D3.3: Runtime Tool Set – Initial version

Accompanying note

**Milestone:** 30/06/2018 (M15)

**Status:** Released

**Coordinator:** SMA

**Contributors:** SMA, BUT, ABO, UPAU, SICS, SOFT, MDH, UCAN, SSF, UAQ, CON, ATOS, FTS, RO, VTT, INT, SOFT, ARM

**Reviewers:** ABO, UAQ

**Dissemination Level:** Public

**Handle:** http://hdl.handle.net/20.500.12004/1/MMART2/D3.3

**Last Edited:** 2/07/2018

MegaM@Rt²
Executive summary

The overall objective of MegaM@Rt2 WP3 is to provide Runtime Analysis tool-supported techniques or methods. These and the related developed tools aim to create and manage large-scale models at runtime in order to support efficient verification and testing, including means for traceability of requirements. To address and reach this objective, WP3 will investigate and develop relevant and efficient tool-supported solutions able to automate and generalize the conceptual runtime approaches, defined within the project, that address the project case-study challenges.

This deliverable D3.3 specifies the initial version at M15 of the MegaM@Rt2 runtime analysis tools developed within Tasks T3.2, T3.3 and T3.4 to support automated code generation and model execution, log analysis, as well as runtime verification and testing activities. Therefore, this deliverable contains the current version of the executable runtime demonstrators developed within the project. It defines the starting point of the MegaM@Rt2 Framework, whose intermediate version and final version will be delivered at M20 and M32, respectively. To reach these future achievements, the delivered initial version of the tools will be refined and extended in the rest of the project to fully support the set of MegaM@Rt2 Framework requirements. These requirements have been defined and scheduled in Deliverable D3.2 [1] to address the case study requirements and end-user KPI given in FPP and refined in Deliverables D1.1 [2] and D1.3 [3]. They also address the related tooling expectations described in Deliverable D1.2 [4].

The present report (subtitled “D3.3 Accompanying Note”) aims to complete tool components released at M15 by providing, for each tool in development, the status and current achievement regarding targeted functional objectives and roadmap such as they have been stated in the previous Deliverable D3.2 [1]. It also gives the basic information and/or references about installation instructions, guidelines to download, deploy and use the delivered tools. Therefore, this document is part of the Deliverable D3.3 of the MegaM@Rt2 project. It specifically aims to highlight the development status of each tool, achieved at M15 of the project, to release all of them and introduce information to clarify their use.
Table of Contents

Executive summary ............................................. 2
Table of Contents ............................................... 4
List of Figures .................................................. 8
Acronyms ......................................................... 9
Introduction ....................................................... 13
  Reminder of the conceptual framework and tool set architecture .... 14
  Objectives and contents of the deliverable D3.3 ................... 15
  Structure of this document .................................. 16
Progress status of the runtime analysis tool set ...................... 16
  CompleteTest ............................................... 19
    Features available at baseline ............................. 19
    Features planned for initial release (M15) ............... 19
  S3D .................................................................. 20
    Features available at baseline ......................... 20
    Features planned for initial release (M15) ............ 22
  PADRE ........................................................... 23
    Features available at baseline ......................... 23
    Features planned for initial release (M15) ............ 24
  CHESS .......................................................... 24
    Features available at baseline ......................... 24
    Features planned for initial release (M15) ............ 26
  XPM ................................................................ 26
    Features available at baseline ......................... 26
    Features planned for initial release (M15) ............ 26
  Papyrus .......................................................... 28
    Features available at baseline ......................... 28
    Features planned for initial release (M15) ............ 31
  AIPHS ............................................................ 31
    Features available at baseline ......................... 31
    Features planned for initial release (M15) ............ 32
  JTL .................................................................. 32
    Features available at baseline ......................... 32
    Features planned for initial release (M15) ............ 33
  PauWare .......................................................... 34
    Features available at baseline ......................... 34
    Features planned for initial release (M15) ............ 34
  CMA .................................................................. 35
Features available at baseline
Features planned for initial release (M15)
CertifyIt
Features available at baseline
Features planned for initial release (M15)
MBeetle
Features available at baseline
Features planned for initial release (M15)
Conformiq Designer
Features available at baseline
Features planned for initial release (M15)
Convex Hull
Features available at baseline
Features planned for initial release (M15)
Clusterability
Features available at baseline
Features planned for initial release (M15)
Moka
Features available at baseline
Features planned for initial release (M15)
MATERA2
Features available at baseline
Features planned for initial release (M15)
Modelio
Features available at baseline
Features planned for initial release (M15)
VeriATL
Features available at baseline
Features planned for initial release (M15)
LIME TestBench
Features available at baseline
Features planned for initial release (M15)
RCRS
Features available at baseline
Features planned for initial release (M15)

Runtime analysis framework status, the roadmap profile
Basic framework requirements
Analysis of execution logs requirements
Monitoring requirements
Code generation requirements
Testing requirements

Tool set resources
CompleteTest
S3D
PADRE
<table>
<thead>
<tr>
<th>Toolset</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modelio</td>
<td>89</td>
</tr>
<tr>
<td>VeriATL</td>
<td>90</td>
</tr>
<tr>
<td>LIME Testbench</td>
<td>90</td>
</tr>
<tr>
<td>RCRS Toolset</td>
<td>91</td>
</tr>
</tbody>
</table>
List of Figures

Figure 1: Runtime Analysis Tool Set Conceptual Architecture
Figure 2: Runtime Analysis Tool Set Functional Interfaces
Figure 3: Overview of the Progress Status of the Runtime Analysis Tool Set
Figure 4: Status of the Basic Framework Requirements Roadmap
Figure 5: Status of the Analysis of Execution Logs Requirements Roadmap
Figure 6: Status of the Monitoring Requirements Roadmap
Figure 7: Status of the Code Generation Requirements Roadmap
Figure 8: Status of the Testing Requirements Roadmap
Figure 9: CompleteTest GUI
### Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AADL</td>
<td>Architecture Analysis and Description Language</td>
</tr>
<tr>
<td>ADL</td>
<td>Architecture Description Language</td>
</tr>
<tr>
<td>AL</td>
<td>Architectural Language</td>
</tr>
<tr>
<td>ALEX</td>
<td>Automata Learning Experience</td>
</tr>
<tr>
<td>ALF</td>
<td>Action Language for Foundational UML</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>APL</td>
<td>Apache Public License</td>
</tr>
<tr>
<td>ARINC</td>
<td>Aeronautical Radio Incorporated</td>
</tr>
<tr>
<td>ASCET</td>
<td>Advanced Simulation/Software and Control Engineering Tool</td>
</tr>
<tr>
<td>ASIL</td>
<td>Automotive Safety Integrity Level</td>
</tr>
<tr>
<td>ASL</td>
<td>Action Specification Language</td>
</tr>
<tr>
<td>ATL</td>
<td>ATLAS Transformation Language</td>
</tr>
<tr>
<td>AUTOSAR</td>
<td>AUTomotive Open System ARchitecture</td>
</tr>
<tr>
<td>BMM</td>
<td>Business Motivation Model</td>
</tr>
<tr>
<td>BPMN</td>
<td>Business Process Model and Notation</td>
</tr>
<tr>
<td>CBSE</td>
<td>Component-Based Software Engineering</td>
</tr>
<tr>
<td>CDO</td>
<td>Connected Data Objects</td>
</tr>
<tr>
<td>CP</td>
<td>Constraint Programming</td>
</tr>
<tr>
<td>CPS</td>
<td>Cyber-physical Systems</td>
</tr>
<tr>
<td>CSP</td>
<td>Constraint Solving Problem</td>
</tr>
<tr>
<td>CSR</td>
<td>Case Study Requirement</td>
</tr>
<tr>
<td>CTS</td>
<td>Conceptual Tool Set</td>
</tr>
<tr>
<td>DL</td>
<td>Description Logics</td>
</tr>
<tr>
<td>DMA</td>
<td>Direct Memory Access</td>
</tr>
<tr>
<td>DSE</td>
<td>Design Space Exploration</td>
</tr>
<tr>
<td>DSL</td>
<td>Domain-Specific Language</td>
</tr>
<tr>
<td>DSML</td>
<td>Domain-Specific Modelling Language</td>
</tr>
<tr>
<td>EAST-EEA</td>
<td>Electronics Architecture &amp; Software Technologies - Embedded Electronic Architecture</td>
</tr>
<tr>
<td>EMF</td>
<td>Eclipse Modeling Framework</td>
</tr>
<tr>
<td>EMOF</td>
<td>Essential MOF</td>
</tr>
<tr>
<td>EPL</td>
<td>Eclipse Public License</td>
</tr>
<tr>
<td>FBD</td>
<td>Function Block Diagram</td>
</tr>
<tr>
<td>FMI</td>
<td>Functional Mock-up Interface</td>
</tr>
<tr>
<td>FOSS</td>
<td>Free Open Source Software</td>
</tr>
<tr>
<td>FUML</td>
<td>Semantics of a Foundational Subset for Executable UML Model</td>
</tr>
<tr>
<td>GMF</td>
<td>Graphical Modeling Framework</td>
</tr>
<tr>
<td>GPL</td>
<td>General-Purpose (modelling) Languages</td>
</tr>
<tr>
<td>GPL</td>
<td>GNU Public License</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HUT-TCS</td>
<td>Helsinki University of Technology - Laboratory for Theoretical Computer Science</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>HW</td>
<td>HARDWARE</td>
</tr>
<tr>
<td>IEC</td>
<td>INTERNATIONAL ELECTROTECHNICAL COMMISSION</td>
</tr>
<tr>
<td>IEEE</td>
<td>INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS</td>
</tr>
<tr>
<td>INCOSE</td>
<td>INTERNATIONAL COUNCIL ON SYSTEMS ENGINEERING</td>
</tr>
<tr>
<td>ISO</td>
<td>INTERNATIONAL ORGANIZATION FOR STANDARDISATION</td>
</tr>
<tr>
<td>ITEA</td>
<td>INFORMATION TECHNOLOGY FOR EUROPEAN ADVANCEMENT</td>
</tr>
<tr>
<td>LBT</td>
<td>LEARNING-BASED TESTING</td>
</tr>
<tr>
<td>LGPL</td>
<td>LESSER GNU PUBLIC LICENSE</td>
</tr>
<tr>
<td>M2C</td>
<td>MODEL TO CODE</td>
</tr>
<tr>
<td>M2M</td>
<td>MODEL TO MODEL</td>
</tr>
<tr>
<td>M2T</td>
<td>MODEL TO TEXT</td>
</tr>
<tr>
<td>MAENAD</td>
<td>MODEL-BASED ANALYSIS &amp; ENGINEERING OF NOVEL ARCHITECTURES FOR DEPENDABLE ELECTRIC VEHICLES</td>
</tr>
<tr>
<td>MAF</td>
<td>MAJOR FRAME</td>
</tr>
<tr>
<td>MARTE</td>
<td>MODELING AND ANALYSIS OF REAL-TIME EMBEDDED SYSTEMS</td>
</tr>
<tr>
<td>MAST</td>
<td>MODELING AND ANALYSIS SUITE FOR REAL-TIME APPLICATIONS</td>
</tr>
<tr>
<td>MBD</td>
<td>MODEL-BASED DEVELOPMENT</td>
</tr>
<tr>
<td>MBSE</td>
<td>MODEL-BASED SYSTEM ENGINEERING</td>
</tr>
<tr>
<td>MBT</td>
<td>MODEL-BASED TESTING</td>
</tr>
<tr>
<td>MDE</td>
<td>MODEL-DRIVEN ENGINEERING</td>
</tr>
<tr>
<td>MDSD</td>
<td>MODEL-DRIVEN SOFTWARE DEVELOPMENT</td>
</tr>
<tr>
<td>MDT</td>
<td>ECLIPSE MODELING DEVELOPMENT TOOLS</td>
</tr>
<tr>
<td>MFR</td>
<td>MMRT FRAMEWORK REQUIREMENT</td>
</tr>
<tr>
<td>ML</td>
<td>MODELLING LANGUAGE</td>
</tr>
<tr>
<td>MoC</td>
<td>MODEL OF COMPUTATION</td>
</tr>
<tr>
<td>MOF</td>
<td>META OBJECT FACILITY</td>
</tr>
<tr>
<td>M&amp;S</td>
<td>MODELLING AND SIMULATION</td>
</tr>
<tr>
<td>NFP</td>
<td>NON FUNCTIONAL PROPERTY</td>
</tr>
<tr>
<td>OAL</td>
<td>OBJECT ACTION LANGUAGE</td>
</tr>
<tr>
<td>OCL</td>
<td>OBJECT CONSTRAINT LANGUAGE</td>
</tr>
<tr>
<td>OCRA</td>
<td>OTHELLO CONTRACTS REFINEMENT ANALYSIS</td>
</tr>
<tr>
<td>ODM</td>
<td>ONTOLOGY DEFINITION METAMODEL</td>
</tr>
<tr>
<td>OMG</td>
<td>OBJECT MANAGEMENT GROUP</td>
</tr>
<tr>
<td>OMT</td>
<td>OBJECT-MODELING TECHNIQUE</td>
</tr>
<tr>
<td>OOA</td>
<td>OBJECT-ORIENTED ANALYSIS</td>
</tr>
<tr>
<td>OOM</td>
<td>OBJECT-ORIENTED MODELLING</td>
</tr>
<tr>
<td>OOSA</td>
<td>OBJECT-ORIENTED SYSTEMS ANALYSIS</td>
</tr>
<tr>
<td>OOSE</td>
<td>OBJECT-ORIENTED SOFTWARE ENGINEERING</td>
</tr>
<tr>
<td>OS</td>
<td>OPERATING SYSTEM</td>
</tr>
<tr>
<td>OWL</td>
<td>WEB ONTOLOGY LANGUAGE</td>
</tr>
<tr>
<td>PAL</td>
<td>PLATFORM-INDEPENDENT ACTION LANGUAGE</td>
</tr>
<tr>
<td>PDM</td>
<td>PLATFORM DESCRIPTION MODEL</td>
</tr>
<tr>
<td>PIM</td>
<td>PLATFORM INDEPENDENT MODEL</td>
</tr>
<tr>
<td>PLC</td>
<td>PROGRAMMABLE LOGIC CONTROLLERS</td>
</tr>
<tr>
<td>PSCS</td>
<td>PRECISE SEMANTICS OF UML COMPOSITE STRUCTURES</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>PSM</td>
<td>Platform Specific Model</td>
</tr>
<tr>
<td>PSSM</td>
<td>Precise Semantics of UML State Machines</td>
</tr>
<tr>
<td>PTA</td>
<td>Priced Timed Automata</td>
</tr>
<tr>
<td>QVT</td>
<td>Query/View/Transformation</td>
</tr>
<tr>
<td>RC</td>
<td>Resource-Constrained</td>
</tr>
<tr>
<td>REAL</td>
<td>Requirement Enforcement Analysis Language</td>
</tr>
<tr>
<td>RTES</td>
<td>Real-Time Embedded Systems</td>
</tr>
<tr>
<td>S3D</td>
<td>Single Source System Design</td>
</tr>
<tr>
<td>SAT</td>
<td>Propositional Satisfiability Problem</td>
</tr>
<tr>
<td>SBSE</td>
<td>Search-Based Software Engineering</td>
</tr>
<tr>
<td>SCADE</td>
<td>Safety-Critical Application Development Environment</td>
</tr>
<tr>
<td>SCRALL</td>
<td>Starr’s Concise Relational Action Language</td>
</tr>
<tr>
<td>SWRL</td>
<td>Semantic Web Rule Language</td>
</tr>
<tr>
<td>SMALL</td>
<td>SHLAER-Mellow Action Language</td>
</tr>
<tr>
<td>SMC</td>
<td>Statistical Model Checking</td>
</tr>
<tr>
<td>SMM</td>
<td>Structured Metrics Metamodel</td>
</tr>
<tr>
<td>SMT</td>
<td>SAT Modulo Theories</td>
</tr>
<tr>
<td>SMUML</td>
<td>Symbolic Methods for UML Behavioural Diagrams</td>
</tr>
<tr>
<td>SOA</td>
<td>Service-Oriented Architecture</td>
</tr>
<tr>
<td>SUT</td>
<td>System Under Test</td>
</tr>
<tr>
<td>SW</td>
<td>Software</td>
</tr>
<tr>
<td>SWRL</td>
<td>Semantic Web Rule Language</td>
</tr>
<tr>
<td>SysML</td>
<td>System Modelling Language</td>
</tr>
<tr>
<td>TADL</td>
<td>Timing Augmented Description Language</td>
</tr>
<tr>
<td>TCP</td>
<td>Tool Component Purpose</td>
</tr>
<tr>
<td>TCTL</td>
<td>Timed Computation Tree Logic</td>
</tr>
<tr>
<td>TIMMO</td>
<td>TIMing MOdel</td>
</tr>
<tr>
<td>TS-MM</td>
<td>MegaModelling Tool Set</td>
</tr>
<tr>
<td>TS-RT</td>
<td>RunTime Tool Set</td>
</tr>
<tr>
<td>TS-SE</td>
<td>System Engineering Tool Set</td>
</tr>
<tr>
<td>TSC</td>
<td>Tool Set Component</td>
</tr>
<tr>
<td>TSP</td>
<td>Time and Space Partitioning</td>
</tr>
<tr>
<td>T&amp;E</td>
<td>Test and Evaluation</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modeling Language</td>
</tr>
<tr>
<td>UCAN</td>
<td>Unconventional Computer Architecture and Networks</td>
</tr>
<tr>
<td>UPA</td>
<td>UPPAAL Timed automata</td>
</tr>
<tr>
<td>UTP</td>
<td>UML Testing Profile</td>
</tr>
<tr>
<td>V&amp;V</td>
<td>Verification and Validation</td>
</tr>
<tr>
<td>W3C</td>
<td>World Wide Web Consortium</td>
</tr>
<tr>
<td>XMCF</td>
<td>XTratU M Configuration File</td>
</tr>
<tr>
<td>XMI</td>
<td>XML Metadata Interchange</td>
</tr>
<tr>
<td>XML</td>
<td>eXtensible Markup Language</td>
</tr>
<tr>
<td>XSD</td>
<td>XML Schema Definition</td>
</tr>
<tr>
<td>xUML</td>
<td>eXecutable Translatable UML</td>
</tr>
<tr>
<td>xUML</td>
<td>EXECUTABLE UML</td>
</tr>
</tbody>
</table>
1. Introduction

The Deliverable D3.3 reports about the initial version (at M15) of the MegaM@Rt2 Runtime Analysis Tool Set, developed within WP3 in Tasks T3.2, T3.3 and T3.4, to support automated code generation and model execution, log analysis, and runtime verification and testing activities.

