

Dicionário de Dados (DD)

O dicionário de dados consiste numa lista organizada de todos os elementos de dados que são pertinentes para o sistema. Sem o dicionário de dados o modelo não pode ser considerado completo, pois este descreve entradas, saídas, composição de depósitos de dados e alguns cálculos intermédios. O DD consiste num ponto de referência de todos os elementos envolvidos na medida em que permite associar um significado a cada termo utilizado.

Dados elementares

Os dados elementares correspondem a elementos atómicos, ou seja, elementos sem decomposição no contexto do utilizador.

Exemplo: Apesar de se utilizar (na página seguinte) o N_telefone, como um exemplo de descrição de um elemento de dados composto, na maior parte dos contextos este dado é considerado elementar.

O DD permite inventariar e descrever os seguintes itens:

- depósitos de dados;
- fluxos de dados;
- dados elementares que constituem fluxos e depósitos de dados;

Cada entrada no DD é constituída por um identificador e respectiva descrição. A descrição de cada entrada inclui:

- o seu significado;
- o seu conteúdo (só para dados compostos);
- os valores permitidos e unidades (só para dados elementares);
- A chave primária (só para depósitos de dados).

Notação utilizada no DD

Para descrever de uma forma precisa e concisa cada componente de dados utiliza-se um conjunto de símbolos simples.

Símbolo	Significado
=	é constituído por ou é definido por
+	e (conjunção ou concatenação)
()	enquadram componentes opcionais
[]	enquadram componentes que são utilizadas alternativamente
	separam componentes alternativas enquadradas por []
{ }	enquadram componentes que se repetem 0 ou mais vezes
**	enquadram comentários
@	identifica a chave primária de um depósito

Exemplos:

- Descrição de um elemento de dados composto:

$N_telefone = (\text{indicativo_internacional} + \text{indicativo_país}) + (\text{indicativo_zona}) + N^\circ_assinante$

$\text{indicativo_internacional} = \{ \text{dígito} \}$

$\text{indicativo_país} = \{ \text{dígito} \}$

$\text{indicativo_zona} = \{ \text{dígito} \}$

$N^\circ_assinante = \{ \text{dígito} \}$

$\text{Dígito} = [0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9]$

- Descrição de elementos de dados elementares:

$\text{Sexo} = * \text{Valores: } [M | F] *$

$\text{Peso} = * \text{Peso do paciente quando é admitido no hospital} *$

$* \text{Unidades: Kg Intervalo: 1-150} *$

Exemplo de Dicionário de Dados

O exemplo apresentado corresponde a uma parte do DD do sistema de Gestão de Bibliotecas e inclui a descrição dos seguintes itens:

- o fluxo de dados Ficha_leitor;
- o depósito de dados Leitor;
- alguns dos dados elementares dos itens anteriores.

...		
BI	=	*Número do Bilhete de identidade do leitor*
Data_admissão	=	*Data de inscrição do leitor*
Ficha_leitor	=	*Dados pessoais do leitor fornecidos para a sua inscrição ou alteração de informação*
		(N_leitor) + Nome + Morada + BI + Telefone + Profissão
Leitor	=	{Leitor_i}
Leitor_i	=	*Informação mantida sobre cada leitor da biblioteca*
		@N_leitor + Nome + Morada + BI + Telefone + Profissão + Data_admissão
Morada	=	*Morada do leitor*
N_leitor	=	*Número de identificação de leitor da Biblioteca*
		{dígito}
...		...

Descrição de elementos de dados elementares

A descrição do significado de dados elementares é desnecessária quando o elemento de dados é óbvio (Exemplo: Sexo). Contudo, é necessário criar uma entrada no DD e especificar as unidades e valores, ou intervalo de valores, para cada elemento de dados.

Especificação de processos

A especificação de processos consiste na descrição interna da tarefa de cada processo primitivo dos níveis inferiores de um DFD.

A descrição de um processo primitivo também é designada por **mini-especificação**.

