Francqui Piet Demeester

1. Internet Support for Multimedia Flows

Dia 3

The general requirements of a telephone network are :
 support the transport of the voice signal
 support the set-up of the connection
 support other functions such as billing, management, etc.
Some basic features are :
Connection oriented : before user data (= the telephone conversation) is exchanged one has to set up a call which will close the appropriate cross-points in the switches (or assign the correct time slots) resulting in an end-to-end circuit.
The end-to-end circuit will support a constant bandwidth (3.4 kHz for analog telephony) or bitrate (64 kbit/s for digital telephony) during the call.
Because the network is circuit switched, hardly any voice sample will be lost.
The network will keep the delay as low as possible. Roundtrip delay values below 15 msec are required when trying to avoid echo. In general this will not be possible.
Delay values below a few 100 msec are required to keep good interactivity during the call. This is a major problem when using satellites to interconnect phones (e.g. during trans-atlantic calls). A phone call via a GEO satellite results in a roundtrip delay of 500 msec.
Delay jitter (=variation in the arrival time between different voice samples) should be kept very low (typically below a few msec) because otherwise the voice quality is degrading very rapidly. In general this is not a real problem in telephone networks. Note that larger jitter is allowed when dejitter control is available (e.g. dejitter buffer)
Blocking probability (due to network congestion) is in general very low.

Dia 4

Today's telephone networks are fully digital inside the network and only the access to the network (from the home telephone to the local telephone switch) is still in most cases analog.
We consider an analog telephone signal which is converted to a digital signal : the analog voice telephone signal (300 - 3400 Hz) will be converted to a digital signal of 64000 bits/s.
We will first sample the signal in time. One has to sample the signal with a frequency which is at least two times the maximum frequency of the signal. For telephony this maximum frequency is 3400 Hz so we have to sample at least at 6800 Hz. In practice one is using a sampling frequency of 8000 Hz (or we take a sample every 125 µs). The next step is to digitize the amplitude of the signal. In telephony one is using 8 bits (or 256 levels) because this is enough to obtain good voice quality. As a result one is generating 8 bits of information every 125 µs. This corresponds to 64000 bits every second : 64 kbit/s (= 8kHz x 8 bit).

Dia 5

An important question still remains : how does the switching element know when it has to close ? Each switch will use a computer (a controller) in order to close the appropriate switching elements (interface between user plane and control plane) but also in order to talk to other controllers and to the end user.

The use of the switch controllers and the signaling between a user and a switch controller or between the different switch controllers results in a kind of overlay network. This overlay network is not carrying the information the users want to exchange (=voice signal) but carries signaling messages required to control the network (to set up or tear down a call). Otherwise stated : during the call, the voice signal travels through the switch hardware, without entering the switch controller.
This overlay network is commonly described as the control plane and the network carrying the voice information is called the user plane (or data plane). This is more clearly shown in the figure.
One can observe two types of interfaces in the control plane : one internal interface (NNI or **Network Node Interface** or **Network Network Interface**, between the switch controllers) and one external interface (UNI or **User Network Interface**,between the user terminal and the Local Exchange or LEX switch controller).
One of the important goals of the control plane is to support the call set-up. This call set-up will make use of signaling (both at the UNI and NNI interfaces) and routing. It will distribute the necessary information to know the switching elements that should be closed in order to set-up a phone call.
Note : as indicated in the animation, a one-way speech channel in the upstream direction is gradually build-up. This is important to carry the ringing tone from the called party to the caller (in-band signaling).

Dia 6

This figure gives a summary of the user plane operation (fully digital network).
During a phone call, a telephone is generating samples of 8 bits every 0.125 µsec. This continues until the call has ended. These voice samples will be transmitted over the twisted pair access network towards the local exchange (LEX). In the LEX, a number of these samples from different phones will be multiplexed (in our example multiplexing in a 4 timeslot frame). In the transit exchange, these calls are switched. We observe that each frame will transport one voice sample from a sending telephone to a receiving telephone (and of course also the other way around, not shown in the figure). The same timeslots will be reused to transport different phone calls (not overlapping in time). Phone calls that overlap in time and use the same line systems will use different timeslots. If no timeslots are available anymore one observes a call blocking.

Dia 7

In order to better understand the control of the network, it is interesting to analyse a call set-up procedure. The different call set-up steps are :
The caller lifts the handset (off-hook) which alerts the exchange that he wants to place a call.
The exchange A will identify the subscriber and prepare itself to receive the digits.
The exchange A will then provide a dial tone to the caller.
Now the caller can dial the digits of the phone number he wants to reach. This is done using DTMF (Dual Tone Multi Frequency) signaling.
The exchange A will wait till all the digits are dialed and then look up its routing information to see if the call is local (we consider here a non-local call via two exchanges).
The exchange A will send a seize signal to the exchange B and sends the digits (after receiving a "proceed-to-send" signal).

The exchange B will now test if the called phone is busy. If not, the exchange B will send the ringing current to the called subscriber and the ringing tone to the caller (via exchange A).
If the called subscriber picks up the phone, the conversation can start.
The connection will be released if the call is finished.

Dia 8

The control plane is an overlay network that is typically using links of 64 kbit/s (or multiples of 64 kbit/s). This network consists of **Signaling Points** or SPs interconnected by **Signaling Links** or SL's (over which the signaling messages can be sent). There are three types of SP's :
Service Switching Point (SSP) where signaling messages can originate or terminate, typically in the local telephone switches or local exchanges
Signaling Transfer Points (STP) where signaling messages can be routed (note that STP and SSP may be combined at a single exchange)
Signaling Control Points (SCP) where intelligence is provided how to handle the calls (e.g. for local number portability, LNP).
This signaling network is the **SS7 network** (Signaling Systems No 7, standardized by the International Telecommunication Union or ITU).
The signaling network is message based (packet based). The signaling link layout should not correspond to the physical links (e.g. on the figure there are less signaling links than physical links). Normally a switch will have a service switch point (SSP), there will be fewer signaling transfer points (STP).
A good overview is provided at : www.pt.com/tutorials/ss7/index.html

Dia 9

An example of the transfer of information in the control plane is shown in the figure. The information to be transferred (e.g. telephone number) is put in messages which are transported in 64 kbit/s channels. The STP's have routing tables which will forward the message to the correct SL (signaling link). *Note that the messages may follow completely different routes than the final call route!*
Typical parts in the message are :
error correction information
S/D : originating point code and destination point code ("source" and "destination"). This is information used to route the message in the control plane
address : information on the call ("telephone number").

Dia 10

Number portability is the possibility to keep your telephone number when physically moving or when changing operator. In the example we show the operation of LNP (Local Number Portability) when moving to a new location. Note that only the principle is explained, the exact details are much more complex and out of scope for this course.
In the example, telephone 09-264 3333 is calling 02-272 4444.
First consider the case where no number portability is used. In this case a simple routing will happen based on the called telephone number. The message in SS7 will use as source 09-264 (the number of the originating LEX) and as destination 02-272 (the number of the destination LEX). The intermediate STP's will have routing tables that will forward the message to the correct destination. Inside the message, the called phone number will be transported (02-272 4444).

Consider now the case where LNP is used. In this case we need a LNP server (typically inside a SCP or Signaling Control Point) and also the routing tables in the exchanges will be adapted. When the calling party (09-264 3333) will transfer to its LEX the destination phone number (using DTMF signaling), the LEX will observe in its local routing data base that the number is a ported number (actually it will be indicated that the whole range 02-272 xxxx is ported, although maybe only one number was ported). As a result an SS7 message will be routed to an intelligent server (LNP server) where it is known that the number 02-272 4444 is ported to the exchange 02-273. The message in SS7 will use as source 09-264 (the number of the originating LEX) and as destination 02-273 (the number of the new destination LEX). The intermediate STP's will have routing tables that will forward the message to the correct destination (02-273 !). Inside the message, the originally called phone number will be transported (02-272 4444). The LEX 02-273 will connect to the phone 02-272 444 (ringing, etc.)

