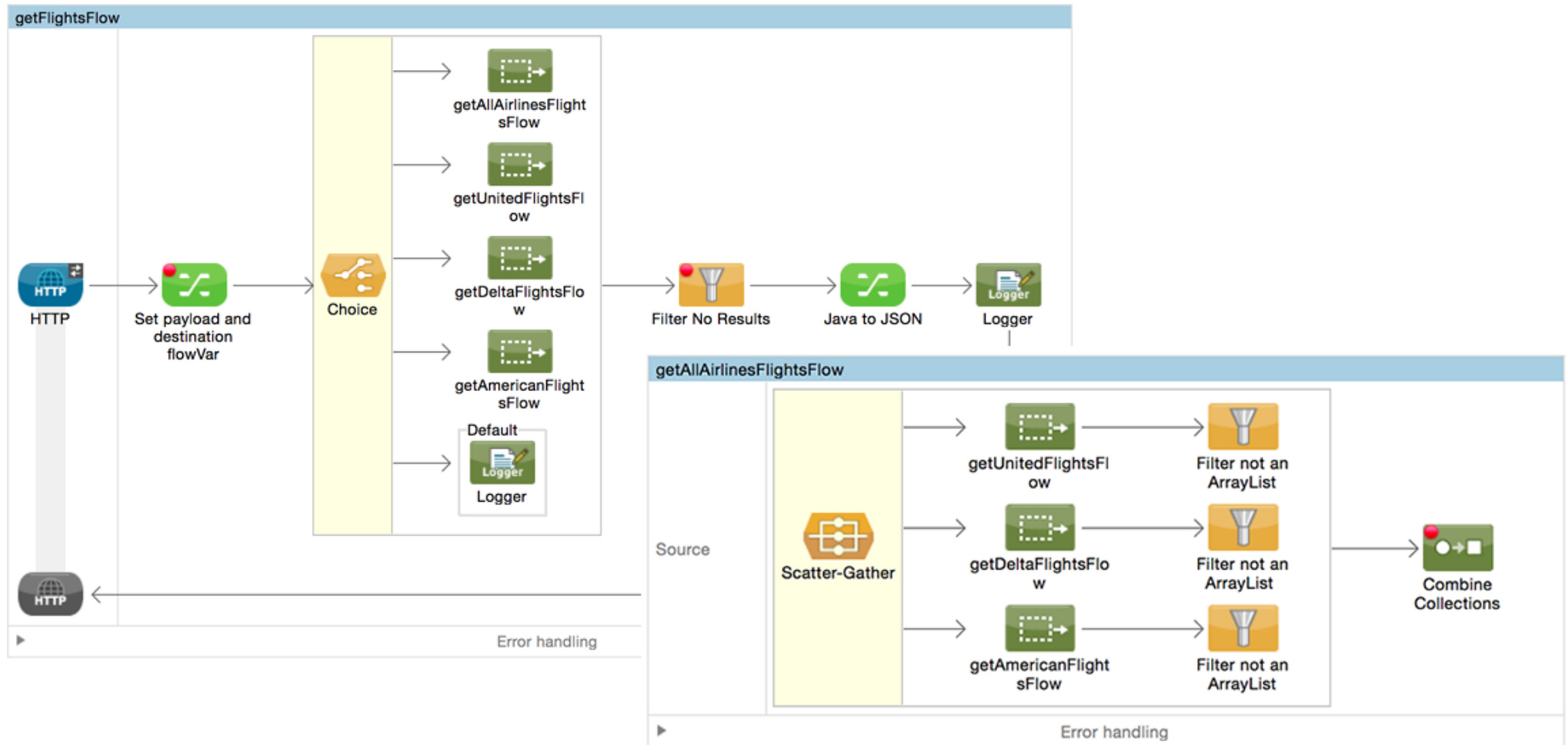




Module 8: Controlling Message Flow



Goal



Objectives

- In this module, you will learn:
 - About flow control and filter elements
 - To multicast a message
 - To route message based on conditions
 - To filter messages
 - About synchronous and asynchronous flows
 - To create an asynchronous flow

