

# Treinamento SOA com IIB

*Nível III*

**Março/2018**

CONTROLE DE VERSÕES

<b><u>Versão</u></b>	<b><u>Data</u></b>	<b><u>Autor (es)</u></b>	<b><u>Descrição das Alterações</u></b>
1.0	03/2018	Sergio Fonseca da Silva	Criação

## ÍNDICE

1	APRESENTAÇÃO .....	4
2	MODELO DE SERVIÇO. ....	4
2.1	ENTENDENDO OS PROJETOS E SUAS CAMADAS.....	4
3	IMPLEMENTAÇÃO.....	5
3.1	IMPLEMENTANDO O MODELO CANÔNICO. ....	5
3.2	RS - IMPLEMENTANDO A CAMADA DE EXPOSIÇÃO OU SERVIÇO CORPORATIVO.....	8
3.3	RS - IMPLEMENTANDO A CAMADA DE VALIDAÇÃO. ....	12
3.4	RS - VINCULANDO O CANÔNICO AO PROJETO DE CORPORATIVO.....	18
3.5	RS - IMPLEMENTANDO A CAMADA ADAPTER – JSON PARA CANÔNICO.....	19
3.6	RS - IMPLEMENTANDO A CAMADA DE MEDIATOR. ....	22
3.7	CP - IMPLEMENTANDO A CAMADA DE COMPOSIÇÃO.....	24
3.8	EN - IMPLEMENTANDO A CAMADA DE MEDIATOR. ....	25
3.9	EN - IMPLEMENTANDO A CAMADA DE ENABLE – PARTE I. ....	26
3.10	EN - VINCULANDO O CANÔNICO AO PROJETO DE ENABLE.....	31
3.11	EN - IMPLEMENTANDO A CAMADA ADAPTER – CANÔNICO PARA JSON. ....	32
3.12	EN - IMPLEMENTANDO A CAMADA DE ENABLE – PARTE II. ....	34
3.13	EN - IMPLEMENTANDO A CAMADA ADAPTER – JSON PARA CANÔNICO. ....	35
3.14	RS - IMPLEMENTANDO A CAMADA ADAPTER – CANÔNICO PARA JSON. ....	38
3.15	LIGAÇÕES FINAIS.....	43
3.16	REALIZANDO DEPLOY.....	46

## 1 Apresentação

Dando continuidade ao modelo de arquitetura corporativa apresenta no curso anterior “Nível II”, vamos manter, praticamente a mesma arquitetura de desenvolvimento, entretanto iremos adicionar os seguintes itens:

- Construção de uma Application.
- Elaboração de uma biblioteca compartilhada.
- Definição do modelo canônico.
- Orquestração e composição de Serviços.
- Invocação de projetos externos.

## 2 Modelo de serviço.

Ainda tendo como referência os cursos anteriores, vamos construir o mesmo serviço de cotação de dólar, porem atendendo os requisitos apresentados acima.

### 2.1 Entendendo os projetos e suas camadas.

Nessa modelo de arquitetura para desenvolvimento, nosso serviço de integração será composto conforme abaixo:

- A **API-REST** que será exposta no barramento com o nome: ***undbr\_busca\_cotacao\_rs\_v2***.
- O **Application**, que será utilizado para fazer a orquestração e ou composição do serviço, com o nome: ***undbr\_busca\_cotacao\_cp\_v1***.
- O **Application** que será responsável pela invocação dos serviços expostos nos legados, que receberá o nome: ***undbr\_busca\_cotacao\_en\_v1***.
- A **Shared-Library** que será responsável em agrupar e organizar as estruturas dos dados, que receberá o nome: ***undbr\_commons\_canonico\_v1***.

### 3 Implementação.

As próximas paginas serão sobre as implementações dos serviços.

#### 3.1 Implementando o modelo canônico.

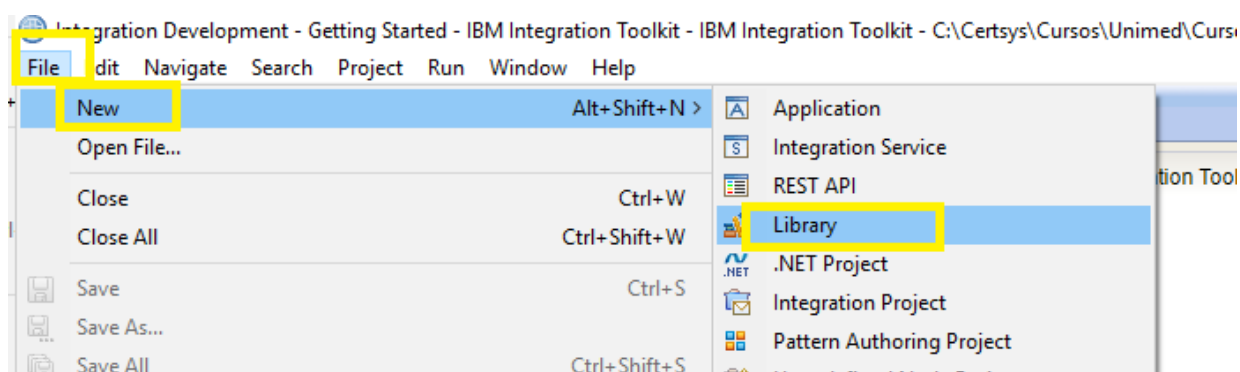
O modelo canônico de forma simplifica resume em agrupar diversos artefatos, que no nosso caso serão arquivos xsd, normalmente o modelo canônico padroniza a forma e define os padrões e estruturas de dados que devem ser compartilhados por todos os sistemas da companhia.

Entretanto esse raciocínio vem sendo polemico dentro da comunidade SOA, alguns defende o raciocínio que o modelo canônico deve estar presente apenas no barramento e deixando os demais sistemas legados autuarem conforme sua própria metodologia de desenvolvimento, não sofrendo influência da área de integração da empresa.

Segundo essa nova tendência e auxiliados pela facilidade que o IIB nos oferece vamos adotar esse novo modelo de implementação, para isso vamos construir uma **Shared-Library**:

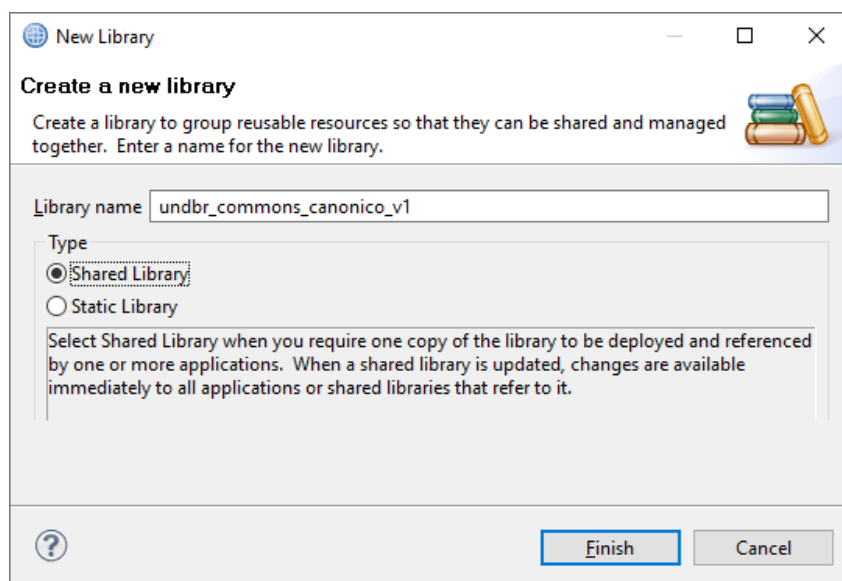
##### Iniciando a construção

##### 3.1.1. Clique em **File, New, Library**.



##### 3.1.2. Em Library name informe: ***undbr\_commons\_canonico\_v1***.

##### 3.1.3. Selecione Shared Library.



3.1.4. Clique em **Finish**.

**Observação:**

O xsd é um arquivo externo, então precisamos importar o mesmo para dentro do projeto canônico.

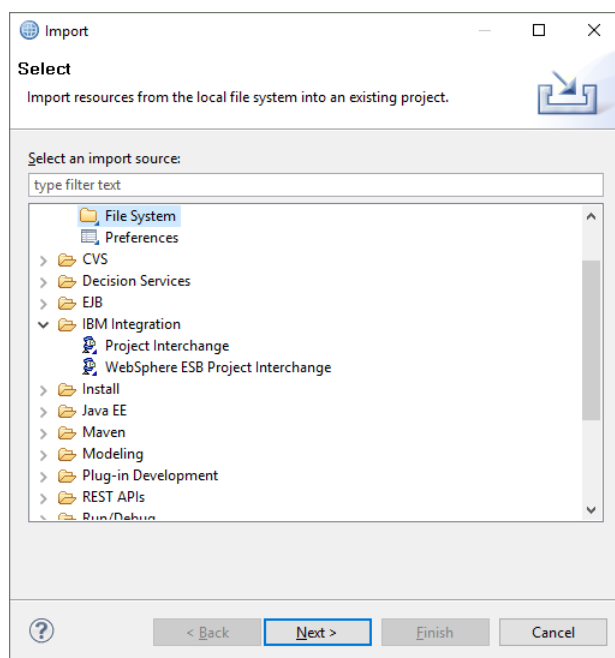
**Importando o XSD**

3.1.5. Selecione o projeto de **Shared Library** do canônico.

3.1.6. Clique em **Import**.

3.1.7. Selecione **File System**.

3.1.8. Clique em **Next**.



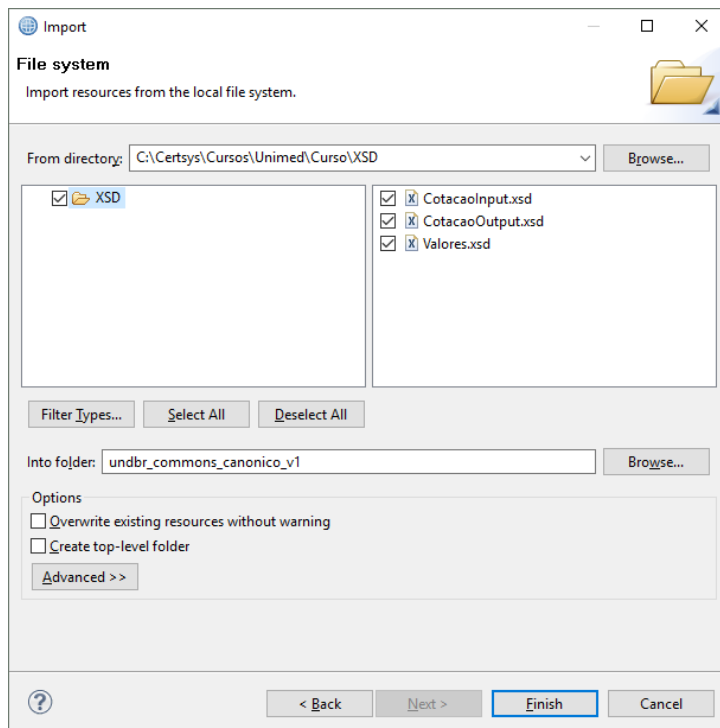
3.1.9. Clique no botão **Browser**.

3.1.10. Selecione a pasta onde o xsd se encontram.

3.1.11. Selecione a direita **XSD**, node que todos os artefatos estão selecionados.

3.1.12. Cerifique-se que em Info folder está setado para o projeto.

3.1.13. Clique em **Finish**.



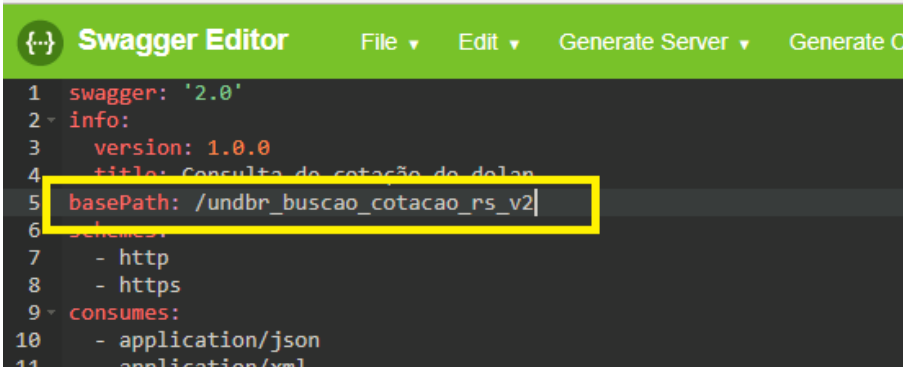
## 3.2 RS - Implementando a camada de exposição ou serviço corporativo.

A primeira etapa em rumo ao novo padrão de desenvolvimento é reutilizarmos o Swagger adotado na apostila anterior, mudaremos a tag **basePath** do arquivo Swagger para suportar a nova versão do serviço.

Essa é uma boa prática para permitir dupla convivência de versões diferentes no mesmo projeto.

Nessa camada termos apenas os patterns: **adapter**, **mediator** e **validate**, já comentados anteriormente, portanto temos que:

- 3.2.1. No browser, abrir o arquivo **Swagger** no editor on-line.
- 3.2.2. Altere o basePath para: **/undbr\_busca\_cotacao\_rs\_v2**.
- 3.2.3. Realize o download do mesmo.
- 3.2.4. Disponibilize na pasta Swagger local da máquina.
- 3.2.5. Renomeei o nome do arquivo json conforme sua convencia, mas que tenha sentido com relação ao projeto.



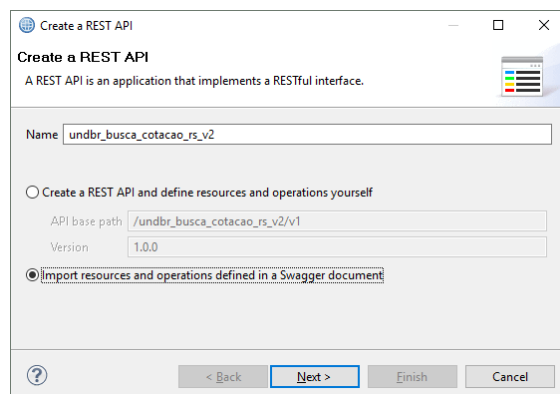
```
{...} Swagger Editor  File Edit Generate Server Generate C
1 swagger: '2.0'
2 info:
3   version: 1.0.0
4   title: Consulta de cotação de dolar
5   basePath: /undbr_busca_cotacao_rs_v2
6 schemes:
7   - http
8   - https
9 consumes:
10  - application/json
11  - application/xml
```

### **Observação:**

Nunca coloque a versão no nome do arquivo Swagger, o fato de trocar o nome do arquivo o IIB perde a referência nos mapeamentos dos fluxos já existentes.

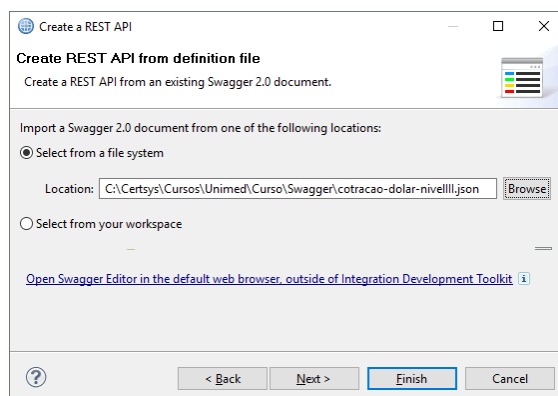


- 3.2.6. No **Workspace**, selecione **File, New, REST API**.  
Em Name digite: **undbr\_busca\_cotacao\_rs\_v2**.



