Modularização

Prof. Gustavo Willam Pereira

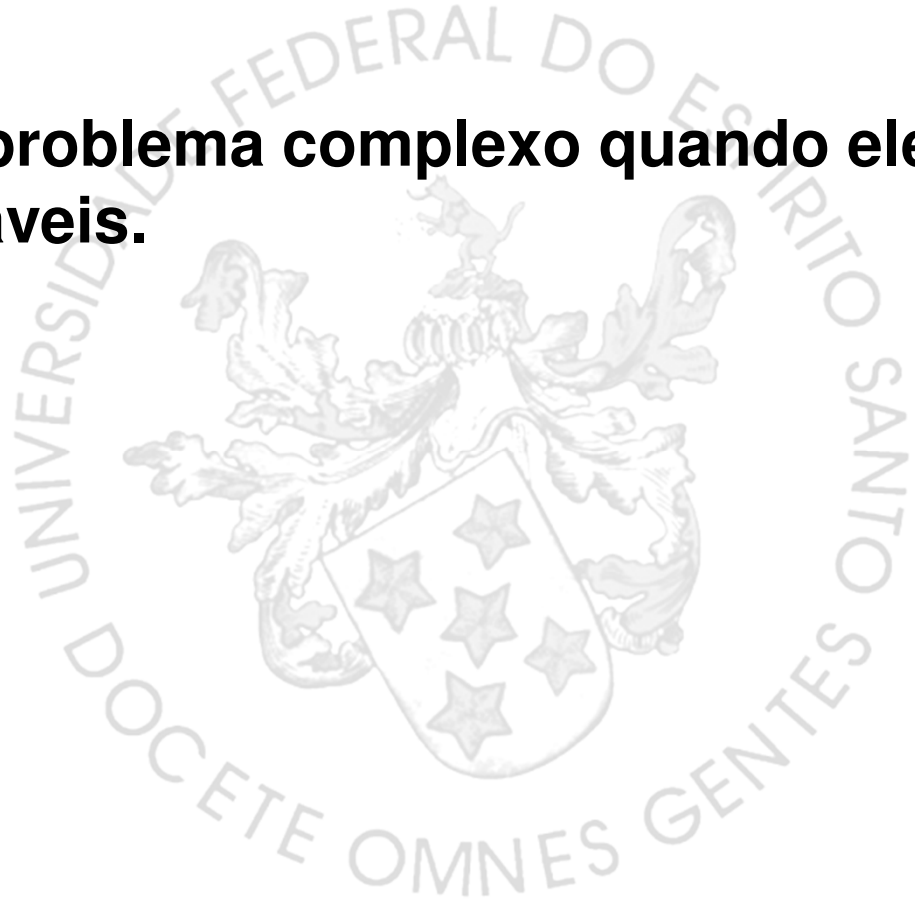
ENG10082 – Programação II

Créditos: Prof. Clayton Vieira Fraga Filho

Modularização

- O software pode ser dividido em módulos, que são integrados para satisfazer aos requisitos do sistema.

- **É mais fácil de se resolver um problema complexo quando ele é dividido em partes administráveis.**



Modularização

Módulos: conjuntos de elementos mantidos pela equipe de desenvolvimento que fornecem e usam serviços de outros elementos do software;

Podem ser compostos de outros elementos mais simples (p.e: procedimentos e funções), ou componentes (provedores de serviços);

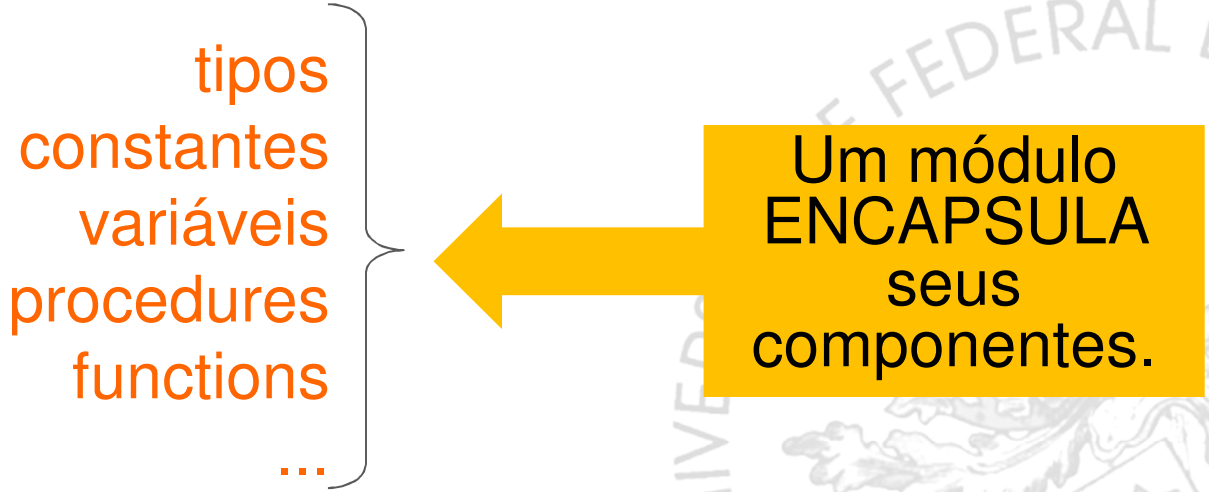
Um módulo bem projetado tem um objetivo simples e apresenta interface reduzida (pequena) para outros módulos. Isso leva o módulo a ser:

- REUSÁVEL (apto a ser incorporado em muitos programas)
- MODIFICÁVEL (pode ser alterado sem forçar grandes modificações nos outros módulos)

- Ex: Sistema de vendas: fazer vendas, efetuar recebimento, lançar no caixa ou contas a receber

Modularização

Módulo pode ser só uma *procedure* ou *function* (prog. in the small), ou pode conter diversos componentes agrupados e com um propósito em comum:



Para conseguir ter uma interface pequena, um módulo deixa poucos componentes visíveis para fora dele. Esses componentes são "exportados" pelo módulo. Os outros componentes permanecem "escondidos " (hidden) dentro do módulo, sendo usados para ajudar na implementação dos componentes exportados.

Modularização

Um módulo é uma unidade cujos elementos estruturais estão fortemente conectados entre si e (relativamente) fracamente conectados com elementos de outras unidades.

(McClelland and Rumelhart, 1995)

São estruturalmente independentes mas que funcionam juntas.

Módulo de vendas (proc. Realizar venda, forma_pag, lançar_caixa_avista, lanca_contas_receber_prazo, cartao_credito)
Este módulo deve receber informação do modulo de Gestao_produto (cod_prod, valor, estoque)
Este módulo deve enviar informação para o modulo Financeiro (total_venda, venda_avista, venda_prazo, venda_cartao)

Modularização

Granularidade dos módulos: A granularidade do artefato de software é um fator relevante que ajuda a definir o conceito de módulo.

