Princípios de Modularidade 1

Baseado em

Arndt von Staa

Sumário

- Abstração
- Níveis de abstração
- Interface
  - Conceituação
  - Relacionamento cliente-servidor
  - Sintaxe e semântica da interface
- Módulos
  - Módulo físico e módulo lógico

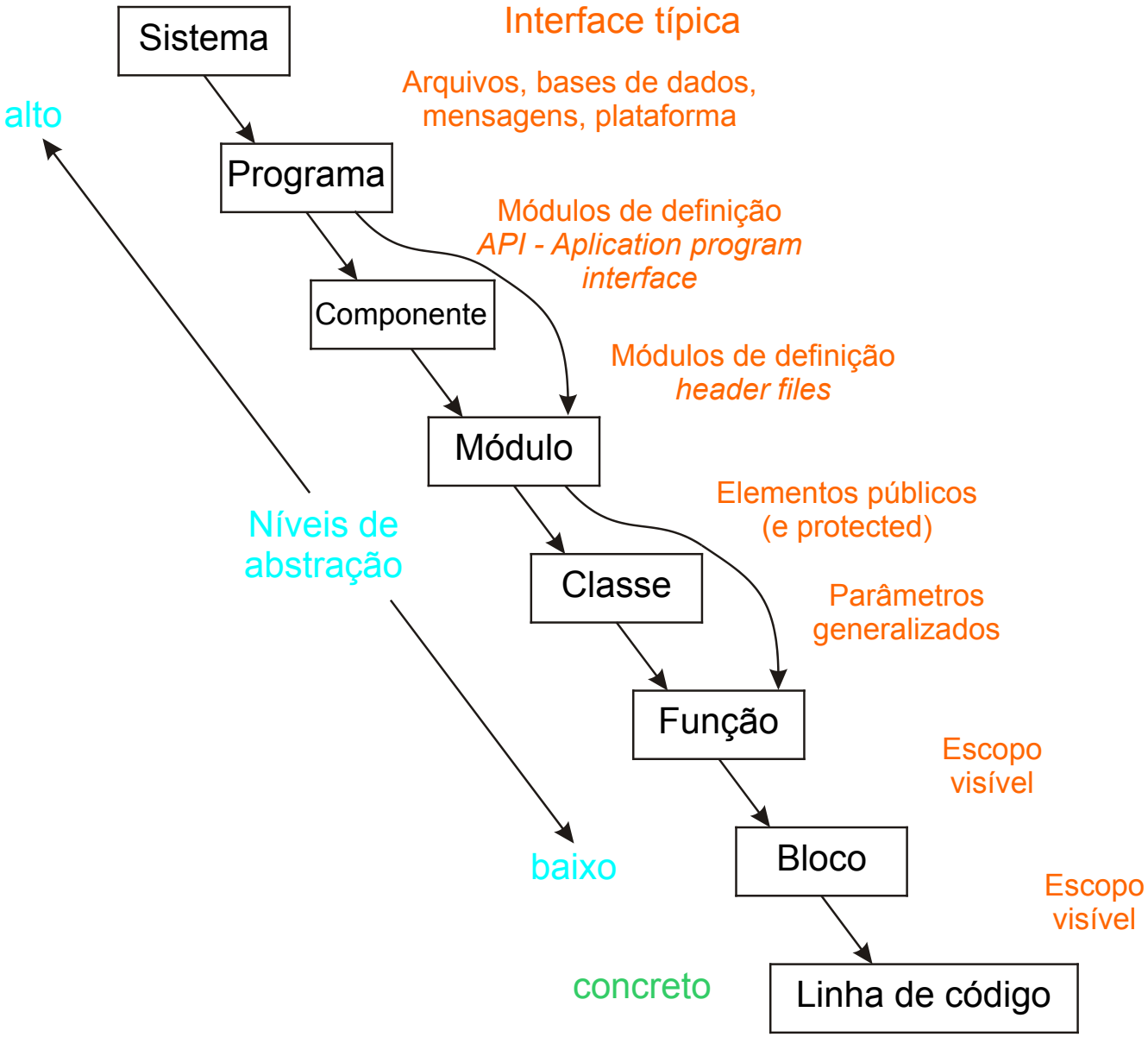
Abstração

- Abstração: omissão de detalhes, mas sem que se perca a compreensão da essência do artefato
- Entretanto, a falta de suficientes detalhes pode impedir a compreensão da sua essência
- O excesso de detalhe torna mais trabalhosa a operação com o artefato

