Captura de Proveniência na Soma de Dois Números com DfAnalyzer

27 de agosto de 2025

1 Sobre Este Documento

Este tutorial foi elaborado por Débora Barbosa Pina com finalidade exclusivamente didática. O exemplo apresentado é propositalmente simples, servindo apenas para ilustrar os conceitos e mostrar o uso da ferramenta DfAnalyzer [1] [1], desenvolvida por Vítor Silva.

2 Introdução

A captura de proveniência é fundamental para compreender e auditar a execução de programas e transformações de dados, pois registra tanto o que foi planejado (proveniência **prospectiva**) quanto o que realmente ocorreu (proveniência **retrospectiva**).

Este exemplo utiliza a **DfAnalyzer** para modelar e registrar a execução de um *script* simples que soma dois números. O *script* está dividido em duas etapas (*i.e.*, atividades) principais:

1. **Extrair números** a partir de um arquivo.
2. **Executar a soma** desses números.

A DfAnalyzer é responsável por registrar, de forma estruturada, quais dados foram utilizados, quais transformações foram aplicadas e quais resultados foram obtidos.

3 Estrutura do Script

O código realiza a soma de dois números, mas com instrumentação para capturar a proveniência. As etapas são representadas como **transformações** nos níveis das proveniências prospectivas e retrospectivas.

As duas etapas são:

- **Transformação 1 – Extrair números**
  - Entrada: arquivo contendo os números.
  - Saída: dois atributos numéricos (`PRIMEIRO_NUMERO` e `SEGUNDO_NUMERO`).

[1] `https://gitlab.com/ssvitor/dataflow_analyzer`

- **Transformação 2 - Executar soma**
  - Entrada: os números extraídos na etapa anterior.
  - Saída: resultado da soma (`RESULTADO_SOMA`).

4 Proveniência Prospectiva

A proveniência prospectiva descreve o **fluxo de dados planejado**, ou seja, a especificação de como o programa deve funcionar, antes de ser executado.

No código, isso é feito com a definição de `Transformations` e `Sets`:

```
tf1 = Transformation("ExtrairNumeros")
tf1_input = Set("iExtrairNumeros", SetType.INPUT,
    [Attribute("SOMA_FILE", AttributeType.FILE)])
tf1_output = Set("oExtrairNumeros", SetType.OUTPUT,
    [Attribute("PRIMEIRO_NUMERO", AttributeType.NUMERIC),
     Attribute("SEGUNDO_NUMERO", AttributeType.NUMERIC)])
```

A transformação `ExtrairNumeros` lê um arquivo (`SOMA_FILE`) e recupera os dois atributos numéricos (`PRIMEIRO_NUMERO` e `SEGUNDO_NUMERO`).

Em seguida, a segunda transformação é definida:

```
tf2 = Transformation("ExecutarSoma")
tf1_output.set_type(SetType.INPUT)
tf1_output.dependency = tf1._tag
tf2_output = Set("oExecutarSoma", SetType.OUTPUT,
    [Attribute("RESULTADO_SOMA", AttributeType.NUMERIC)])
```

A transformação `ExecutarSoma` depende da saída da etapa anterior (`oExtrairNumeros`) e gera o resultado da soma (`RESULTADO_SOMA`).

Essas transformações são adicionadas ao `Dataflow` (`df`) e salvas, compondo o grafo de proveniência.

5 Proveniência Retrospectiva

A proveniência retrospectiva registra o **histórico de execução real**, documentando os dados concretos lidos, processados e produzidos.

5.1 *Task* 1: Extrair números

```
t1 = Task(1, dataflow_tag, "ExtrairNumeros")
t1_input = DataSet("iExtrairNumeros", [Element(["/path/numeros"])
   ])
t1.add_dataset(t1_input)
t1.begin()
PRIMEIRO_NUMERO = 5
SEGUNDO_NUMERO = 1
t1_output= DataSet("oExtrairNumeros", [Element([PRIMEIRO_NUMERO,
                                        SEGUNDO_NUMERO])])
t1.add_dataset(t1_output)
```

```
t1.end()
```

Representa a execução real da transformação `ExtrairNumeros`.

- Entrada real: caminho do arquivo `numeros`.
- Saída real: os números 5 e 1.

Observação: nesta execução, a leitura do arquivo indicado no caminho `/path/numeros` é apenas ilustrativa. Não há processamento real do conteúdo do arquivo; em vez disso, os valores dos números (5 e 1) nas linhas 5 e 6 foram definidos manualmente para fins de demonstração.

5.2 *Task* 2: Executar soma

```
t2 = Task(2, dataflow_tag, "ExecutarSoma", dependency=t1)
t2.begin()
RESULTADO_SOMA = PRIMEIRO_NUMERO + SEGUNDO_NUMERO
t2_output= DataSet("oExecutarSoma", [Element([RESULTADO_SOMA])])
t2.add_dataset(t2_output)
t2.end()
```

Representa a execução real da transformação `ExecutarSoma`.

- Entrada real: números 5 e 1.
- Saída real: resultado da soma 6.

Cada *Task* é ligada à respectiva transformação definida prospectivamente, criando a relação entre **o que foi planejado** e **o que foi executado**.

6 Relação entre Prospectiva e Retrospectiva

- **Prospectiva**: descreve que existe uma etapa para extrair números de um arquivo e outra para somá-los.
- **Retrospectiva**: mostra que, nesta execução específica:
  - O arquivo de entrada foi `/path/numeros`.
  - Os números extraídos foram 5 e 1.
  - O resultado obtido foi 6.

Assim, é possível reconstruir a execução e rastrear a origem do resultado final.

7 Conclusão

O código exemplifica como instrumentar um programa simples para capturar proveniência com a **DfAnalyzer**.

- A proveniência prospectiva define o grafo de fluxo de dados planejado (transformações, entradas e saídas esperadas).

- A proveniência retrospectiva registra o que realmente ocorreu em uma execução, incluindo arquivos lidos, valores processados e resultados obtidos.

Esse mecanismo garante **rastreabilidade, auditoria e reprodutibilidade** do processo, mesmo em um exemplo simples como a soma de dois números.

Observação: Este documento é um tutorial inicial e pode ser melhorado. Caso tenha ideias ou sugestões, entre em contato pelo e-mail: `dbpina@cos.ufrj.br`.

Referências

[1] Vítor Silva, Vinícius Campos, Thaylon Guedes, José Camata, Daniel de Oliveira, Alvaro L.G.A. Coutinho, Patrick Valduriez, and Marta Mattoso. Dfanalyzer: Runtime dataflow analysis tool for computational science and engineering applications. *SoftwareX*, 12:100592, 2020.