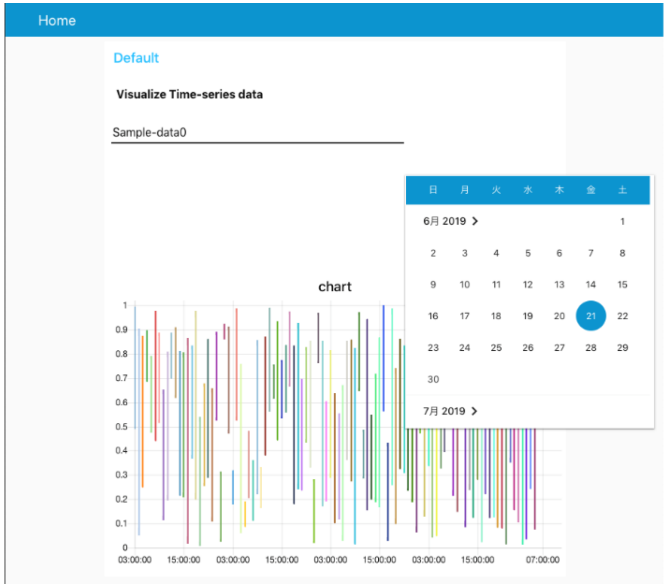
HITACHI
Inspire the Next

Dashboard Enhancements

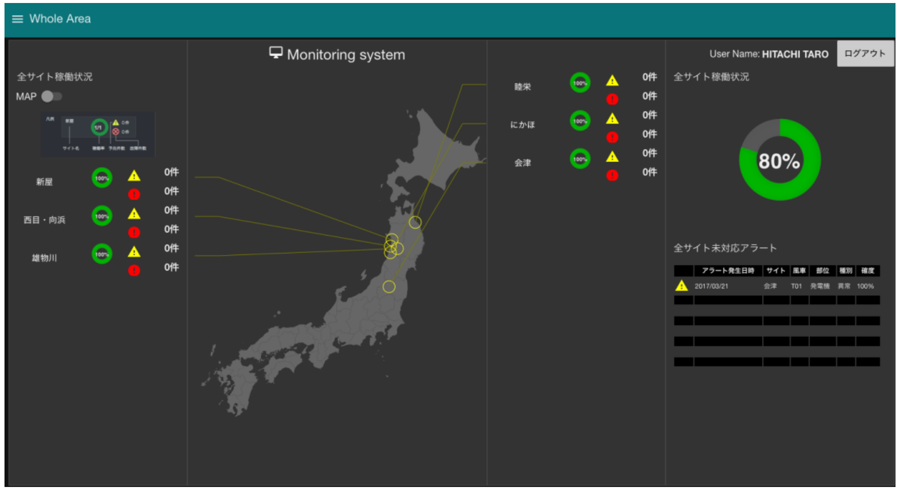
Hiroyasu Nishiyama

Background

- □ 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 dashboard.



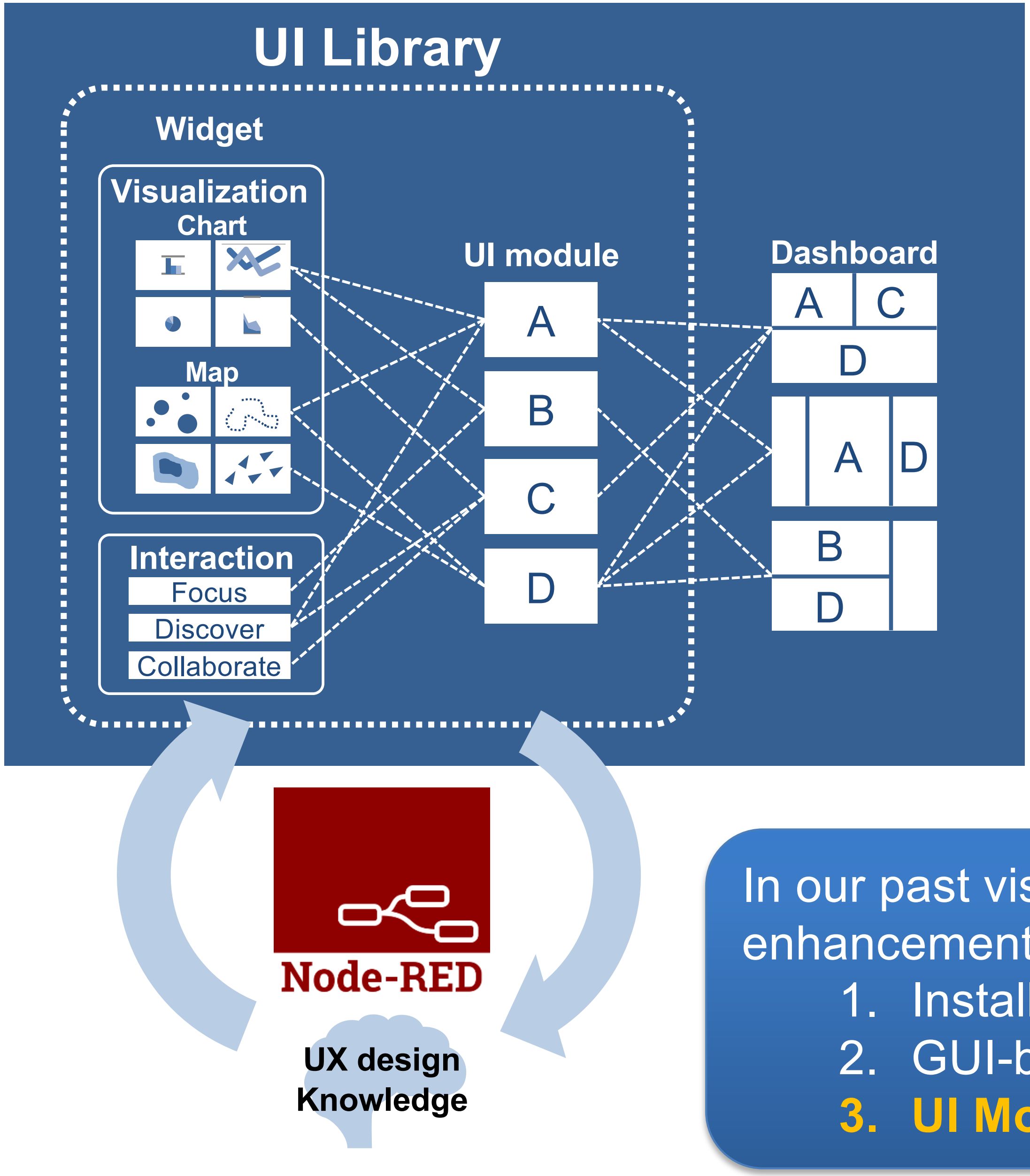
Simple Dashboard



	アラート発生日時	サイト	風車	部位	種別	確度
⚠	2017/03/21	会津	T01	発電機	異常	100%
[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]

Complex Dashboard

UI Library & Node-RED Dashboard Enhance



- **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.

