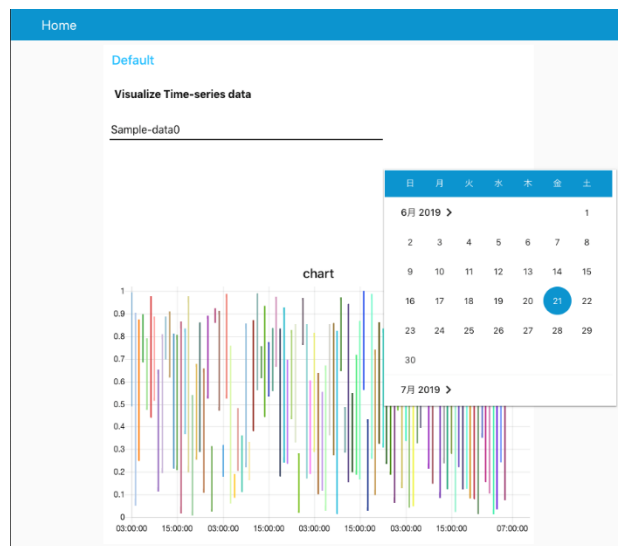




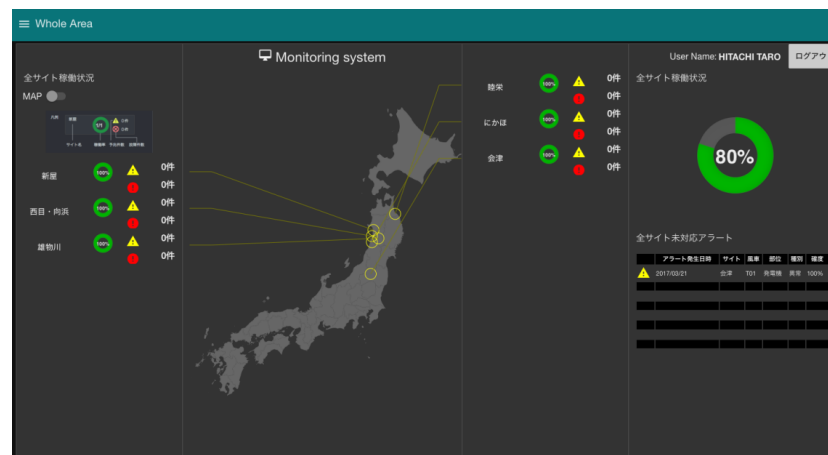
Dashboard Enhancements

Hiroyasu Nishiyama

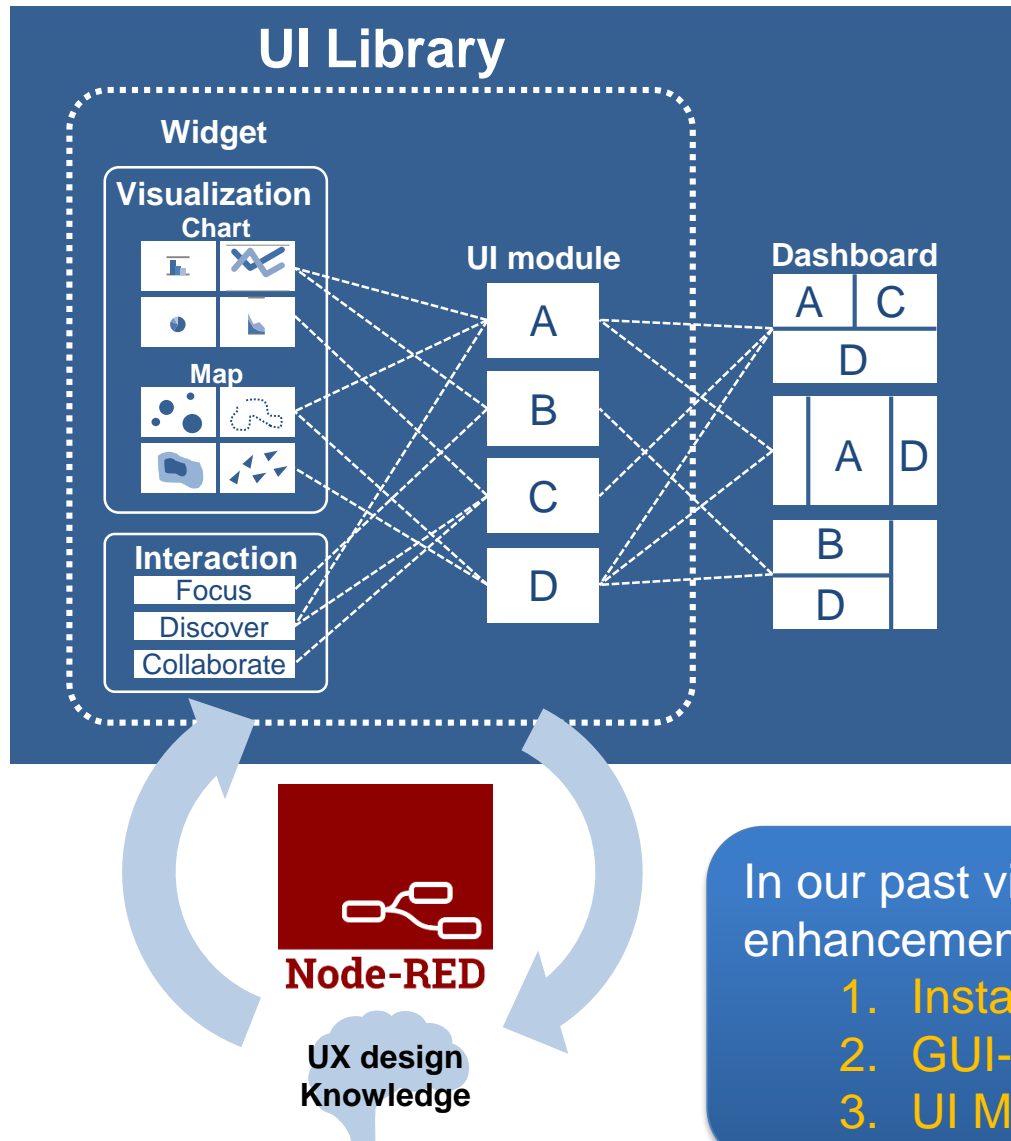
- ❑ Graphical representation of data is effective tool for sharing knowledge among people.
- ❑ Node-RED Dashboard is convenient for creating GUI.
- ❑ But creating complex dashboard is difficult with current Node-RED.



Simple
Dashboard



Complex
Dashboard



- **Widgets**

Basic design elements that use charts and maps to represent data.

- **UI modules**

Visual components made up of multiple elements.

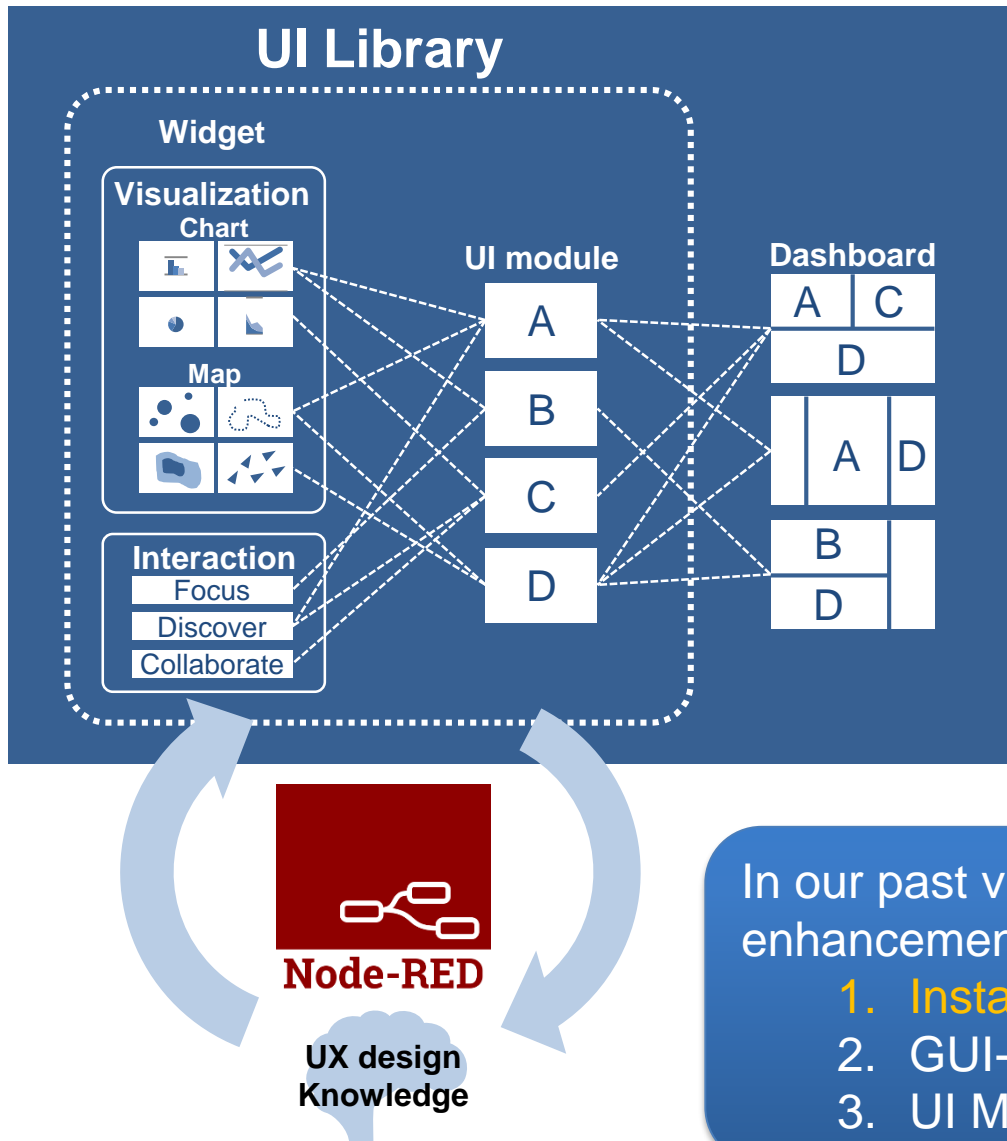
Each UI module is a group of widgets that meets a particular need in terms of the information it conveys or how it is viewed.

