

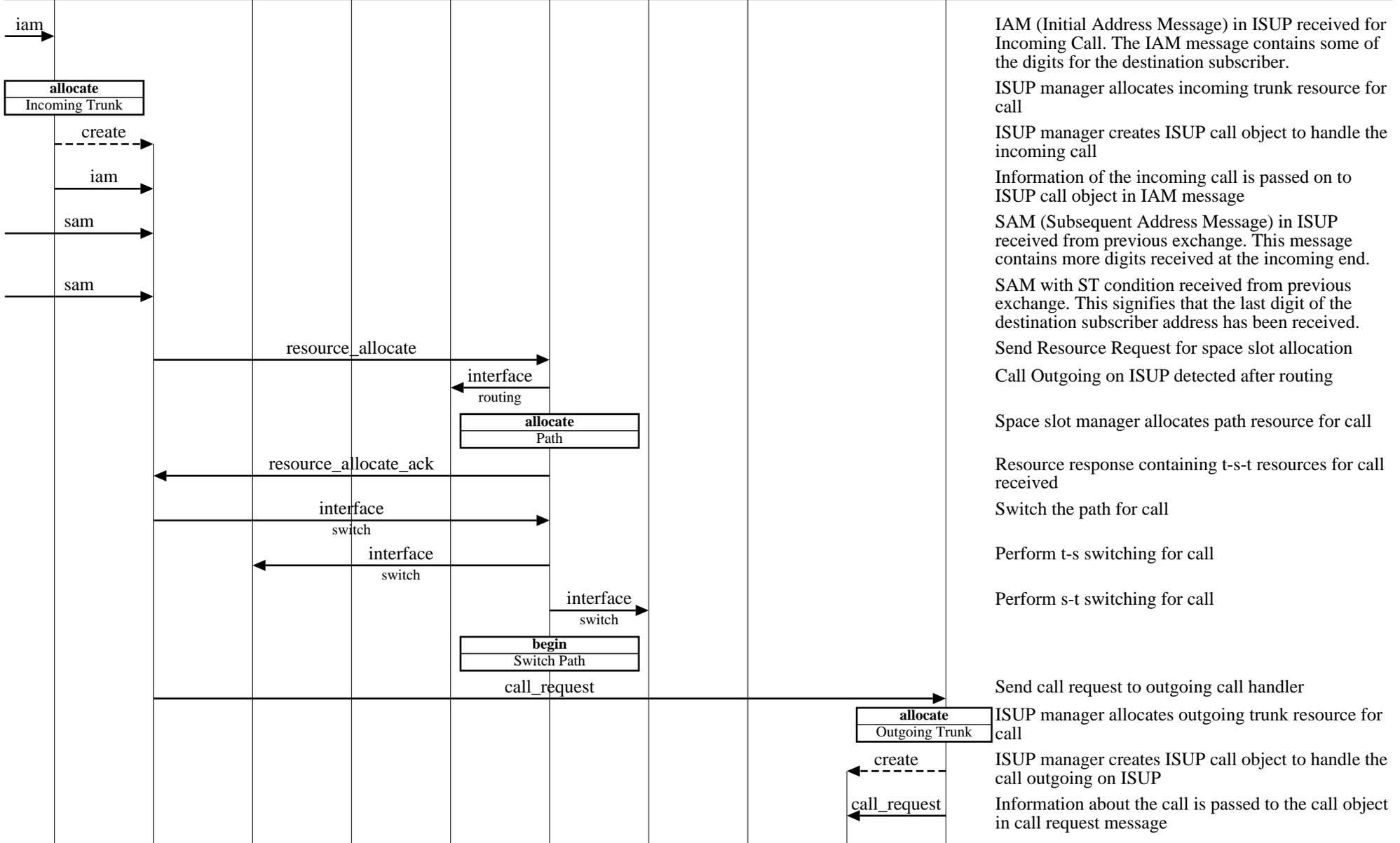
Incoming ISUP and Outgoing ISUP Transit Call (ISUP to ISUP Successful Call - Calling Subscriber Releases First)

switching system										EventHelix.com/EventStudio 1.5	
call handler1				central processor			call handler2			12-Jun-02 22:58 (Page 1)	
isup mgr1	isup call1	ts mgr1	tone mgr1	rout mgr	ss mgr	ts mgr2	tone mgr2	isup call2	isup mgr2		

Copyright (c) 2002 EventHelix.com. All Rights Reserved.

