Aurora

Technology Readiness Assessment Report

D6.3

Document Code :	AUR-UPM-RP-0015
Document Version :	v2.1
Document Date :	12/06/2023
Internal Reference:	DOC00318477

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101004291

Signature Control

Written	Checked	Approved Configuration Management	Approved Quality Assurance	Approved Project Management
M.A. de Miguel Ángel G. Pérez-Muñoz Javier Gómez del Pulgar	Juan Zamorano Antonin Lyko Vaclav Fišer Ana Rodríguez	Rosa Maria Leon	Alfonso López	Ana Rodríguez
Date and Signature	Date and Signature	Date and Signature	Date and Signature	Date and Signature

Signature not needed if electronically approved by route

Changes Record

Rev	Date	Author	Affected section	Changes
v1.1	2022-12-21	M.A. de Miguel	All	Initial issue
v1.1	2023-02-03	Ángel G. Pérez	All	Add UPMSat-2 KPI data + TRL reference values. Add ESA TRL for software development Minor changes to UPMSat-2 description
v1.2	2023-02-09	Ángel G. Pérez	All	Update document structure. Discuss different TRL from ESA, H2020 and NASA.
v1.3	2023-02-13	Ángel G. Pérez	All	Include preliminary TRA report. Include Annex A with ESA and NASA TRLs for software tools.
v1.4	2023-02-15	Ángel G. Pérez	6 & 7	Update TRL levels. They have been analysed in UPMSat-2 and EUCLID independently and together for the deduction of the overall TRL.
v1.5	2023-02-24	Javier G. del Pulgar	5.2	Updated conclusions and section with Euclid AOCS models
v1.6	2023-03-03	A.I Rodríguez	--	Final version after internal review
v2.0	2023-06-02	Ángel G. Pérez A.I Rodríguez	4 6 7	Details provided on the phases of the process definition and the Assessment Strategy including a section to clarify the adopted TRL Calculator. Updated Evaluation Report and conclusions.
V2.1	2023-06-12	Miguel A. de Miguel	All	Final version after final review

Table of Contents

1. Introduction

1.1. Purpose

This document contains the Technical Readiness Assessment (TRA) Report to determine the QGen's technology maturity and provides justification of the obtained TRL which is based on the evidence gathered throughout the. "D6.2 Evidence for the assessment report".

The Report describes the process adopted to conduct the TRA and detail whether QGen has reached the targeted Technology Readiness Levels (TRL), namely 6 or 7. The resulting QGen's TRL has a significant impact on the TRL assessment for the whole AURORA tool suite. Consequently, a higher TRL will enhance the insertion of the tool in future projects with lower technical risks.

The TRLs adopted in this document conform with the H2020's definition: "WP General Annexes - Extract from Part 19 - Commission Decision C (2014)4995 G. Technology readiness levels (TRL)". Although these are the official definitions to be used in H2020 projects, we have also considered ESA's and ECSS's TRLs definitions since they take into consideration readiness levels for software elements and tools, hence, they allow a better understanding and provide further clarifications for the evaluation of QGen. Both definitions are based on the ISO standard "16290 Space systems – Definition of the Technology Readiness Levels (TRLs) and their criteria assessment".

In the AURORA GA (Grant Agreement) and this document, it is stated that the current QGen TRL is TRL-4, i.e.: it is a "technology validated in laboratory". Such assumption is inferred from the following facts:

- QGen is an open-source code generation and model verification toolset that grew out of the European projects Project-P (http://www.open-do.org/projects/p), Hi-MoCo and Gene-Auto.
- In 2021, the QGen code generator is being qualified at Tool Qualification Level 1 (TQL-1), which is the highest level of qualification recognized by the FAA (Federal Aviation Administration) and DO-178C standard.

The technology evaluators for the TRA assessment are the Simulink models from two projects, the UPMSat-2's Attitude Control System (ACS) and EUCLID's Attitude and Orbit Control System (AOCS). QGen has been exercised to generate and validate the code generated from those Simulink models. A quantitative comparison, based on a set of Key Project Indicator (KPI) metric, has been performed. Resulting KPIs figures are analysed in this report. This document is the final report of this study.

1.2. Scope

This document contains the Technical Readiness Assessment (TRA) Report for the AURORA project which results from the work performed in task T6.3 from Work Package 6 (WP6). The TRA process reuses Euclid's and UPMSat-2's testing facilities to determine the QGen's technology maturity.

2. Applicable and reference documents

2.1. Applicable Documents

ID	Title	Reference	Rev.
AD1	ECSS – Technology readiness level (TRL) guidelines	ECSS-E-HB-11A	01/03/2017
AD2	AURORA Grant Agreement	GA number 101004291	N/A
AD3	AURORA Consortium Agreement (CA)	CA Nº 101004291 AURORA	N/A
AD4	ECSS – Software Metrication Handbook	ECSS-Q-HB-80-04A	30/03/2011

Table 1: Applicable Documents

2.2. Reference documents

ID	Title	Reference
RD1	TASTE Toolset	https://taste.tools/
RD2	Technology Readiness Levels Handbook for Space Applications	TEC-SHS/5551/MG/ap
RD3	HORIZON 2020 – WORK PROGRAMME 2014-2015. General Annexes G. Technology readiness levels (TRL)	
RD4	Altunok, Taner & Cakmak, Tanyel. (2010). A technology readiness levels (TRLs) calculator software for systems engineering and technology management tool. Advances in engineering Software.	10.1016/j.advengsoft.2009.12.018
RD5	D6.1 Technology Readiness Level (TRA) Plan	AUR-UPM-PL-0005
RD6	D6.2 Evidences for the assessment report	AUR-ESC-RP-0014
RD7	D3.6 Test cases reporting (PIL & HIL)	AUR-ESC-RP-0008
RD8	U.S. Government Accountability Office (2020). Technology Readiness Assessment Guide: Best Practices for Evaluating the Readiness of Technology for Use in Acquisition Programs and Projects	GAO-20-48G

Table 2: Reference Documents

3. Terms, Definitions, and Abbreviated Terms

The acronyms and abbreviations of this document are listed below.

