
Development Platform for Safe and Efficient Drive

# Guidelines for application development – 2nd release

| | | | |
|---|---|---|---|
| Deliverable n. | D25.6 – Guidelines for application development – 2nd release | | |
| Sub Project | SP2 | ADAS development platform | |
| Workpackage | WP2.5 | Platform System Architecture | |
| Tasks | T2.5.4 | Guidelines for applications development | |
| Authors | Frank Badstübner<br>Ralf Ködel<br>Wilhelm Maurer<br>Martin Kunert<br>F. Giesemann, G. Paya Vaya, H. Blume | Infineon Technologies AG<br><br><br>Robert Bosch GmbH<br>IMS | |
| File name | D25.6_Guidelines_for_applications_development_-_Second_Release_IFAG_Final | | |
| Status | Final, v0.5 | | |
| Distribution | Restricted (RE) | | |
| Issue date | 31/08/2014 | Creation date | 05/10/2014 |
| Project start and duration | 1st of September, 2012 – 36 months | | |

# REVISION CHART AND HISTORY LOG

| Ver | DATE | AUTHOR | REASON |
|---|---|---|---|
| 0.1 | 2014-05-28 | Frank Badstübner | Table of contents and document structure created, content of first version included |
| 0.2 | 2014-07-28 | P.C. Antonello, S. Torchiaro | Added paragraph about ISO 26262 |
| 0.3 | 2014-08-30 | Frank Badstübner | Consolidated version generated |
| 0.4 | 2014-09-24 | Frank Badstübner | Final version generated |
| 0.5 | 2014-10-05 | Frank Badstübner | Reviewer comments considered. Final version prepared for submission. |

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF ABBREVIATIONS

| ABBREVIATION | DESCRIPTION |
|---|---|
| ADAS | Advanced Driver Assistance System |
| ADASIS v2 | ADASIS Forum is an organization of all major vehicle manufactures and System Vendors with the objective to define to standardize data model to represent map data ahead of the vehicle (ADAS Horizon, Electronic Horizon) as well to define standardized interface to enable applications to access this ADAS Horizon.<br><br>ADASIS v2 Specification covers the design and development of the CAN and C API interface between Digital Maps and ADAS applications (Electronic Horizon). After its release in 2010 the ADASIS v2 Specification, backed by the ADASIS Forum, is now accepted as the de-facto standard. |
| ADTF | Automotive Data and Time-Triggered Framework |
| ASIL | Automotive Safety Integrity Level |
| CAN | Controller Area Network |
| E/E | Electrics and Electronics |
| FPGA | Field Programmable Gate Array |
| GigE | Gigabit Ethernet |
| GNU Indent | The *GNU indent* program can be used to make code easier to read. It can also convert from one style of writing C to another. *indent* understands a substantial amount about the syntax of C, but it also attempts to cope with incomplete and misformed syntax. See [9] for details. |
| HMI | Human Machine Interface |

| | |
|---|---|
| HW | Hardware |
| IMS | Institute of Microelectronic Systems |
| ISO | International Organization for Standardization |
| IWI | Information Warning Intervention |
| NIR | Near Infrared |
| LVDS | Low Voltage Differential Signaling |
| PC | Personal Computer |
| SW | Software |
| TCP / IP | Transmission Control Protocol / Internet Protocol |
| UDP | User Datagram Protocol |
| USB | Universal Serial Bus |
| WI | Warning Information |

# EXECUTIVE SUMMARY

D25.6 is dedicated to the guidelines for application development in the DESERVE platform. Work performed refers to task 2.5.4 of the DESERVE project. The development of an application using the DESERVE platform requires to follow a procedure that may be different from the existing methods. This task defines and details such a procedure inside "guidelines" that have to be followed when an application is developed for the DESERVE platform.

Common guidelines for modules and functions concentrating on the co-design methodology for the development of hardware dependent software functionalities as well as application specific guidelines for the DESERVE demonstrators were defined and are described in this document.