This Message Sequence Chart describes an ISUP (ISDN User Part) call. The call is incoming ISUP and after routing it is sent out on an ISUP trunk

Call incoming on ISUP



IAM (Initial Address Message) in ISUP received for Incoming Call. The IAM message contains some of the digits for the destination subscriber.

ISUP manager allocates incoming trunk resource for call

ISUP manager creates ISUP call object to handle the incoming call

Information of the incoming call is passed on to ISUP call object in IAM message

SAM (Subsequent Address Message) in ISUP received from previous exchange. This message contains more digits received at the incoming end.

SAM with ST condition received from previous exchange. This signifies that the last digit of the destination subscriber address has been received.

Send Resource Request for space slot allocation

Call Outgoing on ISUP detected after routing

Space slot manager allocates path resource for call

Resource response containing t-s-t resources for call received

Switch the path for call

Perform t-s switching for call

Perform s-t switching for call

Send call request to outgoing call handler

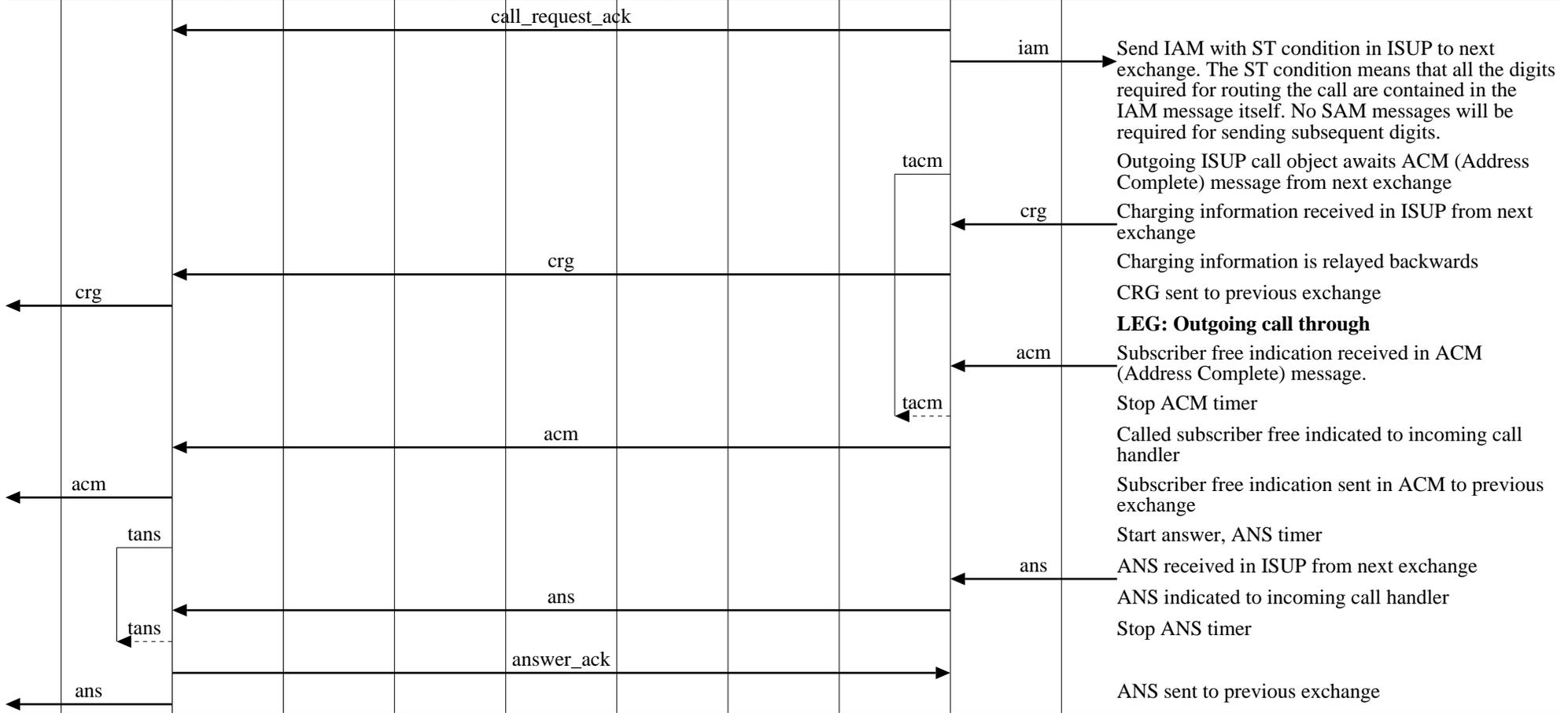
ISUP manager allocates outgoing trunk resource for call

