Building an AGL Telematics Profile Demonstration Platform

(Automotive Linux Symposium - July 2019)

Scott Murray and Matt Ranostay

scott.murray@konsulko.com,matt.ranostay@konsulko.com

Konsulko Group

About Us

- Scott Murray
  - Linux user/developer since 1996
  - Embedded Linux developer starting in 2000
  - Principal Software Engineer at Konsulko Group
- Matt Ranostay
  - Linux user/developer since 2005
  - Embedded Linux developer starting in 2008
  - Senior Software Engineer at Konsulko Group

Konsulko Group

Konsulko Group

- Services company specializing in Embedded Linux and Open Source Software
- Hardware/software build, design, development, and training services.
- Involved with AGL since 2015
- Based in San Jose, CA with an engineering presence worldwide
- https://konsulko.com

Konsulko Group

Agenda

- Telematics and the AGL Telematics profile
- Telematics demonstration platform requirements
- Modem selection and integration issues
- CAN bus adapter selection and integration issues
- Demonstration platform definition
- Demonstration application
- Image size reduction investigation
- Next steps

Konsulko Group

What do we mean by Telematics?

- There are very broad definitions of “telematics” that include basically all transmission of information
- For the automotive domain, it generally refers to technologies that transmit data to or from a vehicle
  - e.g. GPS navigation, integrated cell-phones, emergency warning systems, ADAS
- Other practical applications
  - Vehicle, trailer, or even container tracking
  - Fleet management
  - Carsharing
  - Usage based insurance (UBI)

Konsulko Group

The AGL Telematics profile

- Aims to provide a base platform to build images for headless (non-UI) devices
- Includes:
  - Application framework
  - CAN low-level binding
  - Network binding
    - connman, ofono
- Does not include the GPS binding, since it may not be required for all devices
- Provides agl-telematics-image
  - Not particularly useful on its own as is, base for custom images
- Available in Grumpy Guppy (7.0) and master / Happy Halibut (8.0)

Konsulko Group

agl-image-telematics.bb

```
SUMMARY = "A basic telematics image"

require agl-image-telematics.inc

LICENSE = "MIT"

IMAGE_INSTALL_append = "\
    profile-telematics \
    "
```

Konsulko Group

agl-image-telematics.inc

```
require recipes-platform/images/agl-image-boot.inc

inherit distro_features_check

REQUIRED_DISTRO_FEATURES = "3g"
```

Konsulko Group

packagegroup-agl-profile-telematics.bb

```
SUMMARY = "The middleware for AGL telematics profile"
DESCRIPTION = "The set of packages required for AGL Telematics Distribution"
LICENSE = "MIT"

inherit packagegroup

PACKAGES = "\
    packagegroup-agl-profile-telematics \
    profile-telematics \
    "

ALLOW_EMPTY_${PN} = "1"

...
```

Konsulko Group

packagegroup-agl-profile-telematics.bb (continued)

```
...

RDEPENDS_${PN} += "\
    packagegroup-agl-image-boot \
    packagegroup-agl-core-security \
    ${@bb.utils.contains('VIRTUAL-RUNTIME_net_manager','connman','connman
connman-client','',d)} \
    ${@bb.utils.contains("DISTRO_FEATURES", "3g", "libqmi", "", d)} \
    agl-login-manager \
    agl-service-can-low-level \
    agl-service-network \
    can-utils \
"

RDEPENDS_profile-telematics = "${PN}"
```

Konsulko Group

Telematics profile demonstration requirements

- Show how to extend the base profile with a customized image
- Use a 3G or LTE modem
- Use the CAN bus
- Potentially use GPS
- Simple application to demonstrate using the network, can low-level, and potentially GPS bindings
  - Along the lines of a generic vehicle monitor/tracker or potentially insurance company data collection style application
- Initially targeting smaller platforms like Raspberry Pi and Beaglebone Black
  - But should work on all AGL machine targets

