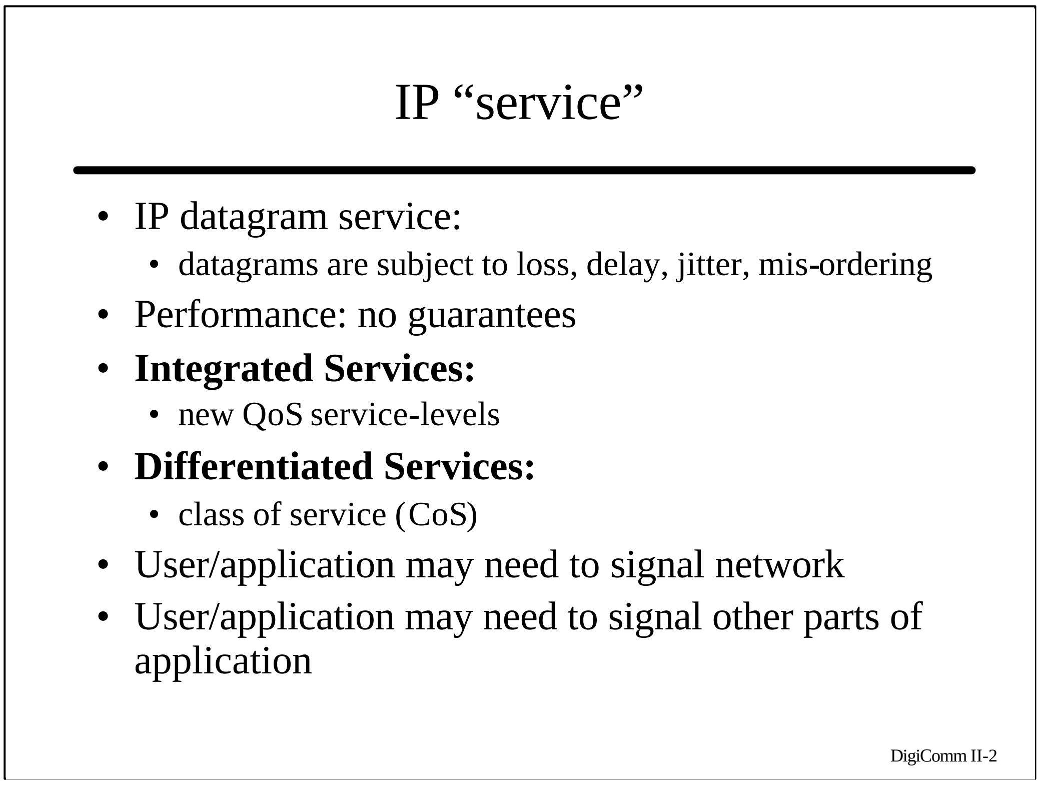
QoS services and application-level service interfaces

DigiComm II-1



IP "service"

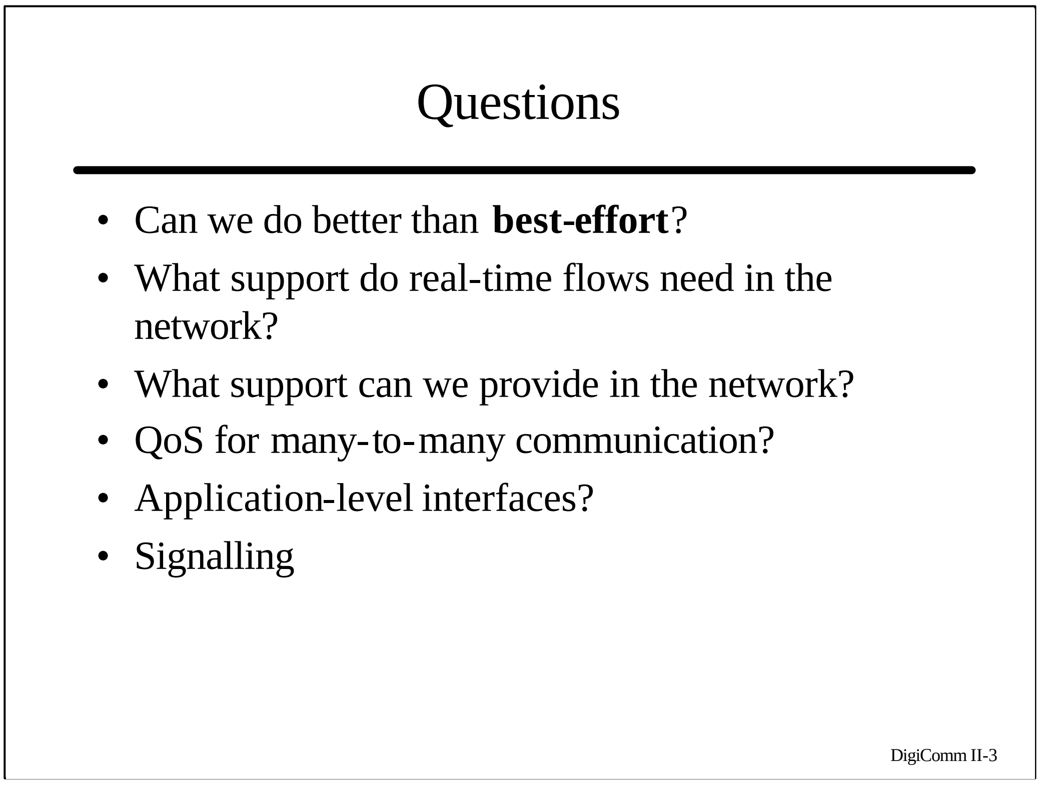
- IP datagram service:
  - datagrams are subject to loss, delay, jitter, mis-ordering
- Performance: no guarantees
- **Integrated Services:**
  - new QoS service-levels
- **Differentiated Services:**
  - class of service (CoS)
- User/application may need to signal network
- User/application may need to signal other parts of application

DigiComm II-2

Internet users have increasing demands to use a range of multimedia applications with QoS sensitive data flows. All these applications may require different QoS guarantees to be provided by the underlying network. An e-mail application can make do with a best-effort network service. Interactive or real-time voice and video applications require (some or all of) delay, jitter, loss and throughput guarantees in order to function. Web access can make do with a best-effort service, but typically requires low delay, and may require high throughput depending on the content being accessed. The Internet was never designed to cope with such a sophisticated demand for services. Today's Internet is built upon many different underlying network technologies, of different age, capability and complexity. Most of these technologies are unable to cope with such QoS demands. Also, the Internet protocols themselves are not designed to support the wide range of QoS profiles required by the huge plethora of current (and future) applications.

To deal with such issues, the IETF have two working groups looking at QoS issues directly. The INTSERV WG is looking at how to extend the IP network to become an **Integrated Services Network (ISN)**. INTSERV have been looking at the definition and support of new **service-levels** (other than best-effort) within an IP network. The DIFFSERV WG is looking at the provision of **differentiated services** within IP networks, allowing service providers to treat packets from different sources with different QoS. The source of packets is defined administratively, e.g. could be a single host or a whole organisation.

In some cases, the user or the application (or both) may need to indicate their requirements to the network by use of some sort of signalling. The notion of user-to-network signalling is not inherent in the IP networking model. At the application-level, there may be a requirement for user-to-user (application-to-application) supported as a network service.



Questions

- Can we do better than **best-effort**?
- What support do real-time flows need in the network?
- What support can we provide in the network?
- QoS for many-to-many communication?
- Application-level interfaces?
- Signalling

DigiComm II-3

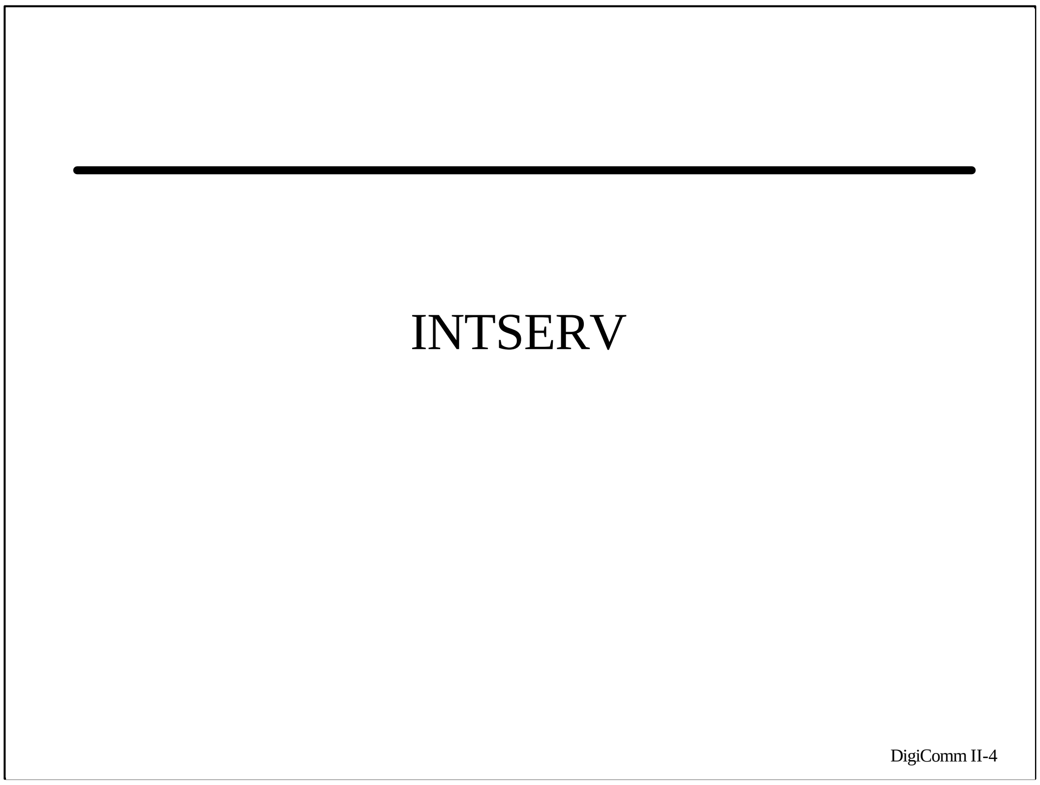
So we can ask ourselves several questions.

Firstly, can we provide a better service that that which IP currently provides – the so-called best-effort?

The answer to this is actually, "yes", but we need to find out what it is we really want to provide! We have to establish which parameters of a real-time packet flow are important and how we might control them. Once we have established our requirements, we must look at new mechanisms to provide support for these needs in the network itself. We are essentially asking trying to establish alternatives to FCFS for providing better control of packet handling in the network as well as trying to support QoS for **multi-party (many-to-many) communication**.

We also need to consider how the applications gain access to such mechanisms, so we must consider any **application-level interface** issues, e.g. is there any interaction between the application and the network and if so, how will this be achieved.

IP, as connectionless network protocol, involves no signalling. However, in order to allow the use of real-time applications, we need to establish a richer set of function in order to allow information about a communication session to sent into (and across) the network. So, we need signalling capability, both user-to-network and user-to-user.



INTSERV

DigiComm II-4

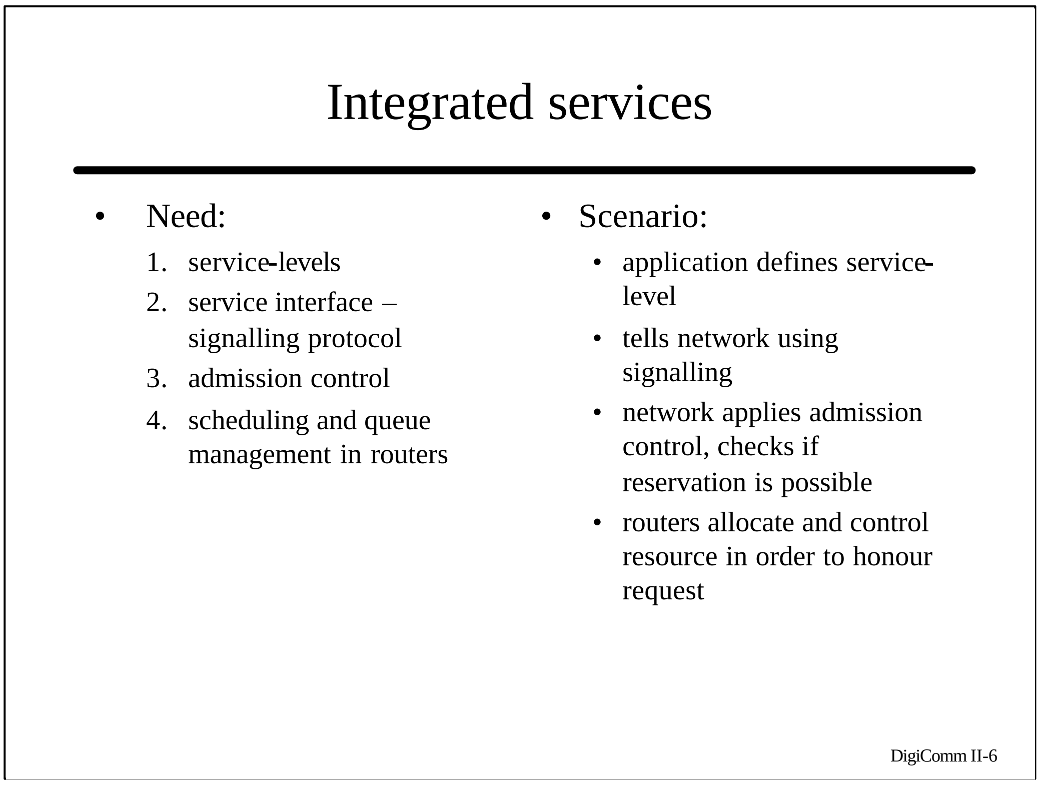
Questions

- What support do we need form the network to give QoS capability to the Transport layer?
- How can we control congestion in the network?
- How can we support legacy network protocols over the Internet?

DigiComm II-5

In the last section we produced a taxonomy of applications with respect to their QoS requirements. Real-time applications need a better service that standard IP unreliable datagram delivery. We will see that there is some support available at the transport layer for real-time applications (e.g. by use of RTP), but this can not give us QoS guarantees. We need direct support form the network so we must modify the operation of the routers somehow so that they can give priority to QoS sensitive traffic.

Finally, we take a brief look at how legacy applications might be operated across an IP-based network.