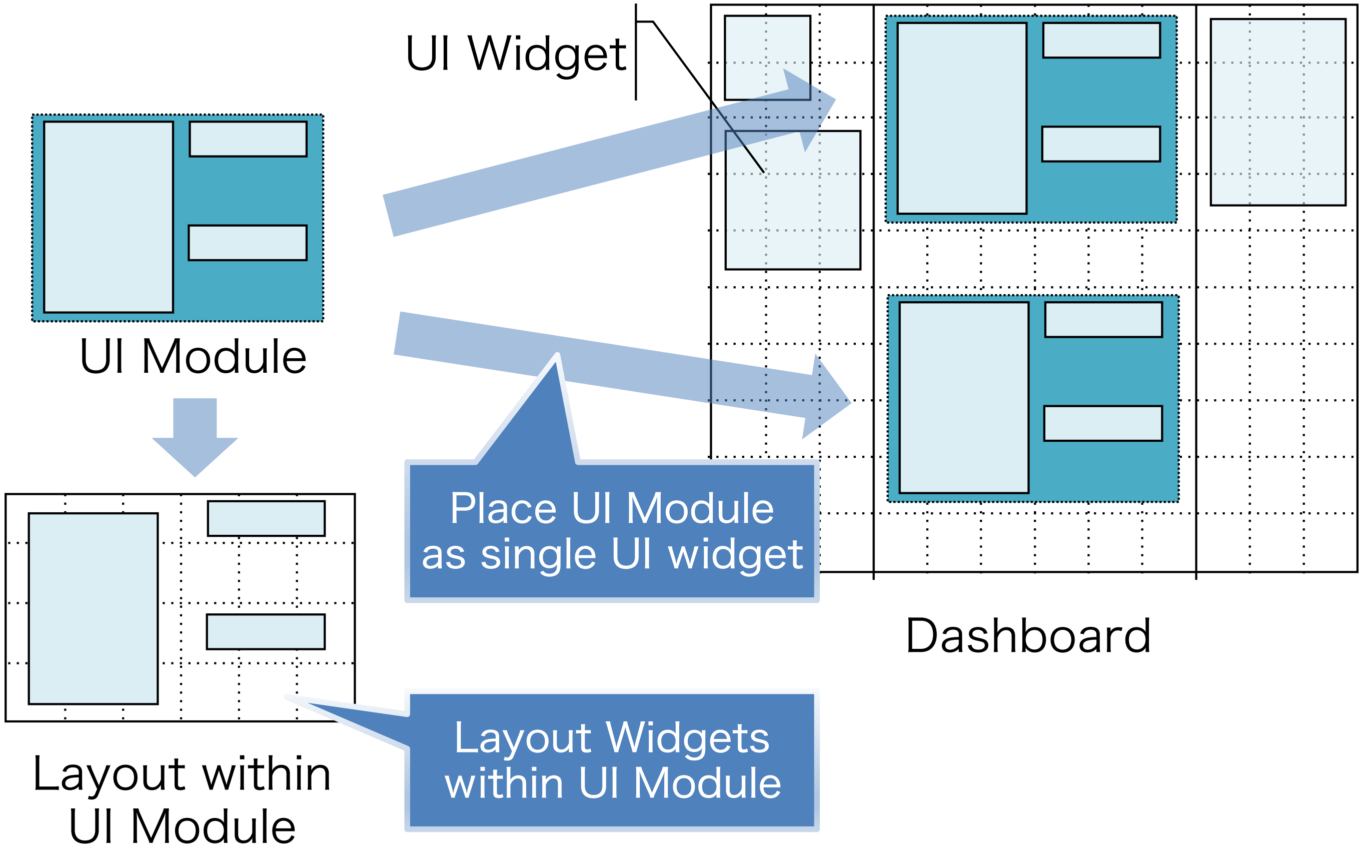
In our past visit to Hursley, we discussed following enhancements on Node-RED Dashboard:

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

UI Module (Compound Widget)

HITACHI
Inspire the Next

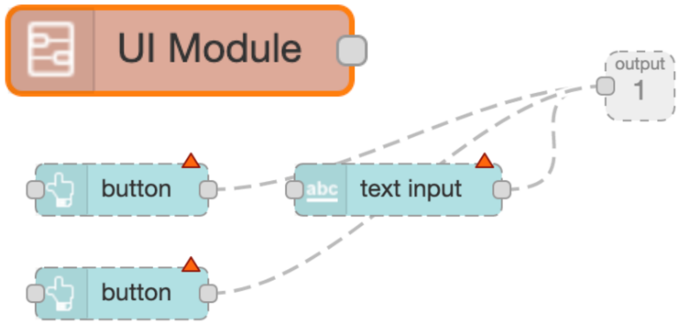
- □ UI module consists of a set of UI widgets.
  It has its own internal layout of containing widgets.
- □ UI modules can be placed on dashboard similar to UI widgets



