HARK-ROS Tutorial

(Optional HARK package for ROS users)

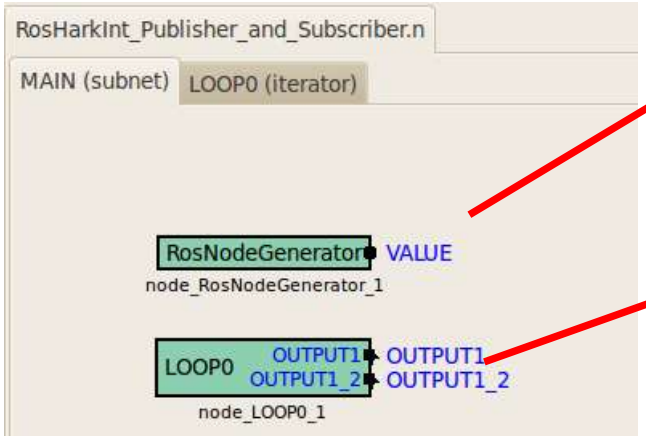
HARK team

Contents

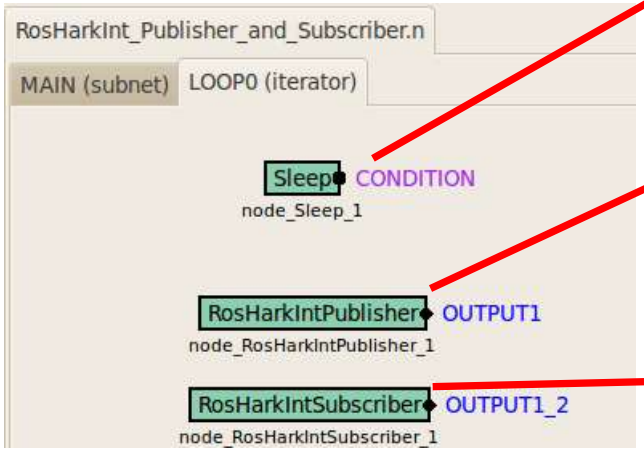
- Publishing/Subscribing “msg” type Topics
  - This tutorial uses HarkInt (just an integer) as the most simple example.
  - Similar to the following ROS tutorial (http://www.ros.org/wiki/ROS/Tutorials/WritingPublisherSubscriber%28c%2B%2B%29)
- Requesting/Responding “srv” type Topics
  - The client sends two integers, and the server returns their summation.
  - Similar to the following ROS tutorial (http://www.ros.org/wiki/ROS/Tutorials/WritingServiceClient%28c%2B%2B%29)
- Dynamic Reconfigure of HARK parameters
  - HarkInt (an integer) is reconfigured as the most simple example.
- Application of publishing/subscribing HARK std messages
- Application for LocalizeMUSIC with dynamic reconfigure

Tutorial1

- **Publishing/Subscribing “msg” type topics**
  - **Main sheet configuration**



  - **Iterator sheet configuration**



ROS node name
Eg) HARK_MASTER_NODE

Keep this blank

Sampling time for HARK Iteration.

- **Published topic name**
- **Buffer size for published topic name**
- **Published value setting (count number + PARAM)**

- **Subscribed topic name**
- **Buffer size for subscribed topic name**
- **Sleep time after subscription**

Tutorial1

Publishing/Subscribing “msg” type topics

Running the HARK network file

- Save the network file before closing [eg) pub_sub_HarkInt.n]
- Open a new terminal and type “% roscore”
- Run the HARK network file : “% ./pub_sub_HarkInt.n”

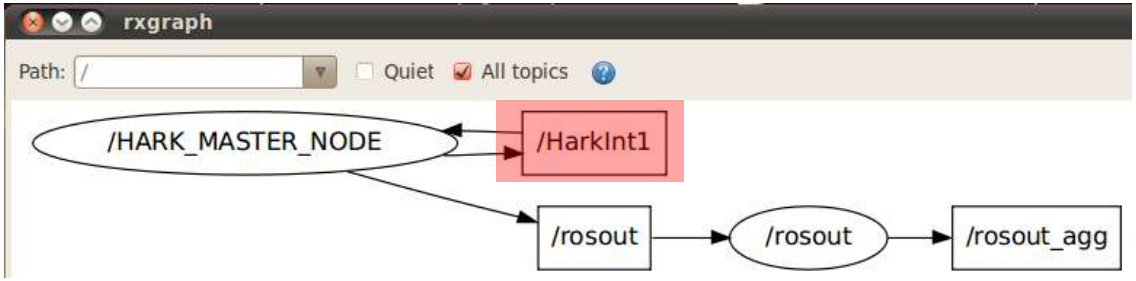
  You’ll see something like following

```
node_RosHarkIntPublisher_1 constructor end...
node_RosHarkIntSubscriber_1 constructor end...
ROS node : HarkRosMasterNode generated...
node_RosHarkIntPublisher_1 initialized...
node_RosHarkIntSubscriber_1 initialized...
ROS initialized...
<Int 0 >
node_RosHarkIntPublisher_1 Published : [0]
node_RosHarkIntSubscriber_1 Subscribed : [0] [ INFO] [1289788809.709227278]: Received [0] [thread=0x8ecd528]
node_RosHarkIntPublisher_1 Published : [1]
node_RosHarkIntSubscriber_1 Subscribed : [0] [ INFO] [1289788809.874858947]: Received [1] [thread=0x8ecd528]
node_RosHarkIntPublisher_1 Published : [2]
node_RosHarkIntSubscriber_1 Subscribed : [1] [ INFO] [1289788810.048147291]: Received [2] [thread=0x8ecd528]
```

If you see some ROS related error here, type
% . ~/ros/setup.sh
on your terminal.