Abstração

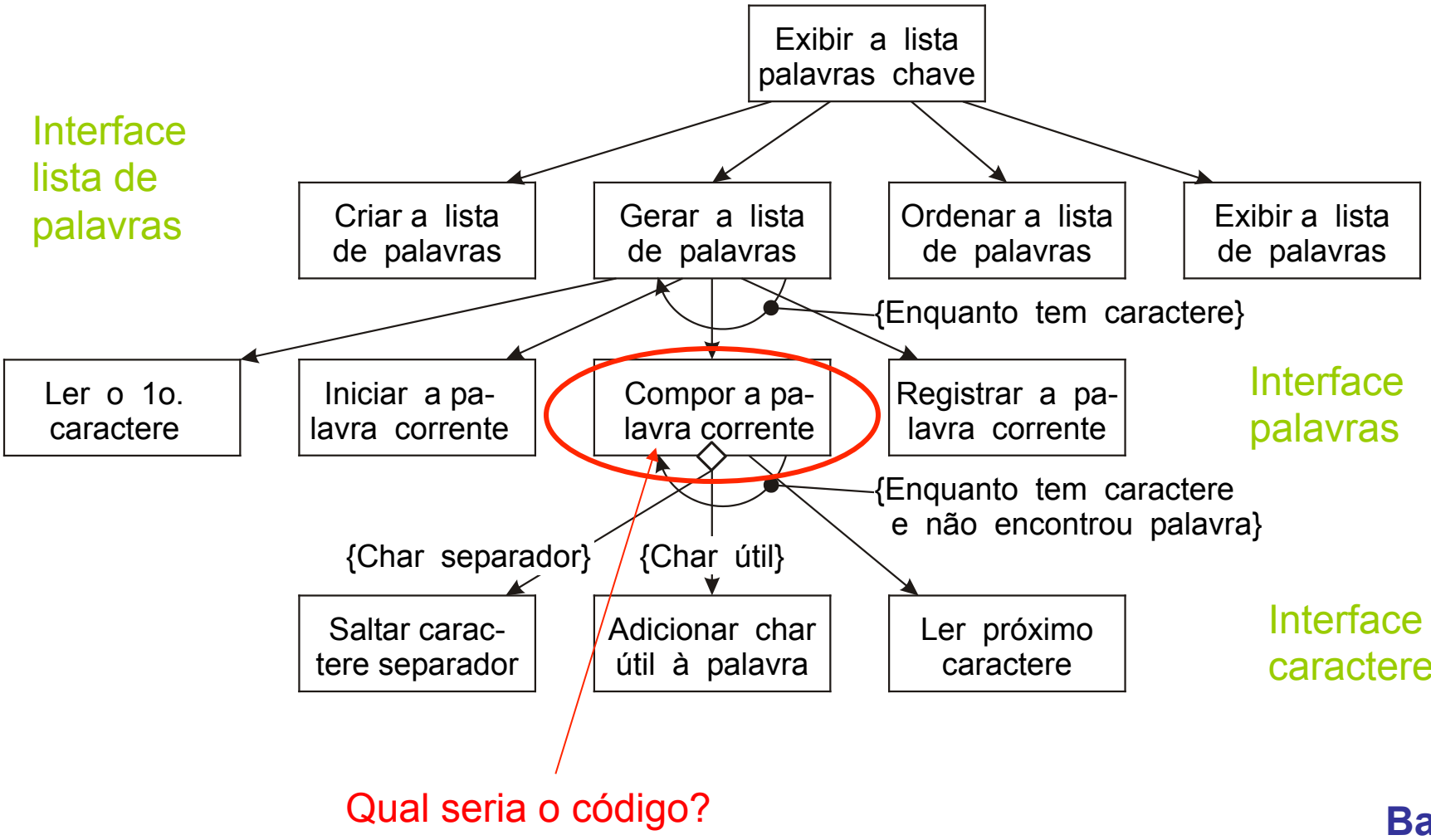
- **Definição:** “Uma abstração denota as características essenciais de um elemento que distinguem este de todos os outros elementos e, assim, provê limites conceituais bem definidos”
- **Exemplos** de abstrações em um sistema de controle de cursos
  - Curso
    - Pessoas: Nome, Endereço, Telefone, Data Nascimento, etc…
      - Alunos
        - » Curso, Data Início, Disciplinas Cursadas, Período, Status, etc…
      - Professores
        - » Formação, Tipo de Dedicação, etc…
  - Certos detalhes podem ser ignorados deste sistema:
    - Hobbies, etnia, altura, peso, etc…
- Temos diferentes **níveis de abstração** em um sistema
  - Linguagens de programação provêem suporte a estes níveis

