CRYSTAL

CRitical S**YST**em Engineering **A**cce**L**eration

WP 306 Milestone Report V1
D306.011

DOCUMENT INFORMATION

Project	CRYSTAL
Grant Agreement No.	ARTEMIS-2012-1-332830
Deliverable Title	WP 306 Milestone Report V1
Deliverable No.	D306.011 (update field only)
Dissemination Level	CO
Nature	R
Document Version	V0.01
Date	2014-05-19
Contact	Pierre du Pontavice
Organization	Valeo
Phone	+33 1 80614441
E-Mail	Pierre.du-pontavice@valeo.com

AUTHORS TABLE

Name	Company	E-Mail
Pierre du Pontavice	Valeo	Pierre.du-pontavice@valeo.com

REVIEW TABLE

Version	Date	Reviewer
0.1	15/05	Serrie Chapman

CHANGE HISTORY

Version	Date	Reason for Change	Pages Affected
0.1	07/01/2014	Initial Version	
0.2	12/05/2014	M12 revision	All
1.0	15/05/2014	Final version	

CONTENT

Content of Tables

No table of figures entries found.

Content of Figures

1 Introduction

1.1 Role of deliverable

This document has the following major purposes:

- Define of the overall use case, including a detailed description of the underlying development processes and the set of involved process activities and engineering methods
- Provide input to WP601 (IOS Development) required to derive specific IOS-related requirements
- Establish the technology baseline with respect to the use-case, and the expected progress beyond (existing functionalities vs. functionalities that are expected to be developed in CRYSTAL)

1.2 Relationship to other CRYSTAL Documents

XXX Relation to WP3.3 (e.g. requirement engineering)

XXX Relations to SP6 WP and Bricks
XXX Relation to WP6.3 (requirement engineering, ARTISAN Studio integration, e topics)

1.3 Structure of this document

This document is articulated in the following chapters, from 2 to 4:

Chapter 2: Use Case Definition and general process description.

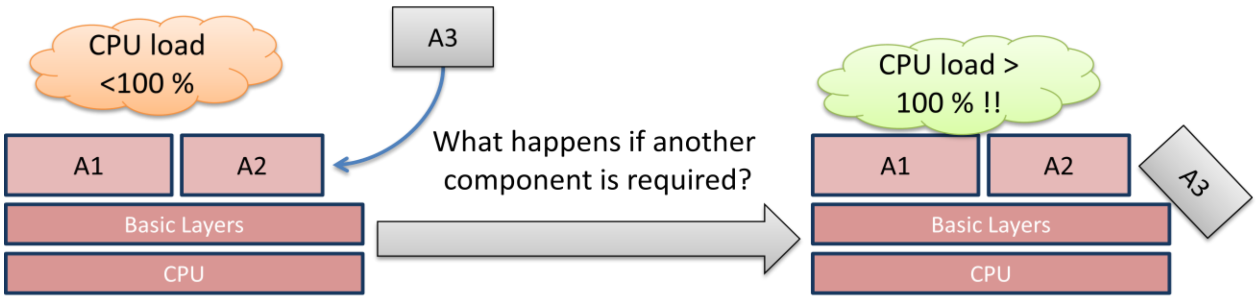
Chapter 3: Detailed description of the use case process.

Chapter 4: Identification of the engineering methods in the engineering environment applied to the use case.

2 Use Case Description

2.1 User Story

- The software complexity increases
  - Vehicles become connected, autonomous, ...
  - AUTOSAR R4.x software standard
  - Dependability constraints (ISO 26262) require dedicated software fault tolerant mechanisms
  - Domain standards (e.g. EURO 6) lead to an increase of code size



- Suppliers roadmaps show that high power single core controller are replaced by multicore

The proposed solution is compliant with the latest automotive standards in the field:
AUTOSAR4.0 for Software Architecture
ISO26262 for functional safety.

Current SW are running on single core microcontrollers. Multicore controllers, uses different memory mapping models (cache, shared and unshared memory). Sw running on different cores need dedicated services to communicate. Moreover, sw architecture have not been design parallelism computing in mind.

These usages can badly affect real time performances and instead of gaining power using multicore controllers, sw may be slower to run and determinism might not be guaranteed

The same applies to the safety requirements.

Hence challenges is to keep SW real time constraints and benefit from multicore advantages without sacrifying Safety functionalities.

2.1.1 Select the suitable candidate

2.1.1.1 Application

Inside Valeo 2 main domains has been considered as Use Case

- ADAS (Advance Driving Assistance system)
- ECM (Engine Controller Management).

Both require compliancy with ISO 26262 and AUTOSAR 4.0.