Integrated services

- Need:
  1. service-levels
  2. service interface – signalling protocol
  3. admission control
  4. scheduling and queue management in routers
- Scenario:
  - application defines service-level
  - tells network using signalling
  - network applies admission control, checks if reservation is possible
  - routers allocate and control resource in order to honour request

DigiComm II-6

To provide Integrated Services for IP applications, we can envisage the following scenario:

• a **service-level** is defined (e.g. within an administrative domain or, with global scope, by the Internet community). The definition of the service-level includes all the service semantics; descriptions of how packets should be treated within the network, how the application should inject traffic into the network as well as how the service should be policed. Knowledge of the service semantics must be available within routers and within applications

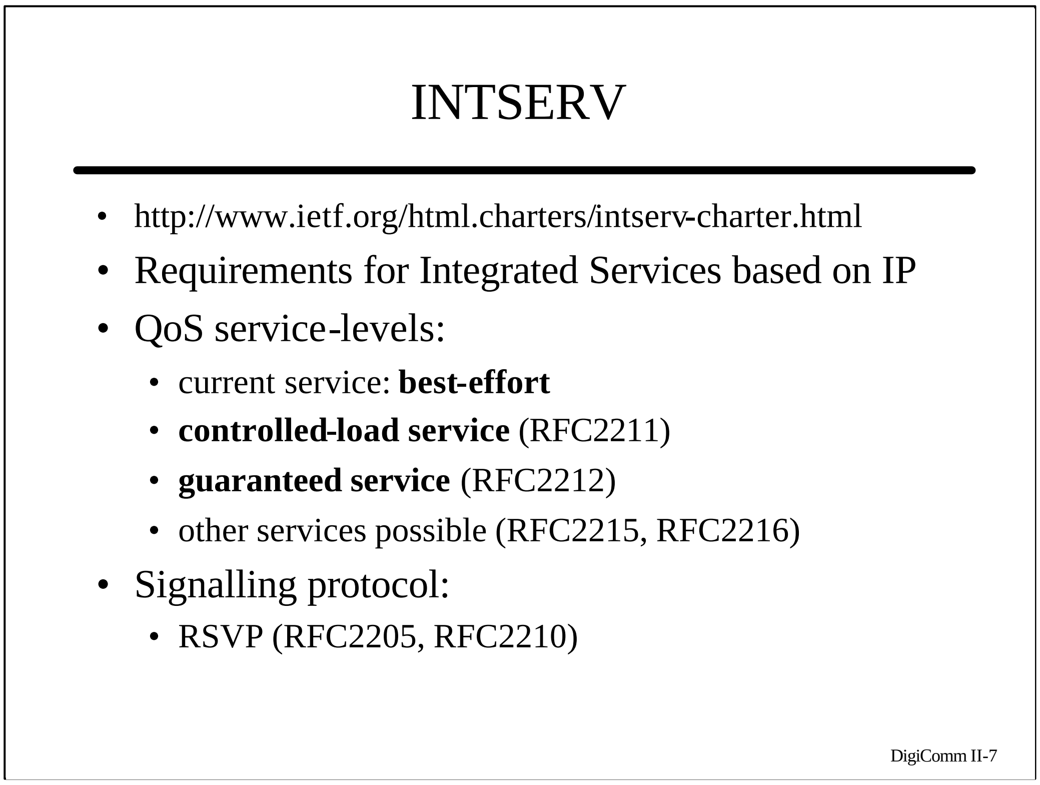
• an application makes a request for service invocation using the **service interface** and a **signalling protocol**. The invocation information includes specific information about the traffic characteristics required for the flow, e.g. data rate. The network will indicate if the service invocation was successful or not, and may also inform the application if there is a service violation, either by the application's use of the service, or if there is a network failure

• before the service invocation can succeed, the network must determine if it has enough resources to accept the service invocation. This is the job of **admission control** that uses the information in the service invocation, plus knowledge about the other service requests it is currently supporting, and determines if it can accept the new request. The admission control function will also be responsible for policing the use of the service, making sure that applications do not use more resources than they have requested. This will typically be implemented within the routers

• once a service invocation has been accepted, the network must employ mechanisms that ensure that the packets within the flow receive the service that has been requested for that flow. This requires the use of **scheduling mechanisms** and **queue management** for flows within the routers

We examine each of the highlighted components.

The Internet Integrated Services architecture is described in [RFC1633].



INTSERV

- http://www.ietf.org/html.charters/intserv-charter.html
- Requirements for Integrated Services based on IP
- QoS service-levels:
  - current service: **best-effort**
  - **controlled-load service** (RFC2211)
  - **guaranteed service** (RFC2212)
  - other services possible (RFC2215, RFC2216)
- Signalling protocol:
  - RSVP (RFC2205, RFC2210)

DigiComm II-7

It is possible to identify four specific technical issues that need to be addressed in the provision of Integrated Services for IP-based networks:

• **service-level:** the nature of the commitment made, e.g. the INTSERV WG has defined **guaranteed** and **controlled-load** service-levels and a set of control parameters to describe traffic patterns

• **service interface:** a set of parameters passed between the application and the network in order to invoke a particular QoS service-level, i.e. some sort of signalling protocol plus a set of parameter definitions

• **admission control:** for establishing whether or not a service commitment can be honoured before allowing the flow to proceed

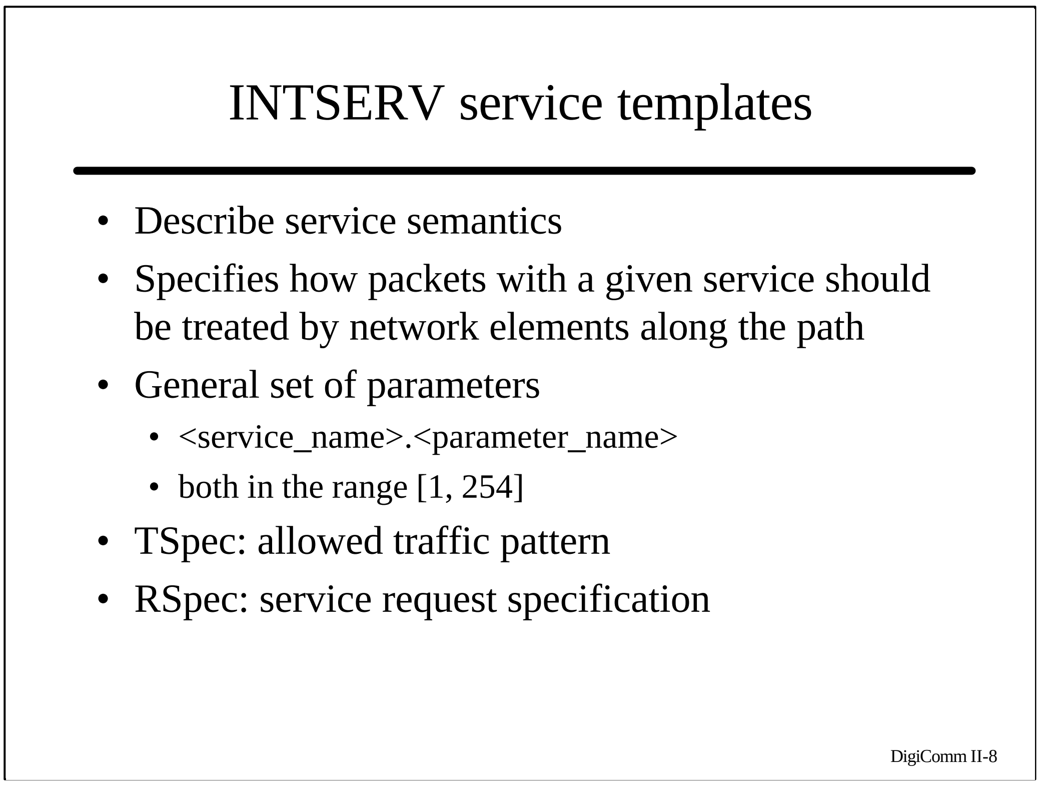
• **scheduling mechanisms within the network:** the network must be able to handle packets in accordance with the QoS service requested

The INTSERV WG addresses only the first two of these issues. However, the INTSERV work does specify the requirements for any mechanisms used to address the last two issues, with some implementation hints. With the present IP service enumerated as **best-effort**, currently, two service-level specifications are defined:

• **controlled-load** service [RFC2211]: the behaviour for a network element required to offer a service that approximates the QoS received from an unloaded, best-effort network

• **guaranteed** service [RFC2212]: the behaviour for a network element required to deliver guaranteed throughput and delay for a flow

The INTSERV signalling protocol is called RSVP (Resource Reservation Set-up Protocol, [RFC2205] [RFC2210]).



INTSERV service templates

- Describe service semantics
- Specifies how packets with a given service should be treated by network elements along the path
- General set of parameters
  - <service_name>.<parameter_name>
  - both in the range [1, 254]
- TSpec: allowed traffic pattern
- RSpec: service request specification

DigiComm II-8

INTSERV have produced a set of specifications for specific QoS service-levels based on a general network service specification template [RFC2216] and some general QoS parameters [RFC2215]. The template allows the definition of how network elements should treat traffic flows, i.e. the QoS granularity here is that of a single logical **flow** (or **session** in RSVP parlance).

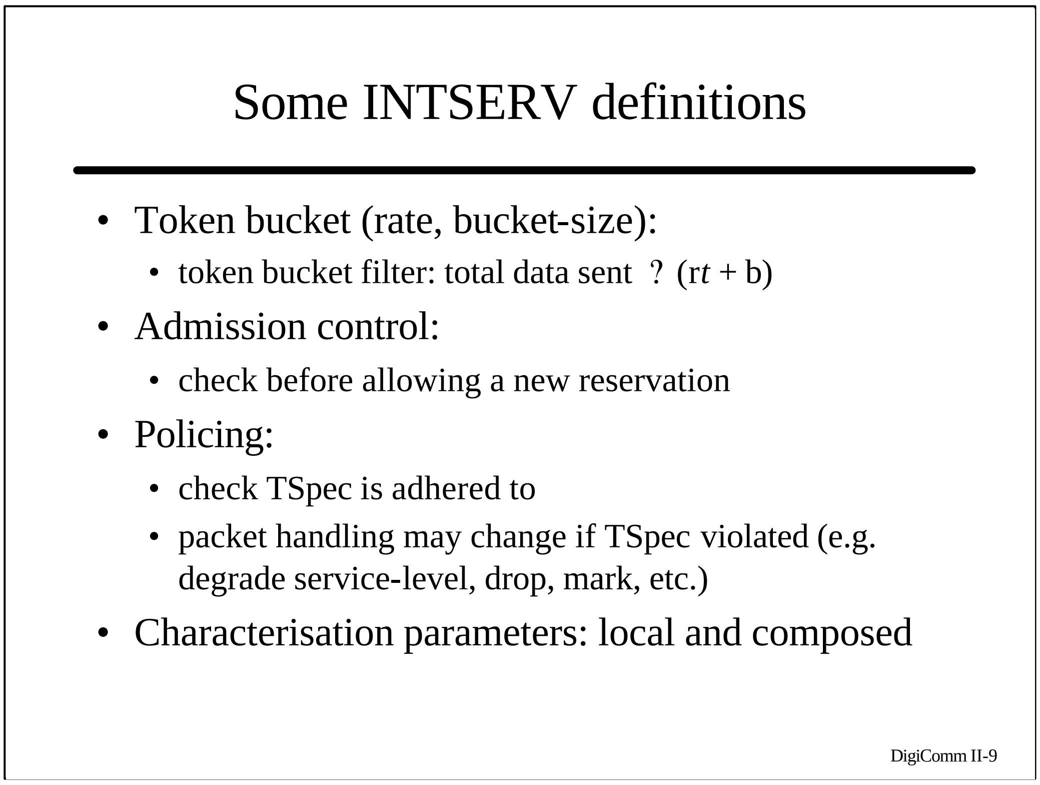
The service template specifies describes the semantics of the services and specifies how packets should be treated as they pass through network elements that would like to implement the service, i.e. packet handling rules.

The genral parameters are identifed using two bytes, one identifying the service name (e.g. controlled-load) and one identifying the parameter.

The use of a service requires a **TSpec** (Traffic Specification) to specify the allowed traffic characteristics for a session. A **RSpec** (Resourse Specification) may also be used during reservation establishment for service specific parameters, but its use is service specific. The service definition includes information on how admission control is applied for the service and how the service is policed within the network (how non-conformant packets should be handled).

The controlled-load service requires a TSpec but no RSpec. For the guaranteed service,, as well as a TSpec, a RSpec should be specified (which will not be discussed here).

Note that this architecture requires that all the network elements along the path, as well as the applications, and applications have semantic knowledge about the service-levels for the application flows, as specified in the service templates.



Some INTSERV definitions

- Token bucket (rate, bucket-size):
  - token bucket filter: total data sent ? (r*t* + b)
- Admission control:
  - check before allowing a new reservation
- Policing:
  - check TSpec is adhered to
  - packet handling may change if TSpec violated (e.g. degrade service-level, drop, mark, etc.)
- Characterisation parameters: local and composed

DigiComm II-9

A key element of the flow description is the TSpec that describes the (expected) traffic characteristics of he flow/session. The traffic characteristic is defined in terms of a **token-bucket** filter which, in general has the following elements:

• **r**: the data rate (bytes/s)

• **b**: the bucket size (bytes)

This specifies that the flow shall have sent no more than (r*t* + b) bytes of data at any time *t*.

This information (along with other, service specific information) may be used by the network for:

• **admission control**: to check if the requested traffic profile and service can be currently be honoured along the intended network path

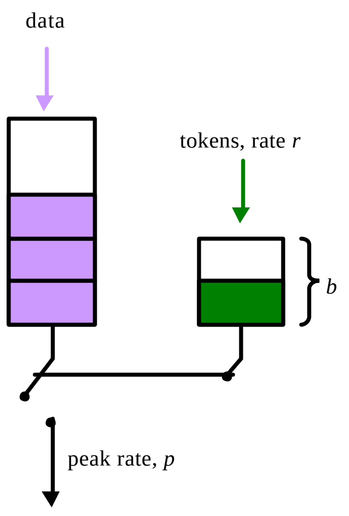
• **policing**: to ensure that the application/user does not exceed the requested traffic specification and to take action (mark/drop packets, degrade the service-level for some packets) as appropriate

The INTSERV specification allows network elements to have **local** value for INTSERV parameters. When a path for a flow/session is selected the **composed** values for a parameter are the combination of the local values for the network elemnts along the path. For example, network elements A, B and C form a path for a flow or session. They have path latency values of 10ms, 12ms and 8ms respectively. Th composed value for the (minimum) path latency will be (10+12+8)ms = 30ms. Different parameters have different composition rules which should be defined when the parameter for a service is defined.

Token bucket (recap)

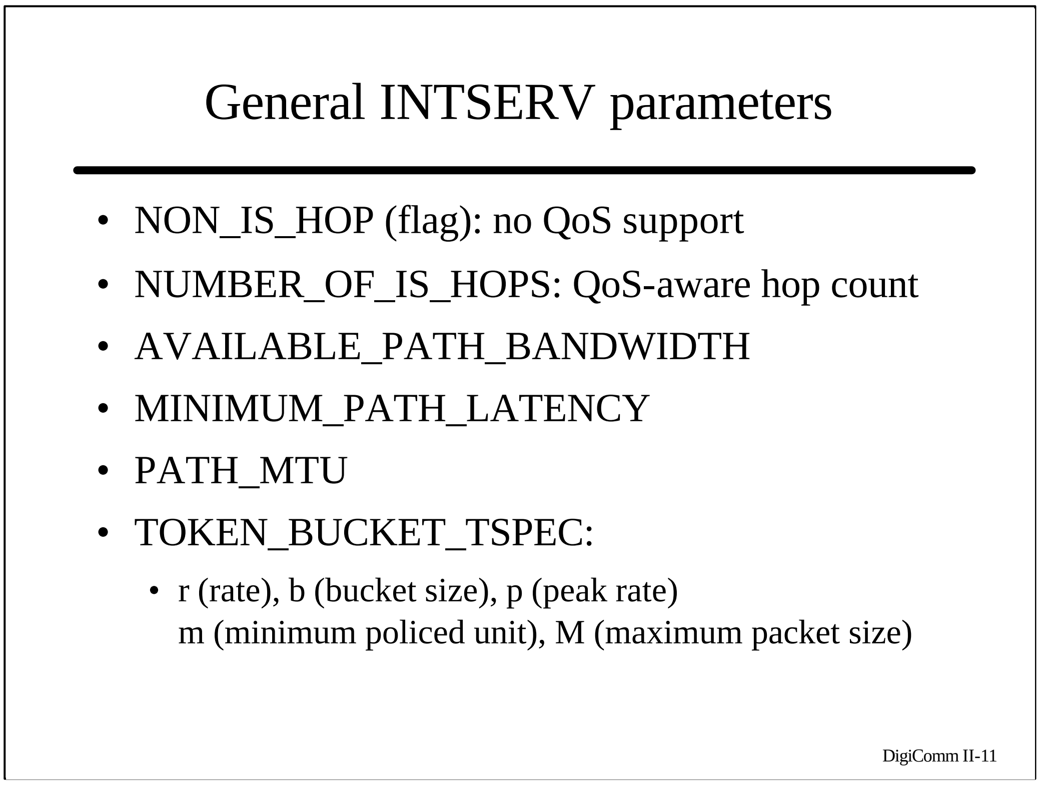
Token bucket

- Three parameters:
  - *b*: bucket size [B]
  - *r*: bucket rate [B/s or b/s]
  - *p*: peak rate [B/s or b/s]
- Bucket fills with tokens at rate *r*, starts full
- Tokens allow transmission
- Burst allowed at rate *p:*
  - data sent < *rt* + *b*
- (Also *m* and *M*)



DigiComm II-10

The **token bucket** has a bucket size, *b*, and a bucket rate, *r*, and allows traffic bursts to be transmitted at peak rate, *p*, under certain conditions. The bucket does not “fill with data” as it does in the leaky bucket, but it fills with tokens, that that allow data to be transmitted. Data can only be transmitted when there are enough tokens to allow transmission to take place. Transmission can then take place at a peak rate of *p*. Nominally, data is transmitted at a rate *r*, the same rate at which the bucket is filled with tokens. However, it can be seen that bursts of traffic, up to the bucket size, can be transmitted at the peak rate, *p*. In fact, we may also need to specify *m*, the minimum packet size, and *M*, the maximum packet size.



