Chp. 02 – Estrutura de Protocolo

2.1 – Introdução
2.2 – Elementos de um Protocolo
2.3 – Exemplo de Protocolo
2.4 – Serviço e Ambiente
2.5 – Vocabulário e Formato
2.6 – Regras Procedimentais
2.7 – Projeto Estruturado de Protocolo
2.8 – Regras de Projeto de Protocolos

Referências Bibliográficas

- Gerard J. Holzmann – Design and Validation of Computer Protocols – Prentice Hall; Englewood Clifs; New Jersey; 1991.

- Paulo Coelho - “Material de Aula” - Arquitetura de Redes de Computadores (FACOM49070 - Mecatrônica)
- Pedro Frosi - “Material de Aula” - Arquitetura de Redes de Computadores (GBC056 - Ciência da Computação)

2.1 - Introdução

- “premissa” - … sem conhecermos o meio de transmissão não há como especificar o conjunto de regras e procedimentos que governam a troca de informação entre elementos pares ! ! !
  - … esta asserção de fato é verdadeira ?!
  - … conhecendo-se ou não o meio, é necessário definir o conjunto de regras e procedimentos que governam as interações.

… 2.1 - Introdução

- “premissa” - … sem conhecermos o meio de transmissão não há como especificar o conjunto de regras e procedimentos que governam a troca de informação entre elementos pares ! ! !
  - … esta asserção de fato é verdadeira ?!

- “protocolo” - … independente do meio de transmissão, faz-se necesssário definir um conjunto de regras:
  - … como mensagens são codificadas;
  - … como uma transmissão é iniciada e encerrada.

… 2.1 - Introdução

- “erros díficeis de evitar” - … são 02 os erros difíceis de evitar no projeto de protocolos, por estarem quase sempre presentes.
  - conjunto incompleto de regras e procedimentos;
  - regras / procedimentos que são contraditórias.

- “objetivo” - … garantir que o conjunto de regras e procedimentos seja, ao mesmo tempo, completo e consistente ...
  - precisão na especificação;
  - modularização das regras;
  - estruturação do protocolo.

… 2.1 - Introdução

- e.g., … Servidor de Arquivo e Servidor de Impressão

- problemas físicos - … cabeamento, codificação, transmissão, velocidade de transmissão, etc.
- problemas de controle - … impressora disponível ?! taxa de transmissão ?! sem papel ?! suspensão?! comunicação apenas no sentido A → B é suficiente ? … etc.

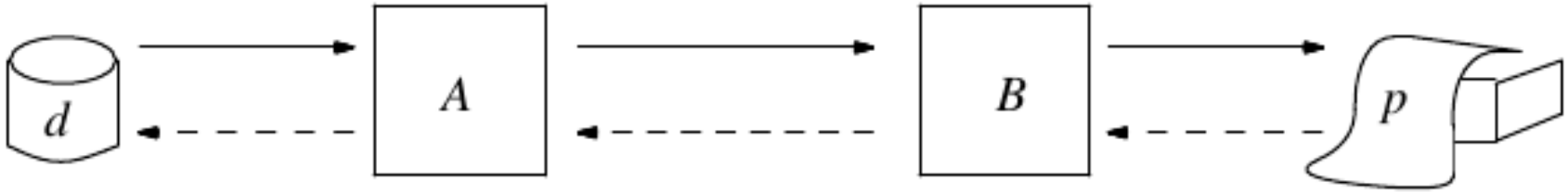


Figure 2.1 — File Server and Print Server

… 2.1 - Introdução

- “protocolo” - … definido pelo conjunto de formatos, mensagens, regras e procedimentos acordado entre os pares.
  - … em outras palavras, o protocolo formaliza as interações padronizando o uso (transmissão/recepção) do canal de comunicação.
- Protocolo pode conter acordos sobre os métodos para:
  - inicialização e finalização a troca de dados;
  - sincronização do transmissor e receptor;
  - detecção e correção de erros de transmissão;
  - formatação e codificação dos dados.

- Obs.: … todas estas fases podem ser definidas em um ou mais níveis de abstração do protocolo.

… 2.1 - Introdução

- Níveis Básicos de Abstração – Formatação.