Níveis de abstração de um sistema



Elementos de um Programa e Interfaces

Exemplo de projeto da estrutura de um algoritmo



Implementação do bloco Compor palavra

```
/* Precisa valer: inxCharPalavra == 0,
                  CharCorr o caractere a ser processado */
/* Compor a palavra chave */
   while ( CharCorr != Char_EOF )
   {
      if ( TipoCaracter( CharCorr ) == TipoUtil )
      {
      /* Adicionar caractere útil à palavra */
         Palavra[ inxCharPalavra ] = CharCorr ;
         inxCharPalavra ++ ;
      } else
      {
      /* Saltar caractere separador */
         if ( inxCharPalavra > 0 )
         {
            break ;
         } /* if */
      } /* if */
      CharCorr = LerProximoChar( ) ;
   } /* while */
/* inxCharPalavra > 0 ➔ existe palavra a registrar */
```

O que é mesmo uma interface?

- Interfaces são os mecanismos (as coisas) através das quais os artefatos interagem
  - Uma interface define os elementos visíveis necessários para a comunicação
  - Outro exemplo de interface em software é: janelas para o usuário
- Para que haja comunicação, o "cliente" e o "servidor" precisam ter um vocabulário (idioma) comum
- Uma função em um programa C:
  - "função cliente" precisa conhecer:
    - Nome da função
    - Tipode de dado utilizado pela função
    - Etc...

Implementação do bloco Compor palavra

```
/* Precisa valer: inxCharPalavra == 0,
                  CharCorr o caractere a ser processado */
/* Compor a palavra chave */
   while ( CharCorr != Char_EOF )
   {
      if ( TipoCaracter( CharCorr ) == TipoUtil )
      {
      /* Adicionar caractere útil à palavra */
         Palavra[ inxCharPalavra ] = CharCorr ;
         inxCharPalavra ++ ;
      } else
      {
      /* Saltar caractere separador */
         if ( inxCharPalavra > 0 )
         {
            break ;
         } /* if */
      } /* if */
      CharCorr = LerProximoChar( ) ;
   } /* while */
/* inxCharPalavra > 0 ➔ existe palavra a registrar */
```

O que é mesmo uma interface?

- Em computação os elementos de interface de um módulo são
  - funções ou operações
    - Exemplos
      - Abrir arquivo
      - Ativar programa xyz
  - dados
    - Exemplo: lista de palavras, lista de caracteres, caracter, etc…
  - eventos
    - uma forma de ação instantânea que informa que algo aconteceu, ex.
      - clicou sobre "abrir arquivo"
      - foi pressionado "enter" com linha de comando contendo "xyz"
      - o relógio atingiu a hora cheia
      - fim do arquivo encontrado
      - eventos excepcionais, tais como: estouro da pilha, "memory overflow",…