For the Inter-Urban Light Assist, for instance, following application specific guidelines will be applied:

1) All modules developed shall be encapsulated and use the standardized interface descriptors as specified in deliverable D2.2.1 [8].
2) The ADTF tool from Elektrobit shall be used as the integrated development framework for the PC-based development work.
3) For the embedded controller platform the MicroAutoBox from partner dSPACE shall be used with a Virtex6 FPGA as first release and possible extensions to Virtex7 as needed.
4) The communication protocol between the independent and modular units either uses the ADTF Message Bus (UDP/IP), the CAN2.0 Bus (for ADASISv2 communication with the navigation unit) or the generic Ethernet links (based on TCP/IP or UDP).
5) Proprietary sensor protocols shall be transcoded with dedicated break out boxes to the standardized interfaces.

6) Design space exploration methods shall be implemented when developing and transferring software code from the PC environment to the embedded FPGA platform.

7) The main working focus shall be put on the improvement of the embedded system on chip development process by efficient and easy transfer of software algorithms to the heterogeneous processor implementations.

The guidelines for the other demonstrators may vary slightly (e.g. use of RTMaps instead of ADTF).

Both common and application specific guidelines will be used by all stakeholders involved in the development of components for ADAS applications, sensors and control units.

# 1. Introduction

## *1.1. Objective and scope of this document*

The objective of this deliverable is to describe the second release of the guidelines for the development of applications into the DESERVE platform system architecture. D25.6 is the third of a series of deliverables documenting the work performed to define the first release of the platform system architecture:

- D25.2 Platform system architecture – 2nd release
- D25.4 Standard interfaces definition – 2nd release
- D25.6 Guidelines for applications – 2nd release and
- D25.8 DESERVE platform – 2nd release

It is to be noted that D25.6 essentially represents an extension to D25.2 and D25.4.

In order to avoid repetitions of other deliverables' contents, readers are kindly referred to following DESERVE deliverables: Detailed description of the DESERVE development system is provided in the DESERVE deliverables D2.1.1 [2] and D2.1.2 [3]. The DESERVE development methodology is described in deliverables D2.1.3 [4] and D2.1.4 [5], while information on the development platform is available in the DESERVE platform require-ments and specification deliverables D1.2.1 [6] and D1.3.1 [7], respectively.

## *1.2. Structure of the deliverable*

This deliverable provides first guidelines for the development and implementation of new ADAS applications onto the DESERVE platform.

The deliverable starts with the general definition of common guidelines for modules and functions (chapter 2). Chapter 3 presents function and application specific guidelines for the DESERVE demonstrators to be developed:

- "Warning Functions" (WP4.1)
- "Control Functions" (WP4.2)
- "Vulnerable Road User Protection Functions" (WP4.3)
- "Automated Functions" (WP4.4)
- "Cooperative System Functions" (WP4.5)
- "Inter-urban light assist" (WP4.6)

# 2. Common guidelines

The DESERVE common guidelines for modules and functions concentrate on the co-design methodology for the development of hardware dependent software functionalities including software framework tools, build process flow and suggestions for using software framework tools.

## *2.1. Co-design methodology*

A first possible co-design platform (which will be utilized for the development of the urban light assist in WP4.6) consists of ADTF and the FPGA-based hardware platform allowing the co-design of software and hardware for applications and algorithms. The whole application or algorithm can be implemented in software using the ADTF concept of filters. Single filters or the whole application can then be implemented in hardware and linked to the ADTF environment. This allows reusability, flexibility and fast verification of the implemented hardware modules.

Quantitative hardware cost models allow to evaluate different characteristics of the hardware modules early in the development phase, e.g., based on parameters derived from the software implementation. They may also provide hints to the software developers that can guide the parameter selection process, in order to make efficient hardware implementations possible.

There are of course further co-design platforms, which may be used in the development process for other DESERVE demonstrators, e.g. RTMaps. Similarly, an FPGA is not the only suitable hardware platform, there are other approved and similar systems too, e.g. powerful multi-core microcontrollers, digital signal processors etc.

