‘Timeout’ flows

Yuma Matsuura

Hitachi Solutions, Ltd.

Background

- Sometimes a function in a node takes a longer time to process. In this case, a lot of flow developers want to treat this behavior as a timeout error.

- I would like to describe enhancements of some nodes for 'timeout' flows.

What is Timeout flow?

- If processing takes time on a certain node, a timeout error occurs after the specified time has elapsed.

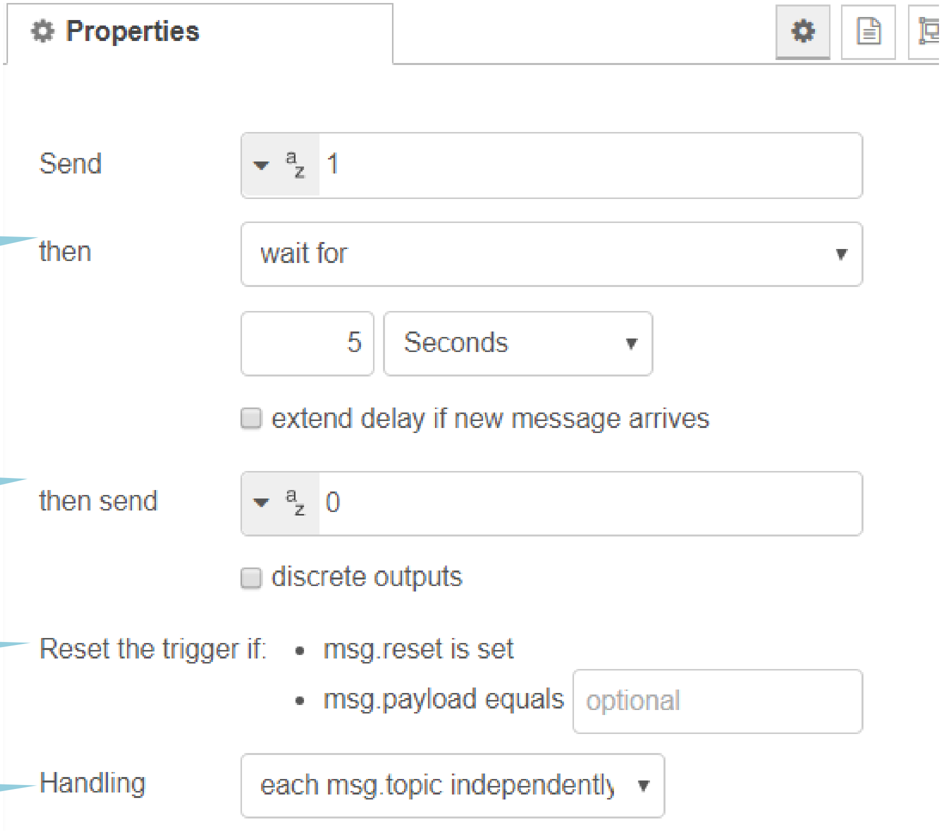
- Do not send a message to the next node when a timeout error occurs.

- Timeout value can be specified for each flow.

- Node-RED user can create timeout flow simply.

Assumption of the Trigger Node

- It is assumed that the Trigger node has following settings when creating a timeout flow.



Send: 1

then: wait for

5 Seconds

□ extend delay if new message arrives

then send: 0

□ discrete outputs

Reset the trigger if:
- msg.reset is set
- msg.payload equals optional

Handling: each msg.topic independently

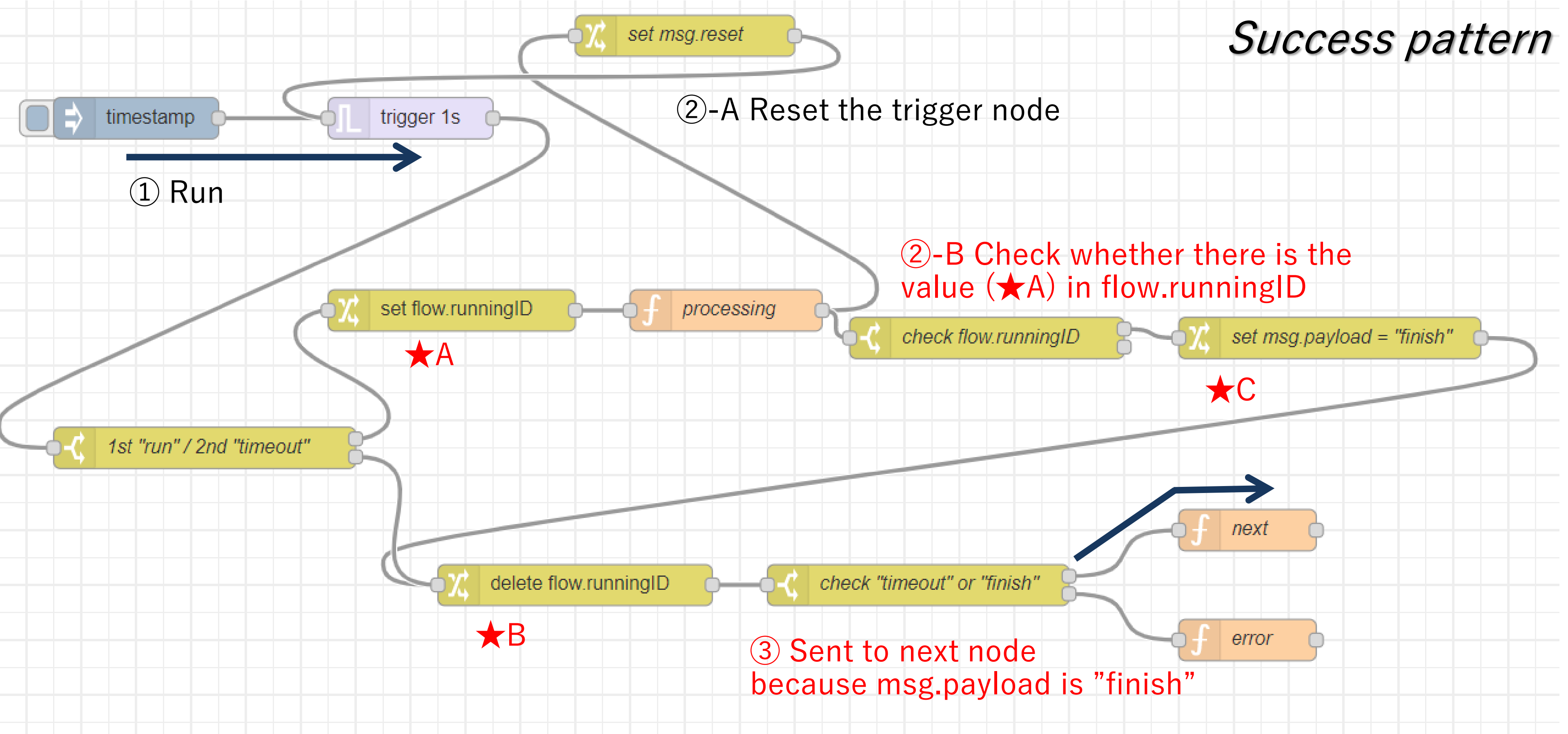
then ： wait for
(It is a timeout value)