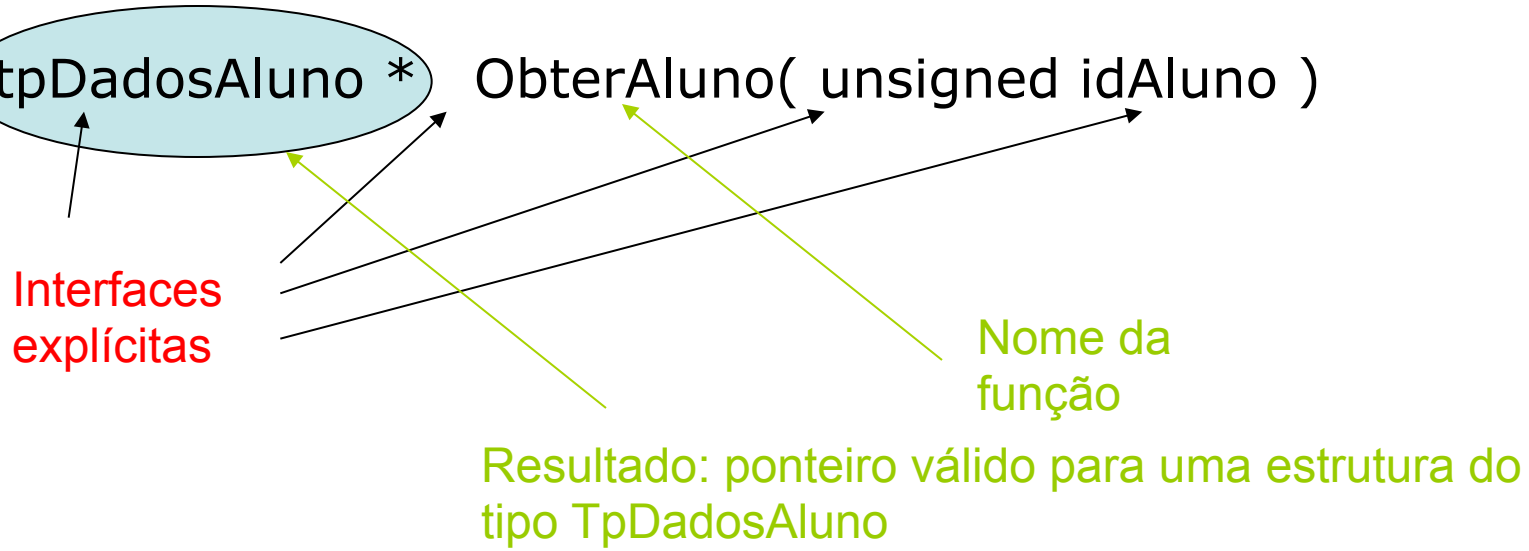
Interfaces ocorrem em diferentes níveis...

- ... de abstração em um sistema de software:
  - Entre blocos de uma função: existe uma série de variáveis que são compartilhadas por ambos
    - Obs: um bloco de estrutura de controle (e.g. *while*)pode definir variáveis locais que não são definidas e usadas pelos blocos subjacentes
  - Entre funções ou métodos de um programa
    - Itens da interface: argumentos, variáveis globais, valores de retorno
  - Entre programas: através da base de dados
  - Entre programas e usuários
  - Entre programas executando em computadores interligados em rede: chamadas de procedimento remotos.

Exemplos de interfaces

- argumentos passados para parâmetros de funções
- valores retornados por funções
- os tipos dos parâmetros
- variáveis globais
- arquivos
- bases de dados
- mensagens exibidas para o usuário
- dados recebidos do usuário
- sinais gerados internamente
  - ex. observação de um erro por uma assertiva executável
- estados de dispositivos
- ...