The application selected is the ECM. The selection has been based on the current state of each project, ECM is really well known but has reach is limits using single core CPU. Internal work has started in Valeo which CRYSTAL can benefit of.

ADAS in the other hand is very new lots has be defined which won't give enough time for CRYSTAL to concentrate on the targeted activities.

2.1.1.2 Microcontroller

Supplier Roadmaps have been studies most of them are confidential at the time this document is written.

2.1.1.2.1 4 suppliers have been considered:

- Infineon for their TriCore
- Renesas for their RH850
- Freescale
- ST

	INFINEON	ST	FREESCALE eGTM	F
HighEnd	SAK-TC297T-128F300	Matterhorn: LFBGA292, PBGA416, LFBGA512	MPC5777M (Matterhorn)	MP
MidEnd	SAK-TC277T-64F200	McKinley (Mid-End): LFBGA292	MPC5746M (McKinley)	MP
MidEnd	SAK-TC275T-64F200	McKinley (Mid-End): eLQFP176	MPC5746M (McKinley)	MP
LowEnd	SAK-TC265D-40F200	K2 (Low-End) : eTQFP144, eLQFP176	MPC5744K (K2)	MP

2.1.1.2.2 Evaluation has been performed against :

- Criterias Core evaluation
- Mathematical performance
- Peripheral performance
- eGTM proved
- Power supply
- Packaging
- Roadmap
- Support
- Tools
- MCAL
- Development cost
- Delay of delivery
- Suppliers
- Production sites (multi-sites)
- Number of suppliers
- Pricing
- Safety
- OS

2.1.1.2.3 Microcontroller selected

The one that came on top was the Infineon TriCore mainly because of its goods performances and availability.

