COSMOS/ORBIT Tutorials (Introduction)
Welcome, Iian

**Quick Links**
- OnlineScheduler
- Status Page
- Change Password
- Change My Profile
- Log Out
- System Administration
- Manage Resources
- Manage Reservations
- Approve Reservations
- Create Blackout Times
- Manage Blackout Times

**Accounts Administration**
- Users
- Groups
- New User
- New Group

**Pending requests**
- Users
- Groups
- Remainder
- Send Reminders
- Send Approvals

**Mailing List**
- PI List
- User List

---

<table>
<thead>
<tr>
<th>Monday, 6/10/2013</th>
<th>12:00am</th>
<th>1:00am</th>
<th>2:00am</th>
<th>3:00am</th>
<th>4:00am</th>
<th>5:00am</th>
<th>6:00am</th>
<th>7:00am</th>
<th>8:00am</th>
<th>9:00am</th>
<th>10:00am</th>
<th>11:00am</th>
</tr>
</thead>
<tbody>
<tr>
<td>grid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>outw</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w1</td>
<td>Dry</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w9</td>
<td>Arrived</td>
<td>Arived</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Tuesday, 6/11/2013</th>
<th>12:00am</th>
<th>1:00am</th>
<th>2:00am</th>
<th>3:00am</th>
<th>4:00am</th>
<th>5:00am</th>
<th>6:00am</th>
<th>7:00am</th>
<th>8:00am</th>
<th>9:00am</th>
<th>10:00am</th>
<th>11:00am</th>
</tr>
</thead>
<tbody>
<tr>
<td>grid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>outw</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**Control Panel**
## Reservation System

### June 2013

<table>
<thead>
<tr>
<th></th>
<th>Mon</th>
<th>Tue</th>
<th>Wed</th>
<th>Thu</th>
<th>Fri</th>
</tr>
</thead>
<tbody>
<tr>
<td>grid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>outdoor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sb1</td>
<td>Dry</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sb2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sb3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sb4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sb5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sb6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sb7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sb8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sb9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>wimax</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table:

<table>
<thead>
<tr>
<th>Time</th>
<th>12:00am</th>
<th>1:00am</th>
<th>2:00am</th>
<th>3:00am</th>
<th>4:00am</th>
<th>5:00am</th>
<th>6:00am</th>
<th>7:00am</th>
<th>8:00am</th>
<th>9:00am</th>
<th>10:00am</th>
<th>11:00am</th>
</tr>
</thead>
<tbody>
<tr>
<td>grid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>outdoor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sb1</td>
<td>Dry</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Shradha</td>
<td>Shradha</td>
</tr>
<tr>
<td>sb2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Shradha</td>
<td>Shradha</td>
</tr>
<tr>
<td>sb3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Shradha</td>
<td>Shradha</td>
</tr>
<tr>
<td>sb4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Shradha</td>
<td>Shradha</td>
</tr>
<tr>
<td>sb5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Shradha</td>
<td>Shradha</td>
</tr>
<tr>
<td>sb6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Shradha</td>
<td>Shradha</td>
</tr>
<tr>
<td>sb7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Prasanthi</td>
<td>Prat</td>
</tr>
<tr>
<td>sb8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Shradha</td>
<td>Shradha</td>
</tr>
<tr>
<td>sb9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>AravindK</td>
<td>AravindK</td>
</tr>
<tr>
<td>wimax</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Shradha</td>
<td>Shradha</td>
</tr>
</tbody>
</table>

### Notes:
- Dry indicating dry.
- Striped cells may require special attention.

---

**Source:** COSMOS
Auto-approval

- Two stage algorithm:
  - "Early bird" – runs once a day (at 2 PM) and resolves conflicts and approves first two hours for all users for the next day
    - (e.g. if you ask for your first slot daily slot from 10-12 the next day, at 2 PM a day earlier you will know wheather you got it).
  - "Just in time" – for reservations made after 2 PM or for more than 2 hours per day per domain, the slots will be automatically approved at the beginning of the slot.

- Conflicts are resolved based on usage in the last three weeks
  - (the less you (ab)use it the more likely you are to get it 😊).

