Signalling Protocol Specific Service Architectures

INSE 7110 – Winter 2004
Value Added Services Engineering in Next Generation Networks
Weeks #6 & 7
Signaling protocol specific architectures ...

1. H.323 specific service architectures
2. SIP specific service architectures
H.323 specific architecture: The supplementary service approach

Specifying services instead of building blocks !!!
H.323 Supplementary services

1. Introduction
2. The services
3. The architecture
4. Beyond supplementary services
5. Pros and cons
Introduction

Architecture based on pre-IN and “PBX” thinking:
- Standardization of services (instead of services capabilities)
  - H.450.1 Recommendation
    - General principles
  - H.450.X
    - Roughly 1 per service

Main architectural principle
- “Overlay” dedicated network for realizing services
  - Functional entities
  - Messages
The services ...

Services standardized so far:

- Call transfer
- Call diversion
- Call hold
- Call park and pick up
- Call waiting
- Message waiting indication
- Name identification
- Call completion
- Call offer
- Call intrusion
The services...

**Call transfer**

Allow a user A in communication with user B to establish a new call between user B and user C

- First case: User A has a call established with user C before the transfer
- Second case: User A does not establish any call with user C before the transfer

**Call diversion**

Divert the call (before answering it) if some conditions are met

- Unconditional
- Busy
- No reply
The services ...

**Call hold**
Allow a user A to put user B on “hold” after the call has been established
- User B can hear music / advertisement in the meantime
Also allow user A to retrieve a call previously put on hold

**Call park and pick up**
Generalization of call hold / retrieve
- Parking places (i.e. identifier for each parked call)
- Retrieval using identifiers
The services ...

**Call waiting**

Allow a busy user to be notified of an incoming call and to decide how to proceed (Classical example; Internet call waiting)
- Accept (i.e. give up on previous call)
- Reject
- Divert

**Message waiting indication**

Self explanatory
- User can call a message center
The services ...

Name indication
Self explanatory ...

Call completion
Camp on ....
- Allow caller to establish a call with a busy callee as soon as callee is free and without having to re-dial callee’s number.
The services ...

**Call offer** ...

Strong form of call completion
Allow caller to offer a call to a busy callee and wait till busy callee accepts the call ...

**Call intrusion**
Allow user A to establish a call with a busy user B by breaking into the call between B and C
- Result: 3 party call
The architecture

Entities
  - Supplementary service control entities
    • Reside in H.323 functional entities
    • Exchange messages for realising services

Messages
  - Service execution related messages
  - Activation / de-activation at user level

Entities and messages are defined on service basis
Call diversion service (H.450.3)

**Flavours**
- Unconditional
- On busy
- On no reply within a given period of time

**Functional entities**
- Activating / de-activating end points
- Original diverting (served) / last diverting (served) end points
  - Points at which the call is diverted
- Interrogating end point
  - Get information (from diverting end points) on activation status / number to which to divert and so on..
- Re-routing end point
  - Entity that does the re-routing
Call diversion service (H.450.3) – Case 1

Terminal 1                                Gatekeeper (Re-routing entity)                                Terminal 2 (Diverting entity)                                Terminal 3

Set up …

Set up

No reply …
Detect that call should
Be re-directed to Terminal 3

Facility (Call re-routing request)

Facility (Call re-routing answer)

Release complete

Set up
Call diversion service (H.450.3) – Case 2

Terminal 1                                Gatekeeper            Terminal 2                          Terminal 3 (Re-routing / Diverting entity)

Set up …

Set up

No reply …
Detect that call should
Be re-directed to Terminal 3

Set up
Beyond supplementary services: Annex K

Annex K

- Cater to additional services such as third party call control
  - HTTP link for user interactions
  - Implementation dependent mapping between actions required by the user and the appropriate H.450.x messages

Annex L

- Allow the introduction of new supplementary services without upgrading terminals
  - New services are introduced in feature server
  - Stimulus based signalling procedures between terminals, feature server and gate keeper.
Beyond supplementary services: Annex K

