Announcements

- NO LECTURE next Tuesday (4.12.)
- NO LECTURE on Thursday (13.12.)
- Assignment 3: Next Wednesday (5.12.)
- Read
  - A Framework for Application Interaction in SIP
    - draft-ietf-sipping-app-interaction-framework-05.txt
  - A SIP Event Package for Key Press Stimulus (KPML)
    - RFC 4730
  - SIP for telephony slides
- Updated schedule

  11.12. Jabber, SIP for telephony, and conferencing

  12.12. SIP in the real-world, exam hints

HELSINKI UNIVERSITY OF TECHNOLOGY
NETWORKING LABORATORY

SIP Extensions and Service Creation

© 2007 Jörg Ott

Basic Extension Mechanisms

- Proxies forward unknown methods and headers
- UAS' ignore unknown headers, reject methods

- Feature negotiation
  - Headers: Require, Proxy-Require, Supported
  - Option tags for feature naming (see below)
  - Error responses:
    - 405 Method not allowed
    - 420 Unsupported
    - 421 Extension Required

UAC and UAS/proxy must agree on common feature subset

- Option tags
  - Identified by unique token
  - Prefix reverse domain name of creator
  - IANA: implicit prefix *org.ietf.*

Some SIP Extensions

- Reliable provisional responses
- Session Timers
- Early Media
- Adjusting session state: UPDATE
- INFO method
- REFERing peers to third parties
- SIP for subscriptions and event notifications
- Instant messaging
- SIP for presence
- ...

- Addressed later today where appropriate...

Reliable Provisional Responses

- Signaling Gateways, ACD systems etc. may depend on provisional responses
  - Such as 180 Ringing

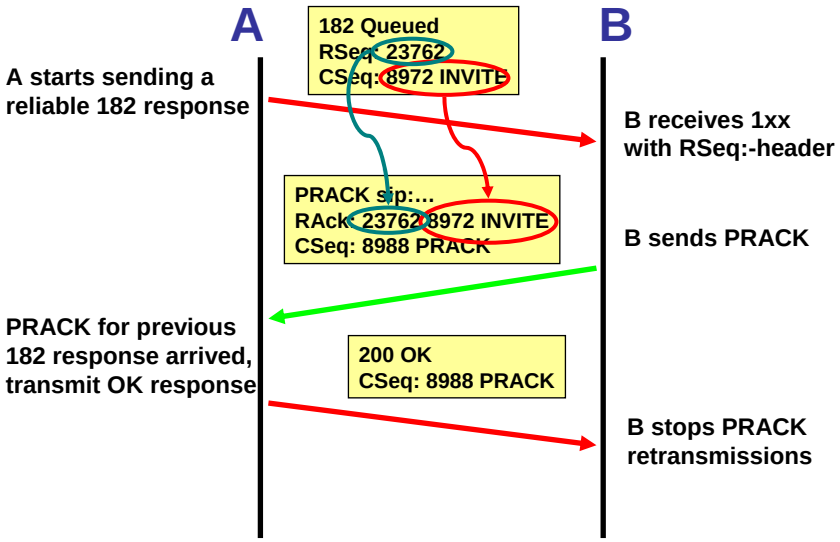
→ need optional reliability

- Option tag 100rel:
  - End-to-end reliability of provisional responses > 100
  - Retain order of reliable responses
- Implementation:
  - Windowing: RSeq/RAck headers
  - Explicit acknowledge: PRACK request

Use of PRACK

- Insert RSeq:-header with random integer value
  - new responses must increment RSeq: by 1
- Retransmit with exponential backoff
- No subsequent reliable provisional responses until first PRACK
  - enables re-ordering at receiver side
- PRACK request
  - RAck: contains RSeq: and CSeq: values to acknowledge
  - May contain body with additional information