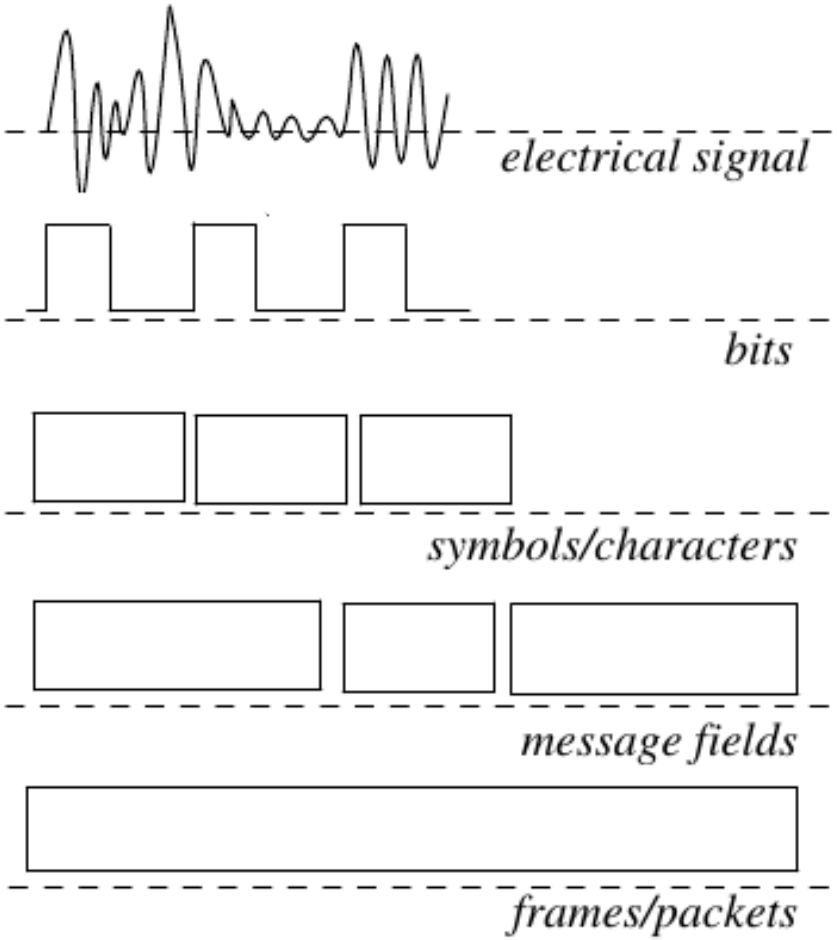


Figure 2.2 — Sample Levels of Abstraction: Formatting

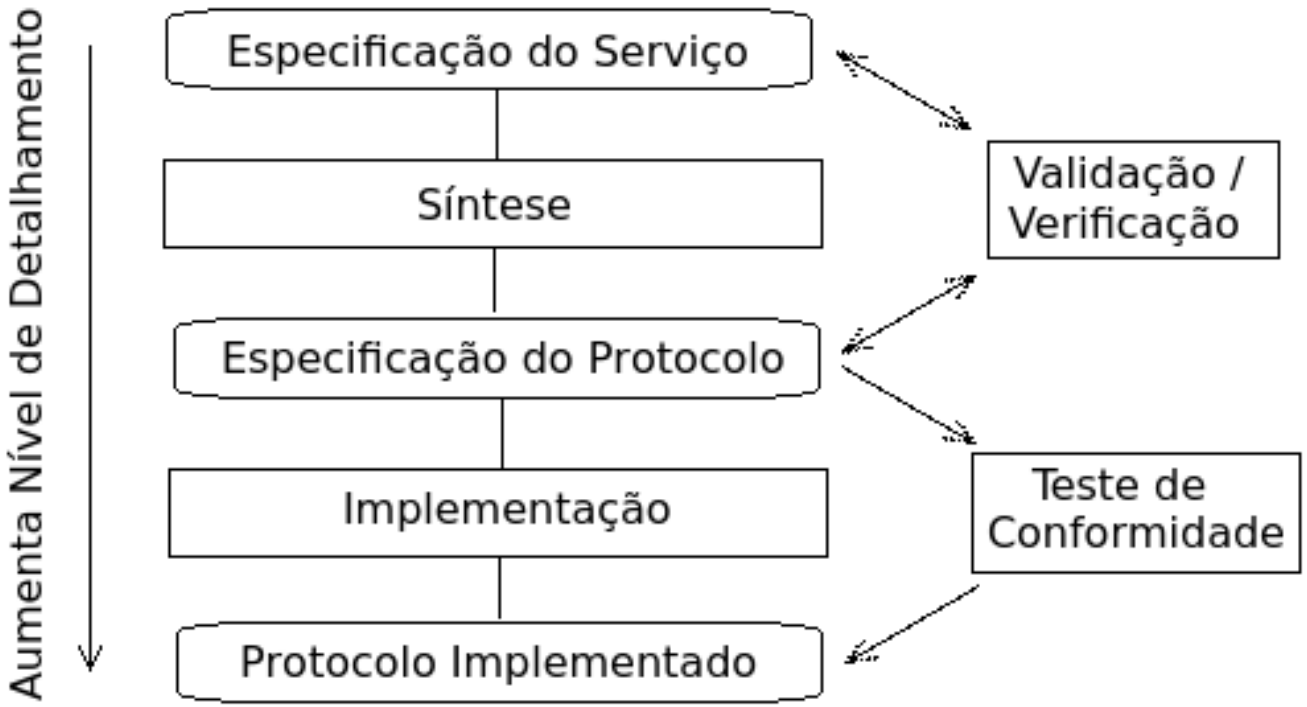
2.2 – Elementos do Protocolo

- “especificação” - ... consiste de **05** elementos distintos:
  - *serviço* a ser oferecido pelo protocolo;
  - definição do *ambiente* no qual o protocolo será executado;
  - *vocabulário* das primitivas usadas para implementar o protocolo;
  - *codificação* (formato) de cada primitiva no vocabulário;
  - *regras procedimentais* → consistência na troca de primitivas.

- Obs.: … último elemento da especificação do protocolo é o mais difícil de projetar e o mais complicado de verificar.
  - razões .. problemas de temporização, condições de corrida, *deadlocks*, problemas decorrentes da concorrência entre os pares, etc.

… 2.2 – Elementos do Protocolo

- “síntese do protocolo” - … geração da especificação do protocolo a partir da especificação do serviço do protocolo.



2.3 – Exemplo de um Protocolo

- W. C. Lynch [1968] - … especificação do **serviço**
- “objetivo” - … transferir arquivos de texto como uma sequência de caracteres através de uma linha telefônica sem erros de transmissão, ou seja, erros podem ser detectados.

- “premissa” - … canal no qual a transferência de mensagens acontece é “full-duplex”, ou seja, ambos os sentidos.
- “premissa” - … Acks e Nacks são enviados pelo canal (B → A) em reconhecimento às transmissões de A → B, e vice-versa.
- “premissa” - … 02 tipos mensagens - msg. propriamente dita (contém informação) e msg. de controle no sentido inverso.

… 2.3 – Exemplo de um Protocolo

- **Ambiente**
- … consiste minimamente de 02 usuários do serviço de transferência de arquivo e um canal de transmissão;
- … usuários submetem uma requisição para transferência do arquivo e esperam até a sua conclusão;
- … canal de transmissão pode causar distorções arbitrárias nas primitivas, mas não perde, não duplica, não insere e nem reorganiza estas primitivas (mensagens);
- … assume-se que um nível inferior é o responsável por corrigir distorções e gerar mensagens de erros.