then send ： message to send when specified time has passed

Reset with msg.reset or msg.payload

Handling：each msg.topic

To achieve with existing nodes

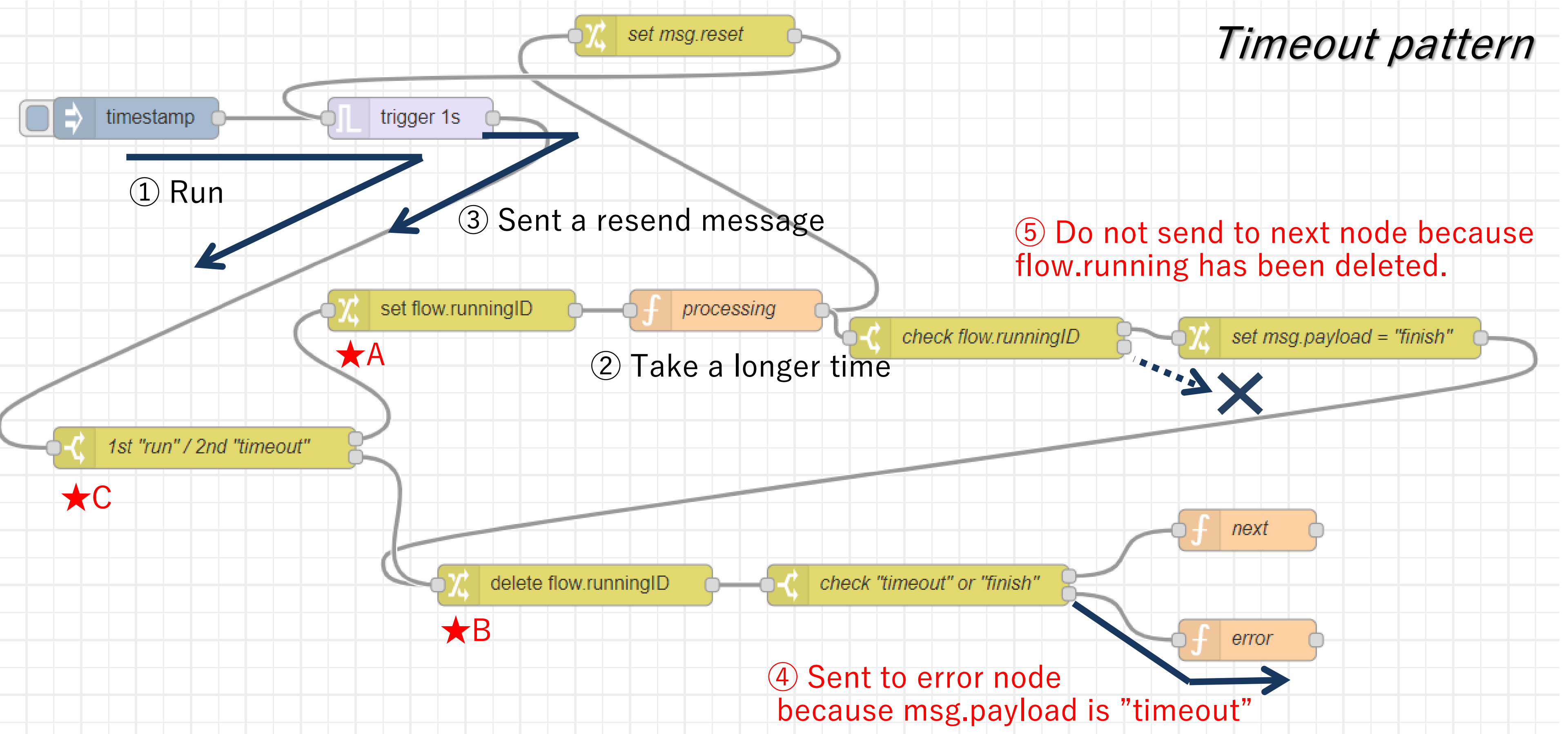

Success pattern
set msg.reset
timestamp
trigger 1s
① Run
②-A Reset the trigger node
②-B Check whether there is the value (★A) in flow.runningID
set flow.runningID
processing
check flow.runningID
set msg.payload = "finish"
★A
★C
1st "run" / 2nd "timeout"
next
delete flow.runningID
check "timeout" or "finish"
error
★B
③ Sent to next node because msg.payload is ”finish”

