Principles of reliable data transfer

(a) provided service
Principles of reliable data transfer

- Characteristics of unreliable channel will determine complexity of reliable data transfer protocol
Principles of reliable data transfer

- characteristics of unreliable channel will determine complexity of reliable data transfer protocol (rdt)
Reliable data transfer: getting started

**send side**

- **rdt_send()**: called from above, (e.g., by app.). Passed data to deliver to receiver upper layer.

**receive side**

- **deliver_data()**: called by rdt to deliver data to upper layer.
- **rdt_rcv()**: called when packet arrives on rcv-side of channel.

- **udt_send()**: called by rdt, to transfer packet over unreliable channel to receiver.
Reliable data transfer: getting started

We’ll:
- incrementally develop sender, receiver sides of reliable data transfer protocol (rdt)
- consider only unidirectional data transfer
  - but control info will flow on both directions!
- use finite state machines (FSM) to specify sender, receiver

**state:** when in this “state” next state uniquely determined by next event

**event causing state transition**

**actions taken on state transition**

---

state 1

---

state 2

CMPE 150 Winter13
**Rdt1.0**: reliable transfer over a reliable channel

- **underlying channel perfectly reliable**
  - no bit errors
  - no loss of packets

- **separate FSMs for sender, receiver**:  
  - sender sends data into underlying channel  
  - receiver reads data from underlying channel

```
Wait for call from above
rdt_send(data)
packet = make_pkt(data)
udt_send(packet)
```

```
Wait for call from below
rdt_rcv(packet)
extract (packet, data)
deliver_data(data)
```

**CMPE 150 Winter13**