… 2.3 – Exemplo de um Protocolo

- **Vocabulário** .. define três tipos de mensagens:
  - Ack – mensagem para reconhecimento positivo;
  - Nack – mensagem para reconhecimento negativo;
  - Err - mensagem para primitivas de erro.

- V = { Ack, Nack, Err }
- … cada tipo de primitiva ou mensagem pode posteriormente ser refinada em várias mensagens de níveis inferiores.

… 2.3 – Exemplo de um Protocolo

- **Síntaxe (formato)** da Primitiva
  - campo de controle - identifica o tipo da primitiva;
  - campo de dados - conjunto de códigos de caracteres.

- Representação “C-like” para uma primitiva:

```
enum control { Ack, Nack, Err };
struct primitiva {
  enum control tag;
  unsigned char data;
};
```

… 2.3 – Exemplo de um Protocolo

- **Regras Procedimentais** - … informalmente descritas como:

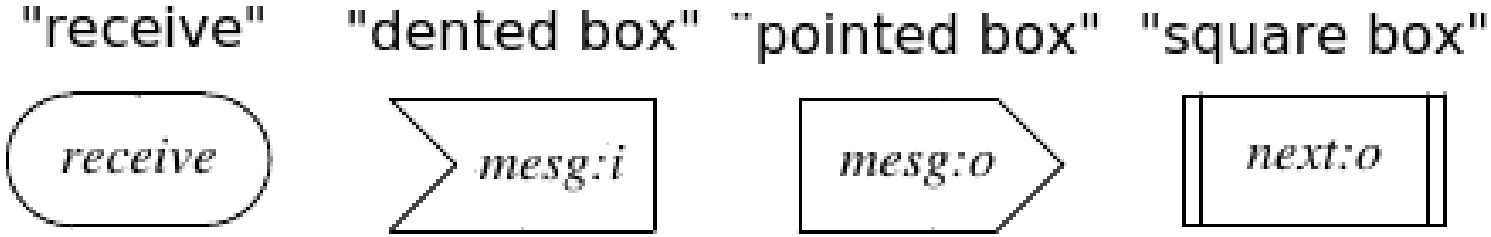
  01. se a recepção anterior foi livre de erro, a próxima msg. no sentido contrário levará um ACK; caso contrário a próxima msg. no sentido contrário levará um NACK.

  02. se a recepção anterior foi um NACK, ou foi uma primitiva de ERR, retransmite a primitiva antiga; qualquer outro caso procure por uma nova primitiva para transmissão.

- Para formalizar estas regras podemos usar diagramas de estado, expressões algébricas, descrição na forma de programa, etc.

… 2.3 – Exemplo de um Protocolo

- “receive” - representa o estado no qual a recepção de uma nova mensagem do canal está sendo aguardada;
- “dented box” - representa o reconhecimento de uma msg. que está associada - “match” com o rótulo - “label” da caixa;
- “pointed box” - indica a transmissão de uma msg. cujo tipo é indicado pelo rótulo - “label” da caixa;
- “square/retangle box” - indica uma ação interna para obter o próximo item de dado, p.ex., caracter a ser transferido.