- **Dashboards**

Dashboards that combine a number of UI modules in a predetermined layout.

In our past visit to Hursley, we discussed following enhancements on Node-RED Dashboard:

1. Installable Widgets (design elements)
2. GUI-based Layout Editing
3. UI Module (Compound Widgets)



- **Widgets**

Basic design elements that use charts and maps to represent data.

- **UI modules**

Visual components made up of multiple elements.

Each UI module is a group of widgets that meets a particular need in terms of the information it conveys or how it is viewed.

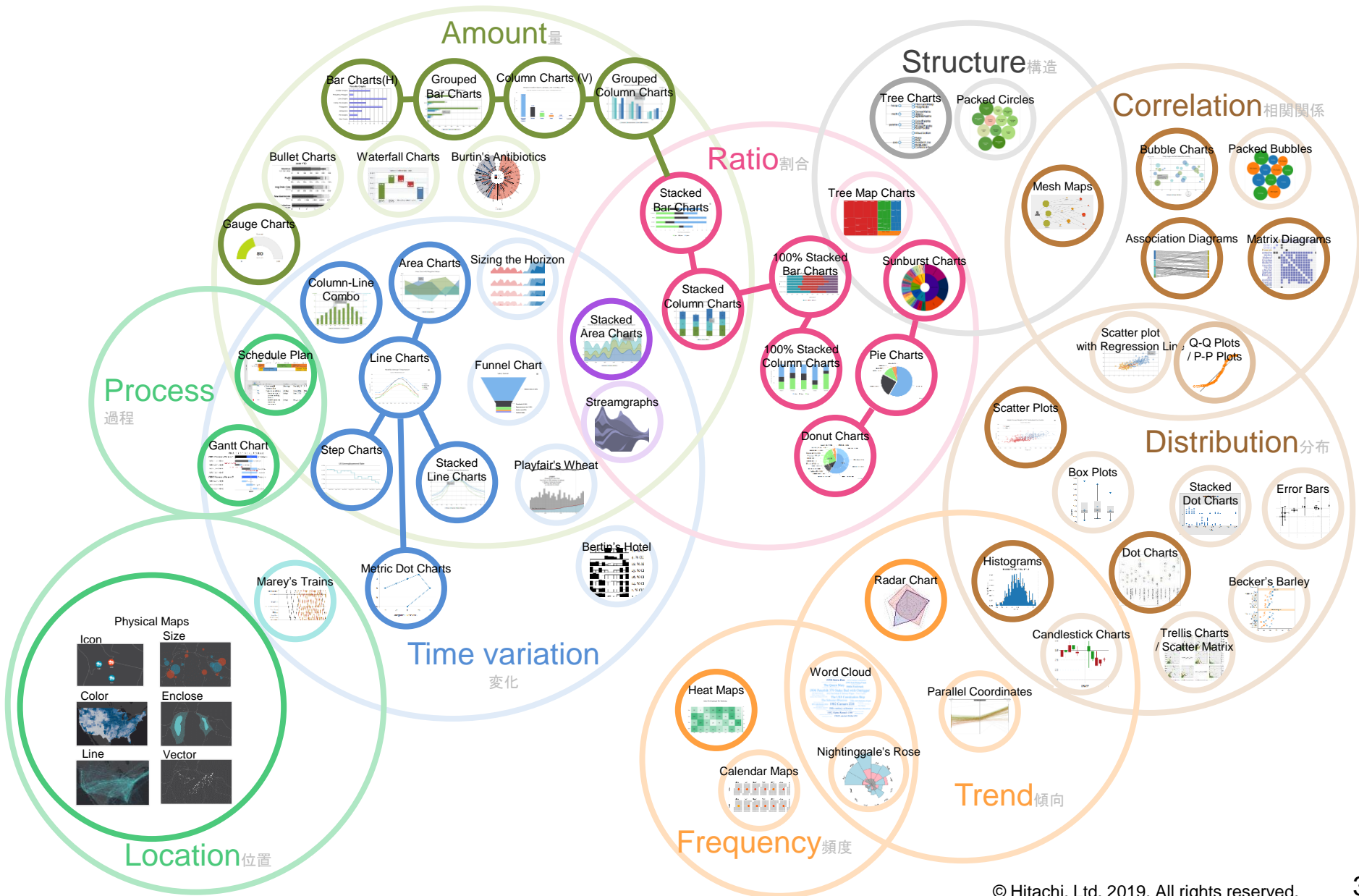
- **Dashboards**

Dashboards that combine a number of UI modules in a predetermined layout.

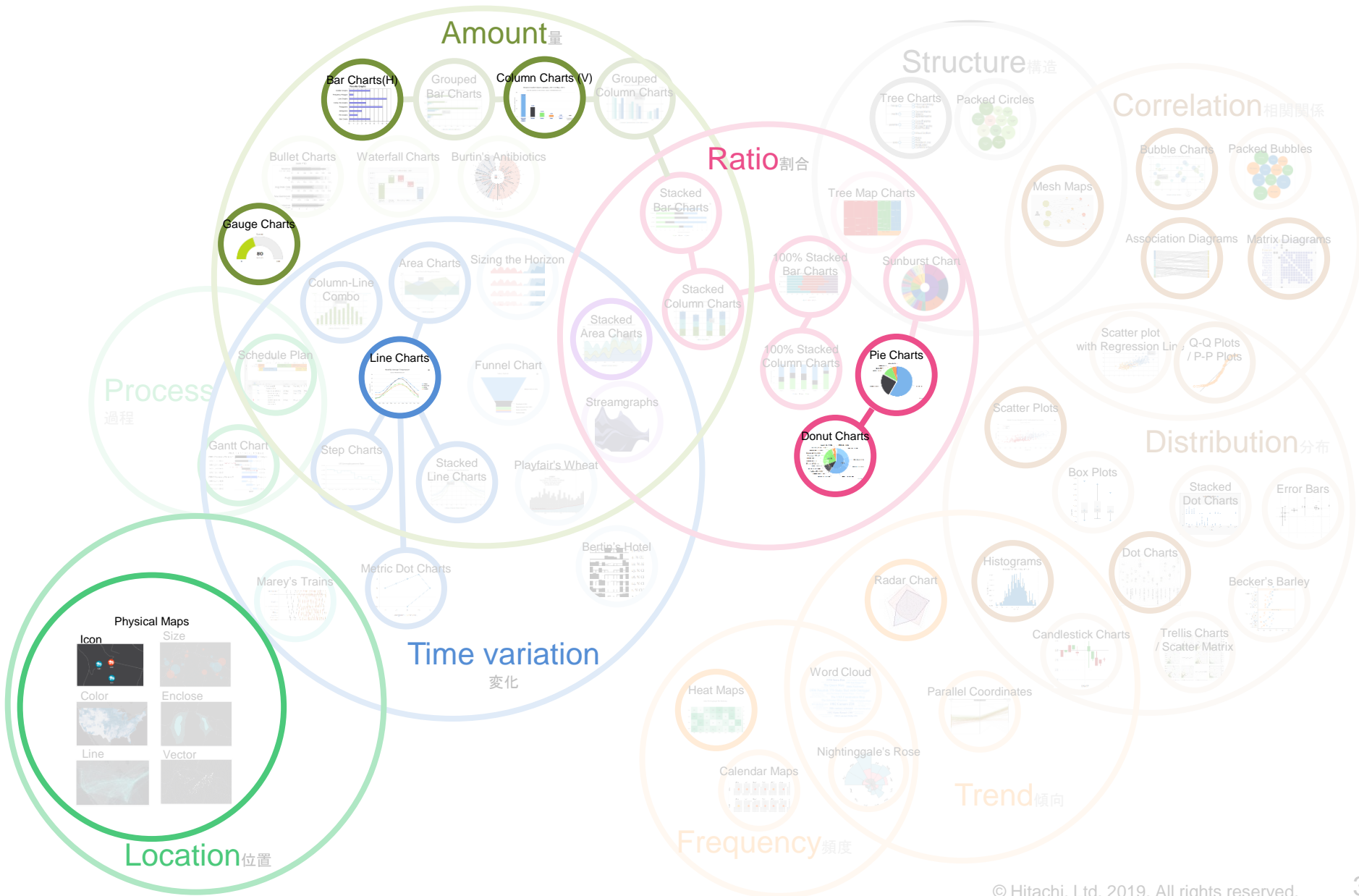
In our past visit to Hursley, we discussed following enhancements on Node-RED Dashboard:

1. Installable Widgets (design elements)
2. GUI-based Layout Editing
3. UI Module (Compound Widgets)

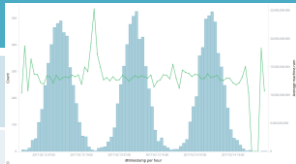
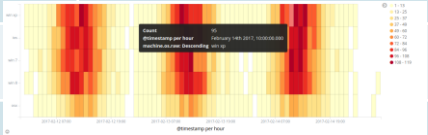
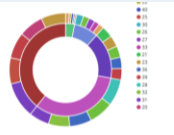
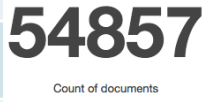
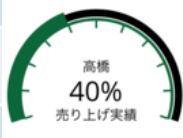

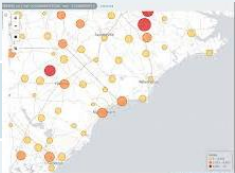


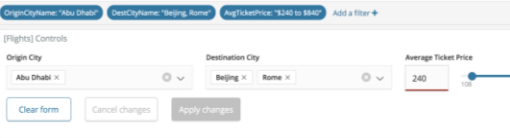


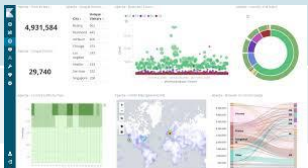
Visualization types of Widgets



Visualization types on Node-RED

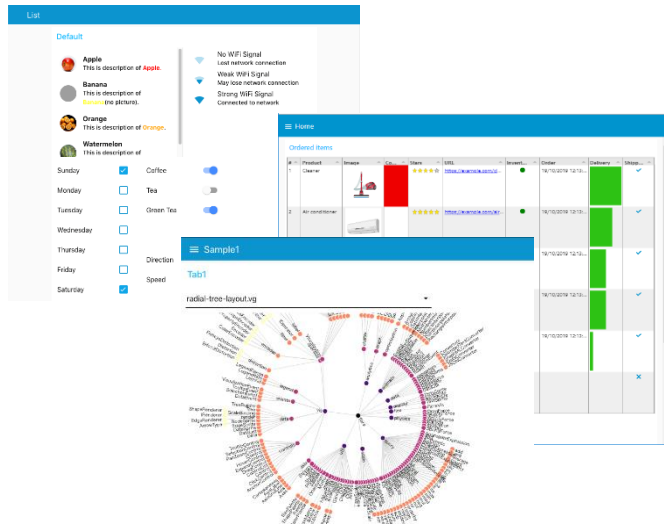


Kibana & Node-RED Widgets

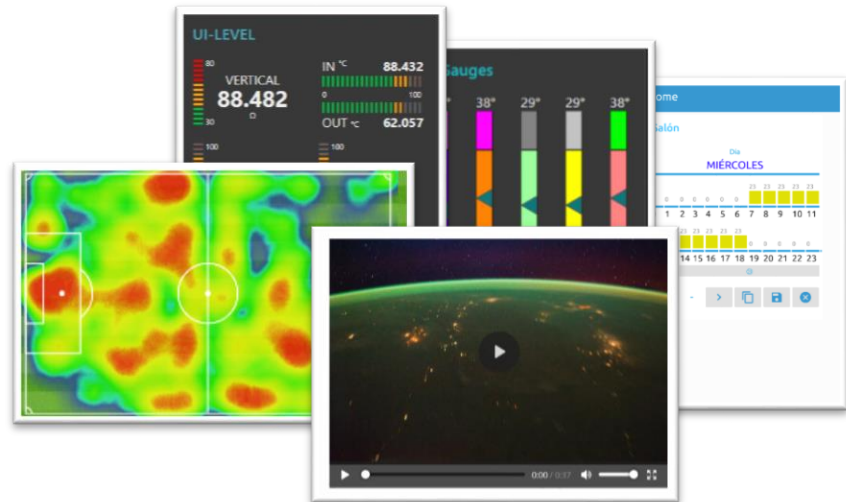
No.	Type	Kibana	Node-RED																				
1	Line, Area, Bar chart		△																				
2	Heat Map		×																				
3	Pie Chart		△																				
4	Data Table	<table border="1"> <thead> <tr> <th>user_age_ranges ▾</th><th>Top 5 user_sex ▾</th><th>Count of documents ▾</th><th>Unique count of user_country ▾</th></tr> </thead> <tbody> <tr> <td>0.0-30.0</td><td>m</td><td>15383</td><td>182</td></tr> <tr> <td>0.0-30.0</td><td>f</td><td>9942</td><td>178</td></tr> <tr> <td>30.0-999.0</td><td>m</td><td>17479</td><td>180</td></tr> <tr> <td>30.0-999.0</td><td>f</td><td>12053</td><td>176</td></tr> </tbody> </table>	user_age_ranges ▾	Top 5 user_sex ▾	Count of documents ▾	Unique count of user_country ▾	0.0-30.0	m	15383	182	0.0-30.0	f	9942	178	30.0-999.0	m	17479	180	30.0-999.0	f	12053	176	×
user_age_ranges ▾	Top 5 user_sex ▾	Count of documents ▾	Unique count of user_country ▾																				
0.0-30.0	m	15383	182																				
0.0-30.0	f	9942	178																				
30.0-999.0	m	17479	180																				
30.0-999.0	f	12053	176																				
5	Metric		×																				
6	Goal, Gauge		△																				
7	Coordinate Map		×																				
8	Region Map		×																				
9	Timeline		△																				
10	Time Series Visual Builder		△																				
11	Controls		△																				
12	Markdown		×																				
13	Tag Clouds		×																				
14	Vega[experimental]		×																				

[Legend] ○:Equivalent Features, △:Smaller Features, ×:No Feature

- After introduction of new dashboard widget API, new UI widgets has been created by HITACHI and other OSS contributors



UI widgets from
HITACHI



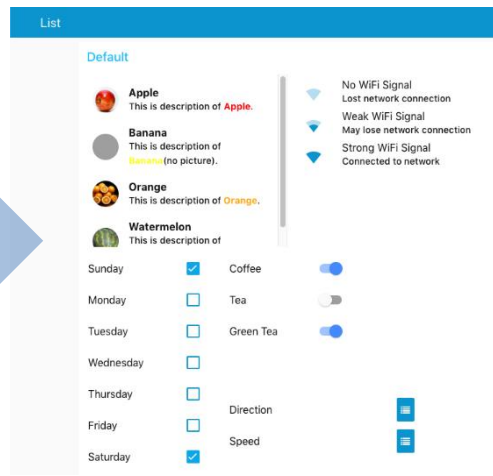
UI widgets from
other OSS contributors

List/Table Node: Simple Data Presentation

- Presenting data in list or table format is commonly used
- Presenting data on Node-RED dashboard can be made easy using List node (ui_list) and table node (ui_table)

```
{
  "$schema": "...",
  "width": 400,
  "height": 200,
  "padding": 5,
  "data": [
    ...
  ],
  "marks": [
    ...
  ],
  ...
}
```

List
Specification



List
Representation

```
{
  "$schema": "...",
  "width": 400,
  "height": 200,
  "padding": 5,
  "data": [
    ...
  ],
  "marks": [
    ...
  ],
  ...
}
```

