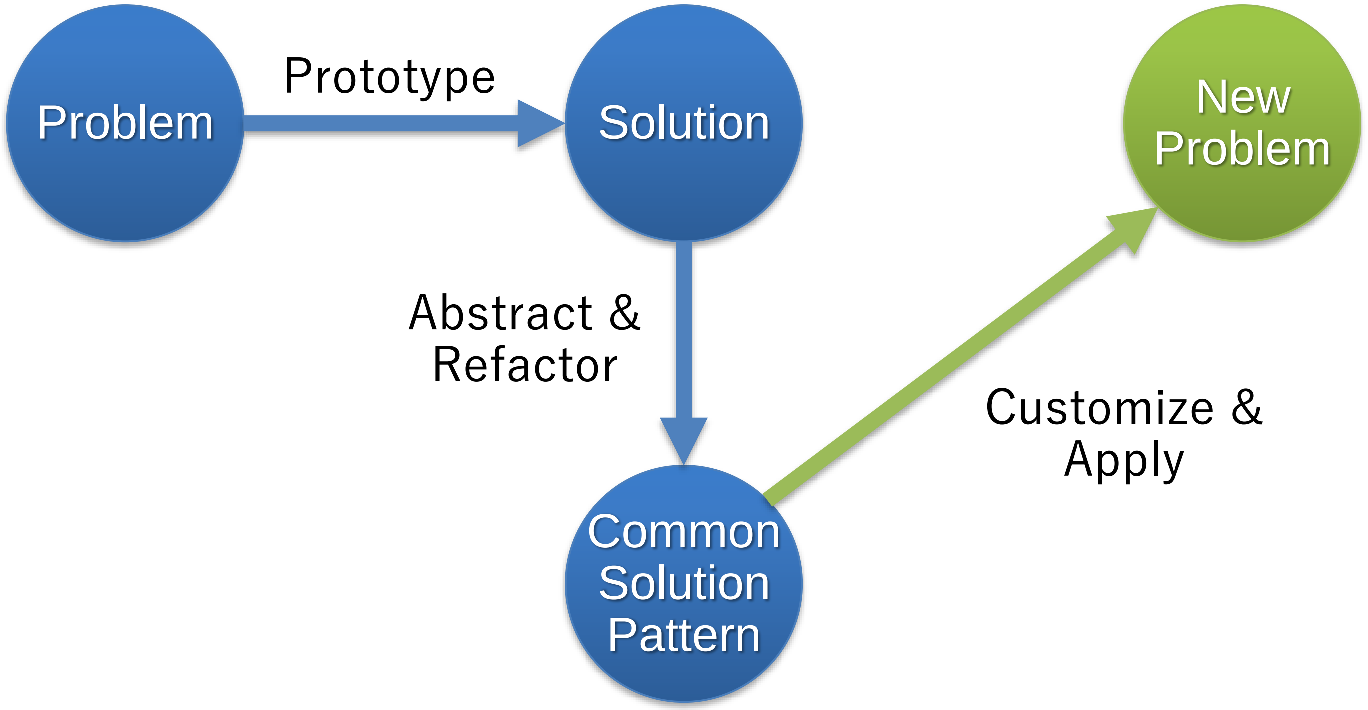
Exporting SUBFLOW as Node

Hiroyasu Nishiyama

Background

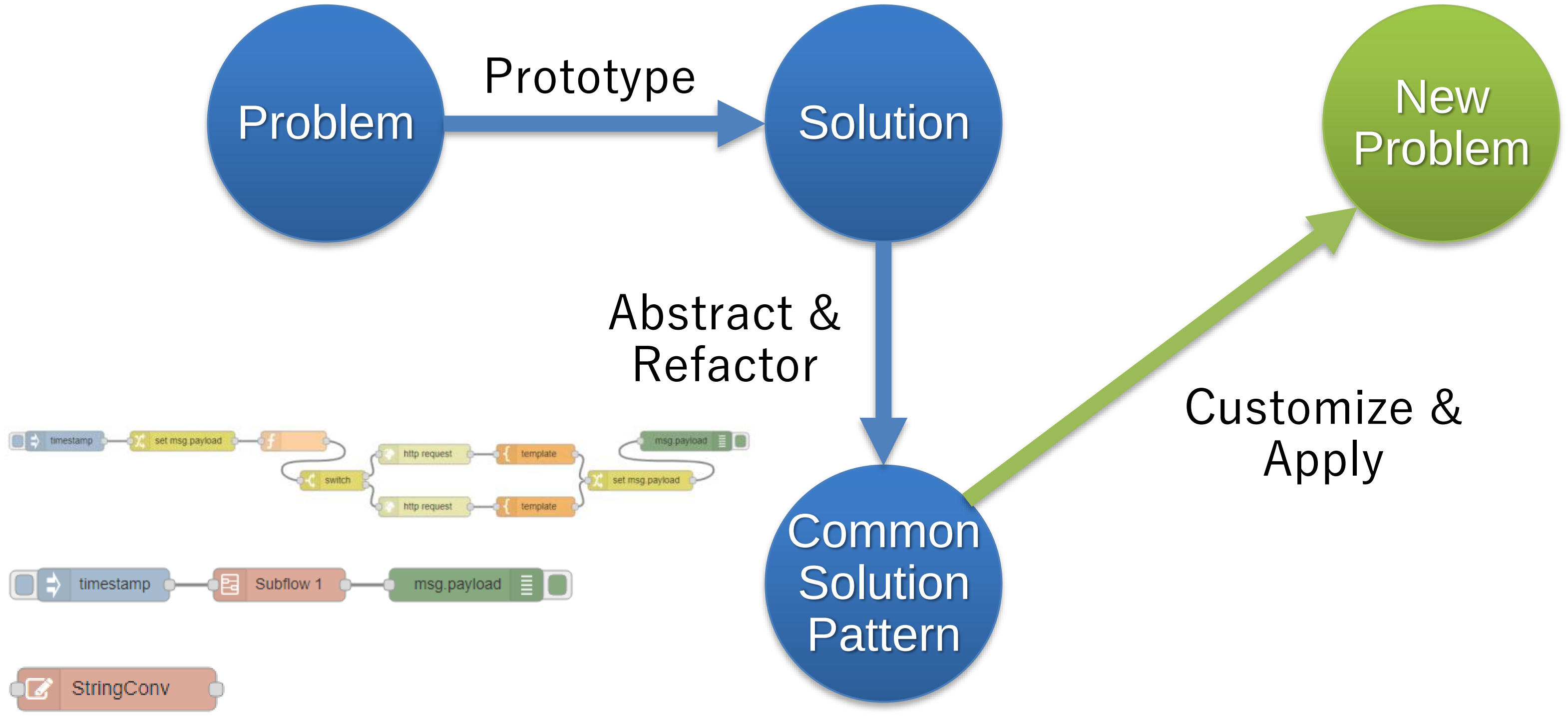
- Node-RED is a highly effective tool for rapid creation of new solutions.
- On the other hand, we would like to create basis for **sharing common solution patterns (or templates)** useful for creating new custom solutions by novice IT users.



