```
diff -r 7cd208d68f6c ns-3.12.1/doc/main.h
--- a/ns-3.12.1/doc/main.h      Sat Dec 17 12:11:54 2011 +0200
+++ b/ns-3.12.1/doc/main.h      Sun Dec 18 14:16:43 2011 +0200
@@ -29,6 +29,7 @@
   *     - aodv
   *     - applications
   *     - bridge
+  *     - brite
   *     - click
   *     - config-store
   *     - core
diff -r 7cd208d68f6c ns-3.12.1/doc/models/Makefile
--- a/ns-3.12.1/doc/models/Makefile     Sat Dec 17 12:11:54 2011 +0200
+++ b/ns-3.12.1/doc/models/Makefile     Sun Dec 18 14:16:43 2011 +0200
@@ -20,6 +20,7 @@
        $(SRC)/aodv/doc/aodv.rst \
        $(SRC)/applications/doc/applications.rst \
        $(SRC)/bridge/doc/bridge.rst \
+       $(SRC)/brite/doc/brite.rst \
        $(SRC)/click/doc/click.rst \
        $(SRC)/csma/doc/csma.rst \
        $(SRC)/dsdv/doc/dsdv.rst \
diff -r 7cd208d68f6c ns-3.12.1/doc/models/source/index.rst
--- a/ns-3.12.1/doc/models/source/index.rst     Sat Dec 17 12:11:54 2011 +0200
+++ b/ns-3.12.1/doc/models/source/index.rst     Sun Dec 18 14:16:43 2011 +0200
@@ -23,6 +23,7 @@
    aodv
    applications
    bridge
+   brite
    click
    csma
    dsdv
diff -r 7cd208d68f6c ns-3.12.1/src/brite/doc/brite.rst
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/doc/brite.rst Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,136 @@
+.. include:: replace.txt
+
+BRITE Integration
+-----------------
+
+This model implements an interface to BRITE, the Boston university
+Representative Internet Topology gEnerator [1]_. BRITE is a standard tool for
+generating realistic internet topologies. The ns-3 model, described herein,
+provides a helper class to facilitate generating ns-3 specific topologies
+using BRITE configuration files. BRITE builds the original graph which is
+stored as nodes and edges in the ns-3 BriteTopolgyHelper class. By accessing
+these nodes and edges, an ns-3 specific topology can easily be built, as
+seen in the example provided, brite-generic-example.
+
+
+Model Description
+*****************
+
+The model relies on building an external BRITE library,
+and then building some ns-3 helpers that call out to the library.
+The source code for the ns-3 helpers lives in the directory
+``src/brite/helper``.
+
+Design
+======
+
+The BRITE ns-3 helpers call out to the external BRITE library, and using a standard
+BRITE configuration file, the BRITE code builds a graph with nodes and edges
+according to this configuration file. Please see the BRITE documenation or the
+example configuration files in src/brite/examples/conf_files to get a better
```

```
+grasp of BRITE configuration options. The graph built by BRITE is returned to
+ns-3, and these nodes and edges are stored in ns-3 structures for use by the
+user.
+
+The ns-3 data structures holding all of the returned BRITE information are
+BriteNodeInfoList and BriteEdgeInfoList. Both of these are simply a vector
+of structs with each Node and Edge struct holding all of the individual
+node and edge information returned from the BRITE graph.
+
+Finally, these two data structures are used to build ns-3 specific topologies.
+It is up to the user to use these nodes and edges and connect them correctly.
+An example, brite-generic-example, is provided to show the user how to do this.
+
+Note: BRITE accepts a seed file to seed its psuedo-random number generator. It also
+spits out a new seed file after every run -- overwriting your old seed file --
+in case you wish to randomize subsequent runs. Finally, it saves the most
+recently used seed file in a file called "last_seed_file." Rather than
+overwriting your old seed file, we have changed it in the ns-3
+interface. You still pass in a seed file, but you also pass in another file for
+BRITE to write the new seed file. This keeps BRITE from overwriting your seed
+file. We believe this makes it easier to run the exact same simulation over and
+over, which may be important to some users. Finally, if you wish to overwrite the seed file
+each run, randomizing each subsequent run as before, you can simply pass in the
+same file name for the seed file and new seed file in ns-3
+
+References
+==========
+
+.. [1] Alberto Medina, Anukool Lakhina, Ibrahim Matta, and John Byers. BRITE: An Approach to Universal
Topology Generation. In Proceedings of the International Workshop on Modeling, Analysis and Simulation of
Computer and Telecommunications Systems- MASCOTS '01, Cincinnati, Ohio, August 2001.
+
+Usage
+*****
+
+The brite-generic-example can be referenced to see basic usage of the BRITE
+interface. In summary, the BriteTopologyHelper is used as the interface point
+by passing in a BRITE configuration and seed file. The BRITE generated nodes
+and edges can then be accessed through this helper to create ns-3 nodes and
+edges.
+
+Within the brite-generic-example, assigning IPs and creating applications
+are done fairly simply, as topology generation is the main topic of the
+BRITE interface. It is very likely that the user will need to change the
+way in which IP addresses are assigned or applications are installed.
+
+Building BRITE Integration
+==========================
+
+The first step is to download and build the ns-3 specific BRITE repository:::
+
+  $ hg clone http://code.nsnam.org/jpelkey3/BRITE
+  $ cd BRITE
+  $ make
+
+This will build BRITE and create a library, libbrite.so, within the BRITE
+directory.
+
+Once BRITE has been built successfully, we proceed to configure ns-3 with
+BRITE support. Change to your ns-3 directory:::
+
+  $ ./waf configure --with-brite=/your/path/to/brite/source --enable-examples
+
+Make sure it says 'enabled' beside 'BRITE Integration'. If it does not, then
+something has gone wrong. Either you have forgotten to build BRITE first
+following the steps above, or ns-3 could not find your BRITE directory.
+
```

```
+Next, build ns-3:::
+
+  $ ./waf
+
+Examples
+========
+For an example demonstrating BRITE integration
+run:::
+
+  $ ./waf --run 'brite-generic-example --verbose=1'
+
+By enabling the verbose parameter, the example will print out the node and
+edge information in a similar format to standard BRITE output. There are
+many other command-line parameters including confFile, seedFile, newseedFile,
+tracing, and nix, described below:
+
+   confFile:    A BRITE configuration file. Many different BRITE configuration
+                file examples exist in the src/brite/examples/conf_files directory, for
+                example, RTBarabasi20.conf and RTWaxman.conf. Please refer to
+                the conf_files directory for more examples.
+
+   seedFile:    BRITE specific seed file to seed a psuedo-random number genrator
+                from BRITE.
