



SIP 协议分析

— SIP 协议基础架构

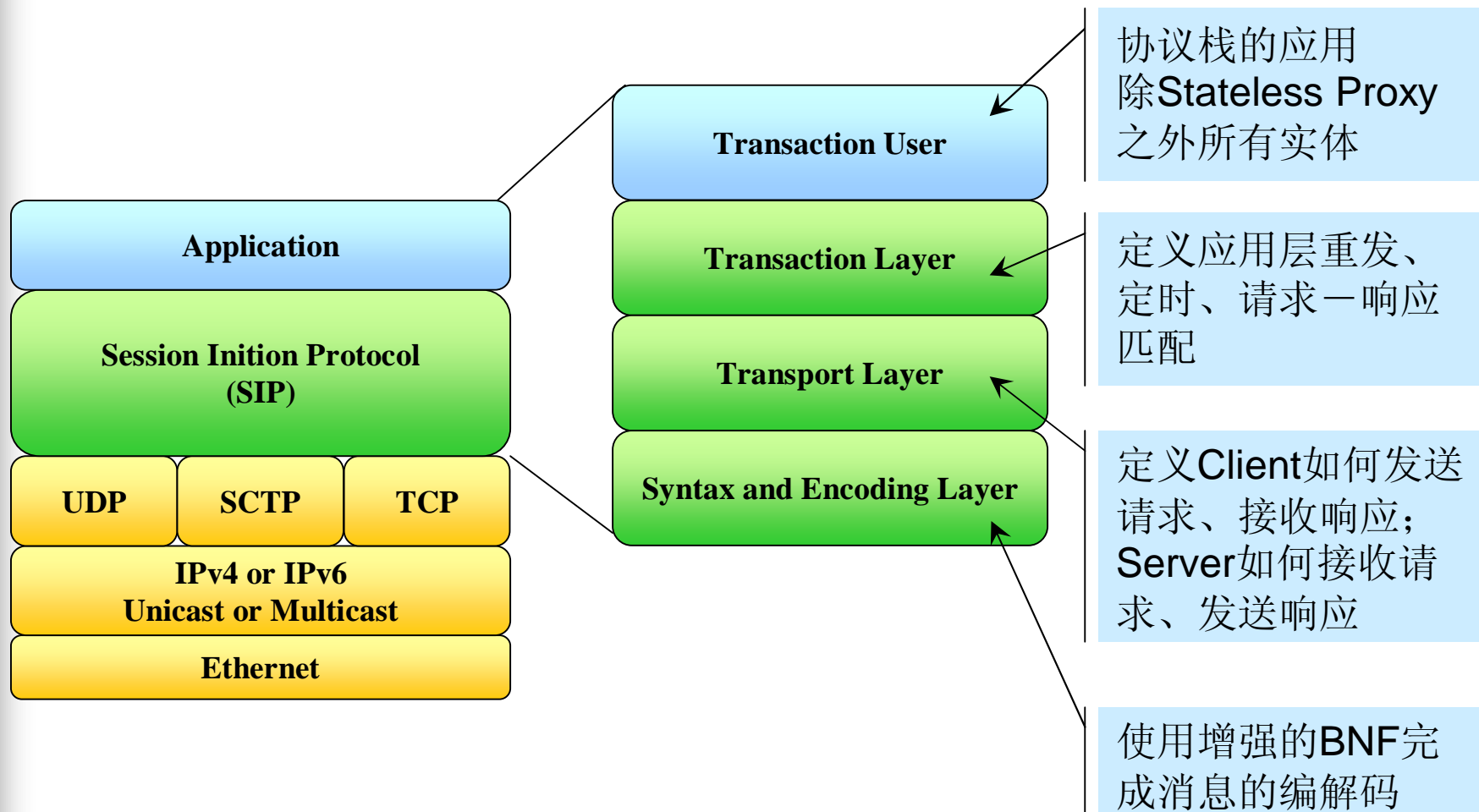
理解协议处理的基本原理与实现模型



- 协议的结构
- 消息的处理规则
- 用户管理
- 网络组网与路由
- 媒体的协商与建立



SIP协议分层结构

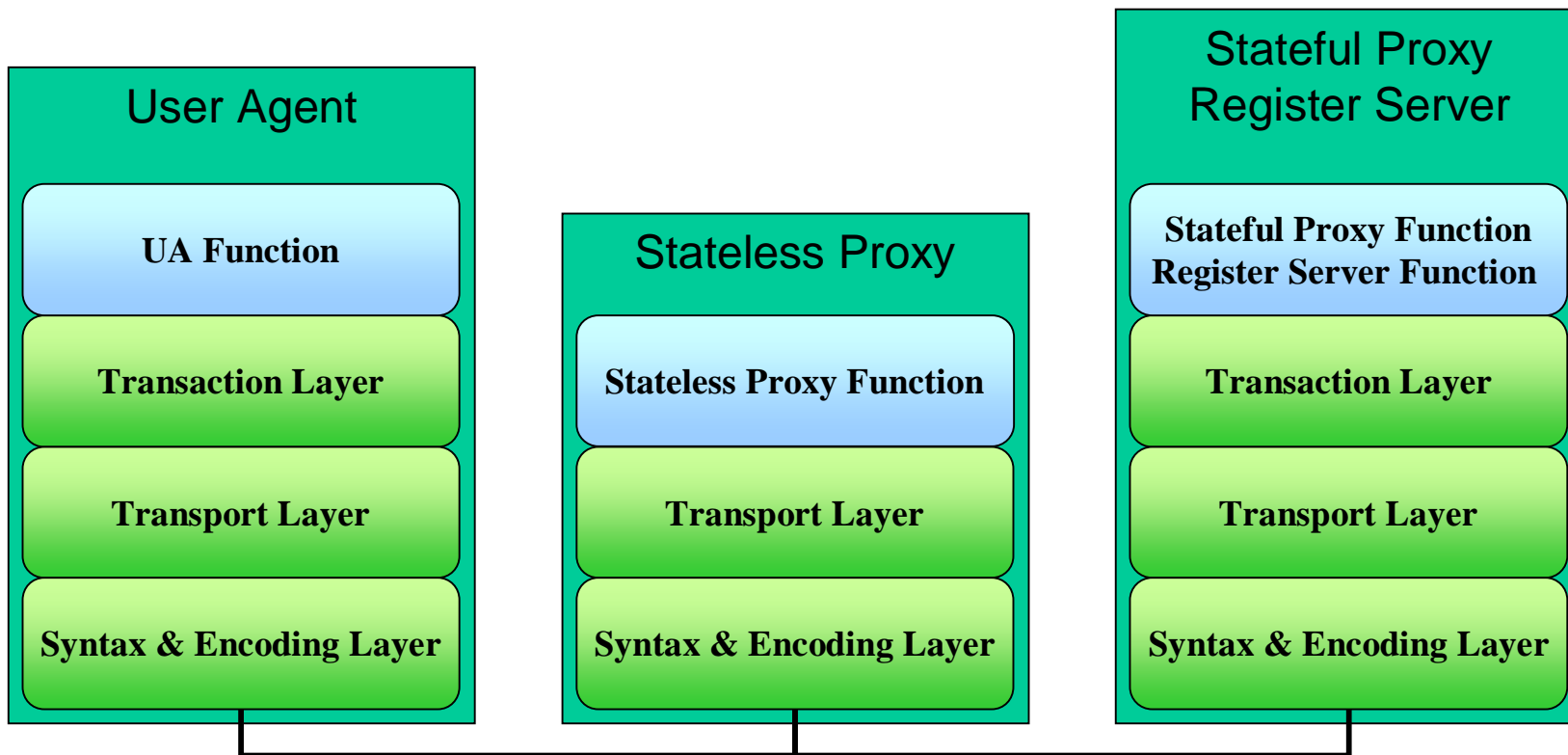


Backus-Naur Form grammar (BNF).