Konsulko Group

Modem selection

- Desirable to use a modem that is:
  - Easily obtainable
  - Known to work with ofono and connman
  - Preferably USB connected to allow using on multiple platforms
- Initially using Sierra Wireless AirPrime 6 mPCIe modules
  - Available in Americas/EMEA (MC7455) and APAC (MC7430) variants
  - Purchasable via Amazon, Digi-key, etc.
  - Firmware and documentation available
  - Though mPCIe, can be used in cheap USB to mPCIe adapters
    - And can be later tested on boards such as UP^2, Hummingboard, etc.
  - LTE likely more futureproof, as carriers are shutting down 2G/3G to reuse spectrum
  - Well documented GPS support, dedicated active antenna an option

Konsulko Group

Modem selection (continued)

- Sierra Wireless MC7455 (Americas/EMEA):
  - https://www.amazon.com/gp/product/B0722HHYFQ/
  - https://www.digikey.com/product-detail/en/sierra-wireless/MC7455_1103578/1645-1057-ND
- Sierra Wireless MC7430 (APAC):
  - https://www.digikey.com/product-detail/en/sierra-wireless/MC7430_1103730/1645-1081-ND
- mPCIe to USB adapter:
  - e.g. https://www.amazon.com/gp/product/B01JGCSPEA
  - Note that the adapter provides a SIM slot
  - No provision for extra GPS antenna connector usually, so modification may be required to use one

Modem mPCIe to USB adapter

Konsulko Group

Other modem investigation

- Narrowband IoT (NB-IoT) modules
  - Also known as LTE Cat NB
  - Low bandwidth usage that fits between guard bands in LTE channels
  - Half duplex, up to 250 kbit/s, high latency (seconds)
  - Available modules from u-blox and SIMCom are not supported in ofono at present
  - Low data rates more suitable for simpler use cases like vehicle tracking
- LTE Cat M1 modules
  - Full duplex, up to 1 Mbit/s, reasonable latency (milliseconds)
  - Can support voice calls with VoLTE
  - As with NB-IoT, available modules have no support in ofono
  - Seems likely to be path in the future for a lot of telematics use cases, as well as emergency call use cases

Konsulko Group

Modem integration

- The Qualcomm chipset in the MC74xx required newer versions of libqmi and ofono in GG to work reliably, and some things enabled
- libqmi updated to 1.18.2 in GG
  - https://gerrit.automotivelinux.org/gerrit/#/c/19915/
- ofono updated to 1.24 in GG
  - https://gerrit.automotivelinux.org/gerrit/#/c/19925/
- “3g” needed to be added to DISTRO_FEATURES
  - https://gerrit.automotivelinux.org/gerrit/#/c/19731/
- Kernel configuration fragment needed to be added to enable USB modem drivers
  - https://gerrit.automotivelinux.org/gerrit/#/c/19697/

Konsulko Group

Modem integration (continued)

- Modem firmware likely will require updating out of the box
- Latest firmware is available from Sierra Wireless:
  - https://source.sierrawireless.com/resources/airprime/minicard/74xx/airprime-em_mc74xx-approved-fw-packages/
- Can be updated with the qmi-firmware-update from libqmi
- Useful guide located at:
  - https://sigquit.wordpress.com/2016/12/09/qmi-firmware-update-with-libqmi/
- Note that there are some carrier specific firmware versions in addition to generic versions
  - Generic versions should work, but you may need to experiment to know for sure
- Currently this has been left as a manual procedure

Konsulko Group

Modem integration (continued)

- Access Point Name (APN) configuration can be an issue
- ofono by default uses the provisioning database from the mobile-broadband-provider-info package
  - Maintained by GNOME project, seems a bit out of date for some providers
- Incorrect APN configuration in the package's serviceproviders.xml file may prevent a working connection
  - Can patch the file with known working APN configuration for the SIM to be used
  - Potentially can remove all APN configuration for the SIM provider and rely on the configuration read from the SIM
  - Solution will depend on carrier

Konsulko Group

Modem integration (continued)

- Example of tweaking serviceproviders.xml for the configuration:
  - APN: bmobile.ne.jp
  - Username: bmobile@4g
  - Password: bmobile
- This is for a prepaid SIM that can be bought at Bic Camera locally
- Note that in general testing has been done with prepaid SIMs, as they are readily available and affordable
  - A product would likely require a large volume purchase agreement with a carrier
  - There are specialized data plans from vendors targeting M2M and IoT

