ns-3 Introduction

Tom Henderson (University of Washington)

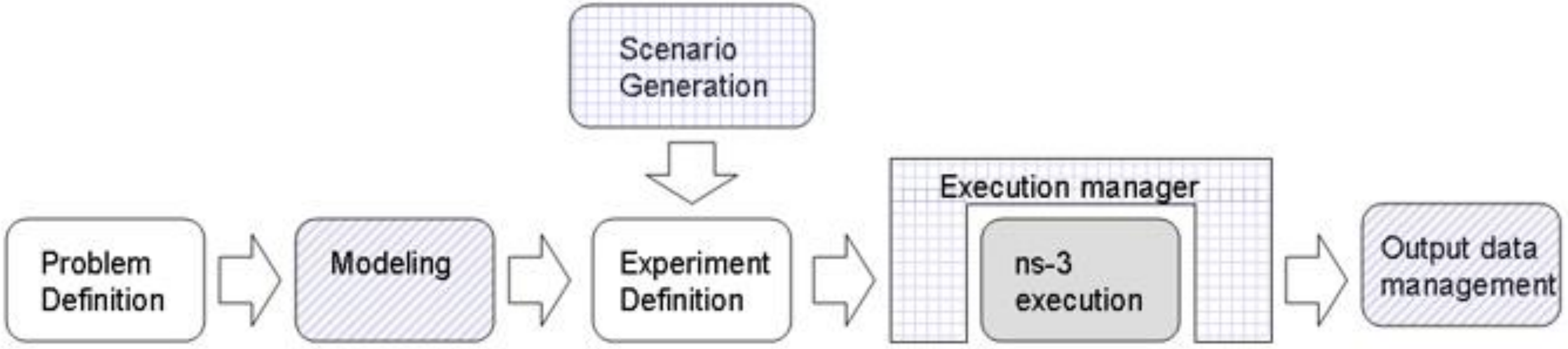
July 2014

Agenda

- ns-3 project overview
  - What is ns-3?
  - Why use ns-3?
  - Project organization
  - Relationship to ns-2
  - Future directions
- Getting started with ns-3

Discrete event network simulator

- Model of the evolution of a networked system through discrete events in time
- Used for experimentation and education



ns-3 simulation basics

- Simulation time advances in discrete jumps from event to event
- C++ functions schedule events to occur at specific simulation times
- A simulation scheduler orders the event execution
- Simulation::Run() gets it all started
- Simulation stops at specific time or when events end

Software overview

- ns-3 is written in C++, with bindings available for Python
  - simulation programs are C++ executables or Python programs
  - ~350,000 lines of C++ (estimate based on cloc source code analysis)
- ns-3 is a GNU GPLv2-licensed project
- ns-3 is mainly supported for Linux, OS X, and FreeBSD
  - Windows Visual Studio port available
- ns-3 is not backwards-compatible with ns-2

Software orientation

Key differences from other network simulators:

1) Command-line, Unix orientation

- vs. Integrated Development Environment (IDE)

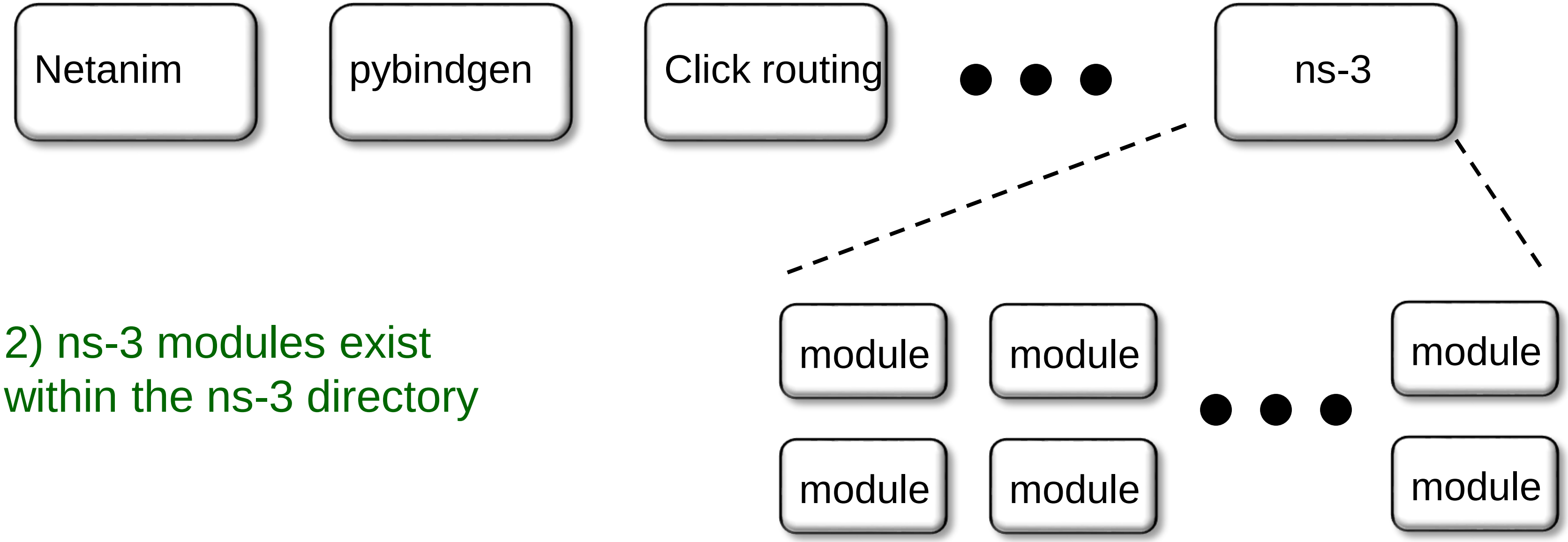
2) Simulations and models written directly in C++ and Python

- vs. a domain-specific simulation language

Software organization

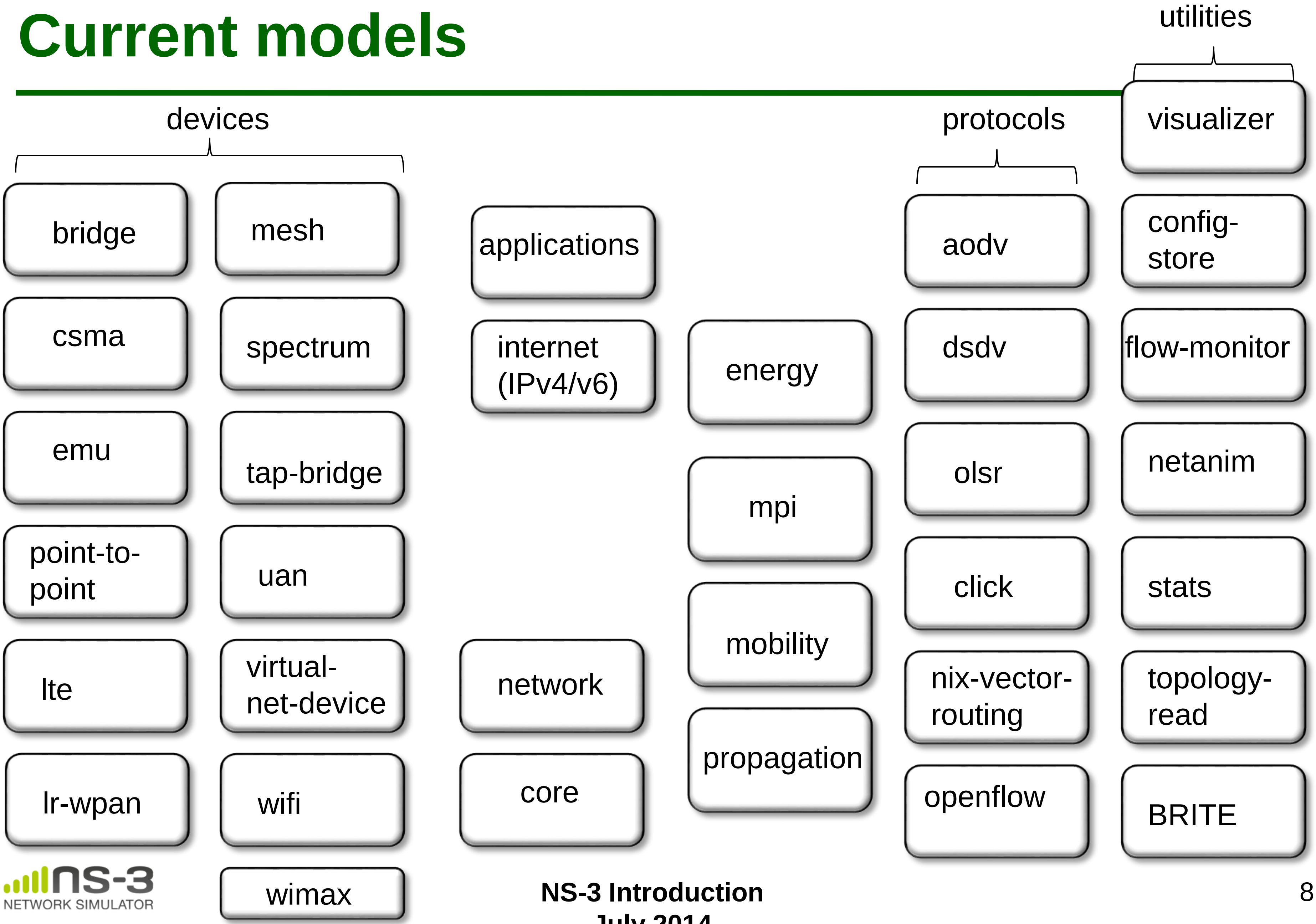
- Two levels of ns-3 software and libraries

1) Several supporting libraries, not system-installed, can be in parallel to ns-3