+
+   newseedFile: BRITE automatically generates a new seed file for you, if you
+                would like to randomize subsequent runs. If you would like
+                this to happen automatically, simply use the same file for
+                seedFile and newseedFile. This way, BRITE will run with the
+                current seeds and then overwrite them for the next run.
+
+   verbose:     Prints out the node and edge information in a similar format
+                to standard BRITE output.
+
+   tracing:     Enables ascii tracing.
+
+   nix:         Enables nix-vector routing. Global routing is used by default.
+
+The generic BRITE example also support visualization using pyviz, assuming
+python bindings in ns-3 are enabled:::
+
+  $ ./waf --run brite-generic-example --vis
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/brite-generic-example.cc
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/brite-generic-example.cc     Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,244 @@
+/* -*- Mode:C++; c-file-style:"gnu"; indent-tabs-mode:nil; -*- */
+/*
+ * This program is free software; you can redistribute it and/or modify
+ * it under the terms of the GNU General Public License version 2 as
+ * published by the Free Software Foundation;
+ *
+ * This program is distributed in the hope that it will be useful,
+ * but WITHOUT ANY WARRANTY; without even the implied warranty of
+ * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
+ * GNU General Public License for more details.
+ *
+ * You should have received a copy of the GNU General Public License
+ * along with this program; if not, write to the Free Software
+ * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
+ *
+ * Author: Josh Pelkey <jpelkey@gatech.edu>
+ */
+
+#include <string>
+#include "ns3/core-module.h"
+#include "ns3/network-module.h"
+#include "ns3/internet-module.h"
```

```
+#include "ns3/point-to-point-module.h"
+#include "ns3/mobility-module.h"
+#include "ns3/applications-module.h"
+#include "ns3/brite-module.h"
+#include "ns3/ipv4-static-routing-helper.h"
+#include "ns3/ipv4-list-routing-helper.h"
+#include "ns3/ipv4-nix-vector-helper.h"
+#include "ns3/ipv4-global-routing-helper.h"
+
+using namespace ns3;
+
+NS_LOG_COMPONENT_DEFINE ("BriteExample");
+
+int
+main (int argc, char *argv[])
+{
+  LogComponentEnable ("BriteExample", LOG_LEVEL_ALL);
+
+  Config::SetDefault ("ns3::OnOffApplication::DataRate", StringValue ("5kb/s"));
+
+  // BRITE needs a configuration file to build its graph. By default, this
+  // example will use the RTWaxman20.conf file. There are many others
+  // which can be found in the BRITE/conf_files directory
+  std::string confFile = "src/brite/examples/conf_files/RTWaxman20.conf";
+  std::string seedFile = "src/brite/examples/conf_files/seed_file";
+  std::string newseedFile = "src/brite/examples/conf_files/newseed_file";
+  bool verbose = false;
+  bool tracing = false;
+  bool nix = false;
+
+  CommandLine cmd;
+  cmd.AddValue ("confFile", "BRITE conf file", confFile);
+  cmd.AddValue ("seedFile", "BRITE seed file", seedFile);
+  cmd.AddValue ("newseedFile", "BRITE automatically generated new seed file", newseedFile);
+  cmd.AddValue ("verbose", "Enable or disable BRITE output", verbose);
+  cmd.AddValue ("tracing", "Enable or disable ascii tracing", tracing);
+  cmd.AddValue ("nix", "Enable or disable nix-vector routing", nix);
+
+  cmd.Parse (argc,argv);
+
+  // Invoke the BriteTopologyHelper and pass in a BRITE
+  // configuration file and a seed file. This will use
+  // BRITE to build a graph from which we can build the ns-3 topology
+  BriteTopologyHelper bth (confFile, seedFile, newseedFile);
+
+  // Grab the node and edge list from the BriteTopologyHelper. From this list
+  // we can extract information to create ns-3 nodes and edges
+  BriteTopologyHelper::BriteNodeInfoList nodeList = bth.GetBriteNodeInfoList ();
+  BriteTopologyHelper::BriteEdgeInfoList edgeList = bth.GetBriteEdgeInfoList ();
+
+  // Create all ns-3 nodes and place them in a node container
+  NodeContainer nodes;
+  nodes.Create (nodeList.size ());
+
+  // Use a constant mobility model to force node placement in pyviz according
+  // to the BRITE output. This is completely unnecessary if you do not plan
+  // to visualize your experiement, or if you don't really care where your
+  // nodes end up in pyviz. Comment out to speed up the simulation.
+  /*
+  MobilityHelper mobil;
+  mobil.SetMobilityModel ("ns3::ConstantPositionMobilityModel");
+  Ptr<ListPositionAllocator> positionAlloc = CreateObject<ListPositionAllocator> ();
+
+  for (BriteTopologyHelper::BriteNodeInfoList::iterator it = nodeList.begin();
+       it != nodeList.end(); ++it)
+    {
+      positionAlloc->Add (Vector ((*it).xCoordinate, (*it).yCoordinate, 0.0));
```

```
+    }
+
+  mobil.SetPositionAllocator (positionAlloc);
+  mobil.Install (nodes);
+  */
+
+  // Iterate the edge list and create all the point-to-point links between
+  // nodes. Save the net-devices in a container.
+  NetDeviceContainer ndc;
+  PointToPointHelper p2p;
+  for (BriteTopologyHelper::BriteEdgeInfoList::iterator it = edgeList.begin();
+       it != edgeList.end(); ++it)
+    {
+      // Set the link delay
+      p2p.SetChannelAttribute ("Delay",
+                               TimeValue (Seconds ((*it).delay)));
+
+      // Set the link bandwidth. NOTE: The bandwidth defaults to bits/s. The
+      // bandwidths are originally specified in the BRITE configuration file
+      // and are unitless. It is your responsibility to ensure these numbers
+      // are correct, either through the BRITE configuration file or modifying
+      // the numbers below, as we've done here (changed to kbps).
+      p2p.SetDeviceAttribute ("DataRate",
+                              DataRateValue (DataRate ((*it).bandwidth * 1000)));
+
+      // Install the link and push netdevice to the container
+      ndc.Add (p2p.Install (nodes.Get ((*it).srcId),
+                            nodes.Get ((*it).destId)));
+    }
+
+  // Setup the internet stack
+  InternetStackHelper stack;
+  Ipv4NixVectorHelper nixRouting;
+  Ipv4GlobalRoutingHelper globalRouting;
+  Ipv4StaticRoutingHelper staticRouting;
+
+  Ipv4ListRoutingHelper listRouting;
+
+  if (nix)
+    {
+      listRouting.Add (staticRouting, 0);
+      listRouting.Add (nixRouting, 10);
+      stack.SetRoutingHelper (listRouting);
+    }
+  else {
+      listRouting.Add (staticRouting, 0);
+      listRouting.Add (globalRouting, 10);
+      stack.SetRoutingHelper (listRouting);
+  }
+
+  stack.Install (nodes);
+
+  // Assign IPs as simple as possible. You may wish to do this manually.