Acronym	Description
ACS	Attitude Control System
AOCS	Attitude, Orbit, and Control System
CA	Consortium Agreement
ECSS	European Cooperation for Space Standardisation
ESA	European Space Agency
GA	Grant Agreement
MGM	Magnetometer
MGT	Magnetic torquer
NA	Not Available
PM	Previous Metric
TRA	Technology Readiness Assessment
TRL	Technology Readiness Level
UPM	Universidad Politécnica de Madrid
WP	Work Package
ACS	Attitude Control System

Table 3 Acronyms

4. TRA Process Overview

This chapter includes an overview of the Technology Readiness Assessment (TRA) process defined for the evaluation of the QGen toolset.

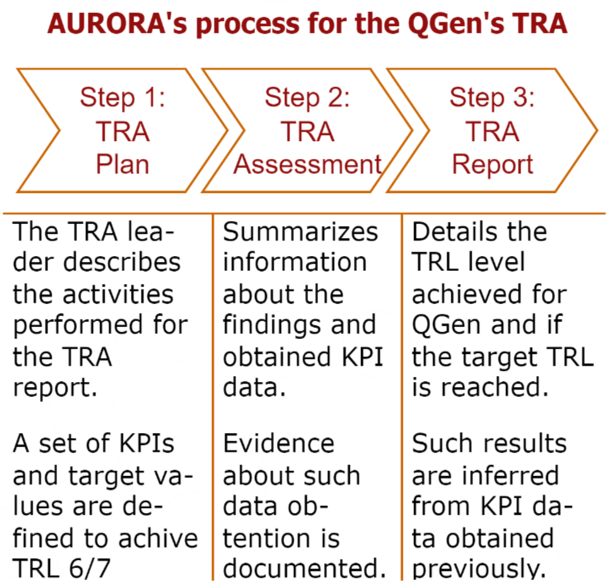

Figure 1 - TRA Process for the QGen evaluation

The whole TRA process is summarized in the Figure 1 that depicts the three tasks which conform a sequential pipeline of steps.

The activities performed in each step has been documented in a deliverable that serves as an input for the next steps in the chain.

The first step to perform this evaluation was **(1)** the definition of a TRA plan which has been defined in deliverable D6.1 [RD5] as the result of the activities carried out in task T6.1. The TRA process is based on a quantitative approach, proof of that is the list of Key Performance Indicators (KPIs) defined in the plan which have associated a set of reference or target values that must be met to reach the desired TRL (6 or 7).

The next step is **(2)** the TRA assessment which describes the UPMSat-2's Attitude Control System (ACS) and EUCLID's Attitude and Orbit Control System (AOCS). These are used as technology evaluators and allow the obtention of a set of measurable values that, in turn, helped to fulfil the KPIs defined in D6.1. As a result, deliverable D6.2 [RD6] was generated. Such document covers the evidence for the applicability of the QGen tool set in the software design, modelling, simulation and verification of Attitude, Orbit and Control Systems (AOCS) applications in space missions.

The next table summarizes the KPIs that are used for the readiness evaluation and data gathering. The objective of every KPI is included too.

KPI Code	KPI Description	Purpose
OIQ01	Integrability in software architecture	Evaluate whether the QGen code is integrable into other software architectures
OIQ02	Interoperability with other software elements	Evaluate if QGen code can reuse other software elements
OIQ03	Integration of Simulink/QGen models and TASTE models	Evaluate whether QGen is integrable into TASTE ([RD1])
OIQ04	Number of Simulink models integrated in the TASTE software architecture model	Evaluate complexity of the process of integration of Simulink/QGen and TASTE
OMQ01	Modelling tools provide support to automatize the QGen code generation process.	Estimate the complexity of QGen usability with modelling tools
OAS01	Number of QGen tool support done to AdaCore	Estimate the support needed in the applicability of QGen
OAS02	AdaCore response time for support requests	Estimate the time needed to solve QGen applicability problems

KPI Code	KPI Description	Purpose
OAS03	Number of issues and bugs sent to AdaCore	Estimate the maturity of QGen tools
OAS04	Time to solve bugs	Estimate the maturity of QGen tools
ORS01	Modified Blocks. Percentage of modified blocks for QGen compatibility: % modified Blocks = nModifBlocks/nBlocks Where the evaluation unit is the reference model (not including internal reference models)	Estimation of effort to reuse Simulink models
DQA01	Number of QGen models used in project development	Estimate applicability of QGen in projects
DQA02	Complexity of Simulink models used in projects: number of Simulink elements	Estimate the complexity of QGen models integrated in practical projects
DQA03	Time development of Simulink models	Estimate the effort to develop Simulink models
DPI01	General increase in productivity	Estimate the reduction of development effort
DPI02	Reduction of tests	Estimate the reduction in testing time
DQA04	Maximum subsystem depth. Number of subsystem nesting levels	Estimate the complexity and maintainability of Simulink/QGen models
DQA05	Maximum number of basic Simulink blocks. Number of basic blocks per function	Estimate the complexity and maintainability of Simulink/QGen models
DQA06	Maximum number of nested bus structures. Number of levels of nested structures in model bus interfaces	Estimate the complexity and maintainability of Simulink/QGen models
DPI03	Deviation from reference models. Error tolerance in the MIL validation environment with respect to the reference Euclid models.	Evaluate a comparison of current Simulink models results and QGen models results.
DPI04	Deviation from MIL reference. Error tolerance in the MIL-SIL validation environment with respect to the MIL reference values.	Evaluate a comparison of MIL-SIL environment and QGen models results
DQA07	Code cyclomatic complexity. Number of linearly independent paths through a function.	Estimate the complexity and maintainability of Simulink/QGen models
DPI05	Maximum number of nested statements in a function. Note: retrieved for comparison to traditional manual code metrics Note: retrieved for comparison to traditional manual code metrics	Evaluate a comparison of complexity of traditional development methods and QGen models results
DPI06	Number of statements. Note: retrieved for comparison to traditional manual code metrics	Evaluate a comparison of complexity of traditional development methods and QGen models results