SIP协议功能实体

- 逻辑设备与物理设备
- 应用可以变换角色，可以变换使用的层次
- Stateless Proxy 不包含Transaction Layer





消息的处理规则

- Three-Way Handshake
- Transaction
 - Transaction的定义
 - Transaction的分类
 - Transaction的处理规则
- Dialog
 - Dialog的定义
 - Dialog的处理规则
- Session
 - Session的定义



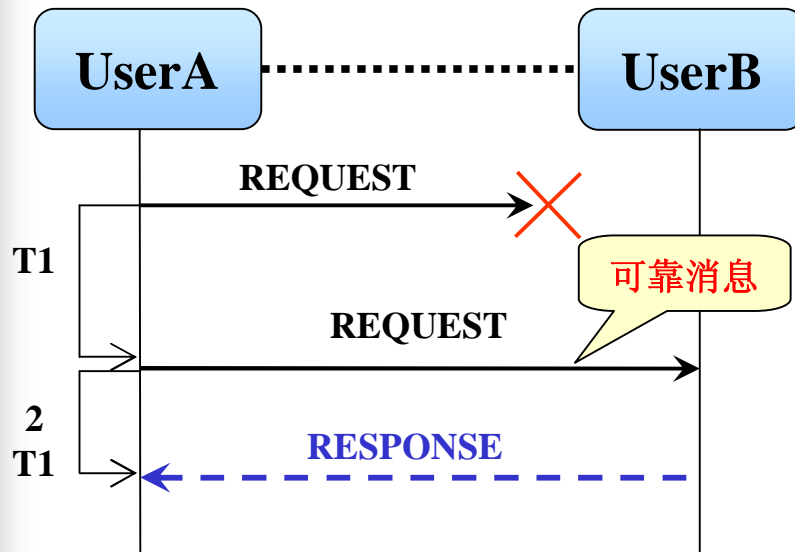
Three-Way Handshake

- Internet条件下，网络状况差异较大，如何保证信令可靠传输？
 - 传输层保证
 - TCP数据可靠传输但并不适合信令协议
 - TCP连接建立时间长
 - TCP协议数据传递的滑动窗口机制
 - UDP协议不保证可靠传输
 - Sigtran是全新协议，应用有限
 - 应用层保证
 - 消息确认
 - 超时重传
- 保证了可靠传输的消息 – 可靠消息
– 不能保证可靠传输的消息 – 不可靠消息

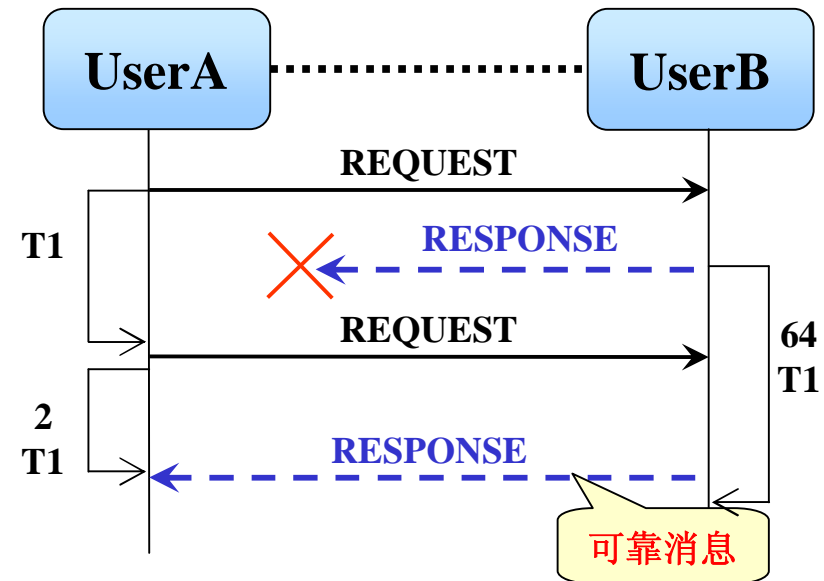
“请求—响应”消息的可靠传输



Two-Way Handshake



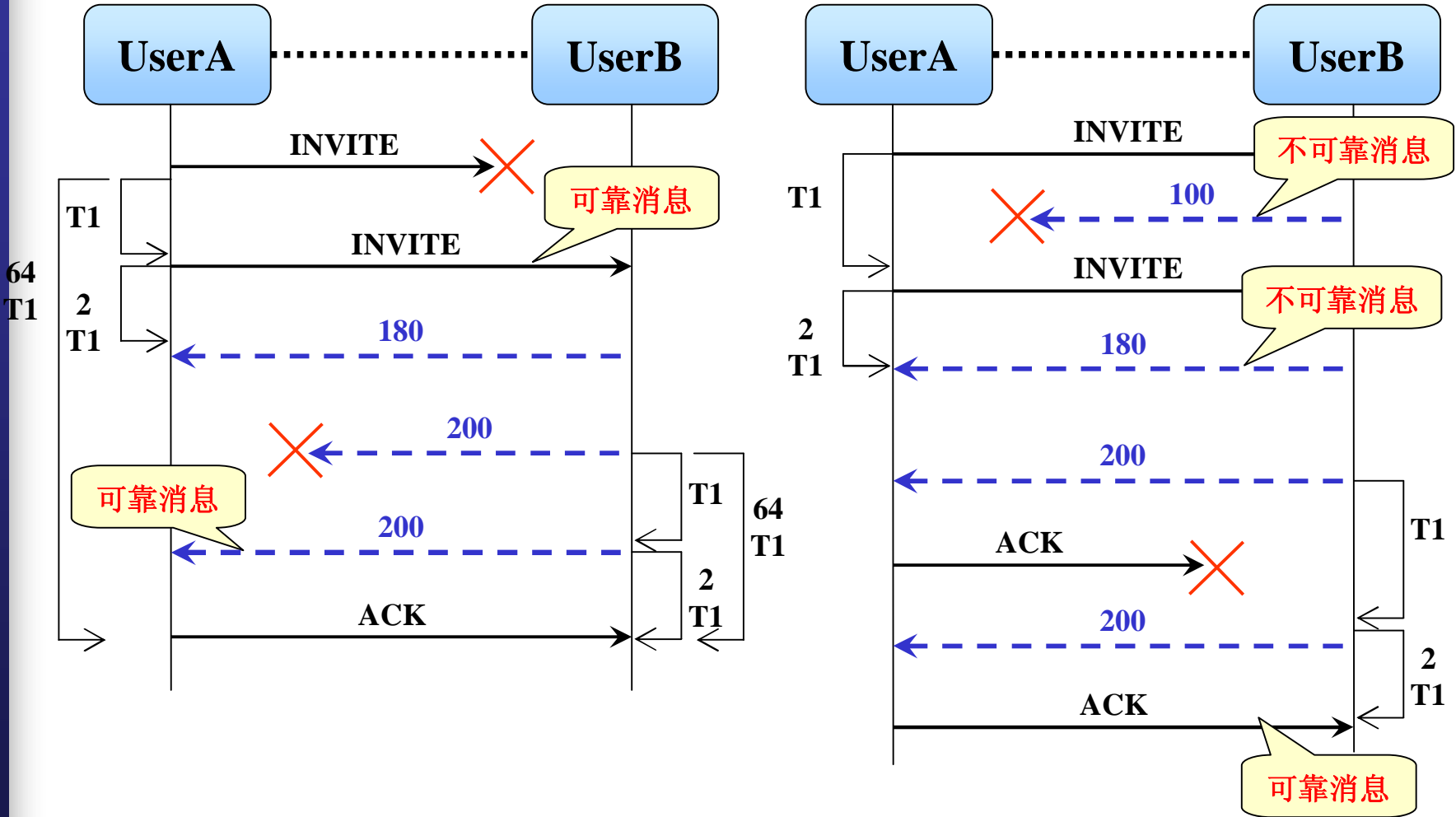
- 1、发出请求之后启动定时器 $T1$ ，等待响应
- 2、 $T1$ 超时，请求“可能”丢失，重发
- 3、重发消息的同时启动定时器 $2T1$
- 4、 $T1$ 、 $2T1$ 、 $4T1$ 、 $8T1$
- 5、请求为可靠消息