+  // Since we do it automatically here using the netdevice container,
+  // and the entries in the container are somewhat random, the IPs
+  // assigned are going to be all over the place on a specfic node's
+  // netdevices. This will make it difficult to determine
+  // which IPs correspond to which node's netdevices later on.
+  Ipv4AddressHelper address;
+  address.SetBase ("10.0.0.0", "255.0.0.0");
+  Ipv4InterfaceContainer ifc = address.Assign (ndc);
+
+  // Install applications. To complete this example, we will simply install a
+  // packet sink on the zero-th node and an OnOff application on a randomly
+  // selected second node. You will likely want to do this differently.
+
+  // Set up packet sink
```

```
+  Address sinkLocalAddress (InetSocketAddress (Ipv4Address::GetAny (), 1024));
+  PacketSinkHelper sinkHelper ("ns3::UdpSocketFactory", sinkLocalAddress);
+  ApplicationContainer sinkApp = sinkHelper.Install (nodes.Get (0));
+  sinkApp.Start (Seconds (5.0));
+  sinkApp.Stop (Seconds (150.0));
+
+  // Set up OnOff application
+  // Remember, the IPs were assigned somewhat arbitrarily, so in order
+  // to get the remote address correct, we have to access it from node 0
+  // directly. In other words, we can't assume that the zero-th entry
+  // in the interface container corresponds to an interface on node 0.
+  Ptr<Ipv4> sink = nodes.Get (0)->GetObject<Ipv4> ();
+  Ipv4InterfaceAddress iaddSink = sink->GetAddress (1, 0);
+  Ipv4Address ipv4Sink = iaddSink.GetLocal ();
+  Address remoteAddress (InetSocketAddress (ipv4Sink, 1024));
+
+  OnOffHelper clientHelper ("ns3::UdpSocketFactory", remoteAddress);
+  uint32_t ranNum = UniformVariable ().GetInteger (1, (nodeList.size () - 1));
+  ApplicationContainer clientApp = clientHelper.Install (nodes.Get (ranNum));
+  clientApp.Start (Seconds (5.0));
+  clientApp.Stop (Seconds (100.0));
+
+  // Set up tracing, if enabled
+  if (tracing)
+    {
+      AsciiTraceHelper ascii;
+      p2p.EnableAsciiAll (ascii.CreateFileStream ("brite-topology.tr"));
+    }
+
+  if (nix)
+    {
+      std::cout << "Using Nix-vector routing" << std::endl;
+    }
+  else
+    {
+      std::cout << "Populate routing tables" << std::endl;
+      Ipv4GlobalRoutingHelper::PopulateRoutingTables ();
+    }
+
+  // Run the simulator
+  Simulator::Stop (Seconds (200.0));
+  Simulator::Run ();
+  Simulator::Destroy ();
+
+  if (verbose)
+    {
+      // Output the BRITE formatted node list
+      NS_LOG_INFO ("\nTotal number of nodes: " << nodeList.size () << "\n");
+      for (BriteTopologyHelper::BriteNodeInfoList::iterator it = nodeList.begin();
+            it != nodeList.end(); ++it)
+        {
+          NS_LOG_INFO ((*it).nodeId << " "
+            << (*it).xCoordinate <<  " "
+            << (*it).yCoordinate << " "
+            << (*it).inDegree << " "
+            << (*it).outDegree << " "
+            << (*it).asId << " "
+            << (*it).type);
+        }
+
+      // Output the BRITE formatted edge list
+      NS_LOG_INFO ("\nTotal number of edges: " << edgeList.size () << "\n");
+      for (BriteTopologyHelper::BriteEdgeInfoList::iterator it = edgeList.begin();
+            it != edgeList.end(); ++it)
+        {
+          NS_LOG_INFO ((*it).edgeId << " "
+            << (*it).srcId <<  " "
```

```
+            << (*it).destId << " "
+            << (*it).length << " "
+            << (*it).delay << " "
+            << (*it).bandwidth << " "
+            << (*it).asFrom << " "
+            << (*it).asTo << " "
+            << (*it).type);
+        }
+    }
+
+  // If you want to output the BRITE file from BRITE, just like the stand-alone
+  // BRITE code would, you can do that too. Just comment out the lines below.
+  /*
+  std::string briteOutput = "briteOutput";
+  std::string otterOutput = "otterOutput";
+  bth.GetBriteTopology ()->BriteOutput (const_cast<char *> (briteOutput.c_str ()));
+  bth.GetBriteTopology ()->OtterOutput (const_cast<char *> (otterOutput.c_str ()));
+  */
+
+  return 0;
+}
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/conf_files/ASBarabasi.conf
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/conf_files/ASBarabasi.conf   Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,21 @@
+#This config file was generated by the GUI.
+
+BriteConfig
+
+BeginModel
+       Name =  4                #Router Barabasi=2, AS Barabasi =4
+       N = 1000                 #Number of nodes in graph
+       HS = 1000                #Size of main plane (number of squares)
+       LS = 100                 #Size of inner planes (number of squares)
+       NodePlacement = 1        #Random = 1, Heavy Tailed = 2
+       m = 2                    #Number of neighboring node each new node connects to.
+       BWDist = 1               #Constant = 1, Uniform =2, HeavyTailed = 3, Exponential =4
+       BWMin = 10.0
+       BWMax = 1024.0
+EndModel
+
+
+BeginOutput                    #**Atleast one of these options should have value 1**
+       BRITE = 1                #0 = Do not save as BRITE, 1 = save as BRITE.
+       OTTER = 0                #0 = Do not visualize with Otter, 1 = Visualize
+EndOutput
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/conf_files/ASWaxman.conf
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/conf_files/ASWaxman.conf     Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,23 @@
+#This config file was generated by the GUI.
+
+BriteConfig
+
+BeginModel
+       Name =  3                #Router Waxman = 1, AS Waxman = 3
+       N = 1000                 #Number of nodes in graph
+       HS = 1000                #Size of main plane (number of squares)
+       LS = 100                 #Size of inner planes (number of squares)
+       NodePlacement = 1        #Random = 1, Heavy Tailed = 2
+       GrowthType = 1           #Incremental = 1, All = 2
+       m = 2                    #Number of neighboring node each new node connects to.