Exemplo

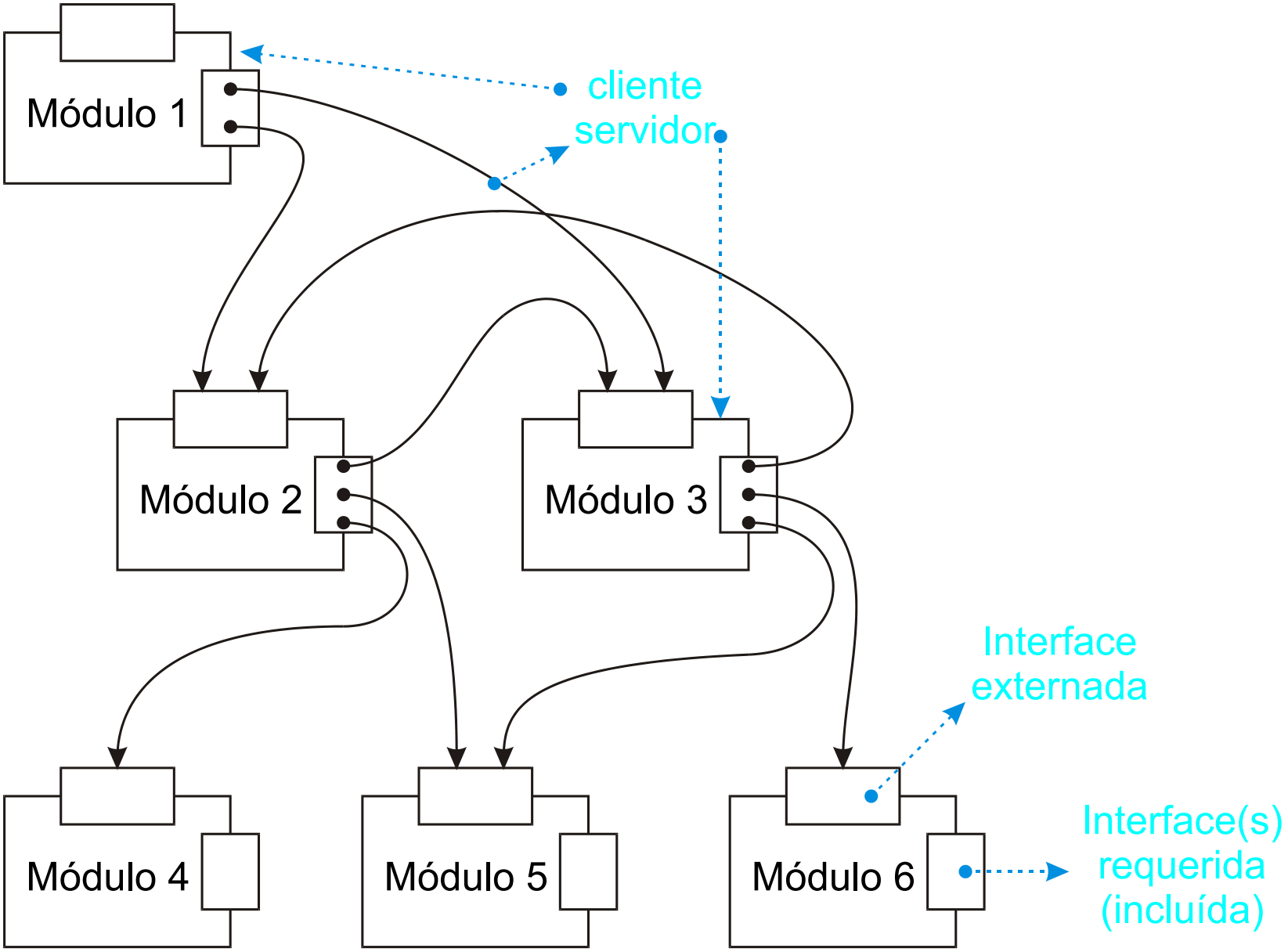


- Interface implícita
  - é um item de interface que não está visível na especificação
  - no exemplo: a base de dados ('valor' global) de onde se extrairá os dados do aluno "idAluno" está "subentendida"

Relacionamento cliente – servidor

- Cada instância de comunicação (conexão) ocorre entre
  - um cliente
    - quem origina a comunicação
    - transmissor
      - solicita um serviço
  - um servidor
    - quem recebe e processa a comunicação
    - receptor
      - presta um serviço
      - um servidor pode dar respostas ao cliente