Dia 12

When considering the use of internet for the support of telephony, one often refers to Voice over IP (VoIP).
The goal of VoIP is to transmit a voice signal over internet with an acceptable quality. As it is clear from the previous paragraph on PSTN, there are many more functions (control and management) that should be supported in order to have a fully operational telephone service on an internet.
Some of the major problems are :
Obtain good voice quality (taking into account the specific problems encountered on internet, such as : delay, jitter, packet loss, connectionless network). This will require the development of a QoS (Quality of Service) enabled internet.
Use the bandwidth as efficient as possible in order to allow as many users as possible on the network and in order to support also the traditional internet services (such as e-mail, FTP, web access, …). This will require the use of advanced voice coders and decoders.
Provide a number of specific control functions in the network (call set-up, resolution between phone number and IP address, routing, etc.). This will require a number of servers supporting specific control services.
Provide interworking with the traditional PSTN network (both at user and control plane level). This will require gateways in order to obtain interworking at control and user plane.
The study of Voice over IP will be an ideal vehicle to introduce a number of basic concepts and technologies used in internet to support multimeDia services.

Dia 13

The goal of Voice over IP is to transport a voice telephone signal over a packet switched network. This will require a number of functions shown on the figure :
Phone connected to internet : a coder (e.g. G.729) will generate a stream of voice samples that will be transported in IP packets. At the receiving end the opposite conversion will be done using a decoder. Because a coder and decoder are always used together, one talks about a codec (CODEC = COder/DECoder).
Phone connected to PSTN : the voice samples (from G.711 codec) will be transported in a circuit (64 kbit/s).
A gateway will be provided in order to support phone calls between users in a PSTN and internet. This gateway should support transcoding (e.g. between G.711 in PSTN and G.729 in

internet), conversion between time-slots and packets, address translation (telephone number <> IP address), signaling translation (PSTN SS7 <> IP SIP), exchange of billing information, etc.
Note : G.711, G.729, … are codecs that are standardized (the number refers to the standard from ITU)

Dia 14

Some basic building blocks of a VoIP network are :
Terminals : Several options are possible : PC with VoIP software and headset or loudspeaker/microphone, PC with VoIP software and interface card to normal telephone, meDia gateway connected to a normal phone, IP phone (which can be directly connected to internet with e.g. an Ethernet interface).
MeDia Gateway (MG) : gateway between PSTN and internet for the user plane information (= the voice signal). Is also used at the terminal side.
Signaling Gateway (SG) : responsible for the interworking of the signaling systems in PSTN and internet.
MeDia Gateway Controller (MGC) : responsible for the set-up of the connection inside internet. This gateway will know the addresses of the different phone subscribers, it will coordinate the call set-up , etc.
Gatekeeper (GK) : additional functionality, for example user authentication and access control.
Billing Server (BS) : server to support billing of the customers.
QoS internet : extension to the current internet protocols in order to provide the necessary quality of service for voice (e.g. low delay, low jitter, low packet loss).
Note : This architecture corresponds to the MEGACO standard (MeDia Gateway Control protocol). This is both an IETF (RFC2885) and an ITU (H.248) standard (both date from the year 2000). Later in this chapter we will use SIP terminology.

Dia 15

The figure illustrates a first example : a simplified call set-up scenario where users in a PSTN network and an IP network are setting up a phone call.
A user in the PSTN (PSTN-user) wants to call a user in internet (IP-user). The PSTN-user will send the phone number of the IP-user to his IN-server (IN = Intelligent Network). This IN-server knows that the person he wants to reach is an internet user and he will forward the request to the SG (Signaling Gateway). This SG will eventually check if the PSTN user is allowed on the IP network and will forward the request to the MGC (MeDia Gateway Controller). This MGC will contact the IP-user and his phone will ring. When he picks up the phone, the end-to-end connection can be set up (signals are exchanged in the other direction, not shown on the figure).
The identification of the two interfaces at the MG is indicated : at the PSTN side this is specified with a physical interface number, a timeslot number (in the TDM frame) and a codec type (G.711); at the IP side one observes a UDP port number, an IP address and a codec number (G.723). The MG will map the timeslot information (from PSTN side) in the packets (at IP side) and vice-versa and this after transcoding between the G.711 and G.723 codecs.

Dia 16

A second example of the use of VoIP is in an intranet environment. A company is using its intranet (=internal internet) to support both data and telephony traffic. This has important advantages in terms of infrastructure, maintenance, investment, etc. (especially in green-field situations where new infrastructure has to be deployed).
In order to connect to the outside world, a gateway is used which will direct all data traffic to the public Internet and the voice traffic to the PSTN (no VoIP in the public Internet).
This is a typical example of today's usage of VoIP (e.g. the research group IBCN is using VoIP in the Zuiderpoort building).

Dia 17

A third example is using VoIP in the core of the public telephone network in order to (partially) replace the transit exchanges. Today we observe many incumbent telephony operators also offering IP services to their customers. In order to support these services, they also deploy a backbone IP network. They are considering to use this backbone IP network also for the support of telephony traffic using VoIP. In this case the VoIP is only used between normal telephone exchanges and no customers have direct access to it. In this way they are able to improve bandwidth usage and to further integrate their network, probably ending up with a single core network technology based on IP.
Note that there are other possible scenarios for the use of VoIP. One example are ITSP’s or Internet Telephony Service Providers. These are telephony service providers who role out a complete internet infrastructure, only for the support of voice traffic. They give direct access to their network for customers (e.g. business customers).

Dia 19

This part will focus on an example architecture for the control of a VoIP network. The figure illustrates the protocols used for the control of the VoIP network and the protocols used for the voice transport itself (control plane and user plane)
At the top of the control protocol stack, SDP is used (Session Description Protocol). This will specify the session (for VoIP we talk about a call) parameters (it is not really a protocol but a specification template). Below SDP the Session Initiation Protocol (SIP) will be used to set up the call (using the SDP to specify the call parameters). SIP is running on the classical TCP/UDP, IP, data link, physical layer protocol stack.
The voice protocol stack will use RTP/RTCP to support the transport of voice samples over internet. UDP is the preferred transport layer (but TCP is used as well). In order to obtain the necessary quality for the voice signal, measures have to be taken in the network itself. Special QoS enabling technologies/protocols will be used (not shown here).

Dia 20

First some background : the Mbone is the part of the Internet that supports IP multicast (see later), and thus permits efficient many-to-many communication. It is used extensively for multimeDia conferencing. Such conferences usually have the property that tight coordination of conference membership is not necessary; to receive a conference, a user at an Mbone site only has to know the conference’s multicast group address and the UDP ports for the conference data streams. Session directories assist the advertisement of conference sessions and communicate the relevant conference setup information to prospective participants. SDP is designed to convey such information to recipients. SDP is purely a format for session

description – it does not incorporate a transport protocol, and is intended to use different transport protocols as appropriate, including the Session Initiation Protocol or SIP, the Real Time Streaming Protocol (RTSP), electronic mail using the MIME extensions, and the Hypertext Transport Protocol (HTTP). SDP is intended to be general purpose so that it can be used for a wider range of network environments and applications than just multicast session directories.
The Session Description Protocol (SDP) is used to describe various parameters of a session. A session is very general (e.g. multimeDia conference), but in this chapter it is used for a point-to-point voice over IP session (a VoIP call).

Dia 21

Different parameter categories used for VoIP are indicated by a single character (c, b, m, a):
The connection information (indicated by **c**) is indicating the type of network (IP), the address format (IPv4 or IPv6), the destination address (unicast or multicast).
The required amount of bandwidth (indicated by **b**), expressed in kbit/s.
The meDia information (indicated by **m**) :
whether it is audio, video, application (e.g. whiteboard, i.e. application data presented to the user), data (e.g. executable data not presented to the user) or control (e.g. an additional conference control channel for a session) information.
the port number (e.g. UDP port number in the range 1024-65535 with even numbers for RTP sessions and the next higher odd number for RTCP).
the transport protocol to be used for the meDia stream (UDP or RTP/AVP over UDP)
the meDia format (e.g. voice codec, video codec, …)
Additional information (indicated by **a**) : **ptime** (packet time in msec)
In this chapter on VoIP, the SDP description of a session will be transported in a SIP message.
Note that SDP may be used both for indicating the capabilities of one endpoint (offer) and the matching of the other endpoint (response). The caller may indicate for example 3 voice codecs that may be used, whereas the called party will respond with a single choice in a response SDP description.