Abstracting Solution Patterns

- Abstraction using FLOW, SUBFLOW or npm module is not sufficient

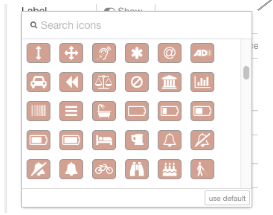
	Creation	Customization
FLOW	easy (visual editor)	difficult (no specific customization point)
SUBFLOW	easy (visual editor)	difficult (no specific customization point)
new node	difficult (HTML+JS)	easy (node settings UI)

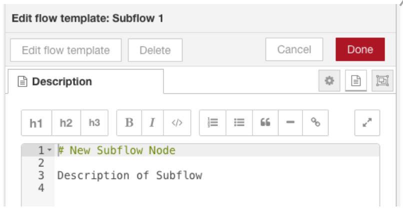


Customizing Subflow

- By adding customization capability to Subflow, Subflow can define similar functionality to normal Node using Node-RED editor GUI.

No.	Item	Node	Subflow (ToBe)
1	Icon	Any Picture, Icon Font (via GUI)	Icon Font (via GUI)
2	Info Text	HTML, Markdown (via GUI)	Markdown (via GUI)
2	Logic	JavaScript	Flow (via GUI)
3	Settings UI	HTML	None → via GUI
4	Color	Any Color	None → via GUI
5	Category	Any Category	Any Category (via GUI)




Edit flow template: Subflow 1

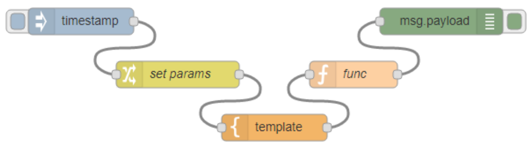
Edit flow template | Delete | Cancel | Done

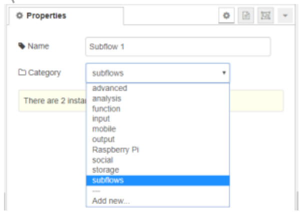
Description

```
# New Subflow Node

Description of Subflow

```




Properties

Name: Subflow 1

Category: subflows

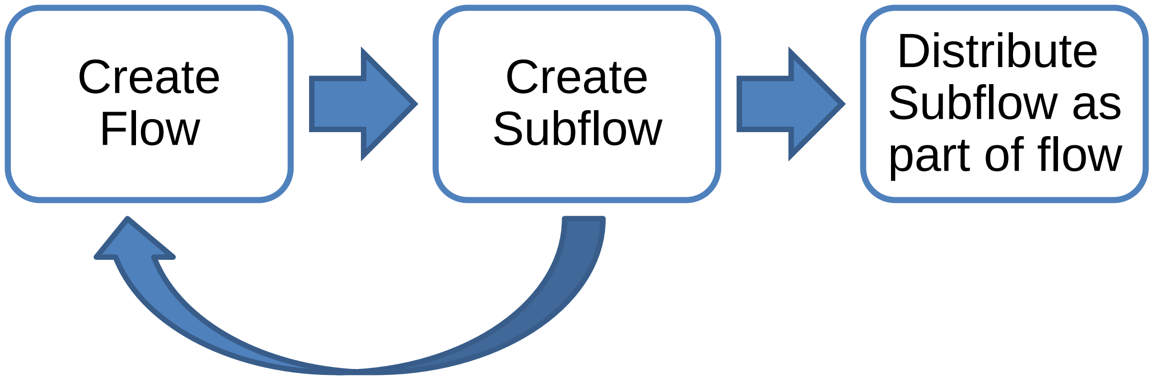
advanced
analysis
function
input
mobile
output
Raspberry Pi
social
storage
subflows

Add new...

