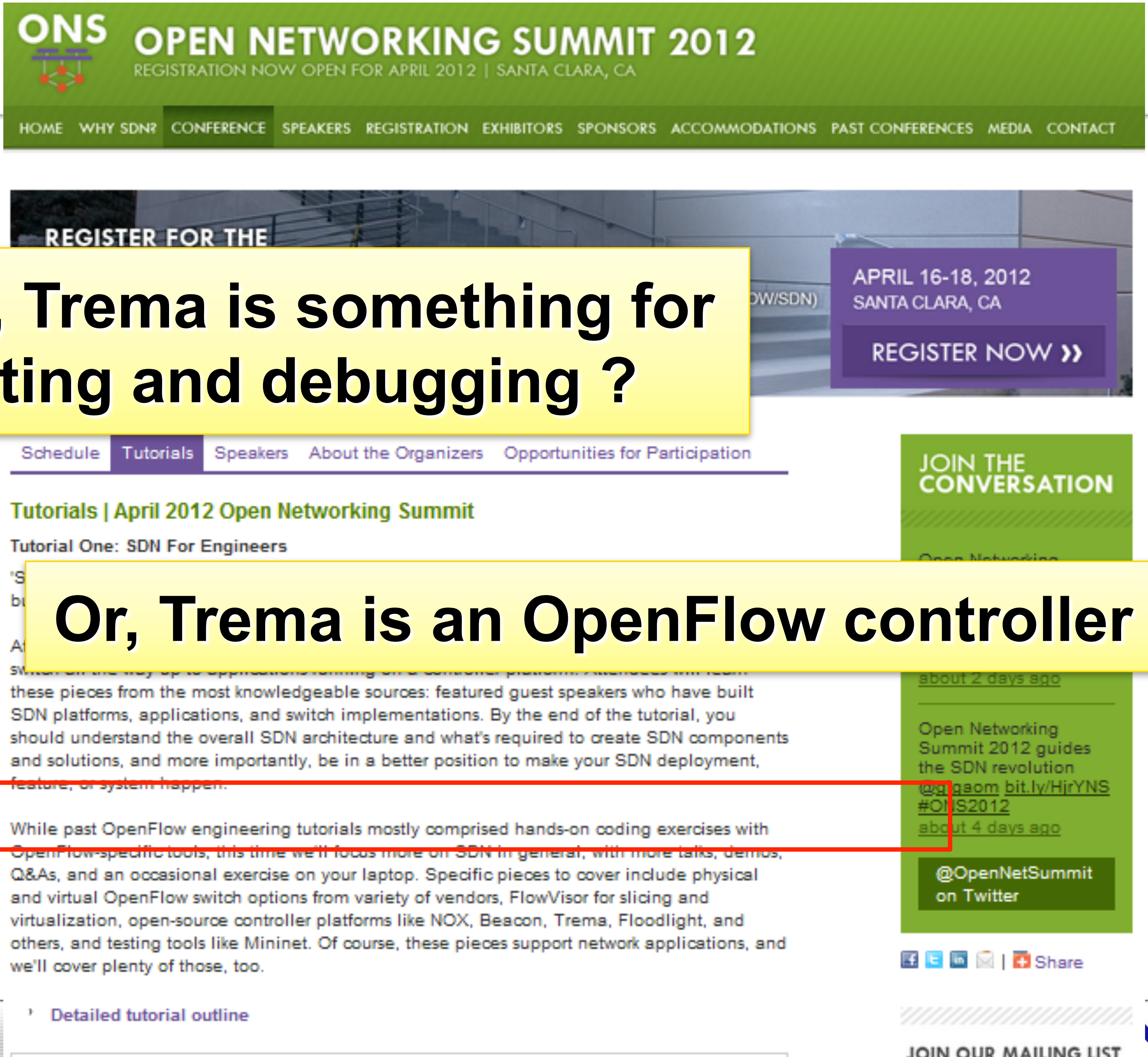
Empowered by Innovation

Trema tutorial

NEC Trema Team
HIDEyuki Shimonishi, Y. Takamiya, K. Sugyo, Y Chiba, K. Suzuki
NEC Corp.

Apr. 16, 2012


ONS
OPEN NETWORKING SUMMIT 2012
REGISTRATION NOW OPEN FOR APRIL 2012 | SANTA CLARA, CA
HOME WHY SDN? CONFERENCE SPEAKERS REGISTRATION EXHIBITORS SPONSORS ACCOMMODATIONS PAST CONFERENCES MEDIA CONTACT
REGISTER FOR THE
APRIL 16-18, 2012
SANTA CLARA, CA
REGISTER NOW ››
Schedule Tutorials Speakers About the Organizers Opportunities for Participation
JOIN THE CONVERSATION
@OpenNetSummit on Twitter
Share
JOIN OUR MAILING LIST

Tutorials | April 2012 Open Networking Summit

Tutorial One: SDN For Engineers

these pieces from the most knowledgeable sources: featured guest speakers who have built SDN platforms, applications, and switch implementations. By the end of the tutorial, you should understand the overall SDN architecture and what's required to create SDN components and solutions, and more importantly, be in a better position to make your SDN deployment, feature, or system happen.

While past OpenFlow engineering tutorials mostly comprised hands-on coding exercises with OpenFlow-specific tools, this time we'll focus more on SDN in general, with more talks, demos, Q&As, and an occasional exercise on your laptop. Specific pieces to cover include physical and virtual OpenFlow switch options from variety of vendors, FlowVisor for slicing and virtualization, open-source controller platforms like NOX, Beacon, Trema, Floodlight, and others, and testing tools like Mininet. Of course, these pieces support network applications, and we'll cover plenty of those, too.

Detailed tutorial outline

about 2 days ago

Open Networking Summit 2012 guides the SDN revolution @gigaom bit.ly/HjrYNS #ONS2012
about 4 days ago

So, Trema is something for testing and debugging ?

Or, Trema is an OpenFlow controller ?

Four things you should know about Trema

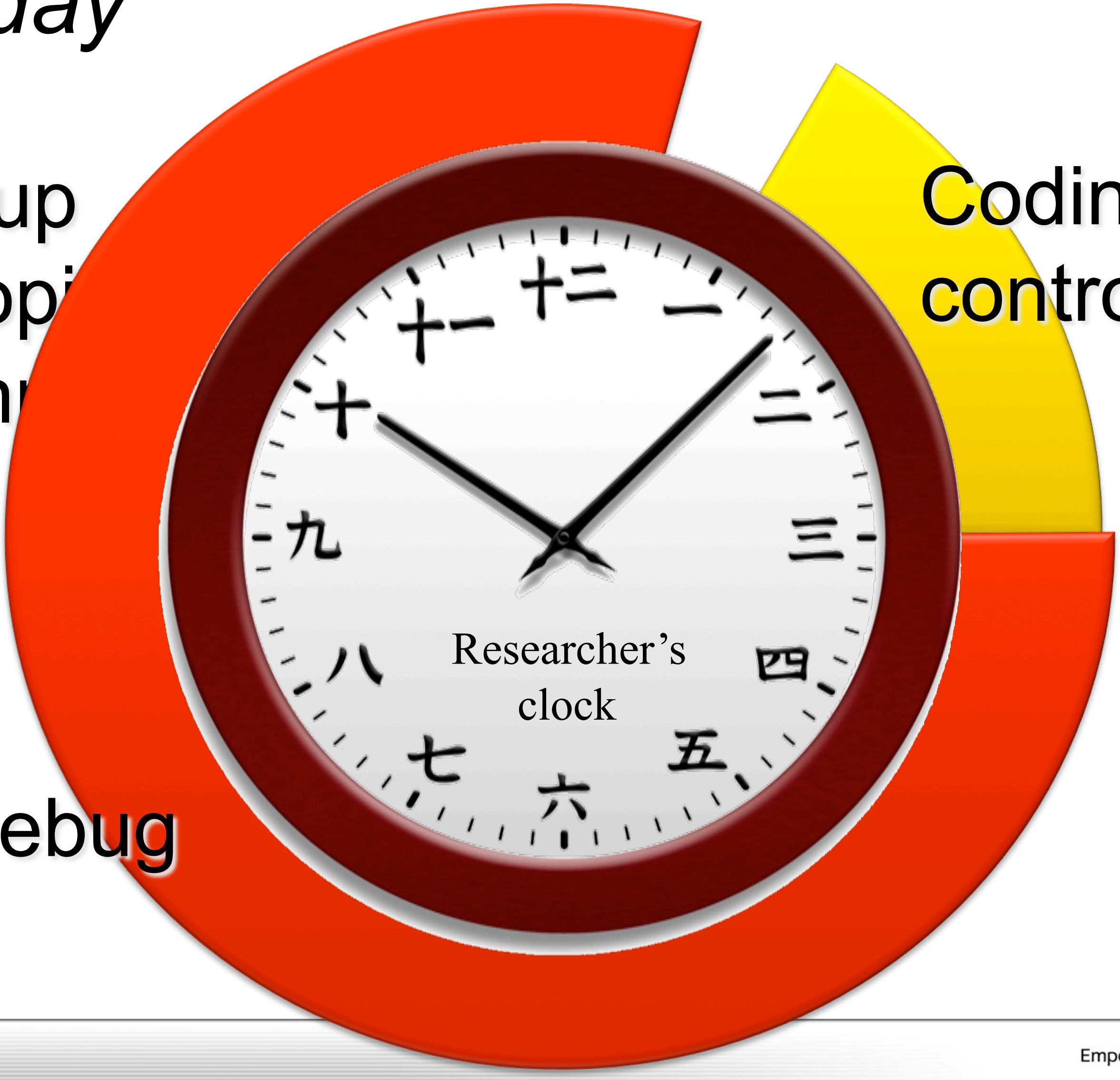
1.

2.

3.

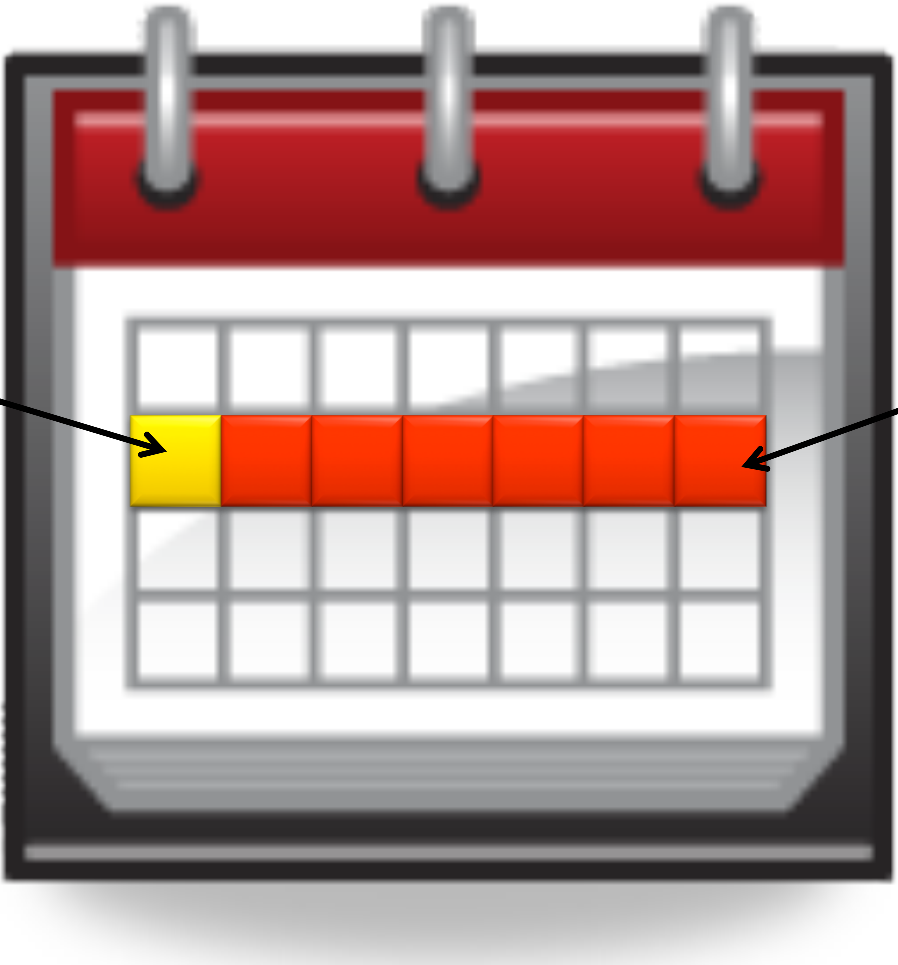
4. What is Trema ?