+       alpha = 0.15             #Waxman Parameter
+       beta = 0.2               #Waxman Parameter
+       BWDist = 1               #Constant = 1, Uniform =2, HeavyTailed = 3, Exponential =4
+       BWMin = 10.0
+       BWMax = 1024.0
```

```
+EndModel
+
+BeginOutput                    #**Atleast one of these options should have value 1**
+       BRITE = 1               #0 = Do not save as BRITE, 1 = save as BRITE.
+       OTTER = 0               #0 = Do not visualize with Otter, 1 = Visualize
+EndOutput
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/conf_files/RTBarabasi.conf
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/conf_files/RTBarabasi.conf   Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,22 @@
+#This config file was generated by the GUI.
+
+BriteConfig
+
+BeginModel
+       Name =  2               #Router Barabasi=2, AS Barabasi =4
+       N = 100                 #Number of nodes in graph
+       HS = 1000               #Size of main plane (number of squares)
+       LS = 100                #Size of inner planes (number of squares)
+       NodePlacement = 1       #Random = 1, Heavy Tailed = 2
+       m = 2                   #Number of neighboring node each new node connects to.
+       BWDist = 1              #Constant = 1, Uniform =2, HeavyTailed = 3, Exponential =4
+       BWMin = 10.0
+       BWMax = 1024.0
+EndModel
+
+
+
+BeginOutput                    #**Atleast one of these options should have value 1**
+       BRITE = 1               #0 = Do not save as BRITE, 1 = save as BRITE.
+       OTTER = 0               #0 = Do not visualize with Otter, 1 = Visualize
+EndOutput
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/conf_files/RTBarabasi10.conf
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/conf_files/RTBarabasi10.conf Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,22 @@
+#This config file was generated by the GUI.
+
+BriteConfig
+
+BeginModel
+       Name =  2               #Router Barabasi=2, AS Barabasi =4
+       N = 10                  #Number of nodes in graph
+       HS = 1000               #Size of main plane (number of squares)
+       LS = 100                #Size of inner planes (number of squares)
+       NodePlacement = 1       #Random = 1, Heavy Tailed = 2
+       m = 2                   #Number of neighboring node each new node connects to.
+       BWDist = 1              #Constant = 1, Uniform =2, HeavyTailed = 3, Exponential =4
+       BWMin = 10.0
+       BWMax = 1024.0
+EndModel
+
+
+
+BeginOutput                    #**Atleast one of these options should have value 1**
+       BRITE = 1               #0 = Do not save as BRITE, 1 = save as BRITE.
+       OTTER = 0               #0 = Do not visualize with Otter, 1 = Visualize
+EndOutput
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/conf_files/RTBarabasi20.conf
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/conf_files/RTBarabasi20.conf Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,22 @@
+#This config file was generated by the GUI.
+
+BriteConfig
+
+BeginModel
```

```
+       Name =  2               #Router Barabasi=2, AS Barabasi =4
+       N = 20                  #Number of nodes in graph
+       HS = 1000               #Size of main plane (number of squares)
+       LS = 100                #Size of inner planes (number of squares)
+       NodePlacement = 1       #Random = 1, Heavy Tailed = 2
+       m = 2                   #Number of neighboring node each new node connects to.
+       BWDist = 1              #Constant = 1, Uniform =2, HeavyTailed = 3, Exponential =4
+       BWMin = 10.0
+       BWMax = 1024.0
+EndModel
+
+
+
+BeginOutput                    #**Atleast one of these options should have value 1**
+       BRITE = 1               #0 = Do not save as BRITE, 1 = save as BRITE.
+       OTTER = 1               #0 = Do not visualize with Otter, 1 = Visualize
+EndOutput
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/conf_files/RTBarabasi5.conf
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/conf_files/RTBarabasi5.conf  Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,22 @@
+#This config file was generated by the GUI.
+
+BriteConfig
+
+BeginModel
+       Name =  2               #Router Barabasi=2, AS Barabasi =4
+       N = 5                   #Number of nodes in graph
+       HS = 1000               #Size of main plane (number of squares)
+       LS = 100                #Size of inner planes (number of squares)
+       NodePlacement = 1       #Random = 1, Heavy Tailed = 2
+       m = 2                   #Number of neighboring node each new node connects to.
+       BWDist = 1              #Constant = 1, Uniform =2, HeavyTailed = 3, Exponential =4
+       BWMin = 10.0
+       BWMax = 1024.0
+EndModel
+
+
+
+BeginOutput                    #**Atleast one of these options should have value 1**
+       BRITE = 1               #0 = Do not save as BRITE, 1 = save as BRITE.
+       OTTER = 0               #0 = Do not visualize with Otter, 1 = Visualize
+EndOutput
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/conf_files/RTWaxman.conf
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/conf_files/RTWaxman.conf     Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,25 @@
+#This config file was generated by the GUI.
+
+BriteConfig
+
+BeginModel
+       Name =  1               #Router Waxman = 1, AS Waxman = 3
+       N = 1000                #Number of nodes in graph
+       HS = 1000               #Size of main plane (number of squares)
+       LS = 100                #Size of inner planes (number of squares)
+       NodePlacement = 1       #Random = 1, Heavy Tailed = 2
+       GrowthType = 1          #Incremental = 1, All = 2
+       m = 2                   #Number of neighboring node each new node connects to.
+       alpha = 0.15            #Waxman Parameter
+       beta = 0.2              #Waxman Parameter
+       BWDist = 1              #Constant = 1, Uniform =2, HeavyTailed = 3, Exponential =4
+       BWMin = 10.0
+       BWMax = 1024.0
+EndModel
+
+
```

```
+
+BeginOutput                    #**Atleast one of these options should have value 1**
+       BRITE = 1               #0 = Do not save as BRITE, 1 = save as BRITE.
+       OTTER = 1               #0 = Do not visualize with Otter, 1 = Visualize
+EndOutput
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/conf_files/RTWaxman10.conf
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/conf_files/RTWaxman10.conf   Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,25 @@
+#This config file was generated by the GUI.
+
+BriteConfig
+
+BeginModel
+       Name =  1               #Router Waxman = 1, AS Waxman = 3
+       N = 10                  #Number of nodes in graph
+       HS = 1000               #Size of main plane (number of squares)
+       LS = 100                #Size of inner planes (number of squares)
+       NodePlacement = 1       #Random = 1, Heavy Tailed = 2
+       GrowthType = 1          #Incremental = 1, All = 2
+       m = 2                   #Number of neighboring node each new node connects to.
+       alpha = 0.15            #Waxman Parameter
+       beta = 0.2              #Waxman Parameter
+       BWDist = 1              #Constant = 1, Uniform =2, HeavyTailed = 3, Exponential =4
+       BWMin = 10.0
+       BWMax = 1024.0
+EndModel
+
+
+
+BeginOutput                    #**Atleast one of these options should have value 1**
+       BRITE = 1               #0 = Do not save as BRITE, 1 = save as BRITE.