Table
Specification

#	Product	Image	Color	Stars	URL	Inventory	Order	Delivery	Shipper
1	Cleaner		Red	★★★★★	https://www.example.com/cleaner	Green	18/10/2019 12:13...	Green	✓
2	Air conditioner		White	★★★★★	https://www.example.com/air-conditioner	Green	18/10/2019 12:13...	Green	✓
3	Toaster		Black	★★★★★	https://www.example.com/toaster	Green	18/10/2019 12:13...	Green	✓
4	Dryer		Blue	★★★★★	https://www.example.com/dryer	Green	18/10/2019 12:13...	Green	✓
5	Washing mach...		Orange	★★★★★	https://www.example.com/washing-machine	Yellow	18/10/2019 12:13...	Green	✓
6	Refrigerator		Brown	★★★★★	https://www.example.com/refrigerator	Red	18/10/2019 12:13...	Green	✗

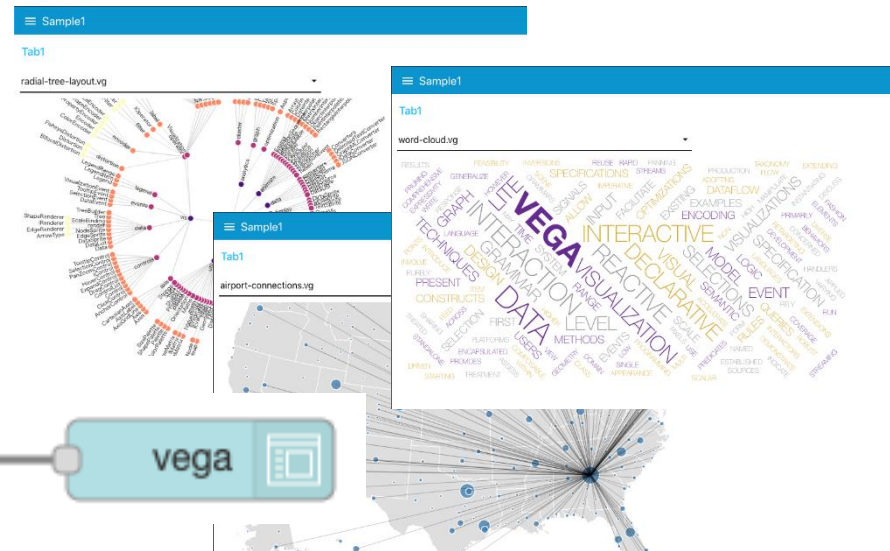
Table
Representation

Vega Node: Complex Data Visualization

- ❑ Vega* is a visualization grammar for interactive visualization design
 - ❑ Vega node (ui_vega) allows Vega-based visualization description presented on Node-RED Dashboard
- Vega supports various kinds of data visualization that can be described in JSON.
 - Naturally fits with JSON based message processing in Node-RED.

```
{
  "$schema": "...",
  "width": 400,
  "height": 200,
  "padding": 5,
  "data": [
    ...
  ],
  "marks": [
    ...
  ],
  ...
}
```

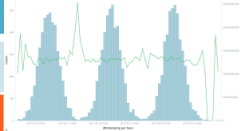
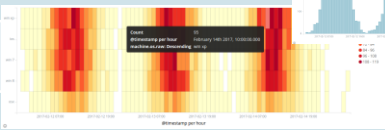
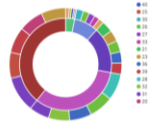
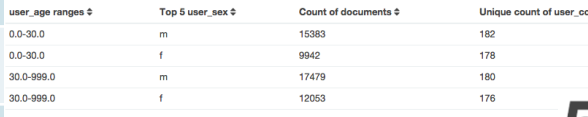
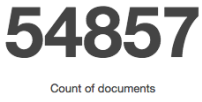

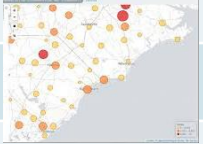

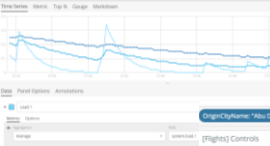

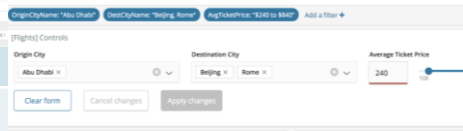



Vega JSON
Specification



Data Visualization
on Node-RED Dashboard

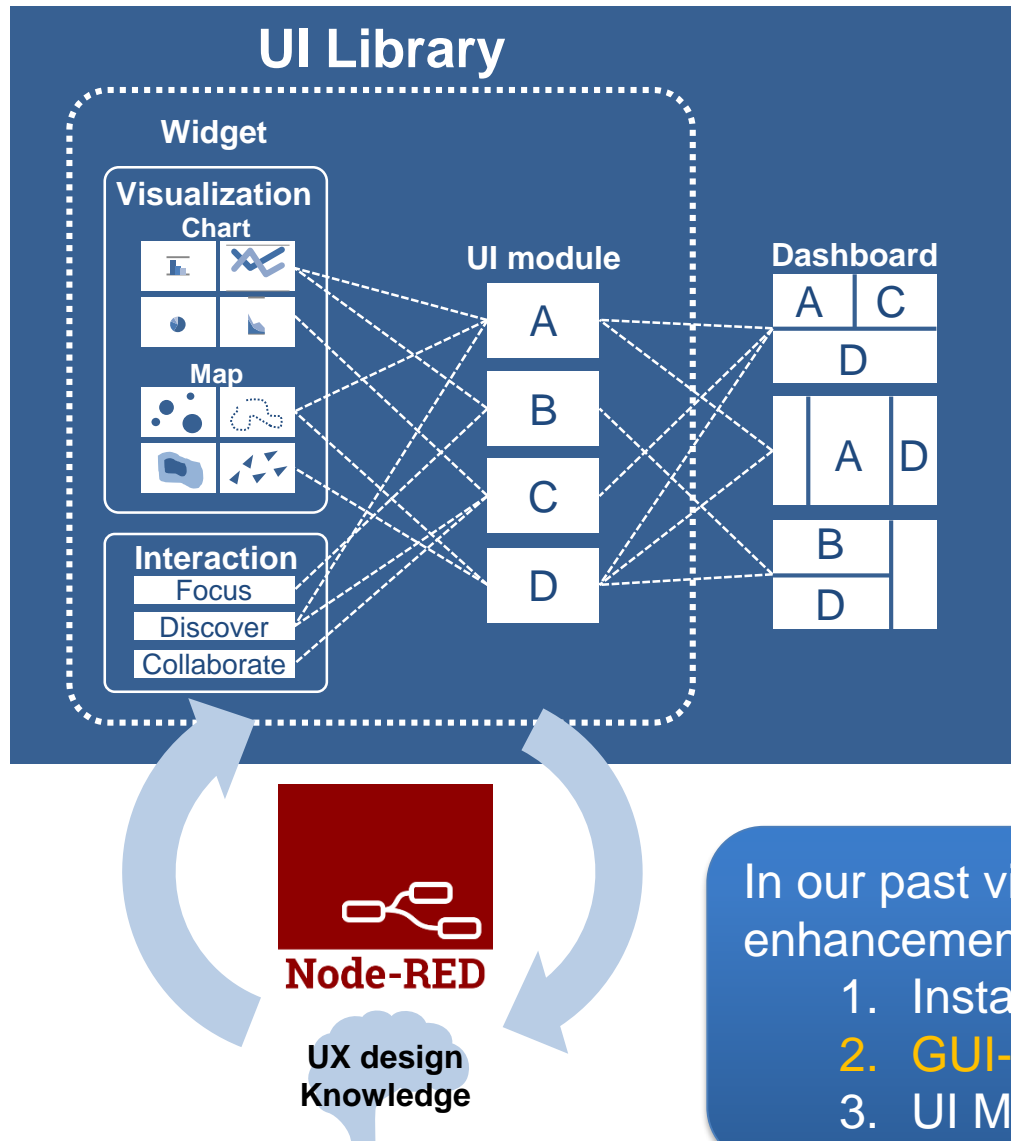
*: Vega: <https://vega.github.io/vega/>

Appendix: Kibana & Node-RED Widgets

No.	Type	Kibana	Node-RED
1	Line, Area, Bar chart		△ ○
2	Heat Map		× ○
3	Pie Chart		△ △
4	Data Table		× ○
5	Metric		× ×
6	Goal, Gauge		△ △
7	Coordinate Map		× △
8	Region Map		× △
9	Timeline		△ △
10	Time Series Visual Builder		△ △
11	Controls		△ △
12	Markdown		× ×
13	Tag Clouds		× ○
14	Vega[experimental]		× ○

with new widgets

[Legend] ○:Equivalent Features, △:Smaller Features, ×:No Feature



- **Widgets**

Basic design elements that use charts and maps to represent data.

- **UI modules**

Visual components made up of multiple elements.

Each UI module is a group of widgets that meets a particular need in terms of the information it conveys or how it is viewed.