Composição de módulos

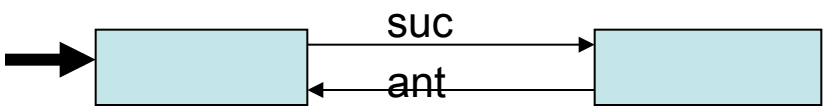


Relacionamento cliente – servidor

- Exemplos
  - O código da função **A** contém uma chamada para a função **B**
    - **A** é cliente do servidor **B**
      - mais precisamente: a chamada de **B** contida em **A** é cliente de **B**
    - a função servidora **B** responde ao cliente <chamada de **B** contida em **A** >
      - retornando um valor
      - atribuindo um valor a parâmetro passado por referência
    - quando a chamada é executada estabelece-se a conexão
      - Conexão é dinâmica: feita em tempo de execução
  - Para poder ser compilado, o módulo `C` necessita incluir (`#include`) o módulo de definição (`D.h`) do módulo `D`
    - o módulo `C` é cliente do módulo servidor `D`
    - quando o módulo `C` for compilado estabelece-se a conexão
      - Conexão é estática: feita em tempo de compilação

Interface, sintaxe

- ***Sintaxe***: é a forma de cada item da interface:
  - Tipos, regras e restrições aplicáveis aos itens intercambiados
- Elementos da sintaxe:
  - identificador (nome)
  - codificação da representação física: (computacional) dos valores, exemplos:
    - inteiro, flutuante, *string* terminado com byte zero,
      `typedef struct ARV_tagArvore *  ARV_tpArvore`
  - codificação de valores: determinam como devem ser redigidos valores de determinado tipo, exemplos:
    - número de matrícula com dígito verificador: 00000-0
    - elementos separados por hífen: 00-00-0000
    - **`NULL`** definido como **`void *`**



Interface, semântica

- ***Semântica***: é o significado de cada item da interface
- Exemplos:
  - O parâmetro `float Veloc` representa ***velocidade em m/s***
    - não basta saber que é `float`, é necessário também saber que o parâmetro é velocidade e que esta é medida em m/s
  - O *string* é um ***nome de pessoa***
    - não basta saber que é *string*, é necessário também saber que denota o nome de uma pessoa e que satisfaz determinadas características, ex.: ressalta o nome ou apelido usual ao se comunicar com a pessoa
  - Dados com significados específicos
    - ex. NULL; ID_NIL; ID_UNDEF
  - Dados satisfazendo condições definidas (contratos, assertivas)
    - ex. 0 <= Nota <= 10
    - O nó corrente é a ***raiz*** de uma estrutura de dados ***árvore n-ária***

Corretude de composição via interface

- Em programação modular, um contrato básico deve ser garantido:
  - O cliente deve assegurar a validade sintática e semântica dos dados transmitidos ao servidor
  - O servidor deve assegurar a validade sintática e semântica dos dados retornados ao cliente
  - Exceção: sempre que não confiar é necessário redigir código de verificação da corretude dos dados

Corretude de composição via interface

- Infelizmente a maior parte das linguagens não é capaz de verificar a corretude total da composição via interface
  - verificam somente a corretude sintática (parcialmente ☹)
- Soluções parciais para esse problema
  - uso de nomes suficientemente explícitos

```
NomeAluno       = NomeDisciplina ;
VelocidadeMedia = DistanciaPercorrida ;
```

    - podem rapidamente ser identificados como um erro
  - comentários

```
float VelocidadeMedia ;
   /* medida em m/s */
```

  - comentários redigidos em uma linguagem natural permitem melhorar a descrição da semântica dos itens da interface

Módulo: Definição geral x programação

- *De forma geral*: é qualquer unidade que podemos tratar de forma independente em um sistema de software
  - a especificação ou implementação interna é substituível!
  - interface bem definida
    - idealmente: deveria ser totalmente explícita
    - objetivo: facilitar compreensão, uso, manutenção do módulo

- *Para o propósito deste curso*, temos uma definição mais específica para módulo:
  - "uma unidade *lógica* de um programa com interface bem definida que pode ser *compilada de forma independente*" [Staa, 2000]
  - duas propriedades são de importância: *modularidade física* e *modularidade lógica*