… 2.3 – Exemplo de um Protocolo

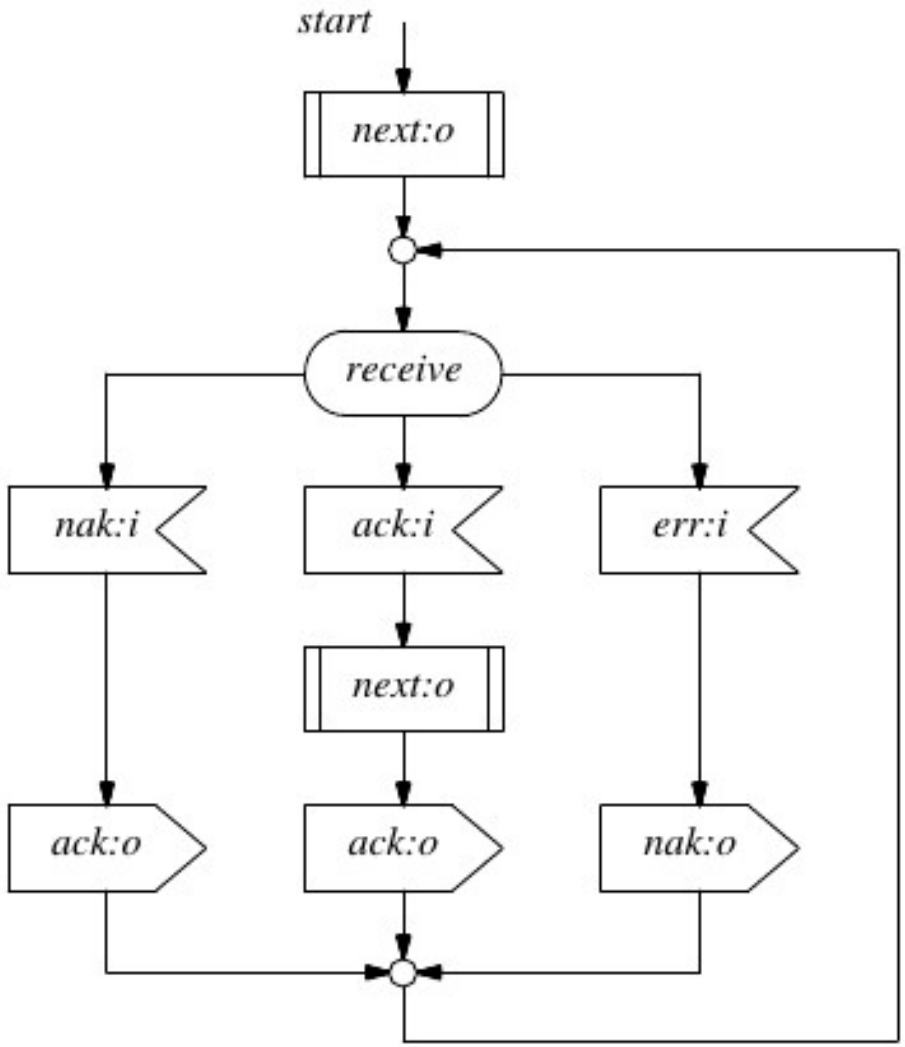


Figure 2.3 — Lynch's Protocol

… 2.3 – Exemplo de um Protocolo

- Falhas do Projeto de Protocolo:
  - … transmissão de dados em um sentido só pode começar se houve uma transmissão de dados no sentido contrário;
  - “dúvida” - … qual dos pares decide como o canal vai ser iniciado ou encerrado ?!

- ?! Para melhor entender/perceber o cenário do início da troca de mensagens, proponha uma sequência de ações entre os 02 pares.
  - … ainda assim, considerando que haja uma sequência para iniciar a troca de mensagens, como trazer os pares de processos em fase ??
- ?! Questão semelhante pode ser colocada quanto a finalização da troca de mensagens entre os pares …
  - ... ainda assim, requer mensagens extras de controle.

… 2.3 – Exemplo de um Protocolo

- Maior deficiência:
- … receptor deve decidir se um dado que foi ou não recebido corretamente e armazenado temporariamente na variável “i”, deve ser aceito (por exemplo, salvo em um arquivo);
- … dados duplicados recebidos corretamente, não deveriam ser aceitos (este problema não tem solução pelas regras procedimentais do protocolo, pois nada foi dito acerca deste caso).
  - … maior deficiência do protocolo é a falta de regra procedimental para o caso de duplicação de mensagens / dados.

… 2.3 – Exemplo de um Protocolo

- e.g., … considerando o protocolo especificado anteriormente, vejamos o que acontece com os 02 processos abaixo:

  1. “A” inicia a transferência enviando deliberadamente uma mensagem de erro para “B”;

  2. “A” tenta transmitir letras de “a” a “z” e “B” responde enviando na ordem inversa, ou seja, de “z” a “a”.

- Diagrama da Fig. 2.4 mostra a sequência de eventos que leva a uma duplicidade ser aceita como correta.