- 3.2.7. Selecione **Import Resources...**

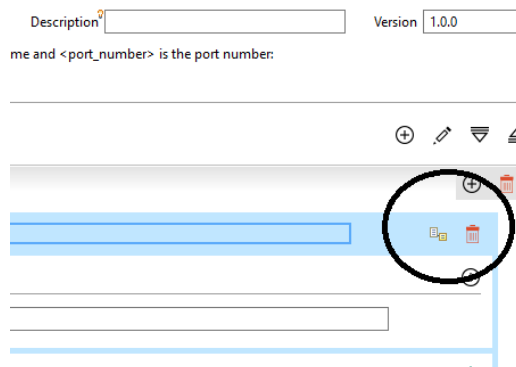
- 3.2.8. Selecione o arquivo JSON: **cotacao-dolar-nivelIII.json** na pasta de download.



- 3.2.9. Clique em **Finish**.

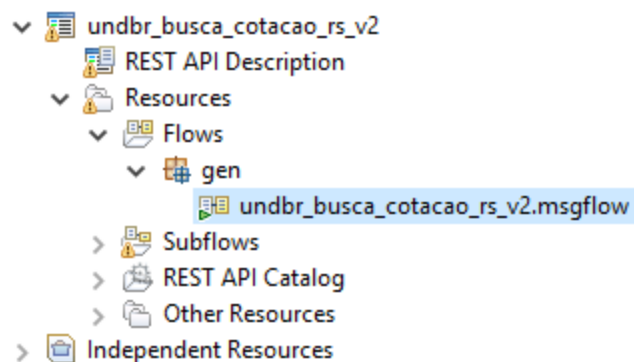
## Implementando o msgflow

- 3.2.10. Clique no botão: **Open the subflow for the operation**, ilustrado abaixo.

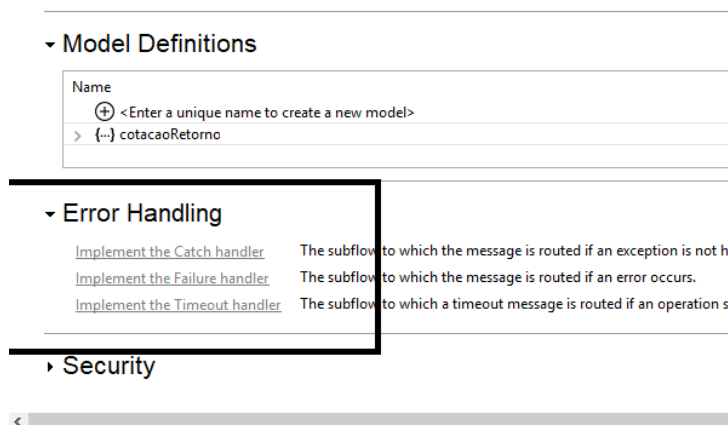


## Implementando os tratamentos para as exceções

- 3.2.11. Duplo clique no flow **undbr\_busca\_cotacao\_rs\_v2**.



- 3.2.12. Role a página para baixo no arquivo json até o bloco **Error Handling**.



3.2.13. Execute duplo clique nos links abaixo, para criar os subflow de erros correspondentes:

3.2.13.1. ***Implements the Cactch handler.***

3.2.13.2. ***Implements the Failure handler.***

3.2.13.3. ***Implements the Timeout handler.***

**Observação:**

No momento oportuno vamos implementar os fluxos de mensagens para tratamento de erro.

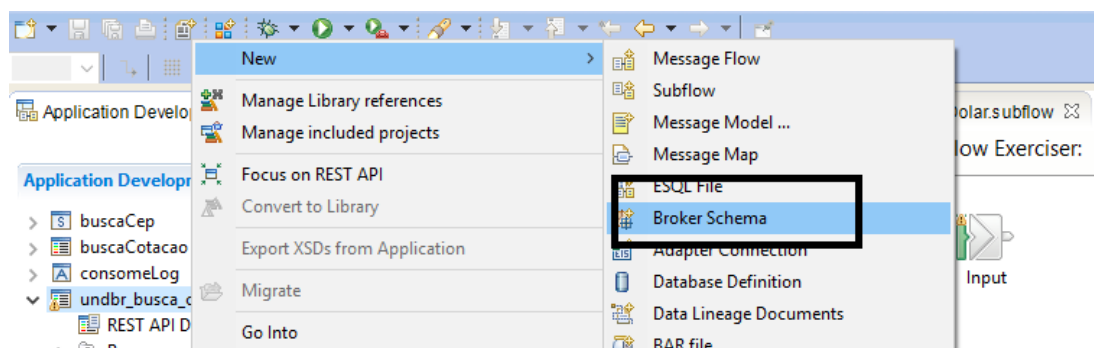
### 3.3 RS - Implementando a camada de validação.

A primeira atividade que o serviço de integração deve realizar é a validação dos dados de entrada, dessa forma caso exista alguma inconformidade com os mesmos o serviço notifica o consumidor e suspende a execução do serviço, economizando processamento desnecessário.

Dando ênfase no nosso modelo de desenvolvimento vamos criar um Broker Schema que é utilizado para definirmos a organização estrutural do projeto.

3.3.1. Selecione o projeto.

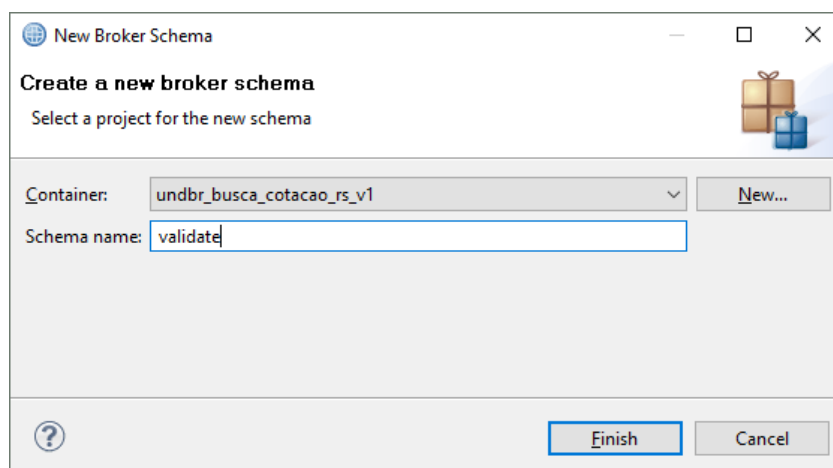
3.3.2. Clique em **New**



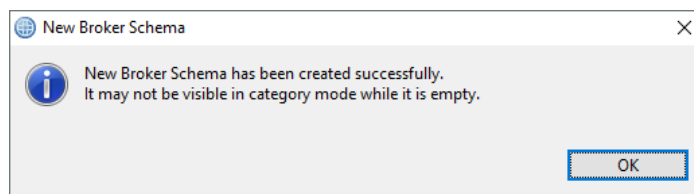
3.3.3. Clique em **Broker Schema**.

3.3.4. Certifique que o **Container** selecionado é o serviço que estamos definindo.

3.3.5. Em **Schema name**, digite o nome de nosso pacote: **validate**.



3.3.6. Ao clicar em **Finish**, será apresentada a seguinte mensagem.

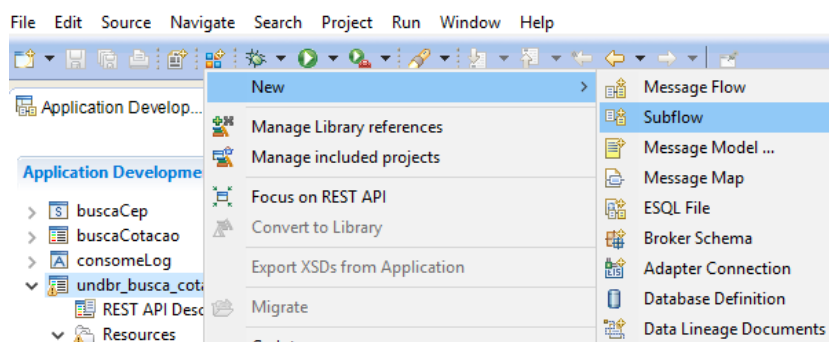


Agora vamos construir nosso subflow de validação, essa validação consiste em verificar se o consumidor do serviço enviou o parâmetro de entrada correspondente a quantidade de dias, caso negativo vamos interromper o fluxo e notificar o consumidor. Para isso:

3.3.7. Selecione o projeto.

3.3.8. Clique em **New**.

3.3.9. Clique em **Subflow**.



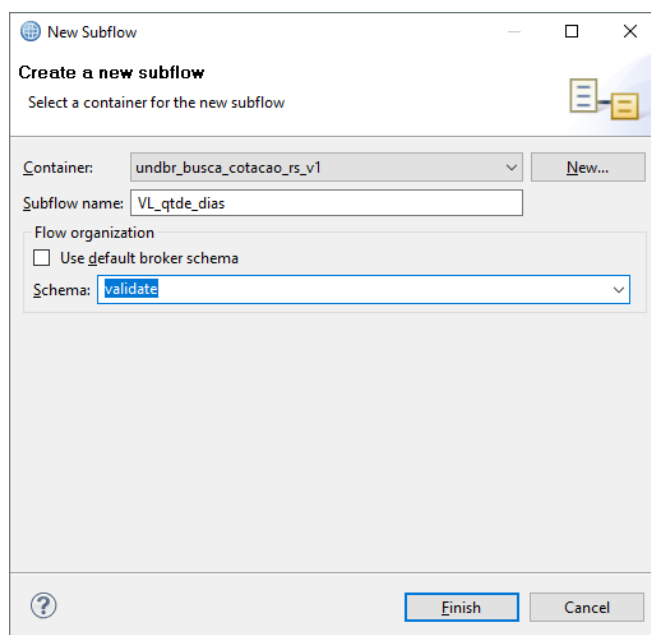
3.3.10. Em Name digite: **VL\_qtde\_dias**, sendo:

3.3.11. Certifique que o Container selecionado é o serviço em implementação.

3.3.12. Digite o seguinte nome para o subflow: **VL\_qtadeDias**

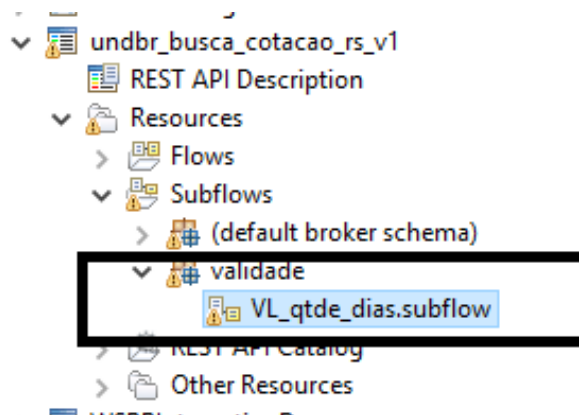
3.3.13. Deselecione a opção: **Use default broker schema**.

3.3.14. Em schema selecione o broker schema que criamos, nesse caso: **validate**.



3.3.15. Clique em **Finish**.

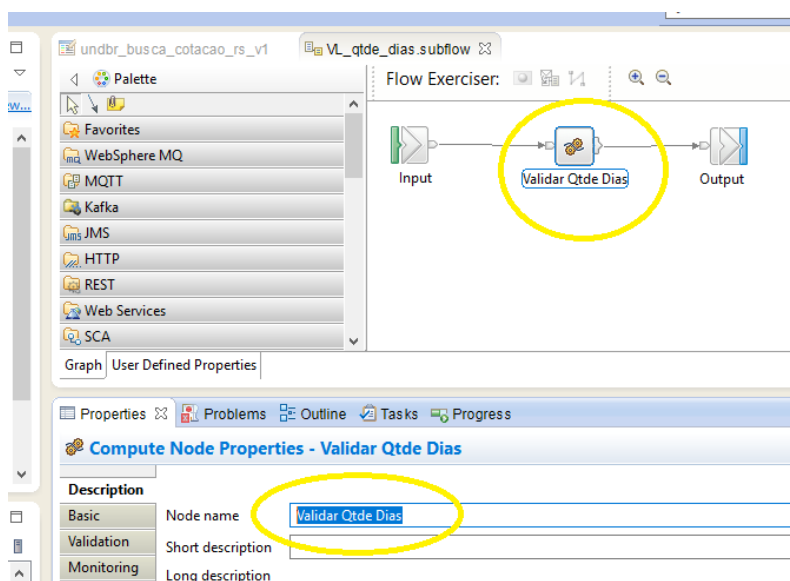
**Ao concluir as etapas teremos**



3.3.16. Duplo clique no subflow **VL\_qtde\_dias**, para abrir a área de trabalho.

3.3.17. Selecione o node de compute e arraste para área de trabalho.

3.3.18. Selecione aba de **description** e em node **name** informe: **Validar Qtde Dias**.



3.3.19. Duplo clique no node de **compute node**, será aberto a área de edição dos comandos **ESQL**.

3.3.20. Entre o **begin** e o **end**, implemente o seguinte código **ESQL** para validação.

```
SET OutputRoot = InputRoot;
SET OutputLocalEnvironment = InputLocalEnvironment;

IF CAST(InputLocalEnvironment.REST.Input.Parameters.dias as INT) > 30 THEN
    THROW USER EXCEPTION MESSAGE 400 VALUES (601, 'Qtdade dias informado, superior
ao numero máximo permitido!');
END IF;

RETURN TRUE;
```

**Abaixo código final do ESQL no node de compute.**

```
BROKER SCHEMA validate

CREATE COMPUTE MODULE VL_qtde_dias_Compute
CREATE FUNCTION Main() RETURNS BOOLEAN
BEGIN
    SET OutputRoot = InputRoot;
    SET OutputLocalEnvironment = InputLocalEnvironment;

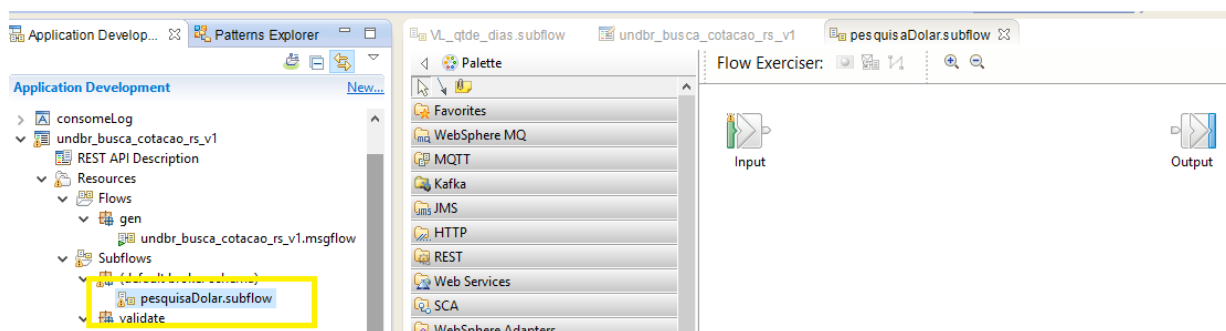
    IF CAST(InputLocalEnvironment.REST.Input.Parameters.dias as INT) > 30 THEN
        THROW USER EXCEPTION MESSAGE 400 VALUES (601, 'Qtdade dias informado, superior ao numero
        END IF;

    RETURN TRUE;
END;

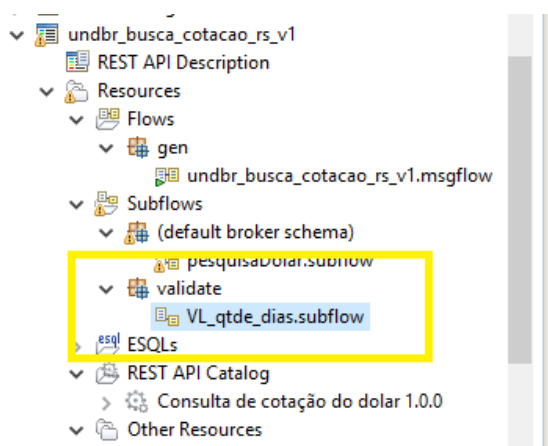
END MODULE;
```

Com as etapas anterior, concluímos a construção do subflow de validação, porem precisamos disponibilizar o mesmo no flow principal da aplicação, então:

- 3.3.21. Na lateral esquerda do toolkit, navegue até encontrar o subflow “**pesquisaDolar.subflow**”, clique duas vezes para abrir a área de desenvolvimento.