A especificação de processos só é elaborada para os processos primitivos e não deve ultrapassar a dimensão de uma página. Se a mini-especificação for complexa é necessário decompor o processo num DFD de nível inferior.

Objectivos da mini-especificação:

- Descrever as regras de transformação dos fluxos de entrada em fluxos de saída, traduzindo a política de transformação e não um método de implementar essa política;
- Descrever actividades sem impor decisões arbitrárias de desenho ou implementação;
- Rever e corrigir o DFD elaborado, pois a especificação de processos permite detectar necessidades de fluxos e actividades adicionais.

Técnicas para a escrita de mini-especificações

- Linguagem estruturada;
- Pré e Pós-condições;
- Tabelas e árvores de decisão;
- Fluxogramas;
- Diagramas de Nassi-Shneiderman;
- Qualquer combinação das técnicas anteriores.

Linguagem Estruturada

A Linguagem estruturada, LE, é uma linguagem de especificação retirada da linguagem corrente à qual foram feitas algumas limitações na sintaxe e no vocabulário. A LE é a técnica de especificação mais utilizada.

A LE é utilizada para escrever vários tipos de instruções:

- **Instruções de atribuição/leitura/escrita**

$X = (Y * Z) / \text{SOMA}(Q, 14)$

ESCREVER (para processo 1.3.2) X

LER (de terminador1) R

$K = (X * R) \wedge \text{SOMA}(Y, 20)$

ESCREVER K

Notas: a) Não é necessário ler dados provenientes de fluxos de dados;

b) Não é necessário especificar terminadores de destino dos fluxos.

- **Instruções de selecção**

SE - FIM SE

SE - SENÃO - FIM SE

SE - SENÃO SE - (SENÃO) - FIM SE

CASO - (CASO SENÃO) - FIM CASO

- **Instruções de repetição**

PARA - ATÉ - FIM PARA

PARA CADA - FIM PARA

ENQUANTO - FIM ENQUANTO

REPETIR - ATÉ

- **Instruções de manipulação de depósitos de dados**

ENCONTRAR lista_atributos EM depósito_dados COM condição_pesquisa

LER registo nome_registo

INSERIR nome_registo EM depósito_dados

ALTERAR lista_atributos EM depósito_dados PARA lista_novos_valores

REMOVER registo EM depósito_dados COM condição_pesquisa

Assim, o vocabulário da LE resume-se a:

- Conjunto limitado de verbos, apresentados no infinito, que permitem especificar qual a acção que é desempenhada;
- Termos definidos no DD correspondentes a objectos do sistema;
- Termos locais, ou seja, palavras definidas, conhecidas, relevantes e com significado somente na especificação de um dado processo. Um exemplo típico de termo local é um cálculo intermédio usado para produzir uma saída final;
- Palavras reservadas para a representação das várias estruturas de controlo;
- Algumas palavras de ligação que permitem dar maior legibilidade à linguagem (EM, DE, PARA);
- Operadores relacionais: = ≠ > < ≥ ≤ e ou
- Operador de concatenação: +
- Operadores aritméticos: soma - / * ^

Exemplos de especificação de processos

Principais entradas do DD relacionadas:

Ficha_leitor	=	*Dados pessoais do leitor fornecidos para a sua inscrição ou alteração de informação*
		(N_leitor) + Nome + Morada + BI + Telefone + Profissão
Leitor	=	{Leitor_i}
Leitor_i	=	*Informação mantida sobre cada leitor da biblioteca*
		@N_leitor + Nome + Morada + BI + Telefone + Profissão + Data_admissão

Processo 1.1 - Registar dados de leitor

ENCONTRAR leitor_i EM **Leitor** COM bi = bi DE **Ficha_leitor**

SE existe registo

LER registo Leitor_i

ALTERAR nome, morada, telefone, profissão EM **Leitor** PARA (nome, morada, telefone, profissão) DE **Ficha_leitor**