★A Set _msgid in flow.runningID

★B Delete flow.runningID

★C Set “finish” in msg.payload

To achieve with existing nodes

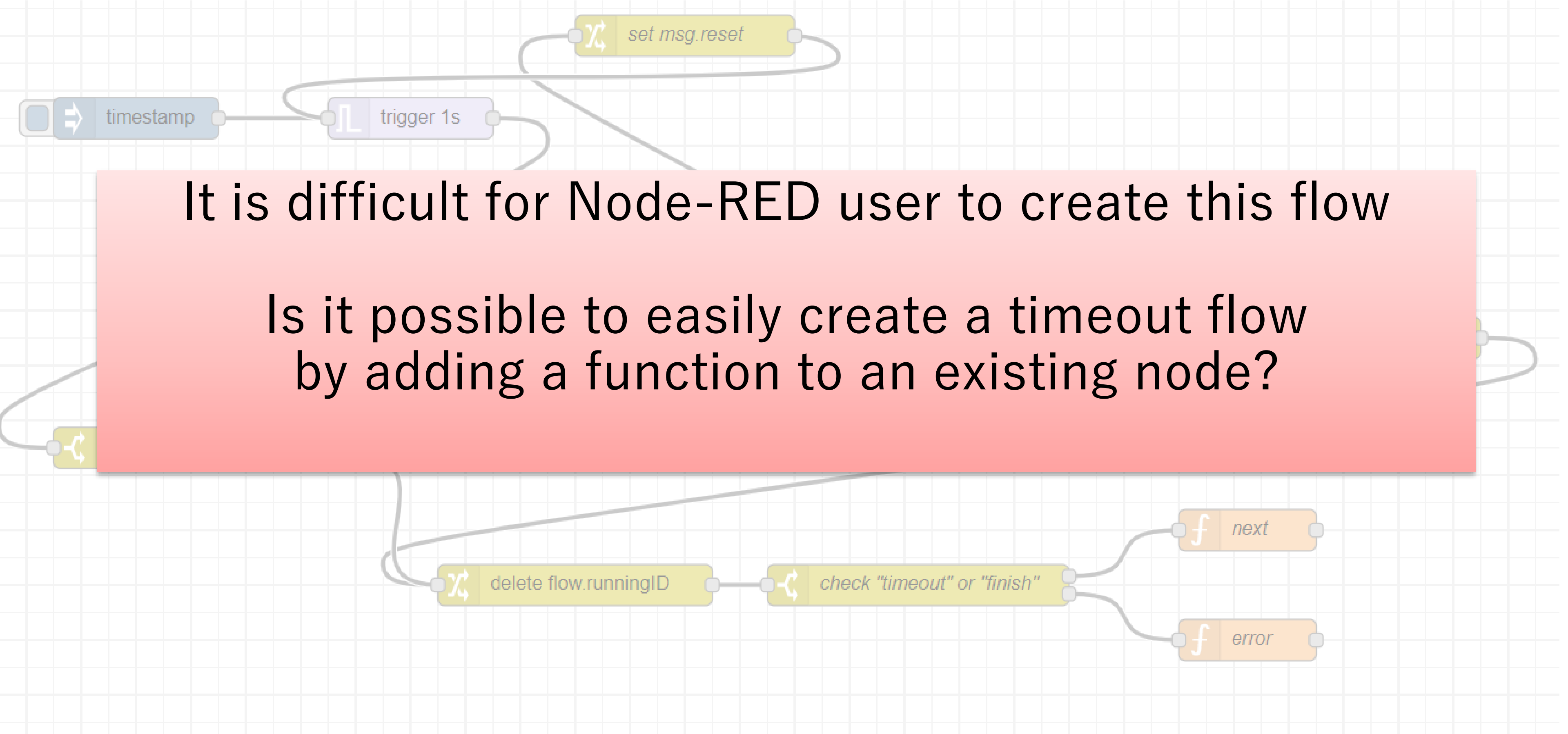

Timeout pattern
set msg.reset
timestamp
trigger 1s
① Run
③ Sent a resend message
⑤ Do not send to next node because flow.running has been deleted.
set flow.runningID
processing
check flow.runningID
set msg.payload = "finish"
★A
② Take a longer time
1st "run" / 2nd "timeout"
★C
next
delete flow.runningID
check "timeout" or "finish"
error
★B
④ Sent to error node because msg.payload is ”timeout”

★A Set _msgid in flow.runningID

★B Delete flow.runningID

★C Determine whether it is the first time or resend message.

To achieve with existing nodes



It is difficult for Node-RED user to create this flow

Is it possible to easily create a timeout flow by adding a function to an existing node?

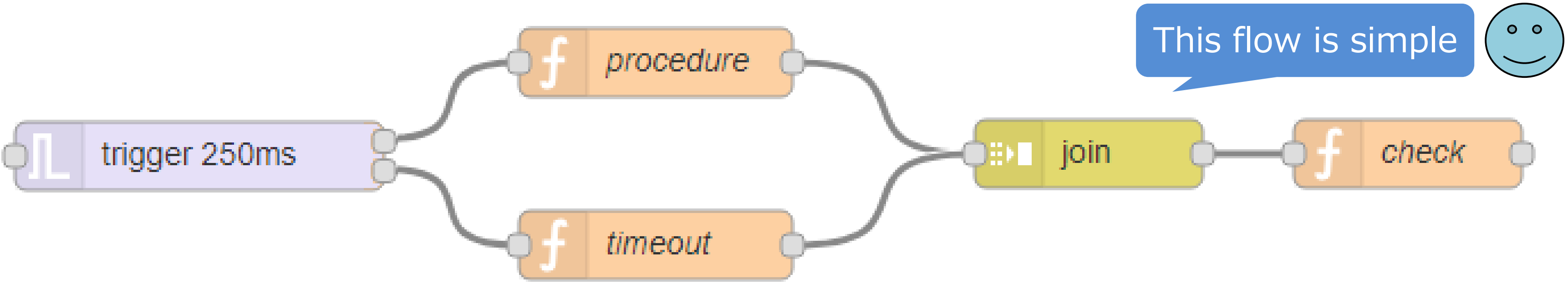
[Plan1] Trigger and Join

- Create timeout flow with *Trigger Node and Join Node*
  - ➢ Trigger Node
    1. Add “discrete outputs” option
       - Send a ‘then send’ message to second port after a set number of seconds.
    2. Add ”memorize ids” option
       - Memorize ‘_msgid’ in flow context.

