

PRINCÍPIOS DE TRANSFERÊNCIA CONFIÁVEL

rdt 2.0

- Canal em que os bits podem ser **corrompidos**
- Assume que os bits foram recebidos na ordem que foram enviados.
- Usa reconhecimentos positivos e negativos. (**ACK = 1 e NAK = 0**)
- Faz o remetente saber o que foi e não foi recebido.
- Usa Protocolos **ARQ**
 - **Automatic Repeat reQuest**
- Quando está no estado de espera (ACK e NAK), ele não recebe dados da camada superior.
- Conhecido como **stop-and-wait**
- Não trata pacotes corrompidos. A solução seria adicionar um novo campo ao pacote de dados que coloca os pacotes em sequência.

rdt 2.1

- Versão corrigida do rdt 2.0

rdt 2.2

- Não precisa do NAK, pois ele pode enviar um ACK para o último pacote que foi recebido corretamente.
- Dessa forma, o remetente entende que, um ACK duplicado do mesmo pacote, significa que o destinatário não recebeu corretamente.
- Agora o destinatário precisa incluir o número de sequência do pacote que está sendo reconhecido por uma mensagem

ACK

- O remetente precisa verificar o número de sequência do pacote que está sendo reconhecido pelo ACK recebido.

rdt 3.0

- Esse canal cuida também de perda de pacotes.
- Remetente faz uma escolha de tempo em que um pacote poderia ser perdido.
- Se não receber um ACK nesse tempo ele envia o pacote de novo, possibilitando pacotes duplicados.
- Aciona um temporizador sempre que um pacote for enviado.
- Responde a interrupções e faz o que for necessário para resolver elas.
- Para o temporizador.
- Conhecido como protocolo bit alternante

Protocolos de transferência confiável de dados com paralelismo

- Todos esses protocolos ainda possuía o mesmo problema de desempenho, por isso não é adotado o tipo **stop-and-wait**.
- A solução para isso é enviar vários pacotes sem esperar os



reconhecimentos para enviar os outros pacotes.

- Se um remetente for autorizado a transmitir 3 pacotes antes de esperar os reconhecimentos, sua utilização é **triplicada**.
- **Faixa de números de sequência** se amplia.
- Precisam de **buffers** para mais de um pacote.
- A faixa de números de sequência e os buffers dependerão de como o protocolo responde a pacotes perdidos.
- Para recuperar esses dados, as duas abordagens comuns são **Go-back-N** e **repetição seletiva** (não aprofundados)