You are here

GENIVI’s Location Based Services for Embedded Automotive
Let me get this out of the way;

I don’t know if HERE will be coming to GENIVI
While some automakers have delegated navigation features to a connected smartphone, many others still see essential value in proposing in-dash, embedded solutions from the automaker. The latter understand that automobiles are about mobility and navigation, and that these functions are as core to the modern automobile as the engine or the chassis. Moreover, navigation becomes a must-have in an autonomous vehicle. Would you let your smartphone drive your car?
“To guarantee people’s safety, autonomous vehicles, need a sophisticated ADAS systems along with reliable positioning information with an accuracy below 10 cm., about the size of an orange."
Why car makers want standard APIs

- Time to market
- The possibility to reuse code and to create systems specified independently from hardware
- The “commodity” layer of the vehicle ought to have cost transparency and competition in the ecosystem
Overview of GENIVI APIs

LBS APIs are

- Navigation Core
  - Location Input
  - Route Calculation
  - Route Guidance
    - Map Viewer, including Map Rendering and View Control
- POI (Point of Interest) search interface with a content access module to extend searching to external data sources
- Traffic Info, providing a Traffic Incidents Service
- Vehicle Positioning
  - GNSS (Global Navigation Satellite Service)
  - SNS Sensor service, used by Geolocation software for Dead Reckoning
  - Enhanced Position Service
Sensors

- Gyroscope
- ‘wheel ticks’ which measure speed and can help with direction and ‘dead reckoning’. One tick generally is one wheel revolution.
- Global Navigation Satellite System
  - Updated with at least 1Hz frequency
  - Position expressed as WGS 84 altitude, longitude and latitude in tenth of microdegree (degree x 10^-7)
  - Course speed in meters per second
  - Climb, relative heading to true north expressed in degrees
  - Timestamp and date in UTC
Navigation

GENIVI created a proof of concept that separates out a typical navigation application into a map viewer (realized in QML) and a navigation core (navit)

Location and other input plugins were provided to navit upstream, dbus bindings were created

1. genivi-navigationcore-locationinput.xml
2. genivi-navigationcore-routing.xml
3. genivi-navigationcore-guidance.xml
4. genivi-navigationcore-session.xml
5. genivi-mapviewer-session.xml
6. genivi-mapviewer-mapviewercontrol.xml
Fuel Stop Advisor

Proof of concept exercising numerous parts of GENIVI subsystems

Enhanced tank distance based on the fuel consumption on the route ahead

Warning if destination not reached

Proposal of reroute to a refill station
Fuel Stop Advisor

- QML
- FSA
- dbus
- Navi Core plugin
- Point of Interest plugin
- Map Viewer plugin
- Navit
- Pol search
- sqlite3
- Open Street Map
Map viewer

dbus interface provided;

- org.genivi.mapviewer.MapViewerControl

Entire dbus interface documented and available online; http://git.projects.genivi.org/?p=lbs/navigation.git;a=blob_plain;f=doc/map-viewer/MapViewerAPI.pdf;hb=f0ddb754ad4e16d8f650485a610818c06e0ceac3
W3C positioning PoC

The PositionWebService is a simple proof of concept (PoC) showing how positioning information provided over D-Bus by the GENIVI EnhancedPositionService can be accessed within a web browser.

This PoC was developed to investigate how to match the already defined positioning dbus interface with the Web API being defined by the W3C.

The translation D-Bus <-> JavaScript is realized using a FireBreath NPAPI plugin.
How do I get the software?

GENIVI DEMO PLATFORM
GENIVI DEMO PLATFORM

Yocto or Baserock based image built as both a virtual image or for target hardware.

Supported targets: Renesas Porter and Koelsch boards, Nvidia Jetson, Intel Minnowboard, and others in development

Uses LTSI kernel, systemd, and includes a complete suite of automotive specific software including lifecycle, layer management, diagnostic log and trace
GDP

- Fully FOSS, all source code available
  http://wiki.minnowboard.org/Where_to_buy
- Working towards a continuous integration and testing system as well as pre-built images
Challenges

What we’re working on
- HMI SDK specific to navigation
- Overall ADK being built
- Continuous Integration
- Automated testing

Nice to have
- Navigation engine (navit doesn’t look it is maintained)
- Complete framework bindings into HMI, i.e. GTK+ and Qt bindings
Future scenarios

- Smart tire sensors
- Web based APIs via W3C
- Vehicle 2 Vehicle (V2V) and Vehicle 2 Infrastructure (V2I)
- ?