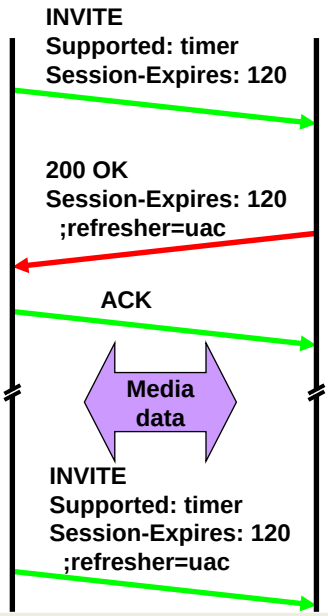
Example Call Flow: PRACK



Session Timers

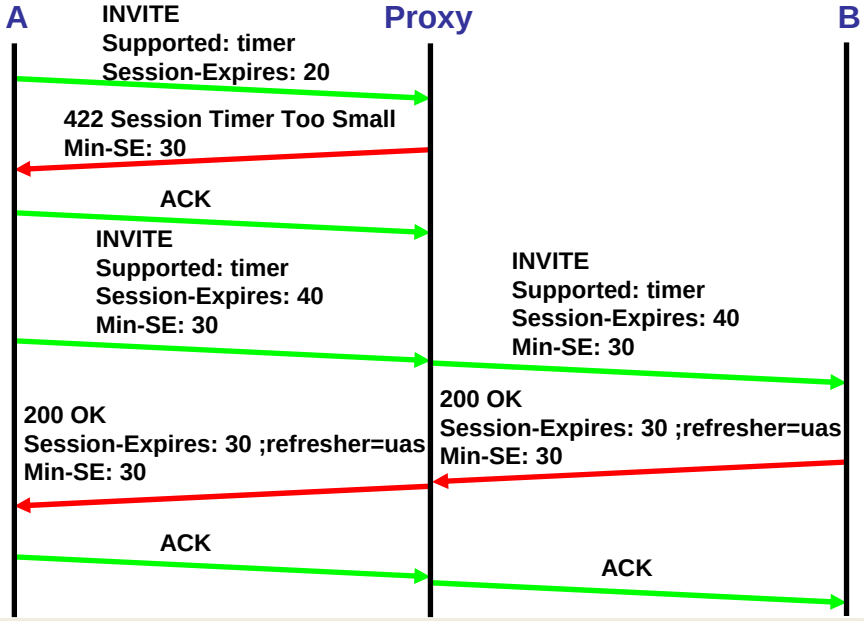
- No keep-alive mechanism for SIP calls
  - Proxies may preserve outdated state information
  - Close dynamically created holes in firewalls
  - Need timeout-mechanism to perform cleanup
- SIP Extension for Session timers
  - Option tag timer
  - Negotiation mechanism for expiry time and refresh responsibility
    - Session-Expires: duration and role
    - 422 response with Min-SE: lower bound for expiry interval
  - *Responsible UA* sends re-INVITE before timer expires
- Session is terminated if no re-INVITE/UPDATE arrives in time
  - *Responsible UA* sends BYE
  - Proxies silently drop state information

Example: Keep-alive



- Caller indicates support of session timers in INVITE
- Propose initial timer value
- Called UA supports timers, accepts proposed value, asks the caller to send refresh
- Setup complete, media streams are established
- After n/2•timer calling party sends re-INVITE
- Call continues after 200 OK and ACK

Example: Interval Negotiation

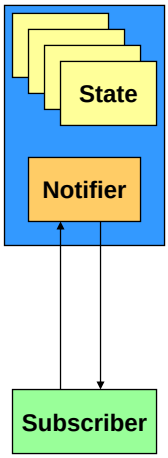


Event Notifications

- Need for flexible event notification
  - Enable presence information
  - Better support for mobile SIP applications
  - Feedback about progress of other calls, conference state, etc.
  - NOTIFY method, Event:-header
- Event subscription
  - SUBSCRIBE, Event:
  - Events may be call-related or third-party generated
  - Security issues: Sensitive Events
    - Privacy
    - Authentication
- Used for personal presence applications
  - Used with PUBLISH method to update presence state
  - Augmented by MESSAGE method for Instant Messaging

Event Concept

- Piece of state information S
  - Identified by some name (“package”)
- SIP entities interested in S
  - Query for the current state
    - “polling”
  - Be notified about changes to S
    - SUBSCRIBE
  - Subscriptions may be created implicitly
    - By means of other (SIP or non-SIP) protocol activity
- Information about S carried in message body
  - NOTIFY
  - Formats to be defined specific to S
- Protection of S
  - Keep control of who gains access, who has access; for how long