- 1、发出响应之后启动定时器 $64T1$ ，等待可能的重发
- 2、收到重发的请求，重发该请求的响应
- 3、 $64T1$ 超时，结束



三次握手 Three-Way Handshake

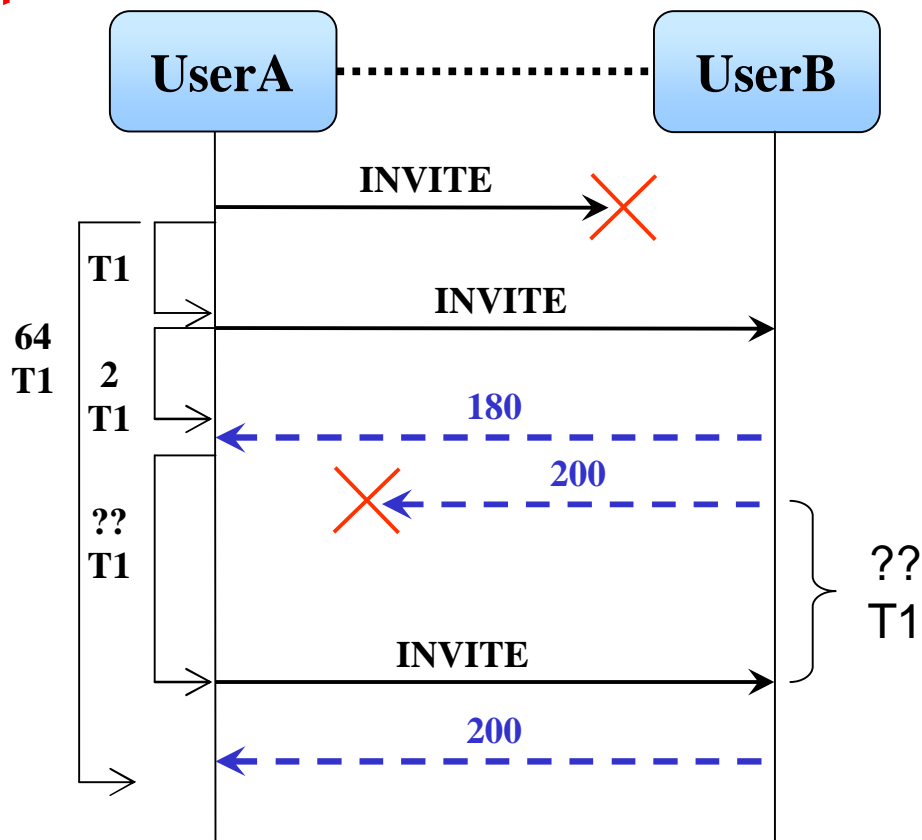


为什么不使用“请求-响应”的可靠传输保证方式？



使用三次握手的原因

- “请求—响应”的基础是**定时器超时重发**，这需要服务侧对请求有较快的响应时间
- 会话建立请求的响应复杂，**最终响应可能不会立即返回**





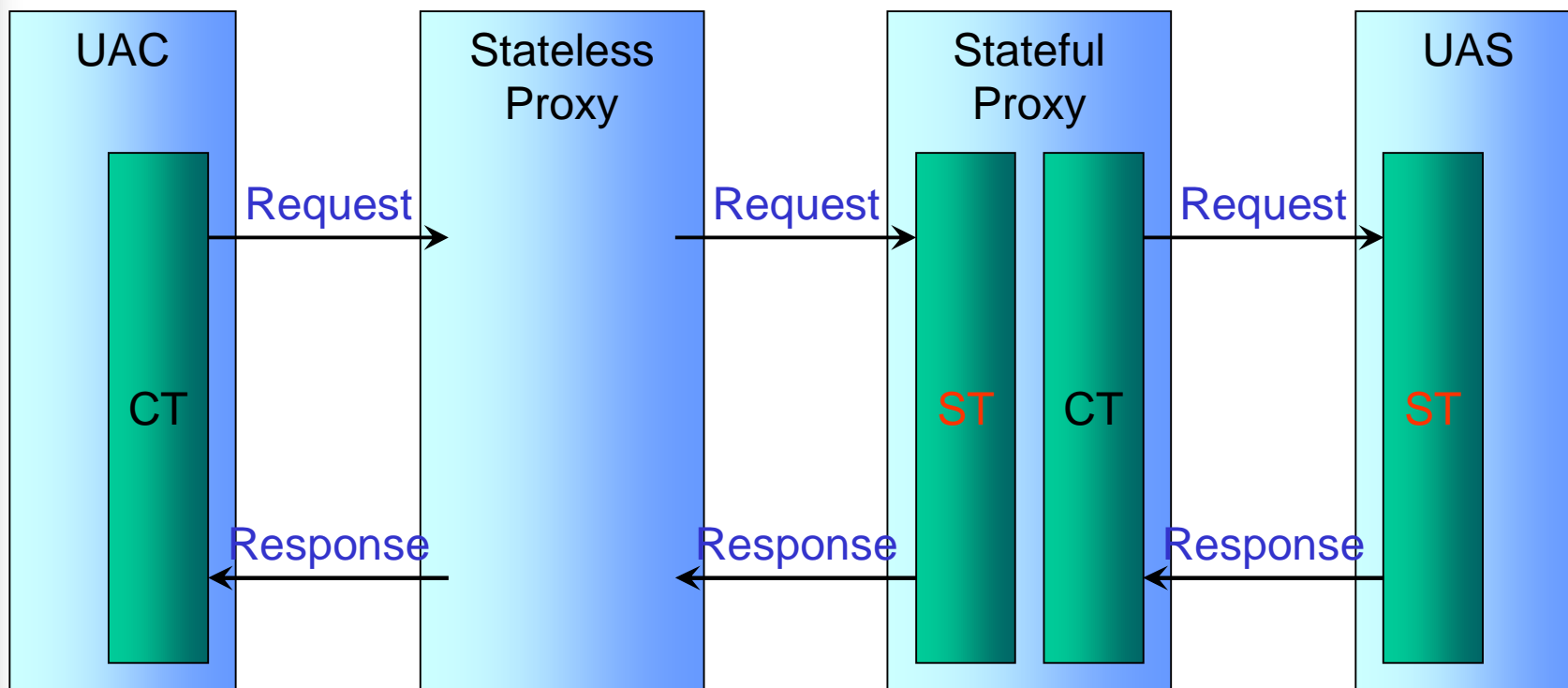
Transaction

- Transaction是Transaction Layer的概念
- Transaction用于应用层重发、定时和请求与响应的匹配
- A SIP transaction occurs between a client and a server and comprises all messages from the **first request** sent from the client to the server up to a **final (non-1xx) response** sent from the server to the client
- SIP协议是基于事务（Transaction）的协议
- 2xx响应是 End-End 的
- 3xx~6xx响应是 Hop-Hop 的



Transaction之间的关联

- Transaction划分为 **Client Transaction (CT)**和**Server Transaction (ST)** C/S结构
- Stateful Proxy同时包含CT和ST
- Stateless Proxy不包含Transaction
- User Agent在特定时刻只能包含CT或ST中的一种





Transaction的分类

- INVITE Transaction
 - INVITE请求及其响应创建的Transaction
 - 在收到200响应的时候，ACK属于本Transaction
 - 在收到非200的最终响应的时候，ACK不属于本Transaction
 - ACK不会创建新的Transaction
 - CANCEL属于INVITE消息建立的Transaction
- Regular Transaction:
 - A regular transaction is any transaction with a method other than INVITE, ACK, or CANCEL



Transaction的标识

INVITE sip:werner.heisenberg@munich.de SIP/2.0

Via: SIP/2.0/UDP 100.101.102.103:5060;**branch=z9hG4bKmp17a**

Max-Forwards: 70

To: Heisenberg <sip:werner.heisenberg@munich.de>

From: E. Schroedinger <sip:schroed5244@aol.com>;tag=42

Call-ID: 10@100.101.102.103

CSeq: 1 INVITE

消息的序号

Contact: <sip:schroed5244@pc33.aol.com>

Content-Type: application/sdp

Content-Length: 159

Transaction标识

SIP/2.0 200 OK

Via: SIP/2.0/UDP 100.101.102.103:5060;**branch=z9hG4bKmp17a**

To: Heisenberg <sip:werner.heisenberg@munich.de>;tag=314159

From: E. Schroedinger <sip:schroed5244@aol.com>;tag=42

Call-ID: 10@100.101.102.103

CSeq: 1 INVITE

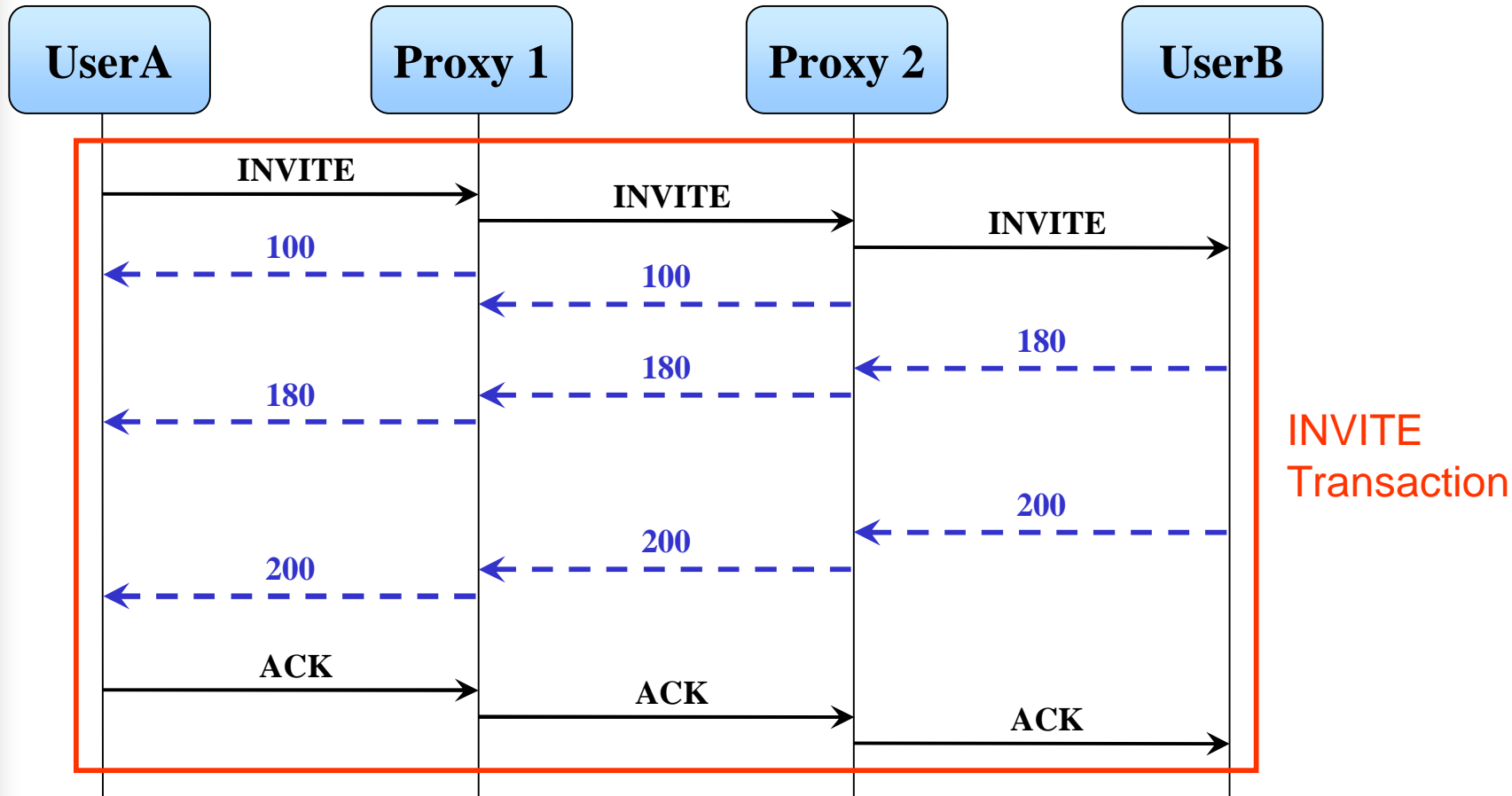
Contact: <sip:werner.heisenberg@200.201.202.203>

