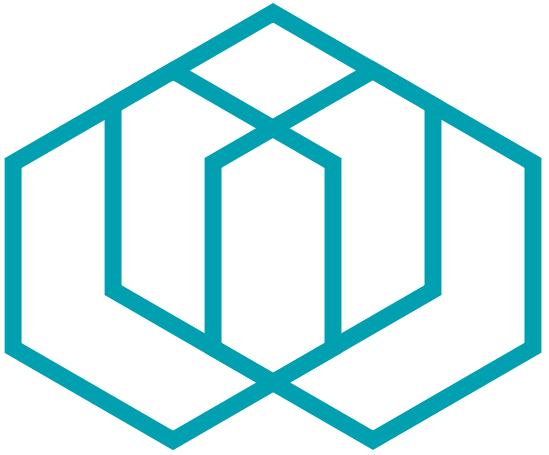


RINASim

BASICS

Vladimír Veselý
Brno University of Technology, Czech Republic
veselyv@fit.vutbr.cz

Agenda

1) Introduction
2) OMNeT++ Handbook
3) RINASim overview
4) Interactive Demo
5) Conclusion

1) Introduction

How to install?
Where to get it?
Documentation and useful links.

Requirements

- OMNeT++ discrete event simulator
  - Windows, Linux, FreeBSD environment
  - Free for non-commercial purposes
  - Supported on version 5.0
    - The last release compatible with OMNeT++ 4.6 is https://github.com/kvetak/RINA/releases/tag/August2016-v4
  - No other libraries or frameworks needed
    - Potential cooperation with INET framework

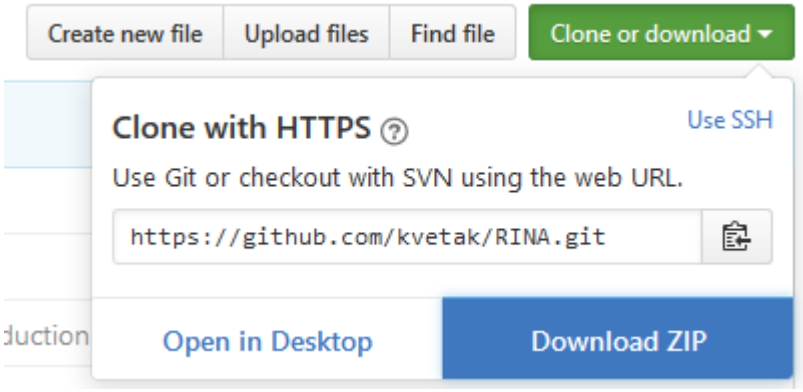
VM Install

- Out-of-the box virtual machine
  - OMNeT++ 5.0 with the newest RINASim
  - Generic OVA appliance
    - MintLinux
  - Created on VMWare Workstation
    - *…should work also on VirtualBox and Qemu*

- Download from http://nes.fit.vutbr.cz/ivesely/vm/RINASim.zip

Windows Installation

- Uses MINGW32_NT-10.0-WOW
- Cookbook
  1) Download OMNeT++
     https://omnetpp.org/omnetpp/send/30-omnet-releases/2307-omnetpp-50-windows
  2) ./configure && make
     http://omnetpp.org/doc/omnetpp/InstallGuide.pdf
  3) Download RINASim
     https://github.com/kvetak/RINA/archive/master.zip
  4) Import RINASim project



Installation Linux

- gcc 4.9.2 with C++11 support
- Cookbook

1) Requirements

   build-essential gcc g++ bison flex perl tcl-dev tk-dev libxml2-dev zlib1g-dev default-jre doxygen graphviz libwebkitgtk-1.0-0

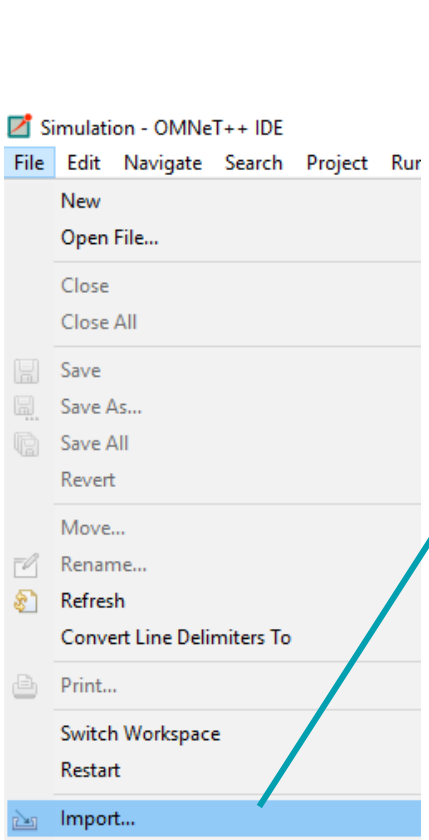
2) Download OMNeT++
   https://omnetpp.org/omnetpp/send/30-omnet-releases/2305-omnetpp-50-linux

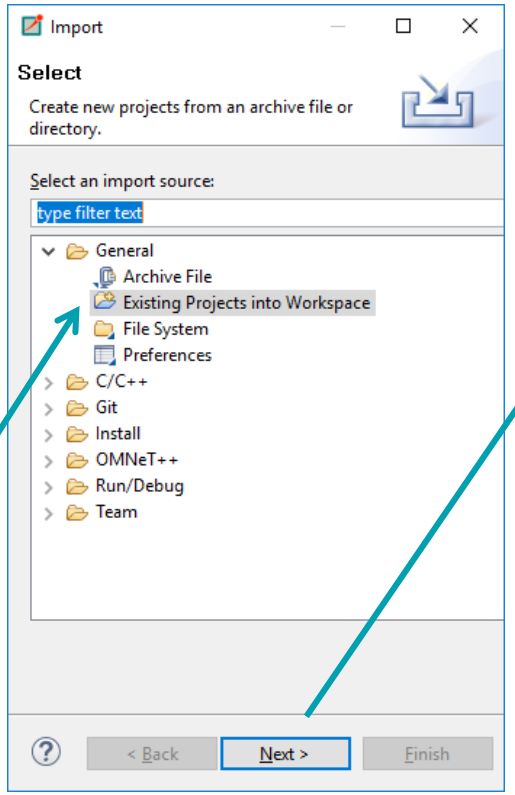
3) ./configure && make
   http://omnetpp.org/doc/omnetpp/InstallGuide.pdf

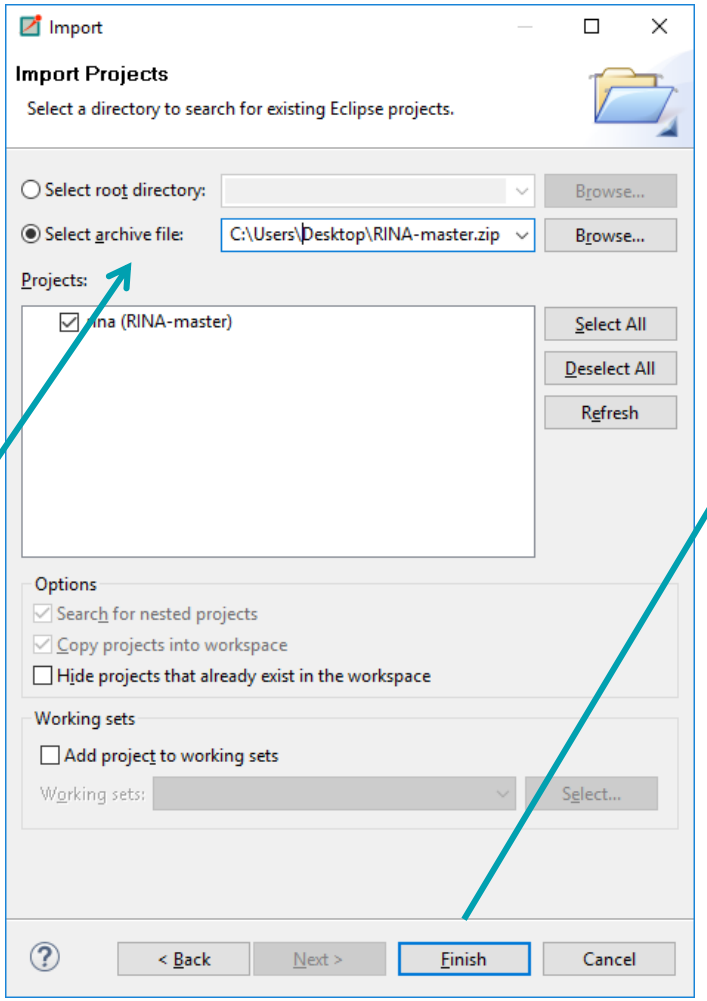
4) Download RINASim
   https://github.com/kvetak/RINA/archive/master.zip

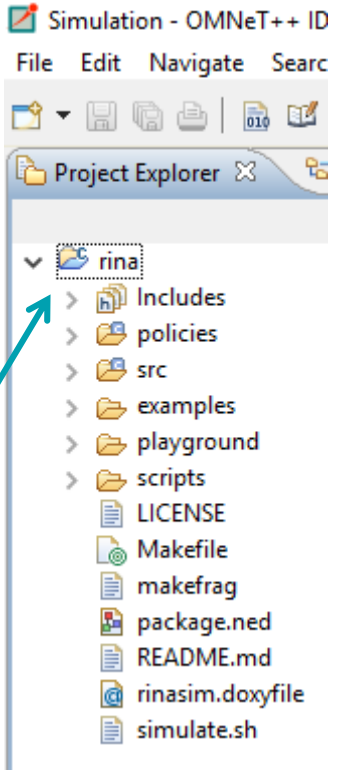
5) Import RINASim project

Importing RINASim Project







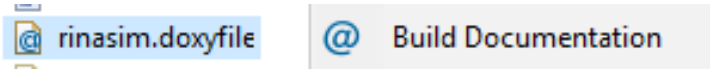


Navigation

- **/src** … RINASim core source codes
  - **/Common** … common or shared ADTs
  - **/CS** … high-level nodes
  - **/DAF** … DAF components
  - **/DIF** … DIF components
- **/policies** … programmable set of policies
- **/examples** … accompanied scenarios
- **/playground** … unmainted/experimental scenarios
- **/out** … compiled project binaries
- **/doc** … Doxygen source code documentation
- **/scripts** … devel batches for SLOC and fingerprinting

Documentation

- Doxygen
  - http://nes.fit.vutbr.cz/ivesely/doxy
  - Important ADTs, ongoing work
  - Build it in folder **/doc**



- PRISTINE D2.4 and D2.6
  - http://ict-pristine.eu/?p=472
  - http://ict-pristine.eu/?p=772

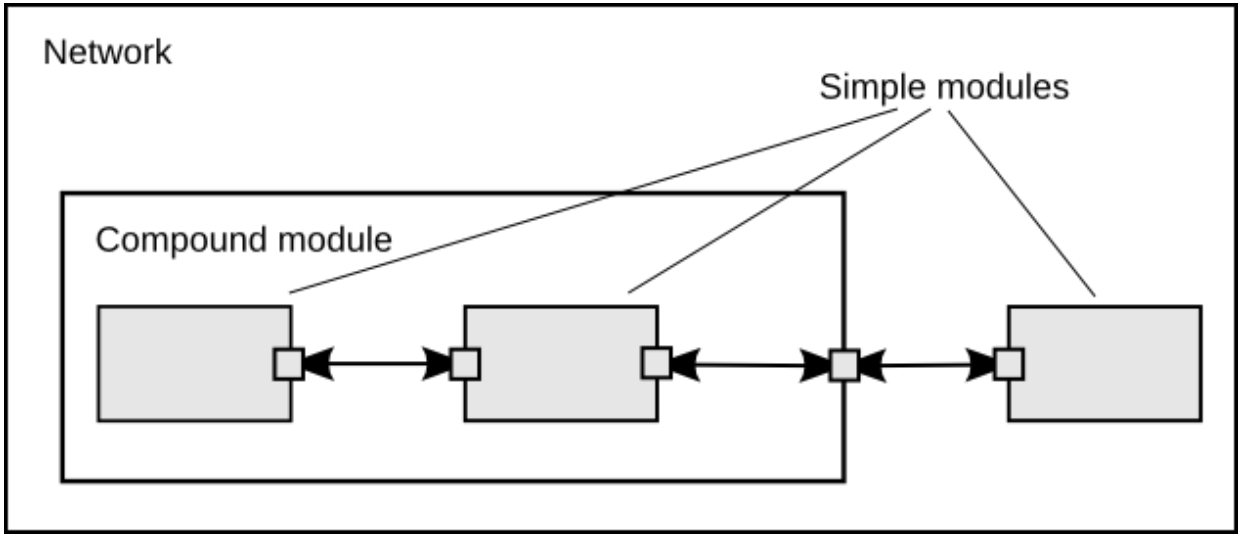
- Nasty details about RINASim are in my dissertation
  - VESELÝ, Vladimír. *A NEW DAWN OF NAMING, ADDRESSING AND ROUTING ON THE INTERNET*. Brno, 2016. Available from: http://www.fit.vutbr.cz/study/DP/PD.php?id=515. PhD. Thesis. Brno University of Technology, Faculty of Information Technology. 2016-04-12. Supervisor Švéda Miroslav.

2) OMNeT++ Handbook

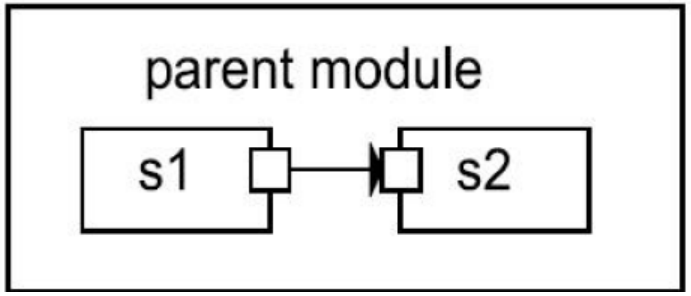
The fast and the furios introductory course for OMNeT++ discrete event simulator

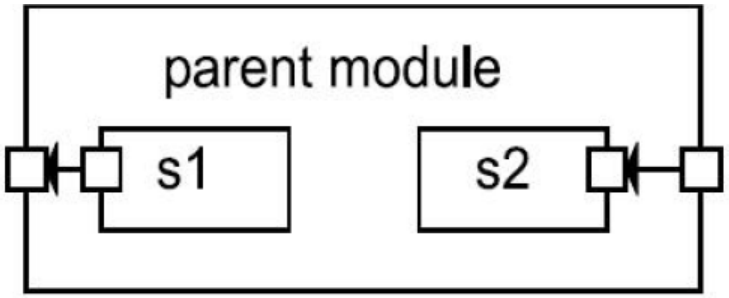
Simulation Module

- Modules



- Gates (input, output, inout), messages





Languages

- NED
  - to define models and interconnections
  - `*.ned`
- C++
  - to implement model behavior
  - `*.h` and `*.cc`
- Message definition
  - to deploy easily C++ message classes
  - `*.msg`

Module's Hierarchy

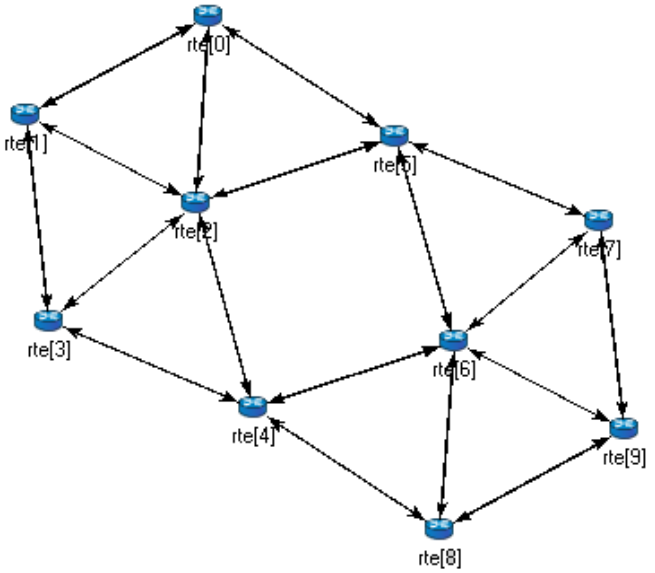
- Simple

```
simple TestModul
{
    parameters:
        @display("i=block/queue");
    gates:
        input in;
        output out;
}
```