+       OTTER = 0               #0 = Do not visualize with Otter, 1 = Visualize
+EndOutput
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/conf_files/RTWaxman20.conf
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/conf_files/RTWaxman20.conf   Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,25 @@
+#This config file was generated by the GUI.
+
+BriteConfig
+
+BeginModel
+       Name =  1               #Router Waxman = 1, AS Waxman = 3
+       N = 20                  #Number of nodes in graph
+       HS = 1000               #Size of main plane (number of squares)
+       LS = 100                #Size of inner planes (number of squares)
+       NodePlacement = 1       #Random = 1, Heavy Tailed = 2
+       GrowthType = 1          #Incremental = 1, All = 2
+       m = 2                   #Number of neighboring node each new node connects to.
+       alpha = 0.15            #Waxman Parameter
+       beta = 0.2              #Waxman Parameter
+       BWDist = 1              #Constant = 1, Uniform =2, HeavyTailed = 3, Exponential =4
+       BWMin = 10.0
+       BWMax = 1024.0
+EndModel
+
+
+
+BeginOutput                    #**Atleast one of these options should have value 1**
+       BRITE = 1               #0 = Do not save as BRITE, 1 = save as BRITE.
+       OTTER = 0               #0 = Do not visualize with Otter, 1 = Visualize
+EndOutput
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/conf_files/RTWaxman5.conf
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/conf_files/RTWaxman5.conf    Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,25 @@
```

```
+#This config file was generated by the GUI.
+
+BriteConfig
+
+BeginModel
+       Name =  1               #Router Waxman = 1, AS Waxman = 3
+       N = 5                   #Number of nodes in graph
+       HS = 1000               #Size of main plane (number of squares)
+       LS = 100                #Size of inner planes (number of squares)
+       NodePlacement = 1       #Random = 1, Heavy Tailed = 2
+       GrowthType = 1          #Incremental = 1, All = 2
+       m = 2                   #Number of neighboring node each new node connects to.
+       alpha = 0.15            #Waxman Parameter
+       beta = 0.2              #Waxman Parameter
+       BWDist = 1              #Constant = 1, Uniform =2, HeavyTailed = 3, Exponential =4
+       BWMin = 10.0
+       BWMax = 1024.0
+EndModel
+
+
+
+BeginOutput                    #**Atleast one of these options should have value 1**
+       BRITE = 1               #0 = Do not save as BRITE, 1 = save as BRITE.
+       OTTER = 0               #0 = Do not visualize with Otter, 1 = Visualize
+EndOutput
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/conf_files/TD_ASBarabasi_RTWaxman.conf
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/conf_files/TD_ASBarabasi_RTWaxman.conf       Sun Dec 18 14:16:43 2011
+0200
@@ -0,0 +1,50 @@
+#This config file was generated by the GUI.
+
+BriteConfig
+
+BeginModel
+       Name = 5                #Top Down = 5
+       edgeConn = 2            #Random=1, Smallest Nonleaf = 2, Smallest Deg = 3, k-Degree=4
+       k = -1                  #Only needed if edgeConn is set to K-Degree, otherwise use -1
+       BWInter = 1             #Constant = 1, Uniform =2, HeavyTailed = 3, Exponential =4
+       BWInterMin = 10.0
+       BWInterMax = 1024.0
+       BWIntra = 3             #Constant = 1, Uniform =2, HeavyTailed = 3, Exponential =4
+       BWIntraMin = 10.0
+       BWIntraMax = 1024.0
+EndModel
+
+BeginModel
+       Name =  4               #Router Barabasi=2, AS Barabasi =4
+       N = 20          #Number of nodes in graph
+       HS = 1000               #Size of main plane (number of squares)
+       LS = 100                #Size of inner planes (number of squares)
+       NodePlacement = 1       #Random = 1, Heavy Tailed = 2
+       m = 10                  #Number of neighboring node each new node connects to.
+       BWDist = 1              #Constant = 1, Uniform =2, HeavyTailed = 3, Exponential =4
+       BWMin = -1.0
+       BWMax = -1.0
+EndModel
+
+BeginModel
+       Name =  1               #Router Waxman=2, AS Waxman =3
+       N = 40          #Number of nodes in graph
+       HS = 1000               #Size of main plane (number of squares)
+       LS = 100                #Size of inner planes (number of squares)
+       NodePlacement = 2       #Random = 1, Heavy Tailed = 2
+       GrowthType = 1          #Incremental = 1, All = 2
+       m = 1                   #Number of neighboring node each new node connects to.
+       alpha = 0.5             #Waxman Parameter
```

```
+       beta = 0.8              #Waxman Parameter
+       BWDist = 1              #Constant = 1, Uniform =2, HeavyTailed = 3, Exponential =4
+       BWMin = -1.0
+       BWMax = -1.0
+EndModel
+
+
+
+BeginOutput                    #**Atleast one of these options should have value 1**
+       BRITE = 1               #0 = Do not save as BRITE, 1 = save as BRITE.
+       OTTER = 0               #0 = Do not visualize with Otter, 1 = Visualize
+EndOutput
+
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/conf_files/newseed_file
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/conf_files/newseed_file      Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,6 @@
+PLACES 46249 4146 19926
+CONNECT 2281 52753 2215
+EDGE_CONN 53489 32975 5141
+GROUPING 34898 48253 21264
+ASSIGNMENT 44174 29196 31893
+BANDWIDTH 2445 3871 43415
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/conf_files/seed_file
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/conf_files/seed_file Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,6 @@
+PLACES 929 18840 38318
+CONNECT 26883 38699 1089
+EDGE_CONN 53489 32975 5141
+GROUPING 34898 48253 21264
+ASSIGNMENT 44174 29196 31893
+BANDWIDTH 2445 3871 43415
diff -r 7cd208d68f6c ns-3.12.1/src/brite/examples/wscript
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/examples/wscript      Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,5 @@
+## -*- Mode: python; py-indent-offset: 4; indent-tabs-mode: nil; coding: utf-8; -*-
+
+def build(bld):
+   obj = bld.create_ns3_program('brite-generic-example', ['brite', 'internet', 'point-to-point', 'nix-
vector-routing', 'applications', 'visualizer'])
+   obj.source = 'brite-generic-example.cc'
diff -r 7cd208d68f6c ns-3.12.1/src/brite/helper/brite-topology-helper.cc
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/helper/brite-topology-helper.cc       Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,219 @@
+/* -*- Mode:C++; c-file-style:"gnu"; indent-tabs-mode:nil; -*- */
+/*
+ * This program is free software; you can redistribute it and/or modify
+ * it under the terms of the GNU General Public License version 2 as
+ * published by the Free Software Foundation;
+ *
+ * This program is distributed in the hope that it will be useful,
+ * but WITHOUT ANY WARRANTY; without even the implied warranty of
+ * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
+ * GNU General Public License for more details.
