WAY2LEARN

LAB- Performance Tuning

In this lab, you will be working on **01-performance-start** project under **06-performance-errorhandling**

In this lab you will understand

a) How to Use Splitter and Aggregator

b) How to configure flow processing strategies

c) How to create a cluster and understand how load balancing will happen in cluster

Step 1

1) Open bids.txt in src/main/resources and observe it. Each line is having BID information. We want to process all the bids and find a highest bidder.

Processing of the bid includes enriching a bid. The logic for enriching a bid is given to you in EnricherAuctionComponent.java. Open it and observe the logic. Observe that we have introduced delay by using Thread.sleep(50) .

Observe that we are printing thread name inside enrichSingle() method

After all the bids are enriched, the logic to find the higest bidder is present in HigestBidderAuctionComponent.java . Open it and observe it.

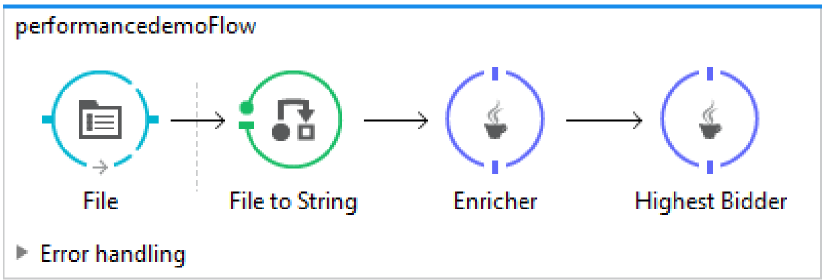
2) Create a new Mule configuration file with name "performancedemo ".Drag a file endpoint and configure it to poll for files after every 5 seconds under some directory like c:\input .

Drag File to String transformer after file endpoint.

Drag a Java component and configure it with EnricherAuctionComponent.

Drag another Java component and configure with HighestBidderAuctionComponent.

The flow should look like below :



3) Run the flow.   copy bids.txt from src/main/resources into c:\input folder . Observe that single thread is processing all the bids.

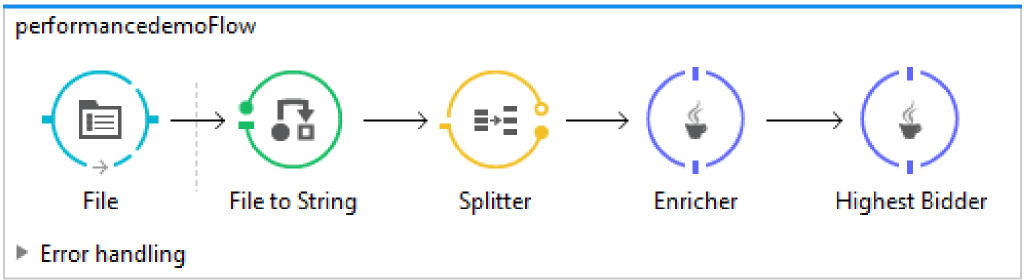
We want bids to be processed concurrently by multiple threads

4) We can leverage splitter in this scenario to split on message in to multiple messages so that parallel processing can be achieved.

So, add splitter after file_to_string transformer. Configure it with the following expression #[groovy:payload.readLines()]

The above expression returns List<String>. If the expression return a collection, the splitter component iterates over the collection and for each message in collection, it will create a message and post it to the channel after it.

Now your flow should look like below :

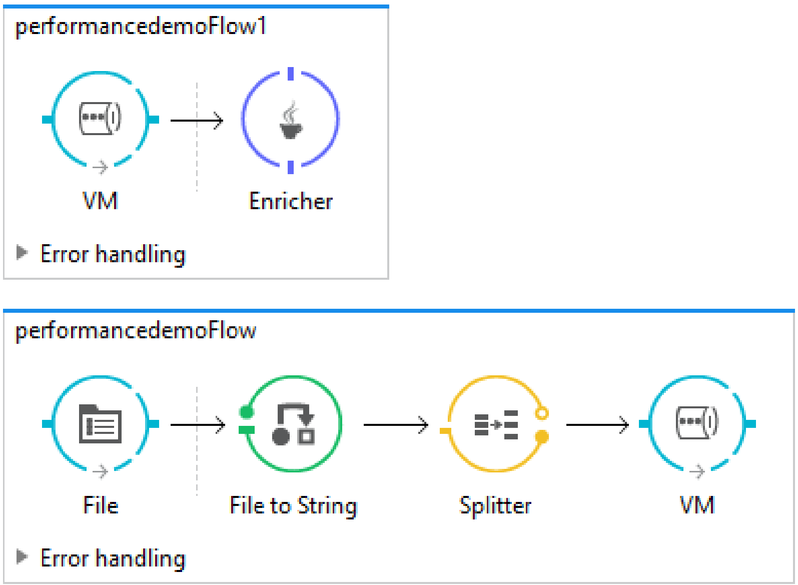


Run it. Keep the bids.txt under input folder.