- Compound

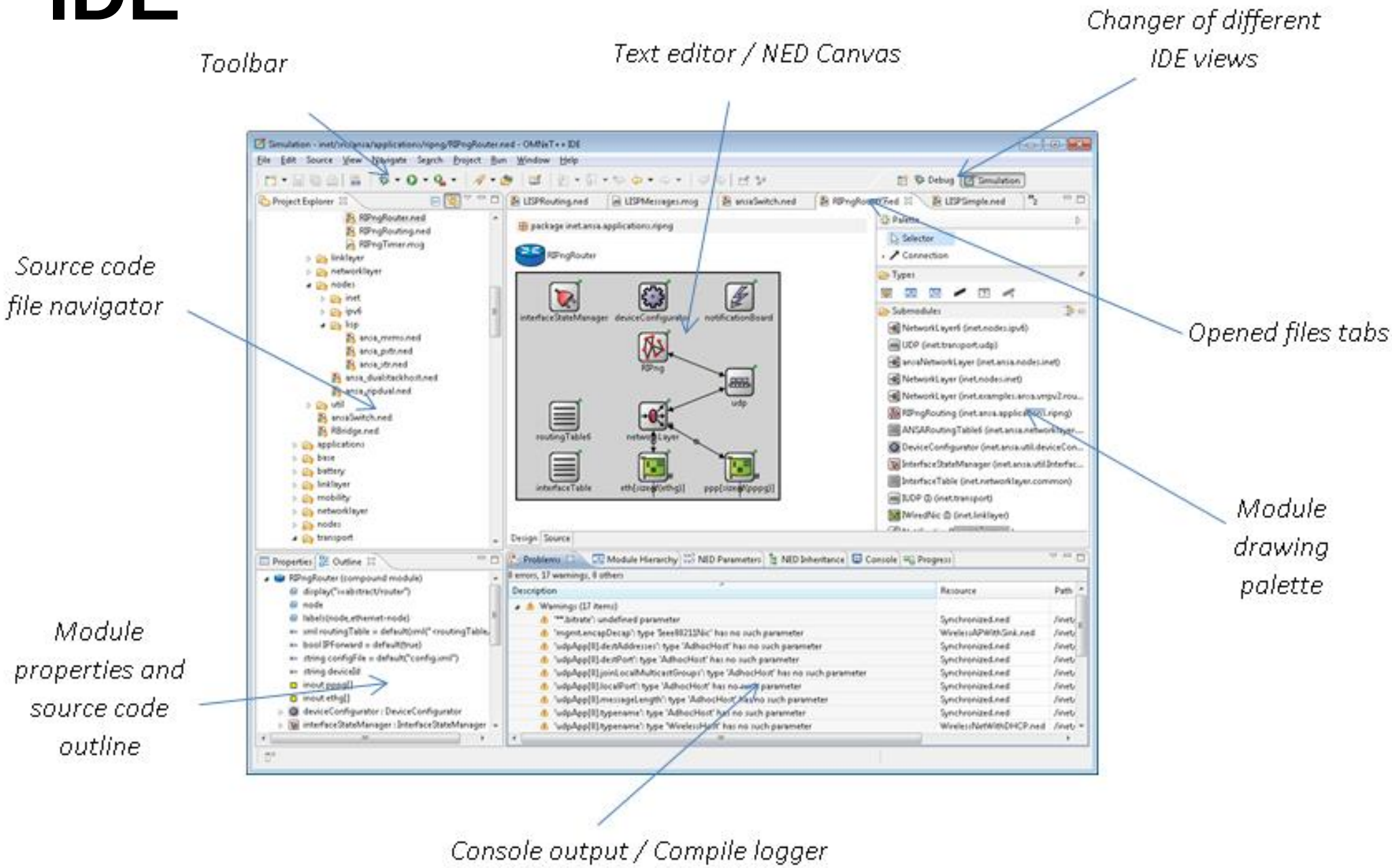
```
module Router
{
    parameter:
        @display("i=block/router");
    gates:
        inout SerialInterface[];
        inout EthInterface[];
    submodules:
        tcp: TCP;
        ip: IP;
        layer1: physicalLayer;
    connections:
        tcp.ipIn <-- ip.tcpOut;
        tcp.ipOut --> ip.tcpIn;
        layer1.ipIn <-- ip.l1In;
        layer1.ipOut --> ip.l1Out;
}
```

- Network




```
//
// A network
//
network Network
{
    submodules:
        node1: Node;
        node2: Node;
        node3: Node;
        ...
    connections:
        node1.port++ <--> {datarate=100Mbps;} <--> node2.port++;
        node2.port++ <--> {datarate=100Mbps;} <--> node4.port++;
        node4.port++ <--> {datarate=100Mbps;} <--> node6.port++;
        ...
}
```

IDE

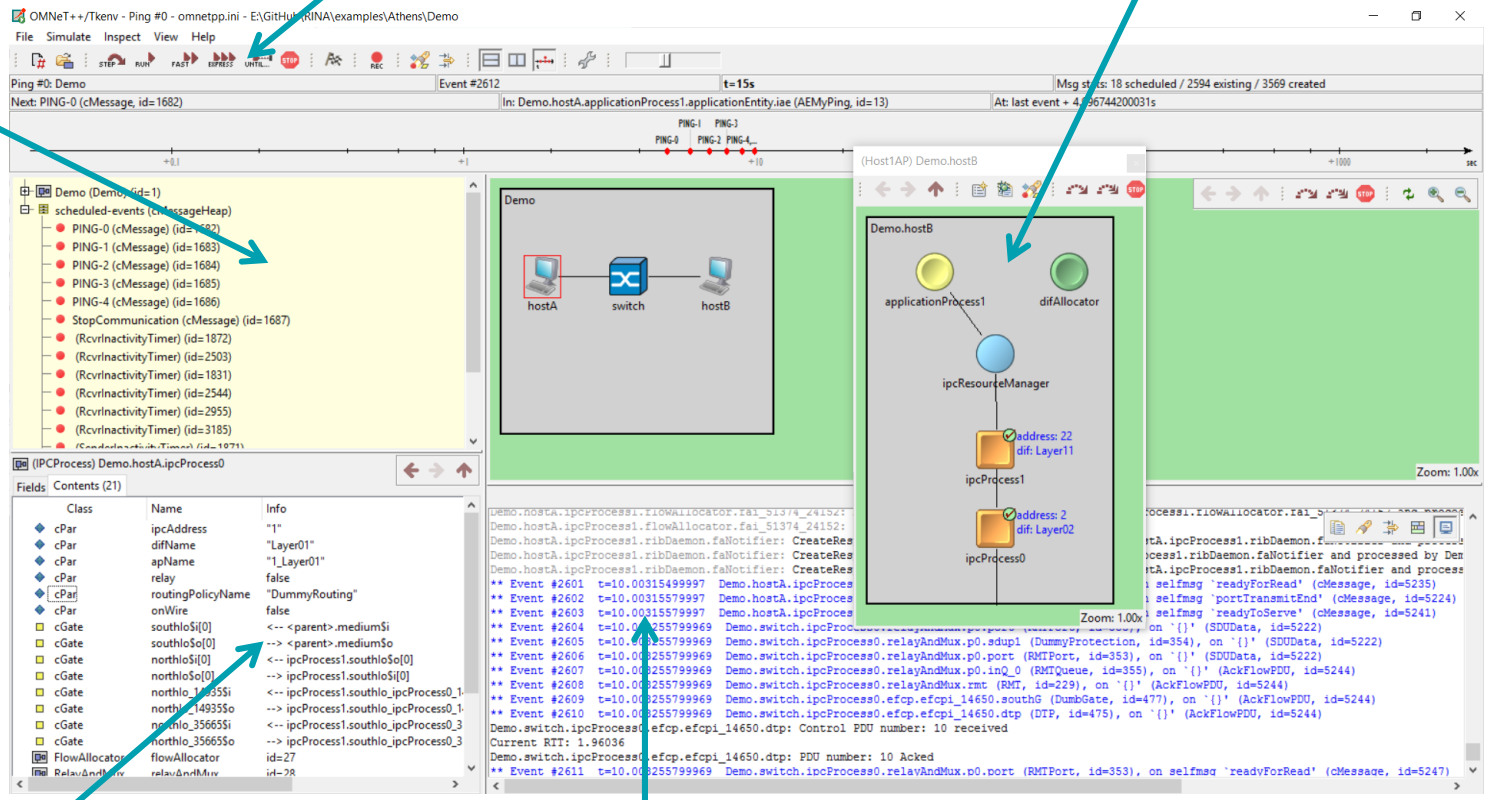

Toolbar
Text editor / NED Canvas
Changer of different IDE views
Source code file navigator
Opened files tabs
Module drawing palette
Module properties and source code outline
Console output / Compile logger

Simulation

Control buttons

Detached graphical component detail

Scheduled messages



Component parameters

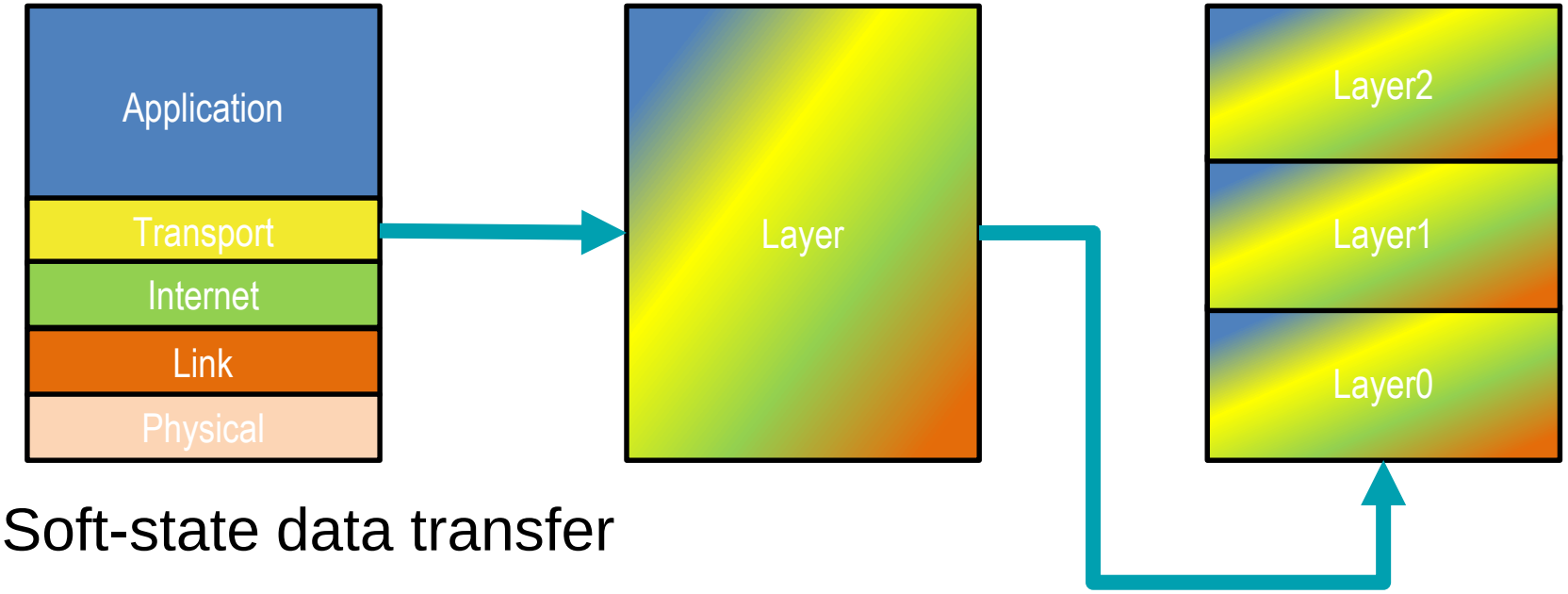
Console log

3) RINASim Overview

How are things stitched together?
Components and their purpose

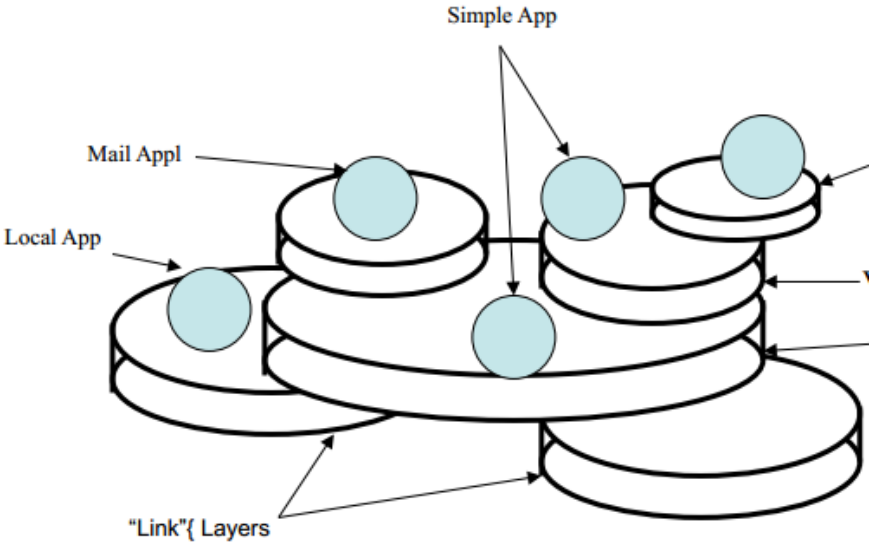
TCP/IP vs RINA

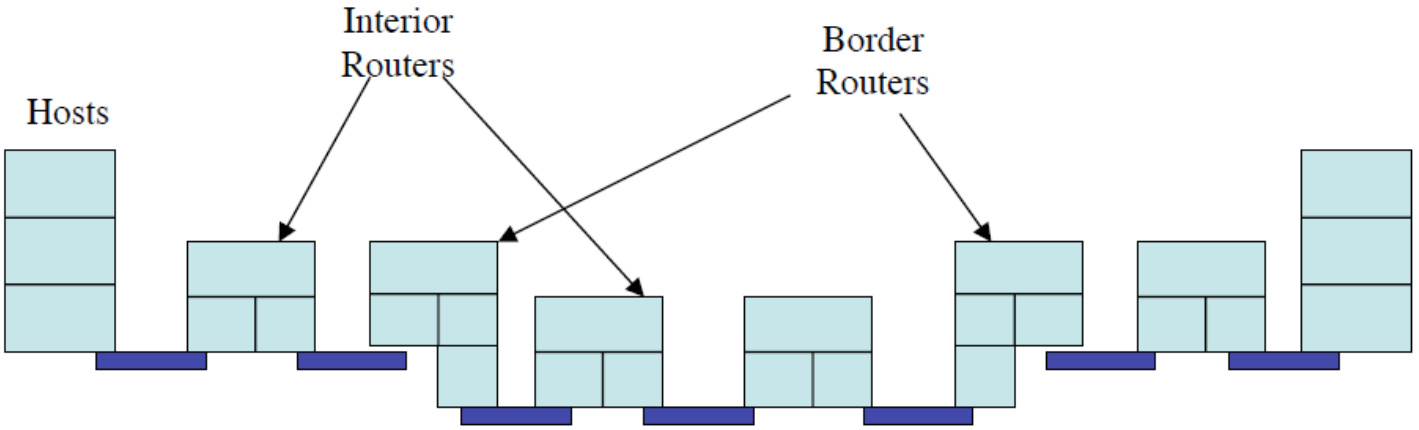
- One generic layer for interprocess communication called **DIF** that limits scope

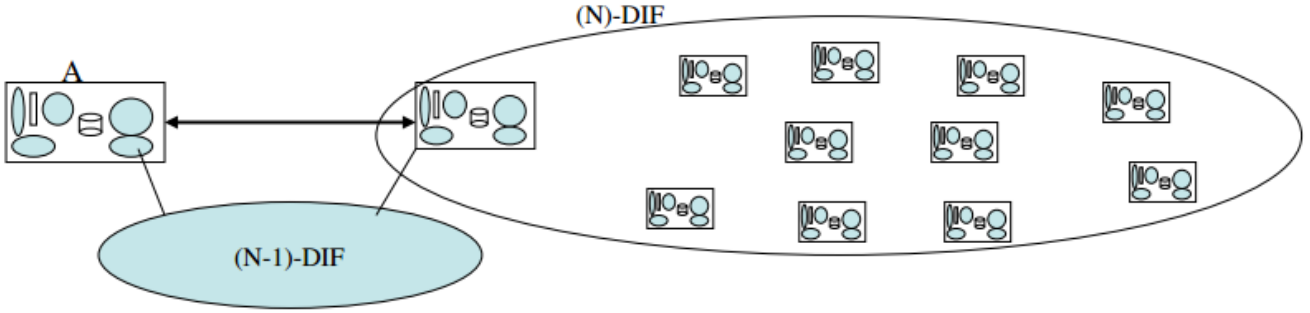


1) Soft-state data transfer
2) Complete addressing and naming
3) Single generic application protocol
4) Split between fixed mechanism and programmable policy

RINA Network







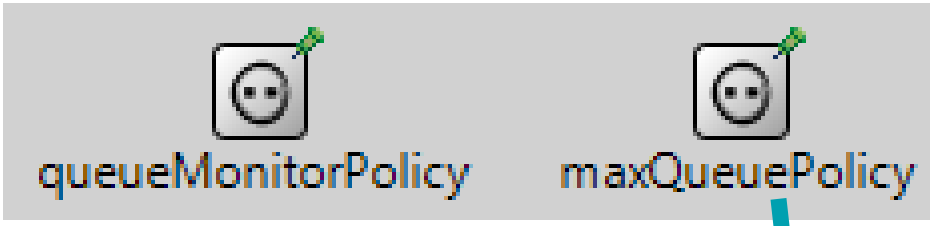
Pictures are taken from John Day's presentation at http://www.slideshare.net/ictpristine/rina-introduction-part-ii

Design

- Split between mechanism and policy
  - Modules as interfaces

NED defines mechanism interface

```
queueMonitorPolicy: <qMonitorPolicyName> like IntRMTQMonitorPolicy {
    @display("p=295,55;is=s");
}
maxQueuePolicy: <maxQPolicyName> like IntRMTMaxQPolicy {
    @display("p=415,55;is=s");
}
```



omnetpp.ini binds a policy

```
#Congestion parameters
**.interiorRouter.relayIpc.relayAndMux.defaultMaxQLength = 5
**.interiorRouter.relayIpc.relayAndMux.defaultThreshQLength = 3
**.interiorRouter.relayIpc.relayAndMux.maxQPolicyName = "REDDropper"
**.interiorRouter.relayIpc.relayAndMux.qMonitorPolicyName = "REDMonitor"
```

Configuring Parameters ①

- *How to setup your simulation?*

- Statically preconfigured
  - NED parameters in `omnetpp.ini`
  - More complex configuration (lists) in `config.xml`

