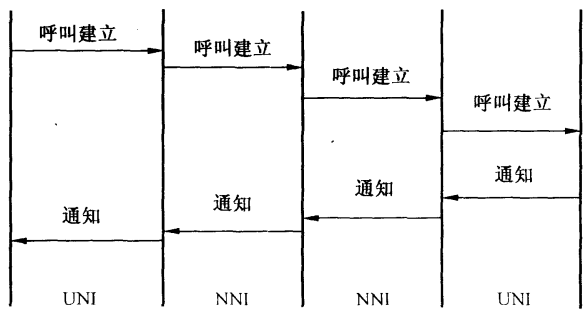


图 B.2 无连接的呼叫建立

呼叫建立的第二种模式允许使用同一呼叫请求来建立连接。CR-LDP 中连接的建立采用 LDP 标签请求和标签映射消息实现。由于需要进行标签分配,因此用相同的连接建立消息来建立呼叫。在这种情况下,标签请求消息将同时携带呼叫参数和连接参数,例如:连接流量参数。图 B.3 给出第二种模式下的呼叫和连接建立过程。通知消息发回被呼叫方用于确认连接是否建立和连接是否准备就绪。B.6.4 描述的通知消息的发送是可选的。

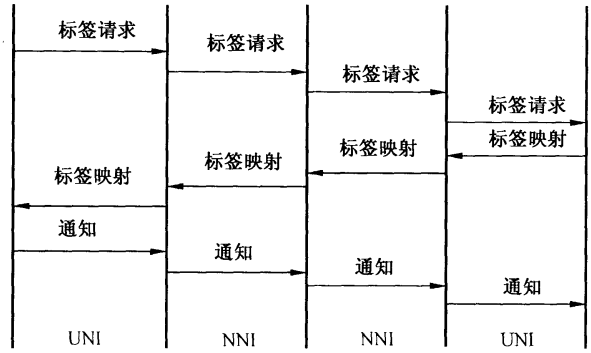


图 B.3　呼叫和相应连接的建立

呼叫建立完成之后，随时都可以进行连接的建立和拆除。需要采用相应的连接和呼叫机制，这可以通过采用呼叫和连接请求中携带的呼叫验证信息获得。

呼叫和连接分离的一个重要应用是建立带保护的多连接呼叫能力，例如，图 B.4 给出了建立两条连接的呼叫消息的传送过程。

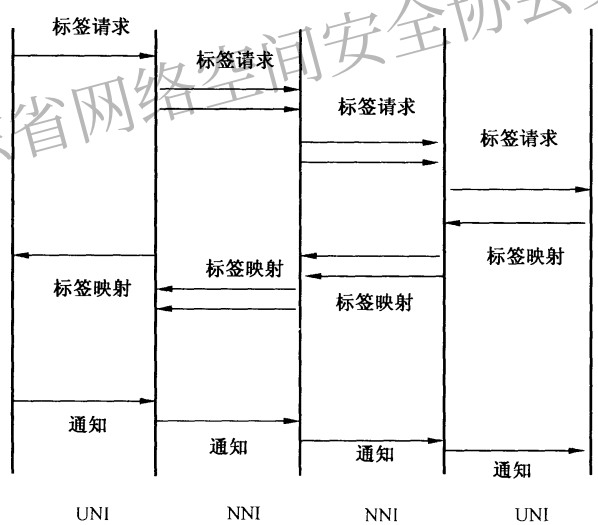


图 B.4　采用 CR-LDP 的 1＋1 应用

图 B.4 所示的过程需要同时建立两条连接。通过在 NNI 接口上生成两个标签请求消息来说明这两条连接。

B.6.3　采用 CR-LDP 释放呼叫

与所采用的建立模式无关，为了拆除呼叫引入了一个新的消息——CR-LDP 呼叫释放消息。呼叫拆除操作必然会导致与此呼叫相关的所有连接的拆除。CR-LDP 中连接的拆除是通过使用标签释放和标签撤消消息实现的。连接的删除也采用相同的过程。当接收到呼叫释放消息时，通过发送标签释放或标签撤消消息触发 CR-LDP 连接释放机制。图 B.5 给出了由呼叫方发出的呼叫拆除请求进程。通知消息包含管理状态 TLV 从而在连接释放之前关闭告警。

