Layering

The OSI 7-layer model (X.200)

When Kleinrock designed CL packet switching
the tremendous complexity was apparent
(as compared to CS or CO networks)

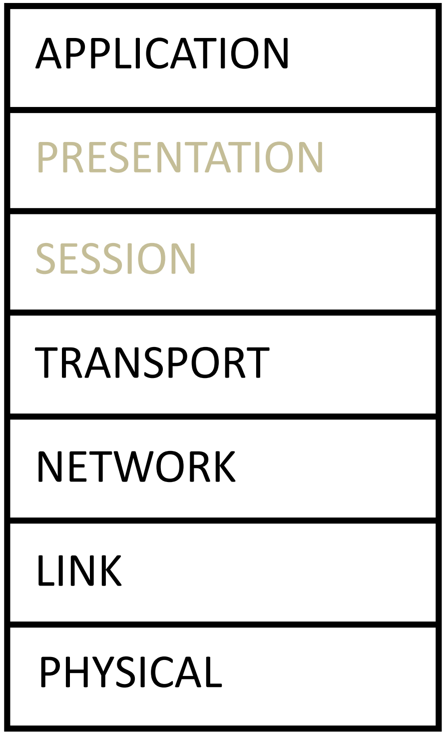
So he did what any scientist would do
he separated the problem into *layers*

Each layer was a discipline with its own experts
Each expert knew his layer well
and how to interface to the layers above and below

Such layering is similar to division of natural science into

- physics
- chemistry
- biology
- etc.

It is only done due to limitations of the human brain
and leads to inefficiencies (since it contradicts Shannon's Separation Law)



Layering inefficiency

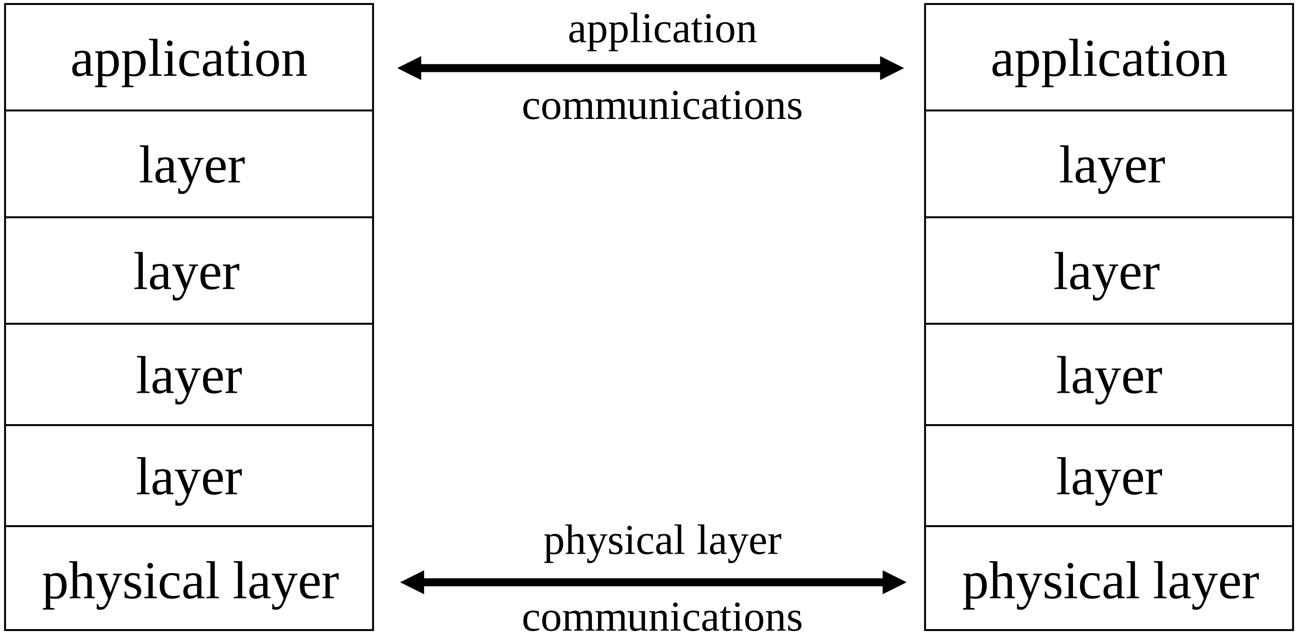
Consider a typical **V**oice **o**ver **IP** scenario

- 10 milliseconds of **64 kbps** telephony-grade voice are extracted (100 pps)
- using G.729 **8 kbps** compression the 80 B are compressed down to 10 B
- the 10 B payload is encapsulated in RTP/UDP/IP/Ethernet
  - minimal RTP header is 12 B
  - UDP header is 8 B
  - IPv4 header (without options) is 20 B (IPv6 is 40 B)
  - Ethernet MAC header is at least 14 B and its trailer (FCS) is 4 B
  - each VLAN tag adds 4 B
  - Ethernet physical layer adds at least 12 B IPG and 8 B preamble
  - altogether at least 88 B (for IPv6 and with 2 VLAN tags 116 B) (78 B or 106 B overhead for 10 B payload)

Leading to 70.4 (92.8) kbps which is greater than the 64 kbps of a DS0 !

Of course, if speak only ½ of the time (and no **C**omfort **N**oise **G**eneration is used) this *is* reduced to 35.2 (46.4) kbps (but the savings are not dramatic)

The layering concept



Layering simplifies communications when each layer is *independent*
i.e., a peer at a given layer believes it is communicating with its peer at the same level

So, an application (e.g., Skype) can be considered to communicate with a peer application

The highest layer is always the *application*, and the lowest the *duct*

Layering options

Two very different layering methodologies have been used

- all layers serve the same entity and each layer performs one function
- each layer serves an entity and performs all functions

The famous OSI model (X.200) conforms to the first methodology while the modern G.80x model conforms to the second

Layering is today used for all networks, including CS ones

OSI was more natural for *simple* packet switched networks

The new G.80x model

- can describe more general networks with an unlimited number of layers
- enforces clean client/server interfacing between layers
- supports network partitioning in addition to network layering
- has a diagrammatic technique
  - which is not just descriptive
  - but has built-in network manipulation mechanisms
- can aid in designing more efficient networks

The OSI layers

- L1 **physical** layer
  - physical (electrical/optical) specifications of the link
- L2 **link** layer
  - provides reliable link between two directly connected NEs
- L3 **network** layer
  - forwards packets between any two NEs in network
  - computes routes and distributes them to forwarding elements
- L4 **transport** layer
  - optionally provides :
    - end-to-end reliability for file/stream packet sequences
    - congestion collapse avoidance

  (IETF currently defines TCP, UDP, SCTP, and DCCP)
- L5 **session** layer and L6 **presentation** layer (mostly obsolete)
- L7 **application** layer
  - user application (or enables user application)

Problems with the OSI model

The OSI model was a good theoretical basis at the time
but no real networks actually work only that way

- OSI layers were subdivided or omitted
  - Ethernet divides L2 into multiple layers
  - L5 and L6 have mostly disappeared
- new layers were added in between existing ones
  - e.g., MPLS was added at layer 2½
- some features only in one place (security, mux) are needed elsewhere
- some functionalities are missing (e.g., OAM, redundancy)
- doesn't support service-provider business models

OSI can't describe important layerings

For example, this layering is frequently used →
but can not be described by the OSI model

application
TCP
IP
MPLS
Ethernet MAC
Pseudowire
MPLS
SONET path
SONET line
SONET section

The top 3 layers make some sense, but then

- MPLS is "in-between" L3 IP and L2 Ethernet
- Ethernet is divided into 2 layers (ETH and ETY) only one of which appears here
- PW is another in-between layer
- MPLS appears again !
- the "physical" layer is divided into three layers

Why is such a complicated layering used ?

The new model (G.80x)

A lesson learned as the PSTN evolved was the importance of **layer networks**

Each layer network is an independent network in its own right
independently designed and maintained

There must be an *operational reason* for each layer network

All layer networks should be described using the same tools

One should be able to add/modify layer networks
without changing neighboring layer networks

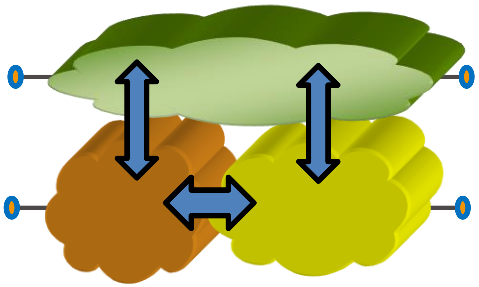
There must be a client/server relationship between neighboring layers

In order for layering to be *clean*
server layer should *transparently* carry the client layer's CI

Each layer network needs its own

- addressing and forwarding mechanisms
- OAM mechanisms to guarantee QoS for its client
- control protocols
- management
- security

Consequences of layer violations



Client/server (G.80x) layering enables Service Providers

- to serve a higher-layer SP
- to be served by a lower-layer SP

Layer violations may lead to security breaches, such as :

- billing avoidance
- misrouting or loss of information
- information theft
- session highjacking
- information tampering

Layer *respect* is often enforced by network element functionality

Some newer technologies (including SDN) do not enforce layer respect

Some references

The new model is described in ITU-T Recommendations

G.805 describes the new model for CO networks

- CO networks transfer information over connections

G.809 describes the new model for CL networks

- CL networks do not have connections but *may* have flows

CS networks are described in G.705 (PDH), G.783 (SDH), and G.872 (OTN)

G.800 Unified functional architecture
G.805 CO networks
G.806 equipment model
G.809 CL networks

