COMPOSITIONAL CONTROL OF IP MEDIA

Pamela Zave and Eric Cheung

AT&T Laboratories—Research

Florham Park, New Jersey, USA

DYNAMIC, POINT-TO-POINT MEDIA SERVICES OVER IP

telephony

home networks

computer-supported cooperative WORK
- teleconferencing
- telemonitoring
- distance learning
- virtual reality

computer-supported cooperative PLAY
- collaborative television
- multiplayer games
- networked music performance

THESE SERVICES USE A DIVERSE SET OF MEDIA ENDPOINTS

CAN ALL THE SERVICES BE IMPLEMENTED IN THESE ENDPOINTS, AS IS OFTEN ASSUMED?

often disconnected, has no persistent network presence

does not belong to a trusted administrative domain

microphone

camera

PDA

the owner will not download new software to contact a business once

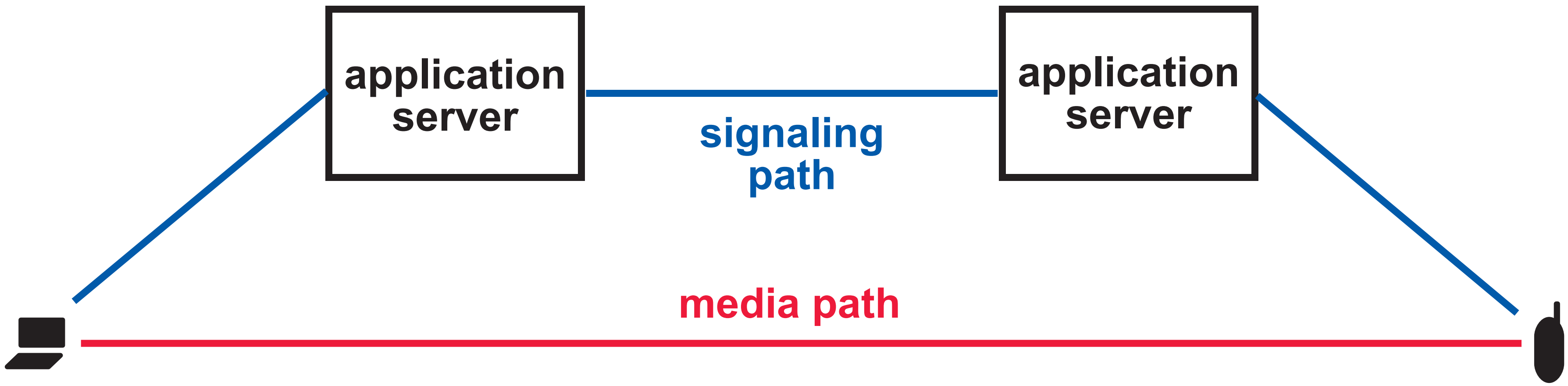
mixer

a vendor's "black box"

record/ play server

not possible to download new software to this device

IN PRACTICE, THESE SERVICES HAVE SIGNALING/MEDIA SEPARATION



SIGNALING PATH

often passes through several servers

low bandwidth + requires reliability
= often uses TCP

MEDIA PATH

should be the shortest end-to-end path

high bandwidth + tolerates packet loss
= often uses RTP

OFTEN, THE APPLICATION SERVERS DO NOT KNOW ABOUT EACH OTHER

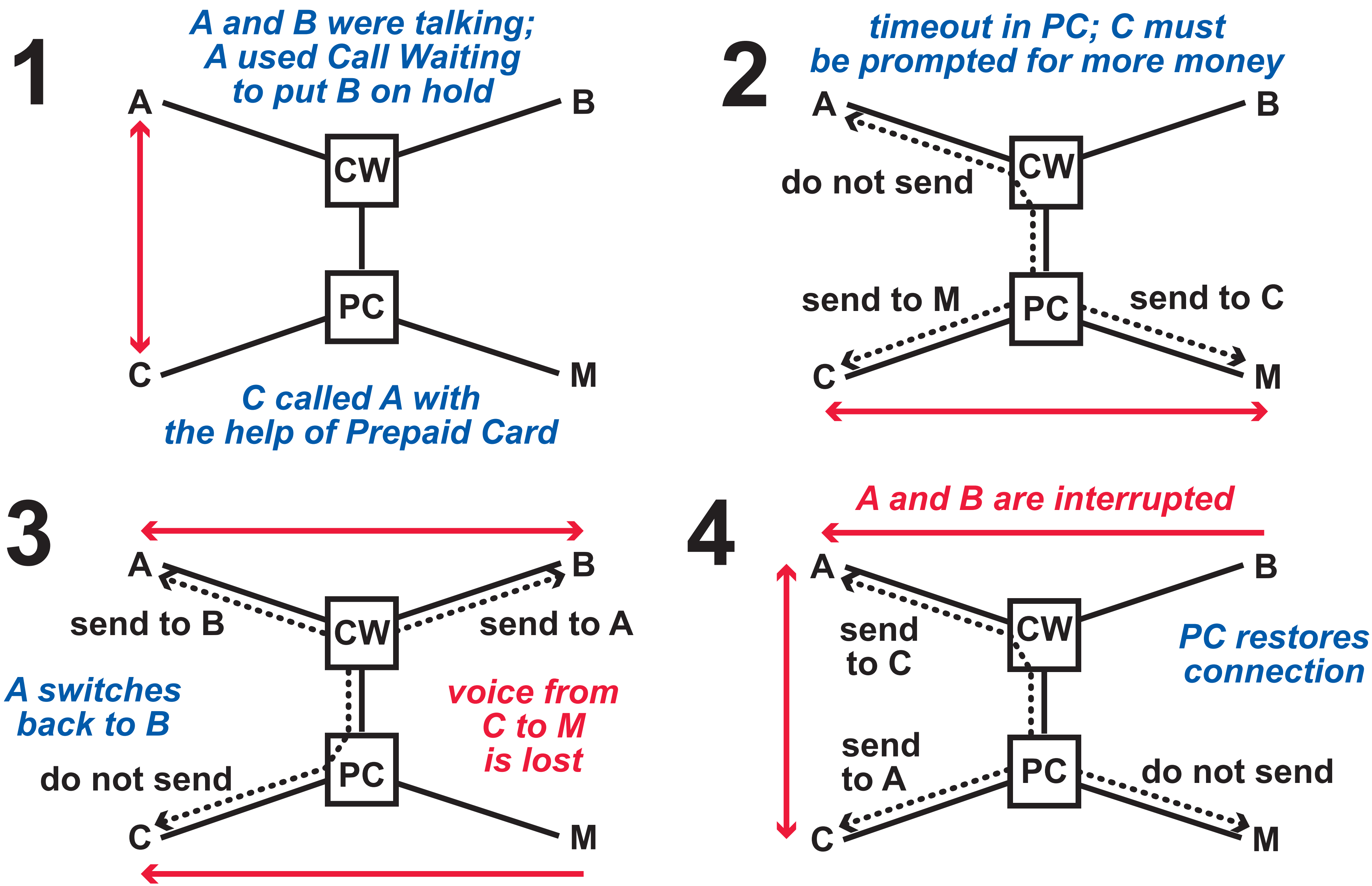
belong to different administrative domains