KPI Code	KPI Description	Purpose
DSQ01	Proportion of comments within the generated functions. Comment frequency = nCommentLines (excluding headers)/nCodeLines (excluding blanks) Note: retrieved for comparison to traditional manual code metrics	Estimate the maintainability of generated code with QGen
DPI07	Code size. Number of lines of generated code per function (including comments but not including blank spaces) Note: retrieved for comparison to traditional manual code metrics.	Estimate increase of software productivity based on autocade application
DSQ02	Code branch coverage results. Coverage % of branches during SIL unitary test verification for each function	Estimate the dependability and reliability of QGen based application
DSQ03	Code statement coverage results. Coverage % of function statements during SIL unitary test verification for each function. This evaluation will be applied at Software SIL.	Estimate the dependability and reliability of QGen based application
DSQ04	Code branch coverage results. Coverage % of branches during SIL unitary test verification for each function. This evaluation will be applied at Software SIL	Estimate the dependability and reliability of QGen based application
DSQ05	Code statement coverage results function. Coverage % of function statements during PIL unitary test verification for each function. This evaluation will be applied at Software PIL.	Estimate the dependability and reliability of QGen based application
DSQ06	SIL test execution. Percentage of exercised SIL test without error execution. This evaluation will be applied at Software SIL.	Estimate the dependability and reliability of QGen based application
DSQ07	PIL test execution. Percentage of exercised PIL test without error execution. This evaluation will be applied at Software PIL.	Estimate the dependability and reliability of QGen based application

Table 4: KPIs used throughout the project for the TRA of Qgen

Finally, step (3) in Figure 1 provides the result of the activities performed in task 6.3. This TRA report "*details whether the targeted TRL is reached and identifies the lacking aspects and associated evidence necessary to reach the targeted TRL* [RD2]".

5. Technology Evaluators

This chapter describes the two technology evaluators for the TRA assessment, that is, the UPMSat-2's Attitude Control System (ACS) and EUCLID's Attitude and Orbit Control System (AOCS).

5.1. Evaluation Models: UPMSAT-2 ACS

UPMSat-2 is an experimental microsatellite designed and developed at the IDR-UPM institute with the collaboration of the STRAST research group. The satellite is in orbit since September 2020, and it was developed for educational and technology demonstration purposes. Consequently, it includes several experiments and subsystems such as the Attitude Control System (ACS) designed and validated by aerospace and software engineers using the MATLAB and Simulink modelling tools.

Specifically, The Simulink Embedded Coder tool was used to transform the Simulink models into C code for later like integration into the final Ada software system. The validation and verification from these models and their auto-generated code were performed with additional MATLAB and Simulink toolboxes following the in-the-loop methodology. The TRA process reuses the simulation and control Simulink models used for the ACS subsystem. These models were taken as reference inputs for the QGen tools. We have used the QGen code generator and QGen Debugger for the SIL and PIL validations.

5.1.1. UPMSat-2 Attitude Control System

The UPMSat-2 ACS oversees the satellite's attitude determination based on the magnetic interaction with the Earth's magnetic field and the one produced through magnetic torques. The ACS sets the satellite rotation rate controlled with a constant angular speed and maintains the vehicle's attitude perpendicular to its orbit plane, thereby the communication antenna is properly oriented to the Earth and a better thermal control is achieved. The satellite is equipped with three magnetometers (MGM) to measure the Earth's magnetic field strength and direction, each one measuring in the three axes. Three magnetorquers (MGT) are used to generate the required torque for attitude control in each axis. The AADL [AADL] diagram presented in Figure 2 depicts the real-time software architecture from the UPMSat-2 ACS. The ACS software process includes three tasks (dashed ▱) and three data (▭) components. Tasks implement the active behaviour and data components represent shared resources allowing the communication and synchronization between the real-time tasks.

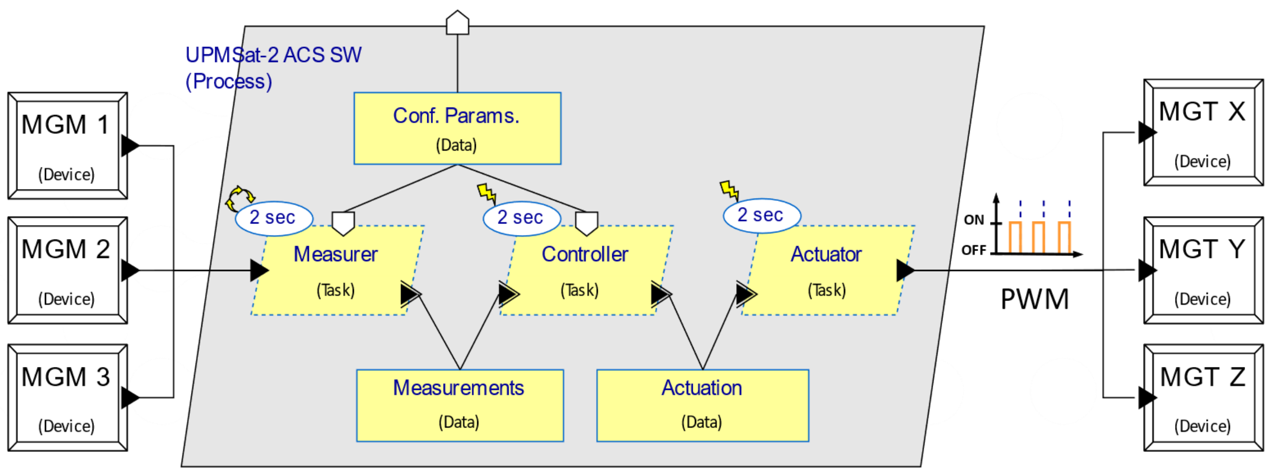

Figure 2 - UPMSat-2 ACS static software architecture specified in AADL

The architecture corresponds to the classical control cycle: sense, compute, and actuate. These three activities are divided respectively into three different tasks to maintain low coupling and independent priorities. The following list describes each task's responsibilities:

- Measurer is a cyclic task that reads five magnetic field samples from the three MGMs and computes their average. These values are passed to the Controller task by means of the Measurements protected object.
- Controller is a sporadic task that imports the auto-generated code from the control algorithm implemented in Simulink. It is activated upon reception of the MGM measurements sent by the Measurer periodic task. At the end of its cycle, it sends the Pulse Width Modulation (PWM) commands to the Actuator task by means of the Actuation protected object.
- Actuator is a sporadic task that is activated upon reception of the actuation commands through the Actuation protected object, which contains PWM duty cycles for each MGT. This task implements a "bit-bang" PWM control, switching on and off all MGTs depending on their duty cycles.