What will spoil SDN



What will spoil SDN

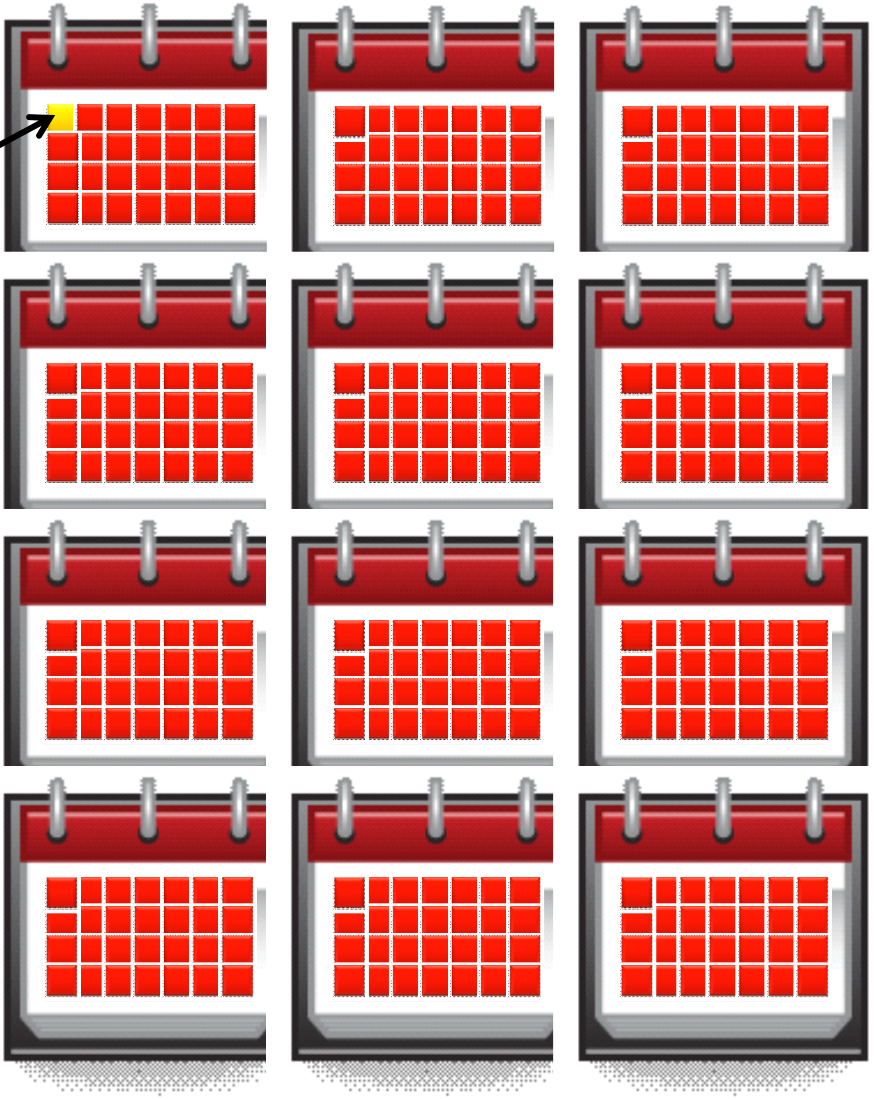
One week



What will spoil SDN

One year

New Year's Day



Our objective

Totally improve productivity of development

- Get start, develop various applications, and debug (not just coding itself)
- Otherwise, my collea id

I am a manager

I am an advisor

I am a coordinator

I'm not a serious programmer

I like to study algorithms

Well, ns2 is OK

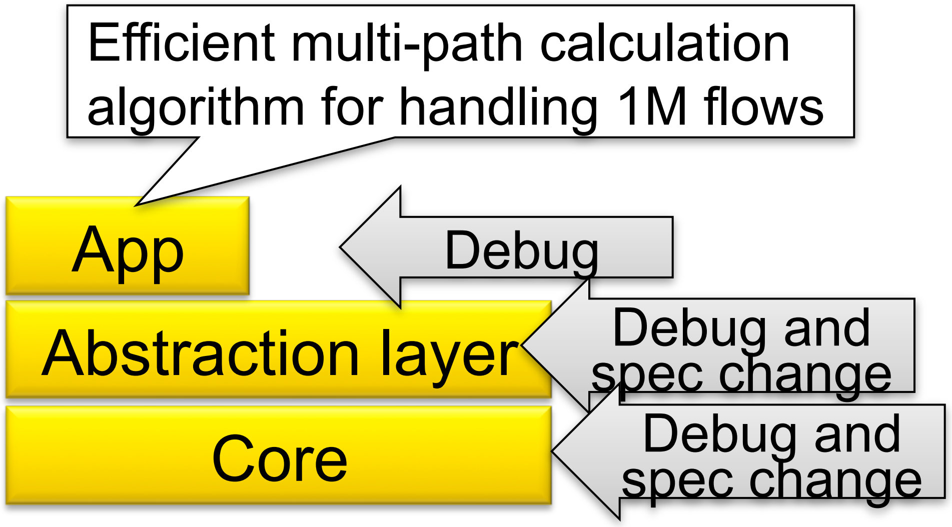
Motivate people to try SDN

Productivity is our key message

Lessons learned

<u>Spectrums of diverse requirement for "controllers"</u>

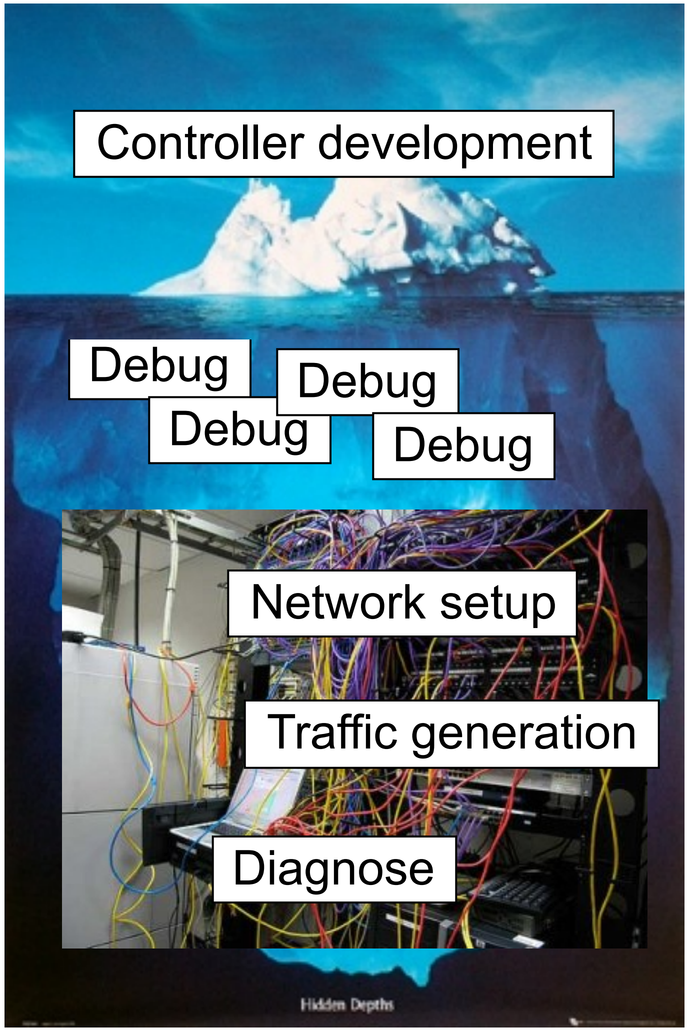
-Networks for research, office, data center, carrier, etc.
-Data consistency requirements
-Scalability requirements
-Static or dynamic flow push, micro flow or aggregated, etc.

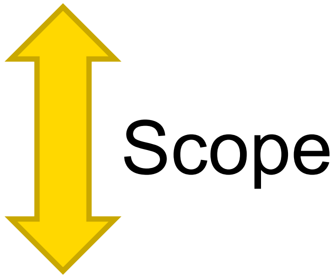


Abstraction layers should be designed flexibly, and maybe co-developed with applications (rather than co-developed with controllers)

Lessons learned

OpenFlow iceberg





Seamless integration of operations and state monitoring among controller applications, switches, hosts, etc.

What is Trema

User application
Your OpenFlow controller (in Ruby or C)

Abstraction mechanisms, and high-level APIs

Core:
OpenFlow controller libraries and modules

OpenFlow controller

Trema

Developing environment:
Network/host emulator, debugging tool, etc

Operation environment:
trema commands

Trema is “OpenFlow programming framework” for Ruby and C (GPL2)

Four things you should know about Trema

1.

2.

3. High-productivity

4. What is Trema ?

Design goal of Trema "Framework"

To improve OpenFlow development productivity…

Run it quick

- Closed-loop development - code, test, debug

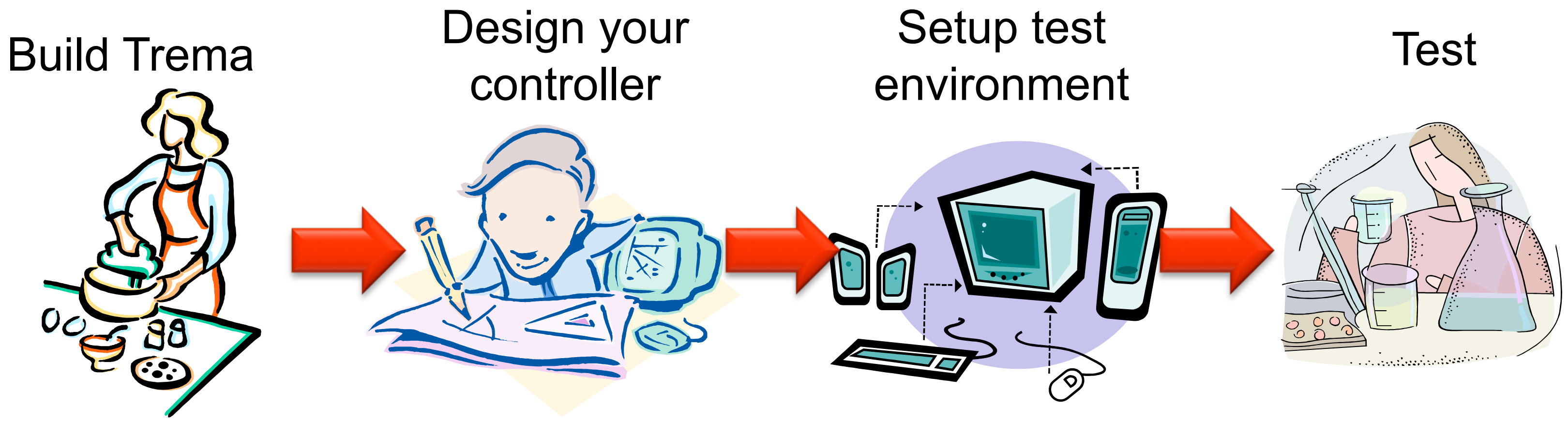
Write it short

- Coding by convention

Fix it easy

- Integrated debugging helps

RUN IT QUICK



How quick ? ➔ Let me show

Demo

-Build Trema

-Design a controller

-Setup test environment

-Test (Packet send and receive)