- On-the-fly
  - not supported everywhere

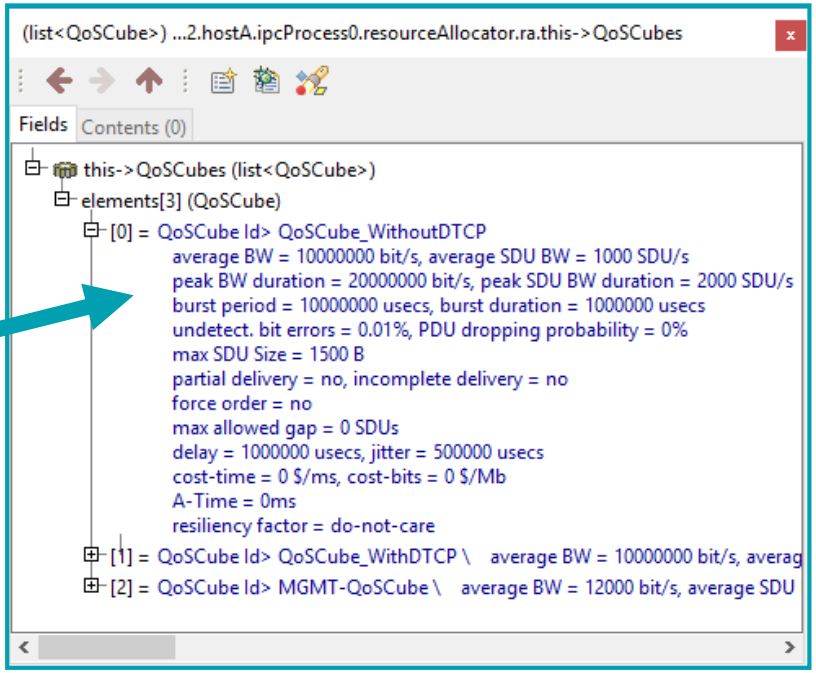
Configuring Parameters ②

omnetpp.ini

```
#Static addressing
**.hostA.ipcProcess0.ipcAddress = "1"
**.hostA.ipcProcess0.difName = "Layer0"
**.hostB.ipcProcess0.ipcAddress = "2"
**.hostB.ipcProcess0.difName = "Layer0"
#QoS settings
**.ra.qoscubesData = xmldoc("config.xml", "Configuration/QoSCubesSet")
```

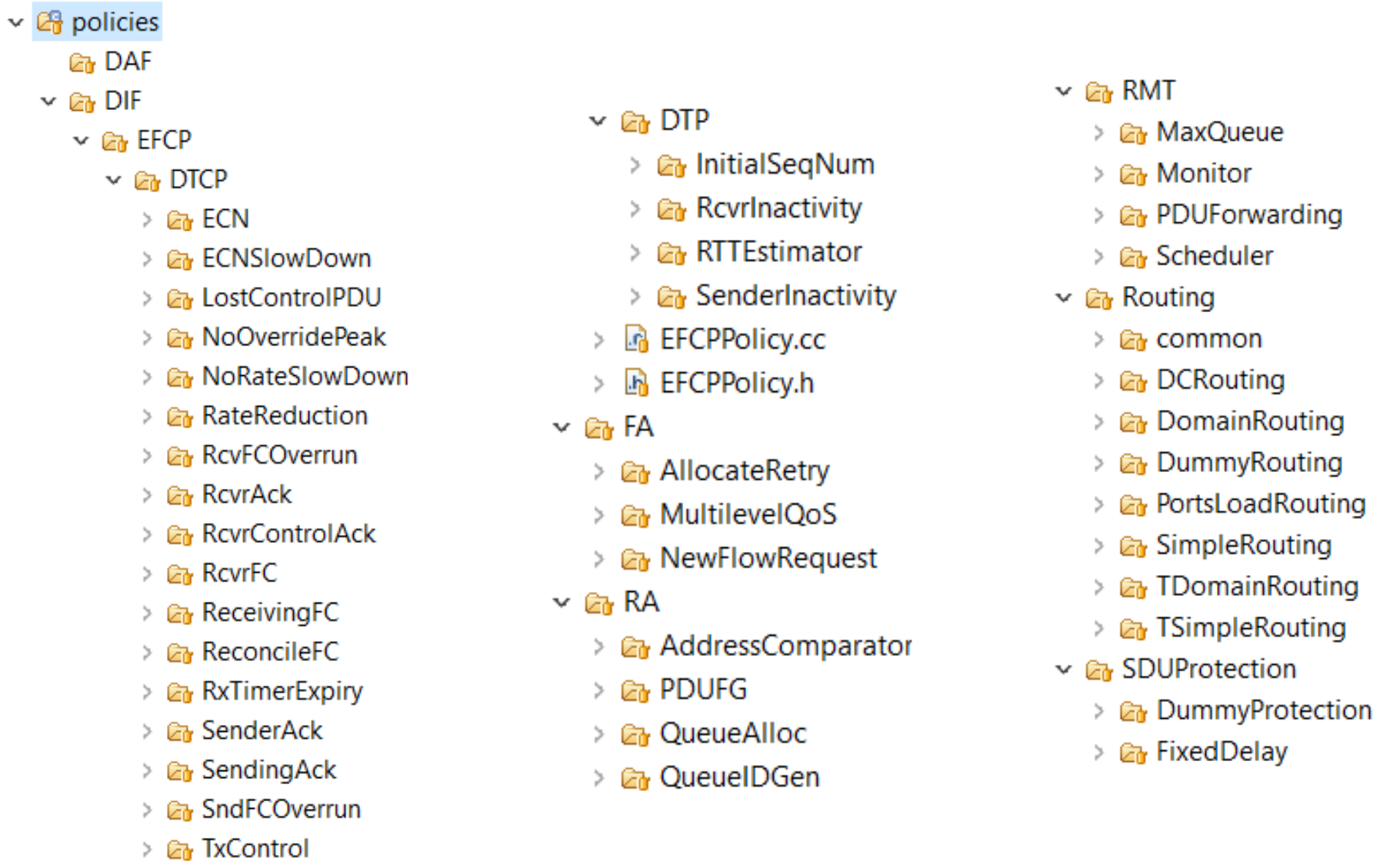
config.xml

```
<Configuration>
    <Host id="host">
        <DA>
            <Directory>
                <APN apn="SourceA">
                    <DIF difName="Layer0" ipcAddress="1" />>
                </APN>
                <APN apn="DestinationB"
                    <DIF difName="Layer0" ipcAddress="2" />
                </APN>
            </Directory>
        </DA>
    </Host>
    <QoSCubesSet>
        <QoSCube id="QoSCube_WithoutDTCP">
            <AverageBandwidth>10000000</AverageBandwidth>
            <AverageSDUBandwidth>1000</AverageSDUBandwidth>
            <PeakBandwidthDuration>20000000</PeakBandwidthDuration>
            <PeakSDUBandwidthDuration>2000</PeakSDUBandwidthDuration>
            <BurstPeriod>10000000</BurstPeriod>
            <BurstDuration>1000000</BurstDuration>
            <UndetectedBitError>0.01</UndetectedBitError>
            <PDUDroppingProbability>0</PDUDroppingProbability>
            <MaxSDUSize>1500</MaxSDUSize>
            <PartialDelivery>0</PartialDelivery>
            <IncompleteDelivery>0</IncompleteDelivery>
            <ForceOrder>0</ForceOrder>
```