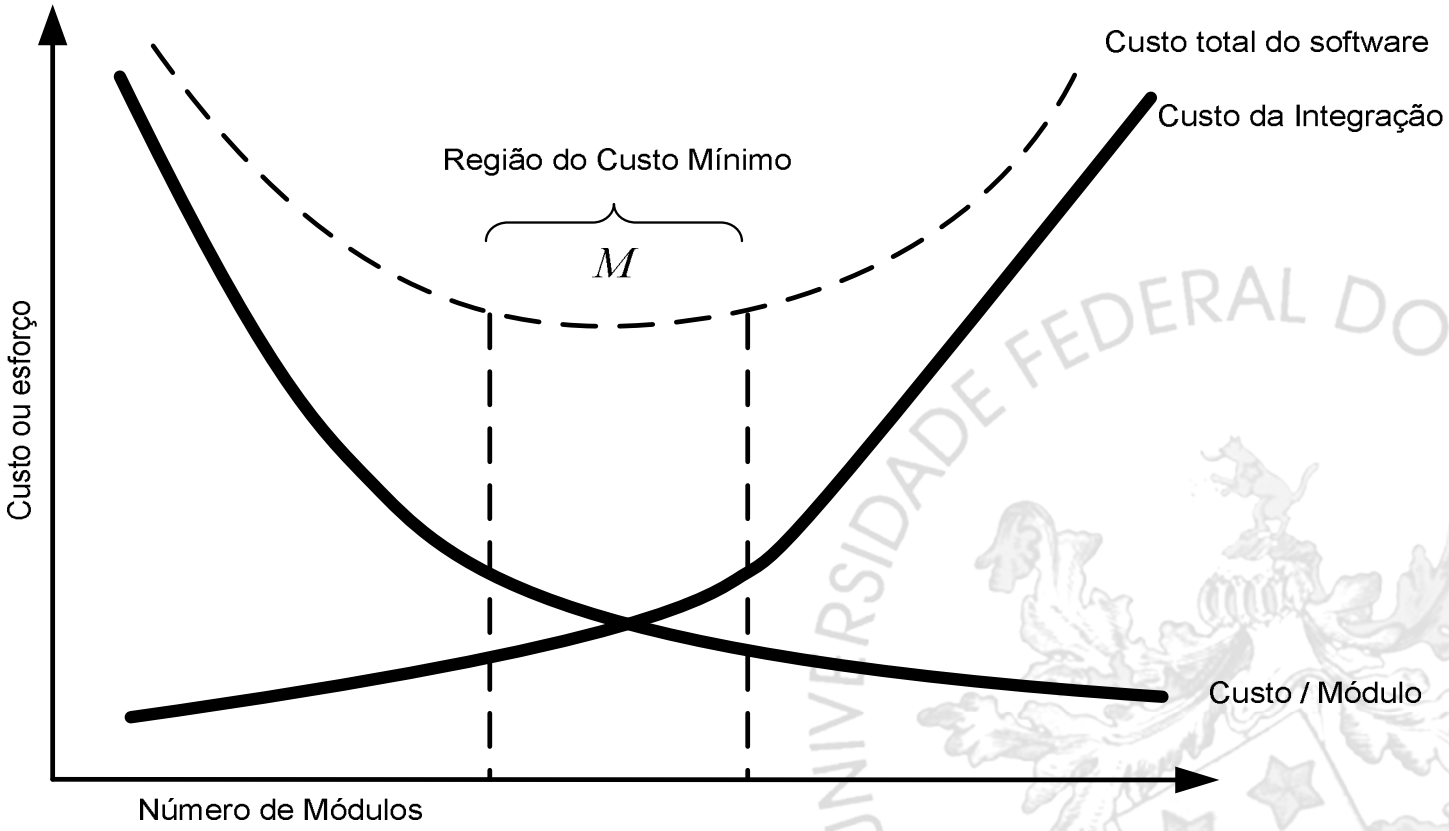
- Uma instrução, por exemplo, tem um nível de granularidade menor, que uma função ou procedimento.
- Uma biblioteca, que agrupa diversos funções e procedimentos tem uma granularidade maior.

Modularização

Um bom projeto modular **reduz** a complexidade, **facilita** a mudança e resulta numa implementação mais **fácil** ao estimular o **desenvolvimento paralelo** de diversas partes de um sistema.

- Ocultamento de informação
- Independência funcional
  - Coesão (alta)
  - Acoplamento (baixo)

Modularização



Submodularidade: módulos grandes e caros, além de difíceis de entender e modificar devido a sua estrutura monolítica;

Supermodularidade: infinidade de pequenos módulos com custo elevado devido a necessidade de integração entre módulos

Modularização

- Vantagens da Modularização
  - Reduz a complexidade
  - Possibilita o desenvolvimento em paralelo
  - Facilita a modificação
  - Possibilita a reutilização

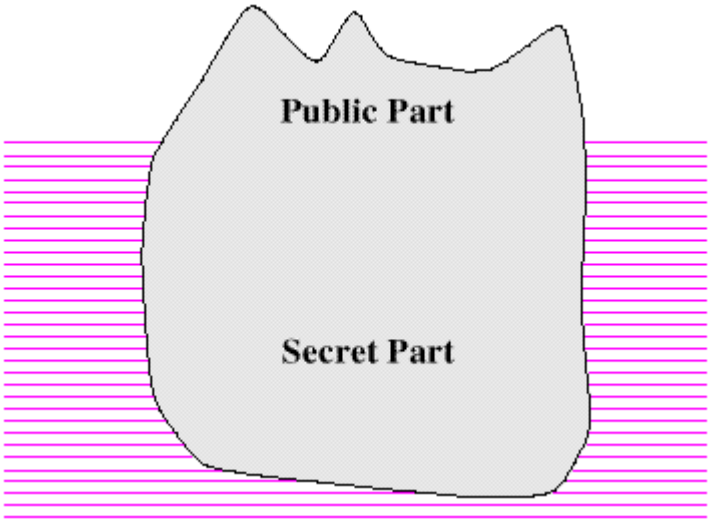
Ocultamento de informação

Módulos devem ser especificados e projetados de tal forma que as informações (procedimentos e dados) contidas num módulo sejam inacessíveis a outros módulos que não necessitem destas informações.

- Reduz a ocorrência de “efeitos colaterais”
- Limita o impacto global das decisões locais de projeto
- Enfatiza comunicação através de interfaces controladas
- Desencoraja o uso de dados globais
- Leva ao encapsulamento - um atributo de projeto de alta qualidade
- Resulta em qualidade de software

Ocultamento de informação

Um subconjunto das propriedades do módulo é escolhido como parte pública, oficial, disponível para os clientes.



Sistema Acadêmico: Modulo RH – disponibiliza parte publica (nome do professor) – parte secreta (salario, número dos dependentes)
Sistema Comercial : Modulo Gestão Produtos - disponibiliza parte publica (produto, preço_venda, estoque) – parte secreta (custo_mercadoria, fornecedores)
Sistema Contábil : Modulo Fiscal - disponibiliza parte publica (total de vendas, total de impostos) – parte secreta (calculo dos impostos).

Independência funcional

A independência funcional é conseguida desenvolvendo-se módulos com função “de um só propósito" e "aversão" a interações excessivas com outros módulos.

Decorrência direta da modularidade e ocultamento de informação

Obtida quanto o Módulo tem “finalidade única” e “aversão” a iteração excessiva com outros módulos

Um software com módulos independentes é mais fácil de ser desenvolvido e mais fácil de ser mantido.

A independência funcional de um módulo é medida usando-se dois critérios estruturais: **coesão** e **acoplamento** .

Ex: Gestão de produtos – para se fazer o calculo do custo de produtos é necessário ter acesso:
as despesas operacionais (que estão disponiveis no modulo contabil)
as despesas de impostos (que estão disponíveis no modulo fiscal)

Independência funcional

- Coesão
  - medida da unidade funcional relativa de um módulo.
  - “cola” que mantém os componentes do módulo juntos
  - Indicação qualitativa do grau em que um módulo focaliza apenas uma coisa
  - Um módulo coeso realiza uma única tarefa dentro de um procedimento de software
  - Alta coesão -> maior independência funcional
  - Coesão é a medida em que os componentes de um módulo estão relacionados, se eles têm responsabilidade em comum ou se foram apenas agrupados por acaso.

