ZYLIA

SmartDevCon^2

12-14.09.2013

Delivering In-vehicle infotainment system based on Tizen and Ford's SmartDeviceLink

Jacek Serafiński	Piotr Szczechowiak	Sean Murphy
jacek@zylia.pl	piotr@zylia.pl	sean.murphy@iname.com

http://www.zylia.pl

About us …

- Sean – B.Eng., PhD in Electronic Engineering (DCU, Ireland)
- Jacek – M.Sc. in Computer Science (PUT)
- Piotr – M.Sc. in Telecommunications (PUT), PhD in Electronic Engineering (DCU, Ireland)
- Co-owners of Zylia (Poznań, Poland) – management buyout of Telcordia Poland (2012) – a Polish research centre of Telcordia Technologies Inc.
- Main areas of interest
  - Embedded software development
  - IVI and telematics systems
  - Speech recognition and digital signal processing

About Carmesh

- Carmesh is an EU FP7 Marie Curie IAPP project which focuses on the delivery of advanced automotive services to the Connected Car
- Partners
  - UCD (University College Dublin), Dublin, Ireland
  - Zylia Sp. z o. o., Poland
- Main parts of the project
  - Networking aspects
  - Mobile/Car Integration
  - Integration with cloud services

About Carmesh

- Architecture of the Carmesh automotive service delivery system

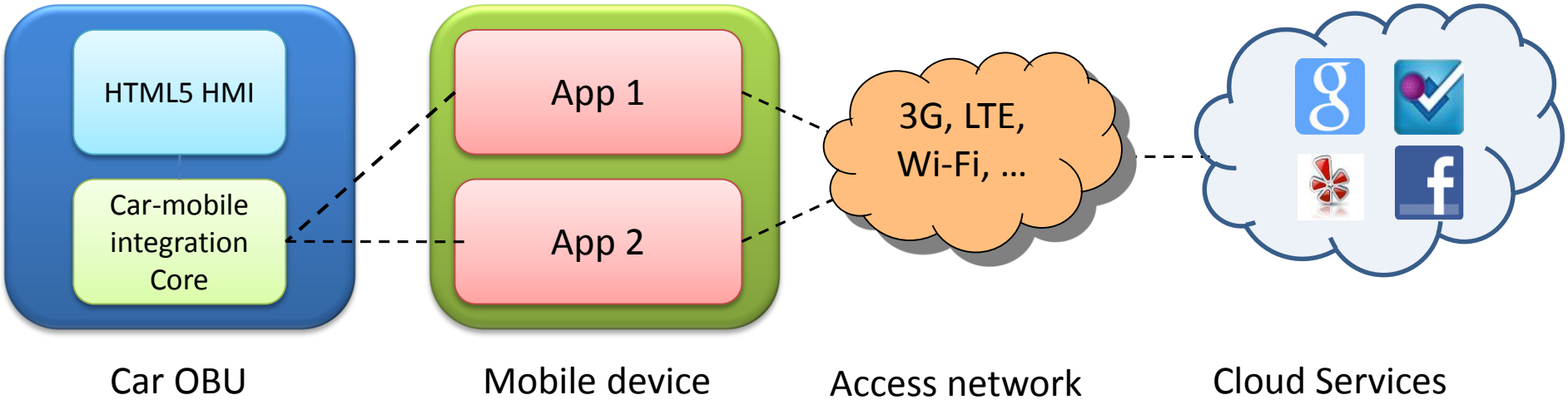


Table of contents

- **Automotive trends in the connected car context (Piotr)**
  - Evolution of the Connected Car
  - Future trends
- **Open Automotive Platforms (Jacek)**
  - Overview of automotive platforms
  - Automotive Grade Linux demonstration
  - Application development for Tizen IVI
- **Application development using SmartDeviceLink (Sean)**
  - SDL overview
  - Application development using Tizen and SDL

Evolution of the Connected Car

- Why connectivity matters? real-time data, social networks integration, novel applications
- It is estimated that by 2025, 60% of the cars on the road will be Internet connected
- Automotive applications market should reach US$ 1.2bn by 2017 (Juniper research)
- Automotive sector sees the connected car market as a big opportunity to offer new services
- The term Connected car was coined some time ago
- This concept covers a mix of different technologies and interfaces that enable in-car Internet services integration

