Distributed Wireless Networking Framework for B3G Simulations

Ricardo Vaz · Martijn Kuipers · Mário Nunes
Outline

• Problem Statement
• Approach
• Demo Video
• Advantages/Disadvantages
• Conclusions
• Future Work
**Problem Statement (1)**

- **Scenario:** WiFi and WiMAX deployment, with vertical handovers.

- **What simulator to use:**
  - Simulator A: Excellent WiFi implementation, but no WiMAX
  - Simulator B: WiMAX only simulator

- **The presented framework allows you to use a “best-of-breed approach”, well almost.**
Approach - Architecture

Central Controller

- The simulators exchange information via the Central Controller and/or Simulation Plan
- Central Controller is a router/(de)multiplexer
- Simulation plan can be simple
Approach (2)

- The Central Controller is the clock-master for the simulators.
- The Simulation Plan informs the Central Controller of the next time period that can be simulated.
  - Either single events (period in the order of milliseconds);
  - Or multiple events (period in the order of seconds);
- The schedulers of the simulators is hijacked in order to control it remotely (outside of the actual simulator).
Approach - SOAP

- Information exchange between the simulators and the central controller, as well as, the simulation plan and the central controller is via webservices.

- SOAP are XML-based webservices, which pass information around on the network.

- Advantage:
  - Gsoap: open source C/C++ software development toolkit for SOAP/XML Web services and generic (non-SOAP) C/C++ XML data bindings takes care of the communication and conversions.
  - Extremely flexible.

- Disadvantage:
  - Messages tend to be verbose (especially binary data).
• In order to change as little as needed in the simulators, they are run inside “incubators”. A closed box, where all output can be caught and forwarded to the simulation planner.

• The incubator controls the process of the simulator and has a named pipe, which is used to hijack the scheduler.
Approach – Simple Client

• Simulation Plan uses synchronous operation (Simulation Plan is blocked, while waiting for an answer from the Central Controller)
  – This gives simplicity to the Simulation Plan, as no message bookkeeping is required, as is the case for asynchronous operation.

• Only a small set of messages is required:
  – Setup simulation (reserve simulators)
  – Config simulation (configure the simulators)
  – Start simulation (initialize the simulators)
  – Step (run the simulators for a specific simulation period)
  – Stop (Abort the simulation)

• Extra messages are needed to:
  – Pass special information to the simulator object
Demo Video
Advantages

- The framework has the following advantages:
  - Simulate larger networks (e.g., NS2 is single threaded, so performance is directly related to the size of the network), “virtually” multithreading the simulators.
  - Mix various simulators (e.g., mix Sphere (WiMAX) with NS2 and NS3).
  - Almost no changes are required to the simulators (minimum change is the control mechanism for the scheduler).
  - Control simulations remotely (e.g., from your lightweight laptop, while using a powerful computer for the actual simulations).
  - Quickly implement some functions in Matlab or Python (e.g., build a handover cost-function).
• Before getting too excited, there are also some noteworthy disadvantages:
  – There is the time-granularity vs. performance trade-off; i.e., smaller simulation periods have more communication over the network, whereas larger simulation periods might not give you the response you wanted. Solution: Dynamic time-steps.
  – Central Controller can become a communication bottleneck, especially when the number of simulators and/or simultaneous Simulation Plans increases.
  – If you need to get fine-grained control over some of the object inside a simulator, more changes are required to the original simulator.
Conclusions

- The framework allows us to combine various simulators to quickly simulate complex networks.
- It is not trivial to set the time-granularity.
- Simple Matlab or Python scripts can be used to evaluate and control the simulators.
- The Central Controller principle works fine as long as a small set of simulators and/or simulation plans are used.
- We can happily mix simulators and operating systems (e.g., Matlab on Windows, with Sphere on Linux and NS2 on OSX).
Future Work

• Solve the time-granularity problem by adjusting the time-step to the states of the simulators.

• Add heuristics to filter only influential actions (e.g., the WiMAX simulator is not interested in beacon information from WiFi).

• Decentralizing the Central Controller could allow for larger number of simulators, allowing even larger scale networks.
Thank you!
Any questions?