Checking by rxgraph

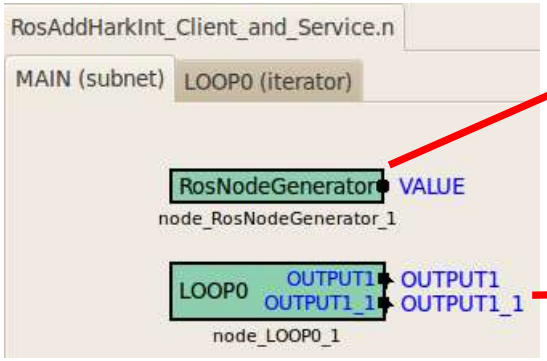
You can see that the HarkInt1 topic is published/subscribed by HARK_MASTER_NODE node.

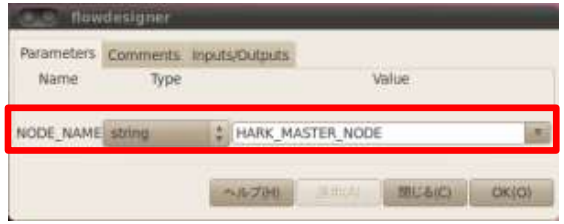


Tutorial2

Requesting/Responding “srv” type topics

Main sheet configuration

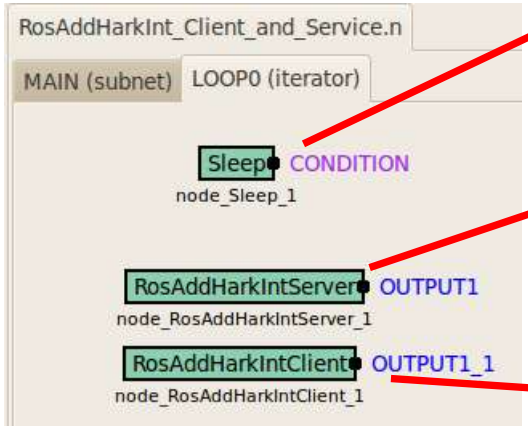




ROS node name
Eg) HARK_MASTER_NODE

Keep this blank

Iterator sheet configuration



Sampling time for HARK Iteration.

- Service topic name
- Buffer size for service
- Sleep time after responce

- Client topic name
- Buffer size for service
- Request parameter setting
  Sends [count number] and [PARAM]

Tutorial2

- **Requesting/Responding “srv” type topics**
  - **Running the HARK network file**
    - Save the network file before closing [eg) srv_cli_HarkInt.n]
    - Open a new terminal and type “% roscore”
    - Run the HARK network file : “% ./srv_cli_HarkInt.n”

      You’ll see something like following

```
node_RosAddHarkIntClient_1 constructor end...
ROS node : HARK_MASTER_NODE generated...
node_RosAddHarkIntClient_1 initialized...
node_RosAddHarkIntServer_1 initialized...
ROS initialized...
<Int 0 >
node_RosAddHarkIntServer_1 Output : [0]
[ INFO] [1289790940.132554321]: Request [0 + 10 = 10] [thread=0xa07fe00]
node_RosAddHarkIntClient_1 Published : [10]
node_RosAddHarkIntServer_1 Output : [10]
[ INFO] [1289790940.300574616]: Request [1 + 11 = 12] [thread=0xa07fe00]
node_RosAddHarkIntClient_1 Published : [12]
node_RosAddHarkIntServer_1 Output : [12]
```

If you see some ROS related error here, type
% . ~/ros/setup.sh
on your terminal.

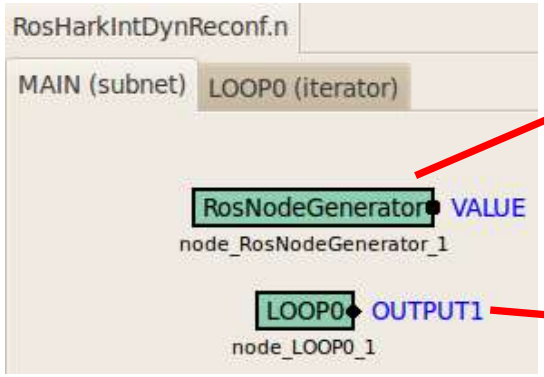
The client sends two integers, [count] and [count + PARAM].

