



D2.5: Design Tool Set – Final version

MILESTONE:	30/11/2019 (M32)
STATUS:	DRAFT
COORDINATOR:	INT
CONTRIBUTORS:	ABO, ARM, ATOS, CON, FTS, INT, MDH, RO, SICS, SMA, SOFT, SSF, UAQ, UCAN, UOC, UPAU, VTT
REVIEWERS:	MDH, TRT
DISSEMINATION LEVEL:	PUBLIC
HANDLE:	HTTP://HDL.HANDLE.NET/20.500.12004/1/P/MMART2/D2.5
LAST EDITED:	29/11/2019

Executive summary

The WP2 relay on the System Engineering Tool Set providing tools and methods to support the system modelling and model validation and verification, in the context of system design activities. The continuous development approach is the key point to enhance the potential of the MegaM@Rt2 framework to create and manage modelling and design of the large-scale and complex systems. To address this objective, WP2 investigates and develops tools and solutions able to exploit and improve the conceptual design approaches, defined within the project, that address the project case-study challenges.

This deliverable D2.5 specifies the development status, at M32 “final” milestone, of the MegaM@Rt2 System Engineering Tool Set, based on the progress of the Tasks 2.2, 2.3, and 2.4 activities. This document proposes the same approach as the previous related deliverables. The status report is split in two main sections. The first sections is focusing on the evolution of the tools (i.e. the tools roadmap), related to the System Engineering framework, in order to provide the detailed picture of the development activities in this final period. The second section refers to the framework roadmap, originally introduced and defined in the D2.2, to give a more global view of the MegaM@Rt framework evolution. The tools roadmap presents the final availability status of each planned tool functionality, evaluating, in case of cancellation, justification and impact to the project and to use case scenarios development.

The status at the M32 “final” milestone of the System Engineering Tool Set demonstrate a satisfactory completion, in line with the expected project objectives, despite the exit of Conformiq from the project. Issues related to the System Engineering Toolset coverage and related to the support to the TEK UC initially based on Conformiq’s CQDesigner tool, arose. But effective countermeasures mitigated the negative effects and allowed reaching final results successfully. Other points depicted in the deliverable are related to few dropped tool features mainly related with capabilities not required by the existing UC needs and not implying any limitation on the global project features.

The report is completed with the description of the relevant activities and initiatives, developed during this period, in particular: as part of the actions T2.2 and T2.3, the cooperation between the tools providers on specific issues and/or respective tools commonalities and, as part of the action T2.4, the hackathon initiatives and the UC support on demonstrators development activities.

Table of Contents

Acronyms	7
Introduction	9
Progress status of the Tool Set	10
Modelio (SOFT)	12
Features planned for final release (M32)	12
Modelio Constellation (SOFT)	12
Features planned for final release (M32)	13
S3D	13
Features planned for final release (M32)	13
PADRE	13
Features planned for final release (M32)	14
CHESS	14
Features planned for final release (M32)	14
Collaboro	15
Features planned for final release (M32)	15
EMFtoCSP	16
Update on delayed features at initial and intermediate milestones	16
Features planned for final release (M32)	16
Xamber	16
Features planned for final release (M32)	16
XPM	16
Features planned for final release (M32)	16
Papyrus (ATOS)	17
Features planned for final release (M32)	17
Moka (ATOS)	18
Features planned for final release (M32)	18
VeriATL	18
Features planned for final release (M32)	18
HepsyCode	19
Update on delayed features at initial and intermediate milestones	19
Features planned for final release (M32)	19
JTL	19
Features planned for final release (M32)	19
PauWare (UPAU)	19
Features planned for final release (M32)	20
CMA (RO)	20
Features planned for final release (M32)	20
CQDesigner	21
MATERA2	21
Features planned for final release (M32)	21
RCRS (SSF)	22
Features planned for final release (M32)	22

System Engineering Status, the Roadmap profile	23
Metamodel and ancillary support	23
Requirement Modelling	24
System Architecture & Design	24
System Model Validation & Verification	25
Relationships and interactions between Work Packages, partners and external projects	28
WP2, WP3 and WP4 harmonisation	28
Support to Use Case development	28
Contribution to UCs and demonstrators development	30
Opportunities for unused tools	34
The internal hackathon and tools fair initiative [INT]	36
Interaction between tools providers	37
Hepsycode - Xamber Integration	37
Hepsycode - S3D Integration	38
S3D - CHESS evaluation	41
Collaboration with external projects	43
Conclusions	45
References	46
Appendix: Hackathons and Tools Fair reports	47
Prague tools fair	47
Prague Hackathon results	48
IKER2 Challenge: AsyncAPI: Single Source of Truth for Asynchronous Applications	48
Santander Hackathon	53
BT Smoke detection system - Variability Modelling	53

Acronyms

ADL	Architecture Description Language
API	Application Programming Interface
ATL	Atlas Transformation Languages
BPMN	Business Process Model and Notation
BVR	Better Variability Result
CPS	Cyber-physical Systems
CSR	Case Study Requirement
DDS	Data Distribution Service
DoDAF	U.S. Department of Defense Architecture Framework
DSE	Design Space Exploration
DSL	Domain-Specific Language
DSML	Domain-Specific Modelling Language
EMF	Eclipse Modeling Framework
EPL	Eclipse Public License
FOSS	Free Open Source Software
GPL	GNU Public License
GUI	Graphical User Interface
HPV	Hypervisor
HW	Hardware
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
INCOSE	International Council on Systems Engineering
ISO	International Organization for Standardisation
LGPL	Lesser GNU Public License
M2C	Model To Code
M2M	Model To Model
M2T	Model To Text
MARTE	Modeling and Analysis of Real-Time Embedded Systems
MAST	Modeling and Analysis Suite for Real-Time Applications
MBD	Model-Based Development
MBSE	Model-Based System Engineering
MBT	Model-Based Testing
MDE	Model Driven Engineering
MDSD	Model Driven Software Development
MDT	Eclipse Modeling Development Tools
MFR	MMRT Framework Requirement
ML	Modelling Language
MODAF	British Ministry of Defence Architecture Framework
NFP	Non Functional Property
OCL	Object Constraint Language

OCRA	Othello Contracts Refinement Analysis
OMG	Object Management Group
OS	Operating System
PDM	Platform Description Model
PIM	Platform Independent Model
PSM	Platform Specific Model
RTES	Real-Time Embedded Systems
S3D	Single Source System Design
SOA	Service-Oriented Architecture
SUT	System Under Test
SW	Software
SysML	System Modelling Language
TCP	Tool Component Purpose
TP	Tool/Method Provider
TS-MM	MegaModelling Tool Set
TS-RT	RunTime Tool Set
TS-SE	System Engineering Tool Set
TSC	Tool Set Component
UC	Use Case
UCP	Use Case Provider
UML	Unified Modeling Language
UTP	UML Test Profile
UPDM	Unified Profile for DoDAF/MODAF
V&V	Verification and Validation
W3C	World Wide Web Consortium
WCET	Worst Case Execution Time
XMI	XML Metadata Interchange
XML	eXtensible Markup Language

1. Introduction

Similarly to the previous D2.4 and D2.3 deliverables, this one reports the progress status at M32, the “final” milestone, of the MegaM@Rt System Engineering Tool Set.

Briefly, the tool set activities are partitioned into three tasks. The task T2.2 and T2.3, to define and develop the system modelling languages, techniques, and appropriate case tools, as well as methods and tools for verification and validation of models, with particular attention to the integration of modelling artefacts in a continuous development toolchain, to ensure a rigorous development process and to early evaluate the design model correctness. The task T2.4 to support Use Case Providers adopting the MegaM@Rt approach and setting up the System Engineering framework demonstrators, to prove the validity of the proposed solutions. Feedback collected from Use Case Providers’ experience with the MegaM@Rt framework are used to constantly improve the current results.

This deliverable proposes and reuses the same approach of the previous D2.3 and D2.4, thus simplifying the evaluation and the comparison of the relative progress with respect to the previous milestones.

D2.5 is completed and released jointly with D2.6. The deliverable D2.6 provides a methodological view of the System Engineering Tool Set and the relevant contributions of each tool belonging to the MegaM@Rt System Engineering Tool Set. Therefore, the set of purposes that are developed and reported in deliverable D2.5, are contextualised in D2.6 with a description of the methodological approach proposed by MegaM@Rt project.

At the M32 “final” project milestone, a number of new features and capabilities, planned for this period, have been implemented into the proposed tool set and released to the Use Case Providers.

This deliverable presents the exact status of the project evolution using the same tools and framework roadmap tables of the previous deliverables, and it is organised as follows.

[Section 2](#) reports the tools roadmap, that is, for each tool, the detailed status of the planned development activities is presented, highlighting the ones definitely cancelled from the plan. For these activities, the justification is provided as well as the analysis of the consequences on the final demonstrators and framework release.

[Section 3](#) reports the framework roadmap, which map the development status, at M32, of the whole System Engineering Tool Set.

[Section 4](#) provides a more general and comprehensive view of the cooperation and joint activities between tools providers and the use case providers as well as the other initiatives (like internal hackathons and tools fair) that were identified as a way to push up the technical results and highlight technical issues during the plenary meetings. Technical details about the hackathon challenges addressed during the last two plenary meetings (Prague and Santander) are collected in the [Appendix](#). The Section 4 reports also the exploitable results from external EU projects, relevant to improve the MegaM@Rt framework functionalities.

2. Progress status of the Tool Set

This section is generated automatically from tool purposes defined in Modelio. It gives an overview of the implementation achievements and development status with regards to the intermediate milestone.

To briefly synthesise the reported status, it should be noted that:

- 19 tools are related to System Engineering framework,
- Among the 7th critical features not completed at the *intermediate* milestone:
 - **MODELIO-060, MODELIO-160 and S3D-060** have been completed thus now are “done”:
 - **MODELIO-060:** *Holistic system engineering support has been implemented according to specific end user scenarios;*
 - **MODELIO-160:** *Modelio Product Line Engineering capabilities have been demonstrated and used in industrial use cases*
 - **S3D-060:** *This activity has been delayed due to two reasons: 1.- CTF has been selected as the format to be used for trace generation as a consequence of the discussions during the Versailles meeting, changing our initial plans. 2.- Delays in some activities of the initial release. This delay was solved and tool has been evaluated by UC providers. In any case, first evaluation by UC providers was already done on the initial release, so small delays in the intermediate release delivery had small impact on UC plans.*
 - **EMFtoCSP-040, PAPYRUS-100, PAPYRUS-120 and PAU-050** have been “cancelled”:
 - **EMFtoCSP-040:** *Although a proposal for the techniques for incremental model verification has been proposed during the project, the implementation has been delayed indefinitely and is therefore not available in EMFtoCSP. This delay, however, does not affect either the Use Case providers or the framework development: the current prototype is fully functional, and EMFtoCSP-040 only provides a better performance without affecting such functionality;*
 - **PAPYRUS-100:** *WP4 has agreed on not developing/providing a central repository for collaborative modeling. Therefore, this feature and the PAPYRUS-110 (planned for final release) have been reevaluated. In particular, existing collaborative central model approaches based on CDO/EMF Store, and current Papyrus support have been evaluated. EMF Store is not longer maintained and CDO already offers a central repository for collaborative modelling. However, CDO has not been integrated in MegaM@Rt2 because the project consortium and industrial UC providers have decided to use a GitHub repository as a central model repository. Git collaborative tool has been adopted hence;*
 - **PAPYRUS-120:** *This baseline purpose of Papyrus was identified as relevant in early stages of the project, but could not be traced back to any framework (system) requirement, so it was discarded (D3.2, pag. 80);*
 - **PAU-050.** *Due to the lack of national funding, developers cannot be hired at the beginning of the project. As a consequence, the development of this functionality has been cancelled. The work of the one-year engineer hired on the project has focused on more important features. Indeed, nobody in the project needs a code generator from an SC-XML specification, thus it was not*

a priority. No impacts are foreseen on the collaboration with Ikerlan as they use UML diagrams and the code generator from UML state machines, corresponding to the PAU-040 capability released as planned.

- 43 features are planned for this **final** milestone,:
 - 38 purposes are in “done” status
 - 5 purposes have been **cancelled** (CHESS-110, PAPHYRUS-110, PAPHYRUS190, VeriATL-050, MATERA2-050) for the following reason:
 - **CHESS-110**: *MOSES methodology aims to early evaluate the performance of a SW system based on the simulation of its high level UML2 model. It has been successfully experimented in the past using the commercial Rational Rose-RT and, later, TAU-Telelogic modelers and simulators. Unfortunately, both tools are no longer available. Preliminary studies to evaluate and select an alternative and effective model execution tool, that is a prerequisite condition for MOSES development, provided negative results. The aim was to integrate MOSES in CHESS adopting Moka as model simulator but the related fUML do not result in an effective approach to allow an easy implementation of the platforms libraries, stimuli and probes models. Moreover, Moka provides an insufficient support to automate the test cases, “transparently” connecting software model to platform model and test bed elements. Since CHESS is not formally adopted on any UC, no impact on demonstrators and UC providers activities are foreseen so far. Other simulation strategies are available in MegaM@Rt (e.g. VIPPE, FENTISS), currently used by the UC providers. Research activities to integrate MOSES in CHESS are still in progress (e.g. integration with S3D in order to experiment VIPPE simulator) but no significant results are expected before the end of MegaM@Rt project.*
 - **PAPHYRUS-110**: *This feature follows the re-evaluation of PAPHYRUS-100. GitHub model repository and Git (as collaborative tool, already integrated in Eclipse) have been elected for providing this feature, not requiring any further implementation nor integration.*
 - **PAPHYRUS-190**: *This feature has been cancelled as its application on any industrial UC has neither been required nor identified by Atos. Resources assigned to this feature have been spent on the development of other implemented Papyrus features (PAPHYRUS-170, PAPHYRUS-180)*
 - **VeriATL-050**: *Some experiments were planned to be performed with EMF Views for integration. However, due to a stronger focus on WP4 / EMF Views for ARM (on other features of this tool), and the lack of clear need from the use cases, this feature has been cancelled.*
- The **exit of Conformiq** from the project resulted in a series of impacts, related with WP2 context, that were carefully evaluated in order to determine a valid and acceptable solution for the continuation and satisfactory conclusion of the project. Impact and countermeasures on Use Cases (in particular to the TEK UC) is discussed in the context of the T2.4 task (WP2 support to UC providers), detailed in section [4.2](#). By toolset point of view, the gap analysis has been iterated to verify the resulting coverage issues and the possible relevant solutions. The following two issues arose:
 - **SYS-060001** “The SE must support the “model-in-the-loop” validation strategy”. It was covered by a set of CQDesigner capabilities available since the “baseline” tool version. The MATERA2 tools is able to solve this gap thanks to the purposes: MATERA2-010, MATERA2-020 and MATERA2-060, made available in this final period.

- **SYS-060303** “The SE must derive integration tests (SW/SW, HW/SW components) from design model”. It was covered by the CQDESIGN-020 purpose, available since the “baseline” tool version. Also in this case, the MATERA2 tools is able to solve this gap thanks to the purposes: MATERA2-010, MATERA2-020 and MATERA2-060, made available in this final period.
- in all the other cases, the coverage of the SYS requirements is guaranteed by other available framework toolset
- All the CQDesigner tool purposes as been formally declared “*cancelled*” as specified in tool subsection [2.17](#). They are no longer counted in the summary statistics.

Summarising:

- the purposes’ **delivery plan was fundamentally respected** during the whole project development until this final release;
- slight **delays in some purpose development**, occurred in previous milestones, **were managed to avoid impacts on UC providers activities**;
- **all the cancelled purposes were carefully evaluated** guaranteeing:
 - **no impact on the project plan**;
 - **no impact on project objectives**, indeed the MegaM@Rt framework provides a complete set of functionalities that cover the user needs and the expected technical advancements beyond the current state of the art;
 - **no impact on demonstrator development** and framework evaluation activities;
 - **the cancellation of the purposes is justified** either by the absence of significant UC requirements, by a revisited framework architecture or by negative research results;
 - **the effort** not spent on the cancelled features **has been redirected on other relevant and more important capabilities**.

2.1. Modelio (SOFT)

All the features planned for previous milestone have been delivered. In particular, the MODELIO-060 and MODELIO-160, that were postponed in the intermediate version, are now implemented and in status “done”, as already mentioned in the above summary.

2.1.1. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
MODELIO-050: Modelio shall support functional properties in holistic system engineering approach.	Criticality: High Release: Final Status: done	Modelio supports functional properties in related use cases.	NOK_02, NOK_01, TRT_03, TEK_09, TRT_02, TRT_04, TRT_05
MODELIO-080: Modelio shall manage traceability on holistic system engineering level.	Criticality: High Release: Final Status: done	this tool capability is now available	BT_01, BT_05, IKER_16, IKER_18, NOK_12, NOK_14, NOK_17, NOK_18, NOK_19, NOK_20, VCE_01, TRT_05, TEK_05, IKER_18, NOK_17, NOK_18, NOK_19, TEK_05, NOK_20, TEK_01, BT_06, NOK_12

Modelio-100: Modelio shall support Model Simulation (through FMI co simulation).	Criticality: High Release: Final Status: done	Modelio supports INTO-CPS SysML profile allowing generation of FMI descriptors.	
---	--	---	--

2.2. Modelio Constellation (SOFT)

All the features planned for previous milestone have been delivered.

2.2.1. Features planned for final release (M32)

No new purposes are planned for the final release.

2.3. S3D

All the features planned for previous milestone have been delivered. In particular, the S3D-060, that was postponed in the intermediate version, is now implemented and in status “done”, as already mentioned in the above summary.