Content-Type: application/sdp

Content-Length: 159

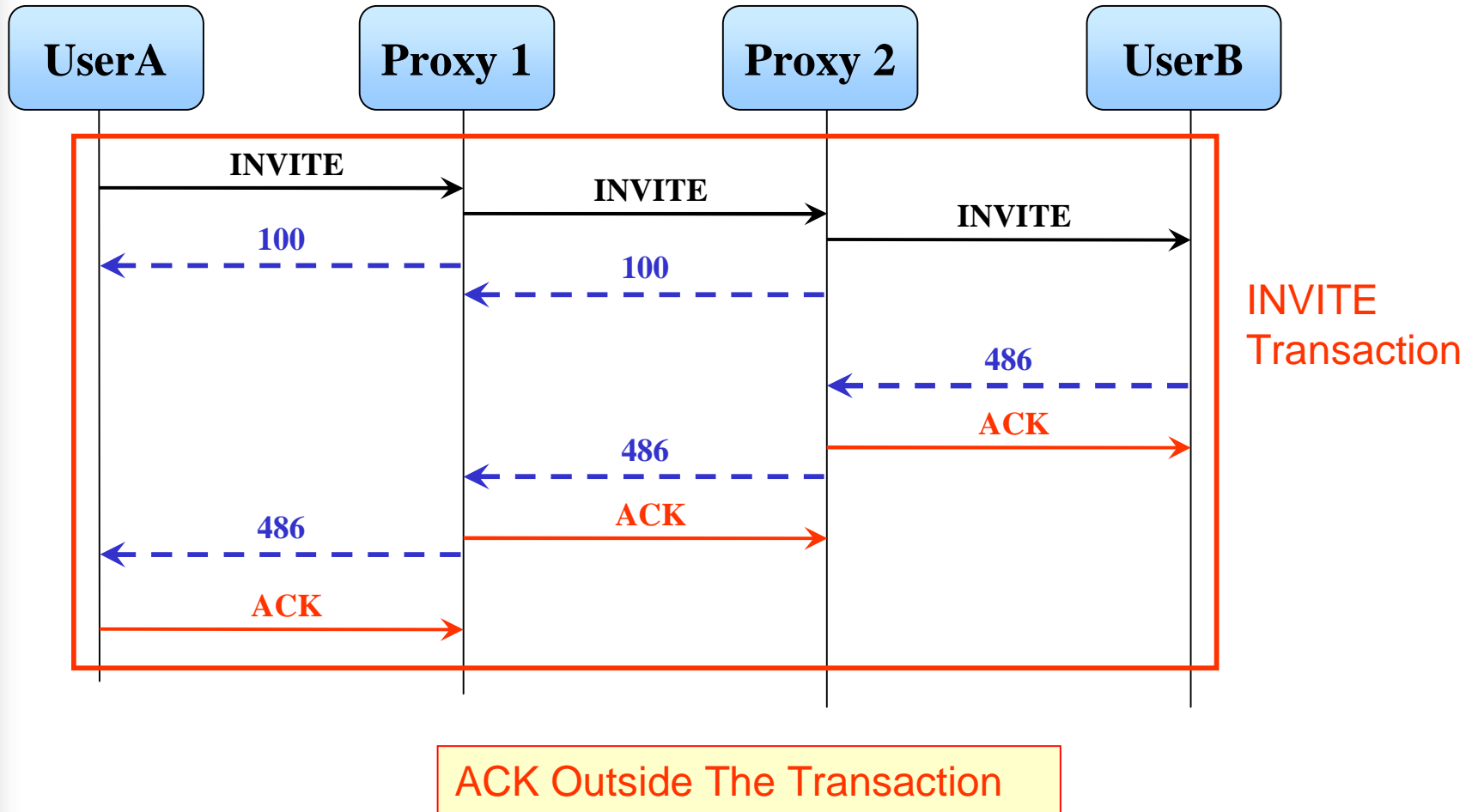


INVITE Transaction



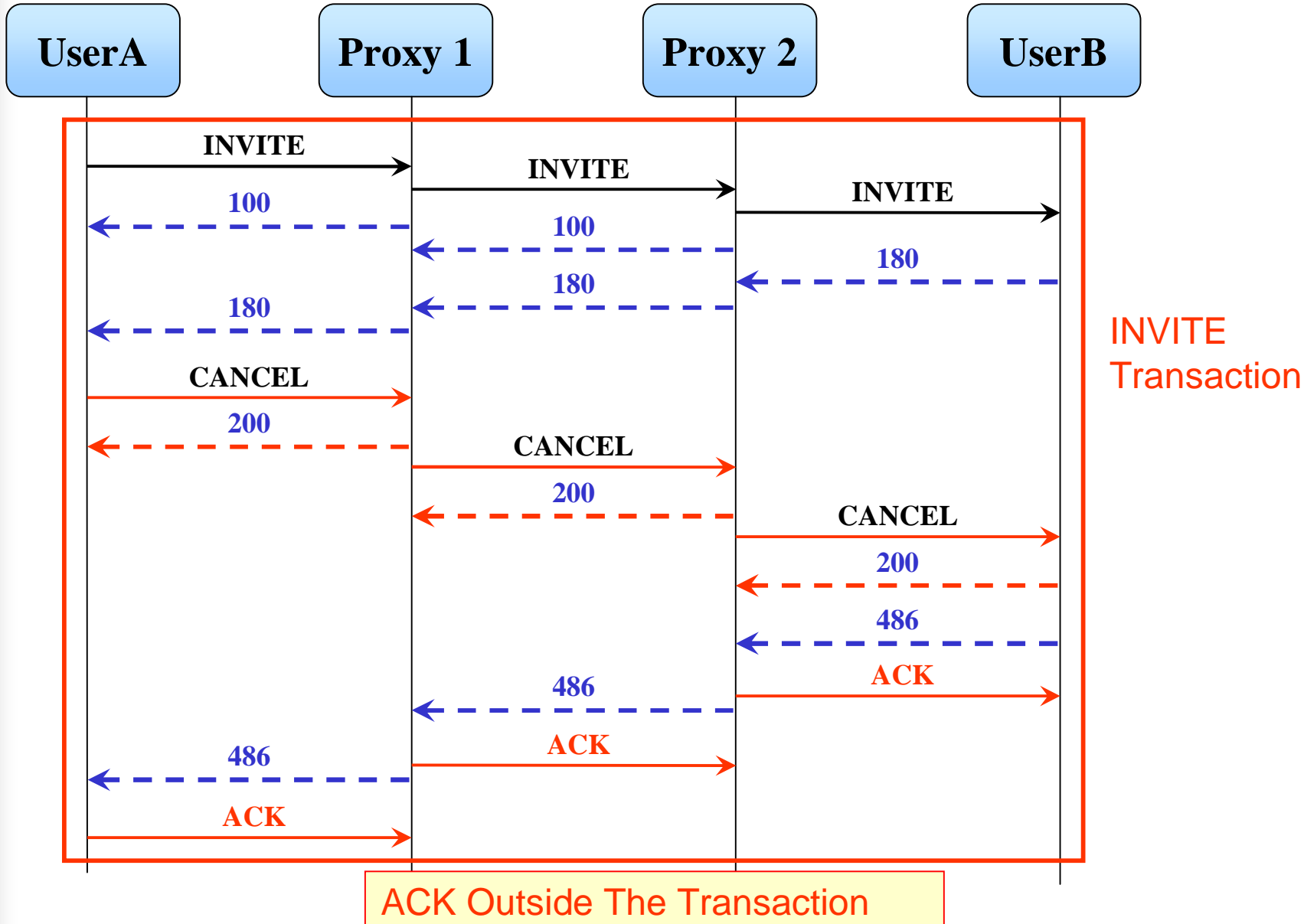


INVITE non-200 Transaction



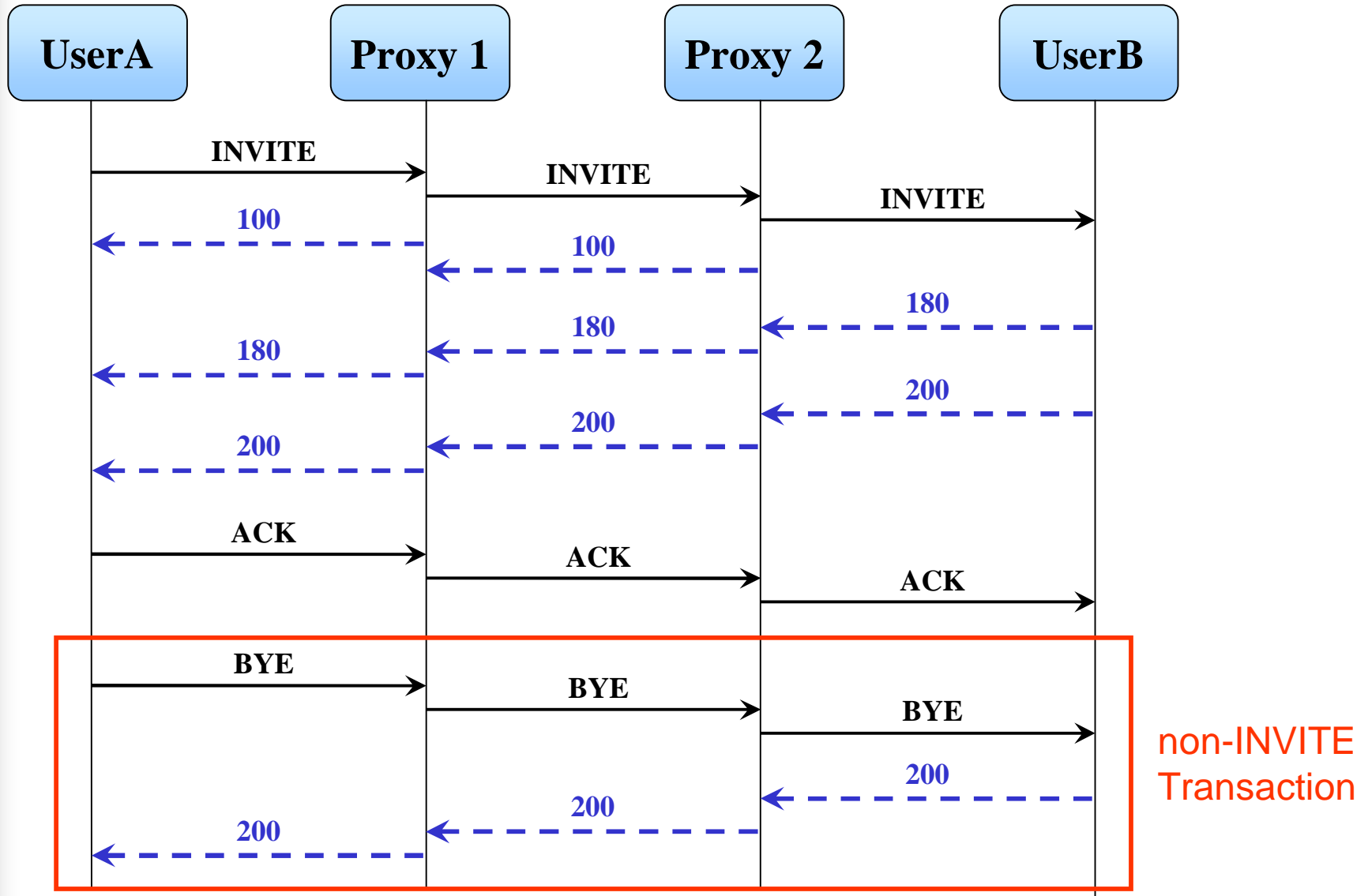


INVITE CANCEL Transaction





non-INVITE Transaction



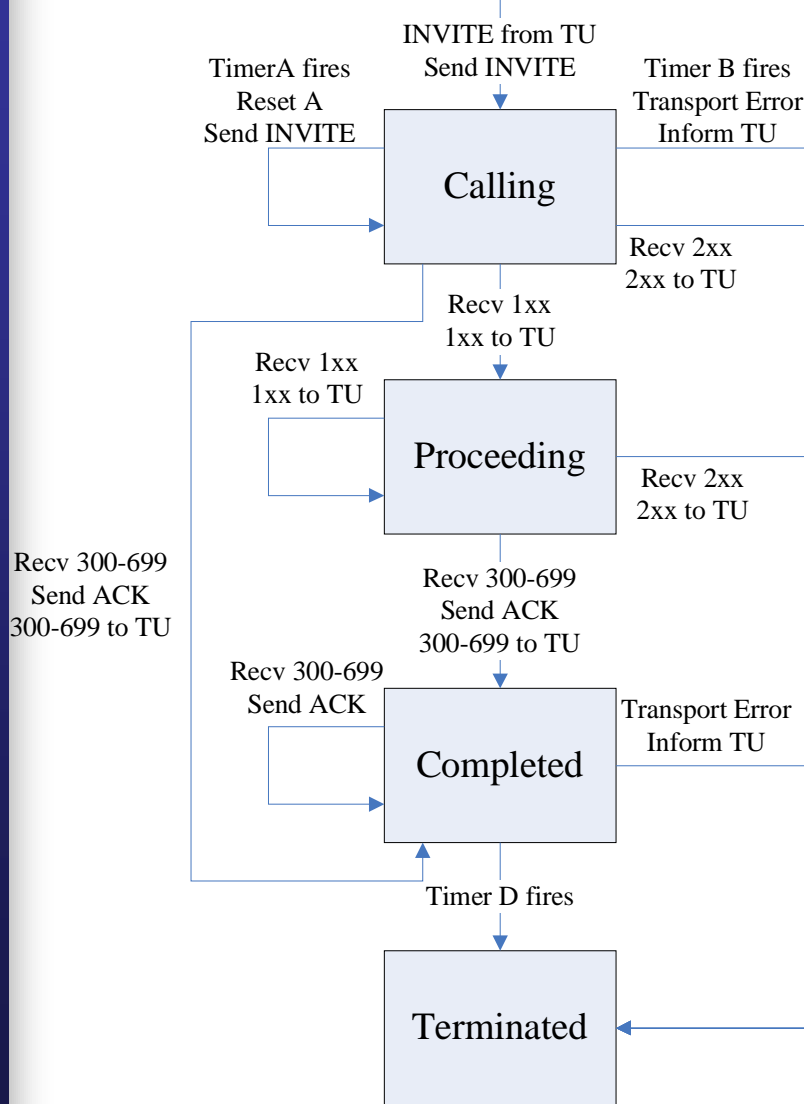


Transaction分类划分原因

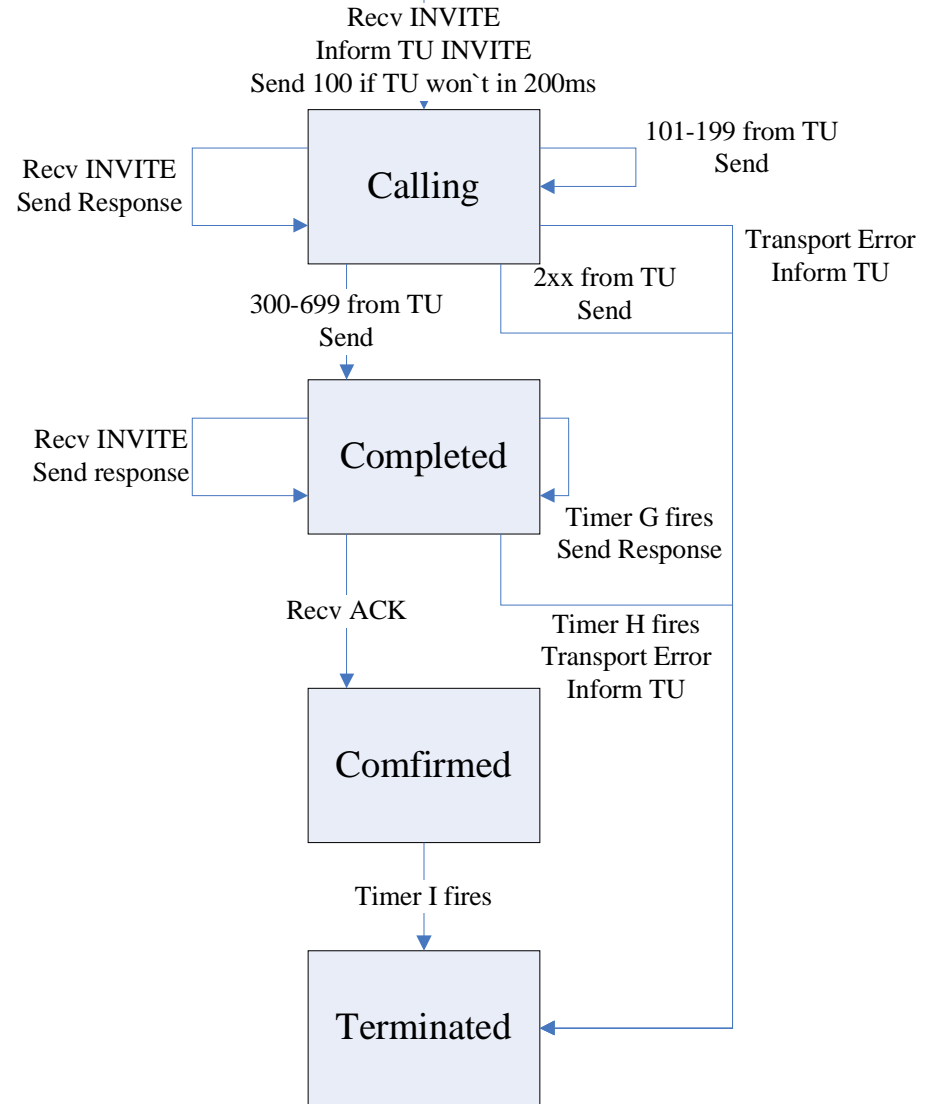
- INVITE Transaction
 - 对应于INVITE三次握手处理规则
- Regular Transaction
 - 对应于标准的“请求—响应”处理规则



Transaction状态机



INVITE Client Transaction



INVITE Server Transaction



ACK的状态哪里去了？

- INVITE Client Transaction收到3xx~6xx响应之后，发送ACK，跳转到Completed状态