The server calculates the total and sends back to the client.

Tutorial3

Dynamic Reconfigure of HARK parameters

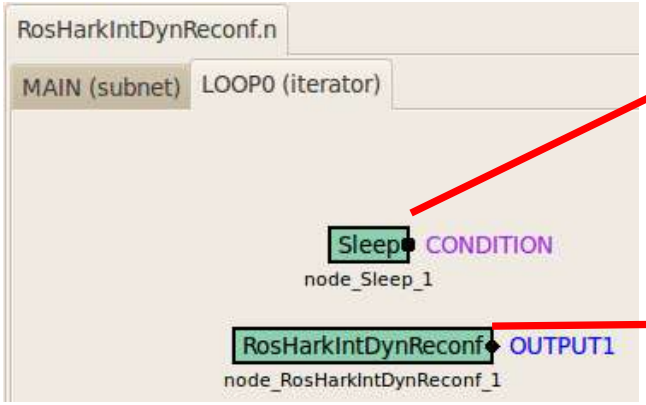
Main sheet configuration



ROS node name
Eg) HARK_MASTER_NODE

Keep this blank

Iterator sheet configuration



Sampling time for HARK Iteration.

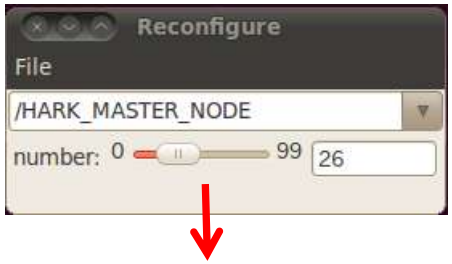
Sleep time after receiving dynamics reconfigure.

Tutorial3

Dynamic Reconfigure of HARK parameters

Running the HARK network file

- Save the network file before closing [eg) DynReconf_HarkInt.n]
- Open a new terminal and type “% roscore”
- Run the HARK network file : “% ./DynReconf_HarkInt.n”
- Open a new terminal and type : “% rosrun dynamic_reconfigure reconfigure_gui”

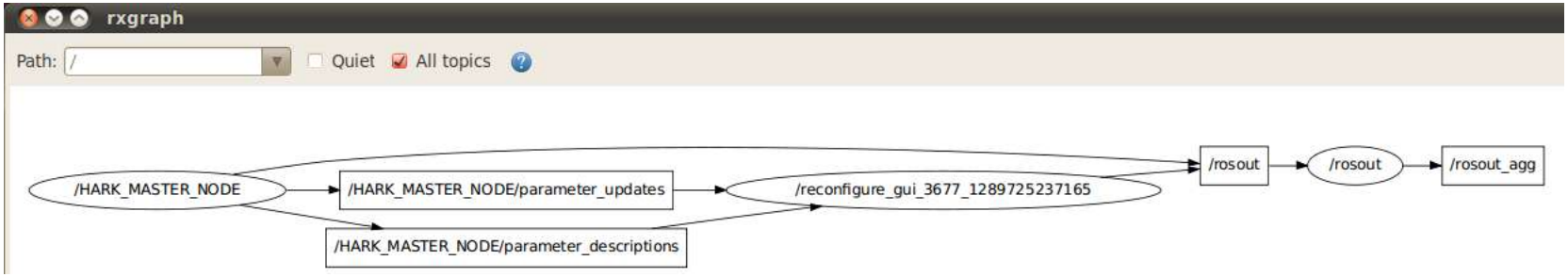


Parameter can be reconfigured through this GUI

```
ROS node : HARK_MASTER_NODE generated...
node_RosHarkIntDynReconf_1 initialized...
[ INFO] [1289793120.876684240]: Received [1] [thread=0x923af18]
ROS initialized...
<Int 0 >
node_RosHarkIntDynReconf_1 Subscribed [1]
node_RosHarkIntDynReconf_1 Subscribed [1]
[ INFO] [1289793129.366559703]: Received [2] [thread=0x91a1c38]
node_RosHarkIntDynReconf_1 Subscribed [2]
 [ INFO] [1289793129.399733376]: Received [3] [thread=0x91a1c38]
 [ INFO] [1289793129.416073718]: Received [5] [thread=0x91a1c38]
 [ INFO] [1289793131.381853268]: Received [26] [thread=0x91a1c38]
node_RosHarkIntDynReconf_1 Subscribed [26]
node_RosHarkIntDynReconf_1 Subscribed [26]
```