## 2.2. Deployment of hardware dependent software functionalities

Objective of the SW framework is to provide a unified platform for the software development to demonstrate upcoming family of microcontrollers. Target users are the application engineers and customers who use the application examples or demo software drivers for prototyping. Target application domain is automotive.

The SW Framework has two main parts (Figure 1):

- Software development tools
- Software objects consisting of microcontroller application examples

The release of SW Framework consists of folders consisting of software development tools and software objects consisting of template projects.

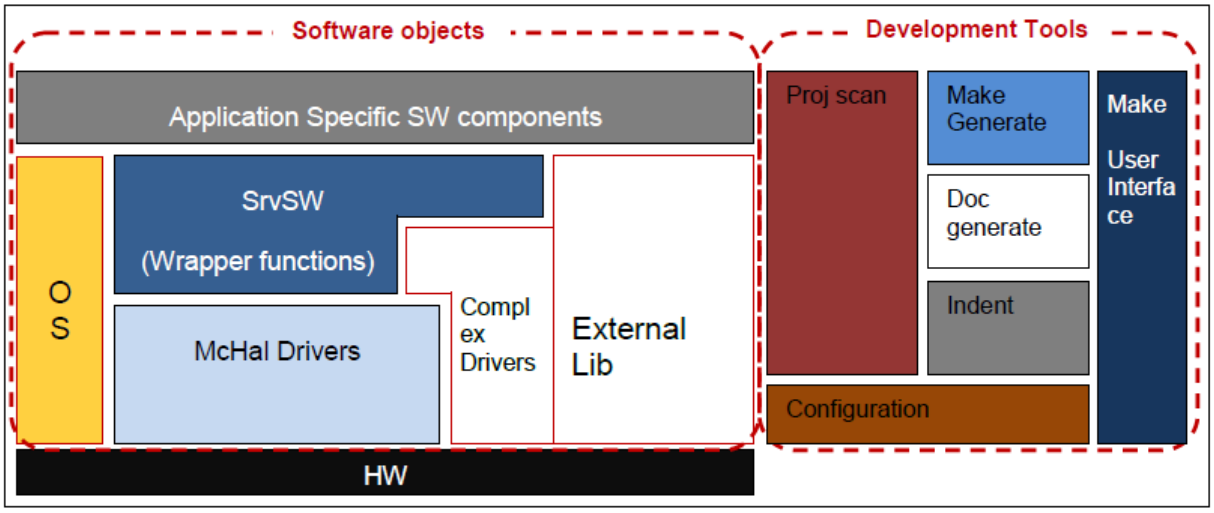


**Figure 1: Software framework**

### *2.2.1. SW Framework Tools Build Process Flow*

Software components are input to the development tools to generate the output as shown in Figure 2 below:

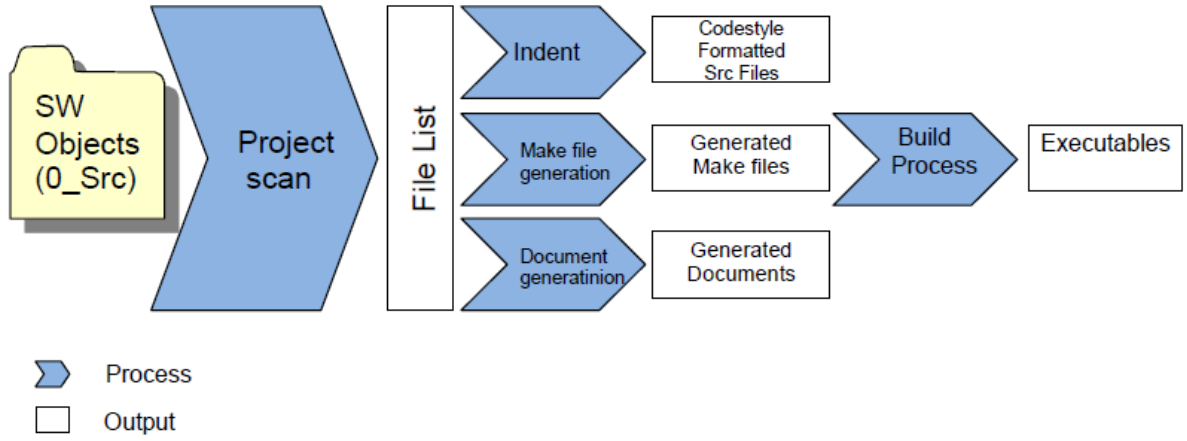


**Figure 2: Software development tools**

Framework tools are responsible for following activities:

- Software components are scanned by the project scan process and the file list is developed for next process.
- Files listed in the file list are used for Indent to make standard code style formatting (GNU Indent is used in the SW Framework).
- The Make file is generated using the file list and these make files are invoked during the build process to generate the executables.
- Doxygen tool runs on the files listed in the file list to generate project documents.

### *2.2.2. Motivation to use the SW Framework Tools*

The framework tool environment is very helpful for demo software development because of the following reasons. The SW Framework

- provides the uniform structure for project files organization, API usage and tool chain environment,
- makes it easy to switch between different tool chains without changing source code,
- builds the make files automatically for the source files (it scans all the C, header and assembly files inside the source folder and it generates the make files depending on the used tool chain),
- provides the documentation possibility and
- provides an easy structure to handle homogeneous and heterogeneous multicore software.

## *2.3. Guidelines related to ISO26262*

ISO 26262 is a derivative of IEC 61508, the generic functional safety standard for electrical and electronic (E/E) systems.

Ten volumes make up ISO 26262. It is designed for series production cars, and contains sections specific for management, concept and development phase, production, operation, service and decommission.

The ISO 26262 requires the application of a "functional safety approach", starting from the preliminary vehicle development phases and continuing throughout the whole product lifecycle.

The DESERVE project focuses on the concept and development (at system, hardware and software level) phases of the lifecycle.

During these phases, the main steps defined by the Standard are:

Item definition: the Item has to be identified and described. To have a satisfactory understanding of the item, it is necessary to know about its functionality, interfaces, and any relevant environmental conditions.

Hazard analysis and risk assessment: to evaluate the risk associated with the item under safety analysis, a risk assessment is required. The risk assessment considers the functionality of the item and a relevant set of scenarios. This step produces the ASIL (Automotive Safety Integrity Level) level and the top level safety requirements.

The ASIL is one of the key concepts in the ISO 26262. The intended functions of the system are analyzed with respect to possible hazards. The ASIL asks the question: "If a failure arises, what will happen to the driver and to associated road users?".

The risk of each hazardous event is evaluated on the basis of:

- Frequency of the situation (or "exposure")
- Impact of possible damage (or "severity")
- Controllability

The ASIL level is standardized in the scale:

- QM: quality management, no-risk
- A, B, C, D: increasing risk with D being the most demanding.

The ASIL shall be determined without taking into account the technologies used in the system. It is purely based on the harm to the driver and to the other road users.

Identification of technical safety requirements: the top level safety requirements are detailed and allocated to system components.

Identification of Software and Hardware safety requirements: The technical safety requirements are divided into hardware and software safety requirements. The specification of the software safety requirements considers constraints of the hardware and the impact of these constraints on the software.

To take into account the functional safety approach, the DESERVE applications should consider the application of the following main points:

1. Analyze risk early in the development process
2. Establish the appropriate safety requirements
3. Consider these requirements in software and hardware development

The impact of the standard is different for the development of warning functions, control functions or automated driving functions.

# 3. Function and application specific guidelines

Function and application specific guidelines will be exemplarily shown for the application "Inter-urban light assist". The general component layout for the DESERVE demonstrator "Inter-Urban Light Assist" was defined and elaborated in work package 3.1 and is depicted in Figure 3.