OBU vs mobile

- So why not get rid of the OBU and replace it entirely with a mobile device ?
  - Safety regulations, strict QA processes
  - Integration with car on-board interfaces (buttons, multimedia displays, speakers microphones, HUDs etc)
  - Auto manufacturers want to montize on automotive applications and want to have some control over the ecosystem
  - Security issues, some car data and functionality needs to be locked from mobile developers (e.g. remote engine start-up, etc)

Car-mobile integration

- In the Carmesh project we believe that the mobile device will play a key role in the Connected Car concept
- Mobile-car integration is a crucial aspect
  - Basic integration – contacts, call support, no data connection
  - Single high-quality mobile app (e.g. Mercedes mobile app)
  - Mirroring solutions (e.g. MirrorLink, iOS 7)
  - More open solutions Open SDK for developers (Ford)
- Applications run on the phone and connect wirelessly to the car
- Car provides a HTML5 based user interface with typical automotive controls
- The applications should by designed to minimize driver distraction

Future trends

- Car is to become a supplier of different kinds of data that will be used in numerous novel applications
- Standardized API will allow to use the same application in different cars
- The data gathered from connected cars will be analyzed in the cloud to provide value added services to consumers
- Examples of novel applications:

Car data	Cloud data	Application
Driving behaviour	Driving models	Lower insurance premiums for safe drivers
Position	Calendar	Navigation with planned next appointments
Diagnostic data	Driver behaviour analysis	Dynamic pricing for car rentals
Position, battery level	Charging stations data	Electric car charging optimization

Automotive Platforms

- Closed platforms
  - Ford Applink (MS Sync) – Open API for communicaion with OBU UI (buttons, screen text)
  - GM Onstar – similar to Ford Applink
  - QNX (Toyota, Honda, BMW, Porsche, Audi)
  - Renault RLink, Mercedes – closed environemnt, both HW and SW maintained in-house
  - Tesla (by nVidia) – currently closed for 3rd party app communication, but will open in the future
- Open platforms
  - Tizen IVI
  - MirrorLink
- Virtual integrating platforms
  - drivencomputing.com

Why Tizen IVI?

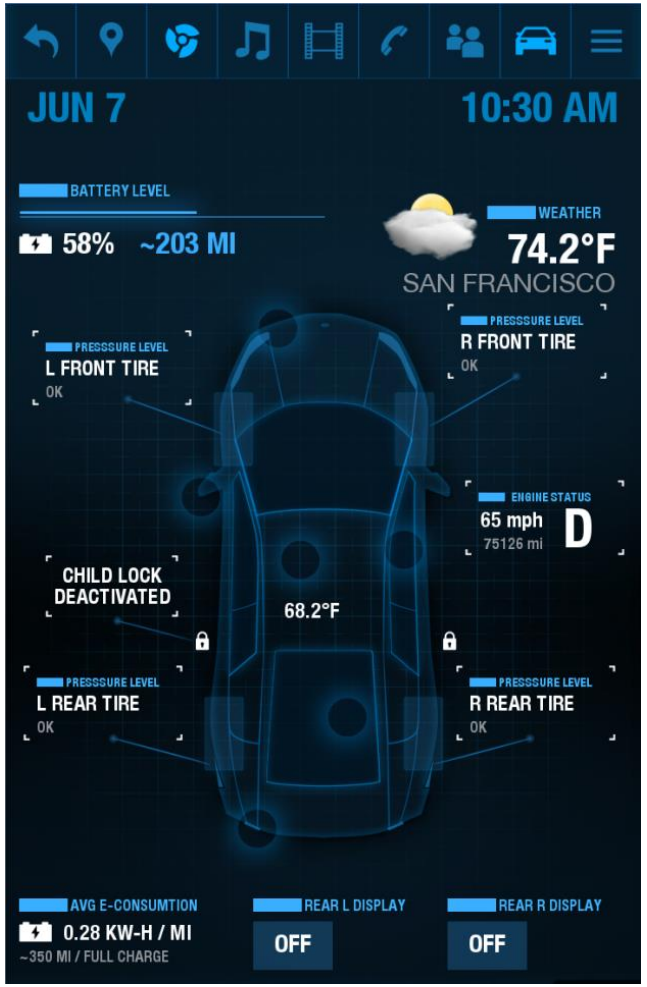
- Why Tizen IVI?
  - Opensource
  - Supported by major players (Samsung, Intel)
  - Demonstrated proof-of-concept by Jaguar (AGL)
  - Communication with mobile phone based on HTML5 (increased lifetime and relevance, no need for frequend updates)
  - Supported by GENIVI Alliance
- Tizen IVI is using Tizen Web Applications
  - SDK (full development toolchain) available
  - Emulator available
  - Tizen IVI snapshots/images available for testing/deployment
- Tizen SDK version 2.2 available at http://tizen.org