The UML Sequence diagram depicted in Figure 3 presents the dynamic architecture of the ACS subsystem. The three tasks are presented as three active objects, while the protected resources as passive objects. All function calls are modelled as UML messages whose sequence number is based on the task context, e.g.: ID messages received and sent by the Controller have a value of 2.

As shown in the sequence diagram, both protected objects activate its "consumer/reader" task whenever new data is inserted by its "producer/writer". At the beginning of the execution, both Controller and Actuator request for new data (messages #2 and #3, respectively) but are blocked since no data is available. After the Measurer execution, Controller is activated receiving the new data from Measurements (messages #1.1 and #2.2). A similar sequence occurs in the Actuator task timeline. However, this does not apply to the Measurer task since it is activated cyclically by the system's dispatcher with a period of two seconds (message #1).

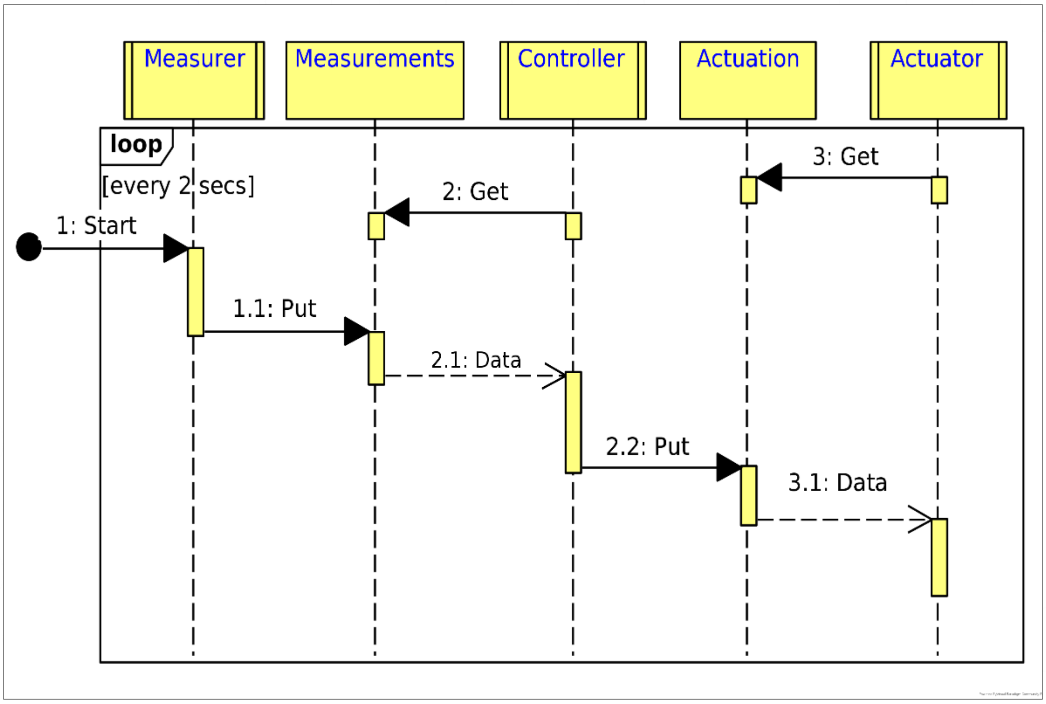

Figure 3 - UML Sequence diagram for the ACS subsystem

5.1.2. Simulink model for the UPMSat-2 Attitude Control System

UPMSat-2 ACS models not only implement the control law, but also the satellite dynamics, the environment, its perturbations, and the spacecraft sensors and actuators. Figure 5 depicts the high-level Simulink model from the UPMSat-2 ACS.

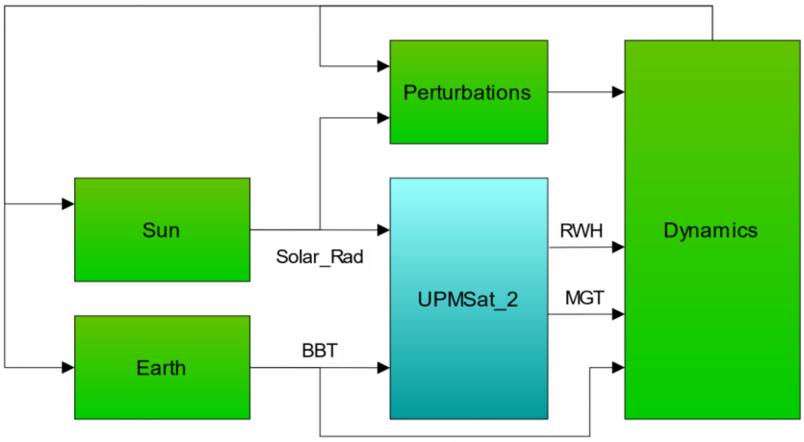

Figure 4 - High level view from the ACS Simulink model

As shown in Figure 4, there are four green blocks that model the real physical environment from the satellite: Earth, Sun, satellite's dynamics, and perturbations. The output signals from the dynamics block are fed-back to the other blocks to close the simulation loop. The last turquoise-blue block represents the UPMSat_2 satellite and contains the ACS algorithm and additional blocks to simulate equipment required by the ACS such as sensors and actuators, namely MGMs and MGTs.

In brief, the UPMSat-2 block receives the solar radiation signal from the Sun block and the Earth's magnetic field (BBT) from the Earth block. These signals are then processed by its inner sensor blocks and analysed by the ACS algorithm to control the MGTs. Alternative ACS equipment such as solar sensors or reaction wheel actuators were used for experimental purposes, which is not in the scope of this evaluation.