2) ns-3 modules exist within the ns-3 directory

Current models



devices

bridge
mesh
csma
spectrum
emu
tap-bridge
point-to-point
uan
lte
virtual-net-device
lr-wpan
wifi
wimax

applications
internet (IPv4/v6)
network
core

energy
mpi
mobility
propagation

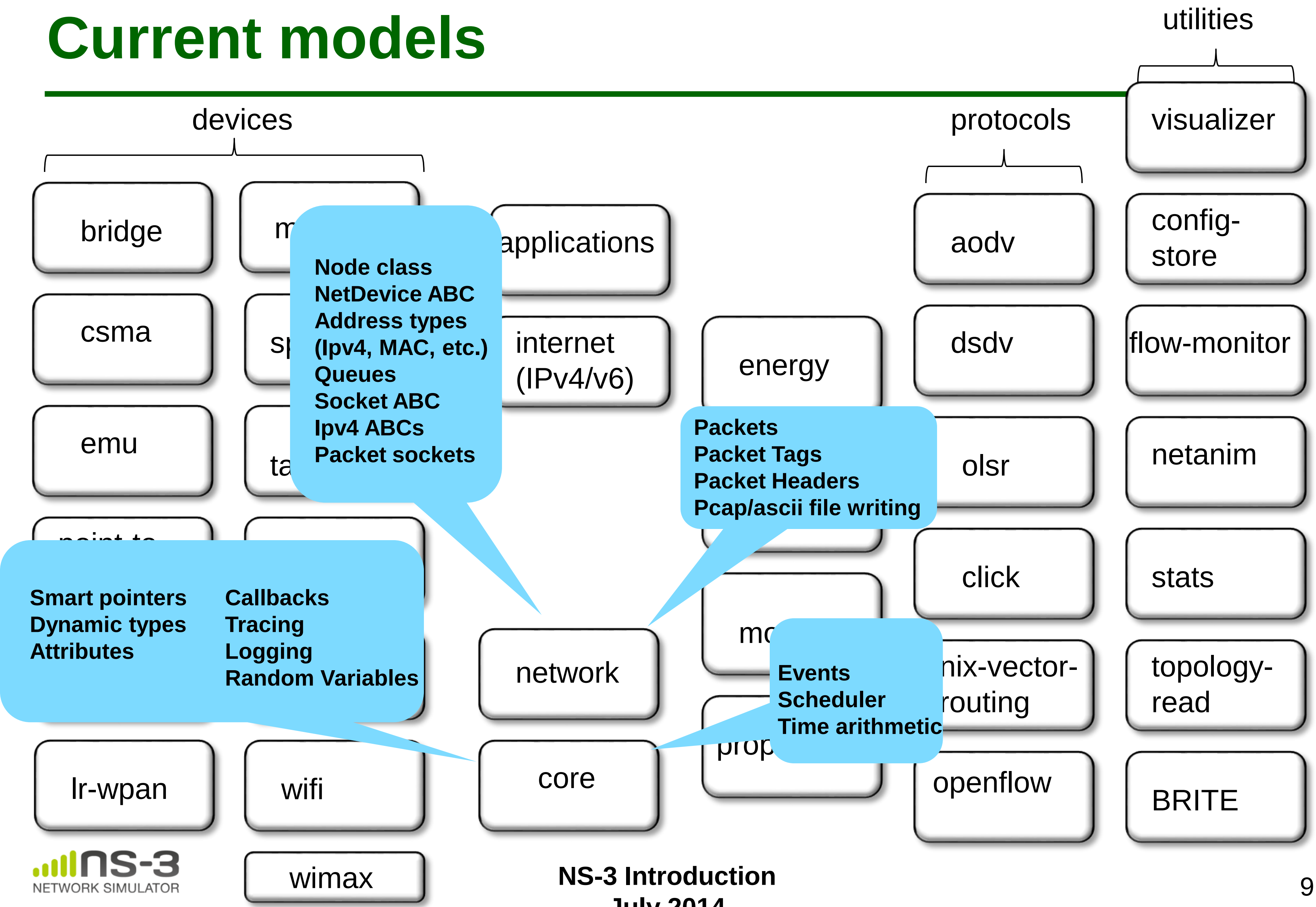
protocols

aodv
dsdv
olsr
click
nix-vector-routing
openflow

utilities

visualizer
config-store
flow-monitor
netanim
stats
topology-read
BRITE

Current models



devices

bridge

m

csma

sp

emu

ta

point-to

lr-wpan

wifi

wimax

applications

internet (IPv4/v6)

network

core

energy

mo

prop

protocols

aodv

dsdv

olsr

click

nix-vector-routing

openflow

utilities

visualizer

config-store

flow-monitor

netanim

stats

topology-read

BRITE

Node class
NetDevice ABC
Address types
(Ipv4, MAC, etc.)
Queues
Socket ABC
Ipv4 ABCs
Packet sockets

Packets
Packet Tags
Packet Headers
Pcap/ascii file writing

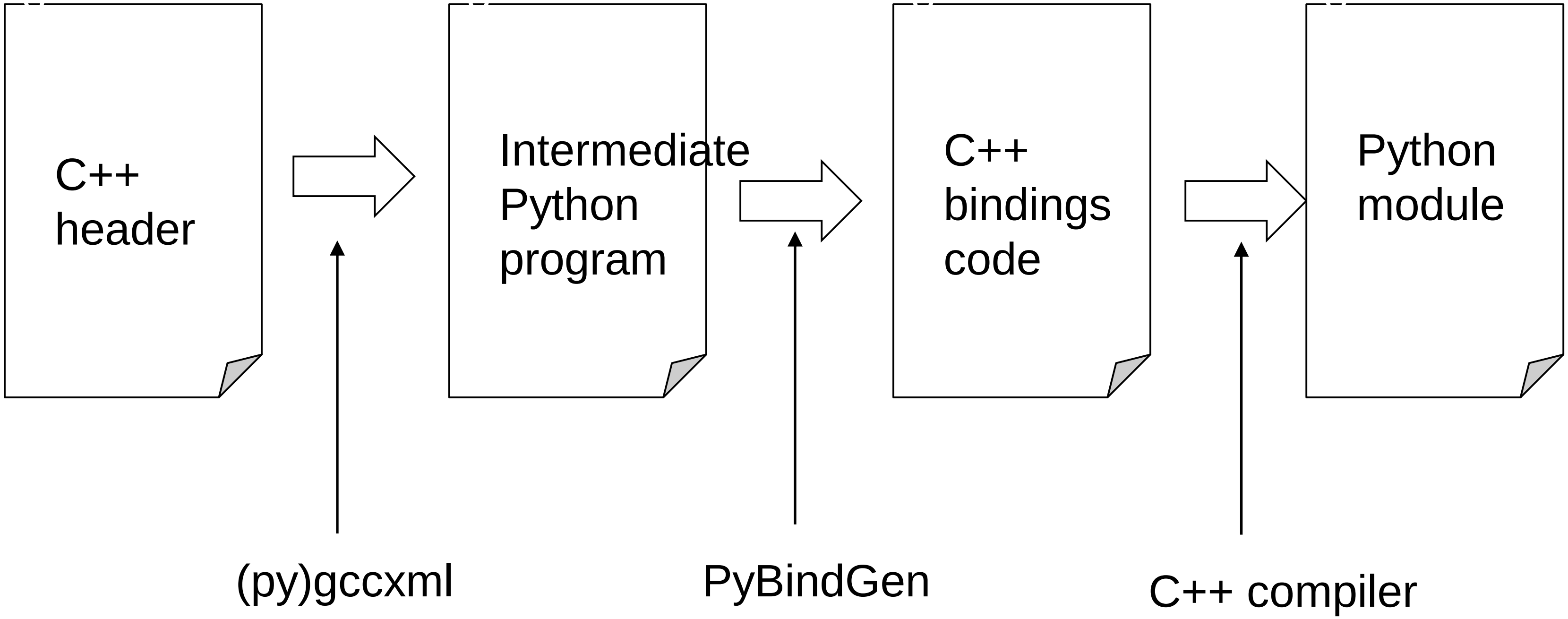
Smart pointers
Dynamic types
Attributes

Callbacks
Tracing
Logging
Random Variables

Events
Scheduler
Time arithmetic

Python bindings

- ns-3 uses a program called PyBindGen to generate Python bindings for all libraries



Agenda

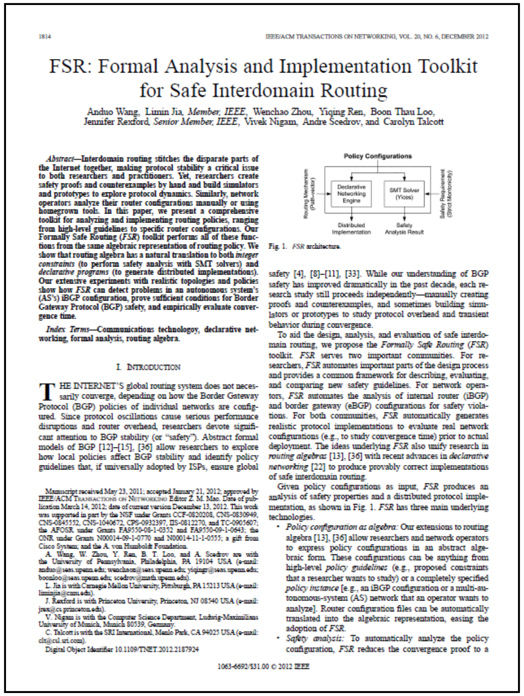
- ns-3 project overview
  - What is ns-3?
  - Why use ns-3?
  - Project organization
  - Relationship to ns-2
  - Future directions
- Getting started with ns-3

Why use ns-3?

- You want to study *network performance* or *protocol operation* in a *controllable* or *scalable* environment
- You are comfortable writing C++ or Python code, and combining ns-3 with other code
- You like the idea of working on an active open source project
- ns-3 has the models you are looking for
  - or you can provide/integrate what is lacking

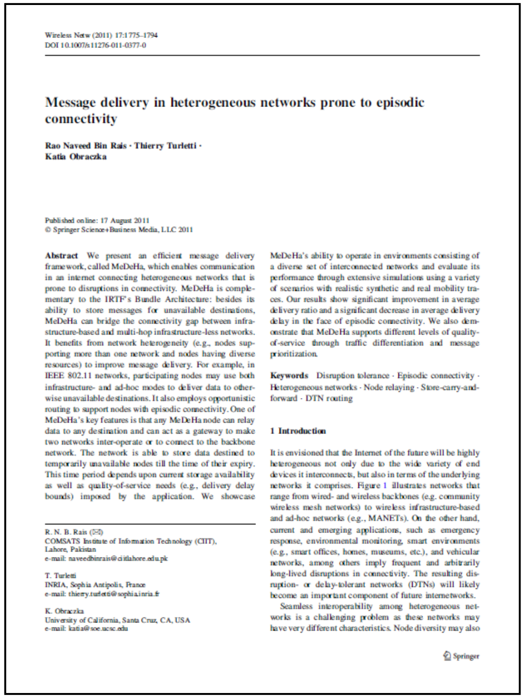
What have people done with ns-3?