Reconfigured

rxgraph shows like…

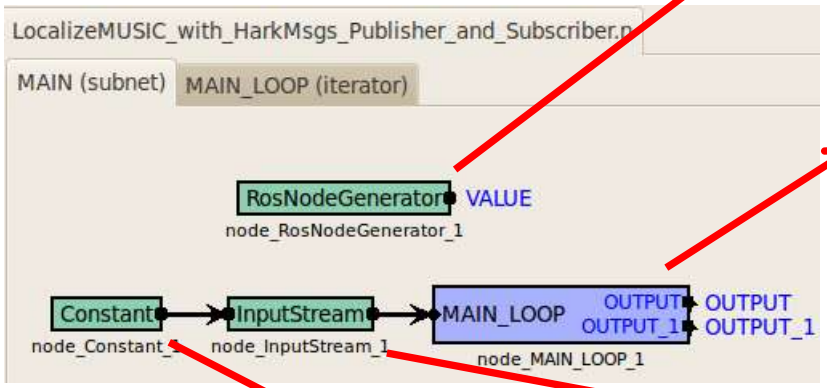


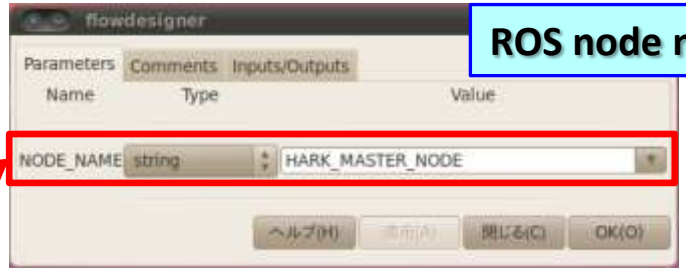
Tutorial4

Publishing/Subscribing HARK standard messages

Here, we construct a network for Sound Source Localization, and sound locations are published/subscribed as ROS topics.