Finally, the nominal control subsystem, which is inside the UPMSat_2 block, is depicted in Figure 6. The model is decomposed into five subsystems that conform a functional chain to process the raw measurements and compute the required actuation on the MGTs.

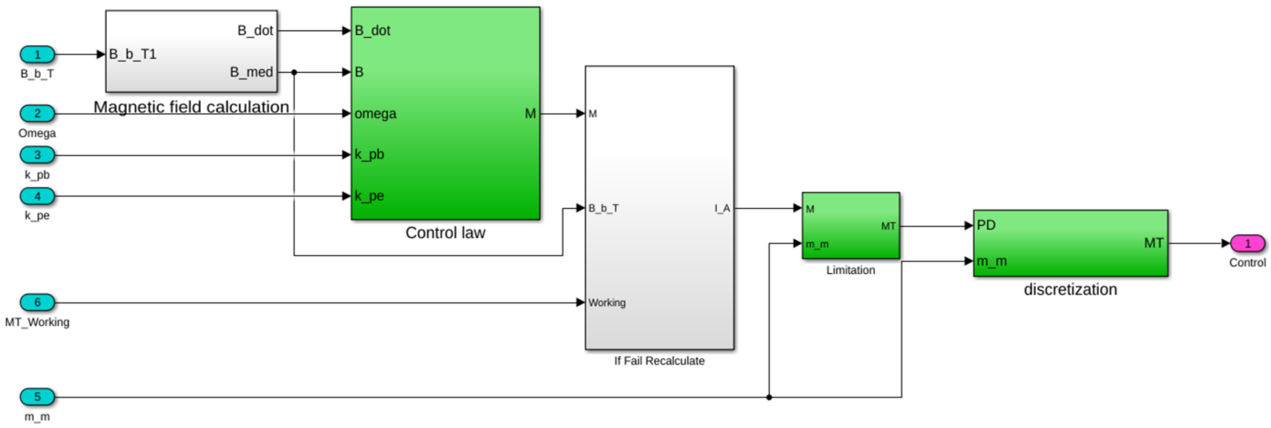

Figure 5 - Simulink model for the ACS control block

The Magnetic field calculation block receives 15 samples from the three MGM sensors (B_b_T signal) and then generates two output signals: (i) B_dot, its derivatives as an array of three floating point elements representing each axis value; and (ii) B_med, that represents the mean from the 15 samples. These data are received by Control_law, which implements the mathematical equation to compute the magnetic torquers. Then, these values are received by the If_Fail_Recalculate block by means of the M signal, which behaves accordingly to the MGMs that are currently operating (MT_Working).

Finally, the M signal is passed through the limitation and discretization for further processing. These steps compute the required torques in a digital value format that represents the PWM duty cycles in milliseconds.

5.2. Evaluation Models: EUCLID AOCS

Euclid is a medium-class mission of ESA's Science Program whose objective is the elucidate the geometry and the nature of the dark energy and dark matter components with unprecedented accuracy, of the order of micro arcseconds when in science mode. For that a complex multi-mode AOCS has been developed by Sener Aeroespacial.

The AOCS is composed of sensors (Sun sensors, IMU, Coarse Rate sensors, STR and Fine Guidance Sensor) and actuators (Reaction wheels, Reaction Control System and Micro Propulsion System) for the different activities needed. These sensors and actuators are needed based on the AOCS operating mode, specifically, the *science* mode is the most demanding mode due to the hard limitations in terms of accuracy and stability.

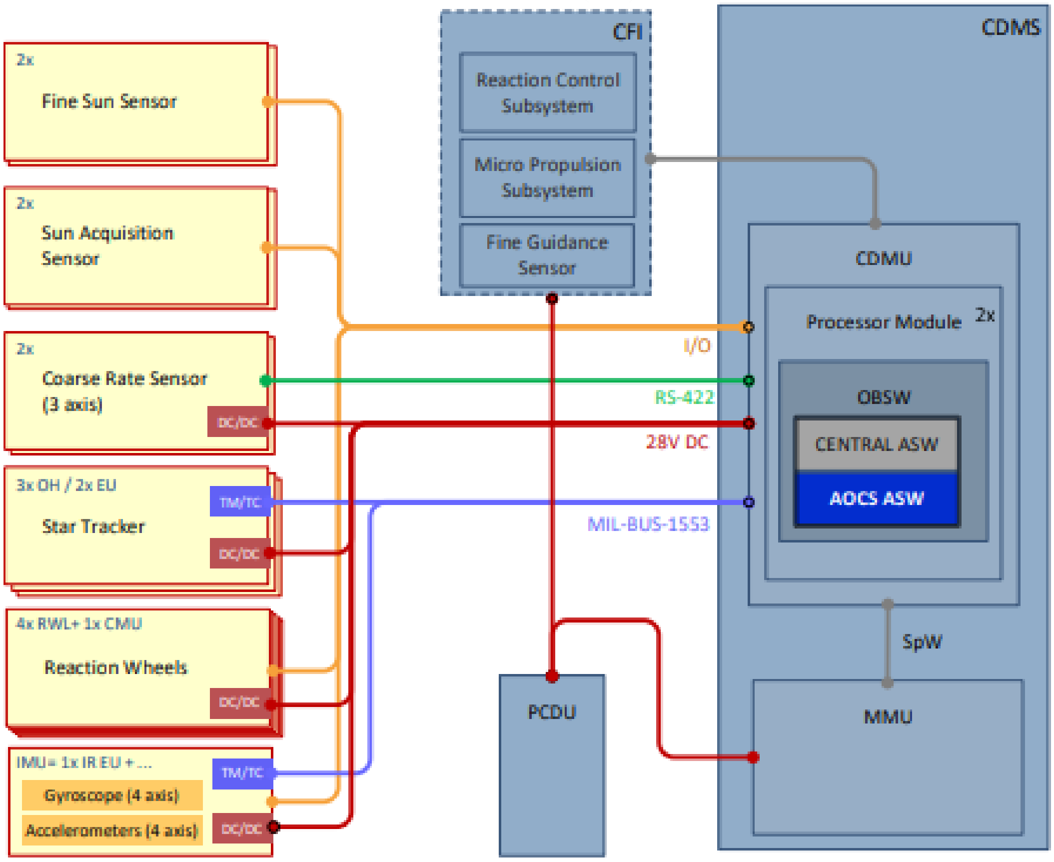

Figure 6 - Euclid AOCS Architecture

Euclid AOCS has the special feature that it is one of the first ESA missions to be launched where the GNC algorithms were autocoded based on Simulink models shifting the traditional manual code validation to an autocoded philosophy, which focus on reduction on developing times. This code, written in C, is then integrated inside a manual coded application software containing the Failure, Detection, Isolation and Recovery (FDIR) functionalities, the communication with the rest of the systems and the mode manager.