- ~750 publications to date
  - search of 'ns-3 simulator' on IEEE and ACM digital libraries


1814 IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 20, NO. 6, DECEMBER 2012

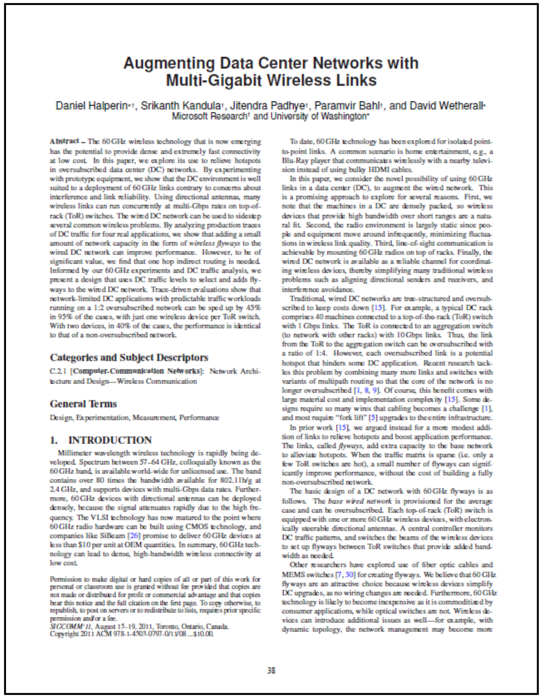
FSR: Formal Analysis and Implementation Toolkit for Safe Interdomain Routing

Anduo Wang, Limin Jia, Member, IEEE, Wenchao Zhou, Yiqing Ren, Boon Thau Loo, Jennifer Rexford, Senior Member, IEEE, Vivek Nigam, Andre Scedrov, and Carolyn Talcott


Wireless Netw (2011) 17:1775–1794
DOI 10.1007/s11276-011-0377-0

Message delivery in heterogeneous networks prone to episodic connectivity

Rao Naveed Bin Rais · Thierry Turletti · Katia Obraczka


Augmenting Data Center Networks with Multi-Gigabit Wireless Links

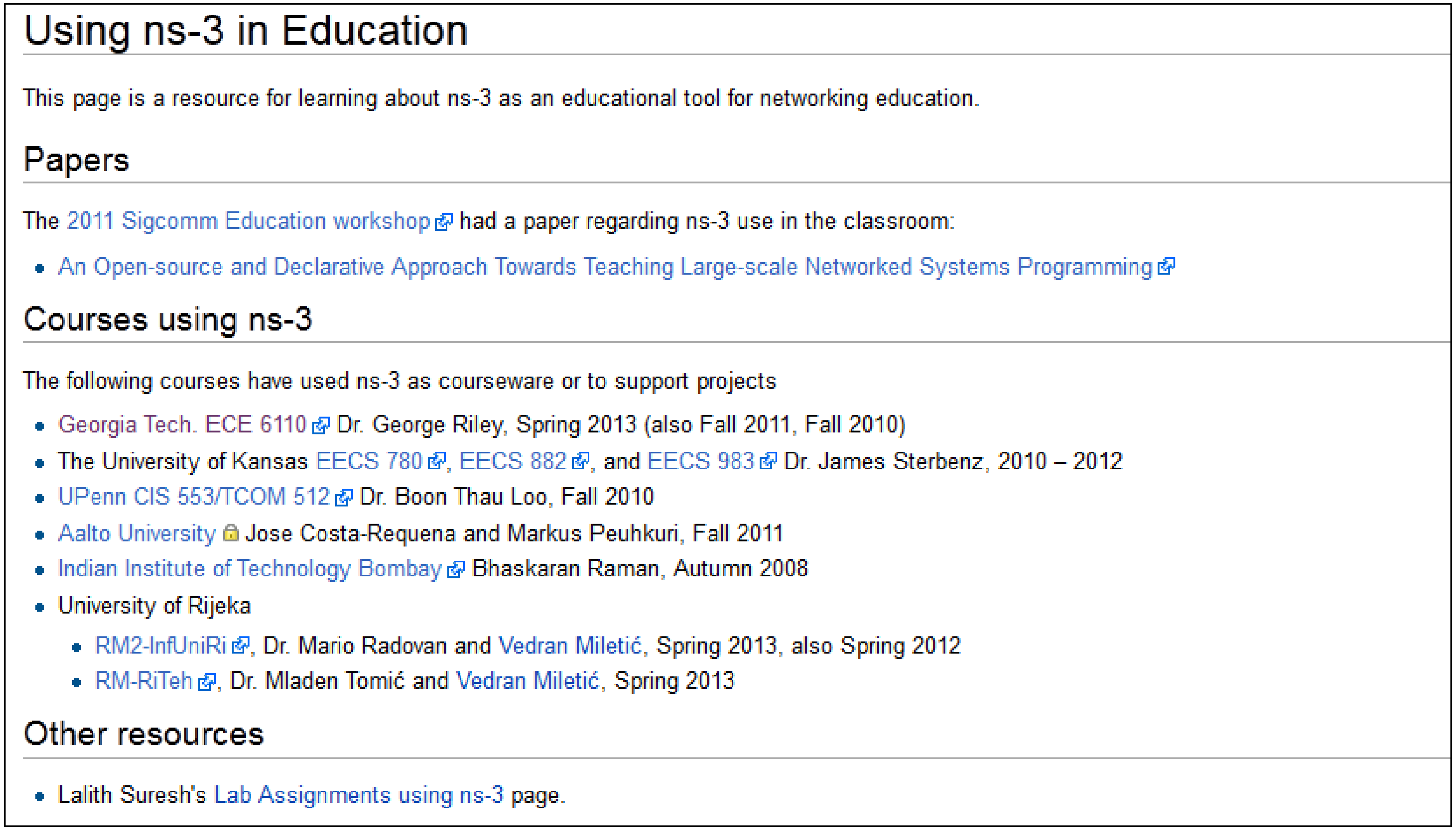
Daniel Halperin, Srikanth Kandula, Jitendra Padhye, Paramvir Bahl, and David Wetherall
Microsoft Research and University of Washington

Examples of recent publications

- L. Salameh et al., "HACK: Hierarchical ACKs for Efficient Wireless Medium Utilization," *Proceedings of 2014 Usenix Annual Technical Conference* (best paper award winner), June 2014.
- A. Azgin et al., "Mobility Study for Named Data Networking in Wireless Access Networks," *Proceedings of IEEE ICC 2014*, June 2014.
- Wong S.-H and Gary Chan, "Topology Optimization for Wireless Mesh with Directional Antennas," *Proceedings of IEEE ICC 2014*, June 2014.
- C. Gouveia et al., "Development and implementation of Portuguese smart distribution system," *Electric Power Systems Research, Elsevier*, vol. 116, June 2014.
- M. Alharthi et al., "An Acumen/NS-3 integration for modeling networked Cyber-Physical Systems," *2014 Biennial Symposium on Communications (QSBC)*, June 2014.
- L. Ciarletta et al., "Simulation and platform tools to develop safe flock of UAVs: a CPS application-driven research," *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, May 2014.

What have people done with ns-3?

- Educational use (from ns-3 wiki)



Using ns-3 in Education

This page is a resource for learning about ns-3 as an educational tool for networking education.

Papers

The 2011 Sigcomm Education workshop had a paper regarding ns-3 use in the classroom:

- An Open-source and Declarative Approach Towards Teaching Large-scale Networked Systems Programming

Courses using ns-3

The following courses have used ns-3 as courseware or to support projects

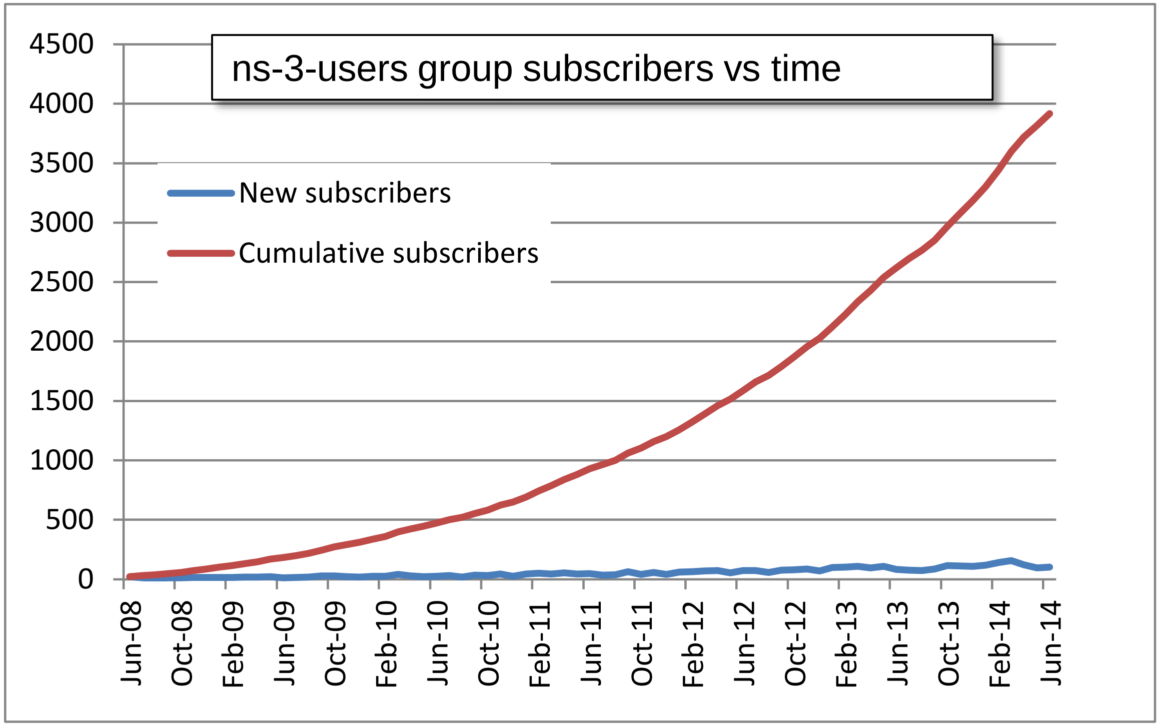
- Georgia Tech. ECE 6110 Dr. George Riley, Spring 2013 (also Fall 2011, Fall 2010)
- The University of Kansas EECS 780, EECS 882, and EECS 983 Dr. James Sterbenz, 2010 – 2012
- UPenn CIS 553/TCOM 512 Dr. Boon Thau Loo, Fall 2010
- Aalto University Jose Costa-Requena and Markus Peuhkuri, Fall 2011
- Indian Institute of Technology Bombay Bhaskaran Raman, Autumn 2008
- University of Rijeka
  - RM2-InfUniRi, Dr. Mario Radovan and Vedran Miletić, Spring 2013, also Spring 2012
  - RM-RiTeh, Dr. Mladen Tomić and Vedran Miletić, Spring 2013

Other resources

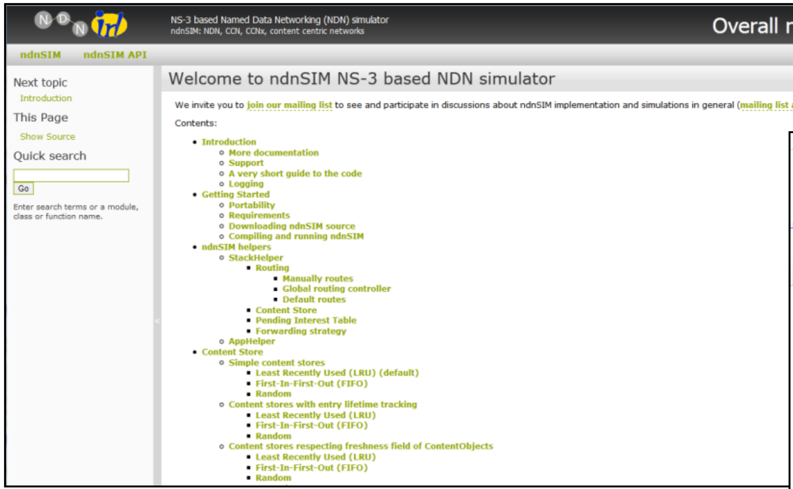
- Lalith Suresh's Lab Assignments using ns-3 page.