serve different users

are produced by different vendors

are added and/or updated individually

THE NEED FOR COMPOSITIONAL MEDIA CONTROL



these problems occur because the actions of the two servers are not coordinated

COMPOSITIONAL MEDIA CONTROL

PROBLEM

find a way to program media control in application servers that . . .

- is sufficient for all applications
- is architecture-independent

no location or function constraints on physical components

- is automated

because programming media control is inherently difficult

- works regardless of how many application servers are controlling media channels concurrently
- is verified

SOLUTION

SPECIFICATION

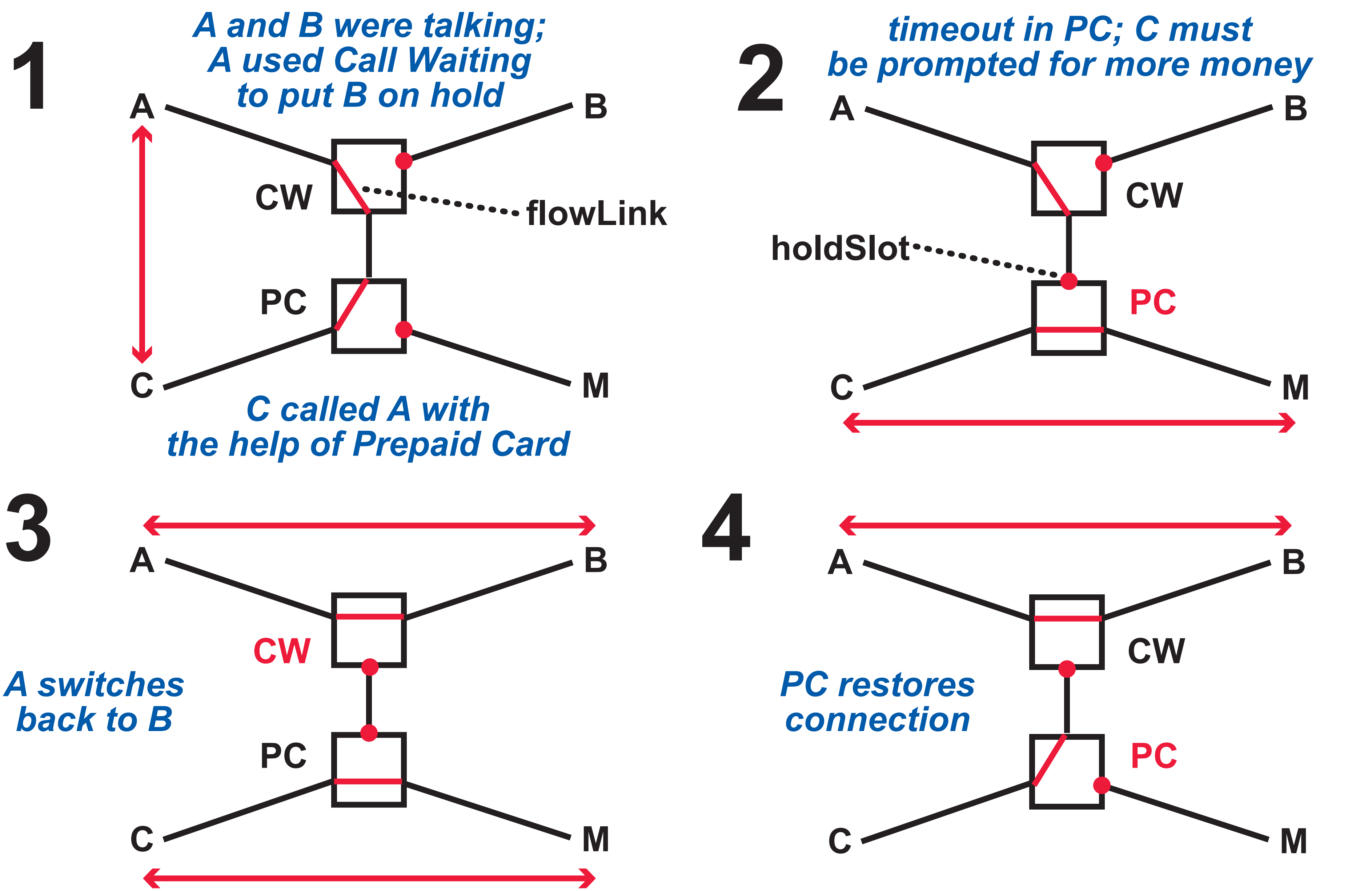
- architecture-independent descriptive model of systems
- a set of high-level primitives for application programmers
- compositional semantics of the primitives in temporal logic

IMPLEMENTATION

- a new signaling protocol
- implementation of programming primitives
- partial verification by model-checking