Dia 22

This is a more general example of SDP (typically used for a multimeDia conference). The details of the different parameters can be found in RFC 2327 : Session Description Protocol (April 1998).
The example is describing the parameters used in a seminar on SDP that will be given by M. Handley. More information can be found at www.cs.ucl.ac.uk/staff/M.Handley/SDP.03.ps and for further information one may contact M. Handley at mjh@isi.edu. The seminar will use multicast address 224.2.1.1 (and the scope is limited to a TTL of 127, meaning that the multicast packets will be discarded after 127 hops). It is also indicated that the participants to the seminar should work in the receive only mode (it is not an interactive seminar). There are two audio codecs possible (PCM μ-law and GSM), there is one video stream possible (H261) and there is a white board application.
A number of the parameters indicated in this example are not appropriate for VoIP.
[NTP : Network Time Protocol : a protocol used to synchronize computers on the internet, RFC 1305]

Dia 24

There are many applications of the Internet that require the creation and management of a session, where a session is considered an exchange of data between an association of participants. The implementation of these applications is complicated by the practices of participants: users may move between endpoints, they may be addressable by multiple names, and they may communicate in several different meDia - sometimes simultaneously. Numerous protocols have been authored that carry various forms of real-time multimeDia session data such as voice, video, or text messages. The Session Initiation Protocol (SIP) works in concert with these protocols by enabling Internet endpoints (called user agents) to discover one another and to agree on a characterization of a session they would like to share. For locating prospective session participants, and for other functions, SIP enables the creation of an infrastructure of network hosts (called proxy servers) to which user agents can send registrations, invitations to sessions, and other requests. SIP is an agile, general-purpose tool for creating, modifying, and terminating sessions that works independently of underlying transport protocols and without dependency on the type of session that is being established.
SIP is an application-layer control protocol that can establish, modify, and terminate multimeDia sessions (conferences) such as Internet telephony calls. SIP can also invite participants to already existing sessions, such as multicast conferences. MeDia can be added to (and removed from) an existing session. SIP transparently supports name mapping and redirection services, which supports personal mobility - users can maintain a single externally visible identifier regardless of their network location.
SIP supports five facets of establishing and terminating multimeDia communications
1. User location: determination of the end system to be used for communication
2. User availability: determination of the willingness of the called party to engage in communications

Dia 25

3. User capabilities: determination of the meDia and meDia parameters to be used
4. Session setup: "ringing", establishment of session parameters at both called and calling party
5. Session management: including transfer and termination of sessions, modifying session parameters, and invoking services

SIP is not a vertically integrated communications system. SIP is rather a component that can be used with other IETF protocols to build a complete multimeDia architecture. Typically, these architectures will include protocols such as the Real-time Transport Protocol (RTP) (RFC 1889) for transporting real-time data and providing QoS feedback, the Real-Time streaming protocol (RTSP) (RFC 2326) for controlling delivery of streaming media, the MeDia Gateway Control Protocol (MEGACO) (RFC 3015) for controlling gateways to the Public Switched Telephone Network (PSTN), and the Session Description Protocol (SDP) (RFC 2327) for describing multimeDia sessions. Therefore, SIP should be used in conjunction with other protocols in order to provide complete services to the users. However, the basic functionality and operation of SIP does not depend on any of these protocols.
SIP will support the set-up phase of a call, the termination of a call and the changes during a call. Therefore it is using a number of request and response messages. Examples are INVITE, OPTIONS, STATUS, ACK, CANCEL, REGISTER.
We will first consider a very simple example : two users (running a user agent on their terminal) are connected to an IP network and want to set-up a call (Bob will set-up a call to

Carol). In the network there is a SIP server to support the call set-up (that SIP server may run several services, as explained next).
Note : Very interesting RFC's : SIP protocol in RFC2543, examples in RFC3665

Dia 26

A simplified message flow is illustrated in the figure. First Carol will REGISTER herself with the registrar server (1). This registrar will store the information in a location service database (2). When Bob wants to call Carol, he will send an INVITE to his proxy server (indicating that he wants to set-up a call with Carol) (3). Bob knows the location of this proxy server (e.g. he knows the IP address or he knows the name "sip.chicago.com" that can be resolved by DNS). The proxy server will consult the location service database (4)(5) to know the location of Carol and will finally invite Carol. This will allow Bob and Carol to set-up the VoIP phone call.
In general the registrar, proxy and location service database are implemented on a single server (but this is not required).

Dia 27

Server : A server is a network element that receives requests in order to service them and sends back responses to those requests. Examples of servers are proxies, user agent servers, redirect servers, and registrars.
User Agent (UA) :A logical entity that can act as both a user agent client and user agent server
User Agent Client (UAC): logical entity that creates and sends a new request. The role of UAC lasts only for the duration of that transaction. In other words, if a piece of software initiates a request, it acts as a UAC for the duration of that transaction. If it receives a request later, it assumes the role of a user agent server for the processing of that transaction.
User Agent Server (UAS): A user agent server is a logical entity that generates a response to a SIP request. The response accepts, rejects, or redirects the request.
Proxy server: An intermediary entity that acts as both a server and a client for the purpose of making requests on behalf of other clients. A proxy server primarily plays the role of routing, which means its job is to ensure that a request is sent to another entity "closer" to the targeted user. Proxies are also useful for enforcing policy (for example, making sure a user is allowed to make a call). A proxy interprets, and, if necessary, rewrites specific parts of a request message before forwarding it.
Redirect server : A redirect server is a user agent server that generates 3xx responses [*Redirection : further action required to complete the request*] to requests it receives, directing the client to contact an alternate set of URI
Registrar: A registrar is a server that accepts REGISTER requests and places the information it receives into the location service for the domain it handles. It is similar to a Home Location Register (HLR) in GSM.
A **location service** is used by a SIP redirect or proxy server to obtain information about called party's possible location(s). It contains a list of bindings of address-of-record keys to zero or more contact addresses. The bindings can be created and removed in many ways; SIP defines a REGISTER method that updates the bindings. Note that contact addresses may have to be resolved using DNS. Note also that the protocol used between location service and proxy/redirect/registrar is not specified by SIP.

Dia 28

SIP will support the set-up phase of a call, the termination of a call and the changes during a call. Therefore it is using a number of request and response messages. Examples are INVITE, OPTIONS, STATUS, ACK, CANCEL, REGISTER. SIP is very similar to the HTTP request/response model and uses the same format for the messages.
The figure illustrates a more complex example of the SIP building blocks. Alice and Bob are in a different domain (atlanta.com and boloxi.com) with their own servers (proxy/redirect, registrar, location service). Note that in practice even more servers will be involved.
The next slides will illustrate the operation of the servers in proxy or redirect mode.
Note : SIP may use TCP or UDP to transport messages. Along an end-to-end path, a mixture of UDP and TCP may be used (SIP has built-in reliability mechanisms). The protocol is indicated in the Via header. Example: TCP between Alice and atlanta.com and between atlanta.com and biloxi.com; UDP between biloxi.com and Bob. Also other transport protocols are possible, e.g. TLS over TCP. It is recommended that a server listens to the default SIP ports (5060 for TCP and UDP, 5061 for TLS over TCP).
Note : TLS (Transport Layer Security) : The primary goal of the TLS Protocol is to provide privacy and data integrity between two communicating applications (very similar to SSL). Dia 29
This first example show the registration and proxy operation of the SIP server. We have a user Alice in the atlanta.com "domain" and a second user Bob in the biloxi.com "domain". They have IP numbers 10.1.3.3 and 10.4.1.4 respectively (private IP numbers used for simplicity). The servers have IP numbers 10.1.1.1 (atlanta.com) and 10.2.1.1 (biloxi.com).
Before Alice can do anything, she has to configure here PC with the IP address of the SIP server where she wants to subscribe (10.1.1.1). This is similar to the IP number of the DNS server you want to use from your PC. The same holds for Bob (using the SIP server at 10.2.1.1). Note that a name of the SIP server is also possible (this name will be resolved by DNS).
The first action Alice and Bob will take is to register themselves to their respective SIP servers (using the **`REGISTER`** message). They will have a specific SIP URI (as indicated by **`sip:alice@atlanta.com`** and **`sip:bob@biloxi.com`**).
Now Alice can set up a call with Bob. In order to do this, she will send an **`INVITE`** message with all necessary information to set up the call. This will be sent to the SIP server of Alice (at 10.1.1.1). This SIP server will then find out where the SIP server for the biloxi.com domain (using a normal DNS) is. It will forward the **`INVITE`** message to the SIP server of biloxi.com (at 10.2.1.1). This SIP server will contact Bob because he knows that the SIP address of bob (**`sip:bob@biloxi.com`**) corresponds to the IP address 10.4.1.4. After sending some response messages, the communication will start. Note that it will no longer make use of the SIP servers (the User Agents of Alice and Bob will exchange information directly).
Note that the operation of the proxies is similar to recursive DNS where each DNS server is involved in a chain of requests. The records for SIP servers in a DNS are indicated by SRV (similar to MX for mail servers), see also RFC 2782, 2915 and 3263.