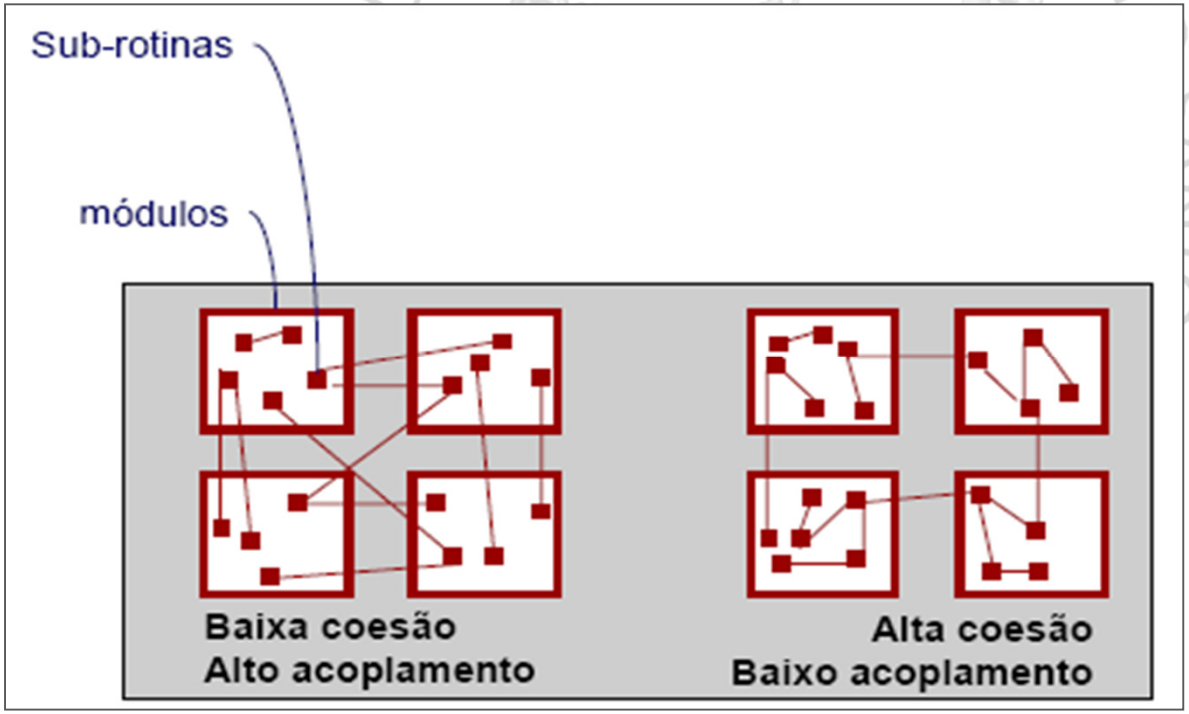
Independência funcional

- Acoplamento
  - medida da interdependência relativa entre os módulos.
  - a “força” da conexão entre módulos
  - O quão "amarrado" uma parte do sistema é às outras partes.
  - O quão dependente uma parte do sistema é das estruturas internas das outras partes do sistema.
  - É a medida de interconexão entre módulos numa estrutura de software
  - Acoplamento diz respeito a independência entre componentes, a medida de impacto que a alteração da implementação de um componente tem sobre outros.

Independência funcional

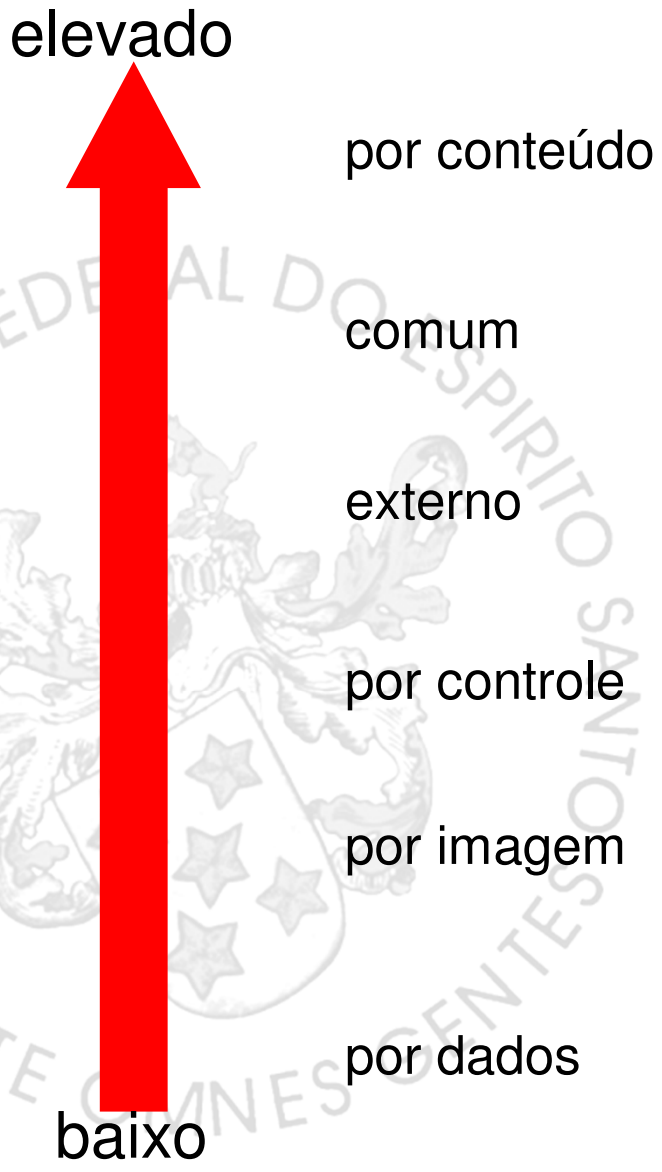
- Cada módulo deve ser **altamente coeso** (*highly cohesive*)
  - módulo é visto como "unidade"
  - componentes internos a um módulo estão relacionados

- Módulos devem apresentar **baixo acoplamento** (*low coupling*)
  - módulos possuem poucas interações com outros módulos
  - podem ser compreendidos separadamente



Acoplamento

- Quanto menor o número de conexões entre os módulos, menor a chance do efeito cadeia (propagação);
- Deseja-se trocar um módulo com um mínimo de riscos de ter de trocar outro módulo;
- Deseja-se que cada mudança do usuário afete o mínimo de módulos;
- Enquanto estiver sendo realizada a manutenção de um módulo, não deve existir a necessidade de se preocupar com a codificação interna de nenhum outro módulo.