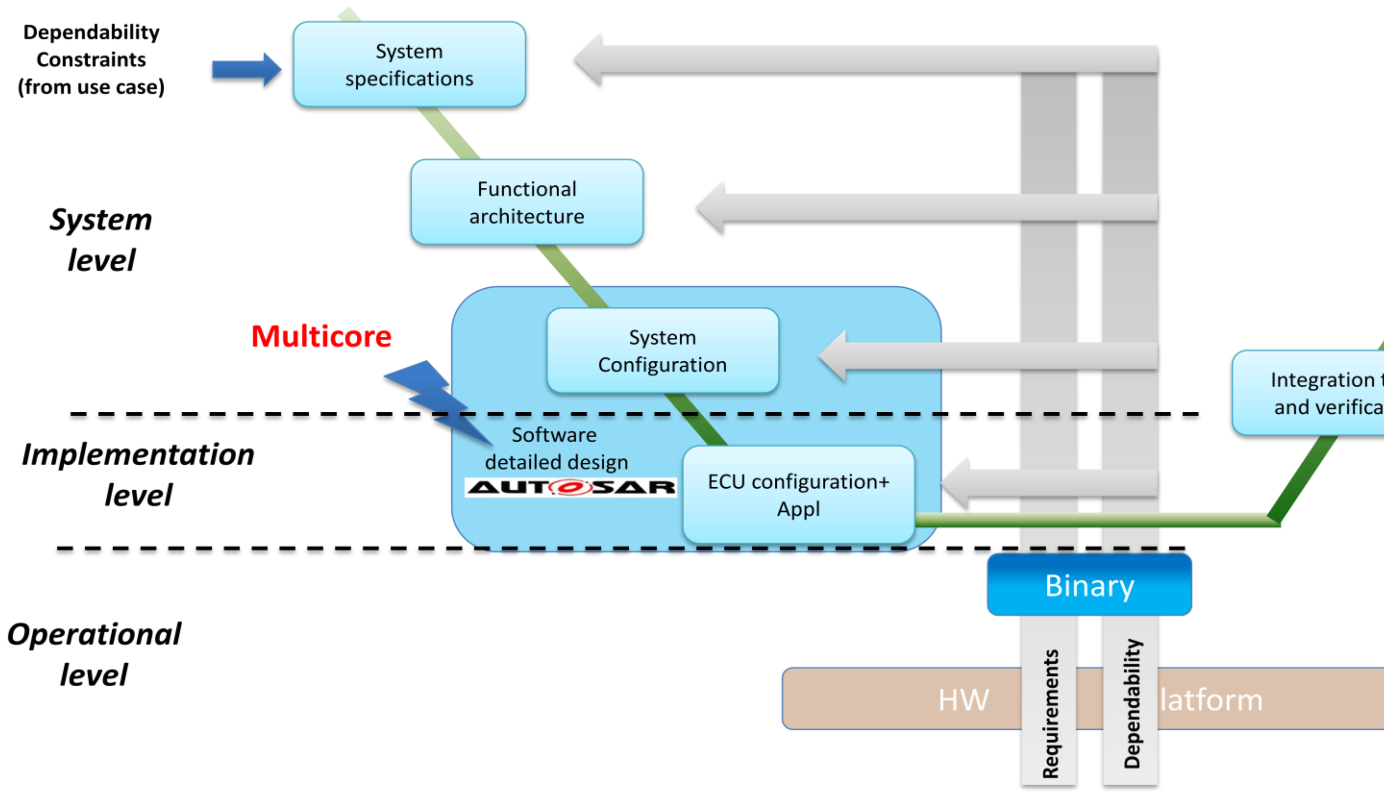
2.1.1.3 OS

As Elektrobit is our partner for this UC we choose the own the responsibility to provide the OS. It will be developed to meet CRYSTAL/VALEO use case that definition of architecture (§3.1.3) will define if EB Safety OS, EB MultiCore OS or EB Safe/MC OS shall be used

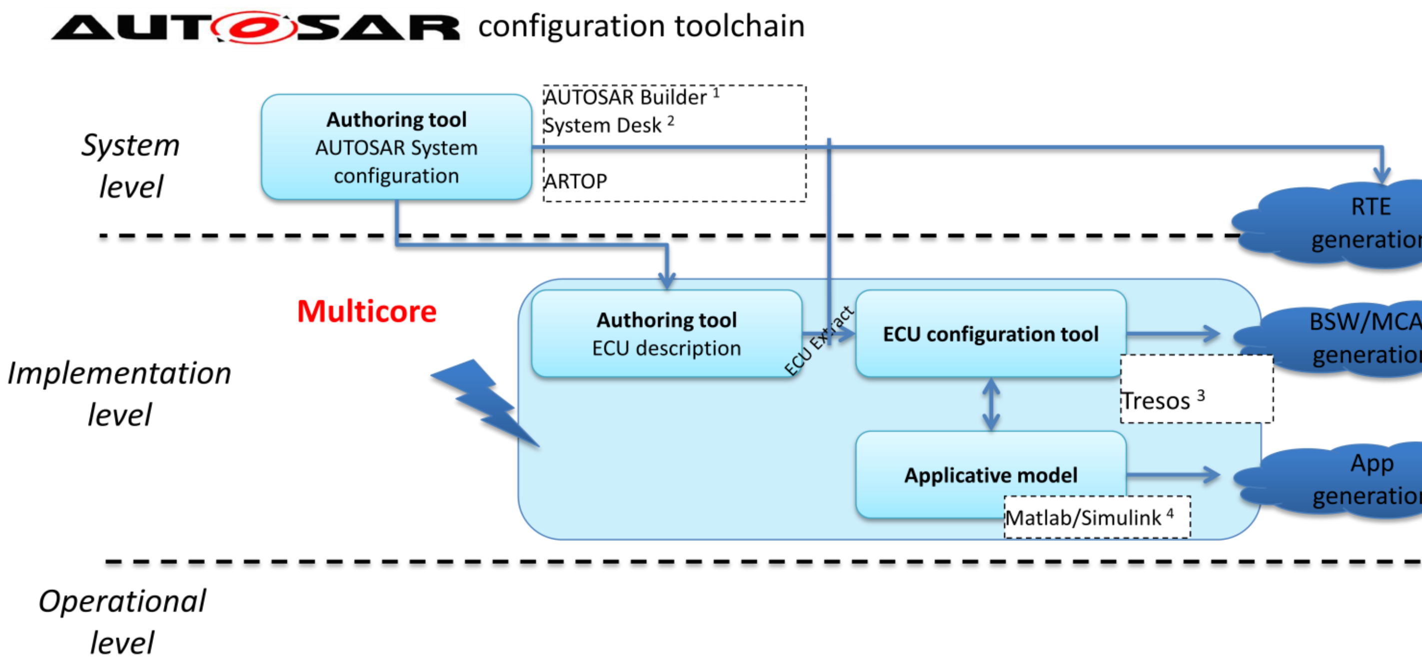
Part of the activities on EB side is to provide an OS that is compliant with both ISO26262 and Autosar 4.x. The current state is that two different product exits AUTOSAR Os and Safe OS.

2.2 Use Case Process Overview

The development will follow the AUTOSAR standard as shown below:



And below is the commonly used tools for each steps.



New requirements are introduced by OEM and ISO26262
Challenge is to be able to track the dissemination of the requirement along the development cycle.

Today's SW on single core SW execution is sequential and manage by a scheduler. To benefit from multicore increase of power SW needs to introduce task parallelism and Mutlicore OS

Challenge is to maintain real-time performances while providing computational reserve for new requirement.

Challenge is also to keep determinism.