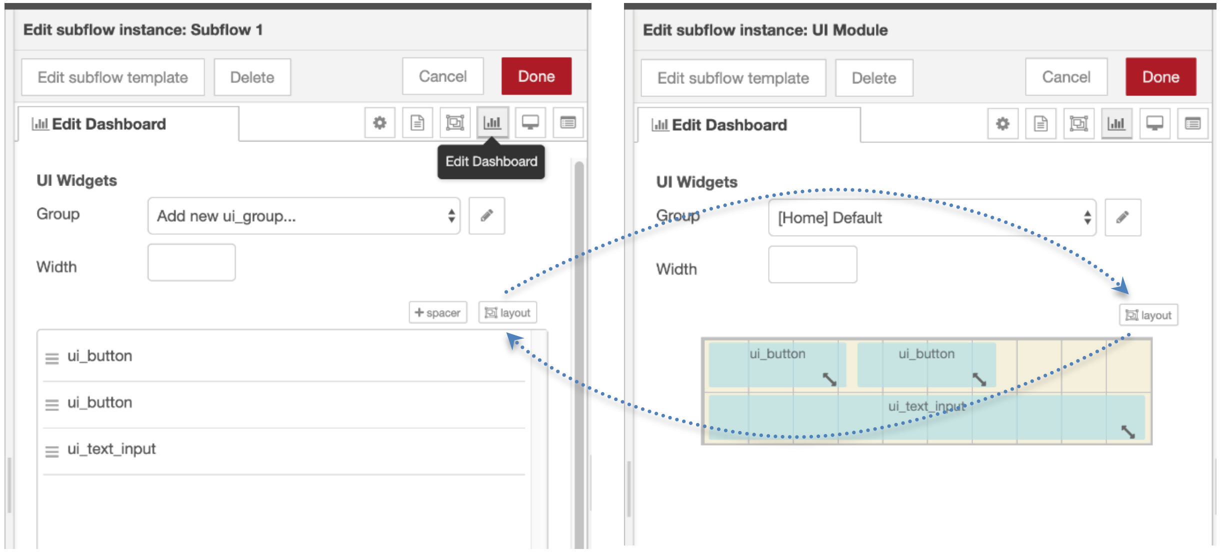
UI Widget
UI Module
Place UI Module as single UI widget
Dashboard
Layout within UI Module
Layout Widgets within UI Module

Initial Proposal: SUBFLOW as UI Module

- UI Module is represented as a SUBFLOW that contains UI widgets
- SUBFLOW is extended to have interface for UI layout



UI Module (SUBFLOW) containing 3 widgets



Edit subflow instance: Subflow 1

UI Widgets

Group: Add new ui_group...

Width:

ui_button
ui_button
ui_text_input

Edit subflow instance: UI Module

UI Widgets

Group: [Home] Default

Width:

ui_button
ui_button
ui_text_input

(a) order mode

(b) layout mode

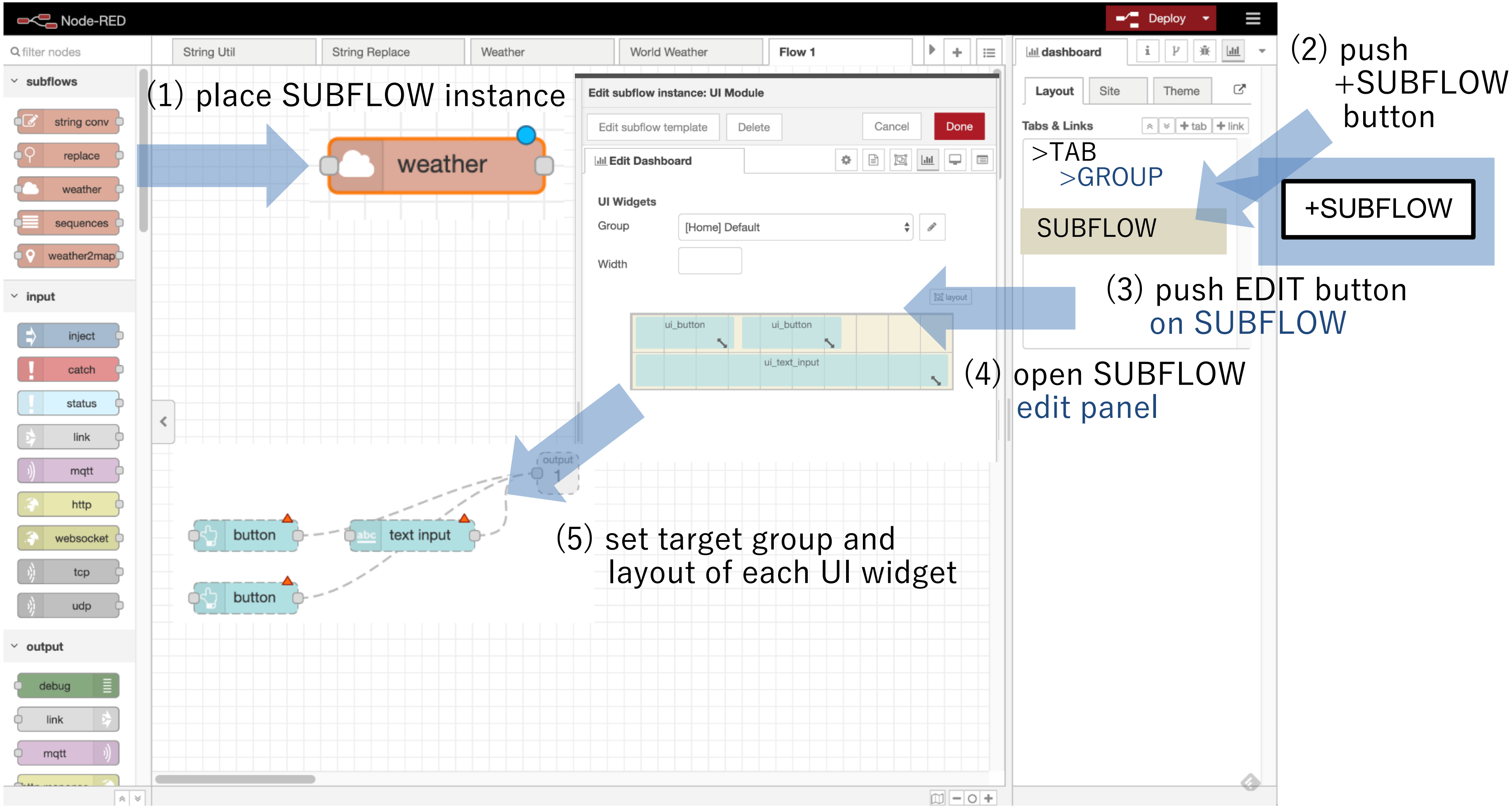
Settings Panel of UI Module

PROBLEM:
Node-RED(SUBFLOW) implementation should be independent of external node module (i.e. Node-RED dashboard).