Acoplamento por conteúdo

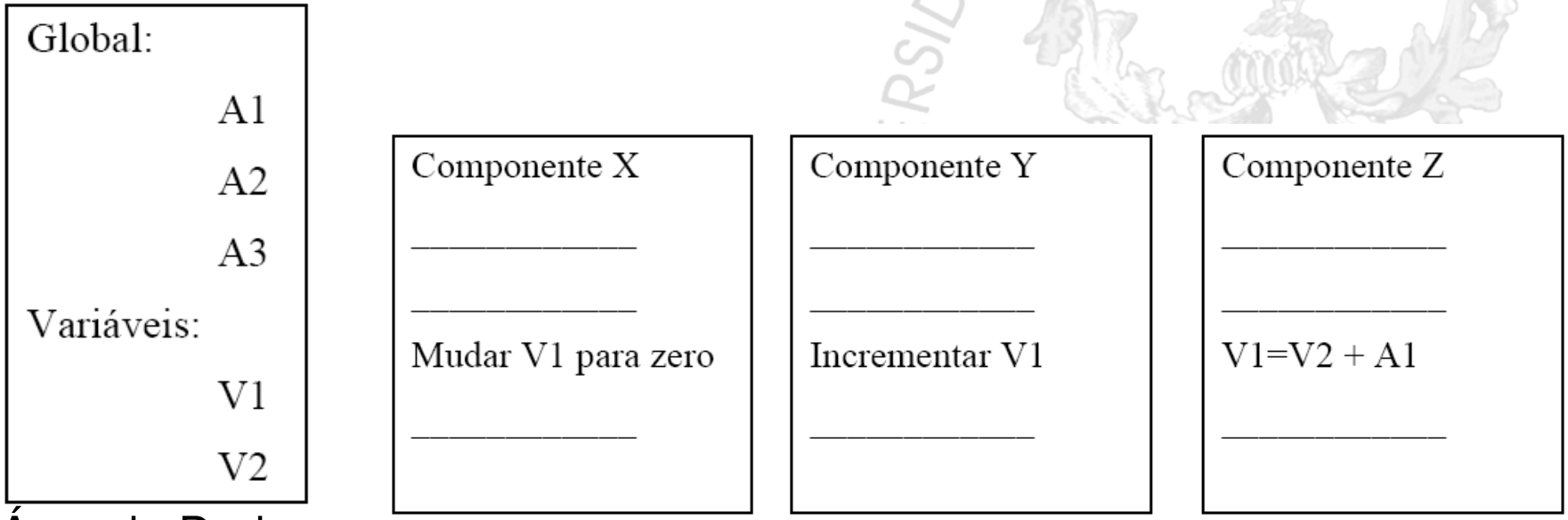
```
Módulo A
        <instruções>
        Valor = Valor * percentual
        Se Valor > 1000
            GO TO P1
         Fim-se
        <instruções>

Módulo B
        <instruções>.
        P1. Efetuar desconto
        <instruções>
```

- Dois módulos apresentam acoplamento por conteúdo se um faz referência ao interior do outro.
- Ocorre quando um módulo faz uso de dados ou controle mantida dentro dos limites de outro modulo
- Ou quanto são feitos desvios para o meio de um módulo
- Tal acoplamento quebra o conceito de caixa preta → caixa branca ou transparente
- Viola o Ocultamento de informação
- Prejudica reuso, manutenção e testes.

Acoplamento por Área Comum

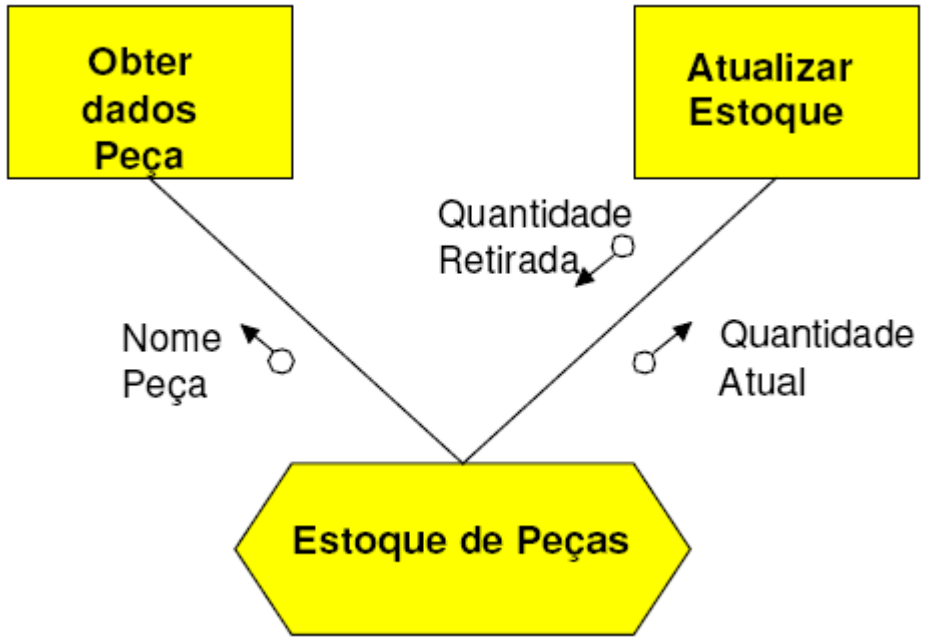
- Quando vários módulos fazem referência a uma área de dados global
- Existe quando o projeto está organizado de modo que os dados estejam acessíveis a partir de um repositório de dados comum:
- Dificuldade em determinar qual componente é responsável pela definição de uma variável com um determinado valor.



Área de Dados Comum

baixo → alto

Acoplamento por Área Comum



baixo → alto

Acoplamento externo

- Ocorre quando um módulo se comunica ou colabora com elementos de infra-estrutura como funções de sistema operacional, facilidades de banco de dados, função de telecomunicações. **Também quando há imposição de padrões externos, para comunicação entre aplicativos.**
- É necessário, mas deve ser limitado para não comprometer a adaptabilidade do software.
- Ex: Software que utiliza a API do Windows

  Software que utiliza serviços do Linux como agendamento de eventos

  Uso de Store Procedures de um Banco de Dados específico (SQL Server)

  Formato de Dados de uma empresa específica.

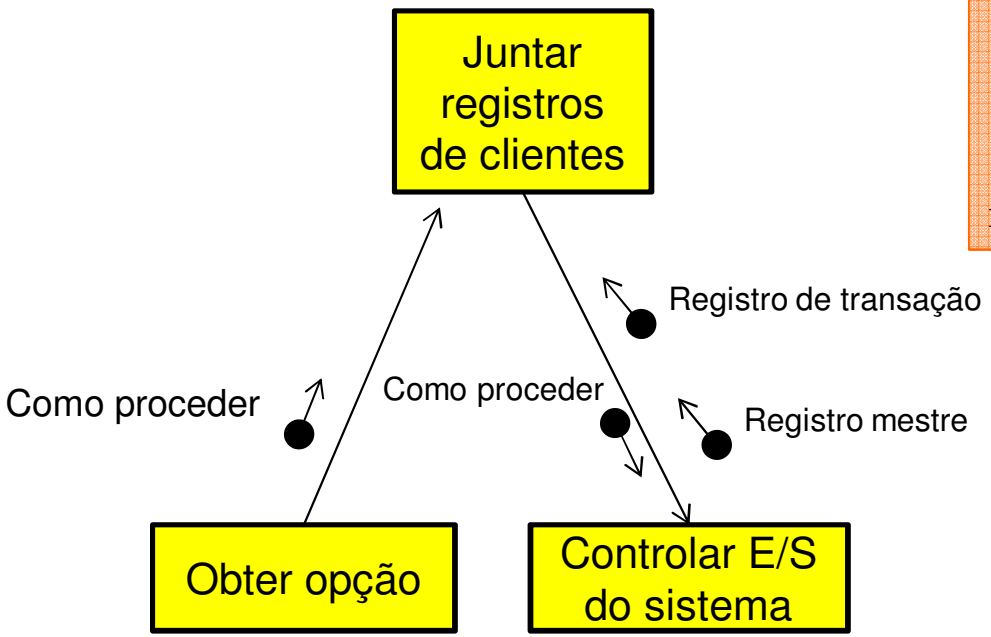
Acoplamento por controle

- Dois módulos são acoplados por controle se um passa um grupo de dados (controle) para o outro para controlar sua lógica interna.
- Em outras palavras: x e y se comunicam por parâmetros sendo que um deles é um flag que controla o comportamento de um dos módulos

- Uma variável que controla decisões em um módulo superior é passada como parâmetro para o módulo subordinado

- É impossível para o componente controlado funcionar sem a orientação do componente que o controla.