Konsulko Group

Example serviceproviders.xml patch

```
@@ -7531,11 +7531,11 @@ conceived.
            <name>b-mobile</name>
            <gsm>
                  <network-id mcc="440" mnc="10"/>
-                 <apn value="dm.jplat.net">
-                       <plan type="postpaid"/>
+                 <apn value="bmobile.ne.jp">
+                       <plan type="prepaid"/>
                        <usage type="internet"/>
-                       <name>u300</name>
-                       <username>bmobile@u300</username>
+                       <name>Visitor SIM</name>
+                       <username>bmobile@4g</username>
                        <password>bmobile</password>
                  </apn>
            </gsm>
@@ -7559,7 +7559,6 @@ conceived.
                  <network-id mcc="440" mnc="02"/>
                  <network-id mcc="440" mnc="03"/>
                  <network-id mcc="440" mnc="09"/>
-                 <network-id mcc="440" mnc="10"/>
                  <network-id mcc="440" mnc="11"/>
```

Konsulko Group

CAN bus adapter selection

- As with the modem, USB connected deemed preferable to allow flexibility
- Tested with a couple of different devices:
  - USB-CAN Plus ISO
    - Bit older, well documented
    - Needs slcand daemon for SocketCAN support
    - Uses a different DB9 CAN pinout than a lot of newer adapters, OBD-II adapter cable harder to find
  - Microchip CANbus Analyzer
    - Bit more expensive
    - Software configurable termination
    - Direct SocketCAN support, avoids needing to use slcand

Konsulko Group

CAN bus adapter selection (continued)

- VScom USB-CAN Plus ISO:
  - http://www.electronicnetwork.com/en/usb-can-plus-iso.html
  - https://shop-visionsystems.de/en/usb-to-can-bus/430-usb-can-plus-iso.html
- USB-CAN Plus DB9 / OBD-II cable (EasySync OBD-M-DB9-F-ES):
  - https://www.mouser.com/ProductDetail/EasySync/OBD-M-DB9-F-ES?qs=%2Fha2pyFadujDlkq35NWa0Zot3gKfnxUIBy5tgxPV7wd6AQNaRDvj%2Fw%3D%3D
- Microchip CAN bus analyzer:
  - https://www.amazon.com/gp/product/B00DK2AJXS
  - https://www.digikey.com/product-detail/en/microchip-technology/APGDT002/APGDT002-ND

Konsulko Group

CAN bus adapter integration

- Devices with native SocketCAN support need something to bring the CAN interface up and configure its speed
  - Using systemd-networkd is one option for this
  - systemd changes to allow this backported into telematics profile in GG, available in core profile in master/HH
    - https://gerrit.automotivelinux.org/gerrit/#/c/20173/
  - Slight issue that ATM the CAN device name and speed are hard-coded in canbus-can0.network
  - Could potentially switch to something based on udev rules or build-time configuration
- Devices that require slcand be used to provide SocketCAN support have a similar problem
  - Demo image adds udev rule and systemd service unit to support the USB-CAN Plus ISO

Demonstration platform definition

- Extends agl-image-telematics
- Adds USB CAN adapter configuration
- Adds GPS binding
  - and gpsd via dependencies
  - Some configuration tweaks required for out of the box GPS
- Adds the demo application
  - And some dependencies that will be discussed later
- Adds some optional debugging tools
- In AGL Gerrit: https://gerrit.automotivelinux.org/gerrit/AGL/meta-agl-telematics-demo

Konsulko Group

agl-telematics-demo-platform.bb

```
DESCRIPTION = "AGL Telematics Demo Platform image."

require agl-telematics-demo-platform.inc

LICENSE = "MIT"

IMAGE_FEATURES_append = " \
    "

IMAGE_INSTALL_append = " \
    packagegroup-agl-telematics-demo-platform \
    "
```

Konsulko Group

agl-telematics-demo-platform.inc

```
# Base image
require recipes-platform/images/agl-image-telematics.inc
```