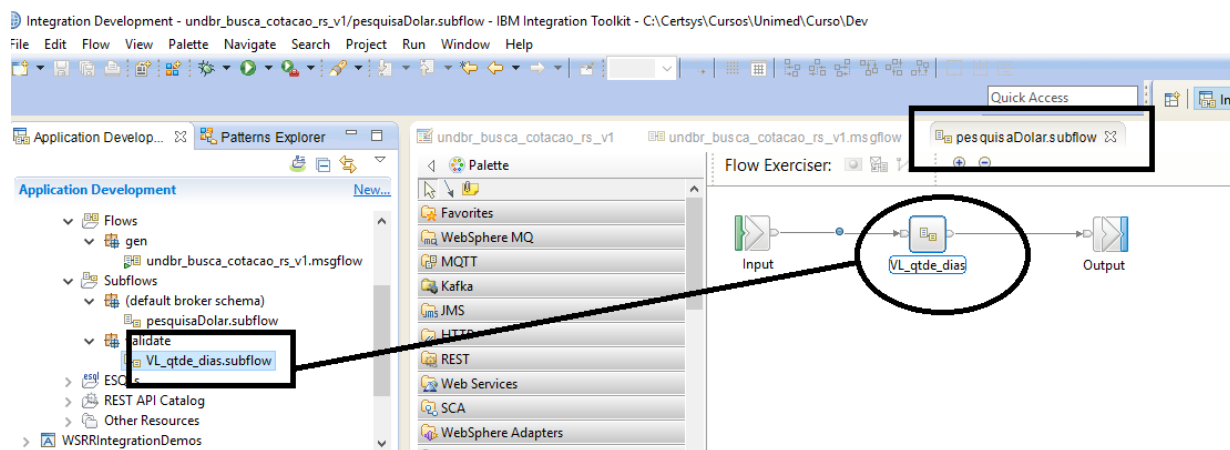


- 3.3.22. Ainda na lateral esquerda do toolkit, navegue até encontrar o subflow : **VL\_qtde\_dias**.



- 3.3.23. Selecione a estrutura Subflows “**VL\_qtde\_dias**” e araste para a área de trabalho entre os nodes **input** e o de **output** do flow **pesquisaDolar**.



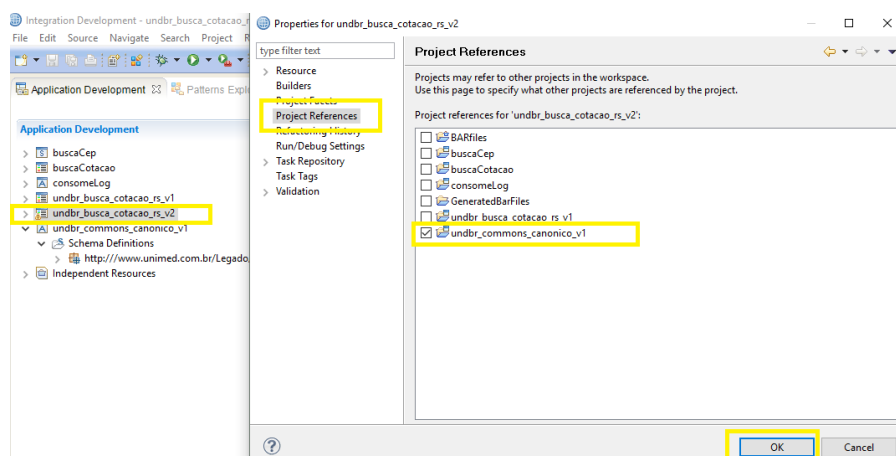


3.3.24. Realize as ligações entre os notes.

### 3.4 RS - Vinculando o canônico ao projeto de corporativo.

Para que aplicação possa ter acesso a estrutura de dados, precisamos criar um vínculo do canônico com o projeto, para isso:

- 3.4.1. Selecione o projeto que receberá o vínculo.
- 3.4.2. Clique em **Properties**.
- 3.4.3. Selecione Project **References**.
- 3.4.4. Selecione o projeto que será vinculado.
- 3.4.5. Clique em **OK**.



### 3.5 RS - Implementando a camada adapter – Json para Canônico.

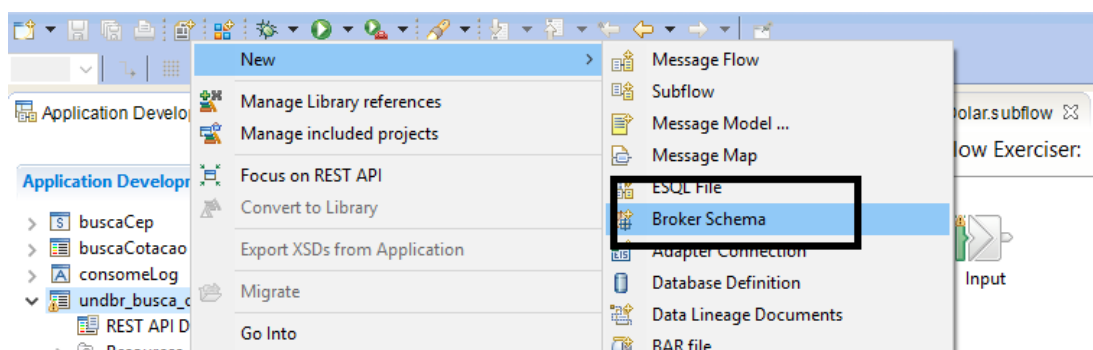
Nesse fluxo de adaptação, terá a responsabilidade de receber uma requisição **JSON** e converter para o modelo **canônico**, disponibilizando nesse formato para as demais camadas internas do barramento.

#### Criando Broker Schema

3.5.1. Selecione o projeto.

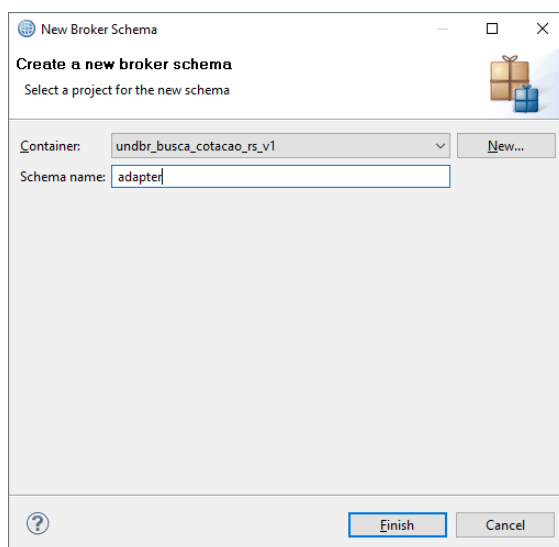
3.5.2. Clique em **New**

3.5.3. Clique em **Broker Schema**.



3.5.4. Certifique que o **container** selecionado é o serviço que estamos definindo.

3.5.5. Em **Schema name**, digite o nome de nosso pacote: **adapter**.



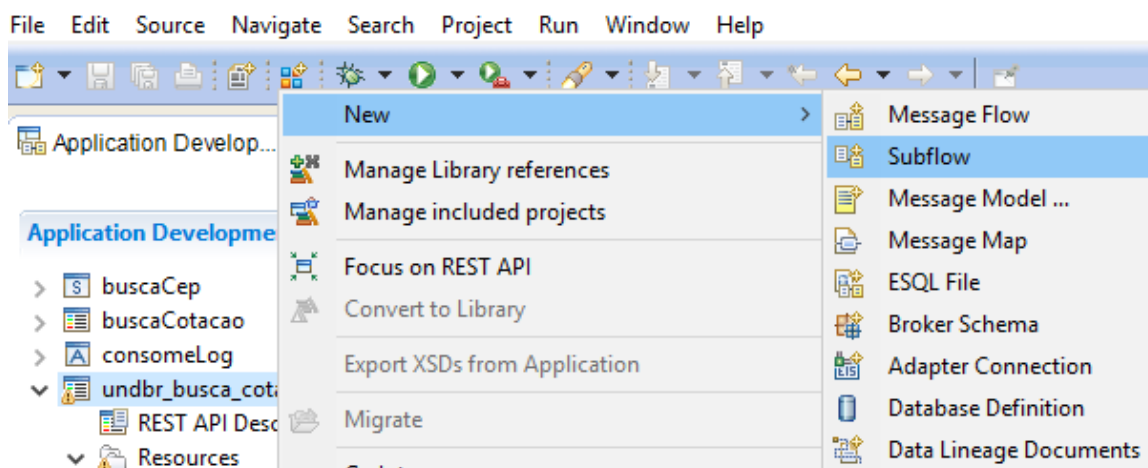
3.5.6. Clique em **Finish**.

**Vamos construir o subflow de adaptação.**

3.5.7. Selecione o projeto.

3.5.8. Clique em **New**.

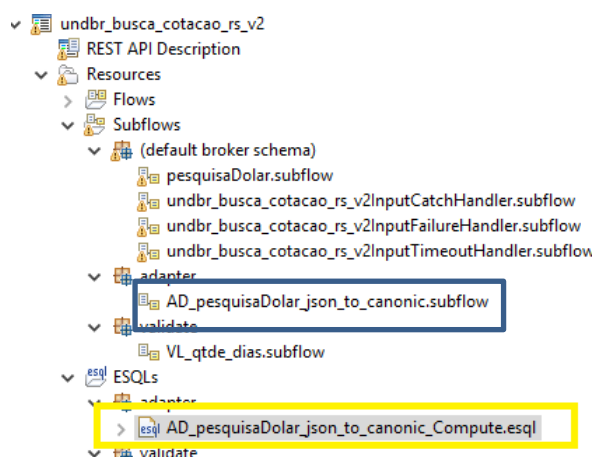
3.5.9. Clique em **Subflow**.



3.5.10. Em Name digite: **AD\_pesquisaDolar\_json\_to\_canonic**.

3.5.11. Clique em **Finish**.

**Teremos a imagem abaixo:**



**Editando o código ESQL.**

3.5.12. Selecione o subflow: AD\_pesquisaDolar\_json\_to\_canonic.  
e realize duplo clique, cole o código **ESQL** abaixo entre a tag **begin** e **end**:

```
SET OutputRoot          = InputRoot;  
SET OutputLocalEnvironment = InputLocalEnvironment;  
  
DECLARE nsDias NAMESPACE 'http://www.unimed.com.br/Legado/CotacaoInput';  
  
SET OutputRoot.XMLNSC.nsDias:Cotacao.qtdeDias =  
InputLocalEnvironment.REST.Input.Parameters.dias;  
  
RETURN TRUE;
```

**Observação:**

Não esqueça de setar o compute mode para **LocalEnvironment** and **Message**.

## 3.6 RS - Implementando a camada de mediator.

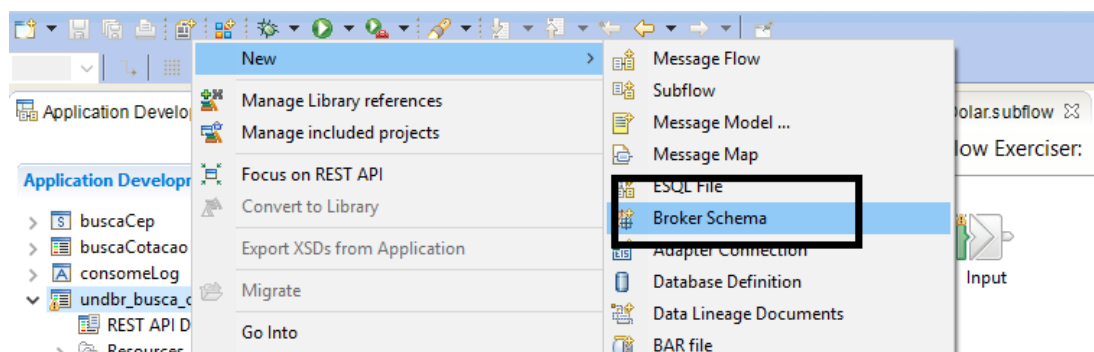
Ainda no **API-Rest**, implementados a camada mediator, responsável em separar a requisição que chega no barramento com as demais camadas ou subflows do projeto.

Dando ênfase ao modelo de desenvolvimento vamos criar um Broker Schema que é utilizado para definirmos a organização estrutural do serviço.

3.6.1. Selecione o projeto.

3.6.2. Clique em **New**

3.6.3. Clique em **Broker Schema**.



**Certifique que o container selecionado e o serviço que estamos definindo.**

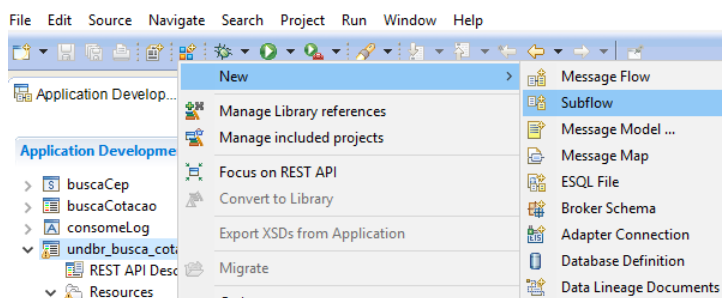
3.6.4. Em **Schema name**, digite o nome de nosso pacote: **mediator**.

