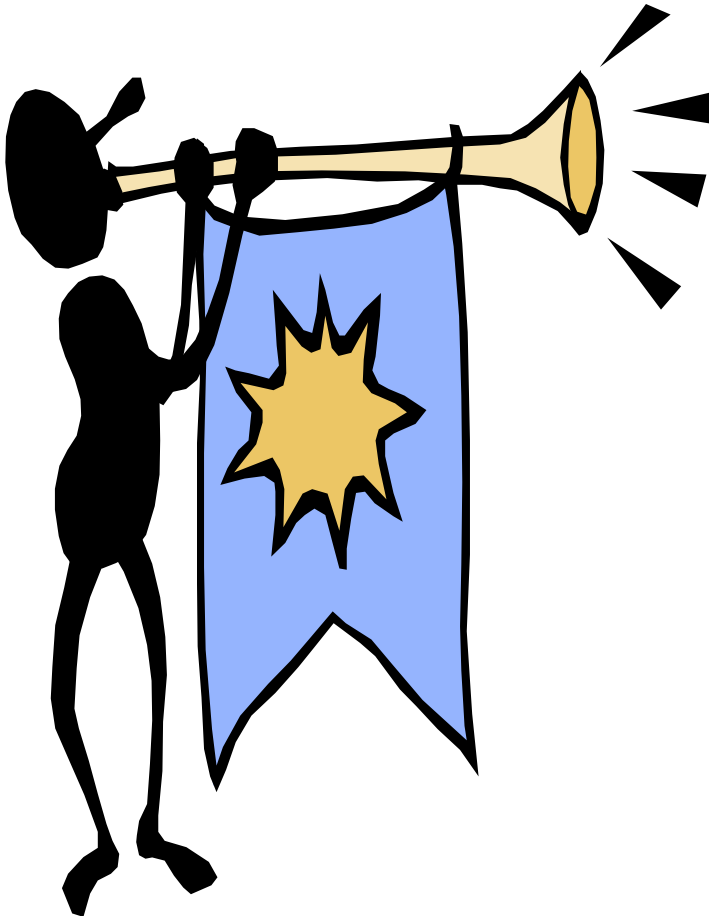
Chapter IV
SIP Session Signaling
And
SIP Specific Value Added Service Technologies

Roch H. Glitho

Part I
SIP Session Signaling

Outline



1. Introduction
2. Core SIP
3. Selected Extensions

Introduction: Signaling vs Media

Signaling:

- Session establishment
- Session tear down
- Changes to the session
- Supplementary services

Media:

- Actual communication data: encoded voice stream, video stream,…

Introduction: SIP

Signaling Protocols:

- SIP and H.323

Media transport protocol:

- RTP

Why SIP?

SIP: Prime signaling system because adopted by all key next generation networks:

- 3GPP
- 3GPP2
- PacketCable:

SIP: Introduction

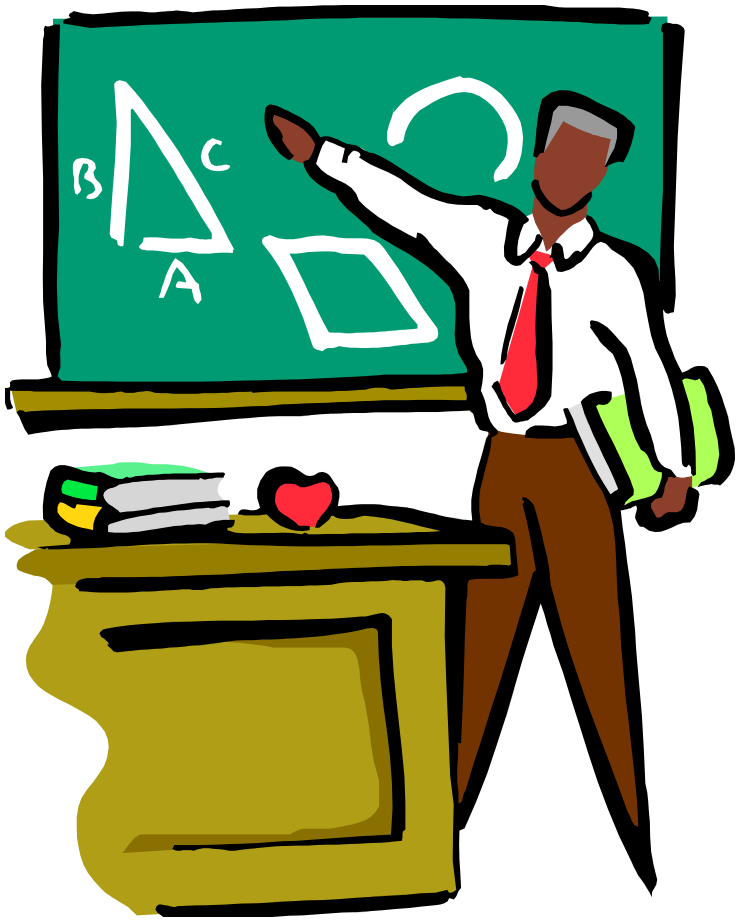
A set of IETF specifications including:

- **SIP core signalling:**
  - RFC 2543, March 1999
  - RFC 3261, June 2002 (Obsoletes RFC 2543)

- **SIP extensions (e.g. RFC 3265, June 2002 - Event notification)**
  - May have nothing to do with signalling
- IMS related extensions.