General INTSERV parameters

- NON_IS_HOP (flag): no QoS support
- NUMBER_OF_IS_HOPS: QoS-aware hop count
- AVAILABLE_PATH_BANDWIDTH
- MINIMUM_PATH_LATENCY
- PATH_MTU
- TOKEN_BUCKET_TSPEC:
  - r (rate), b (bucket size), p (peak rate)
    m (minimum policed unit), M (maximum packet size)

DigiComm II-11

The service template specifies and describes the semantics of the services and specifies how packets should be treated as they pass through network elements that would like to implement the service, i.e. packet handling rules. There is a general set of parameters specified and their values can be defined for each service level, as required. The general parameters include:

• a flag to indicate that a network element is not INTSERV-aware

• hop-count of INTSERV-aware network elements

• available path bandwidth

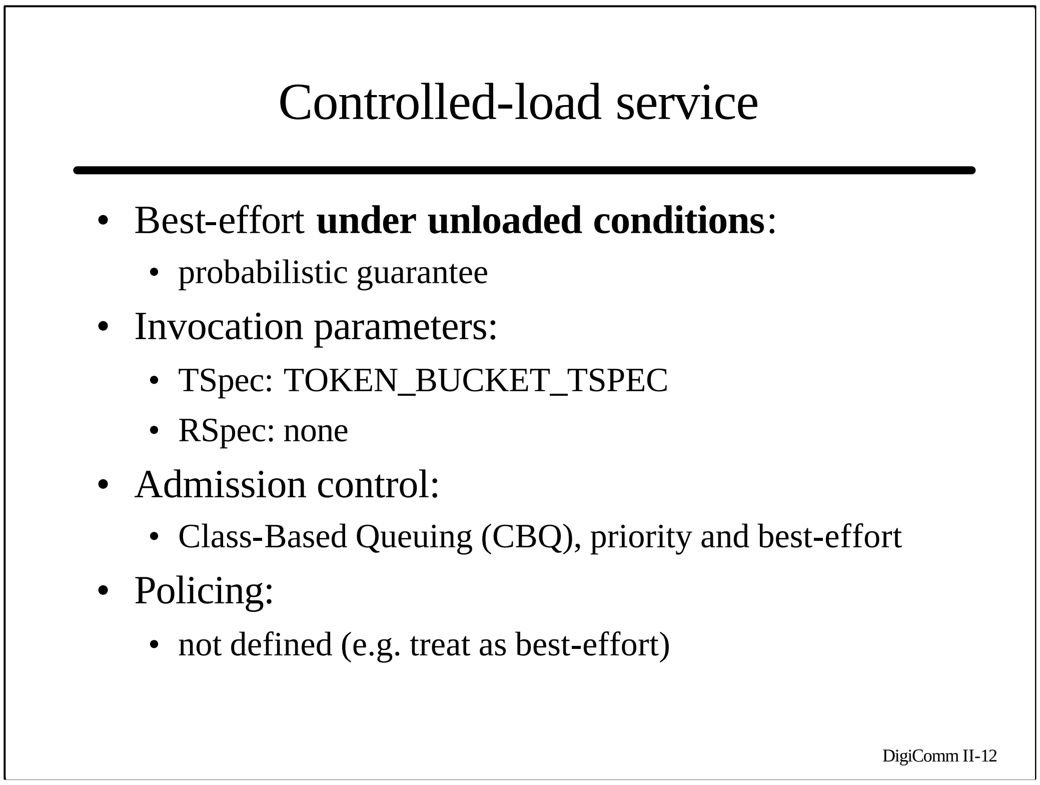
• minimum path latency

• path MTU

• traffic characteristic in terms of a token bucket specification (data rate, bucket size, peak rate), minimum packet size to be policed and maximum packet size allowed

The last of these parameters is used in the **TSpec**. A **RSpec** (Resource Specification) may also be used during reservation establishment for service specific parameters.

Other, service-specific parameters may be defined. Some of the paramters (for example AVAILABLE_PATH_BANDWIDTH and MINMUM_PATH_LATENCY) are carried in a special **AdSpec** data structure (see later) and are filled in by the routers along the network path to form **composed values**, which represent the values of those parameters for the path as a whole.



Controlled-load service

- Best-effort **under unloaded conditions**:
  - probabilistic guarantee
- Invocation parameters:
  - TSpec: TOKEN_BUCKET_TSPEC
  - RSpec: none
- Admission control:
  - Class-Based Queuing (CBQ), priority and best-effort
- Policing:
  - not defined (e.g. treat as best-effort)

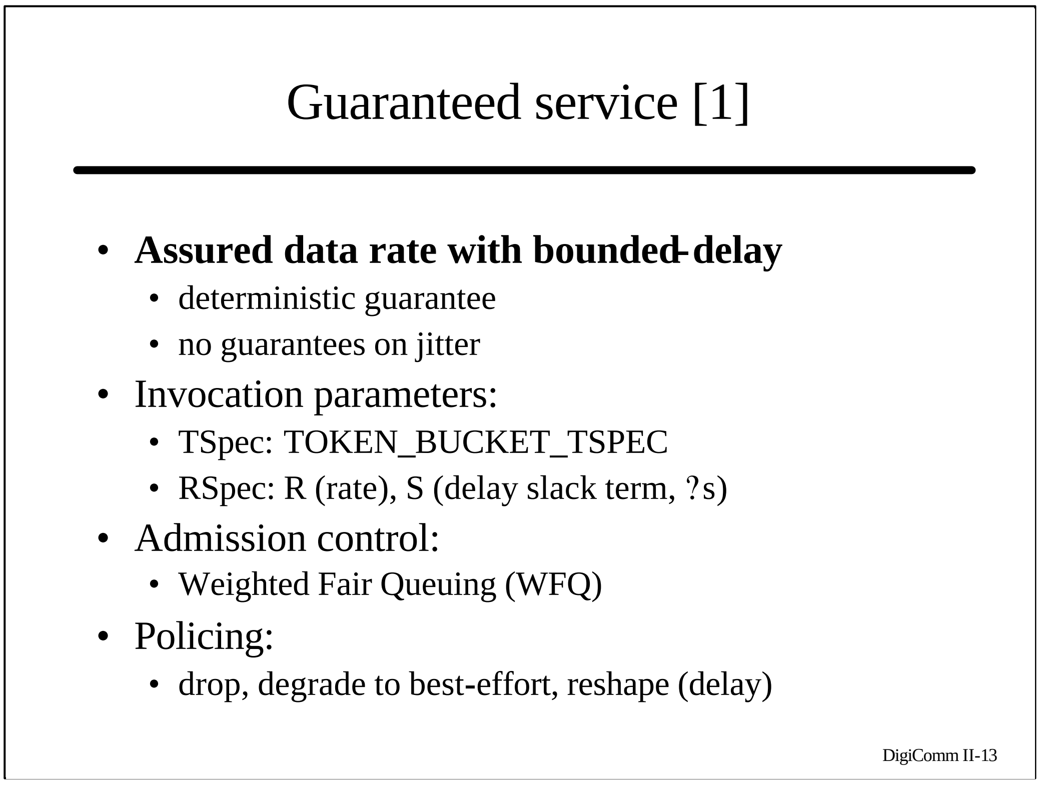
DigiComm II-12

The **controlled-load** service definition [RFC2211] specifies that network elements supporting this service should provide a service that is no worse than a best-effort service that would be seen at that network element under unloaded (lightly-loaded) conditions.

To invoke the service, the TSpec must be specified and a RSpec is not required.

It is suggested that Class-Based Queuing (CBQ) could be used to implement controlled-load service in network elements, e.g. with two classes of service, **priority** for the controlled-load packets and a separate class for normal best-effort packets.

Policing mechanisms are not specified/defined, but it is suggested that non-conformant packets should be degraded to best-effort.



Guaranteed service [1]

- **Assured data rate with bounded-delay**
  - deterministic guarantee
  - no guarantees on jitter
- Invocation parameters:
  - TSpec: TOKEN_BUCKET_TSPEC
  - RSpec: R (rate), S (delay slack term, ?s)
- Admission control:
  - Weighted Fair Queuing (WFQ)
- Policing:
  - drop, degrade to best-effort, reshape (delay)

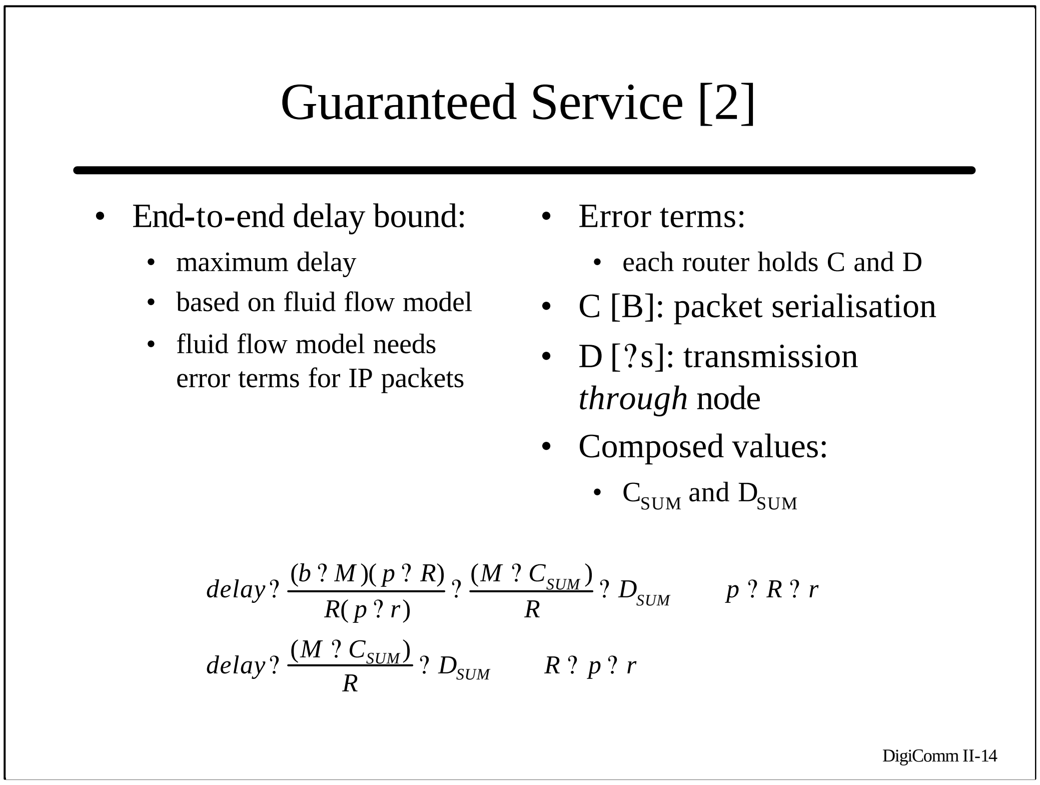
DigiComm II-13

The **guaranteed** service definition [RFC2212] specifies the network elements should treat packets so that there is an assured data rate and all packets have a bounded overall delay. However, the service makes no commitment on jitter (packet inter-arrival delay).

The invocation of the service requires a TSpec and a RSpec. The RSpec contains two parameters, R a required service rate, and S a slack-term for the delay bound. R must be greater than or equal to r (the rate defined in the TSpec token-bucket). S must be non-negative. Defining a bigger value for R helps to decrease the overall path delay. Defining a bigger value for S makes it more likely that the reservation request will succeed, but may result in a larger end-to-end delay. R is used as a suggestion and larger values of S may require routers to use a value lower than R for the reservation. The exact use of R and S are given in [RFC2212].

The suggested admission control mechanism for this service is Weighted Fair Queuing (WFQ) where the weight assignments will be a function of the required rate, R.

The suggested policing function takes one of two forms. A simple policing function (the suggested default) is for non-conformant packets to be dropped or degraded to best-effort. A more complex policing function take the form of reshaping the flow/session by delaying packets so that they conform to the requested parameters.



Guaranteed Service [2]

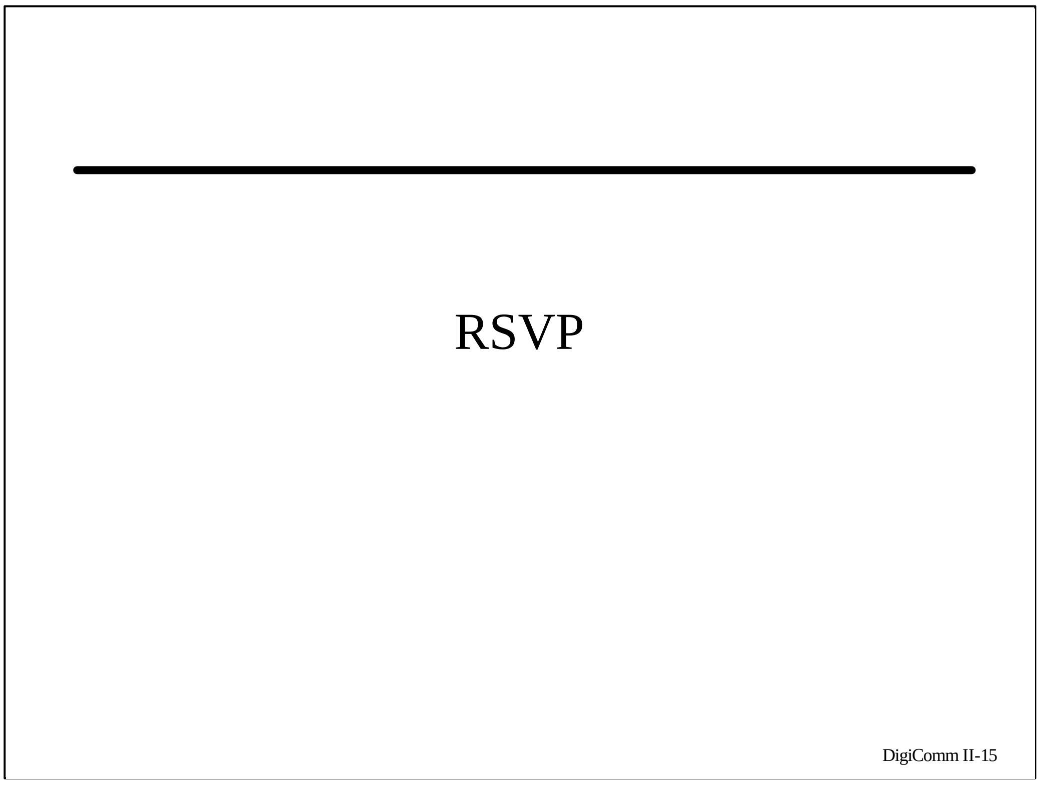
- End-to-end delay bound:
  - maximum delay
  - based on fluid flow model
  - fluid flow model needs error terms for IP packets
- Error terms:
  - each router holds C and D
- C [B]: packet serialisation
- D [?s]: transmission *through* node
- Composed values:
  - C_{SUM} and D_{SUM}

$$delay ? \frac{(b ? M)(p ? R)}{R(p ? r)} ? \frac{(M ? C_{SUM})}{R} ? D_{SUM} \qquad p ? R ? r$$

$$delay ? \frac{(M ? C_{SUM})}{R} ? D_{SUM} \qquad R ? p ? r$$

DigiComm II-14

Wit the Guaranteed Service, it is possible to evaluate the exact end-to-end delay bound from the TSpec and RSpec parameters. The equitions above show how to evaluate the delay, using the TSpec and RSpec parameters discussed earlier. There are also two as well as two new values, C_{SUM} and D_{SUM}. The equations above are based on a fluid flow model, and C_{SUM} and D_{SUM} provide the error terms required to correct for the effect of IP packets being of a size that deviates from the fluid flow model. C_{SUM} and D_{SUM} are discovered using RSVP, and constructed from individual values of C and D held at routers. Full details are given in [RFC2212].