+ *
+ * You should have received a copy of the GNU General Public License
+ * along with this program; if not, write to the Free Software
+ * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
+ *
+ * Author: Josh Pelkey <jpelkey@gatech.edu>
+ */
+
+#include "ns3/log.h"
+#include "ns3/abort.h"
```

```
+
+#include "brite-topology-helper.h"
+
+NS_LOG_COMPONENT_DEFINE ("BriteTopologyHelper");
+
+namespace ns3 {
+
+BriteTopologyHelper::BriteTopologyHelper (std::string confFile,
+                                          std::string seedFile,
+                                          std::string newseedFile)
+{
+  brite::Brite br (confFile, seedFile, newseedFile);
+  m_topology = br.GetTopology ();
+  BuildBriteNodeInfoList ();
+  BuildBriteEdgeInfoList ();
+}
+
+BriteTopologyHelper::~BriteTopologyHelper ()
+{
+  delete m_topology;
+}
+
+brite::Topology*
+BriteTopologyHelper::GetBriteTopology (void)
+{
+  return m_topology;
+}
+
+void
+BriteTopologyHelper::BuildBriteNodeInfoList (void)
+{
+  brite::Graph *g = m_topology->GetGraph ();
+  for (int i = 0; i < g->GetNumNodes (); ++i)
+    {
+      BriteNodeInfo nodeInfo;
+      nodeInfo.nodeId = g->GetNodePtr (i)->GetId ();
+      nodeInfo.xCoordinate = g->GetNodePtr (i)->GetNodeInfo ()->GetCoordX ();
+      nodeInfo.yCoordinate = g->GetNodePtr (i)->GetNodeInfo ()->GetCoordY ();
+      nodeInfo.inDegree = g->GetNodePtr (i)->GetInDegree ();
+      nodeInfo.outDegree = g->GetNodePtr (i)->GetOutDegree ();
+
+      switch(g->GetNodePtr(i)->GetNodeInfo()->GetNodeType())
+        {
+          case brite::NodeConf::RT_NODE:
+            nodeInfo.asId =
+              ((brite::RouterNodeConf*)(g->GetNodePtr(i)->GetNodeInfo()))->GetASId();
+
+            switch (((brite::RouterNodeConf*)(g->GetNodePtr(i)->GetNodeInfo()))->GetRouterType())
+              {
+                case brite::RouterNodeConf::RT_NONE:
+                  nodeInfo.type = "RT_NONE ";
+                  break;
+                case brite::RouterNodeConf::RT_LEAF:
+                  nodeInfo.type = "RT_LEAF ";
+                  break;
+                case brite::RouterNodeConf::RT_BORDER:
+                  nodeInfo.type = "RT_BORDER";
+                  break;
+                case brite::RouterNodeConf::RT_STUB:
+                  nodeInfo.type = "RT_STUB ";
+                  break;
+                case brite::RouterNodeConf::RT_BACKBONE:
+                  nodeInfo.type = "RT_BACKBONE ";
+                  break;
+                default:
+                  NS_FATAL_ERROR ("Topology::Output(): Improperly classfied Router node encountered...");
+              }
```

```
+            break;
+
+          case brite::NodeConf::AS_NODE:
+            nodeInfo.asId =
+              ((brite::ASNodeConf*)(g->GetNodePtr(i)->GetNodeInfo()))->GetASId();
+
+            switch (((brite::ASNodeConf*)(g->GetNodePtr(i)->GetNodeInfo()))->GetASType())
+              {
+                case brite::ASNodeConf::AS_NONE:
+                  nodeInfo.type = "AS_NONE ";
+                  break;
+                case brite::ASNodeConf::AS_LEAF:
+                  nodeInfo.type = "AS_LEAF ";
+                  break;
+                case brite::ASNodeConf::AS_STUB:
+                  nodeInfo.type = "AS_STUB ";
+                  break;
+                case brite::ASNodeConf::AS_BORDER:
+                  nodeInfo.type = "AS_BORDER ";
+                  break;
+                case brite::ASNodeConf::AS_BACKBONE:
+                  nodeInfo.type = "AS_BACKBONE ";
+                  break;
+                default:
+                  NS_FATAL_ERROR ("Topology::Output(): Improperly classfied AS node encountered...");
+              }
+            break;
+        }
+
+      m_briteNodeInfoList.push_back (nodeInfo);
+    }
+}
+
+void
+BriteTopologyHelper::BuildBriteEdgeInfoList (void)
+{
+  brite::Graph *g = m_topology->GetGraph ();
+  std::list<brite::Edge*>::iterator el;
+  std::list<brite::Edge*> edgeList = g->GetEdges ();
+
+  for (el = edgeList.begin (); el != edgeList.end (); el++)
+    {
+      BriteEdgeInfo edgeInfo;
+      edgeInfo.edgeId = (*el)->GetId ();
+      edgeInfo.srcId = (*el)->GetSrc ()->GetId ();
+      edgeInfo.destId = (*el)->GetDst ()->GetId ();
+      edgeInfo.length = (*el)->Length ();
+
+      switch((*el)->GetConf()->GetEdgeType())
+        {
+          case brite::EdgeConf::RT_EDGE:
+            edgeInfo.delay = ((brite::RouterEdgeConf*)((*el)->GetConf()))->GetDelay();
+            edgeInfo.bandwidth = (*el)->GetConf()->GetBW();
+            edgeInfo.asFrom = ((brite::RouterNodeConf*)((*el)->GetSrc()->GetNodeInfo()))->GetASId();
+            edgeInfo.asTo = ((brite::RouterNodeConf*)((*el)->GetDst()->GetNodeInfo()))->GetASId();
+            break;
+
+          case brite::EdgeConf::AS_EDGE:
+            edgeInfo.delay =  -1;   /* No delay for AS Edges */
+            edgeInfo.bandwidth = (*el)->GetConf()->GetBW();
+            edgeInfo.asFrom = ((brite::ASNodeConf*)((*el)->GetSrc()->GetNodeInfo()))->GetASId();
+            edgeInfo.asTo = ((brite::ASNodeConf*)((*el)->GetDst()->GetNodeInfo()))->GetASId();
+            break;
+
+          default:
+            NS_FATAL_ERROR ("Topology::Output(): Invalid Edge type encountered...");
+        }
```

```
+
+      switch ((*el)->GetConf()->GetEdgeType())
+        {
+          case brite::EdgeConf::RT_EDGE:
+            switch (((brite::RouterEdgeConf*)(*el)->GetConf())->GetRouterEdgeType())
+              {
+                case brite::RouterEdgeConf::RT_NONE:
+                  edgeInfo.type = "E_RT_NONE ";
+                  break;
+                case brite::RouterEdgeConf::RT_STUB:
+                  edgeInfo.type = "E_RT_STUB ";
+                  break;
+                case brite::RouterEdgeConf::RT_BORDER:
+                  edgeInfo.type = "E_RT_BORDER ";
+                  break;
+                case brite::RouterEdgeConf::RT_BACKBONE:
+                  edgeInfo.type = "E_RT_BACKBONE ";
+                  break;
+                default:
+                  NS_FATAL_ERROR ("Output(): Invalid router edge type...");
+              }
+          break;
+
+          case brite::EdgeConf::AS_EDGE:
+            switch (((brite::ASEdgeConf*)((*el)->GetConf()))->GetASEdgeType())
+              {
+                case brite::ASEdgeConf::AS_NONE:
+                  edgeInfo.type = "E_AS_NONE ";
+                  break;
+                case brite::ASEdgeConf::AS_STUB:
+                  edgeInfo.type = "E_AS_STUB ";
+                  break;
+                case brite::ASEdgeConf::AS_BORDER:
+                  edgeInfo.type = "E_AS_BORDER ";
+                  break;
+                case brite::ASEdgeConf::AS_BACKBONE:
+                  edgeInfo.type = "E_AS_BACKBONE ";
+                  break;
+                default:
+                  NS_FATAL_ERROR ("BriteOutput(): Invalid AS edge type...");
+              }
+          break;
+
+          default:
+            NS_FATAL_ERROR ("BriteOutput(): Invalid edge type...");
+
+        }
+
+      m_briteEdgeInfoList.push_back (edgeInfo);
+    }
+}
+
+BriteTopologyHelper::BriteNodeInfoList
+BriteTopologyHelper::GetBriteNodeInfoList (void)
+{
+  return m_briteNodeInfoList;
+}
+
+BriteTopologyHelper::BriteEdgeInfoList
+BriteTopologyHelper::GetBriteEdgeInfoList (void)
+{
+  return m_briteEdgeInfoList;
+}
+
+} // namespace ns3
diff -r 7cd208d68f6c ns-3.12.1/src/brite/helper/brite-topology-helper.h
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
```

```
+++ b/ns-3.12.1/src/brite/helper/brite-topology-helper.h	Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,146 @@
+/* -*- Mode:C++; c-file-style:"gnu"; indent-tabs-mode:nil; -*- */
+/*
+ * This program is free software; you can redistribute it and/or modify
+ * it under the terms of the GNU General Public License version 2 as
+ * published by the Free Software Foundation;
+ *
+ * This program is distributed in the hope that it will be useful,
+ * but WITHOUT ANY WARRANTY; without even the implied warranty of
+ * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
+ * GNU General Public License for more details.
+ *
+ * You should have received a copy of the GNU General Public License
+ * along with this program; if not, write to the Free Software
+ * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
+ *
+ * Author: Josh Pelkey <jpelkey@gatech.edu>
+ */
+
+#ifndef BRITE_TOPOLOGY_HELPER_H
+#define BRITE_TOPOLOGY_HELPER_H
+
+#include <iostream>
+#include <string>
+#include <vector>
+
+#include "ns3/channel.h"
+#include "ns3/node-container.h"
+#include "ns3/node-list.h"
+#include "Brite.h"
+
+namespace ns3 {
+
+/**
+ * \brief Interface with BRITE, the Boston university Representative Internet
+ *        Topology gEnerator
+ *
+ * This helper class creates an interface with BRITE and allows the user to
+ * easily create ns-3 topologies from BRITE generated graphs. This class
+ * accepts a BRITE configuration and seed file, much like the stand-alone
+ * BRITE software. Using these files, BRITE generates a graph which is
+ * stored herein. ns-3 examples can then grab the BRITE generated nodes and
+ * edges from this helper and create ns-3 specific topologies.
+ *
+ */
+
+class BriteTopologyHelper
+{
+  public:
+    /*
+     * Construct a BriteTopologyHelper
+     *
+     * \param confFile a BRITE configuration file
+     * \param seedFile a BRITE seed file
+     */
+    BriteTopologyHelper (std::string confFile,
+                         std::string seedFile,
+                         std::string newseedFile);
+
+    ~BriteTopologyHelper ();
+
+    /**
+     * \brief Node information from BRITE
+     *
+     * The BRITE code generates a graph and returns
+     * information on the nodes generated. This is
```

```
+     * stored here in a struct.
+     */
+    struct BriteNodeInfo
+      {
+        int nodeId;
+        double xCoordinate;
+        double yCoordinate;
+        int inDegree;
+        int outDegree;
+        int asId;
+        std::string type;
+      };
+
+    /**
+     * \brief Edge information from BRITE
+     *
+     * The BRITE code generates a graph and returns
+     * information on the edges generated. This is
+     * stored here in a struct.
+     */
+    struct BriteEdgeInfo
+      {
+        int edgeId;
+        int srcId;
+        int destId;
+        double length;
+        double delay;
+        double bandwidth;
+        int asFrom;
+        int asTo;
+        std::string type;
+      };
+
+    /**
+     * The BRITE code generates multiple nodes and edges. Each
+     * node and edge is stored in a BriteNodeInfo or BriteEdgeInfo
+     * struct, and each instance is stored in a vector.
+     */
+    typedef std::vector<BriteNodeInfo> BriteNodeInfoList;
+    typedef std::vector<BriteEdgeInfo> BriteEdgeInfoList;
+
+    /**
+     * \brief Get the BRITE nodes
+     *
+     * \returns a BriteNodeInfoList, i.e. a vector of
+                BriteNodeInfo structs
+     */
+    BriteNodeInfoList GetBriteNodeInfoList (void);
+
+    /**
+     * \brief Get the BRITE edges
+     *
+     * \returns a BriteEdgeInfoList, i.e. a vector of
+                BriteEdgeInfo structs
+     */
+    BriteEdgeInfoList GetBriteEdgeInfoList (void);
+
+    /**
+     * \brief Get the BRITE topology object
+     *
+     * This is useful if you want to BRITE specific things with the
+     * topology object, such as print the BRITE output file
+     * in Otter format
+     *
+     * \returns the BRITE topology object
+     */
+    brite::Topology* GetBriteTopology (void);
```

```
+
+  private:
+    void BuildBriteNodeInfoList (void);
+    void BuildBriteEdgeInfoList (void);
+
+    brite::Topology* m_topology;
+
+    BriteNodeInfoList m_briteNodeInfoList;
+    BriteEdgeInfoList m_briteEdgeInfoList;
+};
+
+} // namespace ns3
+
+#endif /* BRITE_TOPOLOGY_HELPER_H */
diff -r 7cd208d68f6c ns-3.12.1/src/brite/test/examples-to-run.py
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/test/examples-to-run.py       Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,20 @@
+#! /usr/bin/env python
+## -*- Mode: python; py-indent-offset: 4; indent-tabs-mode: nil; coding: utf-8; -*-
+
+# A list of C++ examples to run in order to ensure that they remain
+# buildable and runnable over time.  Each tuple in the list contains
+#
+#     (example_name, do_run, do_valgrind_run).