SENÃO

N_leitor = próximo N_leitor disponível

Data_admissão = Data_actual

Leitor_i = N_leitor + nome + morada + bi + telefone + profissão + data_admissão

INSERIR Leitor_i EM **Leitor**

FIM SE

Conteúdo das principais entradas no DD relacionadas:

Dados_livro	=	N_livro + Título + Ano_edição + Editora + Cota
Lista_título	=	{Dados_livro + {nome_autor}}
Pedido_pesquisa_título	=	título
Resposta_pesquisa_título	=	["Não existe" lista_título]

Processo 2.1 - Pesquisar livros por título

ENCONTRAR livro_i EM **Livro** COM título \subset título DE
Pedido_Pesquisa_título

SE não existem registos

Resposta_pesquisa_título = "Não existe"

SENÃO

ENQUANTO não fim de registos */ de livro /*

LER registo Livro_i

dados_livro = N_livro + título + Ano_edição + Editora + Cota

lista_título = lista_título + dados_livro

ENCONTRAR autor_livro_i EM **Autor_Livro** COM N_livro

ENQUANTO não fim de registos */ de autor_livro /*

LER N_autor de **Autor_livro**

ENCONTRAR autor_i EM **Autor** COM N_autor

LER Nome_autor de **Autor**

lista_título = lista_título + Nome_autor

FIM ENQUANTO

FIM ENQUANTO

Resposta_pesquisa_título = lista_título

FIM SE

ESCREVER **Resposta_pesquisa_título**

Pré / Pós Condições

Constituem uma forma conveniente de descrever a função que tem de ser executada por um processo, sem especificar o respectivo algoritmo.

A utilização de pré / pós condições é apropriada quando:

- o utilizador tem tendência para expressar os procedimentos que utiliza para sustentar uma política e não a política subjacente que tem de ser levada a cabo;
- o analista reconhece que existem vários algoritmos diferentes, que podem ser usados, e pretende adiar a opção por um deles para uma fase posterior;
- o analista quer deixar para o programador a tarefa de exploração de vários algoritmos e não se quer envolver nestes detalhes.

A utilização de pré / pós condições não é apropriada quando:

- o número de passos intermédios para produzir as saídas é elevado, tornando a especificação produzida difícil de interpretar pelo facto de não se visualizar todos os procedimentos executados;
- os relacionamentos entre entradas e saídas a produzir são complexos, sendo mais fácil escrever a especificação através da utilização da “Linguagem estruturada”.

A especificação de processos através desta técnica é constituída pela descrição de dois tipos de condições:

- Pré condições
- Pós condições

Pré condições

As pré condições especificam tudo o que se deve verificar antes da actividade do processo se iniciar. Tipicamente descrevem:

- **Entradas que devem estar disponíveis**

Elementos de dados que constituem um estímulo activador do processo e que correspondem a fluxos de entrada. Contudo, existem casos em que, apesar de existirem vários fluxos de entrada, só um dos fluxos é que é uma pré condição necessária para activar o processo, correspondendo os restantes fluxos de entrada a pedidos de informação.

- **Relacionamentos que devem existir entre os dados das entradas**

Muitas vezes uma pré condição especifica que ocorrem duas entradas com atributos correspondentes (Exemplo: “Ocorrem detalhes_de_encomenda e detalhes_de_envio com o mesmo número_pedido”).

Uma pré condição também pode especificar o intervalo de valores de um elemento de dados de entrada (Exemplo: “Um pedido ocorre com uma data_entrega de 60 dias”)

- **Relacionamentos que têm de existir entre entradas e depós. de dados**

Uma pré condição pode estipular que existem correspondências entre instâncias de um depósito e elementos de uma entrada (Exemplo: “Existe um pedido com #cliente que corresponde a um #cliente do depós. cliente”).