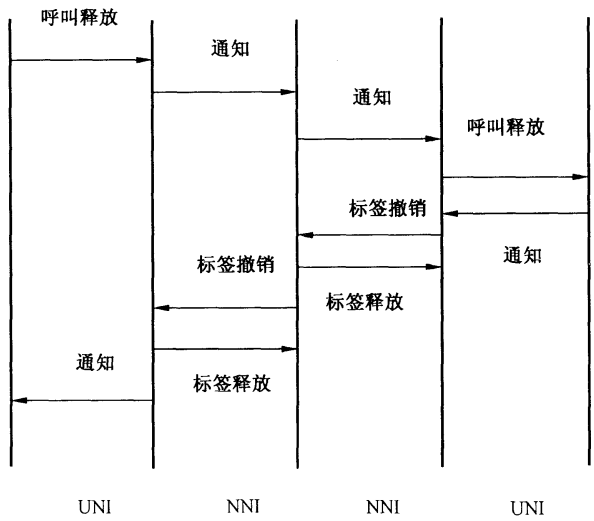


图 B.5　呼叫方发出的呼叫释放

当发出呼叫释放消息的一方接收到通知消息后就认为呼叫拆除过程已完成。在确定所有与该呼叫相关的连接都被删除后，被叫方会发出这个通知消息。

在某些实例中需要不完善的(non-graceful)连接释放(即：预先没有关闭告警)，图 B.6 和图 B.7 给出了由呼叫对应的两个连接发出的不完善的呼叫或连接释放的信令流程。

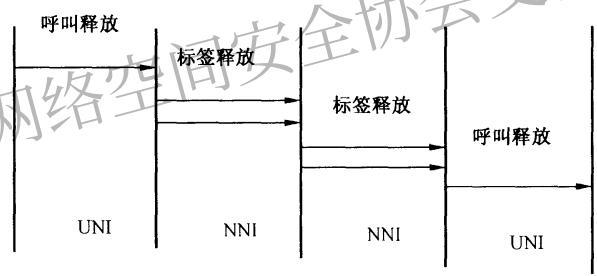


图 B.6　被叫方发起的不完善呼叫或连接释放

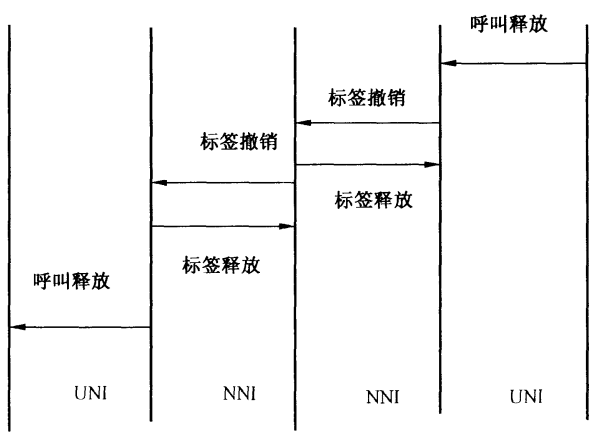


图 B.7　被叫方发起的不完善呼叫或连接释放

B.6.4　采用 CR-LDP 建立连接

采用 CR-LDP 标签请求和标签映射消息建立连接。在传送方向上发出标签请求消息，该消息携带

着相关连接的参数，也可能含有呼叫参数，例如呼叫建立的同时伴随着连接建立的情况。标签请求消息要求在通道经过的每个节点上为被请求的连接分配一个标签。该标签可以是时隙、波长、带宽或物理光纤。