2.3.1. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
S3D-040: S3D Framework shall integrate links to schedulability analysis tools using models consistent with the generated code	Criticality: Medium Release: Final Status: cancelled	The theoretical foundations for the integration the schedulability analysis tool MAST in S3D has been developed, but some difficulties in signing engineers with experience in this context has delayed the accomplishment of the task. This delay is not critical to UC1 since schedulability analysis can be solved by FentISS tools. However this purpose will be available at the end of the project	TRT_03, TRT_04, NOK_22, AINA_04, BT_08, NOK_07, NOK_25, TRT_02, TRT_05
S3D-070: S3D Framework shall support test execution with functional and non-functional validation	Criticality: High Release: Final Status: done	this tool capability is now available	CSY_01, NOK_02, AINA_01, NOK_07, NOK_08, BT_08, BT_12, IKER_19, NOK_25, CSY_07
S3D-100: S3D Framework shall support automatic test generation for design-time component verification	Criticality: Medium Release: Final Status: cancelled	Although, as described in D2.6, the testing infrastructure has been developed, the automatic generation of new tests in order to cover certain testing requirements has not been finished. The main reason is that the development of the testing infrastructure has involved more effort than expected. Automatic test generation is a piece of work that we plan to develop in the short term before the end of the project.	NOK_01, TRT_03, TEK_09, TRT_02, TRT_04, TRT_05, BT_12, NOK_22, NOK_25, BT_12, BT_13, IKER_09, NOK_22, NOK_23, NOK_24, NOK_25, AINA_02

S3D-110: S3D Framework shall support automatic runtime management to collect information for system verification	Criticality: Medium Release: Final Status: done	this tool capability is now available	NOK_07, NOK_08, CSY_05, TRT_02, TRT_05, IKER_16, IKER_14, AINA_01, NOK_25, NOK_26, CSY_06, BT_14, IKER_12, IKER_13, NOK_25, NOK_26, AINA_02, CSY_04, CSY_05
---	--	---------------------------------------	---

2.4. PADRE

All the features planned for previous milestone have been delivered in time.

2.4.1. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
PADRE-050: PADRE shall enable different types of refactoring sessions at different levels of automation.	Criticality: High Release: Final Status: done	this tool capability is now available	IKER_01, NOK_01, TEK_02, CSY_02, CSY_01, TEK_01, BT_04, CAM_02, NOK_22, IKER_16, IKER_17, VCE_01, TRT_05, TEK_05, NOK_16, NOK_21, NOK_25
PADRE-060: PADRE shall integrate runtime traces in the performance antipatterns detection and model refactoring activities.	Criticality: Medium Release: Final Status: done	this tool capability is now available	IKER_01, NOK_01, TEK_02, CSY_02, CSY_01, TEK_01, NOK_07, NOK_08, CSY_05, TRT_02, TRT_05, IKER_16, NOK_25, IKER_16, IKER_17, VCE_01, TRT_05, TEK_05, NOK_16, NOK_21, NOK_25

2.5. CHESS

All the features planned for previous milestone have been delivered.

2.5.1. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
---------	------------	------------------------	--------------

<p>CHES-110: CHES will support performance analysis based on MOSES methodology</p>	<p>Criticality: Low Release:Final Status:cancelled</p>	<p>MOSES methodology aims to early evaluate the performances of a SW system based on the simulation of its high level UML2 model. It has been successfully experimented in the past using the commercial Rational Rose-RT and, later, TAU-Telelogic modelers and simulators. Unfortunately, both tools are no longer available.</p> <p>Preliminary studies to evaluate and select an alternative and effective model execution tool, that is a prerequisite condition for MOSES development, provided negative results. The aim was to integrate MOSES in CHES adopting Moka as model simulator but the related fUML do not result in an effective approach to allow an easy implementation of the platforms libraries, stimuli and probes models. Moreover, Moka provides an insufficient support to automate the test cases, "transparently" connecting software model to platform model and test bed elements.</p> <p>Since CHES is not formally adopted on any UC, no impact on demonstrators and UC providers activities are foreseen so far. Other simulation strategies are available in MegaM@Rt (e.g. VIPPE, FENTISS), currently used by the UC providers.</p> <p>Research activities to integrate MOSES in CHES are still in progress (e.g. integration with S3D in order to experiment VIPPE simulator) but no significant results are expected before the end of MegaM@Rt project.</p>	<p>BT_04, CAM_02, NOK_22</p>
<p>CHES-120: support of a contract runtime monitor for verification of contracts against behavioral conditions</p>	<p>Criticality: High Release:Final Status:done</p>	<p>this tool capability is now available</p>	<p>BT_01, VCE_03, VCE_02, VCE_01, NOK_12, NOK_04</p>
<p>CHES-130: product lines variability modelling support is available with the integration of the BVR open source component</p>	<p>Criticality: Medium Release:Final Status:done</p>	<p>this tool capability is now available</p>	<p>BT_01, VCE_03, VCE_02, VCE_01, NOK_12, NOK_04</p>
<p>CHES-140: CHES supports the definition and instantiation of a parametrized architecture for variability modelling</p>	<p>Criticality: Medium Release:Final Status:done</p>	<p>this tool capability is now available</p>	<p>TEK_04, VCE_05, BT_07</p>

2.6. Collaboro

All the features planned for previous milestone have been delivered in time.

2.6.1. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
---------	------------	------------------------	--------------

COLLAB-060: Collaboro shall provide both Eclipse-based and Web-based GUIs	Criticality: High Release: Final Status: done	Collaboro has been mainly implemented in the Eclipse platform, and thus, development effort has focused on the improvement of the Eclipse-based interface. Additionally, a new prototypical Web-based interface has been created as initially planned. Although the Web-based interface has been implemented, and is available, it is not included in the integrated MegaM@Rt2 framework. UOC focused on the development of the Eclipse based interface. After checking the integration approach of WP5, the Eclipse-based solution provides a tighter integration within the MegaM@Rt2 framework. The web-based interface adds unnecessary complexity to the framework; and affects the homogeneity of the solutions provided.	IKER_04, TRT_04, IKER_16,
--	--	---	---------------------------

2.7. EMFtoCSP

2.7.1. Update on delayed features at initial and intermediate milestones

All the features planned for previous milestone have been delivered with the exception of the EMFtoCSP-040 (EMFtoCSP shall allow the incremental verification of UML/OCL models to improve performance) that has been cancelled. *Although a proposal for the techniques for incremental model verification has been proposed during the project, the implementation has been delayed indefinitely and is therefore not available in EMFtoCSP. This delay, however, does not affect neither the Use Case providers nor the framework development: the current prototype is fully functional, and EMFtoCSP-040 only provides a better performance without affecting such functionality;*

2.7.2. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
EMFtoCSP-050: EMFtoCSP shall provide an extensibility mechanism to support the integration with other tools	Criticality: Medium Release: Final Status: done	Integration with other tools can be provided via plugin imports and dependencies: EMFtoCSP can be imported as a dependency from another Eclipse-based tool, and since EMFtoCSP can analyze EMF models annotated with OCL constraints, any other tool producing EMF/OCL models can use EMFtoCSP to analyze them. A more complex extensibility mechanism to support the integration with other tools based on extension points has not been implemented since it is not required by other components.	

2.8. Xamber

All the features planned for previous milestone have been delivered.

2.8.1. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
XAMBER-110: Xamber internal model shall be agnostic, that is, it does not depend on the partitioning kernel used	Criticality: High Release: Final Status: done	this tool capability is now available	CAM_01, NOK_04, NOK_12, BT_04, BT_05, BT_06

2.9. XPM

All the features planned for previous milestone have been delivered.

2.9.1. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
XPM-060: Given a XtratuM configuration file, XPM shall provide an XPM project with all the model elements required by the system	Criticality: Medium Release: Final Status: done	this tool capability is now available	CSY_01, NOK_02, BT_02, AINA_01, BT_07, BT_11, IKER_01, NOK_23, NOK_24, NOK_25, IKER_02, IKER_05, IKER_06, TEK_06, NOK_22, CSY_07, BT_01, BT_02, BT_03, CAM_01, IKER_09, IKER_10, IKER_15, IKER_18, IKER_27, NOK_04, NOK_10, NOK_11, NOK_13, NOK_23, NOK_24, NOK_25, BT_01, BT_02, BT_03, CAM_01, NOK_11, NOK_21, TEK_03
XPM-070: XtratuM Plug-in shall be able to build the whole system under a Linux-based system	Criticality: Medium Release: Final Status: done	this tool capability is now available	

2.10. Papyrus (ATOS)

All the features planned for previous milestone have been delivered with the exception of PAPYRUS-100 and PAPYRUS-120 that has been cancelled.

For PAPYRUS-100 (Central model repository to allow for collaborative modelling): *WP4 has agreed on not developing/providing a central repository for collaborative modeling. Therefore, this feature and the PAPYRUS-110 (planned for final release) have been reevaluated. In particular, existing collaborative central model approaches based on CDO/EMF Store, and current Papyrus support have been evaluated. EMF Store is not longer maintained and CDO already offers a central repository for collaborative modelling. However, CDO has not been integrated in MegaM@Rt2 because the project consortium and industrial UC providers have decided to use a GitHub repository as a central model repository. Git collaborative tool has been adopted hence;*

About PAPHYRUS-120 (Design applications through a component-based model and deploy them on heterogeneous platforms): *This baseline purpose of Papyrus was identified as relevant in the early stages of the project, but could not be traced back to any framework (system) requirement, so it was discarded (D3.2, pag. 80);*

2.10.1. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
PAPHYRUS-110: Collaborative modelling by means of a central model repository	Criticality: High Release:Final Status:cancelled	This feature follows the re-evaluation of PAPHYRUS-100. GitHub model repository and Git (as collaborative tool, already integrated in Eclipse) have been elected for providing this feature, not requiring any further implementation nor integration.	
PAPHYRUS-160: Model composition through aspect-oriented modeling (model weaving)	Criticality: Medium Release: Final Status: done	This new purpose of Papyrus was identified as relevant in the early stages of the project, but could not be traced back to any framework (system) or case study requirement. This indicated that there was no justification for the development of this feature. However, during the Versailles hackathon, Atos successfully explored the application of model composition with aspect-oriented-modeling techniques in one of the Ikerlan UCs, for the injection of logging and message validation advices (i.e. cross-cutting concerns) into their IoT Gateway component of the Skyline UC. For such a reason, this requirement was reactivated and implemented in release M32	
PAPHYRUS-180: Code generation of aspect-oriented-programming code	Criticality: Low Release: Final Status: done	This code-generation plugin generates AspectJ aspect files from UML models that describe aspect classes annotated with the AOM profile implemented in feature PAPHYRUS-170	
PAPHYRUS-190: System variability modelling through feature models	Criticality: High Release: Final Status: cancelled	This feature has been cancelled as its application on any industrial UC has neither been required nor identified by Atos. Resources assigned to this feature have been spent on the development of other implemented Papyrus features (PAPHYRUS-170, PAPHYRUS-180)	BT_01, VCE_03, VCE_02, VCE_01, NOK_12, NOK_04

2.11. Moka (ATOS)

All the features planned for previous milestone have been delivered.

2.11.1. Features planned for final release (M32)

No new purposes are planned for the final release.

2.12. VeriATL

All the features planned for previous milestone have been delivered.

2.12.1. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
VeriATL-030: VeriATL shall support incremental unbound verification of ATL model transformations	Criticality: Medium Release: Final Status: done	This tool capability is now available (but, as a research prototype, it is not integrated into the tool by default).	NOK_06, CAM_02, BT_09, BT_10, BT_11, BT_08, BT_12
VeriATL-040: VeriATL shall provide scalable unbound verification of ATL model transformations	Criticality: Medium Release: Final Status: done	This tool capability is now available (but, as a research prototype, it is not integrated into the tool by default).	BT_08, IKER_26, NOK_25, AINA_04, AINA_01, NOK_07, NOK_08, BT_08, BT_12, CAM_02, BT_09, BT_10, BT_11, BT_08, BT_12
VeriATL-050: VeriATL shall provide an extensibility mechanism to support the integration with other tools	Criticality: Medium Release: Final Status: cancelled	Some experiments were planned to be performed with EMF Views for integration. However, due to a stronger focus on WP4 / EMF Views for ARM (on other features of this tool), and the lack of clear need from the use cases, this feature has been cancelled.	

2.13. HepsyCode

2.13.1. Update on delayed features at initial and intermediate milestones

All the features planned for previous milestone have been delivered in time.

2.13.2. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
HepsyCode-040: Hepsycode shall provide a framework and related tools to suggest both the platform and the mapping of the SW onto HW architecture starting from UML models.	Criticality: High Release: Final Status: done	this tool capability is now available	CAM_02, CAM_03, CAM_02, BT_09, BT_10, BT_11, BT_08, BT_12, IKER_01, NOK_01, TEK_02, CSY_02, CSY_01, TEK_01, NOK_19, NOK_20

2.14. JTL

All the features planned for previous milestone have been delivered.

2.14.1. Features planned for final release (M32)

No new purposes are planned for the final release.

2.15. PauWare (UPAU)

All the features planned for previous milestone has been delivered with the exception of PAU-050 (PauWare shall be extended with code generation or model interpretation for SC-XML specification (currently, only a partial prototype exists)) that has been cancelled, *due to the lack of national funding, only a one-year engineer was hired on the project. She has worked on more important features. Indeed, nobody in the project needs a code generator from an SC-XML specification, thus it was not a priority.*

2.15.1. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
PAU-010: PauWare shall be integrated in the MegaM@RT tool framework for simulation of UML state machines at design	Criticality: High Release: Final Status: done	This tool capability is now available	BT_04, CSY_04, IKER_04, NOK_22, CSY_03, IKER_04, NOK_08, CSY_04
PAU-020: PauWare shall be extended with runtime verification features that could also be used for i-DSML (interpreted/executable DSML)	Criticality: High Release: Final Status: done	This tool capability is now available	BT_10, CSY_03, NOK_25, IKER_19, NOK_25, CSY_07, BT_11, NOK_25, NOK_07, NOK_08, CSY_05, TRT_02, TRT_05, IKER_16
PAU-030: PauWare shall be extended with execution trace generation features that could also be used for i-DSML (interpreted/executable DSML)	Criticality: High Release: Final Status: done	This tool capability is now available	IKER_23, NOK_25, AINA_04, BT_08, IKER_23, NOK_25, TRT_01, IKER_19, NOK_25, CSY_07, NOK_07, NOK_08, CSY_05, TRT_02, TRT_05, IKER_16

2.16. CMA (RO)

No features was planned for previous milestone.

2.16.1. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
---------	------------	------------------------	--------------

CMA-010: CMA shall make sure that each requirement is covered by functionality.	Criticality: Medium Release: Final Status: done	The tool has been reengineered without graphical interface to better support TEK UC needs.	IKER_01, NOK_01, TEK_02, CSY_02, CSY_01, TEK_01, IKER_04, NOK_01, NOK_03, TEK_02, IKER_01, CSY_02, IKER_18, NOK_17, NOK_18, NOK_19, TEK_05, NOK_20, TEK_01, BT_06, NOK_12, CSY_03, NOK_25, BT_01, BT_05, IKER_16, IKER_18, NOK_12, NOK_14, NOK_17, NOK_18, NOK_19, NOK_20, VCE_01, TRT_05, TEK_05
CMA-020: CMA shall analyze how the coverage of each requirement has been performed.	Criticality: Medium Release: Final Status: done	The tool has been reengineered without graphical interface to better support TEK UC needs.	IKER_01, NOK_01, TEK_02, CSY_02, CSY_01, TEK_01, IKER_04, NOK_01, NOK_03, TEK_02, IKER_01, CSY_02, IKER_18, NOK_17, NOK_18, NOK_19, TEK_05, NOK_20, TEK_01, BT_06, NOK_12, CSY_03, NOK_25, IKER_16, IKER_17, VCE_01, TRT_05, TEK_05, NOK_16, NOK_21, NOK_25

2.17. CQDesigner

Conformiq left the project thus the two purposes **CQDESIGN-160** and **CQDESIGN-170** planned for final release as well as all the other purposes has been formally declared “**cancelled**”, thus unavailable to the project.

2.18. MATERA2

All the features planned for previous milestone have been delivered.

2.18.1. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
MATERA2-010: MATERA2 shall provide UML based modeling and executable specifications.	Criticality: High Release: Final Status: done	This tool capability is now available.	IKER_01, NOK_01, TEK_02, CSY_02, CSY_01, TEK_01
MATERA2-020: MATERA2 shall provide UML based simulation facilities.	Criticality: High Release: Final Status: done	This tool capability is now available.	AINA_01, NOK_07, NOK_08, BT_08, BT_12

MATERA2-030: MATERA2 shall generate a human-friendly report in order to visualize the test and monitoring results.	Criticality: High Release: Final Status: done	This tool capability is now available.	BT_08, NOK_25, TRT_01
MATERA2-040: MATERA2 shall provide feedback to the modeling phase, including timing performance results.	Criticality: High Release: Final Status: done	This tool capability is now available.	AINA_04, BT_08, NOK_07, NOK_25, TRT_02, TRT_05
MATERA2-050: MATERA2 shall provide a monitoring tool to predict future functional and non-functional problems.	Criticality: High Release: Final Status: cancelled	This tool capability was cancelled due to lack of data and test infrastructure in the case studies. The effort was allocated to other tool purposes.	IKER_26, NOK_25, AINA_04
MATERA2-060: MATERA2 shall provide support for functional and non-functional test generation.	Criticality: High Release: Final Status: done	This tool capability is now available.	BT_12, BT_13, IKER_09, NOK_22, NOK_23, NOK_24, NOK_25, AINA_02
MATERA2-090: MATERA2 shall provide support for requirements traceability at model and runtime level.	Criticality: High Release: Final Status: done	This tool capability is now available.	NOK_25, IKER_18, NOK_17, NOK_18, NOK_19, TEK_05, NOK_20, TEK_01, BT_06, NOK_12

2.19. RCRS (SSF)

All the features planned for previous milestone have been delivered.