2.2.1 Implementation

2.2.1.1 System and SW Description

Input OEM : Matlab, SW-C, Requirement as word document

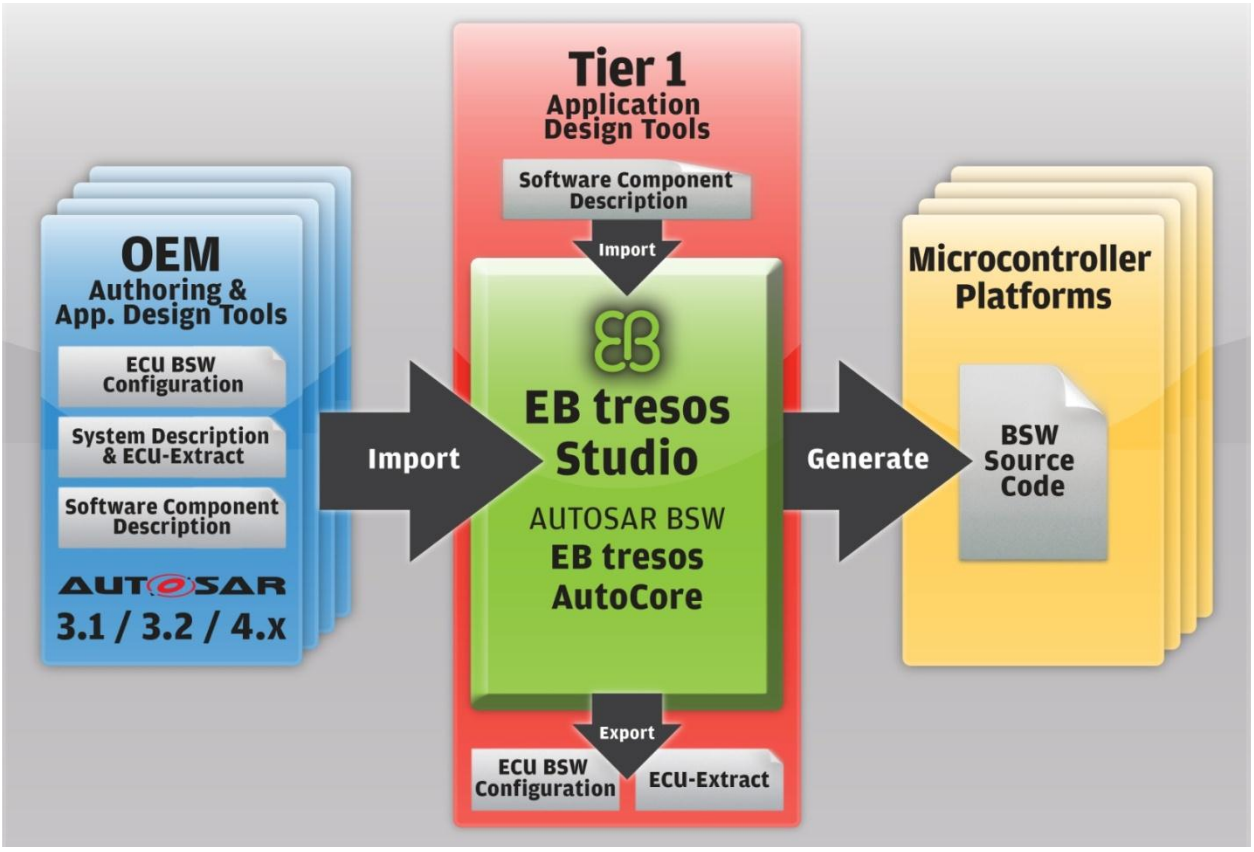
Input from Valeo : Legacy SW, Safety Requirements as word document

2.2.1.2 Configuration (EB Tresos Studio and EB Tresos Autocore)

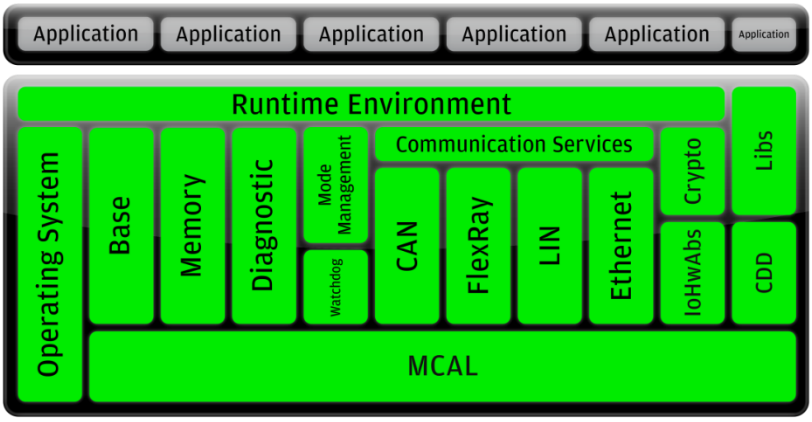
EB tresos Studio based on Eclipse is a graphical user environment allow to configure, validate and generate code of basic software modules that comply with the AUTOSAR standard. Data exchange with other tools is guaranteed through the DBC, LDF, OIL and AUTOSAR formats Import AUTOSAR Sys-D.In addition, EB tresos Studio can be extended with customer-specific basic software modules.