- INVITE Client Transaction收到2xx响应之后，跳转到了Terminated状态，那么谁发送ACK？
- INVITE Client Transaction收到3xx~6xx之后发送了ACK，为什么还说ACK不属于INVITE Transaction？
- INVITE Client Transaction收到2xx之后没有发送ACK，为什么还说ACK属于INVITE Transaction？
- Answer:
 - 200响应的ACK由TU维护状态
 - ACK是没有响应的，因此不能创建新的Transaction
 - 200是End-End的，ACK归属于要确认的Transaction

 - 3xx~6xx的ACK由Transport Layer维护
 - 3xx~6xx是Hop-Hop的，因此ACK虽然不归属于要确认的Transaction，但具有相同的参数

 - TU是看不到错误响应的ACK的



CANCEL的状态哪里去了？

- 既然CANCEL属于INVITE Transaction，为什么在Client Transaction和Server Transaction中都没有CANCEL的处理呢？
- Answer:
 - CANCEL不会创建一个新的Transaction
 - TU将根据存在的Transaction填写CANCEL的消息头，并直接通过Transport Layer发送
 - CANCEL是 Hop-Hop 的，Server端判断CANCEL的Transaction存在，将直接返回200 OK
 - INVITE的Transaction将由UAS的487响应终结



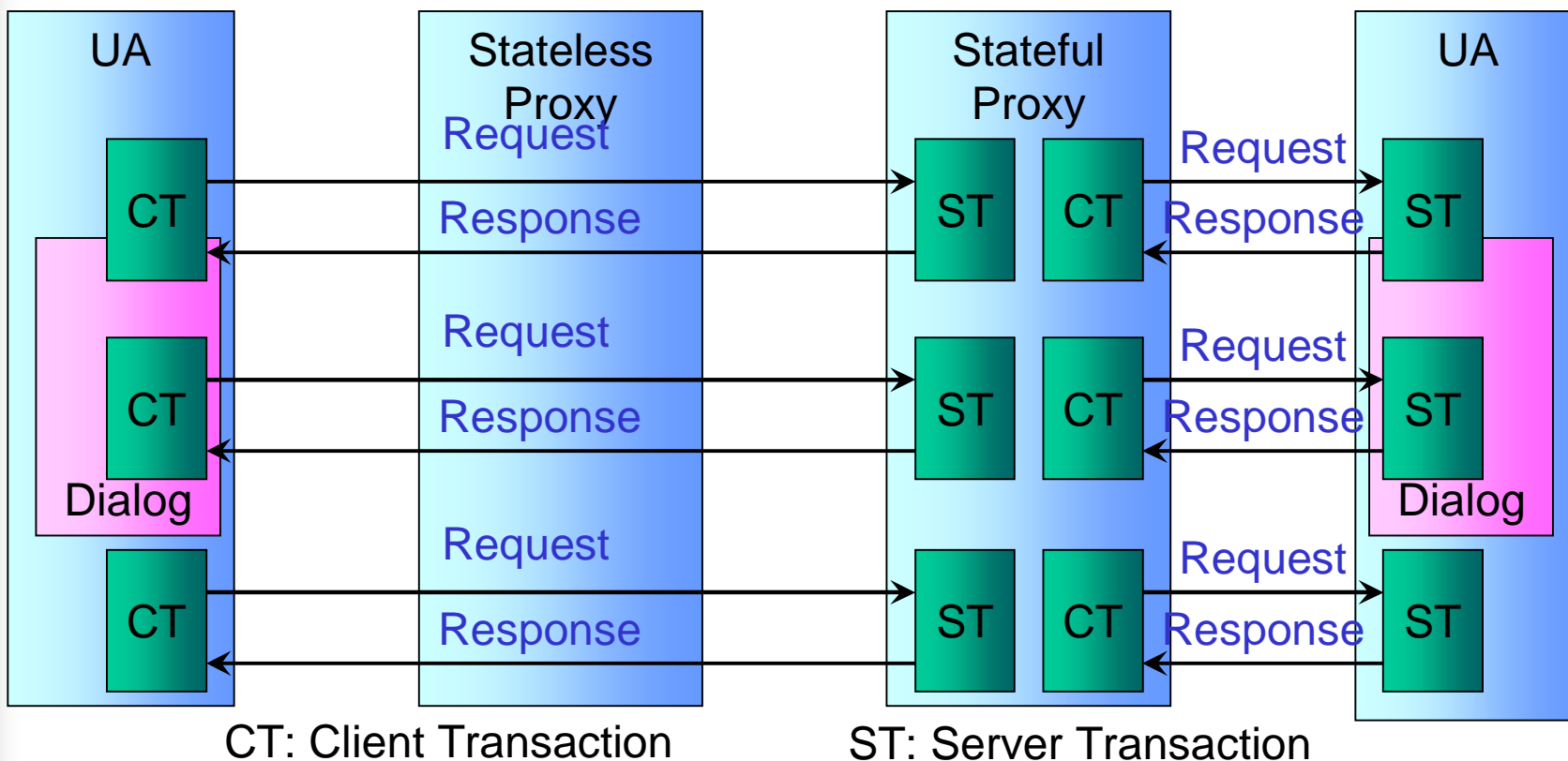
Dialog

- Dialog是Transaction User Layer的概念
- Dialog用于维护**UA之间**的状态
- A dialog is a **peer-to-peer** SIP relationship between two **UAs** that persists for some time.
- A dialog is **established** by SIP messages, such as a **2xx response to an INVITE** request.
- the **BYE** method **terminates** a session and the dialog associated with it



Dialog与Transaction之间的关系

- Dialog是UA-UA之间的对话关系 — 类比Call (Call Leg)
- Transaction: 消息处理层次; Dialog: 对话方关联层次
- 标准的SIP Server是不维护Dialog的
- 三类Transaction: 建立Dialog的, Dialog内的, Dialog外的