There are 2 instar

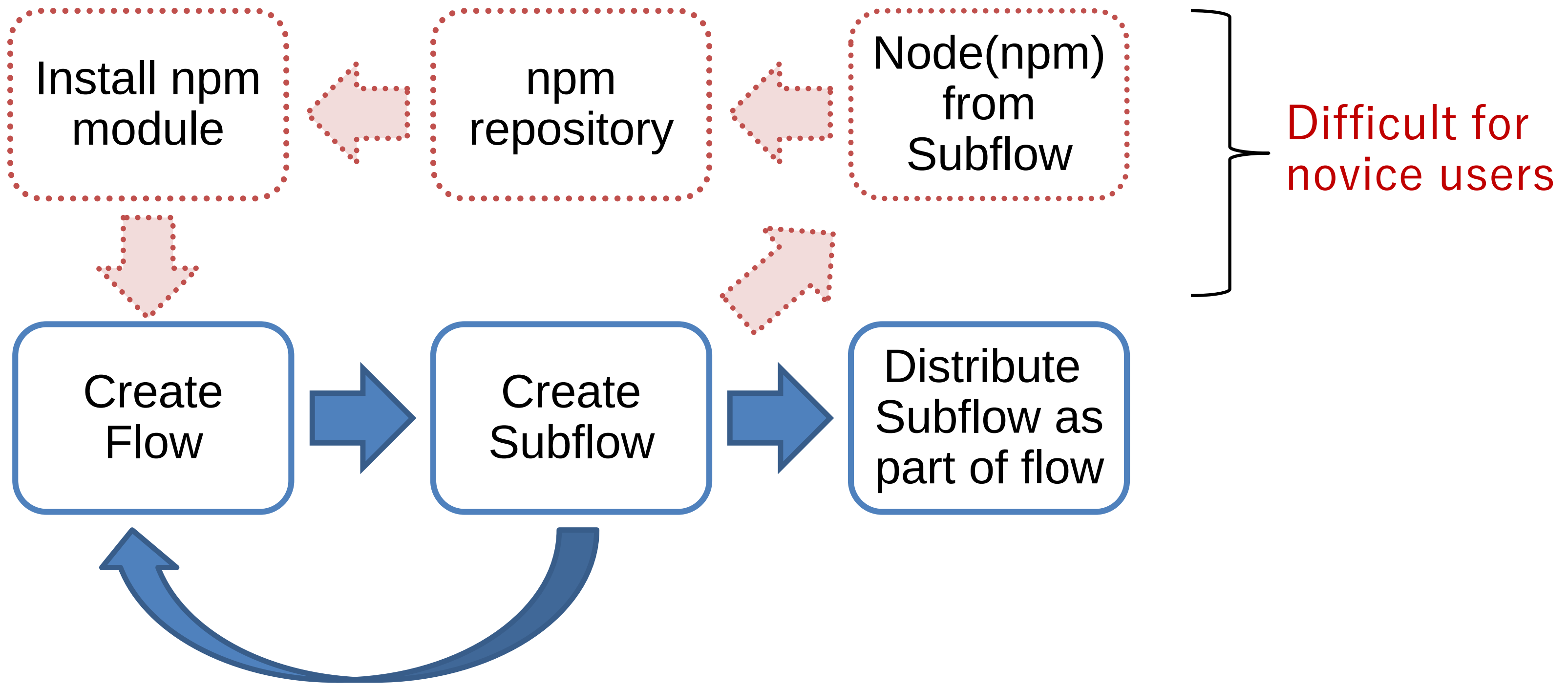
Current Use of Subflow in Flow Creation

- SUBFLOW in current Node-RED can be reused in the belonging flow and can be redistributed as part of the flow.
- Flow format is handy for sharing among users because it is represented by JSON format.



