Towards Smart Grids: Simulation Framework for the Management of the Electric Grid

Introduction
The advent of a smart grid would allow today's electric power grid to be supported by an overlay communications infrastructure, leading to large improvements in efficiency, carbon footprint, stability and cost.

In addition to smart grid applications, an improved communications infrastructure is motivated by the changing landscape of today's electric grid: higher demand, need for wide-area controls, grid security, and market deregulation.

Smart grid applications involve information coming from potentially millions of sources, therefore this information needs to be managed in order to understand the status of the grid and to take the required actions.

The aim of this project is to lay the foundation of a simulation framework on which further studies can be conducted.

An existing implementation of a middleware framework whose generic principle will be used as a starting point is the GridStat middleware. This project is implemented using ns-3, a discrete-event network simulator.

GridStat Overview
GridStat at Washington State University is a middleware framework currently being researched, prototyped, and developed that provides flexible, robust, timely, and secure delivery of operational status information for the electric power grid.

A middleware is a class of software technologies that provide common abstractions across a distributed system, effectively hiding underlying complexities and heterogeneity. GridStat is a status dissemination middleware – a specialization of the publisher-subscribe model – that optimizes delivery of status data and manages subscriptions while providing quality of service (QoS) guarantees.

GridStat Architecture
GridStat’s role is to deliver status events from publishers to subscribers. It provides the following key abstractions:

- **Status Variables** – state events
- **Status Routers** – message forwarding
- **QoS Brokers** – management & path allocation
- **Publishers** – source of events
- **Subscribers** – destination of events

Results
To create the simulation framework, a suitable architecture was designed by mapping ns-3’s key entities to GridStat’s key abstractions. The user of the simulation framework can set up experiments by declaring simulation entities, defining a dynamic network topology, setting simulation parameters, and calling subscription request functions to set up multicast paths from publishers to subscribers.

There are multiple methods to gather data from the simulation: the built-in ns-3 logging system, the simulation framework’s functions for file output, and the tracing system to generate .pcap (packet capture) files.

The following table shows the simulation framework’s performance in running a simple configuration involving 1 QoS Broker, 2 Status Routers, 10 Subscribers, and 10 Publishers (each publishing p status values every 200ms.)

<table>
<thead>
<tr>
<th># of Total Subscriptions</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 (p = 1)</td>
<td>1 second</td>
</tr>
<tr>
<td>50 (p = 5)</td>
<td>5 seconds</td>
</tr>
<tr>
<td>100 (p = 10)</td>
<td>10 seconds</td>
</tr>
<tr>
<td>1000 (p = 100)</td>
<td>100 seconds</td>
</tr>
<tr>
<td>10000 (p = 1000)</td>
<td>750 seconds</td>
</tr>
</tbody>
</table>

Figure 2: Amount of time required to perform a 30-second simulation described above on an Intel Core i5 2.2 GHz machine – more subscriptions (and thus more packets) require more calculations.

Conclusion
This simulation framework models key functionalities of GridStat while maintaining extensibility and providing adequate performance. Future work includes implementing additional features and conducting studies regarding the architecture of data aggregation mechanisms.

Acknowledgements
I would like to thank to Faculty of Engineering for the SURE program, as well as my supervisor Prof. Labeau.