The openness allows to EB tresos Studio to be used as a tool for the company-wide deployment of standard software

The workflow on EB Tresos Studio consist first to import the system description file which contains system and application data + eventual additional database (like DBC,LDF files). This first operation configures automatically a large part of the Basic software. Next, you complete the configuration and validate of your Basic software and finally you generate the source code of your Basic software.



The EB tresos Studio Tool is associated to the EB tresos AutoCore base on AUTOSAR 4.0 containing the basic software for ECUs (OS, COM stack, memory management, Rte...). The EB tresos Autocore is the layer between application and the hardware" (see the figure below)



Currently this EB Tresos Toolchain (EB Tresos Studio + EB Tresos Autocore) doesn't support the development of Automotive Applications using multi-core microcontrollers in compliance with Automotive industry standard (AUTOSAR4.0, ISO26262). The goal is to extend this Toolchain in order to offer an integrated solution for developing automotive applications using multicore microcontrollers.

3 Detailed Description of the Use Case Process

3.1 Activities

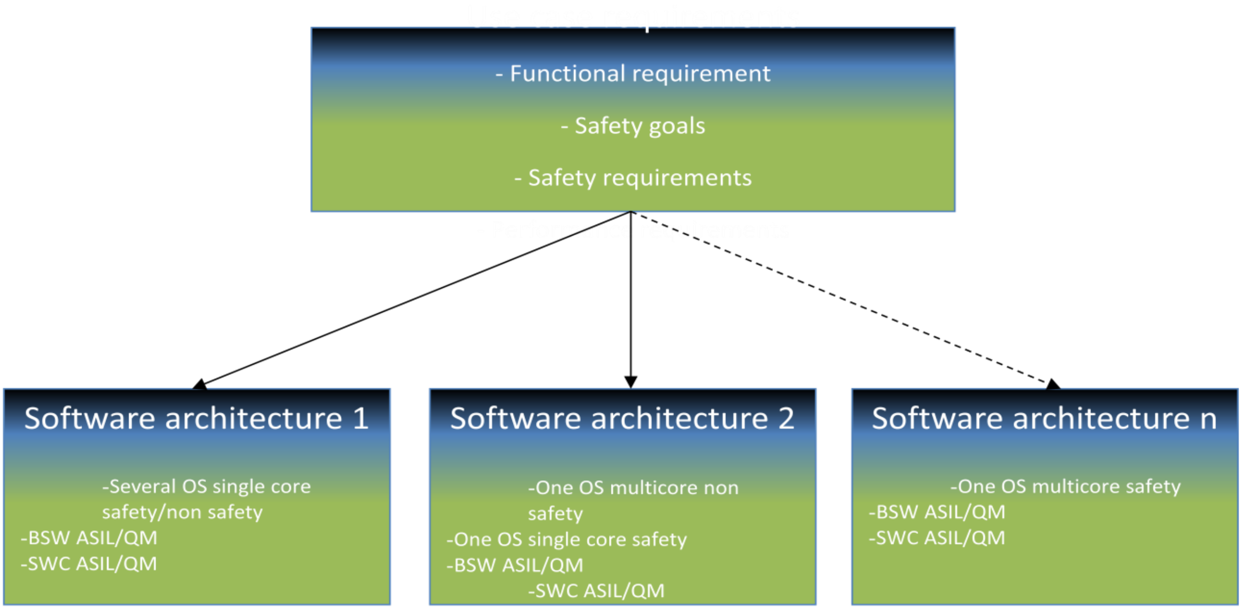
3.1.1 Use-case software architecture definition

Once the use-case analyzed, use-case requirements will be define including :

- Functional requirement
- Safety goals
- Safety requirements
- Performance requirements
- Multicore environment

Based on these requirements, we will define different possibilities of software architectures.