2.19.1. Features planned for final release (M32)

Purpose	Properties	Comments/Release notes	Affected CSR
RCRS-050: Systematic Translation of Simulink diagrams into Isabelle RCRS theories (systematic coverage of Simulink blocks and options, support for action/enable ports).	Criticality: High Release: Final Status: done	This tool capability is now available.	AINA_03, CSY_03, NOK_25
RCRS-060: Translation of other HBD languages (Modelica block diagrams) into Isabelle RCRS theories.	Criticality: High Release: Final Status: done	This tool capability is now available.	AINA_03, CSY_03, NOK_25, BT_04, CSY_04, IKER_04, NOK_22
RCRS-070: Advanced Formal Analyzer – apply advanced automated tools for checking model properties.	Criticality: High Release: Final Status: done	This tool capability is now available.	AINA_03, CSY_03, NOK_25, NOK_25, NOK_22

3. System Engineering Status, the Roadmap profile

This section tracks, for the “FINAL” milestone MS5 at M32, the MegaM@Rt Tool Set implementation plan and the enhancements or upgrades of the tools features. For each SYS requirement the roadmap table, as defined in [D2.2](#), that specifies the plan for the availability and integration of the relevant tool purposes, is reported, synthesizing the current status of the tools developments. Also in this case, the following tables are automatically generated from Modelio.

As already summarised in the previous section, the current status of the MegaM@Rt tool set development successfully reach the final milestone based on the defined plan.

The cancelled capability was carefully evaluated avoiding impacts on the global framework functionalities and the UC providers evaluations.

The criticisms raised by the leaving of Conformiq has been mitigated and solved refining the traceability of new tools purposes. In particular, the SYS-060001 and the SYS-060303 requirements, initially covered by CQDesigner’s baseline purposes, have been remapped to MATERA2 new functionalities delivered during this final period.

For readability, the “cancelled” purposes are highlighted in red colors.

The Conformiq CQDesigner tool purposes are maintained in the following tables but marked with strikeout characters.

3.1. Metamodel and ancillary support

Framework Features	Baseline	Initial	Intermediate	Final
SYS-010100: The SE must support standard modelling languages, standard profiles (i.e. AADL, UML, SysML, MARTE, fUML, UTP) and profile customisation capability.	MODELIO-010 MODELIO-040 S3D-010 S3D-120 PADRE-010 CHESS-010 EMFTOCSP-010 XAMBER-020 XAMBER-040 PAPYRUS-010 MOKA-010 MOKA-020 MOKA-030 VeriATL-010 JTL-010	MODELIO-141 PADRE-020	PADRE-030 HepsyCode-030	PADRE-050 [done] PADRE-060 [done] HepsyCode-040 [done] CMA-010 [done] CMA-020 [done] MATERA2-010 [done]
SYS-010200: The SE must support domain specific modelling languages (i.e. EAST-ADL, FDB).	PAPYRUS-020 PAPYRUS-030 JTL-010	MODELIO-141 COLLAB-030	COLLAB-010	CMA-010 [done] CMA-020 [done]
SYS-010400: The SE must provide the mean to generate project and design documents from the model artifacts		MODELIO-020 XPM-010 XPM-020		XPM-060 [done]

SYS-000001: The SE must support a teamwork collaboration environment.	M.CSTEL-010 M.CSTEL-020 M.CSTEL-030 COLLAB-050 PAPYRUS-080 PAPYRUS-090 HepsyCode-010 QQDESIGN-010	MODELIO-020 COLLAB-020 COLLAB-030 COLLAB-040	COLLAB-010 PAPYRUS-100 cancelled HepsyCode-030	COLLAB-060 [done] PAPYRUS-110 cancelled
SYS-000002: The SE must provide an advanced graphical user interface (GUI) to simplify the modelling activities.	MODELIO-010 XAMBER-010 MOKA-040 QQDESIGN-010	XPM-010	XAMBER-090	COLLAB-060 [done]

3.2. Requirement Modelling

Framework Features	Baseline	Initial	Intermediate	Final
SYS-020201: The SE must allow modelling the functional requirements based on stakeholder needs	MODELIO-030 CHESS-020 XAMBER-010 PAPYRUS-040 RCRS-010			MODELIO-050 [done]
SYS-020202: The SE must allow modelling the non-functional/extra-functional requirements and constraints (e.g. execution delay, power consumption, etc.)	MODELIO-040 CHESS-020 XAMBER-010 PAPYRUS-040 PAPYRUS-050 HepsyCode-020 QQDESIGN-020	S3D-030 S3D-050 XPM-010	MODELIO-060 XAMBER-070 XAMBER-090	S3D-070 [done] XPM-060 [done]
SYS-020300: The SE must extend the "intra-model" requirements traceability across the whole set of a model artifacts	MODELIO-070	MODELIO-020		MODELIO-080 [done] CMA-010 [done] CMA-020 [done] MATERA2-090 [done]

3.3. System Architecture & Design

Framework Features	Baseline	Initial	Intermediate	Final
SYS-000004: The SE must support the HW/SW co-design.	S3D-020 PAPYRUS-070 HepsyCode-010 HepsyCode-020	MODELIO-020 S3D-030 S3D-050 XPM-040	XPM-050	HepsyCode-040 [done]

SYS-000005: The SE must provide smarter design exploration techniques based on a pre-analysis of the model	HepsyCode-010 CODESIGN-020	EMFTOCSP-020 EMFTOCSP-030	EMFTOCSP-040 cancelled VeriATL-020	VeriATL-030 [done] VeriATL-040 [done] HepsyCode-040 [done]
SYS-030101: The SE must support the architectural views definition and modelling.	MODELIO-030 XAMBER-010 XAMBER-030 XAMBER-040 XAMBER-050 HepsyCode-020	MODELIO-020	XAMBER-070 XAMBER-080	HepsyCode-040 [done]
SYS-030301: The SE must provide modelling adopting separation of concerns principle.	MODELIO-030 S3D-010 CHESS-030 CHESS-040 PAPYRUS-040	MODELIO-020	XAMBER-070 XAMBER-080	
SYS-030401: The SE must support the system variability modelling		MODELIO-020	MODELIO-160	CHESS-120 [done] CHESS-130 [done] PAPYRUS-190 cancelled
SYS-030402: The SE must support the reuse of existing models or patterns	XAMBER-050 RCRS-010	MODELIO-020		XAMBER-110 [done]
SYS-040201: The SE must provide the means to model non-functional/extra-functional requirements using specific profiles (i.e. MARTE) or annotations, containing natural language text or formal language formula (i.e. OCL)	MODELIO-040 S3D-010 S3D-120 EMFTOCSP-010 PAPYRUS-050 PAPYRUS-060 MOKA-010 MOKA-040 VeriATL-010 JTL-010 RCRS-010		MODELIO-060	MODELIO-050 [done] S3D-100 [done]

3.4. System Model Validation & Verification

Framework Features	Baseline	Initial	Intermediate	Final
SYS-060000: The SE must provide advanced test design capabilities to support the verification and the validation of both the SE system model and the system application	S3D-020 EMFTOCSP-010 XAMBER-010 VeriATL-010 CODESIGN-020 CODESIGN-070 CODESIGN-080 CODESIGN-100 CODESIGN-110 CODESIGN-120 CODESIGN-130 CODESIGN-140	EMFTOCSP-020 EMFTOCSP-030 XPM-030	S3D-060 S3D-090 VeriATL-020	S3D-070 [done] VeriATL-040 [done] CODESIGN-160 cancelled CODESIGN-170 cancelled MATERA2-020 [done]

SYS-060001: The SE must support the "model-in-the-loop" validation strategy.	QQDESIGN-020 QQDESIGN-070 QQDESIGN-080 QQDESIGN-090 QQDESIGN-110 QQDESIGN-130			MATERA2-010 [done], MATERA2-020 [done], MATERA2-060 [done]
SYS-060002: The SE must refine the verification and validation test cases based on the data collected during runtime executions (e.g. logs, traces, etc..).		S3D-080		S3D-110 [cancelled] PADRE-060 [done] PAU-020 [done] PAU-030 [done]
SYS-060003: The SE should provide a model execution engine to support model analysis like verification & validation tests, dependability, schedulability, time, performance analysis, etc..	RCRS-030		XAMBER-070 XAMBER-080 PAU-050 [cancelled] RCRS-080	MODELIO-100 [done] PAU-010 [done]
SYS-060101: The SE must check the model syntax correctness based on metamodel definitions.	MODELIO-010 XAMBER-010 XAMBER-060 XAMBER-100 PAPYRUS-010 MOKA-010 QQDESIGN-020 QQDESIGN-090 QQDESIGN-110 QQDESIGN-130	XPM-030	JTL-030	
SYS-060102: The SE must provide model's quality metrics based on design guidelines.	COLLAB-050 QQDESIGN-020	MODELIO-020		CHESS-140 [done]
SYS-060103: The SE must provide the support to certify the "safety integrity level" (SIL) of the system based in IEC61508 standard.				CHESS-140 [done]
SYS-060301: The SE must derive system validation and verification tests from requirements model.	EMFTOCSP-010 QQDESIGN-020 QQDESIGN-030 QQDESIGN-040 QQDESIGN-050 QQDESIGN-070 QQDESIGN-080 QQDESIGN-100	EMFTOCSP-020 EMFTOCSP-030	S3D-060 EMFTOCSP-040 [cancelled]	
SYS-060302: The SE must derive components (i.e. modules) tests from design model.	QQDESIGN-020		S3D-060	
SYS-060303: The SE must derive integration tests (SW/SW, HW/SW components) from design model.	QQDESIGN-020			MATERA2-010 [done], MATERA2-020 [done], MATERA2-060 [done]
SYS-060304: The SE must provide incremental verification based on the changes made on the model since the last check to minimize the model subset to reverify.	QQDESIGN-020		EMFTOCSP-040 [cancelled]	VeriATL-030 [done]

SYS-060305: The SE should be able to provide smart bounds pre-analysing of the model to suggest better bug searching areas.	QQDESIGN-020 QQDESIGN-080	EMFTOCSP-030		QQDESIGN-160 [cancelled] QQDESIGN-170 [cancelled]
SYS-060306: The SE should be able to execute a time-constrained verification to stop the verification after a given time amount, associating the partial result with some kind of confidence value regarding that answer.	QQDESIGN-020	EMFTOCSP-020		
SYS-060307: The SE should support assertion-based verification strategy.	QQDESIGN-020 QQDESIGN-090 QQDESIGN-110 QQDESIGN-120 QQDESIGN-130 QQDESIGN-140 RCRS-010 RCRS-040	EMFTOCSP-020		RCRS-070 [done]
SYS-060401: The SE must support the generation of the simulation models from the system model	MOKA-020 MOKA-030 MOKA-050 RCRS-020 RCRS-030	S3D-050	MOKA-090 PAU-040 PAU-050 [cancelled] RCRS-080	PAU-010 RCRS-060 [done]
SYS-060501: The SE must support the generation of the performance analysis models from the system model		S3D-050	JTL-030	PADRE-050 [done] CHESS-110 [cancelled]
SYS-060600: The SE must support the time and schedulability analysis of the system model.	CHESS-050 CHESS-060 CHESS-070 CHESS-080 CHESS-090		XAMBER-070 XAMBER-080	S3D-040 [cancelled]

4. Relationships and interactions between Work Packages, partners and external projects

4.1. WP2, WP3 and WP4 harmonisation

The coordination between WP2, WP3 and WP4 continued following the criteria and the basic principles already indicated in the previous deliverables.

The WP leaders and deliverable leaders are working together to maintain and ensure a significant level of consistency regarding all the outputs of the project. In particular for this final milestone, the coordination involves the final status reports deliverables (D2.5, D3.5 and D4.4) and the tools guidelines deliverables (D2.6, D3.6 and D4.5).

4.2. Support to Use Case development

TOOLS	APPLIED IN UC	REMARKS
MODELIO	03_IKER, 04_TEK, 05_NOK, 06_BT, 07_VCE	EXPERIMENTS IN PROGRESS, SEE §4.2.1 COLLABORATION BETWEEN IKERLAN AND SOFTEAM COLLABORATION BETWEEN BT, VCE, MDH AND SOFTEAM
MODELIO CONSTELLATION	03_IKER, 05_NOK	EXPERIMENTS IN PROGRESS, SEE §4.2.1 COLLABORATION BETWEEN IKERLAN AND SOFTEAM
S3D	01_TRT	EXPERIMENTS IN PROGRESS FOR TRT CASE, DIRECTLY AND IN COLLABORATION WITH HEPSCODE TOOL, SEE §4.2.1. COLLABORATION BETWEEN TRT AND UCAN AND §4.3.2 HEPSYCODE - S3D INTEGRATION
PADRE	04_TEK, IKER	EXPERIMENTS IN PROGRESS FOR TEK CASE, SEE §4.2.1 COLLABORATION BETWEEN TEK, UAQ, RO AND CONFORMIQ CONTACTS WITH IKER, SEE §4.2.1 COLLABORATION BETWEEN IKER AND UAQ .
CHESS		NOT APPLIED YET, SEE §4.2.2 CHESS
COLLABORO		SEE §4.2.2 COLLABORO . AFTER STUDYING ITS APPLICATION TO THE IKER CASE STUDY (SEE COLLABORATION BETWEEN UOC AND IKER), THIS USE CASE HAS LED TO THE DEVELOPMENT OF THE CHALLENGE IKER2 CHALLENGE: ASYNCAPI: SINGLE SOURCE OF TRUTH FOR ASYNCHRONOUS APPLICATIONS
EMFtoCSP		NOT APPLIED, SEE §4.2.2 EMFtoCSP
XAMBER	01_TRT	EXPERIMENTS IN PROGRESS,

		SEE §4.2.1 COLLABORATION BETWEEN TRT AND FTS AND
XPM		NOT APPLIED YET, SEE §4.2.2 XPM
PAPYRUS		PRELIMINARY CONTACT WITH IKER DURING VERSAILLES HACKATHON TO ADOPT ASPECT ORIENTE MODELING (AOM) IN THEIR USE CASE. CONTACT RESUMED IN NOVEMBER 2018. A COMPLETE MODEL OF THE IKERLAN SKYLINE SYSTEM WITH MODELLED ASPECTS FOR ASPECTJ INJECTION OF ADVICES WAS IMPLEMENTED AND SHOWN TO IKERLAN IN PRAGUE AND SANTANDER HACKATHONS. A REFACTORED PROJECT OF IKERLAN SKYLINE SYSTEM WAS REFACTORED FROM ORIGINAL SOURCES FOR ASPECTJ COMPATIBILITY. SEE COLLABORATION BETWEEN ATOS AND IKER
MOKA	IKER	INITIAL DISCUSSION WITH ABO DURING PARIS HACKATHON FOR JOINT COLLABORATION IN MODEL EXECUTION USING MOKA. INITIAL DISCUSSION WITH IKERLAN DURING PARIS. SIMULATION OF IKERLAN SKYLINE SYSTEM WITH MOKA WAS IMPLEMENTED AND PRESENTED TO IKERLAN IN PRAGUE AND SANTANDER HACKATHONS. EXTENDED SKYLINE MODEL SIMULATION WAS IMPLEMENTED (BASED ON SKYLINE CODE BASE) BY SANTANDER HACKATON AND OFFER TO ABO FOR TEST GENERATION USING MATERA2 AFTER SANTANDRE MEETING. SEE COLLABORATION BETWEEN ATOS AND IKER
VERIATL		NOT APPLIED YET, SEE §4.2.2 VERIATL
HEPSYCODE	01_TRT	EXPERIMENTS IN PROGRESS, SEE COLLABORATION BETWEEN TRT AND FTS AND §4.3.1 HEPSYCODE - XAMBER INTEGRATION ON JOINED ACTIVITY WITH FTS TO TRANSFORM THEIR TRT MODEL INTO COMPLIANT HEPYCODE MODEL EXPERIMENT IN PROGRESS, SEE §4.3.2 HEPSYCODE - S3D INTEGRATION ON JOINED ACTIVITY WITH UCAN TO INTEGRATE AND TO COMPARE HEPYCODE AND S3D METHODOLOGIES (MORE DETAILS WILL BE REPORTED IN WP5, IN PARTICULAR D5.3)
JTL	02_CSY, 04_TEK	EXPERIMENTS IN PROGRESS FOR TEK USE CASE, SEE §4.2.1 SUPPORT TO TEK UC . JOINED ACTIVITY WITH PADRE TOOL AND RO PARTNER. EXPERIMENTS IN PROGRESS FOR 02_CSY, SEE §4.2.1 COLLABORATION BETWEEN ARM, CSY AND UAQ . CONTACTS WITH IKER, SEE §4.2.1 COLLABORATION BETWEEN IKER AND UAQ .
PAUWARE	02_CSY, 03_IKER, 08_AINA	EXPERIMENTS IN PROGRESS, SEE §4.2.1, COLLABORATION BETWEEN IKER AND UPAU
CMA		EXPERIMENTS IN PROGRESS FOR TEK UC, SEE §4.2.1, SUPPORT TO TEK UC
CONFORMIX DESIGNER	04_TEK, 05_NOK,	CONFORMIX LIVED THE PROJECT

	06_BT, 07_VCE	
MATERA2	03_IKER, 05_NOK, 06_BT	SEE §4.2.1 COLLABORATION BETWEEN NOK, IKER, BT, ATOS, AND ABO'S MATERA2
RCRS	06_BT	SEE §4.2.1 Collaboration between BT and SSF

4.2.1. Contribution to UCs and demonstrators development

This section provides a brief update on the relevant issues and solutions during demonstrators preparation in the context of the cooperation between UC and Technology providers. The parallel deliverable D2.6 demonstrates the methodology approach supported by the tools and provides concrete examples of the tools instantiation on relevant UCs.

