TOTVS Connector

Last updated by | Felipe Cesar Sepulvida Rampazzo | 14 de jan. de 2022 at 17:53 BRT

Contents

O que é o TOTVS Connector?

O TOTVS Connector é uma ferramenta que permite a integração entre softwares, não importando a forma de distribuição do software, ou seja, integra dados entre aplicações Cloud e aplicações On Premise e vice-versa.

Como assim?

Por meio de dois componentes dispostos em ambientes diferentes, é possível uma aplicação On Premise ficar sincronizada com um aplicação em Cloud, realizando pouca codificação.
Um desses componentes é o **TOTVS Connector Client**, ele é instalado no ambiente On Premise do cliente.
O outro componente é o **TOTVS Connector Server**, ele é gerenciado pela TOTVS e é o responsável de receber todo fluxo de dados.

Arquitetura

Aqui será demonstrada a disposição e requisitos dos componentes do TOTVS Connector.

TOTVS Connector Client

É a parte responsável por ler os dados das aplicações On Premises.
Ele é instalado em uma máquina que possa se conectar com o banco de dados do produto On Premise/Private Cloud.
O TOTVS Connector Client precisa de uma instância do PostgreSQL e, dependendo da configuração standalone, uma instância do RabbitMQ.
Outro requisito é o acesso via *http* ao TOTVS Connector Server (se houver alguma bloqueio na rede, deverá ser liberado).

Requisitos mínimos

- Sugestão do sistema operacional: Debian ou CentOS GNU/Linux 10 (buster) ou distribuição Linux 64 bits;
- Memória RAM: 4GB;
- Processador: Quad Core 1.8 GHZ ou superior;
- Espaço em Disco: Espaço em disco de 40GB;
- Ferramentas/Recursos principais: Docker (containers Linux) e Portainer (opcional, para monitoramento/gestão de ambientes baseados em Docker/containers);
- Requisitos de ambiente/infraestrutura:
  - Comunicação entre os ambientes do TOTVS Agro Connector e bases de dados de outros produtos (por exemplo, TOTVS Agro Bioenergia ou TOTVS Agro Multicultivos, compatibilidade com Oracle, SQL Server e PostgreSQL);
  - Caso exista bloqueio/restrição baseado em whitelist, adicionar o host "https://tcserver-supplyagro.totvs.app/ ";

Como é feita a entrega

Um dos requisitos do TOTVS Connector Client é uma máquina que suporte o Docker (com a configuração para utilizar containers Linux), geralmente distribuições Linux já atendem este requisito. Um dos motivos é a forma de entrega, a aplicação está dentro de um *container* e precisa do Docker para executar o *container*.
Esta forma de *deploy* facilita a instalação e torna possível realizar entrega continua (CI) a partir de pipelines.

O que isso significa?

Um vez que um bug for corrigido ou uma nova funcionalidade estiver pronta, será possível realizar a atualização do TOTVS Connector Client sem a necessidade de intervenção humana.

