Genivi Demo Platform Hands-on
Location-Based Services
Fuel Stop Advisor

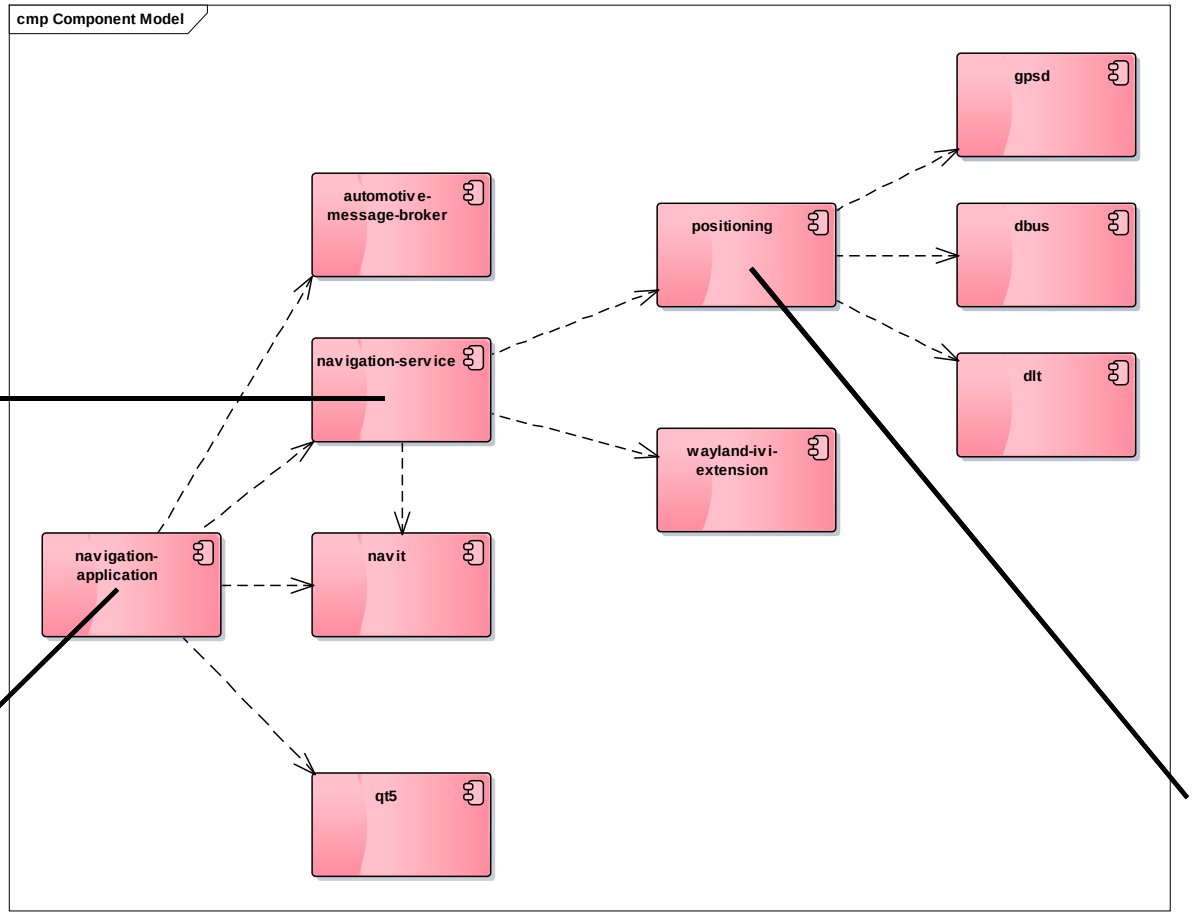
Philippe Colliot - Location-Based Expert
Matthias Bloch - Multimedia Software Architect

PSA PEUGEOT CITROËN

How to build FSA (Yocto recipes)

