Technical Platforms and Local Dynamic Maps

Christian Zott, Robert Bosch GmbH
christian.zott@de.bosch.com

www.safespot-eu.org safespot@crf.it
Perspectives to SAFESPOT’s Platforms

- **Technical**
  - Platform definition
  - Targeted users and their needs
  - Functional, SW, HW views on the platform architectures
  - Development states and future-challenges

- **Business**
  - Derived platform products
  - Cost Estimations and Limits
  - Platform players and allocation to industry (branches)

- **Deployment**
  - Compliance to related standards (ISO, SAE, ETSI, OGC,…)
  - Certification
  - Roadmaps
General Platform Definition

An architecture, composed by components and interfaces, which serves as a foundation or base for the development of applications or systems

HW platform examples:
CPU or computer family (x86, 68000, ARM, powerPC, SPARC)

SW platform examples:
Operating systems (Unix, Linux, Windows,...), SW development frameworks, libraries (e.g. Java, Qt) and APIs, web browsers (e.g. Mozilla Firefox) or tool frameworks (e.g. Eclipse) open for plug-ins

Platforms
• provide base functionality and communicate with other components
• are often third party components, i.e. developers procure and use it, but mostly do not develop or modify it
• save development time by reuse of mature, widely tested and documented design patterns, components, interfaces and tools
• support development organization and planning
SAFESPOT’s Platforms

Two scopes:

I. General, sustainable concept beyond SafeSpot, i.e. for related EC projects like CVIS, future projects and products, defining a generic *functional* architecture and APIs

II. Specialized, detailed concept including SW and HW architecture for SafeSpot prototype, application demonstration and test site development

Targeted platform users:

- Infrastructure and vehicle application *developers* for ITS and ADAS
- Not the drivers itself (compare to web server, IT middleware, etc.)
Some SAFESPOT Platform User Needs

Reusable, flexible, scalable

with respect to varying context, i.e. for

- **different HW configurations:**
  node sensors, node communication/networks, HW platforms, vehicles

- **different SW configurations:**
  OS, frameworks, SW design patterns

- **wide range of applications:**
  driver information, assistance, vehicle control, traffic monitoring, control, comfort to safety critical scenarios, environments, e.g. motorway, rural, urban
Generic SafeSpot Platform Functional Architecture

- **Node Central Database:** (Local) Dynamic Map
- **Message Generation**
- **Data Fusion**
  - **Object Refinement**
  - **Situation Refinement**
- **Ad-hoc Network Routing**

- **Node Gateway**
- **Data Reception**

- **Node Sensors and Data Sources**
- **Ad-hoc Network Routing**

- **Driven by HW configuration**
- **Driven by applications**
- **Re-usable platform components**
Local Dynamic Map

- landmarks for referencing
- temporary regional info
- com nodes, fusion result
- map from provider
- veh id pos...
- ego id pos...
- congestion id pos length dir...
- tree id pos...
- fog id pos a,b...
- accident id pos...
- rsu id pos vel type...
- ego pos vel...
Local Dynamic Map – Contents

Dortmund Test Site
Local Dynamic Map - Contents

Classical Geometry Map Content

- links, center-line
- junctions
- usually rich attributes not shown here
- adapted to navigation systems needs
Local Dynamic Map - Contents

Static Layer for Dortmund Test Site

Data from TeleAtlas

- building facades
- curbs
- lane dividers
- lane markings
- traffic lights
- light poles
- more to come/
  not shown here
Vertically Elevated Objects – Landmark View

Data from TeleAtlas

- building facades
- traffic lights
- light poles
- more to come/
  not shown here
- LDM is more than just a storage device
- Unique representation of environment – not copy of other LDMs
- Moving “horizon” of static information, e.g. road network, landmarks
- **Dynamic** contents governed by Data Fusion process
- Does not only provide data, but **access functions** for data projection, selection, filtering, join, grouping, sorting, etc.
- **Event-triggered notification** can provide extra functionality

*platform output
Data Fusion Levels

- Raising of level in general by finding relations (wrt space, time, attributes)
- Sensor systems, sources often provide data on different levels
- Data fusion levels grouped in
  - Object Refinement
  - Situation Refinement
In-Vehicle Platform View Generation

- Views generated by queries
- Views specified by query conditions
- Views application specific
- Query complexity usually increases with level
- Scope usually increases with level, whereas resolution decreases
- Query rate usually decreases with level
LDM Contents – Structure

- Feature
  - Roads&Ferries
  - ReferenceTrack
  - LandCover
  - RoadFurniture
    - RoadElement
    - Junction

envelope()
LDM Contents – Geometry Objects

Simple Features (see OGC/ISO, used in TA/Bosch implementation)
Overview of Level 1 Queries (Query class methods)

**Feature Generation**
- from 2 features
  - intersection
  - convexHull
  - geomUnion
  - difference
- from 1 feature
  - buffer
  - boundary
  - centroid
  - envelope

**Tests**
- for 2 features
  - intersects
  - contains
  - equals

**Main Query**
- args: names, attributes, conditions
  - select

**Select Result Navigation**
- next
- previous
- first
- last
- gotoRecord
- size
- value

**Transactions**
- args: names, attributes, values, conditions
  - insert
  - update
  - delete

**Maintenance**
- setDB
- getLastError
- clearSelect

**Q-API**

**T-API**
In-Vehicle Platform Functional Architecture

SAFESPOT WATCH-OVER Workshop
Stuttgart, Jan 21-22, 2008
In-Vehicle Platform HW Configurations

Powertrain/body networks

OEM Gateway

GPS PPS

Main & VANET Router PC
In-Vehicle Platform SW Architecture

- LDM database: 2 implementations (TeleAtlas/Bosch, Navteq)
- OS: WindowsXP
- DF process: C++, framework using Qt
In-Vehicle Platform SW Architecture

**Class SP1_MainPC**

- **coordinator**
- **sharedMemory**
- **scheduler**
- **threadList**

**Thread**
- `prio`
- `state`
- `stop()`
- `wait()`
- `run()`

**Thread Methods**
- `run()`

Example threads:
- dataAcquisition
- OR_alignment
- OR_radar
- OR_vanet
- SR_centralLevel
- SR_ego
- SR_obj
- SR_event
- infoProv
- logger

Framework implemented and compiled into exe and DLLs by Bosch

Code pieces (e.g., methods) implemented and compiled into DLLs by partners
In-Vehicle Platform Timing

- IO bound tasks run short, but frequent, often need to get CPU immediately
- Processor bound tasks run longer, but less frequent
- Proc bound tasks need to be pre-empted to limit latencies for non-fusion data
SAFEPROBE Partners and Schedule

**OEMs**
- Centro Ricerche Fiat (SP Co-leader)
- Volvo Technology
- REGIENOVA
- Piaggio

**Suppliers**
- Robert Bosch (SP leader)
- Magneti Marelli Sistemi Elettronici
- IBEO Automobile Sensor
- Siemens

**Research and others**
- Institute of Communication and Computer Systems
- MIRA

**Schedule**
- Feb 2006 – Jan 2007: Needs, requirements and use cases
- Jan 2008 – Jan 2009: Test and Validation