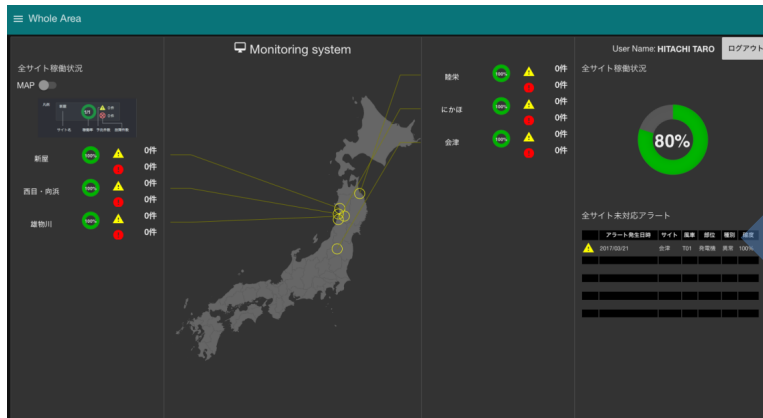
- **Dashboards**

Dashboards that combine a number of UI modules in a predetermined layout.

In our past visit to Hursley, we discussed following enhancements on Node-RED Dashboard:

1. Installable Widgets (design elements)
2. **GUI-based Layout Editing**
3. UI Module (Compound Widgets)

- Node-RED Dashboard is convenient tool for creating

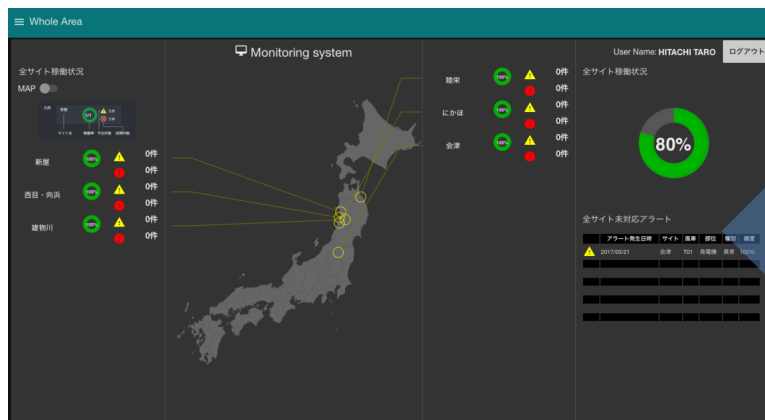


Example of
Complex Dashboard

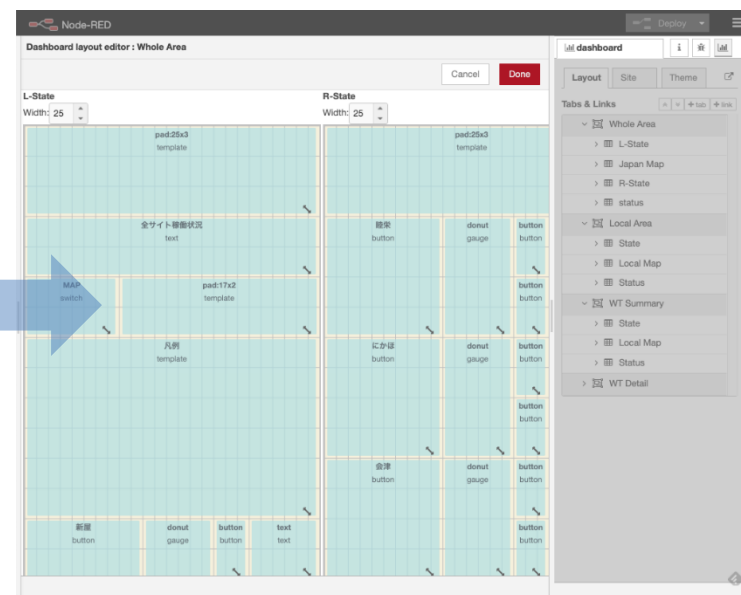


Layout of Widgets using
Dashboard Layout Tab

- ❑ With new Dashboard Layout Editor, we can intuitively create dashboard layout via GUI based tool
- ❑ Can place and resize UI widgets within dashboard

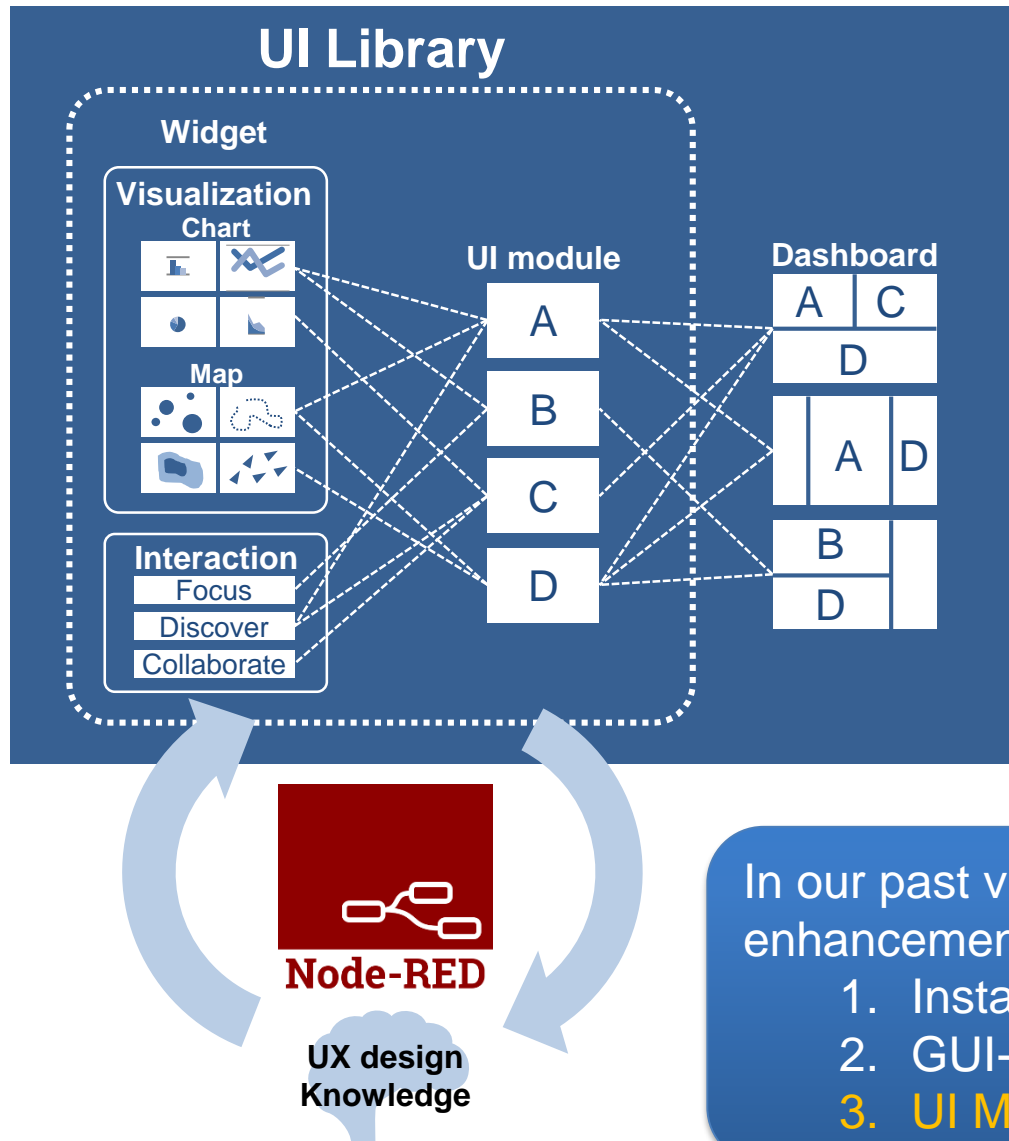


Example of
Complex Dashboard



Layout of Widgets using
Dashboard Layout Tab

DEMO



- **Widgets**

Basic design elements that use charts and maps to represent data.

- **UI modules**

Visual components made up of multiple elements.

Each UI module is a group of widgets that meets a particular need in terms of the information it conveys or how it is viewed.