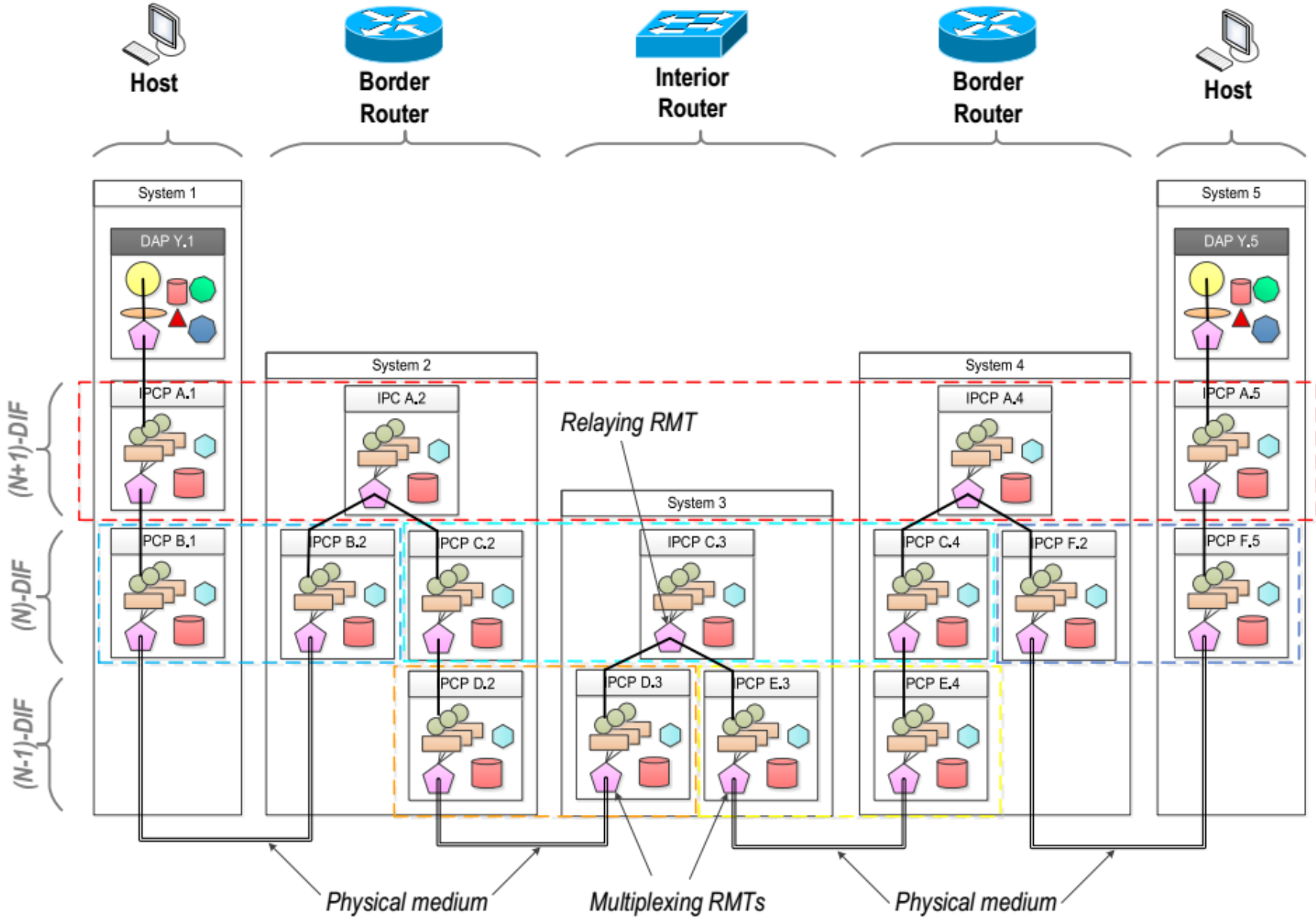


running simulation

Policy Implementations

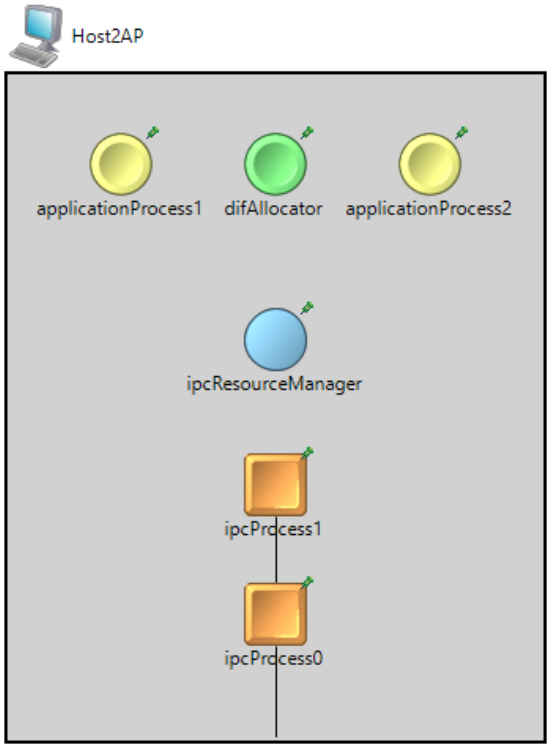


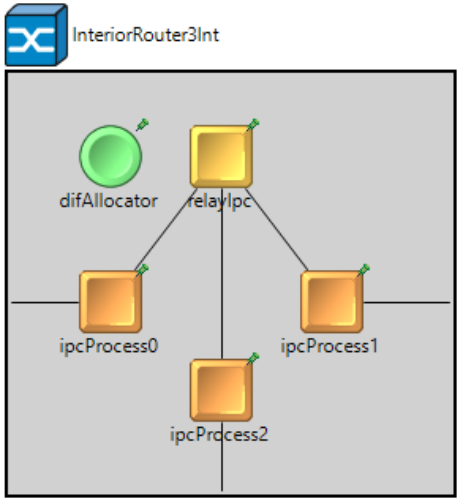
Computation Systems ①

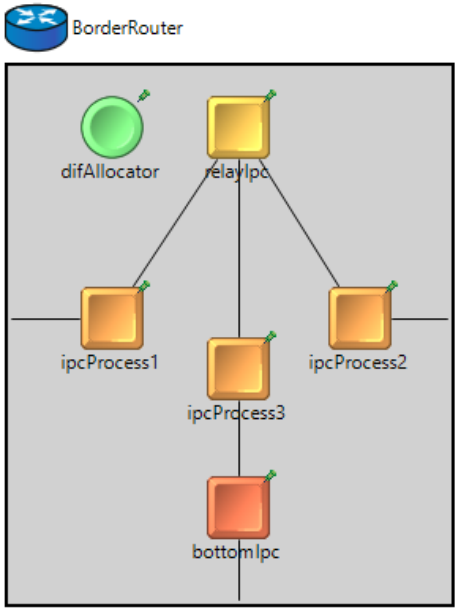


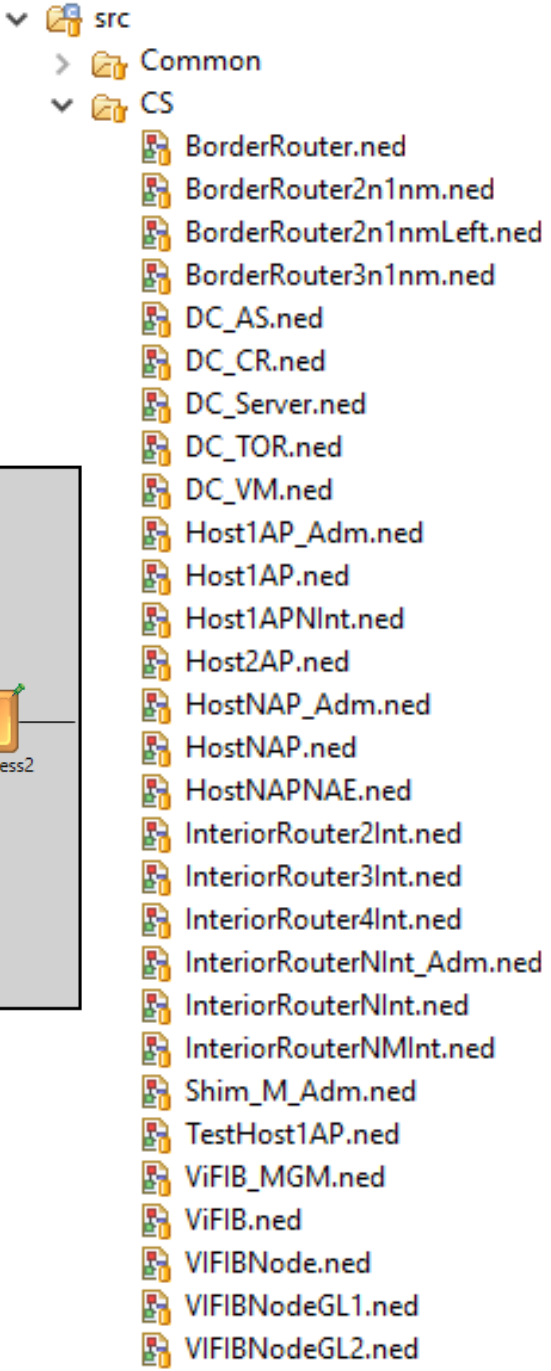
Computation Systems ②

- Hosts and Routers



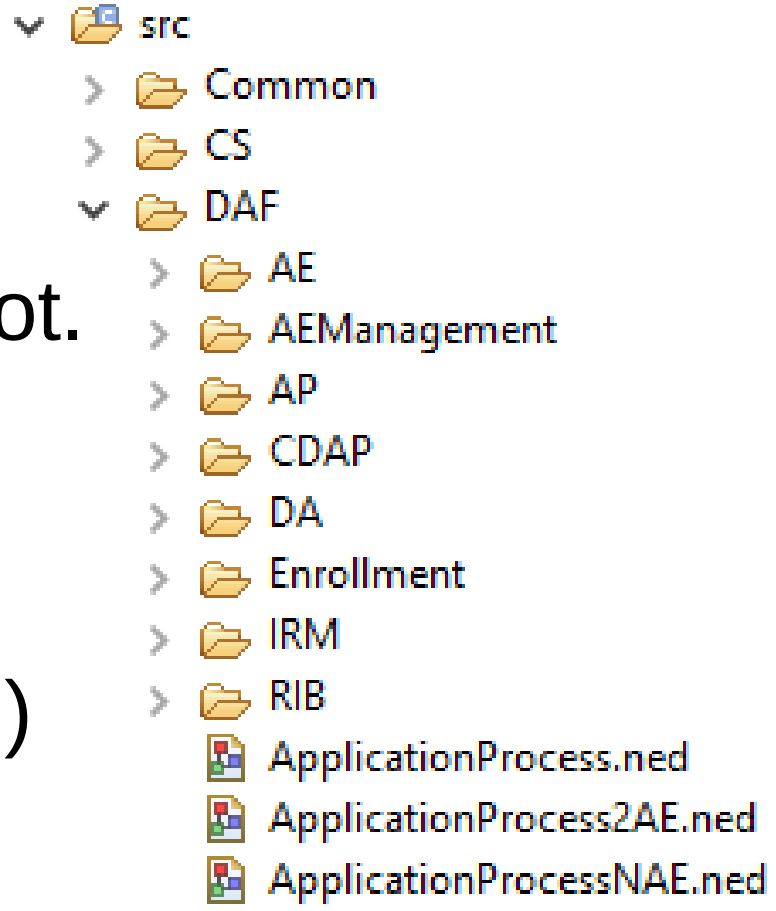






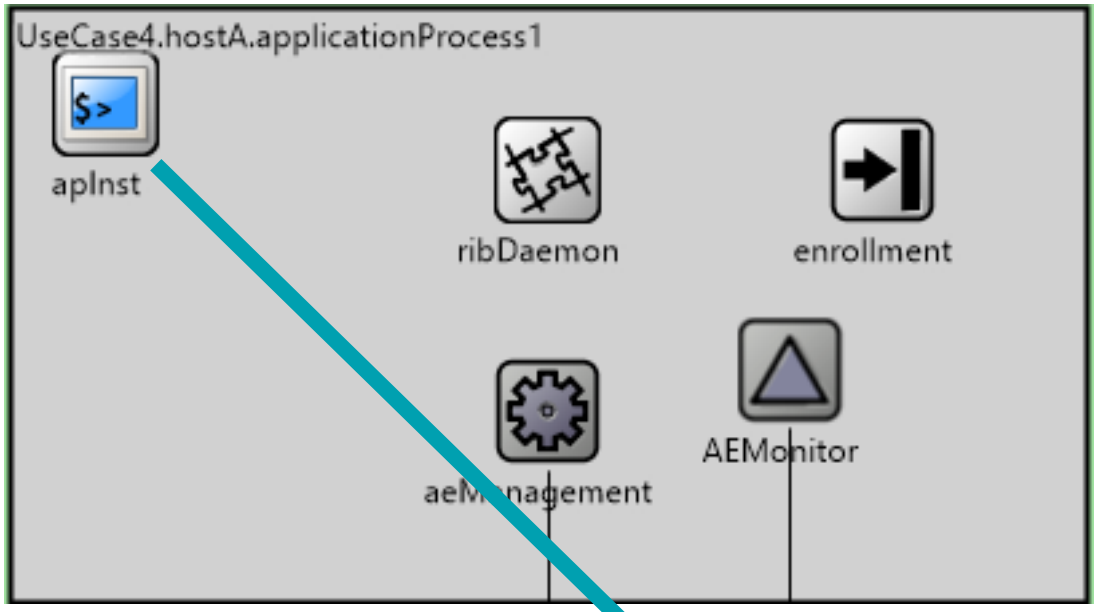
DAF Components

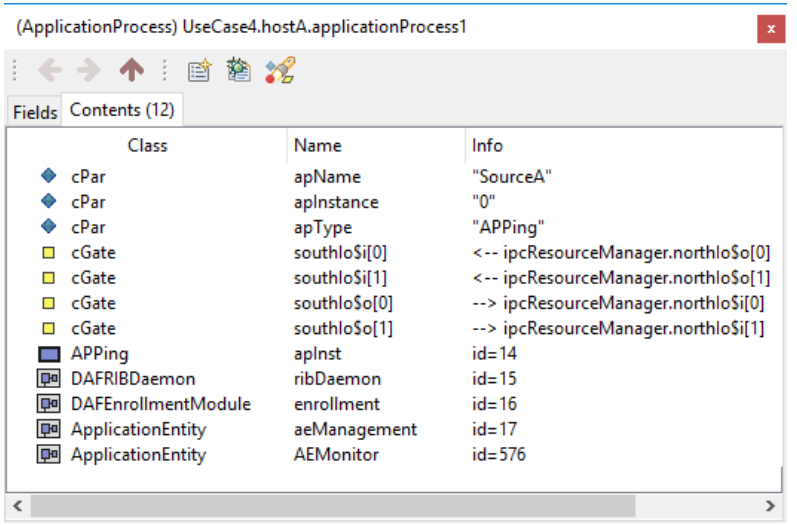
- Application Process (AP)
- Application Entity (AE)
- Common Distribute Apod. Prot.
- DIF Allocator (DA)
- DAF Enrollment
- IPC Resource Manager (IRM)
- DAF RIB Daemon



Application Process

- Contains a single AP instance



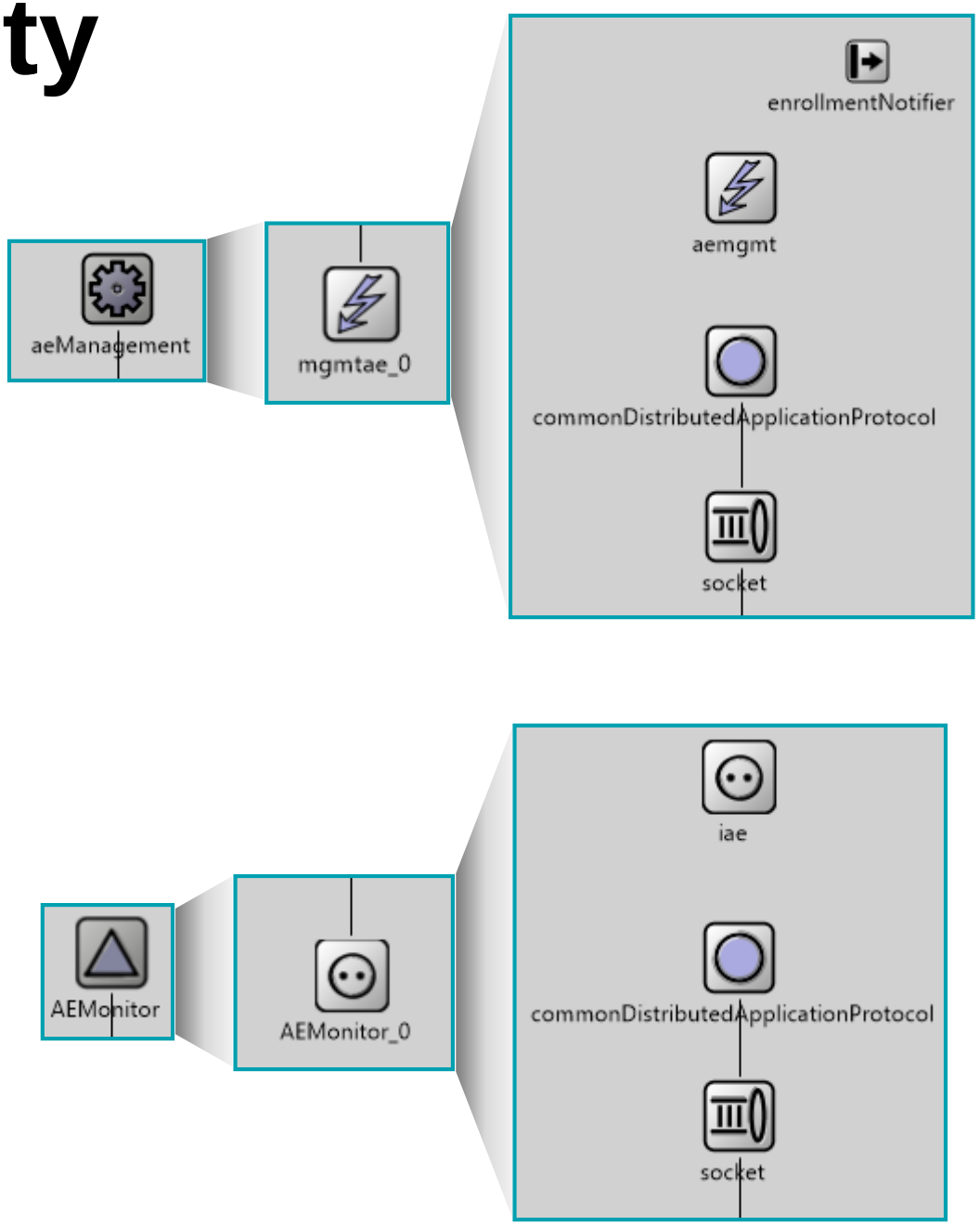


- Application code is written using CDAP API

```
if ( !strcmp(msg->getName(), "start") ) {
    invokeId = getNewInvokeId();
    a_open(invokeId, par("dstApName").stringValue(), "0", "AEMonitor", "-1");
}
else if (!strcmp(msg->getName(), "stop")) {
    a_close(conID);
}
```

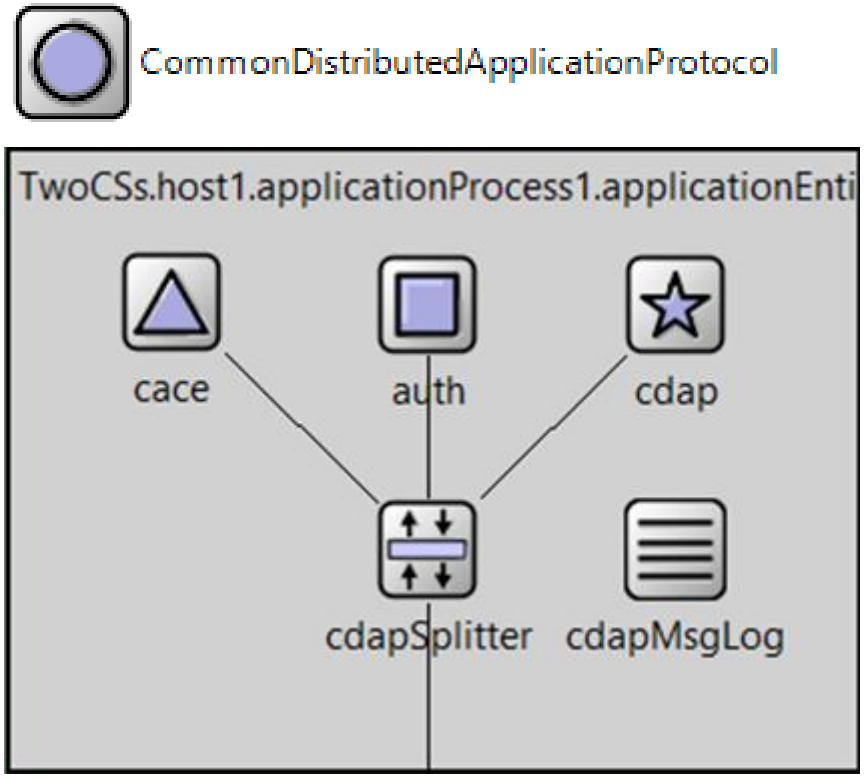
Application Entity

- AP has AE instances
  - suffix _***instanceID***
  - AEManagementModule instance for exchange of management CDAP messages
  - Another AEs delivering IPC for AP instance



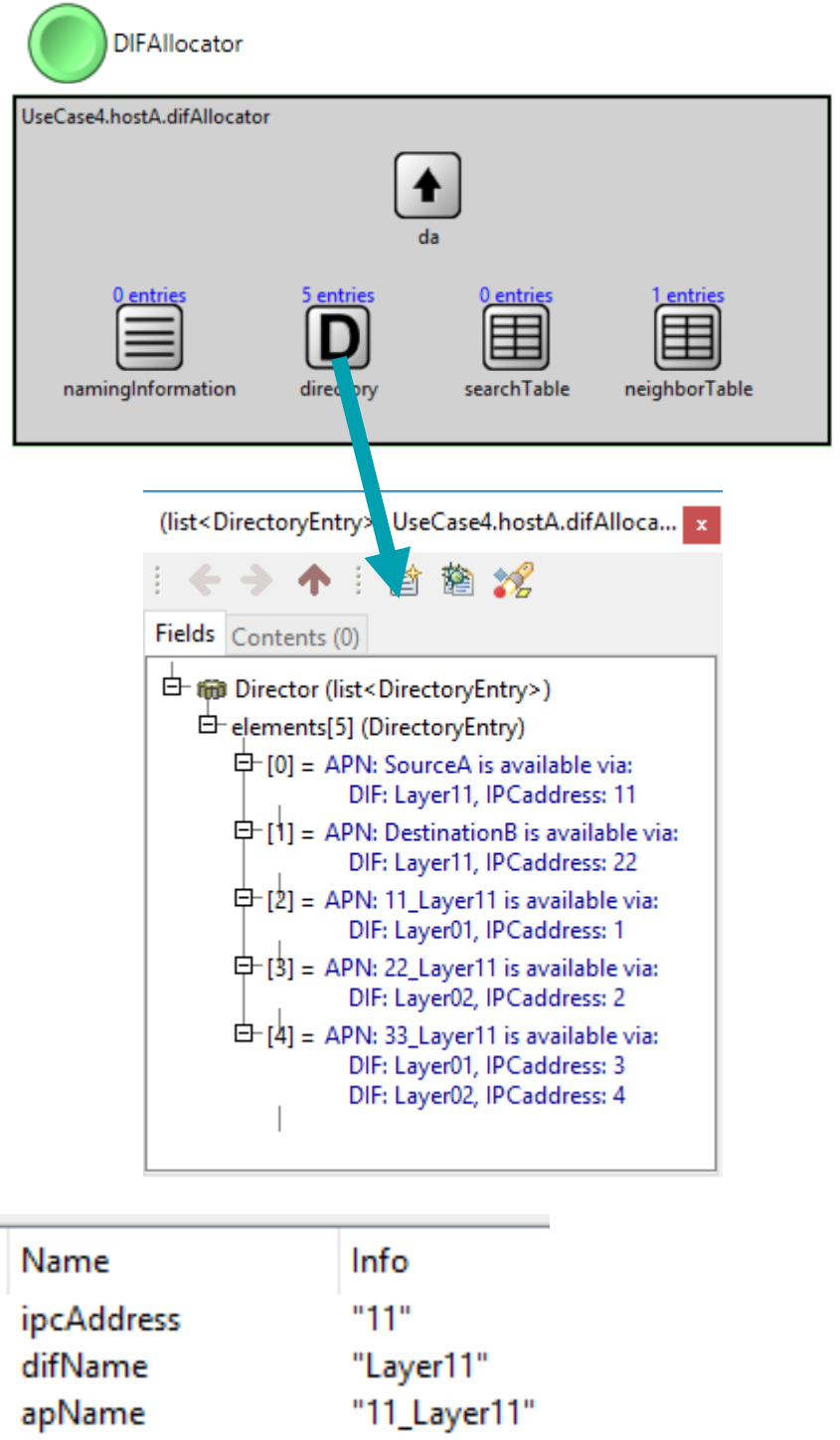
Common Distributed App Protocol

- Simulation module used by AE and RIBd
- **CDAP**
  - Sends/Receives CDAP messages
- **CDAPSplitter**
  - Delegates CDAP message to appropriate module
- **CDAPMsgLog**
  - Statistic collector
- **CACE + AUTH**
  - Used by Enrollment or during authentication phase



DIF Allocator

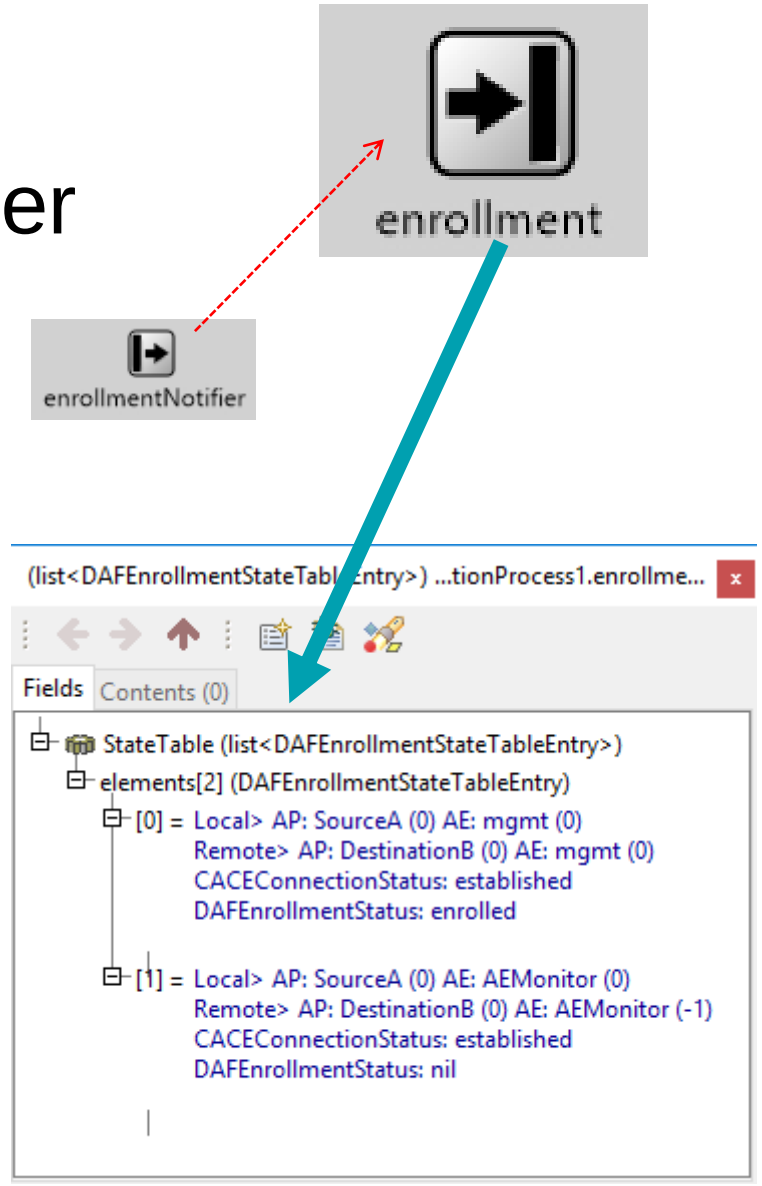
- *Knows how AP/IPCP is reachable via which IPCP*
- Configured externally
- „Unique“ naming scheme
  - whatever string allowing different interpretations
  - IPCP name is concatenation of unambiguous address and DIF name