… 2.3 – Exemplo de um Protocolo

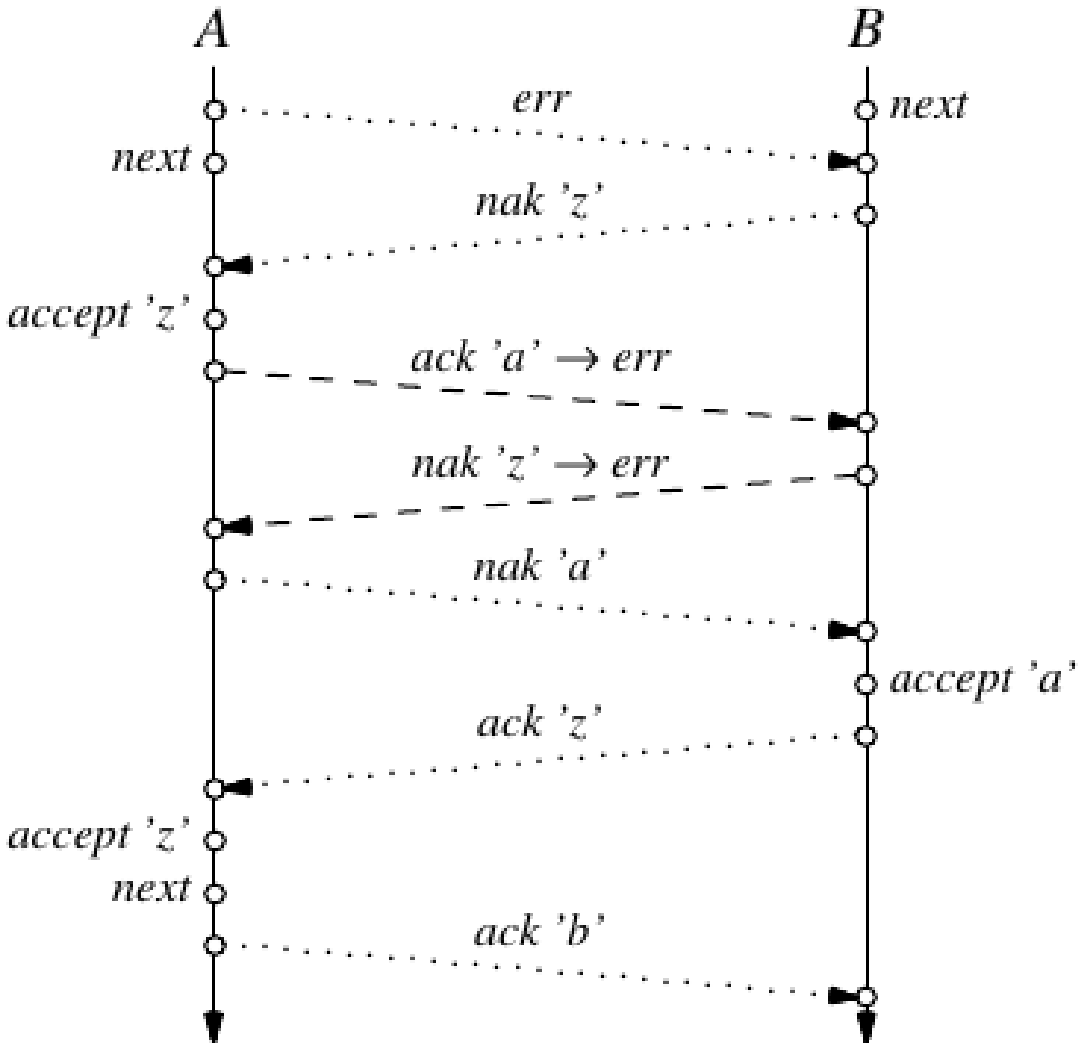


Figure 2.4 — Time Sequence Diagram

… 2.3 – Exemplo de um Protocolo

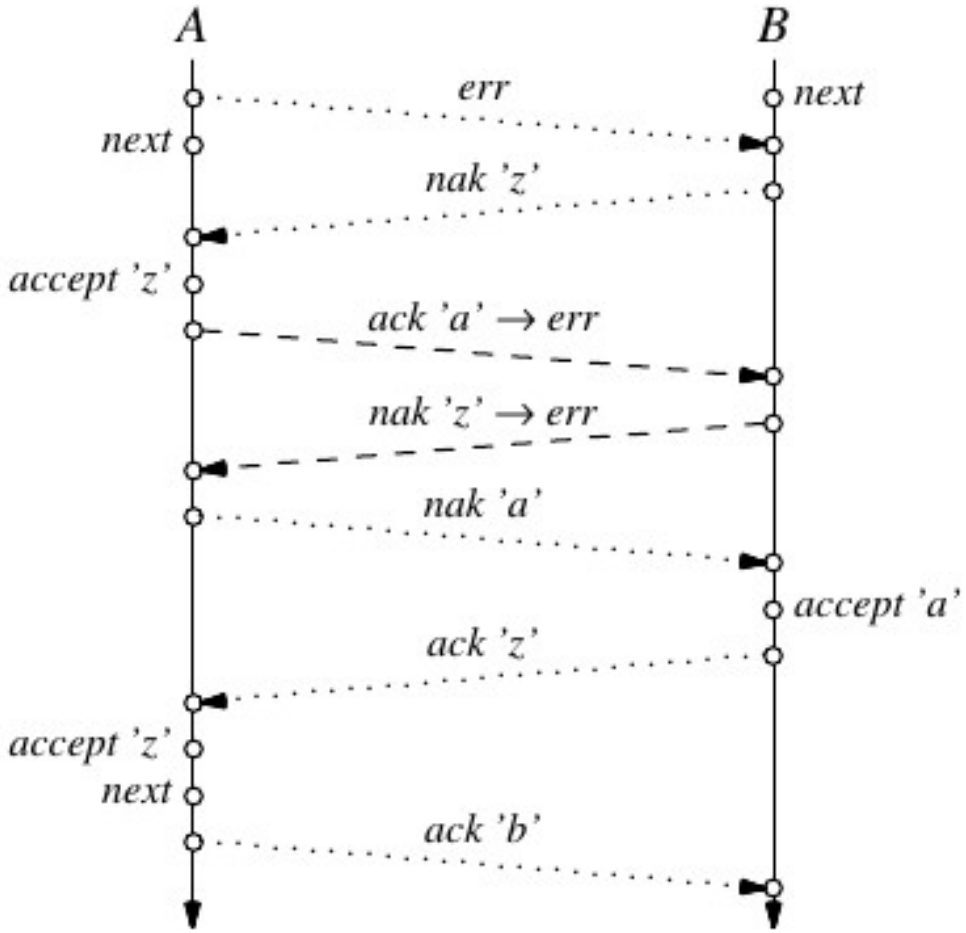


Figure 2.4 — Time Sequence Diagram

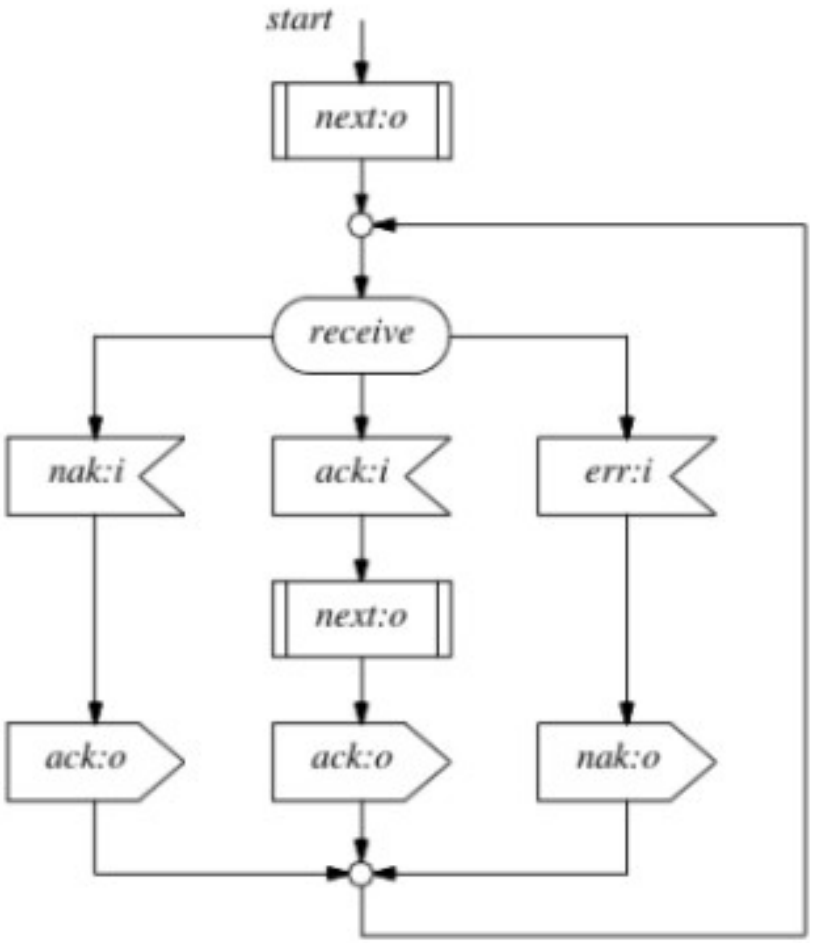


Figure 2.3 — Lynch's Protocol

… 2.3 – Exemplo de um Protocolo

- Diagrama da Fig. 2.4 mostra a sequência de eventos que leva a uma duplicidade ser aceita como correta.