Automotive Grade Linux

- AGL Demonstrator Developed by

  Linux Community for User Experience Contest
- Based on Tizen IVI 1.0
- Presented by Jaguar Land Rover on CES 2012

- Let's see it working!



Building application for IVI

- We are using Tizen SDK 2.2 to build Web Application
- Sign it and upload to running Tizen IVI image in VM Player
- Run applications using tutorial:

  https://wiki.tizen.org/wiki/Using_WRT_to_launch_apps_in_Wayland

- Let's see it working!

Outline of the SDL part

- Overview of SDL
- Why were we interested in SDL?
- SDL - the working parts
- Getting it going
- Limitations of SDL
- Application development
- Final thoughts

Ford AppLink

- Phone-car interaction protocol
  - Evolution of Ford's Applink
- Enables mobile applications to interact with car
  - display information, button controls, control in car audio playout etc
- Being pushed (somewhat) within GENIVI consortium
- On Tizen IVI v3.0 roadmap
  - progress is slow

Ford AppLink

- Mobile phone connected to Car through Microsoft Sync
- Concept
  - OBU as UI
  - Application on mobile phone
  - Mobile phone screen locked
  - Applications precertified by Ford (non-distraction issue)

- Ford provides SDK, samples and emulator
- http://blog.carmesh.eu/2013/04/26/first-impression-with-ford-applink-emulator/
- http://blog.carmesh.eu/2013/04/30/videos-demonstrations-of-ford-applink-emulator/
- Ford Applink is older version of SmartDevice Link (SDL)