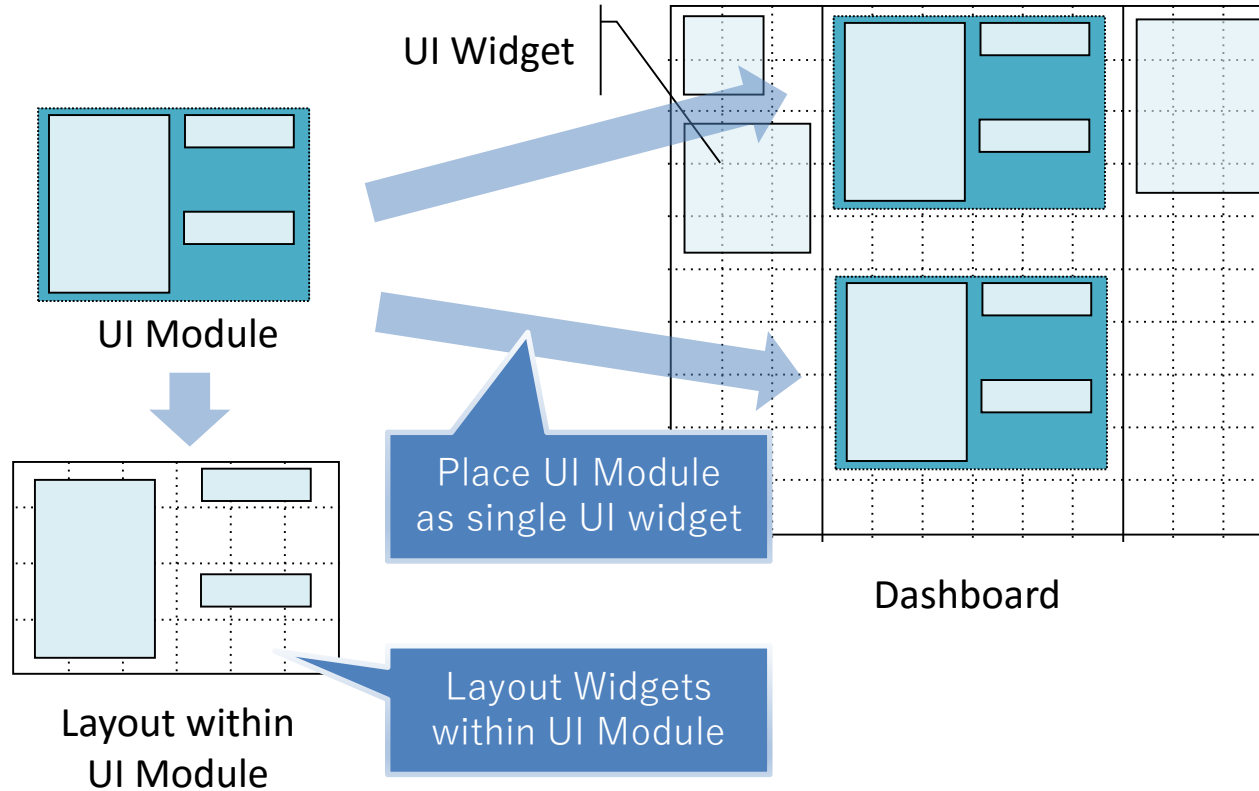
- **Dashboards**

Dashboards that combine a number of UI modules in a predetermined layout.

In our past visit to Hursley, we discussed following enhancements on Node-RED Dashboard:

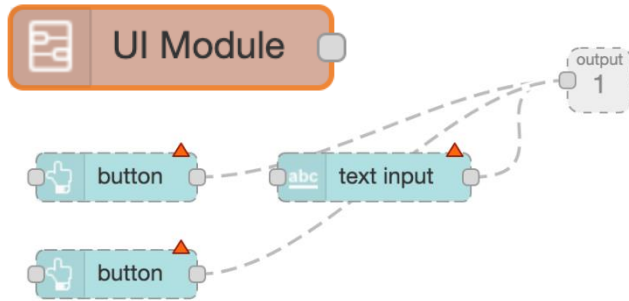
1. Installable Widgets (design elements)
2. GUI-based Layout Editing
3. **UI Module (Compound Widgets)**

- ❑ UI module consists of a set of UI widgets. It has its own internal layout of containing widgets.
- ❑ UI modules can be placed on dashboard similar to UI widgets

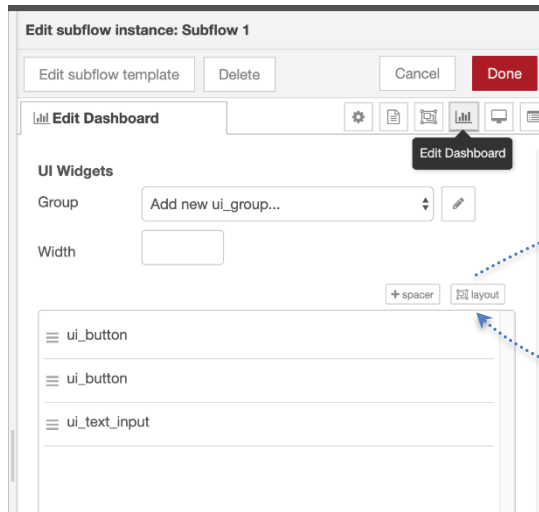


Proposal: SUBFLOW as UI Module

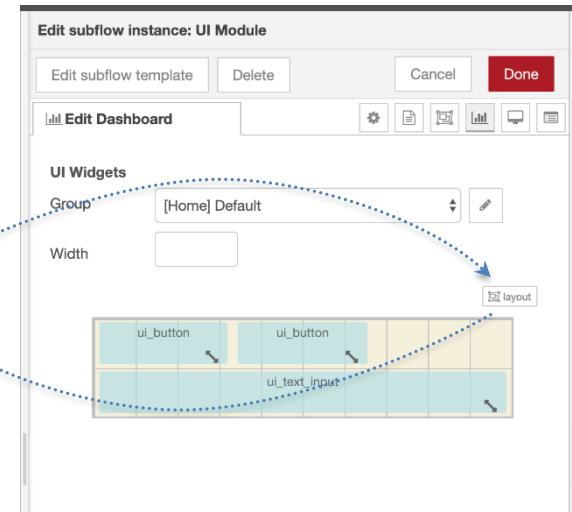
- ❑ Since UI Module consists of a set of nodes, using SUBFLOW as UI Module is natural extension
- ❑ Add interface to specify internal layout of SUBFLOW
 - UI Module (SUBFLOW) has "Edit Dashboard" Tab in settings panel.
 - It can specify group, width, and layout.
 - Layout can be switched between order-based layout and GUI-based layout (toggle by layout button).