As the microcontroller has 3 cores in our use case, we can imagine different software architecture for our need :

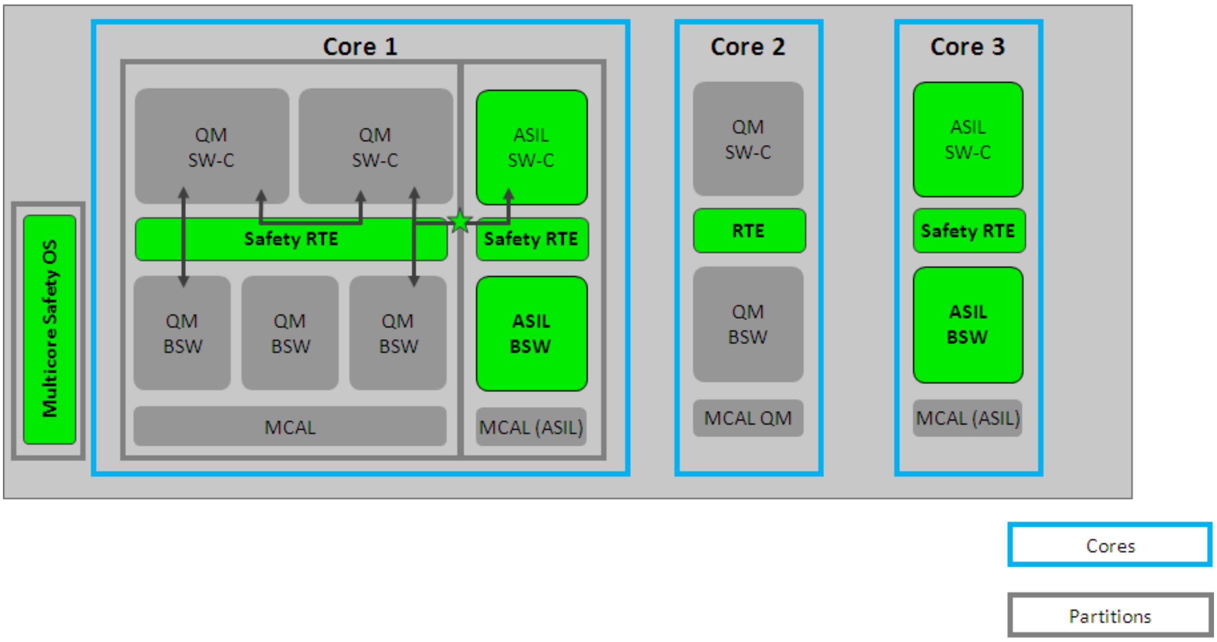


3.1.2 Development of Multicore OS and Basic Software

Based on this software architecture analysis, a preferred software architecture for the selected use-case will be defined. According to the software architecture selected different modification on the EB Tresos Toolchain will be achieved :

- Adaption of the EB Tresos Autocore to be compliant to a multicore and safety environment :
  - Development of a multicore OS prototype
  - Development of Basic software prototype
  - Development of safety mechanism for Multicore OS and Basic Software
- Adaption of the EB-Tresos studio to realize and improve the configuration of the Autocore

See below an example of software architecture, the blocks in green correspond to an example of module which should be developed or adapted in order to work on a multicore and safety environment.



Software architecture example

The Safety RTE allows to protect communication between Memory Partitions.

The multicore Safety OS allows :

- Communication between cores
- Stack Protection
- Context Protection
- OS Protection
- Hardware Error management

3.1.3 Integrate SW

Once the architecture including the safety requirements is completed, we will start the integration and configuration of the complete software which will be composed of the following parts :

- Basic software modules (BSW) : modules below the RTE
- Complex driver modules (CDD) : Additionnal module below the RTE not defined by Autosar
- Software components modules (SWC)

- RTE

First of all, we will use Jenkins tool to ensure a continous integration of the software. This tool provides several plugins for email notification, builds tools, and various version control system (e.g Subversion). A web interface is provided to view the details of the current and previous build jobs, and to configure Jenkins completely. The following automatic features will be used in the use case:

- Build the software code and the specification
- Generate the traceability between the software architecture and the test specification
- Launch the measurements
- Generate tests report

Distribution of SW-C to Cores

The purpose of activity is to define an optimum distribution of SW component across cores.

To achieve these goal criteria that impact the most real time response for the SW has to be analyzed. Once these criteria have been defined a tool will be build to parse AUTOSAR configuration file in XML format. This tool will then try different combination of SW-C placement to find the best response time for each.