RSVP

INTSERV: RSVP [1]

- Provides signalling:
  - user-to-network
  - network-to-network
- Traffic information – *FlowSpec*:
  - *TSpec*
  - sent through network
  - *AdSpec* (optional)
- Receiver confirms reservation:
  - uni-directional reservation

DigiComm II-16

RSVP is a signalling protocol that provides the service invocation interface for applications. The messages are sent between applications, but are acted upon and modified by the network elements en-route, so RSVP provides both user-to-network and network-to-network signalling. Special RSVP message carry *TSpec* and *RSpec* messages that are seen by (INTSERV aware) network elements along the network path as well as by the flow recipients.

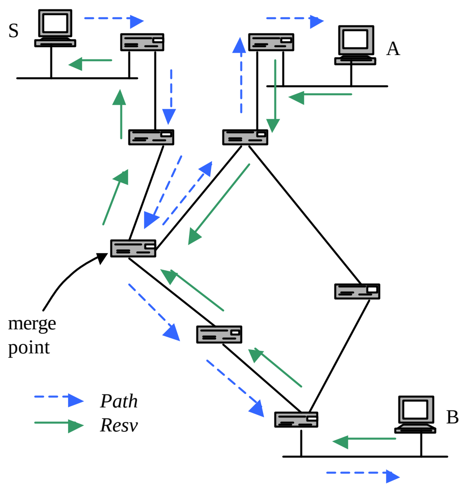
The reservation request consists of a *FlowSpec* identifying the traffic characteristics and service-level required. One part of the *FlowSpec* is a *TSpec*, a description of the traffic characteristic required for the reservation. So it is possible for the same traffic characteristic to be used with different service levels. This difference in QoS service-level could, for example, act as a way for offering cost differentials on the use of a particular application or service.

RSVP can be used to set-up resource reservations for multicast as well as unicast flows. The reservations are unidirectional and in fact it is the receiving application that actually confirms the reservation, i.e. this is a receiver-oriented reservation protocol. The receiver may also be made aware of composed parameter values along the route if an (optional) *AdSpec* is present within the *FlowSpec* transmitted from the sender.

Note that RSVP is a **general** QoS signalling protocol specified in [RFC2205]. For use in a particular QoS architecture additional specification is required. In the case of INTSERV, the additional specification is provided in [RFC2210].

INTSERV: RSVP [2]

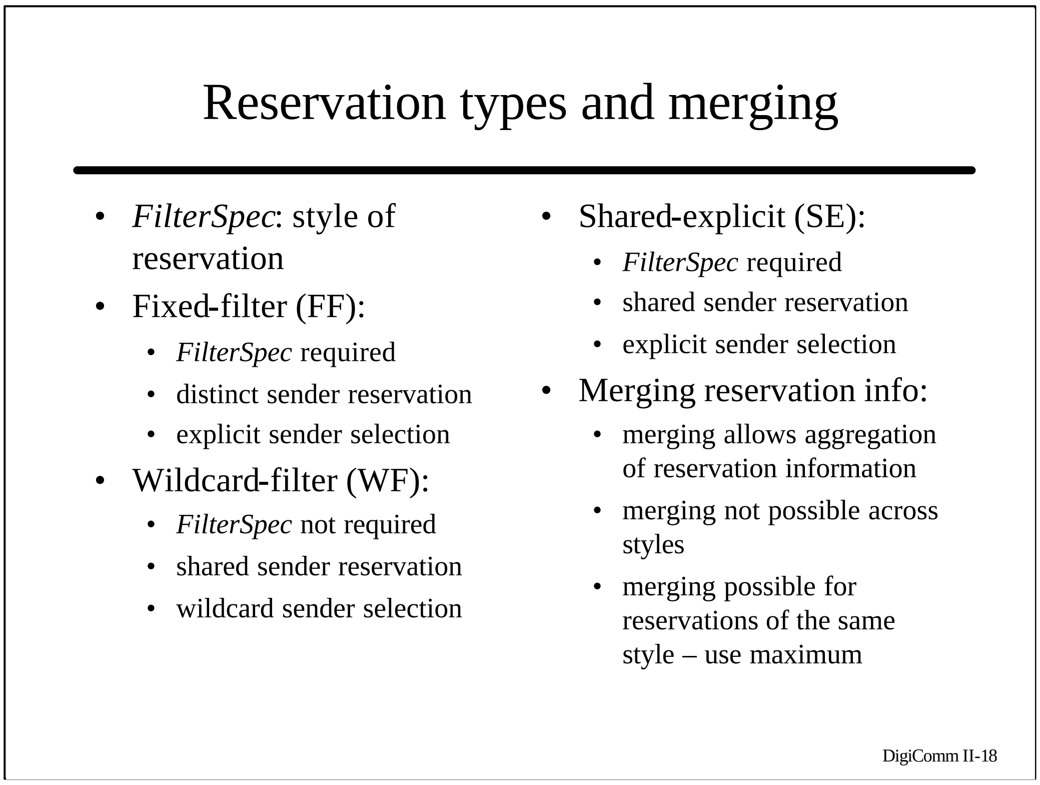
- Two-pass, with soft-state:
  - sender: *Path* message
    - NEs hold **soft-state** until *Resv*, *PathTear* or time-out
  - receiver(s): *Resv* message - TSpec (+RSpec)
  - sender: *PathTear*
  - receiver(s): *ResvTear*
  - soft-state refreshed using *Path* and *Resv*
- Composed QoS params:
  - *AdSpec* for path



To make a resource reservation, an appropriate *FlowSpec* is used along with session IP destination address, the protocol number in the IP packet and – optionally – the destination port number in the service invocation. The reservation procedure is as follows. The sender transmits a *Path* message advertising the session QoS requirements towards the destination IP address. All RSVP routers forwarding the *Path* message hold **soft-state** – information about the resource reservation required – until one of the following happens: a *PathTear* is sent from the sender cancelling the reservation, a *Resv* message is transmitted from a receiver effectively confirming the reservation, or the soft-state times-out. A *Resv* message from a receiver is sent back along the same route as the *Path* message, establishing the reservation and then the application starts sending data packets. *Path* and *Resv* messages are sent by the sender and receiver, respectively, during the lifetime of the session to refresh the soft-state and maintain the reservation. A *PathTear* or *ResvTear* message explicitly tears down the reservation and allows resources to be freed. It is possible for the reservation to be changed dynamically during the lifetime of the session. RSVP can be used for unicast or multicast sessions. (It is assumed that routes are symmetrical and relatively stable, but this is not always true in the wide area.)

As part of the *Path* message, an *AdSpec* data structure may also be sent in a **one pass with advertising (OPWA)** that allow network elements along the path to indicate to the receiver the **composed** (combined) QoS parameter values along the path based on **local** QoS capabilities at each network element. The local and composed capabilities are reported as QoS parameters for each service definition.

Where multicast communication is involved for the same flow, it is possible for a router to effectively merge two reservations instead of making two separate reservations.



Reservation types and merging

- *FilterSpec*: style of reservation
- Fixed-filter (FF):
  - *FilterSpec* required
  - distinct sender reservation
  - explicit sender selection
- Wildcard-filter (WF):
  - *FilterSpec* not required
  - shared sender reservation
  - wildcard sender selection
- Shared-explicit (SE):
  - *FilterSpec* required
  - shared sender reservation
  - explicit sender selection
- Merging reservation info:
  - merging allows aggregation of reservation information
  - merging not possible across styles
  - merging possible for reservations of the same style – use maximum

DigiComm II-18

There are three reservation styles that are permitted with RSVP/INTSERV.

• **fixed-filter (FF)** style: this style sets up a distinct reservation per sender that requires and specifies explicitly the set of sender who can make use of this reservation specification.

• **wildcard-filter (WF)** style: allows a shared reservation for senders, but the senders are not explicitly specified.

• **shared explicit (SE)** style: allows the reservation to be shared amongst an explicitly specified list of senders.

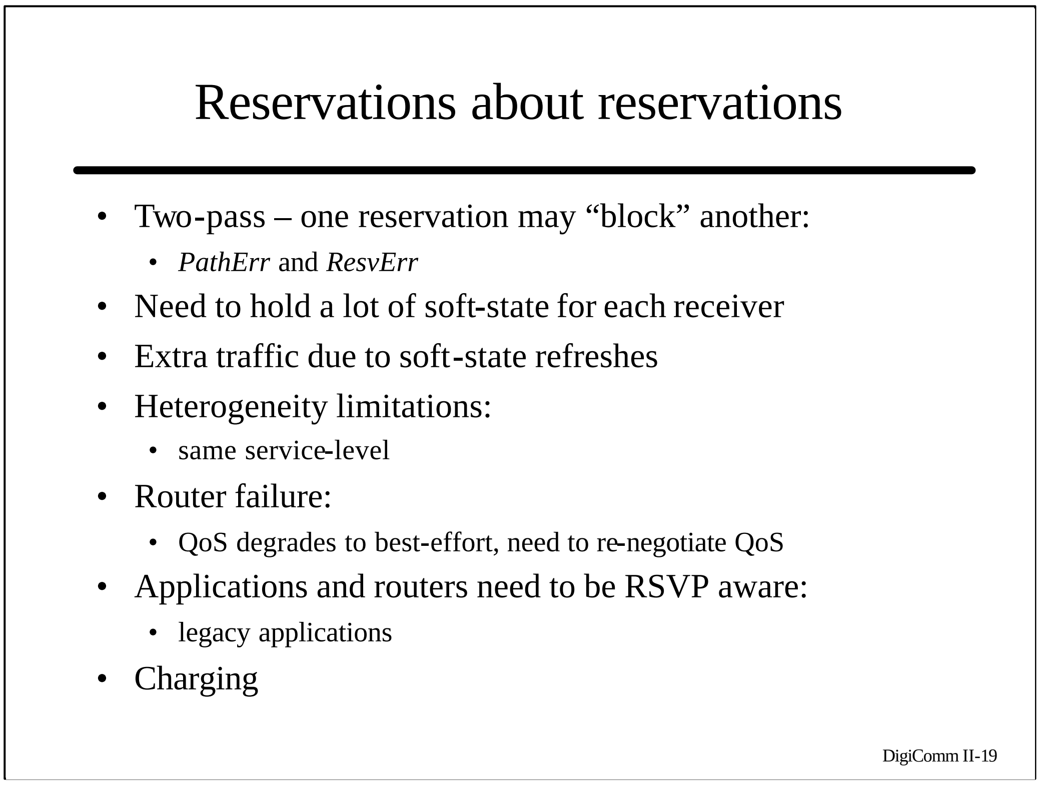
Information about the list of senders for FF and SE is carried in a *FilterSpec* data structure that forms part of the *Resv* message provided by the sender.

Its is possible for the RSVP routers to merge reservations of the same style. This is effectively to allow router to pass upstream a single reservation that is a maximum of the incoming reservations. This is specifically for multicast, where many flows for the same group are merged.

FF would typically be used for unicast communication only

WF would be used for an open conference, where the number of senders and who they will be is not known *a priori*. It would be expected that only one person would be speaking at a time, and perhaps the reservation would be enough for two speakers just in case two people did start to speak at once.

SE would be for a similar situation to WF but the conference would be closed, with the senders known before hand and listed in the FilterSpec.



Reservations about reservations

- Two-pass – one reservation may “block” another:
  - *PathErr* and *ResvErr*
- Need to hold a lot of soft-state for each receiver
- Extra traffic due to soft-state refreshes
- Heterogeneity limitations:
  - same service-level
- Router failure:
  - QoS degrades to best-effort, need to re-negotiate QoS
- Applications and routers need to be RSVP aware:
  - legacy applications
- Charging

DigiComm II-19

We summarise the main problems with RSVP below:

1. Intuitively, we can see that in a network with limited resources, which are heavily utilised (e.g. the Internet), it is likely larger reservations are probably less likely to succeed that smaller reservations. During reservation establishment if the first pass of each of two separate reservation requests are sent through the same network element, where one request is a “super-set” of the other, the lesser one may be rejected (depending on the resources available), even if the greater one eventually fails to complete (of course it is possible to re-try).

2. If the first pass does succeed, the router must then hold a considerable amount of state for each receiver that wants to join the flow (e.g. in a multicast conference)

3. The routers must communicate with receivers to refresh soft-state, generating extra traffic, otherwise the reservation will time out

4. Complete heterogeneity is not supported, i.e. in a conference everyone must share the same service-level (e.g. guaranteed or controlled-load), though heterogeneity within the service-level is supported

5. If there are router failures along the path of the reservation, this results in IP route changes, so the RSVP reservation fails and the communication carries on at best-effort service, with the other routers still holding the original reservation until an explicit tear-down or the reservation times out or the reservation can be re-established along the new path

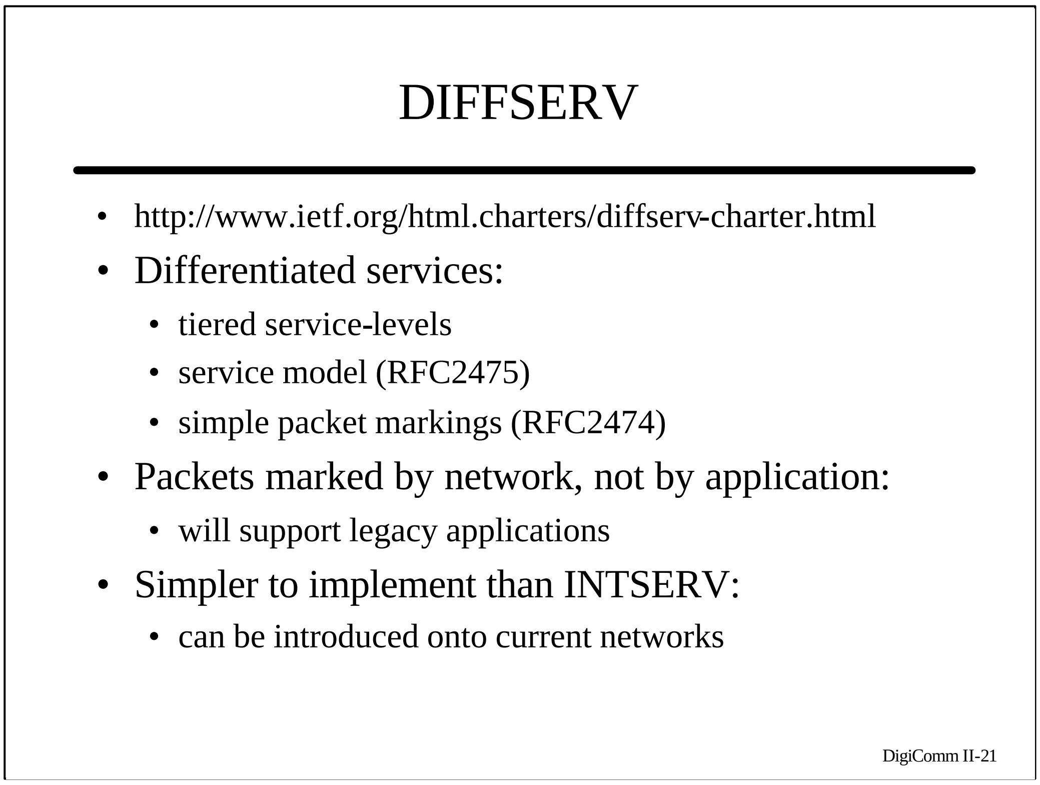
6. The applications must be made RSVP aware, which is a non-trivial goal to realise for the many current and legacy applications that already exist, including multimedia applications with QoS sensitive flows

Resource reservation could be expensive on router resources and adaptation capability is still required within the application to cope with reservation failures or lack of end-to-end resource reservation capability. Indeed, RSVP is now recommended for use only in restricted network environments [RFC2208].

Additionally, there is as yet no agreement as to how to charge for end-to-end QoS guarantees that span the networks of multiple administrations, e.g. across multiple ISPs.

DIFFSERV

DigiComm II-20



DIFFSERV

- http://www.ietf.org/html.charters/diffserv-charter.html
- Differentiated services:
  - tiered service-levels
  - service model (RFC2475)
  - simple packet markings (RFC2474)
- Packets marked by network, not by application:
  - will support legacy applications
- Simpler to implement than INTSERV:
  - can be introduced onto current networks

DigiComm II-21

Concerns about resource reservation have directed the Internet community to consider alternatives; specifically **differentiated services**. In fact the IETF DIFFSERV WG was spawned directly from the INTSERV WG.