- **Used in conjunction with other IETF protocols**
  - QOS related protocol (e.g. RSVP)
  - Media transportation related protocol (e.g. RTP - RFC 1889)
  - Others (e.g. SDP - RFC 2327)

Session Initiation Protocol (SIP) - Core



1. Introduction
2. Functional entities
3. Messages
4. SDP
5. Examples

SIP: Introduction

SIP core Signaling

- A signalling protocol for the establishment, modification and tear down of multimedia sessions
- Based on HTTP

A few key features

- Text based protocol
- Client/server protocol (request/response protocol)

SIP: The Request

Request messages

- Methods for setting up and changing sessions
  - . INVITE
  - . ACK
  - . CANCEL
  - . BYE

- Others
  - . REGISTER (Registration of contact information)
  - . OPTIONS (Querying servers about their capabilities)

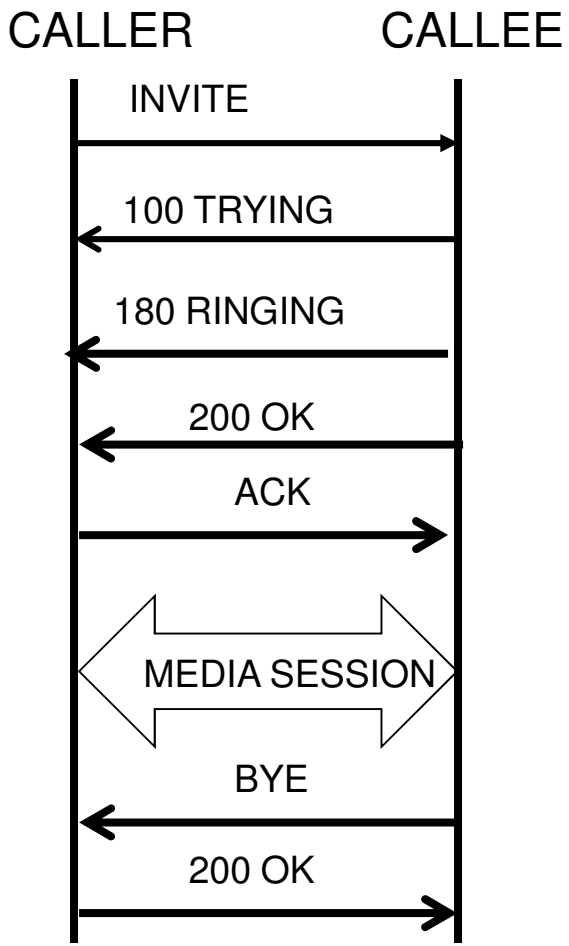
SIP: The Response

Response message

- Provisional
- Final

Examples of status code
1xx: Provisional
2xx: Success
6xx: Global failure

SIP: A basic peer to peer call scenario



SIP: The functional entities

User agents

- **End points, can act as both user agent client and as user agent server**
  - **User Agent Client: Create new SIP requests**
  - **User Agent Server: Generate responses to SIP requests**