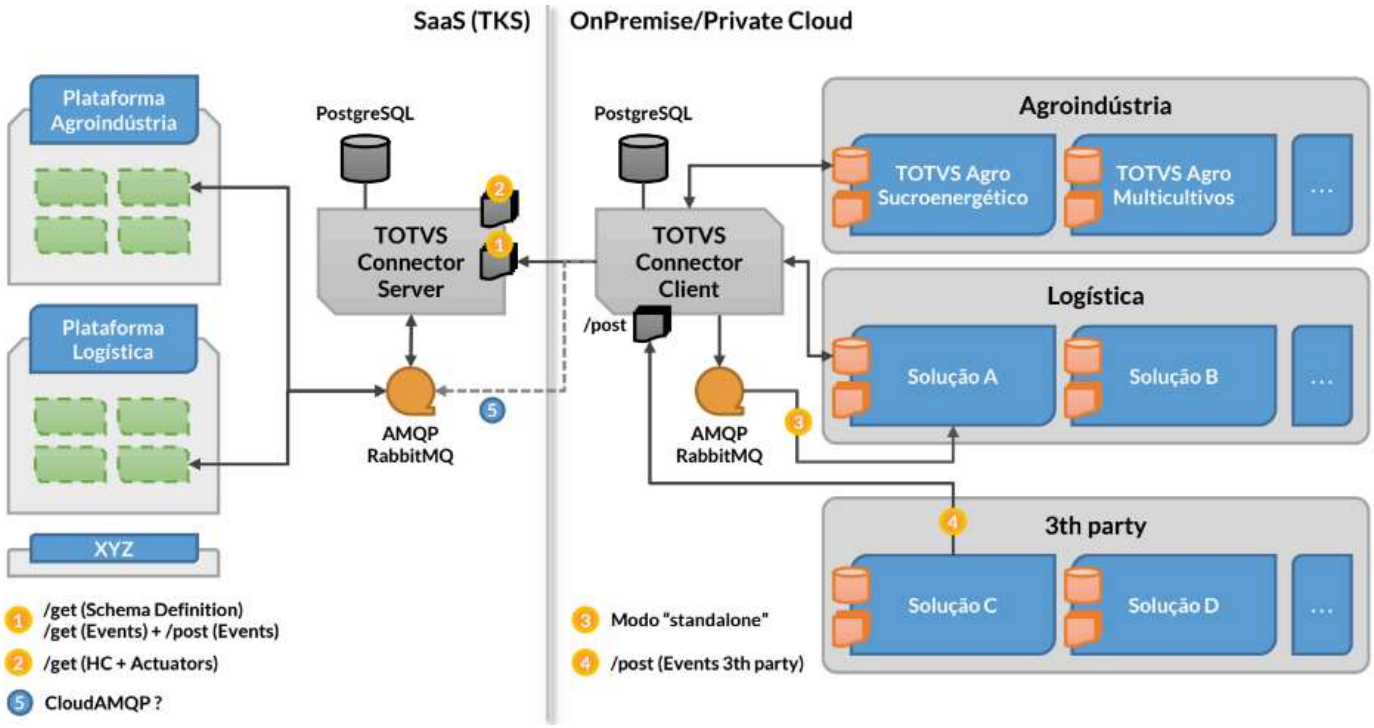
Como isso é feito?

É preciso realizar a instalação de um software chamado Azure Agent Pipeline na máquina que receberá o TOTVS Connector Client. Ele é responsável de conceder acesso a máquina, desta forma o pipeline de integração continua consegue acessar a máquina que irá receber o TOTVS Connector Client e realizar a atualização sem a necessidade de alguém atualizar manualmente o produto.
Para mais informações, acessar o link.

TOTVS Connector Server

É o responsável por receber todos os dados que serão integrados.
Ele deve ficar em um ambiente exposto na internet, já que todos os TOTVS Connector Client devem ser capazes de acessa-lo via *http*.
O TOTVS Connector Server precisa de uma instância do PostgreSQL e de um RabbitMQ (não é o mesmo do TOTVS Connector Client).
A instância do RabbitMQ (ou um RabbitMQ as a Service) precisa ser exposta, já que o TOTVS Connector Server e os produtos Cloud irão se conectar no RabbitMQ.

Abaixo uma imagem que ilustra essa arquitetura:



Funcionalidades

Abaixo serão descritas as funcionalidades do TOTVS Connector e como elas funcionam.

Schema Definition

O Schema Definition é uma estrutura de dados que possui as informações da tabela origem do dado e como ele deve ser convertido para ser enviado ao TOTVS Connector Server.
A gestão do Schema Definition é realizada apenas no TOTVS Connector Server, uma vez publicada uma nova versão do Schema Definition, todos os TOTVS Connector Client irão sincronizar os novos schemas.
A estrutura de um Schema Definition é representada pelo JSON abaixo:

```
{
    "id": "Identificador: UUID",
    "version": "Versão",
    "name": "Nome",
    "description": "Descrição",
    "status": "Status: PRODUCTION ou DEVELOPMENT",
    "createdAt": "Data de criação",
    "updatedAt": "Data da última atualização",
    "attributes": [ -- Array que contém objetos Schema Attributes
        {
            "name": "Nome do atributo",
            "required": true -- se o atributo é obrigatório,
            "type": "Tipo: ELEMENT ou ARRAY",
            "entityPrimaryKey": false -- se o atributo é a primary key da entidade,
            "foreignKeyType": "Tipo de chave estrangeira: NONE, PARENT_SCHEMA ou OTHER_SCHEMA",
            "attributeProducts": -- Array de objetos Schema Attribute Product [
                {
                    "productName": "Nome do produto",
                    "tableName": "Nome da tabela",
                    "foreignTableName": "Nome da tabela de referência a chave estrangeira",
                    "columns": -- Array de objetos Schema Attribute Columns [
                        {
                            "columnName": "Nome da coluna",
                            "parentColumnName": "Nome da coluna na tabela da chave estrangeira",
                            "type": "Tipo da coluna: STRING, NUMBER, DATE e CLOB",
                            "label": "Label da coluna quando o atributo for Primary Key, Parent Schema ou O
                        }
                    ]
                }
            ],
            "children": -- Array de objetos Schema Attributes quando o type for ARRAY []
        }
    ]
}
```