Main sheet configuration

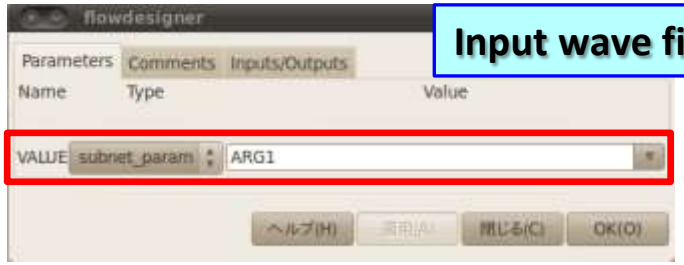




ROS node name

FFT window length & Shift length

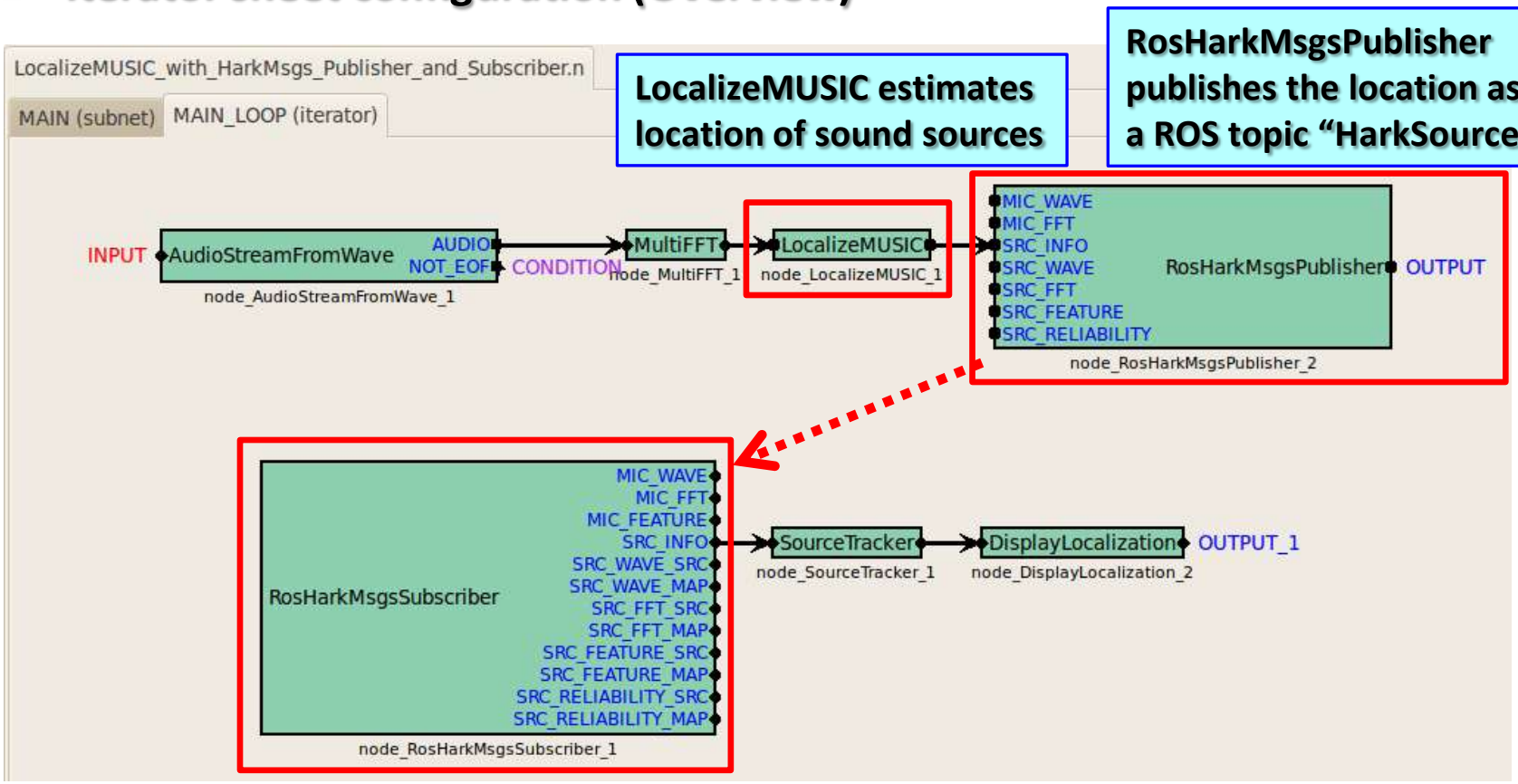
Keep this blank



Input wave file name

Tutorial4

- **Publishing/Subscribing HARK standard messages**
  - **Iterator sheet configuration (Overview)**



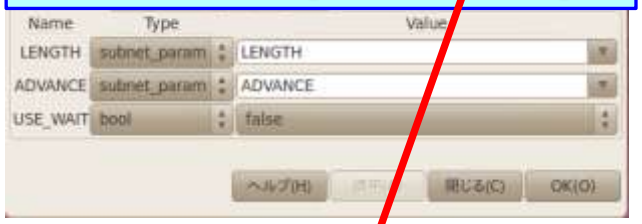
RosHarkMsgsSubscriber subscribes “HarkSource” and outputs it as HarkIO.

Optionally, you can use separated HARK network files for upper and lower networks.

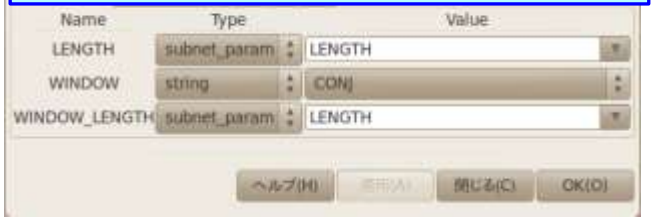
Tutorial4

- **Publishing/Subscribing HARK standard messages**
  - **Iterator sheet configuration (Upper side)**

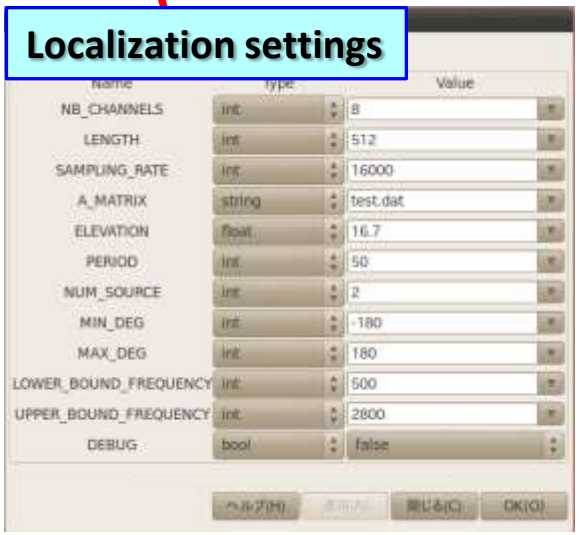
FFT window length & shift length



FFT window setting & shift length



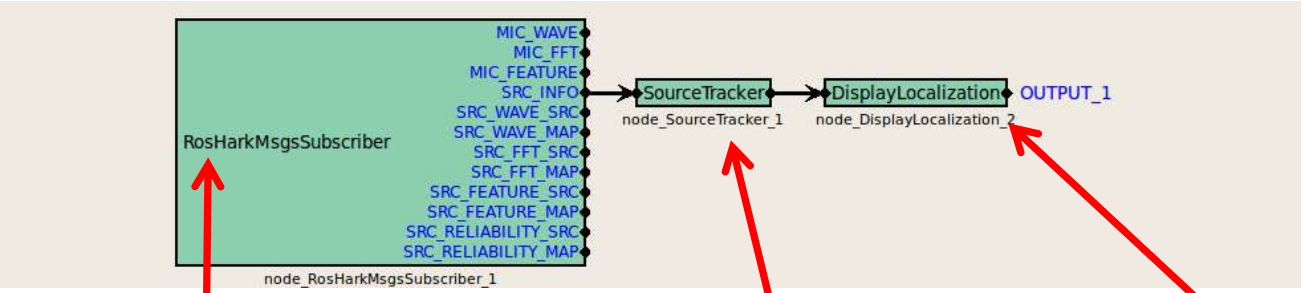
Localization settings



ROS topic name

Tutorial4

- **Publishing/Subscribing HARK standard messages**
  - **Iterator sheet configuration (Lower side)**