Konsulko Group

packagegroup-agl-telematics-demo-platform.bb

```
SUMMARY = "The software for AGL telematics profile demo platform"
DESCRIPTION = "A set of packages belonging to the AGL telematics demo
platform"

LICENSE = "MIT"

inherit packagegroup

PACKAGES = "packagegroup-agl-telematics-demo-platform"

ALLOW_EMPTY_${PN} = "1"

RDEPENDS_${PN} += "packagegroup-agl-profile-telematics"

...
```

Konsulko Group

packagegroup-agl-telematics-demo-platform.bb (continued)

```
...

AGL_APPS = "telematics-recorder"

AGL_APIS = "agl-service-gps"

RDEPENDS_${PN}_append = " \
    gpsd \
    sw-gpsd-udev-conf \
    usb-can-udev-conf \
    ${@bb.utils.contains('DISTRO_FEATURES', 'agl-devel', 'ofono-tests
gps-utils' , '', d)} \
    ${AGL_APPS} \
    ${AGL_APIS} \
"
```

Konsulko Group

Notes on GPS device configuration

- USB attached GPS devices such as the ones exposed by the QMI driver for the Sierra Wireless MC74xx may not have stable device names depending on other attached hardware and probe order
- This can be worked around by using an existing mechanism in the gpsd package's systemd units
  - A udev rule triggers a systemd unit that calls gpsdctl to register or unregister the device
  - Can use the pre-existing 60-gpsd.rules file as a guide on writing rules for a new device

Konsulko Group

Notes on GPS device configuration (continued)

- The GPS support in the Sierra Wireless modems may need to be manually enabled, see:
  - https://source.sierrawireless.com/resources/legato/howtos/nmeaport/
- Additionally, NMEA output needs to be enabled at boot by writing to the associated device
  - Some information at above link
  - https://forum.sierrawireless.com/t/read-mc7455-gps-data-on-ubuntu/8683/7
  - Simple approach is to use gpsd's device hook script to write to the device as needed
- Dedicated antenna port may need to be enabled as well
  - https://techship.com/faq/basic-standalone-gnss-gps-usage-guide-for-sierra-wireless-airprime-em-mc74-series-cellular-modules/

Konsulko Group

Demonstration application

- Checks modem availability and potentially enables it if necessary using the network binding
- Subscribes to CAN bus messages using the CAN low-level binding
  - Currently just vehicle and engine speed
- Optionally gets current location using the GPS binding when processing events
- Sends event data to a MQTT broker
- In AGL Gerrit: https://gerrit.automotivelinux.org/gerrit/apps/agl-telematics-demo-recorder

Konsulko Group

MQTT?

- Message Queuing Telemetry Transport
- An ISO standard publish-subscribe messaging transport
- Designed to require small code footprint and work over potentially high-latency, low-bandwidth links
- Machine to Machine (M2M), Internet of Things (IoT) focused
- Runs on top of TCP/IP, and requires a broker to publish to
- Well supported on Linux via mosquitto and other packages
    - Recipe for mosquitto available in meta-openembedded layer

Why MQTT?