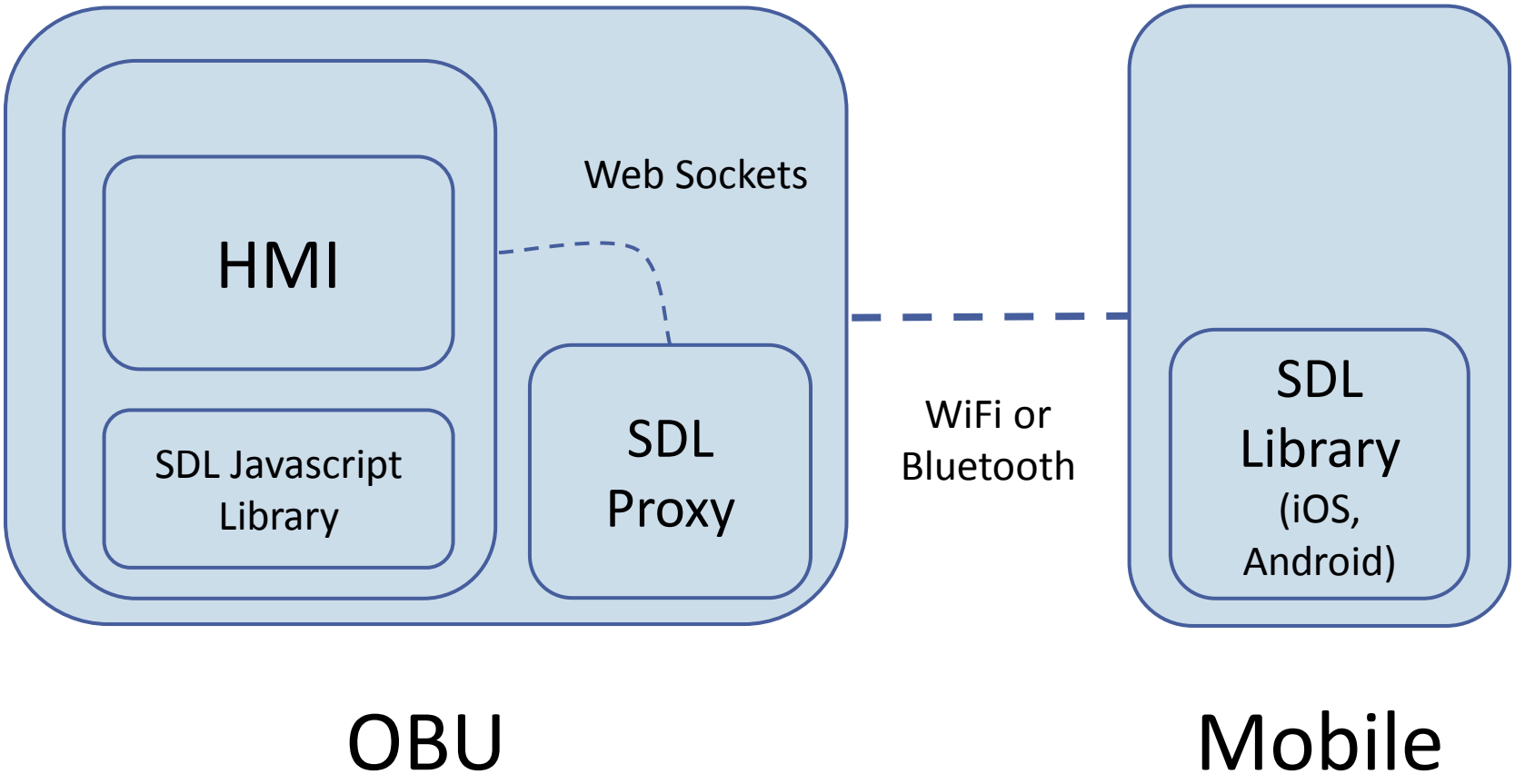
Overview of SDL

- SDL 'richer' view of mobile-car interaction than mirroring solutions
  - iOS, MirrorLink
- Individual applications interact with car
  - multiple applications on multiple devices possible
- There is a quite well defined protocol right now
  - although it is somewhat missing an architecture

Why were we interested in SDL?

- Started looking at it early this year
- Was the most open phone-car interaction solution
  - Ford are championing it and doing quite good developer outreach
  - Although Mirrorlink has more industry support, developer outreach less mature
- We like the vision of brains in phone and car provides UI components
  - very compatible with SDL

SDL - the working parts



SDL - the working parts

- Mobile
  - iOS/Android library(ish)
  - to be included in SDL compatible applications
  - understands SDL primitives and can maintain communications with OBU

SDL - the working parts

- OBU
  - Proxy running as OS process
    - C++ implementation
    - maintains communications with phone and sends to browser
    - quite robust
  - Browser based component
    - Websockets interface to proxy
    - Javascript library for registering for notifications, message composition etc
    - mostly designed around callbacks

Getting it going

- GENIVI source code available
  - Mostly contributed by Ford, although others have done some work on it
- OBU side has its own HMI
- Mobile side has demonstrator application
  - SmartDeviceLinkTester

Getting it going - OBU

- cmake build process
- Did not compile out of the box on either Tizen or Ubuntu
  - Had some issues around Bluetooth and websockets SHA calculation
- Build generates binary which runs proxy and launches HMI
  - chromium by default
- HMI offers main automotive UI and emulates automotive buttons

Getting it going - Android

ZYLIA

- System comprises of library and application
- Straightforward to build and deploy for Android
- Small issue with default settings for Wifi mode being incorrect