ROS topic name

Name	Type	Value
NB_CHANNELS	int	8
FFT_LENGTH	subnet_param	LENGTH
ENABLE_DEBUG	bool	true
TOPIC_NAME_HARKWAVE	string	HarkWave
TOPIC_NAME_HARKFFT	string	HarkFFT
TOPIC_NAME_HARKFEATURE	string	HarkFeature
TOPIC_NAME_HARKSOURCE	string	HarkSource
TOPIC_NAME_HARKSRCWAVE	string	HarkSrcWave
TOPIC_NAME_HARKSRCFFT	string	HarkSrcFFT
TOPIC_NAME_HARKSRCFEATURE	string	HarkSrcFeature
TOPIC_NAME_HARKSRCFEATUREMFM	string	HarkSrcFeatureMFM
ROS_LOOP_RATE	float	1000000
MSG_BUFFER_NUM	int	100
DATA_BUFFER_NUM	int	100

ヘルプ(H) 適用(A) 閉じる(C) OK(O)

Keep this blank

NameTypeValue

ヘルプ(H) 適用(A) 閉じる(C) OK(O)

Sound Activity Detection Parameters

Name	Type	Value
THRESH	float	32
PAUSE_LENGTH	float	1300
MIN_SRC_INTERVAL	float	20
DEBUG	bool	false

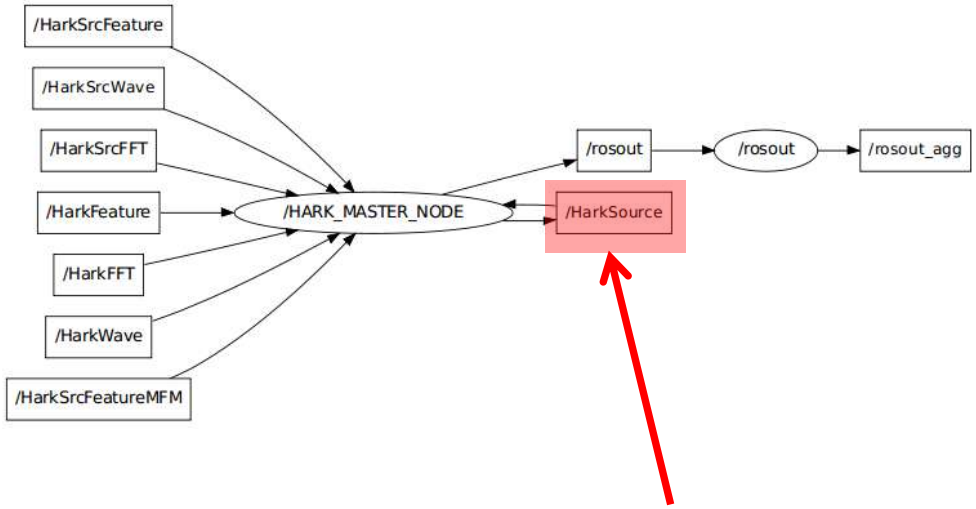
ヘルプ(H) 適用(A) 閉じる(C) OK(O)

Tutorial4

Publishing/Subscribing HARK standard messages

- **Running the HARK network file**
  - Save the network file before closing [eg) pub_sub_Localization.n]
  - Open a new terminal and type “% roscore”
  - Run the HARK network file : “% ./pub_sub_Localization.n your_wav_file.wav”

- **rxgraph shows like…**



- **rostopic list shows like…**

```
/HarkFFT
/HarkFeature
/HarkSource
/HarkSrcFFT
/HarkSrcFeature
/HarkSrcFeatureMFM
/HarkSrcWave
/HarkWave
/rosout
/rosout_agg
```

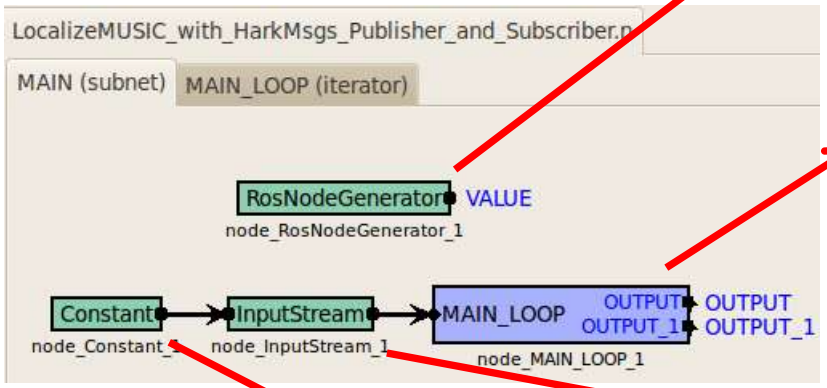
HarkSource is published/subscribed in the network file.