Proxy servers

- Application level routers

Redirect servers

- Redirect clients to alternate servers

Registrars

- Keep tracks of users

SIP: The functional entities

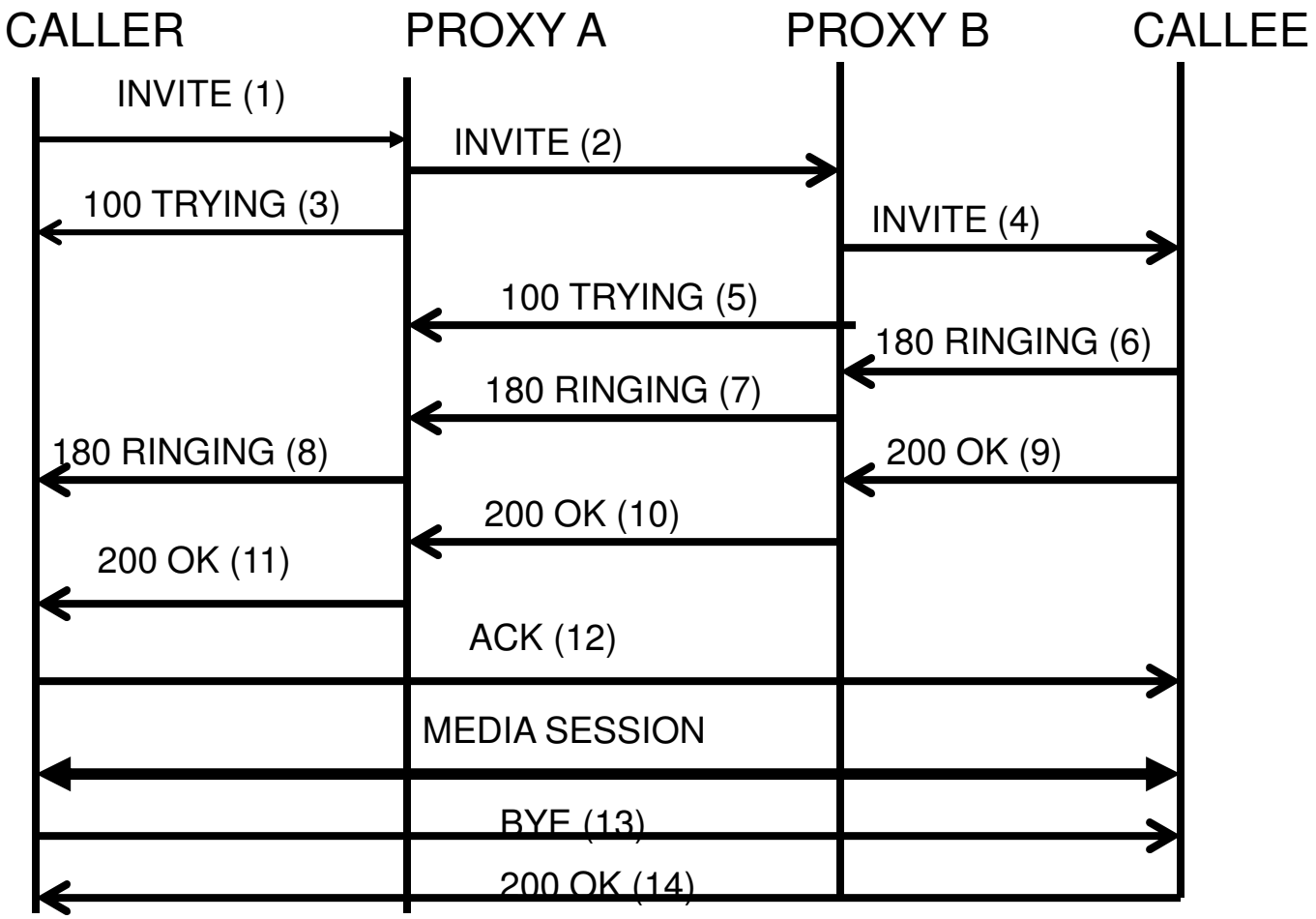
State-full proxy

- Keep track of all transactions between the initiation and the end of a transaction
- Transactions:
  - Requests sent by a client along with all the responses sent back by the server to the client

Stateless proxy

- Fire and forget

SIP: A call scenario



SIP: The messages

Generic structure

- Start-line
- Header field(s)
- Optional message body

Request message

- Request line as start line
  - . Method name
  - . Request URI
  - . Protocol version

Response message

- Status line as start line
  - . Protocol version
  - . Status code
  - . Reason phrase (Textual description of the code)

SIP: Examples of messages from the RFC

An example of an INVITE

```
INVITE sip:bob@biloxi.com SIP/2.0
  Via: SIP/2.0/UDP
 pc33.atlanta.com;branch=z9hG4bK776asdhds
  Max-Forwards: 70
  To: Bob <sip:bob@biloxi.com>
  From: Alice <sip:alice@atlanta.com>;tag=1928301774
  Call-ID: a84b4c76e66710@pc33.atlanta.com
  CSeq: 314159 INVITE
  Contact: <sip:alice@pc33.atlanta.com>
  Content-Type: application/sdp
  Content-Length: 142
```

SIP: Examples of messages from the RFC

An example of RESPONSE to the OPTIONS request

```
SIP/2.0 200 OK
   Via: SIP/2.0/UDP
  pc33.atlanta.com;branch=z9hG4bKhjhs8ass877
    ;received=192.0.2.4
   To: <sip:carol@chicago.com>;tag=93810874
   From: Alice <sip:alice@atlanta.com>;tag=1928301774
   Call-ID: a84b4c76e66710
   CSeq: 63104 OPTIONS
   Contact: <sip:carol@chicago.com>
   Contact: <mailto:carol@chicago.com>
   Allow: INVITE, ACK, CANCEL, OPTIONS, BYE
   Accept: application/sdp
   Accept-Encoding: gzip
   Accept-Language: en
   Supported: foo
   Content-Type: application/sdp
```

SDP

Session Description Protocol

- Convey the information necessary to allow a party to join a multimedia session
  - Session related information
  - Media related information
- Text based protocol

- No specified transport
  - Messages are embedded in the messages of the protocol used for the session
    - Session Announcement Protocol (SAP)
    - Session Initiation Protocol (SIP)

SDP

Session Description Protocol

Use with SIP

- Negotiation follows offer / response model
- Message put in the body of pertinent SIP messages
  - INVITE Request / response
  - OPTIONS Request / response