Support to TEK UC

Since TEK UC was mainly based on CQDesigner tool, the support has been discontinued due to the Conformiq exit from the project. Several countermeasures have been adopted to solve this issue.

As a first approach, TEK, actively supported by RO, is implementing the demonstrators based on the tools and technologies consolidated in their practice, but not part of the MegaM@Rt framework.

The current effort of the project partners is concentrated in defining and implementing a suitable alternative solutions using MegaM@Rt toolset. Evaluations are in progress on MATERA2, CertifyIt, Mbeetle and CHESS tools, to complete the whole chain of scenarios defined by the TEK UC. Additional developments to extend MegaM@Rt toolset capabilities are also under evaluation by RO revisiting and improving their CMA tool features with new implementations.

The final results and a complete description of the TEK UC implementation will be provided in the context of the relevant WP5 deliverables planned by the end of the project.

Collaboration between IKER and SOFT

IKER and SOFT have continued their collaboration regarding its Modelio usage for Java generation from DDS modelling. A Modelio extension has been developed and is currently used by IKER on one use case. The DDS designer has been ported from Modelio version 3.6 to Modelio 3.8.

Collaboration between BT, VCE, MDH and SOFT

BT, VCE and MDH have continued their collaboration with SOFT related to system modelling using Modelio and the automatic extraction of metadata from requirement documents. The resulting model is used by other tools such as CompleteTest and Mbeetle for test generation and execution. Evaluations on real software in two use cases has been performed.

In addition to that VCE and SOFT worked on the Variability modelling in the context of the automotive scenario. In this context SOFT implemented a Modelio extension called Variability Designer that is compatible with Variability Exchange Language (VEL) and pure::variants tool by pure::systems.

Collaboration between TRT and UCAN

TRT and UCAN have continued their collaboration focused on the avionics TRT use case. During the reporting period, the collaboration has focused on the application to the use case and the corresponding evaluation of the features added to the S3D methodology and developed in the S3D toolset. These activities have focused on the exploration of the implementation details of the different communication semantics and models of computation proposed in the S3D methodology, the definition and generation of different memory spaces for the application, the use of different communication channels for their interconnection, and the interaction with the trace generation and analysis infrastructure developed in WP3. Evaluations through native execution and simulation have been performed.

Collaboration between IKER and UAQ

Concerning the collaboration between Ikerlan and University of L'Aquila, a step ahead has been made. We are currently working on defining rules in order to link runtime logs, i.e., logs generated within the Ikerlan's case study, and design elements, i.e., elements within the design model. Furthermore, JTL will be used to create traceability links among logs and design elements, while PADRE will be used to detect performance flaws, and eventually to remove them by changing the initial configuration (i.e., by applying refactoring actions, which are available within PADRE's portfolio).

Collaboration between TRT and FTS

see details reported in [HEPSYCODE - XAMBER INTEGRATION](#)

Collaboration between NOK, IKER, BT, ATOS, and ABO's MATERA2

Åbo Akademi University (ABO)'s MATERA2 is a tool suite that comprises six components: (1) a model-based monitoring and online testing tool called MATERA2-MBMÅA, (2) a model-based performance and load testing tool called MATERA2-MBPeT, (3) a model-based testing tool for executable UML models called MATERA2-AIfTester, (4) a genetic algorithm based performance space exploration tool for web applications called MATERA2-PerfXGA (5) a reinforcement learning based exploratory performance testing tool called MATERA2-iPerfXRL, and (6) a model-based conformance testing tool called MATERA2-ADCT.

In WP2, ABO has participated in the Nokia (NOK) and IKER case studies with two tools: MATERA2-AIfTester and MATERA2-ADCT. Moreover, MATERA2-MBMÅA has been applied to the BT case study. MATERA2-AIfTester generates test inputs for executing and testing fUML activity diagrams containing AIf code. Similarly, MATERA2-ADCT provides a model-based conformance testing approach for system specifications modelled as UML activity diagrams. MATERA2-AIfTester and MATERA2-ADCT were developed and experimented to simulate and test some executable models in the NOK and IKER case studies by generating test inputs that maximize model coverage. ABO has also participated in the NOK and IKER case study discussions at the Paris and Santander Hackathons. The executable UML models in the IKER case study have been developed in cooperation with ATOS. ABO has collaborated with ATOS to apply MATERA2-AIfTester and MATERA2-ADCT on the executable IKER models to generate and execute tests for model-based conformance testing. MATERA2-MBMÅA was applied on BT's case study to perform model-based monitoring and online testing. ABO and BT also had a discussion at the Helsinki Hackathon.

Collaboration between IKER and UPAU

The collaboration has started in previous hackathons and has continued in the last one, in Santander. Ikerlan is modeling UML state machines using the Enterprise Architect tool and is generating a specific code for implementing these state machines. The idea of the collaboration is to use the generic PauWare code generator for developing UML state machine based applications using the PauWare execution engine. PauWare has a Java ME version so it can be used for the IoT Gateway use case of Ikerlan. The PauWare code generator is taking as entry an XML file that can be defined with any UML

modeler and is then independent of the tool used to define the state machine. PauWare Java code has been successfully generated from their first models but it requires to be fully validated on a complete and more complex model.

Collaboration between UOC and IKER

The collaboration that started in previous hackathons between UOC and IKER (i.e., plenary meetings in Helsinki and Paris-Versailles) led to the creation of a DSL for the specification of message-driven APIs using Collaboro. Nevertheless, after some initial experiments and after inspecting the state of the art, it was decided that the usage of Collaboro was not needed, opting for the adoption of AsyncAPI¹. As such the collaboration has evolved in the development of a prototypical DSL and accompanying toolkit as reported in [IKER2 Challenge: AsyncAPI: Single Source of Truth for Asynchronous Applications](#). The initial works have been already published in a national conference (only in spanish) [[Gomez19](#)].

Collaboration between Atos and IKER

Papyrus

Papyrus AOM/AspectJ code generation (Aspects) and fUML modeling.

These extensions to Papyrus enable the modeling of cross-cutting concerns, by adopting Aspect Oriented Modeling (AOM) techniques, in particular for targeting the AspectJ language (WP2). The Papyrus UML profile for AspectJ can be adopted to model cross-cutting concerns as aspects within the system modeling. Aspect models can refer to joint points within the model directly or using pointcuts, and refer to the advices designed elsewhere within the model.

IKERLAN has a case study name Skyline, developed with MicroJava, where IoT devices exchange messages of different kinds. IKERLAN declared its need to verify the conformance of exchanged messages to certain specifications, which is message dependent, in order to ensure the correctness of the IoT devices. Avoiding to entangle message conformance verification code within the main business code is a factor; therefore this cross-cutting concern could be managed with AspectJ techniques at model level.

The suitability of this technology was evaluated by IKERLAN and Atos together in the Paris Hackathon, and this collaboration continued in following hackathons. Atos refactored the existing code base of the Skyline system (in particular the IoT gateway that manages the incoming messages) to centralize the message management code, so that the aspect oriented engineering technologies could be easily applied. This refactored software was sent back to Ikerlan for their inspection and validation. Atos designed, using Papyrus, a UML model for the Skyline system, Then, Atos designed aspects for logging and model verification for Skyline, that were profiled with the AOM Profile for UML that Atos implemented and integrated within the Papyrus tool. This model was presented to Ikerlan in Prague and Santander hackathons. Then, Ikerlan evaluated this technology for modeling aspects for their Skyline system.

Similarly, a complete fUML model for Skyline was implemented by Atos, using Papyrus and fUML support and validation (using the Moka validation extension implemented by Atos for the Papyrus tool). This fUML model focuses on describing the internal behavior of the Skyline system, by specifying:

- The internal behavioral state a UML state machine,
- The internal behavioral process for each Skyline state as a set of activity diagrams,
- Other aside supporting behaviors specified as activity diagrams,
- Structural representation of the Skyline system as class diagrams,
- Simulation scenarios that configures messages sent to the Skyline system.

This complete Skyline model was sent back to Ikerlan for inspection, verification and validation.

¹ <https://www.asyncapi.com/>

Other use case requirements, identified from Deliverable [D2.2](#), where these Papyrus technology can be applied are: IKER_01, NOK_03, NOK_22.

Moka

Moka is a fUML model execution engine integrated within the Papyrus UML editor. Moka enables the execution of behavioral UML2 models.

Moka has been adopted by Atos and Ikerlan in the specification and simulation of the Ikerlan Skyline system. Atos developed a detailed, executable, fUML model with Papyrus (see above section) and used Moka fUML execution engine (integrated within Papyrus) to simulate the Skyline system under different scenarios. This model was presented to Ikerlan in Prague and Santander hackathons. As a result of the feedback received from Ikerlan during the Prague hackathon, the fUML model was largely improved to mimic the Skyline behavior.

The complete executable Skyline fUML model has been shared with ABO for MATERA2 to experimentation to generate test cases for Skyline activity diagrams.

Other use cases for Moka Model Simulation, identified from Deliverable [D2.2](#), where these Moka technology can be applied are: CSY_04, CSY_05, IKER_04, NOK_08.

Collaboration between ARM, CSY and UAQ

The objective of this collaboration is to provide CSY with the means for relating the logs generated by their production system to its design and specifications, in order to better understand how the system behaves in real environments and therefore improve its design.

The use case provided by CSY is a system controls Platform Screen Doors in urban railway stations, namely the Coppilot system. The system has a safety fallback position, to ensure passenger safety in non-nominal situations. Increasing the availability of the system requires a careful analysis of the events sequences leading to the fallback position.

We focused on detecting one specific cause of fallback situations: when two sensors report inconsistent values. However, a fallback may also be caused by faulty sensors or other conditions. CSY is interested in detecting root causes behind different types of fallbacks. To detect these other causes in our current solution, we defined traceability links between logs models, UML models, and the B specification models (CSY uses the B formal method to implement the core of the Coppilot system). Such traceability links are generated by JTL using the relations defined among them. In turn, traceability links are given in input to EMF Views that is able to present the user with a graphical interface clearly visualizing the connection between log events and design elements.

Collaboration between BT and SSF

RCRS is a tool for checking compatibility, refinement, liveness and safety properties of reactive systems. The toolset comes with a Translator of Hierarchical Block Diagrams (Simulink) into the RCRS framework. We collaborated with BT and we used RCRS toolset to analyze some Simulink models used by BT. Using RCRS we could verify that the Simulink models have or in some cases does not have some desired properties.

4.2.2. Opportunities for unused tools

Still some tools has not been adopted by UC demonstrators. However this section aims to demonstrate the relevance of these tools to complete and extend the MegaM@Rt framework providing additional choices to the end users and advanced capabilities to address complex designs. The parallel deliverable D2.6 adds details on the methodological approach supported by these tools and provides concrete examples on how they may be instantiated on relevant UCs.

VeriATL

VeriATL is a tool for allowing the verification of several kinds of functional requirements (e.g. syntactic/semantic correctness) over existing model-to-model transformations defined using the Eclipse ATL model transformation language. Unfortunately, at the end of the project, the tool has not been used in practice in the context of any use case. This is mostly due to the fact that a really few use case scenarios implied the specification and deployment of model-to-model transformations written in the ATL language, and that none of them actually required to formally verify these ATL transformations. However, we still believe VeriATL proposes interesting formal verification capabilities to be added to the MegaM@Rt2 toolbox: it could be potentially used in the future (by partners both inside and outside the MegaM@Rt2 consortium) in any scenario and/or technical solution where ATL transformations are involved and need to be verified.

XPM

During the last months, XPM has been used in TRT use case in order to automate the translation between the Time4Sys model and the Xamber model. Instead of implementing the automatic translation directly from time4Sys to Xamber, XPM plays the role of this task as shown in [Figure 4.1](#).

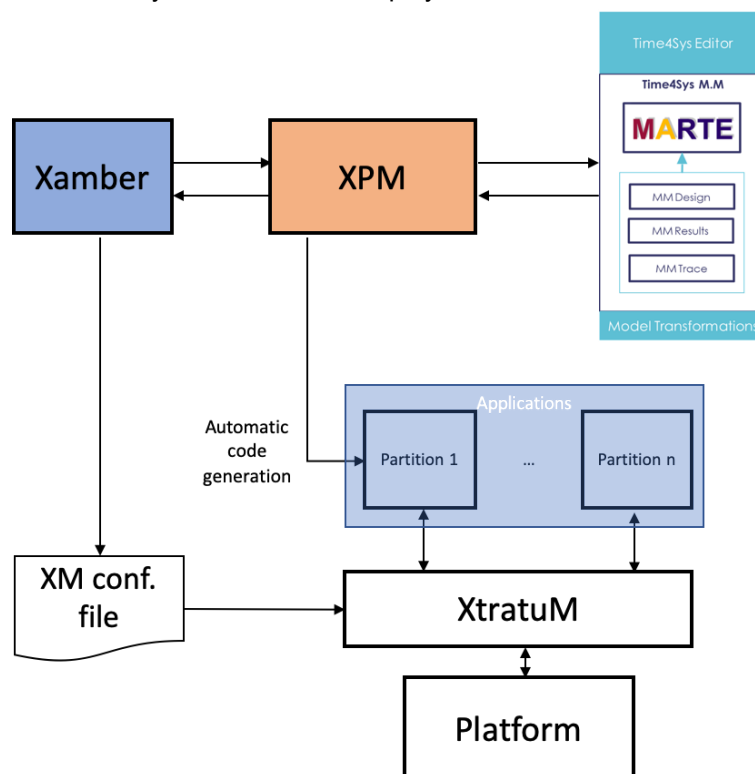


Figure 4.1 Tyme4sys to Xamber translation

Hepsycode

Hepsycode is an electronic system level (ESL) HW/SW co-design tool for heterogeneous parallel dedicated systems design. It offers an integrated framework able to drive designers from input specification to final HW/SW deployment and implementation, using model-based artifacts, SystemC simulation activities, and evolutionary algorithms for system partitioning, mapping and architectural definition. Hepsycode has been developed using Eclipse EMF technologies (Sirius/Accelero/Xtext), and a functional/timing SystemC custom simulator able to extract and predict application timing behavior (and check F/NF input requirements and constraints). The whole plugins have been included into the MegaM@Rt2 toolbox and catalog, so it could be potentially used in industries and academic domains. Section §4.3.1 [Hepsycode - Xamber Integration](#) and §4.3.2 [Hepsycode - S3D Integration](#) describe two possible Hepsycode use inside the MegaM@Rt2 consortium. The joined activities involved FTS and UCAN tools provider MegaM@Rt2 partners. Possible TRT modeling activities in

Hepsycode methodology and framework context, have been started in order to compare, integrate, and offers other parallel/alternative/different analysis that could be of interest to improve the TRT use case using another design and/or V&V activity flow. The work has been made as an initial step to show the potential use of Hepsycode tool inside the MegaM@Rt2 consortium (and also in other European projects, as described in Section §4.5 [Collaboration with external projects](#)).

CHESS

CHESS is a composite tool for real time and embedded system design and development. It is able to support several functional and non-functional properties analysis and verification. Among the number of its valuable functionalities, some are specifically applicable to some of the UCs:

in case of VCE and BT scenarios related with variability modelling, INT is exploiting the results of the activities provided in the context of the AMASS EU project to support product lines and processes variability by means of the integration with the Base Variability Resolution (BVR) built on the Common Variability Language (CVL) technology.

A concrete example is in development phase on the VCE UC based on Santander meeting where CHESS capabilities has been presented.

in case of BT requirement to support the SIL certification, CHESS provide the “correct by construction” and “safety by contract” approaches to support the certification process with a coherent design methodology. In addition it will exploit the results of the SafeCOP EU project to support the contract runtime monitor able to verify the system compliance to the given safety requirements during the application execution. Despite the fact that the current implementation of the contract runtime monitor provides limited capabilities strictly related to the applied SafeCOP UC, effort is in progress trying to develop a concrete example for BT UC before the end of the project.

in the case of TRT UC scenarios, 2 concrete examples has been developed exploiting and demonstrating the CHESS capability. In particular, it provides the ARINC extension profile to model the concepts and the timing properties of partitions, processes and functions, as defined in ARINC standards, and to support the timing and schedulability analysis specific for space and avionic domains. Activity is in progress by INT in cooperation with UCAN and UAQ to exploit interoperability between S3D, CHESS and HepsyCode tools for additional support of the TRT UC, see for details related subsection in §4.2.3 and in particular [S3D - CHESS evaluation](#).

Collaboro

Collaboro is an approach to make language development processes more participative, meaning that both developers and users of the language can collaborate together to create and evolve it. Collaboro supports both the collaborative definition of the abstract (i.e., metamodel) and concrete (i.e., notation) syntaxes for your DSL by providing a collaborative environment enabling the discussion. With Collaboro, anyone has the chance to request changes, propose solutions and give an opinion (and vote) about those from others. Since the different partners involved in the project have opted for the usage of standard languages and specifications, it was not necessary to use Collaboro for the definition of new DLs. However, the initial consideration for using Collaboro in the [Collaboration between UOC and IKER](#) has led the development of the AsyncAPI toolkit reported in [IKER2 Challenge: AsyncAPI: Single Source of Truth for Asynchronous Applications](#).