-Using real switches (without emulator)

WRITE IT SHORT

```
class RepeaterHub < Controller                # Create a new controller class

  def packet_in datapath_id, message          # Packet-in received handler
    send_flow_mod_add(
      datapath_id,                                   Send flow_mod
      :match => ExactMatch.from( message ),
      :actions => ActionOutput.new( OFPP_FLOOD )
    )
    send_packet_out(
      datapath_id,                                   Send packet_out
      :packet_in => message,
      :actions => ActionOutput.r
    )
  end
end
```

An example fo

Reads smooth

Coding by convention

Syntactic sugar

Default options

Love

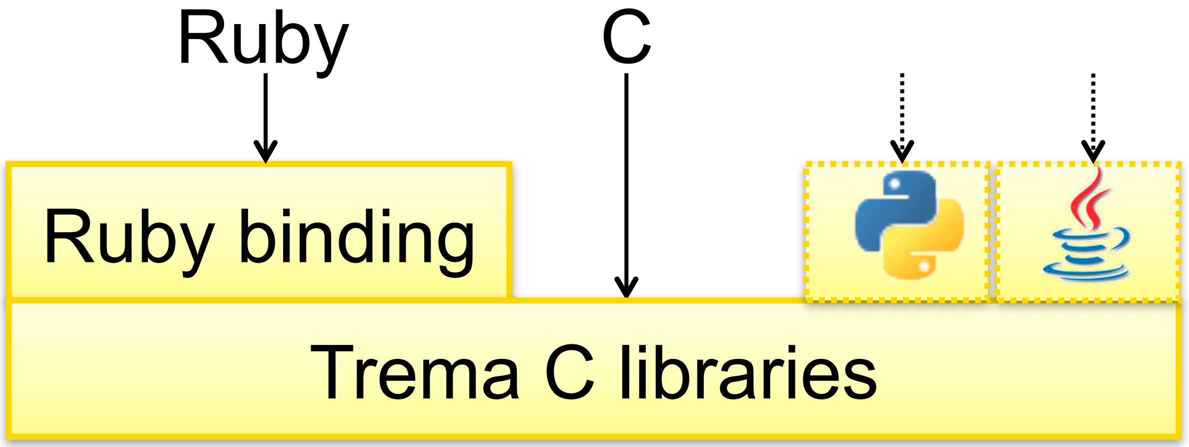
Ruby ?

C ?

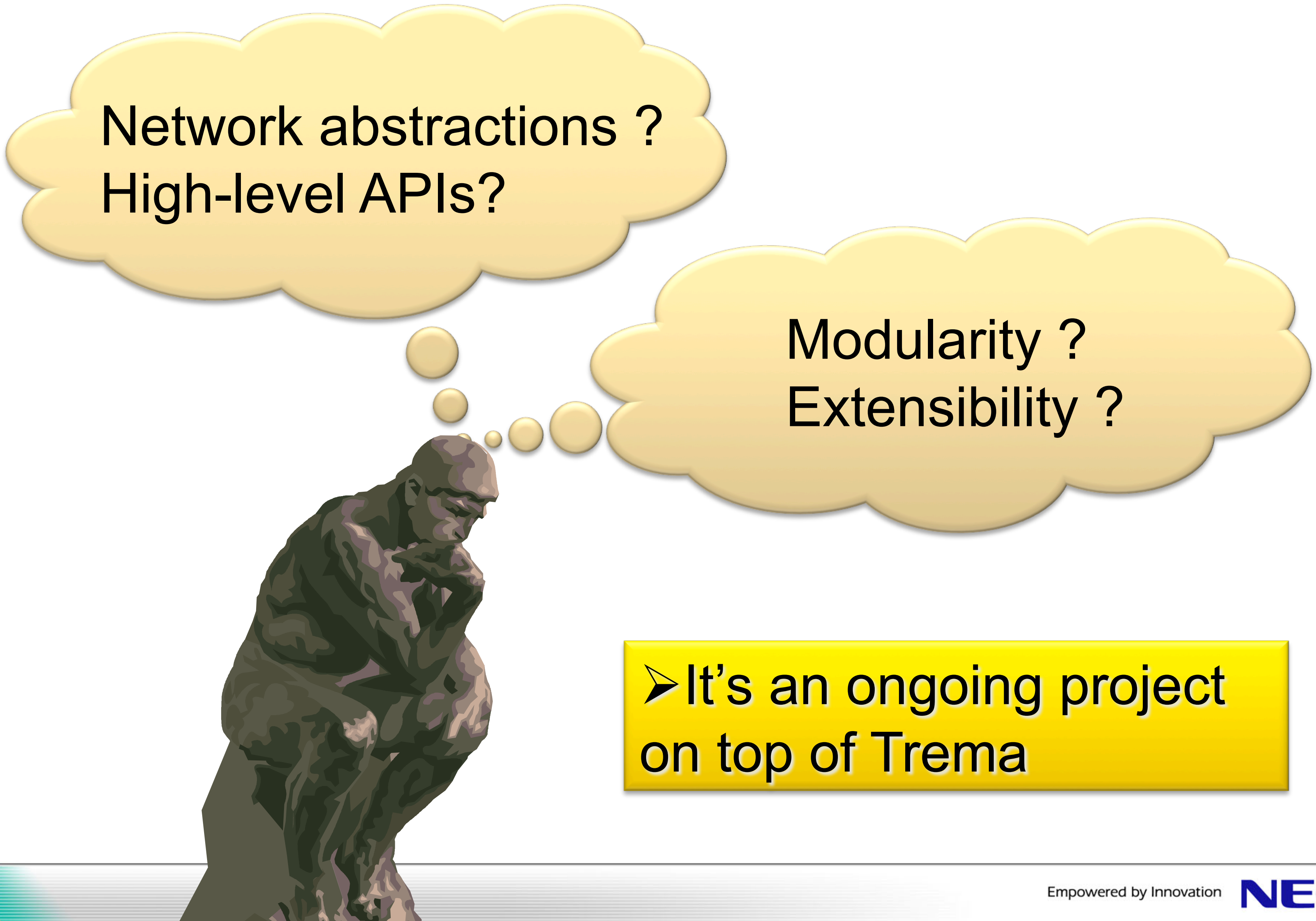
From rapid prototyping to production use

- Trema provides libraries for both Ruby and C
  - Script language for productivity
  - Compile language for performance

- Trema C is also as simple as Trema Ruby
  - Most of practical applications in use are written in C



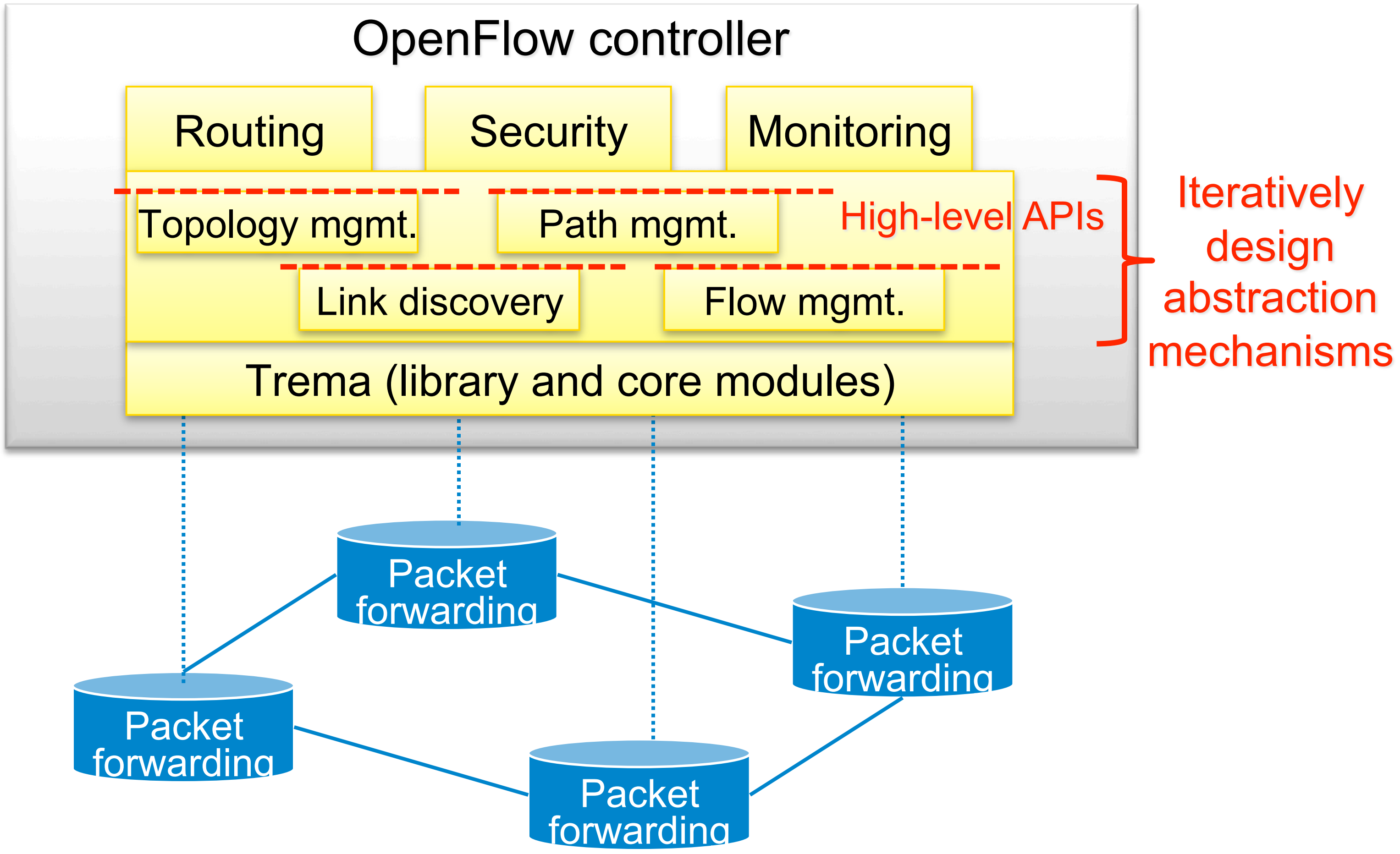
Forget about this ?