The Euclid AOCS architecture is divided into the different modes that will perform the necessary actuation for the mission success. Those are:

- Safe Mode (SFM)
- Standby Mode (SBM)
- Sun Acquisition Mode (SAM)
- Orbit Control Mode (OCM)
- Fine Pointing Mode based on Reaction Wheels (FPMRWL)
- Fine Pointing Mode based on Reaction Control Thrusters (FPMRCS)
- Science Mode (SCM)

Due to the large scale of the AOCS and the intrinsic complexity of some of the modes, only SAM, OCM and FPMRCS were selected as a direct comparison between the previously generated code based on the Simulink Embedded Coder and the QGen autogenerated code.

5.2.1. Sun Acquisition Mode

SAM is in charge of damping the spacecraft (SC) angular rate at entry (launch separation or transition from other mode), acquiring and maintaining Sun pointing. It has also incorporated in its functions the protection against telescope illumination and dark zone situations, where the position of the Sun cannot be estimated due to SAS not generating any current

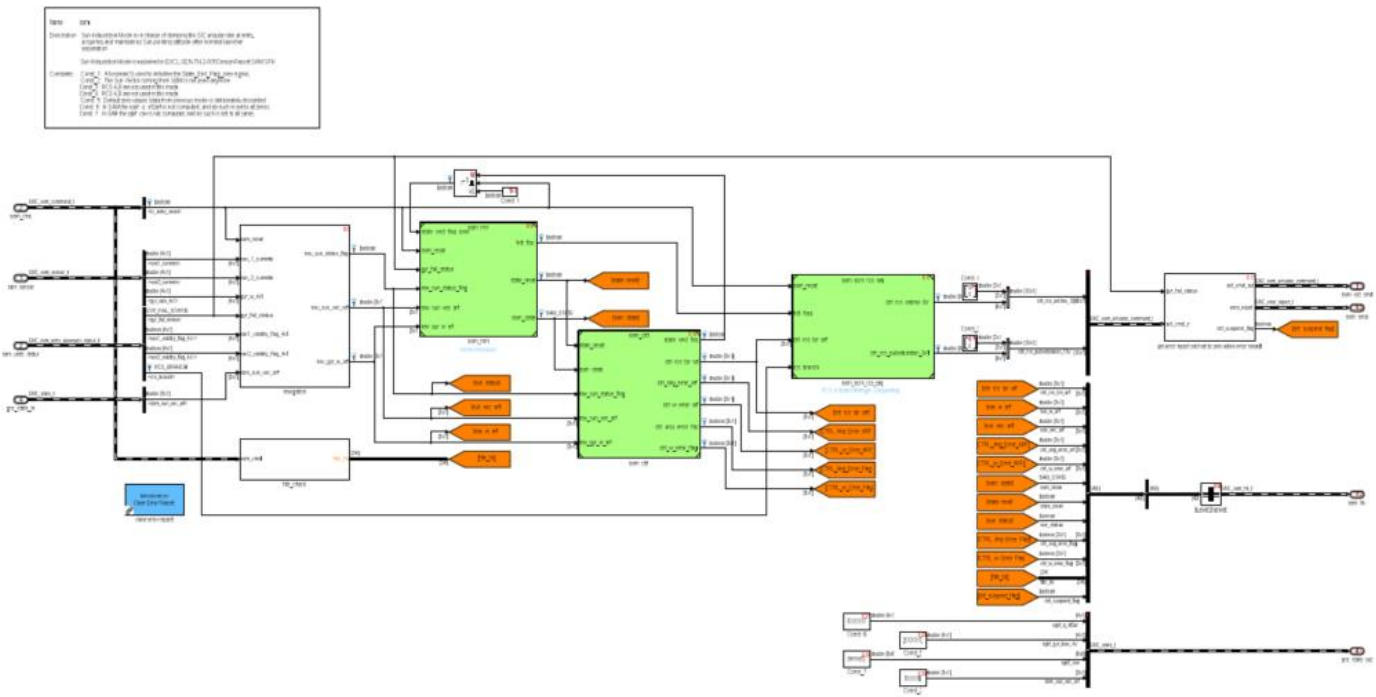

Figure 7 - SAM model

5.2.2. Fine Pointing Mode based on Reaction Control System

The FPMRCS is activated by an external command under certain FDIR alarms. It is in charge of providing three axis inertial pointing during the transition between other modes, when the SC is out of science observation, and it also provides slew capabilities to achieve the required attitude within the operational domain.