Dia 30

SIP proxies are elements that route SIP requests to user agent servers and SIP responses to user agent clients. A request may traverse several proxies on its way to its final destination. Each will make routing decisions, modifying the request before forwarding it to the next

element. Responses will route through the same set of proxies traversed by the request in the reverse order.
Being a proxy is a logical role for a SIP element. When a request arrives, an element that can play the role of a proxy first decides if it needs to respond to the request on its own. For instance, the request may be malformed or the element may need credentials from the client before acting as a proxy. The element may respond with any appropriate error code. When responding directly to a request, the element is playing the role of a UAS.
A proxy can operate in either a stateful or stateless mode for each new request. When stateless, a proxy acts as a simple forwarding element. It forwards each request downstream to a single element determined by making a targeting and routing decision based on the request. It simply forwards every response it receives upstream. A stateless proxy discards information about a message once the message has been forwarded. A stateful proxy remembers information (specifically, transaction state) about each incoming request and any requests it sends as a result of processing the incoming request. It uses this information to affect the processing of future messages associated with that request. A stateful proxy may choose to "fork“ a request, routing it to multiple destinations. Any request that is forwarded to more than one location must be handled statefully.
Important note : a SIP proxy will never be stateful with respect to a complete session (or call).

Dia 31

In some architectures it may be desirable to reduce the processing load on proxy servers that are responsible for routing requests, and improve signaling path robustness, by relying on redirection. Redirection allows servers to push routing information for a request back in a response to the client, thereby taking themselves out of the loop of further messaging for this transaction while still aiding in locating the target of the request. When the originator of the request receives the redirection, it will send a new request based on the URI(s) it has received. By propagating URIs from the core of the network to its edges, redirection allows for considerable network scalability. This is similar to the iterative operation of a DNS server.
The example shows the operation of the SIP server in redirect mode. The requests are always directly answered to the user agent client (the server does not take the initiative to contact the next server).

Dia 32

The layout of a request message is standardised and is shown in the figure. This is the same format as used in HTTP. The message body may be SDP but arbitrary MIME headers may be included : JPEG picture (e.g. a photo of the caller), ISUP (ISDN User Part), charging info, …
The request line contains the method (REGISTER, INVITE, ACK, CANCEL, BYE, OPTION, etc.), the URI (Uniform Resource Identifier) and the version of the SIP protocol.

Dia 33

The request messages will first specify the method. Some examples are :
REGISTER : used when a user agent is registering to a SIP server
INVITE : used when a user agent client is setting up a call in order to invite the other party
ACK : used as part of a three way handshake protocol (**`INVITE/200/ACK`**, see later example)
CANCEL : used when a certain transaction has to be cancelled (by the client).
BYE : used when the call is finished

OPTION : used to negotiate the options of the user agents
...
A URI (Uniform Resource Identifier) will specify the destination for the message. Examples : for an INVITE this will be the destination User Agent, for a REGISTER this will be the responsible SIP server. Note that a URI may also contain a phone number.
A number of header lines are possible. Some important examples are listed above, but there are much more headers possible.
A body is optional (it may contain e.g. SDP information).

Dia 34

Two examples of SIP request messages are shown.
Below we observe a **`REGISTER`** message from Bob to the **`sip:register.biloxi.com`** server. The **`To:`** and **`From:`** fields are both the user agent client (**`sip:bob@biloxi.com`**). There is a call identification and a sequence number. The contact is **`sip:bob@10.4.1.4`** and the validity is 7200 minutes. No further information is included.
On the slide (above) an **`INVITE`** message is illustrated. In this example we observe a **`Via:`** line which indicates where the response to the request message has to be sent (IP address 10.1.3.3 at UDP port 3456). The content type is SDP and the length is 142 bytes.

```
REGISTER sip:registrar.biloxi.com SIP/2.0
Via: SIP/2.0/UDP 10.4.1.4:5060
To: Bob <sip:bob@biloxi.com>
From: Bob <sip:bob@biloxi.com>;tag=456248
Call-ID: 843817637684230@phone21.boxesbybob.com
CSeq: 1826 REGISTER
Contact: <sip:bob@10.4.1.4>
Expires: 7200
Contact-Length: 0
```

Dia 35

A response message looks similar to a request message but different fields are used. The response line starts with the SIP version and then it gives some status information (code + phrase explaining the code). Next again a number of header lines are possible. The last part of the response message contains the requested information.

Dia 36

Status code classes :
`1xx:` Informational : request received, continuing to process the request. [100 Trying, 180 Ringing, 181 Call is Being Forwarded]
`2xx:` Success : the action was successfully received, understood, and accepted.[200 OK]
`3xx:` Redirection : further action required to complete the request [300 Multiple Choices, 301 Moved Permanently, 302 Moved Temporarily]
`4xx:` Client error : the request contains bad syntax or cannot be fulfilled at this server. [400 Bad Request, 401 Unauthorized, 482 Loop Detected, 486 Busy Here]
`5xx:` Server error : the server failed to fulfil an apparently correct request. [500 Server Internal Error]
`6xx:` Global failure : the request cannot be fulfilled at any server. [Busy Everywhere]

Dia 37

The figure illustrates an example of a call set-up when using a SIP proxy server. Alice will phone Bob and she is inviting him by sending an **`INVITE`** request message to her SIP server at atlanta.com. The atlanta.com server will forward the request to the server of Bob and at the same time it will send a response back to Alice (**`100 Trying`**). A similar action is taken by server biloxi.com. The **`INVITE`** from biloxi.com to Bob will result in Bob's phone starting to ring and at the same time a **`180 Ringing`** response message will be sent back to Alice (via the two SIP servers). When Bob picks up the phone a **`200 OK`** message will be sent back to Alice (via the two SIP servers) and Alice will send an **`ACK`** to Bob directly (from now on the SIP servers are no longer required). At that moment the conversation may start (RTP MeDia Session). The session will be stopped by Bob (**`BYE`**). Some examples of the response messages are shown in the next slide.

Dia 38

A few examples of response messages are illustrated. Below we observe the response (**`100 Trying`**) from the SIP server atlanta.com to Alice. The header field is nearly the same as in the **`INVITE`** message. The message above (middle layer, *only visible via animation*) is the response from the biloxi.com SIP server to the atlanta.com SIP server. Note that there is an extra **`Via:`** line indicating that the message has to go to the atlanta.com server. The message above (top, *visible*) illustrates the OK message (from Bob). We observe the 3 Via: lines indicating the way back to Alice. In this OK message, Bob's SDP is also added.

```
SIP/2.0 100 Trying

Via: SIP/2.0/UDP 10.1.3.3:3456

To: Bob <sip:bob@biloxi.com>

From: Alice <sip:alice@atlanta.com>;tag=1928301774

Call-ID: a84b4c76e66710@10.1.3.3

CSeq: 314159 INVITE

Contact-Length: 0
```

Dia 39

This REGISTER example illustrates the support of mobility. Bob registers a number of URI's where he may be reached. He will be first contacted at his personal SIP phone. If he is not responding, one will try his phone at work and next his mobile. If none of these respond, the call will be forward to his voicemail.