- Considerações acerca do Protocolo:
  - protocolo apresentado é muito simples;
  - descrição informal é convicente, sendo que baseado nesta descrição poucos duvidariam da “corretude” do mesmo;
  - especificação incompleta .. qualquer implementação baseada nesta proposta resulta em erros subtos durante a troca de dados;
  - mostra-se que mesmo o mais simples dos protocolos exige uma boa disciplina de projeto e ferramentas analíticas eficazes.

… 2.3 – Exemplo de um Protocolo

- Exercício: … Identifique e descreva os cinco elementos para o protocolo que descreve a operação de impressão remota.
  - Impressão Remota apresentada anteriormente.

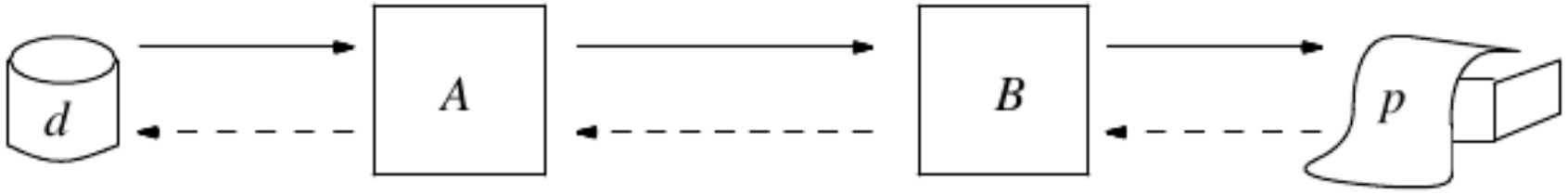


Figure 2.1 — File Server and Print Server

2.4 – Serviço e Ambiente

- Se um problema é muito grande para ser abordado de uma vez; ele pode ser quebrado em problemas menores …
- … muitas funções abstratas são definidas e implementadas em termos de construtores subjacentes, onde cada nível “esconde” certas propriedades indesejáveis do canal de comunicação;
  - … canal de comunicação se transforma em um meio mais idealizado.

- e.g., … considere um protocolo para transmissão de dados que oferece codificação de caracteres em tuplas de 7 bits e um rudimentar detector de erro baseado em paridade;
- … são 02 os serviços: codificação e detecção de erros.

… 2.4 – Serviço e Ambiente

- Pode-se separar estes serviços em dois módulos funcionais, chamados sequencialmente, construindo um canal virtual.
- … de fato, cada camada oferece um serviço diferente e implementa um protocolo separado;
  - 1º Camada implementa P1 → formato 8 bits;
  - 2º Camada implementa P2 → formato 7 bits.

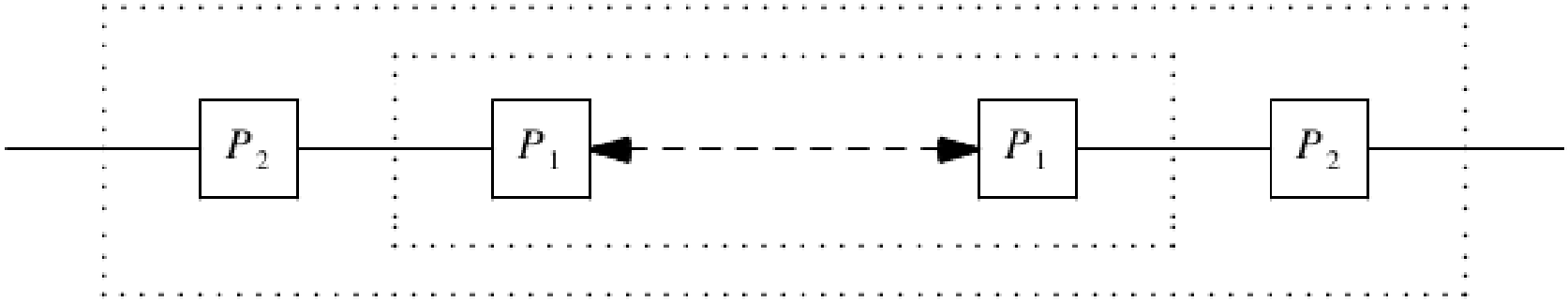


Figure 2.5 — Building a Virtual Channel

… 2.4 – Serviço e Ambiente

- Processo P2 não vê e não sabe nada sobre o 8º bit adicionado e controlado pelo Processo P1
  - … única coisa que P1 sabe é que o “canal” é mais confiável do que diretamente sobre o meio de comunicação abaixo.

- Processo P1 oferece um canal virtual para P2, mas, ao mesmo tempo, P1 é transparente para P2.

- 02 conceitos fundamentais em protocolo:
  - transparente - alguma coisa que existe, mas parece não existir.
  - virtual - alguma coisa que parece existir, mas não existe de fato.

… 2.4 – Serviço e Ambiente

- Para P1, o significado dos bits não importa - apenas o número de bits; de modo similar nem P1, nem P2, conhece qualquer coisa a respeito do protocolo de nível superior.
- … cada nível encerra os dados transmitidos em uma nova “capsula”, consistindo de um “header” e/ou “trailer”, antes de passá-lo a um próximo nível.
- … formato de níveis superiores não precisam ser preservados.