Dialog的标识

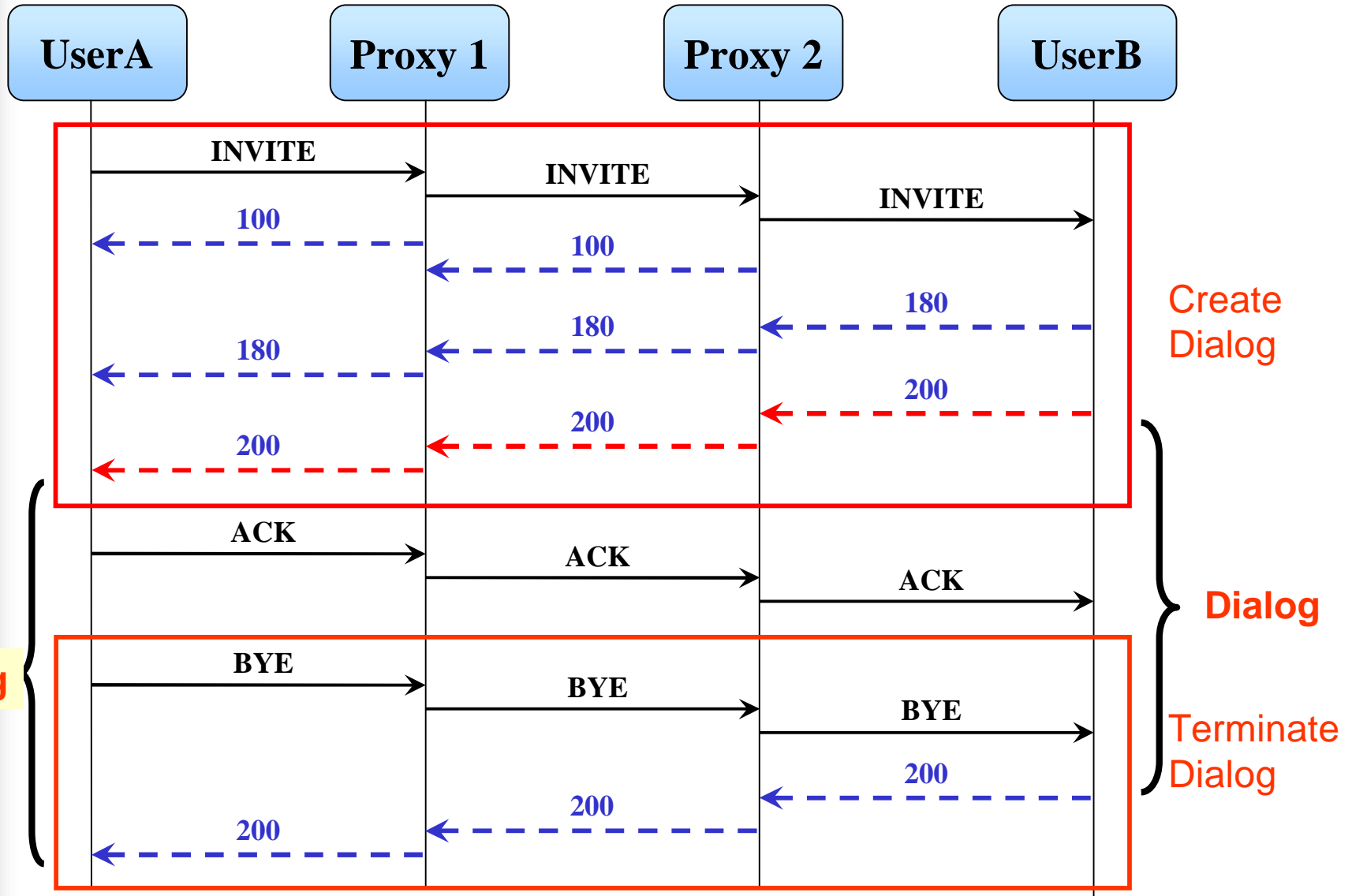


A dialog is identified by a **call identifier**, **local tag**, and a **remote tag**.

```
INVITE sip:werner.heisenberg@munich.de SIP/2.0
Via: SIP/2.0/UDP 100.101.102.103:5060;branch=z9hG4bKmp17a
Max-Forwards: 70
To: Heisenberg <sip:werner.heisenberg@munich.de>
From: E. Schroedinger <sip:schroed5244@aol.com>;tag=42
Call-ID: 10@100.101.102.103
CSeq: 1 INVITE
Contact: <sip:schroed5244@pc33.aol.com>
Content-Type: application/sdp
Content-Length: 159
```