Statistics (July 2014)

- 3900 subscribers to ns-3-users
- 1440 subscribers to ns-developers
- ~ 15 maintainers
- ~ 150 authors/contributors



Contributed code and associated projects



NS-3 based Named Data Networking (NDN) simulator
ndnSIM: NDN, CCN, CCNx, content centric networks

Overall ndnSIM documentation

ndnSIM ndnSIM API next

Next topic
Introduction

This Page
Show Source

Quick search

Go

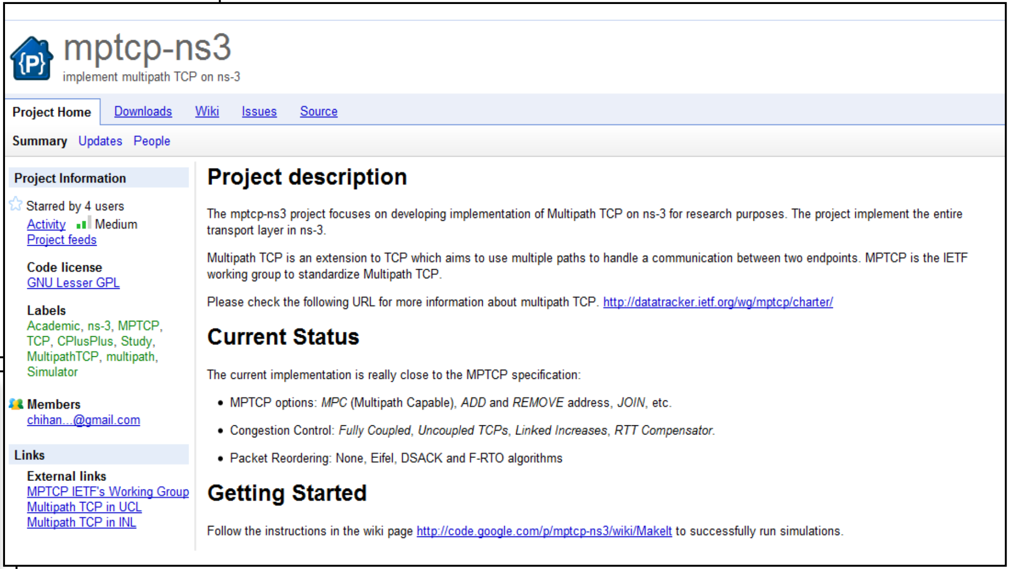
Enter search terms or a module, class or function name.

Welcome to ndnSIM NS-3 based NDN simulator

We invite you to join our mailing list to see and participate in discussions about ndnSIM implementation and simulations in general (mailing list archives).

Contents:

- Introduction
  - More documentation
  - Support
  - A very short guide to the code
  - Logging
- Getting Started
  - Portability
  - Requirements
  - Downloading ndnSIM source
  - Compiling and running ndnSIM
- ndnSIM helpers
  - StackHelper
    - Routing
      - Manually routes
      - Global routing controller
      - Default routes
    - Content Store
    - Pending Interest Table
    - Forwarding strategy
  - AppHelper
- Content Store
  - Simple content stores
    - Least Recently Used (LRU) (default)
    - First-In-First-Out (FIFO)
    - Random
  - Content stores with entry lifetime tracking
    - Least Recently Used (LRU)
    - First-In-First-Out (FIFO)
    - Random
  - Content stores respecting freshness field of ContentObjects
    - Least Recently Used (LRU)
    - First-In-First-Out (FIFO)
    - Random



mptcp-ns3

implement multipath TCP on ns-3

Project Information

Starred by 4 users
Activity Medium
Project feeds

Code license
GNU Lesser GPL

Labels
Academic, ns-3, MPTCP, TCP, CPlusPlus, Study, MultipathTCP, multipath, Simulator

Members
chihan...@gmail.com

Links

External links
MPTCP IETF's Working Group
Multipath TCP in UCL
Multipath TCP in INL

Project description

The mptcp-ns3 project focuses on developing implementation of Multipath TCP on ns-3 for research purposes. The project implement the entire transport layer in ns-3.

Multipath TCP is an extension to TCP which aims to use multiple paths to handle a communication between two endpoints. MPTCP is the IETF working group to standardize Multipath TCP.

Please check the following URL for more information about multipath TCP. http://datatracker.ietf.org/wg/mptcp/charter/

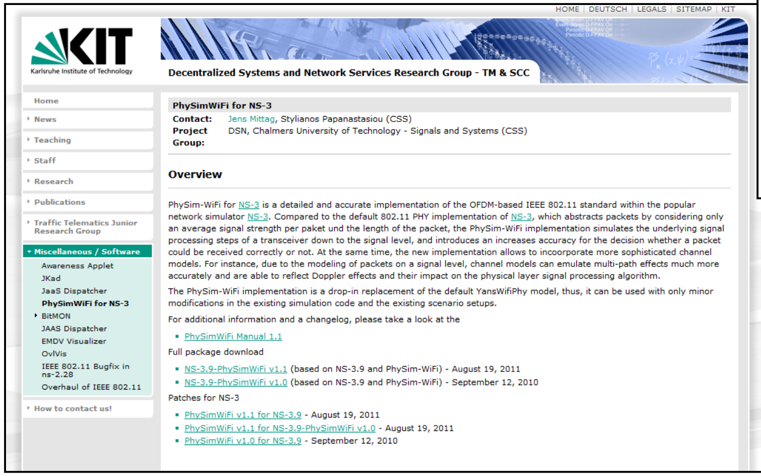
Current Status

The current implementation is really close to the MPTCP specification:

- MPTCP options: *MPC* (Multipath Capable), *ADD* and *REMOVE* address, *JOIN*, etc.
- Congestion Control: *Fully Coupled, Uncoupled TCPs, Linked Increases, RTT Compensator.*
- Packet Reordering: None, Eifel, DSACK and F-RTO algorithms

Getting Started

Follow the instructions in the wiki page http://code.google.com/p/mptcp-ns3/wiki/MakeIt to successfully run simulations.



HOME | DEUTSCH | LEGALS | SITEMAP | KIT

Karlsruhe Institute of Technology

Decentralized Systems and Network Services Research Group - TM & SCC

PhySimWiFi for NS-3

Contact:	Jens Mittag, Stylianos Papanastasiou (CSS)
Project Group:	DSN, Chalmers University of Technology - Signals and Systems (CSS)

Overview

PhySim-WiFi for NS-3 is a detailed and accurate implementation of the OFDM-based IEEE 802.11 standard within the popular network simulator NS-3. Compared to the default 802.11 PHY implementation of NS-3, which abstracts packets by considering only an average signal strength per paket und the length of the packet, the PhySim-WiFi implementation simulates the underlying signal processing steps of a transceiver down to the signal level, and introduces an increases accuracy for the decision whether a packet could be received correctly or not. At the same time, the new implementation allows to incoorporate more sophisticated channel models. For instance, due to the modeling of packets on a signal level, channel models can emulate multi-path effects much more accurately and are able to reflect Doppler effects and their impact on the physical layer signal processing algorithm.

The PhySim-WiFi implementation is a drop-in replacement of the default YansWifiPhy model, thus, it can be used with only minor modifications in the existing simulation code and the existing scenario setups.

For additional information and a changelog, please take a look at the

- PhySimWiFi Manual 1.1

Full package download

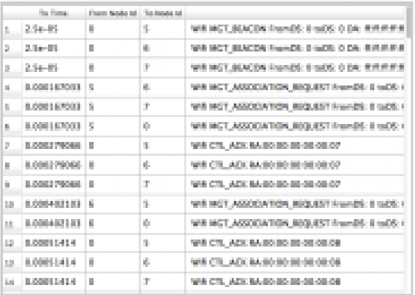
- NS-3.9-PhySimWiFi v1.1 (based on NS-3.9 and PhySim-WiFi) - August 19, 2011
- NS-3.9-PhySimWiFi v1.0 (based on NS-3.9 and PhySim-WiFi) - September 12, 2010

Patches for NS-3

- PhySimWiFi v1.1 for NS-3.9 - August 19, 2011
- PhySimWiFi v1.1 for NS-3.9-PhySimWiFi v1.0 - August 19, 2011
- PhySimWiFi v1.0 for NS-3.9 - September 12, 2010

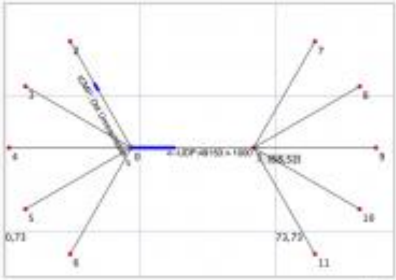
NetAnim

- "NetAnim" by George Riley and John Abraham
  - see the 'ns3share' channel on YouTube

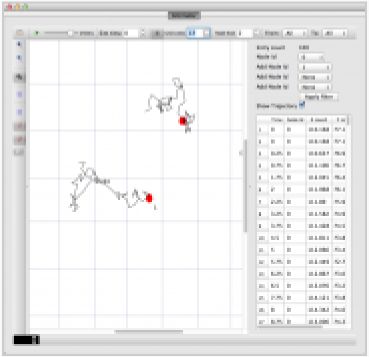


	Tx Time	From Node Id	To Node Id	
1	2.5e-05	8	5	Wifi MGT_BEACON FromDS: 0 toDS: 0 DA: ff:ff:ff:ff
2	2.5e-05	8	6	Wifi MGT_BEACON FromDS: 0 toDS: 0 DA: ff:ff:ff:ff
3	2.5e-05	8	7	Wifi MGT_BEACON FromDS: 0 toDS: 0 DA: ff:ff:ff:ff
4	0.000167003	5	6	Wifi MGT_ASSOCIATION_REQUEST FromDS: 0 toDS:
5	0.000167003	5	7	Wifi MGT_ASSOCIATION_REQUEST FromDS: 0 toDS:
6	0.000167003	5	0	Wifi MGT_ASSOCIATION_REQUEST FromDS: 0 toDS:
7	0.000179066	8	5	Wifi CTL_ACK RA:00:00:00:00:00:07
8	0.000179066	8	6	Wifi CTL_ACK RA:00:00:00:00:00:07
9	0.000179066	8	7	Wifi CTL_ACK RA:00:00:00:00:00:07
10	0.000402183	6	5	Wifi MGT_ASSOCIATION_REQUEST FromDS: 0 toDS:
11	0.000402183	6	0	Wifi MGT_ASSOCIATION_REQUEST FromDS: 0 toDS:
12	0.00051414	8	5	Wifi CTL_ACK RA:00:00:00:00:00:08
13	0.00051414	8	6	Wifi CTL_ACK RA:00:00:00:00:00:08
14	0.00051414	8	7	Wifi CTL_ACK RA:00:00:00:00:00:08