It is possible to specify in a request the way one wants to contact Bob (not sequential). The use of

Accept-Contact:*;feature=voicemail
in an INVITE will direct the call directly to Bobs voice-mail.

Note that a user may also update his location service database every time he gets to a new location (each time using the REGISTER method).

Important note : SIP will also be used in the future for the signaling in the core network of the third generation mobile networks (3GPP : Third Generation Partnership Project, developing 3th generation networks). The system proposed is called IMS (IP MultimeDia core network Subsystem).

Dia 41

In the terminals we have several methods and protocols to support voice over IP. When transmitting voice over internet, one will compress the voice signal (in a codec) in order to reduce the necessary bandwidth, but also other treatments will be done in order to support the transfer of the voice signal with acceptable quality. Examples are : add redundant information for packet loss recovery (e.g. Forward Error Correction), insert timing information (e.g. using RTP), etc.
Also the network will take specific actions to provide QoS (Quality of Service) e.g. to reduce delay or packet loss. Both the routers (e.g. buffer management) as the control of the network (e.g. Differentiated Services) need to be adapted.

Dia 42

Let us first have a look at how the analog signal is transferred over an internet based network. At the sender side the analog voice signal will be converted to a digital signal (sampling in time and amplitude in an A/D converter) and eventually a compression algorithm will be used to reduce the bitrate. In order to reduce the influence of packet loss in the network, error correcting measures may be taken at the sender side. Examples are Forward Error Correction (FEC), interleaving and the addition of sequence numbers. The addition of timing information (timestamp) may be useful in order to eliminate jitter (=delay variation) or to detect silent periods in the conversation. This will result in a digitally coded signal that will be transported over the UDP/IP protocol stack.
The receiver will use a dejitter buffer in order to reduce the influence of the jitter in the network (possibly also using the timestamp information). Error recovery will be based on the FEC information from the sender possibly combined with the interleaving. The influence of lost packets may be reduced by the use of packet concealment : a missing packet may be replaced by e.g. noise or an interpolation. Decompression and D/A conversion will generate the voice signal, which will be a degraded version of the original voice signal.

Dia 43

The major contributor to packet loss is the router in the network. The basic functionality of a router is :
Space switching to transfer a packet from a certain input port (interface) to a certain output port (interface).
Buffering (queuing) for contention resolution : multiple packets arriving at the same time may contend for the same output line. Buffering is also required to provide sufficient time for routing table look-up (longest prefix match).
Control will be responsible for maintaining the routing tables (e.g. use of OSPF routing protocol), controlling the space switch, scheduling the buffers, etc.
The figure illustrates several packets arriving on different input ports, contending for the same (middle) output port. This will result in a delay for certain packets.When the buffer gets full, overflow will occur resulting in packet loss.

Another reason for packet loss is due to transmission errors (not shown on the figure). Transmission errors will result in bit errors (or bursts of bits) but this will result in packet loss (wrong bits will be detected by the checksums used in TCP/UDP and IP headers).
Note : this is a simplified scheme (e.g. when a packet arrives, some time will be required to know the outlet port by using the routing table and doing a longest prefix match)

Dia 44

Routers will also be responsible for the delay variations (jitter) observed when crossing a network. Because the buffers have to resolve the contention, it may happen that packets from the same traffic flow may have to wait more or less in the buffers (depending on the other traffic contending for the same output link).

Dia 45

The delay variation in a router will result in end-to-end delay variations and even out of order arrival becomes possible. This is illustrated in the figure : 6 packets from the same traffic flow are transmitted at regular time intervals. Due to delay variations in a router, they will arrive with a time difference between the packets which will vary (this is the jitter). Eventually packets may arrive out of order.

Dia 46

Another problem is the delay in the network. This delay will impact the influence of echo (if the delay is less than a few 10 msecs, it has no impact in the degradation quality) and will reduce the interactivity of the conversation (if the delay is a few 100 msec, interactivity is still acceptable but starts to degrade).
A first important delay component is the transmission delay. This is the time it takes to transmit a packet over a transmission line with a certain bandwidth. The example shows that it takes 125 msec to transmit a packet of 1 kbyte over an access line of 64 kbit/s (e.g. ISDN line) : this is the time between the first bit and the last bit sent. If one would use an access line of 5 Mbit/s (e.g. possible with a cable modem), the transmission delay would reduce to 1.6 msec.
A second important delay component is the propagation delay, limited by the propagation speed of electromagnetic waves : the speed of light is about 3.10^8 msec in air and 2.10^8 in optical fiber. This component becomes important for transmission over long distances. The example shows a transatlantic connection via a Geo Stationary Satellite (GEO) which is at a distance of about 35000 kilometer from earth. This results in a propagation delay of about 250 msec (from earth to satellite and back to earth). The way to reduce this delay is to use shorter distances. In the case of the use of satellite links, one can replace them eventually by fiber links (e.g. transatlantic cable), resulting in a shorter propagation distance and therefor a lower delay value (e.g. 10.000 km of fiber results in 50 msec delay).

Dia 47

The graph shows the packet loss measured from Gent towards the rest of the world. A drastic improvement is observed over the years.

Dia 48

A similar graph is shown for the roundtrip delay.

Dia 50

The figure shows the general operation of a voice codec. The analog voice signal will be split in voice samples (duration T_V) and each of these voice samples will be coded. In some cases one will also look ahead in order to code the voice sample. This look-ahead time is indicated by T_{LA}. It is only after the T_V and T_{LA} that the coding can start (at that time all the information is available to generate a voice packet). The time required for the encoding is T_{enc}. An IP packet may be constructed of a number of voice packets (in the example one is using 3 voice packets in one IP packet).
The example illustrates how long it takes to construct an IP packet for a G.729A codec. The voice samples are 10 msec long and the look-ahead time is 5 msec. If we group 6 voice packets in an IP packet, this results in a packetization time (delay) of 65 msec (note that this takes into account one look-ahead time).

Dia 51

The table shows a number of codecs with their specific characteristics. The two lower ones are used in GSM (FR = Full Rate and HR = Half Rate).

Dia 52

The type of codec and the network behavior have a profound impact on the voice quality perceived. The characterization of perceived voice quality is however very difficult because it is a subjective matter (note that this is even much more complex for video perceived quality).
One has defined a measure to represent the voice quality : MOS or Mean Opinion Score. This will be based on the opinion of a heterogeneous public listening to a voice stream under well controlled conditions (e.g. noise in the room). From the individual scores (ranging from 0 to 5), an average will be calculated giving a measure for the voice quality.
Note that MOS measurements are very difficult (special room environment is required and very good audio equipment) and time consuming (audience to be invited, tests themselves take also some time, …).

Dia 53

This slide shows the MOS values for different codecs when no network impairment is present (back-to-back measurement : no network delay, jitter or packet loss). A standard telephone call over the PSTN network results in a MOS value of 3.8. The difference in quality is thus caused by the compression used (the compression is not lossless).
The digital PSTN (ISDN or trunks) makes use of the G.711 codec, so normally (G.711 = 4.3 in the graph) a higher MOS Score is expected than 3.8 (because there is no degradation due to the network).
Ilbc is described in IETF RFC 3951.Dia 54
If this graph is compared with the ‘Codec MOS scores’, one can see that some codecs (e.g. G.729, which is commonly used) can combine a low bitrate with a high MOS score.

Dia 55

A typical VoIP quality measurement set-up is shown in the figure. A number of measurements will be illustrated in the next slides, illustrating the problems encountered when sending voice over data networks or the Internet.
A VoIP analyzer will generate analog voice streams. From this reference signal, a voice terminal will generate digital voice streams using different codecs. The packets will be

transmitted over a network (sometimes emulated by using an impairment node*). At the receiving side the digital voice stream will be decoded and the analog (degraded) voice signal will go to the analyzer. The VoIP analyzer will generate the MOS value by comparing the reference and the recorded degraded signal.
The algorithms based on psycho-acoustic used for quality analysis are:
PESQ = Perceptual Evaluation of Speech Quality (ITU-T P.862)
PSQM = Perceptual Speech Quality Measure (ITU-T P.861)
PEAQ = Perceptual Evaluation for Audio quality (ITU-R rec. BS.1387)
* An impairment node will introduce artificial packet loss, delay and delay jitter in a controlled way.