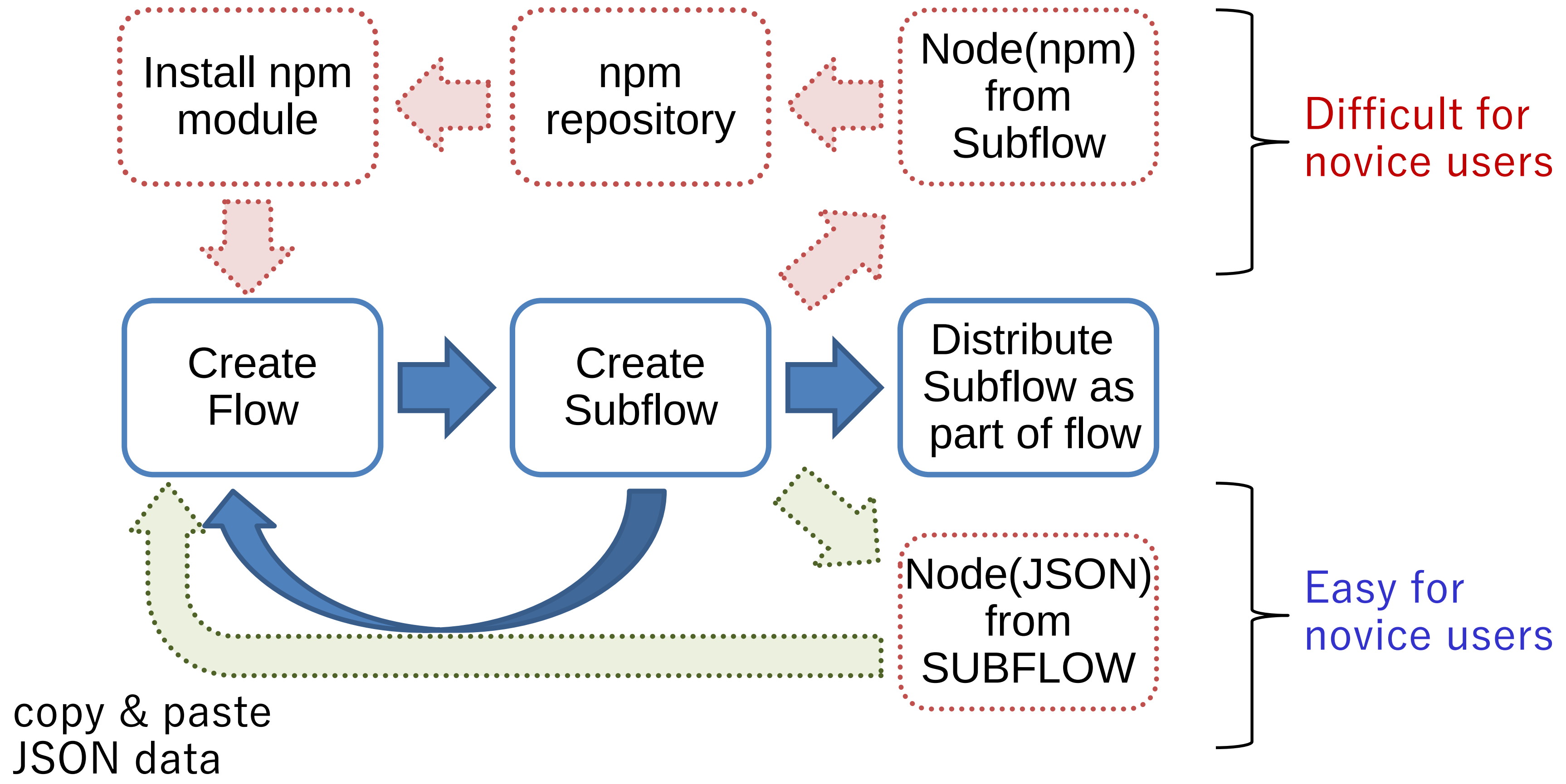
Creating Node in npm module from Subflow

- Create Node in npm module from Subflow.
- Node npm module is shared using npm repository.
  → npm creation/redistribution/etc. are difficult for novice users.
  → unknown node appear if SUBFLOW node is not installed.
  → Installation of a node is needed for flows that uses the node.



Exporting Node in JSON from Subflow

- Export Node in JSON format from Subflow.
- Node npm module is shared using JSON (text) format.
  → Can be redistributed as part of a flow (eliminates unknown nodes)
  → npm repository and explicit node installation is not needed.



Flow Format Extension

- Current FLOW export format in JSON is represented as an array of node specifications.
- Proposal:
  1. Introduce new object-style FLOW/SUBFLOW representation which include meta-data for describing properties of exported JSON data
  2. Current array-style flow representation will be used for compatibility
  3. Add properties for flow data type, hiding flow details, etc.

```
[
    { id": "...", type: "...", ... },
    { id": "...", type: "...", ... },
    ...
]
```

nodes definition

Current FLOW format

```
{
    type: "flow",
    sealed: true,
    flow: [
        { id: "...", type: "...", ... },
        { id: "...", type: "...", ... },
        ...
    ]
}
```

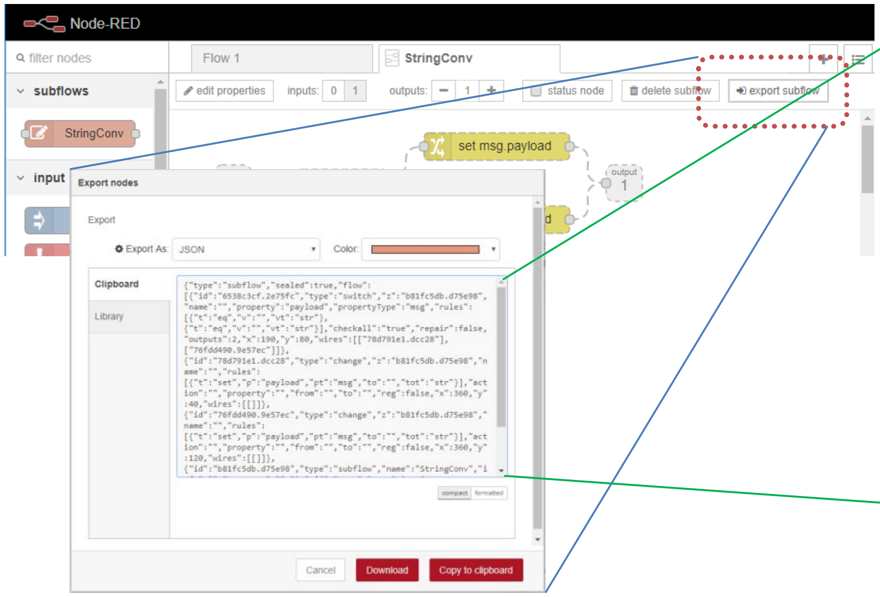
meta-data

New FLOW format

- Various kinds of data can be exported/imported using new format

Exporting SUBFLOW

- Add 「export subflow」 button to SUBFLOW template
- Use new FLOW format for distributing SUBFLOW:
  - type = "subflow"
  - sealed = true: hide details of exported subflow
  - flow: array of SUBFLOW node & nodes definitions in its template