**Vamos construir nosso subflow de mediação. Para isso:**

3.6.5. Selecione o projeto.

3.6.6. Clique em **New**.

3.6.7. Clique em **Subflow**.



3.6.8. Em Name digite: **MD\_pesquisaDolar**.

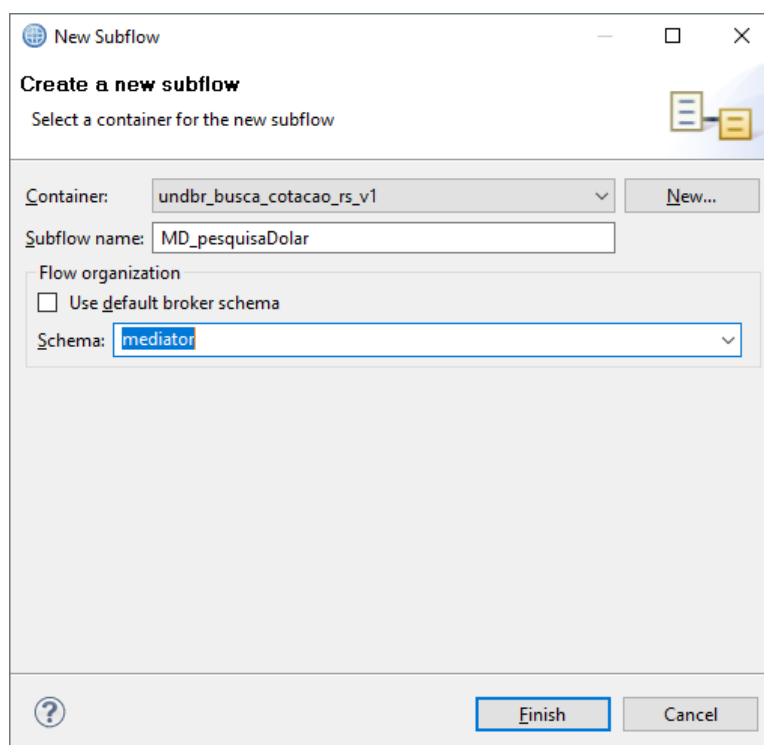
3.6.9. Certifique que o Container selecionado é o serviço em implementação.

3.6.10. Digite o seguinte nome para o subflow: **MD\_pesquisaDolar**.

3.6.11. Deselecione a opção: **Use default broker schema**.

3.6.12. Em schema selecione o broker schema que criamos, nesse caso: **mediator**.

3.6.13. Clique em **Finish**.



**Observação:**

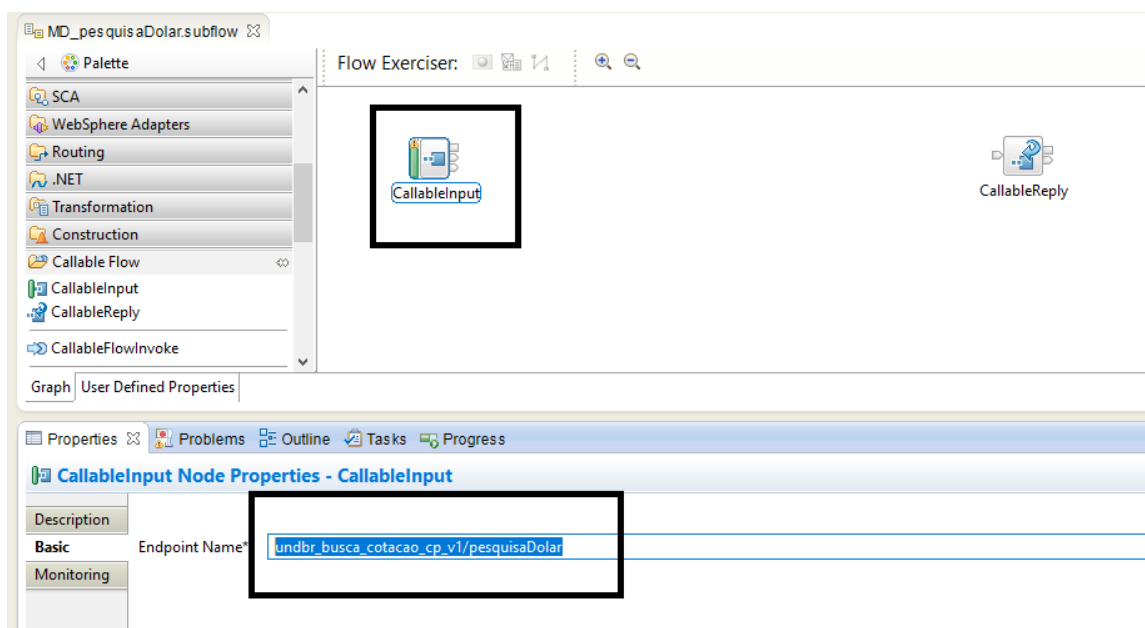
Por hora vamos parar por aqui, antes de continuarmos, precisamos primeiro criar o projeto de composição de serviços, após a conclusão do mesmo voltaremos a implementação dessa camada no item: Implementando a camada de mediator – parte II.

### 3.7 CP - Implementando a camada de composição.

Essa camada poderá ser substituída a medida que novas composição são solicitadas e perfeitamente possível a existência de dupla convivência, sua responsabilidade é realizar a orquestração dos serviços ou implementar os serviços de negócio.

#### Vamos a implementação

- 3.7.1. Clique em **File, New, Application**.
- 3.7.2. Em name informe: **undbr\_busca\_cotacao\_cp\_v1**.
- 3.7.3. Selecione e remova o nodes de **Input** e **Output**.
- 3.7.4. Na paleta de nodes, clique em **Callable Flow**.
- 3.7.5. Arraste e solte na área de trabalho os node **Callable Input** e **Callable Reply**.
- 3.7.6. Selecione o node **Callable Input** em **Endpoint Name**, informe: **undbr\_busca\_cotacao\_cp\_v1/pesquisaDolar**.



#### Observação:

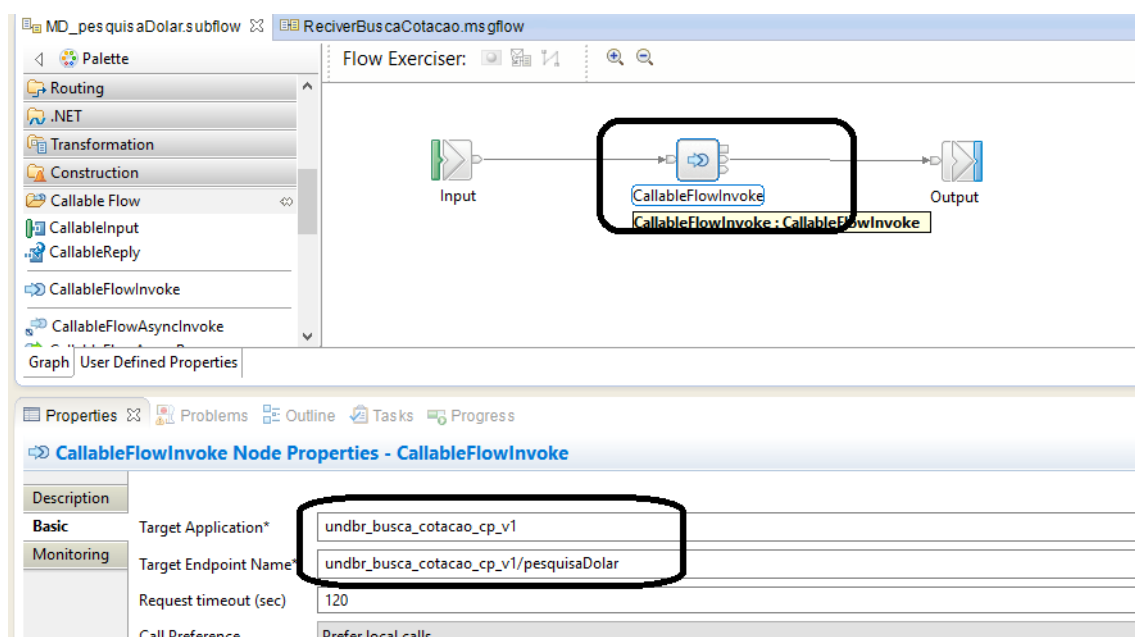
Por hora vamos parar por aqui, antes de continuamos, precisamos primeiro criar o projeto de chamadas atômicas/enable nos legados, após a conclusão do mesmo voltaremos a implementação dessa camada.



### 3.8 EN - Implementando a camada de mediator.

Voltamos novamente a nossa **API-REST**, que está com fluxo de composição parcialmente construída faltando implementar a chamada remota do mediator ao projeto composite, que criamos anteriormente, para isso:

- 3.8.1. Selecione o projeto: **undbr\_busca\_cotacao\_rs\_v2**.
- 3.8.2. Localize e realize um duplo clique no subflow: **MD\_pesquisaDolarw**.
- 3.8.3. Na paleta de nodes, clique em **Callable Flow** e arraste e solte na área de trabalho o node **CallableFlowInvoke**.
- 3.8.4. Com **CallableFlowInvoke** selecionado, clique em **Basic** e preencha as seguintes propriedades:
  - 3.8.4.1. Target Application: **undbr\_busca\_cotacao\_cp\_v1**.
  - 3.8.4.2. Target EndPoint Name: **undbr\_busca\_cotacao\_cp\_v1/pesquisaDolar**. Name:



### 3.9 EN - Implementando a camada de enable – parte I.

Até esse ponto implementamos parcialmente o serviço cooperativo que é a camada **REST-API**, também desenvolvemos o serviço de negócio que é a camada **Composite**.

Portanto nos resta implementar a ultima camada, conhecido como camada de **enable** ou camada de chamada a atômica.

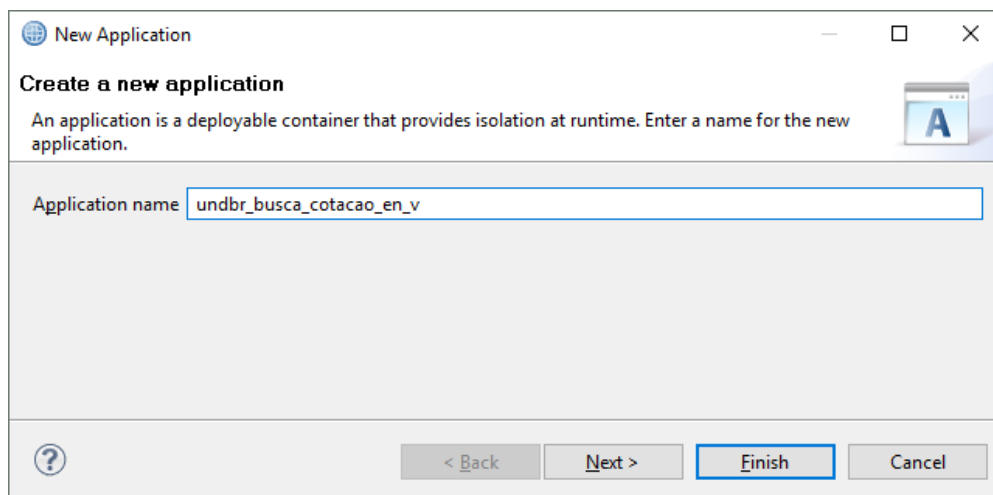
Tal camada tem a responsabilidade de realizar as chamadas aos serviços dos legados, que são os provedores de dados ao barramento.

Para isso:

3.9.1. Clique em **File, New, Application**.

3.9.2. Em name informe: **undbr\_busca\_cotacao\_en\_v1**.

3.9.3. Clique em **Finish**.

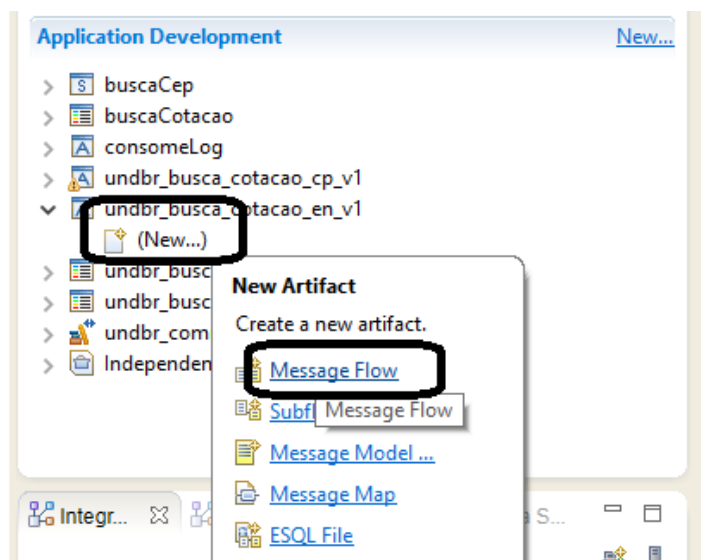


**Construindo o flow principal.**

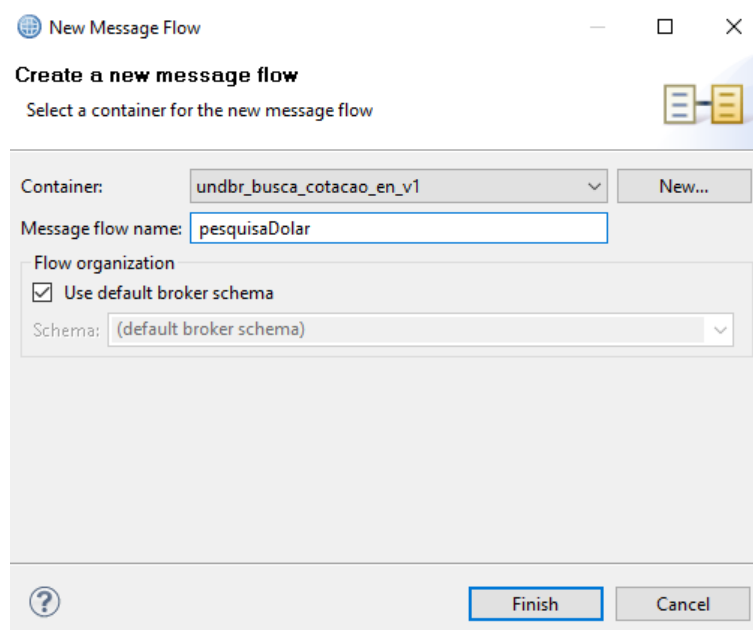
3.9.4. Selecione o projeto.