连接建立假设呼叫已经被建立或正处于建立的过程中。标签请求消息包括呼叫和相应连接的呼叫标识。此标识由网络分配，并且在单一网络中(可能包括多个域)是唯一的。

ASON 的连接通常是双向的。正如 GMPLS 中定义的，双向连接是在标签请求消息中的上游标签指示的。宿端接收到标签请求消息就意味着请求成功了，也就是能满足双向连接的所有连接属性。但是，这并不意味着已经完成连接建立并可以传送数据了。只有当中间的交叉连接配置完成以后才真正建立了有效的连接。中间的交叉连接配置都需要占用一定时间来完成，并且根据采用的技术，这种延时会很明显，一般在 10 ms～100 ms 量级。

宿端在建立了自己的交换结构之后会发出一个标签映射消息作为标签请求的响应。如果要求这样做，目的方会通知发起方要求预置确认的指示。通过使用 LDP 通知消息中状态编码“预留确认”完成确认。一般情况下，单向连接不需要确认指示。

在两个相反方向传递双向连接建立请求时会发生标签竞争现象。这种竞争现象会在同一时刻，两边同时分配相同的资源(标签)时发生。为了避免竞争现象，首先会满足具有高级别 ID 的节点，且必须发出“路由问题或标签分配失败”的通知消息。当收到这种出错消息时，另一个节点应尽力给双向通道分配不同的上游标签。但是，如果无法获得其他的资源，节点必须进行标准的错误处理。

图 B.8 给出了采用 CR-LDP 建立连接的过程。

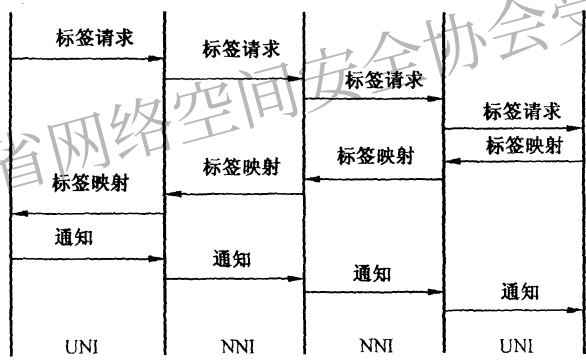


图 B.8 建立连接

连接建立请求失败会有很多种原因，例如：无可获得的带宽或带宽不足，不具备物理连接能力，违反 SLA，远端客户拒绝连接等。在这种情况下，系统会向源端客户发出 LDP 通知消息指示建立请求失败，并以状态码的形式显示失败的原因，例如：无有效资源。图 B.9 给出了被网络拒绝的建立连接请求。

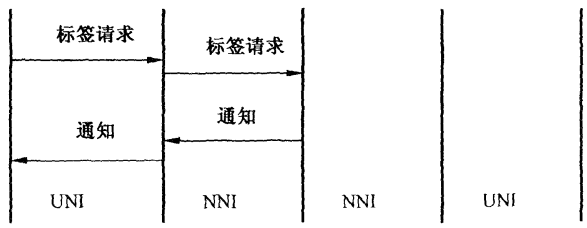


图 B.9 网络拒绝建立连接请求

在发出标签请求消息之后，客户想终止连接建立过程，应发出 RFC3036 的 3.5.9 中定义的 LDP 终止消息。特别指出的是标签请求消息中的 ID 在终止消息中用作临时的本地连接标识符。

B.6.5 采用 CR-LDP 修改连接

CR-LDP 的连接修改功能可以通过呼叫方发出一个新的标签请求消息实现，该标签请求消息包括

被修改的连接标识和说明该消息为修改请求的指示(区别于新的建立请求)。连接 Id TLV 中的激活标志用于指示标签请求消息是一个新的建立,还是一个修改。

连接修改只适用于已存在的连接。标签请求消息中包括一些要进行修改的参数,例如连接业务量参数。连接修改过程与建立一个新的连接过程相同。

B.6.6 采用 CR-LDP 释放连接

LDP 采用两种机制使节点通知它的对等实体停止使用标签。第一种方法是使用标签撤消消息给对等实体发出信号以通知对等实体不必继续使用特定的标签,以指示节点一直在进行广播。第二种方法是采用标签释放消息给对等实体,从而通知它 LSR 不再需要特定的标签来指示对等实体的请求或指示它在进行广播。