Discussion on Last Visit to Hursley

HITACHI
Inspire the Next

Discussed solution:
Add feature to edit layout of SUBFLOW UI Module instance to Node-RED Dashboard



(1) place SUBFLOW instance

(2) push +SUBFLOW button

(3) push EDIT button on SUBFLOW

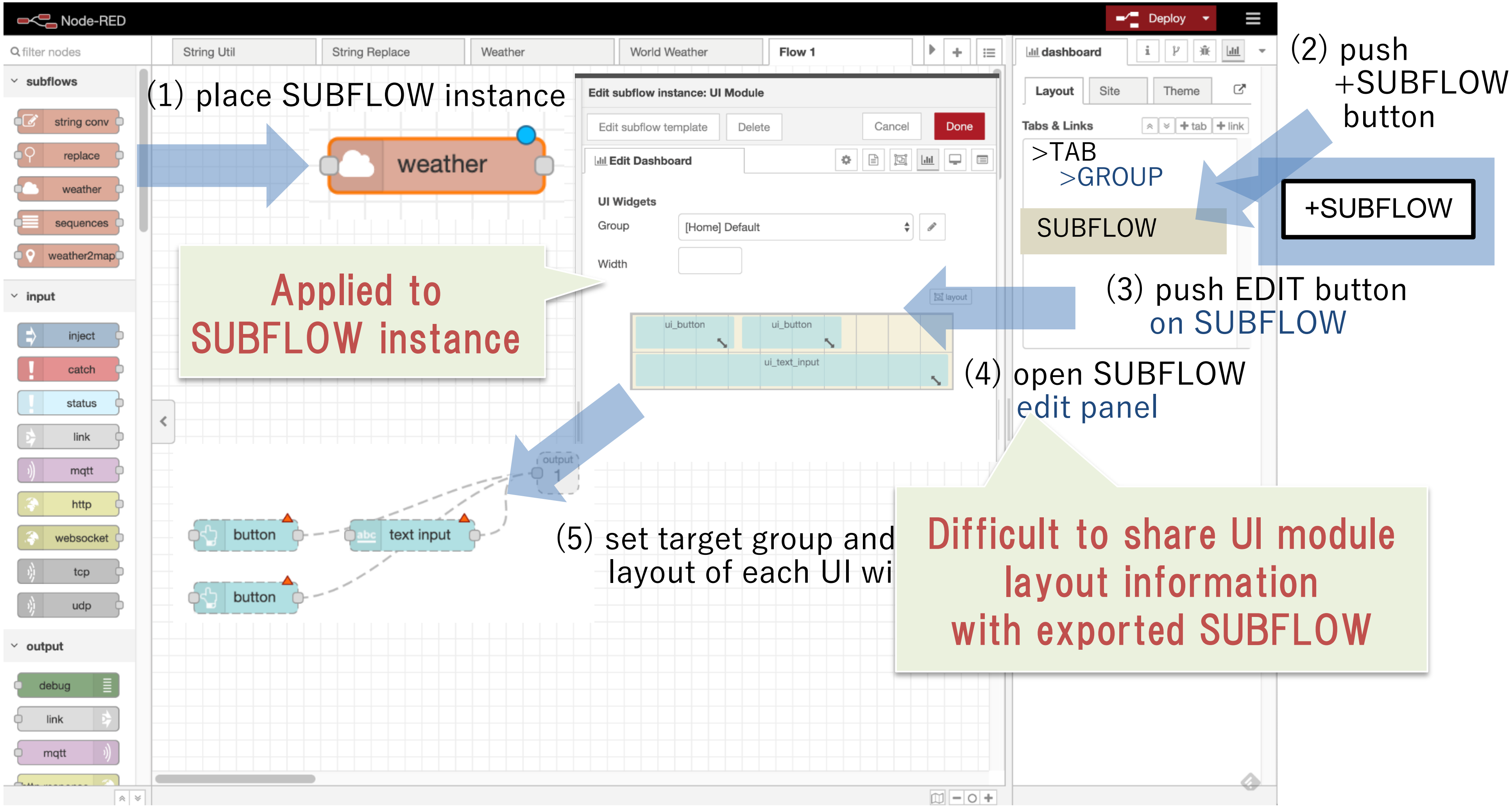
(4) open SUBFLOW edit panel

(5) set target group and layout of each UI widget

Thoughts on the Latest Proposal

PROBLEMS:

- □ Can only be applied to SUBFLOW instance (not template)
- □ Can not share layout information among exported SUBFLOW