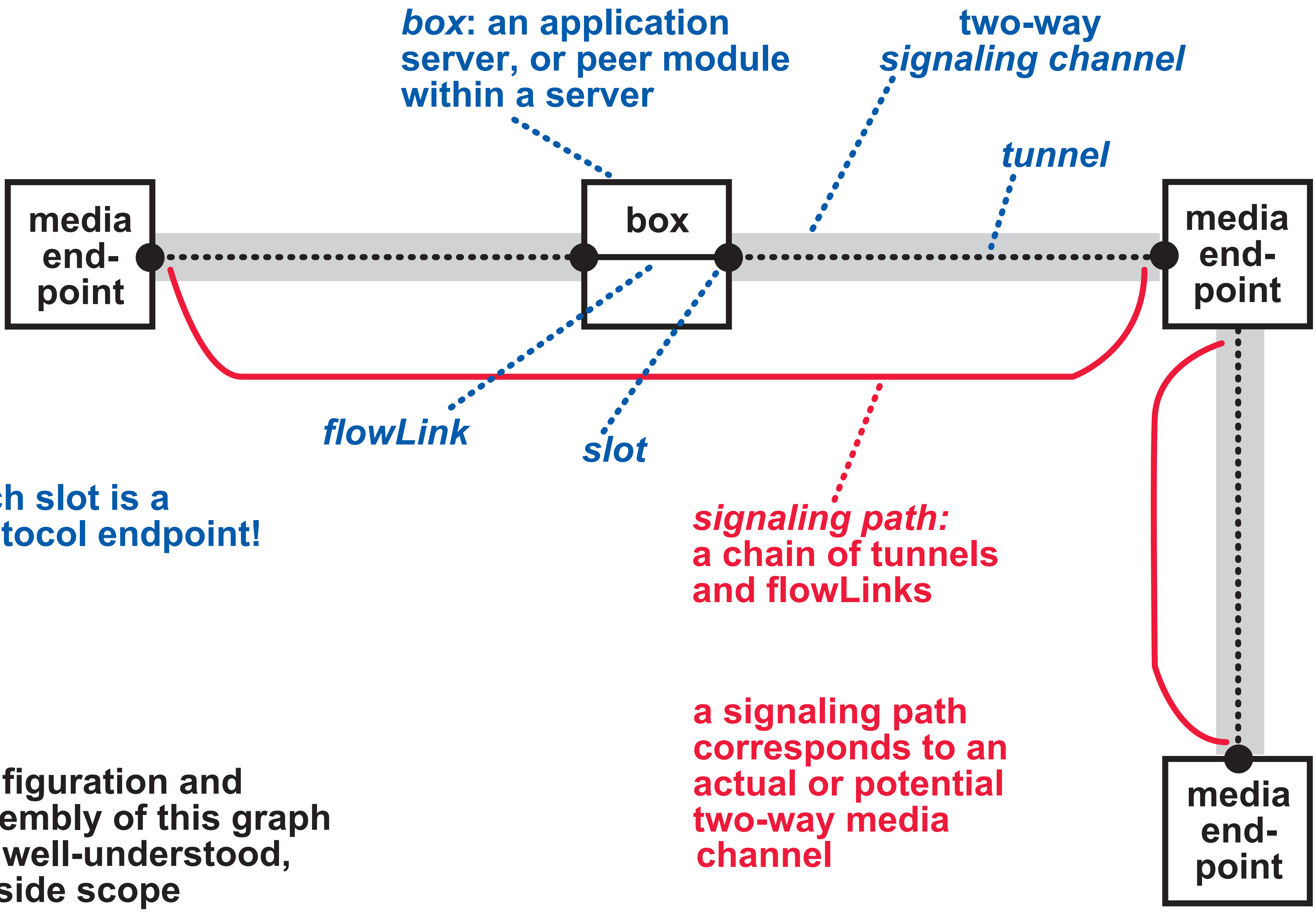
COMPOSITIONAL MEDIA CONTROL IN SIP

HOW MEDIA CONTROL SHOULD WORK (OVERVIEW)



there is a voice channel between two endpoints if and only if there is an unbroken chain of signaling channels and flowLinks between them