- Be aware of major (conference) deadlines
Status Page
First Exercise: Account Creation, Scheduler, Login (ssh) and Status Page
ORBIT Management Framework

OMF is a framework to **use** and **manage** experimental platforms (testbeds)

**Use**
- support “*experiment cycles*” & scientific rigor
- validation, accuracy & reproducibility

**Manage**
- ease operation and maintenance tasks
- optimize resource utilization inside / across testbeds

Written mostly in Ruby
OMF Workflow

OMF Tasks:
1. Controlling experiments
2. Managing the testbed
3. Measuring and collecting results
# OMF Command
(aka “NodeHandler”)

```bash
omf [SUBCOMMAND] [ARGUMENT]...
```

<table>
<thead>
<tr>
<th>Subcommand</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>omf help</td>
<td>Display the help for using omf commands.</td>
</tr>
<tr>
<td>omf exec</td>
<td>Execute an experiment script.</td>
</tr>
<tr>
<td>omf load</td>
<td>Load a disk image on a given set of nodes.</td>
</tr>
<tr>
<td>omf save</td>
<td>Save a disk image from a given node into a file.</td>
</tr>
<tr>
<td>omf tell</td>
<td>Switch a given set of nodes ON/OFF.</td>
</tr>
<tr>
<td>omf stat</td>
<td>Returns the status of a given set of nodes</td>
</tr>
</tbody>
</table>
Second Exercise: Basic OMF commands

`omf {tell, stat, load, save}`
OMF Experiment Description Language (OEDL)

- Domain-specific Language based on Ruby
- Two parts of experiment description (ED):
  - **Resource requirements and configuration**: specifies experimental resources
  - **Task description**: *state-machine* that enumerates tasks to perform
OEDL Commands

8 groups:

• Top-level commands
• Topology-specific commands
• Group-specific commands
• Prototype-specific commands
• Application-specific commands
• Execution-specific commands
• Resource Paths
• Testbed-specific commands
defProperty(name, initialValue, description)

- **name**: name of the property. This name will be used to refer to this property in any consecutive OEDL commands.
- **initialValue**: the initial value of the property. This also determines the type of the property.
- **description**: Textual description. Used in Experiment Controller's help message, as well as for the default web interface.

**Usage:**
```
defProperty('rate', 300, 'Bits per second sent from sender')
defProperty('packetSize', 1024, 'Size of packets sent from sender')
```
OEDL Top-level Commands: prop

prop.propName
prop.propName = newValue

- propName: Name of experiment property.
- newValue: New value to assign to the property.

Usage:
defProperty('rate', 300, 'Bits per second sent from sender') ...
'rate' => prop.rate
...
[500, 1000, 2000].each { |newRate|
  prop.rate = newRate 14
}
OEDL Top-level Commands: logging

- **arg1**: None or more strings to be logged

**Usage:**
- `info("Starting")`
- `debug(i, " resource(s) are up")`

**Note**: DEBUG and INFO log normal progress and can be ignored, while WARNING and ERROR report on abnormal behavior.
OEDL Top-level Commands: wait

wait(time)

• **time**: pause experiment execution for time seconds

Usage:
whenAllInstalled {
  ...
  [500, 1000, 2000].each { |newRate|
    prop.rate = newRate
    wait 30
  }
}
OEDL Topology Commands: defTopology

Used to specify topology consisting of a set of nodes and links each with certain characteristics

```ruby
defTopology( name , arrayOfNodes = nil , &block = nil )
```

- **name**: Name of the defined topology.
- **arrayOfNodes**: (optional) array of resources (e.g. nodes) to include in this topology.
  - the list of valid definition patterns are:
    - `[x,y]`: Describes a single node at location x@y
    - `[x1..x2, y]`: Describes a set of nodes along a line starting at x1@y and ending at x2@y. For instance, `[2..4, 5]` defines the nodes `[2,5], [3,5], [4,5].
    - `[x, y1..y2]`: Same as previous, but for the y coordinate.
    - `[x1..x2, y1..y2]`: This defines a rectangle area of nodes within the grid.
    - `[[x1,y1], [x2,y2], [x3,y3]]`: An arbitrary long list of single nodes.
- **block**: (optional) a block of commands that can be used to build/configure this topology.
### OEDL Topology Commands: `defTopology` (cont'd)