To meet these expectations, the objective of Task T3.2 is to obtain directly and automatically some parts of the final system from the design models. Task T3.3 investigates methods and tools for log analysis in order to extract relevant information needed or produced by the verification process. Task T3.4 aims to develop novel methods and tools for scalable runtime verification and testing of system implementations. It should finally be noted that Task T3.5 will also provides a set of guidelines for applying the methods and tool prototypes developed in this WP for using models at runtime, while feedback from the use case providers is analyzed and used for iterating on and improving the outcomes.

To drive the developments, preliminary steps have been previously conducted and reported. The tooling expectations, and related roadmap to implement the intended features regarding runtime analysis, are especially based on the following inputs:

- the Deliverable D1.1 [2], entitled “Industry Requirements Specification”, defines, from a user point of view, the case study requirements and illustrates them on case study scenarios that will be used to demonstrate the achievements of the tool set capabilities;
- the Deliverable D1.2 [4], entitled “Architecture Specification and Roadmap - initial version”, defines the initial vision of the global architecture of the tooled MegaM@Rt2 Framework by describing the conceptual tool architecture and introducing the individual tools provided by the tool provider partners;
- the Deliverable D3.1 [5], entitled “Foundations for Model-Based Runtime Methods”, provides an overview of the foundations of model-based runtime methods and technologies in order to support the innovation tasks of WP3;
- the Deliverable D3.2 [1], entitled “Runtime Analysis Tool Set”, precisely identifies the Framework requirements to be implemented by the tools to address the case study requirements, and provides a precise roadmap to achieve them.

The Framework requirements identified in the Deliverable D3.2 [1] define for each tool functional objectives that have to be implemented during the project. Their expected achievements are also scheduled in a dedicated roadmap to be able to conduct in time the case study experiments and to allow end users to evaluate the tools at the soonest to give fruitful feedback. The Framework roadmap is structured on three release milestones: initial version at M15, intermediate version at M20, and final version at M32. Within the WP3 project, each milestone gives rise to a dedicated Tool Deliverable: D3.3 at M15, D3.4 at M20 and D3.5 at M32. The present report, subtitled “D3.3 Accompanying Note”, forms part of the Tool Deliverable D3.3. It aims to clarify the implementation status of the MegaM@Rt2 tool set at M15 and to highlight the current achievements regarding the targeted requirements, related to WP3 activities, within the initial version of the framework. It finally introduces the useful information and links to download the tools (source code and/or executable) that constitute the other material delivered within the Deliverable D3.3.

It should be noted that this WP3 scheduling is common with the other technical WP of the project, i.e. WP2 (System Engineering) and WP4 (Global Model and Traceability Management), which also provide similar Deliverables to address their own challenges. This synchronization between all the technical activities of the project makes it possible, within WP5 (Integration, Case Study Development and Evaluation), to deliver in a continuous way a consistent whole framework to the case-study providers for conducting experiments and evaluating the tooling achievements.
1.1. Reminder of the conceptual framework and tool set architecture

Within the MegaM@Rt2 Framework, several tools, demonstrators and prototypes, developed within WP2, WP3 and WP4, shall be combined and shall be able to interact with each other and with the existing different platforms deployed by use case providers. In order to have a seamless integration between all relevant software components and to ease the deployment of such components into case study technical environments, a conceptual framework and a global tool set architecture (addressing in a common way issues of model design of WP2, runtime analyses of WP3 and model management and traceability of WP4) have been defined at the beginning of the project. They make it possible, at the architecture level, to identify relevant interfaces to better satisfy the tooling needs and expected requirements described by the MegaM@Rt2 use case providers. To allow a continuous update and compliance of these keystone artefacts, the Modelio Constellation tool is used as a single reference to synchronize the global tool set architecture of the project.

The conceptual framework and tool architecture have been initially defined and introduced in Deliverable D1.2 [4]. Some refinements and updates have been however provided during the development process. The framework and architecture resulting from these refinements are presented and precisely described in Deliverable D1.4 [6].

Regarding WP3 issues, Figure 1 depicts the current architectural decomposition of the Runtime Analysis Tool Set framework, that consists of a set of subordinate components that cover all the expected needs regarding runtime analysis issues, i.e. trace collection and analysis, automated code generation, probes injection, monitoring and log analysis, and runtime verification and online model-based testing.
Therefore, all the tools delivered in the present Deliverable 3.3 instantiate and contribute to one or more identified subordinate components.

To use and combine the tools of these components, and more generally all the tools integrating the global MegaM@Rt2 framework (including Design tools from WP2 and Model Management infrastructure tools from WP4), a set of interface has been defined (see Figure 2). This common strategy enables on the one hand to facilitate the compliance and the technical interaction between the different tools, and on the other hand it offers an efficient way to manage a continuous development approach.

These conceptual and architectural models that define the current MegaM@Rt2 Framework are fully described and each model element is detailed in Deliverable D1.4 [6].

1.2. Objectives and contents of the deliverable D3.3

The present Deliverable D3.3 reports on the first released version of the MegaM@Rt2 Framework, called the initial version and released at M15. This version is focusing on the runtime analysis tools developed within WP3 and thus dedicated to automated code generation and model execution, log analysis, as well as runtime verification and testing activities.

From the baseline version of the tools, which constitutes the starting point of the project development, new features and capabilities have been implemented and are now available to conduct experiments and evaluate the model-based approaches of the project.
More precisely, the initial version of the MegaM@Rt2 Framework, that constitutes the Deliverable D3.3 of the project, is composed of the following material for each tool:

- the executable files and/or source code of the MegaM@Rt2 runtime analysis tools;
- basic information about installation instructions and usage guidelines;
- the present accompanying note that completes and clarifies the delivered tool components by:
  - providing, for each tool, the references about the above mentioned material,
  - giving the status of the current achievements at M15 about the functional objectives, as stated in Deliverable D3.2 [1],
  - highlighting the progress regarding the roadmap proposed in Deliverable D3.2 [1].

1.3. Structure of this document

The present accompanying note is structured as follows. Section 2 reports, for each tool, the detailed status of the planned activities highlighting the purposes not in line or deferred with respect to the initial timeline. For these activities, the delay justification is provided as well as the analysis of the potential consequences. Section 3 summarizes and applies the tool set status to the WP3 roadmap at M15. Section 4 includes information on the structure and organization of the tool resources. Section 5 identifies the dependencies and relationships of the WP3 activities with the ones developed within other WPs, in particular with the WP1 (related to the framework architecture definition and the use case scenarios implementation), with the technical work packages WP2 and WP4 (that complete the framework capabilities on system modeling and respectively on model management and traceability), and with the WP5 (addressing the integration of the whole set of tools in the MegaM@Rt2 Framework). Finally, after a short conclusion in Section 6, the Appendix collects, for each tool, additional and useful information that should help case study providers on using the MegaM@Rt2 Framework.

It should be noted that this document is structured in a practical form that will be reused to report the status of the next intermediate (at M20) and final (at M32) milestones, simplifying the evaluation and the comparison of the relative progress between them.

2. Progress status of the runtime analysis tool set

This section gives an overview of the implementation achievements and development status of the runtime analysis tools with regards to the initial milestone, as stated in Deliverable D3.2 [1]. The achievements of each tool are thus presented, and potentially commented, to make explicit the achievements of the tool requirements (called purposes) both in the baseline and the initial version of the framework release.

The table introduced in Figure 3 shows an overview of the achievements at M15 according to the tool purposes, as identified and scheduled in Deliverable D3.2 [1]. To briefly synthesize these reported statuses, the following can be noted:

- 21 tools are related to the runtime analysis tool set framework,
- 173 tool purposes are related to the tools participating to runtime analysis issues:
  - 80 concern the release at the baseline milestone
  - 14 concern the release at the initial milestone (M15)
  - 28 concern the release at the intermediate milestone (M20)
  - 51 concern the release at the final milestone (M32)
It should be noted that Deliverable D3.3 focuses on the baseline and initial milestones. That is why, to avoid misunderstanding and to facilitate the review, issues and progress statuses related to intermediate and final versions are thus not presented in Figure 3. However, some development tasks scheduled for these two milestones have already started (see Deliverable D1.4 for further information [6]). This information is nevertheless reported in the next chapter, which gives an overview of the progress status by focusing on WP3 purposes.

<table>
<thead>
<tr>
<th>Tools</th>
<th>Baseline</th>
<th>Initial</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>All Related Purposes</td>
<td>Planned</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In progress</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Done</td>
</tr>
<tr>
<td>CompleteTest</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>S3D</td>
<td>12</td>
<td>3</td>
</tr>
<tr>
<td>PADRE</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>CHESS</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>XPM</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>Papyrus</td>
<td>19</td>
<td>11</td>
</tr>
<tr>
<td>AIPHS</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>JTL</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>PauWare</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>CMA</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>CertifyIt</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>MBeetle</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>Conformiq Designer</td>
<td>16</td>
<td>14</td>
</tr>
<tr>
<td>Convex Hull</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Clusterability</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Moka</td>
<td>9</td>
<td>5</td>
</tr>
<tr>
<td>MATERA2</td>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>Modelio</td>
<td>17</td>
<td>7</td>
</tr>
<tr>
<td>VeriATL</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>LIME Testbench</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>RCRS Toolset</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>TOTAL</td>
<td>173</td>
<td>79</td>
</tr>
</tbody>
</table>

Therefore, at M15, we can note the following assessments with regard to the released initial version of the tool framework:

- Regarding the baseline version, all expected tool purposes have been achieved (i.e. “Done” status), except 1 purpose of the Papyrus tool that has been canceled. This purpose (PAPYRUS-140) aims to support reverse and round-trip engineering. It was identified as relevant in early stages of the project, but it could not be traced back to any framework (system).

- Regarding the initial version, all expected tool purposes have been achieved except:
4 purposes concerning S3D (S3D-030, S3D-050 and S3D-080) and LIME Testbench (LIME-020) tools that are still in progress. S3D-030, S3D-050 and S3D-080 respectively concern the automation of performance estimations, the generation of a performance model, and the generation of C/C++ codes sources and makefiles. LIME-020 is about the generation of relevant test inputs using a constraint solver. However, this delay does not affect the capacity to efficiently use and evaluate both tools, and mitigations are committed, notably in terms of needed effort and user feedback need, to finalize the implementation and deliver the intended functionalities by the next milestone.

1 purpose of Modelio that has been postponed. This purpose (MODELIO-020) is about system engineering practices, and requires further progress on the methodological aspects to be addressed.

It should be noted that this section introduces all the tool purposes related to the tools participating to runtime analysis challenges. However, some purposes are not directly related to WP3 issues, but related to the other technical WPs. In this context, the postponed purpose of Modelio relates to WP2 (MODELIO-020), and among the 4 “in progress” purposes, only two belong to WP3 issues (S3D-080 and LIME-020), the others being part of WP2 (S3D-030 and S3D-050).

To sum up, at M15, except a minor delay on a few (4) tool purposes, no deviation occurs at this stage. Moreover, regarding the 4 tool purposes still in progress, mitigation has been identified and committed.

The rest of the section is structured with one subsection per tool that distinguishes and comments the tool purposes available at baseline and the ones expected for the present initial release at M15. These tool-related statuses are presented using synthetic tables automatically generated from Modelio models, which are used in a continuous way to manage the progress of the development from a functional and tool capabilities point of view.

It should also be noted that some tool purposes are not required by any CSR (empty box in last column). As reported in the Deliverable D3.2 [1], this set of requirements has nevertheless been maintained because they may be necessary to complete other required purposes, or may be simply used by the customers to improve solutions to their current problems.
## 2.1. CompleteTest

### 2.1.1. Features available at baseline

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Properties</th>
<th>Comments/Release notes</th>
<th>Affected CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>COMPTEST-010</strong>: CompleteTest shall provide automated tests for 61131-3 Function Block Diagrams</td>
<td>Criticality: High&lt;br&gt;Release: Baseline&lt;br&gt;Status: Done</td>
<td>The tool capability has been extended for different test goals and combinatorial coverage criteria.</td>
<td>BT_08&lt;br&gt;NOK_25&lt;br&gt;TRT_01&lt;br&gt;AINA_02&lt;br&gt;NOK_25&lt;br&gt;AINA_03&lt;br&gt;TRT_01&lt;br&gt;BT_12&lt;br&gt;BT_13&lt;br&gt;BT_14&lt;br&gt;IKER_09&lt;br&gt;NOK_22&lt;br&gt;NOK_23&lt;br&gt;NOK_24&lt;br&gt;NOK_25&lt;br&gt;AINA_02</td>
</tr>
<tr>
<td><strong>COMPTEST-040</strong>: CompleteTest shall provide coverage information for 61131-3 Function Block Diagram</td>
<td>Criticality: Medium&lt;br&gt;Release: Baseline&lt;br&gt;Status: Done</td>
<td>The tool supports direct coverage of the input space and structure of the diagram.</td>
<td>BT_14&lt;br&gt;IKER_12&lt;br&gt;IKER_13&lt;br&gt;IKER_14</td>
</tr>
</tbody>
</table>

### 2.1.2. Features planned for initial release (M15)

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Properties</th>
<th>Comments/Release notes</th>
<th>Affected CSR</th>
</tr>
</thead>
</table>
2.2. S3D

2.2.1. Features available at baseline

<table>
<thead>
<tr>
<th>PURPOSE</th>
<th>PROPERTIES</th>
<th>COMMENTS/RELEASE NOTES</th>
<th>AFFECTED CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The tool supports decision-based test generation but these decisions are based on the block outputs and not the internal structure. Ongoing work is dedicated to improving this.
<table>
<thead>
<tr>
<th>S3D-010: S3D SHALL SUPPORT UML/MARTE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Criticality:</strong> High</td>
</tr>
<tr>
<td><strong>Release:</strong> Baseline</td>
</tr>
<tr>
<td><strong>Status:</strong> Done</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>UML/MARTE SUPPORT HAS BEEN EXTENDED</td>
</tr>
<tr>
<td>ACCORDING TO THE NEEDS OF THE PROJECT.</td>
</tr>
<tr>
<td>THIS CAPABILITY IS AVAILABLE.</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Concern Participants:</strong></td>
</tr>
<tr>
<td>IKER_01</td>
</tr>
<tr>
<td>NOK_01</td>
</tr>
<tr>
<td>TEK_02</td>
</tr>
<tr>
<td>CSY_02</td>
</tr>
<tr>
<td>CSY_01</td>
</tr>
<tr>
<td>TEK_01</td>
</tr>
<tr>
<td>NOK_01</td>
</tr>
<tr>
<td>TRT_03</td>
</tr>
<tr>
<td>TEK_09</td>
</tr>
<tr>
<td>TRT_02</td>
</tr>
<tr>
<td>TRT_04</td>
</tr>
<tr>
<td>TRT_05</td>
</tr>
<tr>
<td>CSY_03</td>
</tr>
<tr>
<td>NOK_25</td>
</tr>
<tr>
<td>AINA_04</td>
</tr>
<tr>
<td>BT_08</td>
</tr>
<tr>
<td>NOK_07</td>
</tr>
<tr>
<td>NOK_25</td>
</tr>
<tr>
<td>TRT_02</td>
</tr>
<tr>
<td>TRT_05</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S3D-020: S3D SHALL SUPPORT ANALYSIS AND OPTIMIZATION OF MIXED-CRITICALITY EMBEDDED SYSTEMS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Criticality:</strong> High</td>
</tr>
<tr>
<td><strong>Release:</strong> Baseline</td>
</tr>
<tr>
<td><strong>Status:</strong> Done</td>
</tr>
<tr>
<td><strong>Concern Participants:</strong> WITHE THIS TOOL CAPABILITY IS YET AVAILABLE</td>
</tr>
<tr>
<td>CSY_03</td>
</tr>
<tr>
<td>NOK_25</td>
</tr>
<tr>
<td>AINA_01</td>
</tr>
<tr>
<td>NOK_07</td>
</tr>
<tr>
<td>NOK_08</td>
</tr>
<tr>
<td>BT_08</td>
</tr>
<tr>
<td>BT_12</td>
</tr>
<tr>
<td>CAM_02</td>
</tr>
<tr>
<td>CAM_03</td>
</tr>
</tbody>
</table>
S3D-120: S3D supports XMI interface to exchange models with external tools

Criticality: Medium
Release: Baseline
Status: Done

S3D relies on Eclipse capability to support XMI exchange models, and as such, it is available.