➢It's an ongoing project on top of Trema

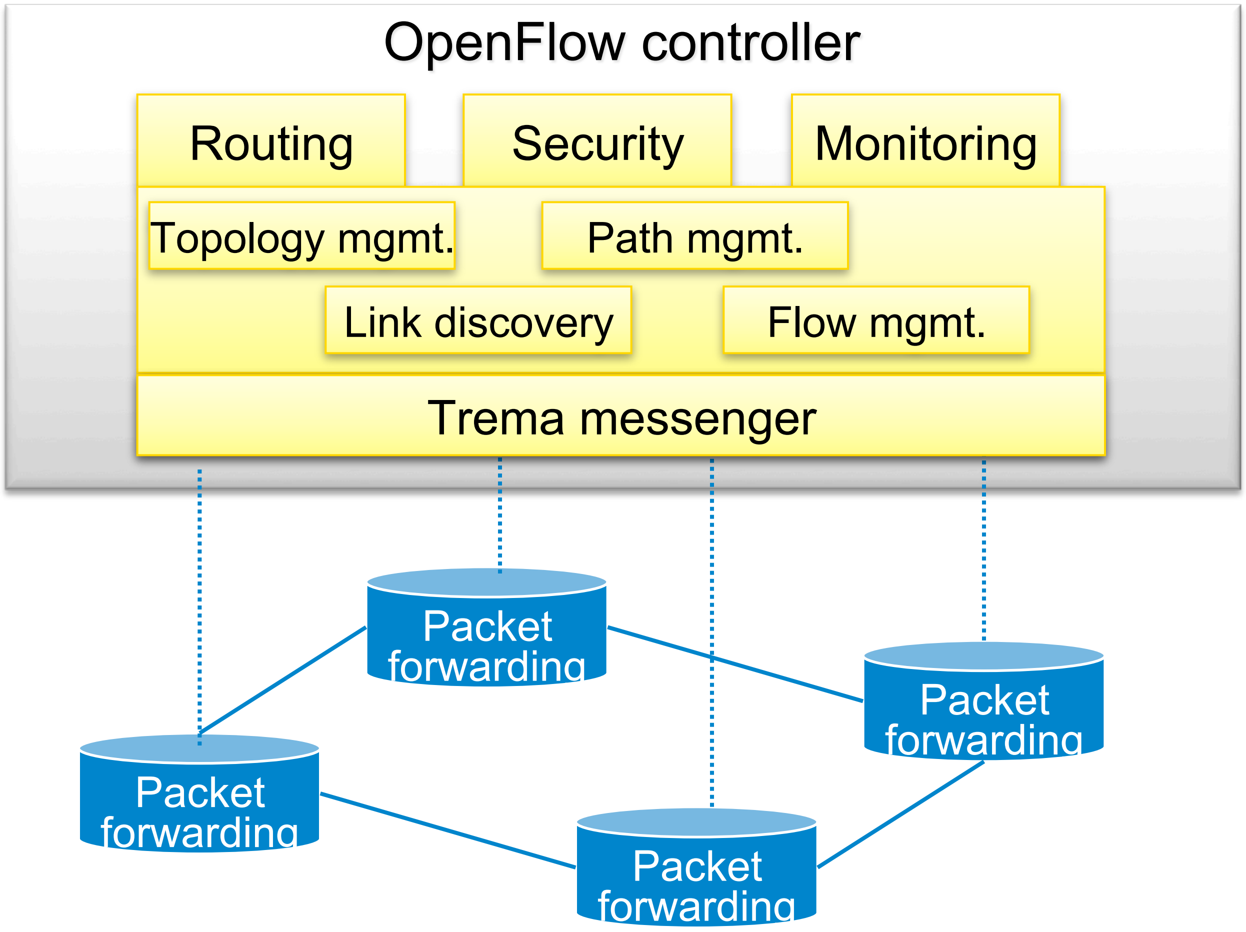
(To study) network abstractions

- Trema provides no high-level APIs or abstractions
- Common function modules provide their own APIs



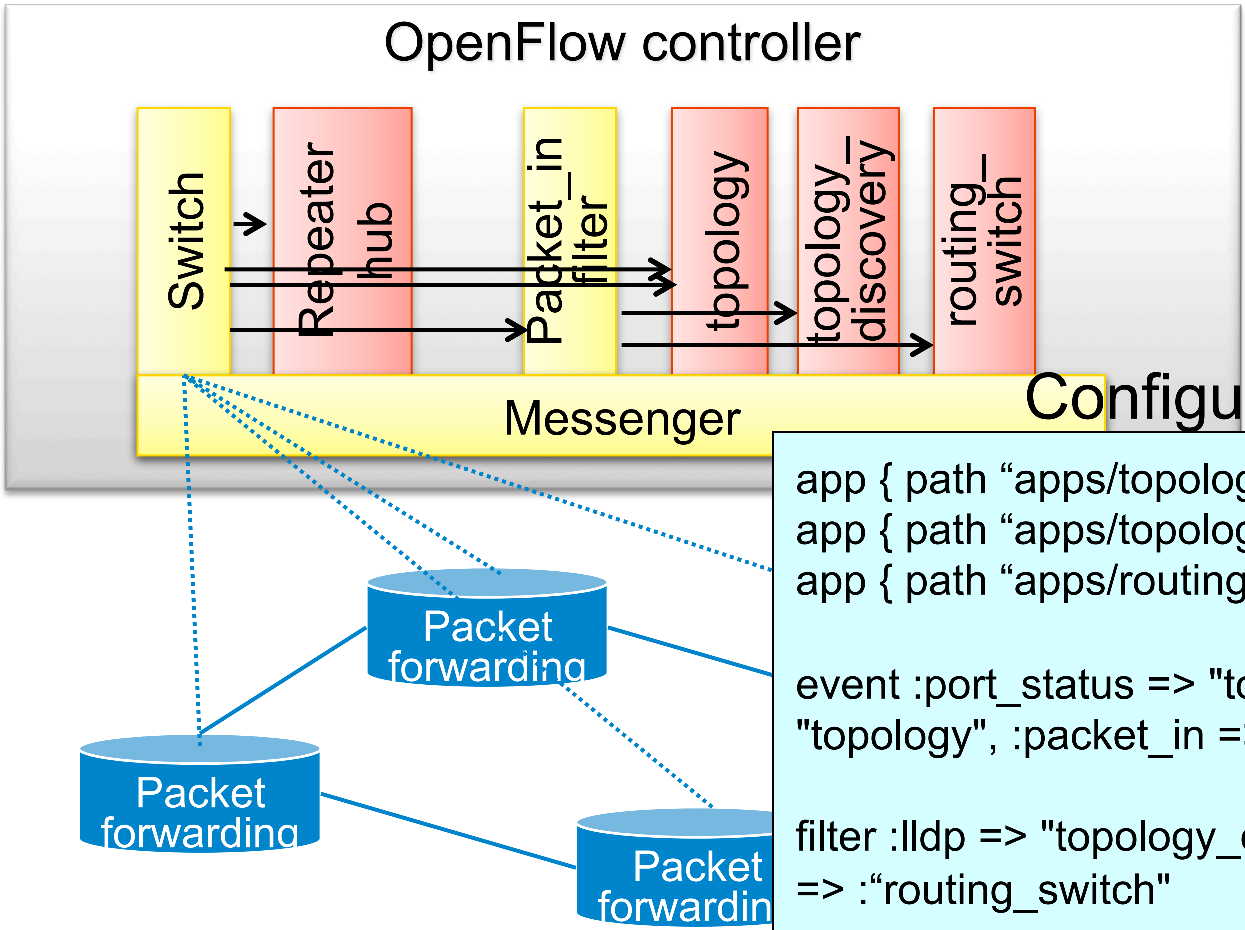
Trema implementation

- Module = process, messenger = IPC
- Better isolation for 3rd party modules, partial updates, etc.



An example

- Repeater-hub
- Routing switch case (@TremaApps)



Configuration file

```
app { path “apps/topology/topology“ }
app { path “apps/topology/topology_discovery“ }
app { path “apps/routing_switch/routing_switch“ }

event :port_status => "topology", :state_notify =>
"topology", :packet_in => "filter”

filter :lldp => "topology_discovery", :packet_in
=> :“routing_switch"
```