DAF Enrollment

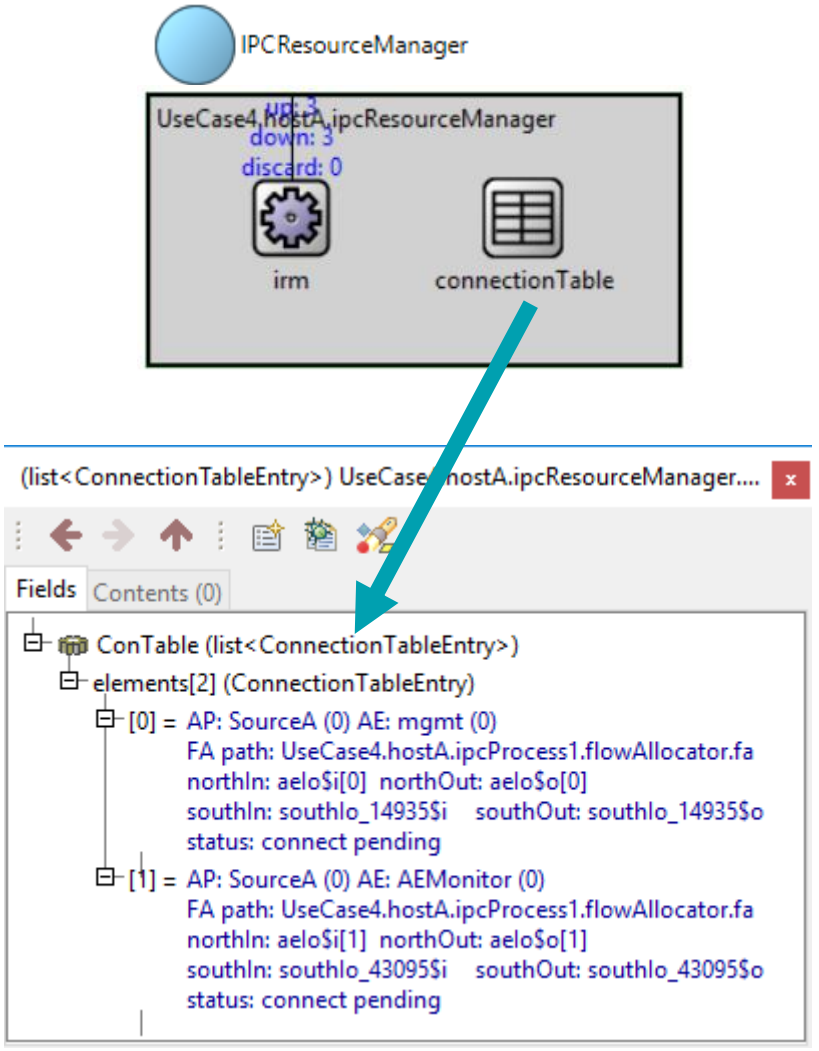
- Notified by Enrollment Notifier within AE

- Maintains state about AE's enrollment



IPC Resource Manager

- Interconnects APs with IPCs
- Passes messages from applications to DIFs



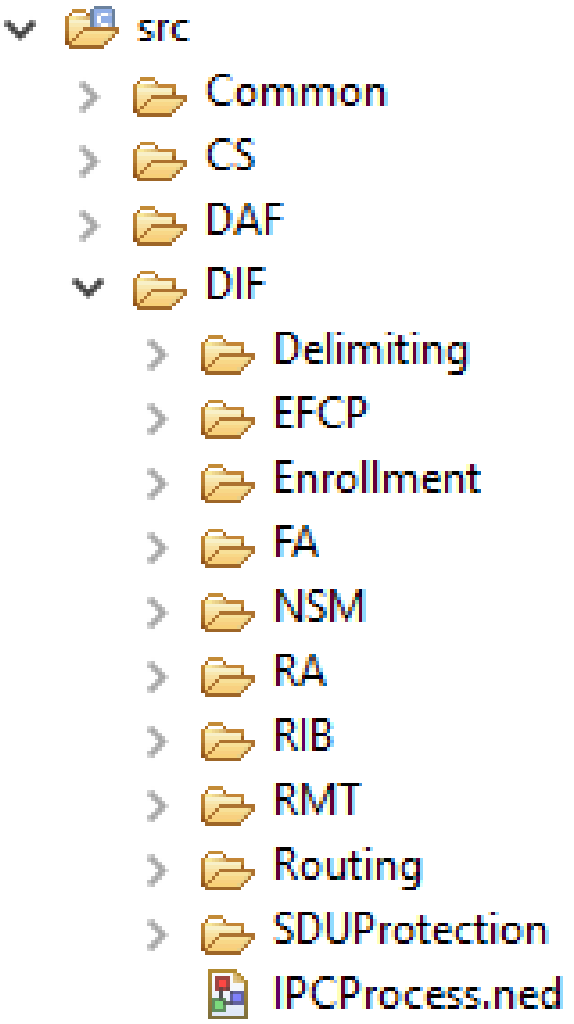
DAF RIB Daemon

- Shared database for AP instance functionality

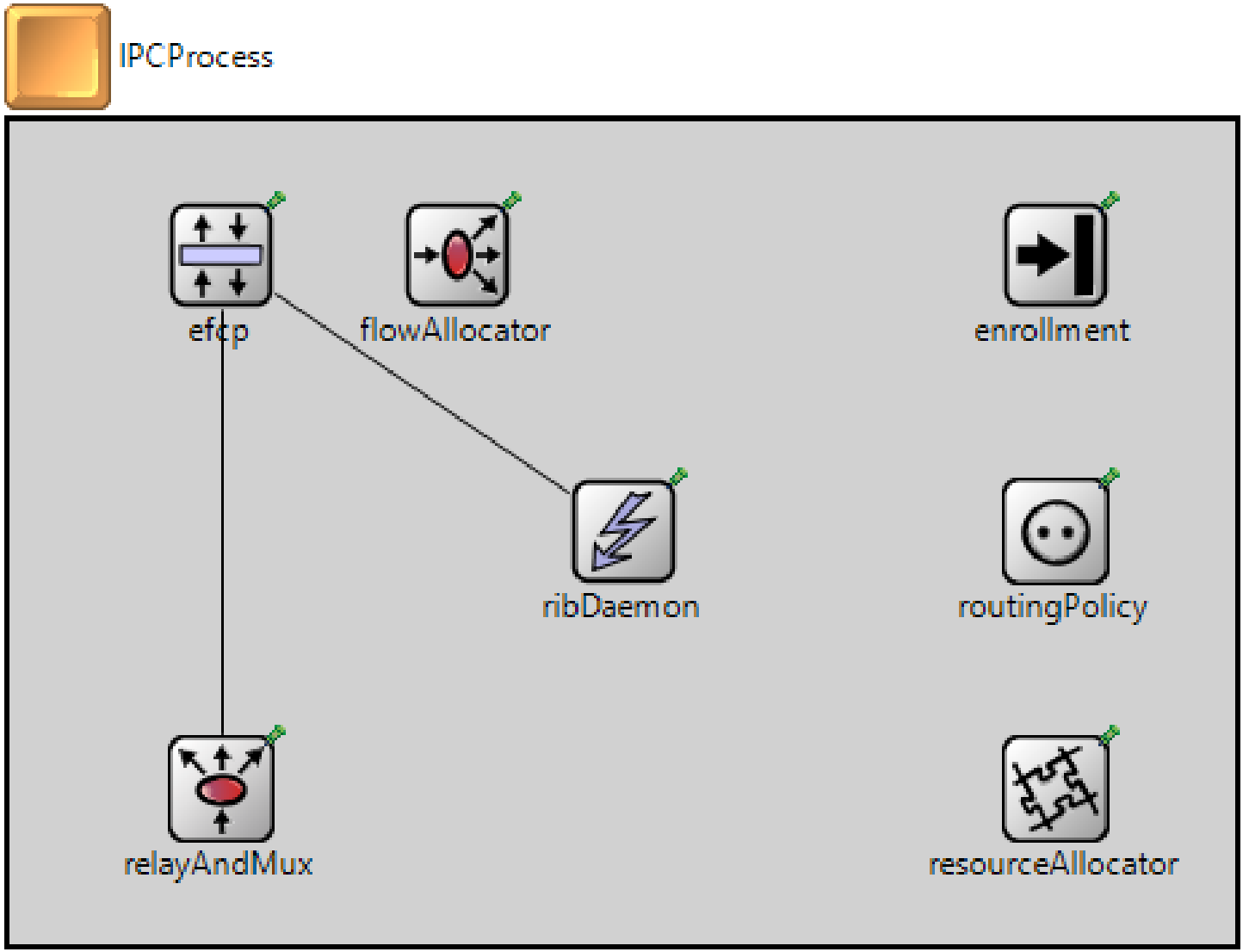
- Its functionality is not bound to other concrete components comparing to IPCP's RIBd

DIF Components

- Flow Allocator (FA)
- Error and Flow Control Prot.
- Delimiting
- Enrollment
- RIB Daemon
- SDU Protection
- Relaying and Multiplexing Task
- Resource Allocator (RA)
- Routing

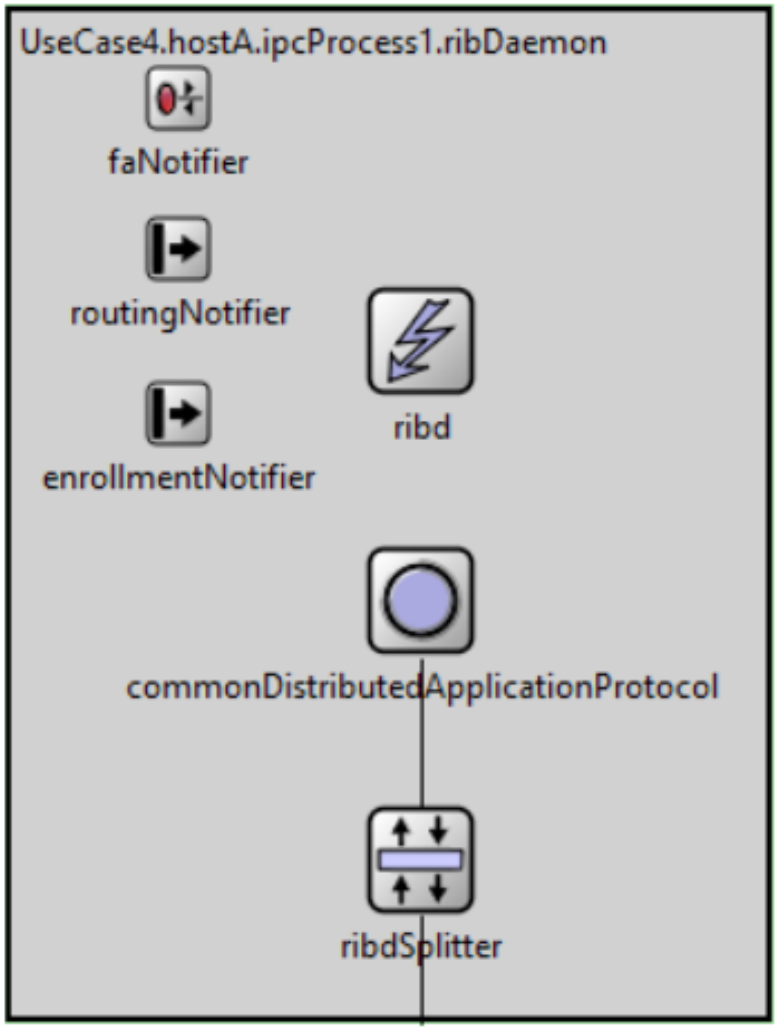


IPC Process


IPCProcess
efcp
flowAllocator
enrollment
ribDaemon
routingPolicy
relayAndMux
resourceAllocator

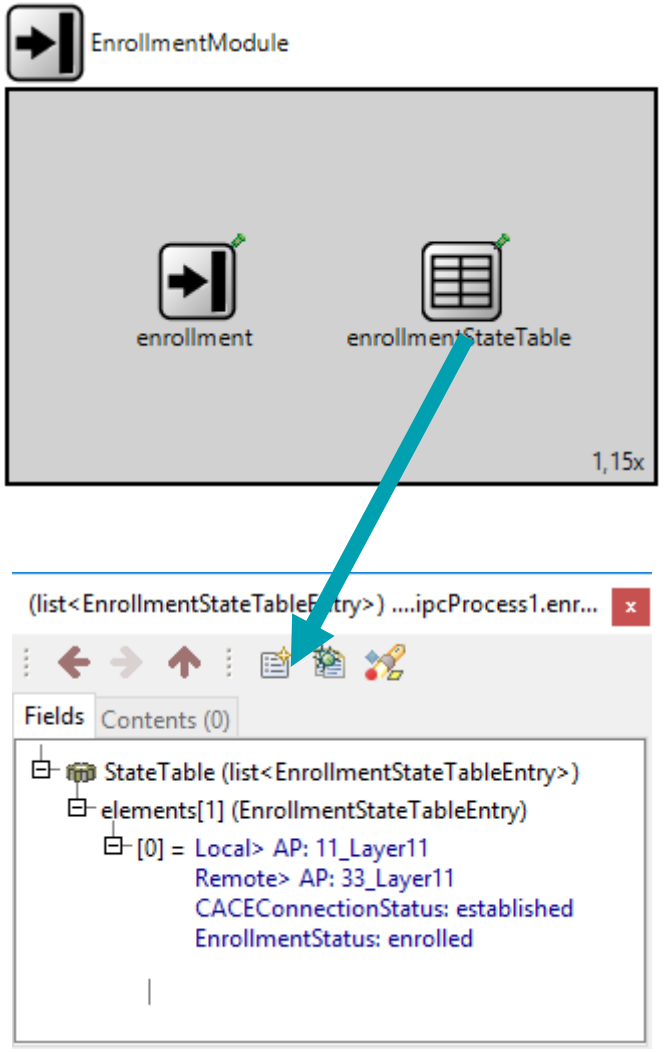
RIB Daemon

- Generates / Processes IPCP management messages
- The main IPCP's AE
- **RIBd**
  - Core functionality
- **CDAP**
  - Socket-like message sender/receiver
- **Notifiers**
  - FA, Routing, Enrollment hooks to RIBd
- **RIBdSplitter**
  - Passes CDAP msgs to/from appropriate EFCPI
  - Placeholder for socket behavior



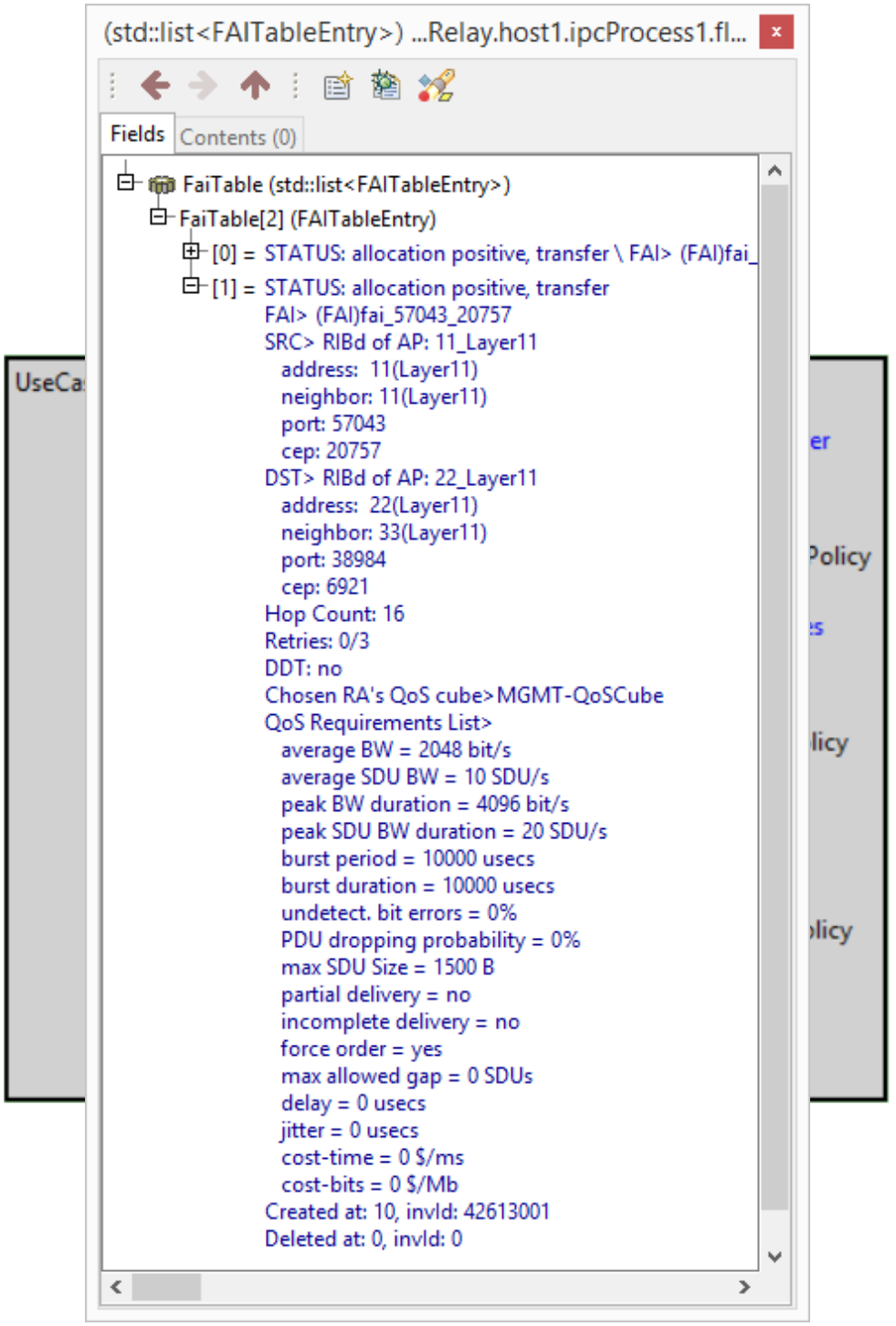
DIF Enrollment

- Maintains enrollment status for a given DIF
- Used when allocating connections between IPCPs