SUBSCRIBE

- SUBSCRIBE used to registered interest in a piece of state
  - Event: identifies the state information to be retrieved
    - 489 Bad Event
    - Syntax reflects package concept:
      package ('.' *template*)* → e.g. "Event: presence.winfo"
  - Allow-Events: used to indicate which event packages are supported
  - Expires: how long to subscribe
    - Subscriptions are soft-state; need to be refreshed periodically
    - May be shortened or lenthened by server
  - Expires: 0
    - Poll the state information once; do not estabilish a subscription
    - Terminate a subscription
  - Body may indicate desired notification policy (filters)
- Each SUBSCRIBE triggers at least one NOTIFY
  - May contain no (useful) information if access not yet authorized

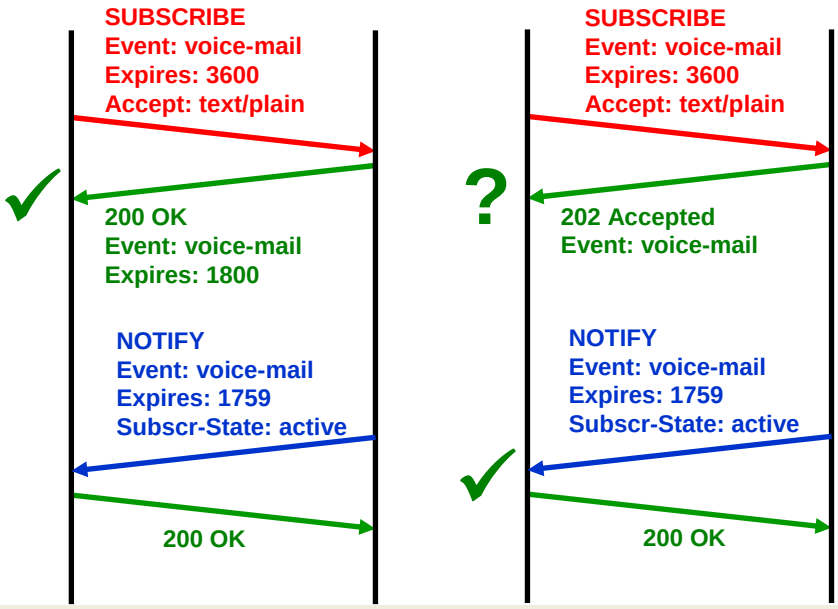
SUBSCRIBE Response

- SUBSCRIBE Responses
  - 200 OK: Everything's fine: event understood, client authorized, etc.
  - 202 Accepted : Request received, working on a final result
  - 401, 603, … if applicable
  - Acceptance or rejection to be reported in NOTIFY
- Accepting / rejecting subscription requests
  - Automated e.g. through configured black / white list
  - Ask user if acceptance cannot be determined automatically
- Watcher Info package used to monitor one's own presence
  - Generates notifications for subscription requests
  - User then needs to authorize subscription
    - Through arbitrary means (e.g. web page)