3.9.5. Clique em **New**.

3.9.6. Clique em **Message Flow**.

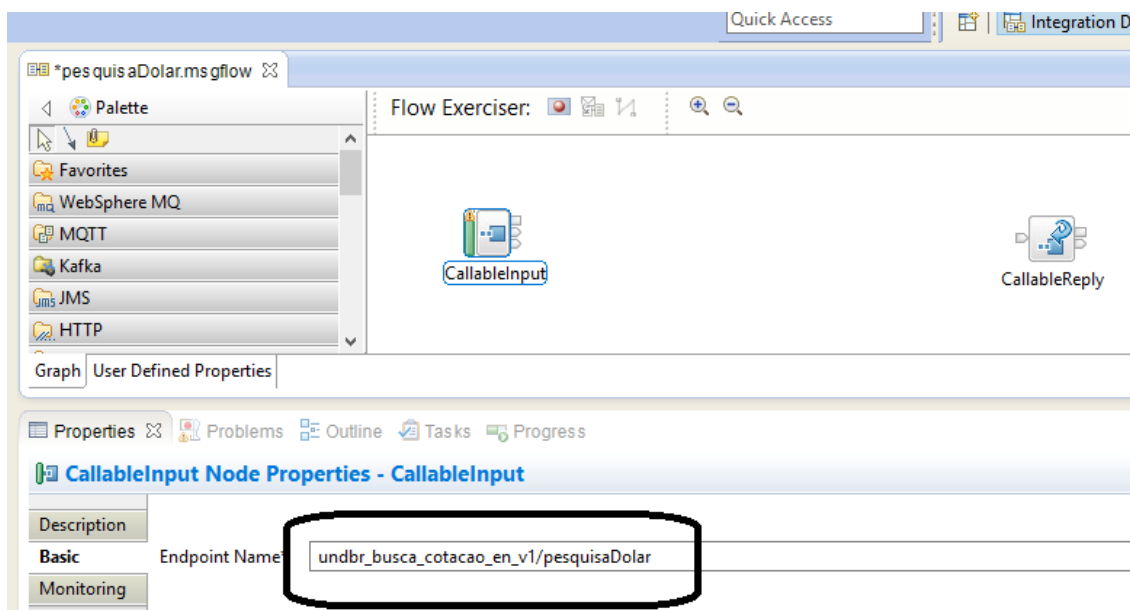


3.9.7. Em **message flow name** informe: **pesquisaDolar**.



3.9.8. Clique em **Finish**.

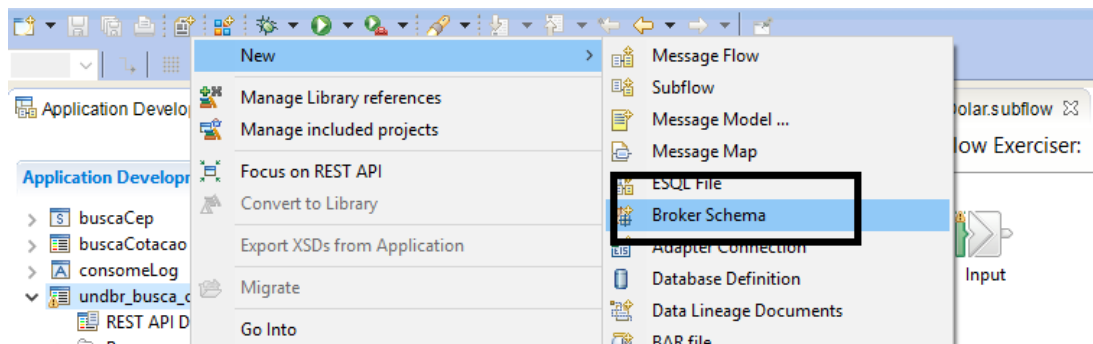
- 3.9.9. Selecione o flow criado e realize duplo clique para abrir a área de trabalho.
- 3.9.10. Na paleta de nodes, clique em **Callable Flow**.
- 3.9.11. Arraste e solte na área de trabalho os node **Callable Input** e **Callable Reply**.
- 3.9.12. Selecione o node **Callable Input** em **Endpoint Name**, informe: **undbr\_busca\_cotacao\_en\_v1/pesquisaDolar**.



**Observação:**

Precisamos construir dois brokers schemas, o de enable e o de adapter, prosseguindo:

- 3.9.13. Clique em **Broker Schema**.

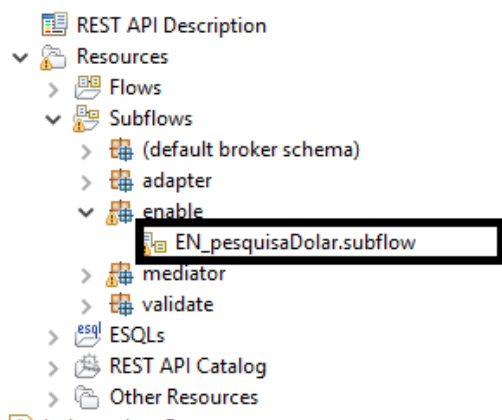


- 3.9.14. Certifique que o **container** selecionado e o serviço que estamos definindo.
- 3.9.15. Em **Schema name**, digite o nome de nosso pacote: **enable**.
- 3.9.16. Clique em Finish.

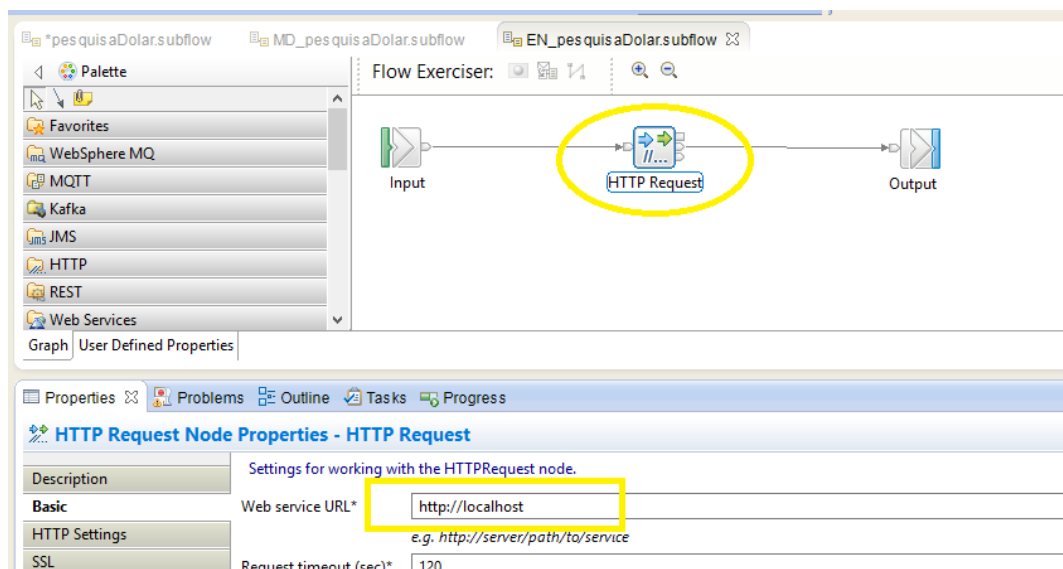
**Observação:**

Repita o procedimento acima e crie o broker schema com o nome **adapter**.

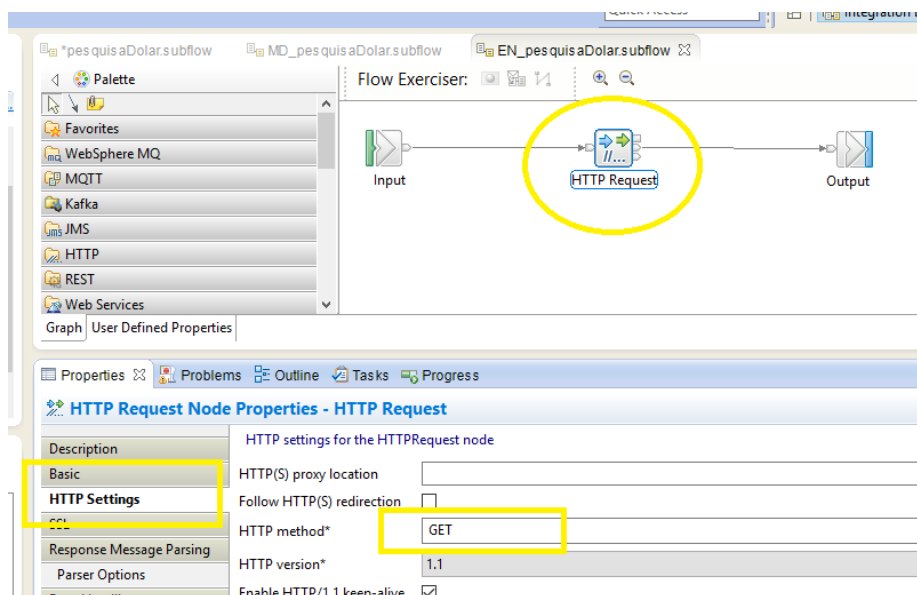
- 3.9.17. Dublo clique no subflow **EN\_pesquisaDolar.subflow**.



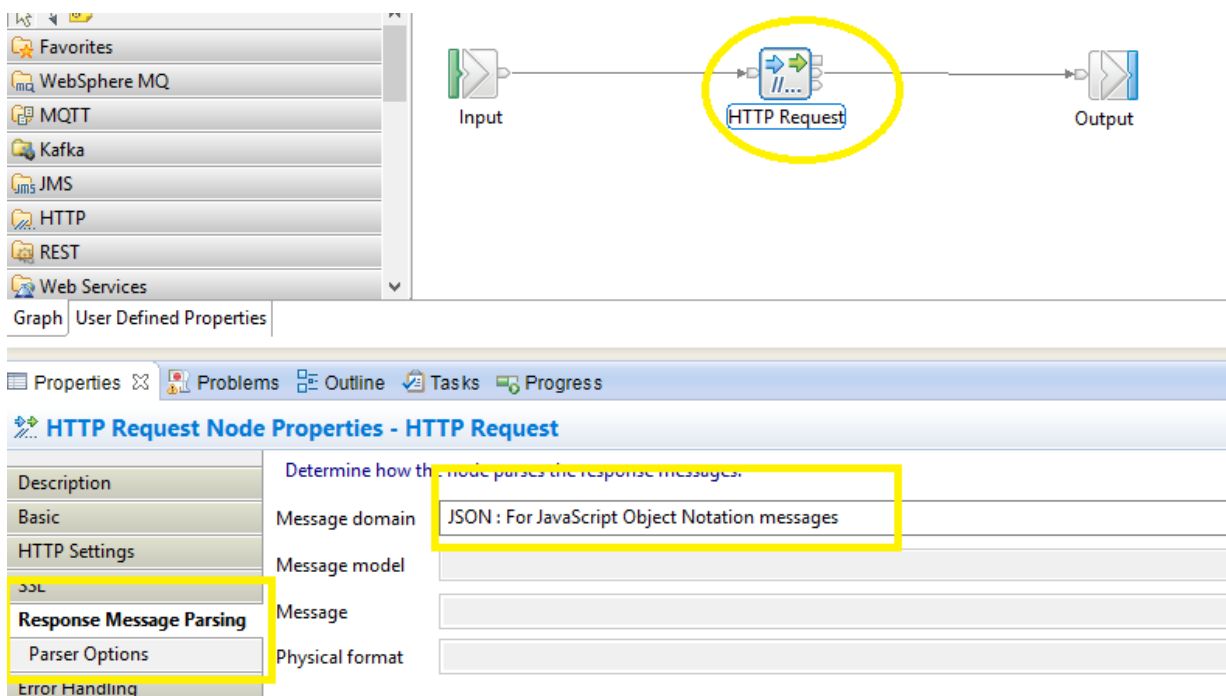
- 3.9.18. Em **Properties**, selecione **Basic** e preencha o campo **Web Service URL** com o valor **http://localhost**.



3.9.19. Em **Properties**, selecione **HTTP Settings** e selecione o campo **HTTP Method** selecione **GET**.



3.9.20. Em **Properties**, selecione **Response Message Parsing** e preencha o campo **Message Domain** selecione **JSON: For JavaScript...**



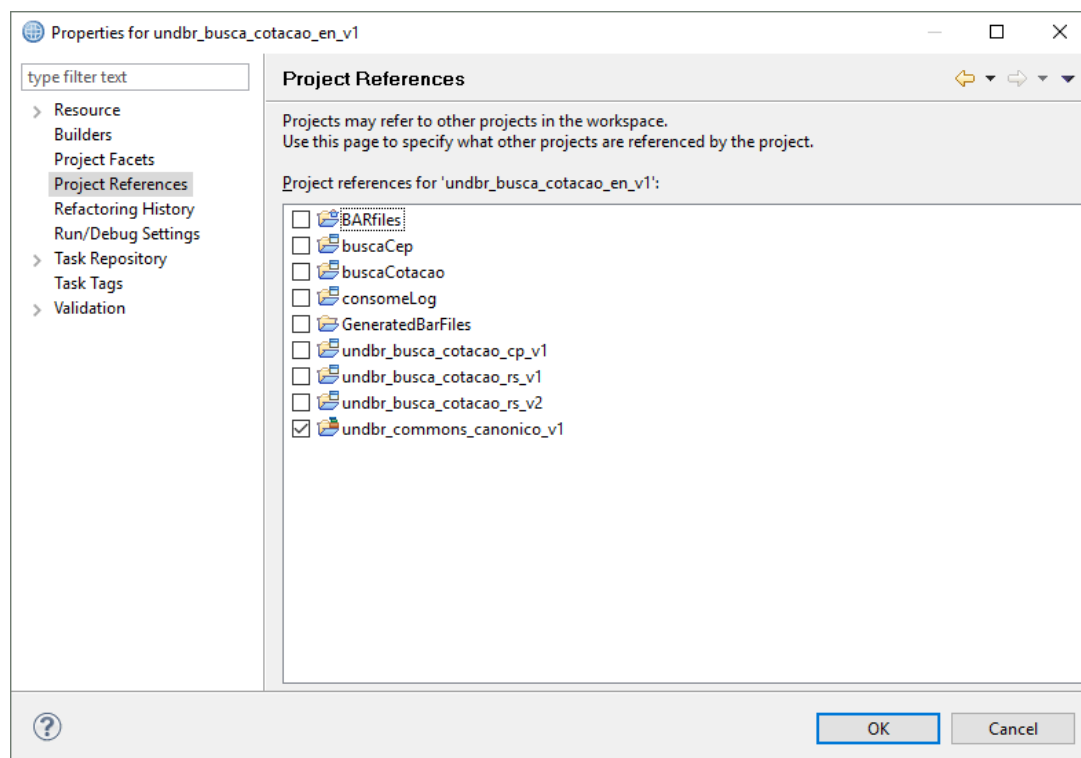
### 3.10 EN - Vinculando o canônico ao projeto de enable.

Após nos certificarmos através do teste anterior, que o serviço de integração está conseguindo realizar a invocação do serviço do legado e o respectivo legado retornando os dados ao barramento, podemos implementar os dois últimos mapeamentos de dados que consiste em:

Capturar o response do legado e converter para o canônico no projeto de **EN** e depois pegar do **canônico** e converter para o formato **JSON** que o consumidor está esperando, isso no projeto **RS**.

Entretanto precisamos criar um vínculo desse projeto com o projeto canônico, uma vez que o xsd para estrutura de dados está contida no mesmo, para isso:

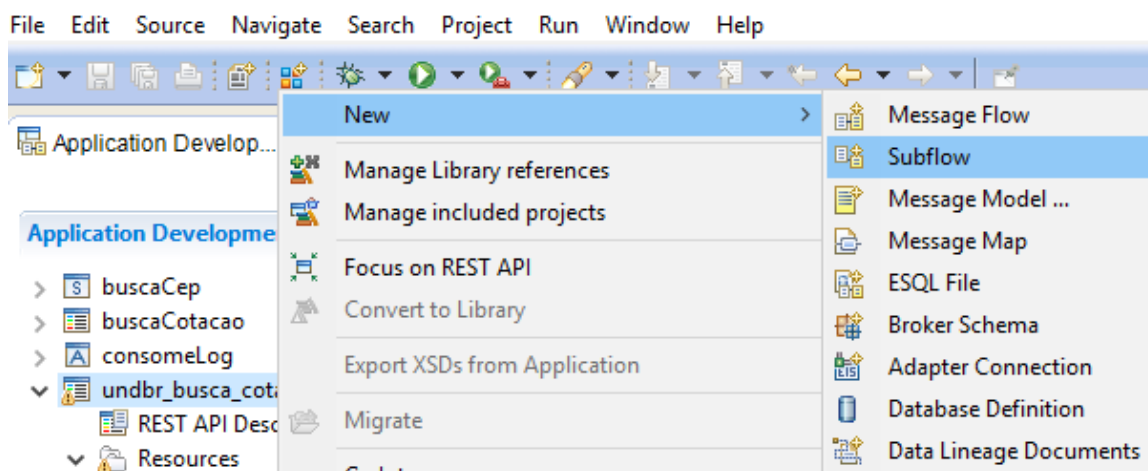
- 3.10.1. Selecione o projeto que receberá o **vínculo**.
- 3.10.2. Clique em **Properties**.
- 3.10.3. Selecione **Project References**.
- 3.10.4. Selecione o **projeto** que será **vinculado**.
- 3.10.5. Clique em **OK**.