Acoplamento por controle



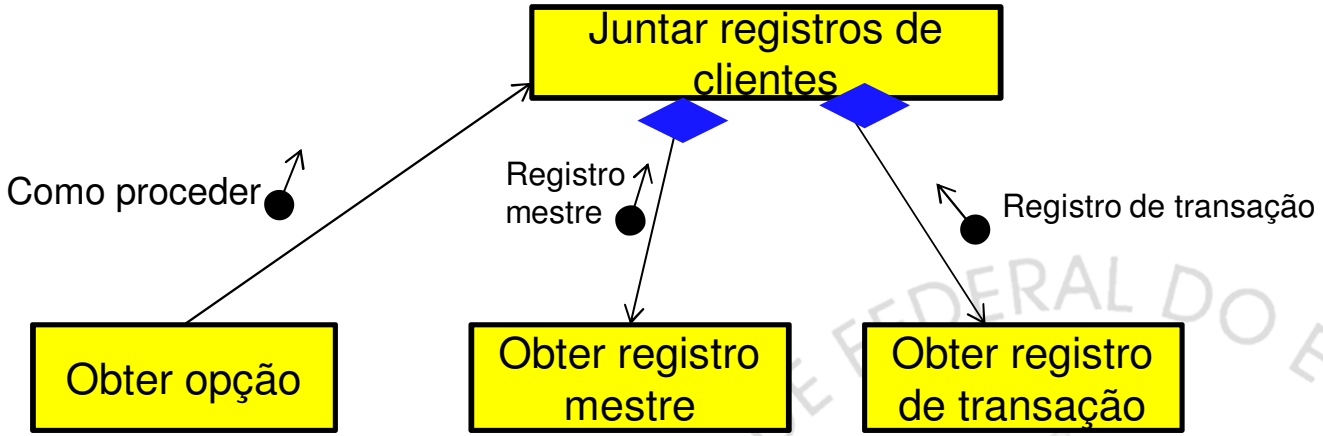

```
Módulo: Juntar registros de clientes
Início
          Obter como_proceder
          Executar Controlar E/S do sistema
                    enviando como_proceder
                    recebendo registro mestre e/ou
                              registro de transação
Fim
```

```
Valores de "Como proceder":
        1 - Obter próximo registro mestre
        2 - Obter próximo registro de transação
        3 - Obter ambos
```

Decompor a operação em múltiplas operações primitivas

baixo ——— alto

Acoplamento por controle

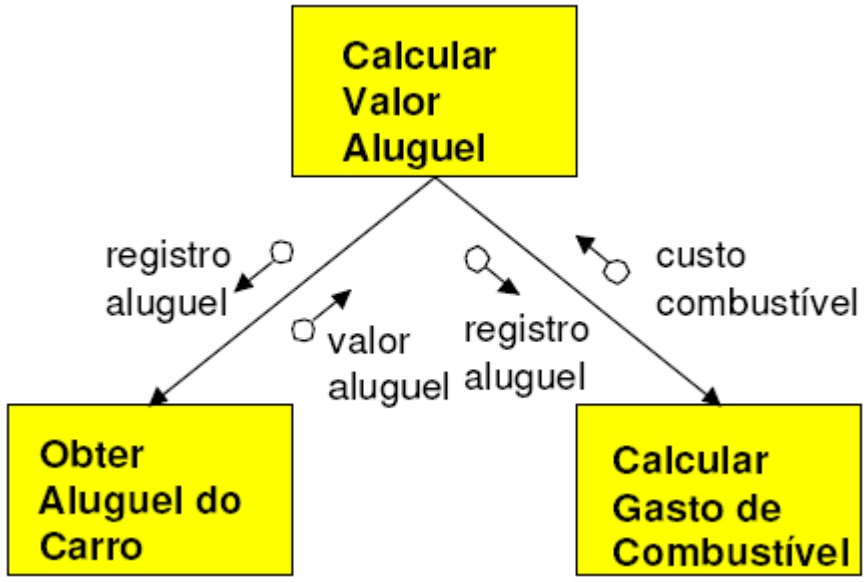



```
Módulo: Juntar registros de clientes
Início
Obter como_proceder
Caso como_proceder
          "1": Executar Obter registro mestre
                    recebendo registro mestre
          "2": Executar Obter registro de transação
                    recebendo registro de transação
          "3": begin
                    Executar Obter registro mestre
                              recebendo registro mestre
                    Executar Obter registro de transação
                              recebendo registro de transação
          fim
Fim
```

baixo ——— alto

Acoplamento por imagem

- Existem estruturas de dados mais complexas na lista de argumentos.
- Ocorre quando dois módulos fazem referência a uma mesma estrutura de dados. Este tipo de acoplamento fornece mais dados do que o necessário a um módulo.

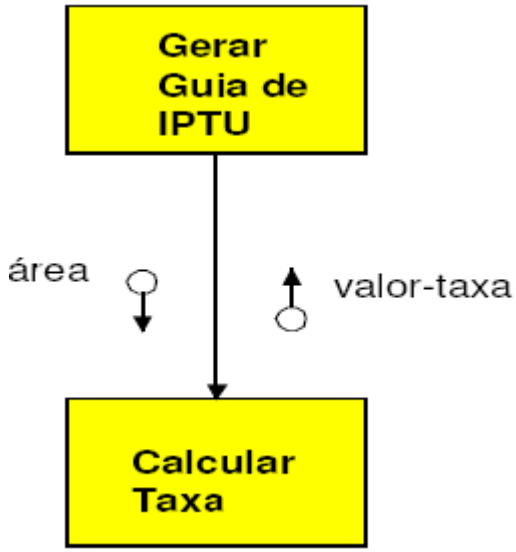



```
registro aluguel =
    combustivel
    modelo do carro
    data de locação
    km percorrida
    locatario
fim
```