EMFtoCSP

EMFtoCSP is a model verification tool that follows a bounded verification strategy to provide a pragmatic approach to assess the quality of EMF models and UML class diagrams. OCL is also supported. The tool works by transforming the question of whether a given input model satisfies a particular correctness property into a Constraint Satisfaction Problem which is then fed into an underlying CSP solver. If the CSP solver finds a solution, then a valid instance of the model is provided as a proof.

Although the framework benefits from the generic verification capabilities of the tool, no use case has been identified where the verification of whether a UML model can be instantiated is an important issue.

4.2.3. The internal hackathon and tools fair initiative [INT]

The hackathon initiative has been productively exploited also during the final period, promoting two events during the Prague and Santander plenary meetings. The hackathons central themes addressed during these meetings was, respectively, on “traceability” (related to WP4) and “integration” (related to WP5), thus not explicitly related to WP2 area. However, these initiatives represented a significant moment of positive synergies between UC and Tool/Method providers allowing discussion on technical items other than the predefined ones. Relevant results for WP2 tools, like prototypes, proof-of-concepts as well as partial demonstrators’ implementations are presented in the [Appendix](#). In addition, during the Prague meeting, a tools fair initiative has been promoted in order to extend the visibility of the tools capabilities and tools applicability to UC context. The [Appendix](#) collects the set of WP2 tools posters prepared for this event.

4.3. Interaction between tools providers

In the context of the System Engineering Tool Set there are in progress three cooperation initiative between tool providers (FTS-UAQ, INT-UCAN, and UCAN-UAQ). These initiatives started during the initial and intermediate development period and are continuing. Heredown details on the results reached until now are presented. More details will be reported in WP5, especially in D5.3 deliverable, where a more general interoperability overview that try to catch some methodological and semantical integration pattern between tools providers and UC providers will be presented.

4.3.1. Hepsycode - Xamber Integration

As presented in D2.4 - Section 4.4.1, a collaboration between UAQ and FTS has been started to integrate Hepsycode and Xamber tools. The two tools are fully integrated, with an automatic project model-to-model transformation, as described in previous deliverables.

This paragraph describes the integration activity between the two tools providers. As shown in [Figure 4.2](#), the two tools instantiate each phase of the MegaM@Rt2 V-Model (an improvement of traditional V-Model) in different manner.

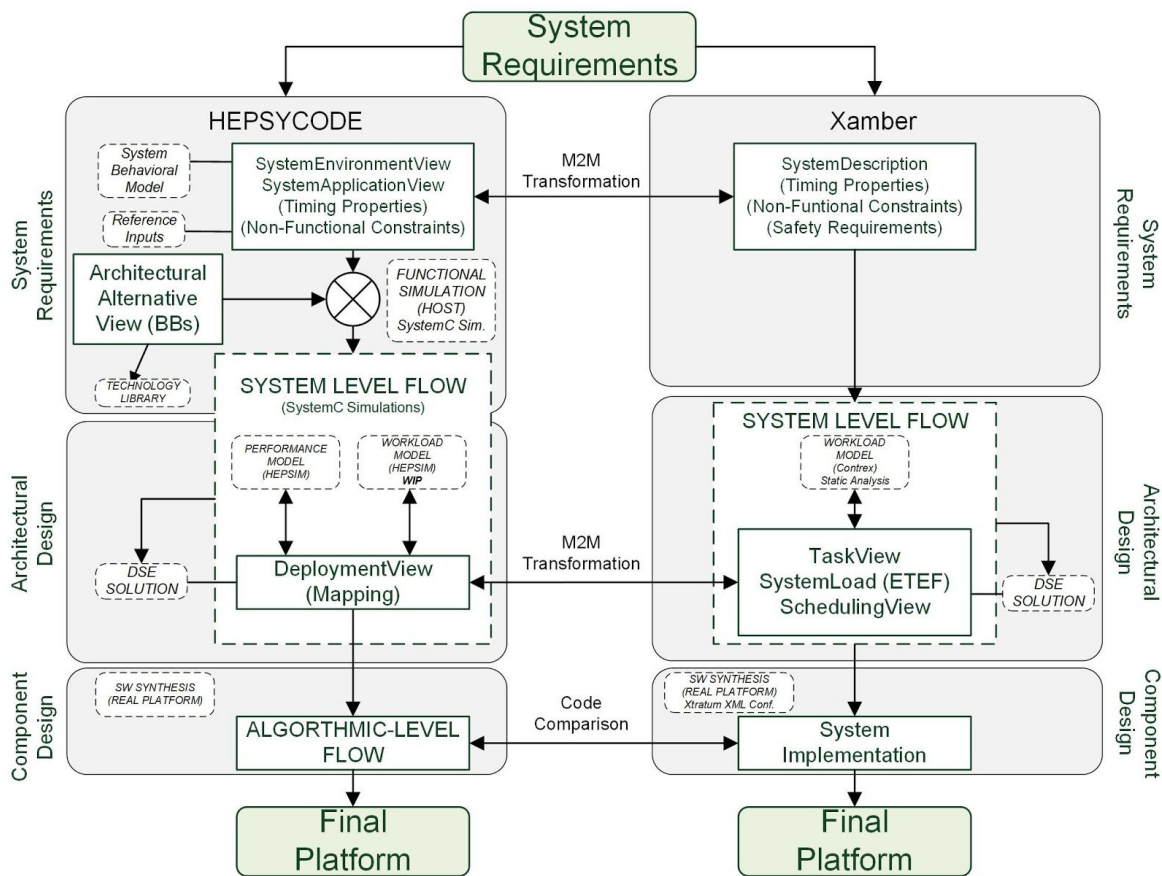


Figure 4.2: Hepsycode – Xamber Transformation Pattern (integration activity flow)

Hepsycode is a model-based framework that try to solve the classical problem of HW/SW co-design of embedded parallel systems. The methodology started from different possible architectural alternatives and apply a system-level flow using SystemC simulations and design space exploration (DSE) activities. Xamber started from a system description model, where all the HW and SW components are defined and instantiated in the main project view. Then the tool allows to check Timing and Non-Functional constraints using static schedulability analysis tools (called Contrex). Finally, Hepsycode entered inside the algorithmic-level flow, where the deployment result has been translated into a HW/SW implementation (using automatic code generator tools or manually designer programming activities), with focus on C/C++ code and VHDL HW description components. But instead, Xamber focus his implementation activities to Xtratum Hypervisor, using XPM tools (also included inside the Megamart2 catalog).

Starting from this integration pattern, Hepsycode can be used to apply performance analysis and DSE functionalities in a parallel view to Xamber tool. As an example, [Figure 4.3](#) shows a possible Hepsycode model of TRT FMS use case. This process view matches one by one to the Xamber TRT FMS use case model (as described in previous deliverables).

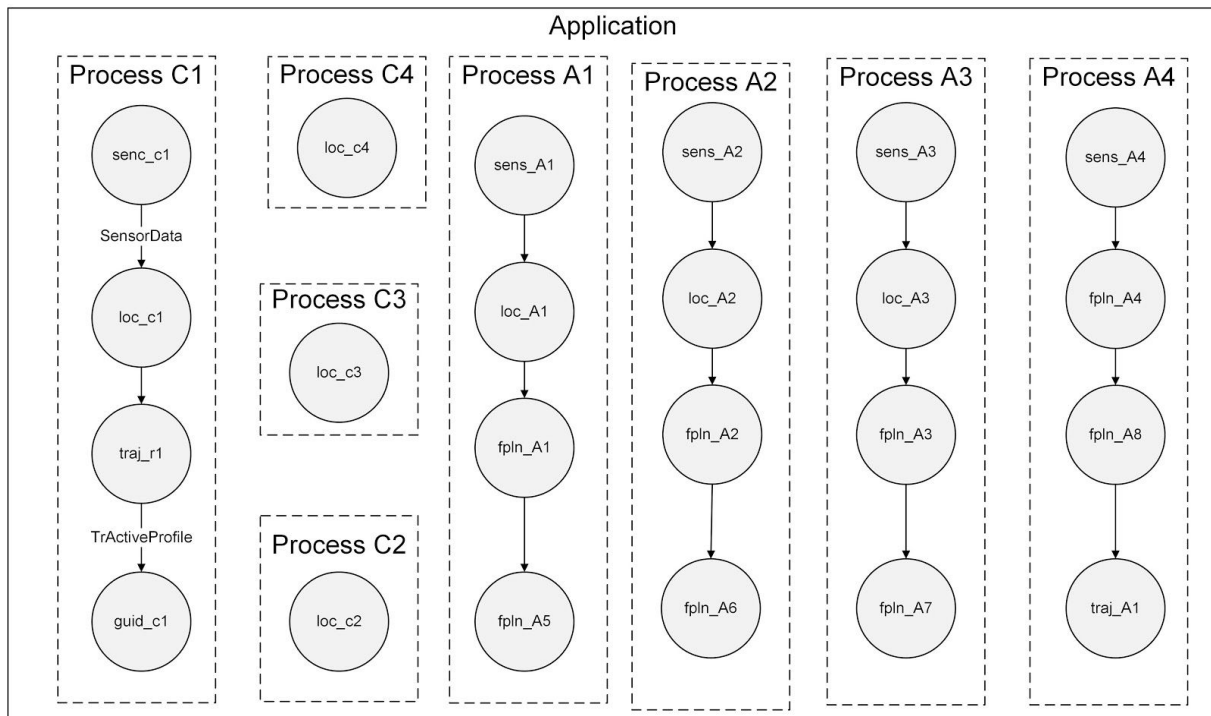


Figure 4.3: TRT FMS use case Hepsycode model w.r.t. Time4Sys and Xamber models

The model in [Figure 4.3](#) describes the FMS application as a set of concurrent processes, where each task inside the processes are connected via CSP channels. Communication between processes will be available using shared variables inside the whole main SystemC module. Starting from this preliminary FMS Hepsycode model, taken from Xamber model (compliant with Time4Sys workload model), it is possible to realize all the design flow activities offered by Hepsycode tool ad framework, and it is also possible to improve performances and check other non-functional input requirements applying all the needed activities, offering the opportunity to find other possible mapping and allocation solutions, introducing hypervisor technologies as a possible alternative for FMS implementation. Further information will be described in WP5, into D5.3 deliverable.

4.3.2. Hepsycode - S3D Integration

During the intermediate MegaM@Rt2 period, a collaboration between UAQ and UCAN has been started. The main goals of this collaboration are the identification of interoperability tool patterns, and the integration between Hepsycode and S3D tools as a reference example for this conceptual pattern. In this paragraph the main integration activities will be described, while postponing the discussion among tools cooperation and interoperability for the WP5 – D5.3.

[Figure 4.4](#) illustrates the main integration pattern between Hepsycode and S3D methodology. The two framework and tools use different Eclipse EMF technologies. S3D starts his activities from a UML/MARTE specification model. The entire projects revolve around a component-based approach. This modeling methodology introduces scalability, composability, reusability and concerns separation at system-level, allowing designer to focus on models and not to source code. The S3D approach divides the whole MegaM@Rt V-model into several activities, platform dependent and independent, using MARTE and ESSYN stereotypes, and introducing a specific model of communication that uses interfaces to exchange data among different system components. This service-oriented approach allows to realize different models of computations, defining the specific services and the provided and required ports parameters. Function Call, rendezvous, CSP, KPN, data flow, non-deterministic and timed model of computations are a subset of S3D possible computational alternatives.

Furthermore, the first integration activity done by UCAN and UAQ was related to the comparison between the two approaches in terms of MoC. Hepsycode fixes the MoC as a CSP process network, where communication between component/processes are rendezvous, synchronous and blocking. Future work will consider the possibility to introduce other MoC inside the Hepsycode methodology, while trying to do not change the entire ESL co-design flow (the use of CSP model guarantee some properties that allows to extract information about concurrency, communication and possible unsafe behavior, while introducing possible deadlock or crash system situations).

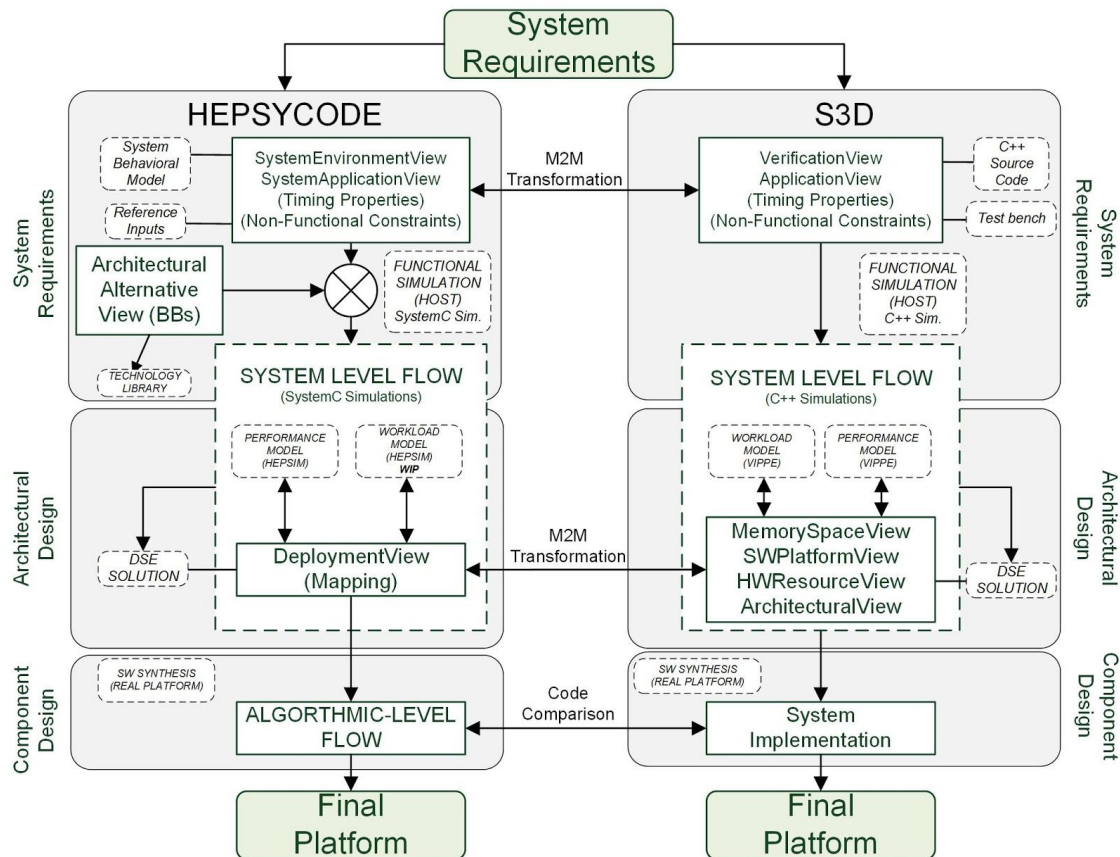


Figure 4.4: Hepsycode – S3D Transformation Pattern (integration activity flow)

The S3D activity mapped on architectural design is related to workload and performance model analysis. Using native simulation (by means of VIPPE simulator), S3D permits to analyze events, communication patterns, and schedulability issues, with a direct link between the High-Level UML/MARTE model and the functional behavioral C++ source code. A similar activity is done by Hepsycode, while the SystemC simulator (called HEPSIM) is an early-stage timing emulator that introduces higher error associated to system simulation, while trying to reduce simulation execution time by means of less accurate clock estimations. Future work will evaluate the possible integration of VIPPE emulator inside the Hepsycode co-design flow, in order to reduce errors and simulation time, while trying to refine and reduce also the SystemC errors associated to the HEPSIM simulator. Finally, the advantage of S3D design flow is the direct link between models and implementation, at component design level, while is possible to cross-compile and execute code and test directly on final target with a minimal modification effort. This is due to the use of C++ code and S3D library released during the MegaM@Rt2 project. Meanwhile Hepsycode involves, at this release time, some manual transformation between SystemC code and final implementation, but it is possible in future to direct link higher abstraction source code with low-level implementation using some external tools. Considering Hepsycode example use case described in D2.6, a first activity was oriented to model

CSP example application inside S3D. This activity has been completed in a short time, while the two methodology sharing the main semantical aspect related to process/components and Model of Computation, as shown in [Figure 4.5](#). Some preliminary results will be presented in future scientific publications.

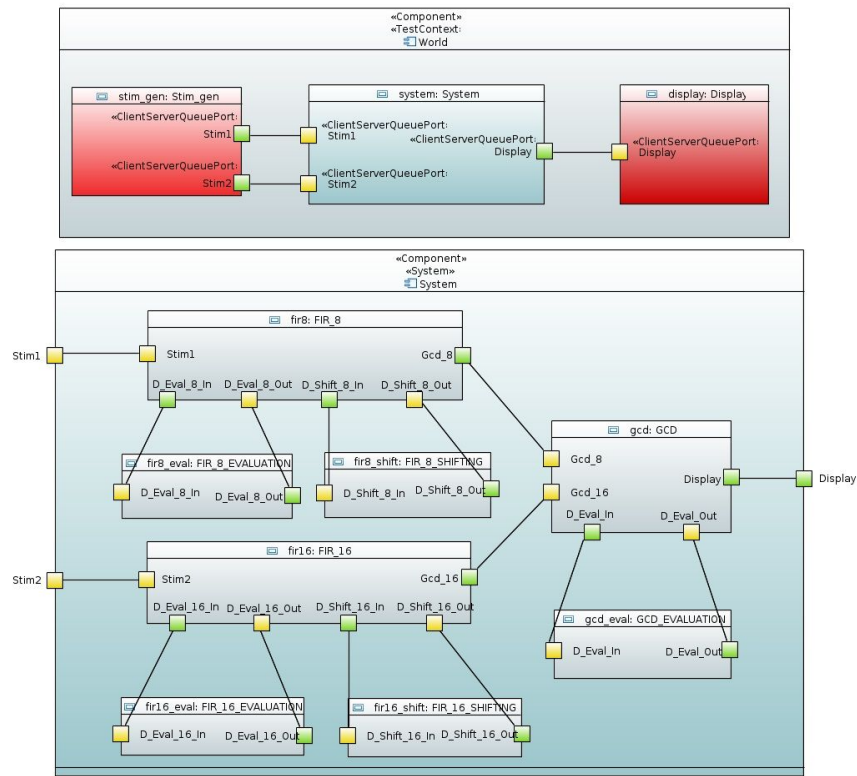


Figure 4.5: Fir-Fir-GCD S3D model (RTService stereotype has been configured to work as CSP)

Finally, respect to TRT use case, the integration activity considered during this collaboration was related to the possibility to model FMS use case inside Hepsycode and SystemC environment, so it could be possible to analyze system behaviors and propose system improvements, while offering an alternative performance and workload analysis approach and introducing possible allocation and mapping refinement, also considering multi-core scenarios. A first FMS model inside Hepsycode is shown [Figure 4.6](#), where each S3D component is mapped on a specific CSP process, and the whole system is synchronous and deterministic. The introduction of Alternation (i.e., barrier guards in the CSP formal model) guarantees the possibility to serve several processes at the same time, introducing the possibility to implement a buffer process that can permits to represent some complex scenarios (also considering the possibility to change the model of computation considering KPN or dataflow MoC). Future work will analyze all the possible modeling alternatives that this collaboration can offer in terms of analysis, simulations and model refinements.