SDP

Session Description Protocol

- <Type> = <Value>
- Some examples
  - Session related
    - v= (protocol version)
    - s= (Session name)
  - Media related
    - m= (media name and transport address)
    - b= (bandwidth information)

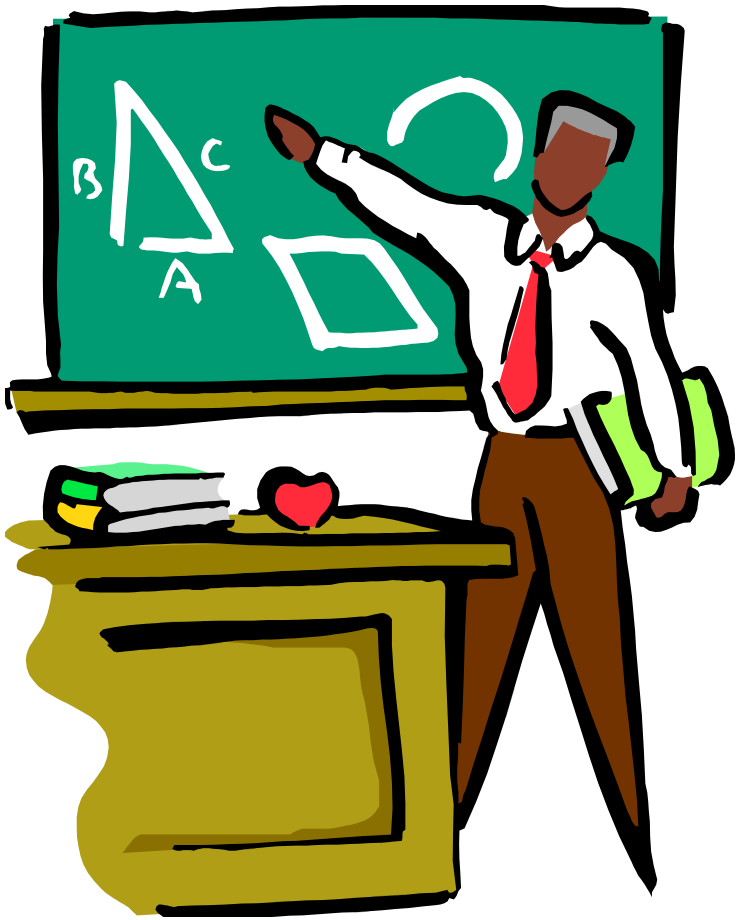
SDP: Examples of messages from the RFC …

Session Description Protocol

An example from the RFC …

v=0
o=mhandley 2890844526 2890842807 IN IP4 126.16.64.4
s=SDP Seminar
i=A Seminar on the session description protocol
u=http://www.cs.ucl.ac.uk/staff/M.Handley/sdp.03.ps
e=mjh@isi.edu (Mark Handley)
c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
a=recvonly
m=audio 49170 RTP/AVP 0
m=video 51372 RTP/AVP 31
m=application 32416 udp wb
a=orient:portrait