This is a relatively new IETF WG and most of the work within this group is currently at the stage of discussion and the formulation of a framework and architecture for the DIFFSERV work.

The IETF charter for the workgroup is:

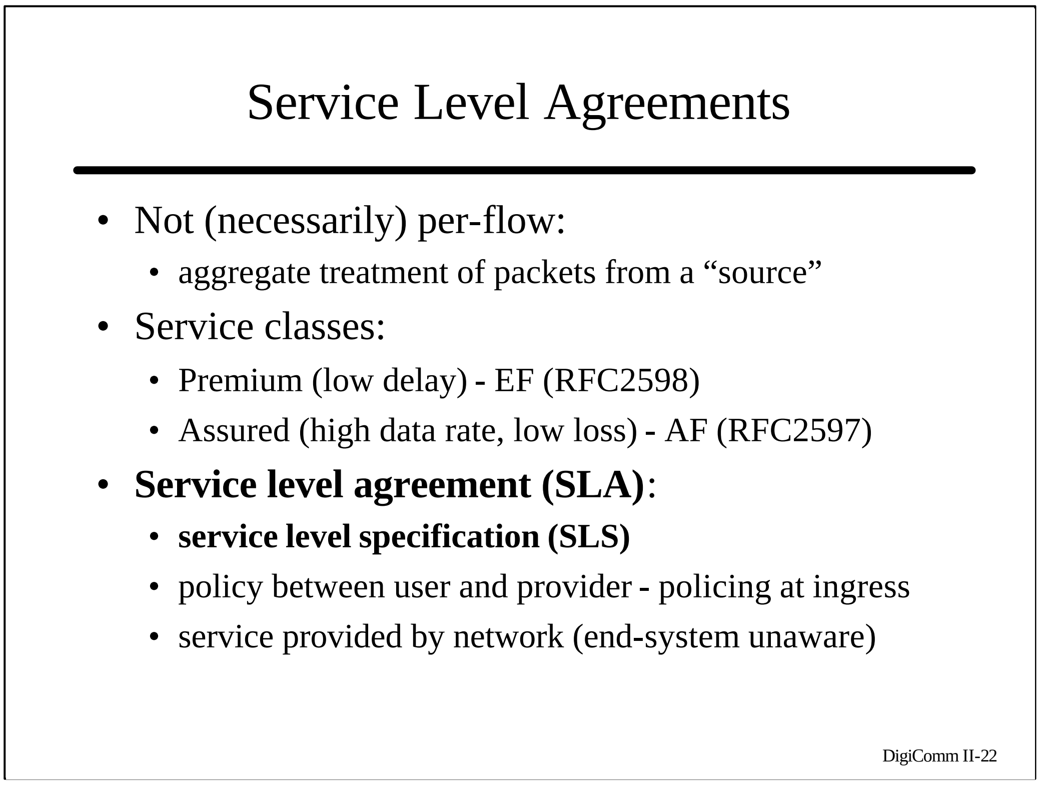
http://www.ietf.org/html.charters/diffserv-charter.html

Two RFC documents have been produced. RFC2474 describes special values to be used for the IPv4 ToS field or IPv6 traffic-class field when DIFFSERV is in use. RFC2475 describes the DIFFSERV architecture.

DIFFSERV hopes to offer a relatively simple, coarse-grained QoS mechanism that can be deployed in networks without needing to change the operation of the end-system applications. The QoS mechanism is based around marking packets with a small-fixed bit-pattern, which maps to certain handling and forwarding criteria at each hop. The WG seeks to identify a common set of such per-hop handling behaviours as well as packet markings to identify these behaviours.

This is a much coarser granularity of service, but reflects a well understood service model used in other commercial areas. The DIFFSERV model is different to INTSERV. A key distinction of the DIFFSERV model is that it is geared to a business model of operation, based on administrative bounds, with services allocated to users or user groups.

The DIFFSERV mechanisms should be simpler to implement than INTSERV mechanisms and will allow some QoS control for legacy applications that are not QoS aware.



Service Level Agreements

- Not (necessarily) per-flow:
  - aggregate treatment of packets from a “source”
- Service classes:
  - Premium (low delay) - EF (RFC2598)
  - Assured (high data rate, low loss) - AF (RFC2597)
- **Service level agreement (SLA)**:
  - **service level specification (SLS)**
  - policy between user and provider - policing at ingress
  - service provided by network (end-system unaware)

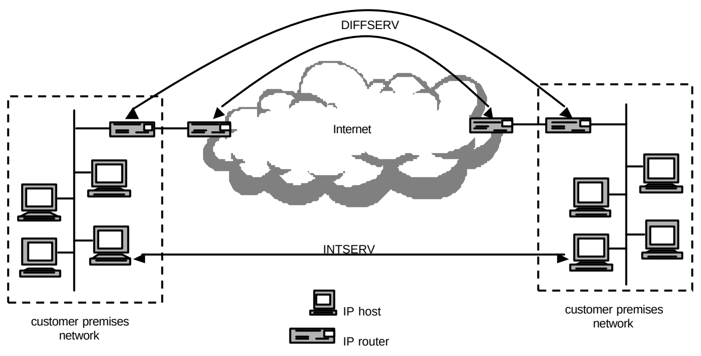
DigiComm II-22

Whereas RSVP can act on a per-flow basis, the DIFFSERV classes may be used by many flows. Any packets within the same class must share resources with all other packets in that class, e.g. a particular organisation could request a Premium (low delay service provided using **Expedited Forwarding**) quality with an Assured (low loss, using **Assured Forwarding** with different drop precedence assignments) service-level for all their packets at a given data rate from their provider.

The exact nature of the packet handling will be based on a policy and **Service Level Specification (SLS)** that forms part of a **Service Level Agreement (SLA)** between user and provider. The policy could be applied to all the traffic from a single user (or user group), and could be set up when subscription to the service is requested, or on a configurable profile basis. The policy implemented by the SLA may include issues other than QoS that must be met, e.g. security, time-of-day constraints, etc.

The DIFFSERV mechanisms would typically be implemented within the network itself, without requiring runtime interaction from the end-system or the user, so are particularly attractive as a means of setting up tiered services, each with a different price to the customer.

Scope of DIFFSERV

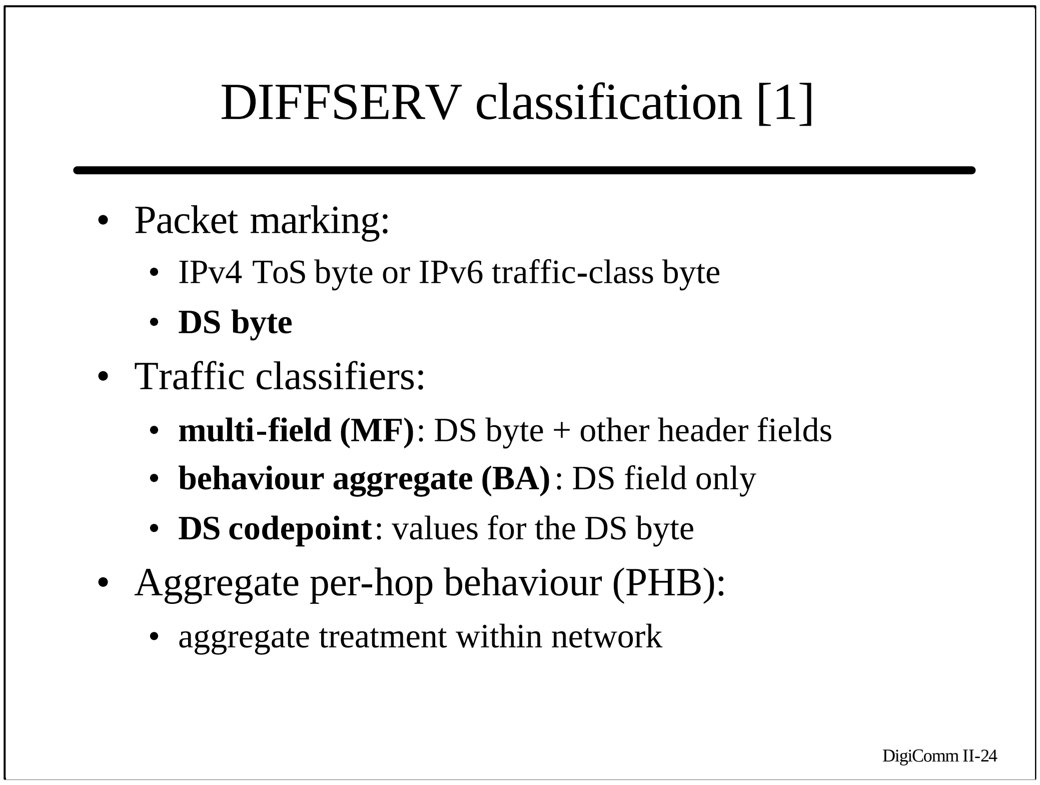


DigiComm II-23

The DIFFSERV-capable routers could be at the edge of the customer network or part of the provider's network. If the DIFFSERV-marking is performed within the customer network, then policing is required at the ingress router at the provider network in order to ensure that customer does not try to use more resources than allowed by the SLA.

The INTSERV mechanism seeks to introduce well-defined, end-to-end, per-flow QoS guarantees by use of a sophisticated signalling procedure. The DIFFSERV work seeks to provide a "virtual pipe" with given properties in which the user may require adaptation capability or further traffic control if there are multiple flows competing for the same "virtual pipe" capacity.

Additionally, the DIFFSERV architecture means that different instances of the same application throughout the Internet could receive different QoS, as different users may have different SLAs with their subscriber. So the application needs to be dynamically adaptable.



DIFFSERV classification [1]

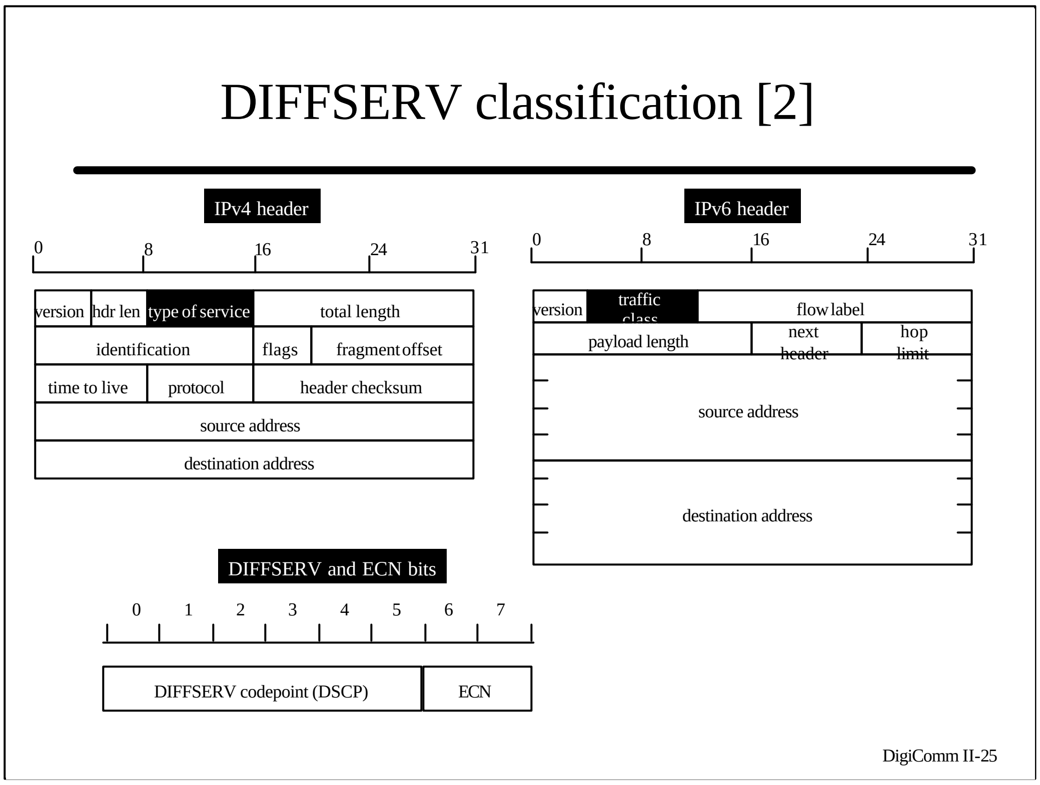
- Packet marking:
  - IPv4 ToS byte or IPv6 traffic-class byte
  - **DS byte**
- Traffic classifiers:
  - **multi-field (MF)**: DS byte + other header fields
  - **behaviour aggregate (BA)**: DS field only
  - **DS codepoint**: values for the DS byte
- Aggregate per-hop behaviour (PHB):
  - aggregate treatment within network

DigiComm II-24

The DIFFSERV work is aimed at providing a way of setting up QoS using policy statements that form part of a service level agreement between service user and service provider. The policy may use several packet header fields to classify the packet, but the classification marking can also be a simple identifier – currently a single byte, the **DS (differentiated services) byte** – within the packet header. The **DS (differentiated services) byte** will be used in place of the ToS (Type of Service) field in IPv4 packets or the traffic-class field in IPv6 packets. The DS byte will have the same syntax and semantics in both IPv4 and IPv6. There are likely to be some global values – **DS codepoints** – agreed for the DS field within the IETF but the intention is that the exact policy governing the interpretation of the DS codepoints and the handling of the packets is subject to some locally agreed SLA. SLAs could exist between customer and Internet Service Provider (ISP) as well as between ISPs. The DS codepoints are used to identify packets that should have the same aggregate **per-hop behaviour (PHB)** with respect to how they are treated by individual network elements within the network. The PHB definitions and the DS codepoints used may differ between ISPs, so there will be need for translation mechanisms between ISPs.

A traffic classifier selects packets based either on the on DS codepoint or on some (policy-based) combination header fields from the packet header and directs them to an appropriate traffic conditioner. When the DS codepoint is used to classify traffic, the classifier is called a **Behaviour Aggregate (BA)** classifier. When other packet header fields are used we have a **Multi-Field (MF)** classifier. And MF classifier may use information such as the port numbers, IP addresses protocol types, as well as the DS byte to make classification decisions.

Although there will be scope for changes to the SLA by agreement between customer and provider, the kind of dynamic, flexible, host-to-host resource reservation that is possible with the INTSERV model using RSVP is not envisaged for DIFFSERV.



DIFFSERV classification [2]

This is the usage proposed by RFC2474 for the ToS (IPv4) and traffic class (IPv6) byte. For bits 6 and 7, marked "currently unused", RFC2481 proposes they be used to provide explicit congestion notification (ECN) at the IP-level. This would allow DIFFSERV and ECN to be used together, the former to provide coarse-grained (class-based) QoS and the latter to provide congestion control signalling, by simply re-using an existing field in the IP header.

DIFFSERV PHBs

- Specify rate/delay in SLS
- **Expedited Forwarding (EF)** (RFC2598):
  - virtual leased line (VLL) service
  - data rate specified in SLS
  - low delay, low jitter, low loss
- **Assured Forwarding (AF)** (RFC2597):
  - 4 classes (1-4)
  - 3 levels of drop precedence per class (1-3)
  - AF11 - “best”, AF43 - “worst”

DigiComm II-26

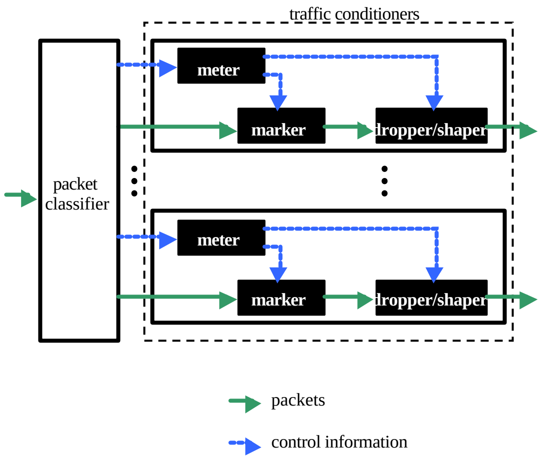
Recently (June 1999), the DIFFSERV WG have defined two PHBs, both of which are proposed standards. Both require that the SLS contain information such as delay, data rates (e.g. token bucket filters), and the scope over which the SLS applies (e.g. between ingress and all end-points, between ingress and specific end-points, etc.), as well as actions to take if the traffic is found to be violating the SLS.