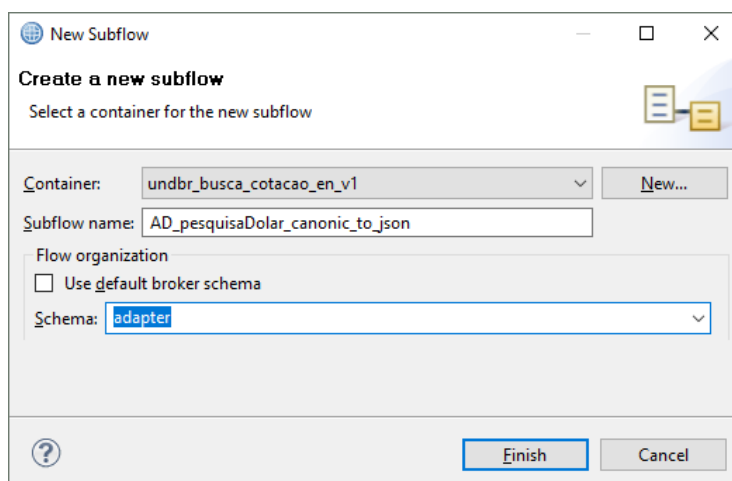
### 3.11 EN - Implementando a camada adapter – Canônico para Json.

Na sequência precisamos construir mais um adapter, que ira receber os dados no modelo canônico e transformar o para a estrutura de dados que o legado está esperando, para isso:

- 3.11.1. Selecione o **projeto**.
- 3.11.2. Clique em **New**.
- 3.11.3. Clique em **Subflow**.



- 3.11.4. Em Name digite: **AD\_pesquisaDolar\_canonic\_to\_json**.



- 3.11.5. Clique em **Finish**.



**Implementando o código ESQL**

- 3.11.6. Selecione o compute node e arraste para a área de trabalho.
- 3.11.7. Em description digite: ***pesquisaDolar\_canonic\_to\_json***.
- 3.11.8. Duplo clique o node para abrir o editor ***ESQL***.
- 3.11.9. Cole o código abaixo entre ***begin*** e ***end***.

```
SET OutputRoot          = InputRoot;
SET OutputLocalEnvironment = InputLocalEnvironment;

DECLARE nsDias NAMESPACE 'http://www.unimed.com.br/Legado/CotacaoInput';

-----
-- Endpoint do provedor que teremos que acessar
-- http://api.bcb.gov.br/dados/serie/bcdata.sgs.1/dados/ultimos/10?formato=json
-----
-- Declarando variaveis
-----

DECLARE uri_parte1 CHAR;
DECLARE dias      CHAR;
DECLARE uri_parte2 CHAR;
DECLARE endpoint  CHAR;

-----
-- Concatenando a URL para compor o endpoint
-----

SET uri_parte1 = 'http://api.bcb.gov.br/dados/serie/bcdata.sgs.1/dados/ultimos/';
SET dias      = InputRoot.XMLNSC.nsDias:Cotacao.qtdeDias;
SET uri_parte2 = '?formato=json';
SET endpoint   = uri_parte1 || dias || uri_parte2;

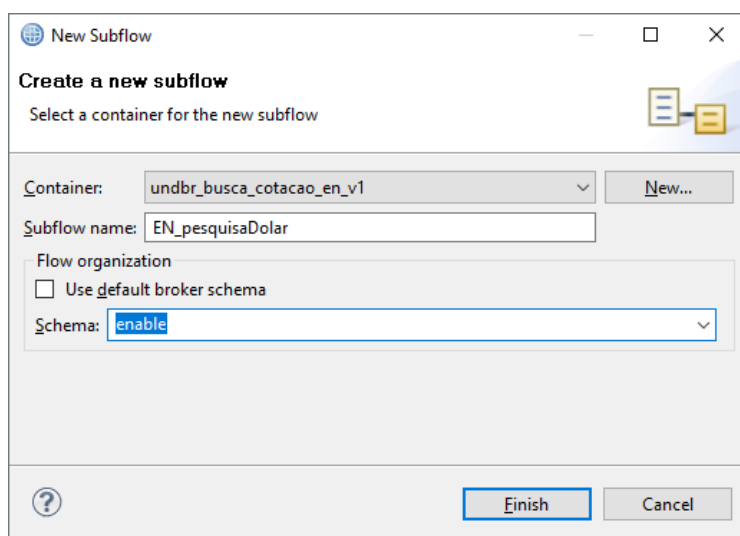
-----
-- Setando o endpoint no node de HTTP Request
-----

SET OutputLocalEnvironment.Destination.HTTP.RequestURL = endpoint;
RETURN TRUE;
```

## 3.12 EN - Implementando a camada de enable – parte II.

Na sequência precisamos construir mais um adapter, que irá receber os dados de retorno do legado e transformar para o modelo canônico, para isso:

- 3.12.1. Selecione o **projeto**.
- 3.12.2. Clique em **New**.
- 3.12.3. Clique em **Subflow**.
- 3.12.4. Em Subflow Name: **EN\_pesquisaDolar**.
- 3.12.5. Selecione o schema: **enable**.

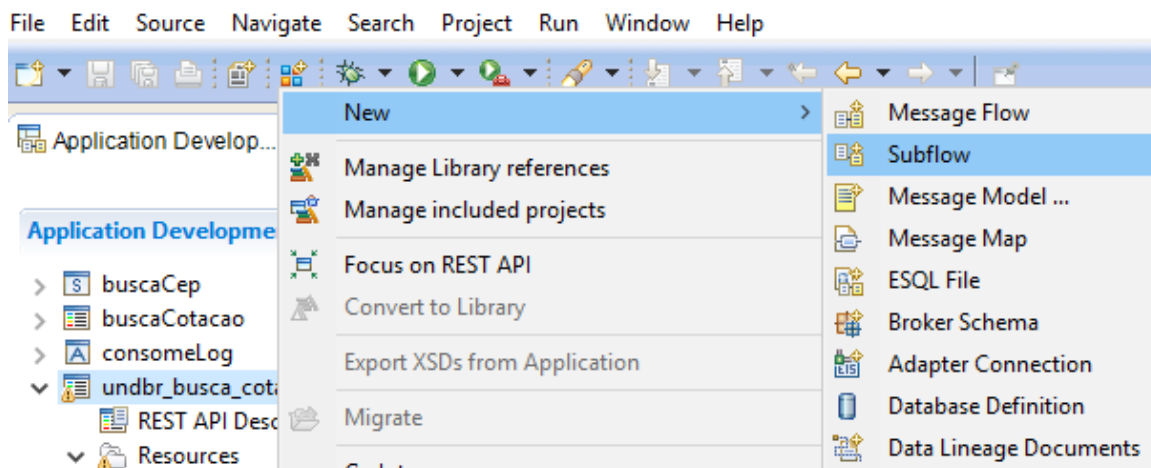


### 3.13 EN - Implementando a camada adapter – JSON para Canônico.

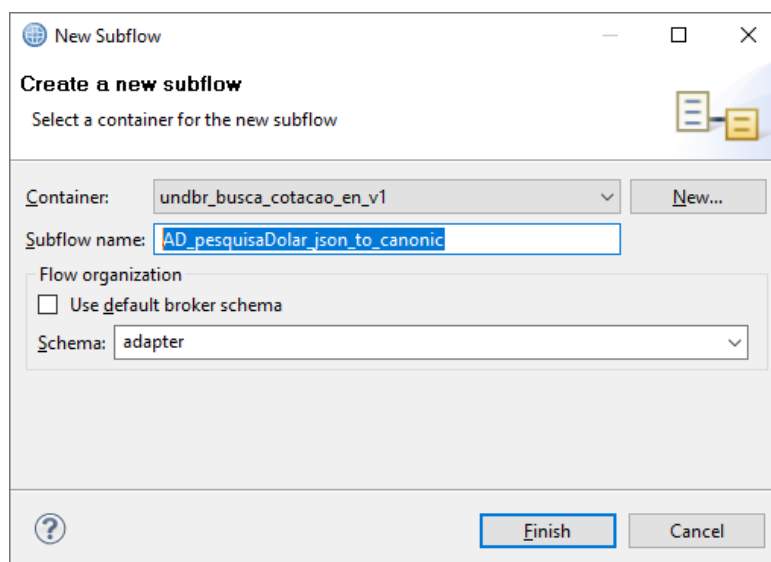
Temos agora que retornar os dados da requisição que o barramento realizou no legado, assim será necessário realizar o mapeamento de resposta, entenda que é a resposta do legado que devemos responder.

**Agora vamos construir nosso subflow de adaptação para a resposta.**

- 3.13.1. Selecione o projeto: ***undbr\_busca\_cotacao\_en\_v1***.
- 3.13.2. Clique em ***New***.
- 3.13.3. Clique em ***Subflow***.



- 3.13.4. Em Name digite: **AD\_pesquisaDolar\_json\_to\_canonic.**
- 3.13.5. *Certifique que o Container selecionado é o serviço em implementação.*
- 3.13.6. Deselecione a opção: **Use default broker schema.**
- 3.13.7. Em schema selecione o broker schema que criamos, nesse caso: **adapter.**



- 3.13.8. Clique em **Finish.**

Vamos implementar o código **ESQL.**

- 3.13.9. Selecione o **compute node.**
- 3.13.10. Duplo clique no node para abrir o editor de **ESQL.**



3.13.11. Insira o **ESQL** abaixo, entre as tag **begin** e a tag **end**.

```
DECLARE ns NAMESPACE 'http://www.unimed.com.br/Legado/CotacaoOutput';

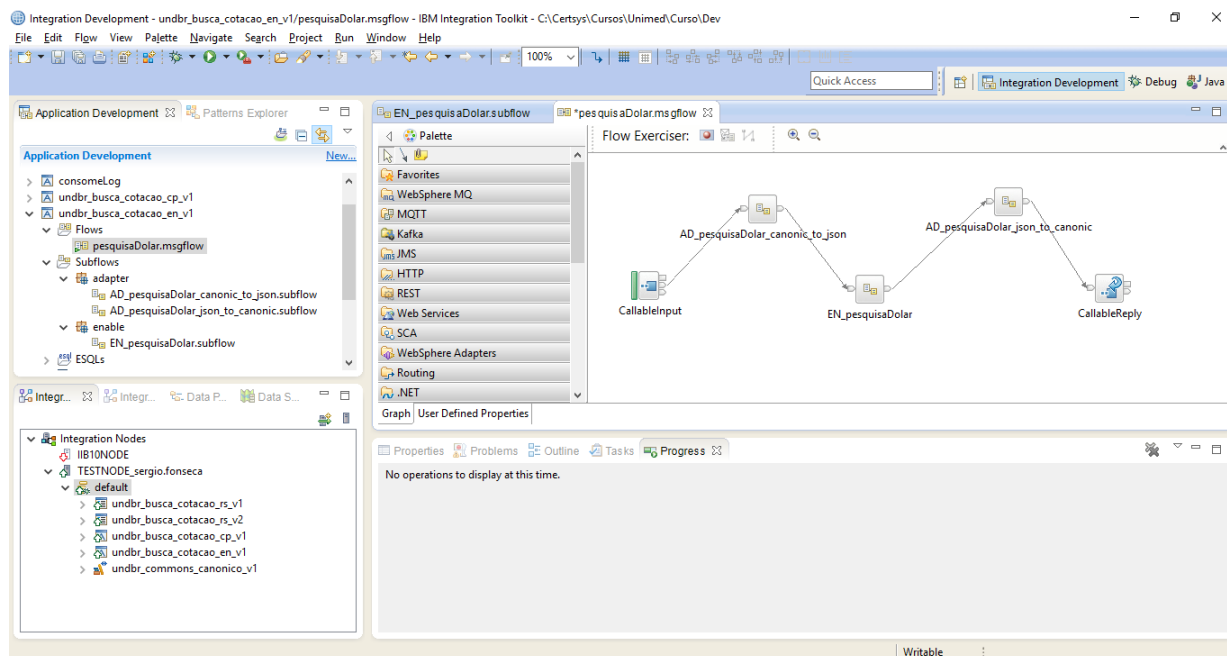
DECLARE I INTEGER CARDINALITY (InputRoot.JSON.Data.*[]);
DECLARE J INTEGER 1;

WHILE J <= I DO
    SET OutputRoot.XMLNSC.ns:CotacaoOutput.valores[J].data = InputRoot.JSON.Data.Item[J].data;
    SET OutputRoot.XMLNSC.ns:CotacaoOutput.valores[J].valor = InputRoot.JSON.Data.Item[J].valor;
    SET J = J + 1;
END WHILE;

RETURN TRUE;
```

3.13.12. Araste o Subflow do adapter para flow **pesquisaDolar**.

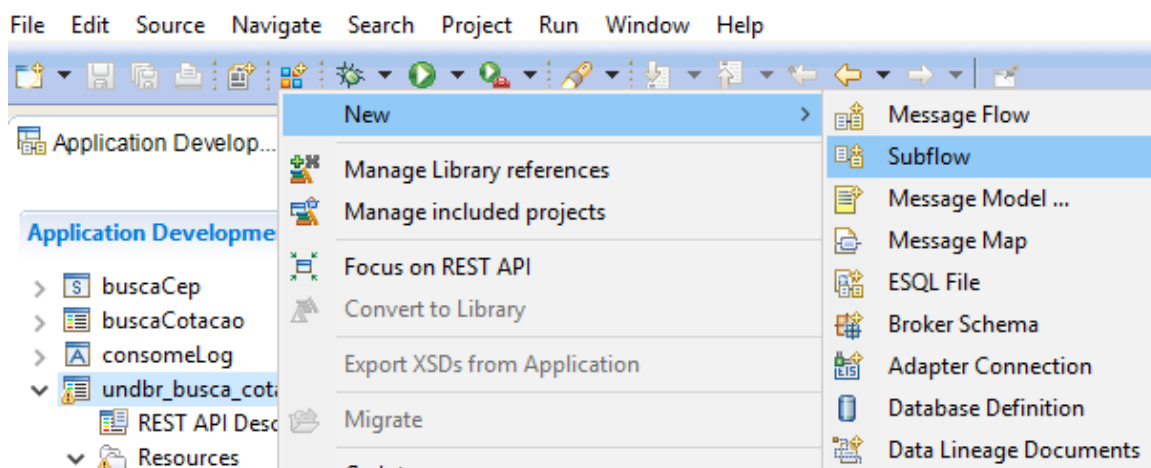
3.13.13. Faça as ligações entre os nodes.