Tutorial5

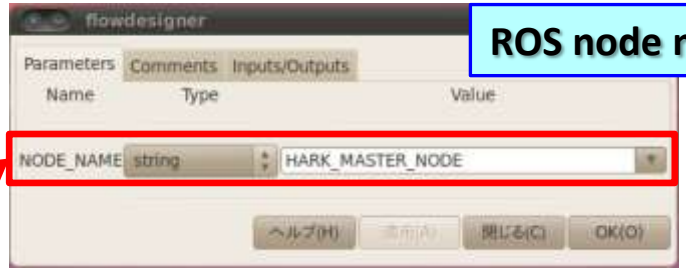
Localization with the Dynamic Reconfigure

Here, parameters of modules, "LocalizeMUSIC" and "SourceTracker" are dynamically reconfigured.

Main sheet configuration



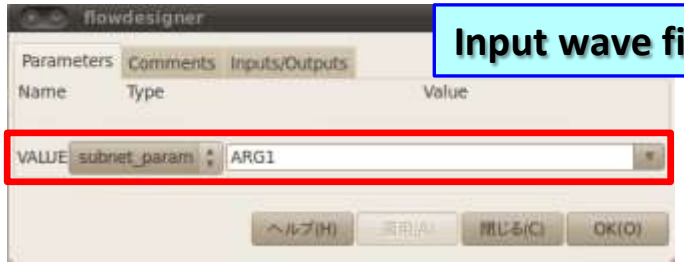
The main sheet is exactly the same as tutorial 4.



ROS node name

FFT window length & Shift length

Keep this blank

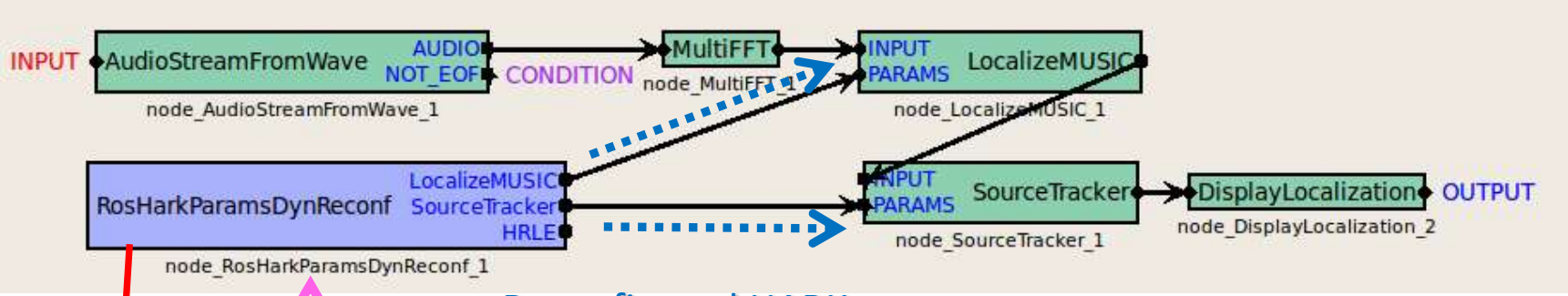


Input wave file name

Tutorial5

Localization with the Dynamic Reconfigure

Iterator sheet configuration

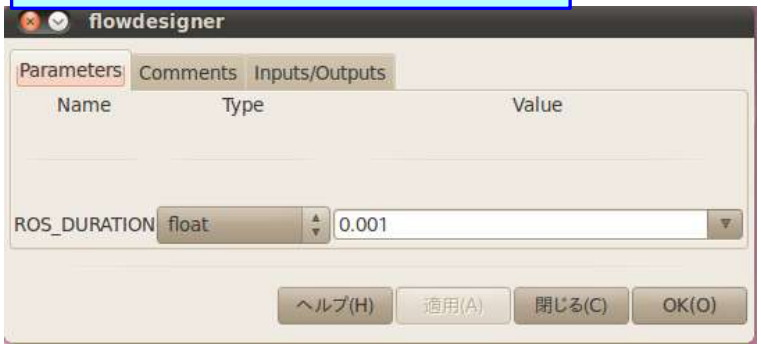


Reconfigured HARK parameters

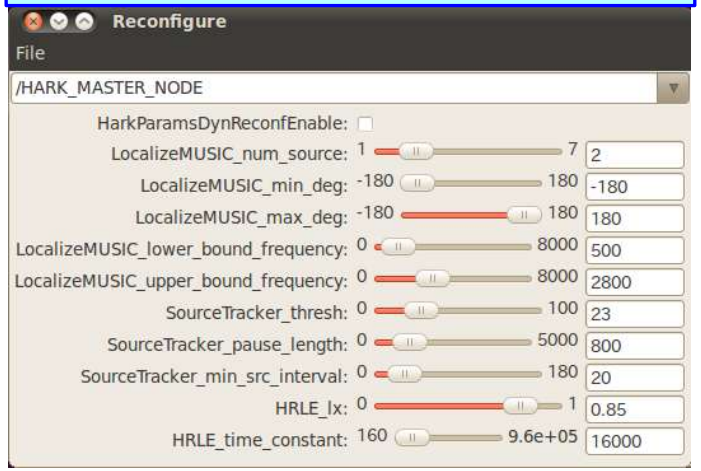
This module receives reconfigured parameters from GUI and outputs them as Hark parameters.