Segue abaixo um exemplo de Schema Definition:

```
{
    "id": "a126a79d-693c-4d2d-bbbd-a3cbff1abfc9",
    "version": "12.1.27",
    "name": "Instancia",
    "description": "Instancia",
    "status": "PRODUCTION",
    "createdAt": null,
    "updatedAt": null,
    "attributes": [
        {
            "name": "nome",
            "required": true,
            "type": "ELEMENT",
            "entityPrimaryKey": false,
            "foreignKeyType": "NONE",
            "attributeProducts": [
                {
                    "productName": "PIMSMC",
                    "tableName": "UNIDADEADM",
                    "foreignTableName": null,
                    "columns": [
                        {
                            "columnName": "DE_UNI_ADM",
                            "parentColumnName": null,
                            "type": "STRING",
                            "label": null
                        }
                    ]
                },
                {
                    "productName": "PIMSCS",
                    "tableName": "PMINSTANCIAS",
                    "foreignTableName": null,
                    "columns": [
                        {
                            "columnName": "DE_INSTANCIA",
                            "parentColumnName": null,
                            "type": "STRING",
                            "label": null
                        }
                    ]
                }
            ],
            "children": []
        },
        {
            "name": "codigo",
            "required": true,
            "type": "ELEMENT",
            "entityPrimaryKey": false,
            "foreignKeyType": "NONE",
            "attributeProducts": [
                {
                    "productName": "PIMSMC",
                    "tableName": "UNIDADEADM",
                    "foreignTableName": null,
                    "columns": [
                        {
                            "columnName": "CD_UNI_ADM",
                            "parentColumnName": null,
                            "type": "STRING",
                            "label": null
                        }
                    ]
                },
                {
                    "productName": "PIMSCS",
                    "tableName": "PMINSTANCIAS",
                    "foreignTableName": null,
                    "columns": [
                        {
                            "columnName": "INSTANCIA",
                            "parentColumnName": null,
```

```
                            "type": "STRING",
                            "label": null
                        }
                    ]
                }
            ],
            "children": []
        },
        {
            "name": "id",
            "required": true,
            "type": "ELEMENT",
            "entityPrimaryKey": true,
            "foreignKeyType": "NONE",
            "attributeProducts": [
                {
                    "productName": "PIMSMC",
                    "tableName": "UNIDADEADM",
                    "foreignTableName": null,
                    "columns": [
                        {
                            "columnName": "ID_UNIDADEADM",
                            "parentColumnName": null,
                            "type": "NUMERIC",
                            "label": "id"
                        }
                    ]
                },
                {
                    "productName": "PIMSCS",
                    "tableName": "PMINSTANCIAS",
                    "foreignTableName": null,
                    "columns": [
                        {
                            "columnName": "INSTANCIA",
                            "parentColumnName": null,
                            "type": "STRING",
                            "label": "id"
                        }
                    ]
                }
            ],
            "children": []
        }
    ]
}
```


Product Connection

Entidade gerenciada no TOTVS Connector Client, ela possui as informações de conexão do banco de dados do produto que será integrado.
Ao criar um Product Connection estamos informando que desejamos integrar dados do banco de dados informado, mas a integração não se dá automaticamente, após a criação precisamos informar quais schemas serão integrados.
É possível trabalhar com múltiplos Product Connection, o TOTVS Connector Client suporta a conexão com vários bancos de dados ao mesmo tempo.
Cada banco de dados deve possuir uma tabela chamada TCC_PRODUCT_METADATA, onde deve existir um registro com o nome do produto e a versão, essas informações serão necessárias para relacionar um Product Connection com o Schema Definition.