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP 100.101.102.103:5060;branch=z9hG4bKmp17a
To: Heisenberg <sip:werner.heisenberg@munich.de>;tag=314159
From: E. Schroedinger <sip:schroed5244@aol.com>;tag=42
Call-ID: 10@100.101.102.103
CSeq: 1 INVITE
Contact: <sip:werner.heisenberg@200.201.202.203>
Content-Type: application/sdp
Content-Length: 159
```

Dialog的建立与删除



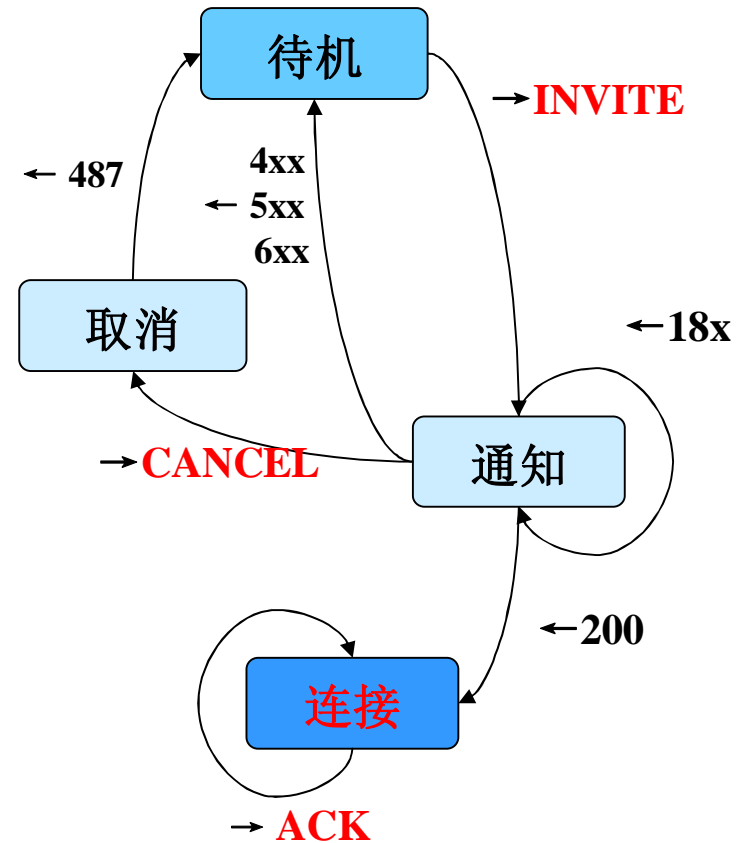
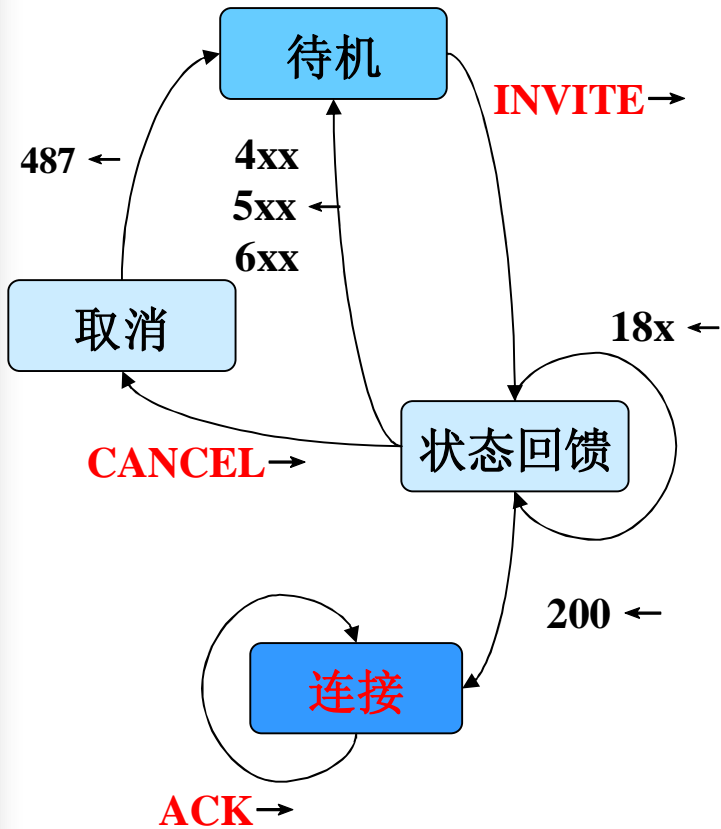
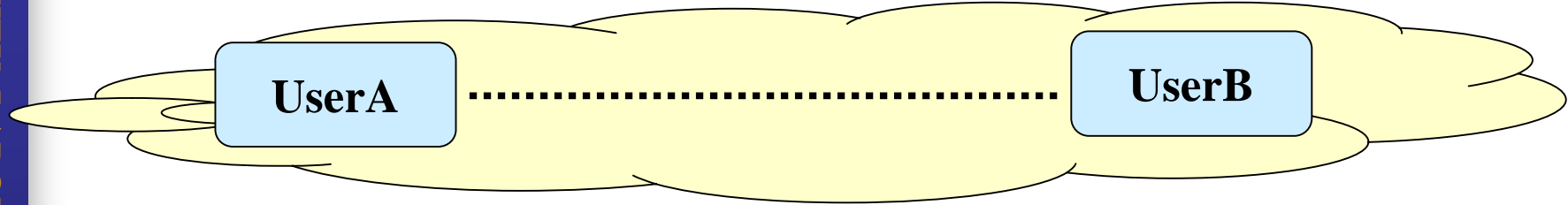
Dialog

Create Dialog

Dialog

Terminate Dialog

Dialog与会话





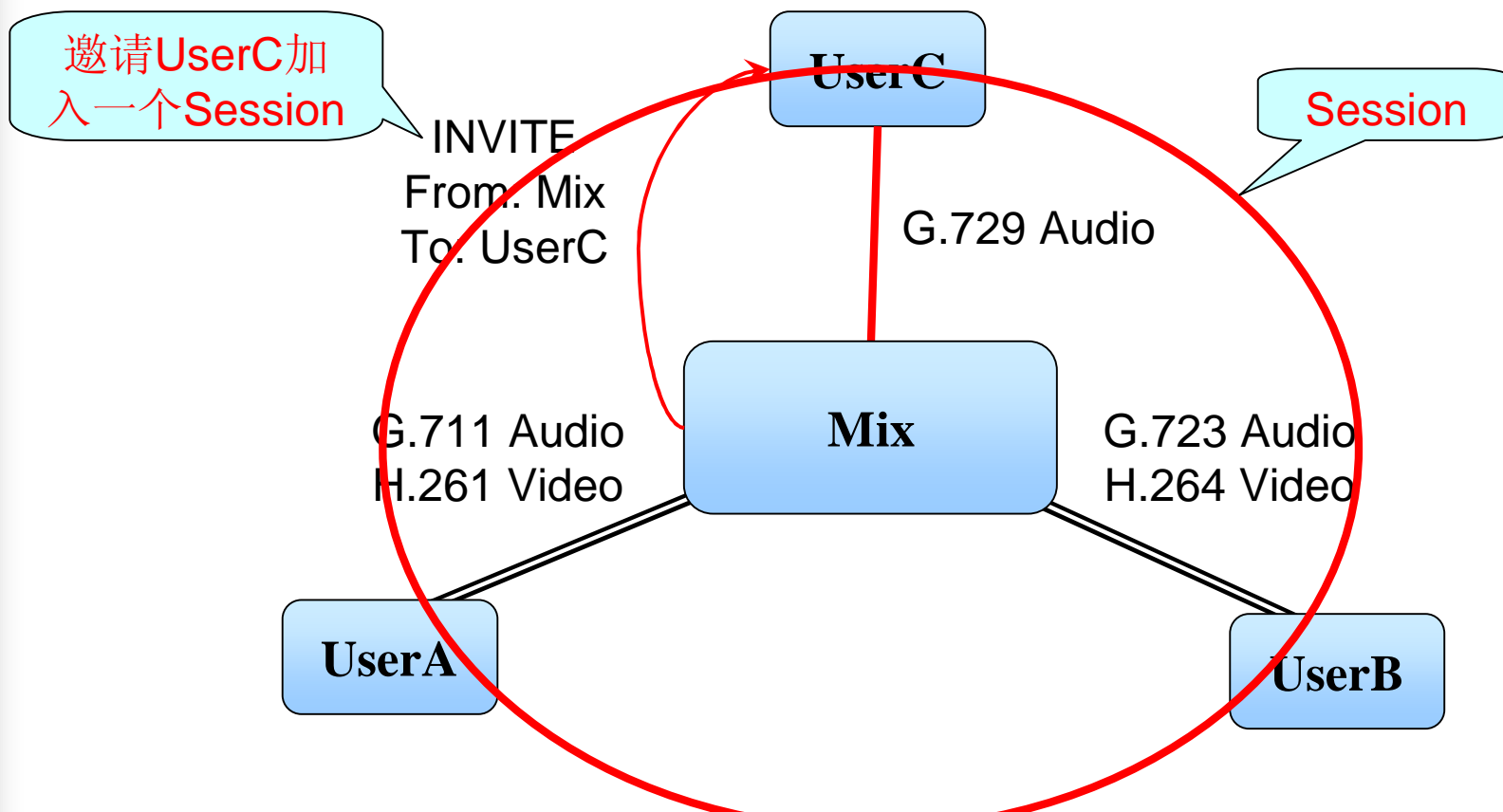
Session

- A multimedia session is **a set of** multimedia **senders** and **receivers** and the **data streams** flowing from senders to receivers (RFC 2327 SDP)
 - A multimedia conference is an example of a multimedia session
 - A session as defined for SDP can comprise one or more RTP sessions
 - As defined, a callee can be invited several times, by different calls, to the same session.
 - If SDP is used, a session is defined by the concatenation of the SDP user name, session id, network type, address type, and address elements in the origin field
- INVITE是在邀请什么？
 - 加入一个**Session**



Session的直观理解

- Mix向UserC发起会话请求，加入一个会议

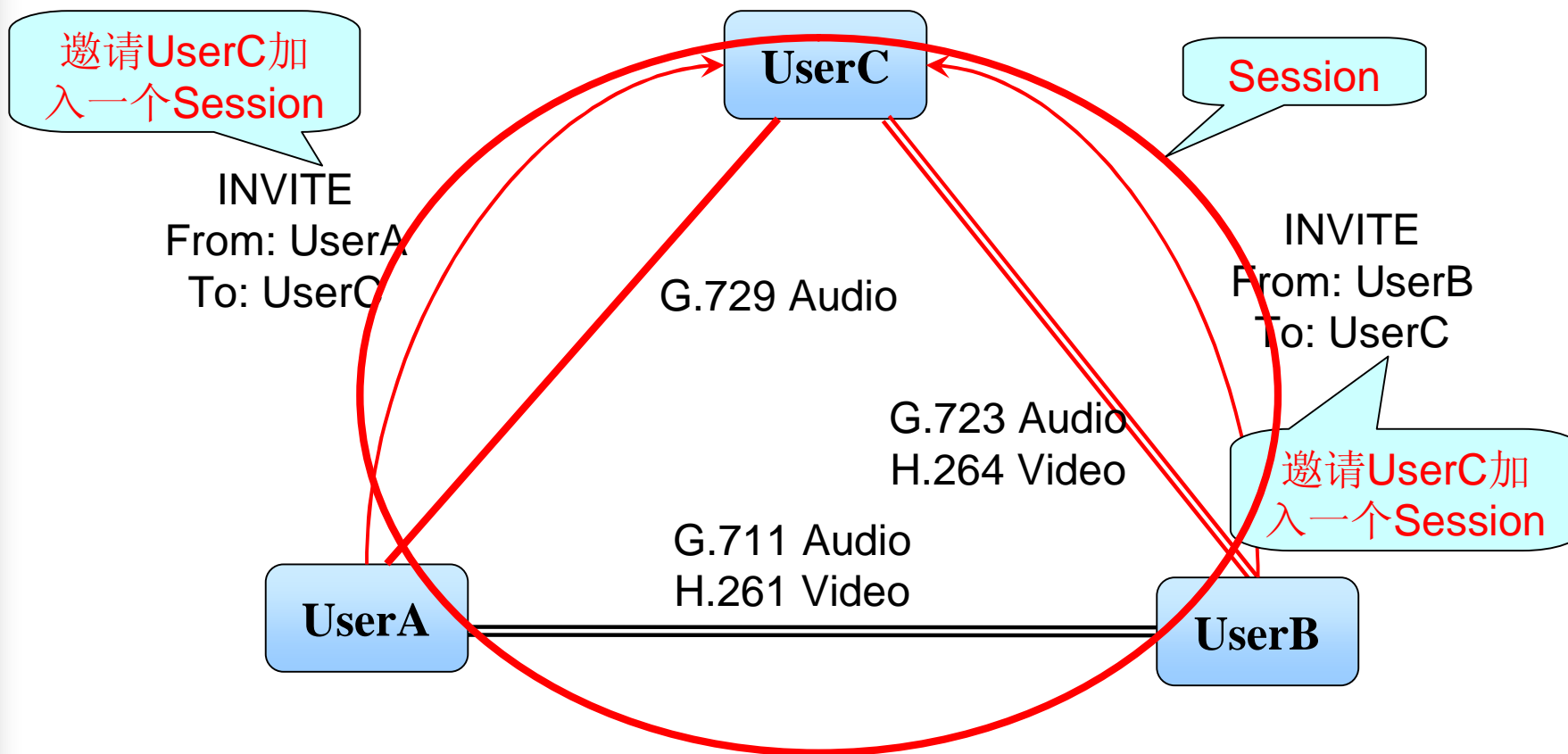


进一步内容在媒体协商与建立一节中说明



Session的直观理解

- UserA、UserB向UserC发起会话请求，建立一个多方会话



进一步内容在媒体协商与建立一节中说明



协议基本架构总结

- SIP协议是分层的协议
- 应用层保证会话可靠建立机制：三次握手
- 协议的核心：Transaction
- 协议的用户层概念：Dialog
- SIP邀请建立的Session概念



思考题

- SIP协议分层
- 使用三次握手的原因
- 如何理解Transaction
- 如何理解Dialog
- 如何理解Session