SIP – Selected Extensions



1. Event framework
2. INFO method

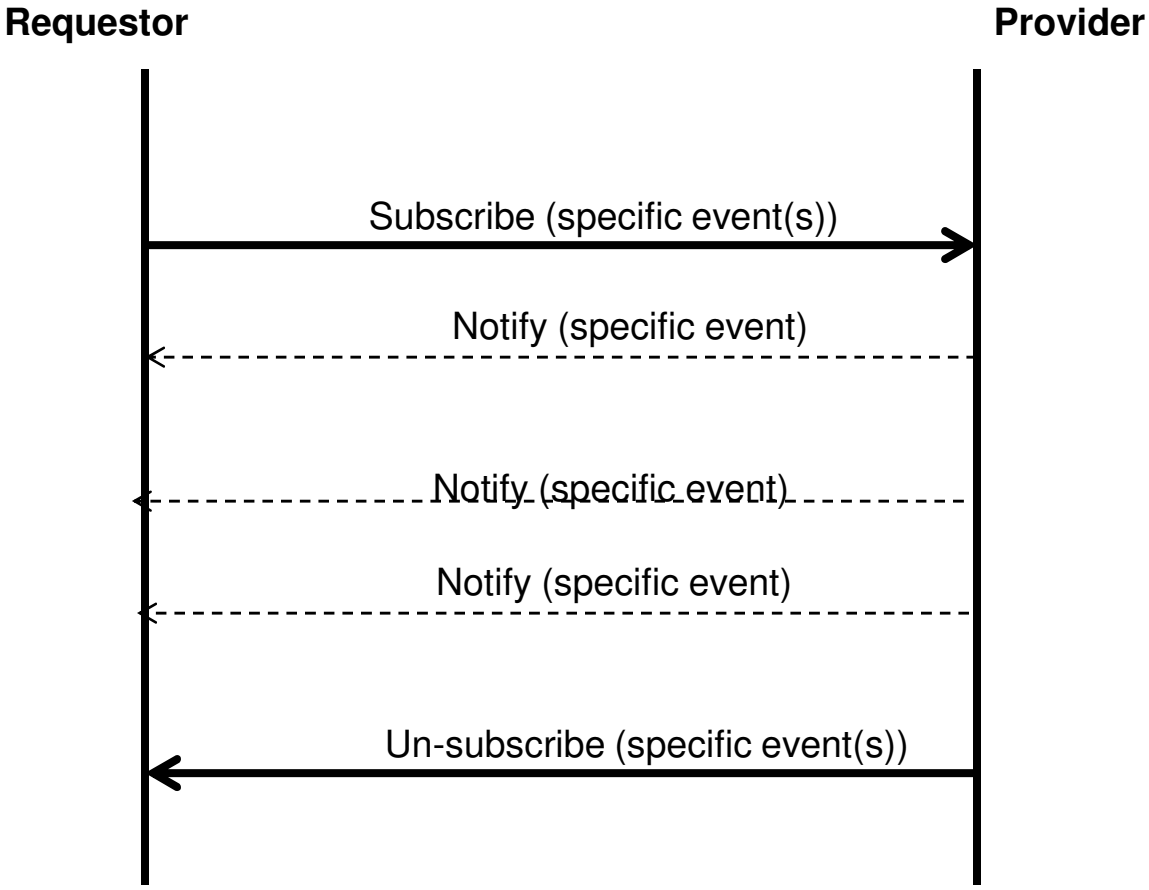
Event Notification

Motivation

- Necessity for a node to be asynchronously notified of happening (s) in other nodes
  - Busy / not busy (SIP phones)
    - A client A can call again a client B when notified that B is now not busy
  - On-line / Off-line
    - Buddy list

Event Notification

Conceptual framework



Event Notification

The SIP Event Notification Framework

- Terminology
  - Event package:
    - Events a node can report
    - Not part of the framework – Part of other RFCs
  - Subscriber
  - Notifier
- New Messages
  - Subscribe
    - Need to be refreshed
    - Used as well for un-subscribing (expiry value put to zero)
  - Notify

Event Notification

The SIP Event Notification Framework

- More on the methods
  - New headers
    - Event
    - Allow-Events
    - Subscription state

Event Notification

An example of use: REFER Method

- Recipient should contact a third party using the URI provided in the CONTACT field
  - Call transfer
  - Third party call control
- Handled as Subscribe / notify
  - REFER request is considered an implicit subscription to REFER event
    - Refer-TO: URI to be contacted
    - Expiry determined by recipient and communicated to sender in the first NOTIFY
    - Recipient needs to inform sender of the success / failure in contacting the third party

Event Notification

Another example of use: Presence

- Dissemination/consumption of presence information (e.g. on/off, willingness to communicate, device capabilities, preferences)
  - Numerous applications
    - Multiparty sessions initiated when a quorum is on-line
    - News adapted to device capabilities
- Several standards including SIMPLE (SIP based)
  - Handled as Subscribe / notify in SIMPLE
    - Watchers / presentities
      - Explicit subscriptions
      - Explicit notifications

INFO Method

Allow the exchange of non signalling related information during a SIP dialog

- Semantic defined at application level
- Mid-call signalling information
  - DTMF digits with SIP phones
- Info carried as
  - Headers and/or
  - Message body

References

Core SIP

- **SIP core signalling:**
- H. Schulzrinne, an J. Rosenberg, SIP: Internet Centric Signaling, IEEE Communications Magazine, October 2000
- RFC 3261, June 2002 (Obsoletes RFC 2543)
- RFC 2327 (SDP)