FIX IT EASY

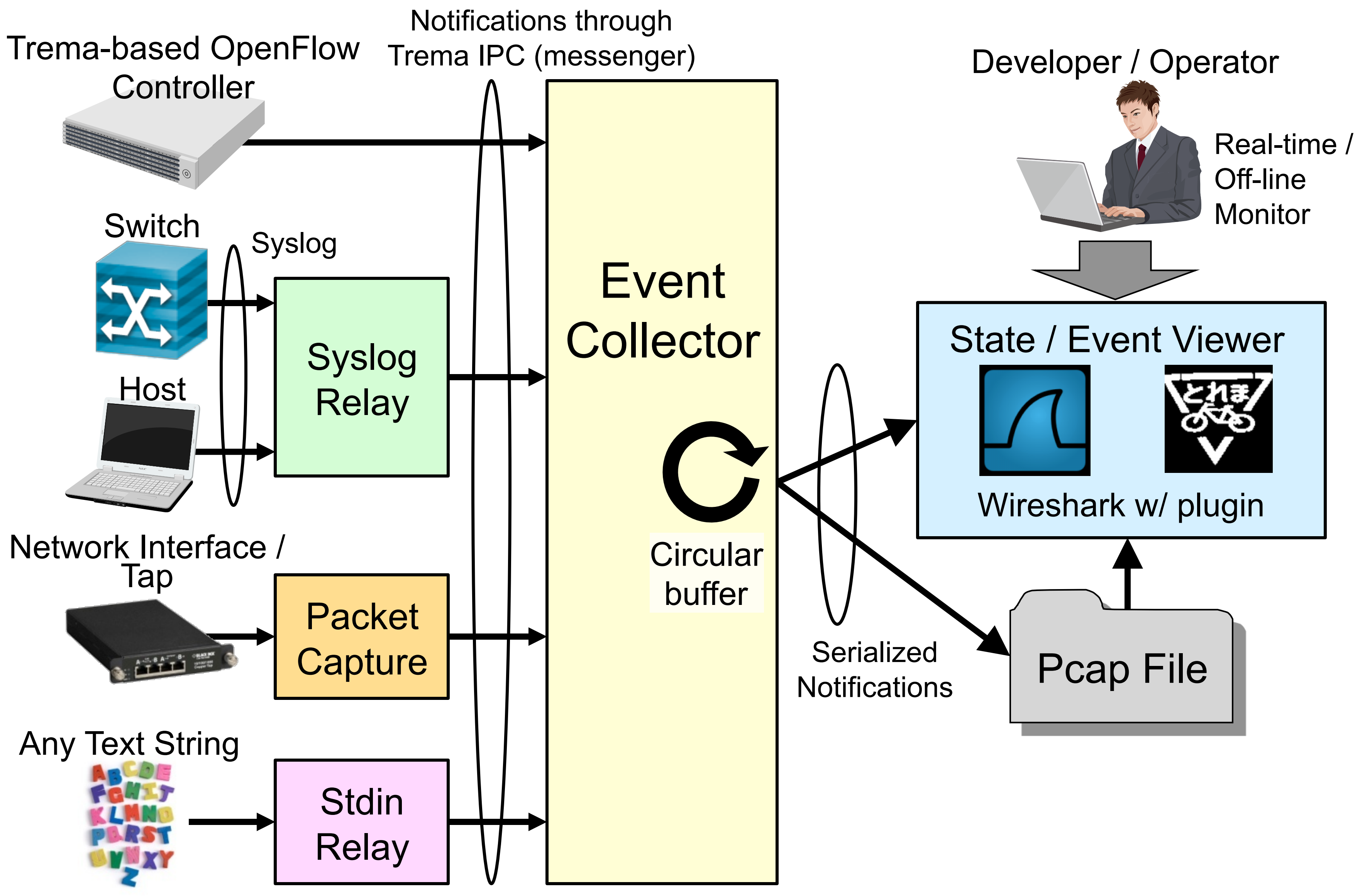
Demo

-Trema dump_flows

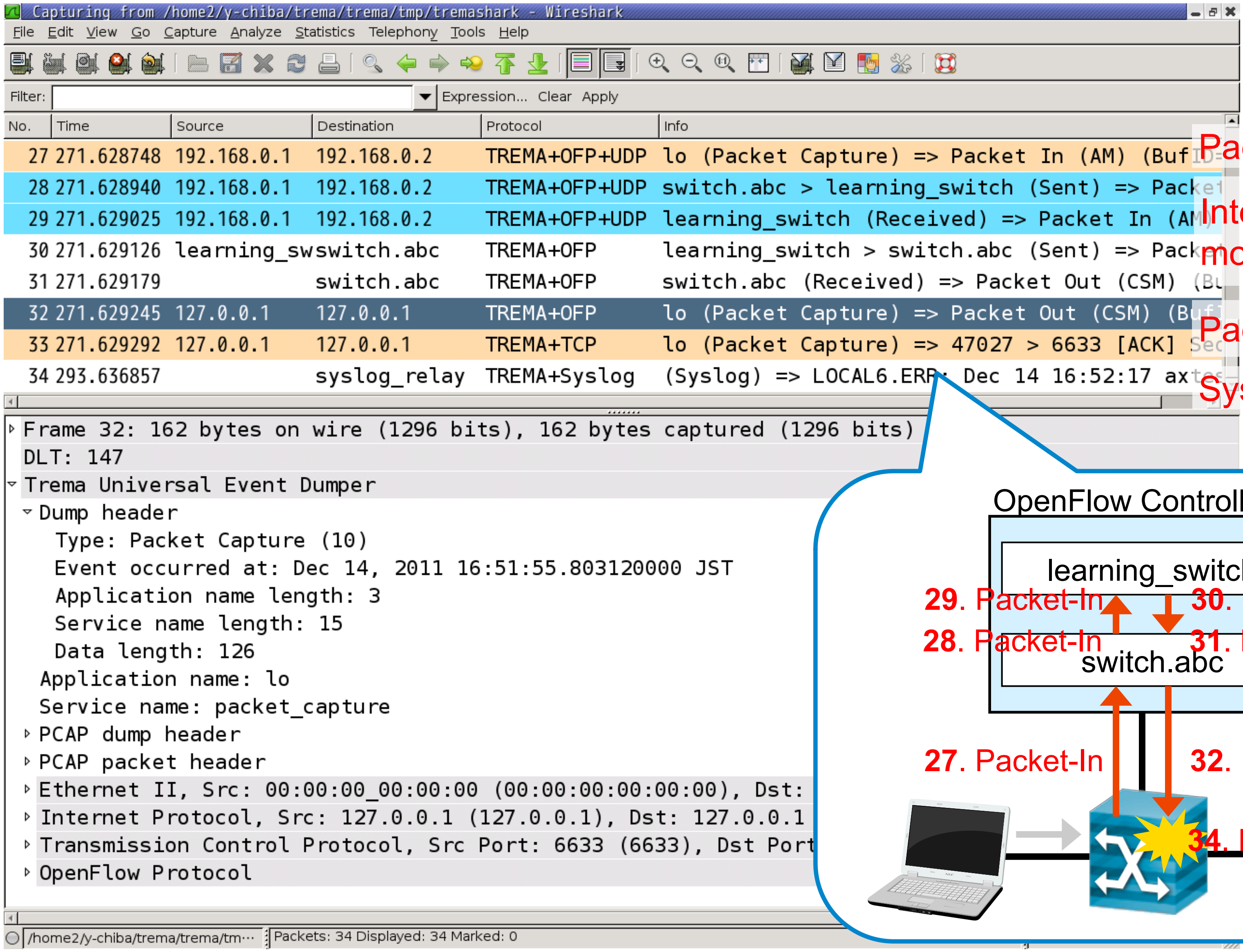
-Trema ruby

-TremaShark

TremaShark


Trema-based OpenFlow Controller
Notifications through Trema IPC (messenger)
Developer / Operator
Real-time / Off-line Monitor
Switch
Syslog
Host
Syslog Relay
Event Collector
Circular buffer
State / Event Viewer
Wireshark w/ plugin
Network Interface / Tap
Packet Capture
Serialized Notifications
Pcap File
Any Text String
Stdin Relay

TremaShark



Capturing from /home2/y-chiba/trema/trema/tmp/tremashark - Wireshark

File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply

```
No. Time       Source          Destination   Protocol       Info
27 271.628748 192.168.0.1     192.168.0.2   TREMA+OFP+UDP  lo (Packet Capture) => Packet In (AM) (Buf
28 271.628940 192.168.0.1     192.168.0.2   TREMA+OFP+UDP  switch.abc > learning_switch (Sent) => Packe
29 271.629025 192.168.0.1     192.168.0.2   TREMA+OFP+UDP  learning_switch (Received) => Packet In (AM
30 271.629126 learning_swswitch.abc         TREMA+OFP      learning_switch > switch.abc (Sent) => Pack
31 271.629179                 switch.abc    TREMA+OFP      switch.abc (Received) => Packet Out (CSM) (Bu
32 271.629245 127.0.0.1       127.0.0.1     TREMA+OFP      lo (Packet Capture) => Packet Out (CSM) (Buf
33 271.629292 127.0.0.1       127.0.0.1     TREMA+TCP      lo (Packet Capture) => 47027 > 6633 [ACK] Sec
34 293.636857                 syslog_relay  TREMA+Syslog   (Syslog) => LOCAL6.ERR: Dec 14 16:52:17 ax
```

```
Frame 32: 162 bytes on wire (1296 bits), 162 bytes captured (1296 bits)
DLT: 147
Trema Universal Event Dumper
  Dump header
    Type: Packet Capture (10)
    Event occurred at: Dec 14, 2011 16:51:55.803120000 JST
    Application name length: 3
    Service name length: 15
    Data length: 126
  Application name: lo
  Service name: packet_capture
  PCAP dump header
  PCAP packet header
  Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst:
  Internet Protocol, Src: 127.0.0.1 (127.0.0.1), Dst: 127.0.0.1
  Transmission Control Protocol, Src Port: 6633 (6633), Dst Port
  OpenFlow Protocol
```

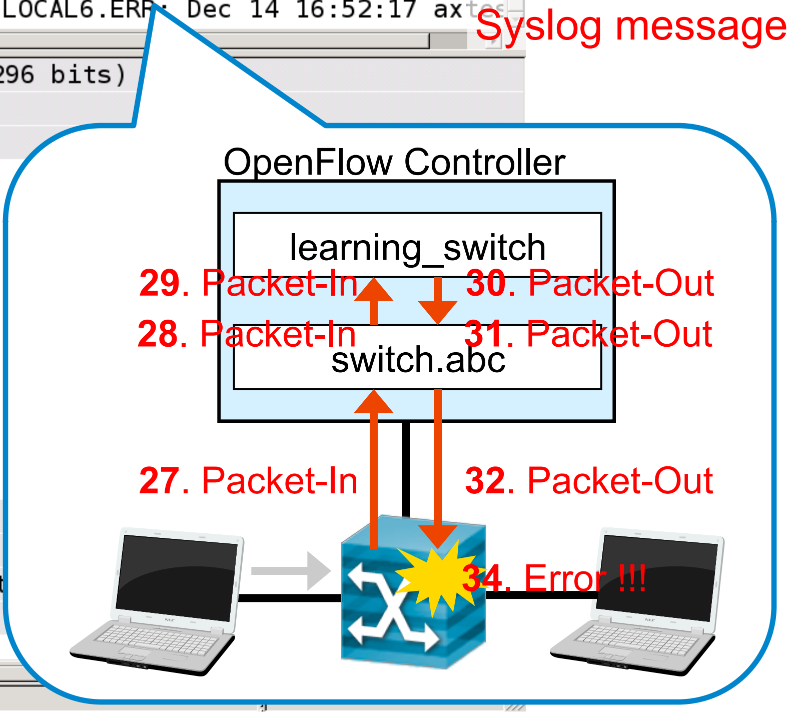
/home2/y-chiba/trema/trema/tm··· Packets: 34 Displayed: 34 Marked: 0