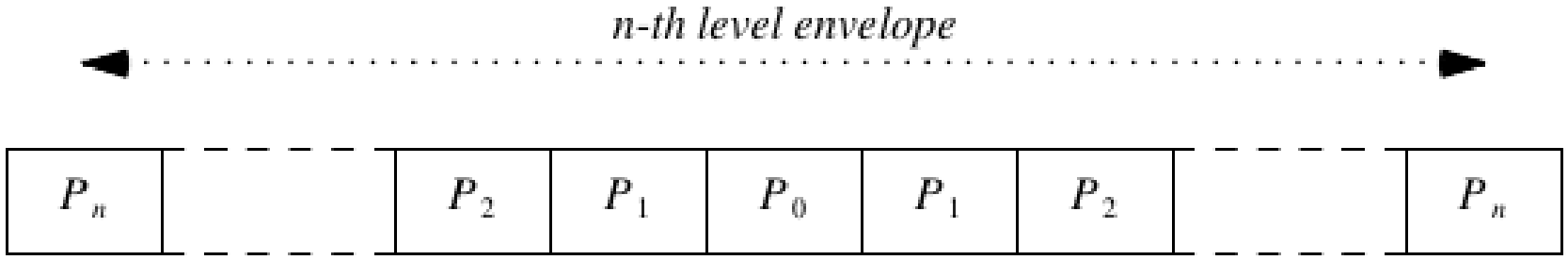


Figure 2.6 — Data Envelopes

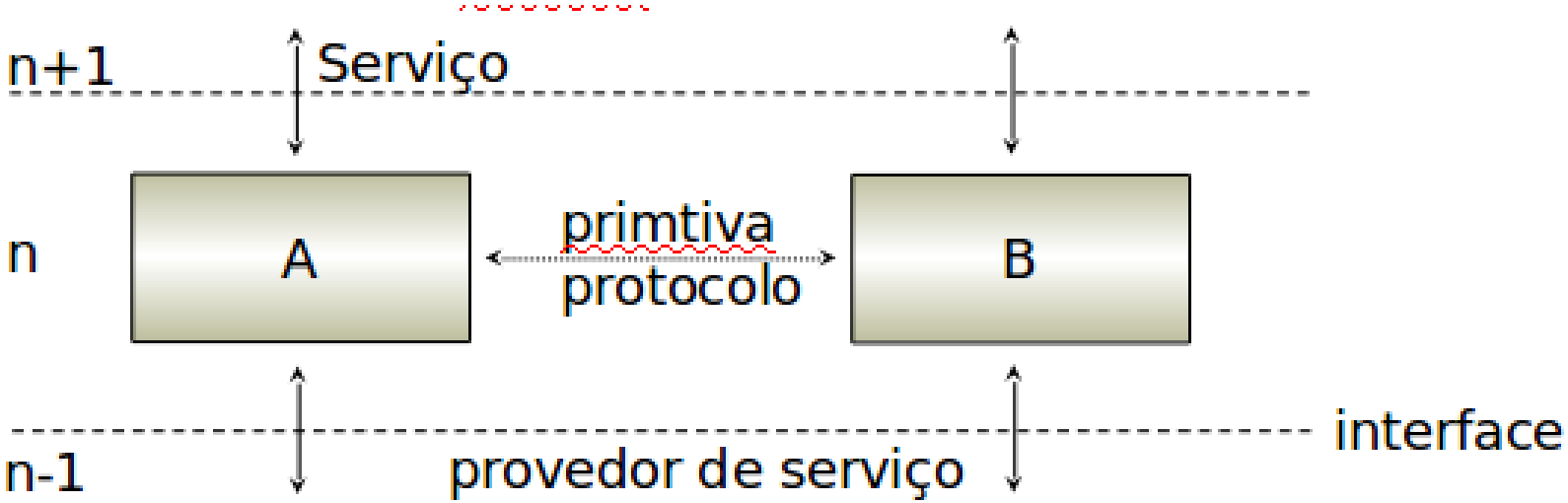
… 2.4 – Serviço e Ambiente

- “princípio de projeto hierárquico” - … bem conhecido em programação sequencial e modular, mas novo em sist. distribuídos.
- … camadas ajudam a indicar a estrutura lógica do protocolo separando detalhes de alto-nível dos de baixo-nível;
- … quando o protocolo é estendido ou requer correções/ajustes, é mais fácil reescrevê-lo ou trocá-lo.

- No início dos anos 70, a ISO reconheceu a necessidade de padronização e criou o Modelo OSI baseado em camadas ...

… 2.4 – Serviço e Ambiente

- Cada camada define um conjunto de serviços distintos e implementa protocolos específicos para aquela camada.
- Formato usado por uma camada é completamente independente dos formatos usados pelas demais camadas.
- e.g., … camada de rede envia pacotes; camada de enlace envia “frames” … e assim por diante.



2.5 – Vocabulário e Formatos

- Cabeçalhos e “Trailers”
- … com os métodos de estruturação descritos, métodos de formatação de alto nível mais sistemáticos podem ser construídos.
- e.g., … considere ???? a ETX e DLE como delimitadores da mensagem bem como a ocorência de erros de transmissão:
  - … se o contador é perdido, ou,
  - … se o caracter ETX (End of Text) ou DLE (Data Link Escape) forem corrompidos, as técnicas de estruturação falham.
- Como veremos adiante, esquemas de detecção de erro requerem transmissão de informações redundantes na msg.
  - … se outros mecanismos forem contemplados, p.ex., controle de fluxo, então outros campos fazem-se necessários, p.ex. nro de sequência.

… 2.5 – Vocabulário e Formatos

- Cabeçalho e o Trailer podem ser refinados em subconjuntos ordenados chamados campos de controle.