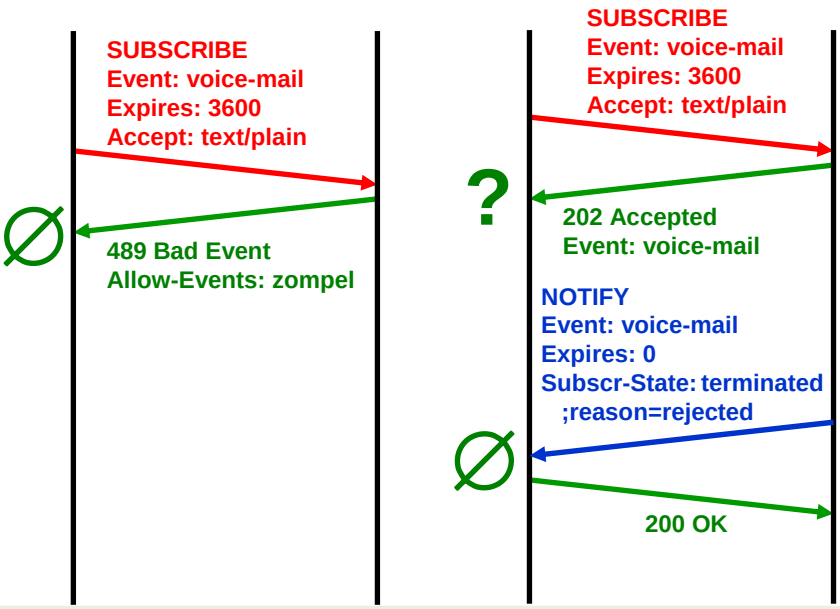
NOTIFY

- NOTIFY carries state information in message body
  - Format depending on Accept: header from SUBSCRIBE
  - Full state vs. state deltas
  - Subscription-State: active/pending/terminated
- NOTIFY indicates acceptance or rejection of SUBSCRIBE
  - If no immediate response could be supplied
- NOTIFY may be used to terminate subscriptions
  - Initiated by the notifier
  - Includes reason code parameter
- NOTIFY may be sent without SUBSCRIBE
  - Implicit subscription
- Response to NOTIFY
  - 200 OK vs. 481 Subscription does not exist
- Notification rate: risk of network congestion
  - Event throttling to be specified in event packages