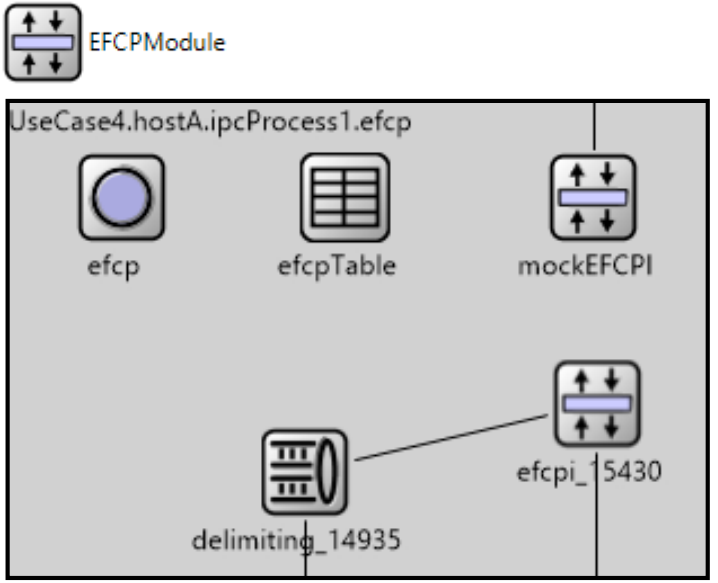
Flow Allocator

- Manages flow lifecycle
- **FA**
  - Core functionality
- **FAI_*portId*_*cepId***
  - Instance
- **NFlowTable**
  - Information about all (N)-DIF connections
- **NewFlowRequestPolicy**
  - Score or Min compare
- **AllocateRetryPolicy**
  - Upon treshold reach
- **QoSComparerPolicy**
  - For multi QoSCube routing purposes



Error and Flow Control Protocol

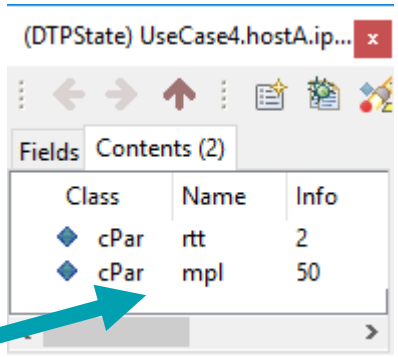
- **EFCP**
  - Manages EFCP instances
- **EFCPTable**
  - Table of known EFCPIs
- **Delimiting_*portId***
  - Creates SDUs from incoming PDUs
- **EFCPI_*cepId***
  - Provides DTP and DTCP services
- **MockEFCPI**
  - Provides unreliable communication for IPCP management messages
  - Simple en/decapsulater between SDUs and PDUs

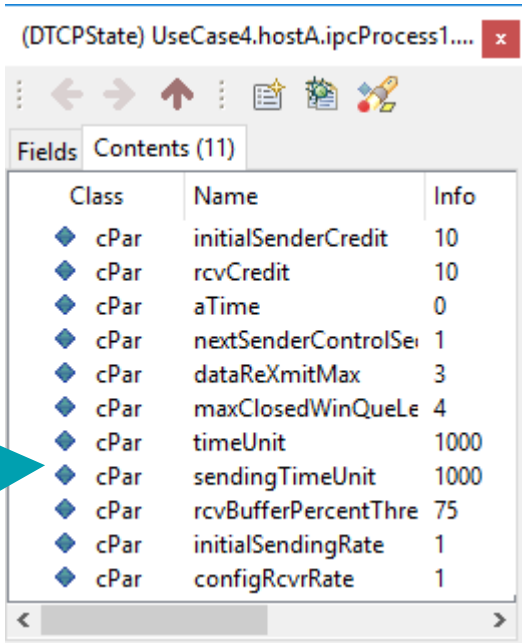


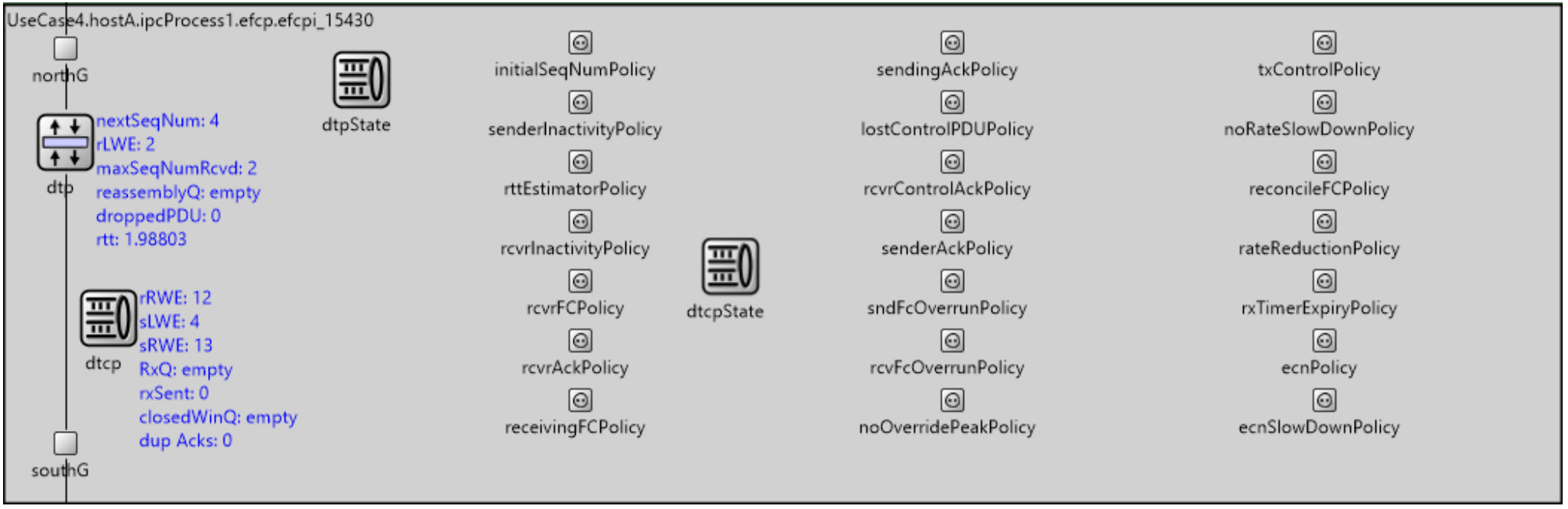
EFCP Instance

- **DTP**
  - Actual Data Transfer
- **DTCP**
  - Handles Flow Control and ReXmission

- **DTPState**
  - Holds all DTP related variables
- **DTCPState**
  - Holds all DTCP related variables



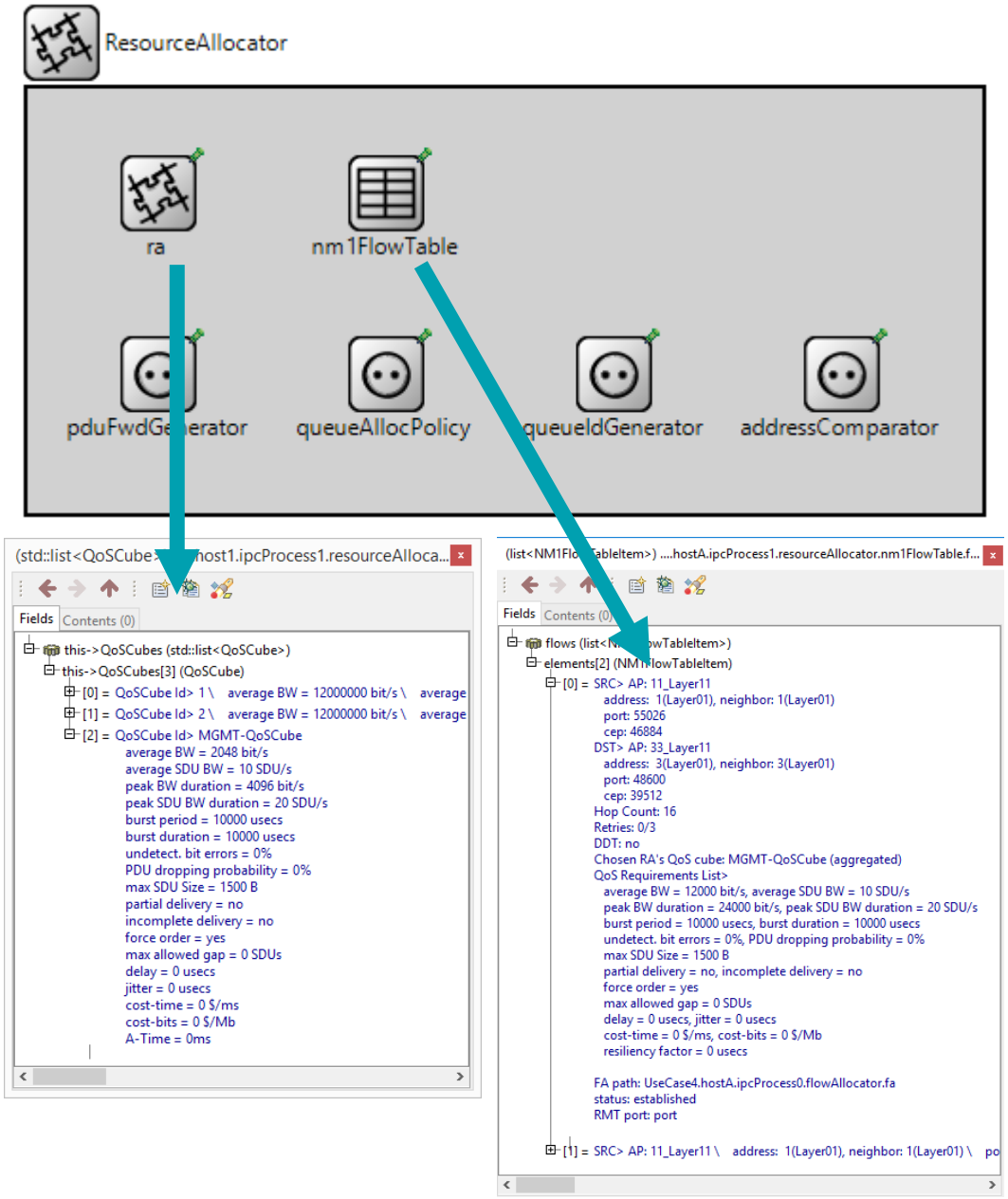




- **EFCP policies**
  - Triggered during various DTP states

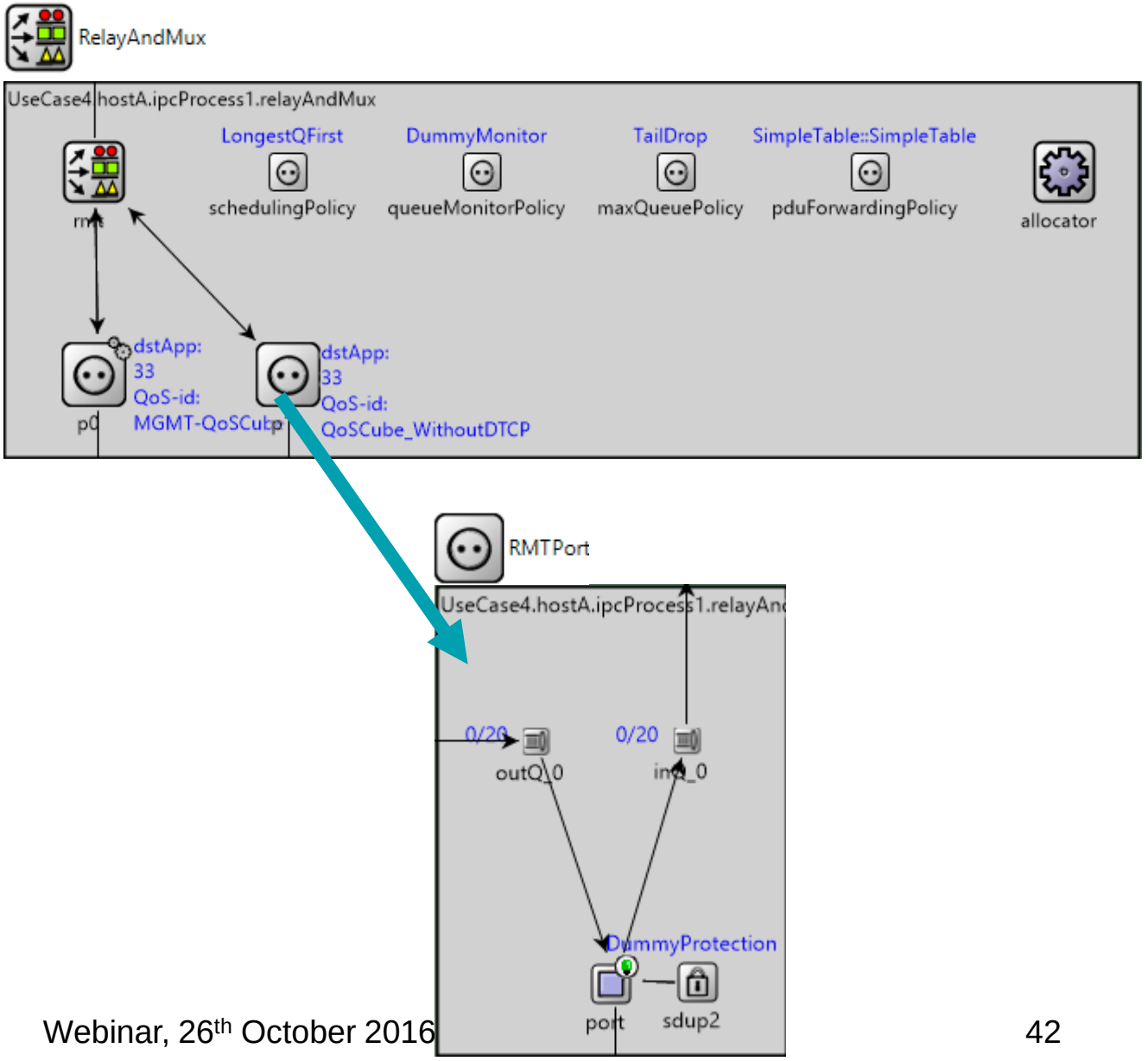
Resource Allocator

- Provides access to (N-1)-DIFs and their resources
- **RA**
  - Core functionality
  - Manages IPCP's QoSCubes
- **NM1FlowTable**
  - Information about current (N-1)-flows
- **PDUFwdGenerator**
  - Forwarding information management
- **QueueAllocPolicy**
  - How and when should RMT queues be allocated?
- **QueueIdGenerator**
  - In which RMT queue should a PDU be stored?
- **AddressComparator**
  - Syntax and comparison of addresses



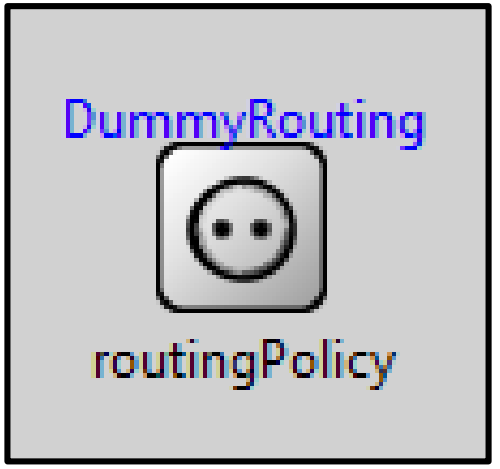
Relaying and Multiplexing Task

- Relays incoming/outgoing PDU to proper destination (either an EFCP instance or an (N-1)-flow)
- **RMT**
  - The core PDU forwarder
- **SchedulingPolicy**
  - When a PDU needs to be sent/received, which queue should it be taken from?
- **QueueMonitorPolicy**
  - Keeping information about port/queue states
- **MaxQueuePolicy**
  - What should happen to a queue when it overflows?
- **PDUForwardingPolicy**
  - Where should be PDU relayed based on a given header?