Did u observe that bids are processed concurrently? Not yet.

So, Splitter alone cannot give you performance.

5) Now modify the flow to look like below :



Make sure you give same queue path to both VM Endpoints . I gave the queue path as **bidsvmq**

We have removed HighestBidder java component temporarily.

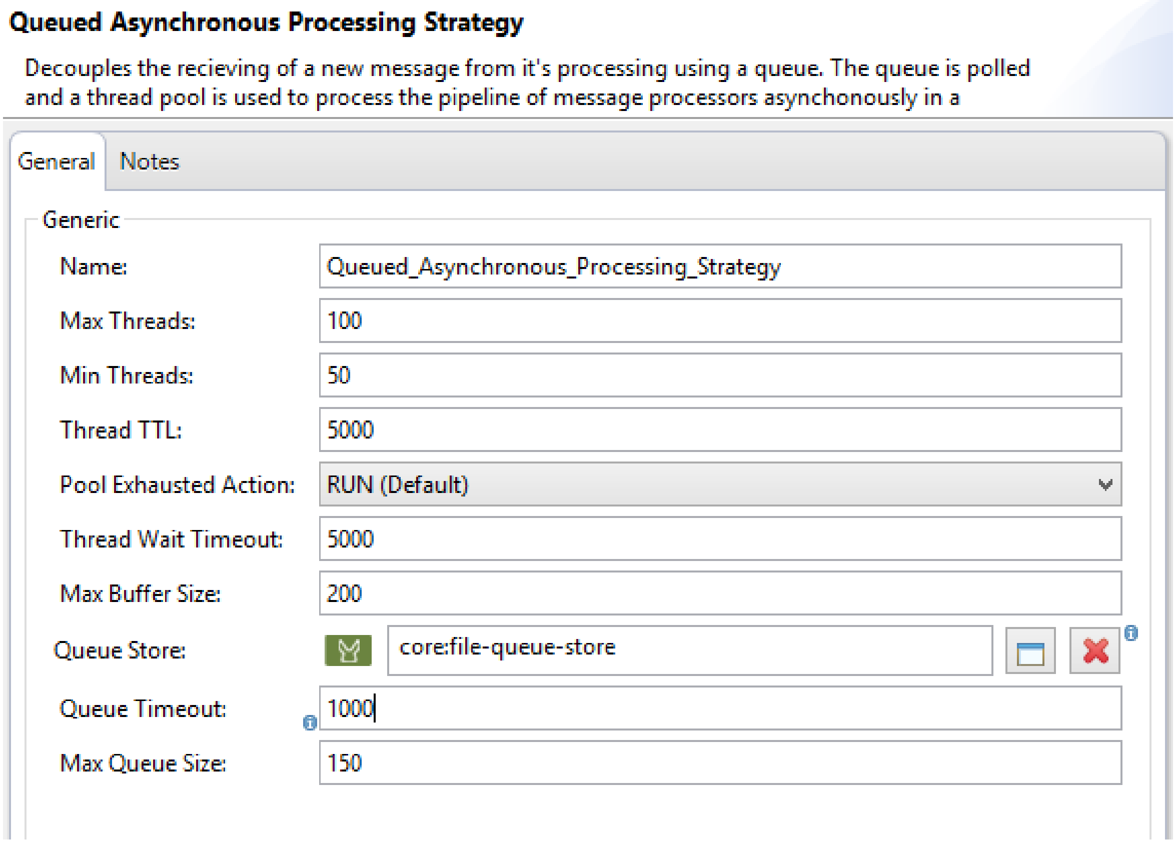
Now run the flow and keep the bids.txt in input folder.

Did u observe that 16 threads are processing the enrichment concurrently?

By default the one-way flow has a flow processing strategy as queued-asynchronous with default 16 threads.

6) Now click on the Flow   and click on + icon after flow processing strategy ref.

Select Queued Asynchronous Processing Strategy and configure as shown below :



Queued Asynchronous Processing Strategy

Decouples the recieving of a new message from it's processing using a queue. The queue is polled and a thread pool is used to process the pipeline of message processors asynchonously in a

Name: Queued_Asynchronous_Processing_Strategy

Max Threads: 100

Min Threads: 50

Thread TTL: 5000

Pool Exhausted Action: RUN (Default)

