delay node enhancements

HITACHI
Inspire the Next

- Use case1: Incoming message control
  Because Node-RED accepts all incoming message from input nodes like mqtt-in node, Node.js process sometimes reaches the upper limit of memory usage. To avoid the risk of message queue overflow, rate limit functionality in delay node will be useful but Node-RED users cannot customize queue handling. For example, the current delay node cannot simultaneously use the drop option and queue to process burst incoming messages.

- Use case 2: Changing properties on demand
  Currently, the delay node supports msg.delay to change property value on demand. If the delay node additionally supports msg.rate to set rate value and msg.queueLength to change queue size, users can create custom delay node using subflow to wrap delay node.

- Use case 3: Logging and error handling
  Because current drop option in rate limit mode doesn't record dropped messages, Node-RED users aren't aware of the lost messages. Therefore, it will be useful for Node-RED users to track background behaviors if the delay node outputs the number of dropped messages to the log stream. Additionally, Node-RED users can add their error handling in their flow if the delay node throws the dropped messages to the catch node when the queue reaches the defined limit.

Options in setting.js which the delay node uses

HITACHI
Inspire the Next

- nodeMessageBufferMaxLength
  The enhanced delay node supports nodeMessageBufferMaxLength as the default buffer length. The buffer length needs to be greater than queue length defined in msg.queueLength or node property UI.

- customQueueLength (default: false)
  When the customQueueLength value is true, the node property UI of the enhanced delay node shows customQueueLength as the text input. The default value is 0 because the current implementation doesn't use the queue.

- throwDroppedMessages (default: false)
  Other nodes which support nodeMessageBufferMaxLength setting throws an empty message to the catch node when they reach the limit of the buffer length. As the default, the behavior of enhanced delay node will be the same as existing other nodes. If the throwDroppedMessages in setting.js is true, the enhanced delay node will throw dropped messages to the catch node. If the Node-RED supports this handling, Node-RED users are able to write their error handling in the flow.

Logging

HITACHI
Inspire the Next

- The number of dropped messages
  - debug level: Node-RED outputs the node ID and the number of dropped messages to the log stream every 15 seconds.
  - trace level: In addition to node ID and the number of dropped messages, Node-RED writes message ID of dropped messages to the log stream every 15 seconds.

- Queue usage rate
  - debug level: Node-RED outputs the node ID and the size of the message queue every 15 seconds.

Properties to change a property on demand

HITACHI
Inspire the Next

- msg.rate
  The enhanced delay node overwrites the existing rate value defined in the node property UI when it receives the message which contains msg.rate value.

- msg.queueLength
  The enhanced delay node overwrites queue length according to msg.queueLength value which received message has. If the number of messages in the queue is greater than the queue length in the msg.queueLength, the last messages will be deleted.

Properties

HITACHI
Inspire the Next

The enhanced delay node newly has queueLength as a node property. The values in the following table are default values.

#	Action			pause Type	Properties which Node-RED users use on node property UI									
					timeout	timeout Units	rate	nbRate Units	rate Units	random First	random Last	random Units	drop	queueLength
1	Delay each Message	Fixed delay	-	delay	5	seconds								
2		Random delay	-	random						1	5	seconds		
3		Override delay with msg.delay	-	delayv	5	seconds								
4	Rate limit	All messages	-	rate			1	1	second				false	0
5		For each msg.topic	Send each topic in turn	queue			1	1	second					
6			Send all topics	timed			1	1	second					