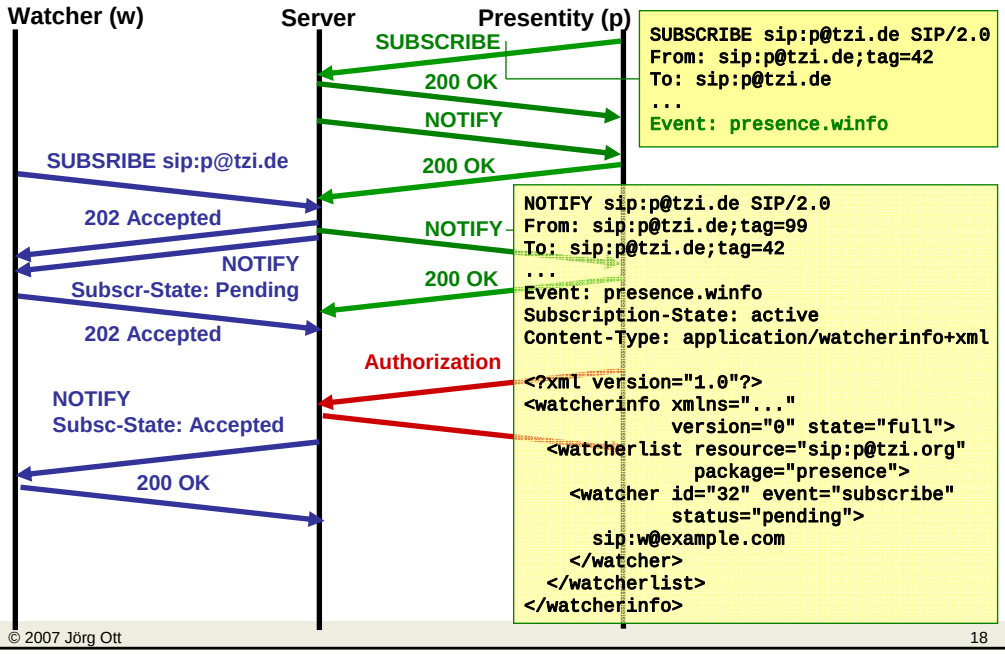
Example: Successful SUBSCRIBE



Example: Unsuccessful SUBSCRIBE



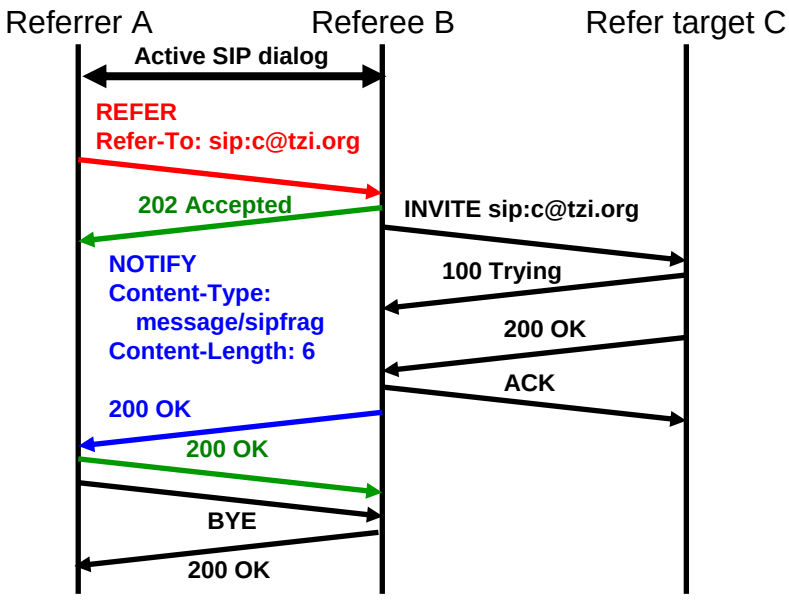
Sample Use of Watcher Information



REFER Method

- Original motivation: Call Transfer
- General idea: make a SIP entity contact a third party
- Originator sends REFER method to peer
  - indicates *refer target* in Refer-To: header
- Referred party accepts or declines immediately
  - 202 Accepted or uses some SIP error code
  - Accepting establishes an *implicit subscription* on the new dialog
- Contacts *refer target* in an entirely independent dialog
  - May indicate who caused the call in Referred-By: header
  - New INVITE – 200 OK – ACK sequence, unrelated
- Reports outcome of new call to originator
  - (exactly) one NOTIFY
  - Message body contains SIP message fragments from new call, e.g.
    - 200 OK 486 Busy Here 503 Service unavailable