2.2.2. Features planned for initial release (M15)

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Properties</th>
<th>Comments/Release notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>S3D-030: S3D shall provide fast performance estimations</td>
<td>Criticality: High</td>
<td>The development of this feature has not been already closed. Issues: Delays on S3D-80 have extended the expected delivery date. Affect: UCAN is in direct contact with TCS, and working in the use case, so the initial evaluation of the tool should not be compromised. Additionally, UCAN has delayed some effort to the second year to close this activity.</td>
</tr>
</tbody>
</table>

CSY_01
NOK_02
CAM_02
CAM_03
2.3. **PADRE**

2.3.1. **Features available at baseline**

<table>
<thead>
<tr>
<th>PURPOSE</th>
<th>PROPERTIES</th>
<th>COMMENTS/RELEASE NOTES</th>
<th>Affected CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PADRE-010:</strong> PADRE SHALL SUPPORT UML SOFTWARE DESIGN MODELS PROFILED WITH MARTE FOR PERFORMANCE ATTRIBUTES.</td>
<td><strong>Criticality:</strong> High&lt;br&gt;<strong>Release:</strong> Baseline&lt;br&gt;<strong>Status:</strong> Done</td>
<td>THIS TOOL CAPABILITY IS YET AVAILABLE</td>
<td>IKER_01&lt;br&gt;NOK_01&lt;br&gt;TEK_02&lt;br&gt;CSY_02&lt;br&gt;CSY_01&lt;br&gt;TEK_01</td>
</tr>
</tbody>
</table>
2.3.2. Features planned for initial release (M15)

<table>
<thead>
<tr>
<th>PURPOSE</th>
<th>PROPERTIES</th>
<th>COMMENTS/RELEASE NOTES</th>
<th>AFFECTED CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>PADRE-020: PADRE SHALL PROVIDE FACILITIES FOR USER-DRIVEN SINGLE-STEP REFACTORING OF UML-MARTE SOFTWARE DESIGN MODELS, DRIVEN BY PERFORMANCE ANTIPATTERNS DETECTION.</td>
<td>CRITICALITY: HIGH RELEASE: Initial STATUS: DONE</td>
<td>THIS TOOL CAPABILITY HAS BEEN RELEASED AS PLANNED AND IT IS YET AVAILABLE</td>
<td>SYS-010100 SYS-060501 MTM-03000 IKER_01 NOK_01 TEK_01 CSY_02 CSY_01 TEK_01 IKER_16 IKER_17 VCE_01 TRT_05 TEK_05 NOK_16 NOK_21 NOK_25</td>
</tr>
<tr>
<td>PADRE-040: PADRE SHALL PROVIDE PORTING FACILITIES FOR TRANSLATING PERFORMANCE ANTIPATTERNS DETECTION RULES AND REFACTORING ACTIONS AMONG DIFFERENT LANGUAGES PROVIDED BY THE EPSILON PLATFORM.</td>
<td>CRITICALITY: MEDIUM RELEASE: Initial STATUS: DONE</td>
<td>THIS TOOL CAPABILITY HAS BEEN RELEASED AS PLANNED AND IT IS YET AVAILABLE</td>
<td>SYS-010100 SYS-060501 MTM-03000</td>
</tr>
</tbody>
</table>

2.4. CHESS

2.4.1. Features available at baseline

<table>
<thead>
<tr>
<th>PURPOSE</th>
<th>PROPERTIES</th>
<th>COMMENTS/RELEASE NOTES</th>
<th>AFFECTED CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHESS-010: CHESS PROVIDES CHESSML MODELLING LANGUAGE, IMPLEMENTED AS UML, MARTE AND SysML PROFILE</td>
<td>CRITICALITY: Low RELEASE: Baseline STATUS: Done</td>
<td>THIS TOOL CAPABILITY IS YET AVAILABLE</td>
<td>IKER_01 NOK_01 TEK_02 CSY_02 CSY_01 TEK_01</td>
</tr>
<tr>
<td>CHESS-020: CHESS PROVIDES A COMPONENT MODEL TO REPRESENT FUNCTIONAL AND NON-FUNCTIONAL ASPECT OF A SYSTEM</td>
<td>CRITICALITY: Low RELEASE: Baseline STATUS: Done</td>
<td>THIS TOOL CAPABILITY IS YET AVAILABLE</td>
<td>NOK_02 CSY_01 NOK_02</td>
</tr>
<tr>
<td>CHESS-030: CHESS SUPPORTS THE MODELLING OF CONTRACTS</td>
<td>CRITICALITY: Low RELEASE: Baseline STATUS: Done</td>
<td>THIS TOOL CAPABILITY IS YET AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>CHESS-040: CHESS PROVIDES CONTRACT REFINEMENT ANALYSIS</td>
<td>CRITICALITY: Low RELEASE: Baseline STATUS: Done</td>
<td>THIS TOOL CAPABILITY IS YET AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>CHESS-050: CHESS SUPPORTS THE MODELLING OF TIMING AND DEPENDABILITY PROPERTIES</td>
<td>CRITICALITY: Low RELEASE: Baseline STATUS: Done</td>
<td>THIS TOOL CAPABILITY IS YET AVAILABLE</td>
<td>TRT_03 TRT_04 NOK_22</td>
</tr>
<tr>
<td>CHESS-060: CHESS PROVIDES SCHEDULABILITY ANALYSIS</td>
<td>CRITICALITY: Low RELEASE: Baseline STATUS: Done</td>
<td>THIS TOOL CAPABILITY IS YET AVAILABLE</td>
<td>TRT_03 TRT_04 NOK_22</td>
</tr>
<tr>
<td>CHESS-070: CHESS PROVIDES STATE BASED ANALYSIS</td>
<td>CRITICALITY: Low RELEASE: Baseline STATUS: Done</td>
<td>THIS TOOL CAPABILITY IS YET AVAILABLE</td>
<td>TRT_03 TRT_04 NOK_22</td>
</tr>
<tr>
<td>CHESS-080: CHESS PROVIDES FAILURE PROPAGATION ANALYSIS</td>
<td>CRITICALITY: Low RELEASE: Baseline STATUS: Done</td>
<td>THIS TOOL CAPABILITY IS YET AVAILABLE</td>
<td>TRT_03 TRT_04 NOK_22</td>
</tr>
<tr>
<td>CHESS-090: CHESS PROVIDES MODEL-TO-MODEL TRANSFORMATION FROM PIM TO PSM TO SUPPORT SCHEDULABILITY ANALYSIS</td>
<td>CRITICALITY: Low RELEASE: Baseline STATUS: Done</td>
<td>THIS TOOL CAPABILITY IS YET AVAILABLE</td>
<td>TRT_03 TRT_04 NOK_22</td>
</tr>
</tbody>
</table>
### CHESS-100: CHESS supports ADA code generation for structural elements

<table>
<thead>
<tr>
<th>CRITICALITY: Low</th>
<th>RELEASE: Baseline</th>
<th>STATUS: Done</th>
<th>THIS TOOL CAPABILITY IS YET AVAILABLE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 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

### CHESS-101: CHESS support XMI interface to exchange models with external tools

<table>
<thead>
<tr>
<th>CRITICALITY: Low</th>
<th>RELEASE: Baseline</th>
<th>STATUS: Done</th>
<th>THIS TOOL CAPABILITY IS YET AVAILABLE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- CAM_01
- CSY_02
- IKER_03
- NOK_04
- NOK_05
- VCE_06

#### 2.4.2. Features planned for initial release (M15)

No new features are planned for initial release.

#### 2.5. XPM

#### 2.5.1. Features available at baseline

No features are available at baseline.

#### 2.5.2. Features planned for initial release (M15)
**XPM-010:** XtratuM Plug-in shall provide a new type of project, the XtratuM Project Manager (XPM)

<table>
<thead>
<tr>
<th>CRITICALITY: High</th>
<th>RELEASE: Initial</th>
<th>STATUS: Done</th>
<th>THIS TOOL CAPABILITY HAS BEEN RELEASED AND IS AVAILABLE.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| CSY_01 | NOK_02 | BT_02 | IKER_04 | TRT_04 | IKER_16 | 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 |

**XPM-020:** XtratuM Plug-in shall provide XPM project management functionality

<table>
<thead>
<tr>
<th>CRITICALITY: High</th>
<th>RELEASE: Initial</th>
<th>STATUS: Done</th>
<th>THIS TOOL CAPABILITY HAS BEEN RELEASED AND IS AVAILABLE.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| BT_02 | 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 |
XPM-030: Given a XtratuM installation path, XtratuM plug-in shall provide the following XPM project model elements: Xtratum, execution environment(s) and partition(s)

<table>
<thead>
<tr>
<th>CRITICALITY: High Release: Initial Status: Done</th>
</tr>
</thead>
<tbody>
<tr>
<td>THIS TOOL CAPABILITY HAS BEEN RELEASED AND IS AVAILABLE.</td>
</tr>
</tbody>
</table>

XPM-040: XtratuM plug-in shall provide support for the following execution environments: Partikle, XAL and LithOS

<table>
<thead>
<tr>
<th>CRITICALITY: High Release: Initial Status: Done</th>
</tr>
</thead>
<tbody>
<tr>
<td>THIS TOOL CAPABILITY HAS BEEN RELEASED AND IS AVAILABLE.</td>
</tr>
</tbody>
</table>

### 2.6. Papyrus

#### 2.6.1. Features available at baseline
<table>
<thead>
<tr>
<th>Purpose</th>
<th>Properties</th>
<th>Comments/Release notes</th>
<th>Affected CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PAPYRUS-010:</strong> Give support to 100% UML standard language, including all diagrams</td>
<td><strong>Criticality:</strong> High, <strong>Release:</strong> Baseline, <strong>Status:</strong> Done</td>
<td>This tool capability is yet available.</td>
<td>IKER_01, NOK_01, TEK_02, CSY_02, CSY_01, TEK_01, TEK_04, BT_01, IKER_18, NOK_16, NOK_19, NOK_20, TEK_05, TRT_05, VCE_01</td>
</tr>
<tr>
<td><strong>PAPYRUS-020:</strong> Leverage UML extension mechanisms to create DSLs</td>
<td><strong>Criticality:</strong> High, <strong>Release:</strong> Baseline, <strong>Status:</strong> Done</td>
<td>This tool capability is yet available.</td>
<td>IKER_04, NOK_01, NOK_03, TEK_02, IKER_01, CSY_02</td>
</tr>
<tr>
<td><strong>PAPYRUS-030:</strong> Customize UML editors, views, perspectives and model explorer based on UML stereotypes</td>
<td><strong>Criticality:</strong> High, <strong>Release:</strong> Baseline, <strong>Status:</strong> Done</td>
<td>This tool capability is yet available.</td>
<td>IKER_04, NOK_01, NOK_03, TEK_02, IKER_01, CSY_02</td>
</tr>
<tr>
<td><strong>PAPYRUS-040:</strong> Give support non-functional aspects modeling</td>
<td><strong>Criticality:</strong> High, <strong>Release:</strong> Baseline, <strong>Status:</strong> Done</td>
<td>This tool capability is yet available.</td>
<td>NOK_02, CSY_01, NOK_02</td>
</tr>
<tr>
<td>Project</td>
<td>Criticality</td>
<td>Status</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>--------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>PAPYRUS-050</td>
<td>High</td>
<td>Baseline</td>
<td>This tool capability is yet available.</td>
</tr>
<tr>
<td>PAPYRUS-060</td>
<td>Medium</td>
<td>Baseline</td>
<td>This tool capability is yet available.</td>
</tr>
<tr>
<td>PAPYRUS-070</td>
<td>High</td>
<td>Baseline</td>
<td>This tool capability is yet available.</td>
</tr>
<tr>
<td>PAPYRUS-080</td>
<td>High</td>
<td>Baseline</td>
<td>This tool capability is yet available.</td>
</tr>
<tr>
<td>PAPYRUS-090</td>
<td>High</td>
<td>Baseline</td>
<td>This tool capability is yet available.</td>
</tr>
<tr>
<td>PAPYRUS-130</td>
<td>High</td>
<td>Baseline</td>
<td>This tool capability is yet available.</td>
</tr>
</tbody>
</table>
2.6.2. Features planned for initial release (M15)

No new features are planned for initial release.

2.7. AIPHS

2.7.1. Features available at baseline

<table>
<thead>
<tr>
<th>PURPOSE</th>
<th>PROPERTIES</th>
<th>COMMENTS/RELEASE NOTES</th>
<th>AFFECTED CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIPHS-010: Distributed hardware monitoring system for target on FPGAs, written in VHDL</td>
<td>Criticality: High Release: Baseline Status: Done</td>
<td>This tool capability is yet available</td>
<td>NOK_25 BT_08 NOK_25 IKER_24 IKER_26 NOK_25 AINA_04 TEK_07 CAM_05 NOK_25</td>
</tr>
</tbody>
</table>
### 2.7.2. Features planned for initial release (M15)

No new features are planned for initial release.

### 2.8. JTL

#### 2.8.1. Features available at baseline

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Properties</th>
<th>Comments/Release Notes</th>
<th>Affected CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>JTL-010: JTL shall support the definition of software design models in a metamodel-independent way (within EMF).</td>
<td>Criticality: Medium Release: Baseline Status: Done</td>
<td>JTL is built on top of EMF and provides support for modeling activities</td>
<td>CSY_03 NOK_25 IKER_01 NOK_01 TEK_02 CSY_02 CSY_01 TEK_01 IKER_04 NOK_01 NOK_03 TEK_02 IKER_01 CSY_02 NOK_01</td>
</tr>
</tbody>
</table>
### 2.8.2. Features planned for initial release (M15)

<table>
<thead>
<tr>
<th>PURPOSE</th>
<th>PROPERTIES</th>
<th>COMMENTS/RELEASE NOTES</th>
<th>AFFECTED CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>TRT_03</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TRT_02</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TRT_05</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CAM_01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CSY_02</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IKER_03</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_04</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_05</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>VCE_06</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>BT_01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>BT_05</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IKER_16</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IKER_18</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_14</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_17</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_18</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_19</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_20</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>VCE_01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TRT_05</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TEK_05</td>
</tr>
</tbody>
</table>
### JTL-020: JTL shall support the definition of runtime models (within EMF).

**Criticality:** High  
**Release:** Initial  
**Status:** Done

JTL provides an initial version of the JTL runtime metamodel representing logs and traces (i.e., system log files).

<table>
<thead>
<tr>
<th>CSY_03</th>
<th>NOK_25</th>
<th>BT_01</th>
<th>BT_05</th>
<th>IKER_16</th>
<th>IKER_18</th>
<th>NOK_12</th>
<th>NOK_14</th>
<th>NOK_17</th>
<th>NOK_18</th>
<th>NOK_19</th>
<th>NOK_20</th>
<th>VCE_01</th>
<th>TRT_05</th>
<th>TEK_05</th>
</tr>
</thead>
</table>

### JTL-050: JTL shall collect traceability links in traceability models conform to a dedicated metamodel.

**Criticality:** Medium  
**Release:** Initial  
**Status:** Done

JTL provides an initial version of the JTL traceability metamodel representing traceability links between design model elements and runtime model elements.

<table>
<thead>
<tr>
<th>NOK_25</th>
<th>BT_01</th>
<th>BT_05</th>
<th>IKER_16</th>
<th>IKER_18</th>
<th>NOK_12</th>
<th>NOK_14</th>
<th>NOK_17</th>
<th>NOK_18</th>
<th>NOK_19</th>
<th>NOK_20</th>
<th>VCE_01</th>
<th>TRT_05</th>
<th>TEK_05</th>
</tr>
</thead>
</table>

## 2.9. PauWare

### 2.9.1. Features available at baseline

No features are available at baseline.

### 2.9.2. Features planned for initial release (M15)

No new features are planned for initial release.
2.10.  CMA

2.10.1.  Features available at baseline

No features are available at baseline.

2.10.2.  Features planned for initial release (M15)

No new features are planned for initial release.

2.11.  CertifyIt

2.11.1.  Features available at baseline

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Criticality</th>
<th>Release</th>
<th>Status</th>
<th>Comments/Release Notes</th>
<th>Affected CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>CERTIFYIT-020: CertifyIt shall provide user-friendly GUI to support model-based testing solution</td>
<td>MEDIUM</td>
<td>Baseline</td>
<td>DONE</td>
<td>All developed functionalities, expected within the project to support model-based testing, can be executed from the GUI of the CertifyIt tool.</td>
<td>BT_12, BT_13, IKER_09, NOK_22, NOK_23, NOK_24, NOK_25, AINA_02</td>
</tr>
<tr>
<td>CERTIFYIT-040: CertifyIt shall be connected to a modeling solution import UML/OCL test models</td>
<td>HIGH</td>
<td>Baseline</td>
<td>DONE</td>
<td>CertifyIt supports RSA UML/OCL modeling (subset of the notation).</td>
<td></td>
</tr>
<tr>
<td>CERTIFYIT-050: CertifyIt shall support syntactic and functional validation of UML/OCL test models</td>
<td>MEDIUM</td>
<td>Baseline</td>
<td>DONE</td>
<td>CertifyIt provides verification to ensure the syntactic and functional consistency of UML/OCL models.</td>
<td></td>
</tr>
<tr>
<td>CERTIFYIT-060: CertifyIt shall support verification to ensure test model consistency and compliance regarding the solution</td>
<td>HIGH</td>
<td>Baseline</td>
<td>DONE</td>
<td>CertifyIt provides verification to ensure the consistency of UML/OCL models regarding CertifyIt required format.</td>
<td></td>
</tr>
</tbody>
</table>
| CERTIFYIT-070: **CertifyIt** shall automate the derivation of functional black-box test cases from UML/OCL test models | **Criticality:** High  
**Release:** Baseline  
**Status:** Done | **CertifyIt** provides algorithms to automatically derive test scenarios from input UML/OCL model. | BT_10  
CSY_03  
NOK_25 |
|---|---|---|---|
| CERTIFYIT-080: **CertifyIt** shall publish specifications describing the generated test cases | **Criticality:** Medium  
**Release:** Baseline  
**Status:** Done | **Excel** files are supported to provide test specification. | NOK_25  
IKER_23  
NOK_25  
AINA_04  
BT_08  
IKER_23  
NOK_25  
TRT_01 |
| CERTIFYIT-090: **CertifyIt** shall publish documentations describing the generated test cases | **Criticality:** Medium  
**Release:** Baseline  
**Status:** Done | **HTML** and **Excel** are supported to provide test documentation. | NOK_25  
BT_08  
NOK_25  
TRT_01  
BT_14  
IKER_12  
IKER_13  
NOK_25  
NOK_26  
AINA_02  
CSY_04  
CSY_05 |
| CERTIFYIT-100: **CertifyIt** shall generate executable scripts implementing the generated test cases | **Criticality:** High  
**Release:** Baseline  
**Status:** Done | **JUnit** is supported to provide executable test scripts. | BT_08  
IKER_23  
NOK_25  
TRT_01 |
| CERTIFYIT-110: **CertifyIt** shall manage traceability between functional requirements, test model, generated test cases and related executables test script | **Criticality:** Medium  
**Release:** Baseline  
**Status:** Done | **TraceAbility** coverage between functional requirements, test model, generated test cases and related executables test script is automatically calculated and provided by **CertifyIt**. | NOK_25 |
2.11.2. Features planned for initial release (M15)

No new features are planned for initial release.

2.12. MBeetle

2.12.1. Features available at baseline

No features are available at baseline.