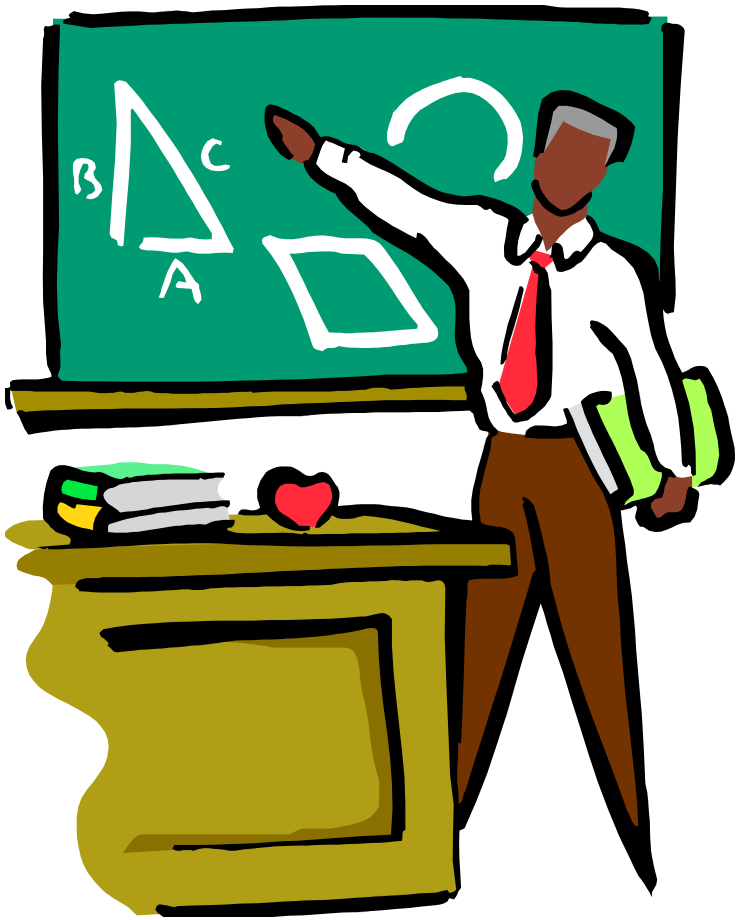
SIP extensions

No overview paper

- **RFC 3265, 3515 (Event framework)**
- **RFC 2976 (INFO Method)**

Part II
SIP Specific Value-Added Service Technologies
By Dr Hechmi Khlifi

SIP Specific Value Added Service Technologies



1. Introduction: SIP specific architectures vs protocol neutral architectures
2. SIP CGI
3. SIP servlet API

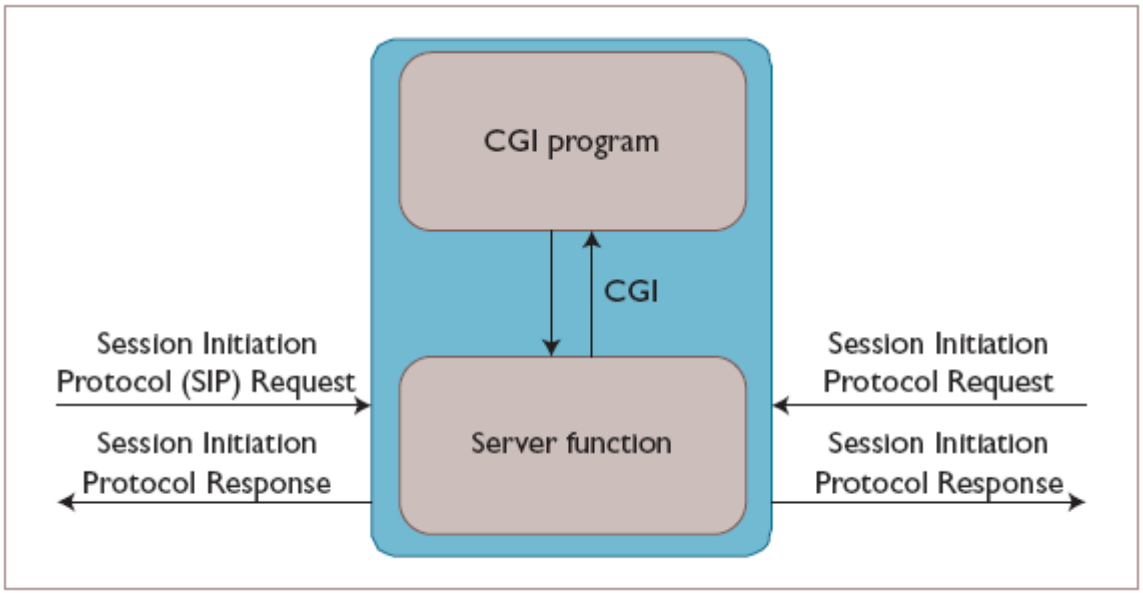
Introduction: SIP specific architectures

- Servers built using SIP specific architectures act as redirect servers, proxy servers, originating user agents, terminating user agents, or back-to-back user agents.
- They have SIP signaling capabilities and are directly involved in the call's signaling flow.
- Implementation techniques: SIP CGI, SIP Servlet

Introduction: Protocol neutral architectures

- Servers built using protocol neutral architectures can provide the same services as the SIP application server, but are:
  - signaling independent (i.e. could be used with any signaling protocol).
  - Are not directly involved in the SIP calls' signaling flow.
- Examples of APIs: TAPI, TSAPI, JTAPI, Parlay and Web services/Parlay X
  - Focus of this lecture: SIP specific value added services technologies (i.e. SIP application servers)
  - Web services / Parlay-X will be discussed in another lecture

SIP CGI



Key features

- Inspired by HTTP CGI
- The server passes the message body to the script through its standard input
- Services are written as CGI scripts

SIP CGI : shortcomings

- Difficult to program
- Require a deep understanding of SIP protocol

SIP Servlet: Introduction

Key features

- Signalling protocol specific (I.e. applicable to SIP only)
- Prime target: trusted parties
  - Service providers
  - Third party developers
- Very few constraints on what can be done
- Reliance on HTTP servlet API
  - HTTP servlet API is widely used in the Internet world
    - A tool which relies on it should attract many users including Web masters.
    - A wide range of developers should favour the development of cool and brand new services

.