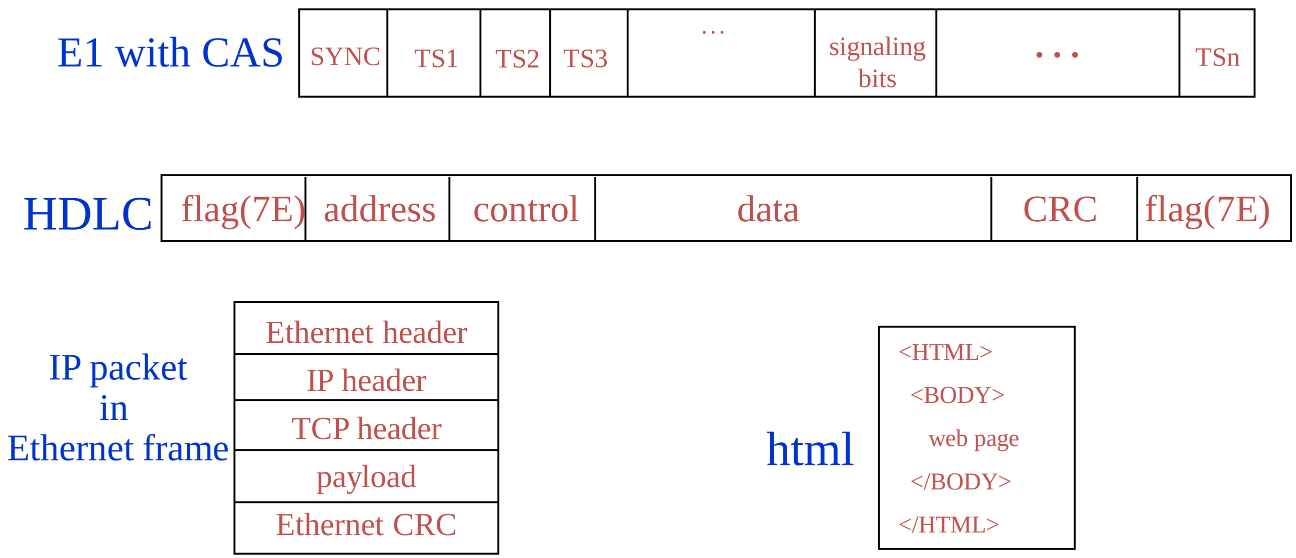
G.705 PDH
G.783 SDH
G.872 OTN
G.781 timing
G.8010 Ethernet
G.8021 Ethernet equipment
G.8110 MPLS

Characteristic Information

The purpose of communications is to move information

Each application and network has its own information *format*

Examples:



The information at a given level is called **characteristic information** (CI)

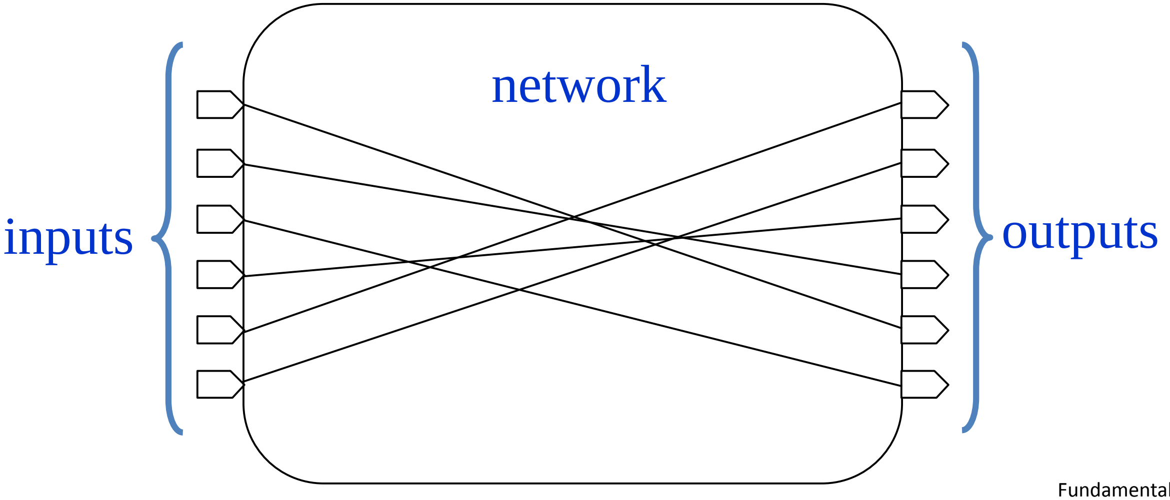
Layer Networks

In the new framework

- each layer is an independent network
- called a **layer network**
    because it exists at one layer
    because it is a network unto itself

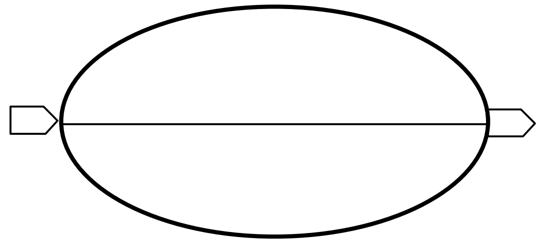
The goal of a layer network is to transport CI with minimal degradation

The association of an input with an output
    is called a connection in a CO layer network
    is called a flow in a CL layer network

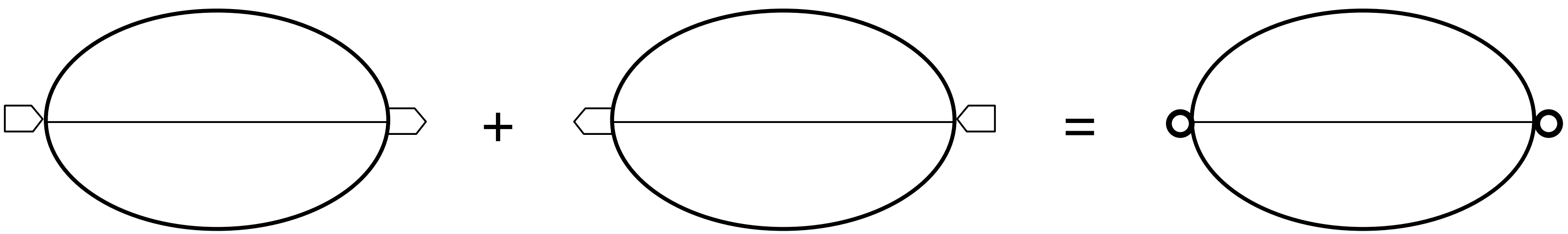


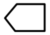
Network Connections (G.805)

A network connection matches one output to one input



often we want to have a **bidirectional** connection



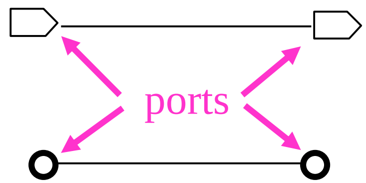
like a transceiver or a modem,  is a  colocated with a 

Network Connection Types

A link connection (LC) is a **fixed** connection between 2 “ports”

unidirectional link connection

bidirectional link connection

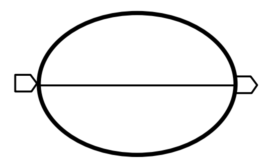


the LC is the smallest unit of manageable capacity

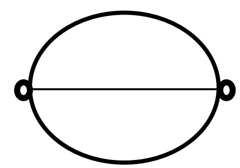
A subnetwork connection (SNC) is a **flexible** connection

For CO networks SNCs are changed by network management functions

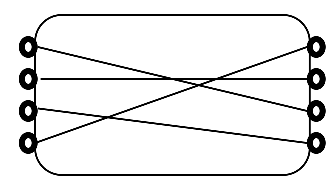
unidirectional subnetwork connection



bidirectional subnetwork connection



The simplest subnetwork is a single Network Element (NE)
such as a matrix, switch, or crossconnect



Transport and Topology

A **transport entity** transfers information from point to point

A **transport processing function** performs some information processing

At a high level of abstraction

only the *possible* connections between inputs and outputs is important

- the geographical location of the endpoints
- the data rate
- the type of physical connection
- etc.

are ignored

G.805 defines a topological component that relates inputs to outputs

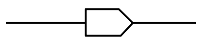
Layer networks and subnetworks are topological components

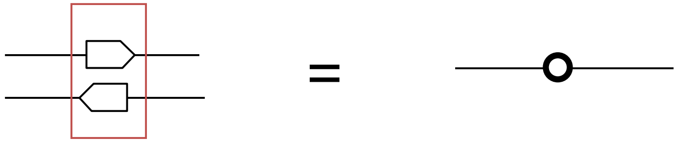
SNCs and LCs are transport entities

We will see some processing functions later

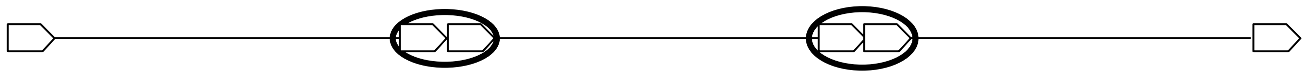
e.g. to adapt format from layer to layer

Reference Points

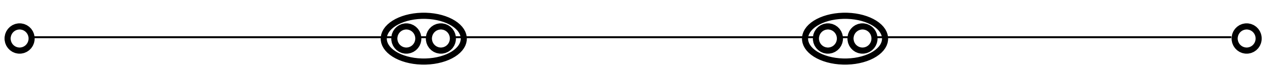

unidirectional input or output point


bidirectional input/output point

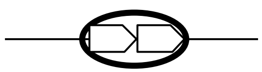
we concatenate connections by binding
the output of one connection
to the input of the next connection

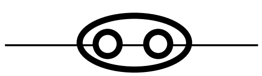


we can do the same thing with bidirectional connections



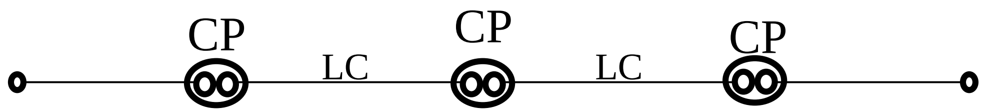
we thus create reference points called **connection points** (CP)


unidirectional connection point

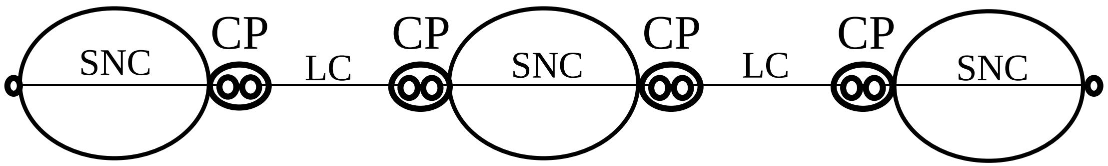

bidirectional connection point

Connection Points

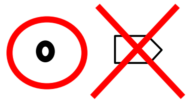
We can concatenate link connections



Similarly, we use link connections to connect subnetwork connections

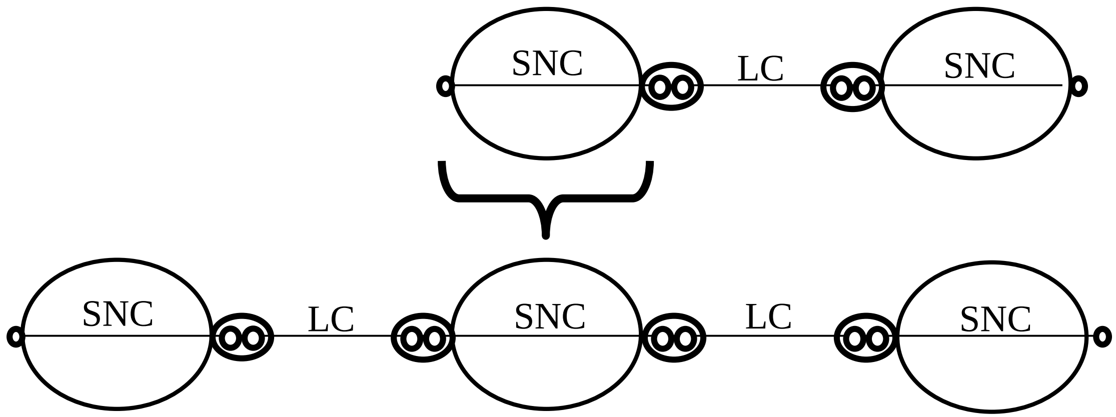


We will mostly focus on bidirectional connections
but remember this merely hides the functionality