- **Relacionamentos que têm de existir entre elementos de dados de depósito(s)**

Exemplo: “Existe um Pedido_encomenda no depósito Encomenda com um #cliente correspondente a um #cliente do depósito Cliente”

Exemplo: “Existe um Pedido_encomenda no depósito Encomenda com data_entrega igual a data_corrente”

Pós condições

Similarmente, as Pós condições especificam tudo o que se deve verificar quando o processo terminar a sua actividade. Ou, seja, especificam o que é feito e não como é feito.

As pós condições descrevem:

- **Saídas a produzir pelo processo**

Exemplo: “É produzida uma factura”

- **Relacionamentos entre valores das saídas e valores originais das entradas**

Utilizada quando uma saída corresponde ao resultado da aplicação de uma função ao valor de uma entrada (Exemplo: “O valor_total_da_factura é calculado através de $\text{Somatório}(\text{quantidade} * \text{preço_unitário}) + \text{despesas_envio}$ ”)

- **Relacionamentos entre valores de saída e valores de um ou mais depósitos**

Utilizada quando a informação, determinada a partir de depósitos de dados, pertence a uma parte do processamento da saída (Exemplo: “As existências do depósito Stock são incrementadas por unidades_recebidas e o novo valor é produzido como saída do processo”)

- **Alterações efectuadas em depósitos de dados**

Engloba itens adicionados, itens alterados ou itens removidos (Exemplo: “O Pedido_encomenda é registado no depósito encomenda”)

Etapas de elaboração de pré / Pós condições:

- Descrever situação(ções) de processamento normal;

Podem existir várias situações de processamento normal, que correspondem a diferentes combinações de diferentes valores de entrada válidos. Nesse caso, elabora-se uma pré condição para cada situação possível;

- Para cada pré condição, descrever o estado do processo quando as saídas forem produzidas e os depósitos forem alterados;
- Descrever pré / Pós condições para casos excepcionais ou situações de erro.

Exemplo: Processo X - Determinar comissão de venda

Pré condição 1

Dados_venda ocorre com *Tipo_item* que corresponde a uma *Categoria* em *Categoria_comissões*

Pós condição 1

$Comissão_venda = Valor_venda_item * Percent_comissão$

Pré condição 2

Dados_venda ocorre com *Tipo_item* que não corresponde a nenhuma *Categoria* em *Categoria_comissões*

Pós condição 2

Gerar mensagem de erro

Tabelas e Árvores de decisão

Tabelas e árvores de decisão são duas técnicas de representação da política de selecção condicional, usada para derivar um conjunto de acções, respectivamente sob a forma:

- tabular;
- diagramática.

Estas duas técnicas também são referidas como **ferramentas de modelação não procedimentais**, pois:

- especificam a política de transformação das entradas em saídas;
- não especificam o algoritmo subjacente da transformação.

A utilização destas técnicas é apropriada quando:

- um processo tem de produzir uma saída ou executar várias acções com base em decisões complexas e
- as decisões dependem de múltiplas variáveis independentes, que podem assumir vários valores diferentes.



Nestes casos, a especificação do processo, utilizando Linguagem Estruturada ou Pré / Pós Condições, torna-se muito complexa.

Etapas de elaboração de uma tabela de decisão:

- listam-se todas as variáveis relevantes, também denominadas por **condições** ou entradas, e todas as **acções** relevantes no lado esquerdo da tabela;
- lista-se cada combinação viável de valores das variáveis numa coluna;
- marca-se com X o conjunto de acções a executar para cada combinação de valores das condições.

Cada coluna da tabela é denominada por **regra**. Uma regra descreve as acções a executar para uma dada combinação de valores das variáveis. Pelo menos uma acção tem de ser especificada para cada coluna da tabela de decisão.