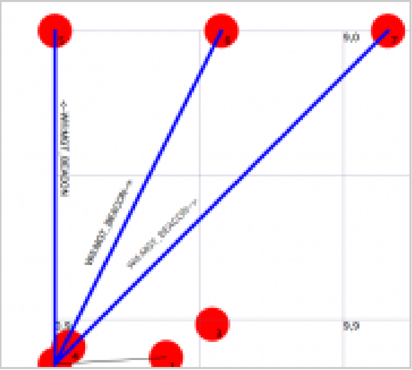
Packet Statistics



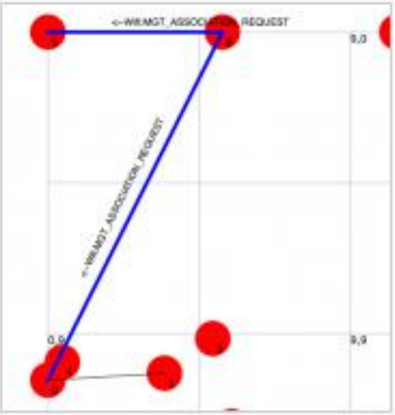
Dumbbell



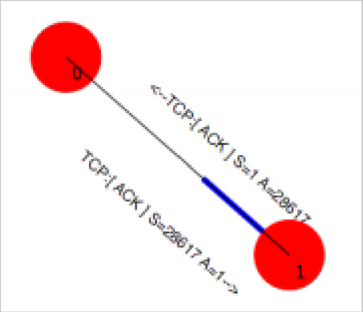
Node Trajectory



Wifi Beacon


<--Wifi MGT_ASSOCIATION_REQUEST
<--Wifi MGT_ASSOCIATION_REQUEST
9,0
0,9
9,9

Wifi Assoc



TCP flags

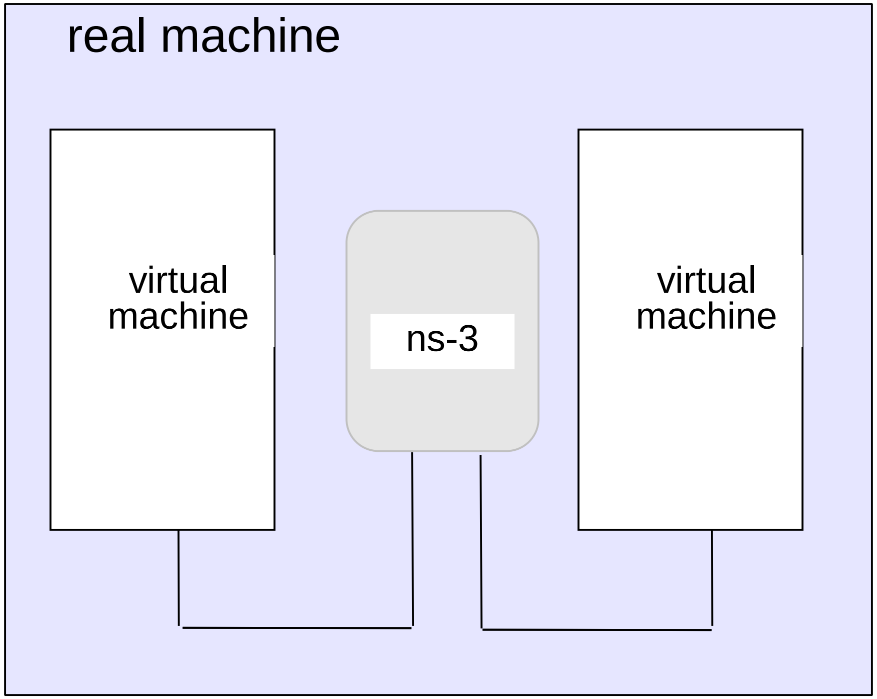
Emulation support

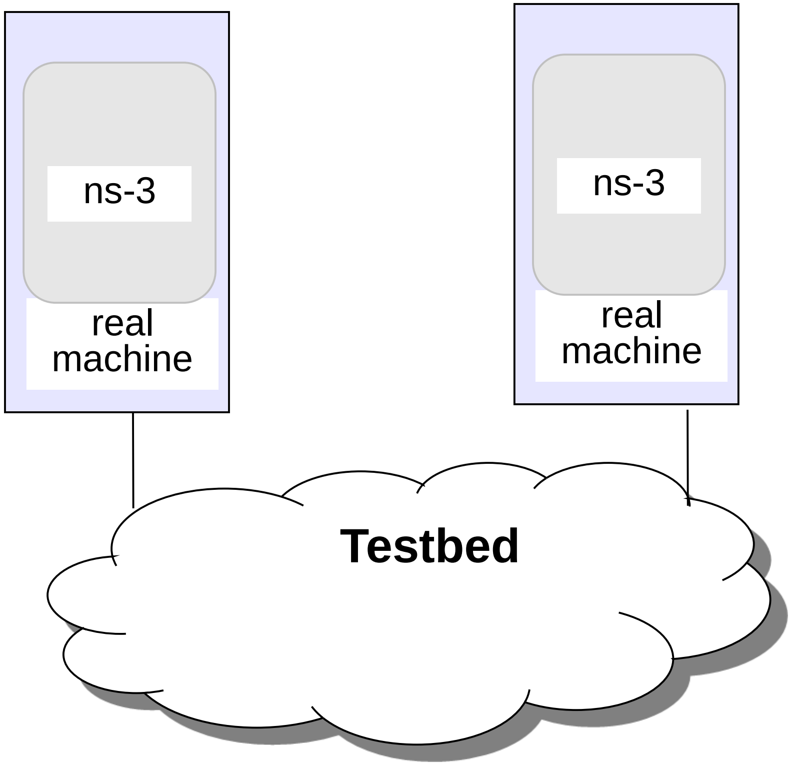
- Support moving between simulation and testbeds or live systems
- A real-time scheduler, and support for two modes of emulation
- Linux is only operating system supported
- Must run simulator in real time

  -
    ```
    GlobalValue::Bind (“SimulatorImplementationType”,
    StringValue (“ns3::RealTimeSimulatorImpl”));
    ```
- Must enable checksum calculations across models

  -
    ```
    GlobalValue::Bind (“ChecksumEnabled”, BooleanValue
    (true));
    ```
- Must sometimes run as root

ns-3 emulation modes





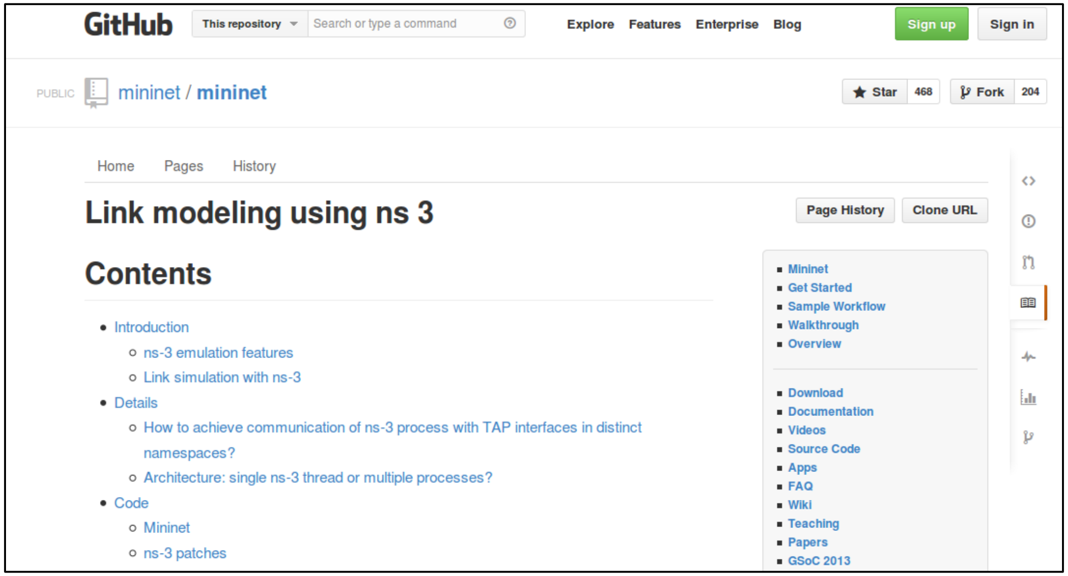
1) ns-3 interconnects real or virtual machines

2) testbeds interconnect ns-3 stacks

Various hybrids of the above are possible

Example use case: mininet

- Mininet is popular in the Software-Defined Networking (SDN) community
- Mininet uses "TapBridge" integration
- https://github.com/mininet/mininet/wiki/Link-modeling-using-ns-3



Link modeling using ns 3

Contents

- Introduction
  - ns-3 emulation features
  - Link simulation with ns-3
- Details
  - How to achieve communication of ns-3 process with TAP interfaces in distinct namespaces?
  - Architecture: single ns-3 thread or multiple processes?
- Code
  - Mininet
  - ns-3 patches

- Mininet
- Get Started
- Sample Workflow
- Walkthrough
- Overview

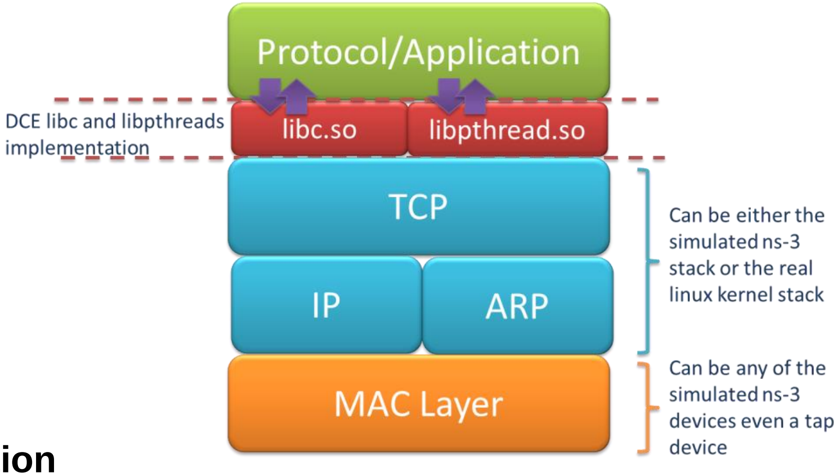
- Download
- Documentation
- Videos
- Source Code
- Apps
- FAQ
- Wiki
- Teaching
- Papers
- GSoC 2013

Direct Code Execution

- Lightweight virtualization of kernel and application processes, interconnected by simulated networks
- Benefits:
  - Implementation realism in controlled topologies or wireless environments
  - Model availability
  - Debugging a whole network within a single process
- Limitations:
  - Not as scalable as pure simulation
  - Tracing more limited
  - Configuration different

Direct Code Execution implementation

- DCE/ns-3 framework requires the virtualization of a series of services
  - Multiple isolated instances of the same protocol on the same machine
- System calls are captured and treated by DCE
- Network stack protocols calls are captured and redirected
- To perform its work DCE re-implement the Linux program loader and parts of *libc* and *libpthread*


Protocol/Application
DCE libc and libpthreads implementation
libc.so
libpthread.so
TCP
IP
ARP
MAC Layer
Can be either the simulated ns-3 stack or the real linux kernel stack
Can be any of the simulated ns-3 devices even a tap device

DCE modes

- DCE modes in context of possible approaches

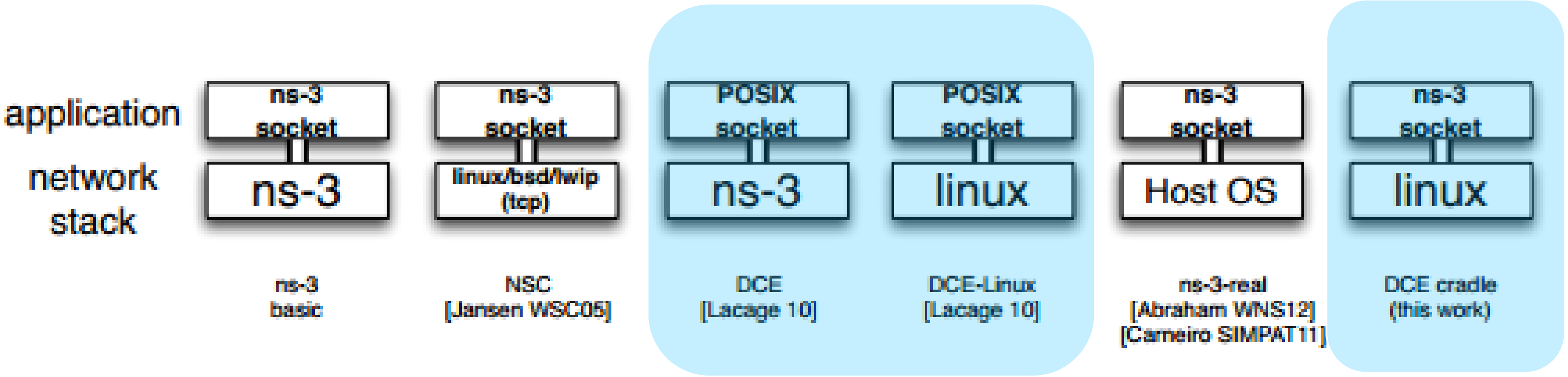


Figure 1: Current possible combinations of network stacks and applications.