Client entity

H.323 endpoint

Service provider

HTTP

HTTP server

H.323 entity (endpoint or GK)

Service control plane

Call control plane

web-browser

H.323

T160891000
Pros and cons

Pros

– More suitable for PBX environment with lower expectations on services
  • Fixed set of supplementary services
  • Easy interoperability due to detailed specifications of messages

Cons

– Highly unsuitable for next generation networks, despite of the recent quick “fixes” (e.g annexes K and L)
  • Limited range of services
  • Third parties are not allowed
The SIP specific tools/ architectures …

1. SIP CGI
2. SIP Servlet API
3. 3GPP SIP Application Server
SIP CGI

1. Introduction
2. HTTP and HTTP CGI
3. SIP CGI
4. Example
5. Pros and cons
Introduction ...

Key features

– Signalling protocol specific (i.e. applicable to SIP only)
– Prime target: trusted parties
  • Service providers
  • Third party developers
– Reliance on HTTP CGI
  • HTTP CGI is widely used in the Internet world for Web page development
    – A tool which relies on it should attract many users including the Web masters.
    – A wide range of developers should favour the development of cool and brand new services
HTTP ...

Object oriented application level request/reply protocol for distributed multimedia information systems

- Clients
  - establish connections
- User agents
  - Initiate requests
- Servers
  - accept connection, serve requests and send responses back
  - Origin servers:
    - servers on which resources are created and reside
  - Resource
    - data object or service
- Proxies
  - act as both servers and clients

...
HTTP ...

**Message**
- Type: request or reply
- Headers
  - General header
    - Applicable to both request and reply (e.g. date)
  - Request (or reply) header
    - Method to be applied to the resource (e.g. GET, POST)
    - Resource id
    - Protocol version
    - Additional information (e.g. host, user agent)
    - In case of reply: status line
  - Entity header: Optional information on body
- Body (optional) (e.g. HTML file)
- length
HTTP CGI ...

**Creation of dynamic Web content**
- Script that can work with most programming language
- Generate resource identified in a request on the fly
  - interface between HTTP request and databases
  - Forms
  - Dynamic information (e.g. date, number of visitors)

**Environment variables allow the script to access**
- HTTP headers
- Non request specific information (e.g. server host name)
HTTP CGI ...Example of dynamic content generation

Source:
http://www.cc.ukans.edu/~acs/docs/other/forms-intro.shtml

set crlf to (ASCII character 13) & (ASCII character 10) set header to "HTTP/1.0 200 OK" & crlf - & "Server: MacHTTP" & crlf set header to header & "MIME-Version: 1.0" - & crlf & "Content-type: text/html" set header to header & crlf & crlf - & "<title>Server Script</title>" set body to "<h2>The time is: </h2>" - & (current date) & "<p><p>" return header & body
HTTP CGI ...Example of dynamic content generation

Source:
http://www.cc.ukans.edu/~acs/docs/other/forms-intro.shtml

GET /scripts/date HTTP/1.0 Accept: www/source Accept: text/html
  Accept: image/gif User-Agent: Lynx/2.2 libwww/2.14 From:
  montulli@www.ku.edu * a blank line *
HTTP CGI ...Example of dynamic content generation

Source:
http://www.cc.ukans.edu/~acs/docs/other/forms-intro.shtml

HTTP/1.0 200 OK" Server: MacHTTP" MIME-Version: 1.0 Content-type: text/html * blank line * <title>Server Script</title> <h2>The time is:</h2> February 3, 2004 3:15 pm <p><p>
HTTP CGI ...