UI Module (SUBFLOW)
containing 3 widgets



(a) order mode

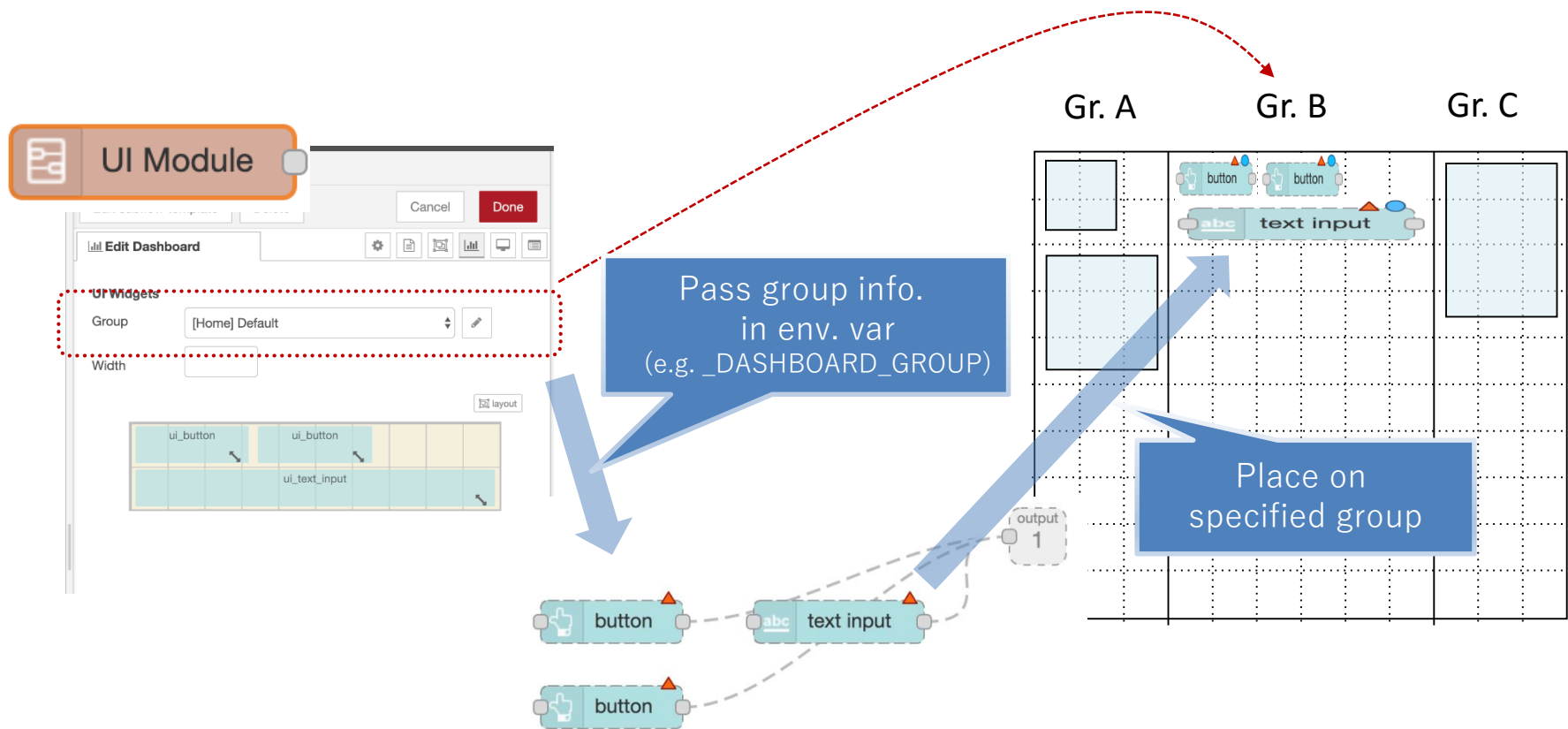


(b) layout mode

Settings Panel of UI Module

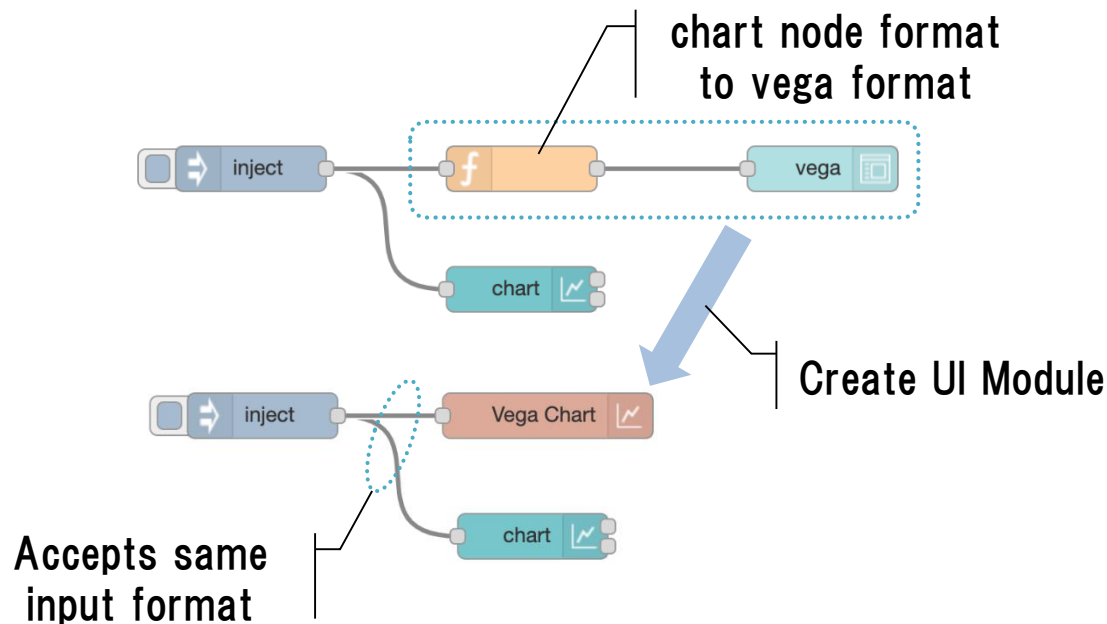
Layout in Dashboard of UI Module

- ❑ UI Module appear as a widget in dashboard layout tab & tool.
- ❑ Group specification of widgets within UI Module is ignored but UI Module's group is passed by environment variable.



Example Usage: Specializing Vega Node

- By using SUBFLOW as UI Module, we can include some logic in UI Module
- One example of this usage is specialization of Vega node
 - Vega node accepts complex visualization specification in JSON
 - The JSON specification is sometimes difficult to write
 - Conversion from light-weight format to Vega specification is useful



DEMO

- In this session, we discussed Node-RED Dashboard enhancements:
 - UI Widgets,
 - GUI based layout editor,
 - UI Module
- We would like to know your opinion on current proposal and implementation

HITACHI
Inspire the Next



Dashboard layout

Kazuhiro Ito

■ Summary

**We are considering adding the function of dashboard layout.
See below.**

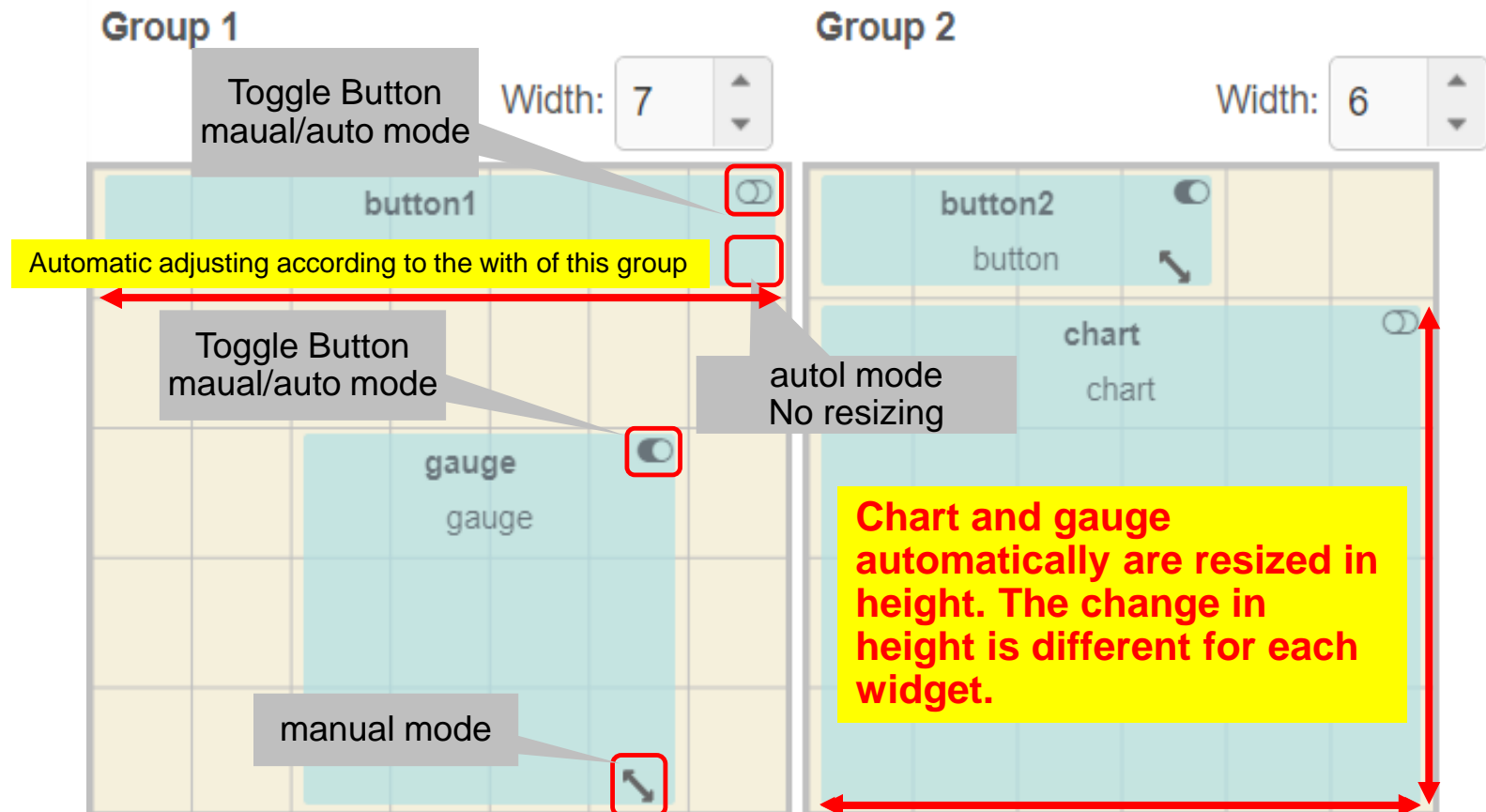
■ Functions

- 1. Support automatic mode for widget size**
- 2. Support layout for sub-flow**

■ Function

Support automatic mode for widget size.

In adding to current manual resizing, support the function to resize widget automatically.



■ Problems

There are widgets whose height is resized according to the group width and the setting of the node. The dealing with the height of standard node of dashboard and the height for each node-red-ui-nodes widget is as follows.