> A estrutura da TCC_PRODUCT_METADATA deve possuir dois campos de texto com o nome NAME e VERSION.

Os bancos de dados suportados são Oracle (11G e 12C), PostgreSQL e Microsoft SQL Server.

Exemplo de um Product Connection (SQL Server):

```
{
    "id": "68eaa1a2-aedb-4ba9-9015-a01a31bead89",
    "productName": "PIMS",
    "productVersion": "12.1.27",
    "url": "jdbc:jtds:sqlserver://localhost:1433/PIMS",
    "username": "PIMS",
    "password": "PIMS",
    "dataBaseType": "MSSQLSERVER",
    "enabled": true
}
```

Exemplo de um Product Connection (Oracle 11G):

```
{
    "id": "68eaa1a2-aedb-4ba9-9015-a01a31bead89",
    "productName": "PIMS",
    "productVersion": "12.1.27",
    "url": "jdbc:oracle:thin:@localhost:1521:PIMS",
    "username": "PIMS",
    "password": "PIMS",
    "dataBaseType": "ORACLE11G",
    "enabled": true
}
```

Exemplo de um Product Connection (PostgreSQL):

```
{
    "id": "68eaa1a2-aedb-4ba9-9015-a01a31bead89",
    "productName": "PIMS",
    "productVersion": "12.1.27",
    "url": "jdbc:postgresql://localhost:5432/PIMS",
    "username": "PIMS",
    "password": "PIMS",
    "dataBaseType": "POSTGRESQL",
    "enabled": true
}
```

Product Connection Schema

A entidade ProducConnectionSchema representa a relação entre as entidades ProductConnection e SchemaDefinition. Esta relação significa que uma determinada conexão integrará determinados SchemaDefinition. Por exemplo: a conexão A possui integração com os schemas Schema_A e Schema_B, e a conexão B, com os schemas Schema_B e Schema_C. Portanto, é possível configurar duas conexões com diferentes SchemaDefinition. A entidade SchemaDefinition é cadastrada no TOTVS Connector Server e a própria aplicação TOTVS Connector Client encarrega-se de sincronizar, automaticamente, esta entidade.

Após realizar o relacionamento, o TOTVS Connector Client iniciará o monitoramento das tabelas definidas no SchemaDefinition. Assim que houver qualquer alteração em um ou mais registros das tabelas monitoradas, o TOTVS Connector Client será notificado e processará este registro, enviando-o à plataforma TOTVS Connector Server.

Atributos Product Connection Schema

- O **atributo id** é o identificador do registro;
- O **atributo idProductConnection** é o id do registro da entidade ProducConnection;
- O **atributo idSchemaDefinition** é o id do registro da entidade SchemaDefinition;
- O **atributo versionSchemaDefinition** é a versão do SchemaDefinition relacionado. Este atributo não precisa ser passado para criar o relacionamento. Preenchido automaticamente;

- O **atributo nameSchemaDefinition** é o nome do SchemaDefinition relacionado. Este atributo não precisa ser passado para criar o relacionamento. Preenchido automaticamente;
- O **atributo enableStandalone** será explicado no tópico Modo Standalone;
- O **atributo enablePublishRemote** será explicado no tópico Modo Publish Remote;

Exemplo da estrutura da entidade ProductConnectionSchema

```
{
    "id": "39956f0f-341a-48c6-9c80-b5b3498e6899",
    "idProductConnection": "20fbd5a8-8ab4-4a16-8cc2-44702a36b8b1",
    "idSchemaDefinition": "8a14924d-7f93-4e2b-87af-c8622cd68859",

    "enablePublishRemote": "true",
    "enableStandalone": "false",

    "versionSchemaDefinitoin": "não_precisa_de_valor",
    "nameSchemaDefinition": "não_precisa_de_valor"
}
```

Métodos de Publicação de Mensagens

Existem dois métodos de publicação de mensagens no TOTVS Connector Client, a publicação local (Standalone) e a publicação remota (PublishRemote). Esses métodos são configurados em dois níveis: TOTVS Connector Client e Product Connection Schema.