baixo ——— alto

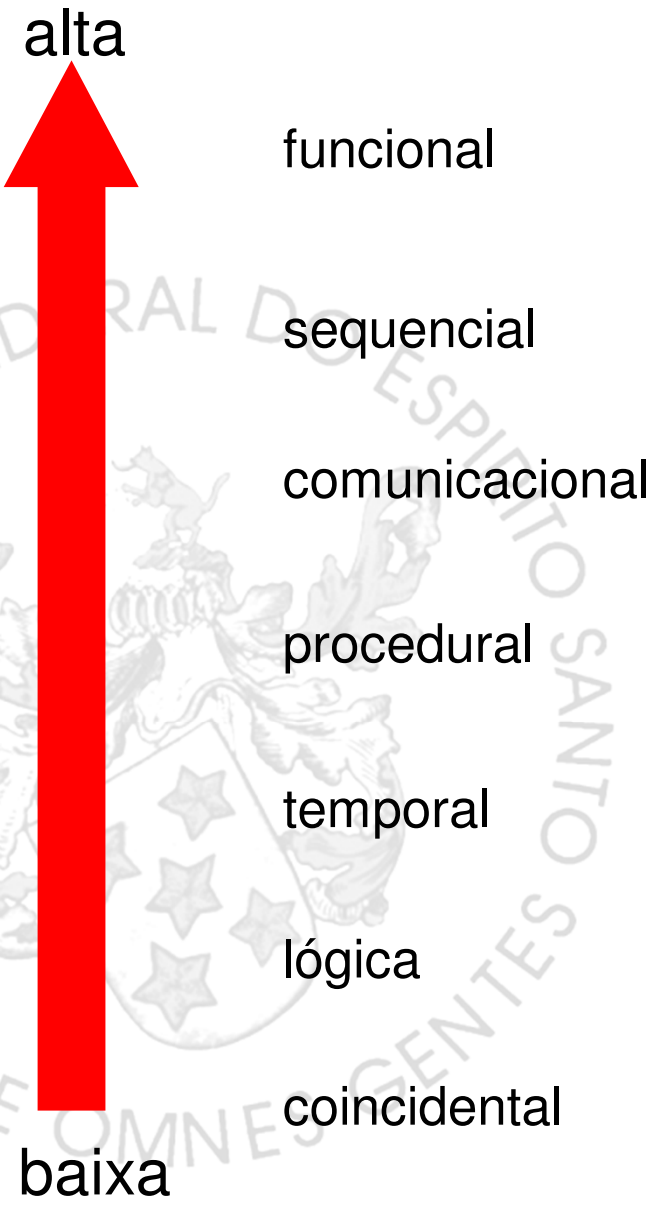
Acoplamento por dados

- Dois módulos são acoplados por dados se eles se comunicam por parâmetros
- Lista simples de argumentos – dados simples são passados, existe a correspondência de itens um a um.



Coesão

- A coesão de um módulo é o grau de relacionamento entre atividades que este realiza (métodos, responsabilidades)
- Quanto maior o grau de coesão melhor
- A coesão e o acoplamento estão inter relacionados, pois a coesão de um módulo geralmente determina o quanto ele será acoplado a outros módulos.
- Boa coesão é uma forma de minimizar acoplamento
- Um módulo coeso realiza uma única tarefa dentro de um procedimento de software
- Alta coesão → maior independência funcional



Coesão

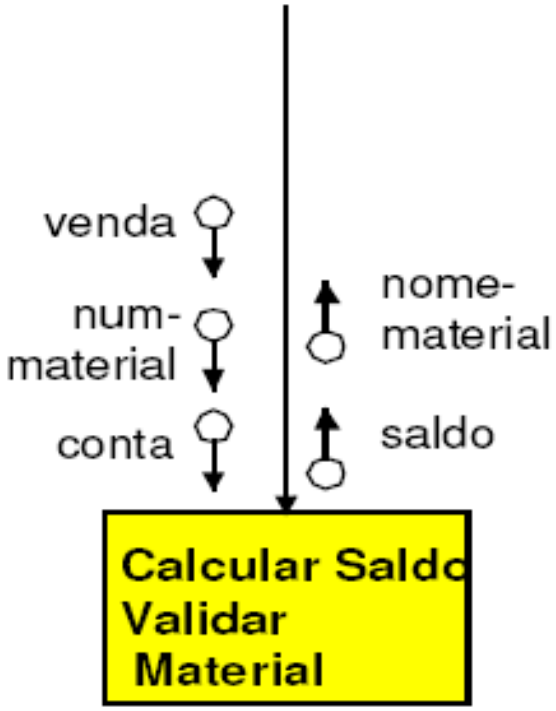
- Indicação qualitativa do grau em que um módulo focaliza apenas uma coisa.
- A escala da coesão é não linear
  - É desnecessário determinar o nível preciso de coesão
  - É importante saber identificar a existência de baixa coesão

Coesão

- Considerando componentes como:
  - Classes em um pacote,
  - Responsabilidades de um classe,
  - Linhas de código em um método
- Coesão é a medida em que os componentes de um módulo estão relacionados, se eles têm responsabilidades em comum ou se foram apenas agrupados por acaso

Coesão Coincidental

- As tarefas executadas no módulo se relacionam fracamente.
- Os elementos não têm razão aparente para estarem juntos.
- Um objeto não representa nenhum conceito do domínio e nem da arquitetura

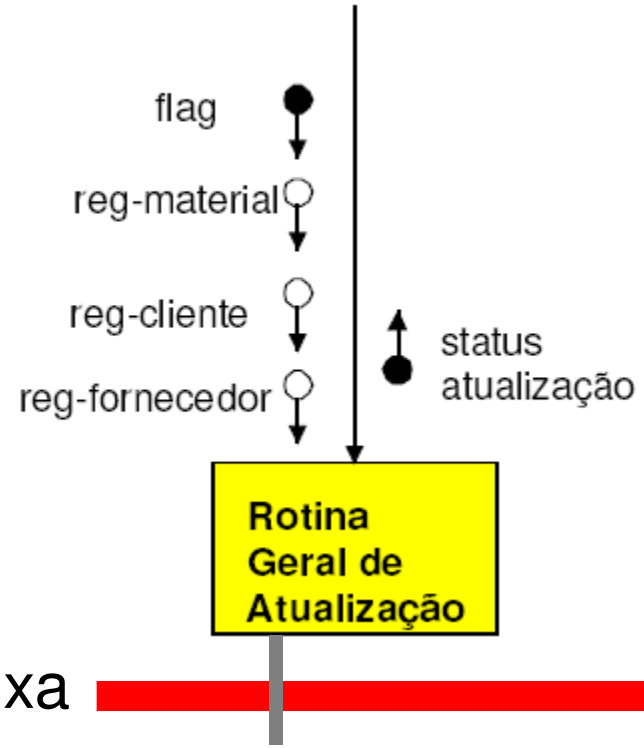


Falsa idéia de reuso e componentização;

baixa → alta

Coesão Lógica (ou de utilidade)

- As tarefas realizadas são relacionadas logicamente
- Os elementos do módulo estão envolvidos em tarefas similares.
- Elementos contribuem para atividades da mesma categoria geral, onde a atividade ou as atividades a serem executadas são selecionadas fora do módulo.



Dificulta a manutenção, testes e reuso;