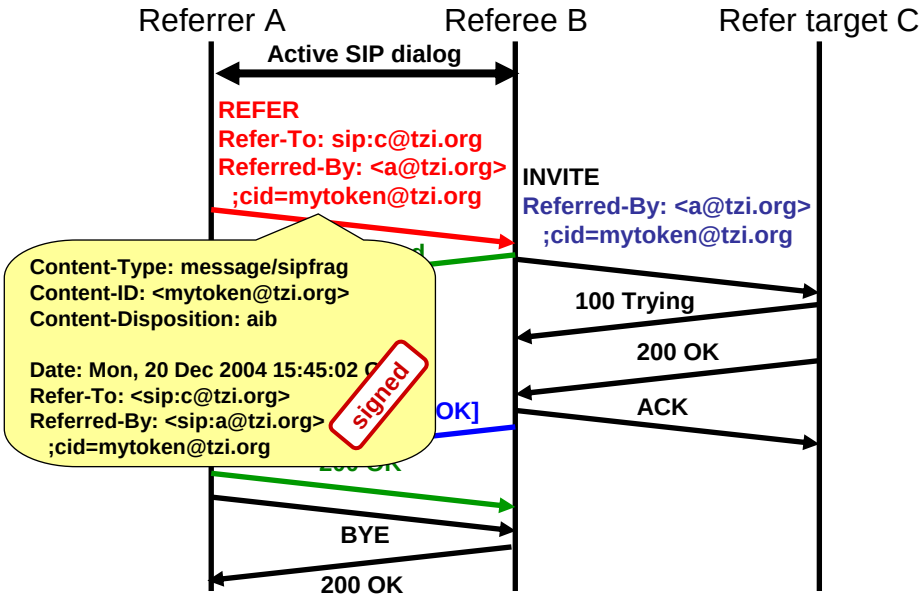
Simple REFER Example



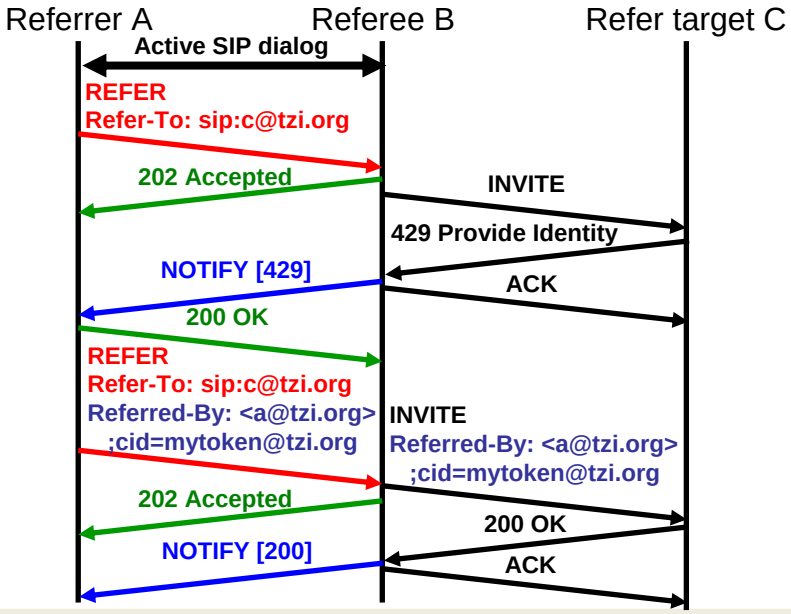
Securing REFER

- Refer Target C receives a request from Referee B
  - May be indistinguishable from any other request from B
  - May be acceptable to C only if A is known as initiator
  - Need to provide authentication information in the request
- Referred-By: header provides first indication
  - But may be faked by B
- Extension to securely pass information from A to C
  - Include signed S/MIME message body in REFER request
    - Content-ID: <sip-message-id>
  - To be copied by Referee into target request
  - Optional cid= parameter of Referred-By: points to signed body
    - Referred-By: <sip:referrer@example.com>;cid="<sip-message-id>"
- Refer target may refuse message without proper Referred-By
  - 429 Provide Referrer Identity

Protected REFER Example



Enforcing Protected REFER Example



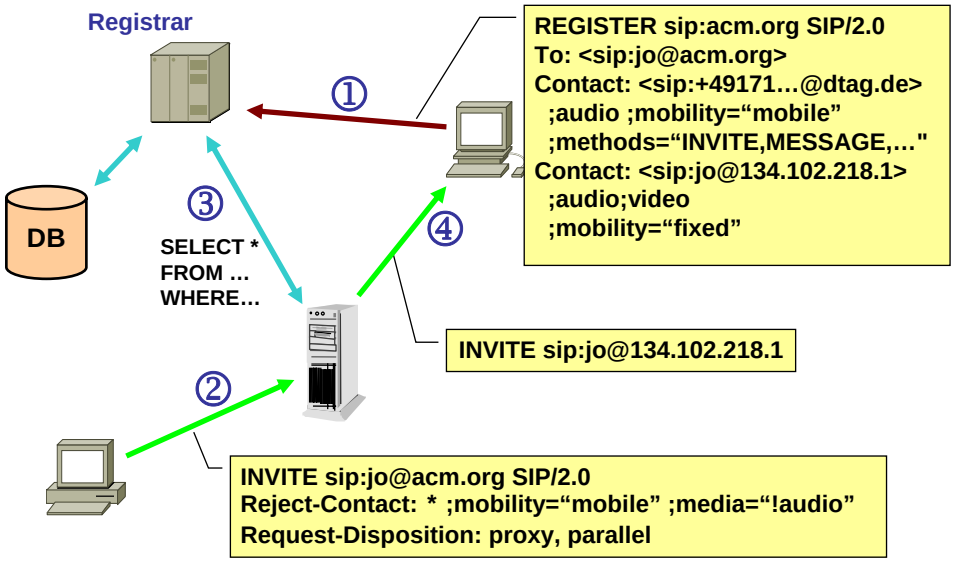
REFER Semantics?

- Refer-To: supports arbitrary parameters
  - Including method= and response=
  - May populate various headers of a message
  - Enables remote control of a SIP UA
- Default: INVITE method for SIP URIs
  - Inside dialog: call transfer
  - Outside dialog: click-to-dial
- But generalized command: Apply <method> to <URI>
  - "Force" a remote UA to do something
  - In principle broad applicability but only little semantics "defined"
    - Fetch the content at http:// …
    - Subscribe to a certain resource
    - Send a certain reply to a request
  - Handle with care…