ARCHITECTURE-INDEPENDENT DESCRIPTIVE MODEL

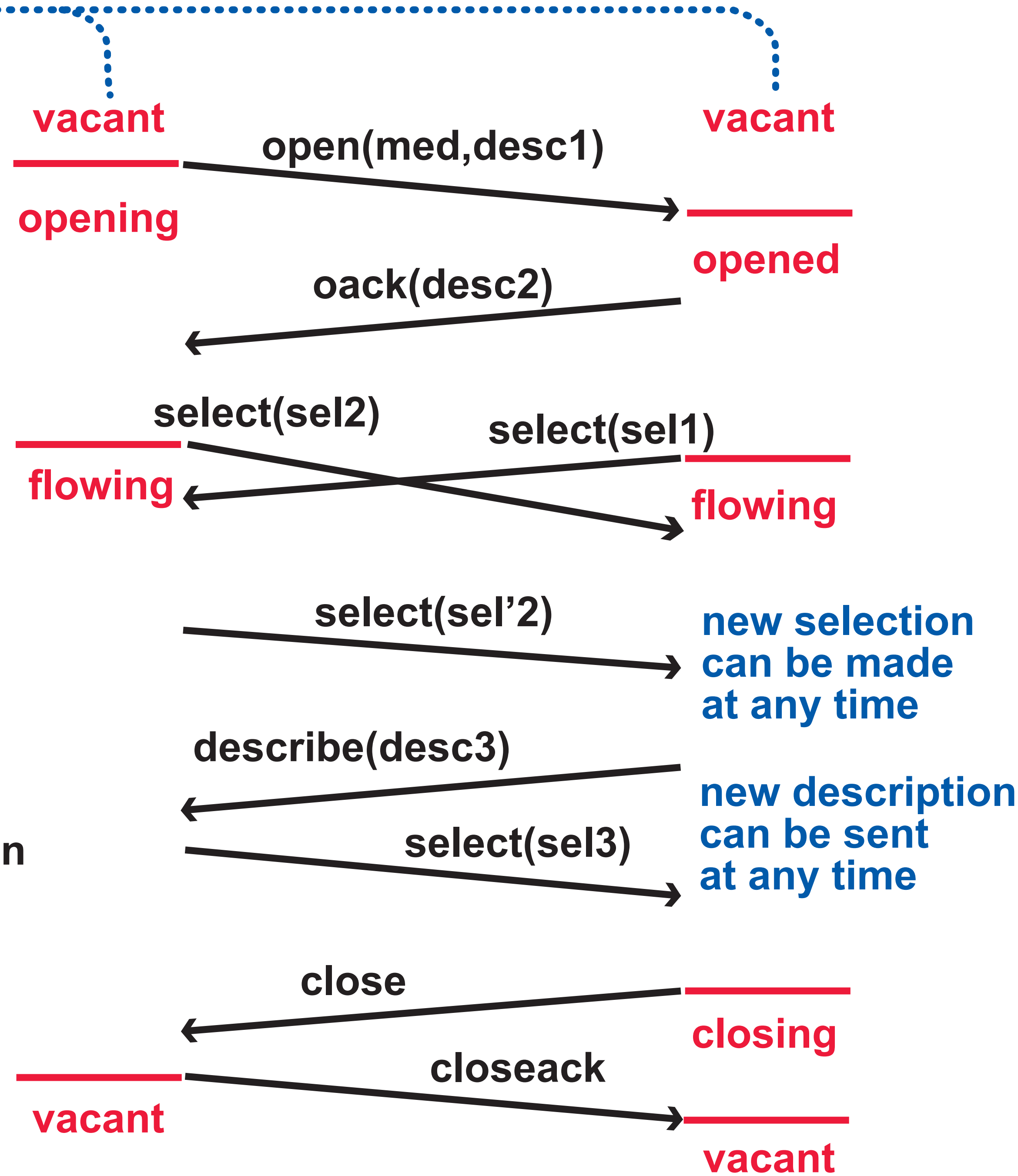


each slot is a protocol endpoint!

configuration and assembly of this graph are well-understood, outside scope

MEDIA-CONTROL PROTOCOL

SLOT STATES



a *descriptor* describes an endpoint as a receiver:

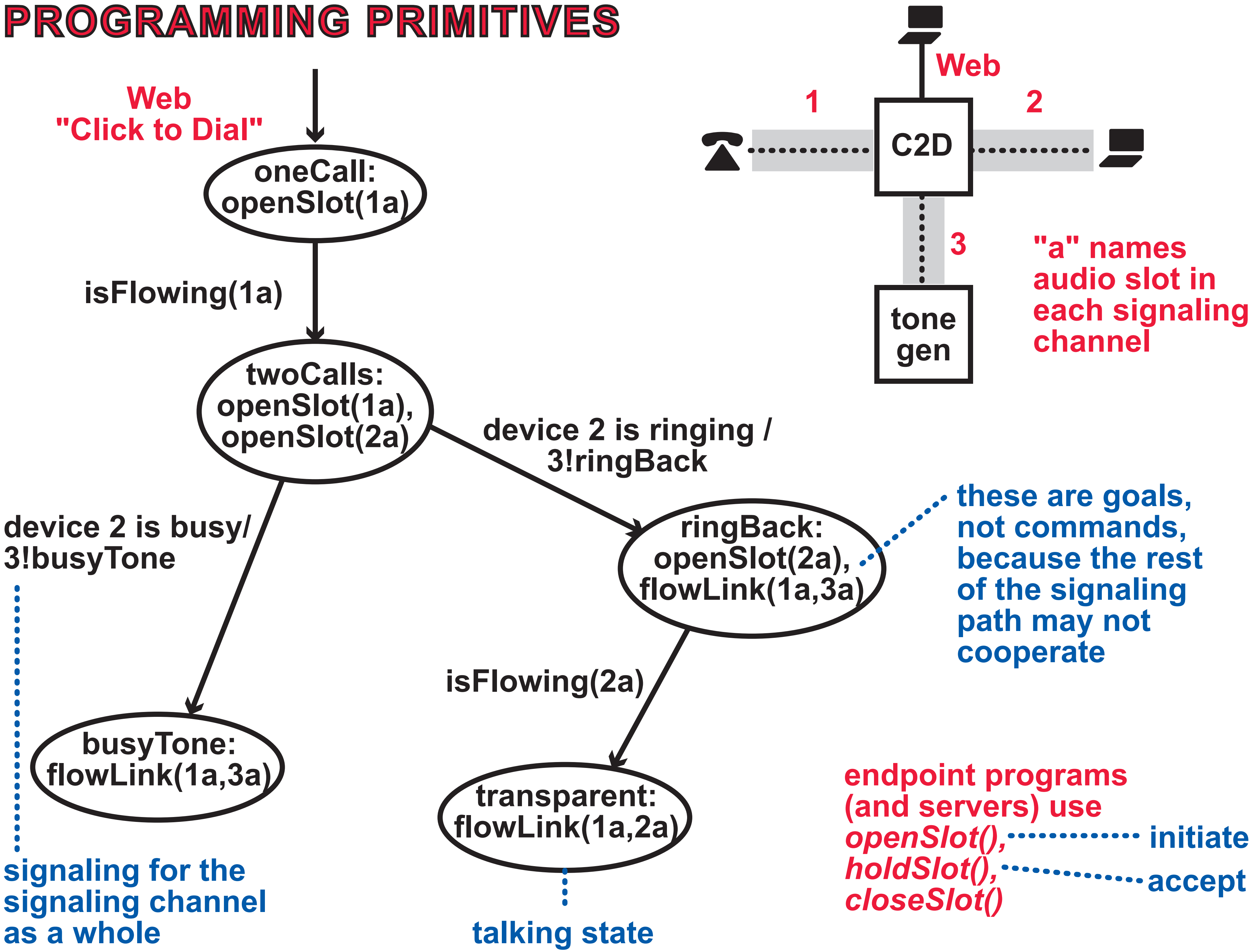
- IP address
- port
- priority-ordered list of codecs

a *selector* responds to a descriptor, chooses sending codec

this protocol can provide optimal codec choice and signaling/media synchronization

description/selection of media flow in each direction is independent and concurrent, making composition easier

