P2p

```
// Client - Server using point to point protocol

#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/applications-module.h"
#include "ns3/netanim-module.h"



using namespace ns3;

int main (){

  LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
  LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);

  std::string animFile="first.xml";

  NodeContainer nodes;
  nodes.Create (2);

  PointToPointHelper pointToPoint;
  pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
  pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));

  NetDeviceContainer devices;
  devices = pointToPoint.Install (nodes);

  InternetStackHelper stack;
  stack.Install (nodes);

  Ipv4AddressHelper address;
  address.SetBase ("10.1.1.0", "255.255.255.0");

  Ipv4InterfaceContainer interfaces = address.Assign (devices);

  UdpEchoServerHelper echoServer (9);

  ApplicationContainer serverApps = echoServer.Install (nodes.Get (1));
  serverApps.Start (Seconds (1.0));
  serverApps.Stop (Seconds (10.0));

  UdpEchoClientHelper echoClient (interfaces.GetAddress (1), 9);
  echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
  echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
  echoClient.SetAttribute ("PacketSize", UintegerValue (1024));

  ApplicationContainer clientApps = echoClient.Install (nodes.Get (0));
  clientApps.Start (Seconds (2.0));
  clientApps.Stop (Seconds (10.0));

  AnimationInterface anim(animFile);
  anim.SetConstantPosition(nodes.Get(0),1.0,2.0);
  anim.SetConstantPosition(nodes.Get(1),45.0,60.0);

  AsciiTraceHelper ascii;
  pointToPoint.EnableAsciiAll(ascii.CreateFileStream("first.tr"));

  Simulator::Run ();
  Simulator::Destroy ();
  return 0;
}
```

P2p & csma

```
// Bus topology using point to point protocol

#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/applications-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/csma-module.h"
#include "ns3/netanim-module.h"
#include "ns3/ipv4-global-routing-helper.h"


using namespace ns3;


int main (){

  LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
  LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);

  std::string animFile="second.xml";

  NodeContainer p2pNodes;
```

```
p2pNodes.Create (2);

NodeContainer csmaNodes;
csmaNodes.Add(p2pNodes.Get(1));
csmaNodes.Create (3);

PointToPointHelper pointToPoint;
pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));

NetDeviceContainer p2pDevices;
p2pDevices = pointToPoint.Install (p2pNodes);

CsmaHelper csma;
csma.SetChannelAttribute ("DataRate", StringValue ("5Mbps"));
csma.SetChannelAttribute ("Delay", StringValue ("2ms"));

NetDeviceContainer csmaDevices;
csmaDevices = csma.Install (csmaNodes);

InternetStackHelper stack;
stack.Install (p2pNodes.Get(0));
stack.Install (csmaNodes);

Ipv4AddressHelper address;
address.SetBase ("10.1.1.0", "255.255.255.0");
Ipv4InterfaceContainer p2pInterfaces = address.Assign (p2pDevices);
address.SetBase ("10.1.2.0", "255.255.255.0");
Ipv4InterfaceContainer csmaInterfaces = address.Assign (csmaDevices);

UdpEchoServerHelper echoServer (9);

ApplicationContainer serverApps = echoServer.Install (csmaNodes.Get (3));
serverApps.Start (Seconds (1.0));
serverApps.Stop (Seconds (10.0));

UdpEchoClientHelper echoClient (csmaInterfaces.GetAddress (3), 9);
echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient.SetAttribute ("PacketSize", UintegerValue (1024));

ApplicationContainer clientApps = echoClient.Install (p2pNodes.Get (0));
clientApps.Start (Seconds (2.0));
clientApps.Stop (Seconds (10.0));

Ipv4GlobalRoutingHelper::PopulateRoutingTables();

AnimationInterface anim(animFile);
```

```
  anim.SetConstantPosition(p2pNodes.Get(0),1.0,2.0);
  anim.SetConstantPosition(csmaNodes.Get(0),45.0,60.0);
  anim.SetConstantPosition(csmaNodes.Get(1),55.0,60.0);
  anim.SetConstantPosition(csmaNodes.Get(2),65.0,60.0);
  anim.SetConstantPosition(csmaNodes.Get(3),75.0,60.0);

  AsciiTraceHelper ascii;
  pointToPoint.EnableAsciiAll(ascii.CreateFileStream("second1.tr"));
  csma.EnableAsciiAll(ascii.CreateFileStream("second2.tr"));

  Simulator::Run ();
  Simulator::Destroy ();
  return 0;
}
```