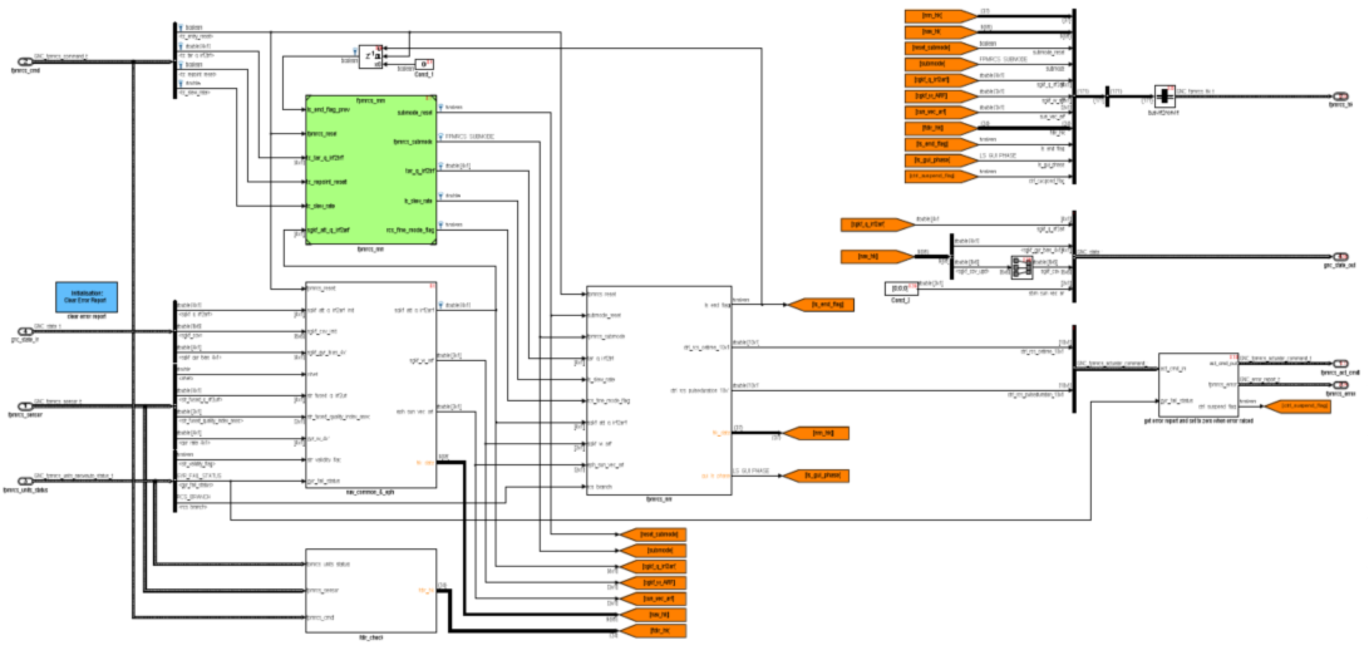

Figure 8 - FPMRCS model

5.2.3. Orbit Control Mode

The OCM is in charge of executing delta -V manoeuvres for orbit correction and orbit station keeping manoeuvres, for providing three axis inertial pointing during orbit correction manoeuvres, using RCS as actuator for both, orbit control manoeuvres and attitude control pulses. In addition, the mode will also manage the angular momentum of the system (SC + reaction wheels, for acquiring any user required angular momentum in the wheels.

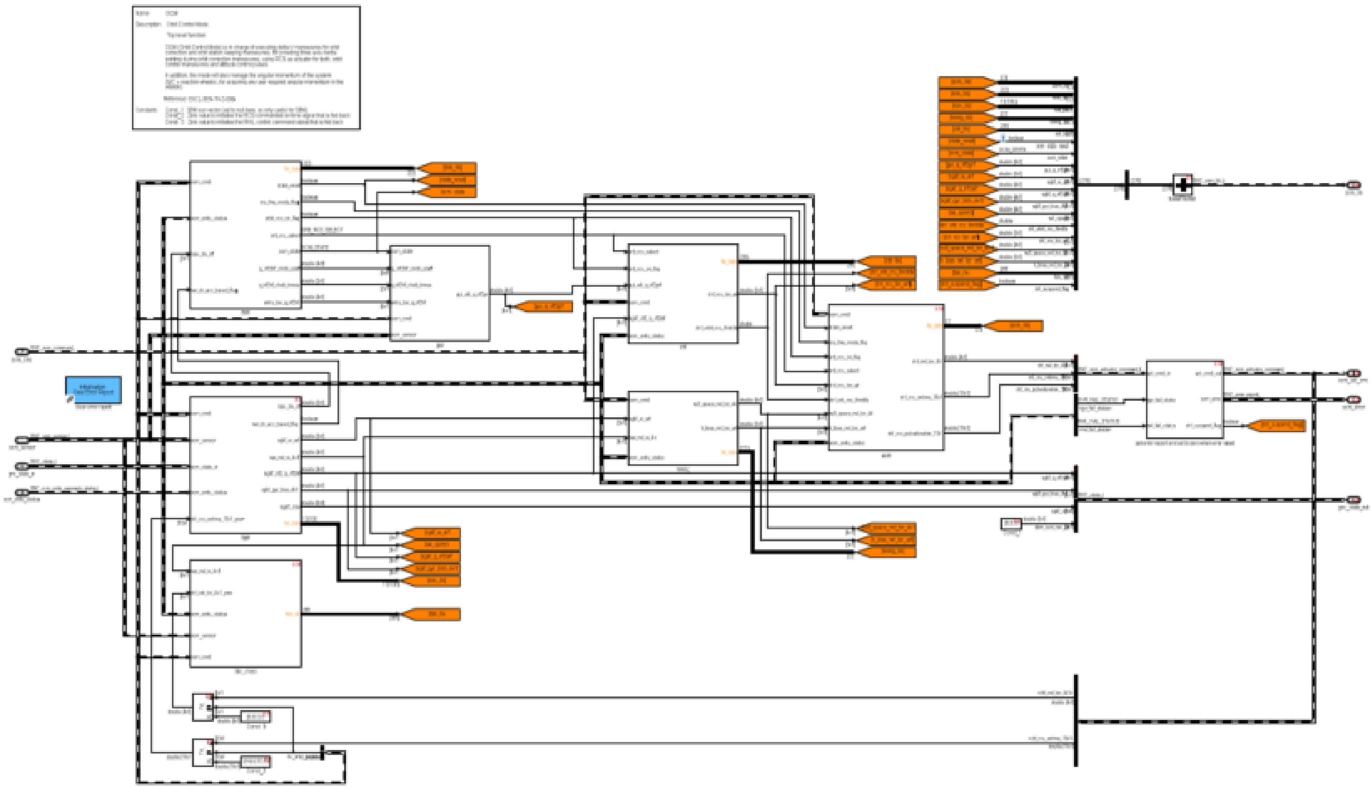

Figure 9 - OCM model

6. Assessment Strategy

The TRL Assessment consists of a systematic process which determines the maturity of certain technologies for its insertion in the industry. The technology maturity is also known as the TRL and, to conduct the TRA, they must be clearly defined and stablished without ambiguity.