Reconfigured parameters

Sleep time after receiving the dynamic reconfigure



Dynamic reconfigure GUI (in ROS)



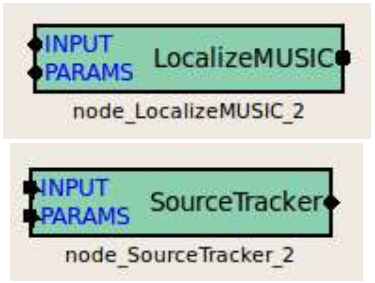
Tutorial5

Localization with the Dynamic Reconfigure

How to add PARAMS input for modules?

Original Modules

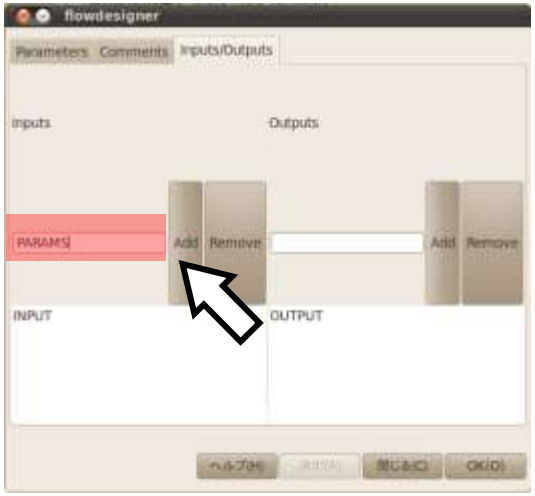
Modules with hidden inputs



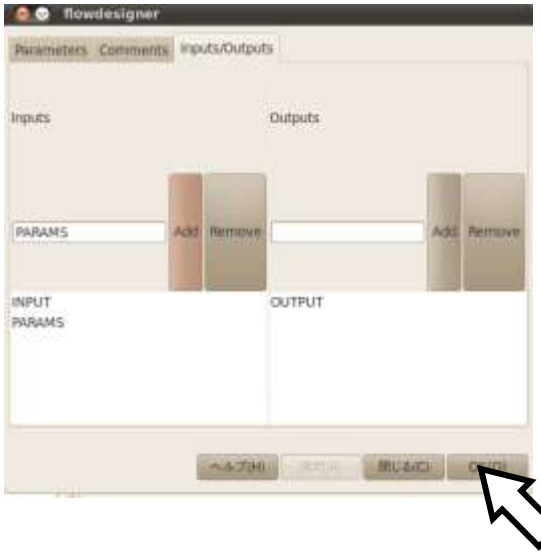
Dynamic reconfigure function uses hidden input ports (PARAMS).

1. Click “Inputs/Outputs” tab

2. Input “PARAMS” in the form. Then press “Add “



3. Press “OK”



Tutorial5

Localization with the Dynamic Reconfigure

Running the HARK network file

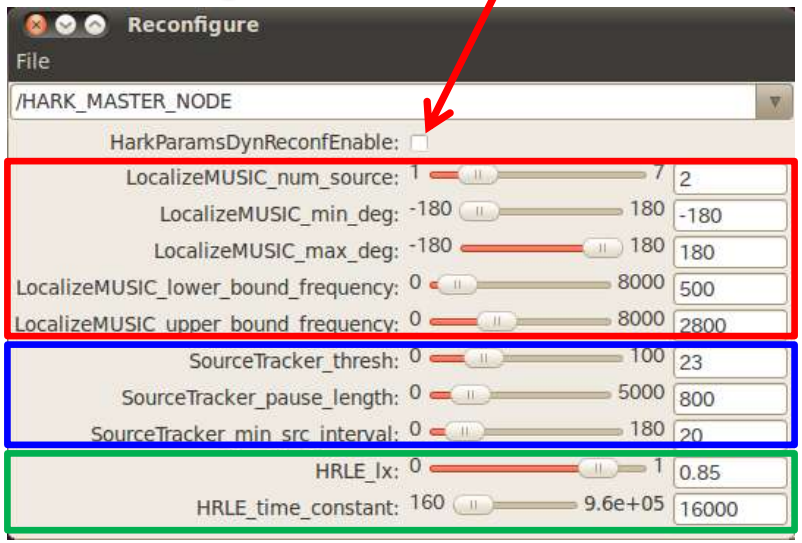
- Save the network file before closing [eg) DynReconf_Localization.n]
- Open a new terminal and type “% roscore”
- Run the HARK network file : “% ./DynReconf_Localization.n your_wav_file.wav”
- Open a new terminal and type : “% rosrun dynamic_reconfigure reconfigure_gui”

rostopic list shows like…

```
/HARK_MASTER_NODE/parameter_descriptions
/HARK_MASTER_NODE/parameter_updates
/rosout
/rosout_agg
```

Reconfigure GUI

Check this to set reconfigure enable



Parameters for Sound Source Localization

Parameters for Sound Activity Detection

Parameters for Speech Enhancement

rxgraph shows like…