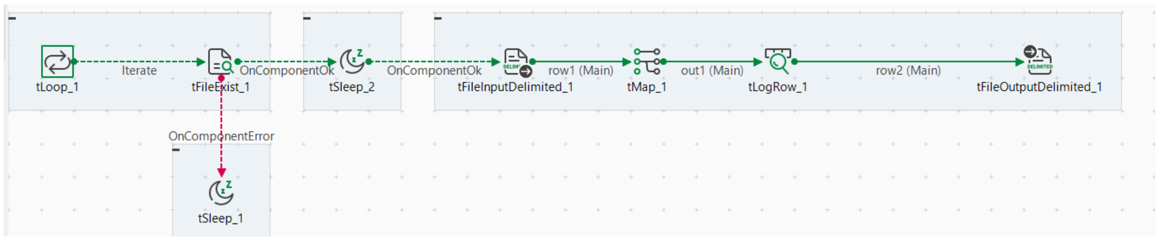
Exercise: Monitor a Folder for New Orders Using tLoop

Objective

Simulate a real-time order notification pipeline by monitoring a folder for new orders. The task will involve processing each new order file, logging the order details, and deleting the processed file. This exercise simulates real-time data integration using Talend Open Studio components.

Workflow



Components Used

1. tLoop
2. tFileExist
3. tFileInputDelimited
4. tMap
5. tLogRow
6. tFileOutputDelimited
7. tFileDelete (optional)

Steps for the Exercise

1. Create a Folder for File Monitoring

Create a folder at `C:/TalendDemo/incoming/`.
Place an initial `orders.csv` file in this folder for testing.

2. Create a New Job

Open Talend Open Studio and create a new job named `MonitorOrders_Folder`.

3. Add Components

To build the job, add the following components to the job:
1. tLoop: This component triggers the process repeatedly. It will check the folder every 10 seconds for new files.
2. tFileExist: This component checks if the `orders.csv` file exists in the folder.
3. tFileInputDelimited: This reads the content of the `orders.csv` file.
4. tMap: This processes the order data and generates an additional `order_message` field.
5. tLogRow: This logs the processed order data to the console.
6. tFileOutputDelimited: This writes the processed order data to `processed_orders.csv`.
7. tFileDelete (Optional): This deletes the `orders.csv` file after processing.

4. Configure tLoop

1. Type: `While`
2. Condition: `true` (This ensures the loop runs indefinitely until the job is stopped)
3. Delay: 10,000 ms (10 seconds)

5. Configure tFileExist

1. Directory: `C:/TalendDemo/incoming/`
2. File Name: `orders.csv`
3. Die on Error: Unchecked (This will prevent the job from failing if the file does not exist)
4. Output:
- IfExists → `True`: This path will be followed if the file exists.
- IfNotExists → `False`: This path will be followed if the file does not exist.

6. Configure tFileInputDelimited

1. File Name: `C:/TalendDemo/incoming/orders.csv`
2. Field Separator: `,`
3. Row Separator: `
` (newline character)
4. Header: 1 (This skips the first line, which is assumed to be a header)
5. Schema: Define columns as follows:
- `OrderID`
- `Customer`
- `Product`
- `Quantity`

7. Configure tMap

1. Input Fields: `OrderID`, `Customer`, `Product`, `Quantity`
2. Output Fields:
- `OrderID`
- `Customer`
- `Product`
- `Quantity`

- `order_message` (new field to generate a description of the order)
3. Expression for `order_message`: `"Order " + row1.OrderID + " by " + row1.Customer + " for " + row1.Quantity + " units of " + row1.Product`

8. Configure tLogRow

1. Mode: `Table`
2. Columns to display: `OrderID`, `Customer`, `Product`, `Quantity`, and `order_message`.

9. Configure tFileOutputDelimited

1. File Path: `C:/TalendDemo/logs/processed_orders.csv`
2. Append Mode: Checked (This ensures new data is appended rather than overwriting the existing file)
3. Map the fields to output: `OrderID`, `Customer`, `Product`, `Quantity`, and `order_message`.

10. (Optional) Configure tFileDelete

1. File Name: `C:/TalendDemo/incoming/orders.csv`
2. Delete File: Yes (This will delete the file after processing)

11. Run the Job

After setting up all the components, run the job.
The job will:
1. Check the folder every 10 seconds to see if `orders.csv` exists.
2. If the file exists, it will:
- Process the file by reading each order.
- Log the order details to the console using `tLogRow`.
- Write the processed data to `processed_orders.csv` using `tFileOutputDelimited`.
- Optionally, delete the `orders.csv` file using `tFileDelete`.

Expected Output

1. Console Output (tLogRow)

```
Order 0001 by John for 2 units of Mouse
Order 0002 by Anita for 1 unit of Laptop
```

2. Processed Orders Log (processed_orders.csv)

```
0001,John,Mouse,2,Order 0001 by John for 2 units of Mouse
0002,Anita,Laptop,1,Order 0002 by Anita for 1 unit of Laptop
```