2.12.2. Features planned for initial release (M15)

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Properties</th>
<th>Comments/Release notes</th>
<th>Affected CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>MBEETLE-040: MBeetle shall be integrated into CertifyIt as a plugin to offer a turnkey solution</td>
<td>Criticality: High Release: Initial Status: Done</td>
<td>CertifyIt and MBeetle deployments are both included into a single process, avoiding configuration needs.</td>
<td></td>
</tr>
<tr>
<td>MBEETLE-050: MBeetle artefacts shall be compliant with those manipulated and provided by CertifyIt</td>
<td>Criticality: High Release: Initial Status: Done</td>
<td>All the input/output file formats are common between CertifyIt and MBeetle.</td>
<td></td>
</tr>
</tbody>
</table>

2.13. Conformiq Designer
### 2.13.1. Features available at baseline

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Properties</th>
<th>Comments/Release Notes</th>
<th>Affected CSR</th>
</tr>
</thead>
</table>
| **CQDESIGN-010:** Designer shall provide GUI to support automated testing | **Criticality:** High  
**Release:** Baseline  
**Status:** DONE | Tool capability readily available | IKER_04  
TRT_04  
IKER_16  
BT_08  
NOK_25  
TRT_01  
AINA_04  
BT_08  
NOK_07  
NOK_25  
TRT_02  
TRT_05 |
| **CQDESIGN-020:** Conformiq Designer shall support fully automatic generation of test cases from models | **Criticality:** High  
**Release:** Baseline  
**Status:** DONE | Tool capability readily available | CAM_02, BT_09  
BT_10, BT_11  
BT_08, BT_12  
CSY_01, NOK_02  
AINA_01, NOK_07  
NOK_08, BT_08  
BT_12, BT_09  
BT_11, NOK_07  
NOK_08, TEK_04  
TEK_04, VCE_05  
BT_10, NOK_07  
NOK_08, NOK_22  
BT_10, IKER_04  
NOK_22, BT_10  
NOK_22, NOK_06  
NOK_06, NOK_22  
BT_10, CSY_03  
NOK_25, NOK_25  
BT_09, CSY_04  
NOK_06, NOK_07  
NOK_25, BT_12  
NOK_22, NOK_25  
BT_12, BT_13 |
<table>
<thead>
<tr>
<th>CQDESIGN-030: Conformiq Designer shall support generation of executable test scripts</th>
<th>Criticality: High Release: Baseline Status: Done</th>
<th>Tool capability readily available</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>IKER_09, NOK_22 NOK_23, NOK_24 NOK_25, AINA_02</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CQDESIGN-040: Conformiq Designer shall support generation of human readable test documentation</th>
<th>Criticality: High Release: Baseline Status: Done</th>
<th>Tool capability readily available</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>BT_10, NOK_07 NOK_08, NOK_22 NOK_25, BT_12 NOK_22, NOK_23 NOK_24, NOK_25 AINA_02, BT_14 NOK_25, NOK_26 AINA_02, BT_14 AINA_01, NOK_25 NOK_26, CSY_06</td>
</tr>
<tr>
<td>CQDESIGN-050: Conformiq Designer shall support generation of test plans for manual test execution</td>
<td><strong>Criticality:</strong> High</td>
<td><strong>Release:</strong> Baseline</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>BT_10, NOK_07 NOK_08, NOK_22 BT_12, NOK_22 NOK_25, BT_12 BT_13, IKER_09 NOK_22, NOK_23 NOK_24, NOK_25 AINA_02, BT_14 IKER_12, IKER_13 NOK_25, NOK_26 AINA_02, CSY_04 CSY_05, IKER_14 AINA_01, NOK_25 NOK_26, CSY_06</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CQDESIGN-060: Conformiq Designer shall support traceability analysis between the generated test cases, test scripts, test documentation, test plans and the model</th>
<th><strong>Criticality:</strong> High</th>
<th><strong>Release:</strong> Baseline</th>
<th><strong>Status:</strong> Done</th>
<th><strong>Tool capability readily available</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>NOK_25, BT_12 BT_13, IKER_09 NOK_22, NOK_23 NOK_24, NOK_25 AINA_02, BT_14 IKER_12, IKER_13 NOK_25, NOK_26 AINA_02, CSY_04 CSY_05</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CQDESIGN-070: Conformiq Designer shall support generation of dependency based test cases</th>
<th><strong>Criticality:</strong> High</th>
<th><strong>Release:</strong> Baseline</th>
<th><strong>Status:</strong> Done</th>
<th><strong>Tool capability readily available</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>AINA_01, NOK_07 NOK_08, BT_08 BT_12, BT_09 BT_11, NOK_07 NOK_08, BT_10 NOK_07, NOK_08 NOK_22, BT_12 NOK_22, NOK_25 BT_12, BT_13 IKER_09, NOK_22 NOK_23, NOK_24 NOK_25, AINA_02 BT_14, IKER_12 IKER_13, NOK_25 NOK_26, AINA_02 CSY_04, CSY_05</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| **CQDESIGN-080:** Conformiq Designer shall support test generation from models based on user selected testing goals | **Criticality:** High  
**Release:** Baseline  
**Status:** Done | **Tool capability readily available** | AINA_01, NOK_07  
NOK_08, BT_08  
BT_12, BT_09  
BT_11, NOK_07  
NOK_08, BT_10  
NOK_07, NOK_08  
NOK_22, NOK_06  
BT_12, NOK_22  
NOK_25, BT_12  
BT_13, IKER_09  
NOK_22, NOK_23  
NOK_24, NOK_25  
AINA_02, BT_14  
IKER_12, IKER_13  
NOK_25, NOK_26  
AINA_02, CSY_04  
CSY_05 |
| **CQDESIGN-090:** Conformiq Designer shall support visualization of test cases in order to provide the end user an understanding of the purpose and origin of the test cases | **Criticality:** High  
**Release:** Baseline  
**Status:** Done | **Tool capability readily available** | BT_09, BT_11  
NOK_07, NOK_08  
TEK_04, NOK_22  
BT_08, NOK_25  
TRT_01, BT_14  
IKER_12, IKER_13  
NOK_25, NOK_26  
AINA_02, CSY_04  
CSY_05 |
| **CQDESIGN-100:** Conformiq Designer shall support test generation in a scalable fashion providing the means of deploying the solution on cloud/cluster | **Criticality:** High  
**Release:** Baseline  
**Status:** Done | **Tool capability readily available while being constantly improved and enhanced** | AINA_01, NOK_07  
NOK_08, BT_08  
BT_12, BT_10  
NOK_07, NOK_08  
NOK_22, BT_12  
BT_13, IKER_09  
NOK_22, NOK_23  
NOK_24, NOK_25  
AINA_02 |
<table>
<thead>
<tr>
<th>CQDESIGN-110: CONFORMIQ DESIGNER SHALL SUPPORT MODEL ANALYSIS AND DEBUGGING FACILITIES</th>
<th>CRITICALITY: High</th>
<th>TOOL CAPABILITY READILY AVAILABLE</th>
<th>AINA_01, NOK_07, NOK_08, BT_08, BT_12, BT_09, BT_11, NOK_07, NOK_08, TEK_04, NOK_22, BT_10, CSY_03, NOK_25, NOK_25, BT_09, CSY_04, NOK_06, NOK_07, NOK_25, AINA_04, BT_08, NOK_07, NOK_25, TRT_02, TRT_05, BT_14, IKER_12, IKER_13, NOK_25, NOK_26, AINA_02, CSY_04, CSY_05</th>
</tr>
</thead>
<tbody>
<tr>
<td>CQDESIGN-120: CONFORMIQ DESIGNER SHALL SUPPORT SIMULATION OF TEST CASES AGAINST THE MODEL</td>
<td>CRITICALITY: High</td>
<td>TOOL CAPABILITY READILY AVAILABLE</td>
<td>AINA_01, NOK_07, NOK_08, BT_08, BT_12, NOK_22, AINA_04, BT_08, NOK_07, NOK_25, TRT_02, TRT_05</td>
</tr>
<tr>
<td>CQDESIGN-130: CONFORMIQ DESIGNER SHALL SUPPORT MODEL PROFILING</td>
<td>CRITICALITY: High</td>
<td>TOOL CAPABILITY READILY AVAILABLE</td>
<td>AINA_01, NOK_07, NOK_08, BT_08, BT_12, BT_09, BT_11, NOK_07, NOK_08, TEK_04, NOK_22, AINA_04, BT_08, NOK_07, NOK_25, TRT_02, TRT_05</td>
</tr>
<tr>
<td>CQDESIGN-140: CONFORMIQ DESIGNER SHALL SUPPORT VERIFICATION OF THE MODEL TO ENSURE INTERNAL MODEL CONSISTENCY</td>
<td>CRITICALITY: High</td>
<td>TOOL CAPABILITY READILY AVAILABLE</td>
<td>AINA_01, NOK_07, NOK_08, BT_08, BT_12, NOK_22</td>
</tr>
<tr>
<td>CQDESIGN-150: CONFORMIQ DESIGNER SHALL PROVIDE GUIDELINES AND DOCUMENTATION TO USE THE SOLUTION</td>
<td>CRITICALITY: High</td>
<td>TOOL CAPABILITY READILY AVAILABLE</td>
<td>AINA_01, NOK_07, NOK_08, BT_12, NOK_22</td>
</tr>
</tbody>
</table>
2.13.2. Features planned for initial release (M15)

No new features are planned for initial release.

2.14. Convex Hull

2.14.1. Features available at baseline

No features are available at baseline.

2.14.2. Features planned for initial release (M15)

No new features are planned for initial release.

2.15. Clusterability

2.15.1. Features available at baseline

No features are available at baseline.

2.15.2. Features planned for initial release (M15)

No new features are planned for initial release.

2.16. Moka

2.16.1. Features available at baseline
<table>
<thead>
<tr>
<th>MOKA-010: Steady and precise semantic foundations for UML and related standards, such as SysML and MARTE</th>
<th>Criticality: High Release: Baseline Status: Done</th>
<th>This tool capability is yet available.</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOKA-020: Execution engine complying with OMG’s Foundational UML (fUML) standard for model animation and simulation</td>
<td>Criticality: High Release: Baseline Status: Done</td>
<td>This tool capability is yet available.</td>
</tr>
<tr>
<td>MOKA-030: Specifying the behaviour of executable models by means of an action language: OMG’s Action Language for fUML (Alf)</td>
<td>Criticality: High Release: Baseline Status: Done</td>
<td>This tool capability is yet available.</td>
</tr>
</tbody>
</table>
2.16.2. Features planned for initial release (M15)

No new features are planned for initial release.

2.17. MATERA2

2.17.1. Features available at baseline
MATERA2-070: MATERA2 SHALL PROVIDE SUPPORT FOR ON-LINE TEST EXECUTION AND MONITORING.

| CRITICALITY: HIGH |
| RELEASE: Baseline |
| STATUS: Done |
| THIS TOOL CAPABILITY IS YET AVAILABLE. |

| BT_13 |
| IKER_11 |
| IKER_12 |
| NOK_25 |
| NOK_26 |
| AINA_01 |
| CSY_03 |
| CSY_04 |
| BT_14 |
| IKER_12 |
| IKER_13 |
| NOK_25 |
| NOK_26 |
| AINA_02 |
| CSY_04 |
| CSY_05 |
| BT_08 |
| NOK_25 |

MATERA2-080: MATERA2 SHALL PROVIDE SUPPORT FOR LOAD TESTING.

| CRITICALITY: HIGH |
| RELEASE: Baseline |
| STATUS: Done |
| THIS TOOL CAPABILITY IS YET AVAILABLE. |

| NOK_25 |

2.17.2. Features planned for initial release (M15)

No new features are planned for initial release.

2.18. Modelio

2.18.1. Features available at baseline
<table>
<thead>
<tr>
<th>Issue</th>
<th>Description</th>
<th>Criticality</th>
<th>Release</th>
<th>Status</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODELIO-010</td>
<td>Modelio shall provide system modelling capabilities in SysML.</td>
<td>High</td>
<td>Baseline</td>
<td>Done</td>
<td>SysML is already available in Modelio.</td>
</tr>
<tr>
<td>MODELIO-030</td>
<td>Modelio shall support functional properties modelling on system level.</td>
<td>High</td>
<td>Baseline</td>
<td>Done</td>
<td>Modelio supports UML, SysML and MARTE for functional properties modeling.</td>
</tr>
<tr>
<td>MODELIO-040</td>
<td>Modelio shall support extra-functional properties modelling with MARTE.</td>
<td>High</td>
<td>Baseline</td>
<td>Done</td>
<td>Modelio supports MARTE for extra-functional properties modeling.</td>
</tr>
</tbody>
</table>

IKER_04
TRT_04
IKER_16
IKER_01
NOK_01
TEK_02
CSY_02
CSY_01
TEK_01
TEK_04

IKER_01
NOK_01
TEK_02
CSY_02
CSY_01
TEK_01
CSY_01
NOK_02
NOK_01
TRT_03
TRT_02
TRT_04
TRT_05
| **MODELIO-070**: Modelio shall manage traceability on Modelio project level. | **Criticality**: High  
**Release**: Baseline  
**Status**: Done | Modelio supports various traceability links that can be queried through the API. In adding the Fast Link Editor provides live diagrams for the diagrams. |
|---|---|---|
| **MODELIO-110**: Modelio provides code generation facilities for specific programming languages (Java, C++, and C for example). | **Criticality**: High  
**Release**: Baseline  
**Status**: Done | Modelio provides code generation for Java, C++, C#, SQL. |

| BT_01  
BT_05  
IKER_16  
IKER_18  
NOK_12  
NOK_14  
NOK_17  
NOK_18  
NOK_19  
NOK_20  
VCE_01  
TRT_01  
TEK_05  
IKER_18  
NOK_17  
NOK_18  
NOK_19  
TEK_05  
NOK_20  
TEK_01  
BT_06  
NOK_12 | 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 |
### 2.18.2. Features planned for initial release (M15)

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Properties</th>
<th>Comments/Release Notes</th>
<th>Affected CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**MODELIO-120:** Modelio shall provide reverse/round trip facilities for specific programming languages (Java, C++, and C for example).

**CRITICALITY:** High  
**RELEASE:** Baseline  
**STATUS:** Done

The reverse engineering is available for Java and SQL. Round-trip engineering is available for all above targets – Java, C++, C#, SQL.

**MODELIO-150:** Modelio shall provide XMI import/export.

**CRITICALITY:** Medium  
**RELEASE:** Baseline  
**STATUS:** Done

Modelio supports XMI in OMG and Eclipse EMF UML2 formats.

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

IKER_19  
NOK_25  
CSY_07  
CAM_01  
CSY_02  
IKER_03  
NOK_04  
NOK_05  
VCE_06
### MODELIO-020: Modelio shall support holistic system engineering practices.

<table>
<thead>
<tr>
<th>Criticality: High</th>
<th>Release: Initial</th>
<th>Status: Postponed</th>
<th>Support for holistic system engineering practices is postponed till further progress on the methodological aspects in WP2 and clarification of requirements and scenarios by the case study providers.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>CAM_02, CAM_03, BT_02, IKER_18, NOK_17, NOK_18, NOK_19, TEK_05, NOK_20, TEK_01, BT_06, NOK_12, NOK_19, NOK_20, BT_01, VCE_03, VCE_02, VCE_01, NOK_12, NOK_04, CAM_01, NOK_04, NOK_12, BT_04, BT_05, BT_06, TEK_04, VCE_05</td>
</tr>
</tbody>
</table>

### MODELIO-141: Modelio shall support the DDS generation.

<table>
<thead>
<tr>
<th>Criticality: High</th>
<th>Release: Initial</th>
<th>Status: Done</th>
<th>Support for generation of the DDS configuration files is required in the IKER scenarios.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>IKER_01, NOK_01, TEK_02, CSY_02, CSY_01, TEK_01, IKER_04, NOK_01, NOK_03, TEK_02, IKER_01, CSY_02</td>
</tr>
</tbody>
</table>
2.19. VeriATL

2.19.1. Features available at baseline

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Properties</th>
<th>Comments/Release Notes</th>
<th>Affected CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>VeriATL-010: VeriATL shall support the unbounded verification of already identified correctness properties on ATL model transformations.</td>
<td>Critciality: High</td>
<td>This tool capability is already available.</td>
<td>AINA_01, NOK_07, NOK_08, BT_08, BT_12, NOK_01, TRT_03, TEK_09, TRT_02, TRT_04, TRT_05, IKER_01, NOK_01, TEK_02, CSY_02, CSY_01, TEK_01</td>
</tr>
</tbody>
</table>

2.19.2. Features planned for initial release (M15)

No new features are planned for initial release.

2.20. LIME TestBench

2.20.1. Features available at baseline

No features are available at baseline.
### 2.20.2. Features planned for initial release (M15)

<table>
<thead>
<tr>
<th>PURPOSE</th>
<th>PROPERTIES</th>
<th>COMMENTS/RELEASE NOTES</th>
<th>AFFECTED CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LIME-010</strong>: Runtime monitoring of the execution of the program under test, already available in Java and C.</td>
<td></td>
<td></td>
<td>AINA_03, CSY_03 NOK_25, NOK_25 BT_08, NOK_25 IKER_23, NOK_25 AINA_04, BT_08 IKER_23, NOK_25 TRT_01, AINA_04 BT_08, NOK_07 NOK_25, TRT_02 TRT_05, IKER_24 IKER_26, NOK_25 IKER_25, NOK_25 AINA_04, CSY_06 TEK_07, NOK_25 NOK_25, IKER_19 NOK_25, CSY_07 CAM_01, NOK_04 NOK_12, BT_04 BT_05, BT_06 NOK_01, TRT_03 TEK_09, TRT_02 TRT_04, TRT_05 CSY_03, IKER_04 NOK_08, CSY_04 NOK_22, TRT_03 TRT_04, NOK_22</td>
</tr>
<tr>
<td><strong>CRITICALITY</strong>: High</td>
<td><strong>RELEASE</strong>: Initial</td>
<td><strong>STATUS</strong>: Done</td>
<td></td>
</tr>
</tbody>
</table>
LIME-020: Auto-generate high coverage test suites by using a constraint solver to produce inputs that force the program to take certain branches during execution.