Estrutura de uma tabela de decisão

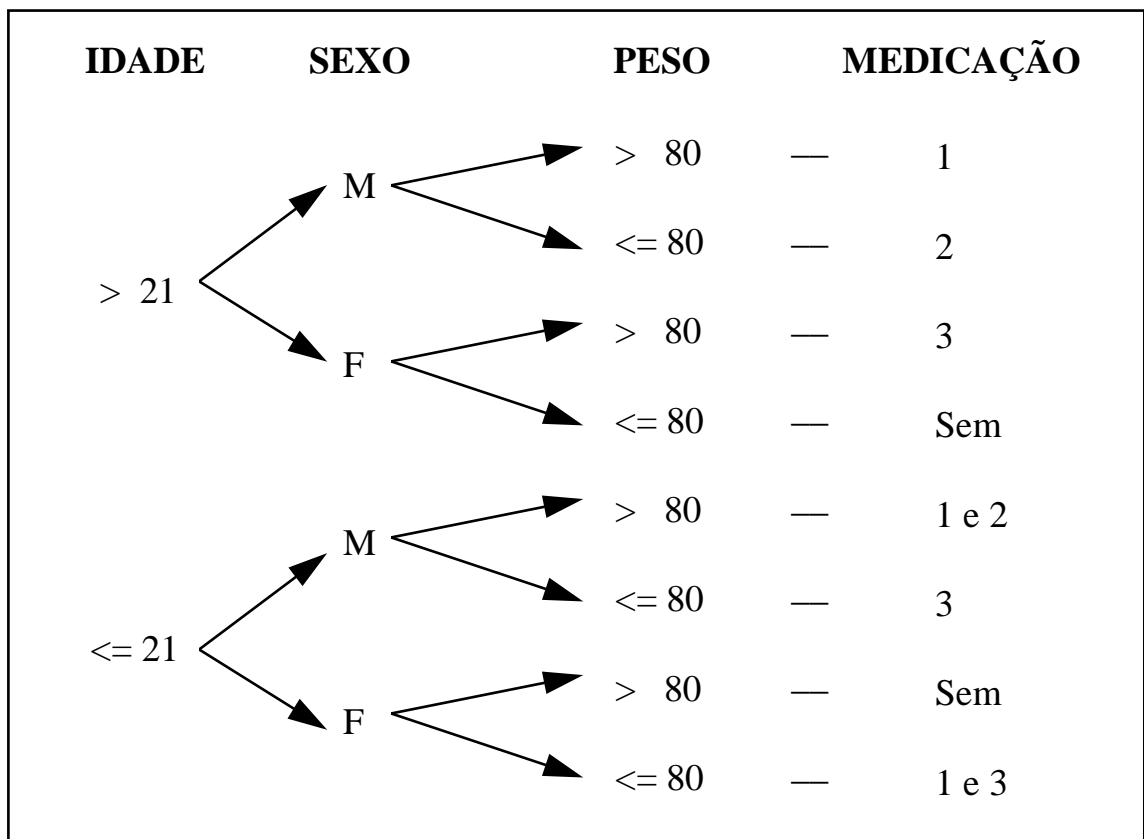
	REGRAS							
CONDIÇÕES								
ACÇÕES								

Exemplo de tabela de decisão: Determinar medicação adequada

	1	2	3	4	5	6	7	8
Idade > 20	S	S	S	S	N	N	N	N
Sexo	M	M	F	F	M	M	F	F
Peso > 80	S	N	S	N	S	N	S	N
Medicação 1	X				X			X
Medicação 2		X			X			
Medicação 3			X			X		X
Sem medicação				X			X	

As árvores de decisão correspondem a uma variante das tabelas de decisão que representam a política de selecção condicional, através da organização hierárquica de séries de decisões.

Exemplo de árvore de decisão: Determinar medicação adequada



Comparando as duas técnicas, verifica-se que uma árvore de decisão permite uma representação mais óbvia, mas menos concisa.

Fluxogramas

Os fluxogramas foram usados no passado como um mecanismo de representação gráfica de algoritmos e do seu fluxo de controlo.