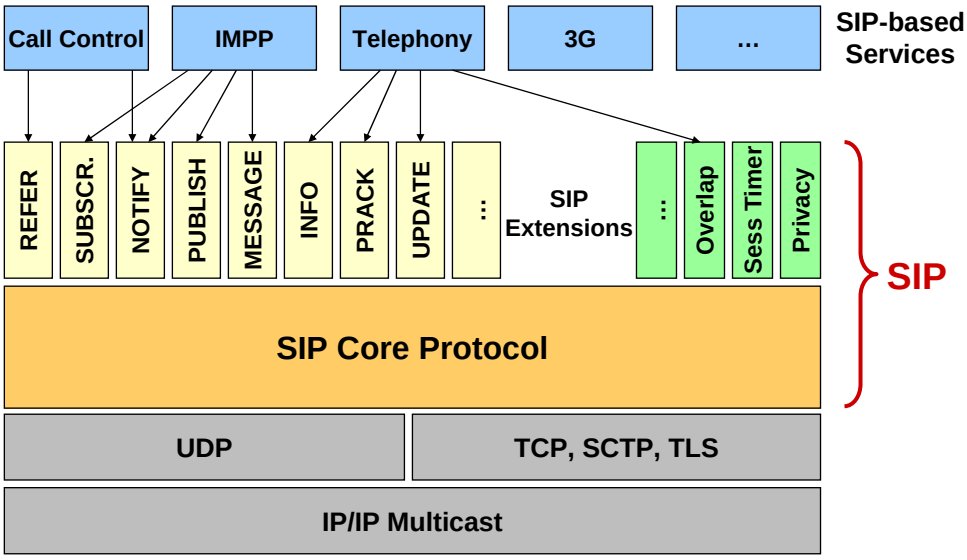
Caller Preferences / Callee Capabilities

- Caller may provide preferences for request routing
  - URI-based Proxy or redirect
  - Refuse particular URIs
  - Forking
  - Recursive search
  - Parallel or sequential search
- SIP Extensions (RFC 3840, 3841)
  - Request-Disposition: Request routing in SIP servers
  - Accept-Contact: Change address ordering
    - Parameters require and explicit to control matching process
  - Reject-Contact: Reject particular URIs
  - Implied predicates from SIP method, events
  - Additional parameters for Contact: headers in REGISTER
    - Modelled following RFC 2533, but with simplified syntax

Preference/Capability Matching



SIP Service Creation Model



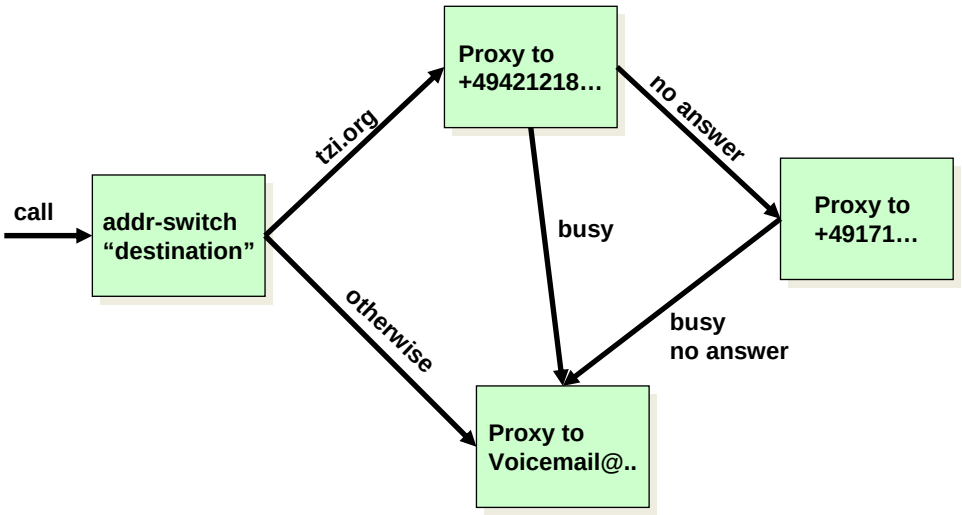
SIP Service Creation

- Extensions to SIP
  - Methods, headers, protocol handling
    (as just discussed for some examples)
- Usages of SIP
  - Combining existing methods, headers, etc. to create a service
  - E.g., SIP for telephony, presence, instant messaging
- Tuning behavior of SIP entities
  - Based upon information available from SIP
  - Causing effects through SIP mechanisms
  - Accessible by means of APIs: in UAs, proxies, and B2BUAs
  - E.g., user location
- Interaction with application servers
  - Stimulus signaling without semantic extensions to SIP
  - E.g., using RTP or other protocols to carry signaling