Packet capture

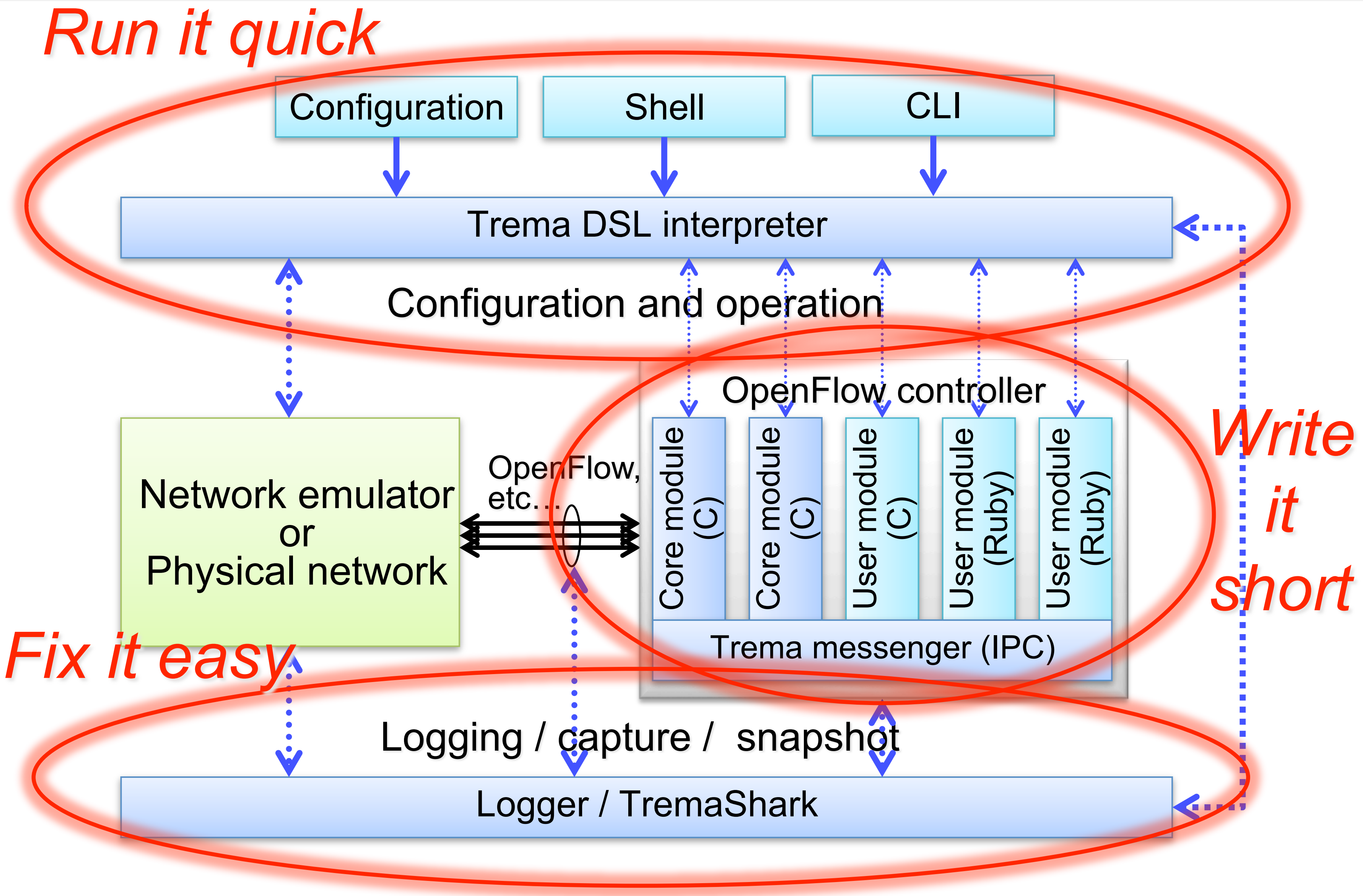
Inter-function module events

Packet capture

Syslog message



Trema architecture overview



Four things you should know about Trema

2. TremaApps

3. High-productivity

4. What is Trema ?

TremaApps

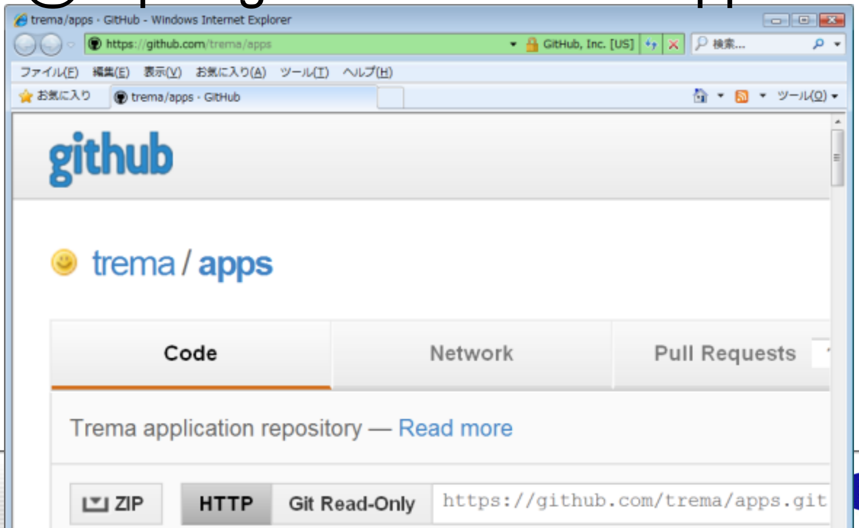
TremaApps

Let us share.
We welcome your contribution

Trema

- Practical/experimental controllers developed on top of Trema
- Experimental abstraction modules
- Good starting point for developing real-world controllers

TremaApps
@https://github.com/trema/apps

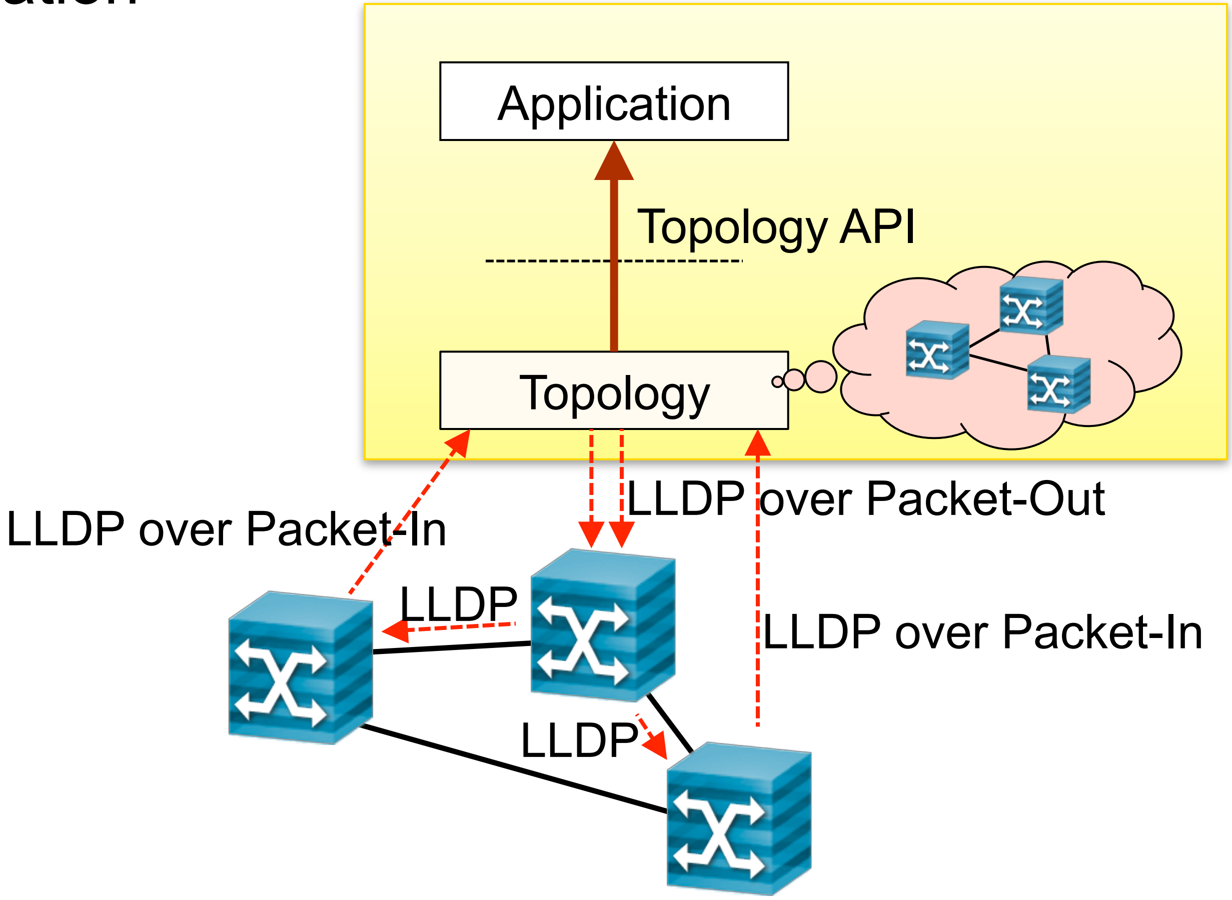

trema/apps - GitHub - Windows Internet Explorer
https://github.com/trema/apps
GitHub, Inc. [US]
検索...
ファイル(F) 編集(E) 表示(V) お気に入り(A) ツール(T) ヘルプ(H)
お気に入り
trema/apps - GitHub
ツール(O)
github
trema / apps
Code
Network
Pull Requests
ZIP
HTTP
Git Read-Only

Trema application repository — Read more

https://github.com/trema/apps.git

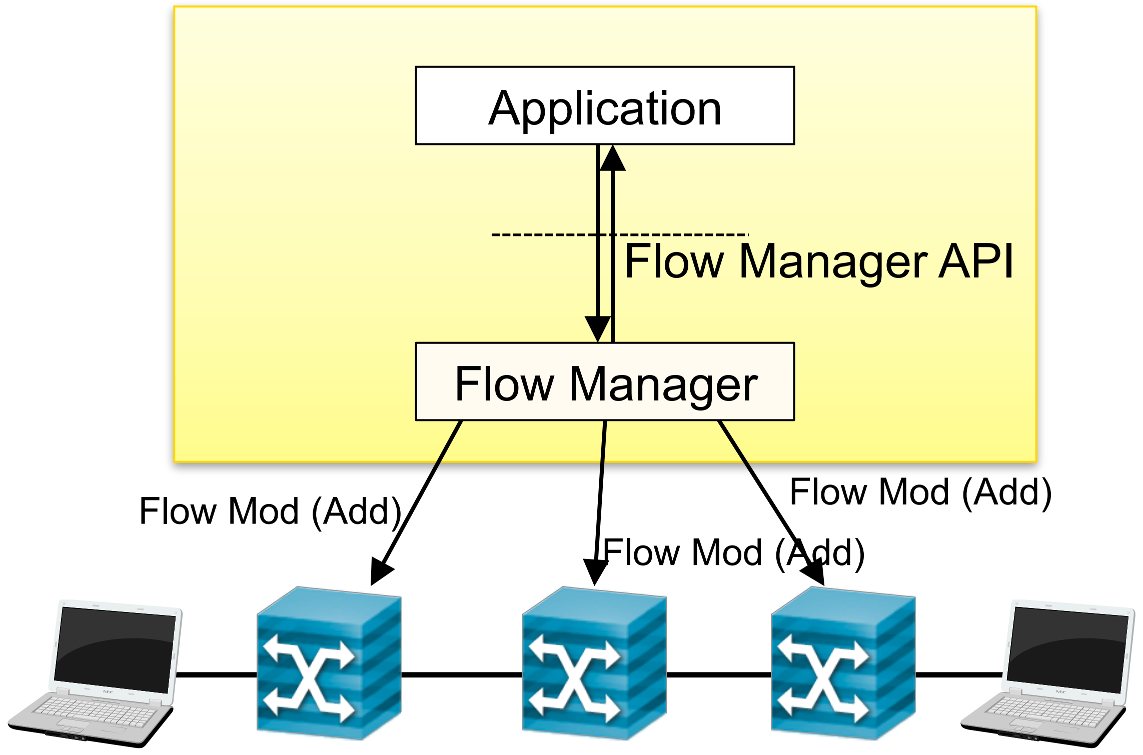
Trema/Apps: Topology

- Discovers network topology using LLDP frames
- Provides API for retrieving a discovered topology from other application



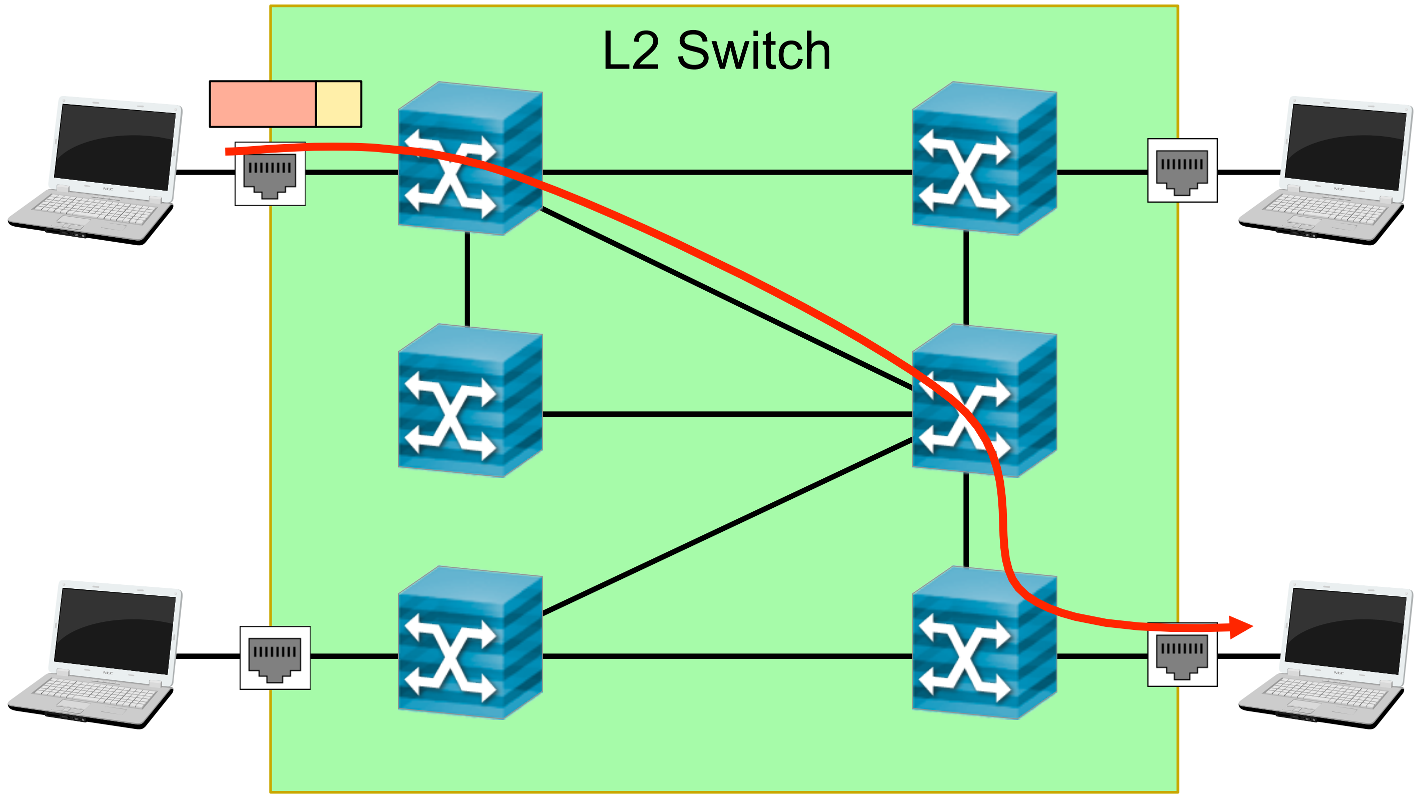
Trema/Apps: Flow Manager

- Provides API for managing a set of flow entries as a single group
- Guarantees all entries in a group are properly installed and removed