ISUP manager creates ISUP call object to handle the call outgoing on ISUP

Information about the call is passed to the call object in call request message

Incoming ISUP and Outgoing ISUP Transit Call (ISUP to ISUP Successful Call - Calling Subscriber Releases First)

switching system										EventHelix.com/EventStudio 1.5	
call handler1				central processor			call handler2			12-Jun-02 22:58 (Page 2)	
isup mgr1	isup call1	ts mgr1	tone mgr1	rout mgr	ss mgr	ts mgr2	tone mgr2	isup call2	isup mgr2		



Send IAM with ST condition in ISUP to next exchange. The ST condition means that all the digits required for routing the call are contained in the IAM message itself. No SAM messages will be required for sending subsequent digits.

Outgoing ISUP call object awaits ACM (Address Complete) message from next exchange

Charging information received in ISUP from next exchange

Charging information is relayed backwards

CRG sent to previous exchange

LEG: Outgoing call through

Subscriber free indication received in ACM (Address Complete) message.

Stop ACM timer

Called subscriber free indicated to incoming call handler

Subscriber free indication sent in ACM to previous exchange

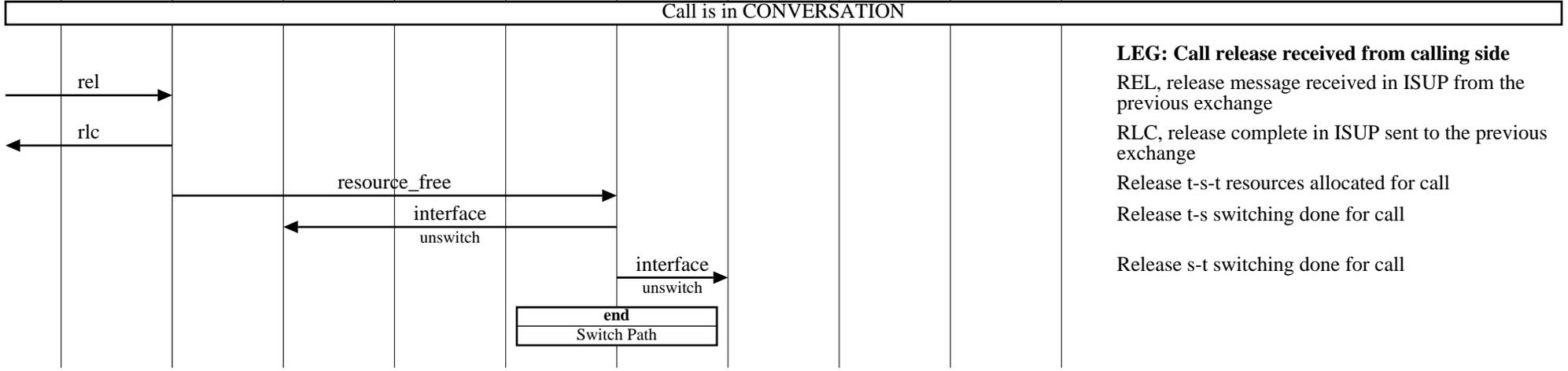
Start answer, ANS timer