The **Expedited Forwarding (EF)** PHB is used to provide a low loss, low delay, low jitter end-to-end service across DS domains. The service if provides is likened to that of a virtual leased line (VLL). Suggested implementation mechanisms include weighted round robin scheduling and class based queuing (CBQ). If simple priority queuing is used (the EF queue is always serviced before any other traffic) then the implementation must ensure that other traffic is not locked out (e.g. by using rate limiting via a token bucket filter). Violating traffic can be dropped. A single DS codepoint is defined for EF.

The **Assured Forwarding (AF)** PHB will allow a DS domain to provide different levels of assurance for forwarding of IP packets. Currently, 4 AF classes are defined with 3 drop precedence levels in each. An AF class mark is indicated by the lexeme *AF**cd*** where ***c*** is the AF class and ***d*** is the drop precedence within that class. An example usage is that each class represents a higher level of service (e.g. 1= platinum, 2=gold, 3=silver, 4=bronze), with low, medium and high (1, 2, 3 respectively) drop precedence levels in each class. So, AF11 would be the “best” AF mark and AF43 the “worst”. Implementation might be using weighted queuing/scheduling with violating traffic being dropped or re-marked to lower classes, higher drop precedence or best effort.

DIFFSERV traffic conditioning

- Traffic conditioners:
  - meter
  - marker
  - shaper/dropper
- Metering of traffic:
  - in-profile
  - out-of profile
- Re-marking:
  - new DS codepoint
- Shape/drop packets



DigiComm II-27

A **DS domain** contains **DS boundary nodes** at its edge and **DS interior nodes** within the domain. DS boundary nodes act as **traffic conditioners**. Traffic conditioners implement the **Traffic Conditioning Agreement (TCA)** part of a SLA. A schematic diagram showing how streams are treated is shown in .

Part of the SLA is the definition of a **traffic profile** for a packet stream. This may, for example, be specified as a token bucket, limiting the way that packets are transmitted into the DS domain. When packets in a stream from a user exceed the negotiated traffic profile, they are said to be **out-of-profile**, else packets are **in-profile**.

After passing through a classifier, information about the packet is passed to a meter that provides control information to other parts of the conditioner. This information includes whether or not the packet is in-profile or out-of-profile. Within the conditioner, the packets follow a path through a marking function and a policing function:

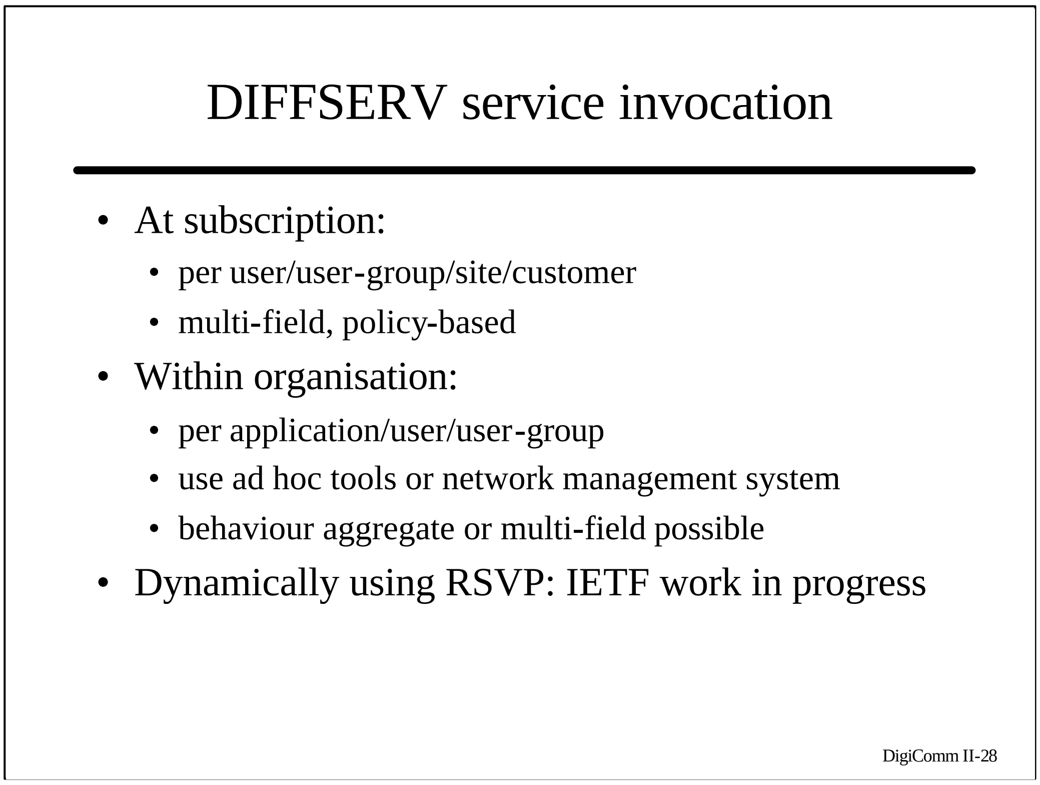
• **marker:** may change the DS codepoint of the packet – **re-mark** the packet

• **shaper:** delays out-of-profile packets in order to enforce the traffic profile for a stream

• **dropper:** drops out-of-profile packets in order to enforce the traffic profile for a stream

Note that a dropper can be implemented by using a shaper with the buffering reduced to zero packets (or a few packets).

DS interior nodes may perform limited BA traffic conditioning, but the intention is that the main traffic conditioning function is performed at the edges of DS domain (as the DS boundary nodes), close to where the packets enter the domain.



DIFFSERV service invocation

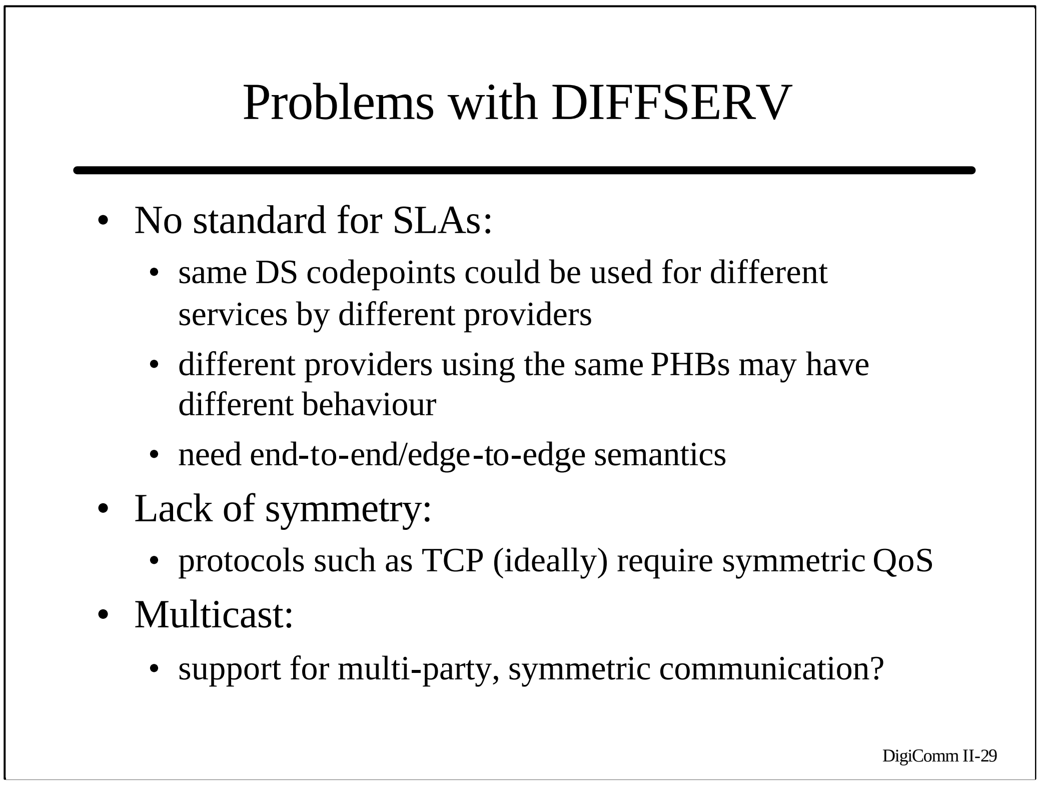
- At subscription:
  - per user/user-group/site/customer
  - multi-field, policy-based
- Within organisation:
  - per application/user/user-group
  - use ad hoc tools or network management system
  - behaviour aggregate or multi-field possible
- Dynamically using RSVP: IETF work in progress

DigiComm II-28

It is intended that the DIFFSERV work will offer a subscription-time mechanism for defining coarse-grained QoS requirements for an organisation. The exact nature of the service level agreement will be left as a matter of negotiation between user and provider. However, DIFFSERV offers an architecture and definitions that will allow an SLA to be defined. The policy for controlling traffic could be based on applications, individual users or user-groups.

It may even be possible to have control of traffic within an organisation, providing the network elements can be persuaded to be DIFFSERV aware. The network elements could be configured, for example, to control the amount of traffic form a particular application appearing on certain segments of the network, e.g. off-site WWW traffic.

It may even be possible to control or invoke SLAs more dynamically using RSVP (with suitable additional specifications) but this is currently work in progress.



Problems with DIFFSERV

- No standard for SLAs:
  - same DS codepoints could be used for different services by different providers
  - different providers using the same PHBs may have different behaviour
  - need end-to-end/edge-to-edge semantics
- Lack of symmetry:
  - protocols such as TCP (ideally) require symmetric QoS
- Multicast:
  - support for multi-party, symmetric communication?

DigiComm II-29

DIFFSERV is not without its own problems, however.

Firstly, there is a problem with service definitions. Only DS-codepoints have been defined, and not end-to-end semantics (though two standard track PHB documents do exit). This means that it will be possible for service providers to implement different services using the same DS codepoints. So, provider must co-operate and ensure that mappings between DS codepoints at network boundaries results in semantically correct service translation as packets go from one network to another.

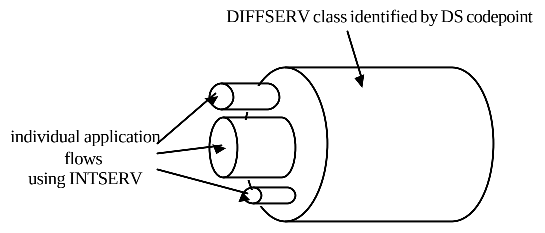
For the current standard-track PHB documents, it is possible that different provider may implement different behaviour across their networks for the same DS codepoints, though this is likely to be more so with AF than with EF. Edge-to-edge semantics are required, so that network boundaries can still be honoured but handling of packets is consistent.

Secondly, note that the SLA/SLS is between a user and their service provider. If that user accesses a server which is connected using a link that has a different, perhaps “worse”, SLA/SLS with its provider then our user would not see the service they expect when paying for the “better” service. This is because the return traffic from the server is treated differently – worse – than the initial request to the server. So, for whizzy web-browsing you need to ensure that the server site has a “good” SLA/SLS as well as getting a “good” SLA/SLS yourself. This lack of symmetry in DIFFSERV connectivity would affect protocols such as TCP which rely on a a two way exchange for reliability.

There is also the issue of support for multicast. DIFFSERV is not as dynamic invocation of services as is INTSERV/RSVP. DIFFSERV, at least currently, I based on the notion of a subscription. Work is in progress to allow dynamic establishment of SLAs/SLS using RSVP. However, many end-to-edn signalling issues remain unresolved.

INTSERV and DIFFSERV [1]

- Complimentary:
  - DIFFSERV: aggregate, per customer/user/user-group/application
  - INTSERV: per flow
- For example:
  - INTSERV reservations within DIFFSERV flows (work in progress)

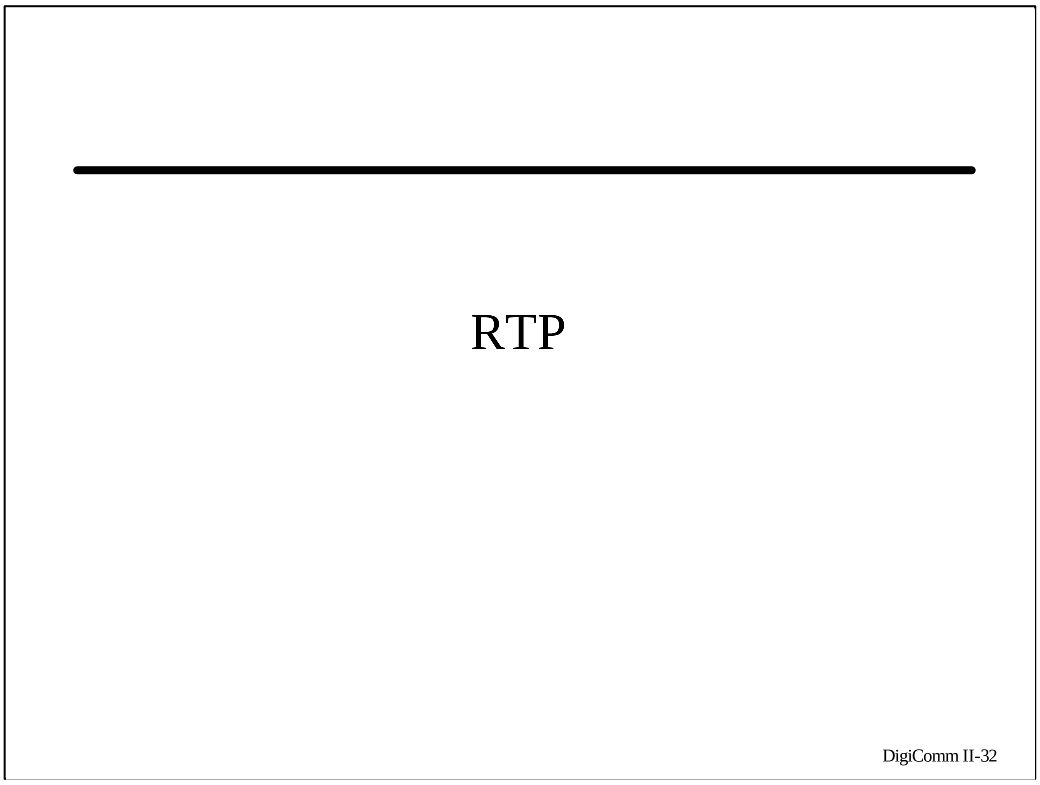

DIFFSERV class identified by DS codepoint
individual application flows using INTSERV

DigiComm II-30

The big gain with DIFFSERV is that the end-to-end signalling and the maintenance of per-flow soft-state within the routers that is required with RSVP is no longer required. This makes DIFFSERV easier to deploy and more scaleable than using RSVP and INTSERV services. However, this does not mean that INTSERV and DIFFSERV services are mutually exclusive. Indeed, it is likely that DIFFSERV SLAs will be set-up between customer and provider for general use, and then RSVP-based per-flow reservations may be used for certain applications as required, e.g. for instance an important video conference within an organisation.