<table>
<thead>
<tr>
<th>CRITICALITY: High</th>
</tr>
</thead>
<tbody>
<tr>
<td>RELEASE: Initial</td>
</tr>
<tr>
<td>STATUS: IN PROGRESS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Properties</th>
<th>Comments/Release notes</th>
<th>Affected CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>AINA_03, CSY_03</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_25, NOK_25</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>BT_12, NOK_22</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_25, BT_12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>BT_13, IKER_09</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_22, NOK_23</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_24, NOK_25</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>AINA_02, BT_13</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IKER_11, IKER_12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_25, NOK_26</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>AINA_01, CSY_03</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CSY_04, BT_14</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IKER_12, IKER_13</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_25, NOK_26</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>AINA_02, CSY_04</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CSY_05, IKER_14</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>AINA_01, NOK_25</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_26, CSY_06</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_25, NOK_01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TRT_03, TEK_09</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TRT_02, TRT_04</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TRT_05, AINA_01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_07, NOK_08</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>BT_08, BT_12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_07, NOK_08</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CSY_05, TRT_02</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TRT_05, IKER_16</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>BT_10, NOK_07</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_08, NOK_22</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>BT_10, IKER_04</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOK_22</td>
</tr>
</tbody>
</table>

2.21. RCRS

2.21.1. Features available at baseline
| RCRS-010: RCRS theory implemented in Isabelle. | Criticality: **High**  
Release: Baseline  
Status: Done | AINA_03  
CSY_03  
NOK_25  
BT_09  
CSY_04  
NOK_06  
NOK_25  
NOK_02  
CAM_01  
NOK_04  
NOK_12  
BT_04  
BT_05  
BT_06  
NOK_01  
TRT_03  
TEK_09  
TRT_02  
TRT_04  
TRT_05  
NOK_22 |
| --- | --- | --- |
| RCRS-020: Translation of Simulink diagrams into Isabelle RCRS theory (good coverage of Simulink blocks and options, limited support for action/enable ports). | Criticality: **High**  
Release: Baseline  
Status: Done | AINA_03  
CSY_03  
NOK_25  
BT_04  
CSY_04  
IKER_04  
NOK_22 |
### RCRS-030: Generation of Python simulation code.

**Criticality:** High  
**Release:** Baseline  
**Status:** Done

<table>
<thead>
<tr>
<th>AINA_03</th>
<th>CSY_03</th>
<th>NOK_25</th>
<th>BT_08</th>
<th>NOK_25</th>
<th>TRT_01</th>
<th>AINA_01</th>
<th>BT_07</th>
<th>BT_11</th>
<th>IKER_01</th>
<th>NOK_23</th>
<th>NOK_24</th>
<th>NOK_25</th>
<th>IKER_02</th>
<th>IKER_05</th>
<th>TEK_06</th>
<th>NOK_22</th>
<th>CSY_07</th>
<th>CSY_03</th>
<th>IKER_04</th>
<th>NOK_08</th>
<th>CSY_04</th>
<th>BT_04</th>
<th>CSY_04</th>
<th>IKER_04</th>
<th>NOK_22</th>
</tr>
</thead>
</table>

### RCRS-040: Formal Analyzer (compatibility check, simplification).

**Criticality:** High  
**Release:** Baseline  
**Status:** Done

<table>
<thead>
<tr>
<th>AINA_03</th>
<th>CSY_03</th>
<th>NOK_25</th>
<th>NOK_25</th>
<th>NOK_22</th>
</tr>
</thead>
</table>

#### 2.21.2. Features planned for initial release (M15)

No new features are planned for initial release.
3. Runtime analysis framework status, the roadmap profile

This section tracks, for the initial milestone at M15, the MegaM@Rt2 Framework implementation plan and the enhancements or upgrades of the tool features regarding WP3 challenges. For each framework requirement related to WP3, the roadmap table (previously defined in D3.2 [1] to specify the plan for the availability and integration of the WP3-relevant tool purposes) is reported with the current status of the tool developments. Therefore, this section defines a synthesis, from a WP3 requirements point of view, of the implementation achievements, introduced per tool in the previous chapter of this document.

For each category of framework requirements related to runtime analysis, a table describes, at M15, the expected tool purposes to be implemented and the current achievements regarding each milestone (baseline, initial at M15, intermediate at M20 and final at M32). It should be noted that, also in this case, the following tables are automatically generated from Modelio.

Regarding the purposes of the whole MegaM@Rt2 tool set, expected at M15, which are either still in progress (4 purposes concerned) or postponed (1 purpose concerned) at this milestone (that have been identified in the previous chapter), only 2 are directly related to runtime analysis issues: LIME-020 and S3D-080 that are still in progress (the other ones are mostly concern WP2 activities about design). For readability, these 2 “in progress” purposes are highlighted in orange color in the different tables. Noted that these purposes have already been identified in the previous chapter, where comments about reasons and proposed mitigation are given. In this regard, as already summarized in the previous chapter, the current status of the MegaM@Rt2 tool set development is basically in line with the defined plan. The slight deviation is controlled and do not require, at the moment, any recovery action.

3.1. Basic framework requirements

<table>
<thead>
<tr>
<th>Framework Features</th>
<th>Baseline</th>
<th>Initial</th>
<th>Intermediate</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RTA-00001:</strong> The framework shall use AOM techniques to facilitate code generation, runtime verification and validation</td>
<td>MODELIO-1 50</td>
<td>MODELIO-130 [in progress]</td>
<td>MODELIO-140 [in progress]</td>
<td></td>
</tr>
<tr>
<td><strong>RTA-00002:</strong> The framework shall support runtime analysis methods combining model-based development, validation and verification techniques</td>
<td>CERTIFYIT-0 70</td>
<td>CERTIFYIT-0 [done]</td>
<td>PAU-020 [in progress]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CQDESIGN-0 20</td>
<td>CQDESIGN-1 10</td>
<td>MBEETLE-010</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PAU-020</td>
<td>MODELIO-100 [in progress]</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>RTA-00003:</strong> The framework shall be able to manage complex and large-scale models</td>
<td>S3D-010</td>
<td>S3D-080 [in progress]</td>
<td>MODÉLIO-100</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S3D-080</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 3.2. Analysis of execution logs requirements

<table>
<thead>
<tr>
<th>Framework Features</th>
<th>Baseline</th>
<th>Initial</th>
<th>Intermediate</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RTA-10000:</strong> The tool chain should allow the parsing execution logs</td>
<td>MODELIO-090 [planned]</td>
<td>COMPTEST-030 ConvexHull-010 Clustera-010</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>RTA-10001:</strong> The framework should provide human-friendly graphs and data visualization, including timing properties</td>
<td>COMPTEST-010 10 CERTIFYIT-0 10 CQDESIGN-0 10 CQDESIGN-0 90 RCRS-030</td>
<td>MBEETLE-110 RCRS-080 [planned]</td>
<td>MODELIO-090 [planned] COMPTEST-030 MBEETLE-020 [in progress] MATERA-030</td>
<td></td>
</tr>
<tr>
<td><strong>RTA-10002:</strong> The framework should allow one to compare simulation to execution logs</td>
<td>MOKA-050</td>
<td></td>
<td>MOKA-070 MOKA-080</td>
<td></td>
</tr>
<tr>
<td><strong>RTA-10003:</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 4: Status of the Basic Framework Requirements Roadmap**
The framework should extract timing properties from logs

<table>
<thead>
<tr>
<th>RTA-10004:</th>
<th>The framework should support root-cause analysis from execution logs</th>
<th>COMPTEST-010</th>
<th></th>
<th></th>
</tr>
</thead>
</table>

RTA-10005: The framework should be able to collect information in a integrated manner (e.g., from a distributed architecture)

<table>
<thead>
<tr>
<th>RTA-10006:</th>
<th>The framework should support online analysis of execution logs</th>
<th>CERTIFYIT-080</th>
<th>LIME-010 [done]</th>
<th>CERTIFYIT-120</th>
<th>MBEETLE-090</th>
</tr>
</thead>
</table>

RTA-10007: The framework should support offline analysis of execution logs

<table>
<thead>
<tr>
<th>RTA-10007:</th>
<th>The framework should support offline analysis of execution logs</th>
<th>CERTIFYIT-080</th>
<th>LIME-010 [done]</th>
<th>CERTIFYIT-090</th>
<th>LIME-010 [done]</th>
<th>CERTIFYIT-1000</th>
</tr>
</thead>
</table>

Figure 5: Status of the Analysis of Execution Logs Requirements Roadmap

3.3. Monitoring requirements

<table>
<thead>
<tr>
<th>Framework Features</th>
<th>Baseline</th>
<th>Initial</th>
<th>Intermediate</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTA-20001: The framework should support validation via model in the loop</td>
<td>CQDESIGN-020</td>
<td>CQDESIGN-110 RCRS-010</td>
<td></td>
<td>COMPTEST-030</td>
</tr>
<tr>
<td>RTA-20002: The framework should allow extracting and comparing timing properties to specifications</td>
<td>AIPHS-030</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1 As reported in the Deliverable D3.2 [1] (see Section 8), the Framework Requirement RTA-10005 (related to distributed architecture management) is not completed by any tool during the project. However, it will be completed within the project by integrating external tools.
tools which allows definition of desired behavior

| RTA-20005: The framework provides a monitoring tool to predict future problems | progress | LIME-030 [in progress] | progress |
| RTA-20006: The framework should provide monitoring tool which allows definition of alarms | AIPHS-010 | LIME-010 [done] | LIME-030 [in progress] |
| RTA-20008: The framework should allow profiling software on FPGA | AIPHS-010 | AIPHS-040 [in progress] |  |

Figure 6: Status of the Monitoring Requirements Roadmap

### 3.4. Code generation requirements

<table>
<thead>
<tr>
<th>Framework Features</th>
<th>Baseline</th>
<th>Initial</th>
<th>Intermediate</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTA-30001: The framework should provide code generation capabilities for different programming languages</td>
<td>MODELIO-110</td>
<td>XPM-030 [done]</td>
<td>XPM-050 RCRS-080 [planned]</td>
<td>XPM-060</td>
</tr>
<tr>
<td>RTA-30002: The framework should provide capabilities for generating code certified SIL0-SIL4</td>
<td>PAPYRUS-130 RCRS-030</td>
<td>PAPYRUS-130</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTA-30003: The framework should provide capabilities for generating monitors</td>
<td>PAPYRUS-130 PAPYRUS-150</td>
<td>LIME-010 [done]</td>
<td>LIME-030 [in progress]</td>
<td></td>
</tr>
<tr>
<td>RTA-30004: The framework should provide support for generating instrumented code</td>
<td>MODELIO-150 PAPYRUS-130</td>
<td>S3D-080 [in progress] LIME-010 [done]</td>
<td>MODELIO-130 [in progress]</td>
<td>MODELIO-140 S3D-070 PAU-020 PAU-030</td>
</tr>
<tr>
<td>RTA-30005: The framework should provide support for generating log configuration files</td>
<td>MODELIO-150 PAPYRUS-130 PAPYRUS-150</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTA-30006: The framework should provide support for the optimization of code at runtime</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 7: Status of the Code Generation Requirements Roadmap

---

2 As reported in the Deliverable D3.2 [1] (see Section 8), the Framework Requirements RTA-30002 and RTA-30006 (related to code generation), as well as RTA-30005 (related to log configuration), are not completed by any tool during the project. Therefore, RTA-30002 and RTA-30006 requirements will be addressed within the project by integrating external SIL-certified tools, while RTA-30005 will be mitigated when covering RTA-30004 issues.
3.5. Testing requirements

<table>
<thead>
<tr>
<th>Framework Features</th>
<th>Baseline</th>
<th>Initial</th>
<th>Intermediate</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTA-40001: The framework should provide support for test selection</td>
<td>CQDESIGN-020, CQDESIGN-030, CQDESIGN-040, CQDESIGN-050, CQDESIGN-070, CQDESIGN-080</td>
<td>LIME-020 [in progress]</td>
<td></td>
<td>COMTEST-030, S3D-100</td>
</tr>
<tr>
<td>RTA-40002: The framework should provide support for test generation</td>
<td>COMPTEST-010, CERTIFYIT-020, CQDESIGN-020, CQDESIGN-030, CQDESIGN-040, CQDESIGN-050, CQDESIGN-060, CQDESIGN-070, CQDESIGN-080, CQDESIGN-100</td>
<td>COMTEST-020, LIME-020 [in progress]</td>
<td>S3D-060, S3D-090, CERTIFYIT-120 [done], MBEETLE-030 [planned], MBEETLE-060 [done], MBEETLE-070 [done], MBEETLE-080 [in progress]</td>
<td>COMTEST-030, S3D-100, MBEETLE-010 [in progress], MBEETLE-020 [in progress], MBEETLE-100, MATERA2-060 [planned]</td>
</tr>
<tr>
<td>RTA-40003: The framework should provide support for test execution</td>
<td>MATERA2-070</td>
<td>LIME-020 [in progress]</td>
<td>S3D-090, CERTIFYIT-120 [done]</td>
<td></td>
</tr>
<tr>
<td>RTA-40004: The framework should provide support for test analysis, including test coverage</td>
<td>CERTIFYIT-090, CQDESIGN-030, CQDESIGN-040, CQDESIGN-050, CQDESIGN-060, CQDESIGN-070, CQDESIGN-080, CQDESIGN-090, CQDESIGN-110, MATERA2-070</td>
<td>COMPTEST-020, LIME-020 [in progress]</td>
<td>S3D-090, MBEETLE-090, MBEETLE-110</td>
<td>COMTEST-030, S3D-110, MBEETLE-020 [in progress], MBEETLE-100</td>
</tr>
<tr>
<td>RTA-40005: The framework should provide support for automatic deployments of tests</td>
<td>CQDESIGN-030, CQDESIGN-040, CQDESIGN-050</td>
<td>LIME-020 [in progress]</td>
<td></td>
<td>S3D-110</td>
</tr>
<tr>
<td>RTA-40006: The framework should provide support for load testing</td>
<td>MATERA2-080</td>
<td>LIME-020 [in progress]</td>
<td></td>
<td>COMTEST-030</td>
</tr>
<tr>
<td>RTA-40007: The system should provide support for runtime simulators</td>
<td></td>
<td></td>
<td></td>
<td>PAU-020</td>
</tr>
</tbody>
</table>

Figure 8: Status of the Testing Requirements Roadmap

4. Tool set resources

At the time of the M15 milestone the integration activity provided by WP5 tasks is in progress, and the first integration results will be available at M20. Thus, to provide a support to case study providers, a preliminary common entry point that collects all the tools has been defined. It consists of a GitHub organization\(^3\) where the main information related to each tool belonging to the MegaM@Rt2 Framework have been made available.

\(^3\) [https://github.com/megamart2](https://github.com/megamart2)
This organisation does not include tool sources or executable code, since each tools provider has already defined its own delivery environment, rather it contains a structured set of information required to access and use the tools.

The organization contains a set of repositories, one for each tool. In each repository a single README page is available, structured based on a given template reporting:

- the download and installation instructions, providing information about:
  - web site,
  - download site,
  - release notes,
  - system requirements,
  - installation/upgrading guide,
  - license information,
  - available support (FAQ, Forums, Help Desk)

- getting started guidelines
  - how to check tool functionalities,
  - examples and preconfigured exercises
  - specific instructions for using the tool for MegaM@Rt2
  - Known limitations.

- additional information
  - links to further documents like: reference guide, user guide, etc...

The next subsections introduce each tool and shortly describe the delivered material. Available material and information for download are detailed for each tool in the Appendix.

4.1. CompleteTest

CompleteTest is composed of a transformation engine translating IEC 61131-3 programs into timed automata models, a validation component, a trace generator and analysis component, and a test generator. The generated test cases can be executed and the results are checked against requirements. The installation guide is provided within the tool.

4.2. S3D

S3D is a UML/MARTE model-driven design framework. S3D supports a “single-source approach” to Electronic System-Level (ESL) design activities, which improve design productivity. For such purpose, S3D requires as input to the design methodology a complete model of the system, integrating all the information to be used by the different steps of the design process. This model relies on existing standards, namely UML and additional OMG standard UML profiles, i.e. MARTE. That way, the UML/MARTE model is used as a single, centralized repository of all the information required to perform the different design activities, such as the production of functional models, of performance models, design space exploration, etc. To about the system relevant for the design. This leverages consistency and modelling efficiency. The framework automates these ESL design activities, through additional tool components, such as eSSYN, used to generate communication and deployment codes, including run-time extensions, or VIPPE, used to generate virtual platform models for simulation and performance evaluation.
4.3.  PADRE

The PADRE tool 1.0.0 release is composed of: i) the performance antipattern detection and removal library, and ii) a validation UML model example, filled with performance values (i.e. indices). In addition we provide an extended version of EPSILON that supports the porting among EPSILON languages (EVL, EPL, and EWL). The installation guide is provided within the tool.

4.4.  CHESS

The PolarSys CHESS tool, rel. 0.10.0, is a Papyrus plug-in is composed by:

- CHESS UML profile, that is a MARTE/UML profile,
- an extension to the Papyrus UML graphical editor that supports the notion of design views,
- a model validator that assesses the well-formedness of the model before model transformations can be undertaken, and
- a set of model to model and model to text transformations, the former for the purpose of model-based schedulability and dependability analysis and the latter for code generation toward multiple language targets.

4.5.  XPM

The XtratuM Project Manager (XPM) is an Eclipse Rich Client Platform (RCP) application to automatically generate the deployment for a target partitioned system based on the XtratuM hypervisor. The main goal is to provide an efficient way to develop real-time and embedded applications with minimal intrusion on the target system. The following main features are Available in XPM 1.0.0:

- Deployment of partitioned systems based on XtratuM hypervisor.
- Different hardware targets: ARM, LEON.
- Different operating systems/environments: XAL, LithOS.