Thread Wait Timeout: 5000

Max Buffer Size: 200

Queue Store: core:file-queue-store

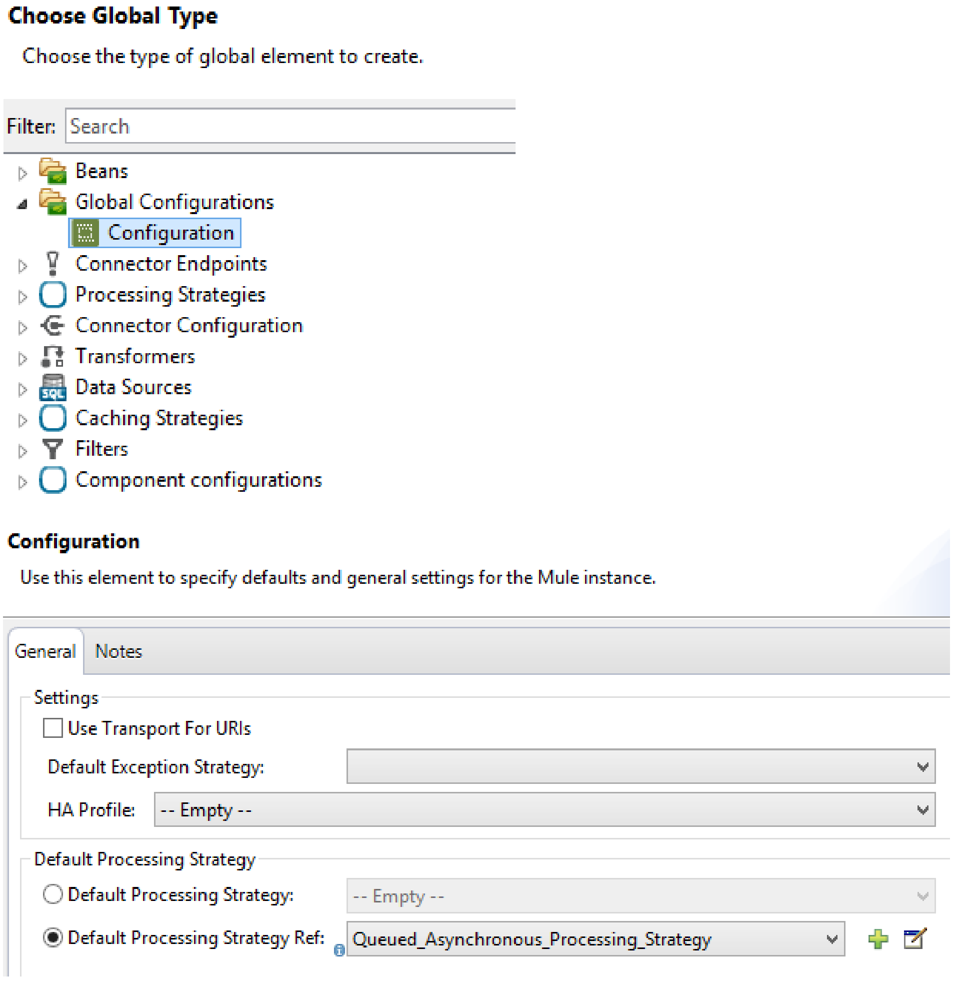
Queue Timeout: 1000

Max Queue Size: 150

7) Now, run the flow and put the bids.txt under input folder . Observe that bids are processed concurrently by maximum 100 threads.

8) Now we want this strategy to become the global default strategy.

Go to Global elements and click on create. Under Global Configurations, select Configuration and select the previous strategy as shown below :



Choose Global Type

Choose the type of global element to create.

Filter: Search

- ▷ Beans
- ◢ Global Configurations
  - Configuration
- ▷ Connector Endpoints
- ▷ Processing Strategies
- ▷ Connector Configuration
- ▷ Transformers
- ▷ Data Sources
- ▷ Caching Strategies
- ▷ Filters
- ▷ Component configurations

Configuration

Use this element to specify defaults and general settings for the Mule instance.

General | Notes

Settings

□ Use Transport For URIs

Default Exception Strategy:

HA Profile: -- Empty --

Default Processing Strategy

○ Default Processing Strategy: -- Empty --

● Default Processing Strategy Ref: Queued_Asynchronous_Processing_Strategy

9) Dont configure as shown below . It is just for information

We can configure default receiver threading profile and dispatcher threading profile as shown below :

```
   <configuration doc:name="Configuration">
        <http:config useTransportForUris="false"/>

        <default-dispatcher-threading-profile maxThreadsActive="20" maxBufferSize="50"
                          maxThreadsIdle="5" poolExhaustedAction="WAIT"
                          threadTTL="1000"/>
        <default-receiver-threading-profile maxThreadsActive="20" maxBufferSize="50"
                          maxThreadsIdle="5" poolExhaustedAction="WAIT"
                          threadTTL="1000"/>

    </configuration>
```

We can configure receiver threading profile and dispatcher threading profile for VM endpoints as shown below :

```
<vm:connector name="VM" validateConnections="true" doc:name="VM">
      <dispatcher-threading-profile maxThreadsActive="20" maxBufferSize="50"
                          maxThreadsIdle="5" poolExhaustedAction="WAIT"
                          threadTTL="1000"/>
        <receiver-threading-profile maxThreadsActive="20" maxBufferSize="50"
                          maxThreadsIdle="5" poolExhaustedAction="WAIT"
                          threadTTL="1000"/>

    </vm:connector>
```