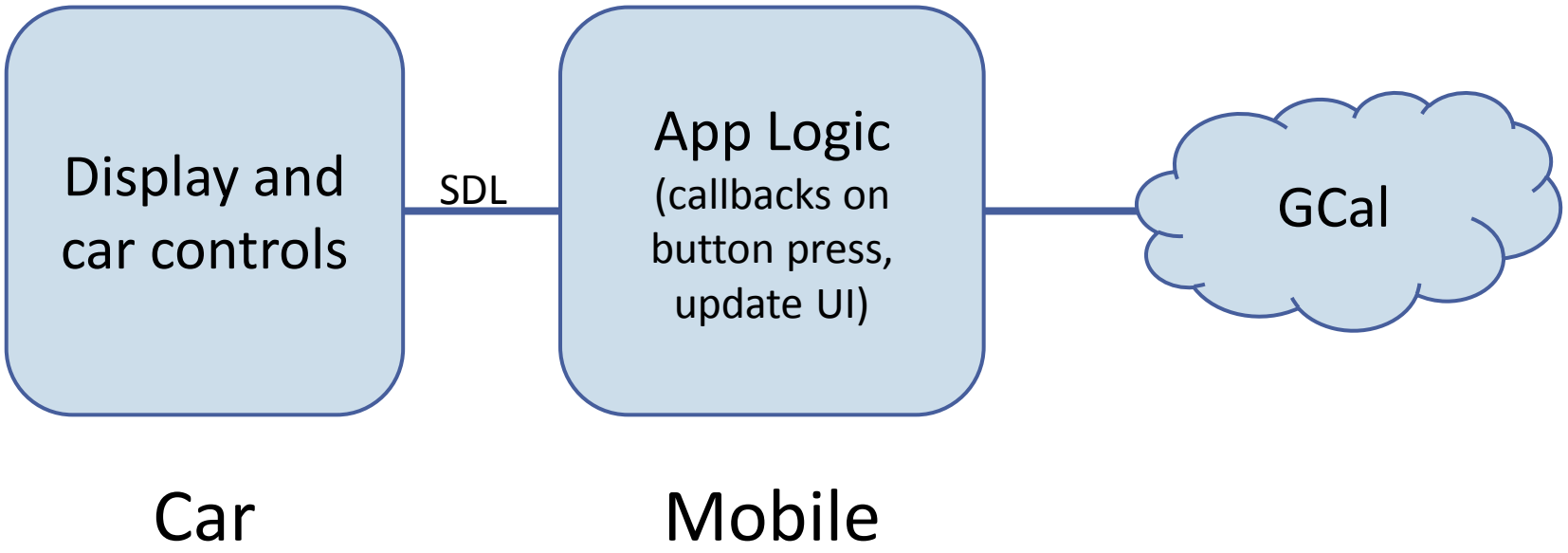
Limitations of SDL (current)

- Control over in car display is very limited
  - Logic required on OBU side to know how to render message
  - No real graphics support
- Little navi integration
  - not possible to send POI to in car navi
  - some TurnByTurn updates available, though
- Features supported by protocol not yet implemented
  - vehicle data, application types/templates, file transfer

Application development

- Basic application to provide Gcal integration in car
- Can see next upcoming appointments and navigate to them
  - assuming location data in gcal event
- Use in car controls to control application

GCal Application



Application Development - OBU

- Changes to HMI required
  - Change default application behaviour
  - Instead of showing basic media player, show calendar interface
    - support update of view to map view
- HMI implemented in ember.js
  - Strong state oriented approach
  - Had to add new states for calendar views and map views
- Strong MVC design
  - somewhat problematic in context of app running on mobile

Application Development - OBU

- Model (in MVC) receives updates from SDL JS library
  - ugh!
- Contents of model updated
  - observables update UI view

Application Development Android

- SDL mobile functionality comprises of service and activity
- Service communicates with OBU
  - regular checks if OBU is available
- Starts activity if not active
- ProxyService implements functions to handle all messages
  - Modified to handle button presses
- Seperation between protocol and application not obivous
- Poor documentation

Application Development - hacks

- Used existing message to support data transfer
  - Show message
    - provides limited support for displaying text content
- Updates info relating to next appointments
  - rendering of next appointments info on specific view
- Change to map view realized by sending specific message with Show

Final thoughts on SDL

ZYLIA

- Applink is available in over 1m (Ford) cars today
  - Backwards compatible with SDL
- Remains unclear if SDL will gain momentum
- Solutions will likely evolve to HTML5
  - Provide HTML5 views in car displays
    - Mirrorlink and SDL can evolve to this
- Developer tools need to evolve
  - heterogeneous platforms (mobile, car)
  - Distributed MVC?

THANKS!

http://www.carmesh.eu

http://www.zylia.pl

jacek@zylia.pl

piotr@zylia.pl

sean.murphy@iname.com