- e.g.,
  - cabeçalho = { tipo , destino , janela , contador , prioridade }
  - trailer = { checksum , endereço de retorno }

2.6 – Regras Procedimentais

- Viu-se até este momento grande similaridade entre as tarefas de projeto de protocolo e o desenvolvimento de software;
- … diferença importante é que as regras procedimentais (*procedure rules*) são interpretadas paralelamente por dois ou mais pares (muito provavelmente em máquinas diferentes).
  - … efeito de cada nova regra adicionada ao conjunto é frequentemente muito maior do que se possa imaginar.

- Para se convencer da corretude do projeto é necessário algo melhor do que a reflexão informal.
  - … infelizmente a ferramenta mais popular para isto é o Diagrama de Ordem Temporal.

2.7 – Projeto Estruturado de Protocolos

- Projeto de Protocolos - toca em várias áreas conhecidas, mas também em outras áreas sem o completo entendimento.
- e.g., ... nível físico do modelo OSI/ISO:
  - … conhece-se precisamente qual o comportamento padrão dos diferentes tipos de informação que são “levados”;
  - … quão rápido pode-se transmitir dados neles;
  - … qual a taxa média de “bit error” resultante.

- Há várias técnicas para codificação binária em sinais analógicos.
- Conhece-se bem técnicas de sincronização *sender/receiver.*

… 2.7 – Projeto Estruturado de Protocolos

- Acima da Camada Física depara-se com problemas, p.ex., controle de acesso ao meio ou problemas de projeto de rede:
  - … roteamento através de redes;
  - … dimensionamento preciso da estrutura de redes;
  - … interconexão de várias redes via gateways;
  - … desenvolvimento em um nível superior de disciplinas para controle de fluxo e controle de congestionamento.

- Obs.: … existem técnicas que podem resolver problemas na camada física, contudo os problemas estão apenas no início.
  - … propôr um conjunto de regras completo e não ambíguas para troca de “informação” em um sistema distribuído é muito difícil e complexo.

… 2.7 – Projeto Estruturado de Protocolos

- Simplicidade - … caso para protocolos *Light-weight*
- … um protocolo bem estruturado pode ser feito a partir de um pequeno número de “pedaços” bem projetados e bem conhecidos.
- … para entender o protocolo basta entender os pedaços;
- … protocolos feitos deste modo são mais fáceis de entender, de implementar e, mais apropriados para verificar e manter.

- Protocolo “Light-Weight” - … simples, robusto e eficiente.

… 2.7 – Projeto Estruturado de Protocolos

- Modularidade - … hierarquia de funções.
- … um protocolo que executa uma função complexa pode ser feito de pedaços que interagem de um modo simples e bem definido;
- … cada módulo, um pedaço, é um protocolo “light weight”;
- … cada módulo individualmente não faz suposições sobre o trabalho de outros, nem mesmo a presença;
- … funções ortogonais não podem ser misturadas, elas são projetadas como entidades independentes;
- … controle de erro e controle de fluxo são funções ortogonais.

… 2.7 – Projeto Estruturado de Protocolos

- Protocolo Bem-Formado - um protocolo bem formado NÃO É:
- … “over-specified”, ou seja, não há regras não alcançáveis ou não utilizadas no conjunto de todas as regras.
- … “under-specified” ou incompleto, ou seja, durante sua execução podem ser requeridas regras que levarão a uma recepção indefinida (não especificada).
- … “bounded” - não pode ultrapassar limites definidos do sistema (ambiente), como capacidade da fila de mensagens.
- … “self-stabilizing” - quando erro arbitrário muda o estado do protocolo, este deve retornar a um estado conhecido.
- … “self-adapting” - pode se adaptar em certas circunstâncias.

… 2.7 – Projeto Estruturado de Protocolos

- Como disse Polybius - "*It is chiefly unexpected occurrences which require instant consideration and help.*"

- Robustez - … não é difícil projetar protocolos que trabalham em circunstâncias normais.
- … é o inesperado que torna o projeto um desafio, i.e., o protocolo deve ser preparado para tratar apropriadamente todas as ações que ocorram, em qualquer sequência, sob quaisquer condições.
- … protocolo deve fazer suposições mínimas sobre o ambiente para evitar dependências de características que mudam.

… 2.7 – Projeto Estruturado de Protocolos

- Consistência - existem alguns modos “padronizados” e temidos nos quais um protocolo pode falhar:
- “deadlocks” - situação na qual não haverá um próximo estado.
- “livelocks” - sequências na execução, repetidas indefinidamente, sem fazer qualquer progresso.
- “no k-limited” – máquina de estado não finita.
- “não reiniciável” - não é possível a partir de um dado estado voltar ao estado inicial.
- “terminação imprópria” - finalização sem satisfazer as condições apropriadas de encerramento.

2.8 – Regras de Projeto de Protocolos

- Regras de Ouro para Projeto de Protocolos:

1. Certificar-se que o Problema foi bem especificado.
2. Definir os serviços em cada nível de abstração.
3. Delinear as funcionalidades externas antes das internas.
4. Manter o projeto simples.
5. Não ligar o que é independente, que tem ortogonalidade.

2.8 – Regras de Projeto de Protocolos

- Regras de Ouro para Projeto de Protocolos:

  6. Não introduzir o que “não é concreto”; não restringir o que é irrelevante; um bom projeto é “*open-ended*”; um projeto resolve uma classe de problemas.

  7. Antes de implementar, faça um protótipo de alto-nível, e verifique se os critérios de projeto foram alcançados.

  8. Implementar o projeto, medir seu desempenho, e se necessário, otimizar o projeto do protocolo.

  9. Garantir que a implementação final, otimizada, é equivalente ao projeto de alto-nível que foi verificado.

  10. Não Pular as Regras de 1 a 7 (MAIS IMPORTANTE !!)

2.8 – Regras de Projeto de Protocolos

- Nota: Regra 10 é a Regra MAIS VIOLADA !!