Partitioning

If we zoom in on an SNC
we discover that it too is made up of SNCs connected by LCs



We can continue *recursively* zooming in until we are left
with LCs and flexible connections internal to NEs

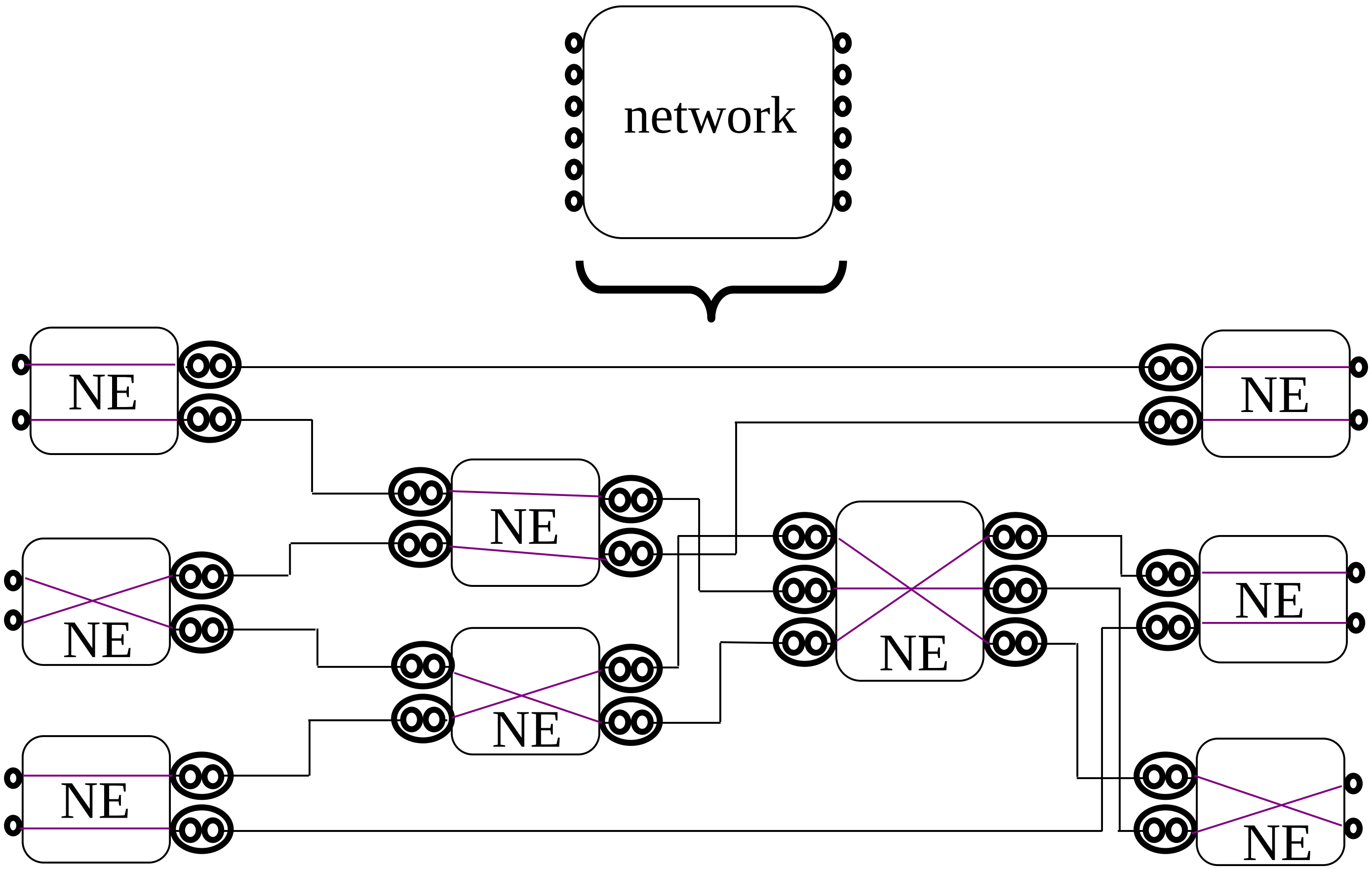
Different degrees of detail are useful for different purposes

Partitioning may be used to delineate:

- routing domains
- administrative boundaries between different operators
- service provider/customer networks

Layer Network Partitioning

The whole layer network can be recursively decomposed into connections internal to NEs and link connections



OAM

Analog channels and 64 kbps digital channels

did not have mechanisms to check signal validity and quality

Thus

- major faults could go undetected for long periods of time
- hard to characterize and localize faults when reported
- minor defects might be unnoticed indefinitely

As PDH networks evolved, more and more overhead was dedicated to

Operations, **A**dministration and **M**aintenance (OAM) functions

including:

- monitoring for valid signal
- defect reporting
- alarm indication/inhibition

When SONET/SDH was designed

overhead was reserved for OAM functions

Today service providers require complete OAM solutions

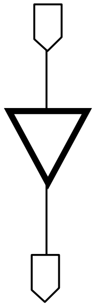
Trails

Since OAM is critical to proper network functioning
OAM must be added to the concept of a connection

A **trail** is defined as a connection along with integrity supervision

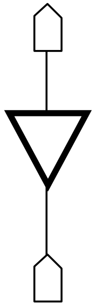
Clients gain access to the trail at access points (AP)

A trail termination (TT) source accepts CI
and adds trail overhead information



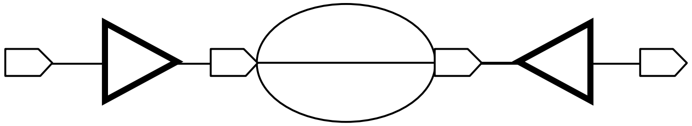
trail terminations are denoted by triangles

A trail termination (TT) sink
supervises integrity of trail
and removes trail overhead



the triangle always points towards the supervised connection

Reference points where trail terminations binds to connections
are called termination connecting points (TCP)

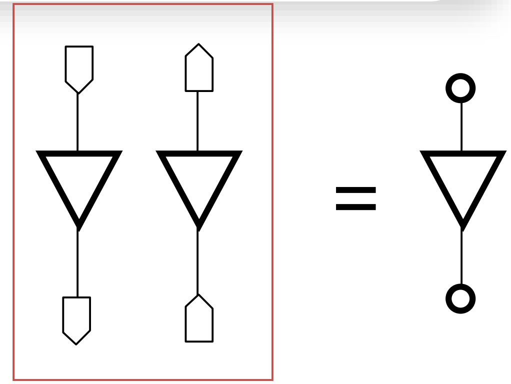


Trails (cont.)

For bidirectional trails

there is a shorthand notation

for colocated termination source and sinks



bidirectional trail termination

A trail is considered to run

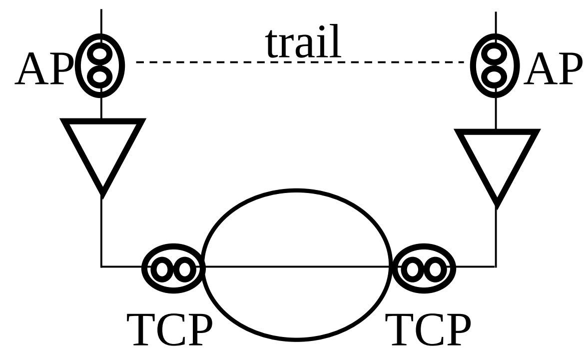
from the *input* to the trail termination source

to the *output* of the trail termination sink

so the access points are

before the trail termination source

after the trail termination sink



sometimes we specify the network inside the triangle

Trail Termination Functions

What precise functionality does the trail add to the connection itself?

- continuity check (e.g. LOS, periodic CC packets)
- connectivity check (detect misrouting)
- signal quality monitoring (e.g. error detection coding)
- alarm indication/inhibition (e.g. AIS, RDI)

Source termination functions:

- generate error check code (FEC, CRC, etc)
- return remote indications (REI, RDI)
- insert trail trace identification information

Sink termination functions:

- detect misconnections
- detect loss of signal, loss of framing, AIS instead of signal, etc.
- detect code violations and/or bit errors
- monitor performance

Defects, Faults, etc.