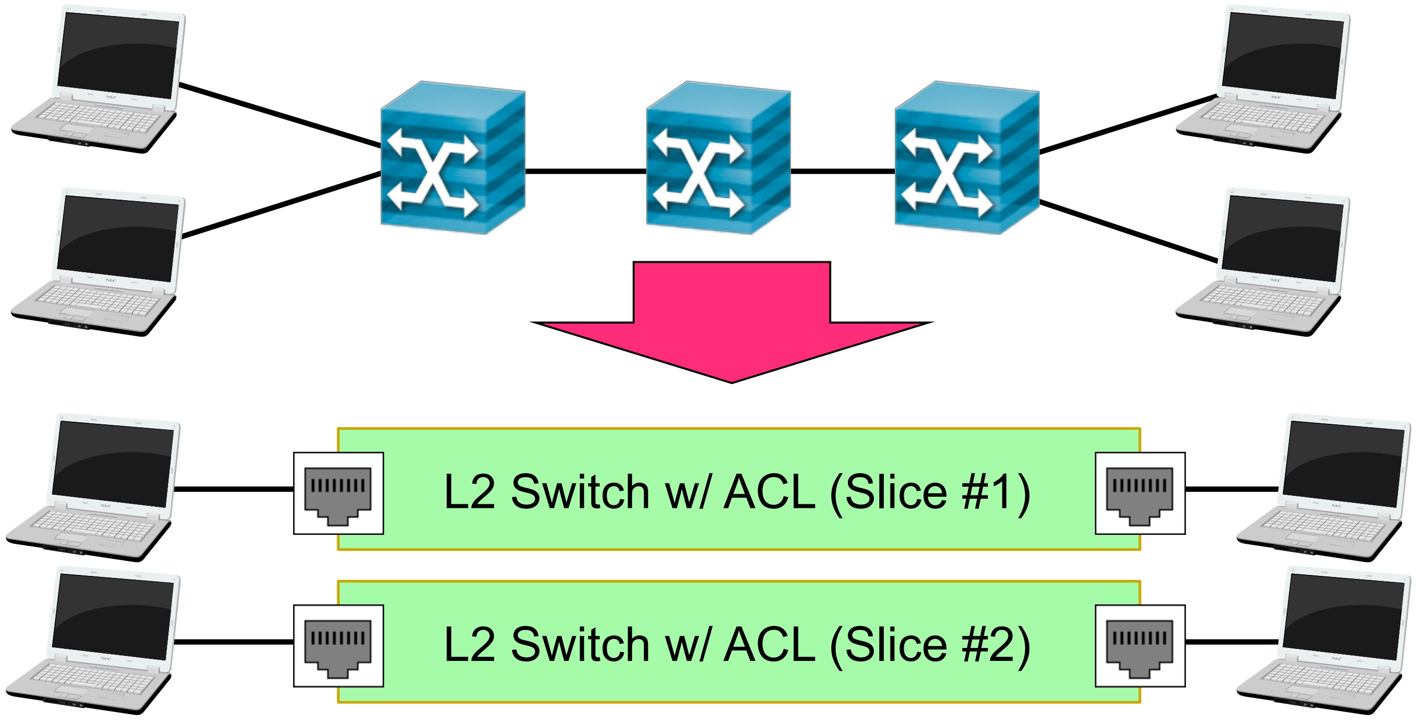
TremaApps: Routing Switch

- Creates a Layer-2 switch consisting of OpenFlow switches
- Resolves internal paths using a shortest path algorithm

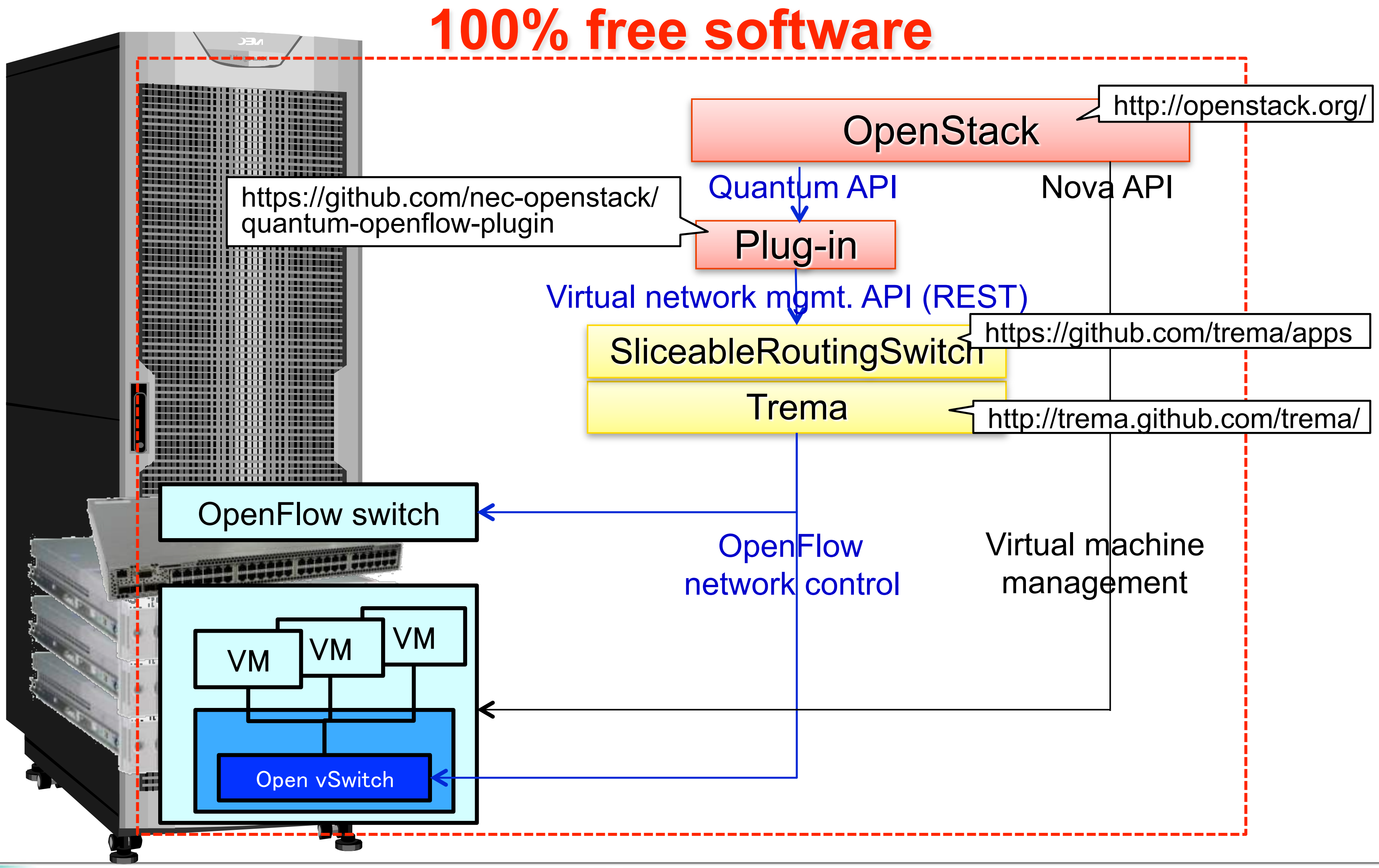


Trema/Apps: Sliceable Routing Switch

- Creates virtual L2 network domains (slices) with L1-4 ACL
- North-bound API (REST) for slice and ACL management
  - Create/delete a slice
  - Attach/detach a host to/from a slice
  - Add/delete an ACL entry



Sliceable Routing Switch and OpenStack


100% free software
OpenStack
http://openstack.org/
Quantum API
Nova API
https://github.com/nec-openstack/quantum-openflow-plugin
Plug-in
Virtual network mgmt. API (REST)
SliceableRoutingSwitch
https://github.com/trema/apps
Trema
http://trema.github.com/trema/
OpenFlow switch
OpenFlow network control
Virtual machine management
VM
VM
VM
Open vSwitch

Four things you should know about Trema

1. **Tested and supported**

2. TremaApps

3. High-productivity

4. What is Trema ?

Tested and supported

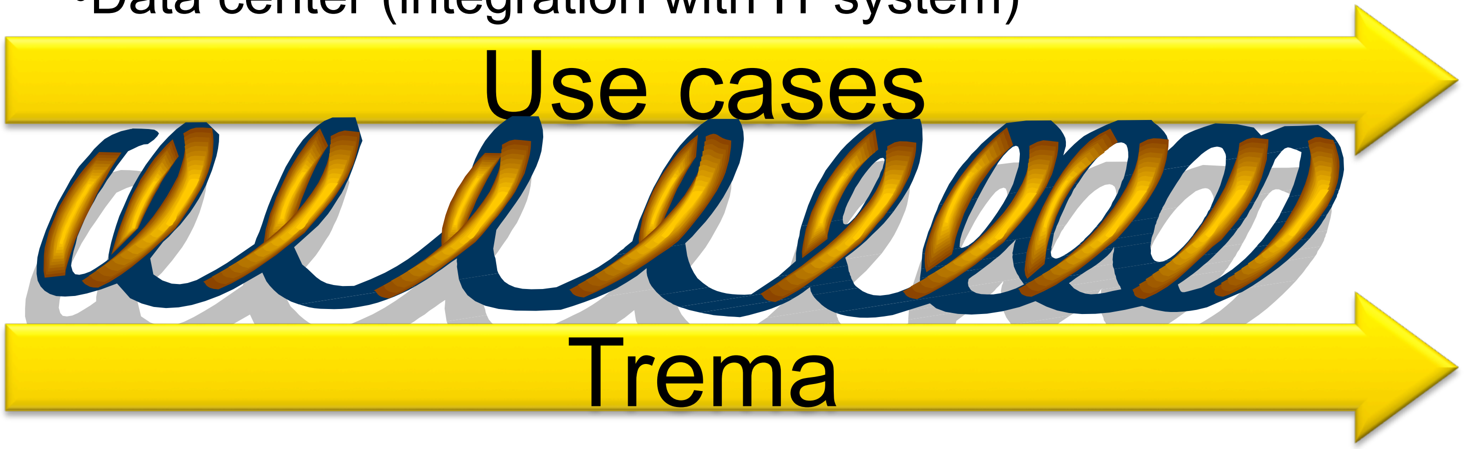
Trema: Full-Stack OpenFlow Framework for Ruby/C - Windows Internet Explorer
http://trema.github.com/trema/
Live Search
ファイル(F)
ツール(T)
ヘルプ(H)
command. You can describe the network topology in
$ cat network.conf
インターネット
105%