ANS received in ISUP from next exchange

ANS indicated to incoming call handler

Stop ANS timer

ANS sent to previous exchange



LEG: Call release received from calling side

REL, release message received in ISUP from the previous exchange

RLC, release complete in ISUP sent to the previous exchange

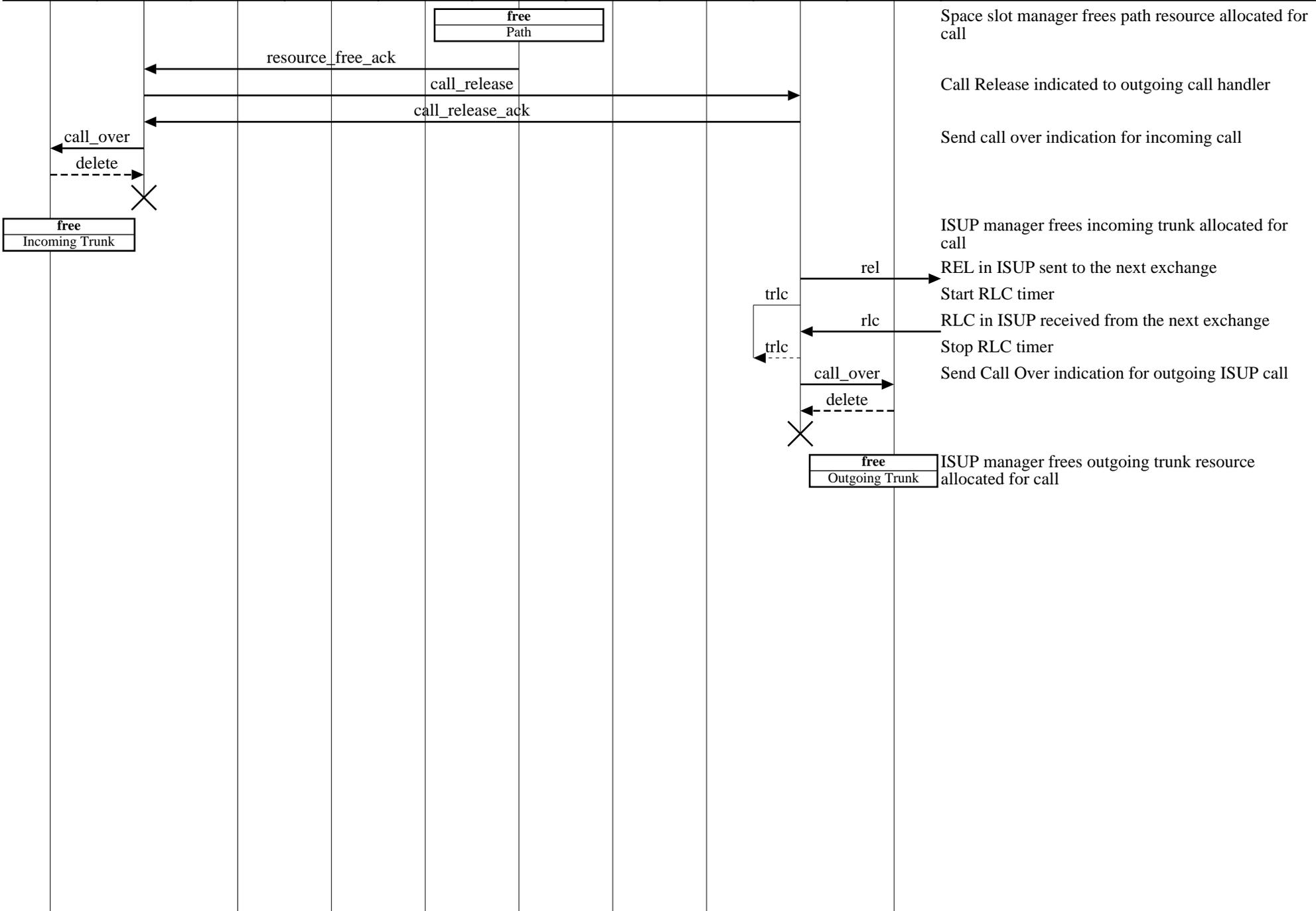
Release t-s-t resources allocated for call

Release t-s switching done for call

Release s-t switching done for call

Incoming ISUP and Outgoing ISUP Transit Call (ISUP to ISUP Successful Call - Calling Subscriber Releases First)

switching system										EventHelix.com/EventStudio 1.5		
call handler1				central processor			call handler2				12-Jun-02 22:58 (Page 3)	
isup mgr1	isup call1	ts mgr1	tone mgr1	rout mgr	ss mgr	ts mgr2	tone mgr2	isup call2	isup mgr2			



Space slot manager frees path resource allocated for call

Call Release indicated to outgoing call handler

Send call over indication for incoming call

ISUP manager frees incoming trunk allocated for call

REL in ISUP sent to the next exchange

Start RLC timer

RLC in ISUP received from the next exchange

Stop RLC timer

Send Call Over indication for outgoing ISUP call

ISUP manager frees outgoing trunk resource allocated for call