4.6.  Papyrus

Papyrus is a graphical editing tool for UML 2 diagrams as defined by OMG to enable model-based software engineering. Papyrus targets to implement 100% of the OMG UML 2.5 specification. Currently, it gives support to the following diagrams:

- Class Diagram
- Object Diagram
- Package Diagram
- Composite Structure Diagram
- Component Diagram
- Deployment Diagram
- Profile Diagram
- Use case Diagram
- Activity Diagram
- State machine Diagram
- Communication Diagram
- Sequence Diagram
- Timing Diagram
- Interaction overview Diagram

---

4  [https://www.omg.org/spec/UML/2.5/About-UML/](https://www.omg.org/spec/UML/2.5/About-UML/)
Papyrus provides also a complete support to SysML 1.1 and 1.4 in order to enable model-based systems engineering. Specific tabular and graphical editors required for SysML are also provided:

- Block Definition Diagram
- Internal Block Diagram
- Requirement Diagram
- Parametric Diagram
- Requirement table
- Allocation table

The features (purposes) integrates in this first version of the MegaM@Rt2 Framework are all baseline purposes which are already provided by Papyrus 3.4:

- Give support to 100% UML standard language, including all diagrams
- Leverage UML extension mechanisms to create DSLs
- Customize UML editors, views, perspectives and model explorer based on UML stereotypes
- Give support non-functional aspects modeling
- Give support to modeling of real-time systems, including rich time and resource modeling and allocation
- Give support to system engineering
- Integration with Git control version system
- Model compare and merge for version control
- Direct code generation for a specific programming language (Java, C++ and C)
- Create custom code generators for component interaction and communication
- Give support to annotation-based modeling of extra-functional properties (e.g., scheduling and performance).

4.7. AIPHS

The AIPHS tool 1.0.0 release is composed of:

- Distributed hardware monitoring system for target on FPGAs, written in VHDL, that can work on Leon3 scenario but that can be adapted also to other soft-core based scenarios
- Runtime generation of logs for WCET analysis, in particular logs composed of ID and timestamp
- Runtime generation of logs for performance measurements on targets with multi-core processors, running bare-metal and Linux based applications, on a Leon3 based scenario but that can be adapted also to other soft-core based scenarios.

The installation guide is provided within the tool.

4.8. JTL

The JTL tool release is provided as an Eclipse Product or a set of Eclipse plugins that can be installed separately. JTL allows to design and manipulate models within EMF, maintain consistency and synchronize software artifacts, keep traceability during design. Its constraint-based and relational model transformation engine is specifically tailored to support bidirectional model transformations, change propagation and traceability.

4.9. PauWare

PauWare is an open source Java tool (current version: 1.3). It enables to program an UML state machine in plain Java, to associate business operations with it and to execute it. It is distributed under
the form of a Java archive file (.jar) that can be included in any Java program (an adaptation for Android has been released). This file provides two components:

- An API to program a state machine: definition of states, transitions, guards, business operations...
- An execution engine that triggers events and make the state machine evolving and executes the required business operations associated with states and transitions

PauWare viewer is an associated prototype for Pauware. It draws in a Web browser the state machine executed by the PauWare engine. It is provided under the form of a Netbeans Java project.

4.10. CMA

Completeness Metric Analyzer (CMA) is a graphical tool that, used in conjunction with DODT, allow users to define, control, check and measure requirements defined in Safety Critical systems. The following main features are available:

- Grouped requirements and relative percentages
- Requirement graphs tracing tool
- Boilerplate combinations for requirements management
- List editor to change items and attributes of related boilerplate
- Ontology editor tab with concept details and import option

Because DODT sources are not available still, it has been decided to analyze other similar tool for Requirements Analysis and specifications, (e.g., SysML Papyrus plugin), so to evaluate and possibly to tailor them into specific use case needs.

4.11. Certifylt

The Certifylt tooling release is composed of the Eclipse-based modeller plug-in for IBM Rational Software Architect (RSA) and the Certifylt core software required installation files. The Eclipse-based UML modeller, especially well-integrated into the RSA platform, allows users to build and edit UML/OCL test model that can be exported into the core software of the Certifylt tool. This one is a model-based testing solution enable to derive executable test cases from those models.

To ease and help engineers to use the release, a user guide is provided and a tutorial shows the certifylt offline testing usage (it will be completed and extended for the next milestone delivery with online testing issues).

4.12. MBeetle

The MBeetle tooling release is composed of a ZIP archive containing the full application, which is completely compliant with the delivered version of the Certifylt tooling release. This version constitutes the initial version of the MegaM@RT online testing approach. A basic algorithm to derive test cases from Certifylt test models and capabilities to execute them at runtime on the system under test are implemented. At present, no dedicated user guide or tutorial is provided to ease and help engineers to use the solution. Such a support will be developed and delivered for the next milestone, including new online test generation strategies to improve the relevance of the test scenarios. The development of these features will be mostly driven according to first user feedback and specific needs appeared during experimentation currently conducted by the project use-case providers.
4.13. Conformiq Designer

Conformiq Designer enables automatic generation of functional box tests from system models. Combining best-of-breed mathematical algorithms with an Eclipse-based IDE for Automated Test Design, Conformiq Designer reduces the risk of missed tests by enabling companies to test for difficult and complex system scenarios.

Conformiq Designer employs client-server architecture in which the client user interface is implemented as an Eclipse plug-in. The server component — Conformiq Computation Server — can be installed on the same computer as Conformiq Eclipse Client or on another node on the local area network.

Conformiq Designer Eclipse Client (QEC) is provided as:
- A standalone software as a rich client application that contains a minimal set of plug-ins collectively known as Rich Client Platform (RCP) as well as the required Java Runtime Environment (JRE).
- An Eclipse plugin that requires an existing Eclipse installation.

Conformiq Designer RCP application and Conformiq Eclipse plugin versions are provided in two distinct installers.

Conformiq Designer comes with a user manual detailed the installation of the product, how to use the product, how to model using Conformiq Designer, how to integrate with other 3rd party test management, test execution and requirement management tools, and more.

Quick introductory video on Conformiq Designer can be found on Conformiq’s YouTube channel (https://www.youtube.com/watch?v=GEemdGj-Ys) with a collection of rather short tutorial style videos (https://www.youtube.com/playlist?list=PLqWF7jabwGA_rbtXuktmCuWXmxh_jOLA).


Convex Hull is a modular tool for analyzing software. Its main functionality is to catch defects in working software as soon as possible. It is composed of a similarity calculator and a similarity analyzer. The similarity calculator defines and measures the similarities and dissimilarities in the software. The similarity analyzer groups the software based on the dissimilarities and finds the source for the deviancy. Convex hull comes with several default modules of similarity calculators and analyzers and a possibility for the user to define her own modules. The user guide will help the user to select the most suitable modules for the analysis.

4.15. Clusterability

The Clusterability evaluation tool release is a Python application for evaluating the clustering structure of a dataset. The clustering structure varies among different datasets and the application can recognize if a given dataset contains enough clusterable structure so that it can clustered in a meaningful way. If a dataset is recognized to contain enough clusterable structure, a clustering is performed. The number of clusters in a dataset is chosen based on the stability of the clusters. User can also provide the number of clusters as an input and the application determines if the dataset can be clustered in a meaningful way so that the output is the user defined amount of clusters.
4.16. Moka

Moka is an open source tool which extends Papyrus (and therefore, Eclipse) to execute UML models using a rich and extensible animation and simulation framework. Apart from UML modeling capabilities to specify the executable behavior of model elements, Moka provides textual notation edition with syntax highlight, completion and content assist for Alf (Action Language for Foundational UML - fUML). The underlying model execution engine implements Foundational UML (fUML), a OMG standard that allows to specify precise (detailed) operational and base semantics for metamodels, in particular, that of UML.

Moka deploys and operate within Eclipse as a customizable feature extending Papyrus with modeling and operational capabilities to run and debug UML models. Therefore, Moka must be understood as module for Papyrus, which is developed under the Eclipse Papyrus project\(^5\) and hence, the PolarSys\(^6\) is an Eclipse Industry Working Group.

The features (purposes) integrated in this first version of the MegaM@Rt2 Framework are all baseline purposes which are already provided by Moka 2.0.0:

- fUML and Alf Modeling:
  - Steady and precise semantic foundations for UML and related standards, such as SysML and MARTE
  - Execution engine complying with OMG’s Foundational UML (fUML) standard for model animation and simulation
  - Specifying the behaviour of executable models by means of an action language: OMG’s Action Language for fUML (Alf)
  - Textual notation edition with syntax highlighting, completion and content assistance
- Model execution (run and debugging).

4.17. MATERA2

MATERA2 provides a model-based monitoring and online testing tool called MBMÅA which monitors or generates test online. The MBMÅA tool uses UPPAAL timed automata for creating the behavioral model of the system under test. The UPPAAL model can be edited using a graphical user interface. The test adapter for MBMÅA tool can be specified using Jython. The model can be simulated and verified using UPPAAL with respect to the requirements of the system. MBMÅA can analyze execution traces specified in textual format.

For model-based performance and load testing, MATERA2 provides the MBPeT tool. MBPeT uses the Markov chain models for specifying the user behavior, which can be specified via the Graphviz language. The test adapter can be specified using Python or proprietary XML. MBPeT provides a graphical user interface for editing the test configuration and parameters.

Both MBPeT and MBMÅA also provide command line interfaces.

4.18. Modelio

Modelio is an open-source modelling environment supporting industry standards like UML and BPMN. Modelio provides a central repository for the local model, which allows various languages (UML2 profiles such as SysML and MARTE) to be combined in the same model, enabling abstraction layers

\(^5\) https://www.eclipse.org/papyrus/
\(^6\) https://www.polarsys.org/
to be managed and traceability between different model elements to be established. Modelio proposes various extension modules and can be used as a platform for building new Model-Driven Engineering (MDE) features such as code generation and reverse engineering of Java and C++. The environment enables users to build UML2 Profiles, and to combine them with a rich graphical interface for dedicated diagrams, model element property editors and action command controls.

In the frame of the MegaM@Rt2 project, SOFTEAM reinforces the current generators and profiles such as SysML, MARTE, and UTP as well as develops the new ones. In particular, the DDS Designer is developed to cover needs of the case study by IKERLAN. In addition, the support for variability modelling is currently under construction with a possibility to manage product variants involving generation.

4.19. VeriATL

VeriATL allows user to specify several kinds of functional requirements of ATL model transformation developers, e.g. requirements of syntactic and semantic correctness, and statically verify them against developed ATL model transformation in a sound manner. On verification failure, it provides debugging clues to help the developer identify the bug. On changes of model transformation, it identifies and incrementally verifies the requirements affected by the changes.

4.20. LIME Testbench

The LIME Testbench (SSF) is a testing and runtime monitoring tool developed in the project LightweIght formal Methods for distributed component-based Embedded systems (LIME), and it's continuation project LIME2. The LIME Testbench implements two separate functions: runtime monitoring of model-based properties, and test case generation. Both functions rely on the instrumentation and monitoring of the execution of the program under test. Instrumenting the program under test allows to monitor and record all function calls, memory access, and register operations during execution without additional effort from the programmer. This in turn allows to add assertions to the program to monitor adherence to properties that can be specified with regular expressions, temporal logic, or finite state machines. It is also possible to automatically generate high coverage test suites by using a constraint solver to produce inputs that force the program to take certain branches during execution. The above features work for Java, but there is also a C version that supports the generation of test cases.

4.21. RCRS Toolset

RCRS Toolset (the Refinement Calculus of Reactive Systems) is a compositional reasoning framework. RCRS can model open (input-output), non-deterministic, and non-input-receptive systems. Components can be specified as symbolic transition systems or as temporal logic (LTL) formulas. RCRS supports checking compatibility of components during composition. RCRS supports refinement, which allows to reason about component substitutability. The RCRS Toolset comes with a Translator of Simulink Hierarchical Block Diagrams into Isabelle theories.
5. **Relationship with other work packages**

This section gives an overview of the dependencies and relationships of the WP3 activities with the other WPs. It mainly concerns relationships with WP1 (related to the framework architecture definition and the use case scenarios implementation), WP2 and WP4 (that complete the framework capabilities), and finally WP5 (addressing the integration of the whole set of tools in the MegaM@Rt2 Framework).

5.1. **WP1, Architecture and Use Case Scenarios**

The WP1 focuses on both: i) case studies and scenarios definition by the case study providers; and ii) the architecture specification by the technology providers.

The case study providers started by identifying their needs (i.e. the case studies requirements, CSR) and defining the scenarios to be used to validate the MegaM@Rt2 Framework. From CSRs, WP3 derived the runtime analysis framework requirements, identifying the match between the tools purposes and CSRs themselves, and defining the roadmap for the framework implementation, as reported in the Deliverable D3.2 [1].

In addition, the case study providers, with the support of the tool providers, set up the baseline experiments evaluating and applying some of the proposed tools. Preliminary feedbacks are already helping the tool providers to identify issues and weakness of their tools and to refine and improve the tool functionalities. The case study providers' activities and the relevant scenarios are reported in Deliverables D1.1 [2] and D1.3 [3]. In particular, Deliverable D1.3 provides several details, for each scenario, about the chosen tools and the relevant selection criteria.

As stated in the FPP, it is important to mention that the advantages in using the MegaM@Rt2 runtime analysis framework, among the MegaM@Rt2 Framework as a whole, must be measured by means of the case study scenarios, evaluating the following performances:

- **KPI 2.1:** Reduction of validation effort in the range of 10%-30% by automated trace collection and analysis.
- **KPI 2.2:** Reduction of time for monitors setup by automated probes injection in the range of 50%-80%.

Details are reported in the Deliverable D1.3 [3].

In addition, the tool providers have elaborated and are continuously refining the details of the global framework architecture that defines:

- the methods and basic criteria to use and apply the participating tools during a system lifecycle,
- the issues and the available supports to integrate the set of tools in a consistent framework.

Finally, the design of the framework architecture has been provided using the Modelio tool that supports, among other things:

- a collaborative environment able to manage the complexity of the consortium structure,
- a configuration control environment to coordinate multiple and asynchronous interactions and
- an extended traceability features to check the specifications consistency.

The architectural results are reported in Deliverables D1.2 [4] and D1.4 [6].

It is important to note that all the modelling activities, developed under Modelio, affect and are provided in the context of all the WPs, notably in WP2, WP3, and WP4 that represent and implement the core of the framework structure and involve all the WP leaders and tool providers. The next subsection introduces details about this active and collaborative aspect.
5.2. WP2, WP3 and WP4 harmonisation

The coordination between all the project WP, especially the technical ones, i.e. WP2, WP3 and WP4, has been constant during the project evolution. The methodological approach followed by the project is an additional element that drives this active cooperation:

- all the three technical WPs are partitioned in a coherent set of tasks;
- they represent the core of the MegaM@Rt2 Framework;
- the timelines are synchronized, and the deliverable contents are aligned;
- the use of an engineering support (i.e. Modelio) further simplifies the harmonization;
- the tool providers need to operate consistently on their tools, independently from the possible WPs they are related to.

This collaborative management notably allows to maintain and ensure a high level of consistency regarding the outputs of the project while enabling efficient approaches to share and disseminate results. For example, one of the more visible effects of the WPs harmonization, concerns the common structure of the deliverables provided until now, including this Dx.3 series. The objective is to significantly increase the readability of the produced deliverables and simplify their content review and comparison.

5.3. WP5 and Integration support

The WP5 aims to define the approach to integrate the MegaM@Rt2 Framework and the tool set identified by WP2, WP3 and WP4. Its time plan is progressing with a slight delay with respect to the WP2, WP3 and WP4, since it requires a preliminary definition of framework architecture details and tool set consolidation. The first report on the integration approach is scheduled for M20 and the initial version of the integrated framework is expected for M22, and will be respectively reported in Deliverables D5.1 [7] and D5.2 [8].

The basics of the integration approach will be based on:
- the interfaces provided and/or used by the tool set,
- the information that are flowing through such interfaces,
- the deployment platforms.

All these elements have been defined and modeled in Modelio, as part of the WP2, WP3 and WP4 tasks, and reported in Deliverables. The current version is reported in Deliverable D1.4 [6].

Additional strategies and implementation structures have to be defined to support framework flexibility and adaptability to different customers’ needs and industrials procedures. At the moment, two basic solutions have been provided, in order to support the case study providers in the baseline experiments and evaluation:
- a communication channel between case studies providers and tools providers based on “slack”,
- a repository that collects the basic information about the whole set of the proposed tools on GitHub.

Slack is a collaboration tool that allows working groups to setup “channels” to exchange messages and information in a chat-like form. In MegaM@Rt2, it can be used as a basis for tools support and to collect CSP feedbacks, i.e. setting up a channel for each case study and/or tool. Slack can be integrated with other application, specifically with GitHub.

GitHub is a well-known environment based on Git version control system. In this preliminary release at M15, it will be used as a repository to collect all the necessary tools information. At the moment, it is not intended to store tool source code or executable files, since each tool provider has already set up a predefined delivery environment for its own tools. Rather, it will collect in a “README” file, for each tool, the relevant information necessary to allow case studies providers to select and experiment the tools themselves. The README file template and the foreseen set of necessary information are described in the previous Section 4, while the Appendix of the present document collects the README file content for the Runtime Analysis Tool Set.
6. Conclusions

This document has introduced the runtime analysis tool set component of the initial version of the MegaM@Rt2 Framework, which constitutes the Deliverable D3.3 of the MegaM@Rt2 project. It provides an overview of the major achievements, features and work done at the “initial” milestone at M15. It also reports information and guidelines to describe the way to use them in an efficient way. The status is presented in two different fashions:

- focusing on tools evolution, in order to provide the detail picture of the development activities,
- referring to the roadmap tables to give a global view of the MegaM@Rt2 Framework evolution.

While all the relevant tables are automatically generated from Modelio, starting from the same information, the two different perspectives allow analysing different issues and impact to the project and the planned activities. By the way, within this management, it should be underlined that Modelio allows to easily update and control the requirements and the architecture evolution, as well as to support reporting from the model artefacts.

A specific section, dedicated to the presentation of the interactions between the different WPs of the project, highlights the efficient collaboration and coordination between partners. To address this issue, a preliminary tools integration step took place with the creation of a repository under the https://github.com/megamart2 address. The repository collects the download links, references and user guides to allow case study providers to get all MegaM@Rt2 tools they need to setup their scenarios and development environments. In addition to the repository, a communication structure, based on the Slack platform, is also available to support interaction and assistance. It should be noted that the benefits resulting from such coordination are confirmed by the current experience and the globally positive results obtained so far.