Figure source: DCE Cradle: Simulate Network Protocols with Real Stacks for Better Realism, Tazaki et al, WNS3 2013.

Agenda

- ns-3 project overview
  - What is ns-3?
  - Why use ns-3?
  - Project organization
  - Relationship to ns-2
  - Future directions
- Getting started with ns-3

ns-3 project goals

Develop an extensible simulation environment for networking research

1) a tool aligned with the experimentation needs of modern networking research

2) a tool that elevates the technical rigor of network simulation practice

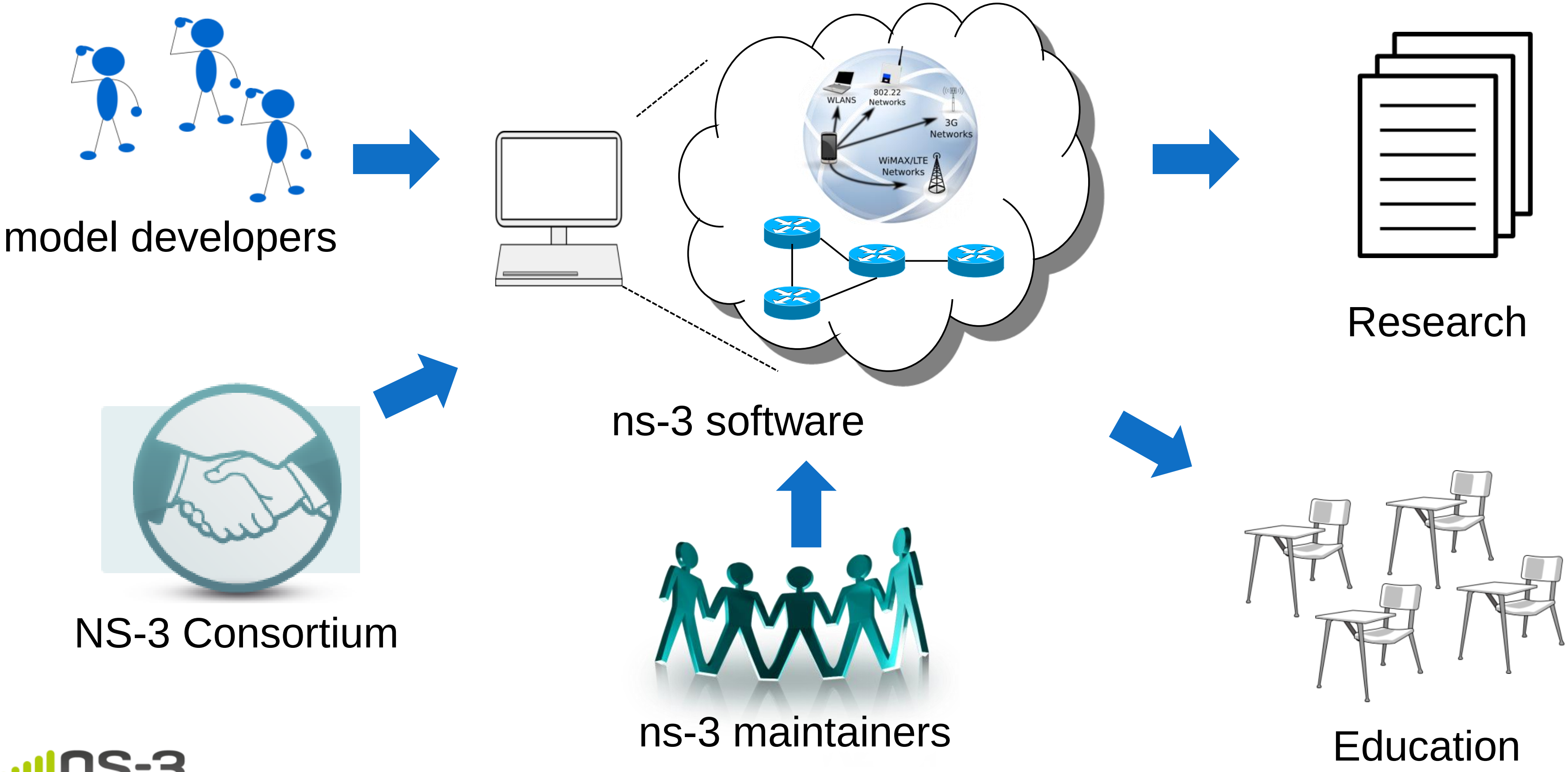
3) an open-source project that encourages community contribution, peer review, and long-term maintenance and validation of the software

How the project operates

- Project provides three annual software releases
- Users interact on mailing lists and using Bugzilla bug tracker
- Code may be proposed for merge
  - Code reviews occur on a Google site
- Maintainers (one for each module) fix or delegate bugs, participate in reviews
- Project has been conducting annual workshop and developer meeting around SIMUTools through 2013
  - ns-3 Annual Meeting in Atlanta, May 2014
- Google Summer of Code (March-August) five of the past six summers

ns-3: An Open Source Network Simulator

- ns-3 is a ***discrete-event network simulator*** targeted for ***research and educational use***



model developers

NS-3 Consortium

ns-3 software

ns-3 maintainers

Research

Education

Goals of the NS-3 Consortium

- The NS-3 Consortium is a collection of organizations cooperating to support and develop the ns-3 software.
- It operates in support of the open source project
  - by providing a point of contact between industrial members and ns-3 developers,
  - by sponsoring events in support of ns-3 such as users' days and workshops,
  - by guaranteeing maintenance support for ns-3's core, and
  - by supporting administrative activities necessary to conduct a large open source project.

Acknowledgment of support

NSF National Science Foundation
WHERE DISCOVERIES BEGIN

NAVAL RESEARCH LABORATORY
WASHINGTON, DC

Inria informatics mathematics

PLANETE

Google SUMMER OF CODE 2012

UNIVERSITY OF WASHINGTON 1861
LVX SIT

Georgia Tech
SEAL OF THE GEORGIA INSTITUTE OF TECHNOLOGY 1885
PROGRESS AND SERVICE

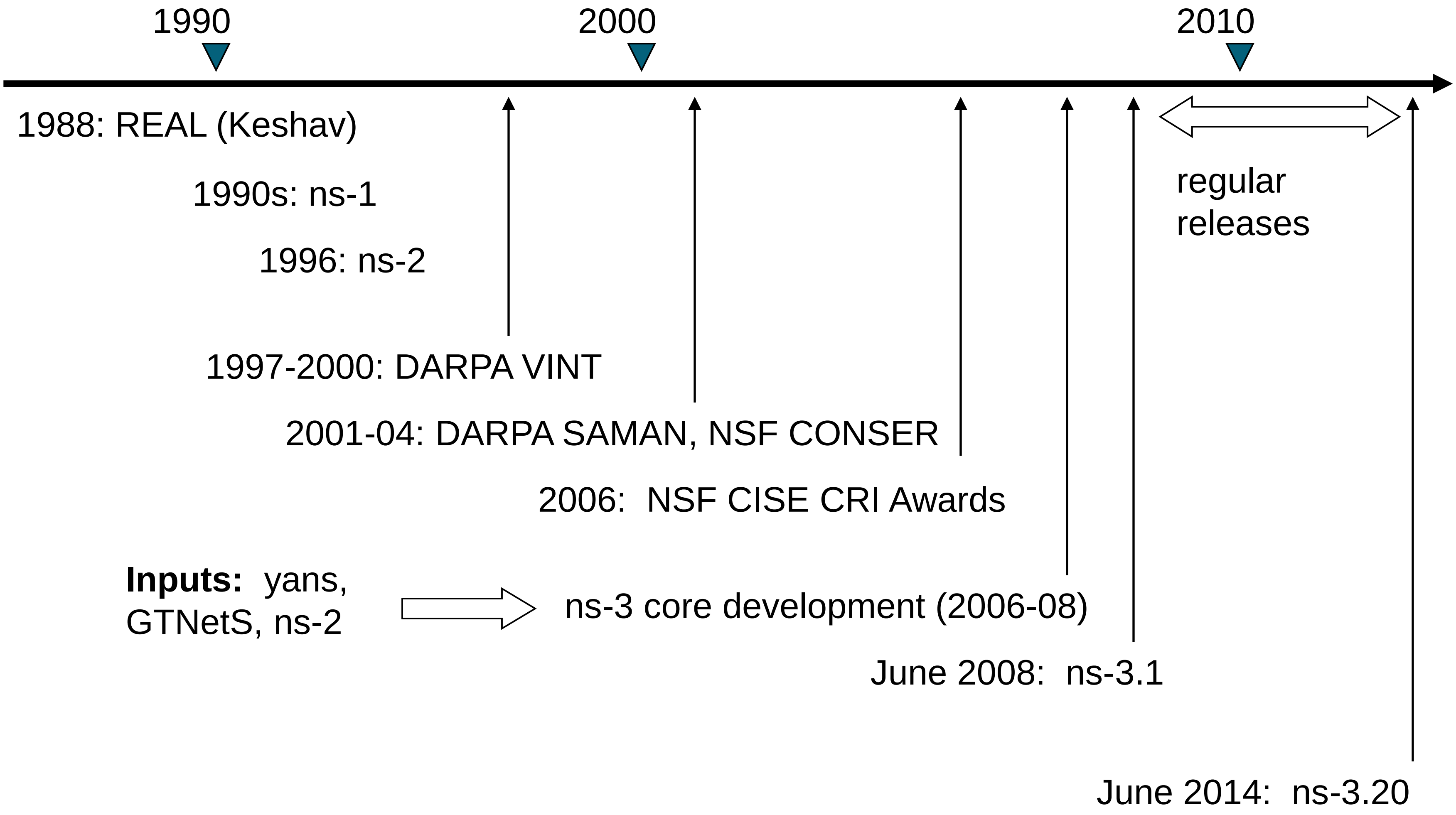
INESCPORTO®
LABORATÓRIO ASSOCIADO

ISI
Information Sciences Institute

Agenda

- ns-3 project overview
  - What is ns-3?
  - Why use ns-3?
  - Project organization
  - Relationship to ns-2
  - Future directions
- Getting started with ns-3

ns timeline



1990

2000

2010

1988: REAL (Keshav)