NG point

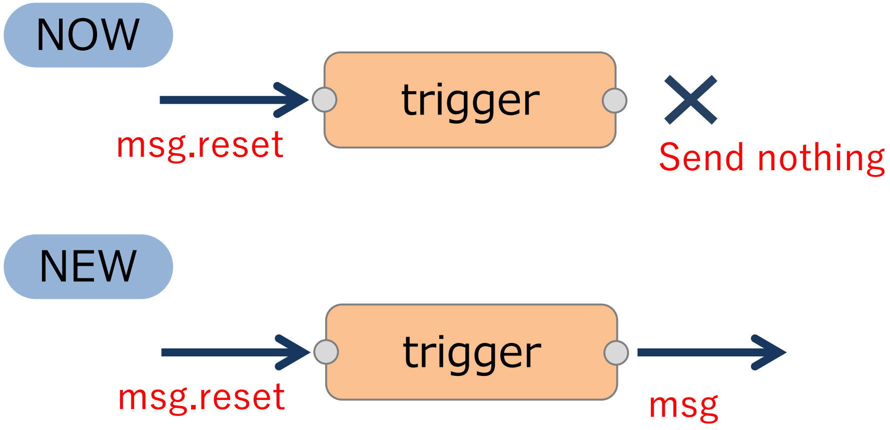
  - ➢ Join Node
    1. Add “once” option
       - Deletes the message id when finds the same message id in the flow context, and then passes the message to the next node.
       - If there is no same message id, the message will be ignored.
       - One message is only passed the first time so that messages arriving after a timeout are no sent to next node.

This flow is simple



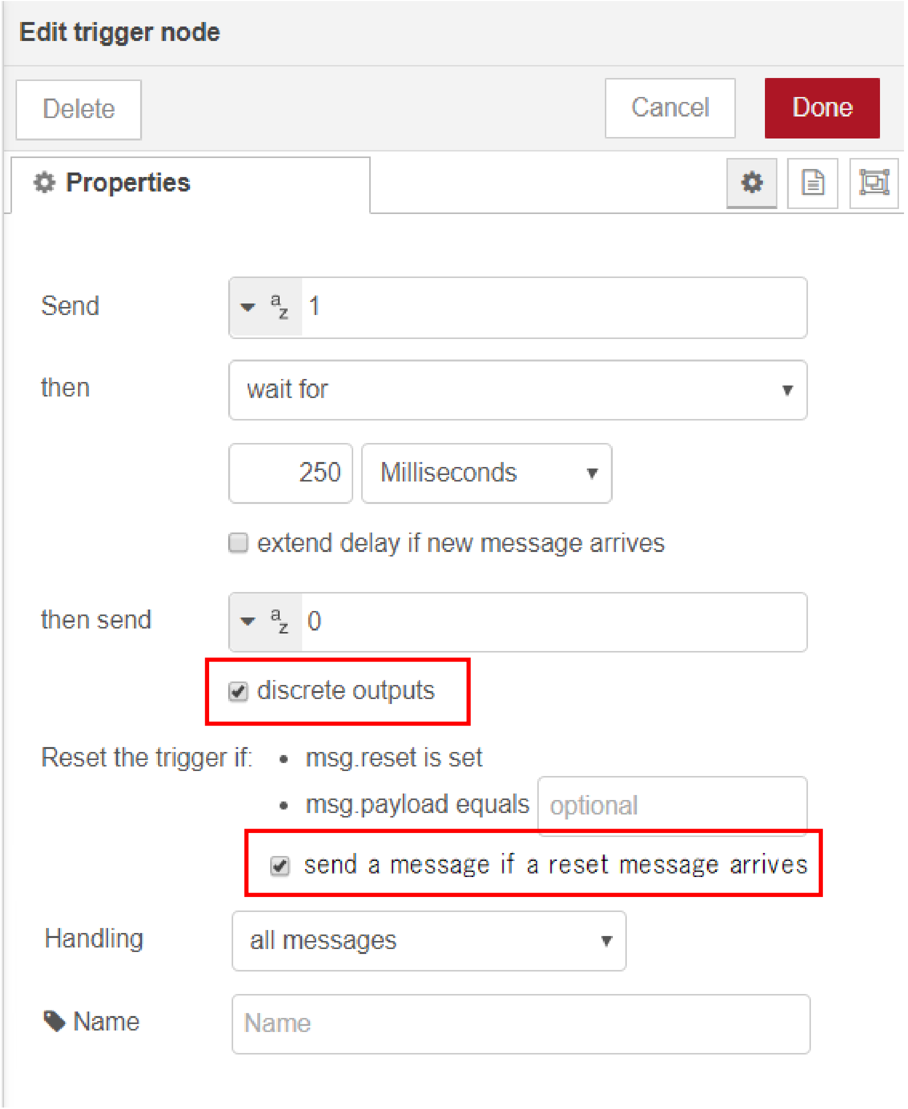
[New idea]Trigger Node improvement

① send a message if a reset message arrives



② discrete outputs

- Trigger node has three output ports .