3.1.4 Validate the SW

The complete toolchain will be validated by checking :

- Verifying that requirements are covered
- Check that impact of a change into input requirement can be trace from architecture to implementation

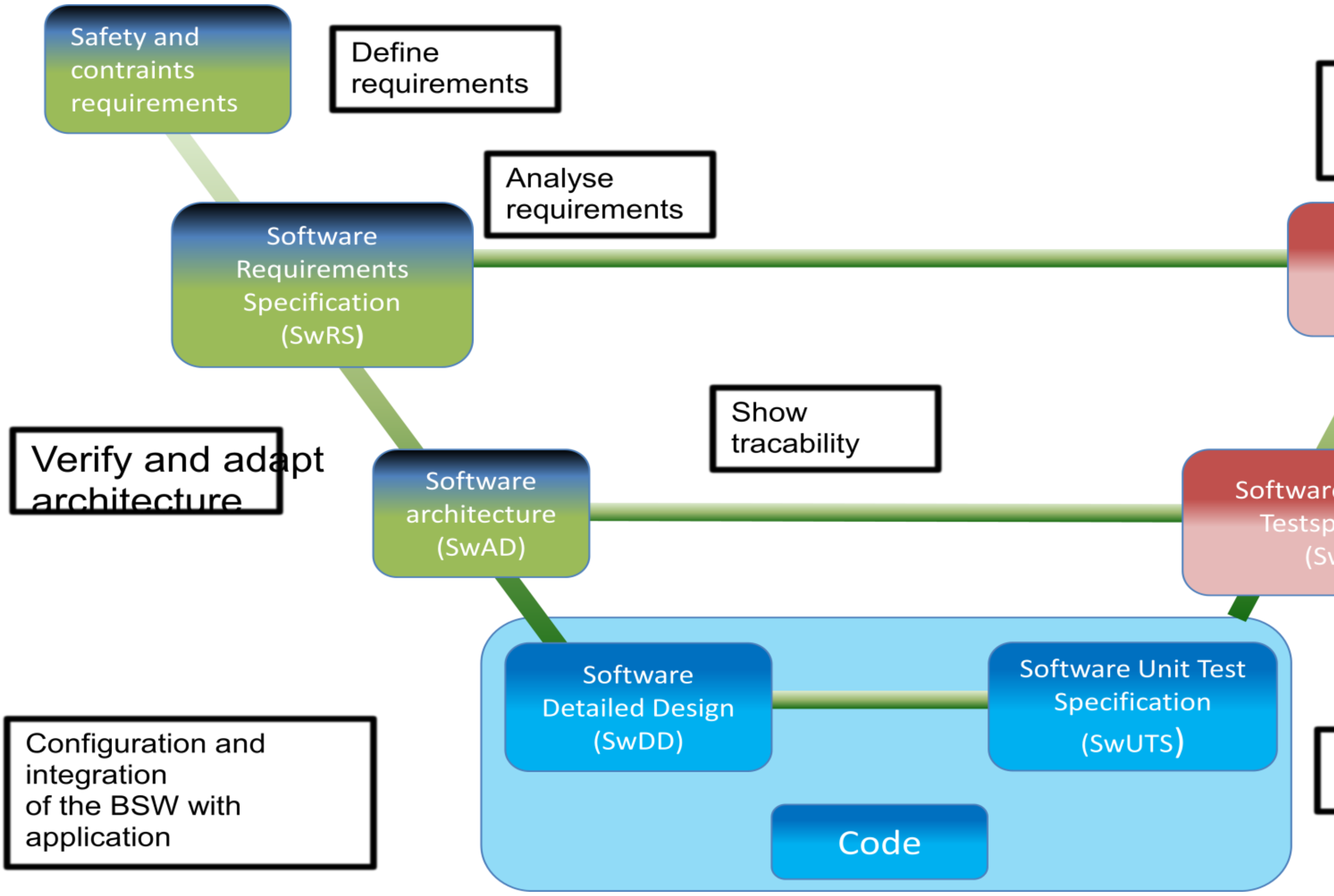
3.2 Stakeholders & Roles

Please identify the stakeholders and their roles with respect to the individual activities.

Stakeholders	**Role**
Requirement Engineer	definition of requirements
SW Architect	Define the SW architecture
Developer	Write the actual SW code
Integrator	Integrates and test the SW