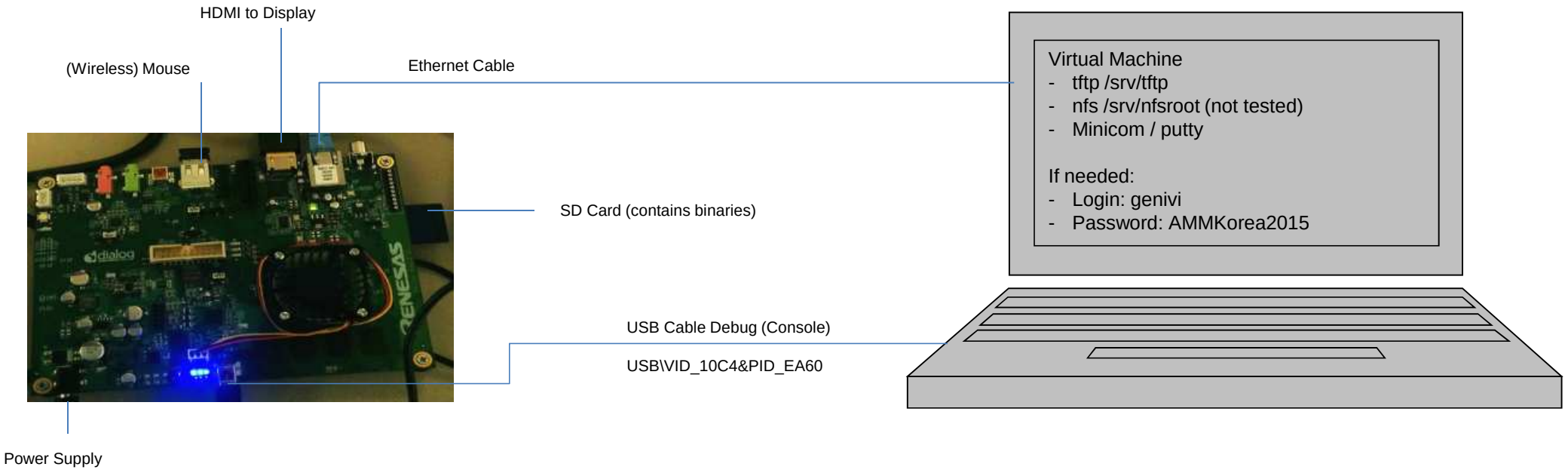
navit plugins
poi-server

fuel-stop-advisor
(genivilogreplayer)
hmi-launcher



enhanced-position-service
log-replayer

Setup Porter



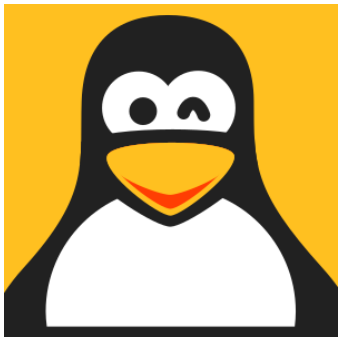
How to launch FSA ?

1. New instance dbus session
2. automotive-message-broker (ambd)
3. poi-server
4. enhanced-position-service
5. navit
    1. navit_genivi_mapviewer.xml
    2. navit_genivi_navigationcore.xml
6. fuel stop advisor
7. hmi-launcher
8. Surfaces / Layer Management: LayerManagerControl

Confused ?

How to launch FSA ?

1. New instance dbus session
2. automotive-message-broker (ambd)
3. poi-server
4. enhanced-position-service
5. navit
    1. navit_genivi_mapviewer.xml
    2. navit_genivi_navigationcore.xml
6. fuel stop advisor
7. hmi-launcher
8. Surfaces / Layer Management: LayerManagerControl

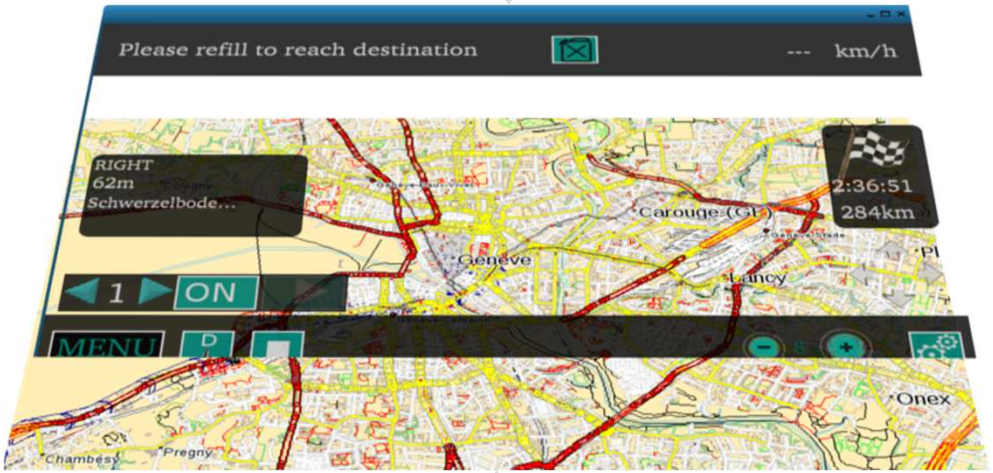


There is a script (temporary until automatic) to launch everything.