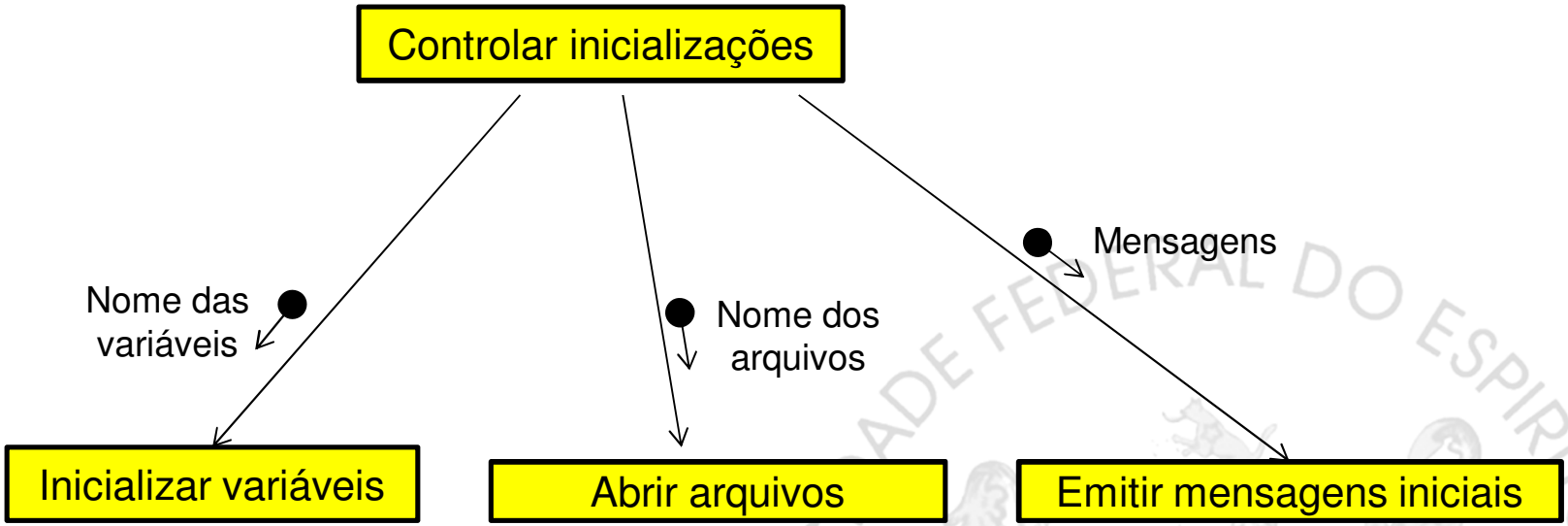
baixa ——— alta

Coesão Temporal

- Módulo cujos elementos estão envolvidos em atividades que estão relacionadas no tempo. Ou seja, estão agrupados no mesmo módulo porque são processados no mesmo intervalo de tempo.
- Tem alto acoplamento com vários outros módulos

- Exemplos comuns:
  - Função de inicialização (main) que provê valores defaults para uma série de funções diferentes
  - Função ou bloco de código de finalização que limpa variáveis internas antes de terminar execução

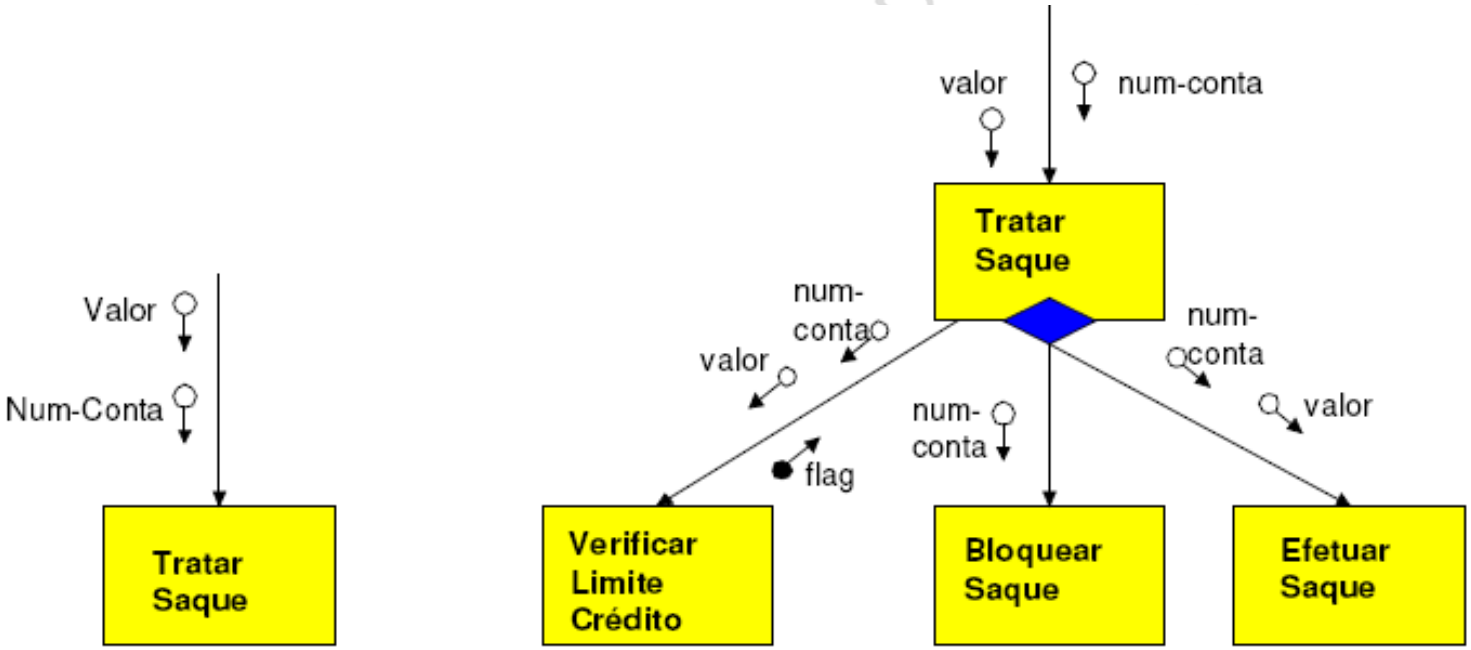
Coesão Temporal



- Difícil de modificar, pois será necessário olhar vários módulos do software (que dependem destas informações) quando houver alguma mudança: utiliza dados de vários módulos.
- Também dificulta o reuso: como reutilizar todo o método **InicializarVariaveis** ?

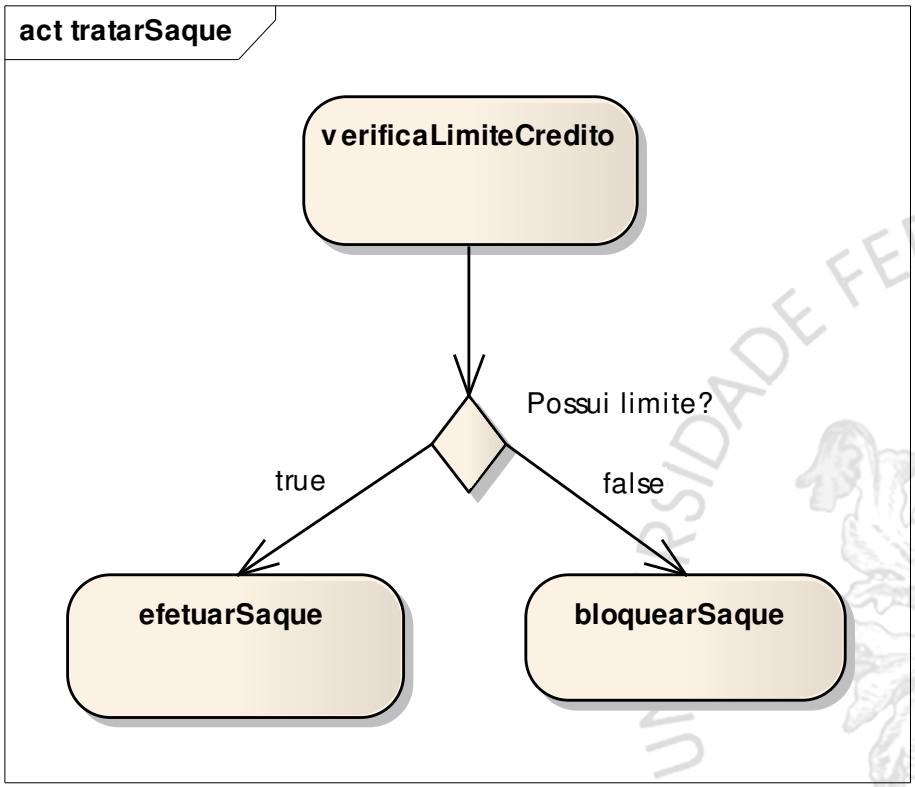
Coesão Procedural (Procedimental)