#	Widget	Height	Dealing with Layout
1	ui_gauge	The height is determined by its width. It depends on its Type (Gauge, Donut / Compass, Level)	Support: depends on its type and width.
2	ui_chart	The height is determined by its width. No difference between Type(Line chart, Bar chart, Bar chart(H) and Pie chart, Poler area chart, Radar chart).	Support: depends on its width
3	ui_form	The height is determined by its condition such as label existence or number of sub-widgets. In the manual mode, you cannot define the height.	Support: depends on node setting.
4	ui_lineargauge	The height is determined by its condition such as unit or name existence.	Support: depends on node setting.
5	ui_list, ui_vega	Height = 5	Always 5
6	ui_table	The height is changed by data input. In the no data input case, it is 1.	Not Support layout error by input.
7	Other ui_xxxxx	Height = 1	Always 1

■ Measure

Layout tool supported each node's height.

Supported adjusting heights by checking the standard nodes of dashboard, or node-red-ui-nodes.

■ Improvement

Support API which returns each height of widget in automatic mode.

(Such as RED.nodes._def.paletteLabel , support API for each widget and, the API returns its height in automatic mode.

example: RED.nodes._def.defaultHeight)

By calling the API from the layout tool, the layout tool obtains its height when a user edits in the automatic mode.

The layout tool supported dealing with height of each node.

Created a function which returns height of each node.

The function calculates height for each widget blow, and returns 1 for unknown widgets.

- ui_guage
- ui_chart
- ui_form
- ui_linearguage
- ui_list
- ui_vega

<node> There are height calculation codes in ui_xxxx.html. However, these do not always same values. Thus, the layout tool deals with heights based on the actual display size.

```
function getDefaultHeight(nodeID, groupWidth) {
  var redNode = RED.nodes.node(nodeID);
  var height = 1;
  if (redNode.type === 'ui_guage') {
    if (redNode.gtype === 'gage') {
      height = Math.round(groupWidth/2)+1;
    } else if (redNode.gtype === 'wave') {
      if (groupWidth < 3) {
        height = 1;
      } else {
        height = Math.round(groupWidth*0.75);
      }
    } else { // donut or compass
      if (groupWidth < 3) {
        height = 1;
      } else if (groupWidth < 11) {
        height = groupWidth - 1;
      } else {
        height = Math.round(groupWidth*0.95);
      }
    }
  } else if (redNode.type === 'ui_chart') {
    height = Math.floor(groupWidth/2)+1;
  } else if (redNode.type === 'ui_form') {
    var optNum = redNode.options.length; // Sub widget number
    if (redNode.label) {
      height = optNum + 2; // Label and Button
    } else {
      height = optNum + 1; // Button only
    }
  } else if (redNode.type === 'ui_lineargauge') {
    if (redNode.unit && redNode.name) {
      height = 5;
    } else {
      height = 4;
    }
  }
  return height;
}
```

■ Function

Support layout for widgets in sub-flows

■ Requirements for sub-flows

- If Dashboard is available and the sub-flow contains a UI widget, specify widget layout for each sub-flow.
- When laying out a widget on the dashboard, replace the sub-flow containing the layout as one UI widget.

■ Specification proposal

Although individual widgets can be resized, subflow definitions (UI Groups) have a layout, so resizing is limited to reproduce the layout correctly. Subflow layout information can be arranged in the layout editor in the same way as widgets, but the following points differ from regular widgets.

- Treat as a group (sub-group) that can be placed in a group.

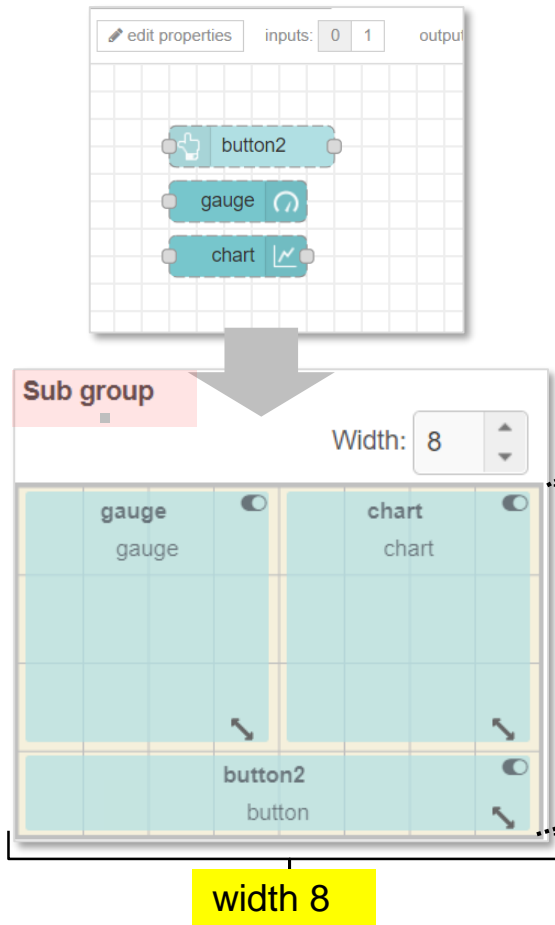
When editing the layout, it can be arranged like a single widget, but it will be expanded and reflected in the widgets in the subgroup when the dashboard is displayed.

- The width and height of subgroups can not be changed.

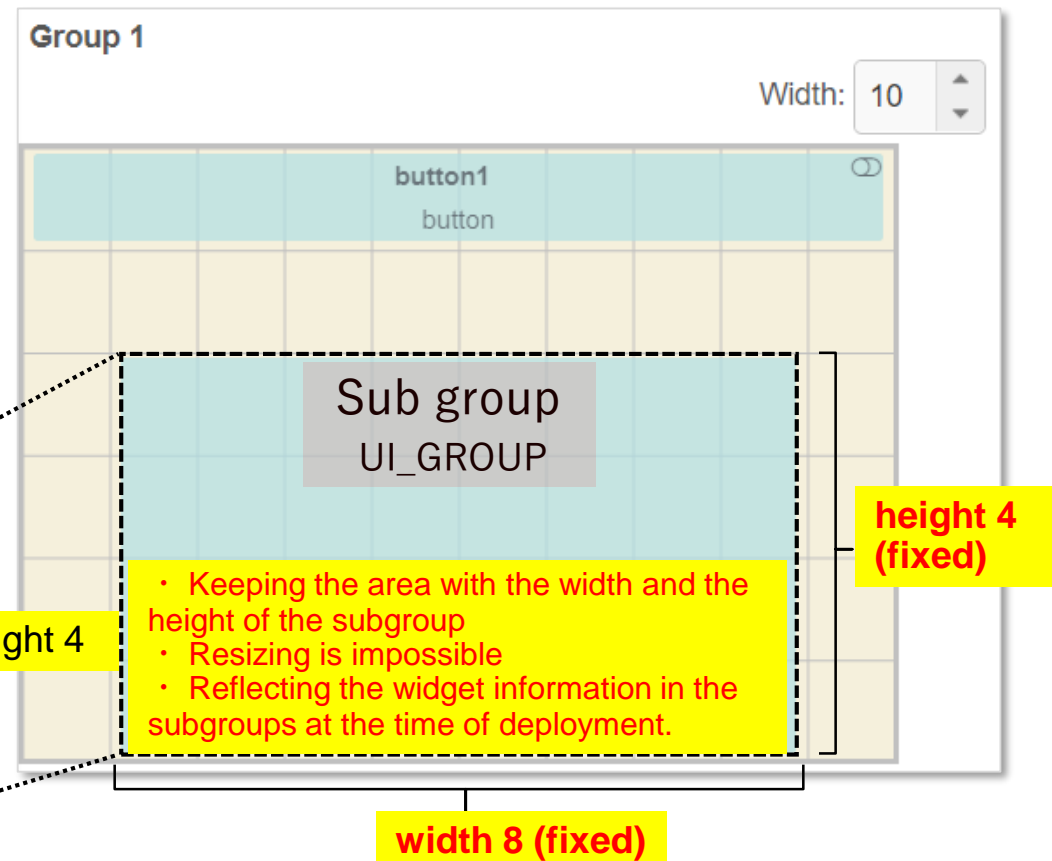
■ Specification proposal

1. Adding the function to layout UI Module in sub-flow.
2. Adding to the layout tool the function to handle sub-groups in groups inside tabs.

1. UI Module layout



2. Group layout inside tabs



■ Problems

When reflecting a subgroup into an existing group, it becomes a problem if the group width in the tab is smaller than the width of the subgroup. We have to consider handling when the group width in the tab is less than the subgroup width.

- Automatically expand the group width in the tab to the width of the subgroup.
- Regarding as an error, placement is not possible (need to consider how to notify errors)

HITACHI
Inspire the Next