1990s: ns-1

1996: ns-2

1997-2000: DARPA VINT

2001-04: DARPA SAMAN, NSF CONSER

2006: NSF CISE CRI Awards

Inputs: yans, GTNetS, ns-2

ns-3 core development (2006-08)

June 2008: ns-3.1

regular releases

June 2014: ns-3.20

Relationship to ns-2

ns-3 is a new simulator, without backward compatibility

Similarities to ns-2:

- C++ software core
- GNU GPLv2 licensing
- ported ns-2 models: random variables, error models, OLSR, Calendar Queue scheduler

Differences:

- Python scripting (or C++ programs) replaces OTcl
- most of the core rewritten
- new animators, configuration tools, etc. are in work
- ns-2 is no longer actively maintained/supported

Agenda

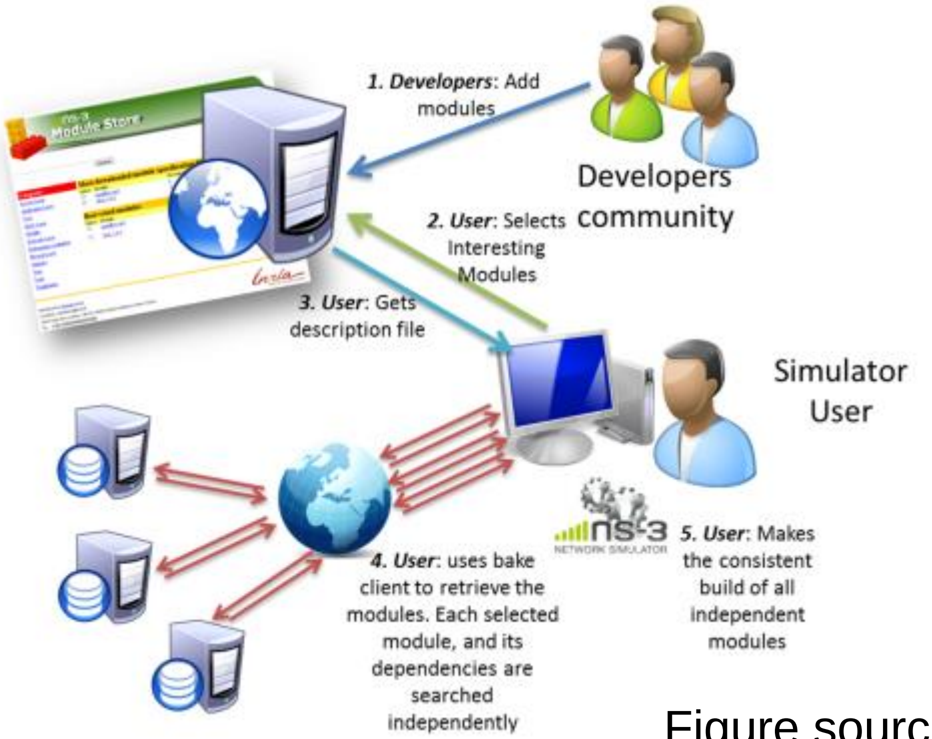
- ns-3 project overview
  - What is ns-3?
  - Why use ns-3?
  - Project organization
  - Relationship to ns-2
  - Future directions
- Getting started with ns-3

Development Priorities

- Software modularity and long-term maintenance
- Improved integration of direct code execution
- Improved integration with container-based and testbed-based experiment infrastructures
- Simulation-based experiment management
- Usability

Modularity

- Open source project maintains a (more stable) core
- Models migrate to a more federated development process


ns-3 Module Store
1. Developers: Add modules
Developers community
2. User: Selects Interesting Modules
3. User: Gets description file
Simulator User
4. User: uses bake client to retrieve the modules. Each selected module, and its dependencies are searched independently
ns-3 NETWORK SIMULATOR
5. User: Makes the consistent build of all independent modules

"bake" tool (Lacage and Camara)

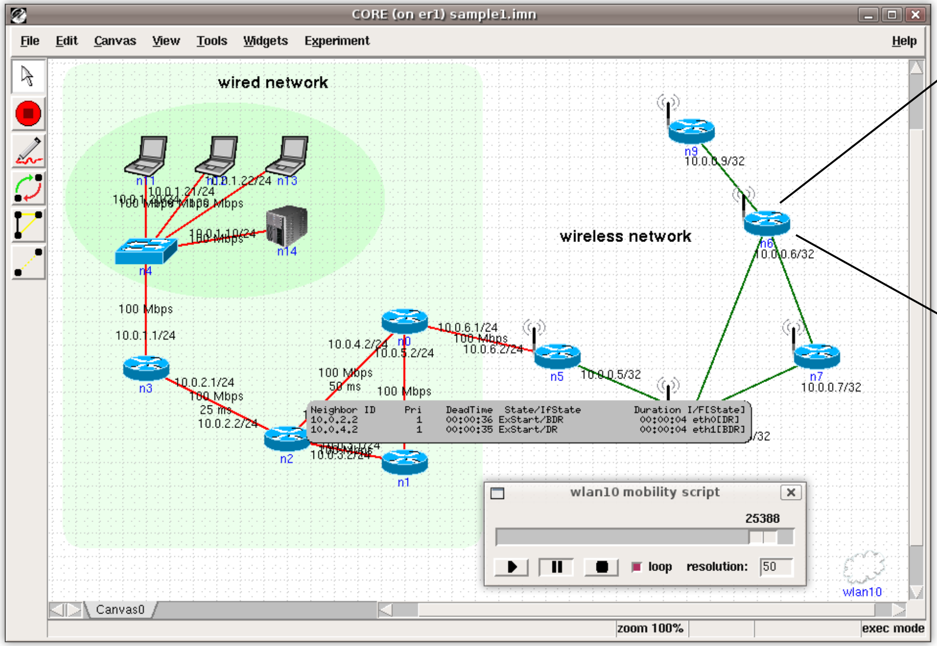
Components:

- build client
- "module store" server
- module metadata

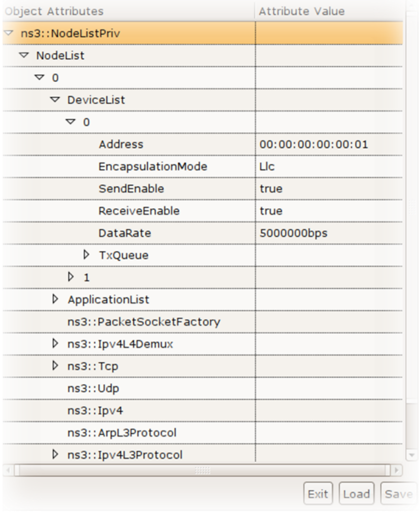
Figure source: Daniel Camara

Container-based Integration

- Common Open Research Emulator (CORE)
  - http://pf.itd.nrl.navy.mil
- Python-based framework using ns-3 Python bindings, distributed computing library, and ns-3 TapBridge framework



```
Neighbor ID     Pri    DeadTime  State/IfState          Duration I/F[State]
10.0.2.2          1    00:00:36 ExStart/BDR             00:00:04 eth0[DR]
10.0.4.2          1    00:00:35 ExStart/DR              00:00:04 eth1[BDR]
```



Object Attributes	Attribute Value
▽ ns3::NodeListPriv	
▽ NodeList	
▽ 0	
▽ DeviceList	
▽ 0	
Address	00:00:00:00:00:01
EncapsulationMode	Llc
SendEnable	true
ReceiveEnable	true
DataRate	5000000bps
▷ TxQueue	
▷ 1	
▷ ApplicationList	
ns3::PacketSocketFactory	
▷ ns3::Ipv4L4Demux	
▷ ns3::Tcp	
ns3::Udp	
ns3::Ipv4	
ns3::ArpL3Protocol	
▷ ns3::Ipv4L3Protocol	

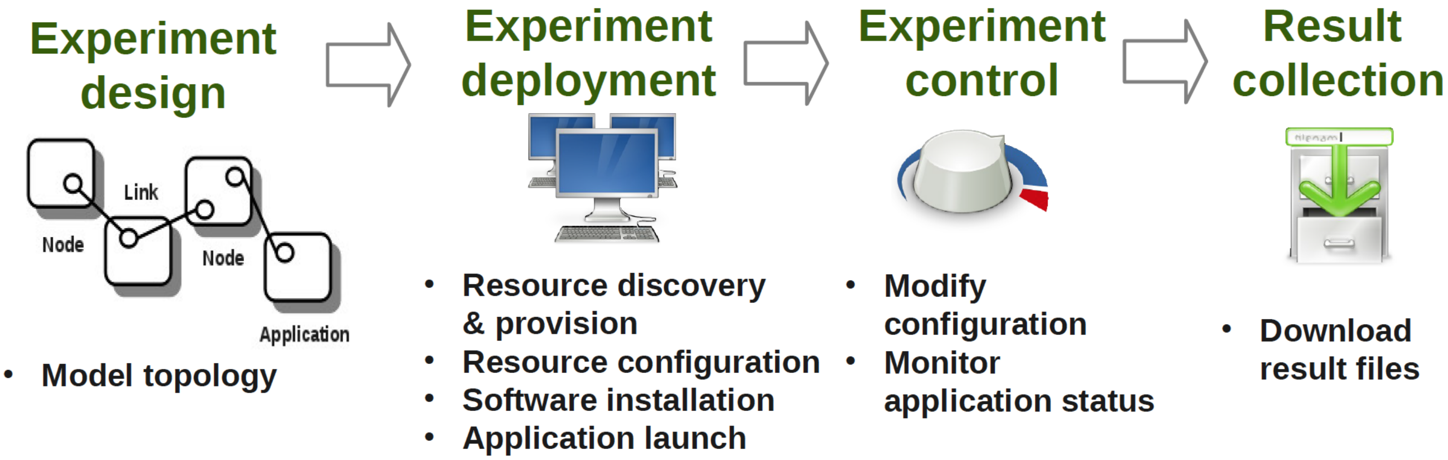
Figure source:
Jeff Ahrenholz

General issues with hybrid environments

- Ease of use
  - Configuration management and coherence
  - Information coordination (two sets of state)
    - e.g. IP/MAC address coordination
  - Output data exists in two domains
  - Debugging
- Error-free operation (avoidance of misuse)
  - Synchronization, information sharing, exception handling
    - Checkpoints for execution bring-up
    - Inoperative commands within an execution domain
    - Deal with run-time errors
  - Soft performance degradation (CPU) and time discontinuities

Network Experiment Management Framework (NEPI)

- Network experiment management framework to automate experiment life-cycle
- Allows scenarios involving heterogeneous resources (ns-3, PlanetLab, netns, …)
- Wiki: http://nepi.inria.fr



Experiment design

- Model topology

Experiment deployment

- Resource discovery & provision
- Resource configuration
- Software installation
- Application launch

Experiment control

- Modify configuration
- Monitor application status

Result collection

- Download result files

Figure source: Alina Quereilhac, INRIA

SAFE: Simulation Automation Framework

- Data collection, transient analysis, management of independent replications, graphical configuration and visualization
- In ns-2 realm, similar to projects like ANSWER, ns2measure, and Akaroa2