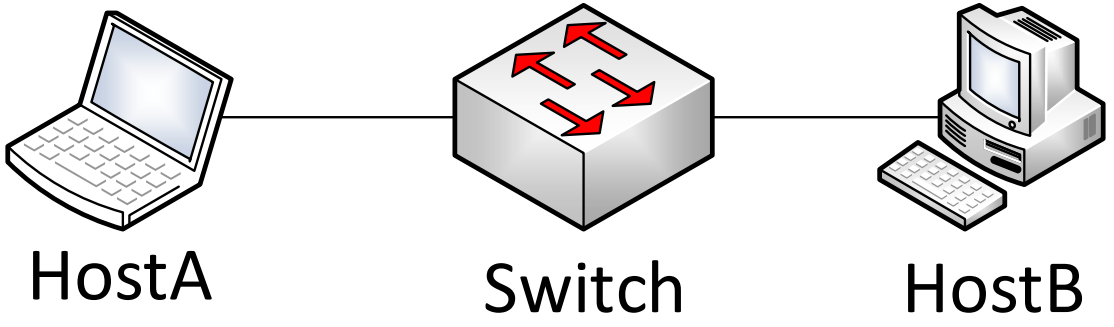
Routing

- The policy computing optimal paths to other destinations by given metrics
- Usually some sort of routing algorithm exchanging information with other members of a DIF

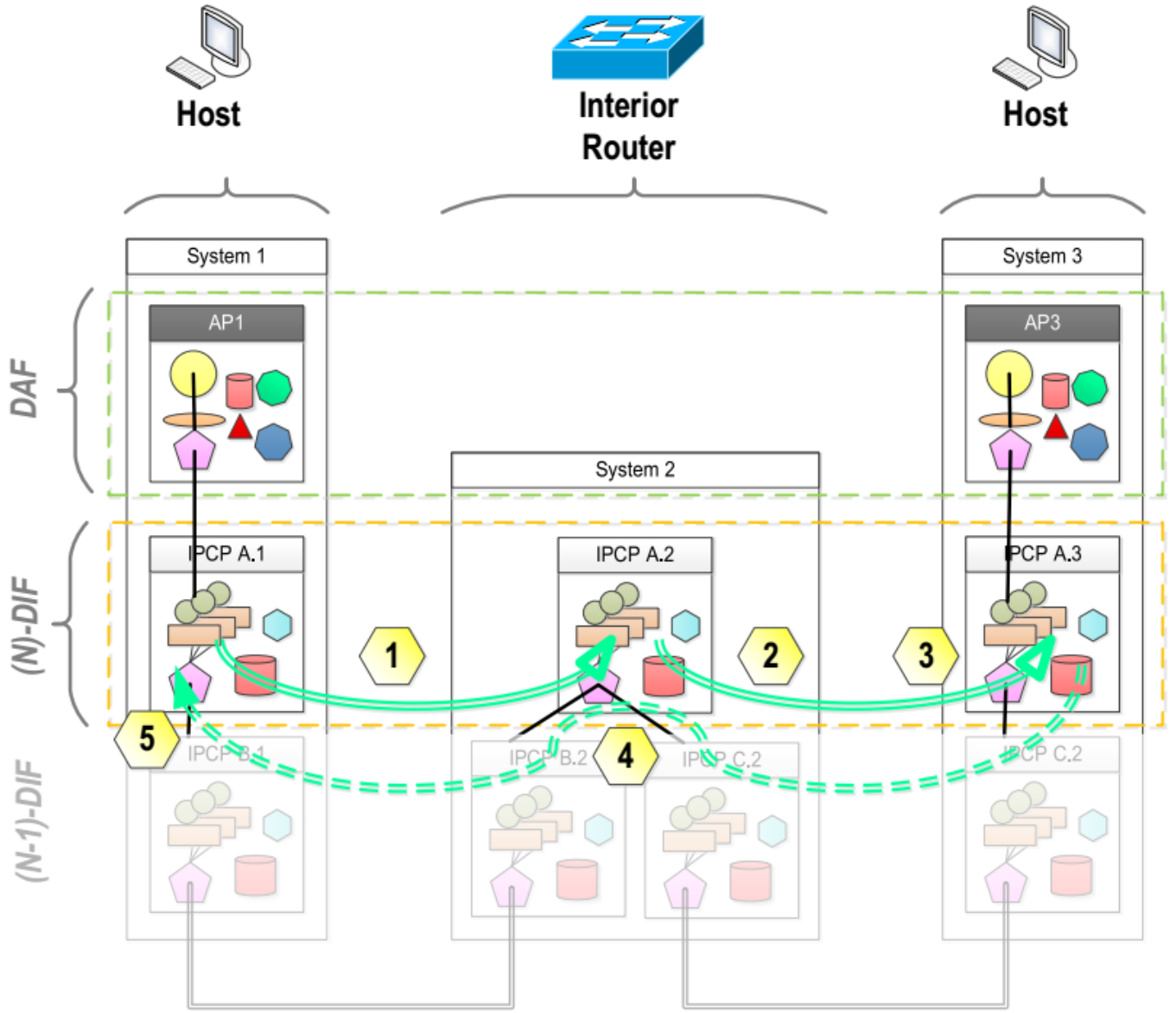


Interactive Demo

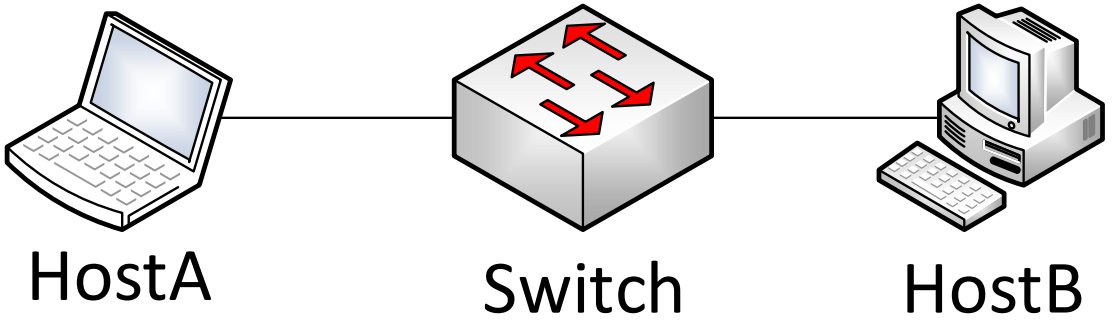
How IPC works between two hosts interconnected to a common node?



Topology


Host
Interior Router
Host
System 1
System 2
System 3
AP1
AP3
DAF
(N)-DIF
(N-1)-DIF
IPCP A.1
IPCP A.2
IPCP A.3
IPCP B.1
IPCP B.2
IPCP C.2
IPCP C.2
1
2
3
4
5

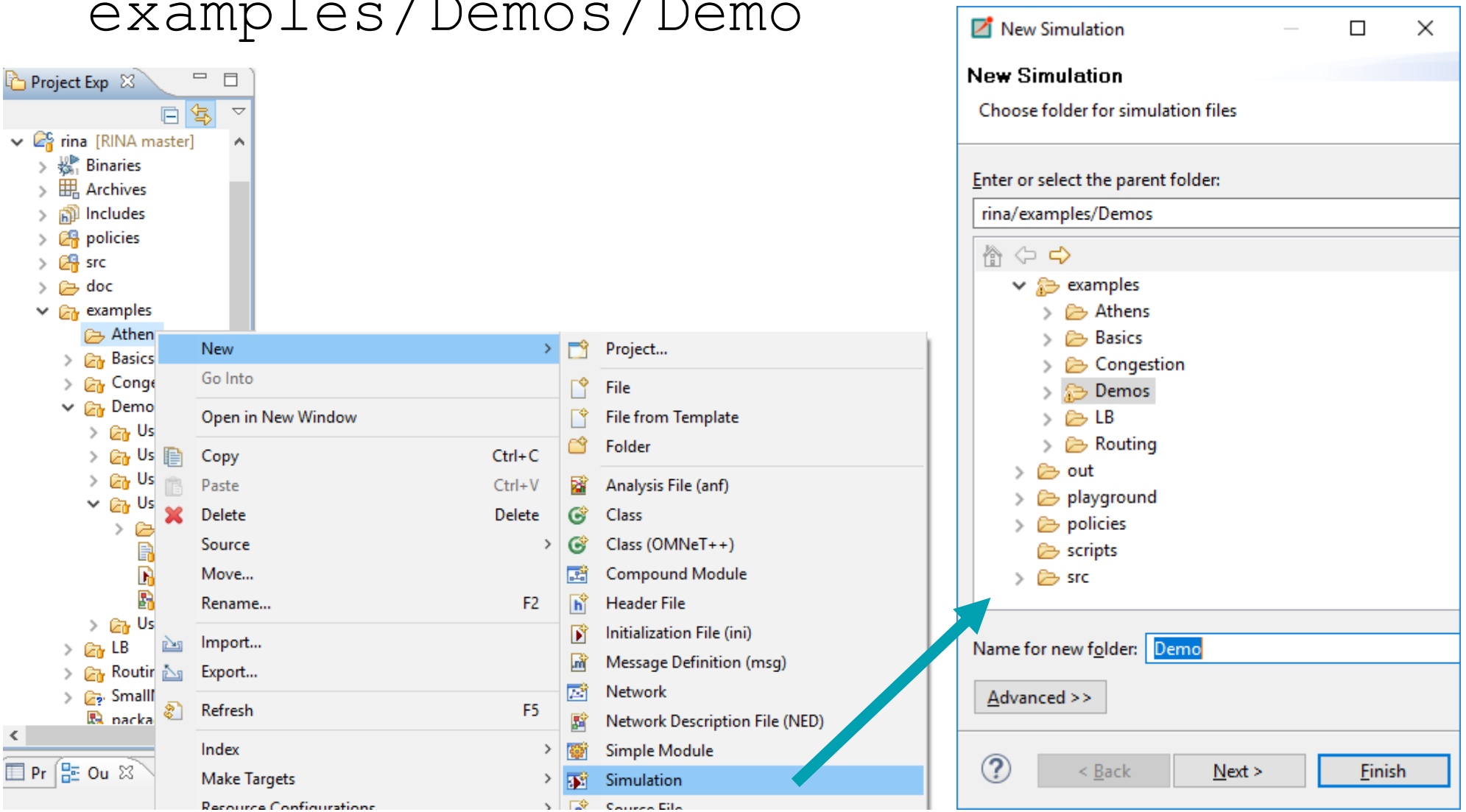
Cookbook



- Topology
  - 2× host with single AP
  - 1× interior router
  - 2× datarate channel between
- Task
  1) Setup network
  2) Schedule simulation
  3) Run
- Goal
  - To observe IPC between two hosts interconnected by a interior router

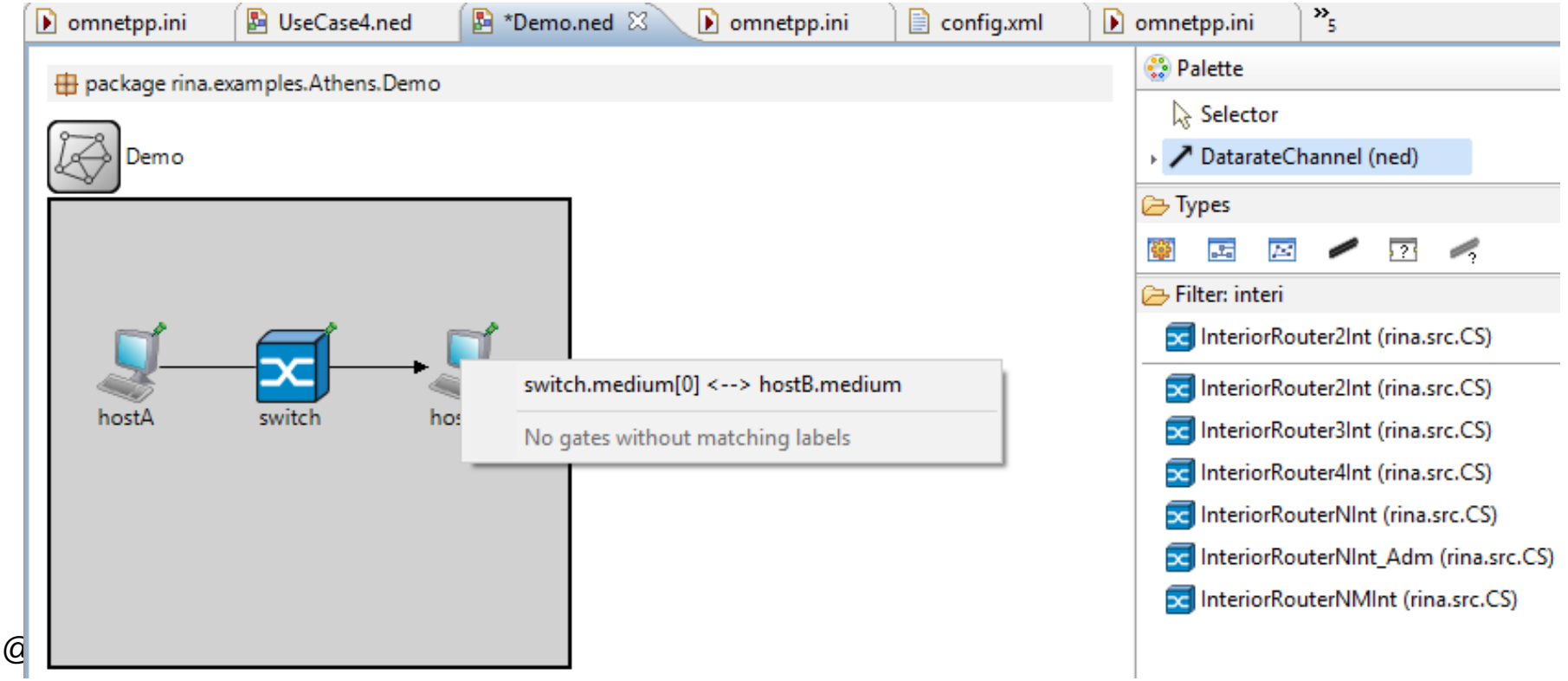
1) Setup network

- Create new simulation in folder `examples/Demos/Demo`



1) Setup network

- Open `Demo.ned` and add two `Host1AP` onto canvas and one `InteriorRouter2Int`
  - Rename them with F6
  - Connect them with `DatarateChannel`



1) Setup network

- Change `DatarateChannel` properties
  - Setup `delay`, `ber`, `datarate`

omnetpp.ini | UseCase4.ned | Demo.ned | omnetpp.ini | config.xml | omnetpp.ini | »5

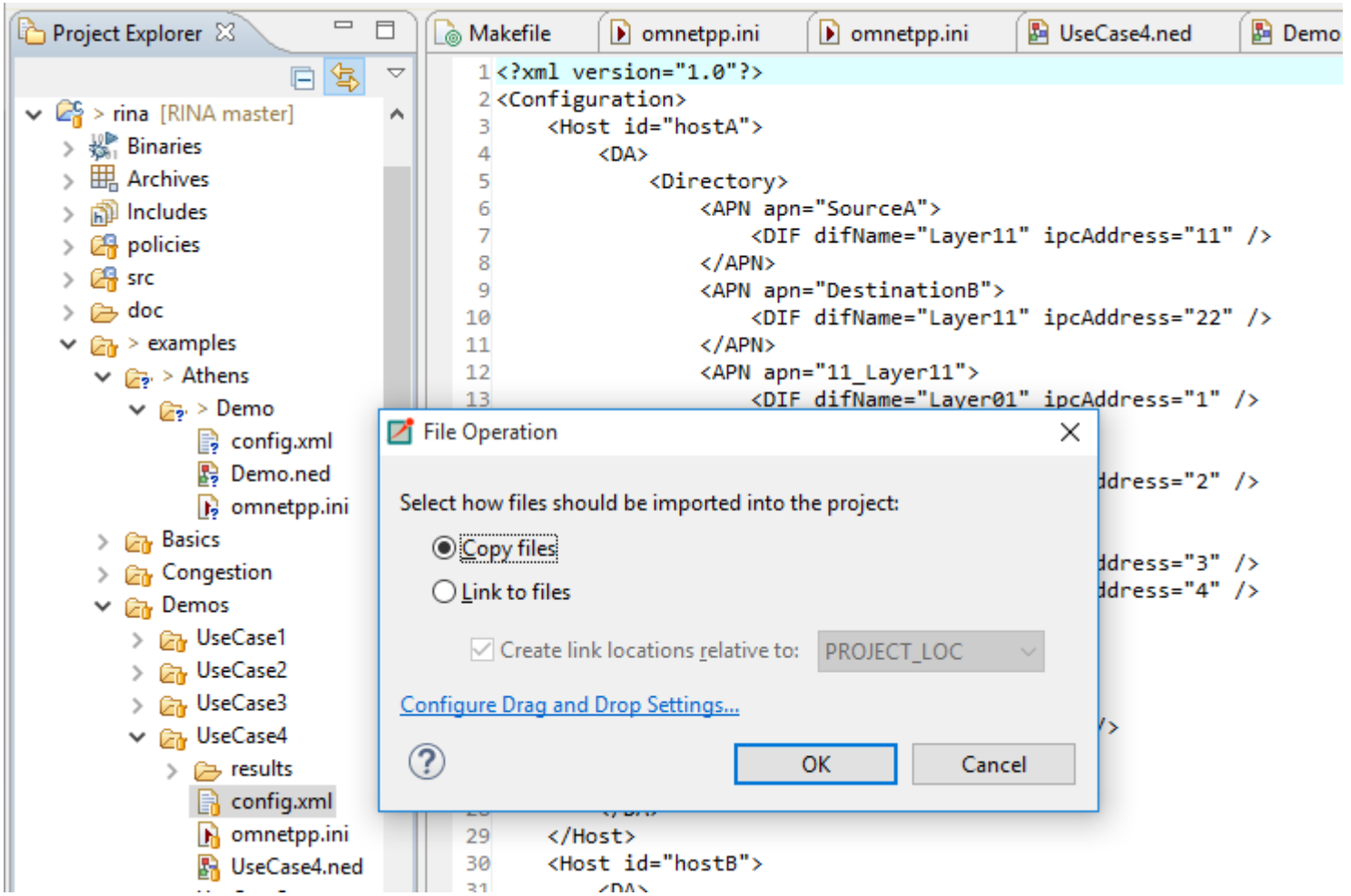
```
1
2  package rina.examples.Athens.Demo;
3
4  import ned.DatarateChannel;
5  import rina.src.CS.Host1AP;
6  import rina.src.CS.InteriorRouter2Int;
7
8  network Demo
9  {
10     @display("bgb=282,253");
11     submodules:
12         hostA: Host1AP {
13             @display("p=42,90");
14         }
15         hostB: Host1AP {
16             @display("p=223,90");
17         }
18         switch: InteriorRouter2Int {
19             @display("p=132,90");
20         }
21     connections:
22         hostA.medium <--> DatarateChannel {  datarate = 100Mbps; delay = 100us; ber = 0; } <--> switch.medium[0];
23         switch.medium[1] <--> DatarateChannel {  datarate = 100Mbps; delay = 100us; ber = 0; } <--> hostB.medium;
24 }
25
```

