ワークフローの作成チュートリアル

ここでは、次の内容について説明します。

ワークフローの作成：チュートリアル

この章では、`operation` および `switch` 状態を使用する簡単な例に基づいてワークフローを構築し、いくつかのシミュレートされたデバイス用の VPN サービスを Cisco NSO で作成する方法について説明します。サンプルワークフロー定義を部分的に説明し、元のワークフローを作成する際にさまざまな定義コンポーネントを使用する方法を示します。

ワークフローの定義方法に関する完全な情報が必要な場合は、「Serverless Workflow 仕様」を参照してください。

サンプルワークフローの概要

ワークフローは、JSON または YAML で記述できます。この例では、JSON 形式を選択します。

このサンプルワークフローの目的は、Cisco NSO デバイス用の VPN サービスを自動的に作成することです。

まず、データ入力でデバイスをポイントし、ポイントしたデバイスに対して NSO `check-sync` 操作の実行を試みます。結果に応じて以下の手順を実行します。

- 同期していない場合は、デバイスをプッシュして `sync-from` を実行してから、そのデバイスの VPN の作成を試みます。
- 同期している場合は、`sync-from` は実行せず、デバイスの VPN を直接作成します。

すべてのステップが正常に実行されると、CWM はワークフロー実行の完了を報告し、最終データ入力を表示します。結果は Cisco NSO でも確認できます。ステップの実行中にエンジンでエラーが発生すると、指定された `retry` ポリシーが使用されます。再試行の制限を超えてエラーが続く場合、実行エンジンは実行を [失敗（Failed）] ステータスで終了します。

データ入力、関数、状態、アクション、およびデータフィルタの定義方法の詳細については、以下のセクションを参照してください。

データ入力の提供

ワークフロー定義には、通常、JSON ファイルの先頭に複数の入力データが含まれています。提供されたデータはワークフローの一部ではありませんが、ワークフロー定義内で参照され、該当するデータ更新が定義されている場合は、状態間でも更新できます。詳細については、Serverless Workflow 仕様の「Workflow data input」を参照してください。

クイックスタートの例では、2 つのユーザー定義の `deviceName` JSON オブジェクトキーと値（ローカル NSO インスタンスのテストデバイスの名前）、およびワークフローで使用される CWM リソースを指定する `nsoResource` キーのみを指定する必要があります。したがって、JSON のワークフローデータ入力は次のようになります。

```
{
"device0Name": "ce0",
"device1Name": "ce1",
"nsoResource": "NSOLocal"
}
```

トップレベルのパラメータと関数

ワークフロー定義は、必須のワークフロー `id` キーで始まります。他のキーでは、`specVersion` も必要です。このキーは、Serverless Workflow 仕様のリリースバージョンを定義します。`start` キーでは、ワークフローの開始状態の名前を定義しますが、必須ではありません。

`functions` キーでは、Cisco NSO アダプタアクティビティ名を関数 `name` として渡し、アダプタアクティビティ ID を関数 `operation` として渡し、`metadata: worker` キーの下にワーカー名を指定します。

```
{
    "id": "CreateL3VPN",
    "name": "Create Layer3 VPN",
    "start": "start",
    "version": "1.0",
    "functions": [
    {
      "name": "NSO.RestconfPost",
      "metadata": {
        "worker": "cisco.nso.v1.0.1"
      },
      "operation": "cisco.nso.v1.0.1.restconf.Post"
    }
  ],
  "description": "Create an L3 VPN for MPLS devices",
  "specVersion": "0.9"
}
```

（注） 実際には、`functions` では、Cisco NSO アダプタで定義され、`main.go` ファイルに表示されるアクティビティの ID をワークフローに提供します。また、`metadata` では、定義された関数を参照するアクションを実行するワーカーの名前を指定します。

再試行ポリシーの指定

retries キーを使用して、アクションが失敗した場合の状態アクションの再試行ポリシーを定義します。このキーでは複数の再試行ポリシーを指定でき、複数の定義済み状態アクションで再利用できます。

```
"retries": [
    {
      "name": "Default",
      "maxAttempts": 4,
      "delay": "PT5S",
      "multiplier": 2
    },
    {
      "name": "Custom",
      "maxAttempts": 2,
      "delay": "PT30S",
      "multiplier": 1
    }
  ],
```

（注） 表示されているように、Default ポリシーは、失敗したアクションが最大 4 回再試行されることを前提としています。連続した再試行の場合、試行の間の遅延秒数は5、10、20、40秒と増加します。