Nível TOTVS Connector Client:

Ao subir um TOTVS Connector Client, temos os dois parâmetros de publicação para definirmos se publicará mensagens localmente, remotamente (TOTVS Connector Server) ou em ambos, ou seja, um é independente do outro. Por padrão temos a publicação remota (PublishRemote) ativa e a publicação local (Standalone) desativa. Em um cenário com apenas um método de publicação ativo, como por exemplo o cenário padrão, o TOTVS Connector Client irá publicar somente de maneira remota, ou seja, mesmo que em nível de Product Connection Schema seja requerido a publicação local, não irá ser publicada a mensagem localmente.

Nível Product Connection Schema:

Tendo um cenário com os dois métodos de publicação ativos, ou seja, o TOTVS Connector Client terá suporte para as duas opções, não necessariamente a mensagem será publicada das duas maneiras, a partir de agora isso é definido por um ProductConnectionSchema, sendo por meio dos atrubitos enablePublishRemote e enableStandalone a definição de onde os dados gerados serão publicados.

Modo Standalone

O modo standalone é uma forma de trabalhar apenas no ambiente On Premise, ou seja, um cenário onde é necessário integrar 2 ou mais produtos que estão em ambientes On Premise / Private Cloud.
Para habilitar o modo standalone é preciso iniciar o TOTVS Connector Client com o standalone ligado. Ao ligar o standalone o TOTVS Connector Client irá exigir uma conexão com uma instância do RabbitMQ, geralmente instalada no mesmo ambiente.
Uma vez definido o TOTVS Connector Client como standalone, precisamos habilitar Product Connection Schema como standalone.
Depois de tudo configurado teremos o seguinte comportamento após alguma das seguintes ações:

> Um registro é modificado no banco de dados do produto On Premise

> Um novo dado é recebido do TOTVS Connector Server

O TOTVS Connector Client irá enviar esse dado para o RabbitMQ, tornando possível qualquer aplicação que estiver conectada a uma fila receber o dado e integrar.
No modo standalone também é possível realizar o caminho inverso, uma aplicação pode enviar um dado para o RabbitMQ, seguindo a estrutura do Schema Definition, dessa forma o TOTVS Connector Client irá receber este dado e enviar para o TOTVS Connector Server (permitindo algum produto Cloud receber o dado) e inserir no banco de dados do produto On Premise.

External event

Um evento externo é um endpoint no TOTVS Connector Client que aceita mensagens via *http POST*, sendo possível enviar dados por aplicações que não possuem integração com o RabbitMQ.
Este endpoint aceita apenas o envio de mensagens, não há um endpoint que seja possível recuperar mensagens, as mensagens só são disponibilizadas no RabbitMQ via modo standalone.

Fluxo de dados

Iremos chamar de Cloud-A um produto hospedado em Cloud, com características de multi-tenancia, sem acesso externo ao banco de dados e o ambiente não está sobre gestão do cliente.
Já o OnPremise-A é um produto instalado em um ambiente do cliente ou em uma Private Cloud, onde é possível que o TOTVS Connector Client consiga acessar o banco de dados.

Produto Cloud-A -> Produto OnPremise-A

O Cloud-A precisa enviar uma mensagem para o RabbitMQ que é de gestão da TOTVS (uma instância do CloudAMQP), essa mensagem deve ser no formato TOTVSMessage (classe utilitária do TJF) e o conteúdo deve ser um dado em formato JSON definido pelo Schema Definition.
Esta mensagem será recebida pelo TOTVS Connector Server que irá guarda-lá e esperar um TOTVS Connector Client requisitar novas mensagens para o *token do cliente*.
De tempos em tempos o TOTVS Connector Client realiza a checagem se existe novos dados no TOTVS Connector Server, sempre usando um token válido. Quando uma nova mensagem é encontrada, é realizada algumas validações relacionadas ao conteúdo da mensagem, como por exemplo, se a mensagem é válida para o Schema Definition ativo no TOTVS Connetor Client.
A mensagem sendo válida e o TOTVS Connector Client estiver configurado para o OnPremise-A, a mensagem será processada conforma a sua ação, ou seja, se a mensagem for do tipo INSERT, o TOTVS Connector Client irá inserir o dado no banco de dados do OnPremise-A.
Os tipos de ação válidos são INSERT, UPDATE e DELETE.

