What is TCP ?

"Provides bidirectional, flow-controlled, error-free streams between processes"

- **Streams**

  By using TCP, applications can send arbitrary amounts of data. TCP splits the data, puts it into IP datagrams, and reassembles it at the other end

- **Error-free**

  TCP packets contain a **sequence number** to ensure the stream is reassembled correctly and a **checksum** to detect transmission errors

  The other end sends an **acknowledgement number** to indicate how much of the stream has been successfully received. Datagrams are resent if not acknowledged within a suitable time

- **Flow-controlled**

  TCP will not send more than a certain amount of data (the 'window') before waiting for an acknowledgement. The receiver controls the rate at which data is sent to it, preventing overflow

- **Bidirectional**

  A TCP connection consists of a pair of streams, one in each direction

- **Processes**

  TCP includes a **Source Port** and **Destination Port** which, together with the Source IP number and Destination IP number, uniquely identify the stream

Other protocols in “TCP/IP” suite

- **User Datagram Protocol (UDP)**

  For when the overhead of setting up a TCP connection is not required, and application can cope with datagrams going missing

  Adds very little functionality to IP: only Source and Destination Ports and a checksum

  Main uses: short transactions (request—response)

  - DNS lookups
  - bootp

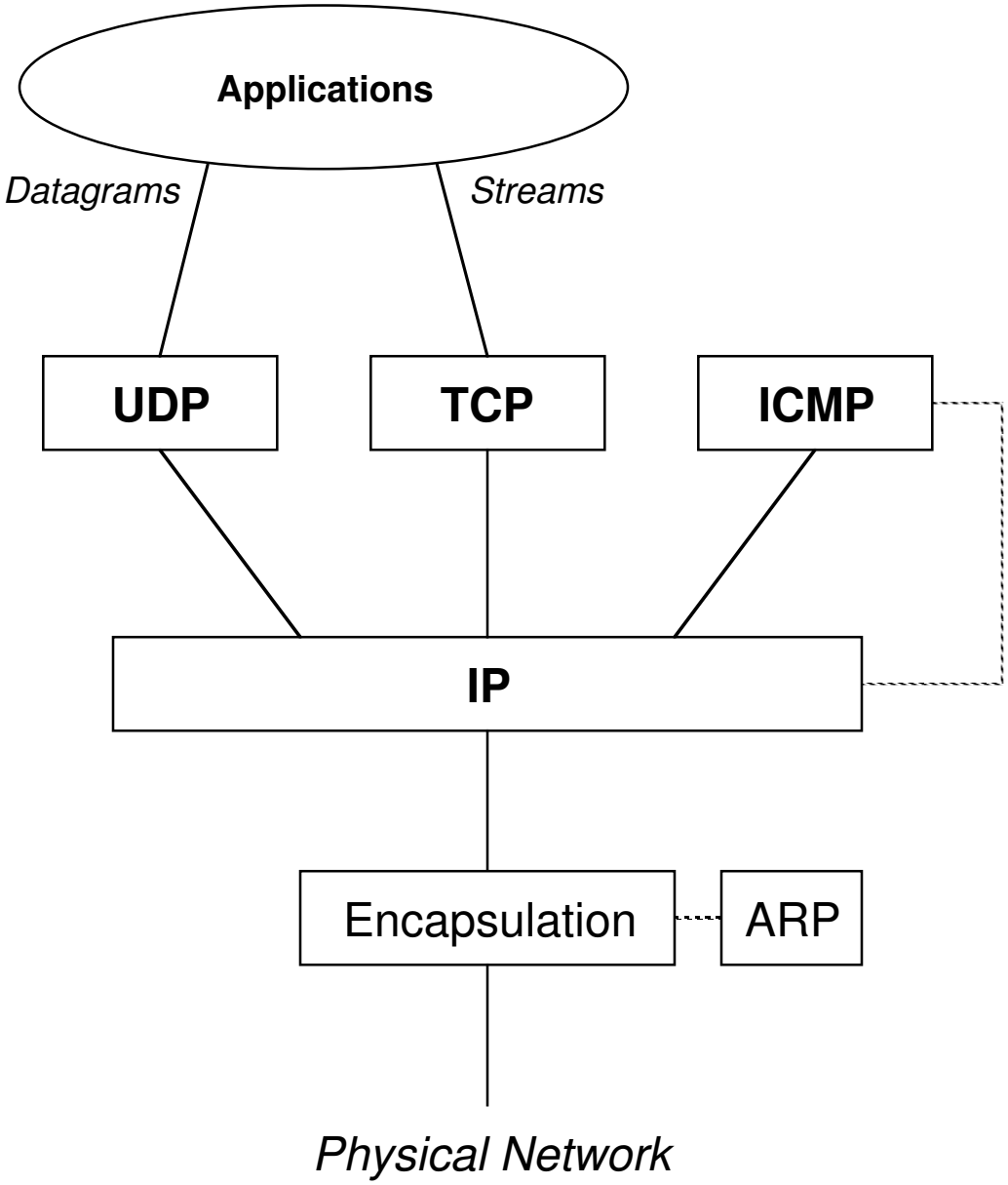
- **Internet Control Message Protocol (ICMP)**

  For indicating status at the IP layer – NOT a user protocol

  Adds even less to IP

  Main messages:

  - Host/Network Unreachable (no route)
  - Time exceeded (too many hops)
  - Source quench – primitive flow control
  - "Echo request" elicits an "Echo response" – Ping!

  ICMP errors are not sent when unable to deliver other ICMP datagrams

The full TCP/IP stack