Send: 1

then: wait for

250 Milliseconds

☐ extend delay if new message arrives

then send: 0

☑ discrete outputs

Reset the trigger if:
- msg.reset is set
- msg.payload equals optional

☑ send a message if a reset message arrives

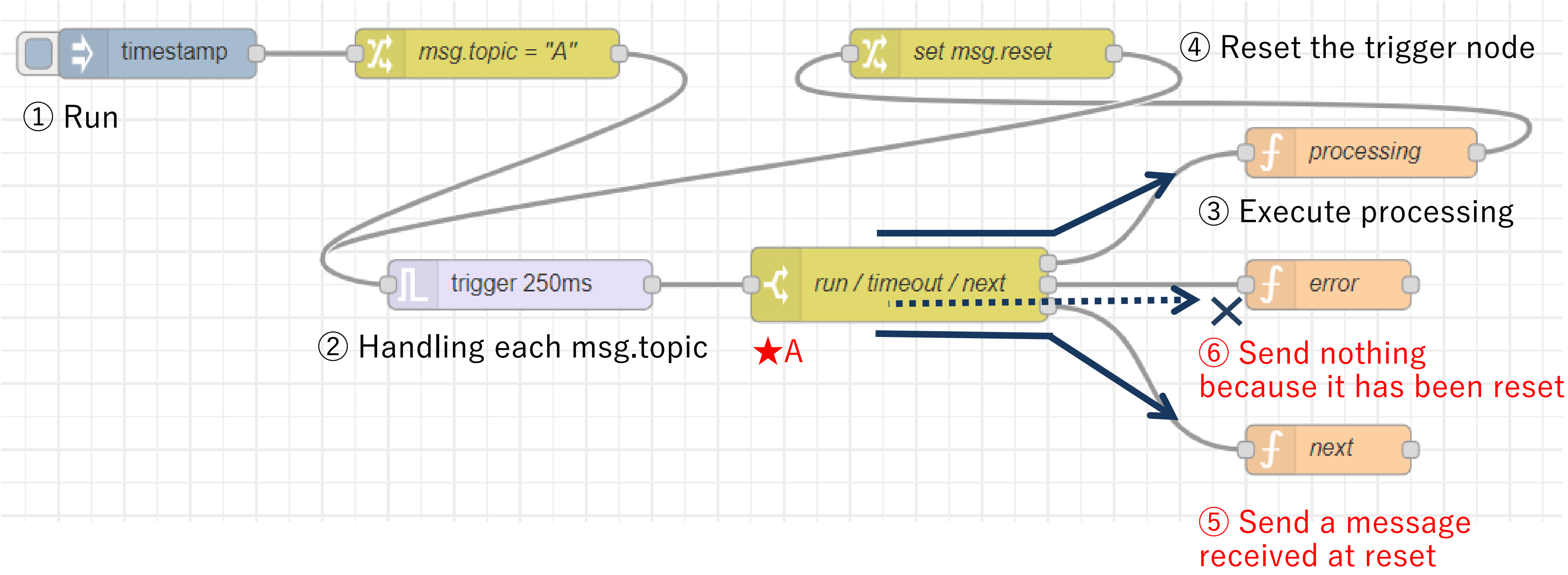
Handling: all messages

Name: Name

[Plan2]

- Create timeout flow with *Trigger Node and Switch Node*
  - Add “send a message if a reset message arrives” option

Success pattern

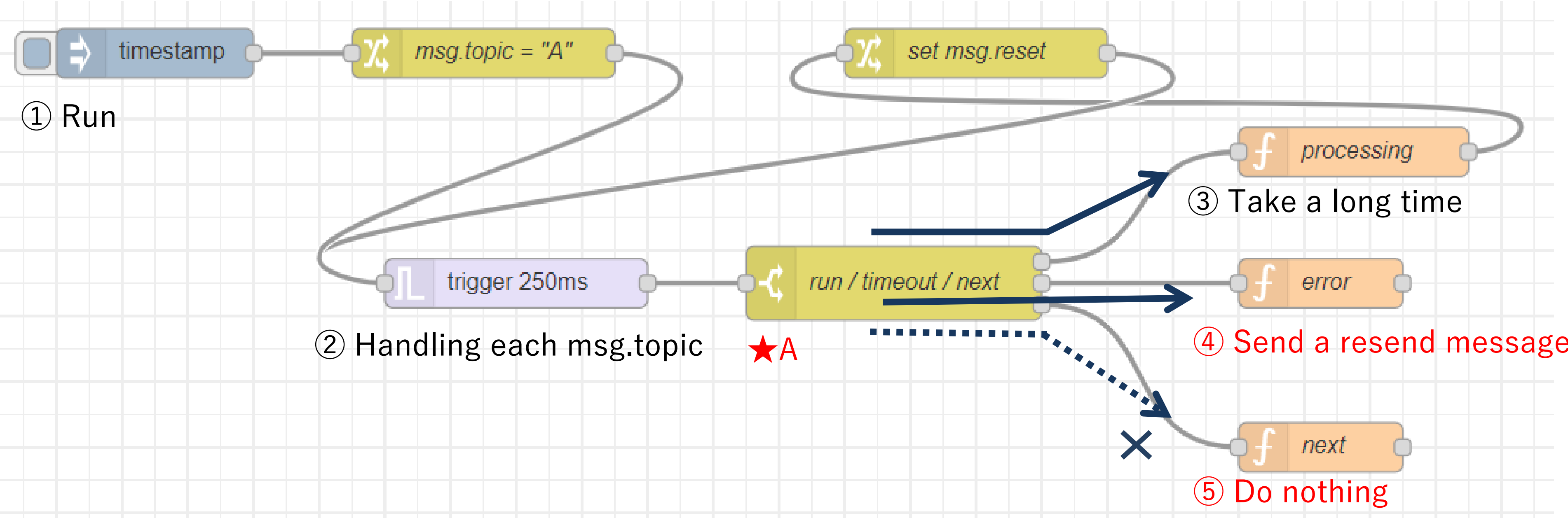


★A　Determine whether it is the first time, resend message, or reset message

[Plan2]

- Create timeout flow with *Trigger Node and Switch Node*
  - Add “send a message if a reset message arrives” option