HTTP servlet API ...

Creation of dynamic Web content

- Servlet
  - Java component
  - Generate content on the fly, just like HTTP CGI
    - interface between HTTP request and data bases
    - Forms
    - Dynamic information (e.g. date, number of visitors)

HTTP servlet API ...

Servlet container (also know as servlet engine)

- Servlet container (or servlet engine)
  - Contains the servlets
  - Manage the servlets through their life cycle
    - Creation
    - Initialisation
    - Destruction
    - Receives and decodes of HTTP requests
    - Encodes and sends of HTTP responses

HTTP servlet API ...

Pros

Address most HTTP CGI shortcomings

- **Performance**
  - Can keep data base connections open
- **Scalability**
  - Servlet containers can be accessed remotely

Cons

- Language dependence

SIP servlet API...

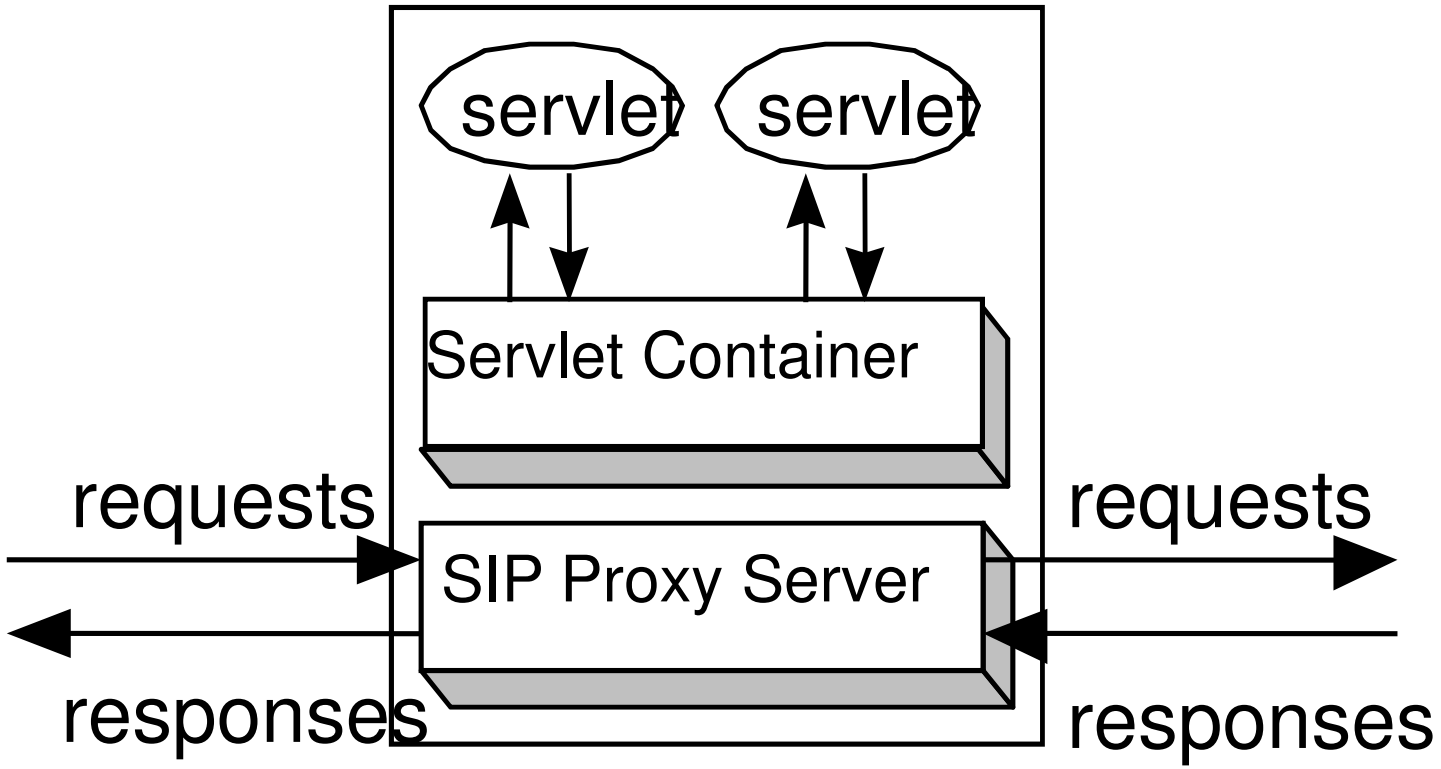
Adjustments made to HTTP servlet:

- Initiate requests
  - Needed for some services
    - wake up call
- Receive both requests and responses
  - Needed for some services
    - Terminating services (e.g. call forward on busy)
- Possibility to generate multiple responses
    - Intermediary responses, then final response
- Proxying requests, possibly to multiple destinations
  - Needed for applications such as intelligent routing

.

SIP Servlet container ...