**Pros**
- Programming language independence

**Cons**
- Poor performance
  - Scripts are not persistent: connection to a data base needs to be established each time
- Lack of scalability
  - Scripts need to reside on same server as resource
SIP CGI vs. HTTP CGI (HTTP CGI model)
SIP CGI vs. HTTP CGI (SIP CGI model)
SIP CGI vs. HTTP CGI

Examples of adjustments

- Script output is not necessarily the response to send
  - Case of call forward
    - Script will instruct the server to proxy the request to right location
- Scripts are persistent
  - Several interactions are required between script and server for some services
Algorithm implemented by a script for call forward

- Get the destination from the SIP request
  - Done by retrieving the To_Field from the environment variable HTTP_TO
- Obtain the forwarding address from a data base
- Forward the call
  - Done by using the CGI-PROXY-REQUEST-TO CGI action
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
- Language independence

Cons:

- CGI is less and less used in the Web world
- SIP CGI is not exactly the same thing as HTTP CGI
- Lack of scalability (e.g. scripts need to reside on same server)
- Performance issues
SIP Servlet API

1. Introduction
2. HTTP servlet API
3. SIP servlet API
4. Examples
5. Pros and cons
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 database 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
A container collocated with a proxy server
SIP servlet Request/response hierarchy...

Build on the generic servlet API, like HTTP servlet

- javax.servlet.sip (just like javax.servlet.http)
- Container must support
  . javax.servlet
  . Javax.servlet.sip
SIP servlet request response hierarchy...

Request-response Hierarchy

- ServletRequest
  - SIPServletRequest
  - SIPServletMessage
  - SIPServletResponse

- ServletResponse
  - SIPServletResponse
  - Javax.servlet
  - Javax.servlet.sip
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
**SIP servlet Message interface ...**

**SIP specific message handling methods (Access to message header):**

- `getHeader`
- `getHeaders` (Used when there are several headers)
- `setHeader`
- `addHeader`

**Note:** system headers cannot be manipulated by servlets (Call-ID, From, To)
SIP servlet Message interface ...

SIP specific message handling methods (Access to message content):

- `getContentLength`
- `setContentLength`
- `getContentType`
- `getContent`
- `getRawContent`
- `setContent`
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

Algorithm 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.
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
3GPP SIP Application Server

1. Introduction

2. Interactions between SIP AS and S-CSCF

3. S-CSCF service control model
Introduction ...

3GPP SIP application server

- Provide value added services
- Reside in user’s home network or in a third party location
- Interact with serving CSCF via IP Multimedia Service Control Interface (ISC)
  - SIP plus some enhancements
- No tool is specified and any SIP specific tool can be used
  - SIP CGI
  - SIP servlet API
- State-full models
  - Incoming call legs
  - Outgoing call legs
Interactions – SIP AS as terminating user agent (e.g. call screening)

Application Server

SIP leg #1
From: X
To: Y
Call-ID: Z

S-CSCF

SIP leg #1
From: X
To: Y
Call-ID: Z
Interactions – SIP AS as originating user agent (e.g. wake up call)
Interactions – SIP user agent as a proxy (e.g. call forward)...

Application Server

SIP leg #1
From: X
To: Y
Call-ID: Z

SIP leg #1
From: X
To: Y
Call-ID: Z

S-CSCF

SIP leg #1
From: X
To: Y
Call-ID: Z

SIP leg #1
From: X
To: Y
Call-ID: Z
Interactions – SIP AS as third party call control

Application Server

SIP leg #1
From: X
To: Y
Call-ID: Z

SIP leg #2
From: P
To: Q
Call-ID: R

S-CSCF

SIP leg #1
From: X
To: Y
Call-ID: Z

SIP leg #2
From: P
To: Q
Call-ID: R
Service control model ...

- Application Server
  - AS-ILCM
  - AS-OLCM
- ISC
- ILCM
- OLCM
- ILSM
- OLSM
- Model
- S-CSCF
- Call Records
- Incoming Leg
- Outgoing Leg
To probe further...

**H.323 specific approaches**

- ITU-T specifications: H.450.x series, recommendation H.323

**SIP specific approaches**

- IETF RFCs: SIP CGI - RFC 3050
- Java Developers Community draft JSR 116
  [http://www.jcp.org/aboutJava/communityprocess/review/jsr116/](http://www.jcp.org/aboutJava/communityprocess/review/jsr116/)
- W. Leekwijck and D. Brouns, SIPlets: Java Based Service Programming for IP Telephony
- 3GPP TS 23.228, 23.002, [http://www.3gpp.org/](http://www.3gpp.org/)