Contudo, os fluxogramas perderam interesse devido a problemas relacionados com a sua utilização em duas áreas:

- **Como ferramenta de modelação de alto nível:**

- * Os fluxogramas representam a lógica procedimental de uma forma sequencial, que não é adequada para modelar uma rede de processos comunicantes assíncronos. Assim sendo, os DFD constituem uma ferramenta de modelação de alto nível mais apropriada.
- * Os fluxogramas praticamente só focam aspectos de fluxo de controlo e, em contraste com os DFD's, dão pouca informação sobre o fluxo de dados e estruturas de dados.

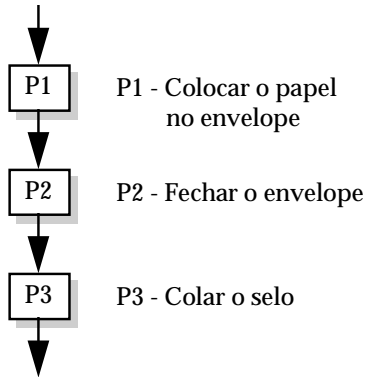
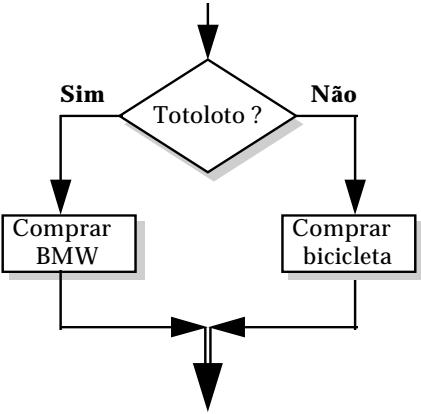
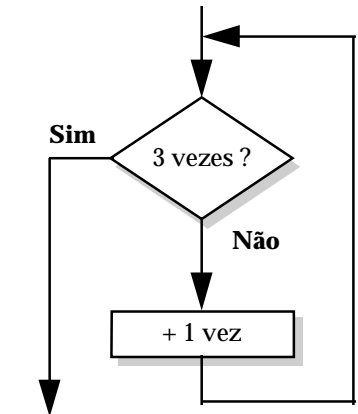
- **Como ferramenta de modelação de processos:**

- * Nada impede o analista de sistemas de construir um fluxograma complexo e não estruturado, pois estes incorporam símbolos que viabilizam a especificação das actividades de uma forma não estruturada;
- * Os fluxogramas documentam um processo a um nível muito baixo, praticamente numa base linha a linha de código, o que lhes retira grande utilidade uma vez que os analistas necessitam de especificações fáceis de actualizar e os designer's necessitam de uma descrição com um nível de abstracção mais elevado.

Assim, apesar desta técnica ser pouco utilizada, deve ter-se em conta as seguintes restrições na utilização de fluxogramas:

- restringir a utilização de fluxogramas à modelação interna de processos;
- restringir a utilização de fluxogramas à combinação aninhada de símbolos equivalentes a estruturas de controlo da programação estruturada.

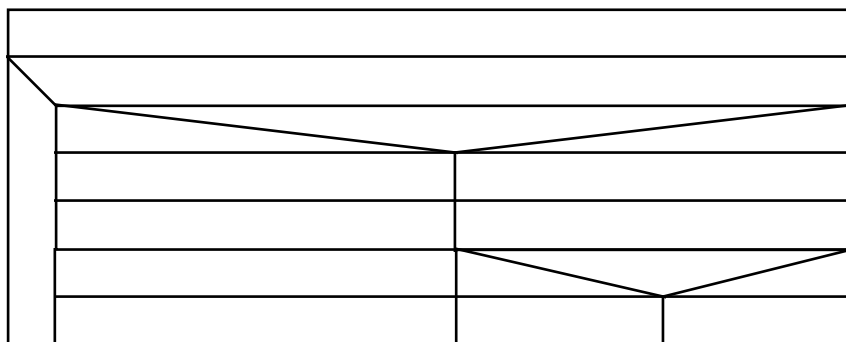
Símbolos de fluxogramas equivalentes a estruturas de controlo da programação estruturada