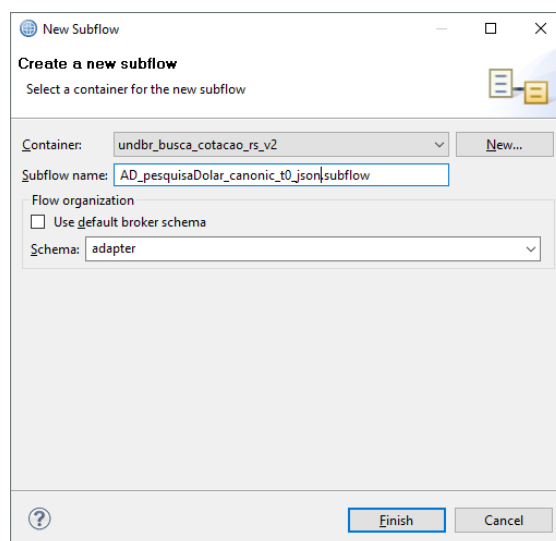
### 3.14RS - Implementando a camada adapter – Canônico para JSON.

Os dados já estão retornando, entretanto está no formato XML, precisamos agora capturar as informações e disponibilizar no formato JSON, para isso:

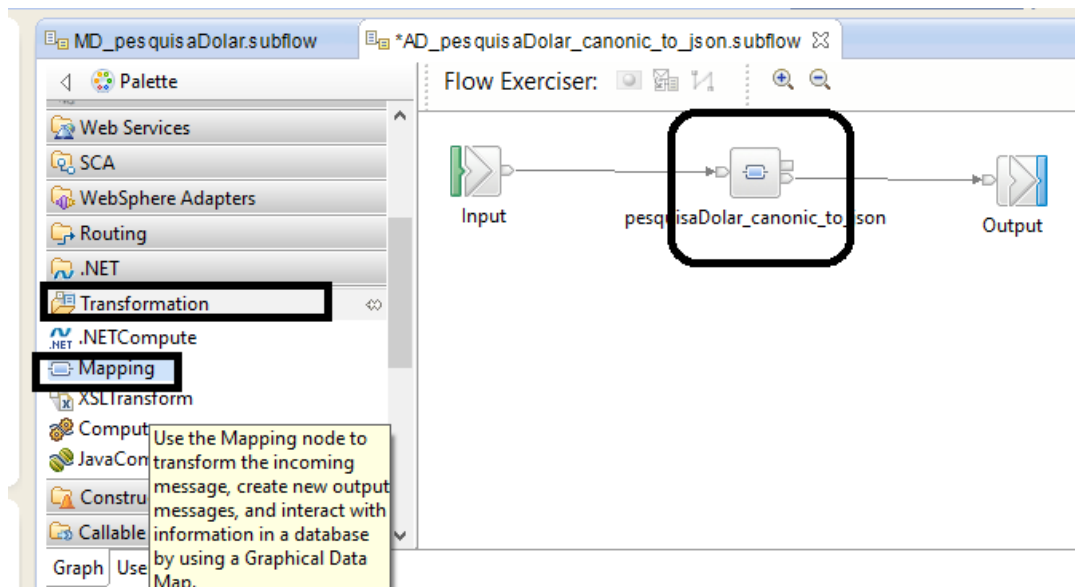
- 3.14.1. Selecione o projeto: **undbr\_busca\_cotacao\_en\_v1**.
- 3.14.2. Clique em **New**.
- 3.14.3. Clique em **Subflow**.



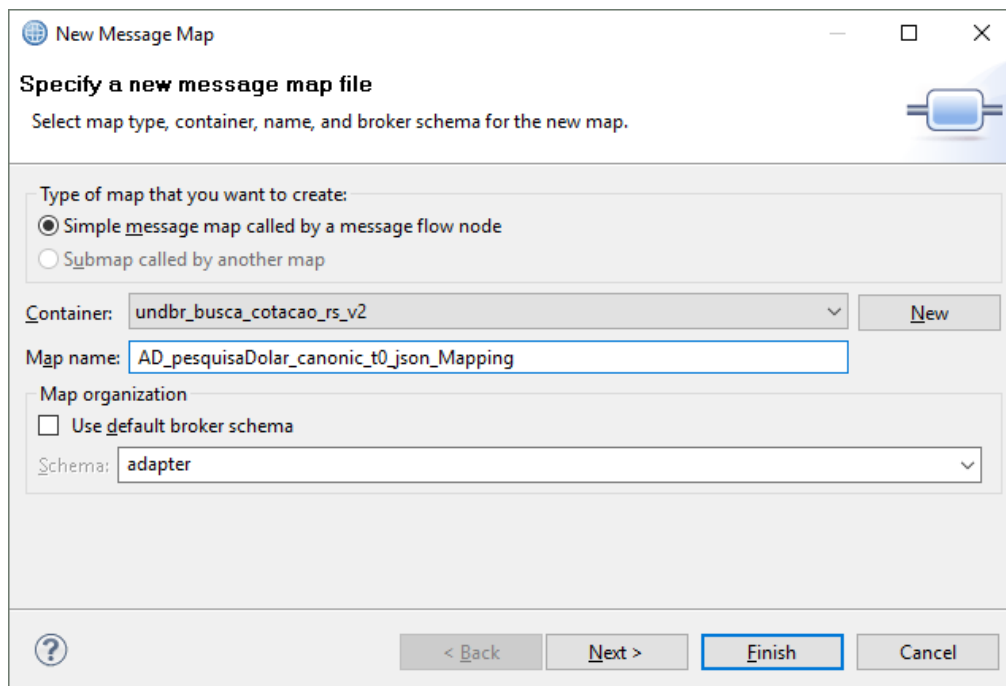
- 3.14.4. Em Name digite: **AD\_pesquisaDolar\_canonic\_to\_json**.
- 3.14.5. *Certifique que o Container selecionado é o serviço em implementação.*
- 3.14.6. Deselecione a opção: **Use default broker schema**.
- 3.14.7. Em schema selecione o broker schema que criamos, nesse caso: **adapter**.



- 3.14.8. No paleta de **notes**, clique **Transformation** e selecione **Mapping** e arraste para a área de trabalho.



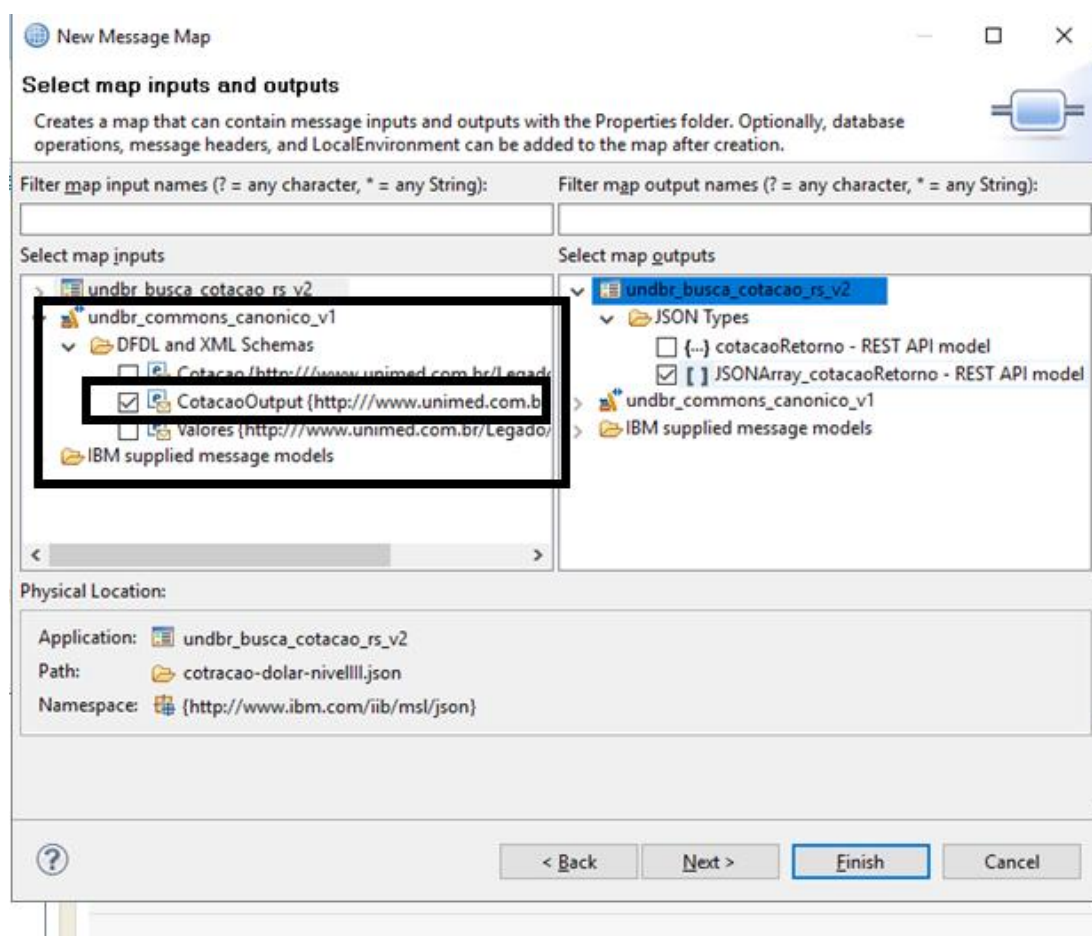
- 3.14.9. Execute duplo clique em **Mapping**, para abrir área de mapeamento de dados.



- 3.14.10. Clique em **Finish**.

Será apresentada a tela para selecionar os artefatos para os mapeamentos de dados, ao lado esquerdo são os dados de entrada de dados para o fluxo de mapeamento, como estamos recebendo conforme o modelo canônico, devemos selecionar o xsd **CotacaoOutput** contido no mesmo, para isso:

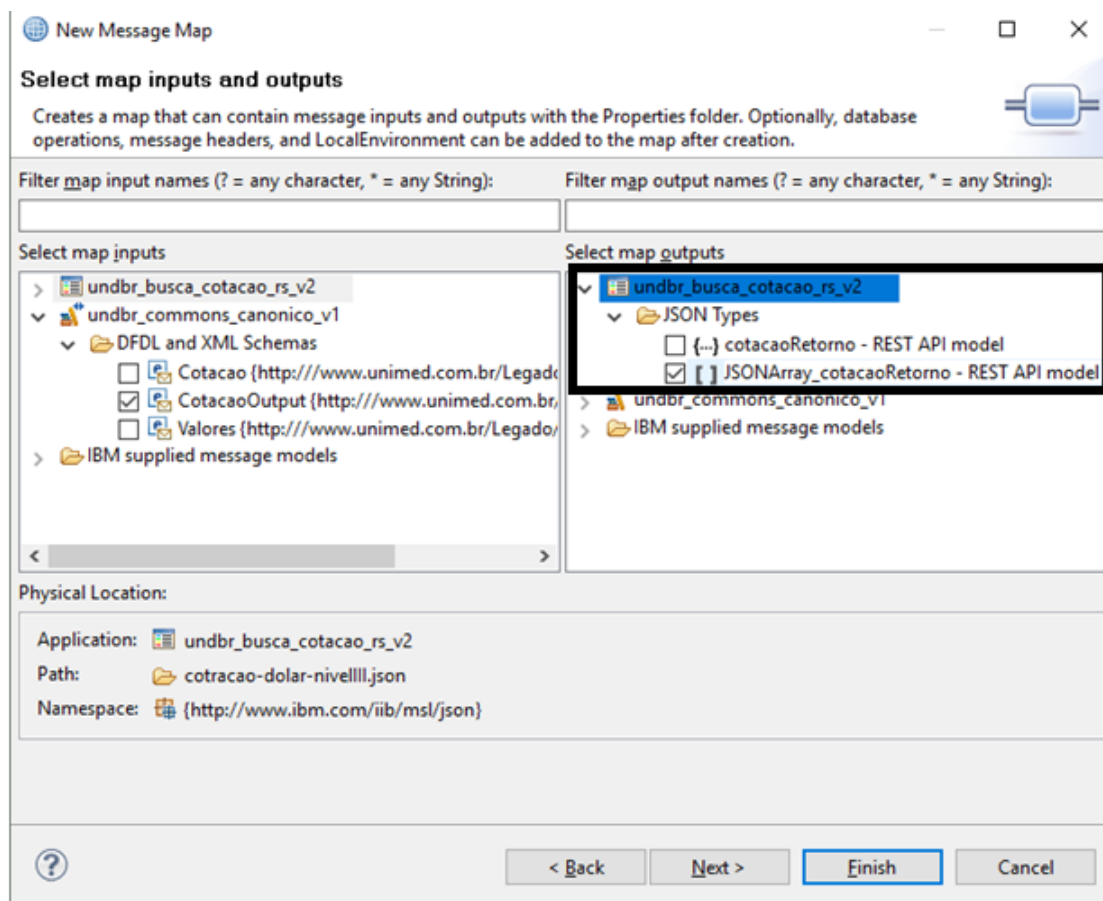
- 3.14.11. Expanda o projeto **undbr\_commons\_canonico\_v1**.
- 3.14.12. Selecione o artefato **CotacaoOutput**.



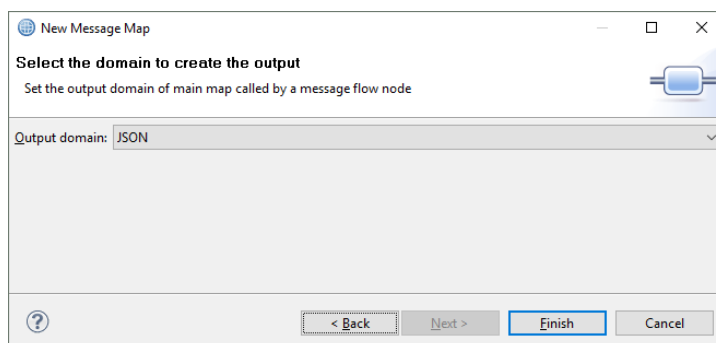


Ao lado direito é idem ao acima, porem são os artefatos para os dados de saída, devemos selecionar o artefato correspondente ao JSON que queremos devolver ao consumidor do barramento.