状態の定義

ワークフロー状態は、ワークフロー定義の構成要素です。このクイックスタートの例では、operation と switch の状態を使用しますが、他の状態も使用できます。詳細については、Serverless 仕様の「Workflow states」セクションを参照してください。

operation 状態

```
"states": [
        {
        "name": "start",
        "type": "operation",
        "stateDataFilter": {
            "input": "${ . }"
        },
        "actions": [],
        "transition": {
            "nextState": "syncFromOrCreateVPN"
            }
        }
    ]
```

operation 状態の内部では、状態の name と type とは別に以下を定義します。

- stateDataFilter：サンプル JSON ファイルの先頭に定義されているデータ入力を指します。input パラメータでは、${ . } と記述します。これは、ワークフロー実行時点で存在するデータ入力全体を使用することを意味する jq 式です。

（注） ワークフローでの jq 式の使用方法の詳細については、Serverless Workflow 仕様の「Workflow expressions 」の章を参照してください。

- `actions`：アクションで使用する関数と、2 つの基本 `arguments`：`input` と `config` を指定します。詳細については、以下のサブセクションを参照してください。
- `transition` または `end`：現在の状態を実行後にワークフローが遷移する次の状態を指します。追加で実行する手順がない場合は、`end` を使用します。

switch 状態

```
    {
      "name": "syncFromOrCreateVPN",
      "type": "switch",
      "dataConditions": [
         {
           "name": "shouldSyncFrom",
          "condition": "${ if (.checkSyncResult0) then  .checkSyncResult0 != \"in-sync\"
else null end }",
            "transition": {
              "nextState": "syncFrom"
            }
          },
          {
            "name": "shouldCreateVPN",
           "condition": "${ if (.checkSyncResult0) then  .checkSyncResult0 == \"in-sync\"
else null end }",
            "transition": {
              "nextState": "createVPN"
            }
         }
       ]
    }
```

`switch` 状態の内部では、状態の `name` と `type` とは別に以下を定義します。

- `dataConditions`：指定された次の状態に遷移するデバイスが満たす条件を定義します。`switch` 状態は、ステータスに基づいてデバイスを適切な状態に誘導するワークフローの「ゲートウェイ」として表示できます。`condition` パラメータで jq 式 `${ if (.checkSyncResult0) then .checkSyncResult0 == \"in-sync\" else null end }` を使用して、ブール値を作成します。ブール値が `true` の場合、デバイスを `CreateVPN` 状態に直接遷移するために使用されます。

アクションの指定

デバイス `ce0` の `operation` 状態の `checkSync` アクションに基づいて `actions` を分析します。

```
{
"name": "checkSync",
"retryRef": "Default",
"functionRef": {
```

```
  "refName": "NSO.RestconfPost",
  "arguments": {
    "input": {
     "path": "restconf/operations/tailf-ncs:devices/device=${ .device0Name }/check-sync"

      },
    "config": {
      "resourceId": "${ .nsoResource }"
      }
    }
  },
"actionDataFilter": {
  "results": "${ if (.data) then .data | fromjson.\"tailf-ncs:output\".result else null
 end }",
  "toStateData": "${ .checkSyncResult0 }"
  }
}
```

使用可能なパラメータの中でも、次の 2 つは特に役立ちます。

- `functionRef`：アクションの実行で使用される関数（NSO アダプタの観点からはアクティビティ）を参照します。ここでは、いくつかの引数を渡す必要があります。
  - `input`：
    - `path`：要求を送信するアダプタのパスを指します。
    - `data`：要求に含めるデータを転送します（`checkSync` アクションには適用されません）。
  - `config`：
    - `resourceId`：外部サービス用に作成したリソースの ID を指定します。サンプルワークフローでは、Cisco NSO インスタンスのローカルホストとデフォルトポートが指定されています。リソースは、外部サービスの認証データを提供するために使用されるシークレット ID も指します。この場合は、`username` と `password` を Cisco NSO インスタンスに提供します。
- `actionDataFilter`：NSO からの `checkSync` 応答で渡されるデータの処理方法を定義します。
  - `results`：jq 式 `"${ if (.data) then .data | fromjson.\"tailf-ncs:output\".result else null end }"` を使用して、着信 NSO データを処理します。`fromjson` を使用して、結果の `tailf-ncs:output` を JSON 形式に変換し、`.result` を使用して `result` のキー値をチェリーピックします。この場合（デバイスが in-sync 状態の場合）、式の出力は「`in-sync`」になります。
  - `toStateData`：前述の `results` パラメータで定義された式の出力を取得し、選択した名前（この場合は `.checkSyncResult0`）でワークフロー入力データ内にキーと値のペアとして保存します。