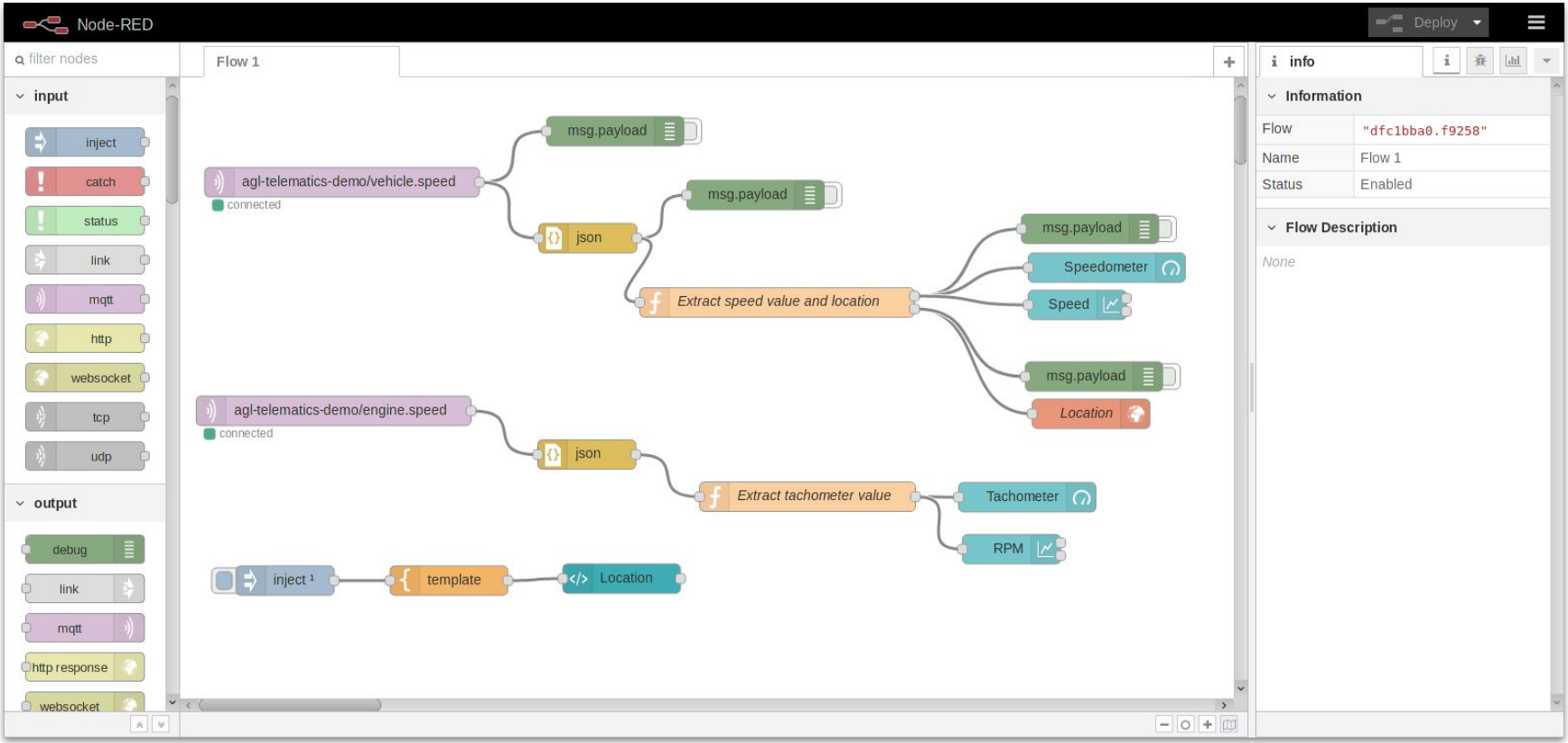
- Potentially could be used in a product
- Simple, and easy to configure for testing
- Allows flexibility for server side processing demos
  - potentially multiple demo apps can subscribe to the broker
- Widely supported by IoT dashboards, etc.
  - e.g. Node-RED
- Can be easily replaced with some other V2C messaging if desired

Konsulko Group

Node-RED Dashboard

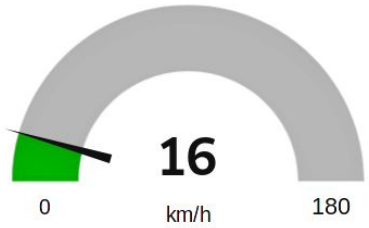
- Node-RED is a visual programming (“flow based programming”) tool originally from IBM for wiring together hardware devices and online services for the IoTs
  - https://nodered.org/
- It allows easily wiring up a demonstration dashboard with MQTT sources
- Demo configuration uses additional modules:
  - node-red-dashboard: https://flows.nodered.org/node/node-red-dashboard
  - node-red-contrib-web-worldmap: https://flows.nodered.org/node/node-red-contrib-web-worldmap