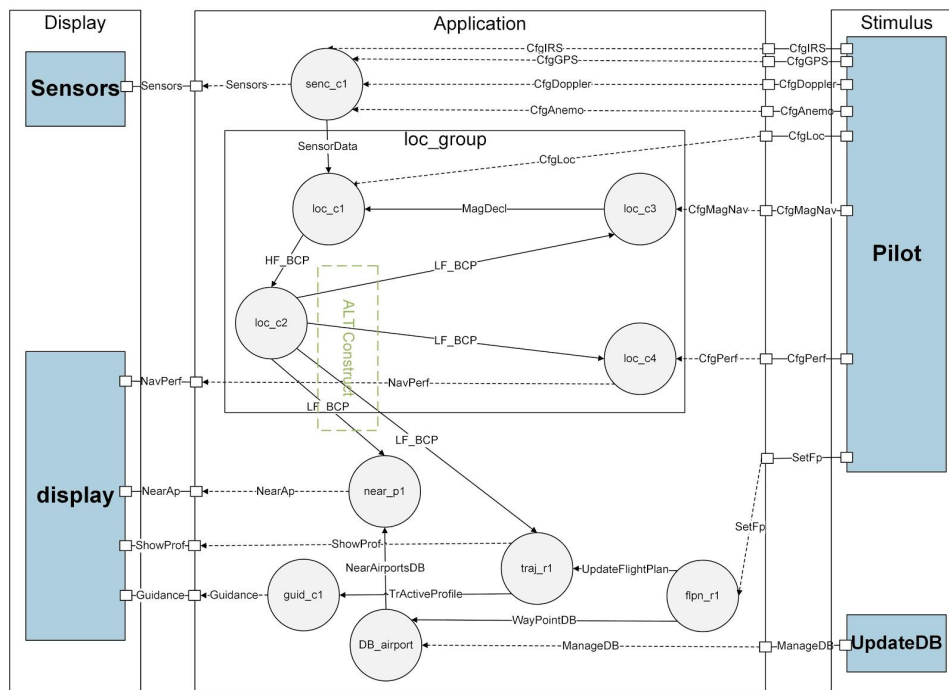


Figure 4.6: TRT FMS use case model inside Hepsycode w.r.t. S3D and Capella models

4.3.3. S3D - CHES evaluation

UCAN and INT started a collaboration oriented to find interoperability patterns between S3D and CHES methodologies. Both the approaches defined a multi-concern component-based methodology and toolset with modeling language derived from OMG modeling languages and a graphical modelling environment, that fits multiple industrial domains, based on Eclipse technologies. The tools allow to specify functional and extra-functional properties at component level and offer tools for the verification of component properties.

S3D and CHES considers models as the central development artifacts, and offer two tools that assist designers using an automated development environment upon UML/MARTE and Papyrus plugins. CHES introduced specific profiles and integrates other tools, like OCRA and OpenCert for contract-based development method, specialized to capture the non-functional properties of components, while S3D considers the direct link between components and their implementations, focusing on simulation in contrast to formal verification. Furthermore, the former and the latter are able to model real-time requirements and dependability issues, where applying a separation of concerns between application and architectural description of the system.

[Figure 4.7](#) illustrates CHES and S3D interoperability conceptual pattern, respect to the mapping between methodology layer and MegaM@Rt2 design flow. As shown in this figure, the two tools started from a similar view of the system, while extra-functional properties are asserted and verified at design time and preserved/guaranteed at run time, using also monitoring and tracing activities to check input constraints.

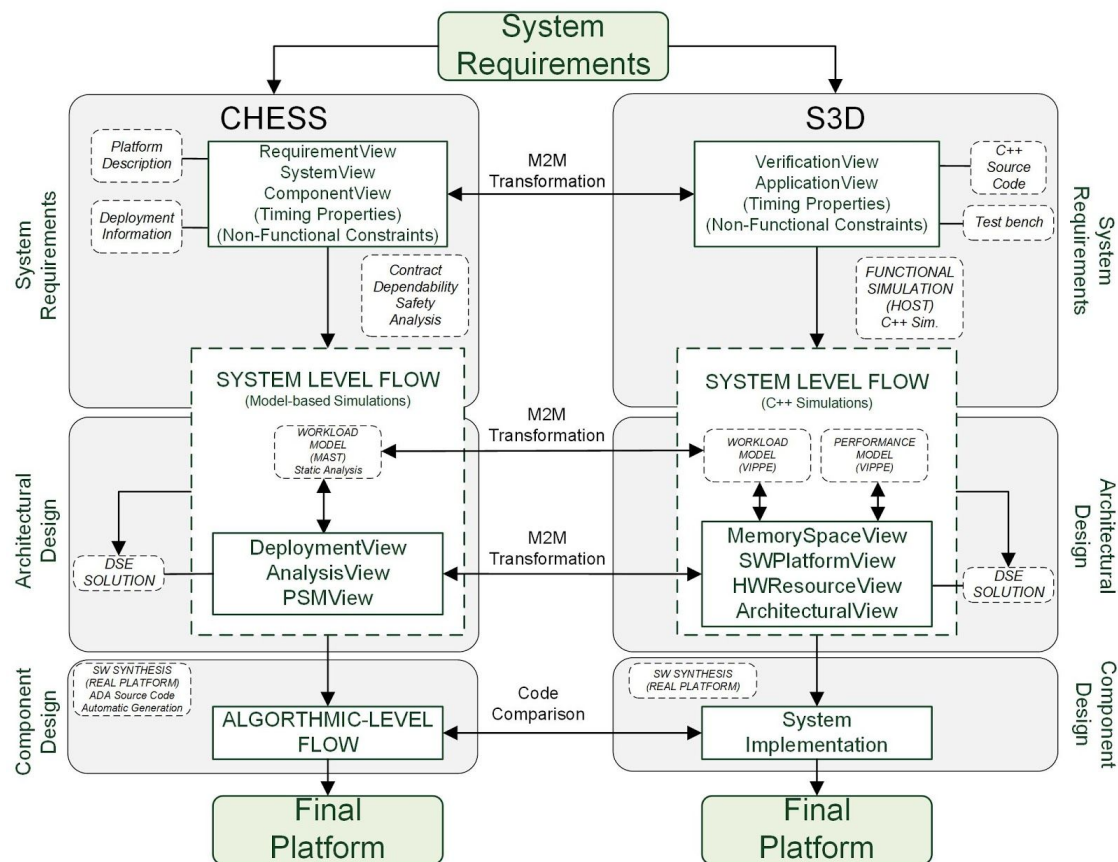


Figure 4.7: CHES – S3D Transformation Pattern (integration activity flow)

Both the approaches use several simulation approaches (model-based or source-code centered), while it is possible to use external schedulability tools (MAST) to check real-time static timing behaviors. The main differences between the two approaches are: (1) the different Model of Computation considered (ADA/Ravenscar profile based on real-time models for CHES, possibility to define several MoC, e.g., KPN, Dataflow, CSP, time triggered dataflow for S3D); (2) the use of different simulation activities, where CHES use model-based simulation to verify and validate models and contracts, with static schedulability analysis, and the possibility to check non-functional requirements dictated by standards (i.e., ARINC 653), and S3D use VIPPE, a cycle accurate C++ simulator, and offers performance and workload results using accurate model transformation to be compliant with VIPPE representations (with the possible use of schedulability analysis like MAST as a static analyzer); (3) automatic code generation (i.e., ADA source code) for CHES, code oriented approaches, where components are linked to their direct C++ implementation, in S3D). At this time S3D do not offers automatic code generation, while this functionality will be released in future works.

REVAMP2² is an Itea3 project focused on the Software Product Lines Engineering methods and tools. As a partner of this project, SOFT brought its knowledge in Variability Exchange Language and system level variability to be applied in the VCE case study. The work has been reflected in the scientific paper [\[Bilic2019\]](#).

² <http://www.revamp2-project.eu/>

5. Conclusions

The present deliverable D2.5 reports the development status of the System Engineering Tool Set component (SYS) of the MegaM@Rt framework at the “final” milestone at M32. The status is presented in two different fashion (tools roadmap and framework roadmap), conforming to the template used to report the development status at previous milestones.

The “tools roadmap”, which provides a detailed picture of the development activities per each tools, shows that:

- **the purposes’ delivery plan was fundamentally respected** during the whole project development until this final release;
- **slight delays** in some purpose development, occurred in previous milestones, **were managed to avoid impacts on UC providers activities**;
- **all the cancelled purposes was carefully evaluated** guaranteeing:
 - **no impact on the project plan**,
 - **no impact on project objectives**, indeed the MegaM@Rt framework provides a complete set of functionalities that cover the user needs and the expected technical advancements beyond the current state of the art;
 - **no impact on demonstrators developments** and framework evaluation activities,
 - **the cancellation of purposes is justified** either by the absence of significant UC requirements, by a revisited framework architecture or by negative research results.
 - **the effort** not spent on the cancelled features **has been redirected on other relevant and more important capabilities**.

The “framework roadmap”, providing a global view of the MegaM@Rt framework evolution, demonstrates that **the weakness identified during the gap analysis was solved** completing the mitigation actions that were identified, mainly by exploiting specific results from external EU project participated by some of the MegaM@Rt consortium partners.

In addition to the project status report, this deliverable presents a more general view of the whole set of WP2 tasks’ activities and significant event like: hackathon initiatives, the follow up activities in use case experiments, and the cooperation between MegaM@Rt partner to better exploit the commonalities between the tools. Indeed, these initiatives demonstrate to be strategic to focus our job on real users needs and on concrete results.

To summarise, **the project development plan for WP2 has been fulfilled with satisfactory results**. All the main project objectives have been reached by adapting the tools during the development progress based on the concrete users needs but providing as well advanced features that are expected to be exploited in future applications.

References

- [Technical Annex]** MegaM@Rt² “MegaModelling at Runtime - scalable model-based framework for continuous development and runtime validation of complex systems”
- [D1.1]** MegaM@Rt² “D1.1: Industry Requirements Specification
- [D1.2]** MegaM@Rt² “D1.2: Architecture specification and roadmap– initial version”
- [D1.3]** MegaM@Rt² “D1.3: Case studies scenarios definition”
- [D1.4]** MegaM@Rt² “D1.4: Architecture specification and roadmap – final version”
- [D2.1]** MegaM@Rt² “D2.1: Foundation for Model-driven Design Methods”
[\[http://hdl.handle.net/20.500.12004/1/P/MMART2/D2.1\]](http://hdl.handle.net/20.500.12004/1/P/MMART2/D2.1) [December 2017]
- [D2.2]** MegaM@Rt² “D2.2: Design Tool Set Specification”
[\[http://hdl.handle.net/20.500.12004/1/P/MMART2/D2.2\]](http://hdl.handle.net/20.500.12004/1/P/MMART2/D2.2) [February 2018]
- [D2.3]** MegaM@Rt² “Design Tool Set - Initial version”
[\[http://hdl.handle.net/20.500.12004/1/P/MMART2/D2.3\]](http://hdl.handle.net/20.500.12004/1/P/MMART2/D2.3) [June 2018]
- [D4.3]** MegaM@Rt Model Management & Traceability tool set - initial version [November 2018]
- [D3.4]** MegaM@Rt Runtime Analysis tool set - intermediate version
- [D5.1]** MegaM@Rt Integration Approach [November 2018]
- [MARTE2.0Req]** L.Rioux, J.Medina, “Toward MARTE 2.0 Needs & Orientation”, internal survey, https://drive.google.com/open?id=1LY3VOjEvBW21yqCUZ_3HGfe852QdqPLs
- [MAST]** Universidad de Cantabria. Mast: Modeling and Analysis Suite for Real-Time Applications. <http://mast.unican.es/>
- [FitOptiVis]** “From the cloud to the edge – smart IntegraTion and OPTimisation Technologies for highly efficient Image and VIdeo processing Systems”, ECSEL-2017-2 - RIA, <https://fitoptimis.eu/>
- [AMASS]** “Architecture-driven, Multi-concern and Seamless Assurance and Certification of Cyber-Physical Systems”, ECSEL-07-2015 - Design Technology, <https://www.amass-ecsel.eu/>
- [SafeCOP]** “Safe Cooperating Cyber-Physical Systems using Wireless Communication”, ECSEL-08-2015 - Cyber-physical Systems, <http://www.safecop.eu/>
- [AQUAS]** “Aggregated Quality Assurance for Systems”, ECSEL-2016-1-RIA, <https://aquas-project.eu/>
- [Gomez19]** Abel Gómez; Iker Fernandez de Larrea; Markel Iglesias-Urkiá; Beatriz Lopez-Davalillo; Aitor Urbietá; Jordi Cabot. Una Aproximación Basada en Modelos para la Definición de Arquitecturas Asíncronas. Actas de las XXIV Jornadas de Ingeniería del Software y Bases de Datos (JISBD 2019), Sistedes, 2019. <http://hdl.handle.net/11705/JISBD/2019/035>
- [Bilic2019]** Bilic, D., Brosse, E., Sadovykh, A., Truscan, D., Bruneliere, H., & Ryssel, U. An Integrated Model-based Tool Chain for Managing Variability in Complex System Design. 13th Workshop on Models and Evolution 2019 (ME2019). September 2019.

Appendix: Hackathons and Tools Fair reports

This appendix reports technical details and results of the hackathon challenges related to the System Engineering Tool Set application.

A.1 Prague tools fair

Tool Posters

- Ábo Akademi University: J.Iqbal, A.Ashraf, D.Truscan, I.Porres, “Exhaustive Simulation and Test Generation Using fUML Activity Diagrams”, MATERA2, <https://drive.google.com/open?id=1UPPkpduBpIPWN3ESPCeLmKifT3vIIftV>
- Innopolis University: A.Naumchev, M.Khazeev, “Intelligent Formalization, Verification and Validation of Traceable Requirements”, https://drive.google.com/open?id=1HaaO16o1mET8fTz_LTFsDGnyLKII5Uif
- Intecs Solutions S.p.A.: S.Mazzini, P.Pierini, S.Puri, “CHESS”, CHESS, https://drive.google.com/open?id=1_QAb3G9wMzdJyIOrRg0NR_HLcph_8a-c
- L’Aquila University: V.Muttillio, L.Pomante, M.Santic, E.Incerto, “HEPSYCODE-MC - Electronic System-Level Methodology for HW/SW Co-Design of Mixed-Criticality Embedded Systems”, HEPSYCODE, <https://drive.google.com/open?id=1ncMqDproAzDFgdlwxTNMSuiAhQbGsIVY>
- Pau University (Université de Pau et des pays de l’Adour): L.Brunschwig, É.Cariou, O.Le Goaër, “Code generation for the PauWare API from UML state machine models”, PAUWARE, https://drive.google.com/open?id=1WhJGYHFE3EDXtZC-3mbyUHMola_sZ3oy

Use Case Posters

- Ikerlan, Softeam: “Smart Warehouse Supervision System”, MODELIO, , <https://drive.google.com/open?id=1WuHBsrc5fkQGbk4KUM7EPPXv2vRxR6R>
- Nokia, Ábo Akademi University: S.Salow, P.Tuttilla, “Performance optimisation in multiprocessor products to meet temperature requirements”, MATERA, <https://drive.google.com/open?id=1KfxfoPS-VbvCUwRQriEU2lx8vztniL42>
- Thales, Cantabria University (Universidad de Cantabria), Fentiss: “Real-Time Design &Verification for the Flight Management System Architecture”, S3D, XAMBER, <https://drive.google.com/open?id=1Sj6-r6SkC266xTmn95LY1oHU0g0tuikf>

A.2 Prague Hackathon results

A.2.1. IKER2 Challenge: AsyncAPI: Single Source of Truth for Asynchronous Applications

CONTEXT

Typically, IIoT (Industrial IoT) architectures are distributed and asynchronous, with communication being event driven, such as the publication (and corresponding subscription) of messages. These asynchronous architectures enhance scalability and tolerance to change, but they have their disadvantages; for example, it is easy for knowledge about messages and their categorization (topics) to be diluted among the elements of the architecture, arising interoperability problems between the agents involved.