Routing messages

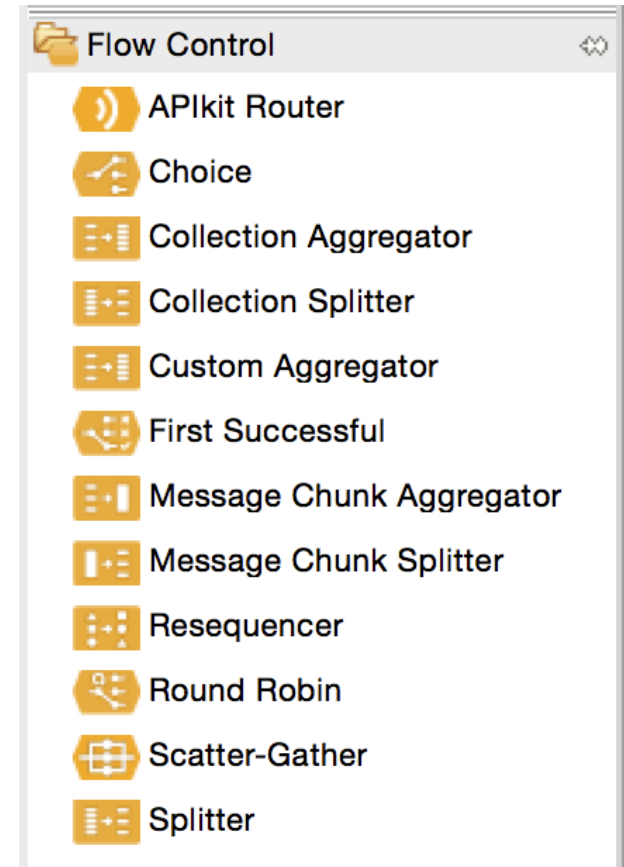


Routers

- Route messages to various destinations in a Mule flow
- Some incorporate logic to analyze and possibly transform messages before routing takes place
- Some change the payload, some don't

Available flow controls

- Three main types, those that
 - Split and/or aggregate
 - Scatter-Gather
 - Multicast and aggregate
 - Scatter-Gather
 - Check logic and route
 - Choice