As a result, the first version of the MegaM@Rt2 Framework, called initial version, dedicated to runtime analysis approaches is composed of 21 tools that implement all expected and required results at M15, as scheduled in the roadmap introduced in the previous Deliverable D3.2 [1], except two purposes that are still in progress. Therefore, at M15, no critical deviation occurs according to the initial proposed timeline, and thus no recovery action is needed.

This released version defines the preliminary integrated framework deployed to address the project case study. Within WP5, this version, and future developments as well, will be intensively experimented (some experiments are already in progress) in a continuous way using the project case studies to allow a quick feedback about the proposed runtime analysis toolsed strategies according to the requirements and evaluation plan established by the end users within WP1. The current technology will also be the starting point for future developments aiming to implement and to integrate the next results expected within the next version of the framework, which will be delivered at the "intermediate” milestone at M20.
References


[7] MegaM@Rt2, "D5.1: MegaM@Rt2 Integration Approach," To be delivered in November 2018. [Online].


[14] MegaM@Rt2, “D4.3: MegaM@Rt2 Model Management & Traceability tool set - initial version,” To be delivered in November 2018. [Online].
Appendix: tool set details

A.1 CompleteTest

The tool is suitable for transforming an FBD program to a formal representation of both its functional and timing behavior. This is done by implementing an automatic model–to–model transformation from FBDs to timed automata. This is used for runtime verification and testing of FBD programs.

WEBSITE
http://www.completetest.org

DOWNLOAD
CompleteTest is an academic tool and it is currently in an early beta version:
http://www.completetest.org/download/
You can always grab the latest version here and use it freely, but only as part of your academic work. Once you have downloaded both the CompleteTest and Uppaal, extract the completetest.zip archive and place verifyta.exe from Uppaal bin-Win32 folder to verifyta\bin-Win32 folder of CompleteTest.
After you have placed verifyta.exe to the correct folder, run the tool either by double-clicking on CompleteTest.jar or from a command line by typing: java -jar CompleteTest.jar

INSTALLATION
Setup Requirements:
This tool is developed in Java and it requires java version 1.7 to be present on the system. You can always check the version of java you have installed from a command line by typing: java –version.
In addition, this tool is using Uppaal model checker, which has to be downloaded separately from www.uppaal.org.

Obtaining the binaries
Once you have downloaded both the CompleteTest and Uppaal, extract the completetest.zip archive and place verifyta.exe from Uppaal bin-Win32 folder to verifyta\bin-Win32 folder of CompleteTest.
After you have placed verifyta.exe to the correct folder, run the tool either by double-clicking on CompleteTest.jar or from a command line by typing: java -jar CompleteTest.jar

Using the GUI

![Figure 9: CompleteTest GUI](image-url)
1. **Step & Time**
   Step & Time columns are used to present to the user the number of test vectors needed for achieving the maximum coverage. Steps represent a cycle scan in FBDs while Time represent an external clock.

2. **Input values**
   Columns in this area are showing the values of input variables to a given FBD. These values are automatically generated by CompleteTest.

3. **Output values**
   Columns in this area by default showing false for Boolean variables and 0 for Numeric values. It is expected that a user will manually change those values to fit the expected behaviour of the system, based on the input values provided in the previous area.

4. **Achieved coverage**
   In this field CompleteTest is displaying the percentage of coverage items that are covered by the generated test inputs. It is important to note that the tool always provide the maximum achievable coverage value. This means that if we have a value of 80% this is the maximum that could be covered using the selected logic coverage. Most likely this is a sign of the dead code.

5. **Diagnostic information**
   In this field, a user is presented with diagnostic information regarding the state space that was explored and memory consumption that was consumed during the model-checking execution.

6. **Validate Test Outputs**
   Once a user has provided expected values, it is possible to click on Validate Test Outputs in order to compare expected and the actual output vector for each test input vector.

**SYSTEM REQUIREMENTS**
This tool is developed in Java and it requires java version 1.7 to be present on the system. You can always check the version of java you have installed from a command line by typing: java -version
We suggest to have a look at examples located in the samples folder.

**LICENSE**
CompleteTest can be used for academic use only, the same as the underlying engine Uppaal. Any other use requires a license of Uppaal: “As academic use, we consider only work performed by researchers or students at institutions delivering academic degrees.”

A.2 S3D

**WEBSITE**
umlmarte.teisa.unican.es

**DOWNLOAD**
umlmarte.teisa.unican.es

**INSTALLATION**
S3D is a framework, requiring the installation of different modules depending on the activities to be performed. The main (core) module is based on eclipse, and it can be directly downloaded from the S3D download web page, or installed as an Eclipse component both in precompiled or source form, following the next steps:

2. Open Eclipse and install Papyrus and Acceleo from “Help” Menu -> “Modeling Components”

3. Install “Papyrus MARTE profile” from “Help” Menu -> “Install Additional Papyrus Components”

4. Download and install the S3D plugin from the S3D download site.

Apart from the S3D core component, VIPPE and Essyn components are also required form MegaM@Rt2 activities. In order to install both components, it is required to download them from the S3D downloading page and execute the corresponding makefiles.

**SYSTEM REQUIREMENTS**

- Linux OS
- Gcc compiler >= 4.8
- Libxml2
- Qt4
- Eclipse Neon Modelling Tools with the following plugins:
  - Acceleo
  - Papyrus
  - Papyrus MARTE profile
- LLVM and SystemC are installed automatically

**RELEASE NOTES**

The latest Release is 1.0.0. This release extends previous CONTREP tool, and it provides new modelling capabilities to support complex communication semantics among components, specially oriented to handle different models of computation, and improvements in the management of the environment and hierarchy.

**LICENSE**

S3D is provided under a dual license. The components can be freely downloaded from the web page as binary or source code under a non-commercial, non-distributable license. Additionally, it can be licensed ad hoc, for further use. More details can be found in the downloading web page.

**SUPPORT**

We currently provide support through email: posadash@teisa.unican.es

**DOCUMENTATION**

Documents describing the S3D modeling methodology and a user guide can be found in the S3D website.

**EXAMPLES**

Several examples are provided for downloading, based on a client-server system, showing how to model different mappings and different communications mechanisms with the S3D methodology.

**Research publications:**


A.3 PADRE

The PADRE tool provides a performance antipattern detection and removal library. In addition, we use an extended version of EPSILON that supports the porting among EPSILON languages (EVL, EPL, and EWL).

Our framework deals with UML models made of the following diagrams: a component Diagram that describes the software components and their Interfaces/Operations; a Deployment Diagram that describes the allocation of artifacts, corresponding to components, on platform nodes; a Use Case Diagram that describes the actors and the use cases that they can execute; a number of Sequence Diagrams, one for each use case, that describe the system behavior in terms of interactions among components.

DOWNLOAD

Official Github repository: https://github.com/SEALABQualityGroup/padre

INSTALLATION

2. Installing from its Update Site (http://download.eclipse.org/epsilon/1.3/updates/) in Eclipse the following EPSILON plugins only:
   a. Core
   b. development tools
   c. EMF integration
   d. GMF integration
   e. UML integration
3. Cloning PADRE from https://github.com/SEALABQualityGroup/padre.git
4. Import in eclipse PADRE projects
5. Create a running configuration in:
   a. EVL
      i. Select an EVL file (e.g.
      padre/it.spe.disim.epsilon.antipattern_d-s/evl/AP-UML-MARTE.evl)
   ii. Give a name (e.g. BGCS)
   iii. Select a model from workspace (e.g.
        padre/it.spe.disim.epsilon.antipattern_d-s/model/uml/BGCS/BGCS.uml)
   iv. Select only this metamodels:
       1. UML
       2. UML standard profile
       3. Ecore
   b. EPL
      i. Select an EPL file (e.g.
      padre/it.spe.disim.epsilon.antipattern_d-s/epl/AP-UML-MARTE.epl)
ii. Give a name (e.g. BGCS)

iii. Select a model from workspace (e.g. padre/it.spe.disim.epsilon.antipattern_d-s/model/uml/BGCS/BGCS.uml)

iv. Select only this metamodels:
   1. UML
   2. UML standar profile
   3. Ecore

c. EWL
   i. Add new EMF/GMF wizard (from Eclipse preferences -> epsilon -> EMF/GMF wizards)
   ii. Select an EWL files from workspace (e.g. padre/it.spe.disim.epsilon.antipattern_d-s/ewl/AP-UML-MARTE.ewl)
   iii. Select the UML namespace uri (http://www.eclipse.org/uml2/5.0.0/UML)

   6. enjoy :-)

SYSTEM REQUIREMENTS

- MacOS Sierra (10.12.06);
- Eclipse Oxygen Modelling Tools with the following plugins:
  o Epsilon 1.3
  o Papyrus
  - Papyrus MARTE profile (downloaded from http://download.eclipse.org/modeling/mdt/papyrus/updates/releases/neon)

RELEASE NOTES

The latest Release is: 1.0.0.
This release provides several major features and supports several performance antipattern detection and removal (Blob, Pipe and Filter, Extensive Processing, Concurrent Processing Systems, Empty Semi Truck, Tower of Babel).

LICENSE
Eclipse Public License - v 1.0 (see http://www.eclipse.org/org/documents/epl-v10.php)

DEVELOPER RESOURCES
Source Repositories: https://github.com/SEALABQualityGroup/padre.git
- ssh: git@github.com:SEALABQualityGroup/padre.git
- https: https://github.com/SEALABQualityGroup/padre.git

You can use the code from these repositories to experiment, test, build, and create patches, issue pull requests.

SUPPORT
We currently support:
- email: daniele.dipompeo@graduate.univaq.it (please take care to use [PADRE SUPPORT] as object
- issues on github.com

EXAMPLES
We provide an example UML project that is a representation of a Botanical Garden Control System (BGCS). It is stored in padre/it.spe.disim.epsilon.antipattern_d-s/model/uml/BGCS/BGCS.uml. It is already filled with performance indices, and it is ready to use for a dummy execution.
Research publications:


A.4 CHESS

PolarSys CHESS implements the CHESS UML profile, a specialization of the Modeling and Analysis of Real-Time and Embedded Systems (MARTE) profile, by producing extensions to Papyrus that provide component-based engineering methodology and tool support for the development of high-integrity embedded systems in different domains like satellite on board systems. The CHESS tool environment is composed by:

1. a MARTE/UML profile,
2. an extension to the Papyrus UML graphical editor that supports the notion of design views,
3. a model validator that assesses the well-formedness of the model before model transformations can be undertaken, and
4. a set of model to model and model to text transformations, the former for the purpose of model-based schedulability and dependability analysis and the latter for code generation toward multiple language targets.

WEBSITE
https://www.polarsys.org/chess

DOWNLOAD
https://www.polarsys.org/chess/download.html

INSTALLATION

The PolarSys CHESS Eclipse full distribution can be installed simply uncompressing it. The Polarsys CHESS is an Eclipse plug-in that can be installed using the standard update-site feature; detailed instructions are available on the website.

SYSTEM REQUIREMENTS

- Windows 7 x86 64-bit Oracle Java 7 or
- Ubuntu 14.04 x86 64-bit Open JDK 7;
- Eclipse Luna with the following plugins in place: Acceleo Core SDK, ATL SDK, QVT Operational SDK, Papyrus (update available on newer Eclipse versions)

RELEASE NOTES

The latest Release is: 0.10.0, 2016-10-31.
This release provides several major features:

- New profile for dependability
- Tool support for State Based Analysis and Failure Logic Analysis.
- Support for contract based modelling; integration with OCRA FBK tool for contract based analysis.
- New Instance View to navigate and check/create timing properties for software component instances.
● Support for multiple software to hardware deployment modeling and analysis.
● New specialized properties tabs.
● Several bugs fixed.

Older release: 0.9.0, 2015-07-30

LICENSE
Eclipse Public License - v 1.0 (see http://www.eclipse.org/org/documents/epl-v10.php)

DEVELOPER RESOURCES
● Source Repositories: http://git.polarsys.org/c/chess/chess.git

You can use the code from these repositories to experiment, test, build, and create patches, issue pull requests, etc. (Review With Gerrit Browse Repository)

SUPPORT
● Wiki: https://wiki.polarsys.org/CHESS
● Mailing list subscription: https://dev.polarsys.org/mailman/listinfo/chess-dev
● Getting started: https://www.polarsys.org/chess/start.html

EXAMPLES
● Producer-Consumer: http://download.polarsys.org/chess/models/ProtectedOperationSample.zip
● IndustrialDrive: http://download.polarsys.org/chess/models/IndustrialDrive.zip
● Wheel Braking System (example about contract-based design): http://download.polarsys.org/chess/models/WBS.zip

ADDITIONAL INFORMATION
● CHESS-ML Profile: https://www.polarsys.org/chess/publis/CHESSMLprofile.pdf

A.5 XPM
INSTALLATION
In order to build the executable file of the XPM from source code, a complete customized Eclipse environment must be set-up. Before following into the set-up, please make sure a Java JDK (1.7 or higher) is ready to be used in the development platform.

● Step 1: Download Eclipse Modelling Kepler SR2 from: http://eclipse.org/downloads/packages/eclipse-modeling-tools/keplersr2 (i.e. The file is "eclipse-modeling-kepler-SR2-macosx-cocoa-x86_64.tar.gz" if the development platform is MacOSX)

● Step 2: Install the package by uncompressing it into a folder with path does not contain blank spaces.
● Step 3: Start the installed eclipse.
**Step 4:** Update the eclipse distribution: Help -> Check for updates.

**Step 5:** If the Java release in your development platform is "Java 1.8", Eclipse Kepler SR2 must be patched in order to use it correctly, otherwise skip this step and go straight to step 6.
- Help > Install New Software… (it opens the “Available Software” dialog)
- Select Work with: [http://download.eclipse.org/eclipse/updates/4.3-P-builds/](http://download.eclipse.org/eclipse/updates/4.3-P-builds/)
- Check “Eclipse Java 8 Support (for Kepler SR2)”. Next, Next, accept the license, and click “Finish”
- Installing Software…
- Restart Eclipse

**Step 6:** Install RCP components. Most RCP components are included in the installed EMF distribution. Nevertheless we are going to check it and install additional ones.
- Help->Install new software
- Select Work with The Eclipse Project Updates [http://download.eclipse.org/eclipse/updates/4.3](http://download.eclipse.org/eclipse/updates/4.3)
- Check "Equinox Target Components"
- Next, Next
- Accept the license, Finish
- Installing Software…

**Step 7:** Install “DeltaPack” to allow the development for multiple target platforms
- Window -> Preferences
- Select "Plug-in Development" -> “Target Platform”
- Check and select the “Active Platform”
- Click Edit.
- Click Add
- Select "Software Site"
- Click Next
- Type in "Work With" [http://download.eclipse.org/eclipse/updates/4.3](http://download.eclipse.org/eclipse/updates/4.3)
- Check "Eclipse RCP Target Components"
- Check "Equinox Target Components"
- Uncheck "Include required software"
- Check "Include all environments"
- Press Finish
- Downloading…
- Press Finish
- Press OK
- Restart Eclipse

Now the eclipse environment is ready to build the XPM bundle of projects and generate XPM binary executable files for many target platforms (i.e. Windows, Linux and MacOS).

**SYSTEM REQUIREMENTS**

XPM binary distribution contains a directory for each platform, that are:
- linux.gtk.x86_64 for Linux operating systems.
- macosx.cocoa.x86_64 for OSX operating system.
- win32.win32.x86_64 for Windows operating systems.

Inside each directory the XPMAppl directory contains the eclipse executable that is the XPM executable. The minimum requirement for hardware is any desktop PC with a general purpose operating system that runs efficiently a Java Virtual Machine (JVM). No network connection is required to execute pre-built binaries, however, to build the XPM tool requires access to the internet to download eclipse packages and plugins.
Examples of minimum hardware configurations are:

- Intel, AMD, or PowerPC CPU 1.5 GHz
- 1024x768 monitor resolution
- 512 MB of RAM

As XPM is entirely written in Java, it is platform neutral. JDK (Java) must be installed and ready to run. Most operating systems come with a pre-installed Java Virtual Machine. If you don’t know whether you have such a thing, simply open a Command-Prompt and type “java -version” or “java --version”. XPM requires a JVM v1.7 or higher.

You can download Java SE 7 update 75 from: http://java.com/en/download/manual_java7.jsp

- For Windows 8 (64 bits): win32
  - Official Git repository: https://alkalir@bitbucket.org/alkalir/aiphs.git
  - win32.x86_64
  - Package jre-7u75-windows-x64.exe

- For Mac OSX 10.10.2 (Yosemite): macosx.cocoa.x86_64
  - Package jre-7u75-macosx-x64.exe

A.6 Papyrus

PolarSys\(^7\) is an Eclipse Industry Working Group integrated by large industry players and by tools providers, aiming to foster collaboration on the creation and support of an advanced, industry-ready, open source tools for the development of embedded and cyber-physical systems. One of the main initiatives of Polarsys is Papyrus, the Polarsys solution for SysML and UML modeling\(^8\), used extensively in large industrial solutions. Papyrus delivers an advanced, industry-ready, open source, model-based engineering (MBE) tool suite, and open tool platform based on the Eclipse Papyrus project\(^9\), to which contribute important researchers and/or tool providers\(^10\) such as CEA LIST and EclipseSource. Papyrus intents to provide application and system lifecycle integration through both Eclipse-based offerings and domain-specific tools contributed by partners

Papyrus is a graphical editing tool for UML 2 diagrams as defined by OMG to enable model-based software engineering. Papyrus targets to implement 100% of the OMG UML 2.5\(^11\) specification. Currently, it gives support to the following diagrams:

- Class Diagram
- Object Diagram
- Package Diagram
- Composite Structure Diagram
- Component Diagram
- Deployment Diagram
- Profile Diagram
- Use case Diagram
- Activity Diagram
- State machine Diagram
- Communication Diagram
- Sequence Diagram
- Timing Diagram
- Interaction overview Diagram

\(^7\) https://www.polarsys.org/
\(^8\) https://www.polarsys.org/solutions/
\(^9\) https://www.eclipse.org/papyrus/
\(^10\) https://projects.eclipse.org/projects/modeling.mdt.papyrus
\(^11\) https://www.omg.org/spec/UML/2.5/About-UML/
Papyrus provides also a complete support to SysML 1.1 and 1.4 in order to enable model-based systems engineering. Specific tabular and graphical editors required for SysML are also provided:

- Block Definition Diagram
- Internal Block Diagram
- Requirement Diagram
- Parametric Diagram
- Requirement table
- Allocation table

The last most stable version of Papyrus 3.4 for Eclipse 4.7 Oxygen.