G.7713 中扩展 CR-LDP 利用标签释放和标签撤消消息删除连接。所采用的消息取决于发起删除连接的实体。标签撤消消息适用于在上游方向删除连接。根据 RFC3036 3.5.10 中对 LDP 进程的描述,标签释放消息适用于确认删除请求的情况。并且标签释放消息用于在下游方向删除连接。在这种情况下,通过采用 LDP 通知消息中的状态码"删除成功"确认删除请求。

图 B.10 和图 B.11 分别显示了由源端和宿端发起的连接删除请求。图 B.10 显示出删除请求是在接收到具有管理状态 TLV 的 LDP 通知之后发出,该 LDP 通知指示出释放一条连接。在光网络中,光丢失的速度快于删除请求的传播速度,这样下游节点将检测到光丢失,并且因此产生告警。该告警可能会触发不正确的恢复或保护。为了解决这一问题,应沿着连接路由发送通知消息,从而告知将要删除的所有节点。在收到该消息之后,每个节点在所指示的连接上应禁用告警报告和保护机制。

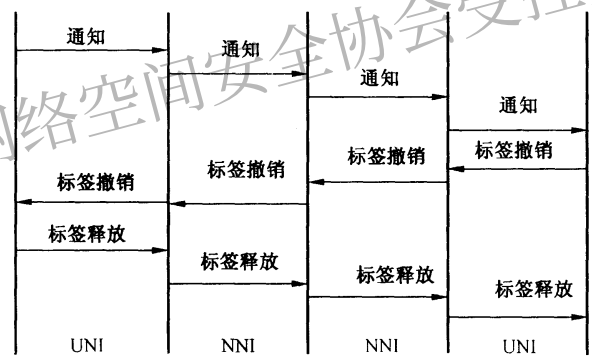


图 B.10 源端发起的连接删除过程

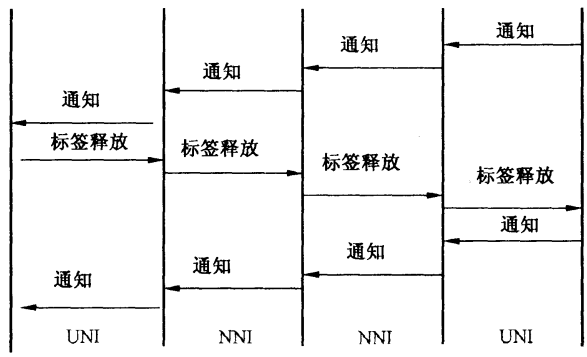


图 B.11 宿端发起的连接删除过程

图 B.12 显示出由网络发起的连接删除过程,以及通过 OAM 实体被迫发出删除请求而发起的连接删除过程。在这种情况下,标签撤消和标签释放消息用于发起删除请求。

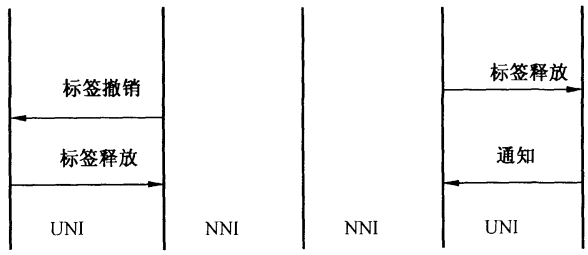


图 B.12　网络发起的连接删除过程

附 录 C
（资料性附录）
回溯（CrankBack）信令机制

C.1 CrankBack 概述

当连接建立请求未成功并从失败点返回建立失败的信息时，回溯（CrankBack）机制允许发起新的连接建立请求，避免资源的阻塞并尝试重新建立连接。回溯机制也可以用于连接的恢复机制。