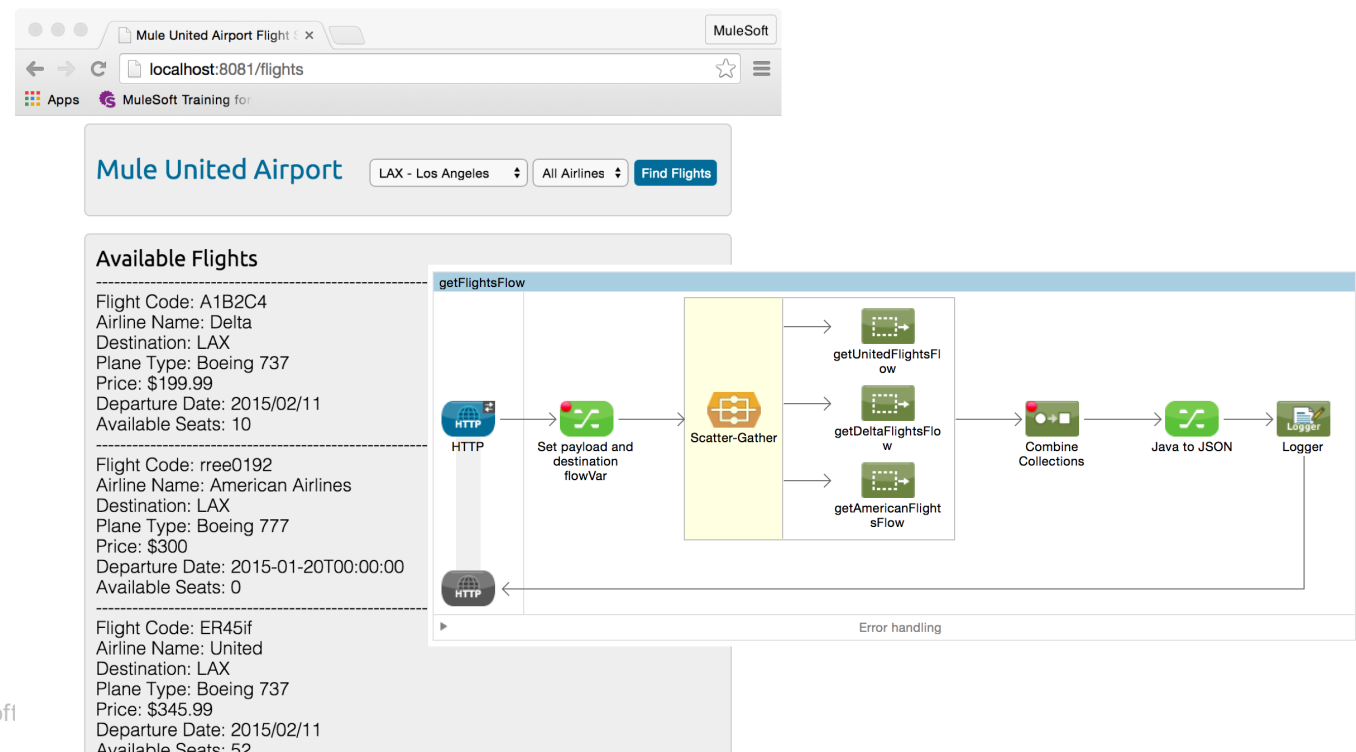


The Scatter-Gather router

- Scatter-Gather sends the message to each route concurrently and returns a collection of all results
- Is often used with the Combine Collections transformer
 - Flattens a collection of collections into one collection

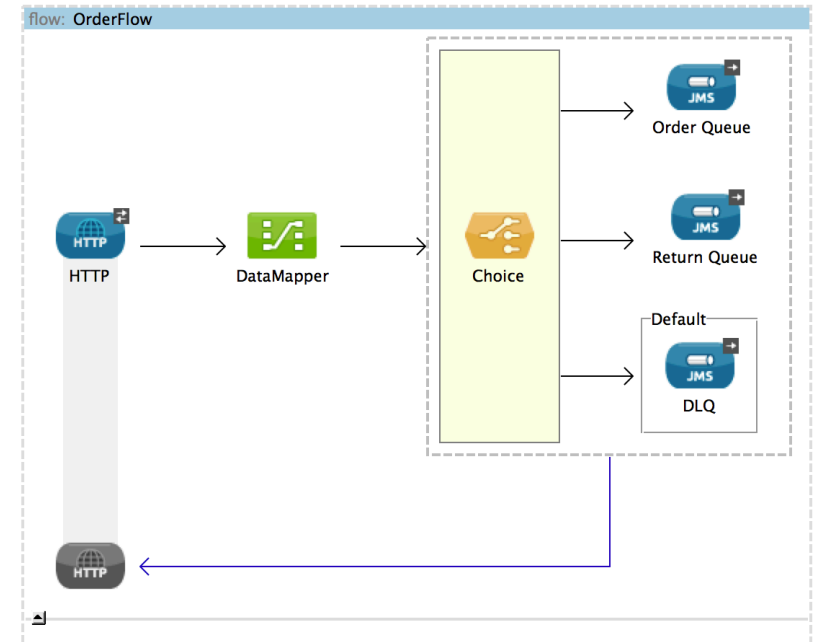
Walkthrough 8-1: Multicast a message

- Use a Scatter-Gather router to concurrently call all flight services
- Use a Combine Collections transformer to combine a collection of three ArrayLists of objects into one collection
- Use DataWeave to sort the flights and return them as JSON to the form






The Choice router

- Sends the message to one route based on conditions
 - Each path can include multiple message processors