> Se o modo standalone estive habilitado, a mensagem será enviada para o RabbitMQ local, geralmente instalado no ambiente On Premise.

Produto OnPremise-A -> Produto Cloud-A

Quando acontece um insert, update ou delete em uma das tabelas monitoradas, uma trigger irá salvar essa ação em uma tabela chamada TCC_EVENT, dessa forma o TOTVS Connector Client irá ler essa informação e disparar um processamento que irá recuperar o registro alterado e realizar uma conversão usando como base o Schema Definition.
Após essa conversão o TOTVS Connector Client verifica se o modo standalone está habilitado, se sim, esse dado será enviado para o RabbitMQ local.
Por fim o dado sempre será enviado ao TOTVS Connector Server. Ao receber o dado, será verificado se ele está de acordo com o Schema Definition informado, após essa verificação o TOTVS Connector Server irá enviar o dado para o RabbitMQ que está em Cloud.
Uma vez no RabbitMQ, qualquer aplicação Cloud que estiver conectada a uma fila irá receber esse dado e poderá integra-lo.

Client Environment

Para o funcionamento do TOTVS Connector Client é necessário realizar um cadastro de ambiente do cliente.
Este cadastro é feito no TOTVS Connector Server e irá gerar um token.
Esse token deve ser informado no momento da instalação do TOTVS Connector Client, por que no momento de iniciar o software ele irá validar se o token é válido.
Com um token válido, o TOTVS Connector Client pode sincronizar Schema Definition, buscar novos dados e enviar informações ao TOTVS Connector Server. Outra funcionalidade importante do token é a identificação do cliente, ou seja, todo dado trafegado deve possuir um token válido, só assim é possível separar os dados dos clientes.

> Os produtos Cloud devem ter o token de cada cliente para conseguirem diferenciar os dados enviados e recebidos.

Database Structure

Por causa dos vários objetos de banco de dados necessários para o funcionamento do TOTVS Connector Client, existe uma funcionalidade que checa de tempos em tempos a consistência desses objetos.
Cada tabela do Schema Definition irá criar uma trigger e duas colunas no banco do produto On Premise, caso algum desses objetos sofrer alterações indevidas o TOTVS Connector Client irá demonstrar essa falta de conformidade, facilitando verificações de problemas no funcionamento desejado.

Estrutura das mensagens

Toda mensagem trafegada possui uma estrutura em comum, o que varia é o conteúdo, que é definido pelo Schema Definition.
Segue abaixo um exemplo de uma mensagem com um schema fictício chamado PESSOA:

```
{
    "header": {
        "type": "PESSOA",
        "generatedOn": "2000-01-01T00:00:00.000000Z",
        "locale": "pt_BR"
    },
    "content": {
        "originApp": "PESSOAS",
        "appVersion": "1",
        "schemaName": "PESSOA",
        "schemaVersion": "1",
        "action": "INSERT",
        "data": {
            "id": {
                "id": 1
            },
            "rg": "10.10.10.10",
            "cpf": "100.100.100/10",
            "nome": "FULANO",
            "originId": "HASH_GERADO_PELO_APP_DE_ORIGEM",
            "tipoSangue": "O-",
            "dataNascimento": "1950-01-01T00:00:00+00:00"
        },
        "createdAt": "2000-01-01T00:00:00.335846Z",
        "token": "TOKEN_DO_CLIENTE_QUE_GEROU_O_DADO"
    }
}
```

O dado real que deve ser convertido e usado para integrar é a tag *data*, as outras informações são complementares para fins de informação.
Este é o formato padrão para as mensagens, ou seja, não importa se é standalone ou em Cloud, **todo** dado deve trafegar neste formato.

Informações importantes (para instalação)

Os endereços abaixo devem estar liberados no firewall para o correto funcionamento do tcclient:

- http://agrodata-gateway.agro.totvs.io:8092
- http://mdm.agro.totvs.io:8090
- https://totvstfs.visualstudio.com
- https://docker.totvs.io
- https://discordapp.com
- https://tcserver-supplyagro.totvs.app