G.806 defines:

anomaly (n): smallest observable discrepancy between desired and actual characteristics

defect (d): density of anomalies that interrupts some required function

fault cause (c): root cause behind multiple defects

failure (f): persistent fault cause - ability to perform function is terminated

action (a): action requested due to fault cause

performance parameter (p): calculatable value representing ability to function

for example:

- dLOS = loss of signal defect
- cPLM = payload mismatch cause
- aAIS = insertion of AIS action

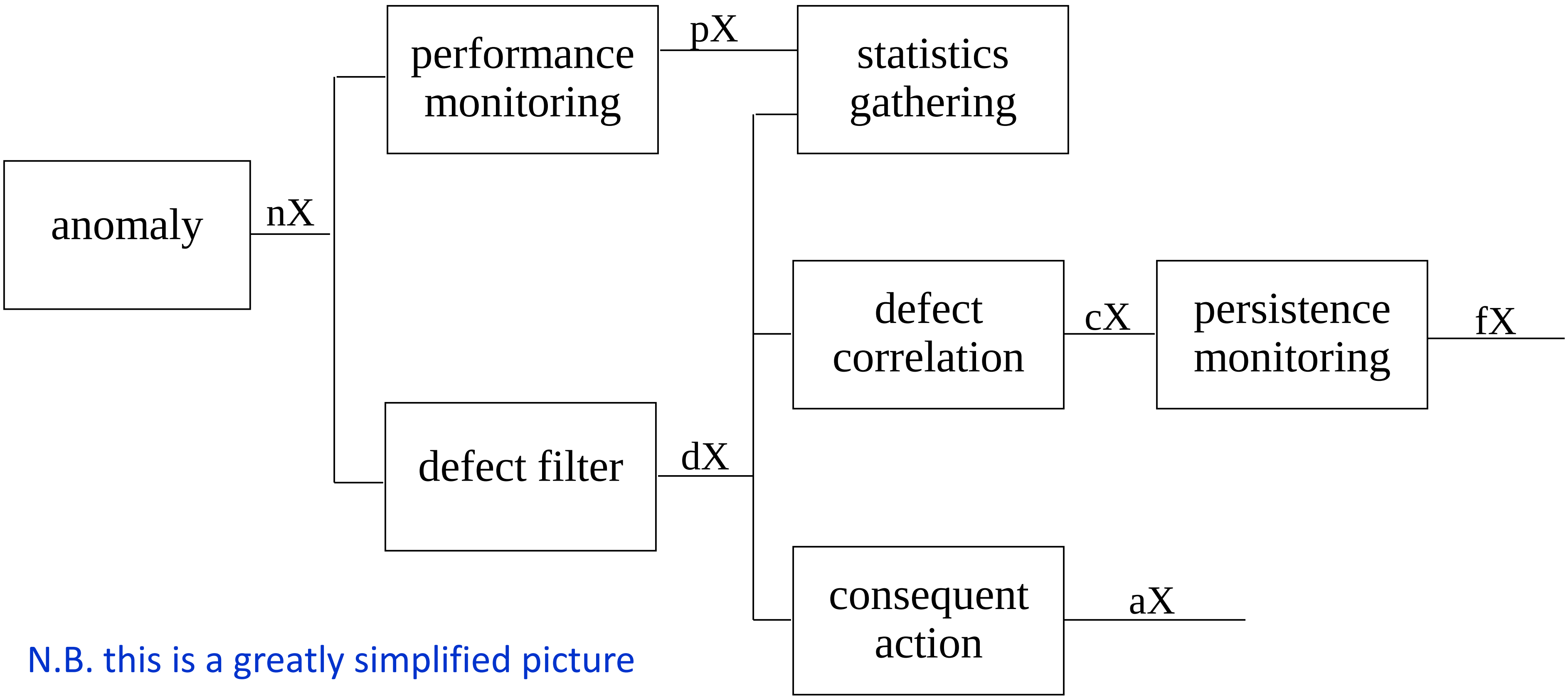
equipment specifications define relationships

e.g.

aAIS <= dAIS or dLOS or dLOF

alarms are human observable failure indications

Supervision Flowchart



N.B. this is a greatly simplified picture

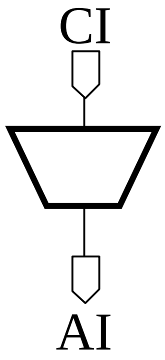
more generally there are external signals, time constants, etc.

Adaptation

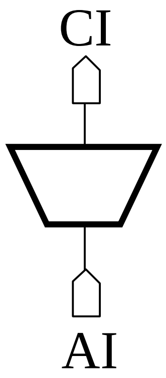
Although all layer networks are created equal
the format of their CI is different

So in order to put the client information into the server;s format
we have to *adapt* it

This is done by an *adaptation function*



An adaptation source accepts client CI
usually *encapsulates* it for transfer over the server trail
creating adapted information (AI)



An adaptation sink accepts the AI
and recovers the client layer CI

adaptations are
denoted by *trapezoids*

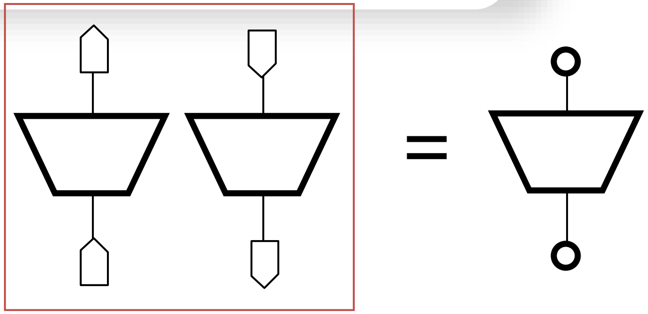
the trapezoid always points
towards the server layer

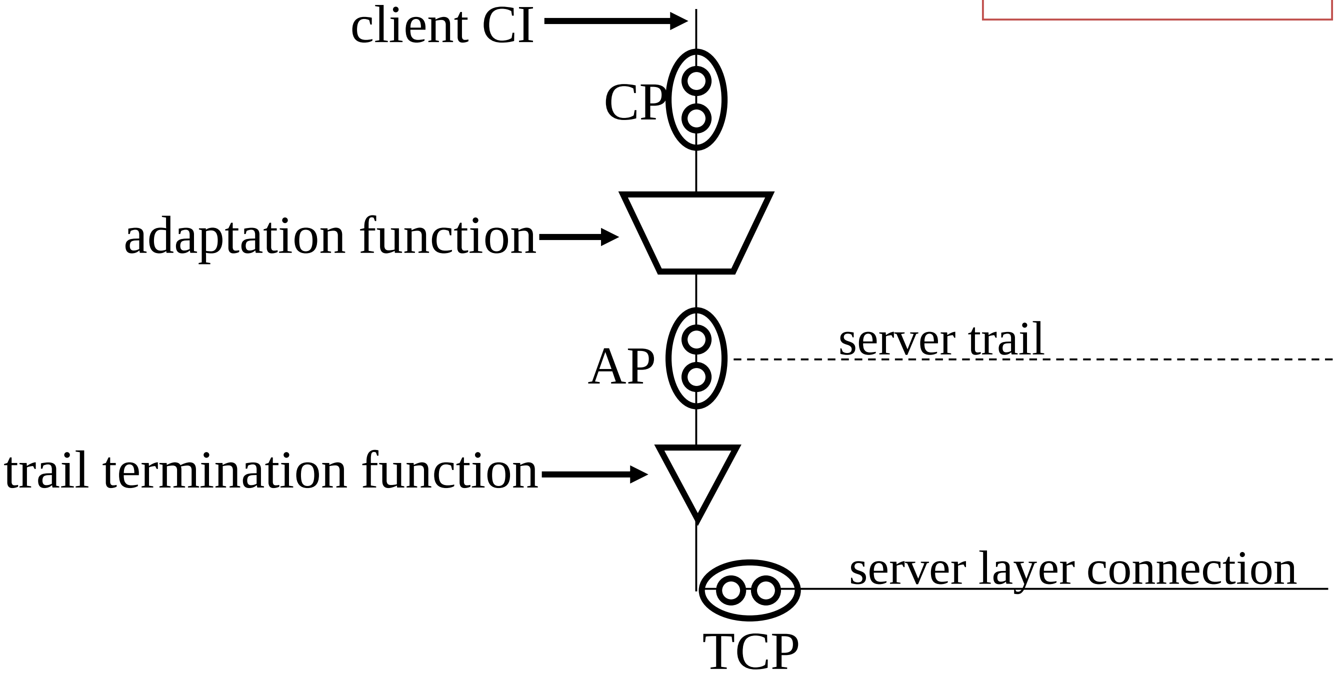
Adaptation (cont.)

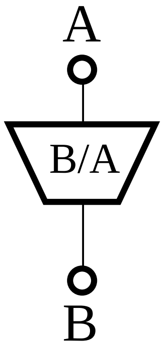
For bidirectional trails

there is a shorthand notation

for colocated adaptation source and sinks







sometimes we specify the layer networks inside the trapezoid

order - server/client

Adaptation Functions

What precise functionality does the adaptation perform?

source adaptation may include:

- bit scrambling
- encoding
- framing
- encapsulation
- bit-rate adaptation
- multiplexing, inverse multiplexing
- etc.

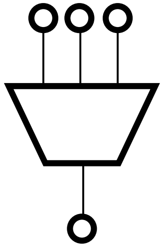
sink adaptation:

- descrambling
- decoding
- deframing
- decapsulation
- bit-rate adaptation
- demultiplexing
- timing recovery
- monitoring for AIS
- etc.