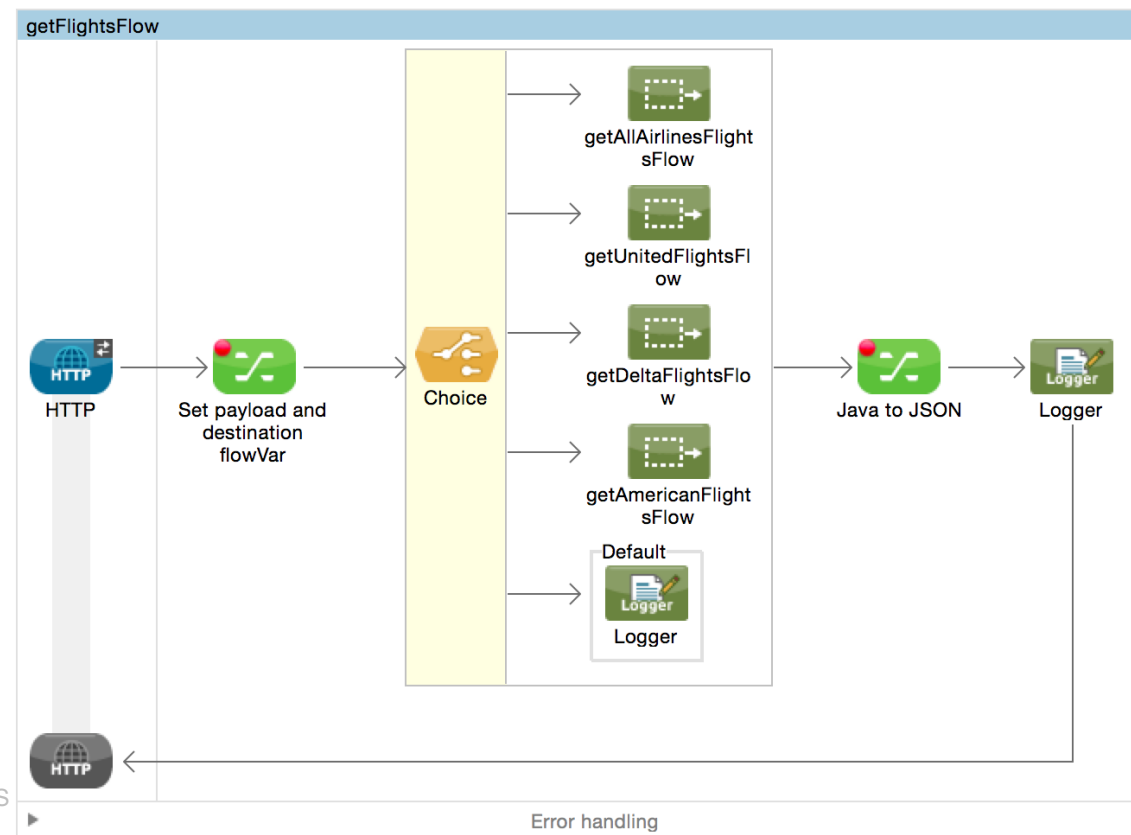


- The conditions are written with MEL

When	Route Message to
<code>#[message.inboundProperties['requestType']] == 'order'</code>	 Order Queue
<code>#[message.inboundProperties['requestType']] == 'return'</code>	 Return Queue
Default	 DLQ

Walkthrough 8-2: Route messages based on conditions

- Use a Choice router to get flight results for all three airlines or only a specific airline
- Set the router paths based on the airline value sent from the flight form



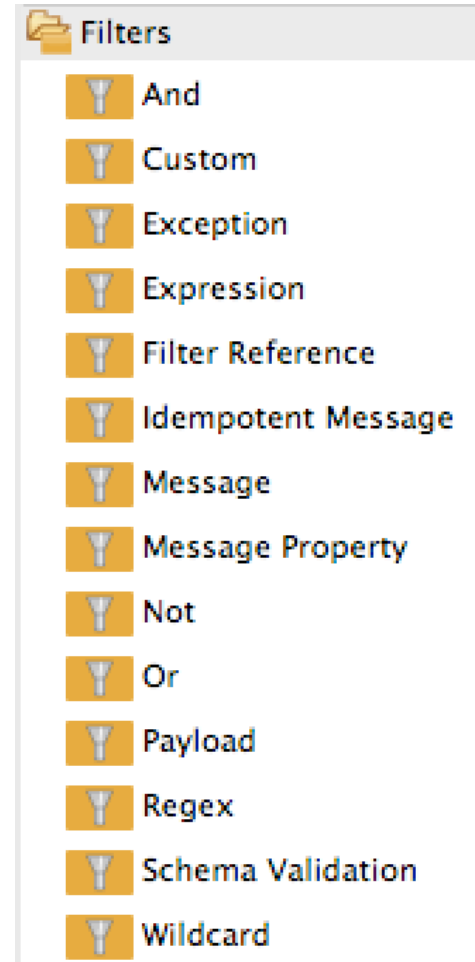
Filtering messages



- Determine whether a message can proceed in a Mule flow
- By default, filtered messages are dropped and processing of the message ends
 - Keeps subsequent processors from receiving irrelevant or incomprehensible messages
 - Filters can be configured to throw an exception