The features (purposes) integrates in this first version of the MegaM@Rt2 Framework are only baseline purposed which are already provided by Papyrus 3.4.

- Give support to 100% UML standard language, including all diagrams
- Leverage UML extension mechanisms to create DSLs
- Customize UML editors, views, perspectives and model explorer based on UML stereotypes
- Give support non-functional aspects modeling
- Give support to modeling of real-time systems, including rich time and resource modeling and allocation
- Give support to system engineering
- Integration with Git control version system
- Model compare and merge for version control
- Direct code generation for a specific programming language (Java, C++ and C)
- Create custom code generators for component interaction and communication
- Give support to annotation-based modeling of extra-functional properties (e.g., scheduling and performance)

WEBSITE
https://www.eclipse.org/papyrus/
https://projects.eclipse.org/projects/modeling.mdt.papyrus

DOWNLOAD
The last most stable version of Papyrus 3.4 for Eclipse 4.7 Oxygen.
http://projects.eclipse.org/projects/modeling.mdt.papyrus/releases/3.4.0

INSTALLATION
You can either use an update site to download Papyrus related features in your Eclipse installation or as a stand-alone application (RCP).
https://www.eclipse.org/papyrus/download.html

For example, to install Papyrus in a distribution Eclipse 4.7 Oxygen, use the following update site:
http://download.eclipse.org/modeling/mdt/papyrus/updates/releases/oxygen/

SYSTEM REQUIREMENTS
- Windows 7 x86 64-bit Oracle Java 7 or
- Ubuntu 14.04 x86 64-bit Open JDK 7;
- Eclipse Oxygen with the following plugins in place: Acceleo Core SDK, ATL SDK, QVT Operational SDK, Papyrus

RELEASE NOTES
The latest release is: 3.4.0
The features (purposes) integrates in this first version of the MegaM@Rt2 Framework are only baseline purposed which are already provided by Papyrus 3.4.

- Give support to 100% UML standard language, including all diagrams
- Leverage UML extension mechanisms to create DSLs
- Customize UML editors, views, perspectives and model explorer based on UML stereotypes

12 http://projects.eclipse.org/projects/modeling.mdt.papyrus/releases/3.4.0
Give support non-functional aspects modeling
Give support to modeling of real-time systems, including rich time and resource modeling and allocation
Give support to system engineering
Integration with Git control version system
Model compare and merge for version control
Direct code generation for a specific programming language (Java, C++ and C)
Create custom code generators for component interaction and communication
Give support to annotation-based modeling of extra-functional properties (e.g., scheduling and performance).

LICENSE
- Eclipse Public License - v 1.0 (see http://www.eclipse.org/org/documents/epl-v10.php)

SUPPORT
Wiki:
- Polarsys Eclipse Working Group: https://wiki.polarsys.org/Main_Page
- Papyrus Industry Consortium: https://wiki.polarsys.org/Papyrus_IC

EXAMPLES
We recommend you follow the get started guide: https://wiki.eclipse.org/Papyrus_User_Guide
- UML Profiles:
  - MDT-UML2Tools How To Use UML Profiles http://wiki.eclipse.org/MDT-UML2Tools_How_To_Use_UML_Profiles
- Code Generation
  - C++ http://wiki.eclipse.org/Papyrus/Codegen/Cpp_description
  - Custom code generation http://wiki.eclipse.org/Papyrus/Codegen/Adding_a_New_Code_Generator

A.7 AIPHS
AIPHS, acronym of Adaptive Profiling HW Sub-system, is basically conceived to support designers on the development of On-Chip Monitoring Systems (OCMSs) able to satisfy given Monitorability Requirements, namely requirements about possibility to observe the behaviour of a system with the goal of collecting metrics (e.g. related to execution time), without inserting SW overhead. It is a flexible framework that targets SoCs implemented on Field Programmable Gate Arrays (FPGAs), or on Integrated Circuits (ICs) integrating some reconfigurable logics.

INSTALLATION
1. Copy and paste the AIPHS folder in the working directory of your VHDL project
2. Insert the component declaration in the VHDL module where the monitoring system is connected to
3. Build the whole system using an FPGA synthesis tool
**SYSTEM REQUIREMENTS**
For using Xilinx FPGAs: Xilinx ISE Design suite or Xilinx Vivado Design Suite software.
For using Intel FPGAs: Intel Quartus software.

**RELEASE NOTES**
The latest Release is: 1.0.0

**LICENSE**
Apache License, Version 2.0

**SUPPORT**
Email: giacomo.valente@univaq.it

**EXAMPLES**
There are examples related to LEON3.

A.8 JTL

**DOWNLOAD**
The JTL Eclipse Product can be downloaded from the releases page. Last as well as previous releases can be found on the same page. Release notes are attached to each release.

New releases can be used with old projects and workspaces. An update site is available and already configured in the product to perform updates without downloading again the entire package.

**INSTALLATION**
Once downloaded, the package does not need any installation procedure. It can be extracted anywhere and launched using the JTL executable.

**SYSTEM REQUIREMENTS**
JTL is available for Linux, MacOS and Windows. The only requirement is an updated Java version (at least Java 8).

Only on MacOS

If you get the error *Library not loaded: /usr/local/lib/libtldl.7.dylib*:

```
$ brew install libtool --universal
$ brew link libtool
```

**LICENSE**
JTL is distributed under the Eclipse Public License - v 2.0

**SUPPORT**
Tutorials, case studies and examples can be found at jtl.di.univaq.it. To obtain support one can create a new issue on the github repository.
**DOCUMENTATION**

The best way to getting started is by following the tutorial. Some benchmarks and case studies can be found here.

**ADDITIONAL INFORMATION**

Additional information like research publications and applications involving JTL can be found at jtl.di.univaq.it.

---

A.9 PauWare

PauWare engine enables to program and execute UML state machines in plain Java programs. PauWare viewer is a prototype drawing in a Web browser a state machine executed by the PauWare engine.

**DOWNLOAD**

PauWare JAR files and source code, examples and the manual are available on the PauWare Web site: [http://www.pauware.com/](http://www.pauware.com/) (follow the “technology” link on the top right of the page).

**INSTALLATION**

PauWare requires no particular installation. It is simply a Java archive file (.jar) to add in a Java project with any IDE.

**SYSTEM REQUIREMENTS**

PauWare is provided as is under the form of a Java Netbeans project. The compatibility with another IDE has not been checked.

**LICENSE**

PauWare is provided under the LGPL version 3 license.

**DOCUMENTATION**

A quick starting guide and simple examples are also available on this page: [http://ecariou.perso.univ-pau.fr/MegaM@RT2/pauware-presentation.html](http://ecariou.perso.univ-pau.fr/MegaM@RT2/pauware-presentation.html)

---

A.10 CMA

**DOWNLOAD**

Please contact luciano.bozzi@rotechnology.it

**INSTALLATION**

Please contact luciano.bozzi@rotechnology.it

**SUPPORT**

Please contact luciano.bozzi@rotechnology.it

**DOCUMENTATION**

Please contact luciano.bozzi@rotechnology.it
A.11 CertifyIt

**DOWNLOAD**
Download site for last release: [http://extranet.smartesting.com/users/sign_in](http://extranet.smartesting.com/users/sign_in), please contact julien.botella@smartesting.com (or fabien.peureux@smartesting.com) to obtain your credentials.

- Windows: Smartesting CertifyIt 6.4.3-68841_68842.exe
- Linux (Ubuntu): Smartesting CertifyIt 6.4.3-68841_68842.tgz

**SYSTEM REQUIREMENTS**
Windows or Linux, Java 7 or 8, RSA.

**LICENSE**
CertifyIt is a commercial tool, so please contact julien.botella@smartesting.com (or fabien.peureux@smartesting.com) to obtain a license file (machine hostname should be provided).

**SUPPORT**
Issues is available on the slack CertifyIt project

**DOCUMENTATION**
Getting started guidelines are available on the Smartesting extranet: [http://extranet.smartesting.com/users/sign_in](http://extranet.smartesting.com/users/sign_in):

- tutorial slides: DABDemo.pptx
- a demonstration project (models, execution layer, etc.): Demo.zip

A.12 MBeetle

**DOWNLOAD**
Download site for last release: [http://extranet.smartesting.com/users/sign_in](http://extranet.smartesting.com/users/sign_in), please contact julien.botella@smartesting.com (or fabien.peureux@smartesting.com) to obtain your credentials.

- Windows / Linux (Ubuntu): mbeetle.zip

**SYSTEM REQUIREMENTS**
Windows or Linux, Java 7 or 8, RSA, Smartesting CertifyIt.

**LICENSE**
No license needed at present.

**SUPPORT**
Issues is available on the slack MBeetle project

A.13 Conformiq Designer

**DOWNLOAD**
Conformiq Designer can be downloaded from Conformiq website by first creating a user account ([https://www.conformiq.com/extranet/register.php](https://www.conformiq.com/extranet/register.php))
**SYSTEM REQUIREMENTS**
Conformiq Designer can be installed on Windows (Windows 7 or newer) or Linux (a reasonable modern version). The detailed system requirements are provided in the Conformiq Designer user manual (Help > Conformiq Manual > System Requirements)

**LICENSE**
License requests can be sent to Kimmo Nupponen (kimmo.nupponen@conformiq.com)

**SUPPORT**
Support requests can be sent to Conformiq support (support@conformiq.com) or directly to Kimmo Nupponen (kimmo.nupponen@conformiq.com)

**DOCUMENTATION**
Conformiq Designer comes with a user manual that has a chapter about getting started. The user manual then goes into details on how to use the product. Quick introductory video on Conformiq Designer can be found on Conformiq’s YouTube channel available at the following address: https://www.youtube.com/watch?v=_GEemdGj-Ys with a collection of rather short tutorial style videos (https://www.youtube.com/playlist?list=PLqWF7jabwGA_rblXuktmCuDWXmxh_iOLAl)

A.14 Convex Hull

**DOWNLOAD**
To be done. Not yet available online. Please contact: timo.lintonen@vtt.fi

**SYSTEM REQUIREMENTS**
Windows or Ubuntu, Python3 interpreter for custom modules.

**LICENSE**
Convex Hull is a commercial tool. To be specified.

**SUPPORT**
For the time being, provided by the author timo.lintonen@vtt.fi

**DOCUMENTATION**
Will be provided together with the software.

A.15 Clusterability

**DOWNLOAD**
To be done. Not yet available online. Please contact: juho.jokinen@vtt.fi

**SYSTEM REQUIREMENTS**
Windows or Ubuntu, Python3 interpreter for custom modules.

**LICENSE**
The clusterability tool is a commercial tool. To be specified.
**SUPPORT**
For the time being, provided by the author juho.jokinen@vtt.fi

**DOCUMENTATION**
Will be provided together with the software.

### A.16 Moka

Moka is an open source tool which extends Papyrus (and therefore, Eclipse) to execute UML models using a rich and extensible animation and simulation framework. Apart from UML modeling capabilities to specify the executable behavior of model elements, Moka provides textual notation edition with syntax highlight, completion and content assist for Alf (Action Language for Foundational UML - fUML). The underlying model execution engine implements Foundational UML (fUML), a OMG standard that allows to specify precise (detailed) operational and base semantics for metamodels, in particular, that of UML. Moka deploys and operate within Eclipse as a customizable feature extending Papyrus with modeling and operational capabilities to run and debug UML models.

Moka is developed under the Eclipse Papyrus project\(^{13}\) and hence, its development steered by the PolarSys\(^{14}\) is an Eclipse Industry Working Group.

**WEBSITE**
https://www.eclipse.org/papyrus/
https://www.polarsys.org/

**DOWNLOAD**
The last most stable version of Moka is 2.0.0 for Papyrus 3.4 (over Eclipse 4.7 Oxigen).

**INSTALLATION**
You need to use an update site to download and install Moka features on top of a Papyrus installation. For example, to install Moka in a Papyrus in a Papyrus 3.4 (over Eclipse 4.7 Oxigen), use the following update site: [http://download.eclipse.org/modeling/mdt/papyrus/updates/releases/oxygen/](http://download.eclipse.org/modeling/mdt/papyrus/updates/releases/oxygen/)

**SYSTEM REQUIREMENTS**
- Windows 7 x86 64-bit Oracle Java 7 or
- Ubuntu 14.04 x86 64-bit Open JDK 7;
- Eclipse Oxygen with the following plugins in place: Acceleo Core SDK, ATL SDK, QVT Operational SDK, Papyrus

**RELEASE NOTES**
The latest release is: 2.0.0
The features (purposes) integrates in this first version of the MegaM@Rt2 Framework are only baseline purposed which are already provided by Moka 2.0.0.
- fUML and Alf Modeling:
  - Steady and precise semantic foundations for UML and related standards, such as SysML and MARTE
  - Execution engine complying with OMG’s Foundational UML (fUML) standard for model animation and simulation
  - Specifying the behaviour of executable models by means of an action language: OMG’s Action Language for fUML (Alf)

\(^{13}\) [https://www.eclipse.org/papyrus/](https://www.eclipse.org/papyrus/)

\(^{14}\) [https://www.polarsys.org/](https://www.polarsys.org/)
Textual notation edition with syntax highlighting, completion and content assistance

- Model execution (run and debugging).

LICENSE
- Eclipse Public License - v 1.0 (see http://www.eclipse.org/org/documents/epl-v10.php)

DEVELOPER RESOURCES

SUPPORT
Wiki:
- Polarsys Eclipse Working Group: https://wiki.polarsys.org/Main_Page
- Papyrus Industry Consortium: https://wiki.polarsys.org/Papyrus_IC

EXAMPLES
We recommend you follow the get started guide:
- Model execution: https://wiki.eclipse.org/Papyrus/UserGuide/ModelExecution

ADDITIONAL INFORMATION

A.17 MATERA2
MATERA2 consists of two components MBMAÅ and MBPET for model-based monitoring and respectively for performance testing and evaluation.

- MBMAÅ

DOWNLOAD
MBMAÅ binaries soon will be available at ÅAU website (https://gitlab.abo.fi/matera2/mbm), meanwhile one can contact the authors (i.e., junaid.iqbal@abo.fi and dragos.truscan@abo.fi) to get the tool.

SYSTEM REQUIREMENTS
The tool can be installed on Windows and Linux (a reasonable modern version). The detailed system requirements are provided in README file.

LICENSE
MBM AA can be used for academic use only. Any other use would require permissions from the authors. We consider only work performed by researchers or students at institutions delivering academic degrees, as academic use.

**SUPPORT**
For the time being, provided by the author junaid.iqbal@abo.fi

**DOCUMENTATION**
README file has a section about the usage of the tools. Please contact the authors of the tool if you have any further questions or suggestions.

- **MBPeT**

**DOWNLOAD**
MBPeT binaries soon will be available at ÅAU website (https://gitlab.abo.fi/matera2/mbpet), meanwhile the authors (i.e., tanwir.ahmad@abo.fi and dragos.truscan@abo.fi) can be contacted to get the tool.

**SYSTEM REQUIREMENTS**
The tool can be only installed on Linux (a reasonable modern version). The detailed system requirements are provided in README file.

**LICENSE**
MBPeT can be used for academic use only. Any other use would require permissions from the authors. We consider only work performed by researchers or students at institutions delivering academic degrees, as academic use.

**SUPPORT**
For the time being, provided by the author tanwir.ahmad@abo.fi

**DOCUMENTATION**
README file has a section about the basic usage of the tools. The detailed guide of the tool is available as MBPeT report on the website. Please contact the authors of the tool if you have any further questions or suggestions.

### A.18 Modelio

**DOWNLOAD**
Please follow instructions from the Modelio MegaM@Rt guide for download, license request and deployment. DDS Designer modules are delivered at the DDS Project under Modelio.org forge.

**DOCUMENTATION**
General documentations about Modelio are available at Modelio.org. Instruction about DDS Designer is available at DDS Project.
A.19  VeriATL

WEBSITE
https://github.com/veriatl/VeriATL/

SYSTEM REQUIREMENTS
VeriATL is an Eclipse-plugin. It is developed on top of Eclipse Modelling Project (Neon), and requires the following external plugin-dependencies to build / be functional:

- EMFTVM v.3.8
- Xtend v.2.10

LICENSE
Eclipse Public License - v 1.0 (see http://www.eclipse.org/org/documents/epl-v10.php)

DOCUMENTATION
Demo on Youtube: https://www.youtube.com/watch?v=zFqbcK4jd9I
Verifying hierarchical state machine to flatten state machine example:

ADDITIONAL INFORMATION
Developer resources are available at: https://github.com/veriatl/VeriATL.git (Clone)
There are currently 4 major branches to track (a branches merging and download site are underway.):

- Master branch: developed for basic functional requirement verification
- FaultLoc branch: developed for basic functional requirement verification + fault localization
- Scalability branch: developed for basic functional requirement verification + fault localization + scalable verification
- Experiment branch: developed for basic functional requirement verification + incremental verification

Publication is also available:

A.20  LIME Testbench

DOWNLOAD
LIME Testbench is available for download from https://megamart.ssf.fi/lime/

INSTALLATION
Installation and usage instructions are available in the download archive in the file README.txt. More detailed documentation about different subcomponents is available in the directory doc in the same download archive.
SYSTEM REQUIREMENTS
The tool has been tested and developed on Linux Ubuntu 18.04, and its main dependency is Java 8
(OpenJDK). Other packages required are gcc, g++, clang, llvm-dev, xutils-dev, doxygen, and make.

LICENSE
The tool is released under GNU General Public License.

A.21 RCRS Toolset

DOWNLOAD
RCRS Toolset is available from https://megamart.ssf.fi/rcrs/

INSTALLATION
Installation and usage instructions are available in the download archive in the file README.txt. More
detailed documentation about different subcomponents is available in the directory doc in the same
download archive.

SYSTEM REQUIREMENTS
The tool uses Python 2.7 and Isabelle theorem prover. Optionally Simulink is need for creating and
editing models. The tool can be used in Windows, Linux, and OS X. Instructions are available on the
web page of the tool.

LICENSE
The tool is released under MIT License.