SEQUÊNCIA	SELECÇÃO	REPETIÇÃO
<p>“Enviar Carta”</p>	<p>“Se me sair o totoloto compro um BMW, senão compro uma bicicleta”</p>	<p>“Tocar a campainha 3 vezes”</p>
 <pre> graph TD Start(()) --> P1[P1] P1 --> P2[P2] P2 --> P3[P3] P3 --> End(()) </pre>	 <pre> graph TD Start(()) --> D{Totoloto?} D -- Sim --> P1[Comprar BMW] D -- Não --> P2[Comprar bicicleta] P1 --> Join(()) P2 --> Join Join --> End(()) </pre>	 <pre> graph TD Start(()) --> D{3 vezes?} D -- Sim --> P1[+ 1 vez] P1 --> D D -- Não --> End(()) </pre>

Diagramas de Nassi-Shneiderman

Os diagramas de Nassi-Shneiderman foram introduzidos como uma técnica de elaboração de fluxogramas estruturados.

Diagrama de Nassi-Shneiderman típico

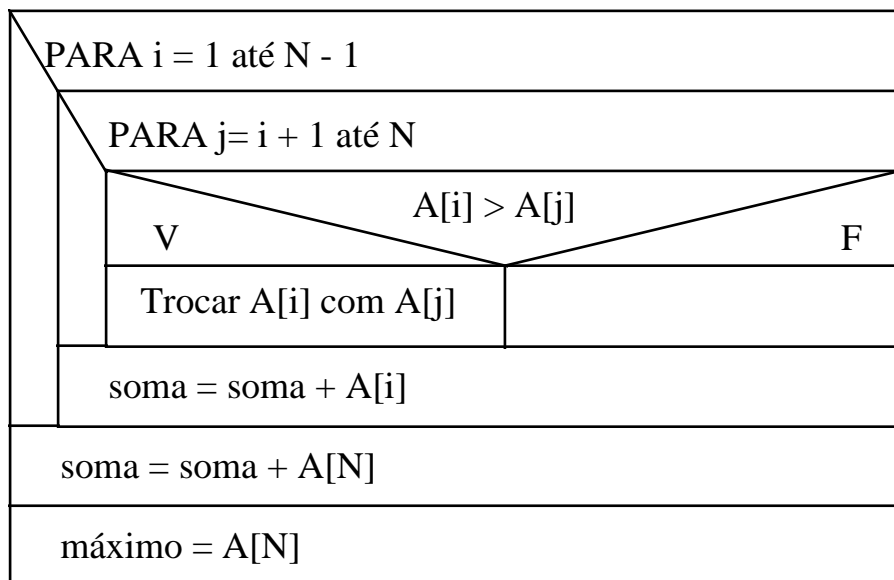


Componentes dos diagramas de Nassi-Shneiderman

COMPONENTE	REPRESENTAÇÃO						
<table border="1"> <tr> <td>$V = \text{Quant} * \text{Preço}$</td> </tr> <tr> <td>$\text{Comiss} = \text{perc} * V$</td> </tr> </table>	$V = \text{Quant} * \text{Preço}$	$\text{Comiss} = \text{perc} * V$	Representam frase declarativa ou um bloco de frases sequenciais				
$V = \text{Quant} * \text{Preço}$							
$\text{Comiss} = \text{perc} * V$							
<table border="1"> <tr> <td colspan="2" style="text-align: center;">$X < 0$</td> </tr> <tr> <td style="text-align: center;">V</td> <td style="text-align: center;">F</td> </tr> <tr> <td style="text-align: center;">$Va = - X$</td> <td style="text-align: center;">$Va = X$</td> </tr> </table>	$X < 0$		V	F	$Va = - X$	$Va = X$	Representam estruturas de selecção
$X < 0$							
V	F						
$Va = - X$	$Va = X$						
<table border="1"> <tr> <td colspan="2" style="text-align: center;">PARA $i = 1$ até N</td> </tr> <tr> <td colspan="2" style="text-align: center;">$\text{Total} = \text{Total} + i$</td> </tr> <tr> <td colspan="2" style="text-align: center;">$\text{Totq} = \text{Totq} + i^2$</td> </tr> </table>	PARA $i = 1$ até N		$\text{Total} = \text{Total} + i$		$\text{Totq} = \text{Totq} + i^2$		Representam estruturas de repetição
PARA $i = 1$ até N							
$\text{Total} = \text{Total} + i$							
$\text{Totq} = \text{Totq} + i^2$							