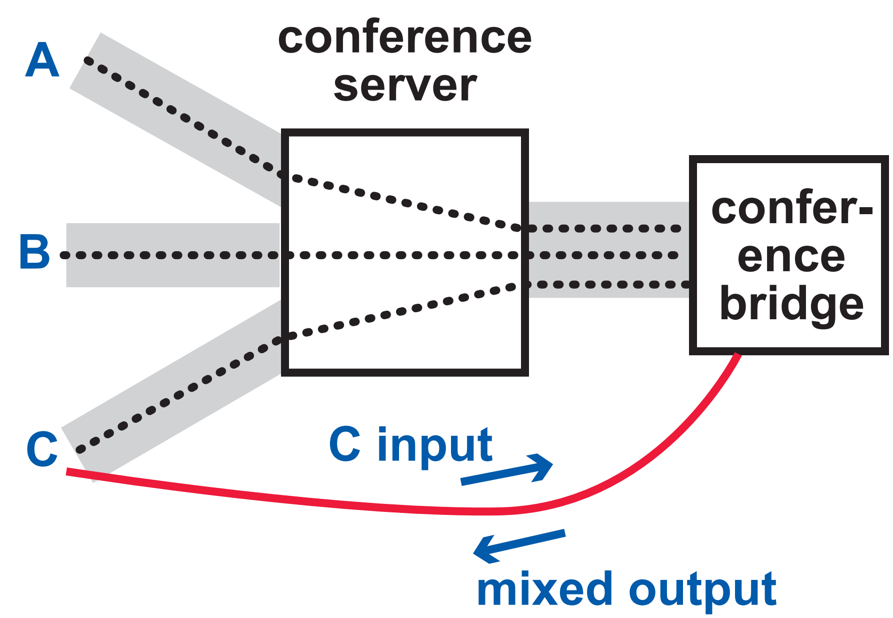
PROGRAMMING PRIMITIVES

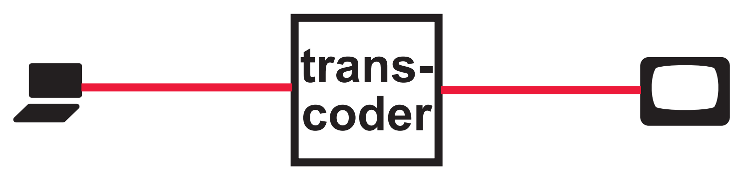


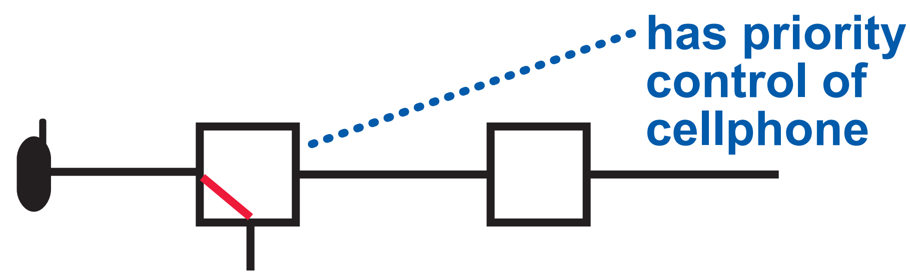
GENERALITY

THE DESCRIPTIVE MODEL, PROTOCOL, AND PROGRAMMING PRIMITIVES ARE SUFFICIENT FOR ALL MULTIMEDIA SERVICES

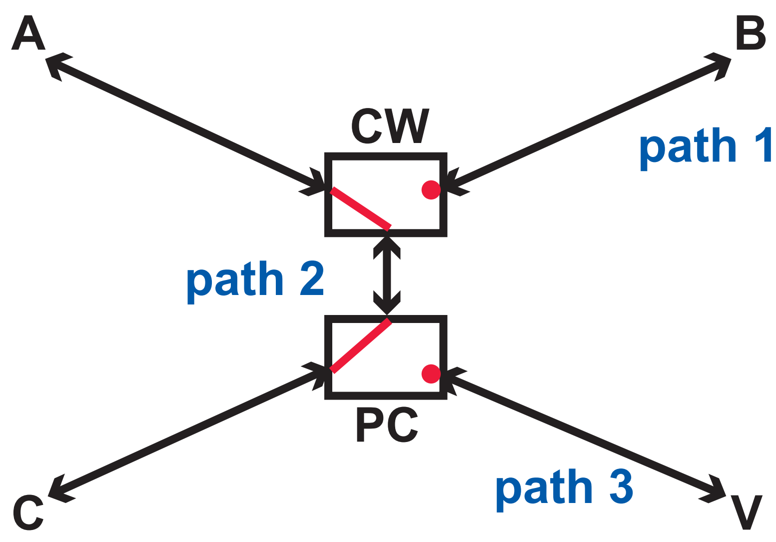
- a signaling channel can have any number of tunnels
- a signaling path can establish a channel of any medium
- temporary muting of channels is included (in either or both directions)
- multiparty connections always require the use of resources for mixing or replication
- a transcoded media channel looks like two in this model
- feature priority is determined by proximity in the signaling graph