在分布式的路由环境中，用于约束路径计算的资源信息有可能过期，这就意味着连接建立请求有可能被拒绝。例如，连接选择的路径上的某个节点出现了资源不足的情况。如果路径的源节点或者中间区域的边界节点能够获知资源阻塞节点或者链路的位置，它可以分配一条备用路径，并重新发出建立请求。对阻塞链路或者边界节点的标识与认证可以通过 CrankBack 路由功能实现。

当网络中多个节点或者链路同时发生故障时，分布式路由计算所获取的资源信息和实时的网络状态存在差异，这种情况下，恢复路径可能使用了不正确的信息，因而导致了阻塞，不能建立成功。CrankBack 信令和路由功能应该支持这种情况下的故障恢复。

C.2 CrankBack 功能需求

CrankBack 需要回溯到中间节点（一般是域边界节点）或是回溯到源节点进行重路由。当连接建立失败时，回溯机制允许避开阻塞的节点或链路并重新尝试连接建立。回溯机制还可以用于对连接的保护和恢复，通过指示失效节点或链路的位置，重新尝试连接恢复，提高恢复率。回溯功能应满足以下要求：

1) CrankBack 重试的次数应可设置。
2) 任何一次域内的 CrankBack 和造成 CrankBack 的原因都要向管理平面上报。
3) 信令协议可支持动态的、不同程度的 CrankBack，可以回溯到源节点或中间节点进行重路由。对于回溯到中间节点（一般是域边界节点）的方式，在 CrankBack 过程中信令的控制权应依次向上游传递，由上游节点尝试重新建立这个连接。
4) 具有后向兼容的能力，没有 CrankBack 能力的节点，不对 CrankBack 相关消息做处理，继续向上回溯。
5) 支持域内和域间的 CrankBack。

C.3 CrankBack 信令操作

当连接建立请求由于资源的不可用而导致阻塞时，应该发出错误指示信息，指示阻塞点的位置标识符，返回连接的源节点，区域边界节点、自治区域边界节点或者其他能够进行 CrankBack 的节点。

错误指示信息中应该携带产生错误的原因以及错误发生点的节点或链路信息。CrankBack 机制还需要其他的信息。

对这些错误的指示信息，需要进行保持，具体的操作如下：

1) CrankBack 节点应该存储阻塞点的位置标识信息，直到连接成功的重新建立，或者 CrankBack 节点终止了重路由尝试。
2) 因为 CrankBack 节点有可能发起多次 CrankBack 信令，因此它应该维护所有阻塞节点的历史记录信息，以便在进行路径计算时可以避开所有的阻塞点。
3) 当 CrankBack 节点接收到新的错误指示信息以后，它应该对历史记录信息进行更新，进行下一步的重路由信息计算时可以使用最新的信息。历史记录信息的一个很重要的作用就是能够避免发生重复的 CrankBack 尝试。

在请求建立一条连接时，入口节点应该指定是否需要 CrankBack 机制，并且是否需要有中间节点

发起重路由操作。因此，需要在连接建立请求中添加重路由标志区域，相应的值包括：

1) 没有重路由(No Re-routing)：入口节点在连接建立失败时可以进行重路由尝试，中间节点则不能进行重路由尝试。节点探测到故障以后，必须发送错误指示信息，其中可以包括CrankBack信息。
2) 端到端的重路由(End-to-end Re-routing)：入口节点在连接建立失败时可以进行重路由尝试，中间节点则不能进行重路由尝试。节点探测到故障以后，必须发送错误指示信息，其中应该包括CrankBack信息。
3) 边界重路由(Boundary Re-routing)：如果中间节点是区域边界上的节点，在连接建立失败时它可以进行重路由尝试。边界上的中间节点可以向上游的入口节点发送错误指示信息，也可以向其他出口边界节点发送错误指示信息。其他的中间节点不能进行重路由尝试。节点探测到故障以后，必须发送错误指示信息，其中应该包括CrankBack信息。
4) 基于段的重路由：任何节点在接收到错误指示信息以及向上游转发以前，都可以发起重路由尝试。节点探测到故障以后，必须发送错误指示信息，其中应该包括完全的CrankBack信息。