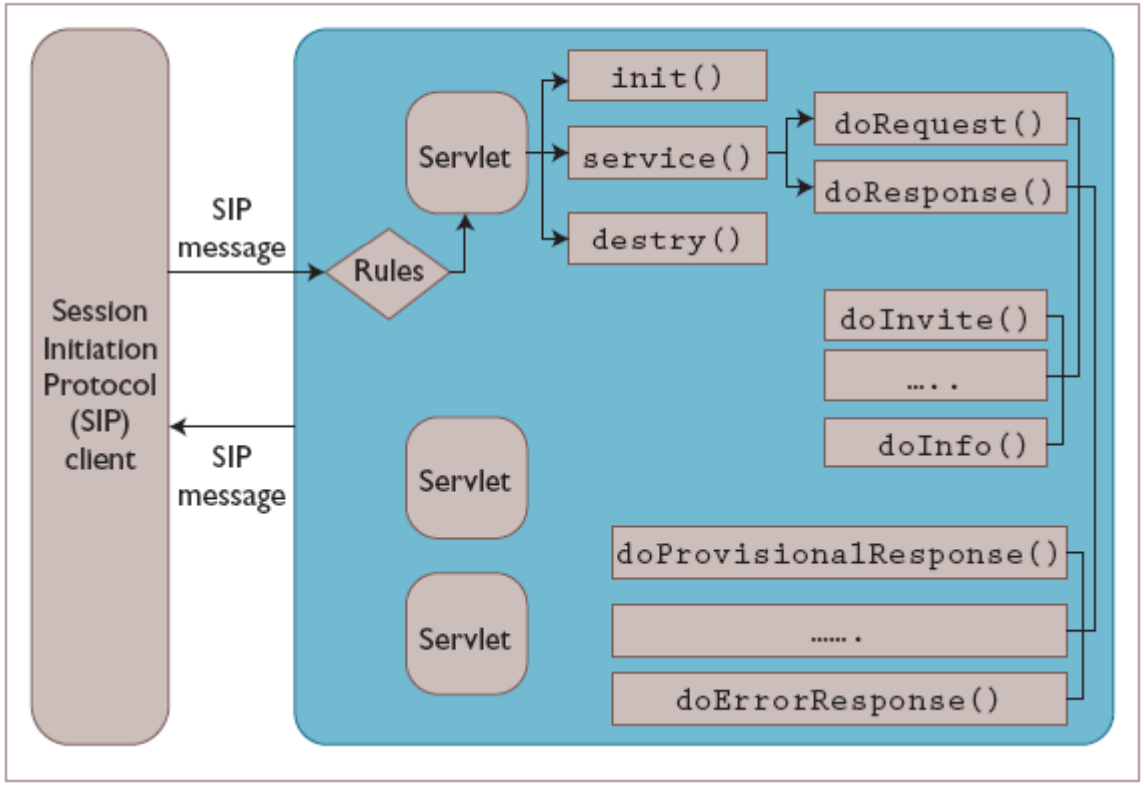
A container collocated with a proxy server



SIP servlet Request interface ...

SIP specific Request handling methods (Based on both core SIP and SIP extensions)·

- doInvite
- doAck
- doOptions
- doBye
- doCancel
- doRegister
- doSubscribe
- doNotify
- doMessage
- doInfo



SIP servlet Response interface ...

SIP specific Response handling methods (Based on both core SIP and SIP extensions):

- doProvisionalResponse
- doSuccessResponse
- doRedirectResponse
- doErrorResponse

An example of service:

Algorithm for call forward

- Get the destination from the SIP request
  - Done by retrieving the To_Field by using the GetHeaders
- Obtain the forwarding address from a data base
- Forward the call
  - Done by setting the Request_URI (and not the To_field) using the setHeader

Another example:

Algorithm for a centralized dial-out conference

Assumptions

- INVITE is used
- URIs of participants are put in the INVITE body

Agorithm used in servlet:

- Use GetContent to get the participant's URIs from INVITE Request
- Use doINVITE to generate and send an INVITE to each participant.

Example

```
public class RegistrarServlet extends SipServlet{

    protected void doRegister(SipServletRequest request) throws     ServletException, IOException {

        SipServletResponse response = request.createResponse(200);

            response.send();
            logger.log(Level.FINE, "Sent 200 response.");
        } catch(Exception e) {

            response.setStatus(500);
            response.send();
        }

    }
```

Pros and cons

Pros

- Possibility of creating a wide range of services due to the full access to all the fields from the SIP Request
- More performance and more scalability
- Possibility to create services that combine both HTTP and SIP

Cons:

- SIP Servlet is not exactly the same thing as HTTP Servlet
- Language dependence

.

References

- R.H. Glitho, “Advanced Services Architectures for Internet Telephony: A Critical Overview,” IEEE Network, July 2000, pp. 38–44.
- Hechmi Khlifi, Jean-Charles Grégoire: IMS Application Servers: Roles, Requirements, and Implementation Technologies. IEEE Internet Computing 12(3): 40-51 (2008)