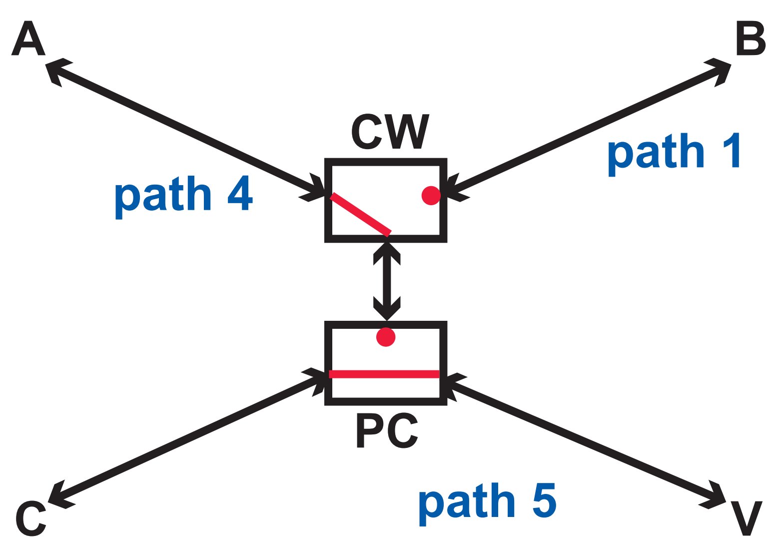
FROM



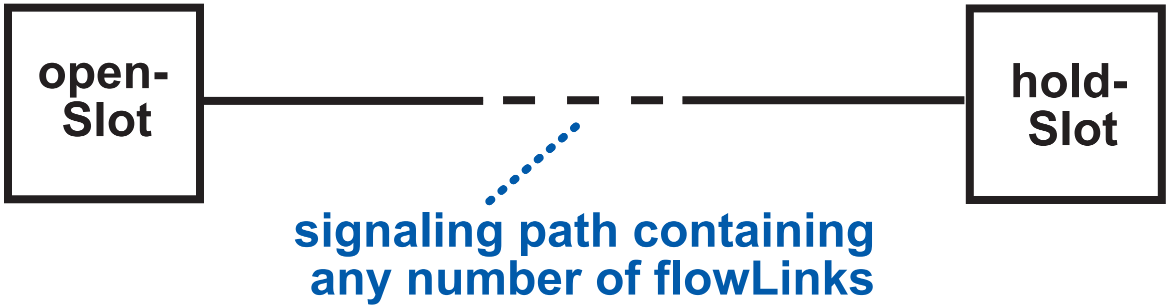
IMPLEMENTATION CHALLENGE:

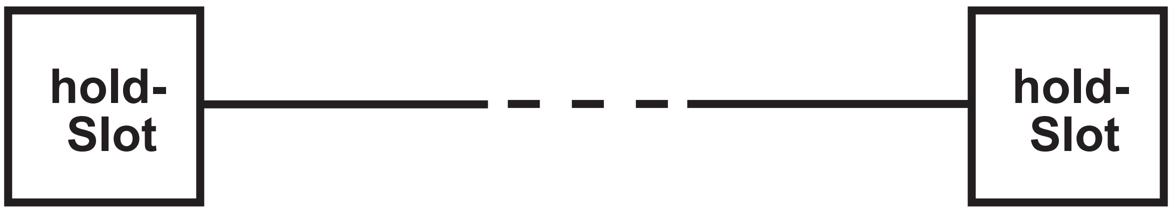
When the last flowLink of a new path "clicks into place", its slots can be in any states whatsoever.

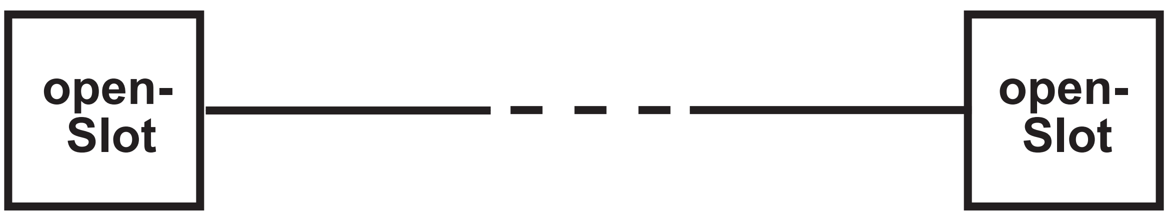
TO



PATH CONFIGURATIONS







for paths with zero or one flowLink, protocol and implementation proved correct by model-checking

cannot model-check larger configurations, but checkable configurations might yield lemmas for an inductive proof

PATH SPECIFICATIONS

stability property:

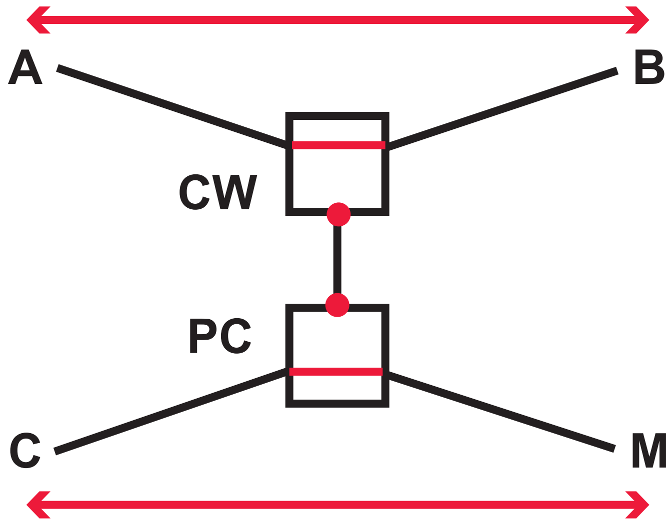
$$\Diamond\Box\,(\mathit{Lflow} \wedge \mathit{Rflow} \wedge \mathit{LselRcvd} = \mathit{noMedia} \wedge \mathit{RselRcvd} = \mathit{noMedia})$$

$$\Diamond\Box\,(\mathit{Lvac} \wedge \mathit{Rvac}) \vee (\mathit{Lflow} \wedge \mathit{Rflow} \wedge \mathit{LselRcvd} = \mathit{noMedia} \wedge \mathit{RselRcvd} = \mathit{noMedia})$$

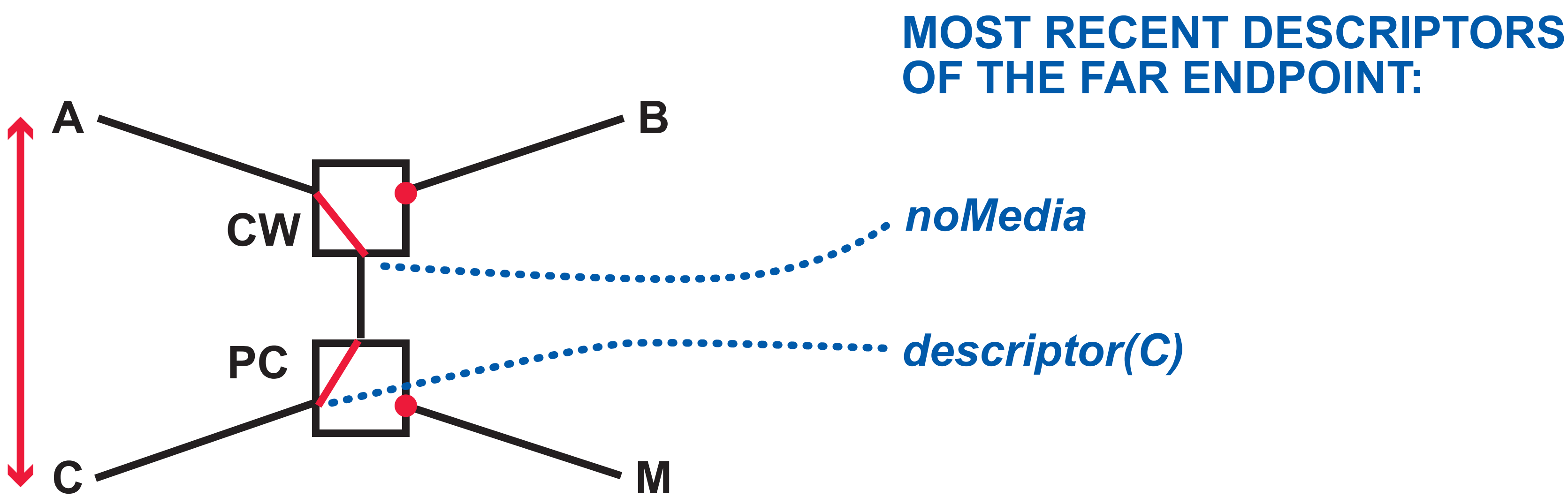
recurrence property:

$$\Box\Diamond\,(\mathit{Lflow} \wedge \mathit{Rflow} \wedge \mathit{LdescRcvd} = \mathit{RdescSent} \wedge \mathit{RdescRcvd} = \mathit{LdescSent} \wedge \mathit{LselRcvd} = \mathit{LdescSent} \wedge \mathit{RselRcvd} = \mathit{RdescSent})$$

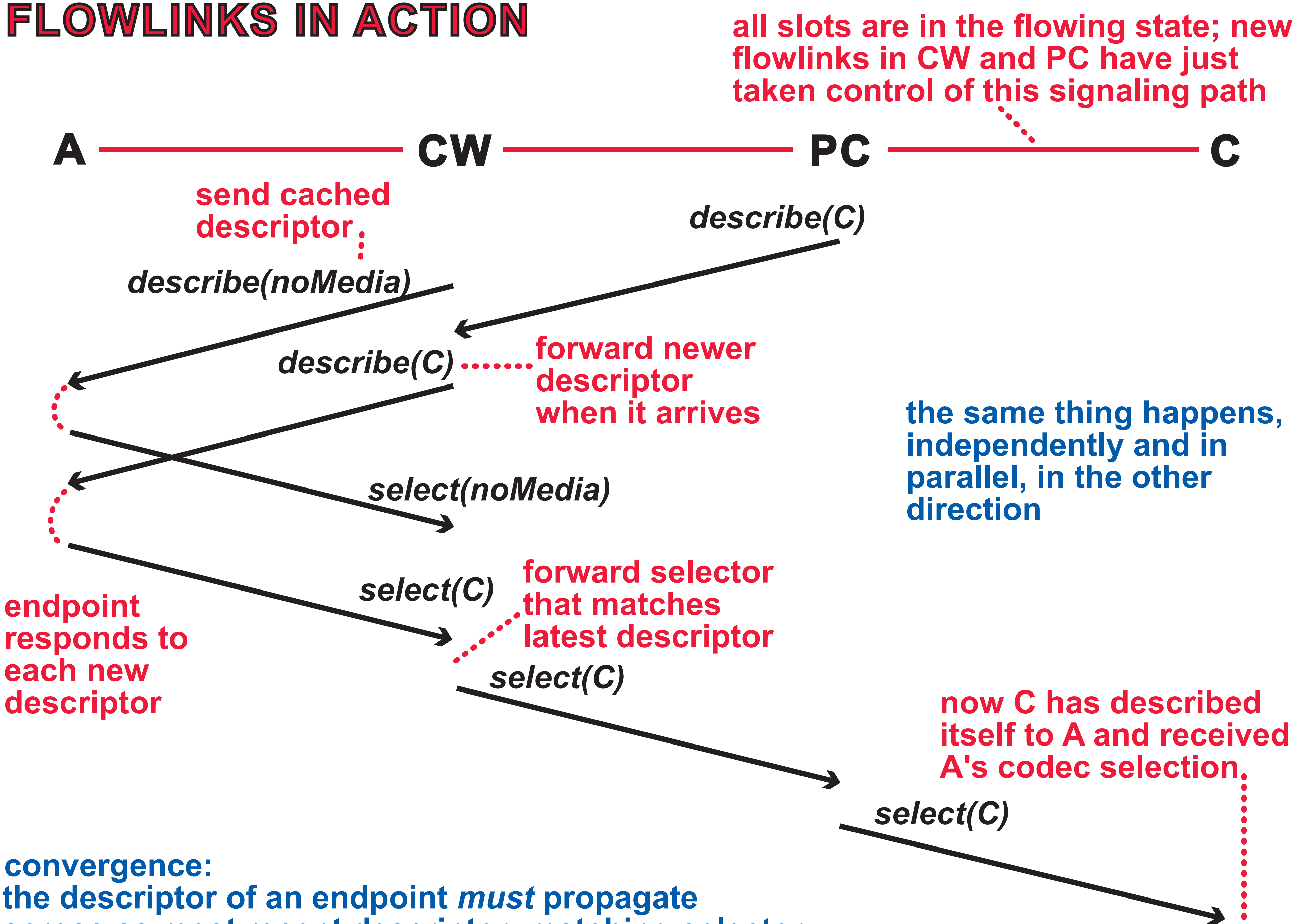
FROM


A
B
CW
PC
C
M

TO


MOST RECENT DESCRIPTORS
OF THE FAR ENDPOINT:
A
B
CW
noMedia
PC
descriptor(C)
C
M

FLOWLINKS IN ACTION



convergence:
the descriptor of an endpoint *must* propagate across as most recent descriptor; matching selector *must* propagate across as response to descriptor

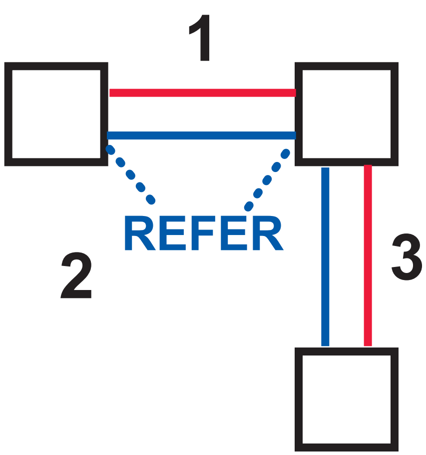
SIP IS THE DOMINANT SIGNALING PROTOCOL FOR IP MEDIA SERVICES

SIP was designed with a strong end-to-end philosophy .... *no application servers, all services in media endpoints*

"MULTIPARTY CALL CONTROL" (MPC)

This style of SIP uses the REFER method, Replace and Join headers.