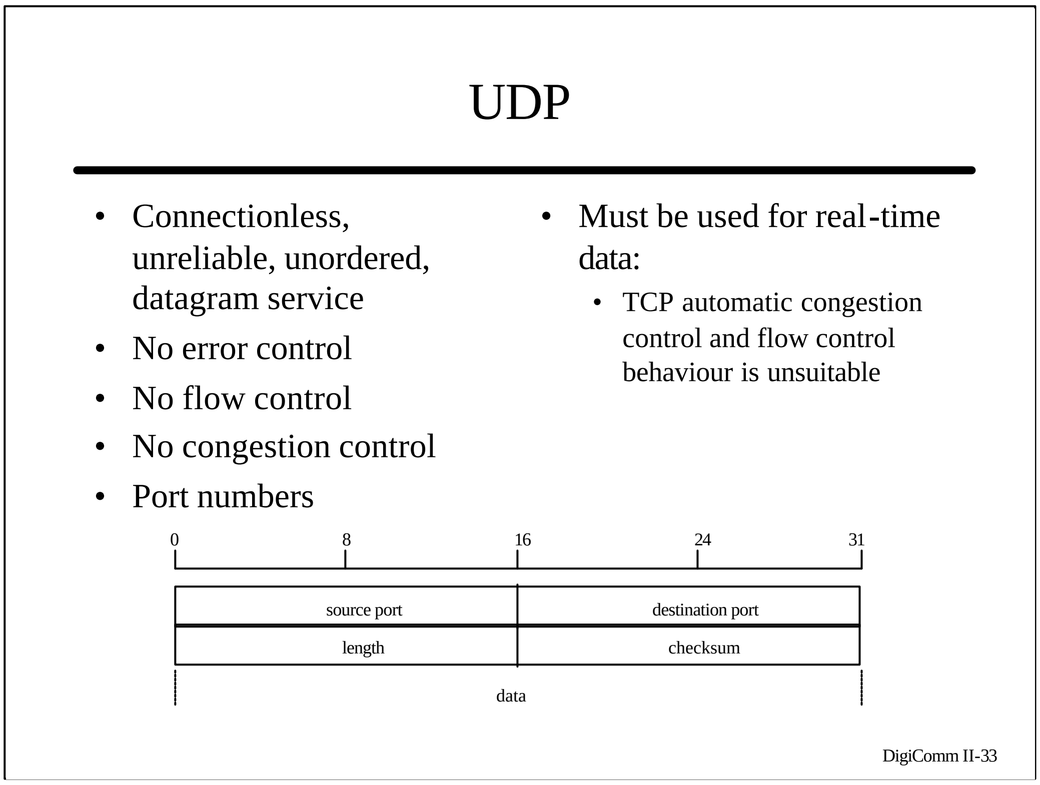
INTSERV and DIFFSERV [2]

	INTSERV	DIFFSERV
signalling	from application	network management, application
granularity	flow	flow, source, site (aggregate flows)
mechanism	destination address, protocol and port number	packet class (other mechanisms possible)
scope	end-to-end	between networks, end-to-end



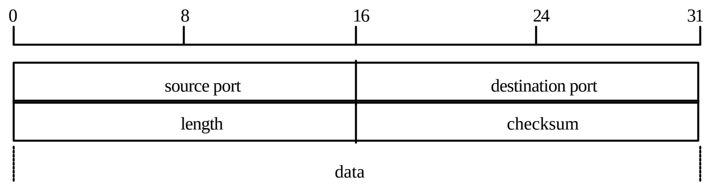
RTP

DigiComm II-32



UDP

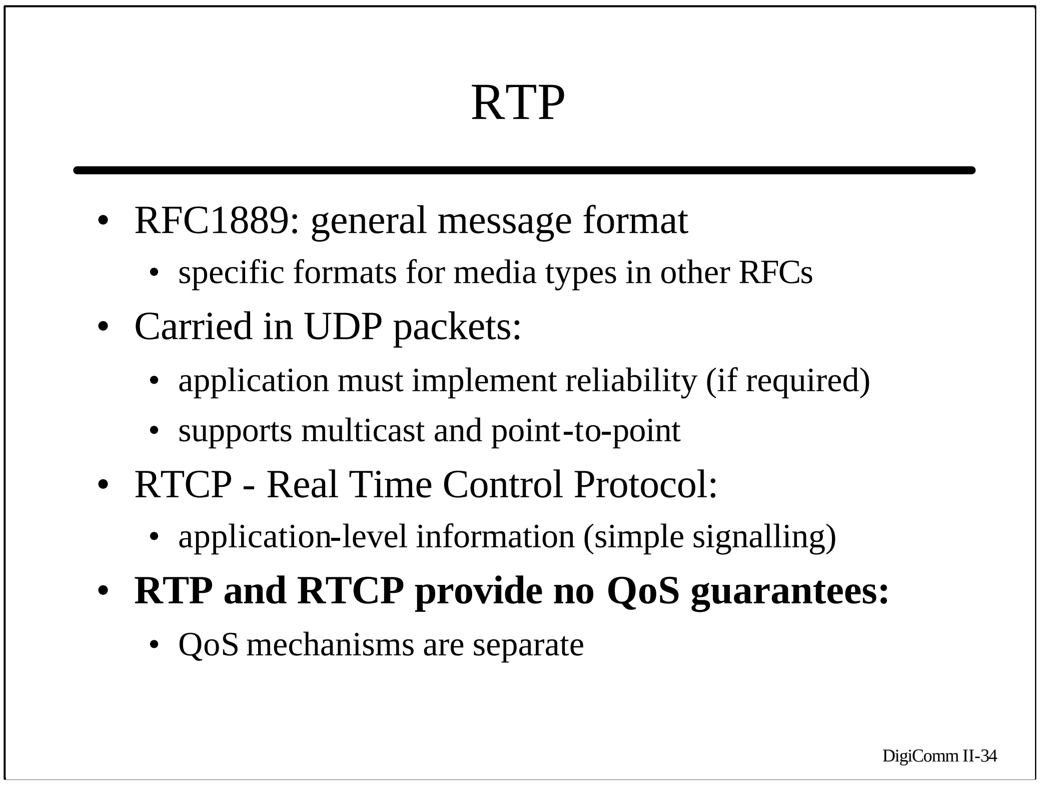
- Connectionless, unreliable, unordered, datagram service
- No error control
- No flow control
- No congestion control
- Port numbers
- Must be used for real-time data:
  - TCP automatic congestion control and flow control behaviour is unsuitable



DigiComm II-33

UDP provides an unreliable, connectionless datagram service. It does not guarantee delivery or ordering, and individual packets may be duplicated within the network. Exactly how "unreliable" the service is depends very much on the network environment. In a lightly loaded LAN, it is unlikely that you will observe much packet loss. Across a wide area backbone, however, there may be significant packet loss, especially over paths involving large numbers of routers or heavily loaded routes.

UDP is very simple to implement, and this is reflected in the packet header for UDP. The port number work in a similar way to those for TCP, identifying a local UDP end-point.



RTP

- RFC1889: general message format
  - specific formats for media types in other RFCs
- Carried in UDP packets:
  - application must implement reliability (if required)
  - supports multicast and point-to-point
- RTCP - Real Time Control Protocol:
  - application-level information (simple signalling)
- **RTP and RTCP provide no QoS guarantees:**
  - QoS mechanisms are separate

DigiComm II-34

The **Real time Transport Protocol (RTP)** is an Internet Proposed Standard and is widely used for multimedia applications (including voice and video) within the Internet community. Its use as the underlying transport mechanism for packetised voice and video is specified in H.323.

RTP carries "time-slices" of audio and video flows in UDP packets, with synchronisation information and application-specific identifiers, QoS parameter information and user information. RTP itself is a general mechanism and the are specific RTP usage **profiles** available for different media types, each described in their own RFC document, e.g.

• RFC2032 for H.261

• RFC2038 for MPEG1 and MPEG2

• RFC2190 for H.263

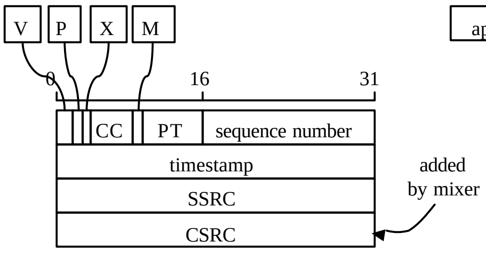
• RFC2198 for redundant (fault tolerant) audio

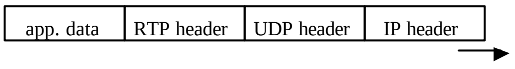
and many others. RTP is designed to support multicast and unicast communication.

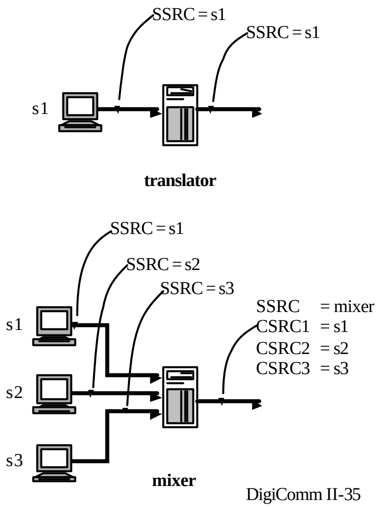
RTP has an associated with it a simple application-level signalling protocol, the **Real Time Control Protocol (RTCP)** that allows application using RTP to pass resource usage information, flow QoS parameters and other information between senders and receivers.

RTP and RTCP themselves do not provide QoS control or resources reservation - they are protocols that enable the transport of real-time media flows.

RTP header information







V	2-bits, version number (=2)
P	1-bit, indicates padding
X	1-bit, indicates extension header present
CC	4-bits, number of CSRCs (CSRC count)
M	1-bit, profile specific marker (defined elsewhere)
PT	7-bits, payload type, profile specific (defined elsewhere)
SSRC	synchronisation source
CSRC	contributing source

timestamp has profile/flow-specific units

DigiComm II-35

RFC1889, the RTP specification, defines some general header information that is used by all RTP applications.

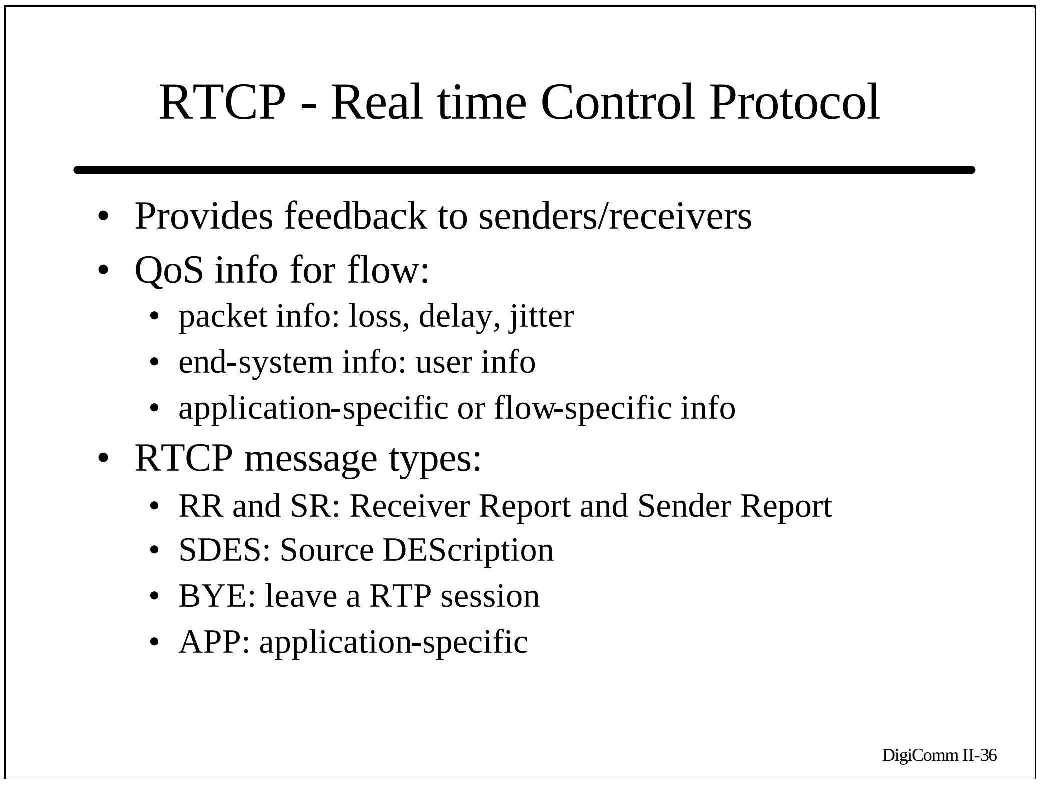
All RTP packets carry a sequence number to allow detection of loss and misordering at the receiver.

There is an application-specific timestamp indicates where in the flow this packet should be with respect to the rest of the flow. This allows synchronisation of the flow playback at the receiver, and also allows packets to be disregarded if they are delayed beyond the point that they have far exceeded their playout time.

RTP uses unique identifiers - SSRC (synchronisation source) and CSRC (contributing source) to identify originators of flows within and RTP session. Any ?IP address, SSRC? pair must be unique so that multiple flows from the same host can be distinguished. The SSRC is randomly generated.

An end-station will generate an SSRC to be carried in the RTP header. The packets in a flow may pass through a **translator** or a **mixer**. When passing through a translator, the flow may be altered, e.g. transcoded, but this is transparent to the receiver. When a flow passes through a mixer, the mixer may decide to merge and/or translate flows. When flows are merged (mixed) the mixer identifies itself as the SSRC, but also identifies the original sources of the mixed flows by putting their respective SSRC IDs into the CSRC list of the packet header. (A maximum of 15 flows can be mixed.)

Media specific header extensions are defined in the relevant RFC documents.



RTCP - Real time Control Protocol

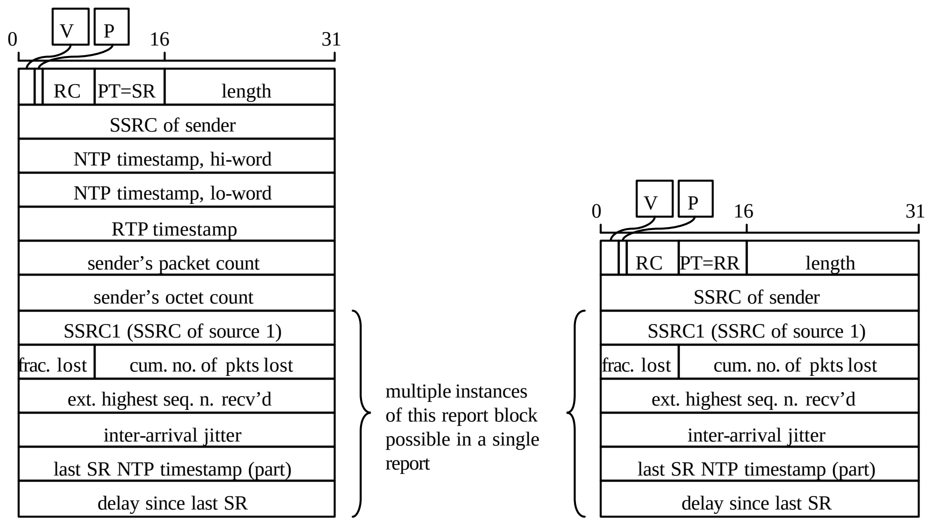
- Provides feedback to senders/receivers
- QoS info for flow:
  - packet info: loss, delay, jitter
  - end-system info: user info
  - application-specific or flow-specific info
- RTCP message types:
  - RR and SR: Receiver Report and Sender Report
  - SDES: Source DEScription
  - BYE: leave a RTP session
  - APP: application-specific

DigiComm II-36

RTCP provides simple information about the flow. Reports are sent by senders and receivers. The RTCP messages defined in RFC1889 carry information about the loss, delay and jitter for a flow, as well as some end-system user information. Additionally, application-specific information is defined for particular media -flows in he relevant RFC documents.

The generation of control message is controlled by an algorithm that seeks to limit the amount of RTCP traffic to around 5% the available network capacity.

SR and RR messages



DigiComm II-37

The Receiver Report and Sender Report are used to convey information about the flow throughout the lifetime of the flow.

