Preamble
The Navigation features are part of Location Based Services (LBS). LBS means a piece of code that uses geolocalized data, either the location of the vehicle itself, or the location of content (map, POI...).

The Location-Based-Services Expert Group (LBS-EG), as part of the GENIVI Alliance, defines Application Programming Interface (API), Proof Of Concept (POC) and application (APP), that are included into the GENIVI compliant, Open Source, automotive IVI software stack.

The "In Vehicle Infotainment (IVI) Navigation" set of projects host the whole set of deliverables provided by the LBS-EG. The APIs and code standardized by GENIVI are covering non-differentiating features, that are shared by the whole car manufacturer's community. This wiki gives all needed explanations that allow to use the APIs into a concrete product.

Scope
The IVI-Navigation domain cover basic features:

Vehicle Positioning:

- Positioning, including GNSSService, SensorsService and EnhancedPositionService, tools for doing Dead Reckoning

Basic navigation:

- Navigation Core, including Location Input, Route Calculation, Route Guidance
- Map Viewer, including Map Rendering and View Control

Basic extensions:

- Point Of Interest (POI), providing a POI Search interface with a content access module mechanism allowing to extend the search to external data sources
- Traffic Info, providing a TrafficIncidentsService

HMI extension:

- Speech Services (output, input and dialog).

Projects
All the requirements and use cases can be found into the following wiki pages:

- IVI_Positioning Location of the vehicle, including dead reckoning.
- IVI_NavigationAndMapViewer Basic features of a navigation.
- IVI_POIService POI service features (search and manage).
- IVI_TrafficInfo Traffic info service features.
- IVI_SpeechInputAndOutput Speech features for the whole IVI features.
Interfaces

Positioning

Positioning - SensorsService API
Positioning - GNSSService API
Positioning - Constants
Positioning - EnhancedPosition API
Positioning - PositionFeedback API
Positioning - Configuration API

Navigation

NavigationCore - Constants
NavigationCore - Session API
NavigationCore - Guidance API
NavigationCore - Routing API
NavigationCore - LocationInput API
NavigationCore - Configuration API
NavigationCore - MapMatchedPosition API
MapViewer - MapViewerControl API
MapViewer - MapViewerConfiguration API
MapViewer - MapViewerSession API
MapViewer - MapViewer Constants

TrafficIncidentsService

TrafficService - TrafficIncidents API
TrafficService - TrafficIncidentsFeed API

POIService

POIService - Constants
POIService - POISearch API
POIService - POIContentAccess API
POIService - POIContentAccessModule API

Available material

In order to publish an API into the GENIVI compliance, it’s required to provide use cases, requirements and POC. The LBS-EG APIs are also promoted by a demonstrator, that is running on the so-called GENIVI Demo Platform (GDP). The demonstrator is called Fuel Stop Advisor (FSA).

Proof of concepts

What is required to build and run the code from scratch

<table>
<thead>
<tr>
<th>Status of the Proof Of Concepts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
</tr>
<tr>
<td>Positioning</td>
</tr>
<tr>
<td>Traffic incidents service</td>
</tr>
<tr>
<td>NavigationCore and MapViewer</td>
</tr>
<tr>
<td>POI Search</td>
</tr>
<tr>
<td>POI CAM with add/remove feature</td>
</tr>
<tr>
<td>Fuel Stop Advisor</td>
</tr>
</tbody>
</table>

Demonstrator

The main APIs are implemented by the Fuel Stop Advisor (FSA) demonstrator, running under the Genivi Demo Platform (GDP) and tested on two boards, Intel based (MinnowBoard MAX) and ARM based (R-Car M2 and Porter).
**GENIVI Workflow**

Here's the workflow put in place in GENIVI, and the deliverables that are covered by IVI-Navigation:

- **Stakeholder needs**: OEMs give their needs for the LBS domain and some rationales
- **Vehicle requirements**: The team defines top level requirements
- **Use cases**: Typical use cases in order to make it more understandable (and to promote the features)

According to the architecture, especially the split into components:
- **Software platform requirements**: requirements mapped on the components
- **APIs**: the APIs that allows to implement the requirements

For each API defined, there's also a proof of concept that implements the main features and validates the dynamic behaviour.

**Top level use cases**

There's a significant bunch of use cases and requirements for Navigation, but, in a first approach, we can consider that the main use cases could be counted on the fingers of two hands:

![Use case diagram](image)

**Actors**

<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>A person operating a GENIVI based IVI system</td>
<td>Driver or passenger of a car</td>
</tr>
</tbody>
</table>
Use cases

All the use cases are detailed into the project pages, but let's have a look at the main ones:

<table>
<thead>
<tr>
<th>ID</th>
<th>Title</th>
<th>Description</th>
<th>Subpart</th>
</tr>
</thead>
<tbody>
<tr>
<td>UC-NAV-GDN-004</td>
<td>guide to a destination</td>
<td>Provide guidance instructions to reach a destination</td>
<td>Guidance</td>
</tr>
<tr>
<td>UC-NAV-ROU-006</td>
<td>initiate a route calculation</td>
<td>Get a destination, calculate route</td>
<td>Routing</td>
</tr>
<tr>
<td>UC-TRI-002</td>
<td>warn of an event around a location</td>
<td>Get the event on the route (traffic, weather ..)</td>
<td>Traffic Info</td>
</tr>
<tr>
<td>UC-MAP-002</td>
<td>display vehicle position</td>
<td>Display the position of the vehicle</td>
<td>Map Viewer</td>
</tr>
<tr>
<td>UC-POS-001</td>
<td>get position</td>
<td>Get the position of the vehicle</td>
<td>Positioning</td>
</tr>
<tr>
<td>UC-MAP-003</td>
<td>display map and position</td>
<td>Display a map with relevant positioning</td>
<td>Map Viewer</td>
</tr>
<tr>
<td>UC-LBS-001</td>
<td>explore the surroundings of a location</td>
<td>Get all the location based content for a given position</td>
<td>Location based services</td>
</tr>
</tbody>
</table>

Quick links

- Positioning git repository can be found here
- Navigation git repository can be found here
- Navigation application git repository can be found here
- Bugzilla tickets for bug tracking can be found here

Components

The components and the APIs hosted by the LBS-EG are displayed into the following diagram:

Current state of Franca Interfaces
The migration to Franca is on-going. It has been decided by the LBS-EG team to maintain in parallel the Franca IDL files and the DBus xml files. Important notice: For compatibility with the Common-API tool chain, some rules are applied:

- 'enumeration' type in Franca matches with 'i' in DBus (int32_t)
- 'union' type in Franca matches with '(yv)' in DBus (structure(uint8_t, variant))
- 'client' is a reserved word in Common-API, so it's prohibited
- 'attributes' is a keyword in C++ and is detected as an error by Xtext checker of Franca, so it's prohibited

The current version of the fidl files can be found under the folder api/franca

Preliminary work with W3C

A working page is available here to evaluate what is covered/not covered by the GENIVI APIs for the LBS domain.

Recent space activity

Space contributors

- Gunnar Andersson (old account, disabled) (1420 days ago)
- Brent Soles (1496 days ago)
- Nicholas Contino (1637 days ago)
- Philippe Colliot (1741 days ago)
- Nicholas Contino (1911 days ago)