It works by moving signaling paths as well as media paths.



It is hostile to application servers: any servers in the original signaling path are removed from signaling path.

"THIRD-PARTY CALL CONTROL" (3PCC)

This style of SIP uses re-INVITEs and back-to-back user agents.

We have developed an implementation of our specification in SIP with 3PCC.

It is verified to the same extent that the "new protocol" version is.

SOME ANALYTIC PERFORMANCE COMPARISONS

SCENARIO: The example in which CW and PC switch at about the same time.

ASSUMPTIONS:

all components distributed

average server response time 20 ms

average message latency 34 ms

AVERAGE TOTAL LATENCY

new protocol	128 ms
SIP/3PCC	378 ms
	3560 ms

with or without
a race condition

The difference is caused by radical differences in the protocol designs.

SCENARIO: There are two interacting services, one using switching and one using conferencing. In this scenario, both services are used.

TOTAL NUMBER OF INTER-NODE MESSAGES

SIP/3PCC, both services in separate application servers	48
SIP/MPC, both services in one of the media endpoints	81

This comparison is significant because proponents of the MPC style claim that it has simpler and faster signaling than the 3PCC style.

FUTURE WORK

StratoSIP

- StratoSIP is a high-level, domain-specific language for programming IP media services

- technology base is SIP and SIP Servlets (a standardized architecture for SIP application servers)

- incorporates compositional media control

- first version due end of 2008

SIP

- we have many ideas for improving or replacing SIP

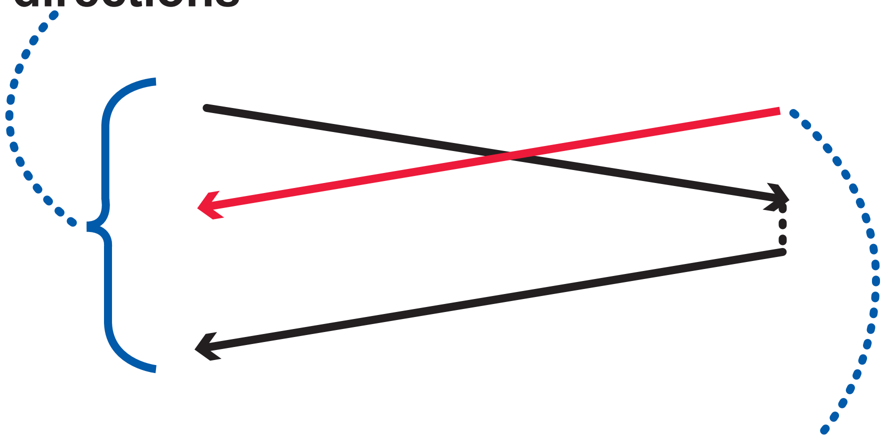
- however, such changes are unlikely

PROTOCOL COMPARISON 1

SESSION INITIATION PROTOCOL (SIP)

"transactional"

a transaction needs exclusive use of the signaling channel in both directions



if there is a race between transactions in opposite directions, *both lose*

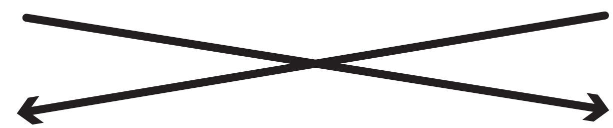
note that this cannot happen in a client/server application, which is what transactions are good for

OUR COMPOSITIONAL PROTOCOL

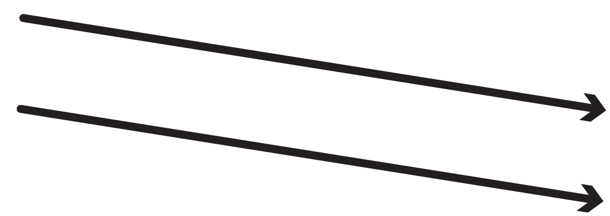
"idempotent"

a signal provides updated information about *one* end of the channel

similar signals from opposite directions are independent



an endpoint can send updated information at any time



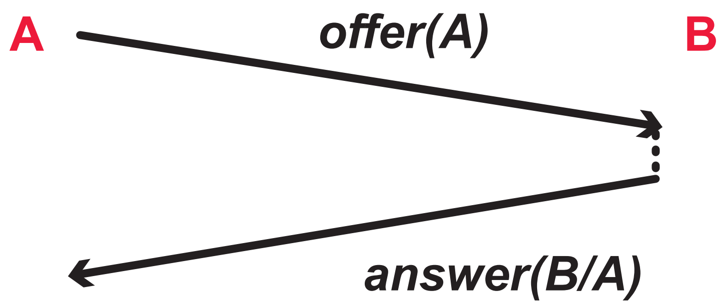
maximized *program state*
program complexity
latency **minimized**

PROTOCOL COMPARISON 2

SESSION INITIATION PROTOCOL (SIP)

codec choice by negotiation

codecs in offer describe the capabilities of endpoint A



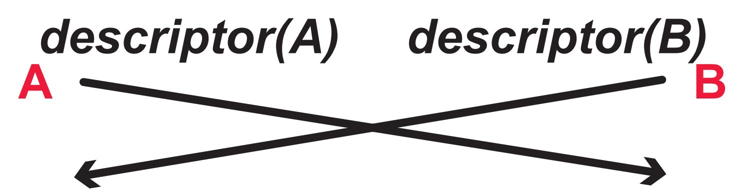
codecs in answer describe the capabilities of B in relation to A

- endpoint descriptions cannot be cached for re-use, new ones must be solicited
- description is sequential

OUR COMPOSITIONAL PROTOCOL

codec choice by unilateral description

codecs in descriptor unilaterally describe the capabilities of an endpoint



- endpoint descriptions can be cached for re-use
- description of the two endpoints can proceed in parallel

maximized

program state
program complexity
latency

minimized