<table>
<thead>
<tr>
<th>Topology Sub-Commands</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>addNode(x,y)</code></td>
<td>Add node at location x@y to the topology.</td>
</tr>
<tr>
<td><code>removeNode(x,y)</code></td>
<td>Remove node at location x@y from the topology.</td>
</tr>
<tr>
<td><code>addLink(x, y, spec)</code></td>
<td>Adds a link between nodes x and y and configures it with the characteristics defined in the 'spec'. 'spec' is a hash with the following valid keys: {:rate, :per, :delay, :asymmetric}</td>
</tr>
<tr>
<td><code>removeLink(x, y)</code></td>
<td>Severs the link between nodes x and y.</td>
</tr>
<tr>
<td><code>size()</code></td>
<td>Return the number of nodes in this topology.</td>
</tr>
<tr>
<td><code>getNode(index)</code></td>
<td>Return the node at the position index in this topology. Return nil if index is greater than the number of nodes in the topology.</td>
</tr>
<tr>
<td><code>getFirstNode()</code></td>
<td>Return the node at the 1st position in this topology.</td>
</tr>
<tr>
<td><code>getLastNode()</code></td>
<td>Return the node at the last position in this topology.</td>
</tr>
<tr>
<td><code>getRandomNode()</code></td>
<td>Return a random node from this topology.</td>
</tr>
<tr>
<td><code>getUniqueRandomNode()</code></td>
<td>Return a unique random node from this topology. When all the available nodes in this topology have been drawn, this method will return nil and output a warning message to the console.</td>
</tr>
<tr>
<td><code>eachNode(&amp;block)</code></td>
<td>Execute the commands in block on each node within this topology.</td>
</tr>
<tr>
<td><code>setStrict()</code></td>
<td>Set the strict flag for this topology. By default, the strict flag is NOT set for a topology.</td>
</tr>
<tr>
<td><code>unsetStrict()</code></td>
<td>Clear the &quot;strict&quot; flag. By default, the strict flag is NOT set for a topology.</td>
</tr>
<tr>
<td><code>hasNode(x, y)</code></td>
<td>Return true if the node at location x@y is part of this topology, return false otherwise.</td>
</tr>
</tbody>
</table>
defTopology('test:topo:circle') { |t|
    nodeNum = 8
    xCenter = 10
    yCenter = 10
    radius = nodeNum
    # use simple 4-way algorithm to pick the nodes
    r2 = radius * radius
    t.addNode(xCenter, yCenter + radius)
    t.addNode(xCenter, yCenter - radius)
    (1..radius).each { |x|
        y = (Math.sqrt(r2 - x*x) + 0.5).to_i
        t.addNode(xCenter + x, yCenter + y)
        t.addNode(xCenter + x, yCenter - y)
        t.addNode(xCenter - x, yCenter + y)
        t.addNode(xCenter - x, yCenter - y)
    }
}
OEDL Group Commands: defGroup

```
defGroup( groupName, selector, &block = nil )
```

- `groupName`: name of the defined set of resources
- `selector`: selects the resources to be contained in this set. Group selector can be also defined with topology URI (i.e. set of nodes that form the topology)
- `block`: instructions for all resources in the group

**Usage:**
```
defGroup('sender1', [1, 1])    # set contains 1 resource
defGroup('sender2', [2, 1..8]) # set contains 8 resources [2,1], [2,2], ... [2,8]
defGroup('sender', ['sender1', 'sender2', [3, 1..8]]) {|node|    
  node.prototype("test:proto:sender", {
    'destinationHost' => '192.168.1.1',
    ...
  })
  node.net.w0.mode = "master" #802.11 Master Mode
}
```
# OEDL Group Commands: defGroup (cont’d)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>addApplication</td>
<td>Install an application on a node</td>
</tr>
<tr>
<td>exec</td>
<td>Execute a command on all nodes in this group.</td>
</tr>
<tr>
<td>image</td>
<td>Check whether a node boots in the required image. (not available in version 4.4 of the NH)</td>
</tr>
<tr>
<td>netmask</td>
<td>This is the network mask resource path.</td>
</tr>
<tr>
<td>onNodeUp</td>
<td>Execute a block of commands when a node is up.</td>
</tr>
<tr>
<td>pxImage(...)</td>
<td>Instructs a resource to boot from a network PXE image (recommended for expert users only).</td>
</tr>
</tbody>
</table>
OEDL Group Commands: group and allGroups

```java
group(groupSelector).command()
group(groupSelector).resource_path = value
group(groupSelector).resource_path {...}
```