(1) place SUBFLOW instance

(2) push +SUBFLOW button

(3) push EDIT button on SUBFLOW

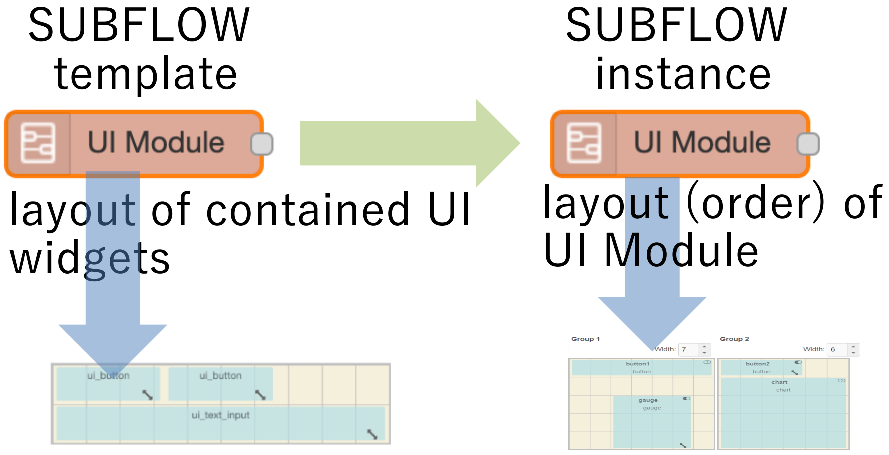
(4) open SUBFLOW edit panel

(5) set target group and layout of each UI wi

Applied to SUBFLOW instance

Difficult to share UI module layout information with exported SUBFLOW

Ideas on Solving UI Module Layout Problem



SUBFLOW template

UI Module

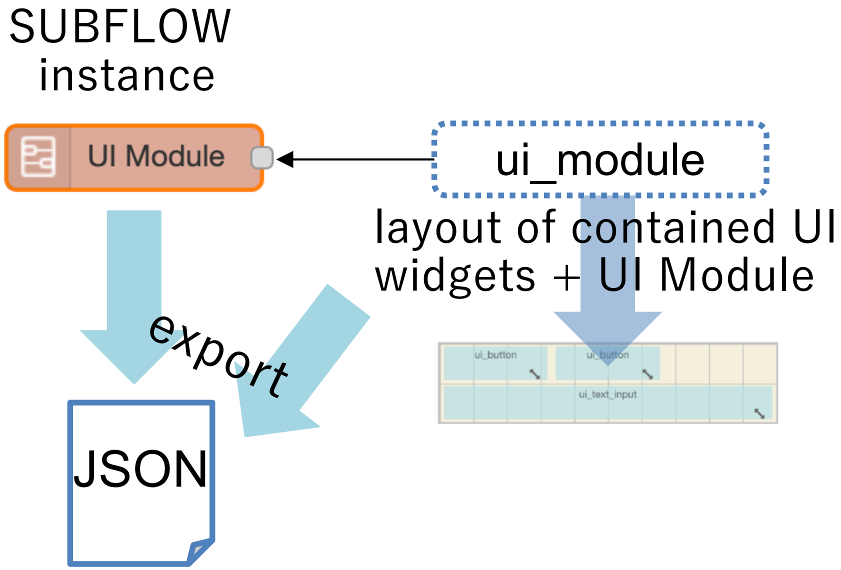
layout of contained UI widgets

SUBFLOW instance

UI Module

layout (order) of UI Module

(a) Extend SUBFLOW



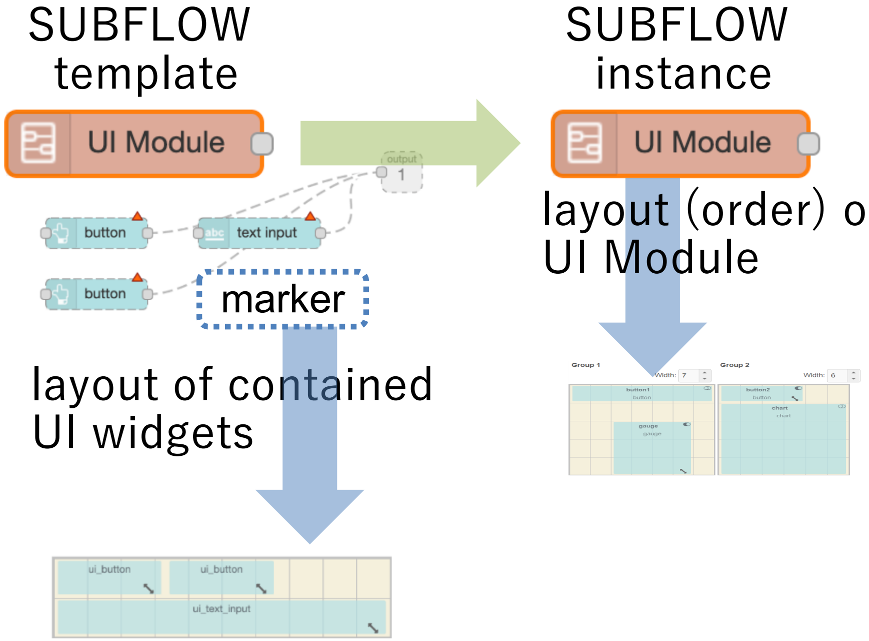
SUBFLOW instance

UI Module

ui_module

layout of contained UI widgets + UI Module

(b) Extend Import/Export



SUBFLOW template

UI Module

SUBFLOW instance

UI Module

marker

layout of contained UI widgets

layout (order) of UI Module

(c) Use Marker Node

New Proposal: Extending SUBFLOW UI

HITACHI
Inspire the Next

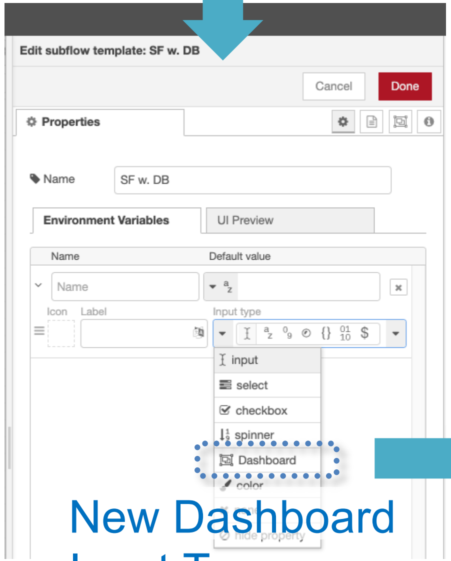
- □ Add an API for registering a new SUBFLOW env var type
- □ UI for UI module layout information is added using this interface by Node-RED Dashboard.

Node-RED
Rumtime

Register input type on startup

RED.nodes.registerSubflowInputType(...)

Node-RED
Dashboard



Edit subflow template: SF w. DB

Cancel Done

Properties

Name: SF w. DB

Environment Variables | UI Preview

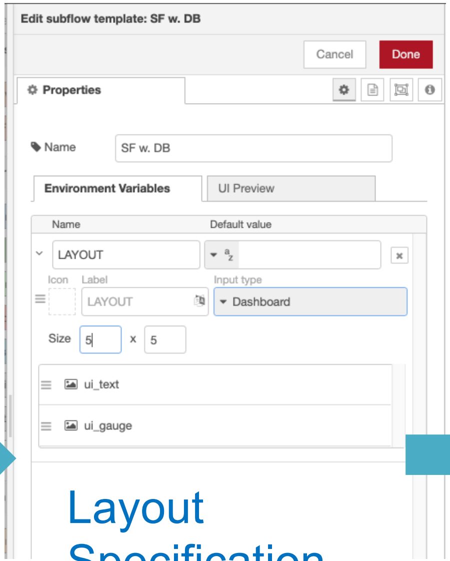
Name | Default value

Name

Icon Label Input type

input
select
checkbox
spinner
Dashboard
color
hide property

New Dashboard
Input Type



Edit subflow template: SF w. DB

Cancel Done

Properties

Name: SF w. DB

Environment Variables | UI Preview

Name | Default value

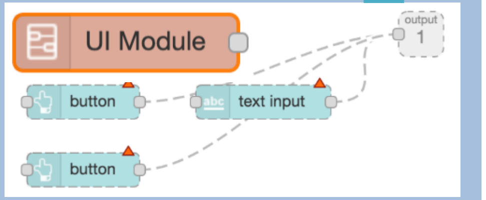
LAYOUT

Icon Label: LAYOUT Input type: Dashboard

Size 5 x 5

ui_text
ui_gauge

Layout
Specification



+

Layout Info
in env. var.