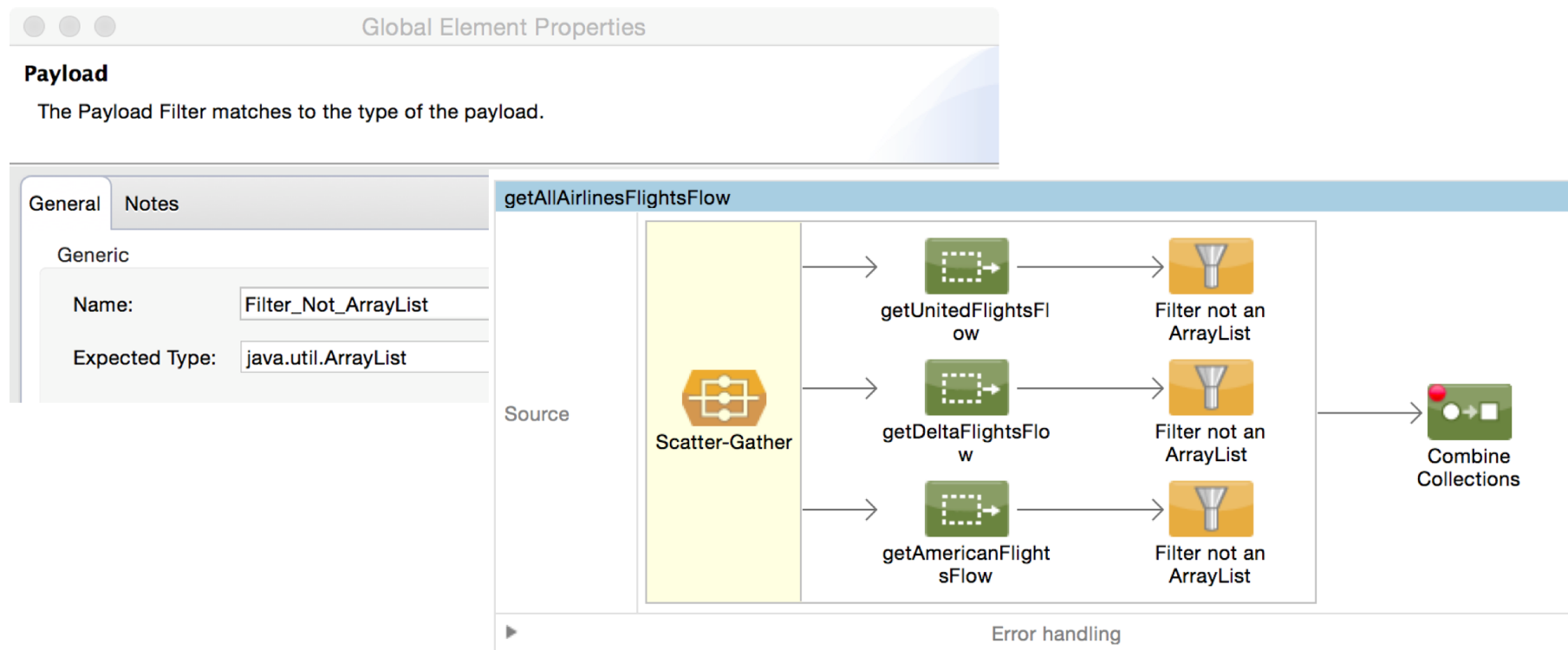
Available filters

- There are 12 bundled filters
 - Expression and Payload are often used
 - And, Or, Not apply Boolean logic
 - Message filter nests other filters for more complex logical conditions
 - Idempotent ensures a message is not delivered more than once
- You can create custom filters
- You can define global filters and reference them for reuse



Walkthrough 8-3: Filter messages

- Filter the results in the multicast to ensure they are ArrayLists and not exception strings
- Use the Payload and Expression filters
- Create and use a global filter