+#
+# See test.py for more information.
+cpp_examples = [
+    ("brite-generic-example", "ENABLE_BRITE == True", "False"),
+]
+
+# A list of Python examples to run in order to ensure that they remain
+# runnable over time.  Each tuple in the list contains
+#
+#     (example_name, do_run).
+#
+# See test.py for more information.
+python_examples = []
diff -r 7cd208d68f6c ns-3.12.1/src/brite/waf
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/waf   Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,2 @@
+#! /bin/sh
+exec "`dirname "$0"`"/../../../waf "$@"
diff -r 7cd208d68f6c ns-3.12.1/src/brite/wscript
--- /dev/null   Thu Jan 01 00:00:00 1970 +0000
+++ b/ns-3.12.1/src/brite/wscript       Sun Dec 18 14:16:43 2011 +0200
@@ -0,0 +1,78 @@
+## -*- Mode: python; py-indent-offset: 4; indent-tabs-mode: nil; coding: utf-8; -*-
+import os
+import Options
+
+
+def set_options(opt):
+    opt.add_option('--with-brite',
+                   help=('Use BRITE integration support, given by the indicated path,'
+                         ' to allow the use of the BRITE topology generator'),
+                   default='', dest='with_brite')
+
+def configure(conf):
+    conf.env['ENABLE_BRITE'] = False
+
+    if Options.options.with_brite:
+        if os.path.isdir(Options.options.with_brite):
+            conf.check_message("BRITE location", '', True, ("%s (given)" % Options.options.with_brite))
+            conf.env['WITH_BRITE'] = os.path.abspath(Options.options.with_brite)
+    else:
```

```
+        brite_dir = os.path.join('..','BRITE')
+        if os.path.isdir(brite_dir):
+            conf.check_message("BRITE location", '', True, ("%s (guessed)" % brite_dir))
+            conf.env['WITH_BRITE'] = os.path.abspath(brite_dir)
+        del brite_dir
+    if not conf.env['WITH_BRITE']:
+        conf.check_message("BRITE location", '', False)
+        conf.report_optional_feature("brite", "BRITE Integration", False,
+                                     "BRITE not found (see option --with-brite)")
+        return
+
+    test_code = '''
+#include "Brite.h"
+int main ()
+{
+  return 0;
+}
+'''
+    conf.env['DL'] = conf.check(mandatory=True, lib='dl', define_name='DL', uselib='DL')
+
+    for brite_module in ['.']:
+        conf.env.append_value('NS3_MODULE_PATH',
+                              os.path.abspath(os.path.join(conf.env['WITH_BRITE'], brite_module)))
+
+    conf.env['CPPPATH_BRITE'] = [
+            os.path.abspath(os.path.join(conf.env['WITH_BRITE'],'.')),
+            os.path.abspath(os.path.join(conf.env['WITH_BRITE'],'Models'))
+            ]
+    conf.env['LIBPATH_BRITE'] = [os.path.abspath(os.path.join(conf.env['WITH_BRITE'], '.'))]
+
+    conf.env['BRITE'] = conf.check(fragment=test_code, lib='brite', uselib='BRITE DL')
+    conf.report_optional_feature("brite", "BRITE Integration",
+                                         conf.env['BRITE'], "BRITE library not found")
+
+    if conf.env['BRITE']:
+        conf.env['ENABLE_BRITE'] = True
+        conf.env.append_value('CXXDEFINES', 'NS3_BRITE')
+        conf.env.append_value('CPPPATH', conf.env['CPPPATH_BRITE'])
+
+def build(bld):
+
+    module = bld.create_ns3_module('brite', ['network', 'core'])
+    module.source = [
+        ]
+
+    if bld.env['BRITE'] and bld.env['DL']:
+        module.uselib = 'BRITE DL'
+
+    headers = bld.new_task_gen('ns3header')
+    headers.module = 'brite'
+    headers.source = [
+        ]
+
+    if bld.env['ENABLE_BRITE']:
+        module.source.append ('helper/brite-topology-helper.cc')
+        headers.source.append ('helper/brite-topology-helper.h')
+
+    if bld.env['ENABLE_EXAMPLES'] and bld.env['ENABLE_BRITE']:
+      bld.add_subdirs('examples')
diff -r 7cd208d68f6c ns-3.12.1/src/wscript
--- a/ns-3.12.1/src/wscript     Sat Dec 17 12:11:54 2011 +0200
+++ b/ns-3.12.1/src/wscript     Sun Dec 18 14:16:43 2011 +0200
@@ -56,6 +56,7 @@
     'wimax',
     'lte',
     'mpi',
+    'brite',
```

```
     'topology-read',
     'energy',
     'tools',
@@ -69,6 +70,7 @@
     opt.sub_options('core')
     opt.sub_options('click')
     opt.sub_options('openflow')
+    opt.sub_options('brite')

     opt.add_option('--enable-rpath',
                    help=("Link programs with rpath"
@@ -93,6 +95,7 @@
     conf.sub_config('openflow')
     conf.sub_config('stats')
     conf.sub_config('visualizer')
+    conf.sub_config('brite')

     blddir = os.path.abspath(os.path.join(conf.blddir, conf.env.variant()))
     conf.env.append_value('NS3_MODULE_PATH', blddir)
diff -r 7cd208d68f6c ns-3.12.1/test.py
--- a/ns-3.12.1/test.py Sat Dec 17 12:11:54 2011 +0200
+++ b/ns-3.12.1/test.py Sun Dec 18 14:16:43 2011 +0200
@@ -54,6 +54,7 @@
     "ENABLE_PYTHON_BINDINGS",
     "ENABLE_CLICK",
     "ENABLE_OPENFLOW",
+    "ENABLE_BRITE",
 ]

 NSC_ENABLED = False
@@ -63,6 +64,7 @@
 ENABLE_TESTS = True
 ENABLE_CLICK = False
 ENABLE_OPENFLOW = False
+ENABLE_BRITE = False
 EXAMPLE_DIRECTORIES = []

 #
```