Timeout pattern



① Run

② Handling each msg.topic

③ Take a long time

★A

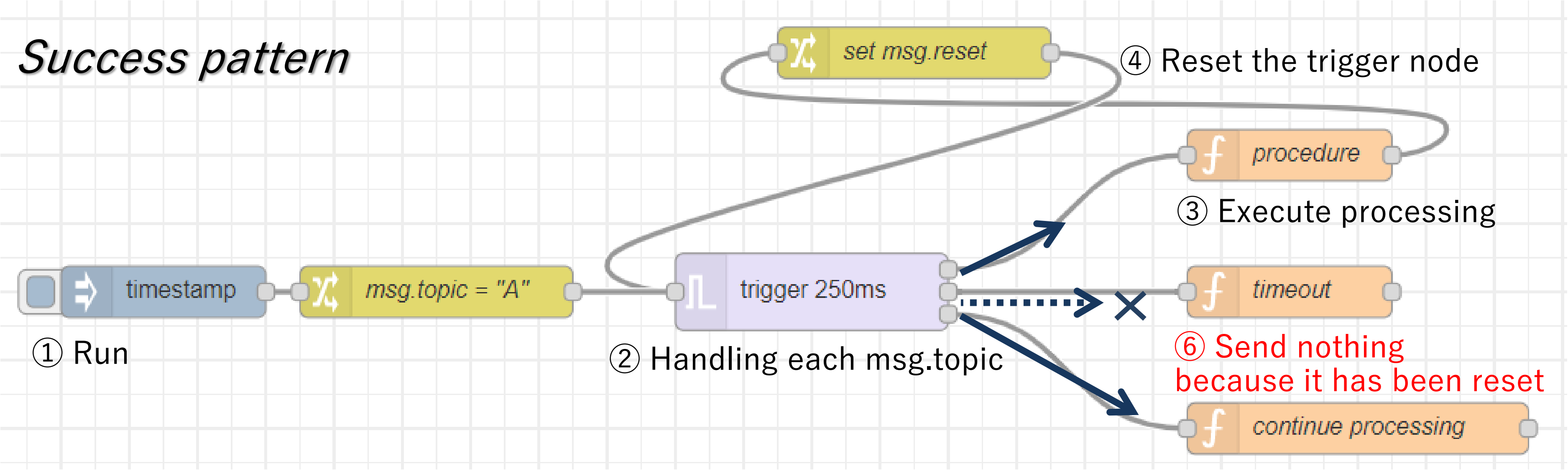
④ Send a resend message

⑤ Do nothing

★A Determine whether it is the first time, resend message, or reset message

[Plan3]

- Create timeout flow with *Trigger Node* only
  - ➢ Add “discrete outputs” & “send a message if a reset message arrives” option
    - Send a resend message to <u>second</u> port after waiting
    - Send a message to <u>third</u> port if a reset message arrives



Success pattern

① Run

② Handling each msg.topic

③ Execute processing

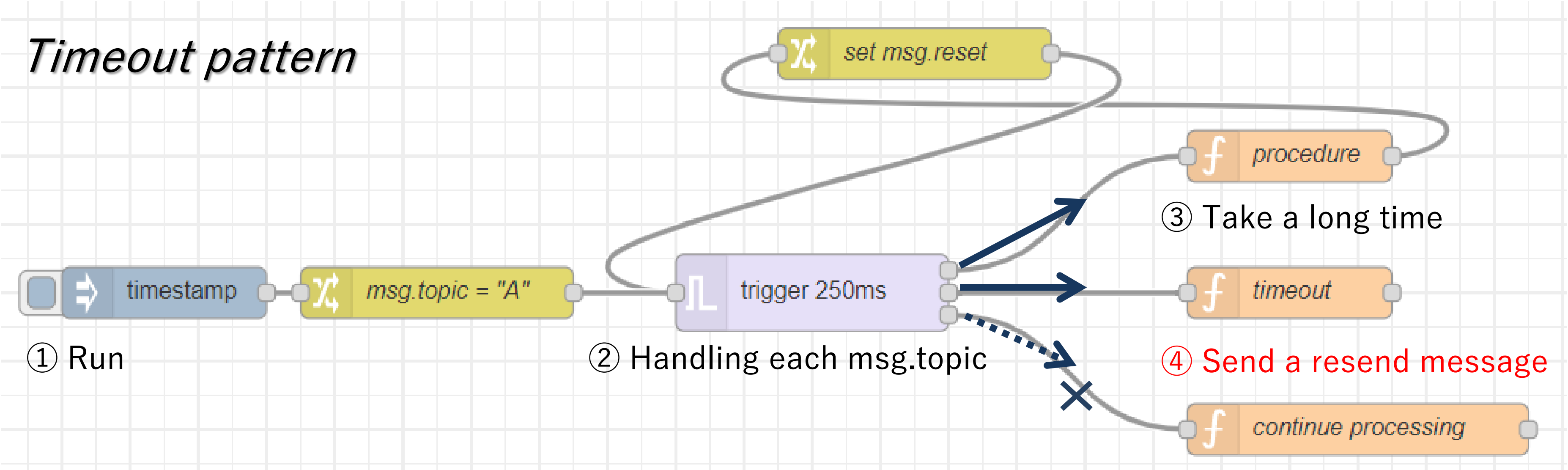
④ Reset the trigger node

⑤ Send a message received at reset

⑥ Send nothing because it has been reset

[Plan3]

- Create timeout flow with *Trigger Node* only
  - Add “discrete outputs” & “send a message if a reset message arrives” option