サンプルワークフローの定義

次のサンプルワークフローの定義は、この章で説明するワークフロー作成プロセスの最終結果です。

```
{
  "id": "CreateL3VPN-1.0",
  "name": "CreateL3VPN",
  "start": "start",
  "states": [
    {
      "name": "start",
      "type": "operation",
      "actions": [
        {
          "name": "checkSync",
          "retryRef": "Default",
          "functionRef": {
            "refName": "NSO.RestconfPost",
            "arguments": {
              "input": {
                "path": "restconf/operations/tailf-ncs:devices/device=${ .device0Name
}/check-sync"
              },
              "config": {
                "resourceId": "${ .nsoResource }"
              }
            }
          },
          "actionDataFilter": {
            "results": "${ if (.data) then .data | .\"tailf-ncs:output\".result else
null end }",
            "toStateData": "${ .checkSyncResult0 }"
          }
        },
        {
          "name": "checkSync",
          "retryRef": "Default",
          "functionRef": {
            "refName": "NSO.RestconfPost",
            "arguments": {
              "input": {
                "path": "restconf/operations/tailf-ncs:devices/device=${ .device1Name
}/check-sync"
              },
              "config": {
                "resourceId": "${ .nsoResource }"
              }
            }
          },
          "actionDataFilter": {
            "results": "${ if (.data) then .data | .\"tailf-ncs:output\".result else
null end }",
            "toStateData": "${ .checkSyncResult1 }"
          }
        }
      ],
      "transition": {
        "nextState": "syncFromOrCreateVPN"
      },
      "stateDataFilter": {
        "input": "${ . }"
      }
```

```
    },
    {
      "name": "syncFromOrCreateVPN",
      "type": "switch",
      "dataConditions": [
        {
          "name": "shouldSyncFrom",
         "condition": "${ if (.checkSyncResult0) then  .checkSyncResult0 != \"in-sync\"
 else null end }",
          "transition": {
            "nextState": "syncFrom"
          }
        },
        {
          "name": "shouldCreateVPN",
         "condition": "${ if (.checkSyncResult0) then  .checkSyncResult0 == \"in-sync\"
 else null end }",
          "transition": {
            "nextState": "createVPN"
          }
        },
        {
          "name": "shouldSyncFrom",
         "condition": "${ if (.checkSyncResult1) then  .checkSyncResult1 != \"in-sync\"
 else null end }",
          "transition": {
            "nextState": "syncFrom"
          }
        },
        {
          "name": "shouldCreateVPN",
         "condition": "${ if (.checkSyncResult1) then  .checkSyncResult1 == \"in-sync\"
 else null end }",
          "transition": {
            "nextState": "createVPN"
          }
        }
      ],
      "defaultCondition": {
        "end": {
          "terminate": true
        }
      }
    },
    {
      "name": "syncFrom",
      "type": "operation",
      "actions": [
        {
          "name": "syncFrom",
          "retryRef": "Default",
          "functionRef": {
            "refName": "NSO.RestconfPost",
            "arguments": {
              "input": {
                "path": "restconf/operations/tailf-ncs:devices/device=${ .device0Name
}/sync-from"
              },
              "config": {
                "resourceId": "${ .nsoResource }"
              }
            }
          },
          "actionDataFilter": {
```

```
            "results": "${ if (.data) then .data | .\"tailf-ncs:output\".result else
null end }",
            "toStateData": "${ .syncFromResult0 }"
          }
        },
        {
          "name": "syncFrom",
          "retryRef": "Default",
          "functionRef": {
            "refName": "NSO.RestconfPost",
            "arguments": {
              "input": {
                "path": "restconf/operations/tailf-ncs:devices/device=${ .device1Name
}/sync-from"
              },
              "config": {
                "resourceId": "${ .nsoResource }"
              }
            }
          },
          "actionDataFilter": {
            "results": "${ if (.data) then .data | .\"tailf-ncs:output\".result else
null end }",
            "toStateData": "${ .syncFromResult1 }"
          }
        }
      ],
      "transition": {
        "nextState": "createVPN"
      }
    },
    {
      "end": {
        "terminate": true
      },
      "name": "createVPN",
      "type": "operation",
      "actions": [
        {
          "name": "createVPN",
          "retryRef": "Custom",
          "functionRef": {
            "refName": "NSO.RestconfPost",
            "arguments": {
              "input": {
                "data":
'[illegible]
 ",
                "path": "restconf/data/l3vpn:vpn"
              },
              "config": {
                "resourceId": "${ .nsoResource }"
              }
            }
          },
          "actionDataFilter": {
            "results": "${ if (.status) then .status else null end }",
            "toStateData": "${ .createServiceResult }"
          }
        }
      ]
    }
  ],
  "retries": [
```

```
    {
      "name": "Default",
      "delay": "PT30S",
      "multiplier": 2,
      "maxAttempts": 4
    },
    {
      "name": "Custom",
      "delay": "PT10S",
      "multiplier": 1,
      "maxAttempts": 2
    }
  ],
  "version": "1.0",
  "functions": [
    {
      "name": "NSO.RestconfPost",
      "metadata": {
        "worker": "cisco.nso.v1.0.1"
      },
      "operation": "cisco.nso.v1.0.1.restconf.Post"
    }
  ],
  "description": "",
  "specVersion": "0.9"
}
```

CWM および Cisco NSO で具体的な結果を得るためのサンプルワークフローの実行方法に関する完全な手順については、スタートアップガイド [英語] を参照してください。

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。