Csma

```
// Client - Server using CSMA protocol

#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/applications-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/csma-module.h"
#include "ns3/netanim-module.h"
#include "ns3/ipv4-global-routing-helper.h"

using namespace ns3;

int main (){

  LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
  LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);

  std::string animFile="third.xml";

  NodeContainer csmaNodes;
  csmaNodes.Create (4);

  CsmaHelper csma;
  csma.SetChannelAttribute ("DataRate", StringValue ("100Mbps"));
  csma.SetChannelAttribute ("Delay", StringValue ("2ms"));

  NetDeviceContainer csmaDevices;
  csmaDevices = csma.Install (csmaNodes);

  InternetStackHelper stack;
  stack.Install (csmaNodes);

  Ipv4AddressHelper address;
  address.SetBase ("10.1.1.0", "255.255.255.0");

  Ipv4InterfaceContainer csmaInterfaces = address.Assign (csmaDevices);

  UdpEchoServerHelper echoServer (9);

  ApplicationContainer serverApps = echoServer.Install (csmaNodes.Get (0));
  serverApps.Start (Seconds (1.0));
  serverApps.Stop (Seconds (10.0));

  UdpEchoClientHelper echoClient (csmaInterfaces.GetAddress (0), 9);
  echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
  echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
  echoClient.SetAttribute ("PacketSize", UintegerValue (1024));

  ApplicationContainer clientApps = echoClient.Install (csmaNodes.Get (3));
  clientApps.Start (Seconds (2.0));
  clientApps.Stop (Seconds (10.0));

  Ipv4GlobalRoutingHelper::PopulateRoutingTables();

  AnimationInterface anim(animFile);
  anim.SetConstantPosition(csmaNodes.Get(0),45.0,40.0);
  anim.SetConstantPosition(csmaNodes.Get(1),55.0,40.0);
  anim.SetConstantPosition(csmaNodes.Get(2),65.0,40.0);
  anim.SetConstantPosition(csmaNodes.Get(3),75.0,40.0);

  AsciiTraceHelper ascii;
  csma.EnableAsciiAll(ascii.CreateFileStream("third.tr"));

  Simulator::Run ();
  Simulator::Destroy ();
  return 0;
}
```

Ping

```
// Pinging over network topology consisting of 3 nodes
```

```cpp
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/applications-module.h"
#include "ns3/csma-module.h"
#include "ns3/internet-apps-module.h"
#include "ns3/netanim-module.h"

using namespace ns3;

int main (){
 std::string animFile="fifth.xml";

  NodeContainer nodes;
  nodes.Create (3);

  CsmaHelper csma;
  //csma.SetChannelAttribute ("DataRate", DataRateValue (DataRate (5000000)));
  //csma.SetChannelAttribute ("Delay", TimeValue (MilliSeconds (2)));
  csma.SetChannelAttribute ("DataRate", StringValue ("5Mbps"));
  csma.SetChannelAttribute ("Delay", StringValue ("2ms"));

  NetDeviceContainer devices;
  devices = csma.Install (nodes);

  InternetStackHelper stack;
  stack.Install (nodes);

  Ipv4AddressHelper address;
  address.SetBase ("10.0.1.0", "255.255.255.0");
  Ipv4InterfaceContainer interface = address.Assign (devices);

  V4PingHelper ping = V4PingHelper (interface.GetAddress (2));

  NodeContainer pingers;
  pingers.Add (nodes.Get (0));
  pingers.Add (nodes.Get (1));

  ApplicationContainer apps = ping.Install (pingers);
  apps.Start (Seconds (2.0));
  apps.Stop (Seconds (5.0));


  csma.EnablePcapAll ("csma-ping", true);

  AnimationInterface anim(animFile);
  anim.SetConstantPosition(nodes.Get(0),10.0,60.0);
  anim.SetConstantPosition(nodes.Get(1),10.0,100.0);
  anim.SetConstantPosition(nodes.Get(2),50.0,60.0);


  Simulator::Run ();
  Simulator::Destroy ();
}
```

Star

```
// Star topology using point to point protocol

#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/applications-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/point-to-point-layout-module.h"
#include "ns3/netanim-module.h"

using namespace ns3;

int main ()
{
 std::string animFile="fourth.xml";

 PointToPointHelper ptp;
 ptp.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
 ptp.SetChannelAttribute ("Delay", StringValue ("2ms"));

 PointToPointStarHelper star (8, ptp); //8 nodes

 InternetStackHelper internet;
 star.InstallStack (internet);
 star.AssignIpv4Addresses (Ipv4AddressHelper ("10.1.1.0", "255.255.255.0"));
 Address hubLocalAddress (InetSocketAddress (Ipv4Address::GetAny (), 50000));

 PacketSinkHelper sink ("ns3::TcpSocketFactory", hubLocalAddress);
 ApplicationContainer hubApp = sink.Install (star.GetHub ());
 hubApp.Start (Seconds (1.0));
 hubApp.Stop (Seconds (10.0));

 OnOffHelper onOffHelper ("ns3::TcpSocketFactory", Address ());
 onOffHelper.SetAttribute ("OnTime", StringValue
("ns3::ConstantRandomVariable[Constant=1]"));
 onOffHelper.SetAttribute ("OffTime", StringValue
("ns3::ConstantRandomVariable[Constant=0]"));
```

```
 ApplicationContainer spokeApps;
 for (uint32_t i = 0; i < star.SpokeCount (); ++i)
 {
     AddressValue remoteAddress (InetSocketAddress (star.GetHubIpv4Address
(i), 50000)); // 50000 is the port number
     onOffHelper.SetAttribute ("Remote", remoteAddress);
     spokeApps.Add (onOffHelper.Install (star.GetSpokeNode (i)));
 }
 spokeApps.Start (Seconds (1.0));
 spokeApps.Stop (Seconds (10.0));

 Ipv4GlobalRoutingHelper::PopulateRoutingTables ();

 ptp.EnablePcapAll ("star");

 AnimationInterface anim(animFile);
 anim.SetConstantPosition(star.GetHub(),10.0,60.0);

 Simulator::Run ();
 Simulator::Destroy ();
 return 0;
}
```