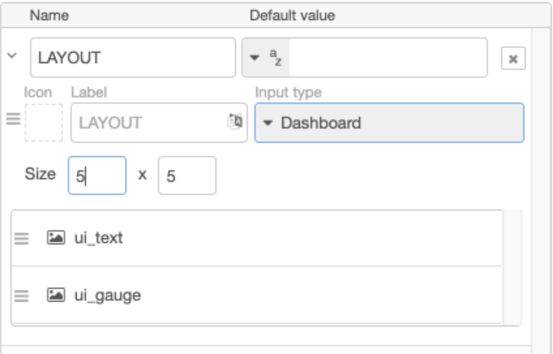
use instance

Nod-RED dashboard uses layout info. in env. var. for placing UI module

export as node
(UI module)

Layouting UI Module

- UI module (SUBFLOW) must define UI_LAYOUT env. var. which contains dashboard layout information of contained UI widgets within SUBFLOW



Name	Default value
LAYOUT	ᵃz

Icon Label: LAYOUT

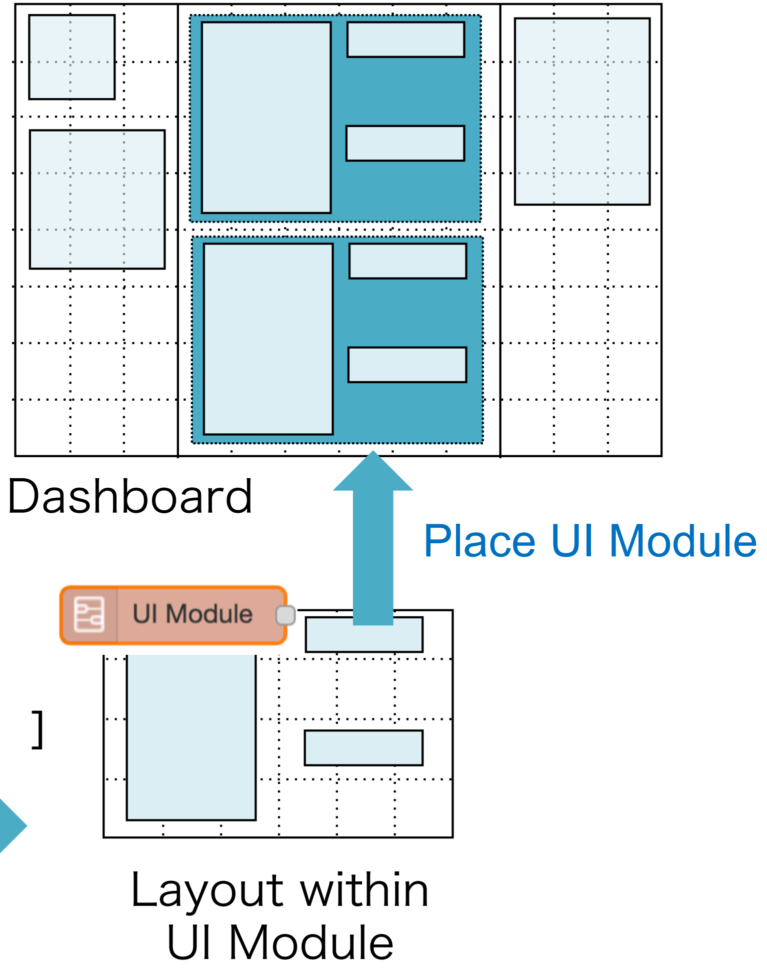
Input type: Dashboard

Size 5 x 5

- ui_text
- ui_gauge

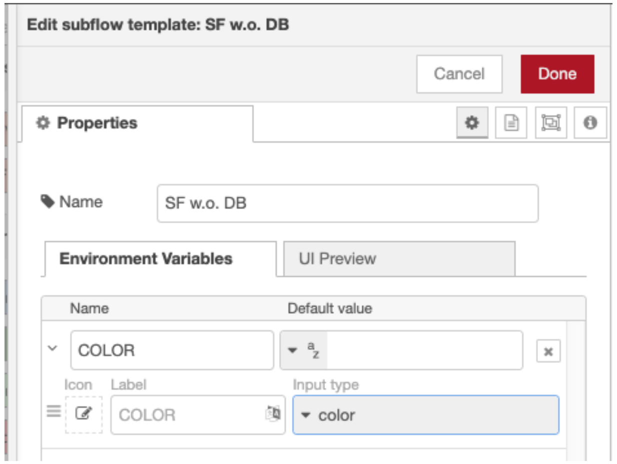
```
{
  size: [ 6, 4 ],
  order: [ "<node1>", "<node2>", "<node3>" ]
}
```

Layout Info in UI_LAYOUT variable



User-Defined SUBFLOW Input Type

- □ Also useful for various kind of data input
- □ Allow selective activation based on SUBFLOW implementation (e.g. activate dashboard input if SUBFLOW contains widgets)



Edit subflow template: SF w.o. DB

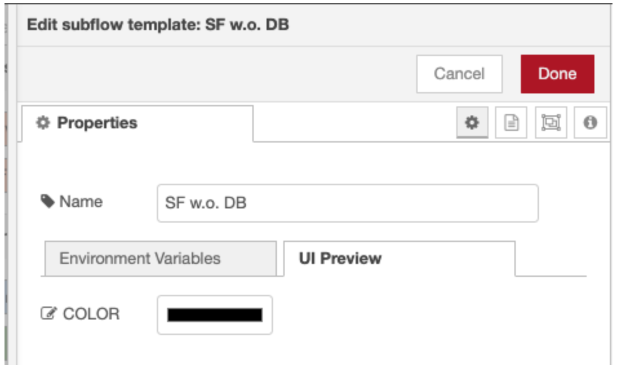
Name: SF w.o. DB

Environment Variables

Name	Default value
COLOR	

Icon / Label: COLOR / Input type: color

(a) COLOR input definition



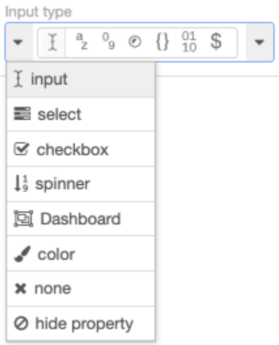
Edit subflow template: SF w.o. DB

Name: SF w.o. DB

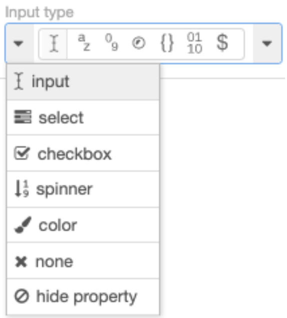
UI Preview

COLOR: ■

(b) COLOR input UI



(a) w. dashboard



(b) w.o. dashboard

registerSubflowInputType API[T.B.D.]