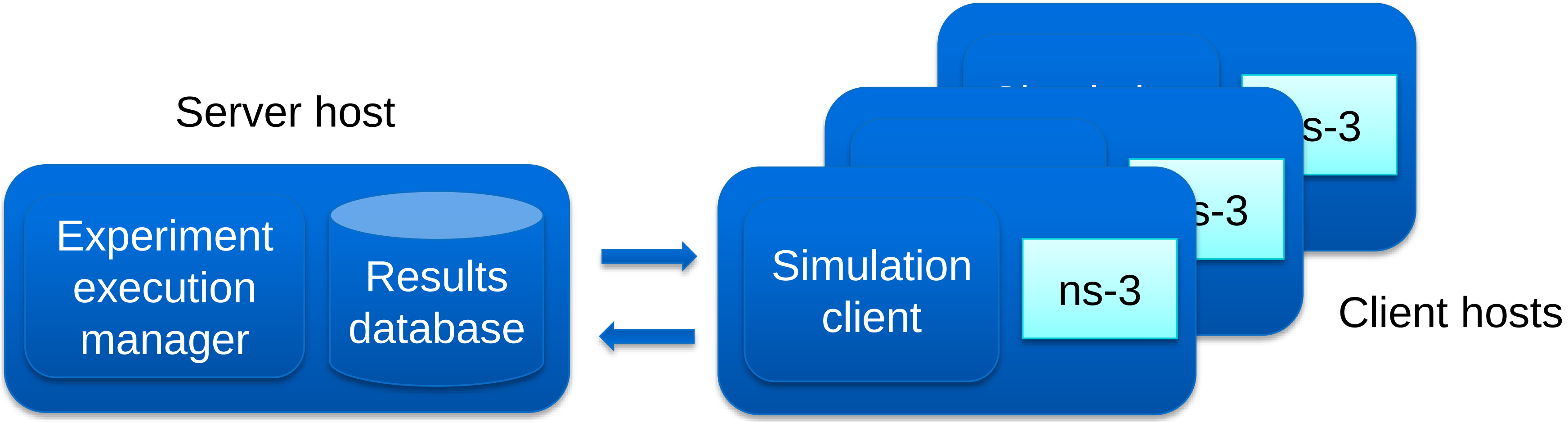
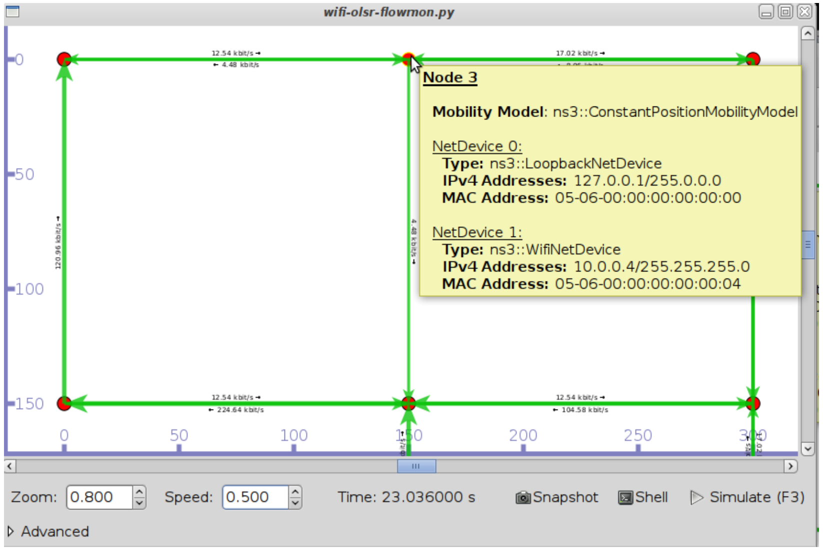


Figure source: Felipe Perrone

Usability

- Animation and visualization



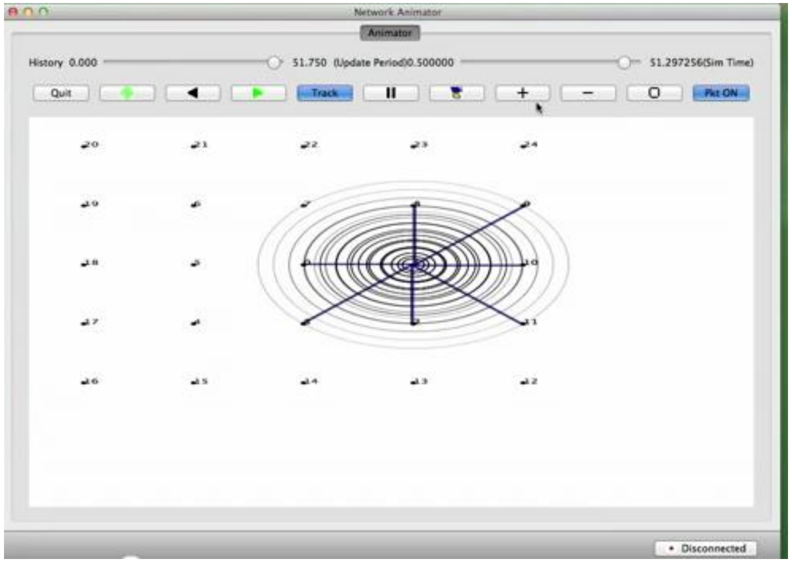
Node 3

Mobility Model: ns3::ConstantPositionMobilityModel

NetDevice 0:
Type: ns3::LoopbackNetDevice
IPv4 Addresses: 127.0.0.1/255.0.0.0
MAC Address: 05-06-00:00:00:00:00:00

NetDevice 1:
Type: ns3::WifiNetDevice
IPv4 Addresses: 10.0.0.4/255.255.255.0
MAC Address: 05-06-00:00:00:00:00:04

Zoom: 0.800 Speed: 0.500 Time: 23.036000 s



History 0.000 51.750 (Update Period)0.500000 51.297256(Sim Time)

PyVis (Carneiro)

NetAnim (Riley and Abraham)

- Linkage to external tools (topology, mobility, statistics)
- Improved helper APIs

Agenda

- ns-3 project overview
  - What is ns-3?
  - Why use ns-3?
  - Project organization
  - Relationship to ns-2
  - Future directions
- Getting started with ns-3

Getting started with ns-3

- Finding what you need
- Contributing to the project

Resources

Web site:

http://www.nsnam.org

Mailing lists:

https://groups.google.com/forum/#!forum/ns-3-users

http://mailman.isi.edu/mailman/listinfo/ns-developers

Wiki:

http://www.nsnam.org/wiki/

Tutorial:

http://www.nsnam.org/docs/tutorial/tutorial.html

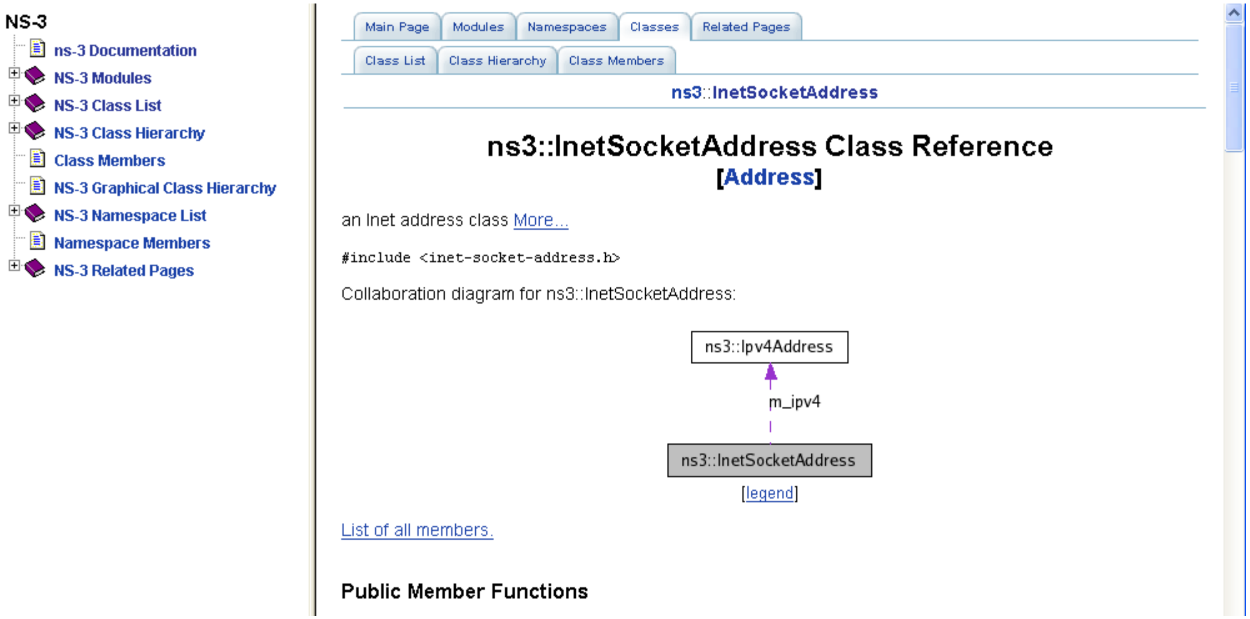
IRC: #ns-3 at freenode.net

Suggested steps

- Work through the ns-3 tutorial
- Browse the source code and other project documentation
  - manual, model library, Doxygen, wiki
  - ns-3 Consortium tutorials (March 2013)
    - http://www.nsnam.org/consortium/activities/annual-meeting-march-2013/
- Ask on ns-3-users mailing list if you still have questions
  - We try to answer most questions

APIs

- Most of the ns-3 API is documented with Doxygen
  - http://www.stack.nl/~dimitri/doxygen/



ns3::InetSocketAddress Class Reference
[Address]

an Inet address class More...

```
#include <inet-socket-address.h>
```

Collaboration diagram for ns3::InetSocketAddress:

List of all members.

Public Member Functions

Reading existing code

- Much insight can be gained from reading ns-3 examples and tests, and running them yourselves
- Many core features of ns-3 are only demonstrated in the core test suite (src/core/test)
- Stepping through code with a debugger is informative
  - callbacks and templates make it more challenging than usual

FAQs

- Does ns-3 have a Windows version?
  - Yes, for Visual Studio 2012
  - http://www.nsnam.org/wiki/Ns-3_on_Visual_Studio_2012
- Does ns-3 support Eclipse or other IDEs?
  - Instructions have been contributed by users
  - http://www.nsnam.org/wiki/HOWTO_configure_Eclipse_with_ns-3
- Is ns-3 provided in Linux or OS X package systems (e.g. Debian packages)?
  - No

Contributing

- Any amount of help is appreciated!
  - Reporting stale documentation to webmaster@nsnam.org
  - Contributing small patches
  - Writing new documentation
  - Reporting bugs
  - Fixing bugs
  - Reviewing code of others
  - Contributing new code
  - Becoming a maintainer

New project ideas

- Visit the wiki under "Project Ideas" tab
  - http://www.nsnam.org/wiki/Project_Ideas
- Students, consider to apply for Google Summer of Code 2015
  - A 10-week summer job that mentors a student project on ns-3
  - Students apply in March 2015 timeframe
  - http://www.nsnam.org/wiki/GSOC2014Projects

Questions?