Dia 56

We notice that software (Skype, SJPhone) in general has higher delays than a hardware solution (Siemens IP Phone). This is due to the buffering on several levels (soundcard, audio drivers, dejitter buffer, …).

Dia 57

From 0-1 % packet loss we see a variable MOS score for different codecs (some codecs are more robust), but from 1% on, the MOS scores converge.

Dia 58

Burst loss: More realistic than uniform packet loss when considering congestion behaviour in networks/the Internet.
A number of consecutive packets are dropped in a burst.
When the burst loss occurs in a silent part of the sample, there is no degradation of the MOS Score

Dia 59

A number of video codecs are listed above. From the bitrates, it is clear that video will result in a much higher load on the network. It is also clear that uncompressed video (bitrates > 100 Mbit/s) could not be supported in the Internet.

Dia 60

This figure illustrates the evolution in video codecs.

Dia 61

A feature a sender may add is the use of RTP (Real Time Protocol). This protocol will add some extra information between the UDP header and the voice data (payload). The main features are : payload type, sequence number, time stamp and connection identifier. The RTP header consists of a number of fields (at least 4 x 32 bits).
version (V): 2 bits : This field identifies the version of RTP.
padding (P): 1 bit : If the padding bit is set, the packet contains one or more additional padding octets at the end which are not part of the payload. The last octet of the padding contains a count of how many padding octets should be ignored, including itself. Padding may be needed by some encryption algorithms with fixed block sizes or for carrying several RTP packets in a lower-layer protocol data unit.
extension (X): 1 bit

CSRC count (CC): 4 bits. The CSRC count contains the number of CSRC identifiers that follow the fixed header.
marker (M): 1 bit
payload type (PT): 7 bits : This field identifies the format of the RTP payload and determines its interpretation by the application.
sequence number: 16 bits : The sequence number increments by one for each RTP data packet sent, and may be used by the receiver to detect packet loss and to restore packet sequence. The initial value of the sequence number SHOULD be random (unpredictable).
Note : **RTP session:** The association among a set of participants communicating with RTP. For each participant, the session is defined by a particular pair of transport addresses (one network address plus a port pair for RTP and RTCP).

Dia 62

timestamp: 32 bits : The timestamp reflects the sampling instant of the first octet in the RTP data packet. The sampling instant must be derived from a clock that increments monotonically and linearly in time to allow synchronization and jitter calculations. The resolution of the clock must be sufficient for the desired synchronization accuracy and for measuring packet arrival jitter (one tick per video frame is typically not sufficient). The clock frequency is dependent on the format of data carried as payload and is specified statically in the profile or payload format specification that defines the format, or may be specified dynamically for payload formats defined through non-RTP means. As an example, for fixed-rate audio, the timestamp clock would likely increment by one for each sampling period. If an audio application reads blocks covering 160 sampling periods from the input device, the timestamp would be increased by 160 for each such block, regardless whether the block is transmitted in a packet or dropped as silent. The initial value of the timestamp should be random, as for the sequence number.
SSRC: 32 bits : The SSRC field identifies the synchronization source (SSRC = Synchronization SouRCe). This identifier should be chosen randomly, with the intent that two synchronization sources within the same RTP session can not have the same SSRC identifier.
CSRC list: 0 to 15 items, 32 bits each : The CSRC (Contributing SouRCe) list identifies the contributing sources for the payload contained in this packet. The number of identifiers is given by the CC field (maximum of 15). Example: for audio packets, the SSRC identifiers of all sources that were mixed together to create a packet are listed, allowing correct talker indication at the receiver.

Dia 63

The figure illustrates an example of the use of a timestamp for a constant bitrate voice stream (G.711). The G.711 is producing every 125 μsec voice samples of 8 bits which are grouped in IP packets (e.g. 160 voice samples per IP/RTP packet) . When starting to send out the voice stream, every IP/RTP packet will contain a timestamp (starting at some random value, e.g. 101). This value is incremented by 160 for every IP/RTP packet. If there is a silence period (resulting in no IP/RTP packets), the timestamp will be increased internally in the sender with the same speed (1 per 125 μsec) and when a next IP/RTP packet is sent out, it will have a much larger value (e.g. 561 in the example). The sequence number is also illustrated (also starting at a random value, e.g. 35).

Dia 64

The major solution used to counteract jitter in the network is the use of a dejitter buffer. This buffer allows to reduce drastically the influence of jitter in the network, at the expense of an extra delay and some possible packet loss. The dejitter buffer works as follows :
A sender will send packets at a constant rate to the receiver (e.g. every 10 msec). When transiting the network, delay variations will occur resulting in a degraded signal when played out without counter measures. When the first packet arrives however in a receiver with a dejitter buffer (e.g. voice RTP stream), it will be delayed for a certain time (it will be kept for some time in the dejitter buffer, e.g. 15 msec) before playout. From then on, a timer will start that will play out the next arriving voice packets at a fixed rate (e.g. every 10 msec). The next packet may observe a higher delay in the network and therefor it will be delayed in the dejitter buffer for a shorter time in order to keep the playout rate constant.
It may happen that a packet arrives after its scheduled playout time (e.g. packet 4). In that case it will be discarded and regarded as a lost packet.
In general the use of a dejitter buffer will keep the overall delay constant, from sending the packet on the network till the playout. Packets arriving too late will be discarded.

Dia 65

If a part of the voice stream is lost (e.g. packet loss in the network or packet discard in the dejitter buffer), the receiver may try to resolve the problem. There are several ways to do this, as illustrated in the figure. Silence insertion will insert silence during the time when no information is available. Noise insertion will insert noise and interpolation will send a signal that is an interpolation between the last received value (before the lost part) and the first received value after the lost part. Another possibility is to replay the last received part (with or without fading).
The processing complexity and voice quality of the different techniques is illustrated in the lower graph.
Note that packet concealment is only useful when there is only a small part of information missing (not a burst).
Of course FEC and interleaving will also be used at the receiver side to recover from packet loss.

Dia 67

Because the terminals can only partly resolve the problems with delay, delay jitter and packet loss, other measures have to be taken by the network. The network will classify the traffic and treat the traffic classes differently (e.g. data traffic is not sensitive to delay or delay jitter, therefore voice traffic may have priority compared to normal data services). The network control may also limit the access to the network (CAC or Customer Access Control) in order to avoid excess delay or packet loss due to congestion in the network.
Two important aspects will be discussed : specific supporting techniques in the network nodes (routers) and global aspects related to network coordination.

Dia 69

The typical building blocks of a QoS-aware IP router are depicted in the figure above. Classification means the identification of the flows. The flows will be either shaped or policed (or go directly to the buffers). The packets are then stored in one of the queues depending on the desired QoS. Each queue can for example correspond to a service category (e.g. real time

streaming video or non real time data). The scheduler then determines which queue is allowed to send packets to the output port. Buffer acceptance will be important in case the buffers are filling up. It will be responsible for discarding some packets.
The next slides will illustrate each of these building blocks.
Note : the building blocks should not be at the output. Part of the classification could be done at the inlet (e.g. if forwarding decision is depending on traffic class).Dia 70

Classification is very important in order to identify different packet flows in the network. Differentiation could be based on the type of traffic (traffic classes, e.g. voice or data) but could also be based on a differentiation between the endpoints. In this way the flows can be treated differently inside the router (e.g. priority).
Classification can be done using a specific classifier (e.g. in the TOS byte of the IP header) or label, as will be discussed in the Diffserv case. Marking (= filling in the correct classifier value) may be done at the edge of the network. It is also possible to classify based on the IP address(es) (of source and/or destination), on the port number(s) (of source and/or destination) or on the combination of address and port numbers.