2) Schedule simulation

- `config.xml`
  - DIF Allocator settings
  - Schedule enrollment events
  - Available QoS Cubes
- `omnetpp.ini`
  - Assign addresses
  - Bind `config.xml`
  - Schedule AEMyPing

2) Schedule simulation: config.xml

- Copy `config.xml`
  from `examples/Demos/UseCase4/config.xml`



2) Schedule simulation: config.xml

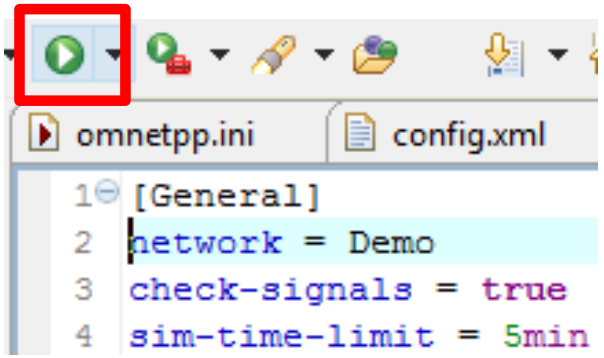
```xml
<?xml version="1.0"?>
<Configuration>
    <Host id="hostA">
        <DA>
            <Directory>
                <APN apn="SourceA">
                    <DIF difName="Layer11" ipcAddress="11" />
                </APN>
                <APN apn="DestinationB">
                    <DIF difName="Layer11" ipcAddress="22" />
                </APN>
                <APN apn="11_Layer11">
                    <DIF difName="Layer01" ipcAddress="1" />
                </APN>
                <APN apn="22_Layer11">
                    <DIF difName="Layer02" ipcAddress="2" />
                </APN>
                <APN apn="33_Layer11">
                    <DIF difName="Layer01" ipcAddress="3" />
                    <DIF difName="Layer02" ipcAddress="4" />
                </APN>
            </Directory>
            <NeighborTable>
                <APN apn="22_Layer11">
                    <Neighbor apn="33_Layer11" />
                </APN>
            </NeighborTable>
        </DA>
    </Host>
    <Host id="hostB">
        <DA>
            <Directory>
                <APN apn="SourceA">
                    <DIF difName="Layer11" ipcAddress="11" />
                </APN>
                <APN apn="DestinationB">
                    <DIF difName="Layer11" ipcAddress="22" />
                </APN>
                <APN apn="11_Layer11">
                    <DIF difName="Layer01" ipcAddress="1" />
                </APN>
                <APN apn="22_Layer11">
                    <DIF difName="Layer02" ipcAddress="2" />
                </APN>
                <APN apn="33_Layer11">
                    <DIF difName="Layer01" ipcAddress="3" />
                    <DIF difName="Layer02" ipcAddress="4" />
                </APN>
            </Directory>
            <NeighborTable>
                <APN apn="11_Layer11">
                    <Neighbor apn="33_Layer11" />
                </APN>
            </NeighborTable>
        </DA>
        <Enrollment>
            <Preenrollment>
                <SimTime t="5">
                    <Connect src="22_Layer11" dst="33_Layer11" />
                </SimTime>
            </Preenrollment>
        </Enrollment>
    </Host>
    <Router id="switch">
        <DA>
            <Directory>
                <APN apn="SourceA">
                    <DIF difName="Layer11" ipcAddress="11" />
                </APN>
                <APN apn="DestinationB">
                    <DIF difName="Layer11" ipcAddress="22" />
                </APN>
                <APN apn="11_Layer11">
                    <DIF difName="Layer01" ipcAddress="1" />
                </APN>
                <APN apn="22_Layer11">
                    <DIF difName="Layer02" ipcAddress="2" />
                </APN>
                <APN apn="33_Layer11">
                    <DIF difName="Layer01" ipcAddress="3" />
                    <DIF difName="Layer02" ipcAddress="4" />
                </APN>
            </Directory>
        </DA>
    </Router>
    <QoSCubesSet>
        <QoSCube id="QoSCube_WithoutDTCP">
            <AverageBandwidth>10000000</AverageBandwidth>
            <AverageSDUBandwidth>1000</AverageSDUBandwidth>
            <PeakBandwidthDuration>20000000</PeakBandwidthDuration>
            <PeakSDUBandwidthDuration>2000</PeakSDUBandwidthDuration>
            <BurstPeriod>10000000</BurstPeriod>
            <BurstDuration>1000000</BurstDuration>
            <UndetectedBitError>0.01</UndetectedBitError>
            <PDUDroppingProbability>0</PDUDroppingProbability>
            <MaxSDUSize>1500</MaxSDUSize>
            <PartialDelivery>0</PartialDelivery>
            <IncompleteDelivery>0</IncompleteDelivery>
            <ForceOrder>0</ForceOrder>
            <MaxAllowableGap>0</MaxAllowableGap>
            <Delay>1000000</Delay>
            <Jitter>500000</Jitter>
            <CostTime>0</CostTime>
            <CostBits>0</CostBits>
        </QoSCube>
        <QoSCube id="QoSCube_WithDTCP">
            <AverageBandwidth>10000000</AverageBandwidth>
            <AverageSDUBandwidth>1000</AverageSDUBandwidth>
            <PeakBandwidthDuration>20000000</PeakBandwidthDuration>
            <PeakSDUBandwidthDuration>2000</PeakSDUBandwidthDuration>
            <BurstPeriod>10000000</BurstPeriod>
            <BurstDuration>1000000</BurstDuration>
            <UndetectedBitError>0.01</UndetectedBitError>
            <PDUDroppingProbability>0</PDUDroppingProbability>
            <MaxSDUSize>1500</MaxSDUSize>
            <PartialDelivery>0</PartialDelivery>
            <IncompleteDelivery>0</IncompleteDelivery>
            <ForceOrder>1</ForceOrder>
            <MaxAllowableGap>0</MaxAllowableGap>
            <Delay>1000000</Delay>
            <Jitter>500000</Jitter>
            <CostTime>0</CostTime>
            <CostBits>0</CostBits>
        </QoSCube>
    </QoSCubesSet>
</Configuration>
```

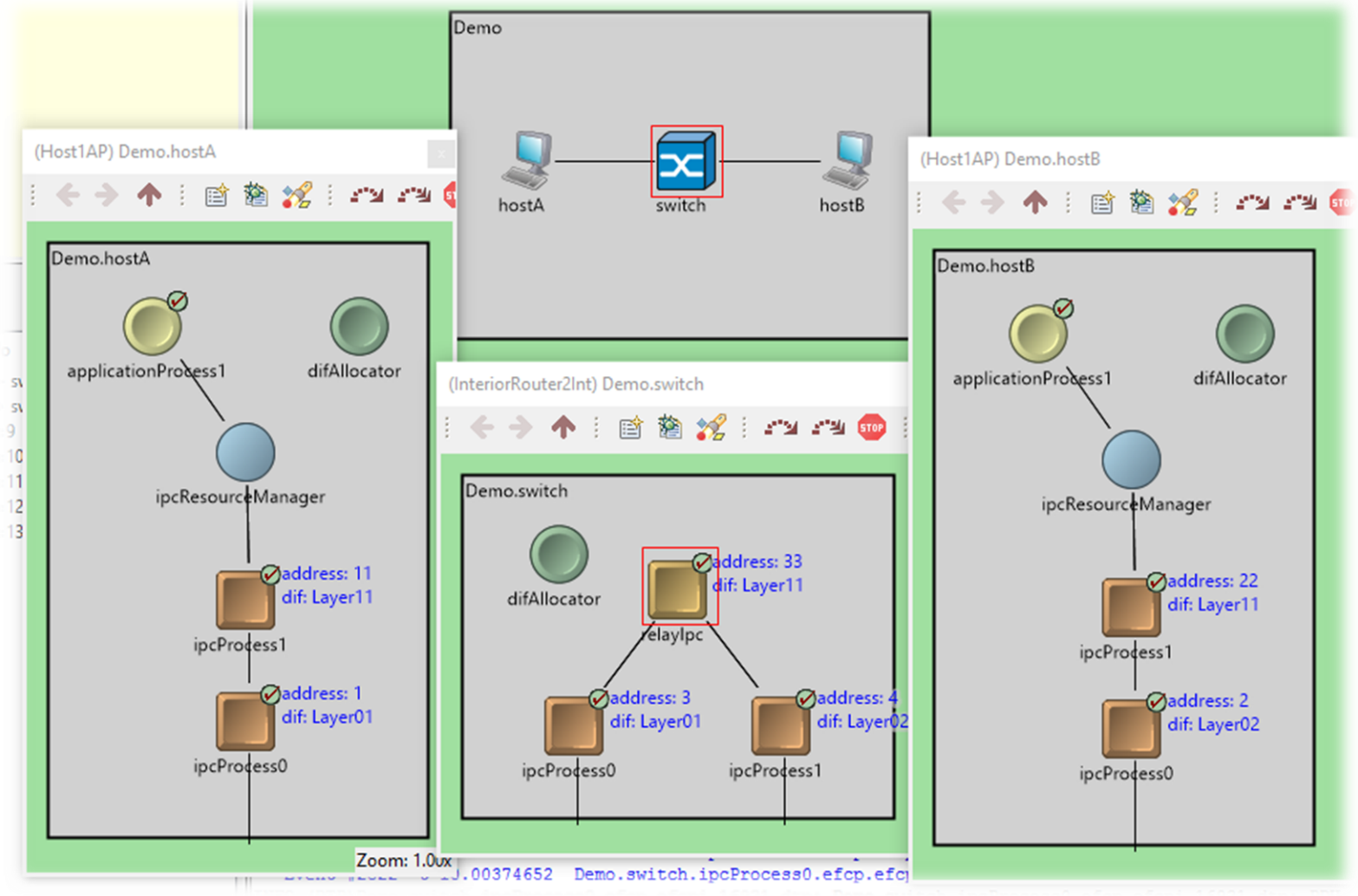
2) Schedule simulation: omnetpp.ini

```
[General]
network = Demo
check-signals = true
sim-time-limit = 5min
debug-on-errors = true
#Application setup
**.hostA.applicationProcess1.apName = "SourceA"
**.hostB.applicationProcess1.apName = "DestinationB"
**.apType = "APPing"
#Static addressing
#  Bottom DIF HostA<->Switch
**.hostA.ipcProcess0.ipcAddress = "1"
**.hostA.ipcProcess0.difName = "Layer01"
**.switch.ipcProcess0.ipcAddress = "3"
**.switch.ipcProcess0.difName = "Layer01"
#  Bottom DIF HostB<->Switch
**.hostB.ipcProcess0.ipcAddress = "2"
**.hostB.ipcProcess0.difName = "Layer02"
**.switch.ipcProcess1.ipcAddress = "4"
**.switch.ipcProcess1.difName = "Layer02"
#  Top DIF HostA<->Switch<->HostB
**.hostA.ipcProcess1.ipcAddress = "11"
**.hostB.ipcProcess1.ipcAddress = "22"
**.host*.ipcProcess1.difName = "Layer11"
**.switch.relayIpc.ipcAddress = "33"
**.switch.relayIpc.difName = "Layer11"
#DIF Allocator settings
**.hostA.difAllocator.configData = xmldoc("config.xml", "Configuration/Host[@id='hostA']/DA")
**.hostB.difAllocator.configData = xmldoc("config.xml", "Configuration/Host[@id='hostB']/DA")
**.switch.difAllocator.configData = xmldoc("config.xml", "Configuration/Router[@id='switch']/DA")
#Enrollment settings
**.switch.**.enrollment.isSelfEnrolled = true
**.hostB.ipcProcess1.enrollment.configData = xmldoc("config.xml", "Configuration/Host[@id='hostB']/Enrollment")
#QoS Cube sets
**.ra.qoscubesData = xmldoc("config.xml", "Configuration/QoSCubesSet")

[Config Ping]
fingerprint = "dff8-6343"
#PingApp setup
**.hostA.applicationProcess1.apInst.dstApName = "DestinationB"
**.hostA.applicationProcess1.apInst.startAt = 10s
**.hostA.applicationProcess1.apInst.stopAt = 20s
**.hostA.applicationProcess1.AEMonitor.**.iae.size = 1024B
```

3) Run


omnetpp.ini
config.xml
1 [General]
2 network = Demo
3 check-signals = true
4 sim-time-limit = 5min


Demo
hostA
switch
hostB
(Host1AP) Demo.hostA
Demo.hostA
applicationProcess1
difAllocator
ipcResourceManager
address: 11
dif: Layer11
ipcProcess1
address: 1
dif: Layer01
ipcProcess0
Zoom: 1.0x
(InteriorRouter2Int) Demo.switch
Demo.switch
difAllocator
address: 33
dif: Layer11
relayIpc
address: 3
dif: Layer01
address: 4
dif: Layer02
ipcProcess0
ipcProcess1
(Host1AP) Demo.hostB
Demo.hostB
applicationProcess1
difAllocator
ipcResourceManager
address: 22
dif: Layer11
ipcProcess1
address: 2
dif: Layer02
ipcProcess0
Demo.switch.ipcProcess0.efcp.efcp

Notable Events

- t=5
  - hostB enrolls to Layer02 and Layer11
- t=10
  - hostA creats flows for AP communication
- t=10.3
  - SourceA and DestinationB apps exchange ping messages
- t=20
  - hostA deallocates Layer11 flow

5) Conclusion

Final remarks
Usual problems

Summary

- Educational tool
  - A way how to visualize what is happening in the native RINA network
  - Helps to improve learning curve

- Research tool
  - Check the growing list of citations https://rinasim.omnetpp.org/#references

Something Is Not Working

1) Simulation crashes before it starts
   - Serious bug in your NED/C++ code
2) Simulation crashes after initialization
   - Error in static preconfiguration

     → check omnetpp.ini and config.xml
3) Simulation does not do what is supposed to
   - Error in your code logic

     → check the output of the Console

     → try to debug using GDB

Other Topics

- We did not cover
  - *How to play with policies? How to implement own policy?*
  - *How to collect and interpret results?*

- Interested?

You Want to Use RINASim

- Messages like:

my research project is to study architecture with RINA (Recursive Internetworking Architecture). I am willing to work with RINASim simulator, but unfortunately does not work with programming. I intend to work with simulator to collect data and analyze with another framework (INET). The goal of my research is to make a performance study between RINA architectures and TCP/IP using the RINA and INET frameworks.

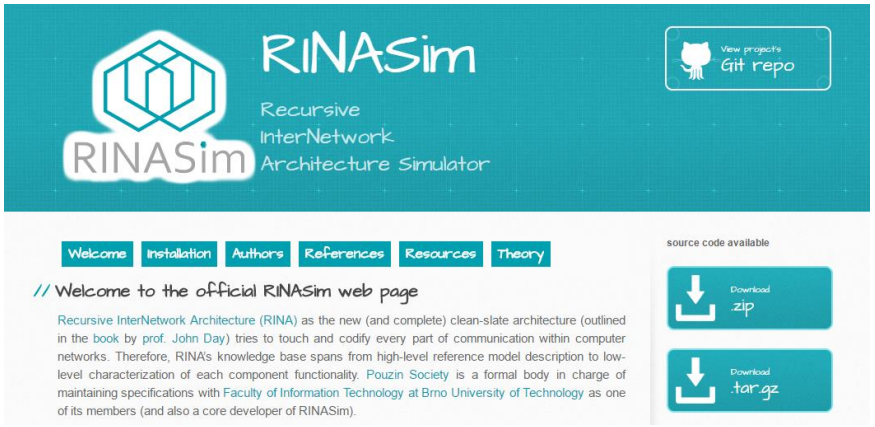
Initially I was thinking of running some tests using wireless models with RINASim and then comparing them to the INET Framework models. Since I am not fluent in OMNeT, using the 4.6 version is an extreme pain because I haven't seen any working examples other than with LTE, which is currently out of my scope. OMNeT 5.0 finally has a semi-understandable wireless tutorial so I was trying to make my life easier.

- Currently, RINASim does not offer any real data-link layer technology

- In order to use RINASim, you need to be programmer whether you like it or not ☺

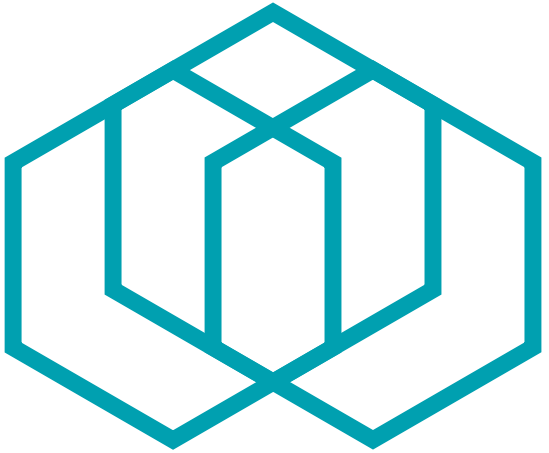
Need Help?

- Check the official webpage
  - Visit https://rinasim.omnetpp.org



- Skype group chat
  - skype:?chat&blob=-bdq6qH_uDXIlbRk_4_XwqZyplfXPl4IzCq4P-S0BrsttjgPR8CNJKV9-Yyn1TYopaYZD2g3bIC_Yv0C
  - https://join.skype.com/B9Tt5aTPd0nC

- Sign to mailing-list rinasim@fit.vutbr.cz
  - Use http://www.fit.vutbr.cz/mailman/listinfo/rinasim



RINASim

Thank you for your attention!