4 Identification of Engineering Methods

4.1 Overview

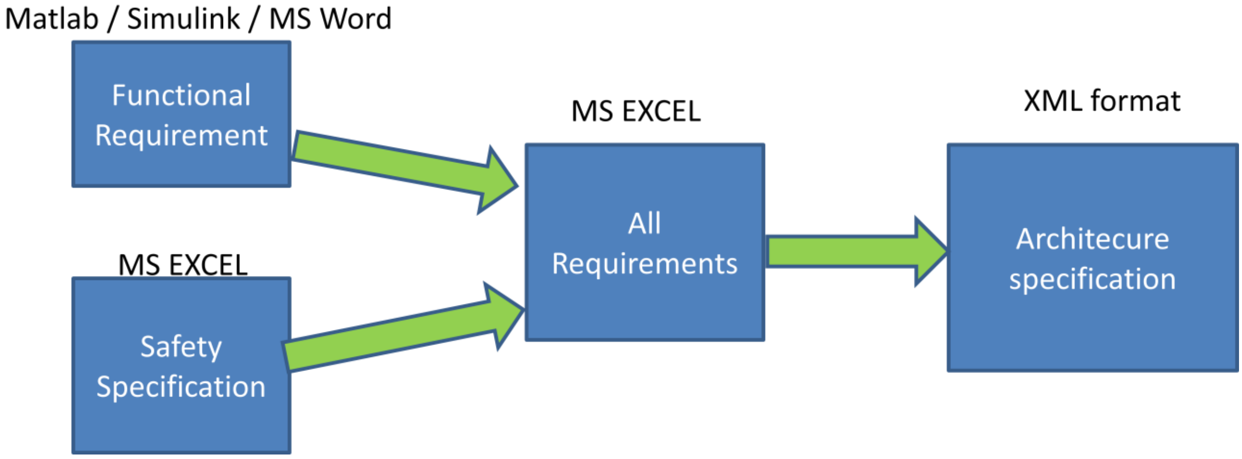


The SW will be developed and build folling the V model and with continuous integration

Number	Process Activitiy	Engineering Methods	Tools Involved
1	Define Initial Requirements	• **Formalizing of requirement**	Matlab, Simulink, MS-
2	Analyze Requirements	• **Analyze Requirement** • **Requirement verification**	REQM, Docbook, Jen
3	Define measurements Concept and achieve measurement	• **Requirement verification**	Docbook, Jenkins, RE
4	Measurements implementation	• **Configuration and measurements iteration**	Docbook, Jenkins, La
5	Show traceability between all specification	• **Requirement Traceability**	REQM, Jenkins

4.2 Formalizing of requirement

Requirements are defined using different tools. Functional requirements are often described by Matlab/Simulink models and Safety Requirements are provided in a MS-OFFICE format (WORD or EXCEL).



The purpose of this engineering method is to use one and only one input format (WORD or EXCEL) for the functional and safety requirements. This format allow with the continuous integration system (Jenkins tool described below) to :

- generate automatically the traceability
- to simplify documents review
- Facilitate the comparison of documents in our version control system (subversion)
- To generate automatically with docbook the document in .pdf format

4.3 Analyse the requirement.

4.3.1 Architecture Definition

The purpose is to collect all safety requirements, adapt and redesign the current software architecture according to these new safety requirements and in the Multicore environnement.

Autosar builder is used at this stage to define the SW architecture, to identify SW components, runnables and communication mechanisms. SW-Component partionning across cores is defined at this level too. Traceability is also managed inside the tool. Files are in arxml format.

4.3.2 Distribution of SW-C to Cores

The purpose of this engineering method is to propose an optimum distribution of SW component across cores.

To achieve these goal criteria that impact the most real time response for the SW has to be analyzed. Once these criteria have been defined a guideline will be defined to help developers to achieve the best SW-Component distribution.

4.4 Requirement verification :

Purpose of this engineering method is the verification of requirements against *measurements results*. The tool used for this method will be REQM for the requirement management.

4.5 Requirement Traceability :

The purpose of this engineering method is to validate the traceability of the requirements and the measurements. The tool used for this method will be REQM for the requirement management as well as Jenkins tools for the build the traceability report.

4.6 Configuration and measurements iteration :

The purpose of this engineering method is to automate the test measurements execution and to adapt the BSW configuration in order to ensure that the test results are correct. The automation could be managed by Jenkins tool. This tool will execute the following tasks :

- build code can be triggered periodically or on event (i.e : at each software or configuration update)
- Test execution (drive the debugger, execute the measurements)
- Test Report

5 Terms, Abbreviations and Definitions

Please add additional terms, abbreviations and definitions for your deliverable.

OS	Operating system
SW-C	Software Component as describe by Autosar
AUTOSAR	AUTomotive Open System ARchitecture

Table 5-1: Terms, Abbreviations and Definitions

6 References

Please add citations in this section.

[Author, Year]	Authors; *Title*; Publication data (document reference)

7 Annex I: Detailed Descriptions of the Engineering Methods

These are captured by the Excel templates. The Excel files will be inserted here, when this document is in the final version before it is submitted to the ARTEMIS JU

8 Annex II: Technology Base Line & Progress Beyond

This information will be collected globally, and the respective part will be inserted here. Basically it could be something like a table with a row for each engineering method and a column for the current functionality, which is the technology baseline (e.g., “data has to be transferred by hand”), and a column for the expected progress in CRYSTAL (e.g., to be implemented in CRYSYTAL / “future work”).
The exact content of this section will be defined in the next technical Board Meeting.