Interoperability problems are not exclusive to asynchronous architectures. In fact, synchronous architectures also manifest them. However, in this field, the industry has already proposed standardized solutions to support the development of synchronous messaging. An example is OpenAPI and its complete ecosystem of tools for generating code and adapters, documentation, etc. For asynchronous architectures, and inspired by OpenAPI, the AsyncAPI³ proposal has appeared recently. In AsyncAPI, the specifications of an API can be defined in YAML or JSON, which allows to specify, for example, the message brokers, the topics of interest, or the different message formats associated to each one of the topics, among other aspects. AsyncAPI is, however, in early stages of development, and therefore its tools are underdeveloped, mainly limited to the generation of documentation to be consumed by humans.

GOALS

- Automate the design and implementation of asynchronous architectures using model-based techniques
 - Obtain a metamodel for AsyncAPI, in such a way that a specification of an asynchronous API can be automatically and directly represented as a model according to the metamodel of AsyncAPI
 - Develop a grammar that validates definitions of asynchronous APIs conforming to the specification of AsyncAPI
- Exploit the entire set of tools available in the Eclipse Modeling Framework (EMF) ecosystem to generate the code that will be deployed in each of the participating agents
 - Given the availability of both a metamodel for AsyncAPI and the specification of an API as a conforming model
- Generate a series of internal Domain Specific Languages (DSL) to assist developers in the creation of well-trained messages.

PARTICIPANTS

- Proposer: Ikerlan (IKER)
- Implementer: Universitat Oberta de Catalunya (UOC)

RESULTS

³ <https://www.asyncapi.com/>

As a result of the hackaton, the proposal in [Figure A2.1](#) was adopted. First, based on the AsyncAPI specification, an Xtext⁴ grammar has been designed. From this grammar, an Ecore metamodel is automatically derived, together with a set of editors and Eclipse-based tools. These editors allow creating JSON-based specifications of message-driven APIs using AsyncAPI. Specifications created using these editors are automatically parsed and reified as instances of the AsyncAPI metamodel, which, in turn, are transformed into Java code.

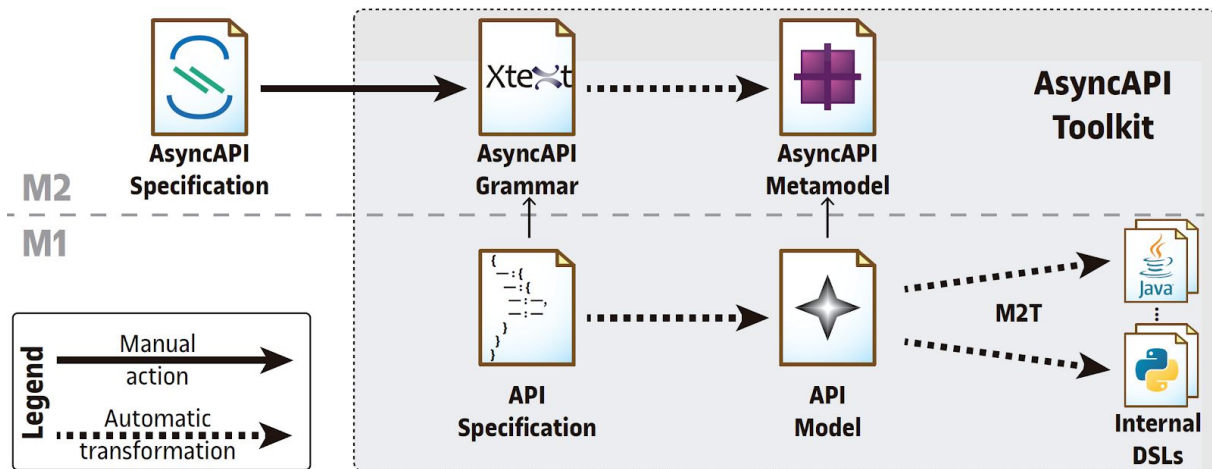


Figure A2.1: Proposal for Single Source of Truth for Asynchronous Applications

The above proposal has led to the *AsyncAPI toolkit*, for which a prototypal implementation is available at <https://github.com/SOM-Research/asyncapi-toolkit>. At the end of the hackathon, the prototype implements a first version of the AsyncAPI metamodel (version 1.2) shown in [Figure A2.2](#).

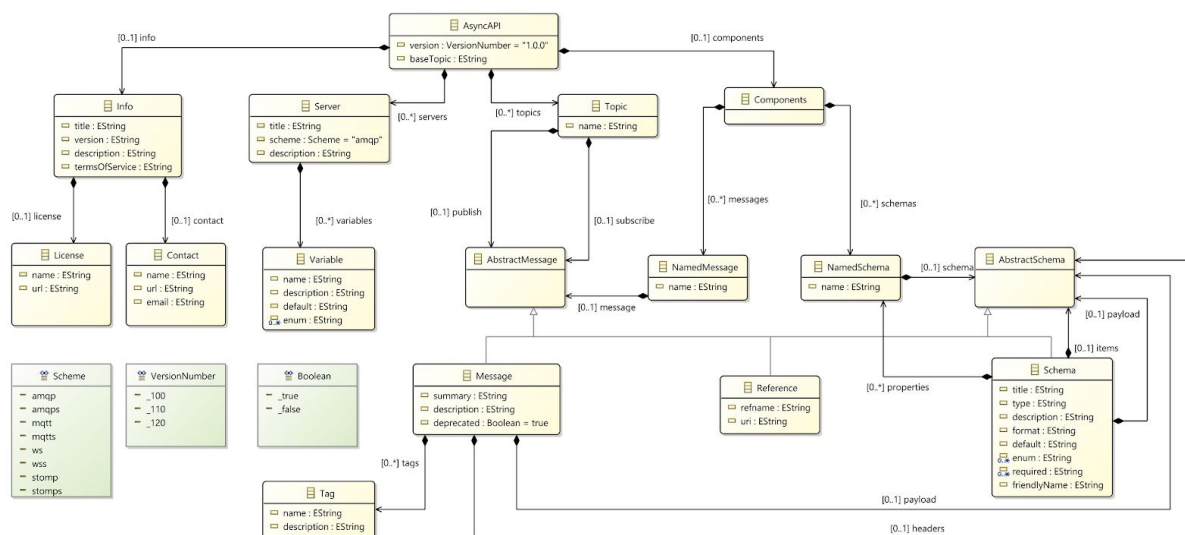


Figure A2.2: Simplified metamodel for AsyncAPI v1.2

Additionally, the prototype is able to generate Java code supporting the creation and serialization of JSON-based message payloads, including nested JSON objects. No support for arrays is provided yet at this point however. The excerpt below shows an example of an AsyncAPI specification supported by the prototype:

⁴ <https://www.eclipse.org/Xtext/>

```

{
  "asyncapi": "1.2.0",
  "info": {
    "title": "Sample AsyncAPI specification",
    "version": "0.1.0",
  },
  "servers": [
    {
      "url": "broker.url:{port}",
      "scheme": "mqtt",
      "description": "This is an example description",
      "variables": {
        "port": {
          "default": "1883",
          "enum": [ "1883", "8883" ]
        }
      }
    }
  ],
  "topics": {
    "messages/device2controller": {
      "publish": { "$ref" : "#/components/messages/request" }
    }
  }
},
"components": {
  "schemas": {
    "protocol_version": {
      "title": "Protocol version",
      "type": "integer",
      "default": 2,
      "x-friendly-name": "ProtocolVersion"
    },
    "id": {
      "title": "ID",
      "type": "string",
      "format": "XXXXXX YY ZZZZZ W"
    },
    "status": {
      "title": "Status",
      "type": "string",
      "enum": ["OK", "ERROR"],
      "x-friendly-name" : "Status"
    },
    "environment": {
      "title": "Environment",
      "type": "string",
      "enum": ["DEV", "STAG", "PROD" ],
      "x-friendly-name" : "Environment"
    }
  }
}

```



```

        .withContent("Content")
        .build()
    ).build();

    // Request.publish(payload); // Not implemented yet

    System.out.println(payload.toJson(true));
    System.out.println(Request.Payload.fromJson(payload.toJson()).toJson(true));
}
}

```

See a screenshot of what the prototype looks like in [Figure A2.3](#).

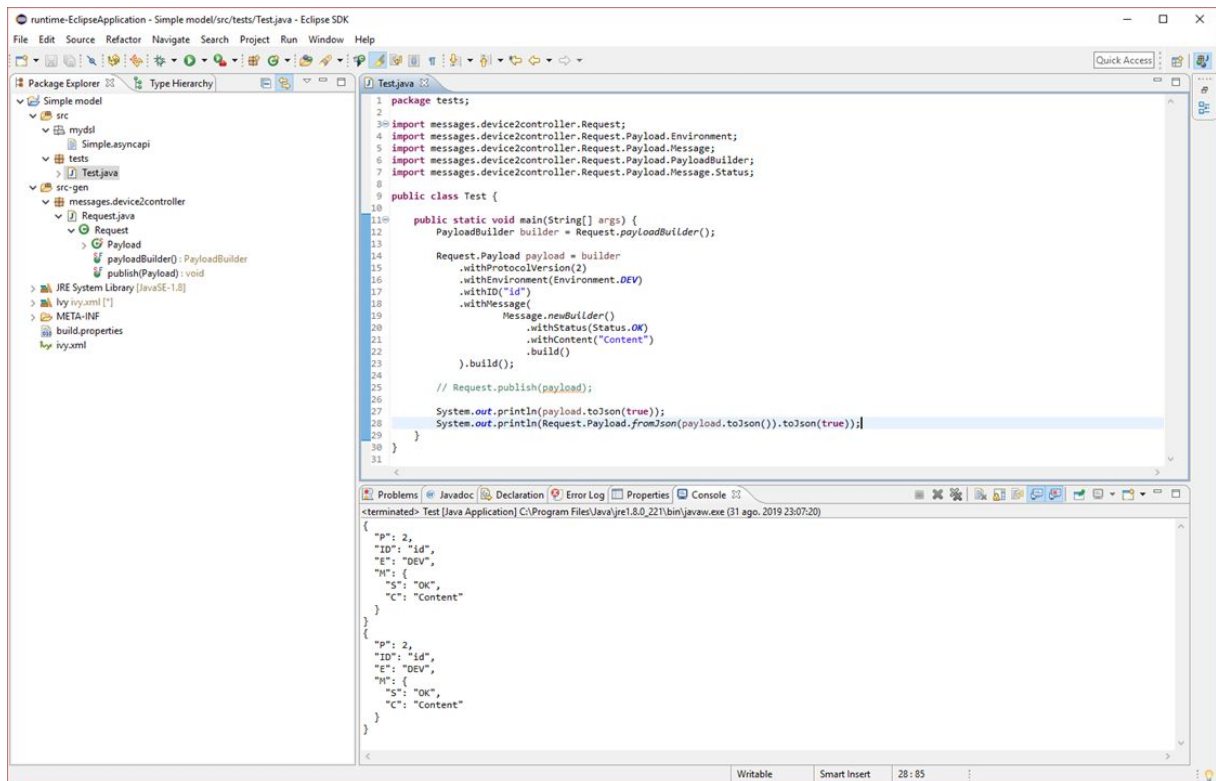


Figure A2.3: Screenshot of the AsyncAPI toolkit prototype

A.3 Santander Hackathon

A.3.1. BT Smoke detection system - Variability Modelling

CONTEXT

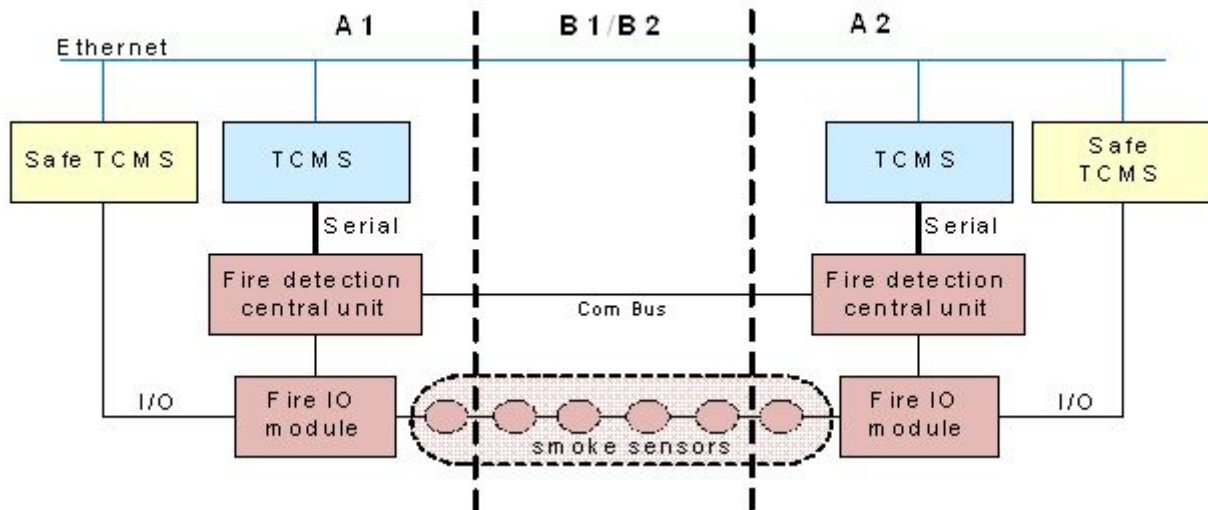


Figure A3.1: Smoke detection system onboard a metro train

Responses in Subsystems

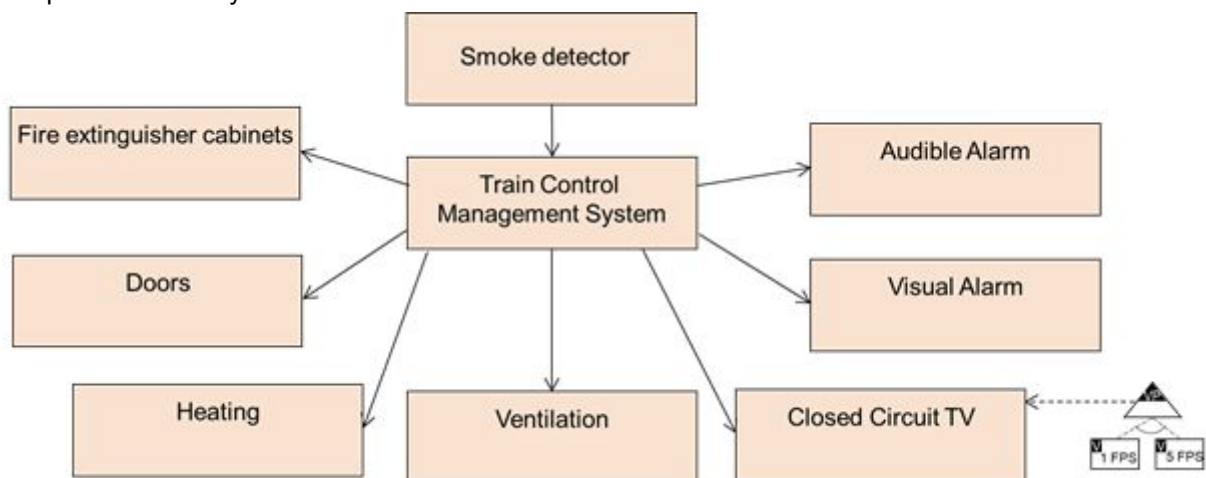


Figure A3.2: Train Control Management System

Main scenario

When a smoke detector senses smoke in a metro train passenger area, a sequence of responses are triggered:

1. Audible alarm sounds in driver's cab.
2. Visual alarm indication is presented on the driver's display.
3. The position of the affected smoke detector is presented on the driver's display.

4. Live CCTV images from the camera covering the smoke detector's area are shown on the driver's display. (VP)
5. Heating Ventilation Air Conditioning (HVAC) system in the affected car is turned off.
6. The fire doors at both ends of the affected car are closed.
7. All fire extinguisher cabinets in the affected train-set are unlocked.
8. High voltage power supply to the heating in the HVAC system is turned off in the affected train-set.

The audible alarm (1.) is silenced when the driver presses button "Acknowledge alarm" on the driver's display.

Reset of all other smoke alarm responses (2. – 8.) is achieved when the driver presses button "Acknowledge alarm" on the driver's display at a point in time when no smoke detector indicates smoke. *CPT-0031 (v.7)*

High level requirements

ID	[2F VehicleFuncts-VF_TRS.-C30-REQ-0214 (v.3)]
Description	In case of any detected fire, indications shall be presented to the driver.
Safety level	0
Safety argument	Fire hazard.

ID	[2F VehicleFuncts-VF_TRS.-C30-REQ-0400 (v.3)]
Description	In case fire is detected in passenger area the heating contactor in affected trainset shall be opened
Safety level	1
Safety argument	Fire hazard.

GOALS

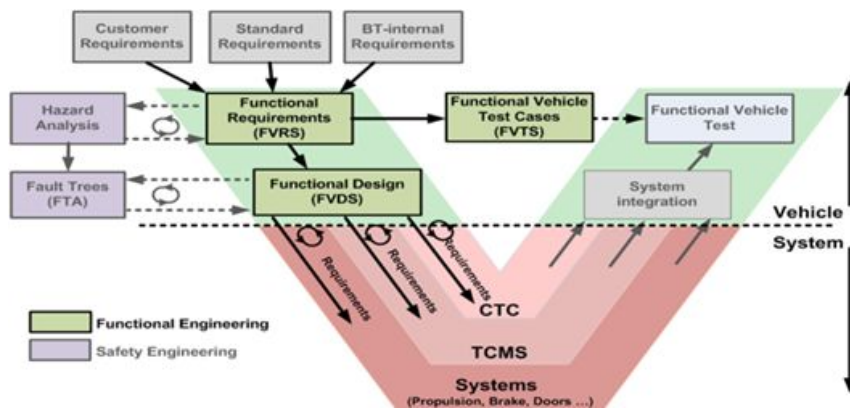
The goal of the challenge is to use and experiment several modelling tools for the purpose of designing a smoke detector sub-system by taking into account the common features and variants.