Surfaces / Layers Management

LayerManagerControl

Qtwayland Surface #: pidof `hmi-launcher`+ 8000
Layer #: 700




```
PID=`pidof hmi-launcher`
HMI_LAYER=700
FSA_LAYER=600

HMI_SURFACE=`expr 8000 + $PID`
LayerManagerControl set surface $HMI_SURFACE destination region 0 0 800 412
LayerManagerControl set surface $HMI_SURFACE visibility 1
LayerManagerControl set layer $HMI_LAYER render order $HMI_SURFACE
LayerManagerControl set layer $HMI_LAYER visibility 1
LayerManagerControl set layer $FSA_LAYER visibility 1
LayerManagerControl set screen 0 render order ...$FSA_LAYER,$HMI_LAYER
```

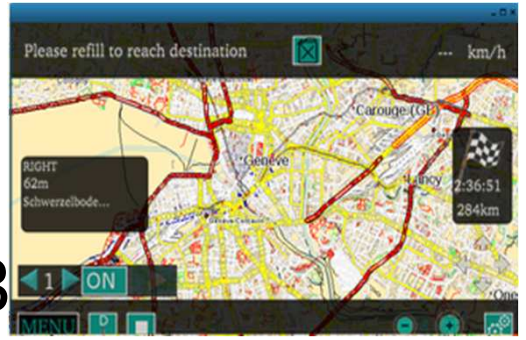
OpenGL Surface #: 601 (fixed)
Layer #: 600

School cases: Think about what happen if …

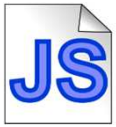
#1

#2

#3



#1

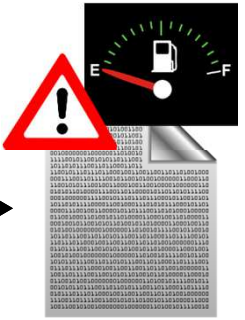


genivi.js
constants.js

```
...
function demonstrator_message(par, iface, func, args)
{
        return
par.message("org.genivi.demonstrator."+iface,"/org/genivi/demonstrator/"+ifac
e,"org.genivi.demonstrator."+iface, func, args);
}

function fuel_stop_advisor_message(par, func, args)
{
        return demonstrator_message(par, "FuelStopAdvisor", func, args);
}
...
```

dbus



fuel-stop-advisor

```
...
std::map< uint16_t, ::DBus::Variant >
GetTripData(const uint8_t& number)
{
   ...
}...
```

Starting point: TripComputer.qml

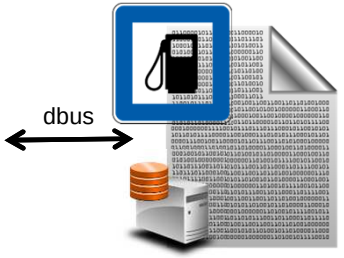
```
...
res=Genivi.fuel_stop_advisor_message
(dbusIf,"GetTripData",["uint8",tripnr-1]);
...
```

#2

genivi.js
constants.js

```
...
function poi_message(par, iface, func, args)
{
    return
par.message("org.genivi.poiservice."+iface,"/org/genivi/poiservice/"+iface,"o
rg.genivi.poiservice."+iface, func, args);
}

function poisearch_message(par, func, args)
{ //session handle sent
    return poi_message(par, "POISearch", func,
poisearch_handle(par).concat(args));
}
...
```



poi-server

```
...
void poiSearchServer::StartPoiSearch
(const handleId_t& poiSearchHandle,
const std::string& inputString, const
uint16_t& sortOption)
{
    ...
}
...
```

Starting point: POI.qml

```
...
Genivi.poisearch_message
(dbusIf,"StartPoiSearch",
["string","","uint16",Genivi.POISERVICE_SORT_BY_DISTANCE]);
...
```

Genivi API dbus connection


```
<node xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="/org/genivi/whatever“
xsi:noNamespaceSchemaLocation="introspect.xsd">
  <interface name="org.genivi.Whatever.WhateverControl">
    <version>3.1.0 (03-03-2014)</version>
    <doc>
       <line>WhateverControl = This interface offers functions to control the Whatever</line>
    </doc>
    <method name="GetVersion">
      <doc>
        <line>GetVersion = This method returns the API version implemented by the server application</line>
      </doc>
      <arg name="version" type="(qqqs)" direction="out">
        <doc>
          <line>version = ...</line>
        </doc>
      </arg>
    </method>
    <method name=“InterfaceFunction_X">
      <doc>
        <line> InterfaceFunction_X = This method ...</line>
      </doc>
      <arg name=“arg_Y" type="u" direction="in">
        <doc>
          <line>arg_Y = Argument Y ...</line>
        </doc>
      </arg>
      <error name="org.genivi.mapviewer.WhateverControl.Error.NoMoreWhateverInstanceHandles">
        <doc>
          <line>This error is generated if no more whatever instance handles are available</line>
        </doc>
      </error>
   </method>
  </interface>
</node>
```

