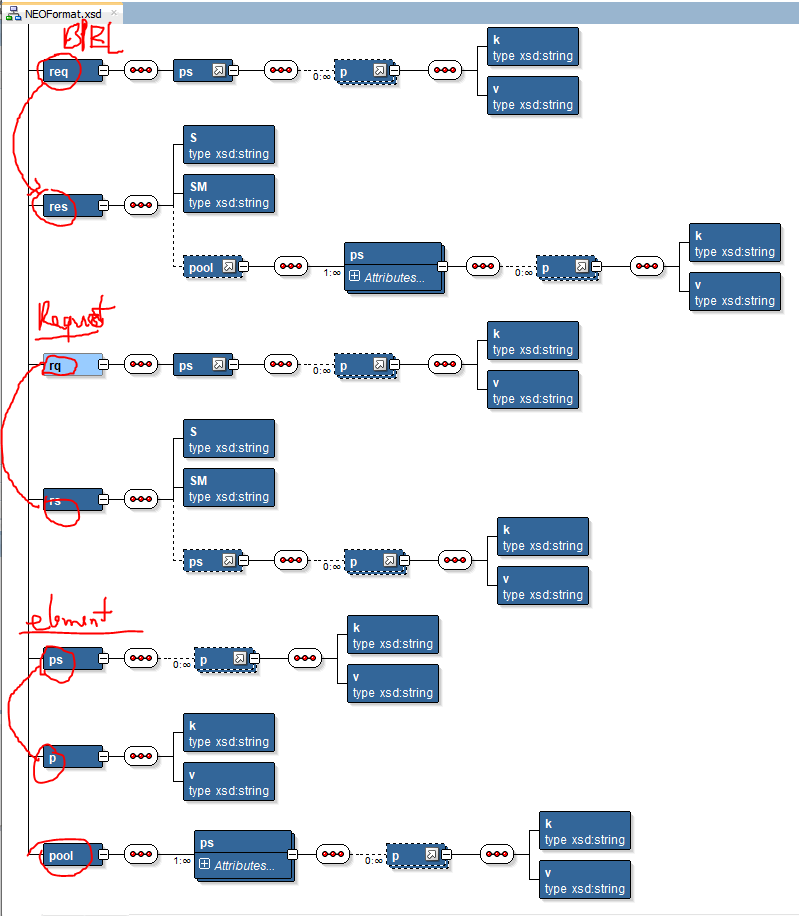
**Phân tích các yếu tố trong kịch flow xử lý SMS**

# **Những bước tạo: FlowSMS0608**

Bước 1: Create SOA Application, SOA Project

Bước 2: Thiết kế ra ParamSMS:



Param: gồm kiểu key-value

Params: tập hợp từ các Param

Pool: là tập hợp của nhiều params -> nơi chứa một loạt các biến đó

Request: chứa một tập các params

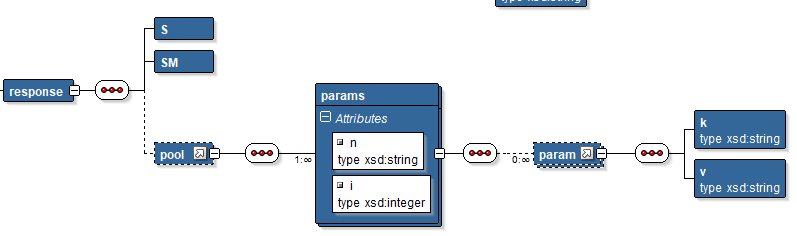
Response: Chứa thông tin phản hồi của BPEL process.

Trong đó có các thông số: S là status -> trạng thái trả về

SM là tin nhắn trạng thái trả về -> có thể là success hoặc fail

# **Thành phần bên trong của response:**

Pool là tập các thông tin chứa về params -> chứa các param -> theo kiểu key – value



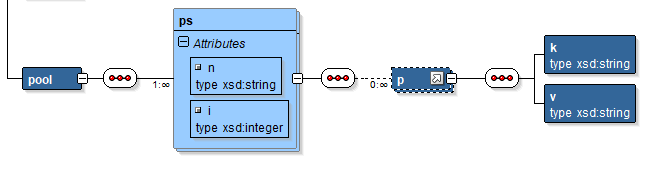
Trong đó Attributes của pool có n là name của cái gì đó, I là index của cái gì đó.

Những thành phần bên trong của một flow BPEL Process:

|  |  |
| --- | --- |
| **Biểu đồ** | **Mô tả** |
|  | Thành phần **InitParam**: -> khởi tạo giá trị param  Gồm có các thành phần: FlowInitParam, FlowUniqueParam, getInstanceName  SMS:  SMS2:  **mapResponse**: -> Map thành công  gồm các thành phần: mapSucess, FlowResponse |
|  | Khởi tạo giá trị:  **FlowInitParam** -> FlowInitParam.xsl -> map những giá trị đầu vào -> cho cái gì đó  **FlowUniqueParam** -> Đồng nhất giá trị -> cho flow sử dụng  **getInstanceName** -> lấy thông tin của instance  những mã code java vào chương trình |
|  | **mapSuccess**: assign map giá trị sao đúng  Flow Response: transaction  FlowResponse.xsl ->  Từ cái pool params -> for each  Tham số n -> attribute n  Tham số i -> attribute i  Những tham số tieeps theo map cho việc for each |

Phân biệt các thành phần URI có hay không trong tài liệu:

# **Phân tích Pool param trong Schema:**



Chứa tập giá trị key – value

Pool là tập chứa cảu các parameter

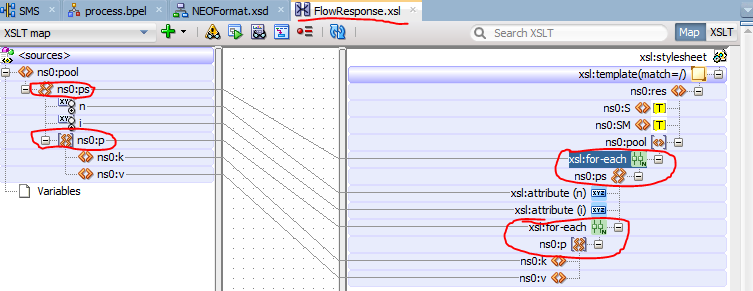
Đầu vào của URL chính là rq: request -> ứng với các giá trị trong Schema đã có

Đầu ra của URL chính là rs: response -> ứng với giá trị trong Schema đã có

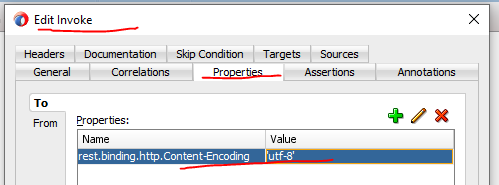
Transaction: map với hoặc là map trong với cái gì đó

Ps sẽ được map bên trong của for-each: vì trong for-each chứa một tập ps

P sẽ được map bên trong của for-each: vì bên trong for-each chứa một tập của p:

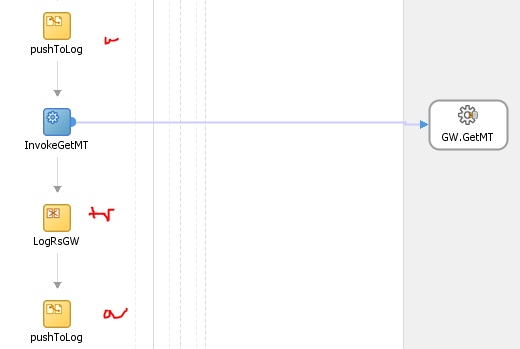


Thuộc tính cho việc Invoke thành công:



# Giá trị trả về của invoke REST:

Mục tiêu chính: Nhận, và chuyển đổi giá trị của một Invoke trả về



Assign + invoke + transformation + assign

Trong LogRsGW làm nhiệm vụ nhận kết quả từ đầu ra InvokeGetMT -> sử dụng một poolparamtemp -> nhận giá trị:

input: InvokeGetMT\_execute\_InputVariable

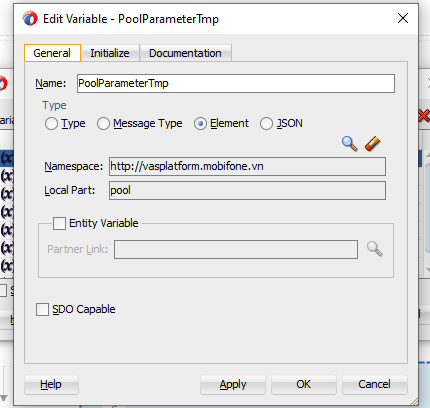
output: InvokeGetMT\_execute\_OutputVariable

Với transformation: LogRsGW -> log respond Gateway -> giá trị phản hồi từ gateway. Sử dụng **PoolParameterTmp** để lưu các phản hồi từ sự trả về -> có cấu trúc tương tự như một **poolprameter** thường.

Chú ý:

# Hướng dẫn tạo mới một PoolparameterTemp

Mục đích chính:

Xây dựng nhanh một pool từ một trong những cái có sẵn. 

Chọn vào phần quản lý giá trị biến của khu vực đó -> Những type có thể là:

Type:

**Message Type**: kiểu … chưa rõ công dụng

**Element**: kiểu trả về như một trong những schema đã tạo trước đó. Giống như pool, req, res…

**JSON**: kiểm giá trị key value

Tùy vào những giá trị đầu vào mà giá trị đầu ra của XSLT transformation tương ứng đến nó.

**Namespace**: miền được đặt

**Local Part**: phần biến trong file schema đó.

**Entity Variable**: có thể có hoặc không đều được.

**SDO Capache**: chưa biết là thuộc tính nào.

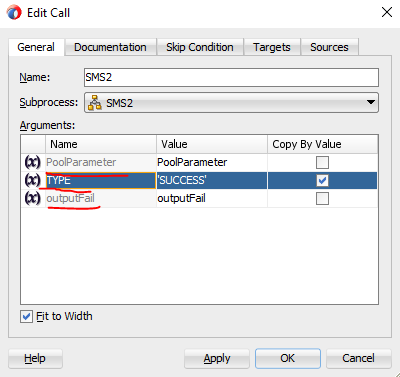
Chú ý:

Phân tích từ phần template

Thông tin phần dưới:

PoolParameterTmp -> biến temp ->

Những giá trị trong phần call SMS2:



Những agruments: PoolParameter, TYPE, outputFail

select="/ns0:pool/ns0:ps[@i = 1]/ns0:p[ns0:k = 'MTKEY']/ns0:v

Lấy từ key pool -> chuyển sang params -> chuyển tiếp sang param -> lấy giá trị value từ key này.