① Run

② Handling each msg.topic

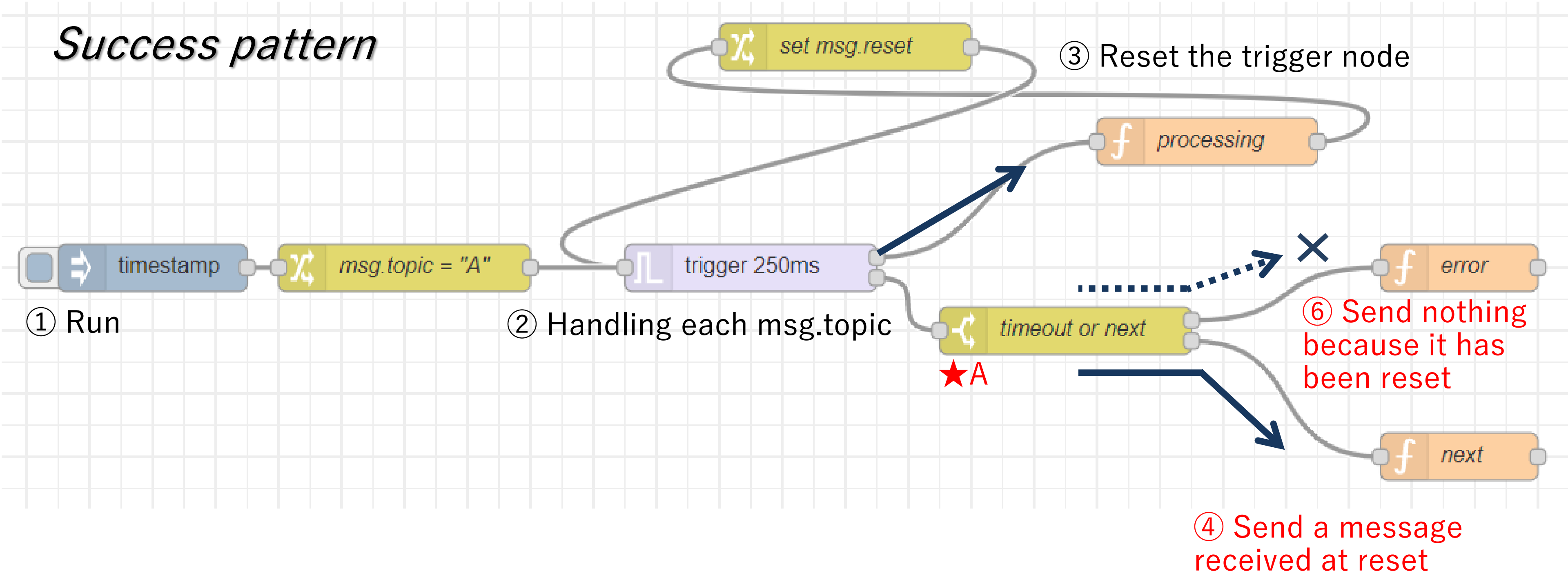
③ Take a long time

④ Send a resend message

⑤ Do nothing

[Supplement] Two output ports

- Create timeout flow with *Trigger Node and Switch Node*
  - ➢ Add “discrete outputs” & “send a message if a reset message arrives” option
    - Send a resend message to second port after waiting
    - Send a message to second port if a reset message arrives



Success pattern

① Run

② Handling each msg.topic

③ Reset the trigger node

④ Send a message received at reset

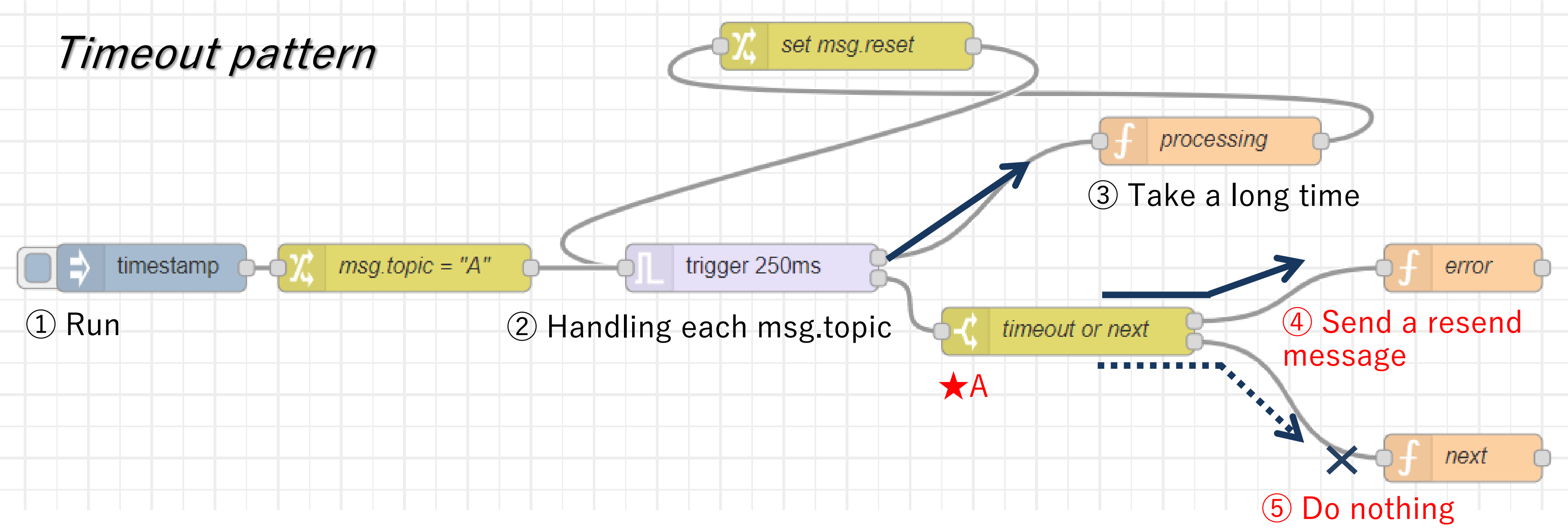
⑥ Send nothing because it has been reset

★A

★A Determine whether it is a resend or reset message

[Supplement] Two output ports

- Create timeout flow with *Trigger Node and Switch Node*
  - Add “discrete outputs” & “send a message if a reset message arrives” option
    - Send a resend message to second port after waiting
    - Send a message to second port if a reset message arrives



★A Determine whether it is a resend or reset message

HITACHI

Inspire the Next

Current Status

- Design: https://github.com/node-red/designs/pull/14/files
- Draft PR: https://github.com/node-red/node-red/pull/2377

[Plan1] Enhancements of “join” node

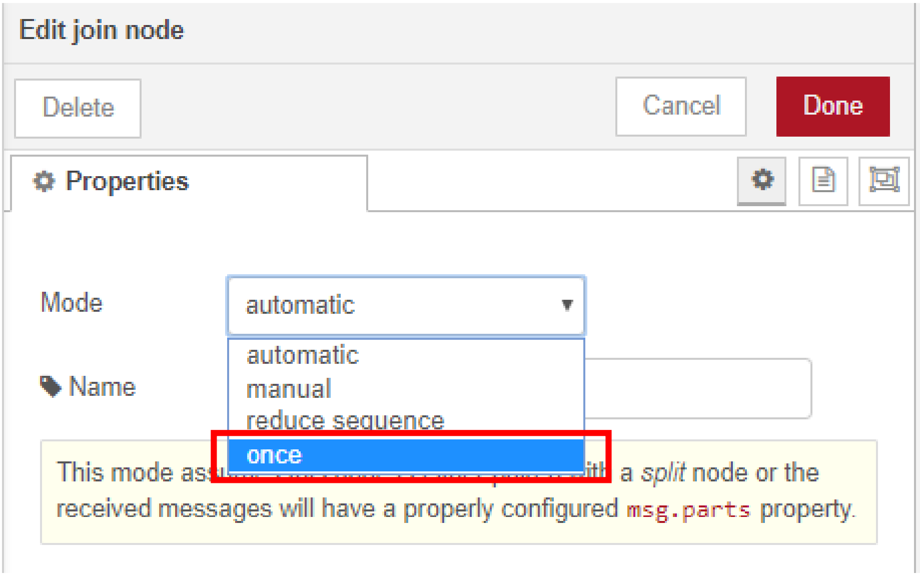
HITACHI
Inspire the Next

A "join" node waits for multiple messages and combines them.