- Design the common features in a generic kernel with just a few subsystems and low complexity, and add additional variants (feature packages) on top of that.
- For example additional packages can be added with extensions to the main common scenario.
- Alternatively, one can add extensions to the main scenario or add subsystems (e.g., using a state machines and or feature trees).
- Finally, it will be interesting to add quality attributes e.g. performance/timing, reliability, availability, maintainability, safety.

Required Inputs

One basic main scenario as a document containing:

- HiLevel reqs
- functional concept
- low level requirements
- test specs.

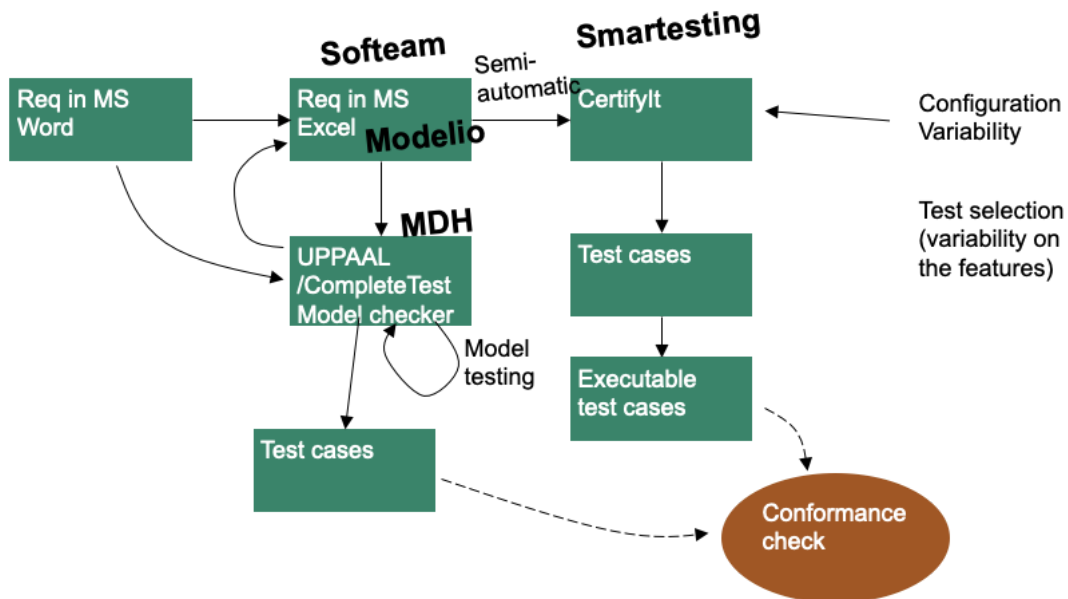


PARTICIPANTS

- Bombardier Train (proposer)
- Softeam
- RISE SICS
- Smartesting
- Malardalen University

Process and Integration of MM2 Techniques between Modelio, Smartesting, UPPAAL/CompleteTest

Global view



Step 1: From Word to SysML Requirement (Modelio)

ID	[2F-VehicleFuncs-VF.TRS.-C30-REQ-0198(v.2)]
Description	The HVAC system shall be shut down in the car where smoke is detected.
Safety level	0
Safety argument	Fire hazards, however power supply to heaters is closed down directly by TCMS.

↑

ID	[2F-VehicleFuncs-VF.TRS.-C30-REQ-0199(v.2)]
Description	The driver shall have the possibility to close/open the fresh air dampers.
Safety level	0
Safety argument	Comfort function, as smoke density can't be more heavy than acceptable for operation in the tunnel.

↑

ID	[2F-VehicleFuncs-VF.TRS.-C30-REQ-0200(v.2)]
Description	OCC shall have the possibility to close/open the fresh air dampers.
Safety level	0
Safety argument	Comfort function, as smoke density can't be more heavy than acceptable for operation in the tunnel.

5.1.2 -> Concept

Smoke-detected-in-the-train

Both HVAC in the cab and the HVAC in passenger area will behave the same way in case of smoke.

To avoid smoke to migrate to other cars TCMS will order the HVAC, in the car where smoke is detected, to turn off (*off mode*) so the air circulation is stopped. Internal doors between cars will be closed by TCMS as well, but that is not part of this concept. (ref[4])

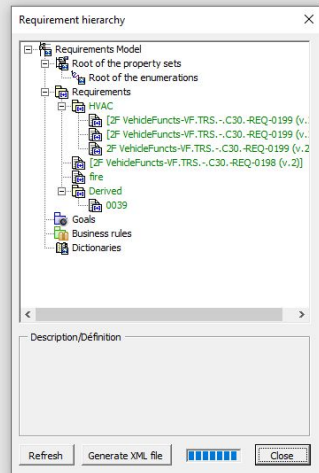
If the train stands at platform (ref[2]) when the fire alarm activated (not reset fire alarm), the HVACs in all cars of the affected train set will be turned off (*off mode*).

↑

The cab is considered to be a separate part of the car. Smoke detected in the passenger area will not affect the HVAC in the cab. If smoke is detected within the cab the cab HVAC will go to *off mode*. -CPT-0018 (v.2)

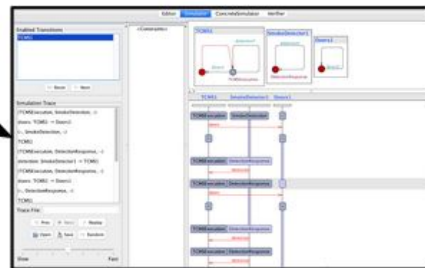
5.1.3 -> Derived Requirements

Req ID	Description	Derived to	SIL
0039 (v.1)	If fire (smoke) is detected in the passenger area, when the train is between platforms, the HVAC in the affected car shall be put in <i>off mode</i> .	4S_09.03.05.-TCMS Software	0
0040 (v.1)	If fire (smoke) is detected in the passenger area, when the train is standing at a platform, the HVACs in all cars of the affected train set shall be put in <i>off mode</i> .	4S_09.03.05.-TCMS Software	0



Step 2: From Requirements to Model Checking and Model Testing

Req ID	Description	Derived to	SIL
0039 (v.1)	If fire (smoke) is detected in the passenger area, when the train is between platforms, the HVAC in the affected car shall be put in <i>off mode</i> .	4S_09.03.05.-TCMS Software	0
0040 (v.1)	If fire (smoke) is detected in the passenger area, when the train is standing at a platform, the HVACs in all cars of the affected train set shall be put in <i>off mode</i> .	4S_09.03.05.-TCMS Software	0
0155 (v.1)	If fire (smoke) is detected in a car, fire shall be signaled to the local HVAC.	4S_09.03.05.-TCMS Software	0

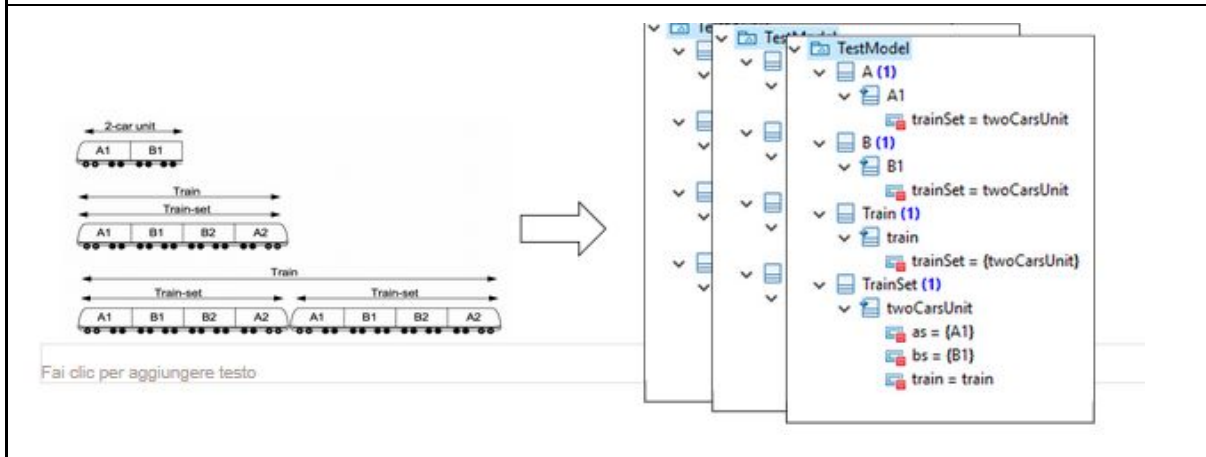
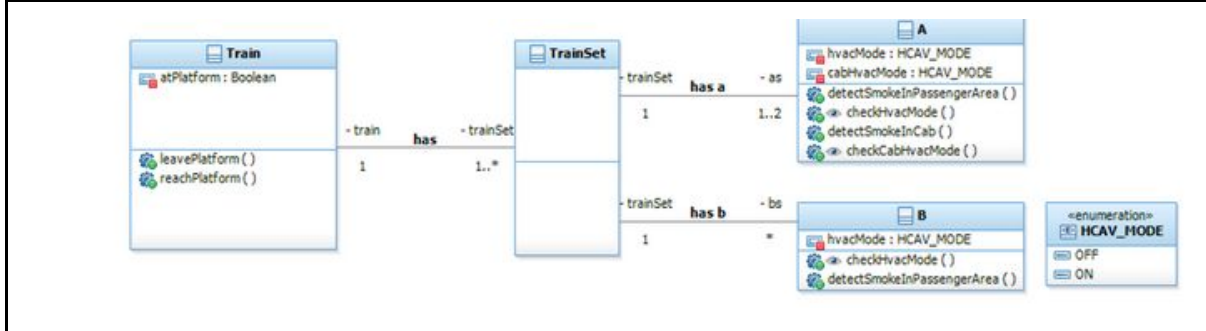
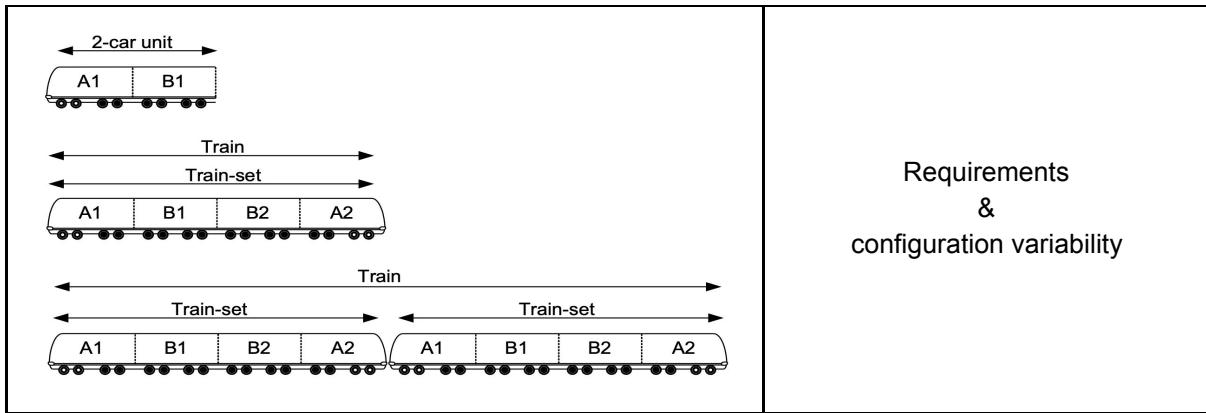


E<> TCMS1.TCMSExecution and SmokeDetector1.DetectionResponse



Step 3: From requirements to test generation with CertifyIt and test configuration variability

Req ID	Description	Derived to	SIL
0039 (v.1)	If fire (smoke) is detected in the passenger area, when the train is between platforms, the HVAC in the affected car shall be put in <i>off mode</i> .	4S_09.03.05.-TCMS Software	0
0040 (v.1)	If fire (smoke) is detected in the passenger area, when the train is standing at a platform, the HVACs in all cars of the affected train set shall be put in <i>off mode</i> .	4S_09.03.05.-TCMS Software	0
0155 (v.1)	If fire (smoke) is detected in a car, fire shall be signaled to the local HVAC.	4S_09.03.05.-TCMS Software	0



Req ID	Description	Derived to	SIL
0039 (v.1)	If fire (smoke) is detected in the passenger area, when the train is between platforms, the HVAC in the affected car shall be put in <i>off mode</i> .	4S_09.03.05.-.TCMS Software	0
0040 (v.1)	If fire (smoke) is detected in the passenger area, when the train is standing at a platform, the HVACs in all cars of the affected train-set shall be put in <i>off mode</i> .	4S_09.03.05.-.TCMS Software	0
0155 (v.1)	If fire (smoke) is detected in a car, fire shall be signaled to the local HVAC.	4S_09.03.05.-.TCMS Software	0

Project 'SmokeDetectionSystem'

Tags	In model	Validation	Desc	Derived to
All tags	13% 3/23	0% 0/23		
REQ: 0018 (v.2)	+	✓	Smoke detected in the passenger area will not affect the HVAC in the cab. If smoke is detected in the passenger area, when the train is between platforms, the HVAC in the affected car:	09.03.05.-.TCMS Softw
REQ: 0039 (v.1)	+	✓	ssenger area, when the train is standing at a platform, the HVACs in all cars of the affected	09.03.05.-.TCMS Softw
REQ: 0040 (v.1)	+	✓	Train-sets of the fire detection system shall independently of each other indicate fire in tra	09.03.05.-.TCMS Softw
REQ: 0081 (v.1)	+	✗	If any central unit indicates fire start inhibit shall be set.	09.03.05.-.TCMS Softw
REQ: 0082 (v.3)	+	✗	If any central unit indicates fire start inhibit shall be set.	09.03.05.-.TCMS Softw

```

1  lif (self.trainSet.train.atPlatform) then
2    ---@REQ: 0040 (v.1)
3    self.trainSet.train.trainSet->forall(ts : TrainSet |
4      ts.as->forall(a : A |
5        a.checkCabHvacMode(a.cabHvacMode) and
6        a.hvacMode = HCAV_MODE::OFF and
7        a.checkHvacMode(a.hvacMode)
8      ) and
9      ts.bs->forall(b : B |
10       b.hvacMode = HCAV_MODE::OFF and
11       b.checkHvacMode(b.hvacMode)
12     )
13   )
14 else
15   ---@REQ: 0039 (v.1)
16   self.hvacMode = HCAV_MODE::OFF and
17   self.checkHvacMode(self.hvacMode)
18 endif

```

Smartesting CertifyIt 6.4.3 - SmokeDetectionSystem [C:\Users\yakaldir\Desktop\Bombardier\workspace\SmokeDetectionS...

Project Preferences Help

Run test generation | JUnit publisher

Stories Tests Requirements

Artifacts Tests

Project 11

- TestSuite1 6
 - detectSmokelnCab (16-fd-53)
 - detectSmokelnPassengerArea (16-01-e2)
 - detectSmokelnPassengerArea (16-8b-ab)
 - detectSmokelnPassengerArea (16-ea-76)
 - detectSmokelnPassengerArea (16-fb-ce)
 - reachPlatform (16-28-b3)
- TestSuite2 5
 - detectSmokelnCab (fb-fd-53)
 - detectSmokelnPassengerArea (fb-01-e2)
 - detectSmokelnPassengerArea (fb-79-e2)
 - detectSmokelnPassengerArea (fb-a1-a6)
 - detectSmokelnPassengerArea (fb-fb-ce)

Test detail

Steps

- Default model instance
- Initialized model instance
- B1.detectSmokelnPassengerArea**
 - A2.checkCabHvacMode(ON)
 - A2.checkHvacMode(OFF)
 - A1.checkCabHvacMode(ON)
 - A1.checkHvacMode(OFF)
 - B1.checkHvacMode(OFF)
 - B2.checkHvacMode(OFF)

Point of view

Tags of the suite reached by the test (bold for current step)

REQ: + 0040 (v.1)

Reached tags Activated tags Parameters Model instance

1: Console

```

1 package smoke.detectionSystem.testSuite2;
2
3 import junit.framework.TestCase;
4 import SmokeDetectionSystem.AdapterInterface;
5 import SmokeDetectionSystem.AdapterFactory;
6
7 import static SmokeDetectionSystem.TestSuite2.TypesDefinition.*;
8
9 /**
10  * @REQ (v.1)
11  */
12 public class DetectSmokelnPassengerArea_fb_fb_ce_extends TestCase {
13
14     private AdapterInterface adapter;
15
16     public void setUp() throws Exception {
17         adapter = AdapterFactory.getForSuite("SmokeDetectionSystem", "TestSuite2");
18     }
19
20     public void testDetectSmokelnPassengerArea_fb_fb_ce() throws Exception {
21         adapter.TestModelBdetectSmokelnPassengerArea(B.B1);
22         adapter.TestModelAcheckCabHvacMode(A.A2, HCAV_MODE.ON);
23         adapter.TestModelAcheckHvacMode(A.A2, HCAV_MODE.OFF);
24         adapter.TestModelBcheckCabHvacMode(B.B1, HCAV_MODE.ON);
25         adapter.TestModelAcheckHvacMode(A.A1, HCAV_MODE.OFF);
26         adapter.TestModelBcheckHvacMode(B.B1, HCAV_MODE.OFF);
27         adapter.TestModelBcheckHvacMode(B.B2, HCAV_MODE.OFF);
28     }
29
30     public void tearDown() throws Exception {
31         adapter.closeAdapter();
32     }
33
34 }

```