The SSRC is used to identify the sender of the RR/SR and then the rest of the message consists of Report Blocks. Each report block identifies the source using an SSRC and the gives the following information for each:

• fraction of lost packets for the flow

• cumulative number of lost packets

• the last received sequence number, and also the number of times the sequence number has cycled (wrapped)

• estimate of the variance of inter-packet arrival time

• part of the last NTP timestamp sent in the SR as received by this SSRC

• the delay since the last SR was received

The SR also has:

• NTP timestamp

• RTP timestamp (flow-specific)

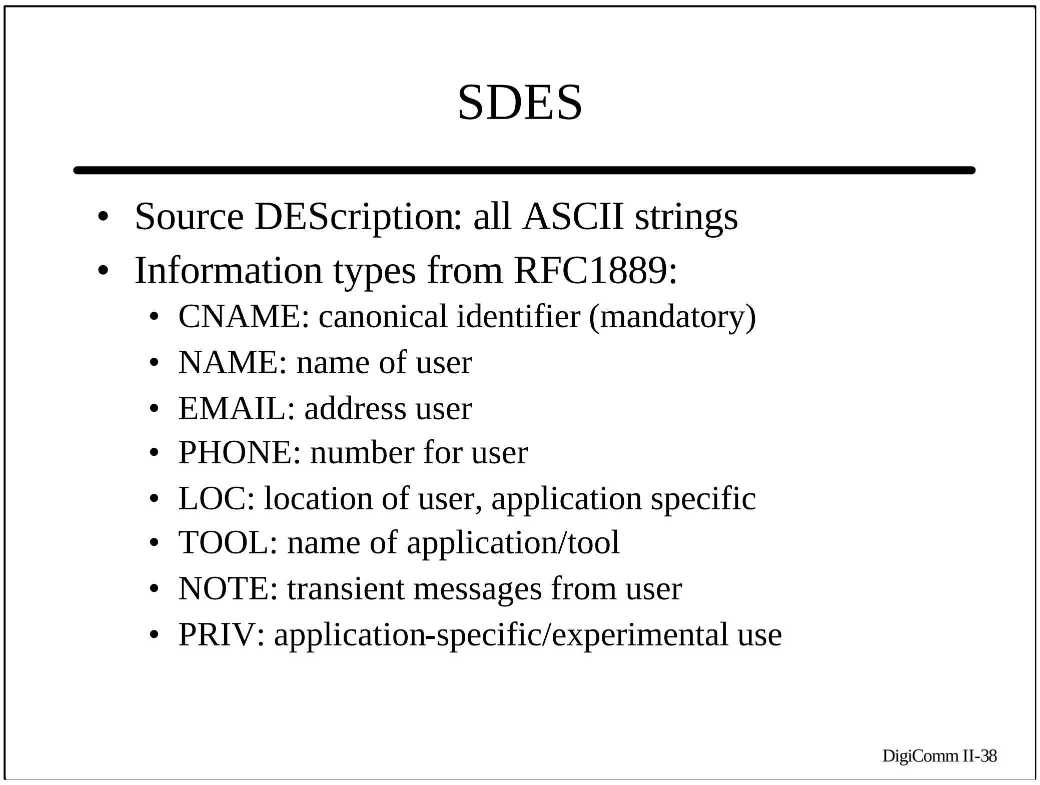
• sender’s packet count

• sender’s octet count

This information allows the applications to evaluate the QoS being received by particular flows from particular senders. This may allow the application to co-ordinate adjustments to the flow based on the QoS information.

NTP is the Network Time Protocol. The RTP timestamp provides application/flow specific timing information whilst the NTP timestamp provides a measure of global time.

Both the RR and SR can be extended with profile specific information.



SDES

- Source DEScription: all ASCII strings
- Information types from RFC1889:
  - CNAME: canonical identifier (mandatory)
  - NAME: name of user
  - EMAIL: address user
  - PHONE: number for user
  - LOC: location of user, application specific
  - TOOL: name of application/tool
  - NOTE: transient messages from user
  - PRIV: application-specific/experimental use

DigiComm II-38

SDES message are simple ASCII strings that contain information that is typically application-specific. RFC1889 defines 8 types that can be carried in the SDES message, most of which will have application-specific values:

• CNAME: this is the only mandatory type and is used to uniquely identify a participant in a conference. It is normally generated automatically by the application and usually takes the form: *user@host* (or just *host* on single user systems), e.g. saleem@darhu.cs.ucl.ac.uk

• NAME: the real name of the user (or any other identifying string, e.g. nickname, etc.)

• EMAIL: RFC822 e-mail address of the user, e.g. jon.crowcroft@cl.cam.ac.uk (this value could also be used for CNAME)

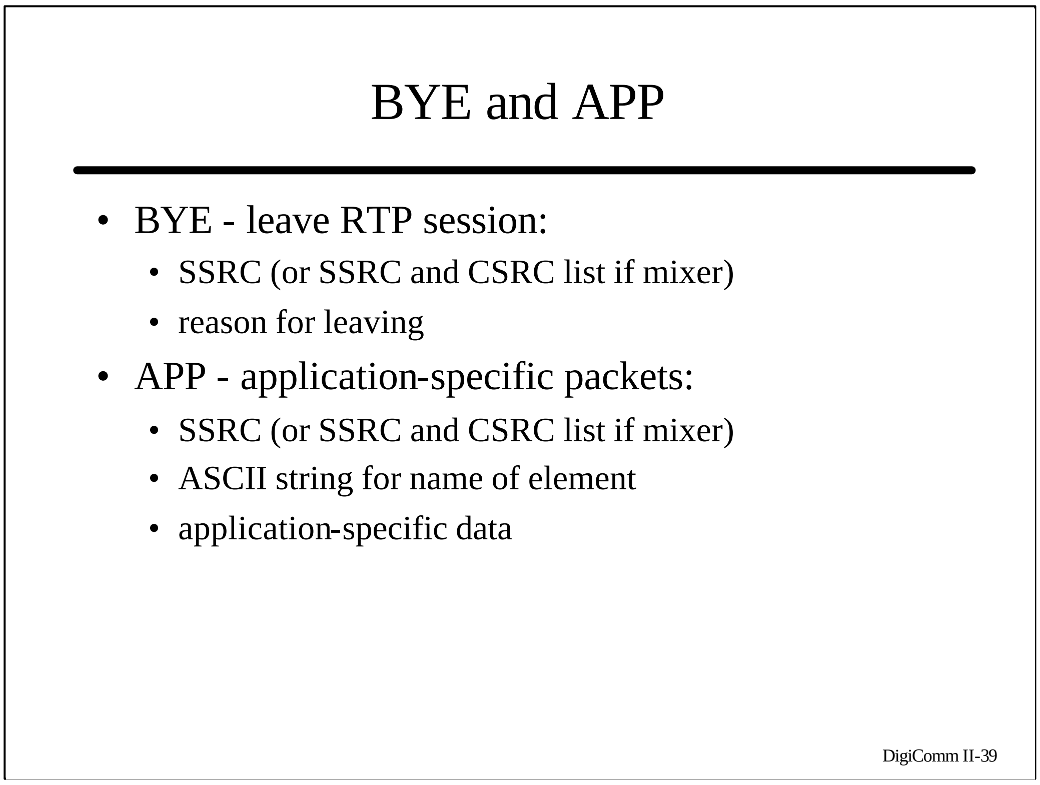
• PHONE: international format phone number, e.g. "+44 20 7679 3249"

• LOC: physical location of the user (application-specific detail required here)

• TOOL: identifies the name of the application/tool e.g. "blob-talk audio tool v42"

• NOTE: for transient message from the user, e.g. "out to lunch"

• PRIV: to allow application-specific SDES contents and for experimental use



BYE and APP

- BYE - leave RTP session:
  - SSRC (or SSRC and CSRC list if mixer)
  - reason for leaving
- APP - application-specific packets:
  - SSRC (or SSRC and CSRC list if mixer)
  - ASCII string for name of element
  - application-specific data

DigiComm II-39

The BYE message allows end-points to signal that they are leaving a session. The packet can contain a SSRC if sent by a single system or an SSRC and CSRC list if sent by a mixer. Optionally, a string giving the reason for leaving may be included.

If a mixer receives a BYE message, it should forward it unchanged. If the mixer itself shuts down, then it should send a BYE message with itself as the SSRC and CSRC for all its contributing sources.

The APP message is a mechanism that can be used for application specific messages. This mechanism is also intended for use in development and testing of a new media flow or application before making specific RTP/RTCP modifications that may be documented a separate RFC.

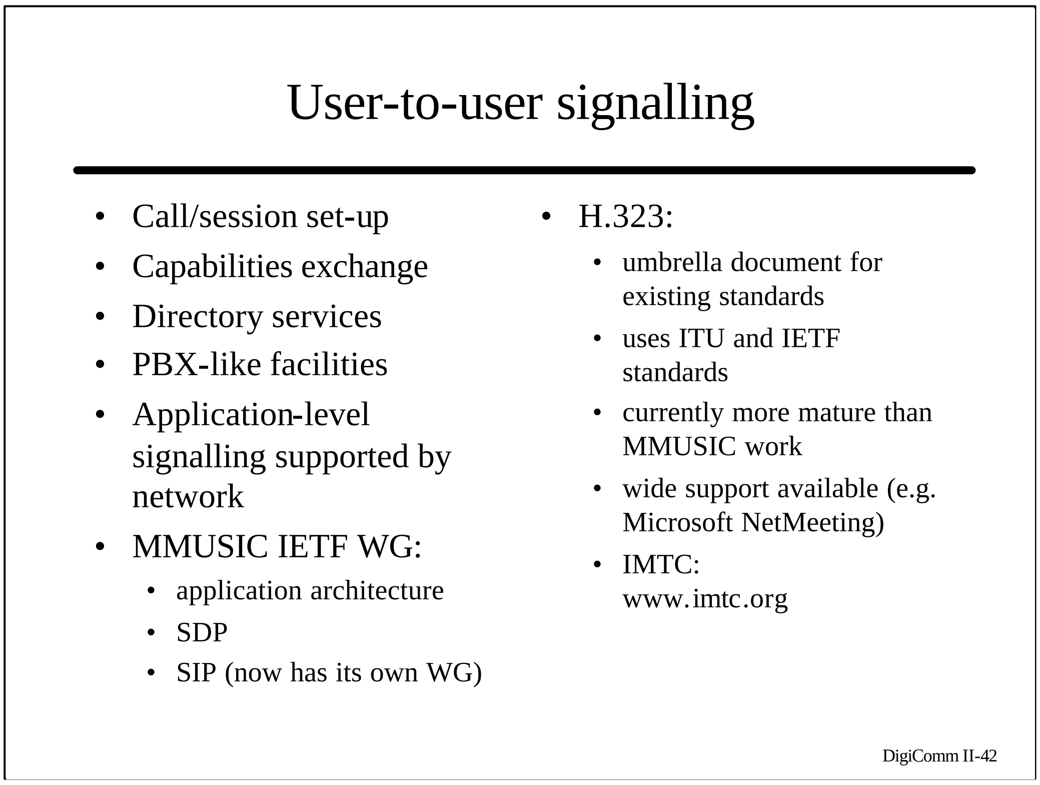
Application-level signalling

User-to-network

- Telco network:
  - common channel signalling (CCS)
  - separate data path and signalling path
  - equipment designed to handle data and signalling separate
- IP:
  - RSVP carried in IP packets along data path
  - scaling issues (RFC2208)
  - need aggregated signalling towards the core (use INTSERV with DIFFSERV?)

DigiComm II-41

Telco networks use **common channel signalling (CCS)**, which provides physically separate channels for signalling. Telco equipment is designed to have separate data paths and signalling paths. Signalling also allows the switching of channels to be aggregated. IP has none of these facilities, and these of signalling is relatively new to the IP world. While level-4 protocols such as TCP do have handshaking, and there are application-specific session information exchanges, these are all carried as IP packets along the same path that will eventually carry the data. This means that routers must look for signalling packets as they handle data, a function that slows down the processing of data packets. Also, signalling such as RSVP can not be aggregated in the same way as can telco signalling. Indeed we have already seen that [RFC2208] points out the scaling limitations of RSVP, as used in its current form. Perhaps the solution would be to use RSVP as an edge-system mechanism, and map flows into DIFFSERV pipes, e.g. map Guaranteed service-level request to EF PHB pipes, and Controlled-load service level to AF11 PHB pipes.



User-to-user signalling

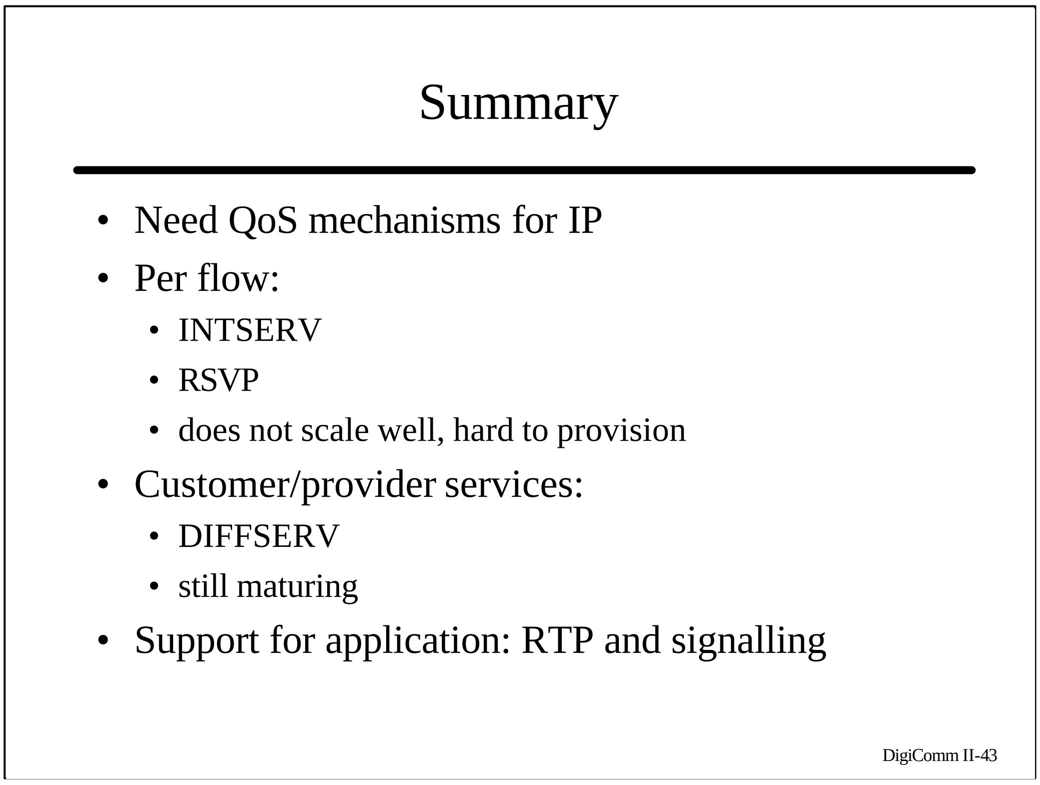
- Call/session set-up
- Capabilities exchange
- Directory services
- PBX-like facilities
- Application-level signalling supported by network
- MMUSIC IETF WG:
  - application architecture
  - SDP
  - SIP (now has its own WG)
- H.323:
  - umbrella document for existing standards
  - uses ITU and IETF standards
  - currently more mature than MMUSIC work
  - wide support available (e.g. Microsoft NetMeeting)
  - IMTC: www.imtc.org

DigiComm II-42

There is also a need for application-specific signalling in establishing multimedia sessions. The kind of information that is required is typically configuration and control information to allow a session to take place, e.g. multicast address, time and duration of session, audio and video profile to use, etc. Additional signalling mechanisms can be envisaged that allow capabilities negotiation (allowing terminals to establish negotiate use of various audio and video codecs), and directory services allowing location of users to be determined. Also, there is the desire to build in more traditional PBX-like functions into the software environment, such as call forwarding, call waiting etc. While this is application-level signalling, the transmission of the this signalling information may need to be supported by service providers for Internet-wide use, but of course as with telco PBXs, virtual private networks (VPNs) are possible.

Within the Internet community, the Multi-party Multimedia Session Control (MMUSIC) workgroup of the IETF is defining an architecture for multimedia applications as well as protocols for describing sessions (Session Description Protocol – SDP) and initiating calls or session (Session Initiation Protocol – SIP). SIP supports functions such as call waiting, call forwarding etc. SIP is designed to be very compatible with HTTP and other existing Internet standards.

The ITU world has documented a similar infrastructure in the Recommendation H.323. This umbrella document from the ITU describes how existing ITU and Internet protocols can be used together to offer build multimedia terminal equipment as well as control infrastructure such as Gatekeepers for call control, multi-point control units (MCUs) for conferencing as well as resource control. The H.323. work is more mature than the MMUSIC work, with H.323v1 (1996) and H.323.v2 (1998) now fairly widely accepted and implemented. More information about H.323 and related standards can be found at the WWW site of the Internet Multimedia Technical Consortium (IMTC) which is an industry forum promoting the use of H.323 and related standards.



Summary

- Need QoS mechanisms for IP
- Per flow:
  - INTSERV
  - RSVP
  - does not scale well, hard to provision
- Customer/provider services:
  - DIFFSERV
  - still maturing
- Support for application: RTP and signalling

DigiComm II-43