Dia 71

Policing will be important in order to check if an incoming flow is conform to a certain specification. A user may ask the network a certain bandwidth and the network may accept this (e.g. if there is still enough bandwidth available).
Policing can be done using the token bucket algorithm. If IP packets are not conform to the specification (e.g. bitrate too high which means that packets are arriving too fast), they may be discarded or marked (with a lower priority) or just made best-effort packets (= lowest priority).

Dia 72

The token bucket algorithm works as follows :
A token bucket is filled up at a certain rate (e.g. r tokens/sec) and it has a certain depth. The tokens will be used to control the transfer of incoming IP packets (with a byte scale resolution). When a packet arrives with a length of 1000 bytes and there are 1500 tokens in the bucket (bucket level X = 1500), the packet is transferred to the output and the number of tokens in the bucket is reduced by the amount of bytes transferred to the output (1000 bytes). The new value is now X=500. The token bucket is however filled again. When a packet arrives with a length of 2000 bytes and the bucket has only 1500 tokens (X=1500), then the packet will be discarded or marked or made best-effort. In this way it is possible to control the average rate of a certain flow.
Note that the bucket depth will limit the maximum number of tokens available (to a maximum b). This will also limit the maximum burst length that will be allowed. Suppose that for some time no packets had arrived and the bucket is completely full (the bucket depth b is 10000). Suppose also that a burst of 15 packets arrives with a length of 1000 bytes for each packet. In that case the first 10 packets will be transferred without any problem (corresponding to a total of 10000 bytes) but the next 5 packets will be discarded or marked or made best effort.
Example : suppose one wants to limit the speed of a certain flow to 1.6 Mbit/s and the maximum allowed burst size to 5000 bytes. In that case one will use the following parameters:
r = 1.6 Mbit/s / 8 bits = 200.000 tokens/s
b = 5000

Dia 73

An example of policing is illustrated.

Dia 74

Another mechanism used in a router is shaping of the traffic flows. This will reduce the possible strong variation in packet rate or bitrate (e.g. it may limit the burst size, without however discarding any packets). This will also be based on the token bucket algorithm.

Dia 75

The major difference with the token bucket algorithm used for policing is that non conforming packets are delayed until enough tokens are available in the token bucket. This will effectively smooth out the traffic.

Dia 76

An example of shaping is illustrated.
In many cases a traffic flow will first be shaped before entering a policer.

Dia 77

After policing and/or shaping (note that the two may be combined !), the packets have to be buffered. Different possibilities exist : use of a single buffer (queue), use of a queue per class of information flow (e.g. voice versus data) or use of a queue for each individual flow (e.g. for each voice connection transiting the router). When multiple queues are used for the same output, a scheduler is required to decide which packet (from which queue) will be sent to the output line.

Dia 78

When too many packets arrive in the buffers, measures have to be taken (discard packets). The decision on when a packet should be discarded is based on the observation of the buffer occupancy (buffer completely full or above a certain threshold value). This can be done in a straightforward manner : drop the arriving packets when the buffer is full. But it may be much better to drop a random packet or a packet from the same flow or queue.

Dia 79

A first example of buffer acceptance strategy is RED or Random Early Detection. In this case the buffer filling will be measured continuously (an average value (avg) will be calculated). Two thresholds will be used : min and max. The incoming packets will be accepted if the average buffer filling is below the min threshold. The incoming packets will be discarded with a certain (increasing) probability if the average filling is between min and max. If the average filling is above the max threshold, all packets will be discarded.
This probabilistic dropping will improve the network performance. By randomly dropping packets from different flows, different sources will reduce their sending rates due to the TCP behaviour (slow start, fast retransmit). Random dropping will also result in a dropping rate proportional to the network usage of the different flows.

Dia 80

Weighted RED will (e.g.) use two traffic classes : high and low priority packets. Both will have different min and max values, different PMAX and different slopes. The net result is that low priority packets have a higher chance to be dropped.

Dia 81

The scheduler will indicate which queue is allowed to send a packet to the outlet (in case of multiple queues).
The example illustrates a very simple scheduler for two queues (high and low priority). If there is a packet in the H queue, the scheduler will always give priority to that queue. It is only in case the H queue is empty that the L queue is allowed to send packets to the outlet. The major advantage is that this is a very simple scheduler and that the high priority traffic will observe a very good treatment. The low priority traffic on the other hand may observe a very bad behaviour. In case of a large volume of high priority traffic, the low priority traffic will be completely blocked, resulting in packet loss (because the buffers have a finite depth).

Dia 82

A more advanced scheduling algorithm is weighted round robin. Each queue will have a certain weight (corresponding to its relative priority) and the queues will be served based on their weighting factors. The example illustrates 4 queues with weight : 5/10, 3/10, 1/10 and 1/10. It is a scheduling algorithm which is easy to implement if the number of queues is limited and it allows to differentiate different traffic flows. Also the low priority traffic will be served.
A special case is the round robin scheduler where all the weights are equal.
Note : if there is a queue without a packet then the scheduler will immediately move to the next queue ("work-conserving" mode of operation).
Note : WRR is often called WFQ or Weighted Fair Queuing.We will use this terminology.

Dia 84

There are two architectures presented in the IETF in order to support QoS in the network : IntServ (Integrated Services) and DiffServ (Differentiated Services). The figure illustrates the complexity and the flow differentiation.
In case of a Best Effort (BE) internet, no differentiation is made between the different traffic flows. They are al treated in the same way and complexity is very low.
In case of Differentiated Services (DiffServ) one will introduce a number of service classes with different requirements. The different service classes will be identified by using the TOS (Type Of Service) byte in the IP header (the DSCP or DiffServ Code Point). This approach requires only a limited amount of state in the routers (state related to each service class).
In case of Integrated Services (IntServ) one will treat the individual flows separately, resulting in a large amount of state in the routers (for each flow the router has to maintain state). This architecture results in general in more flexibility and tailoring towards specific needs of individual flows, but on the other hand it has a big problem with scalability (due to the large amount of state). IntServ will require extensive signaling in the network. This will be based on RSVP (Resource Reservation Protocol).
Note : Today, IntServ is only considered as a viable solution for small (typical local or access) networks. Most interest goes to the DiffServ architectures. From a conceptual point of view, it is interesting to describe both.

Dia 85

This figure illustrates the IntServ approach : all individual flows have to be treated separately in the network. Each router where the flow is transiting will keep some information (state) about this specific flow.

Dia 86

This figure illustrates the DiffServ approach : different classes are treated separately in the network (only state about the different classes has to be maintained in the routers). There is no need to separate individual flows, only classes (which may contain many flows) are separated. The figure illustrates two classes.

Dia 87

Differentiated Services (DiffServ) will make use of different service classes to provide a certain level of QoS to different flows. The term “behaviour class” is used (see RFC 2475). Inside the network, the nodes (DiffServ routers) will support certain Per Hop Behaviours (PHB) to provide the required QoS for the different behaviour classes. This PHB will influence the sharing in a router of buffers and link bandwidth between the different classes of traffic.
In practice, two service classes (behaviour classes) are defined : expedited forwarding (EF) and assured forwarding (AF).

Dia 88

The functions required in a DiffServ enabled network are indicated on the figure.
At an edge router, the flow has to be classified based on e.g. source and destination IP addresses and port numbers (but also other means are possible, e.g. RTP versus non-RTP packets). Based on this classification one will mark the packets using the DSCP (DiffServ Code Point). Policing and shaping may be required because flows have to be conforming a certain traffic profile (otherwise it would not be possible to provide any QoS in the network because a certain uncontrolled flow could overwhelm the network using all resources of lower quality classes).
In a core router one has to support certain Per Hop Behaviours (PHB). These PHB’s will define differences in the performance behaviour amongst different classes. Note that a PHB is only a description of a behaviour at a node (node is considered as a “black box”) and does not specify how this behaviour should be achieved. The available means to support a certain PHB are buffering, buffer management and scheduling. There are two important PHB’s : Expedited Forwarding (EF) and Assured Forwarding (AS).
In general, a traffic flow has to be allowed to a certain traffic class (“behavior aggregate”). This could be done in a static way, meaning that the network operator is manually configuring the edge router in order to allow a certain flow to a certain traffic class (static SLA or Service Level Agreement). The operator could keep track of all the current traffic already allowed on the network, and based on this information a decision is made whether to accept the additional flow. This could also be done in a more dynamic way using a dynamic SLA and a centralised computer (Bandwidth Broker or BB) which will interact with the customers (to negotiate and agree on a SLA), with the edge routers for configuration and (eventually) with the core routers to obtain information about the current traffic and to set certain parameters (e.g. weight factors in WFQ). Note that the latter (traffic measurement) is not necessary because the Bandwidth Broker may keep track of the current flows already allowed to a certain traffic class. The

interaction of BB with customer and edge router could be based on the use of RSVP (which was originally developed in conjunction with IntServ).