- Os elementos do módulo encontram-se juntos em um procedimento adotado para solucionar o problema (em uma mesma unidade do algoritmo).
- Os elementos de processamento de um módulo estão relacionados devem ser executados numa ordem específica



baixa ——— alta

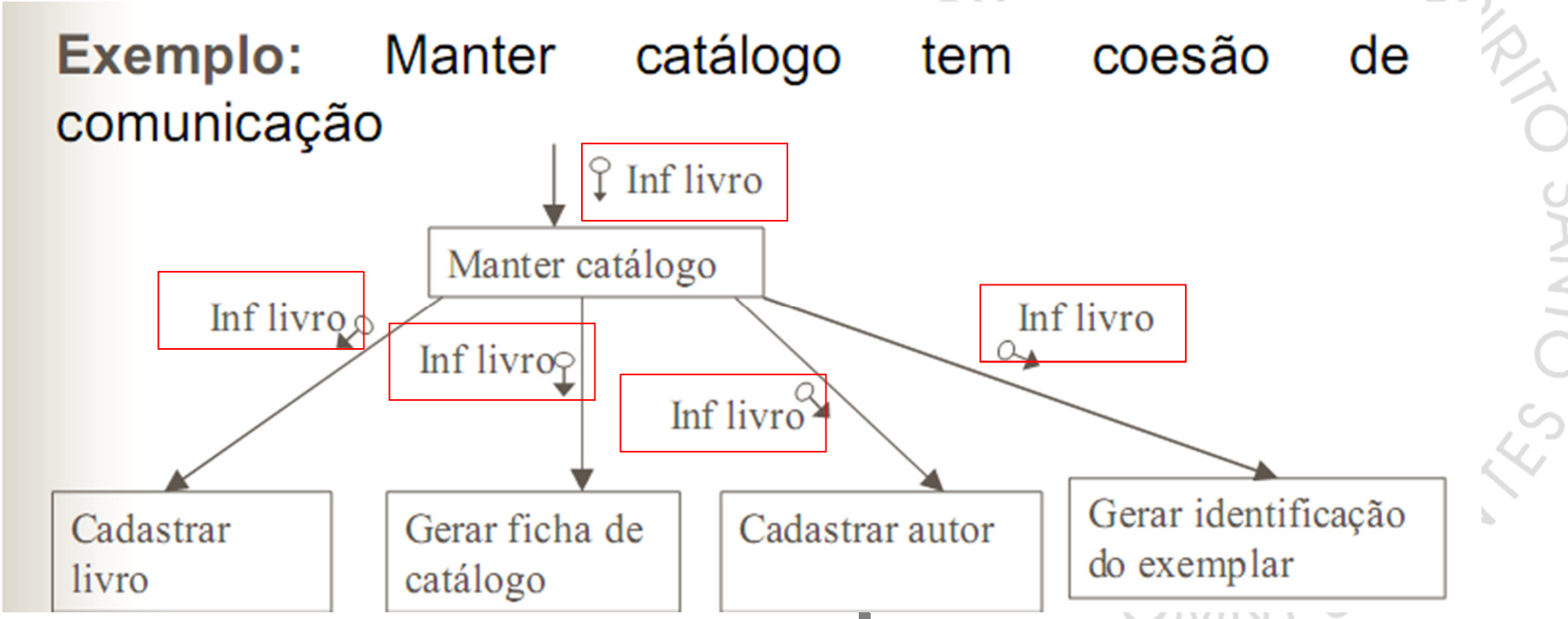
Coesão Procedural



Coesão Comunicacional

- Um módulo tem coesão de comunicação quando suas atividades estão relacionadas pelo uso da mesma entrada ou da mesma saída
- Os elementos de processamento utilizam-se da mesma área de estrutura de dados.

Exemplo: Manter catálogo tem coesão de comunicação

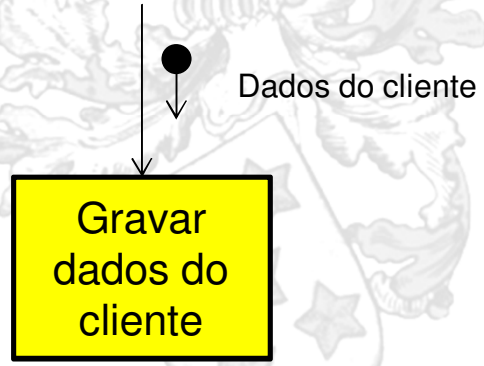


baixa ——— alta

Coesão seqüêncial

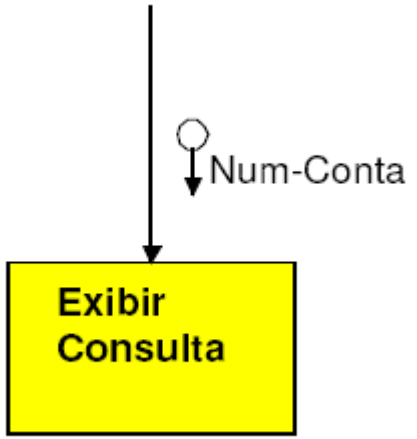
- As funções internas estão envolvidas em atividades de tal forma, que os dados de saída de uma atividade sirvam como dados de entrada para a próxima.
- Este fluxo estabelece uma seqüência de execução das funções.

```
Módulo: Gravar dados do cliente
Início
    Abrir arquivo
    Gravar dados
    Fechar arquivo
Fim
```



Coesão seqüêncial

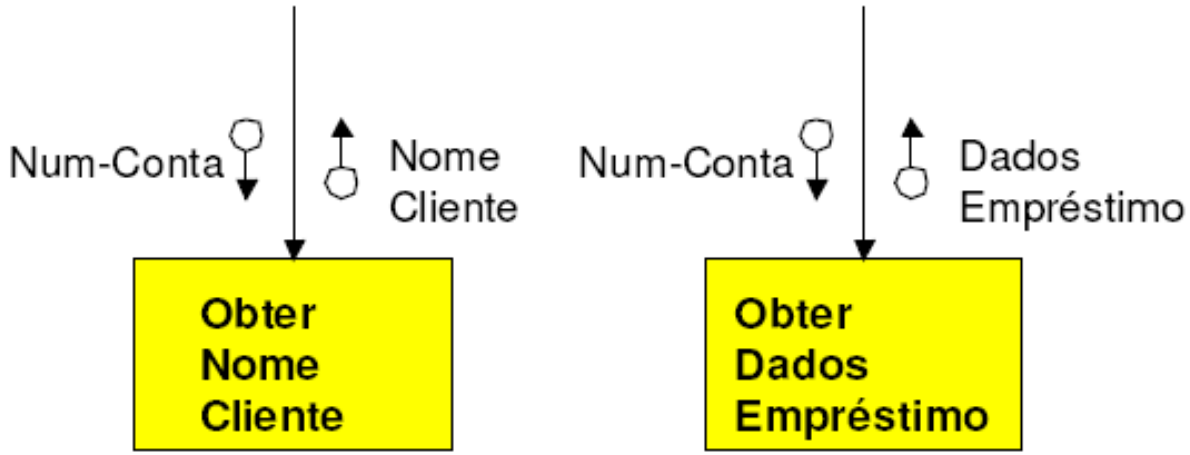
```
Módulo: Exibir consulta
Início
    Abrir arquivo da conta
    Consultar número da conta
    Exibir dados da conta
Fim
```



baixa → alta

Coesão funcional

- Um módulo com coesão funcional contém elementos que contribuem para execução de uma [e apenas uma] tarefa relacionada ao problema.
- Contém todos os elementos e apenas aqueles necessários para realizar uma única tarefa bem definida.



baixa → alta