Os diagramas de Nassi-Shneiderman orientam-se por blocos sendo mais organizados, mais estruturados e mais compreensíveis do que fluxogramas. Contudo, para além de ser necessário elaborar um grande número de esquemas não triviais, estes esquemas não introduzem grande valor acrescentado. De acordo com o proferido por muitos analistas, “estes diagramas correspondem a Linguagem Estruturada com caixas à volta”.

Exemplo de diagrama de Nassi-Shneiderman



A utilização de fluxogramas e diagramas de Nassi-Shneiderman não é abençoada por muitos analistas de sistemas. Contudo, existem estudos que comprovam a preferência deste tipo de ferramentas por parte de estudantes de programação, como um mecanismo de aprendizagem. Se os estudantes preferem estas representações, é natural supor que os utilizadores também as prefiram.

Comparação entre as várias técnicas de mini-especificação

TÉCNICA	PRINCIPAIS VANTAGENS
Linguagem Estruturada	<ul style="list-style-type: none"> • Estruturada • Abstracta • Versátil (+ ou - detalhes) • Curva de aprendizagem mínima para analista • Abrangente • Convertida facilmente em programa
Pré/Pós Condições	<ul style="list-style-type: none"> • Fornece perspectiva do assunto • Alta abstracção • Boa entrada para designer/programador • Facilmente mantida
Tabelas e Árvores de Decisão	<ul style="list-style-type: none"> • Gráficas e facilmente entendidas • Alta abstracção • Exaustivas • Facilmente mantidas • Tratam conjuntos complexos de decisões e acções
Fluxogramas	<ul style="list-style-type: none"> • Gráfica • Correspondência com código subjacente
Diagramas de Nassi-Shneiderman	<ul style="list-style-type: none"> • Estruturada • Gráfica • Convertida directamente em programa • Alguma versatilidade na especificação de detalhes (devido a orientação por blocos)

TÉCNICA	PRINCIPAIS DESVANTAGENS
Linguagem Estruturada	<ul style="list-style-type: none"> • Tende a se parecer com código • Curva de aprendizagem elevada para utilizador • Alguma dificuldade de manutenção
Pré/Pós condições	<ul style="list-style-type: none"> • Utilização limitada • Curva de aprendizagem moderada tanto para o analista como para o utilizador
Tabelas e Árvores de Decisão	<ul style="list-style-type: none"> • Utilização específica e limitada • Necessitam de explicação textual
Fluxogramas	<ul style="list-style-type: none"> • Trabalhosa • Presta-se à utilização de programação não estruturada • Muito difícil de manter actualizado se se alterarem detalhes de especificação ou de desenho
Diagramas de Nassi-Shneiderman	<ul style="list-style-type: none"> • Trabalhosa • Curva de aprendizagem moderada • Difícil de manter actualizada se se alterarem detalhes de especificação ou de desenho.

CRITÉRIOS DE SELECÇÃO DE TÉCNICAS DE ESPECIFICAÇÃO

- natureza de cada processo a especificar, o que poderá conduzir à utilização de várias técnicas;
- facilidade de comunicação com utilizador;
- utilidade da especificação produzida não só para o analista como para o designer/programador;
- tipo de aplicação que está a ser construída;
- facilidades proporcionadas pela ferramenta CASE (*Computer Aided Software Engineering*) que está a ser utilizada.