Dia 89

So far the routing in the internet is based on a shortest path routing algorithm (typically OSPF) using a simple metric (typically hop count). In addition traffic between a certain source and destination (area) is following a single route (although this might change over time due to reconfigurations).
The use of a simple shortest path routing algorithm has a number of important limitations. In general a shortest path today is defined by the number of hops from source to destination (the shortest path metric is the hop count). This does not give any information about the delay along the path or the available bandwidth. In that respect it becomes interesting to consider constrained based routing, where different metrics could be used (for time critical traffic the delay is important, for high bandwidth traffic the available bandwidth is important). Of course these metrics are not fully independent (e.g. a higher available bandwidth will in general result in a lower delay, but a high bandwidth route may also result in a long delay if the physical length of the route (e.g. via satellite) is much longer than the lower bandwidth route (e.g. via fiber)).
A second important remark is that classical (single route) routing may result in traffic concentration on certain links and in routers whereas on other parts of the network a lot of resources are still available. Load balancing (or more general traffic engineering) may help to more evenly distribute the traffic over the network, resulting in less congestion on certain links. In order to support load balancing, MPLS (Multi-Protocol Label Switching) may be a very useful supporting technology. This allows to set up explicit routes in the network (not the shortest routes).

Dia 90

An important question is how to find the best route to fulfil a certain constraint (e.g. delay, bandwidth). Both IntServ and DiffServ rely on the standard routing protocols used in the current internet (e.g. OSPF). This will result however in a route that takes into account only one metric (most of the times this is the hop count). In order to find the optimum route through a network it may be beneficial to take into account several and more advanced metrics (e.g. delay, bandwidth, packet loss rate). The example shows two routes in the network. A first (low delay) route will pass over a fiber link which is however close to congestion (due to traffic already using this link) resulting in a limited bandwidth. A second route will run over a satellite link (large propagation delay) but is hardly used (large bandwidth still available). Depending on the application one may opt for one or the other route. This will be supported by constrained based routing.
In order to use constrained based routing, information about the different metrics has to be distributed in the network. This is possible by extending the existing routing protocols. OSPF for example is currently only providing topology information with a single metric (typically set to 1 for the hop count). By adding other metrics one could extend OSPF (Q-OSPF where Q stands for QoS) and obtain in this way a topology database with multiple metrics per link. From that topology database it will be possible to calculate generalized shortest paths (e.g. path with smallest delay, largest bandwidth, …).
These calculations may become very complex depending on the specifics of the metrics. For bandwidth one can take the smallest value of the route (concave constraint), for delay one has

to add the different contributions from the different links (additive constraint) and for loss rate one has to multiply the link specific values (multiplicative constraints). These are known to be very computational intensive calculations.
It is clear that the use of constraint based routing algorithms is very attractive for IntServ and DiffServ but they are also useful on their own (in the normal best effort forwarding strategy).
Note : the use of constraint based routing will result in much more complex routing tables ! Also the stability of the protocol becomes more problematic.

Dia 91

An important problem in the current internet routing is the fact that shortest paths are used resulting in some concentration of traffic on certain links (resulting in congestion, delay and packet loss). In order to counteract this, one could use load balancing in order to distribute the traffic over the network more evenly. This requires however the use of multiple paths between source and destination (area).
A first possibility is the equal cost multipath algorithm. This will be useful when there are several shortest routes between the same two endpoints. In this case one could try to use the different shortest routes (and not choose one). A possibility is to use a hash function (e.g. half of the destination domains will go on shortest path 1 and the other half on shortest path 2).
A second possibility is to set up explicit tunnels in the network which may follow a route, different from the shortest route. A supporting technology is MPLS or Multi Protocol Label Switching.

Dia 93

MPLS will replace the normal packet forwarding based on a router table look-up with a forwarding based on a label (and label swapping). This is illustrated in the figure.
An MPLS packet will have an additional header in front of the IP-packet header (in fact the MPLS header is inserted between the IP header and the layer 2 header, e.g. Ethernet header). This header (of 32 bits) contains a label (20 bits), 3 bits for experimental use, a stacking bit and an 8 bit TTL. The label will play a central role in the forwarding process (also called label switching).
Suppose an MPLS packet arrives in a router supporting MPLS (=LSR or Label Switched Router). Upon arrival one will look up a database, the Label Information Base (LIB), in order to know where the packet has to go and what the new label is on the outgoing link. The example illustrates an incoming label of 5 on input link 1 that is mapped to an outgoing label of 4 on output link 1 (as indicated in the LIB). Another MPLS packet arriving on IN 2 with label 3 will be mapped to OUT 1 with label 5.
Important : labels do only have local significance. An end to end path (LSP or Label Switched Path) consists of labels on each link and a relation between them defined in the label information base ("label swapping table"). As a result, the same label may be reused on different links.
Note : ATM or Asynchronous Transfer Mode is using a similar concept.

Dia 94

The goal of MPLS is to set up dedicated routes in the network where the forwarding is based on a label swapping operation (and not based on a longest prefix match as in normal IP

forwarding). The principle of path set-up in MPLS (Multi Protocol Label Switching) is as follows : suppose one wants to set up a label switched path (LSP) from X to Y.
First a label request message is sent from source X to destination Y using the normal forwarding of IP packets (following the shortest path as obtained from e.g. OSPF). When the label request arrives in Y, this computer (could also be a router, typically an edge router) will assign a label for this request (e.g. 100). This should be a label that is not yet used on the link from the computer to the last router (C). The computer will send a label mapping message back to the last router. This router will assign a label (e.g. 200) on the link towards the previous router B (a label that is not yet used on this link) and it will keep a label swapping table (Label Information Base) in its memory (swapping from 200 to 100 when information will come from the source X). A label mapping message will be sent from C to B and B will assign a label on the link towards router A (e.g. 100). Router A will take similar actions assigning a label on the link to X (e.g. 300) and filling in its label swapping table (from 300 to 100). In this way a LSP is set up from source to destination.
The protocol used to set up this path (to assign and distribute the labels) is RSVP (in the past, the Label Distribution Protocol (LDP) was also considered).
Some terminology :
MPLS capable router : Label Switched Router (LSR)
MPLS path : Label Switched Path (LSP)

Dia 95

This figure illustrates the principle of the load-balancing support of MPLS. From terminal X one can send IP packets to terminal Y via two different routes (support by MPLS). Sending an IP packet with MPLS label 300 will follow the lower route (label swapping sequence : 300 – 100 – 200 – 100), sending an IP packet with label 50 will follow the upper route (label swapping sequence : 50 – 150 – 450 – 100). Choosing the appropriate label will allow to send traffic over two different routes, enabling the spreading of the traffic over the network.

Dia 96

MPLS may be used to set up explicit paths in a network, not following the shortest path as illustrated previously. In this case the path will be set up using explicit routing : the route will be calculated at the source (e.g. constraint based routing) and will be explicitly carried in the path set-up message (label request message). The path set-up message will follow this explicit route and not the "OSPF-based" shortest route. In this way, a Label Switched Path will be set up following a specific route (with some specific features).
The example shows the routing table in case MPLS is not used (table on top). In this case a packet arriving at the incoming interface of router A with destination 145.12.134.3 will be forwarded to router B, as indicated in the routing table.
In case of the use of MPLS, a Label Switched Path (LSP) is set up between router A and router C and the routing table entry for network 145.12.0.0 is directing the packets into the LSP from A to C via D, E and F.