Understanding flow processing strategies

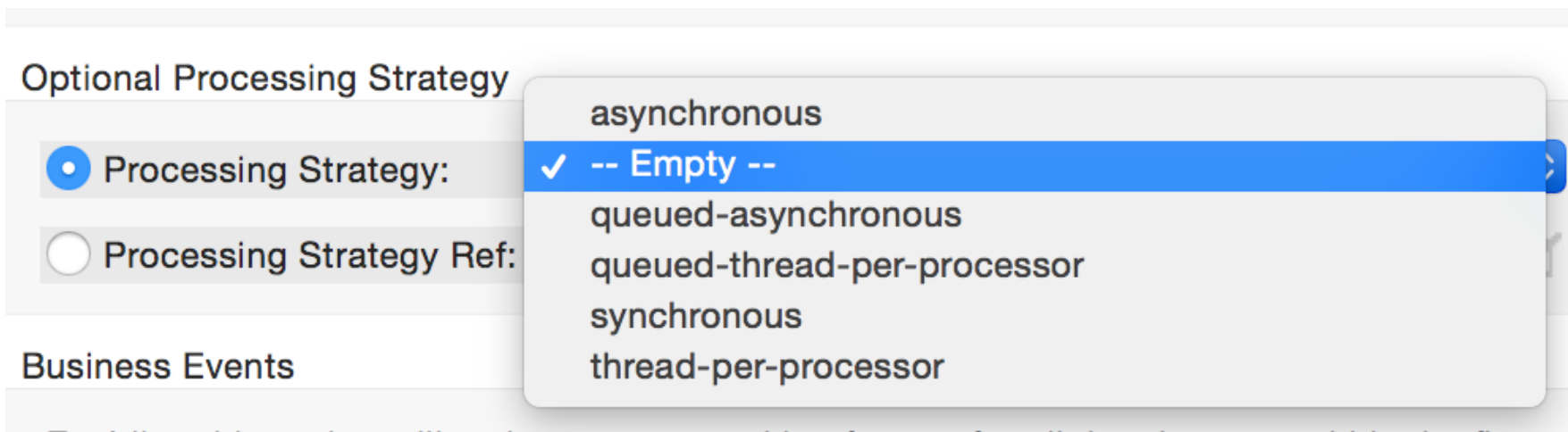


What is a flow processing strategy?

- A flow **processing strategy** determines how Mule implements message processing for a given flow
 - Should the message be processed synchronously (on the same thread) or asynchronously (on a different thread)?
 - If asynchronously, what are the properties of the pool of threads used to process the messages?
 - If asynchronously, how will messages wait for their turn to be processed in the second thread?

Flow processing strategies

- All Mule flows have an implicit processing strategy which Mule applies automatically
 - Either synchronous or queued-asynchronous
 - Each of these is optimal for certain flows
- The processing strategy can be changed



The common flow processing strategies

- Synchronous
 - After the flow receives a message, all processing, including the processing of the response, is done in the same thread
 - With the exception of asynchronous scope Async
- Queued-asynchronous
 - Default if no other processing strategy is configured
 - Higher throughput
 - Uses a queue to decouple the flow's receiver from the rest of the steps in the flow
 - Works the same way in a scope as in a flow
 - Can fine tune number of threads, number queued, object store
 - Asynchronous strategy is same except it doesn't use a queue so cannot be distributed across nodes in a cluster

What determines a flow's processing strategy?

- Mule selects a processing strategy for a flow based the flow's exchange pattern (and if its transactional)
- The flow exchange pattern is determined by the exchange pattern of the inbound endpoint
 - A **request-response exchange pattern** is used when the sender of the messages expects a response
 - Mule applies a synchronous processing strategy
 - A **one-way exchange pattern** is used when no response is expected
 - Mule applies a queued-asynchronous processing strategy