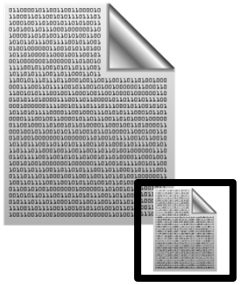
#3

genivi.js
constants.js

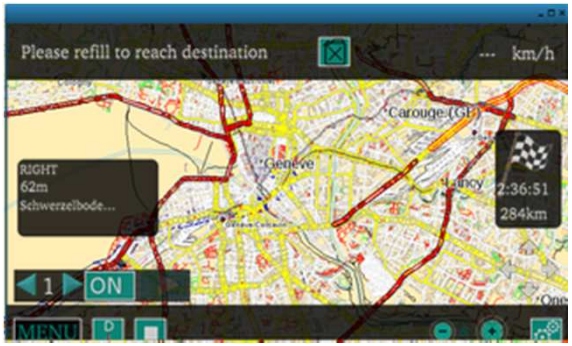
```
...
function map_message(par, iface, func, args)
{
   return
par.message("org.genivi.mapviewer."+iface,"/org/genivi/mapviewer","org.genivi
.mapviewer."+iface, func, args);
}

function mapviewercontrol_message(par, func, args)
{
   return map_message(par, "MapViewerControl", func,
map_session(par).concat(g_map_handle,args));
}...
```

navit



navit plugins



Starting point: Navigation*.qml

```
...
function showZoom()
{
   var res=Genivi.mapviewercontrol_message(dbusIf,
"GetMapViewScale", []);
   ...
}
...
Genivi.mapviewercontrol_message(dbusIf,
"SetMapViewScaleByDelta", ["int16",1]);
showZoom();
...
```

```
...
void GetMapViewScale(const uint32_t&
MapViewInstanceHandle, uint8_t& ScaleID, uint16_t&
IsMinMax)
{
...
}

void SetMapViewScaleByDelta(const uint32_t&
SessionHandle, const uint32_t& MapViewInstanceHandle,
const int16_t& ScaleDelta)
{
   MapViewerControlObj
*obj=handles[MapViewInstanceHandle];
   if (!obj)
      throw DBus::ErrorInvalidArgs("Invalid
mapviewinstance handle");
   obj->SetMapViewScaleByDelta(SessionHandle,
ScaleDelta);
}...
```

Genivi API dbus connection & Navit plugin source code

```
<node xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="/org/genivi/whatev
er“
xsi:noNamespaceSchemaLocation="introspect.xsd">
  <interface name="org.genivi.Whatever.WhateverControl">
    <version>3.1.0 (03-03-2014)</version>
    <doc>
       <line>WhateverControl = This interface offers functions to control the Whatev
er</line>
    </doc>
    <method name="GetVersion">
      <doc>
        <line>GetVersion = This method returns the API version implemented by the se
rver application</line>
      </doc>
      <arg name="version" type="(qqqs)" direction="out">
        <doc>
          <line>version = ...</line>
        </doc>
      </arg>
    </method>
    <method name=“InterfaceFunction_X">
      <doc>
        <line> InterfaceFunction_X = This method ...</line>
      </doc>
      <arg name=“arg_Y" type="u" direction="in">
        <doc>
          <line>arg_Y = Argument Y ...</line>
        </doc>
      </arg>
      <error name="org.genivi.mapviewer.WhateverControl.Error.NoMoreWhateverInstance
Handles">
        <doc>
          <line>This error is generated if no more whatever instance handles are ava
ilable</line>
        </doc>
      </error>
   </method>
  </interface>
</node>
```


```
class code_class_1:
{
        //Where all the logic is in
}
class code_class_2:
{
        //Where all the logic is in
}
class code_class_X:
{
        //Where all the logic is in
}
class Whatever :
public yyy_adaptor,
public DBus::IntrospectableAdaptor,
public DBus::ObjectAdaptor
{
        public: zzz(DBus::Connection &connection)
        :DBus::ObjectAdaptor(connection, "/org/genivi/whatever")
        {
        }

        uint32_t InterfaceFunction_X(arg_Y)
        {
                dbg(lvl_debug,"enter\n");
                ...
        }
}
void plugin_init(void)
{
event_request_system("glib","xxx");
Create dbus connection
conns[CONTROL_CONNECTION]->request_name("org.genivi.Whatever.WhateverControl");
server=new Whatever(*conns[CONTROL_CONNECTION]);
}
```

Thank you !!