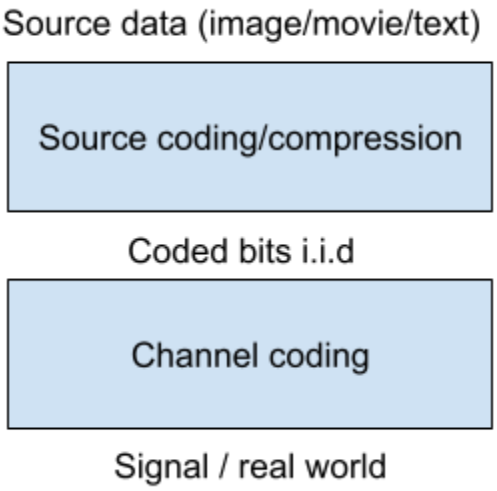
- Last time: Physical layer - clock synchronization and Shannon's capacity
- Today: The abstraction of the Physical Layer
    - TCP: "unreliable" datagrams -> "reliable" Byte Streams
    - Physical Layer: "unreliable" signals —---(coding/modulation)------> "reliable" bits
        - One argument: different coding/modulation for different data (email vs images)
        - Shannon (again): in fact, coding/modulation can be picked independent of the meaning of the bits



The two layers are **independent**. This is the same as the stack of service abstractions in previous lectures, that each layer only cares about what comes in and what goes out.

-
- Another "nice" abstraction: the sender of an Ethernet packet can send without caring about the receiver's clock
    - This is similar to ByteStream: writer and reader of the ByteStream can operate at a different speed, and no synchronization is needed between the writer side and the reader side.
    - A buffer between the receiver and the processor that is reading from the receiver (similar to the data structure for storing bits you have in your ByteStream) makes this nice abstraction possible - (**Elasticity Buffer**)
- It's hard to buy a clock that operates at the frequency it claims it operates at (electronics are cheap because expensive ones does not go to consumer markets)
    - Therefore it is hard to synchronize the rate of the sender and the receiver, since they would have clocks of different frequencies (but only differ by a little bit)
    - How can we control the size of the elasticity buffer given this? (<u>and without feedback</u>)
        - Feedback is too slow at the signal level
- **Inter-packet gap**: the sender has to pause between two packets
- **Maximum transmission unit** (MTU): the maximum size of a single packet
- What is the size of an elasticity buffer given the gap and MTU?
    - Given the gap and MTU, we need to avoid both 1) the buffer being filled up (overflow) and 2) the buffer being drained (underflow).
    - Hold the buffer at $\frac{B}{2}$:

        - For a new packet, allow the buffer to fill up to $\frac{B}{2}$ or
        - Drains the buffer to $\frac{B}{2}$ before a new packet
        - (I think this is equivalent to saying the receiver drains the buffer when buffer size is greater than or equal to $\frac{B}{2}$, therefore the inter-packet gap is not part of the calculation for buffer size, but is there to allow the two side can synchronize around the $\frac{B}{2}$ buffer size.)
        - (And therefore, inter-packet gap needs to be long enough for draining half of the buffer).
    - To prevent overflow (when sender is faster than the receiver):
        - $\frac{B}{2} + (R_{send} - R_{receive}) * \frac{P_{max}}{R_{send}} \leq B$
    - To prevent underflow (when receiver is faster than the sender):
        - $(R_{receive} - R_{send}) * \frac{P_{max}}{R_{receive}} \leq \frac{B}{2}$
    - $|R_{send} - R_{receive}| \leq R_{max} - R_{min}$ (R_{max} is the maximum clock rate and R_{min} is the minimum clock rate, and R_{send} and $R_{receive}$ is roughly the same to the claimed clock rate R).
    - Therefore: $(R_{max} - R_{min}) * \frac{P_{max}}{R} \leq \frac{B}{2}$ guarantees that there is no overflow or underflow
    - And $\frac{R_{max} - R_{min}}{R} = 2 * clock\ tolerance$
- How to multiplex the physical link?
    - Time Division Multiplexing (TDM) (or Time Division Multiple Access (TDMA)): divide the link by time usage, so each connection uses the link for a given time
    - Frequency Division Multiple Access: each connection uses a limited range of frequencies
    - Carrier Sense Multiple Access (CDMA): first listen, if no one is using the link, send the packet