- registerSubflowInputType API adds new input type for SUBFLOW UI definition
- Example

```
var createUI = function(row, id, ui) {
    $("<input/>", { type: "color", id: id }).css({ width: "100px" }).appendTo(row);
};
var exportValue = function(input, item) {
    item.type = "color";
    item.value = input.val();
};
var def = {
    def: {                                    // Basic info. for UI definition I/F
        value: "color",
        label: "color",
        icon: "fa fa-paint-brush",
        hasValue: false
    },
    createUI: createUI,                        // Callback for custom UI creation
    export: exportValue,                       // Callback for exporting to env. var.
    isActive: function() { return true; },     // Callback for selective activation
    onSelection: undefined                     // Callback for custom UI definition I/F
};
RED.nodes.registerSubflowInputType(def);
```

[DEMO] User-Defined Input Type

DEMO

Summary

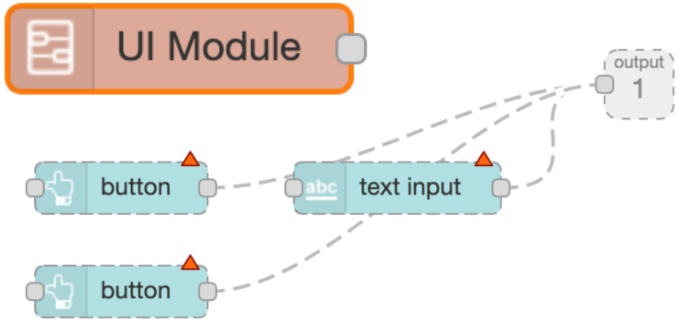
- Proposed API for adding User-defined Type for SUBFLOW env. var.
- This allows dashboard layout information for SUBFLOW UI module can be specified using SUBFLOW UI
- New API is also useful for making other data input for SUBFLOW UI such as color, date, etc. easy and intuitive.

HITACHI
Inspire the Next

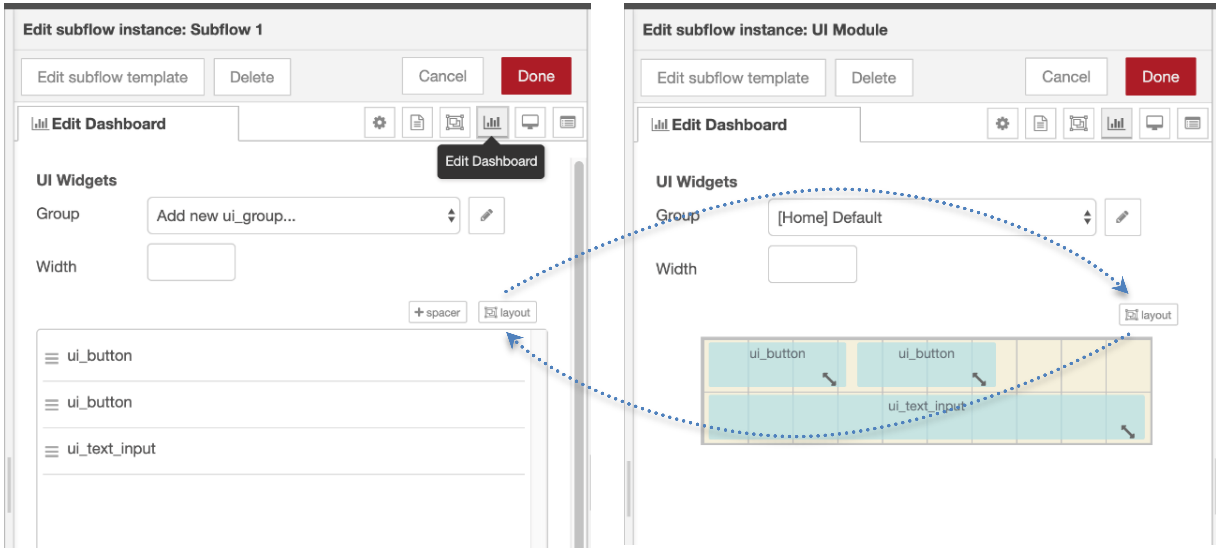
Proposal: SUBFLOW as UI Module

- 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



Edit subflow instance: Subflow 1

UI Widgets

Group: Add new ui_group...

Width:

ui_button
ui_button
ui_text_input

Edit subflow instance: UI Module

UI Widgets

Group: [Home] Default

Width:

ui_button ui_button
ui_text_input

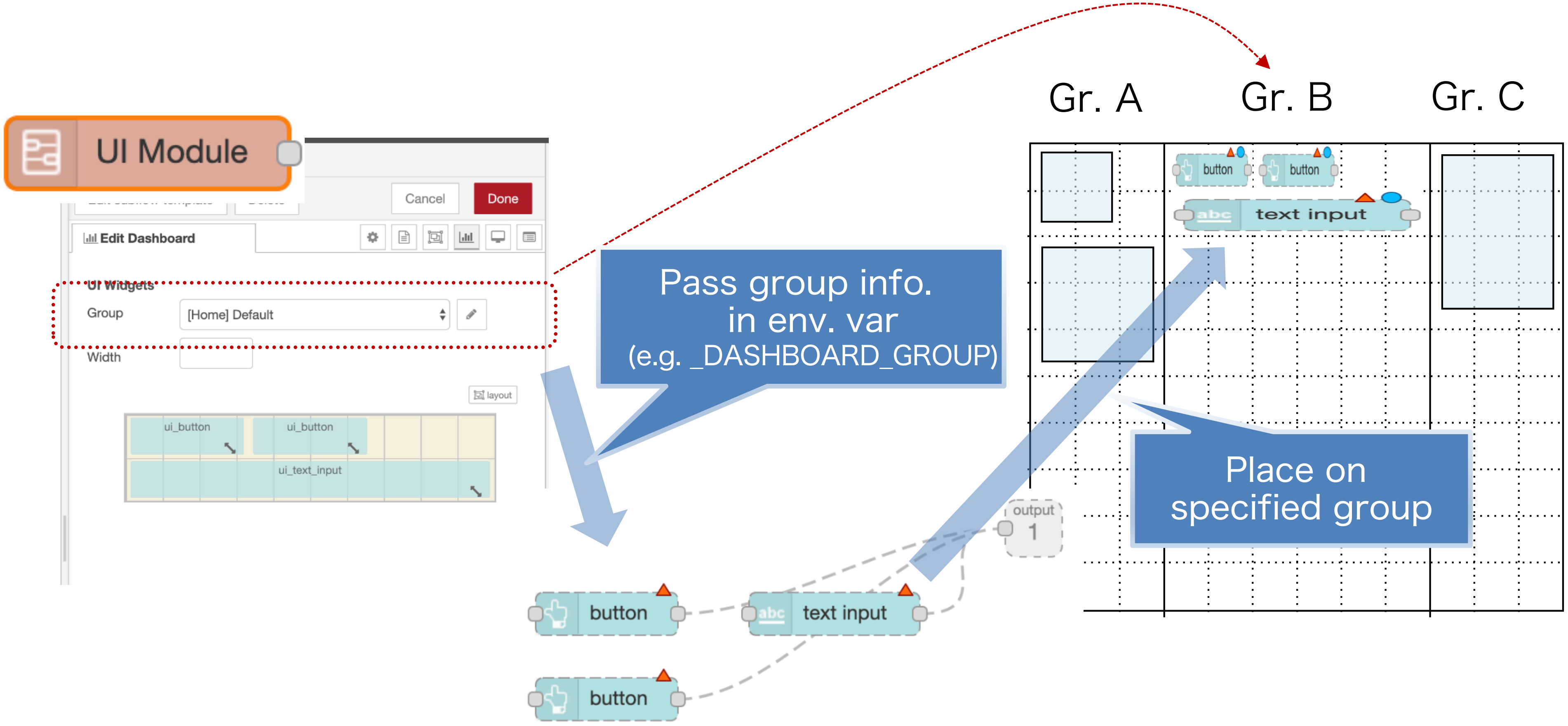
(a) order mode

(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.



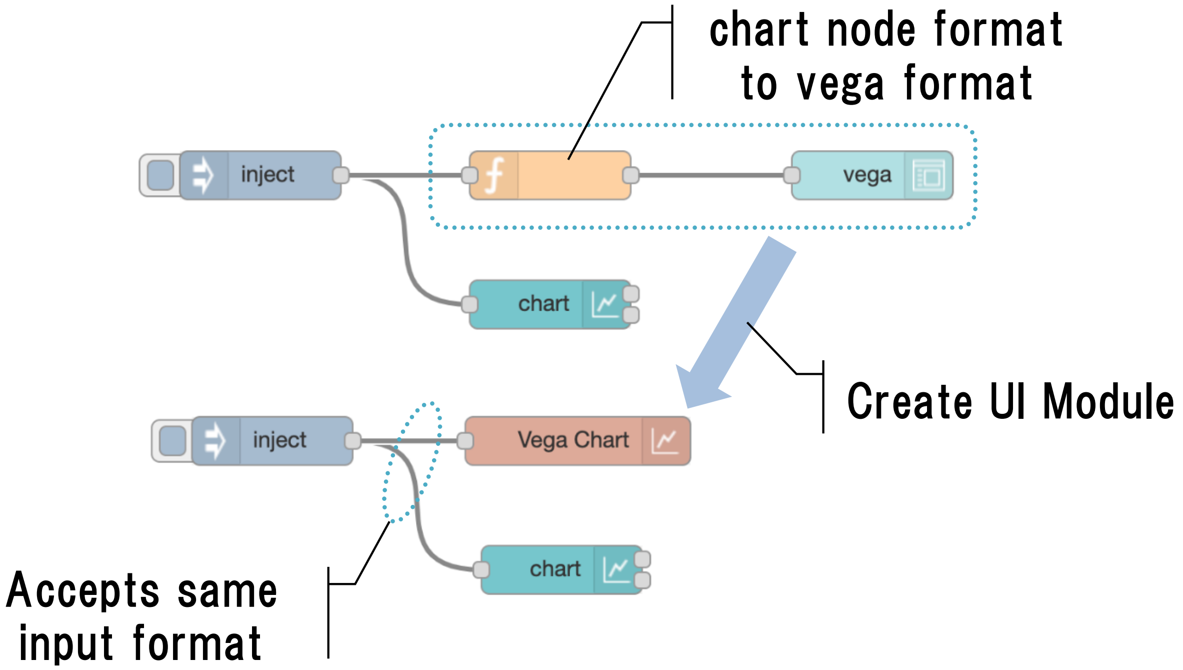
Pass group info.
in env. var
(e.g. _DASHBOARD_GROUP)

Place on
specified group

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

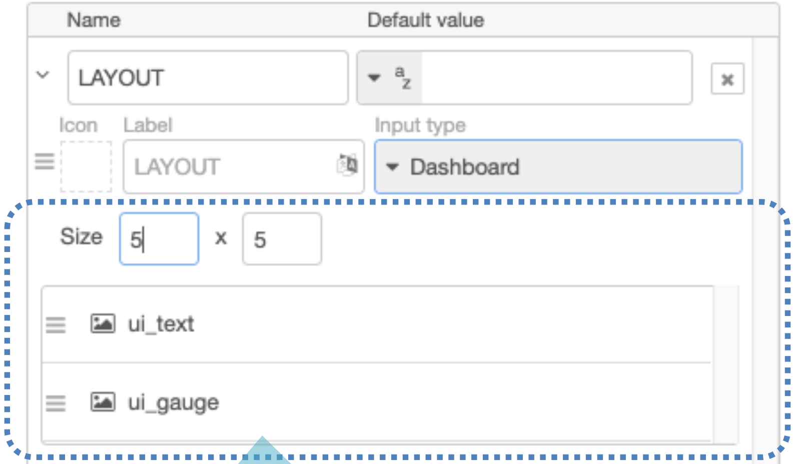
- ■ Vega node accepts complex visualization specification in JSON
- ■ The JSON specification is sometimes difficult to write
- ■ Conversion from light-weight format to Vega specification is useful



registerSubflowInputType API

Specify following properties:

- □ Type name, Label, Icon, …
- □ UI for template definition, SUBFLOW env. var. input
- □ importing/exporting values



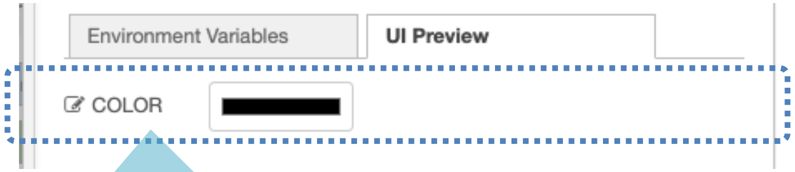
Name	Default value
LAYOUT	^a_z

Icon | Label: LAYOUT | Input type: Dashboard

Size 5 x 5

- ui_text
- ui_gauge

UI for template definition



COLOR: ■

UI for SUBFLOW env. var. input