Visualization in the INET Framework
Motivation Example

Wireless11: Adding obstacles to the environment

packets received: 0
Implemented Visualizations for Communication

- Physical layer
  - Transmission & interference ranges, transmissions, propagating signals, receptions, and physical links
- Data link layer
  - Data links and link breaks
- Network layer
  - Routing tables and active network routes
- Transport layer
  - Transport connections and active transport routes
- Packet drops and numerical statistics
Other Implemented Visualizations

- Street maps and earth
- Playground with axis
- Physical objects and obstacle losses
- Network nodes and network connections
- Mobility and movement trajectory
- Module info strings
Visualization Goals

- De-couple visualization from simulation model
- Make it configurable in terms of *what* and *how*
- Support both 2D and 3D visualization
- Allow multiple different visualizations simultaneously
- First informative, then beautiful and interactive
Motivation Example Solved

Wireless11: Adding obstacles to the environment

packets received: 0
Motivation Example Extended

Wireless11: Adding obstacles to the environment

packets received: 0

hostA
hostR1
hostB
hostR2
hostR3
visualizer
configurator
radioMedium
figureHelper
physicalEnvironment
3D Example
How to Use Visualizers?

- Add visualizer submodules in the NED file
  ```
  network SimpleVisualizationExample
  {
    submodules:
    canvasVisualizer: RoutingTableCanvasVisualizer;
  }

  network WirelessTutorial
  {
    submodules:
    visualizer: IntegratedCanvasVisualizer;
  }
  ```

- Configure visualizers in the Ini file
  ```
  *.canvasVisualizer.destinationFilter = "/.*.destination"

  *.visualizer.dataLinkVisualizer.packetNameFilter = "UDPData*"
  *.visualizer.networkRouteVisualizer.packetNameFilter = "UDPData*"

  *.visualizer.osgVisualizer.sceneVisualizerType = "SceneOsgEarthVisualizer"
  *.visualizer.osgVisualizer.sceneVisualizer.mapFile = "boston.earth"
  *.visualizer.osgVisualizer.sceneVisualizer.playgroundShading = false
  *.visualizer.osgVisualizer.sceneVisualizer.playgroundColor = "#000000"
  *.visualizer.osgVisualizer.sceneVisualizer.playgroundOpacity = 0.1
  ```
What Parameters Visualizers have?

- Target canvas
- Turning various sub-visualizations on/off
- Filtering
  - Node filter, packet filter, etc.
- Styling
  - Colors, shapes, images, sizing, opacity, etc.
Where are the Visualizers?

- Visualizers can be found under ‘INET/src/visualizer’
- Simple visualizers are called something like
  - DataLinkCanvasVisualizer (2D)
  - DataLinkOsgVisualizer (3D)
- Combined visualizers exist for convenience
  - IntegratedCanvasVisualizer (2D)
  - IntegratedOsgVisualizer (3D)
  - IntegratedVisualizer (2D & 3D)
How do Visualizers Work?

- Visualizers are separate modules from network models
- They communicate with network models using
  - OMNeT++ signals (emit & subscribe)
  - C++ functions (lookup & call)
- Communication parameters
  - Subscription module paths for OMNeT++ signals
  - Module paths for C++ calls
Thank you for your kind attention!