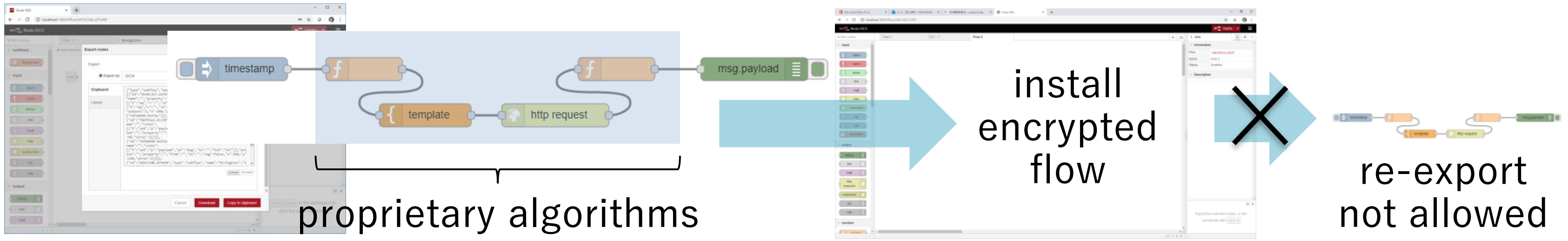
```
{"type":"subflow","sealed":true,"flow":
[{"id":"6538c3cf.2e75fc","type":"switch","z":"b81fc5db.d75e98",
"name":"","property":"payload","propertyType":"msg","rules":
[{"t":"eq","v":"","vt":"str"},
{"t":"eq","v":"","vt":"str"}],"checkall":"true","repair":false,
"outputs":2,"x":190,"y":80,"wires":[["78d791e1.dcc28"],
["76fdd490.9e57ec"]]},
{"id":"78d791e1.dcc28","type":"change","z":"b81fc5db.d75e98","n
ame":"","rules":
[{"t":"set","p":"payload","pt":"msg","to":"","tot":"str"}],"act
ion":"","property":"","from":"","to":"","reg":false,"x":360,"y"
:40,"wires":[[]]},
{"id":"76fdd490.9e57ec","type":"change","z":"b81fc5db.d75e98","
name":"","rules":
[{"t":"set","p":"payload","pt":"msg","to":"","tot":"str"}],"act
ion":"","property":"","from":"","to":"","reg":false,"x":360,"y"
:120,"wires":[[]]},
{"id":"b81fc5db.d75e98","type":"subflow","name":"StringConv","i
```

```
{
    type: "subflow",
    sealed: true,
    flow: [
        { id: "<ID>", type: "subflow", ... },
        { id: "...", z: "<ID>", type: "...", ... },
        ...
    ]
}
```

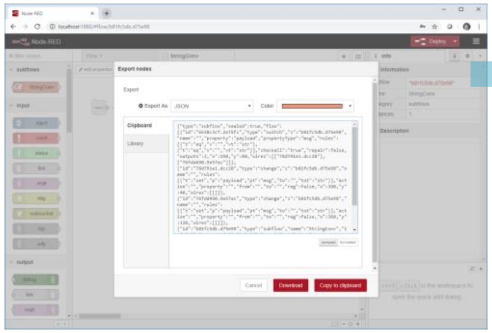
- How do we deal with nested SUBFLOW?
  1. only export single level - current prototype implementation
  2. include all called nodes in exported nodes list

Encrypting FLOW

- In some cases, we want to hide details of FLOW/SUBFLOW because it may contain intellectual property

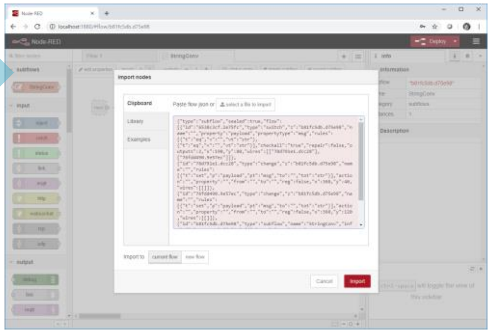


- By using new flow format, we can encrypt FLOW/SUBFLOW for distribution and decrypt it on installation. Details of FLOW/SUBFLOW can be marked hidden by sealed property.


