

Teoria da Computação

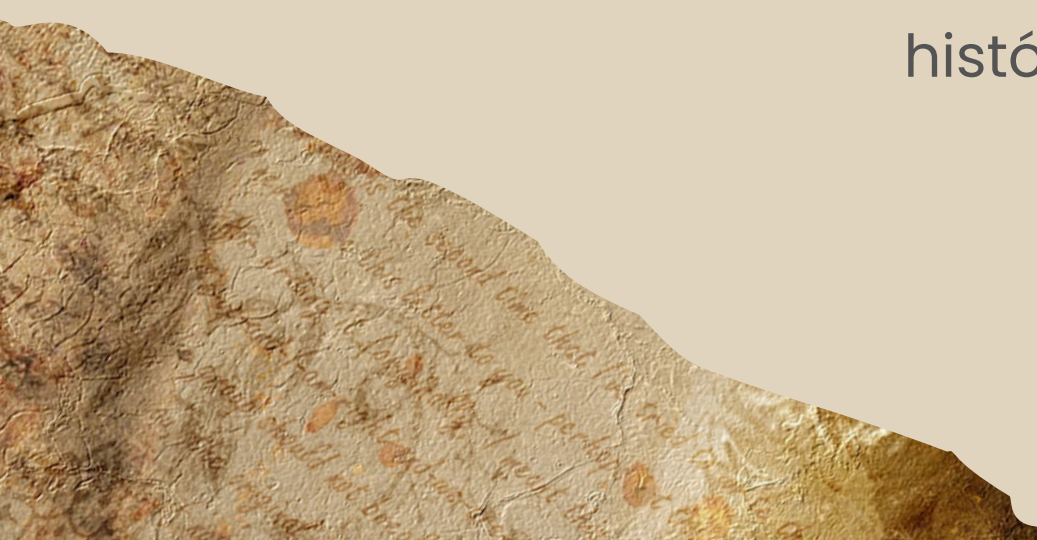
Mais pérolas de
Programação

Por Davi André,
Caio Alves e
Anderson Santos

Column 4: Self-Describing Data



- ✓ Dados não devem depender exclusivamente do código-fonte para serem compreendidos.
- ✓ As duas técnicas abordadas por Bentley:
 - ✓ **Pares Nome-Valor:** Para embutir contexto na estrutura do dado.
 - ✓ **Proveniência:** Para registrar a história e a origem do dado.



Caso 4 - O Pesadelo dos Dados Soltos



- ✓ Dados sem contexto são inúteis e propensos a erros graves de interpretação.
- ✓ Variáveis sem nomes ou unidades claras, Dificuldade em recriar o experimento original.
- ✓ Ausência de procedência dos dados

Exemplo:

3.2% -12.0% 1.1%
12.7% 0.8% 8.6%
1.6% -8.3% 9.2%

Caso 4 - Solução 1: Pares Nome-Valor



Exemplo:

- ✓ Fácil leitura para humanos e máquinas.
- ✓ A ordem dos campos não altera o processamento
- ✓ Estrutura clássica e legível: Atributo | Valor

```
name | Nimitz  
class | CVN  
speed | 30
```



Caso 4 - Otimização: Dicionários de Dados



Exemplo de dicionário:

```
name -> na (texto)  
speed -> sp (nós/knots)
```

O dado salvo:

```
naNimitz|sp30|of447
```

- ✓ O gargalo: Repetir os nomes dos atributos gasta muito espaço.
- ✓ A solução: Formato compactado + Dicionário de Dados.

Caso 4 - Solução 2: Proveniência de Dados



- ✓ O que é? A origem, o histórico ou o pedigree do dado.
- ✓ O problema: "Como esse gráfico perfeito foi gerado?"
- ✓ A regra de ouro: O arquivo de dados deve armazenar um log de suas próprias transformações.



Caso 4 - Proveniência na Prática



Exemplo:

- ✓ Uma cadeia de programas (Pipeline)
sim.events → sample → bins (Um envia a saída para o outro)
- ✓ Cada programa copia o histórico do programa anterior e adiciona o seu próprio comando no topo do arquivo.

```
!trail sim.events -k 1.5 -l 3
```

(Adicionado pelo 1º programa)

```
!trail sample -t .01
```

(Adicionado pelo 2º programa)

```
!trail bins -t .01
```

(Adicionado pelo 3º programa)

```
!dict bin_bottom_value item_count
```

(Dicionário das colunas)

Caso 4 - O Laboratório de Ordenação



✓ O Pedido:

Apenas os parâmetros do teste.

```
n | 100000
alg | quicksort
```

✓ A Prova Documentada:

Contexto + Resultados no mesmo lugar.

```
n | 100000 (Proveniência)
alg | quicksort (Proveniência)
cpu | 0.1083 (Resultado)
```

Reprodutibilidade Garantida:

Qualquer pessoa com o arquivo de saída tem a "receita" exata para refazer o experimento.

Os Princípios



Design Linguístico

A melhor documentação muitas vezes não é um manual, mas uma linguagem de programação limpa e um design de dados claro.

Legibilidade dos dados

O melhor lugar para armazenar a proveniência do dado é dentro do próprio dado.

A regra de I/O

A saída de um programa deve ser um formato legível, preferencialmente adequado para ser a entrada de outro programa..

Obriqada