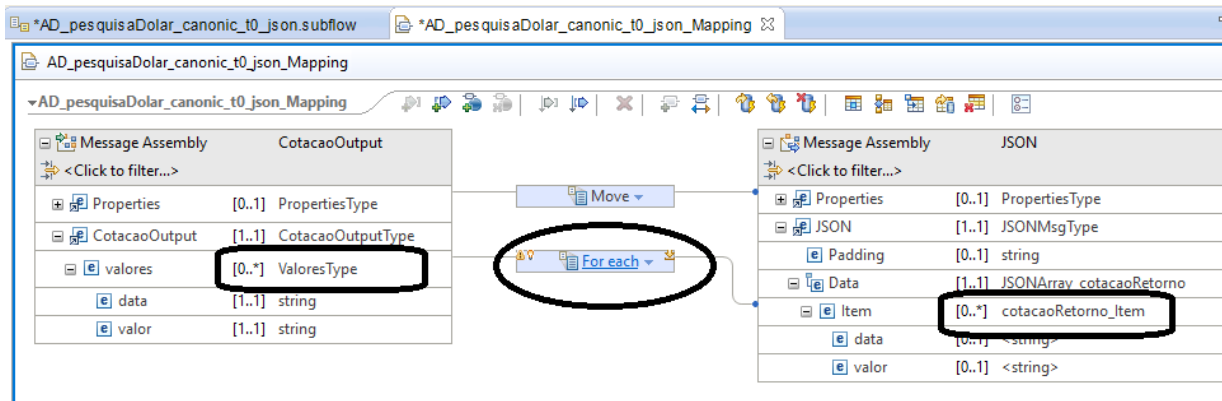
- 3.14.13. Expanda o projeto **undbr\_busca\_cotacao\_rs\_v2**.
- 3.14.14. Selecione o artefato **JSONArray\_cotacaoRetorno**.



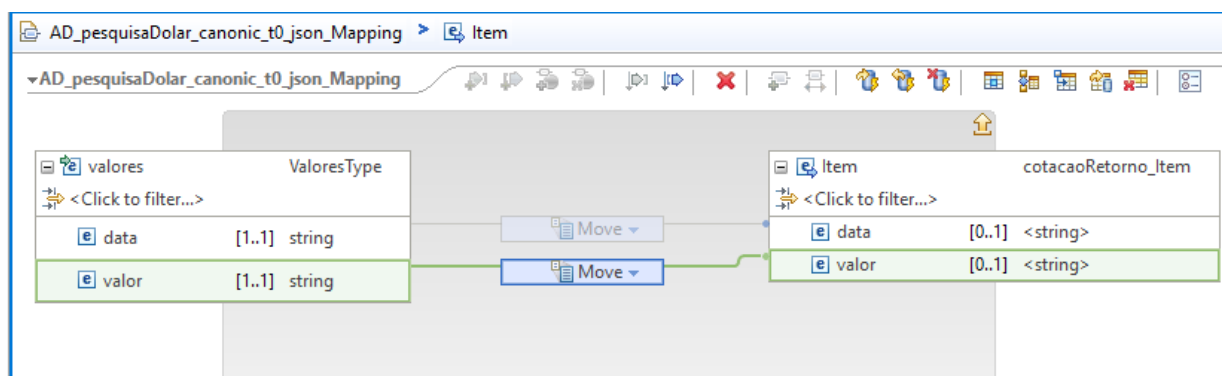
- 3.14.15. Clique em **Next**, para visualizar o formato de retorno.
- 3.14.16. Clique em **Finish**.



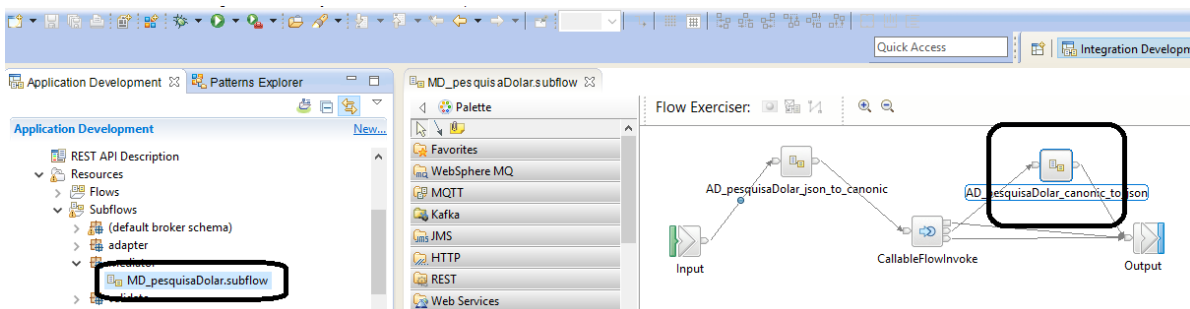
- 3.14.17. Clique em **Valores** do lado esquerdo e arraste para **Item** do lado direito.
- 3.14.18. Duplo clique em **For each**, localizado no centro da ligação.



- 3.14.19. Duplo clique em **For each**, localizado no centro da ligação.
- 3.14.20. Faça as ligações conforme abaixo.



- 3.14.21. Salve o projeto e saída da tela de mapeamento.
- 3.14.22. Selecione o subflow de mediator e insira o subflow de conversão do Canônico para o Json.

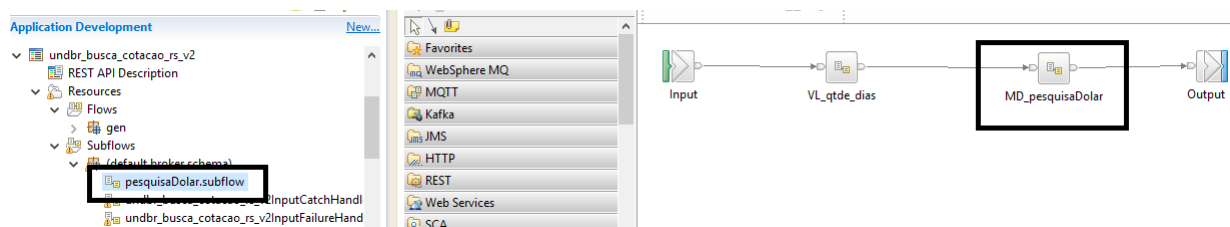


### 3.15 Ligações finais.

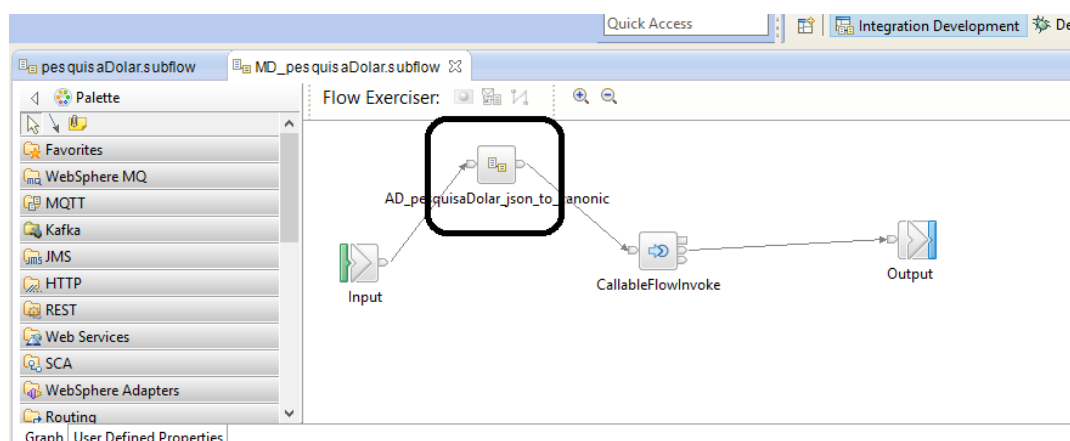
Concluimos a implementação até a chamada do serviço do legado, vamos revisar e realizar os ajustes finos, o que nos permitirá realizar um teste completo no serviço no sentido de realizar uma requisição.

Revisando:

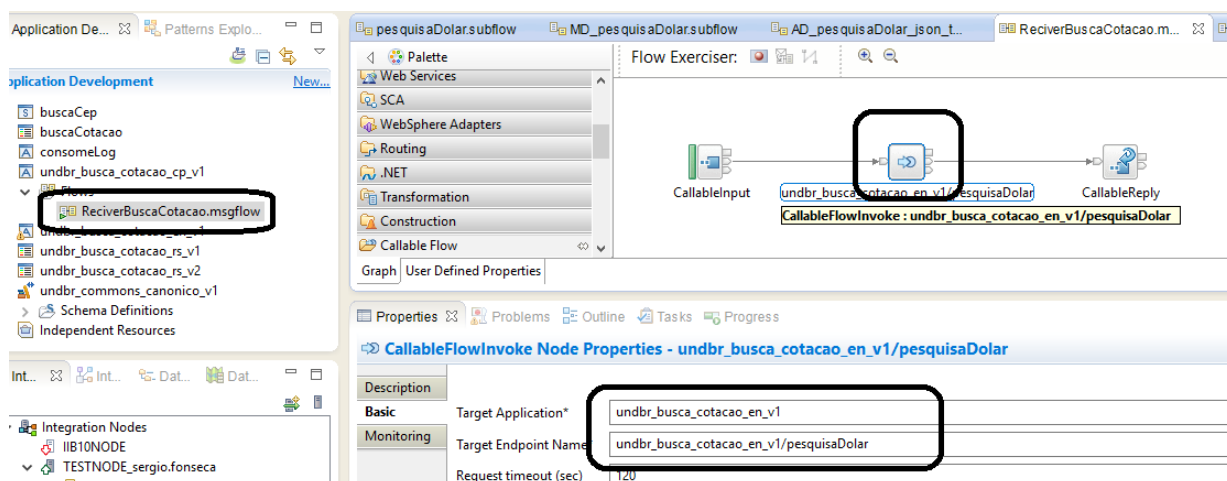
- 3.15.1. No projeto corporativo: **undbr\_busca\_cotacao\_rs\_v2**.
- 3.15.2. Selecione o subflow: **pesquisaDolar** e abra a área de trabalho.
- 3.15.3. Selecione o subflow: **MD\_pesquisaDolar** e arraste para a área de trabalho.
- 3.15.4. Realize as ligações entre os nodes.



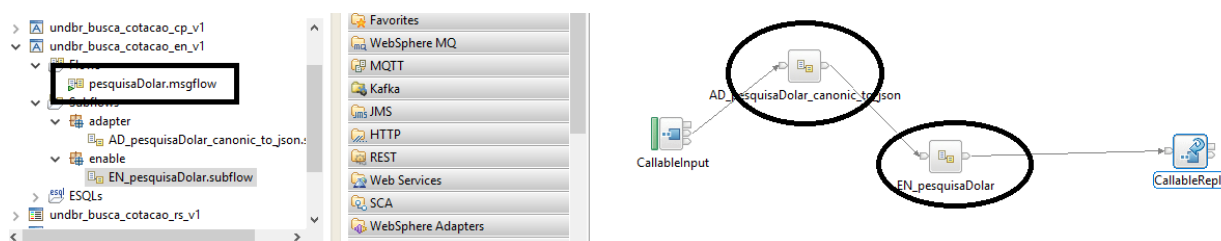
- 3.15.5. No projeto corporativo: **undbr\_busca\_cotacao\_rs\_v2**.
- 3.15.6. Selecione o subflow: **MD\_pesquisaDolar** e abra a área de trabalho.
- 3.15.7. Arraste e solte o Subflow **AD\_pesquisaDolar\_json\_to\_canonic**.
- 3.15.8. Realize as ligações entre os nodes.



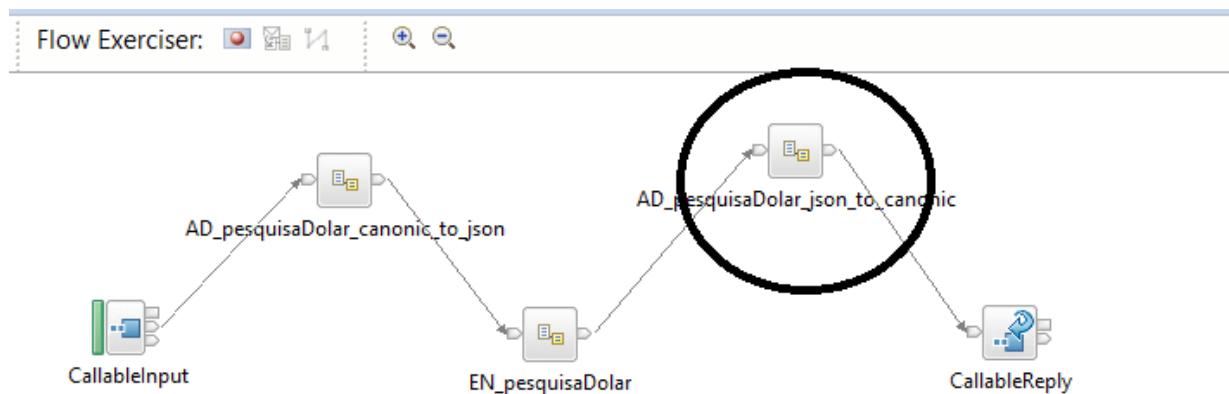
- 3.15.9. No projeto corporativo: **undbr\_busca\_cotacao\_cp\_v1**.
- 3.15.10. Selecione o flow: **reciverBuscaCotacao** e abra a área de trabalho.
- 3.15.11. Na paleta de nodes, clique em **Callable Flow**.
- 3.15.12. Arraste e solte na área de trabalho os node **CallableFlowInvoke** e preencha os seguintes dados:
  - 3.15.12.1. Target Application: **undbr\_busca\_cotacao\_en\_v1**.
  - 3.15.12.2. Target Endpoint Name: **undbr\_busca\_cotacao\_en\_v1/pesquisaDolar**.



- 3.15.13. No projeto corporativo: **undbr\_busca\_cotacao\_en\_v1**.
- 3.15.14. Selecione o flow: **pesquisaDolar** e abra a área de trabalho.
- 3.15.15. Arraste e solte o Subflow **AD\_pesquisaDolar\_canonic\_to\_json**.
- 3.15.16. Arraste e solte o Subflow **EN\_pesquisaDolar**.
- 3.15.17. Realize as ligações entre os nodes.



- 3.15.18. No projeto corporativo: **undbr\_busca\_cotacao\_en\_v1**.
- 3.15.19. Selecione o flow: **pesquisaDolar** e abra a área de trabalho.
- 3.15.20. Arraste e solte o Subflow **AD\_pesquisaDolar\_json\_to\_canonic**.
- 3.15.21. Arraste e solte o Subflow **EN\_pesquisaDolar**.



3.15.22. Realize as ligações entre os nodes.

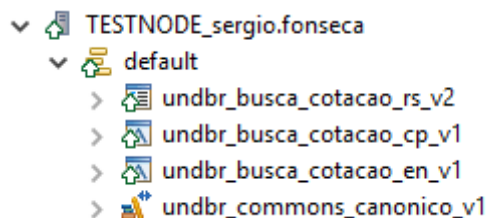
### 3.16 Realizando deploy.

Para realizarmos o deploy precisamos entender as dependências dos projetos, no nosso projeto, o fluxo de integração inicia-se no projeto **RS**, que invoca o projeto **CP** que por sua vez invoca o projeto **EN**.

Portanto o deploy do projeto será em ordem inversa a mencionada acima, porque para que ocorra o deploy precisa que as dependências já estejam disponíveis no servidor.

O raciocínio é o seguinte: primeira deploy dos projetos que não tem dependências normalmente os projetos comuns e depois os demais projetos, na seguinte ordem:

1. undbr\_commons\_canonico\_v1
2. undbr\_busca\_cotacao\_en\_v1
3. undbr\_busca\_cotacao\_cp\_v1
4. undbr\_busca\_cotacao\_rs\_v2

**Observação:**

Faça um teste para validar o fluxo, via **soapui**, mas antes habilite o modo de debug e veja a entrada dos dados e a passagem dos mesmo entre os fluxos.