Exporting Node-RED

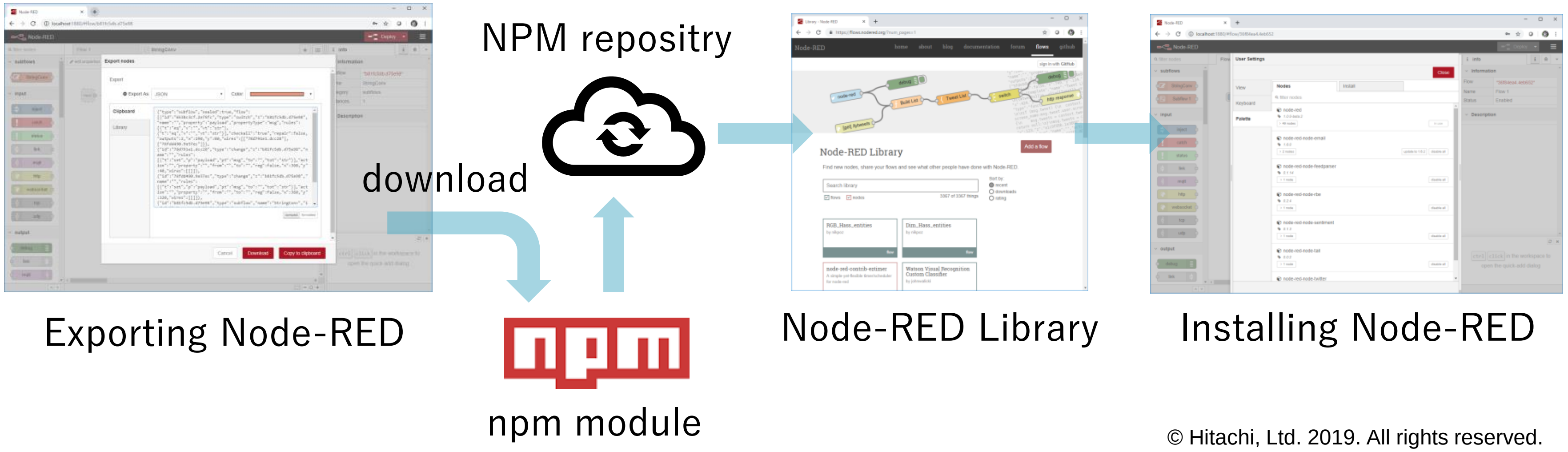
```
{
    type: "subflow-enc",
    sealed: true,
    flow: "W3siaWQiOiI2NTM..."
}
```

Encrypted FLOW format


Importing Node-RED

Exporting SUBFLOW as NPM Module

- Current node distribution uses NPM module as its format
- If we allow JSON based node (SUBFLOW) representation, redistributing SUBFLOW as NPM module may be useful:
  - automatic detection of node update,
  - embedding example flows,
  - listing in flow library by crawling npm repository,
  - …
- Allow exporting SUBFLOW as NPM module, or command to create NPM module from SUBFLOW JSON data



Demo

DEMO

Discussions on new JSON representation

- Updating/Deleting imported SUBFLOW
  - update/delete interface on manage palette
  - how to check SUBFLOW equality (node id sufficient?)

- How do we deal with nested SUBFLOW?
  1. only export single level - current prototype implementation
  2. include all called nodes in exported nodes list

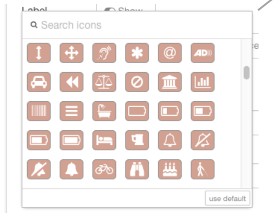
- For exporting encrypted FLOWs, sealing (hiding) all flow details is not appropriate (e.g., flow injection, examples).
  Introduction of sealing attribute to each tab is needed?

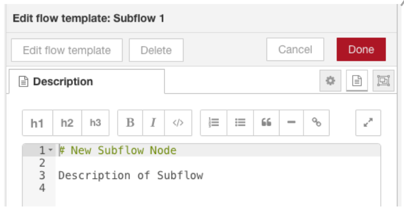
- Need runtime extension of APIs for encrypted SUBFLOW/FLOW not re-exported in decoded form.

Customizing Subflow

- By adding customization capability to Subflow, Subflow can define similar functionality to normal Node using Node-RED editor GUI.

No.	Item	Node	Subflow (ToBe)
1	Icon	Any Picture, Icon Font (via GUI)	Icon Font (via GUI)
2	Info Text	HTML, Markdown (via GUI)	Markdown (via GUI)
2	Logic	JavaScript	Flow (via GUI)
3	Settings UI	HTML	None → via GUI
4	Color	Any Color	None → via GUI
5	Category	Any Category	Any Category (via GUI)



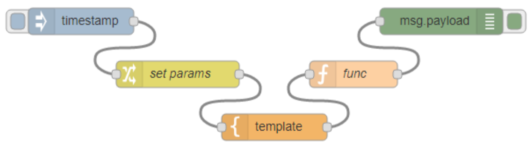


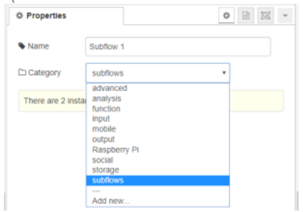
Edit flow template: Subflow 1

```
# New Subflow Node

Description of Subflow

```





Name: Subflow 1

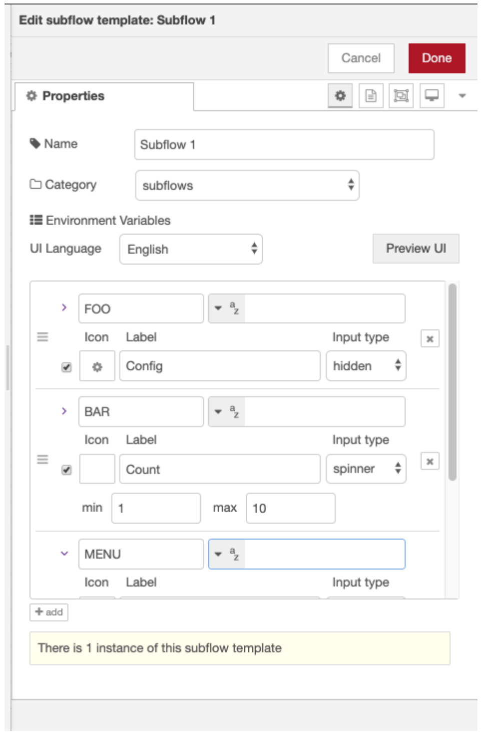
Category: subflows

advanced
analysis
function
input
mobile
output
Raspberry Pi
social
storage
subflows