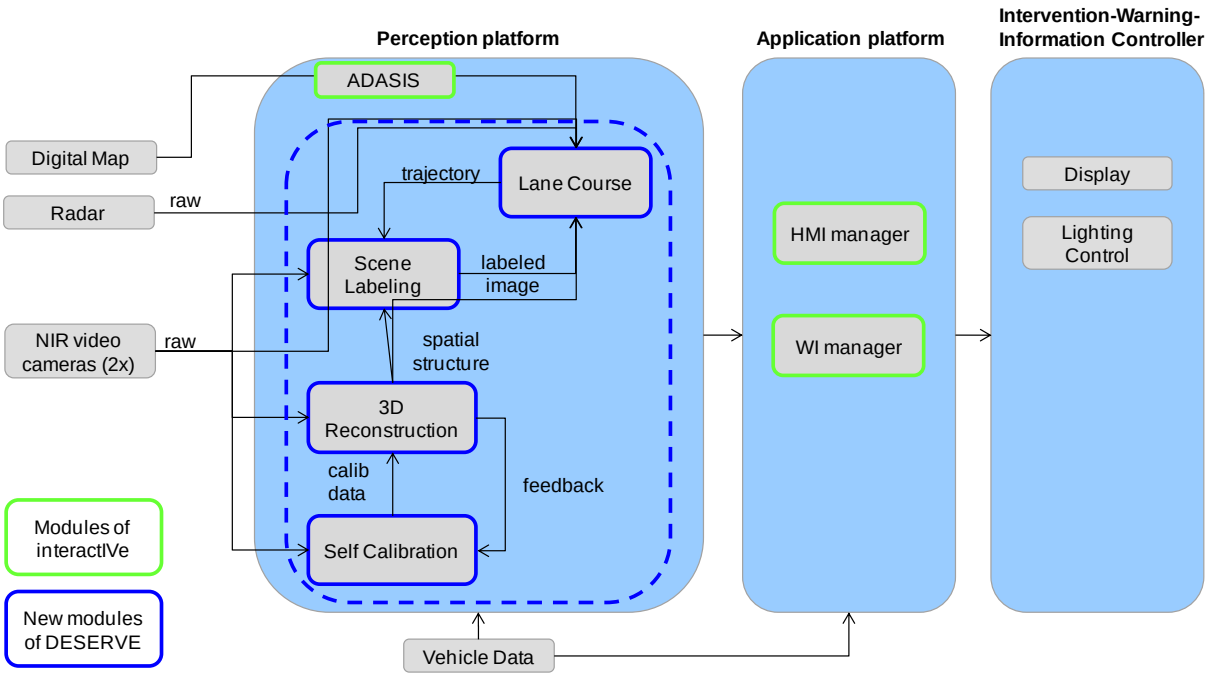


**Figure 3: Inter-Urban Light Assist components layout**

The environmental sensing components are one long-range radar sensor, two NIR (near infrared) video cameras and a navigation unit. Within the perception platform this information has to be processed in specific modules to provide the necessary information to the HMI (human machine interface) and WI (warning information) manager of the

application platform. The driver information and control of the matrix headlamp is finally done in the IWI (information, warning, intervention) controller.

While the processing and handling of the digital map data is already well standardized by the ADASISv2 protocol and a general module is yet available, all other modules in the perception platform still have to be developed.

The development of these new modules shall follow the general DESERVE project principles and is applied to the already agreed hardware components and software tools of the German partners participating in the WP4.6 demonstrator build-up (see Figure 4).