Konsulko Group

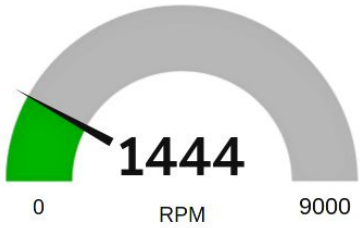


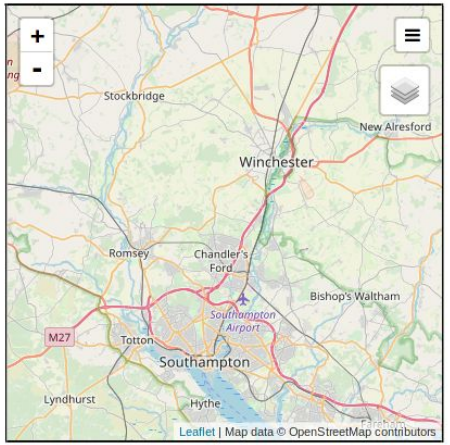
Current

Speedometer



Tachometer





Notes on integration issues

- GPS binding would sometimes timeout on gpsd access at start up due to slow startup of the modem stack
  - Fix was to use a longer timeout as some other sample gpsd clients use
- In-vehicle testing proved problematic
  - Looked like either the Raspberry PI 3 USB stack or USB-CAN Plus had issues with high rate of CAN messages from the OBD-II port, or potentially an issue with power supply under load
  - Have not had a chance to go back and retest
- It’s easy to trigger rate-limiting on some free MQTT brokers
  - The resulting behavior was initially very puzzling, as there was not an obvious error message
  - Solved by switching brokers and being more careful with limiting message rates
  - Actual insurance company ODB-II recorders for UBI store locally and use upload mechanisms like FTP for batch updates

Notes on integration issues (continued)

- The ofono test utilities list-modems and list-contexts, located in /usr/lib/ofono/test are very useful when debugging connectivity issues
- As well, “gpspipe -r” is useful when debugging GPS issues, as it shows incoming NMEA messages (or lack thereof)

Konsulko Group

Image Size Reduction

- Strong interest in reducing image size to reduce system storage requirements and ease upgrades
- The current raspberrypi3 demo image is ~250 MB
- One potential low-hanging fruit is disabling package management support
  - Image decreases to ~165 MB, saves 85 MB
  - Lose the ease of installation of rpm's, but the size saving is large
- Other potential simple changes
  - Remove OpenSSH, or replace it with dropbox (saves ~6 MB)
  - Consider making Bluetooth and Wifi support optional (currently part of DISTRO_FEATURES), likely saves a few MB
  - Investigate whether Unicode and other native language support can be disabled
    - libicu and libicudata are ~30 MB

Konsulko Group

Image Size Reduction (continued)

- There are more extreme changes potentially possible
- Removing connman and ofono in favor of systemd-networkd
  - agl-service-network binding would need substantial rework to support something other than connman
  - Would be use case dependent, complicating a lot of build and runtime code
- Switch to musl C library instead of glibc
  - Not clear if entirely a win for AGL as it will be hard to make images small enough where the 5+ MB saving will be justified
  - Potential complication in AGL with respect to application binary compatibility
    - But that may be less important in telematics use cases
  - Experiments with building the agl-telematics-demo-platform image determine that Cynara is a bit of a blocker, it has some glibc specific code that would need to be reworked
  - Another issue is upstream OpenEmbedded may stop supporting systemd with musl

Konsulko Group

Next steps

- Build and test on other platforms
  - Beaglebone Black next target, currently a slight BSP issue holding it back
- Add support for other CAN adapter hardware
  - PiCAN 2 HAT for Raspberry Pi
    - https://copperhilltech.com/pican-2-can-interface-for-raspberry-pi-2-3/
    - Required support present in GG and master / HH, needs to be tested
  - SanCloud automotive cape for Beaglebone Black or the Beaglebone Enhanced
    - https://www.sancloud.co.uk/?p=1192
    - Provides mPCIe socket for modem, and OBD-II connector
    - Potentially getting one soon to test
- Potentially support other modems
  - Need to find one that does not require significant ofono modification

Konsulko Group

Next steps (continued)

- Investigate size reduction further, upstream changes for package management removal from default DISTRO_FEATURES definition
- Implement some UBI style events in the demo application
  - e.g. speed past a threshold, detect hard-braking
  - Potentially add a separate UBI mode with local data storage and batch upload data transfer
- Improve demo application CAN usage, make filtering configurable
- Investigate CAN simulators further
  - Using CAN dump files with canplayer works, but is somewhat limiting
  - Have not yet found a lightweight, simple to set up FOSS simulator
  - Currently using a simple Python script

Questions?