Muxing and Inverse Muxing

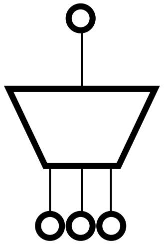
There may be a many-to-one relationship between clients and server

- one server layer trail simultaneously multiplexing many client layer networks (the client layer networks could be of the same or of different types)



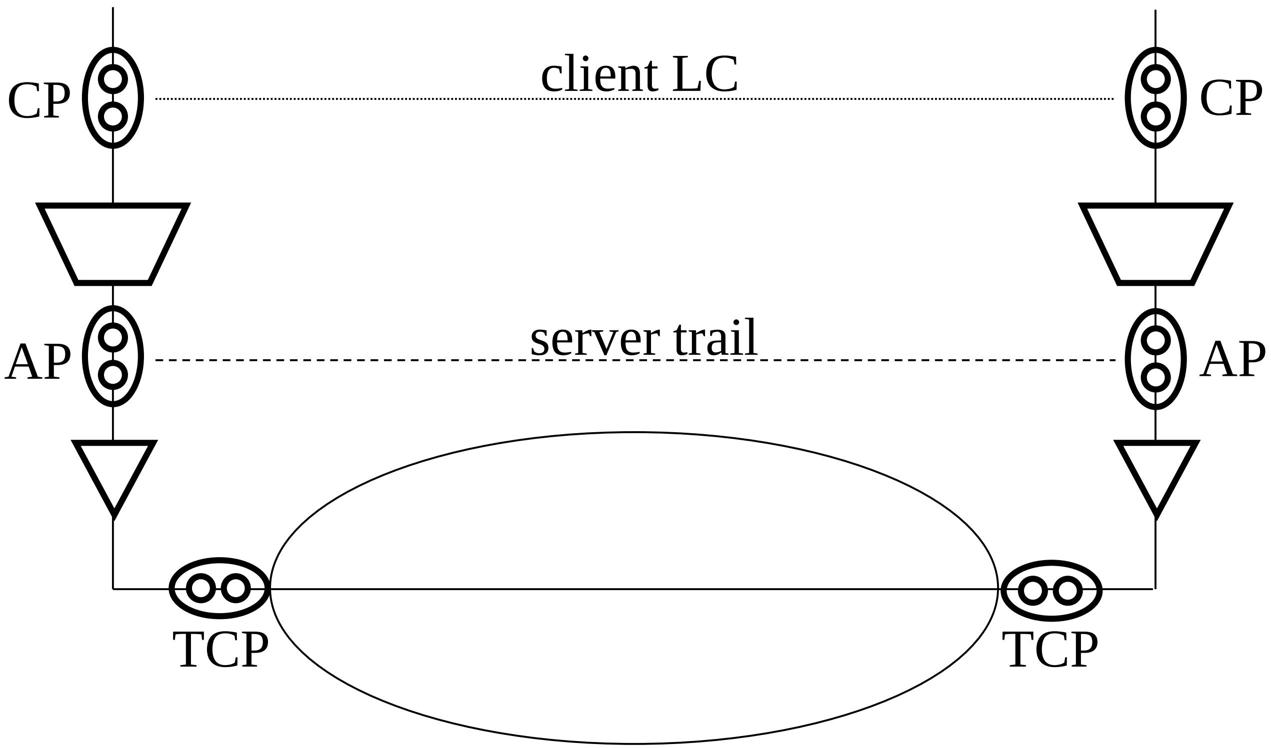
There may be a one-to-many relationship between a client and servers

- multiple server layer trails simultaneously inverse mux a client layer network (the server layer networks could be of the same or of different types)



The BIG Picture

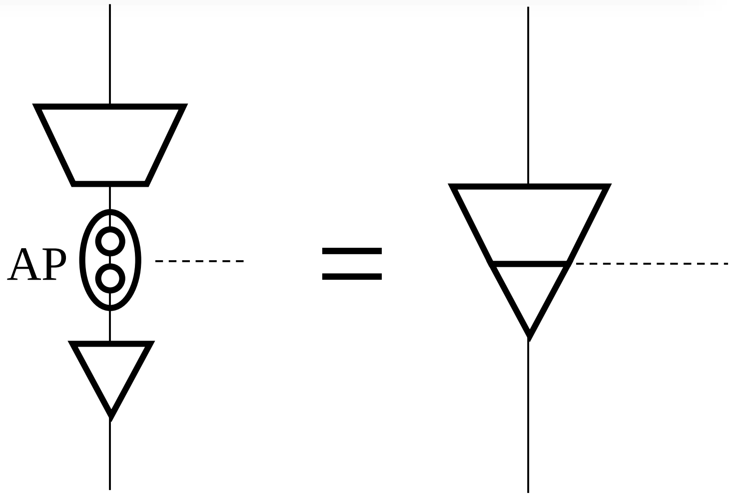
A link connection in the client layer
is supported by a trail in the server layer



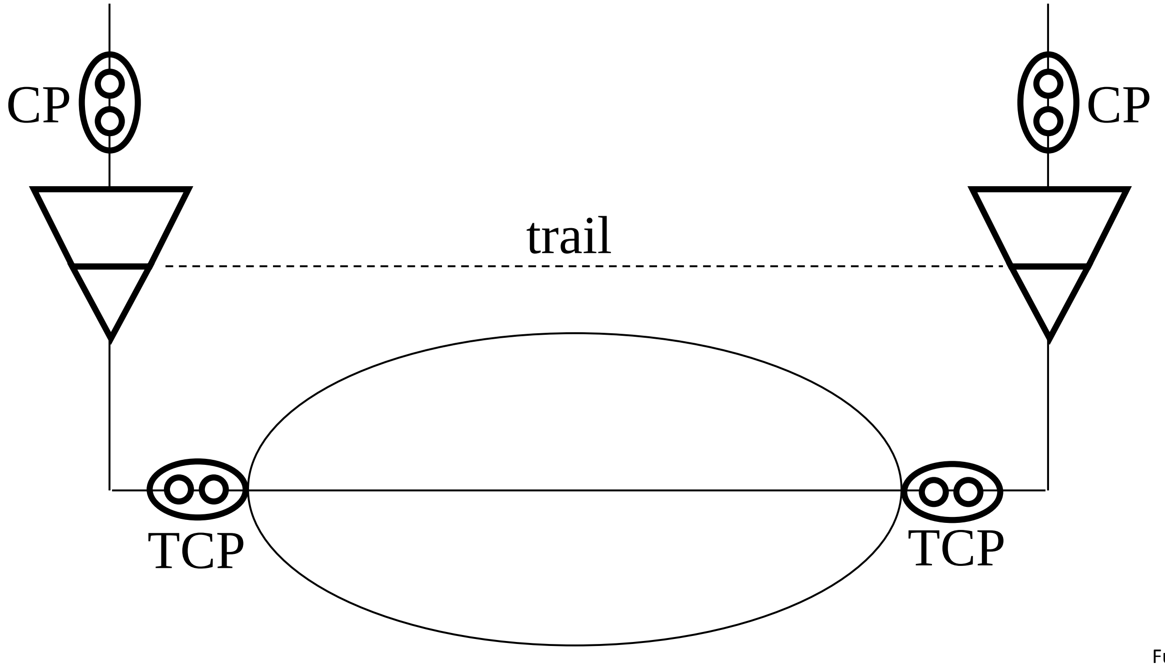
N.B. the flexibility of the server layer connections
is unavailable to the client layer

Shorthand notation

it is often convenient to combine adaptation and trail terminations

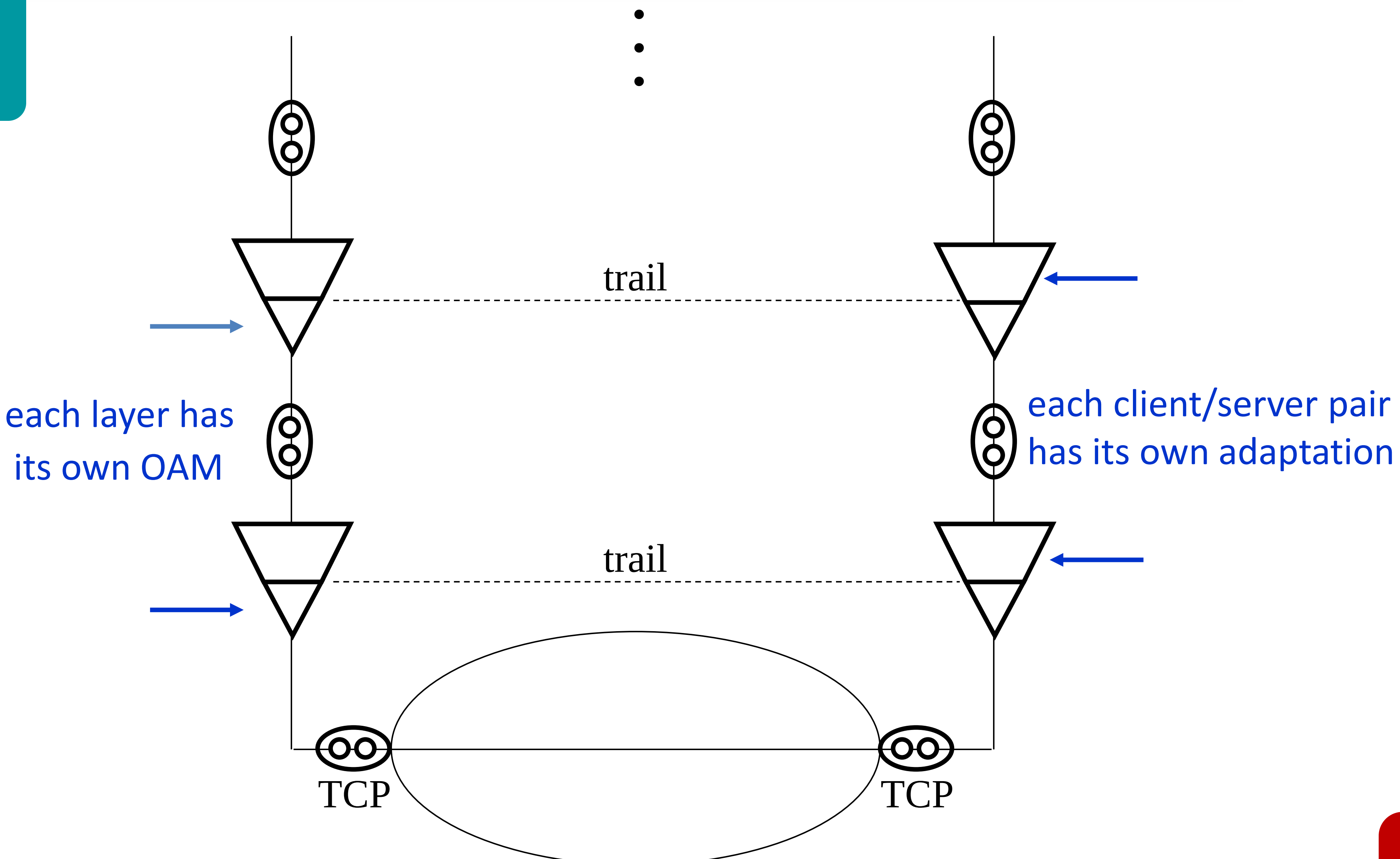


and we obtain the simpler diagram:

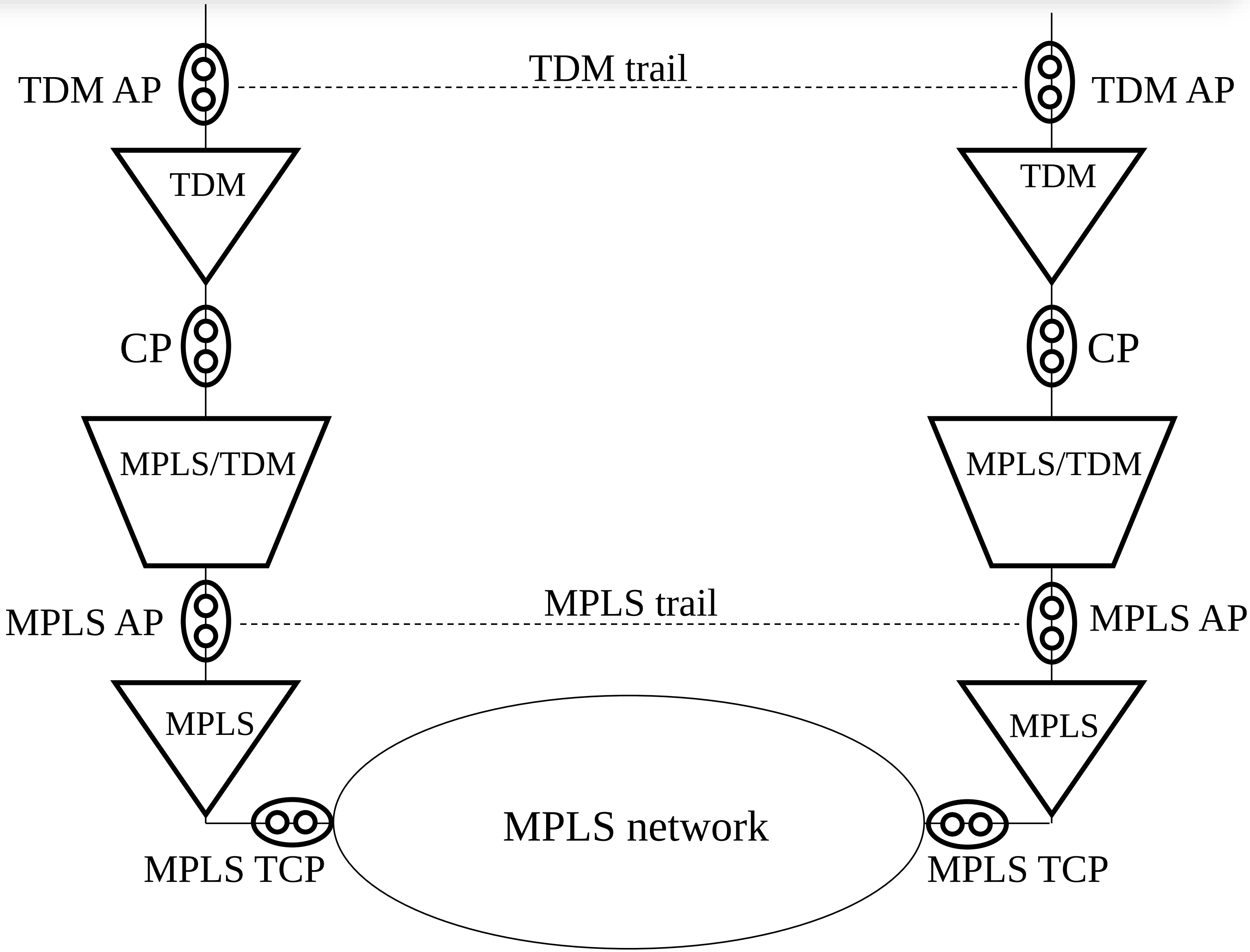


but AP is hidden

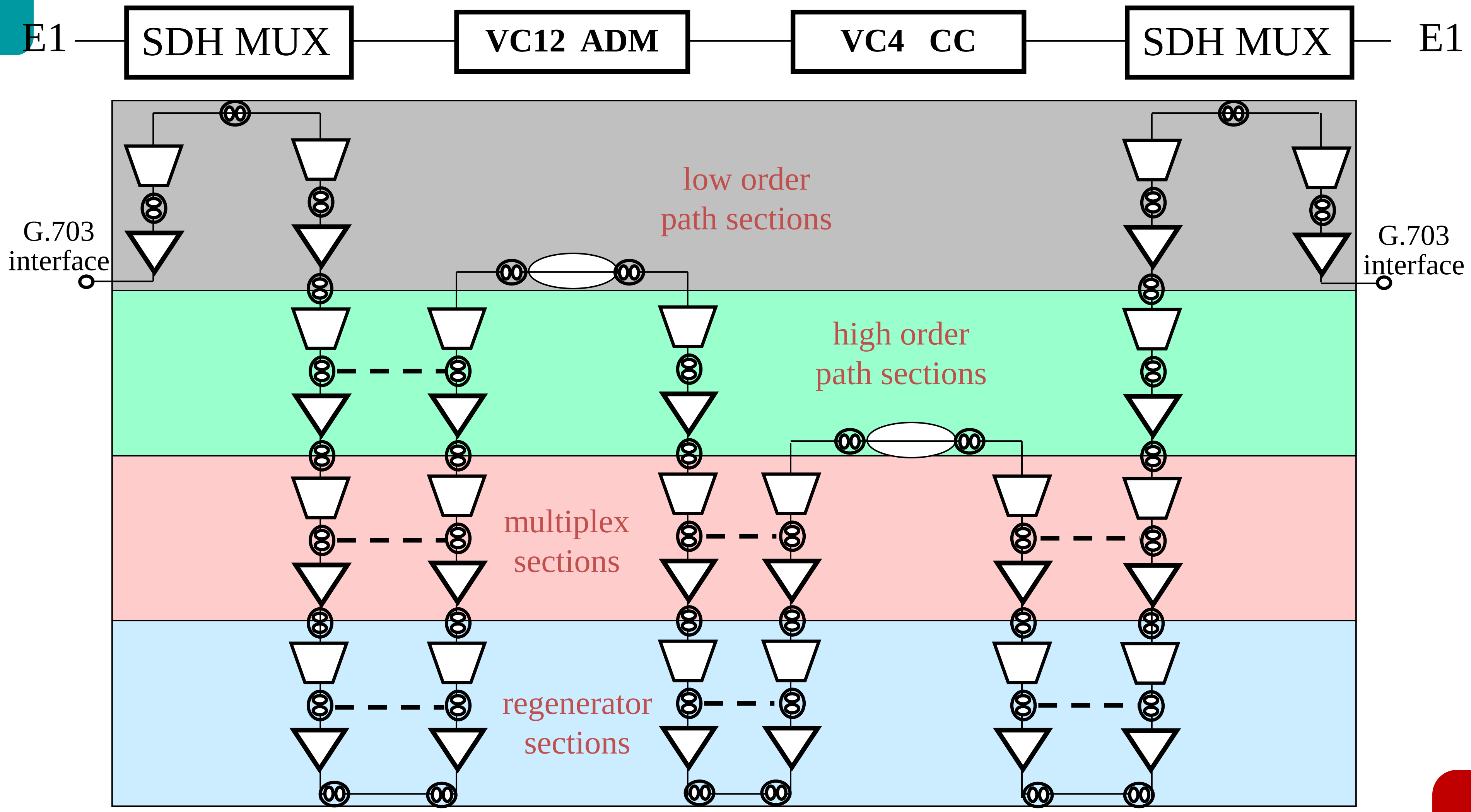
More and more layers



Simple Example: SAToP-MPLS



More Complex Example
PDH over SDH



Layering vs. Partitioning

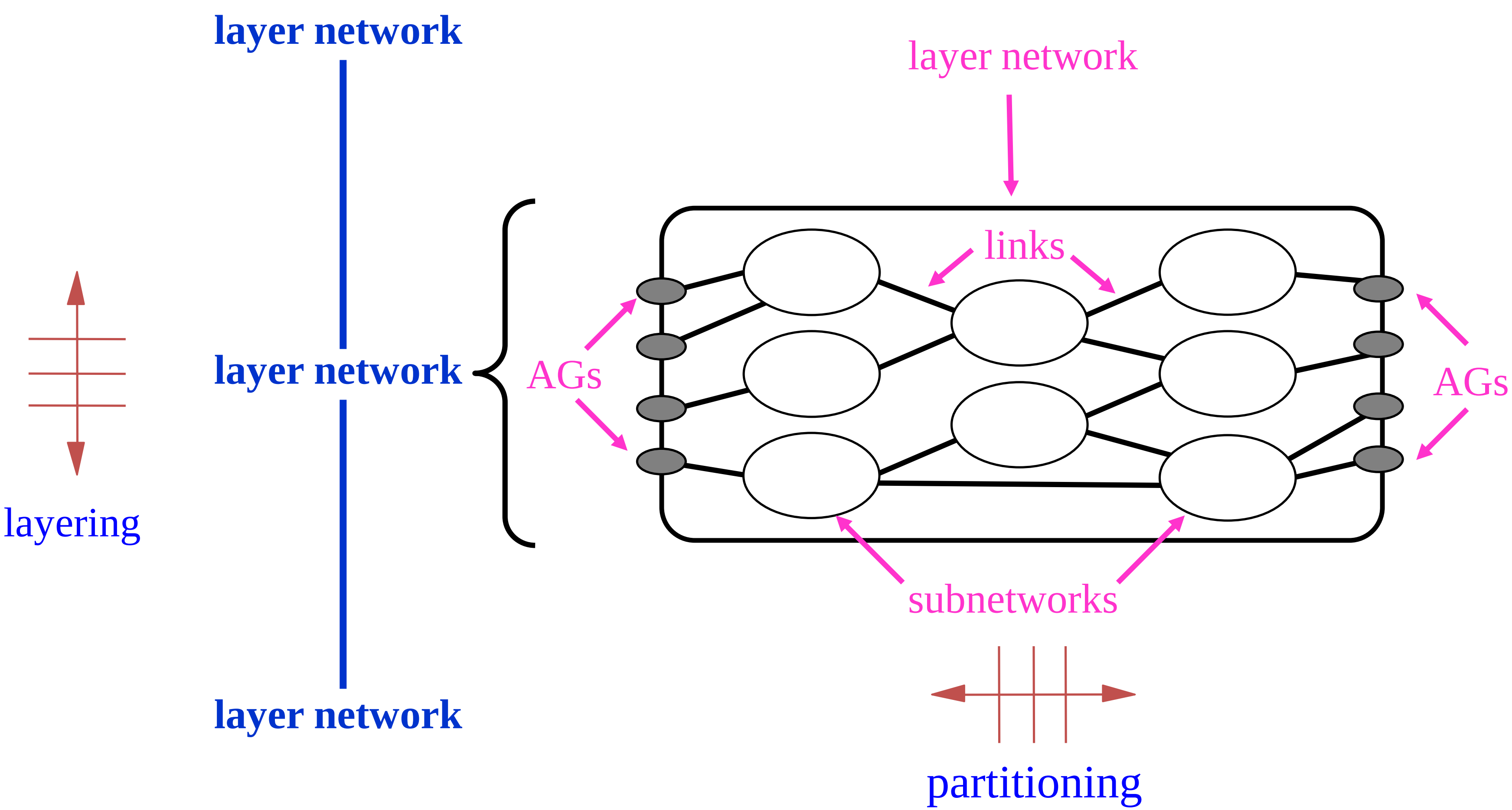
Each layer network may be separately partitioned