Passing messages to an asynchronous block

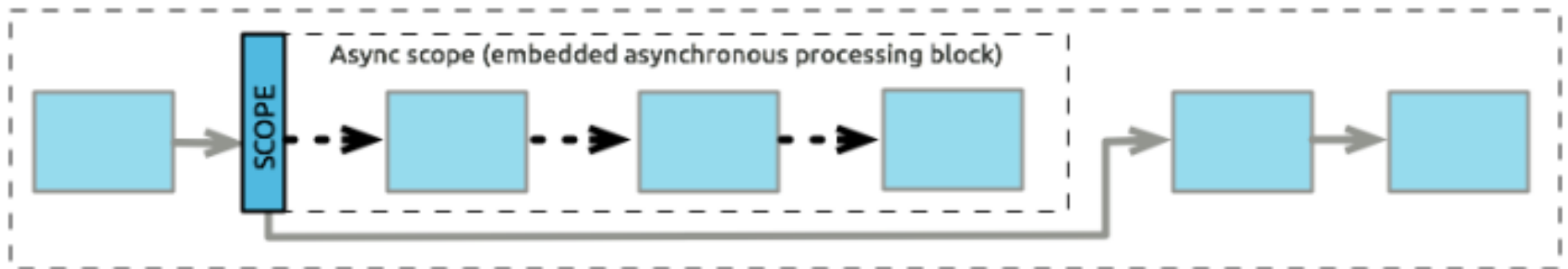


Async scope

- A branch processing block that executes simultaneously with the parent message flow
 - Useful for executing time-consuming operations that do not require sending a response back to the initiating flow
 - Printing a file or connecting to a mail server

Async scope

- Sends a message copy to
 - The first message processor in its own processing block
 - The next message processor in the main flow
- The payload is not copied
 - The same payload object(s) will be referenced by both messages in the two flows

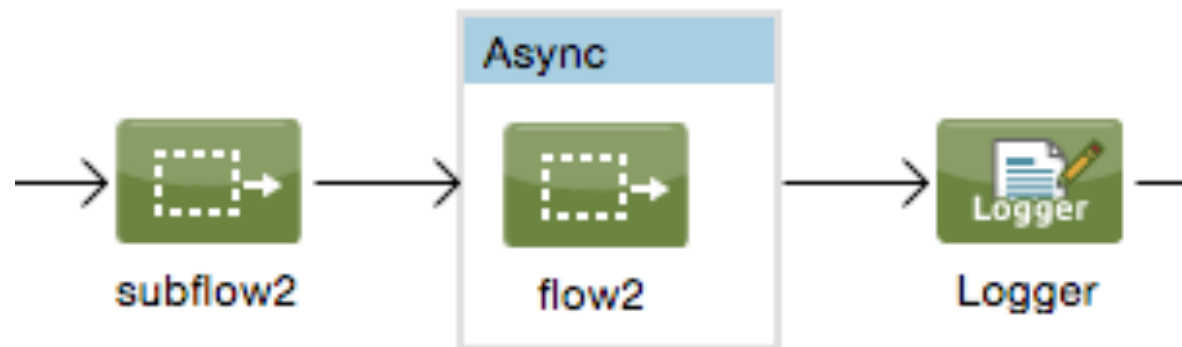


Async scopes vs asynchronous flows

- Similarities
 - Processes the message asynchronously with the main flow without pausing the processing in the main flow thread
 - Does not pass data back into the main flow thread
 - Can have its own processing strategy
- Differences
 - Exists in-line with the main flow
 - Is not called by a flow reference component
 - Is not re-usable
 - Cannot have its own exception handling strategy

Walkthrough 8-4: Pass messages to an asynchronous flow

- Use the Async scope element to create an asynchronous flow
- Use the Mule Debugger to watch messages flow through both flows



Summary



Summary

- In this module, you learned to use different, routers, filters, and scopes to control message flow
- Use the Scatter-Gather router to send a message concurrently to multiple routes
 - A collection of all results is returned
 - Use Combine Collections to flatten the collection
- Use the Choice router to send a message to one route based on conditions
- Use filters to determine whether a message can proceed in a Mule flow

Summary

- A flow processing strategy determines how Mule implements message processing for a given flow
- All Mule flows have an implicit processing strategy which Mule applies automatically
 - Endpoints with a request-response exchange pattern are set to synchronous
 - Endpoints with a one-way exchange pattern are set to queued-asynchronous
- Use Async scope to create a branch processing block that executes simultaneously with the parent message flow