There are 2 instal

Defining Parameter Input UI for Subflow

- Made PR to for UI definition in SUBFLOW settings. And rewriting based on your review.
- An UI for environment variables can be specified in SUBFLOW template.
  UI input type can be specified using environment variable extension
  - checkbox, spinner, menu, label only (no input), and typedInput input from selected types are supported



Edit subflow template: Subflow 1

Cancel | Done

Properties

Name: Subflow 1

Category: subflows

Environment Variables

UI Language: English

Preview UI

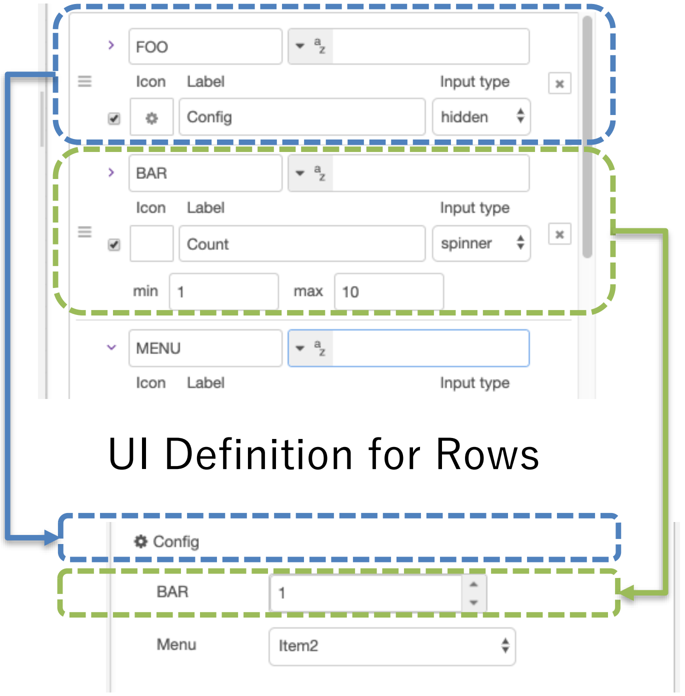
	Icon	Label	Input type
FOO		Config	hidden
BAR		Count	spinner
MENU			

BAR: min 1, max 10

+ add

There is 1 instance of this subflow template

SUBFLOW UI Definition



	Icon	Label	Input type
FOO		Config	hidden
BAR		Count	spinner
MENU			

BAR: min 1, max 10

UI Definition for Rows

Config

BAR: 1

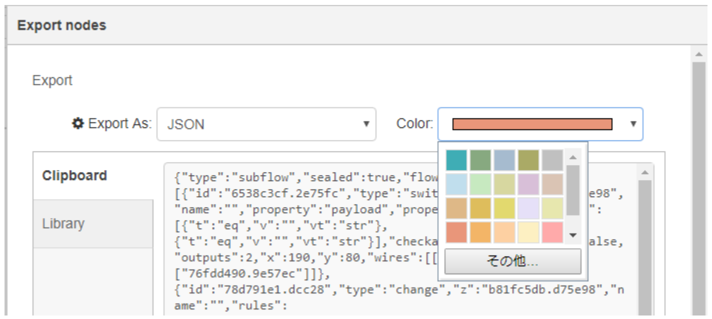
Menu: Item2

Generated Input UI

Node Color Specification

HITACHI
Inspire the Next

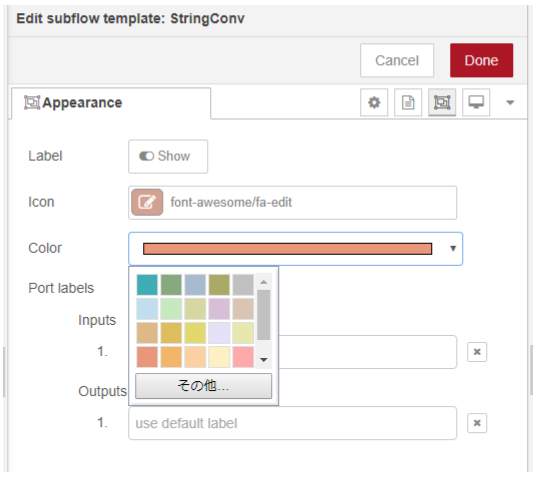
- When exporting SUBFLOW as NODE, we would like to change colors of exported SUBFLOW node.
- Following two approaches are possible:
  a. specify color on export dialog
  b. specify color on node appearance tab



Export As: JSON
Color:

```
{"type":"subflow","sealed":true,"flow
[{"id":"6538c3cf.2e75fc","type":"swit                e98",
"name":"","property":"payload","prope                ":
[{"t":"eq","v":"","vt":"str"},
{"t":"eq","v":"","vt":"str"}],"checka                alse,
"outputs":2,"x":190,"y":80,"wires":[[
["76fdd490.9e57ec"]]},
{"id":"78d791e1.dcc28","type":"change","z":"b81fc5db.d75e98","n
ame":"","rules":
```