Layering and partitioning are thus orthogonal analyses

- layering is *vertical*
  - client layer network is “above” the server layer network
- partitioning is *horizontal*
  - subnetworks and links belong to same layer network

A trail in a server layer network
supports a LC in its client layer network

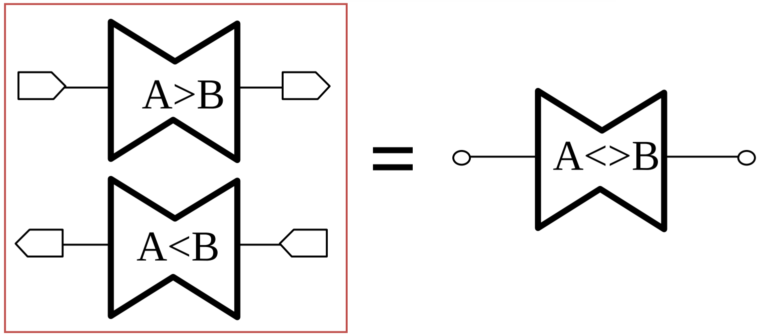
Layering vs. Partitioning (cont.)



Access Groups (AG) are colocated APs that belong to the same client

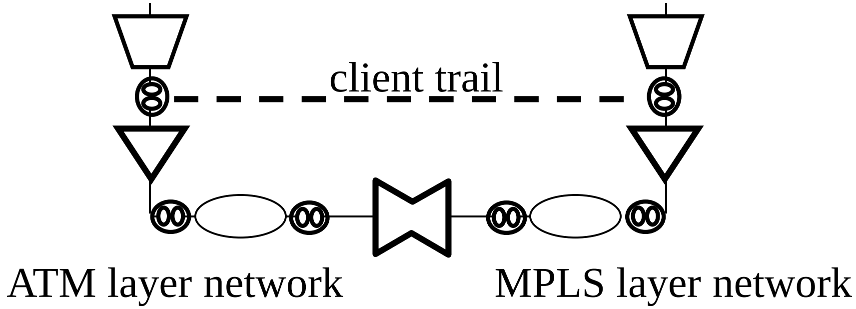
Service Interworking

We have seen how to carry traffic
from network A *over* network B
client/server relationship



There is also layer network interworking (AKA service interworking - SI)
terminate network A and carry its client over network B
peer to peer relationship

Example: Service Interworking of ATM with MPLS

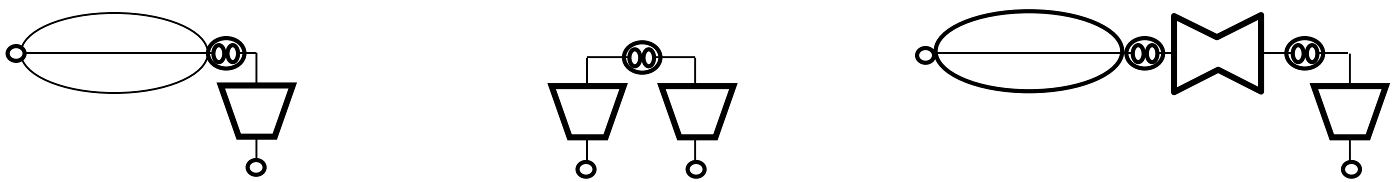


N.B. SI is usually limited to a specific client type

Permissible Bindings

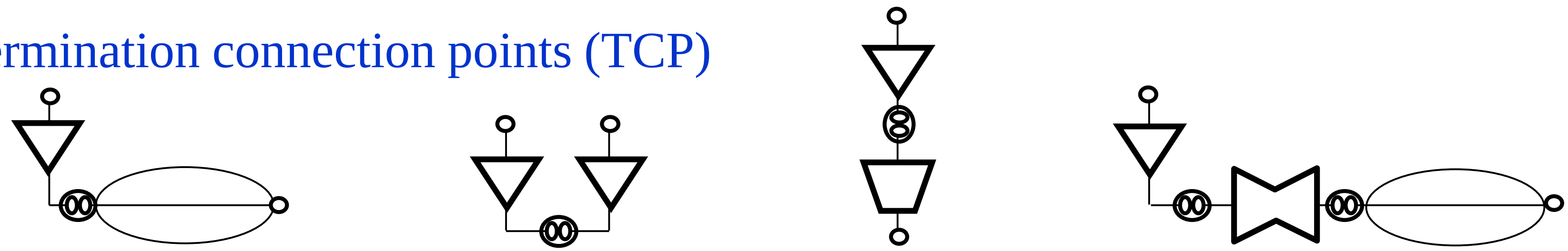
Inputs and outputs may be bound together
iff they share CI or adapted information

connection points (CP)



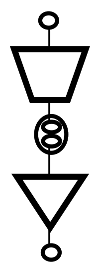
connection - adaptation | adaptation - adaptation | SI - adaptation

termination connection points (TCP)



TT - connection | TT - TT | TT - adaptation | TT - SI

access points (AP)



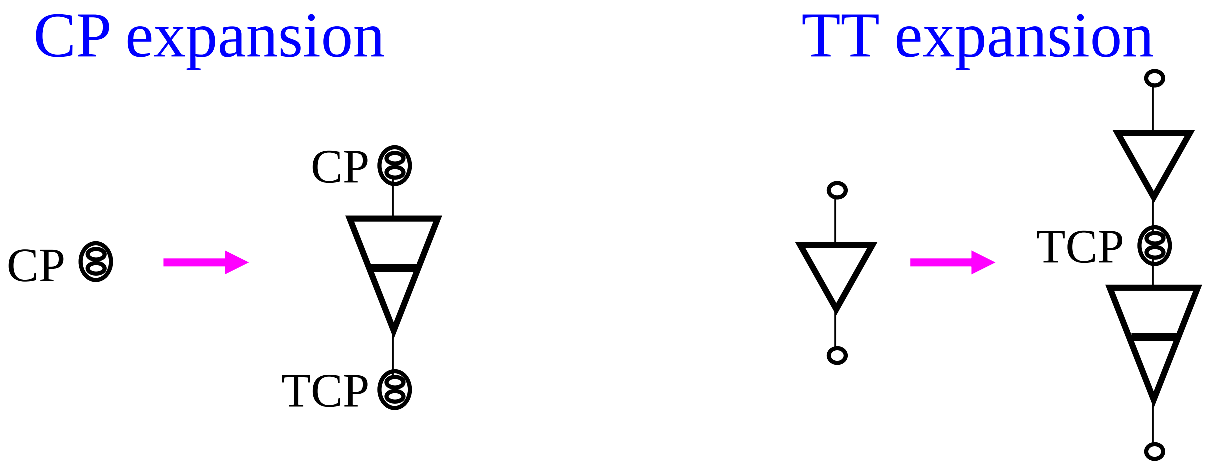
adaptation - TT

the difference between a LNC and a SNC:
- network connections are delineated by TCPs
- SNCs are delineated by CPs

Expansions

New functionality is formally introduced

by *expanding* a CP or a TT



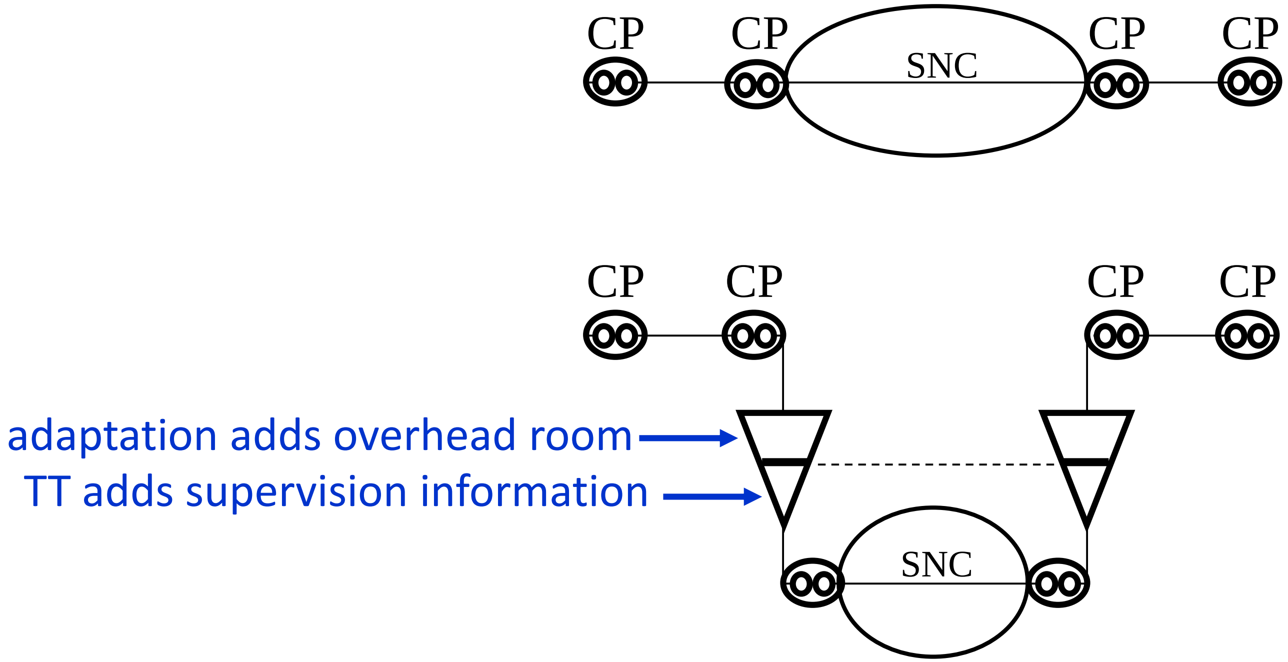
We will show one example of each of these expansions:

- CP expansion to monitor SNC
- TT expansion for trail protection

Example - tandem monitoring

If we need to separately monitor subnetworks

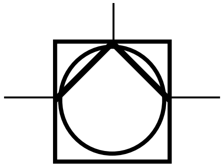
For example, in order to provide defect localization
we can expand a CP to make them into full layer networks

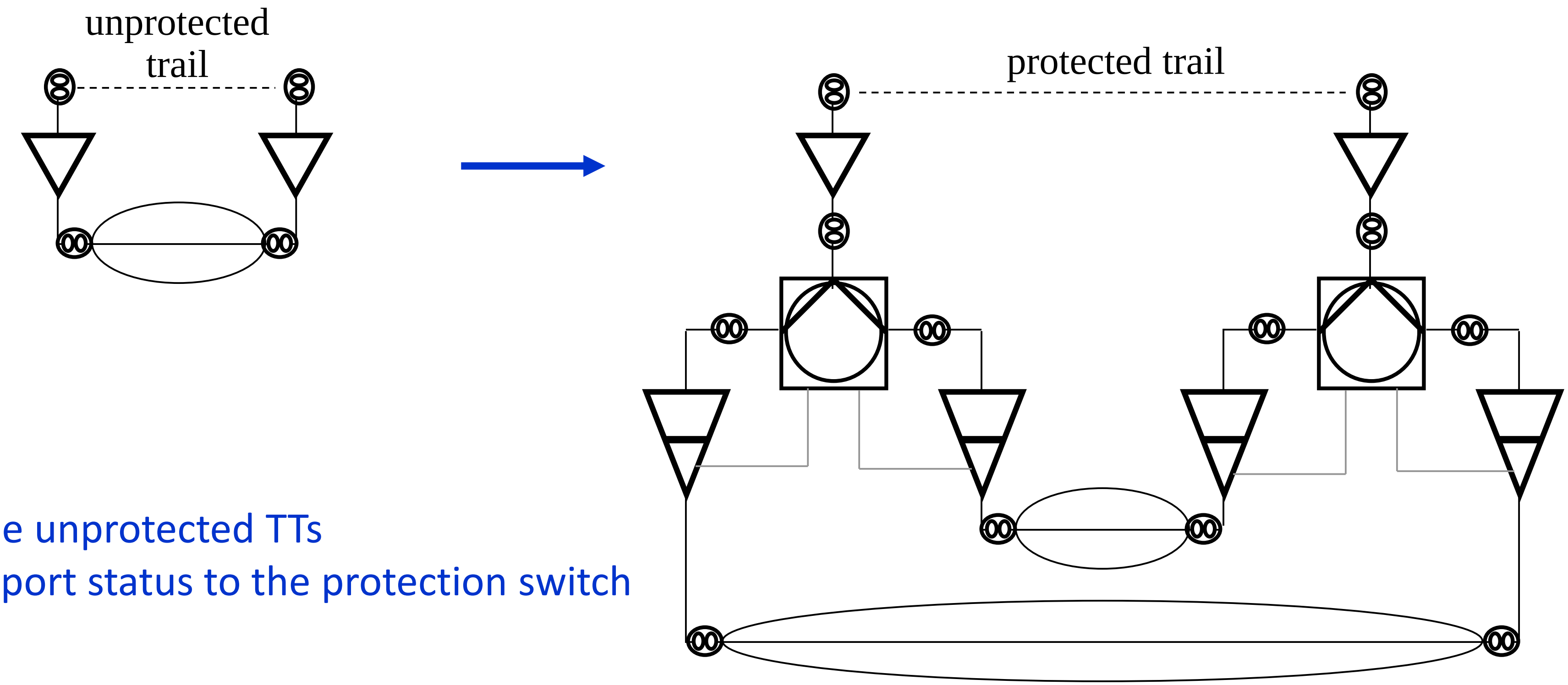


Example - trail protection

To add 1+1 protection for a trail, we can expand a TT

we use a special transport processing function - the protection switch





the unprotected TTs
report status to the protection switch

G.809

CL networks can be partitioned and layered just like CO ones

  but in CL networks there are no connections

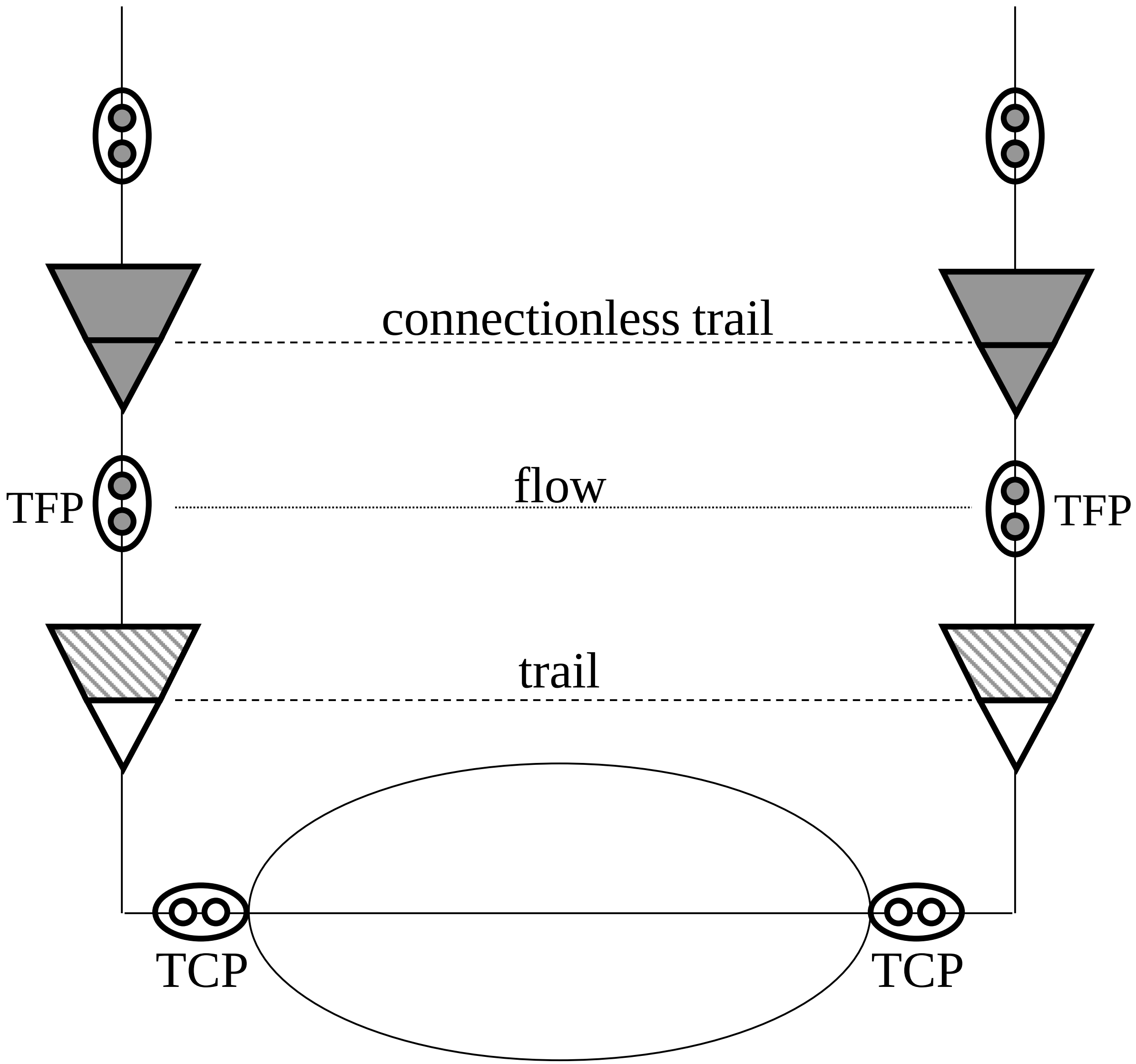
Instead we have a new concept - a *flow*
(there are link flows, flow domain flows, and network flows)

Once monitored, adapted CI is transported on a connectionless trail

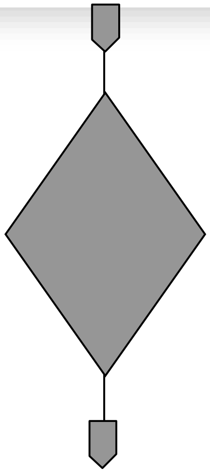
G.809 diagrams are similar to G.805 ones

  but shading indicates CL components

CL client / CO server



CL traffic conditioning



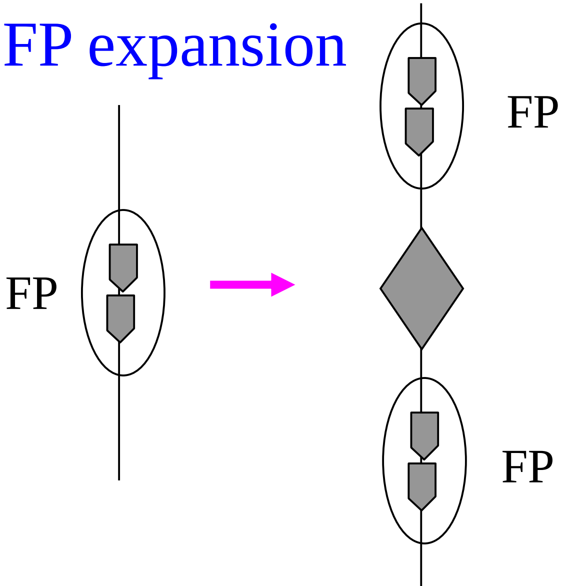
CL networks have some unique requirements

For example, G.8010 defines a *traffic conditioning* function

This transport processing function classifies packets
and then meters / polices within each class

You can add the TC function by *expanding a FP*

FP expansion



Research topics

The new model has not been extensively studied in academic circles

- Diagrammatic techniques are extensively used in DSP and quantum physics
  They are used by network planners to increase reliability of packet networks
  How can they be used in academic circles ?
  Can they be used for numerical computations ?

- Can G.80x and the diagrammatic techniques be used to
  - reduce layering overhead ?
  - increase efficiency (including energy efficiency) ?