Adding an option on the node, this node behave an opposite action.

The node with **“once”** option will pass through a message to the next node only once.

After that, the node will ignore the same message.



Mode: automatic

Name

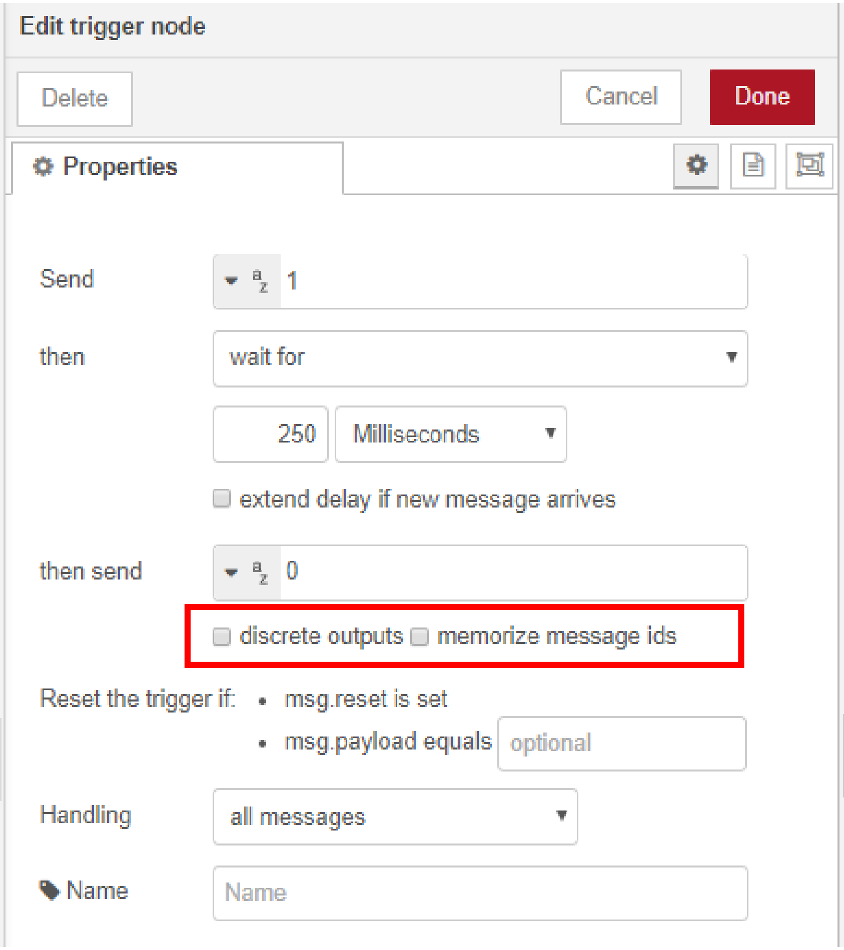
This mode as… …ith a *split* node or the received messages will have a properly configured `msg.parts` property.

[Plan1]Enhancements of “trigger” node

A "trigger" node pass a message to the next node, and wait for the defined time to send a message again.

Adding options, which are **"discrete outputs"** and **"memorize message ids",** on the node, this node sends 2nd message to the other wired node, and memorise the message id in the flow context.

These options can be selected when "with for" is selected in the "then" property.



Send: 1

then: wait for

250 Milliseconds

□ extend delay if new message arrives

then send: 0

□ discrete outputs □ memorize message ids

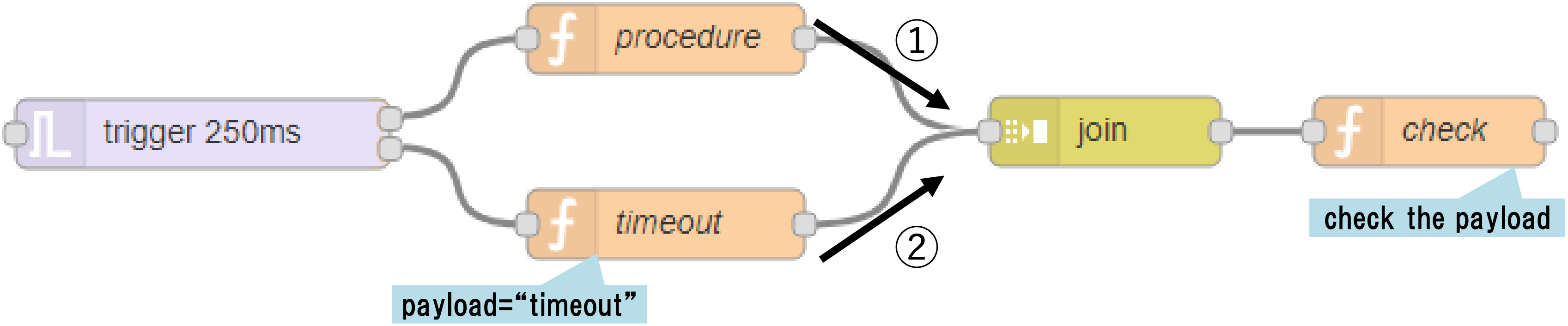
Reset the trigger if:
- msg.reset is set
- msg.payload equals optional

Handling: all messages

Name: Name

[Plan1]Example

In this case, the "trigger" node sends a message to the "procedure" node, and keeps its id in the flow context.



①The "procedure" node sends the message to the "join" node.
If the "join" node could find its id in the flow context, the message would be sent to the "check" node and the "join" node would delete the id in the flow context.

②After 250ms, the "trigger" node sends the same message to the "timeout" node, and the "timeout" node sends it to the "join" node. The "join" node cannot find the message id because the id was already deleted. The message will be ignored.

If the "procedure" node takes time, the "join" node sends a message from the "timeout" node to the "check" node. The "check" node checks where the message comes from. After that the "check" node will be able to deal with a timeout error.