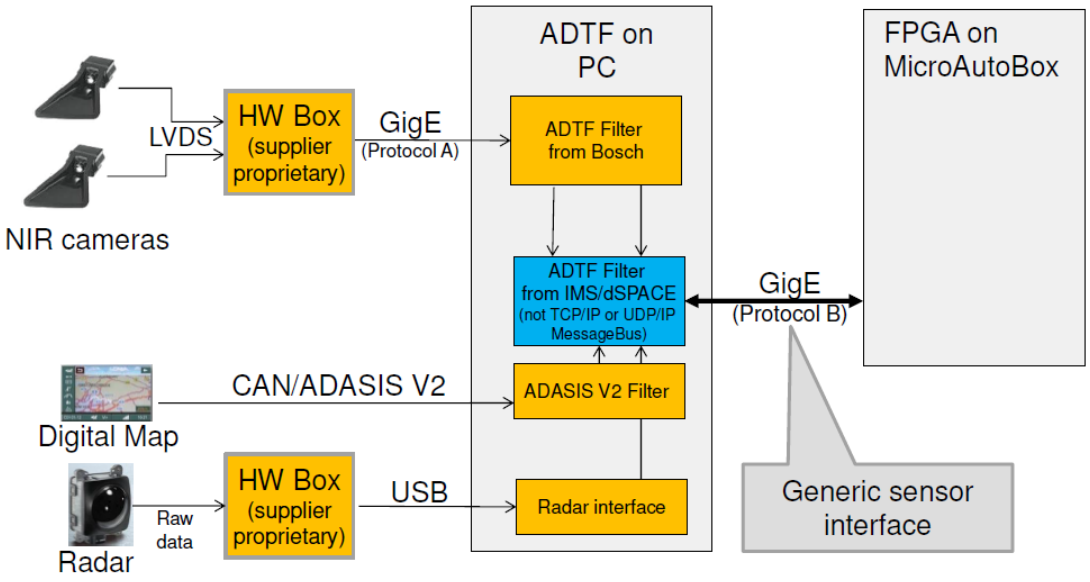


**Figure 4: Inter-Urban Light Assist demonstrator build-up concept**

The following development guidelines are proposed for execution of the underlying work packages and tasks for the Inter-Urban Light Assist:

1) All modules developed shall be encapsulated and use the standardized interface descriptors as specified in deliverable D2.2.1 [8].
2) The ADTF tool from Elektrobit shall be used as the integrated development framework for the PC-based development work.
3) For the embedded controller platform the MicroAutoBox from partner dSPACE shall be used with a Virtex6 FPGA as first release and possible extensions to Virtex7 as needed.
4) The communication protocol between the independent and modular units either uses the ADTF Message Bus (UDP/IP), the CAN2.0 Bus (for ADASISv2 communication with the navigation unit) or the generic Ethernet links (based on TCP/IP or UDP).
5) Proprietary sensor protocols shall be transcoded with dedicated break out boxes to the standardized interfaces.
6) Design space exploration methods shall be implemented when developing and transferring software code from the PC environment to the embedded FPGA platform.
7) The main working focus shall be put on the improvement of the embedded system on chip development process by efficient and easy transfer of software algorithms to the heterogeneous processor implementations.

# 4. Conclusions

In addition to common guidelines, D25.6 presents the function and application specific rules to be followed for the development of DESERVE demonstrators "Warning Functions", "Control Functions", "Vulnerable Road User Protection Functions", "Automated Functions", "Cooperative System Functions" and "Inter-urban light assist".

Seven rules were worked out and agreed by the involved partners. These rules will be employed during the next development steps.

# REFERENCES

[1]. International Organization for Standardization (ISO), ISO 26262, Road vehicles -- Functional safety, 2011.

[2]. DESERVE deliverable D2.1.1 – Development system (first release)

[3]. DESERVE deliverable D2.1.2 – Development system (final release)

[4]. DESERVE deliverable D2.1.3 – Development method (first release)

[5]. DESERVE deliverable D2.1.4 – Development method (final release)

[6]. DESERVE deliverable D1.2.1 – Development platform requirements

[7]. DESERVE deliverable D1.3.1 – Development platform specification

[8]. DESERVE deliverable D2.2.1 - Perception layer - Preliminary Release

[9]. GNU indent documentation: http://www.gnu.org/software/indent/manual/