- **groupSelector**: set of resources to use.
- **command**: command to run for that set.
- **resource_path**: is the parameter to be set
- **value**: is the value to assign to the resource path parameter

**Usage:**
```java
group('sender1').startApplications
group(['s1', 'r1']).net.w0.essid = "orbit"
allGroups.net.w0 { |w|
  w.essid = "orbit"
}
```
Resource Paths

- A resource path allows the access and the value assignment of a specific configuration parameter of a resource
- **Can be** used in any section of the ED.
- Follow a hierarchical organization:
  \[<\text{resource\_selector}.<\text{hierarchical\_path}>\]

**Example:**

\[allGroups.net.w0.type = g\]

- Set of all nodes defined in the ED are the resource
- **Network parameter** of the first wireless device
- **Type attribute** set to 802.11g
net - network resource path

- \( \{e0, e1\} \) Ethernet interface
  - \( \text{arp} = \text{true|false} \) En/disable ARP
  - \( \text{forward} = \text{true|false} \) Enable forwarding
  - \( \text{ip} = \text{address/netmask} \) IP address of interface
  - \( \text{up} = \text{true|false} \) En/disable interface

- \( \{w0, w1\} \) Wireless interface
  - All the above
  - \( \text{channel (intel only)} = 1..11; 36, 40, 44, 48, 52, 56, 60, 64, 149, 153, 157, 161 \)
  - \( \text{frequency (intel only)} = 2.412..2.462\text{GHz (5 MHz steps)}; 5.18\text{GHz (20MHz steps)} \)
  - \( \text{essid} = \text{arbitrary string} \)
  - \( \text{mode} = \text{master|managed|monitor, ad-hoc (intel only)} \)
  - \( \text{rts (atheros only)} = \text{packetSizeThreshold [bytes]} \)
  - \( \text{rate (intel only)} = 1, 5, 11; 6, 9, 12, 18, 24, 36, 48, 54 \)
  - \( \text{tx_power} = -12..15 \text{dBm (intel), 0..20 dBm (atheros)} \)
  - \( \text{type} = a/b/g \)
defProperty('res1', 'node1-1.grid.orbit-lab.org', "ID of sender node")
defProperty('res2', 'node1-2.grid.orbit-lab.org', "ID of receiver node")
defProperty('duration', 60, "Duration of the experiment")

defGroup('Sender', property.res1) do |node|
  node.addApplication("test:app:otg2") do |app|
    app.setProperty('udp:local_host', '192.168.0.2')
    app.setProperty('udp:dst_host', '192.168.0.3')
    app.setProperty('udp:dst_port', 3000)
    app.measure('udp_out', :samples => 1)
  end
  node.net.w0.mode = "adhoc"
  node.net.w0.type = 'g'
  node.net.w0.channel = "6"
  node.net.w0.essid = "helloworld"
  node.net.w0.ip = "192.168.0.2"
end

defGroup('Receiver', property.res2) do |node|
  node.addApplication("test:app:otr2") do |app|
    app.setProperty('udp:local_host', '192.168.0.3')
    app.setProperty('udp:local_port', 3000)
    app.measure('udp_in', :samples => 1)
  end
  node.net.w0.mode = "adhoc"
  node.net.w0.type = 'g'
  node.net.w0.channel = "6"
  node.net.w0.essid = "helloworld"
  node.net.w0.ip = "192.168.0.3"
end

onEvent(:ALL_UP_AND_INSTALLED) do |event|
  info "This is my first OMF experiment"
  wait 10
  allGroups.startApplications
  info "All my Applications are started now..."
  wait property.duration
  allGroups.stopApplications
  info "All my Applications are stopped now."
  Experiment.done
end
Focus on ultra high bandwidth, low latency, edge cloud
Open platform (building on ORBIT) integrating mmWave, SDR, and optical x-haul
1 sq mile densely populated area in West Harlem
Local community outreach
Research community:
  – Develop future experiments, provide input
  – (short term) get involved in the educational outreach

More information:

http://omf.orbit-lab.org  http://oml-doc.orbit-lab.org