节点探测到故障以后，执行的操作应该基于连接建立请求消息中携带的重路由标志信息。

——如果是基于段的重路由，探测到故障的节点应该立即发起重路由尝试；

——如果是基于边界的重路由，并且探测到故障的节点为边界节点，此节点应该立即发起重路由尝试；

——如果是基于端到端的重路由操作，或者是在基于段以及基于边界的重路由操作下，探测到故障的节点不能进行重路由尝试(或者是进行了所有的重路由尝试以后仍不成功)，探测到故障的节点应该发出错误指示信息，在其中包括完整的CrankBack信息。

对于重路由尝试操作，应该配置重路由操作尝试的次数上限。这主要是为了过多的消耗网络中的资源，同时可以后退到其他的修复点进行重路由操作尝试。

C.4 CrankBack信令扩展

RSVP-TE协议可以用于CrankBack的重路由机制：

1) 连接建立请求在某些情况下，可能会失败，如在链路故障情况下或者发生了资源不可用的情况。当连接建立失败时，应向连接的入口返回PathErr消息；
2) 资源预留过程中也有可能会发生失败的情况。在Resv处理过程中，如果发生了链路和资源节点不可用的情况，会向出口节点发送ResvErr消息，指示“接纳控制失效”。出口节点运行改变FLOWSPEC参数，并且重新进行资源预留尝试。如果不支持资源预留重试操作，出口节点应该执行下列操作之一：
   ——忽略此失效状况，通过Path刷新消息或者消息刷新的过期来进行恢复操作；
   ——发送PathErr消息到入口节点，指示“接纳控制失效”。

在RSVP-TE中，失败的连接建立尝试会导致发送PathErr消息到上游节点。PathErr消息中应携带错误编码(ERROR_SPEC)对象，指示某些节点或者接口导致了连接建立失败。CrankBack在进行重路由时，会避免通知中指出的节点或者接口。

GMPLS-RSVP-TE对上述的错误通知进行扩展，在指示错误的接口信息之外，还上报发出错误指示的节点的信息。

GMPLS引入了一个通知(Notify)消息，可以向一个指定的节点报告连接建立失败。此消息可以携带与上述描述相同的信息。此通知消息可以加速错误信息的传播，但是当网络中的多个节点具备CrankBack功能时，在各个节点之间需要协商确立是由PathErr消息还是由通知消息触发CrankBack操作。这是因为由于通知消息的目标节点有可能不是入口节点，同时多个信息在到达入口节点时会沿着不同的路径传输，这会导致多个节点尝试重新建立LSP。

RSVP-TE定义了新的错误编码信息——“重路由操作越限”，指示某个节点不能进行重路由操作，因为它超过了重路由操作尝试的次数。

参 考 文 献

[1] ITU-T Q.2931(06/1995) 宽带综合服务数字网络(B-ISDN)——数字信令系统 第2部分(DSS2)——用于基本呼叫和连接的用户网络接口(UNI)3层规范

[2] IETF RFC3036 标签交换协议(LDP)规范

[3] IETF RFC 3212 采用LDP建立基于约束的LSP

中华人民共和国
国家标准
自动交换光网络(ASON)技术要求
第4部分:信令技术
GB/T 21645.4—2010
*
中国标准出版社出版发行
北京复兴门外三里河北街16号
邮政编码:100045
网址 www.spc.net.cn
电话:68523946 68517548
中国标准出版社秦皇岛印刷厂印刷
各地新华书店经销
*
开本 880×1230 1/16 印张 6.75 字数 191千字
2011年2月第一版 2011年2月第一次印刷
*
书号:155066·1-41253 定价 87.00元

GB/T 21645.4-2010