Future directions and roadmap

- Keep update Trema
  - We use it as our research platform and production platform

- Keep spiral development of Trema and its use cases

•Carrier network (scalability)
•Data center (integration with IT system)



•Many feedbacks from application development
•Better define RoutingSwitch, and high-level APIs / north-bound APIs
•Stronger integration with network environment (new software switch)

Another option for hands-on

Self-study tutorial

- Just follow the slides. No Ruby knowledge needed in advance.
- Files: https://github.com/trema/tutorial.files
- Slides: http://trema-tutorial-gec13.heroku.com/

Useful examples in “[trema]/src/examples”

- Simple samples demonstrating API usage
- Both Ruby and C samples

Explore TremaApps

- https://github.com/trema/apps

Help desk for hands-on

- If you get any troubles, call our help desk
  - Tweet to “@yasuhito” with hash-tag “#trema”
  - Email to Trema-ML
  - Engineers are standing by for you [3:15pm-4:15pm]

THANK YOU

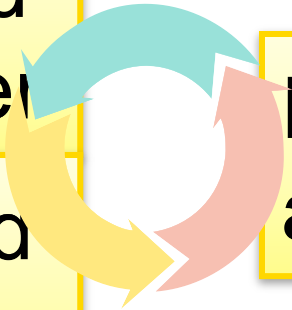
Backup

How we design this time

Design a controller with all high-level features → Develop many applications

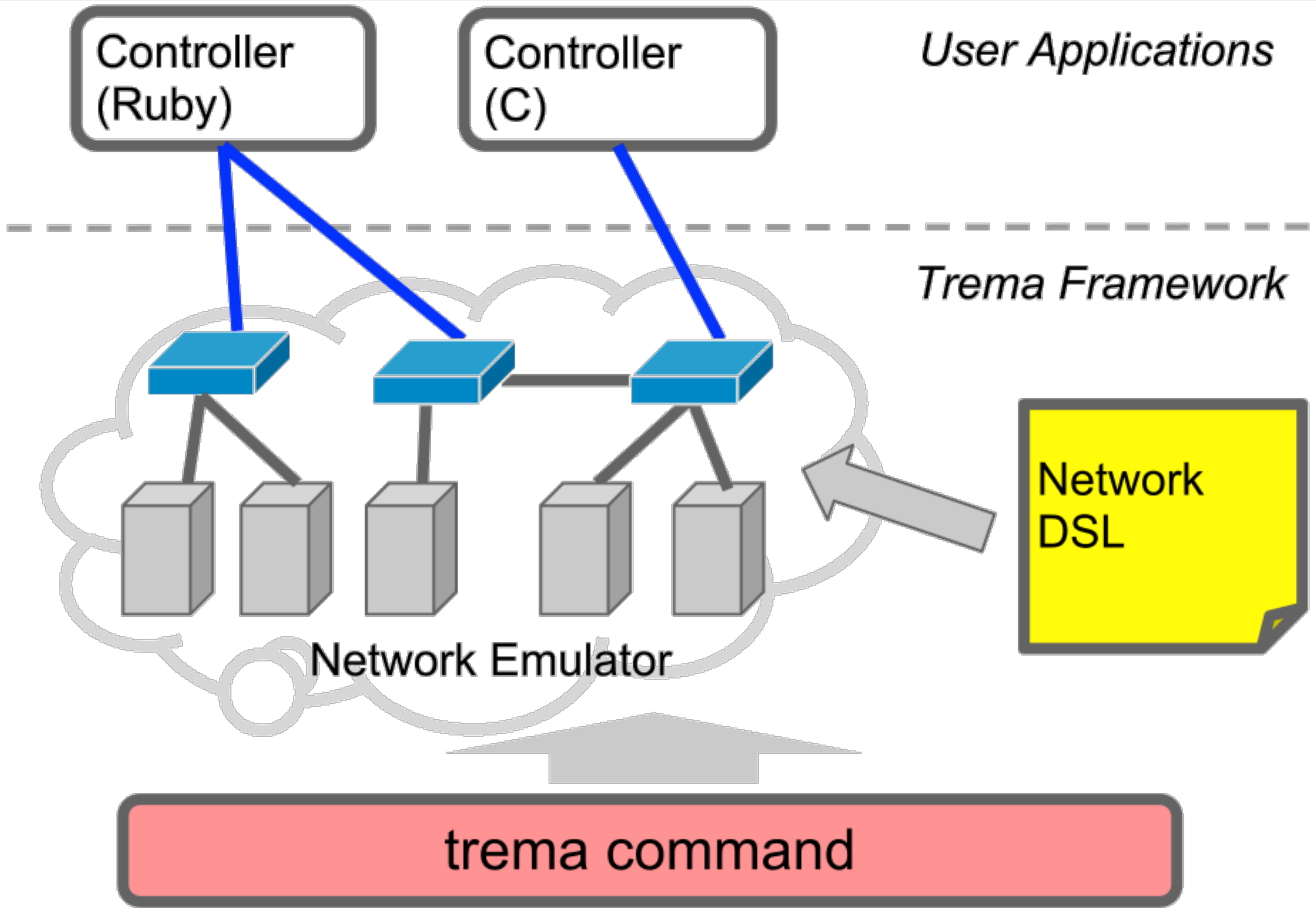
Design a "**simple**" and "**productive**" controller

Design useful tools and abstraction layers



Develop many applications

Scope of Trema



Auto handler dispatch by naming convention

```
class MyController < Controller

  def packet_in dpid, msg          # Packet-in received handler
    # ...
  end

  def features_reply dpid, msg     # features-reply handler
    # ...
  end

  # ...
end
```

```
public Command receive(IOFSwitch sw, ...)
{
  switch (msg.getType()) {
    case PACKET_IN:
      return this.handlePacketIn(sw, ...);
    case features_reply
…
}
```

- Coding conventions for concise and compact code
  - No need to write dispatchers

Syntactic Sugar

```
ExactMatch.from( message )
```

v.s.

```
Match.new(
   :in_port => message.in_port,
   :nw_src => message.nw_src,
   :nw_dst => message.nw_dst,
   :tp_src => message.tp_src,
   :tp_dst => message.tp_dst,
   :dl_src => message.dl_src,
   :dl_dst => message.dl_dst,
 ...
)
```

Default option

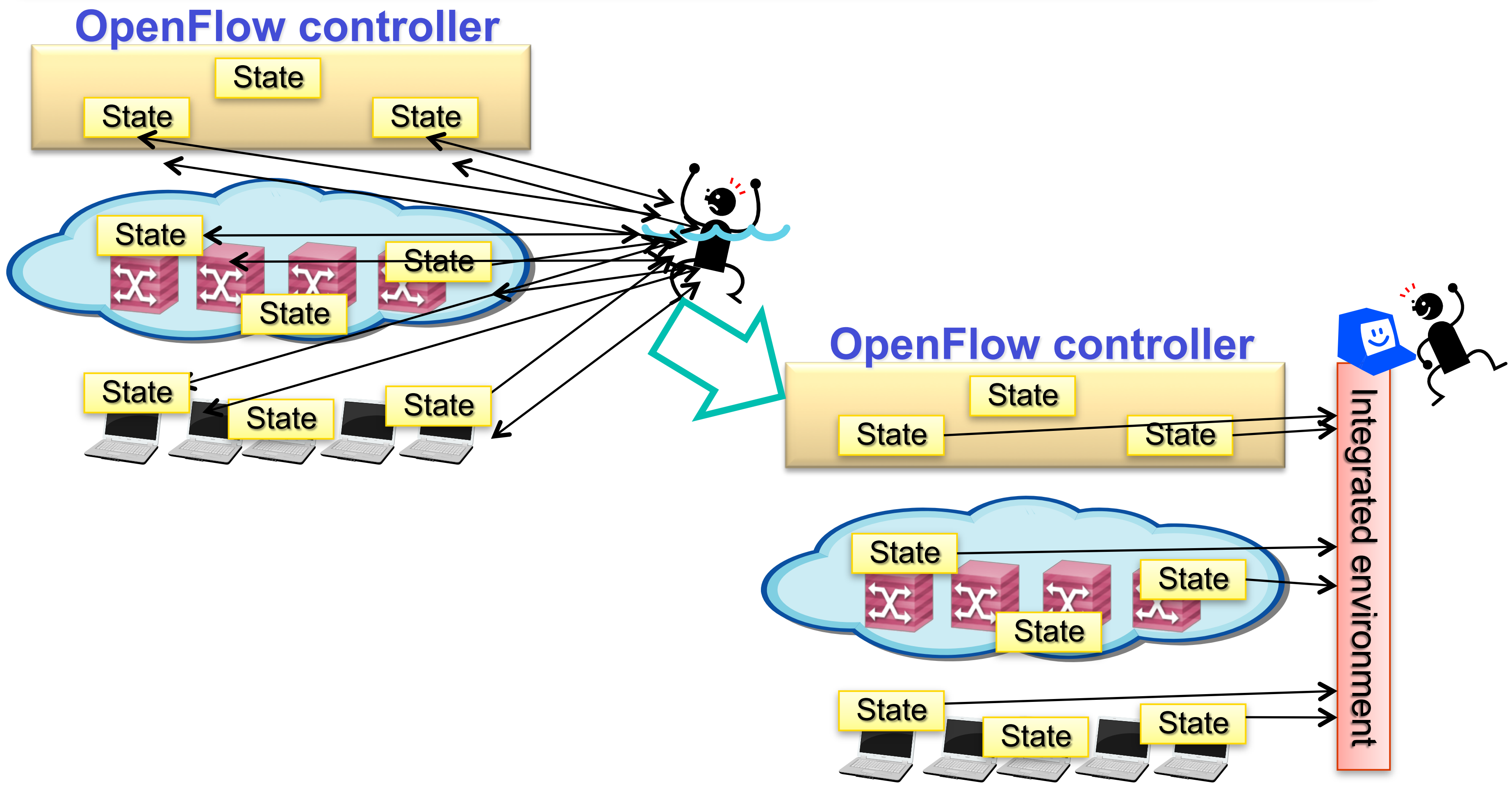
```
send_flow_mod_add(
  dpid,
  :match => ExactMatch.from( message ),
  :actions => ActionOutput.new( port_no )
)
```

v.s.

```
inst.install_datapath_flow(
  dpid,
  extract_flow(packet),
  CACHE_TIMEOUT,
  openflow.OFP_FLOW_PERMANENT,
  [[openflow.OFPAT_OUTPUT, [0, prt[0]]]],
  bufid,
  openflow.OFP_DEFAULT_PRIORITY,
  inport,
  buf
)
```

Integration of controller and network environment

Network programming is essentially distributed system programming



Systematic support to manage, monitor, and diagnosis entire system

Wanted

✓Share your applications at TremaApps

✓Discussions on north-bound API and OpenStack integration

✓Discussions on network abstraction and high-level APIs

✓Any comments and suggestions

Conclusion

What is Trema ?

- Trema is “OpenFlow programming framework” for Ruby and C (GPL2)

High-productivity

- Run it quick
- Write it short
- Fix it easy

TremaApps

- Many useful application
- Fully open source cloud software suite

Tested and supported

What's Trema

"Trema is something for testing and debugging ?"

"Trema is an OpenFlow controller ?"

Trema is "OpenFlow programming framework"

for Ruby and C (GPL2)

WHY ?

High-productivity for this "Post-Rails" era