Call Processing Language (CPL)

- Server provides information about incoming message
  - Destination
  - Originator
  - Caller preferences
  - Contents of several important header fields
  - Media description
  - Security parameters
  - …
- CPL scripts can specify several actions to take
  - Reject, redirect or proxy incoming message
  - Set timeout values for actions
  - Context-dependent choice of different actions
  - Perform location lookups
  - …

Example CPL Script



```
<?xml version="1.0" ?>
<cpl>
  <subaction id="vm">
    <location url="sip:voicemail@dmn.tzi.org">
      <proxy />
    </location>
  </subaction>
  <incoming>
    <address-switch field="destination" subfield="host">
      <address subdomain-of="tzi.org">
        <location url="tel:+49421218…">
          <proxy timeout="10">
            <busy> <sub ref="vm" /> </busy>
            <noanswer>
              <location url="tel:+49171…">
              […]
            </noanswer>
          </proxy>
        </location>
      </address>
    </address-switch>
  </incoming>
</cpl>
```

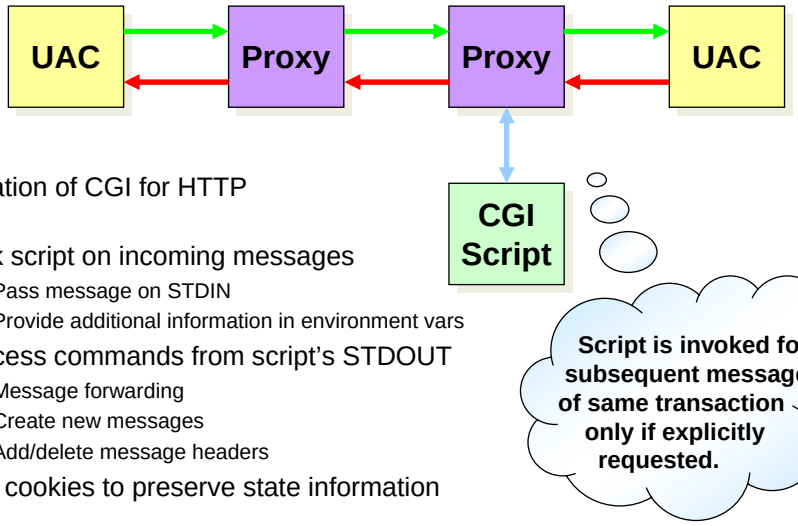

Service Creation

- Flexible processing of SIP messages in *proxy*
  - Rapid development of supplementary services
  - Pass incoming requests to external script for processing

→ SIP Common Gateway Interface (CGI)

- Adaptation of HTTP CGI
- Support SIP's idiosyncrasies:
  - Transport protocols UDP, TCP
  - Central role of proxy servers
  - Persistent transaction state
  - Registrations
  - Request forking, recursive search, timeouts
  - …

SIP CGI



Adaptation of CGI for HTTP

- Fork script on incoming messages
  - Pass message on STDIN
  - Provide additional information in environment vars
- Process commands from script's STDOUT
  - Message forwarding
  - Create new messages
  - Add/delete message headers
- Use cookies to preserve state information

SIP Service APIs

- SIP Servlets
  - Standardized model and API for Java-based service creation
  - Built similar to HTTP servlets
- Vendor-specific APIs
  - for SIP servers
  - SDKs for SIP UAs
- JAIN (Java APIs for Intelligent Networks)
  - Java-based API for service creation across SIP and circuit-switched networks
  - JAIN SIP
  - JAIN SIP Lite
  - SIP Servlets (see above)
- Parlay
  - Joint effort of ETSI, 3GPP, and Parlay group

Interaction with Application Services

- Traditional PSTN / PBX services often controlled via DTMF
  - Stimulus signaling with semantics defined by the application
  - Assumes that signaling and media paths are identical
- Does not apply to SIP
  - Media and signaling may go separate paths
  - Application servers may be on neither path
- SIP defines Application Interaction Framework
  - Taxonomy for interactions between users and application servers
  - Focus on stimulus-based signaling
- KPML Approach
  - SUBSCRIBE to register interest of an application server in user input
  - XML document to describe input expected from user
  - NOTIFY to inform server about key events

Signaling-based Application Servers