Specifying Color
in Export Dialog



Label: Show
Icon: font-awesome/fa-edit
Color:
Inputs 1.
Outputs 1. use default label

Specifying Color
in Node Appearance Tab

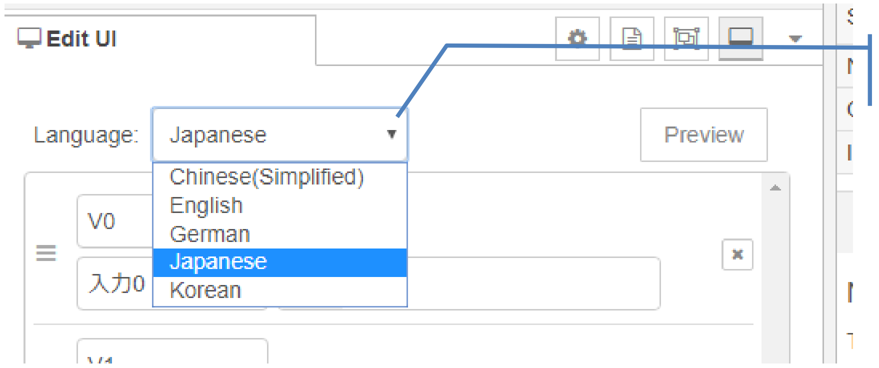
- We think (b) is natural extension of node appearance specification. But may needs discussion because it may allow changing color of every node types.

Demo: UI definition

DEMO

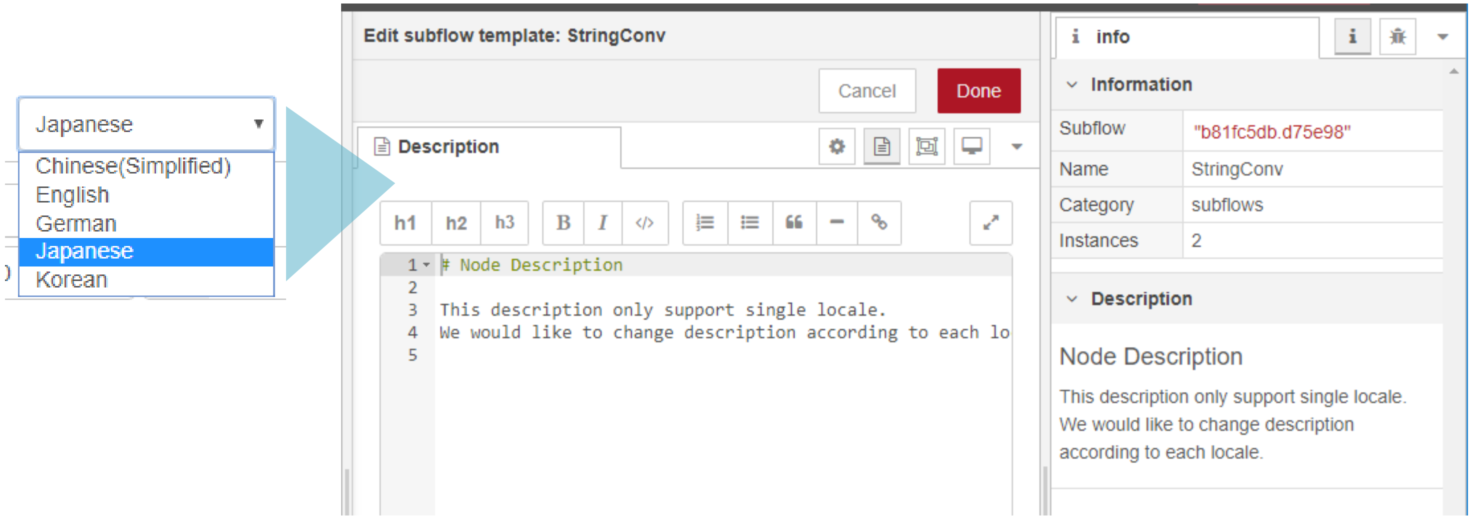
i18n of Node Description

- Similar to this, we would like to make exported SUBFLOW i18 ready.
- UI definition feature allow row labels can be defined for each locales.



locale selector

- We think node description should have similar feature.



```
# Node Description

This description only support single locale.
We would like to change description according to each lo
```

Subflow	"b81fc5db.d75e98"
Name	StringConv
Category	subflows
Instances	2

Description

Node Description

This description only support single locale. We would like to change description according to each locale.

Extension of Function node

- When describing logic in SUBFLOW, function node plays a central role for expressing complex algorithms
- It has following problems:
  a. Can't use external libraries without modifying settings.js,
  b. Execution of function body is performed in VM environment (incur overhead),
  c. Function body is executed each time message is received.
  So, describing common initialization or shutdown code is difficult.
- Proposal:
  1. Add setting to allow use of external libraries and execution without VM module.
  2. Add interface to add npm module,
  3. Add tab to define initialization & shutdown code.

(1)(2) should be disabled or partially enabled by settings.js for safety

1. Background
2. Subflow as Redistributable Node
4. Subflow Enhancements
5. Summary

Summary

- In this session, we described following SUBFLOW related extensions:
  - Exporting SUBFLOW and JSON representation extension,
  - Use of new exporting format,
  - Customization of Subflow (Setting UI and color)

- We would like to know your opinion on current proposal and implementation

HITACHI
Inspire the Next