6.1. TRL Definition

For a first approximation, we have adopted Annex G from the Horizon 2020 Work program 2014-2015 [RD3] that defines six TRLs applicable to H2020 projects.

In addition, we have analysed additional TRL/TRA definitions provided by other organizations. The major finding was ESA's ECSS-E-HB-11A Handbook [AD1] which defines its own TRLs and provides guidelines to perform the TRA in the space realm. ESA's TRL definitions are based on the ISO standard "16290 Space systems – Definition of the Technology Readiness Levels (TRLs) and their criteria assessment". The following table summarizes the definitions of H2020 and ESA's TRL levels which, as shown below, are equivalent. However, the latter is more oriented to space systems and contains detailed definitions and concepts, while the former targets generic systems and provides brief definitions. It is important to pay special attention to the differences of definitions of levels 6 and 7.

TRL	H2020's Definition	ESA's Definition
1	Basic principles observed	Basic principles observed and reported. *Description:* Lowest level of technology readiness. Basic research begins to be translated into applied research and development.
2	Technology concept formulated	Technology concept and/or application formulated. *Description:* Once the basic principles are observed, practical applications can be invented, and R&D started. Applications are speculative and may not be proven.
3	Experimental proof of concept	Analytical and experimental critical function and/or characteristic proof-of-concept. *Description*: Active research & development is initiated, including analytical/laboratory studies to validate predictions regarding the technology.
4	Technology validated in lab	Component and/or breadboard function verification in laboratory environment. *Description*: Basic technological components are integrated to establish that they will work together.
5	Technology validated in relevant environment (industrially relevant environment in the case of key enabling technologies)	Component and/or breadboard critical function verification in a relevant environment. Description: The basic technological components are integrated with reasonably realistic supporting elements so that they can be tested in a simulated environment.
6	Technology demonstrated in a relevant environment (industrially relevant environment in the case of key enabling technologies)	Model demonstrating the critical functions of the element in a relevant environment. *Description*: A representative model or prototype system is tested in a relevant environment.

TRL	H2020's Definition	ESA's Definition
7	System prototype demonstration in operational environment	Model demonstrating the element performance for the operational environment. *Description*: A prototype system that is near, or at, the planned operational system.
8	System complete and qualified	Actual system completed and accepted for flight ("flight qualified"). *Description*: In an actual system, the technology has been proven to work in its final form and under expected conditions.
9	Actual system proven in operational environment (competitive manufacturing in the case of key enabling technologies; or in space)	Actual system "flight proven" through successful mission operations. *Description*: The system incorporating the new technology in its final form has been used under actual mission conditions.

Table 5: Horizon 2020 Work program 2014-2015 TRLs definition

ECSS's handbook not only provides definitions and guidelines of the TRL levels and TRA process for the evaluation of space systems, but it refines those definitions to evaluate software components and tools used in space applications.

The H2020's TRLs contain definitions that are not discipline specific, i.e.: they are applicable to any type of systems and projects. Software components have their own development and qualification processes which must be extrapolated to the generic TRA and TRL defined in such standards. This evaluation adopts the guidelines discussed by ESA in [AD1] Annex A which gives a clear definition of TRLs for software development and presents a generic TRA process targeting the following types of software:

- Software tool: software element that is used for supporting specific activities of the software life cycle. Examples are tools like automatic code generators (TASTE, QGen, Embedded coder), modelling/development tools (Simulink or Space Creator) or software compilers.
- Building block: software element that interacts with other parts of the system and that are conceived to be reused in several applications. Examples are middleware like PolyORB-Hi or NASA's OSAL.
- Tailored generic software product: it may be a software tool or building block which is adapted to a dedicated environment but does not involve source code modification.

As stated above, QGen belongs to the software tool category since it used to accomplish the development, verification and validation (V&V) of software components. However, its automatically generated code is not a building block neither a tailored generic product, but it is a specific software element because it targets a specific software application (UPMSat-2 and EUCLID case studies).

Next table summarizes the links defined by ESA between TRL 6/7 and the development status of a software tool. Annex A contains Table 9 which presents a detailed description for all ESA's software TRLs.

TRL	Description	Requirements	Verification	Viability
6	The tool is ready for use in an operational or production context.	All use cases and error handling implemented. User friendliness validated.	V&V process is complete for the intended scope, including robustness. V&V activities must be executed in an end-to-end fully representative laboratory environment including real target.	It is demonstrated that the tool was applied in an operational project.
7	The tool is ready for market deployment.	Validity of the tool confirmed within intended application.	The tool was successfully validated in a pilot case, representative of the intended project application.	Engineering support and maintenance organization in place, including helpdesk

Table 6: ESA's definition of TRL 6 and 7 for Software

6.2. Reference values for TRL 6/7

In the AURORA project, the estimated initial TRL for QGen technology is 4 and the target TRL is 6 or 7 for an operational certified tool. The TRA report is a metric-based process, thereby, to carry out the assessment the TRA plan has included the definition of a set of KPIs. In deliverable "D6.1–TRA Plan" we have included the definitions and estimation of the reference values for each KPI (defined in section 5) in order to achieve a TRL level of 6 or 7. This evaluation is based on a quantitative analysis, where each KPI data is characterized by numerical units. Hence, the achievement of the desired TRL (level 6/7) is accomplished by comparing the obtained measurements from these reference values.

The original reference values defined in D6.1 and referenced in D6.2 [RD6] have been improved considering the metrication handbook from ECSS [AD4] to provide KPI reference values for static code analysis (e.g.: cyclomatic complexity, nested statements, among others) that are stablished as target values for flight software with criticality category A, the most rigorous level for safety-critical software.

The KPIs Reference values are described in the following table.

KPI Code	Units	Reference Values for TRL 6/7[a]
OIQ01	Boolean	True
OIQ02	Boolean	True
OIQ03	Boolean	True
OIQ04	Number of models	> 2
OMQ01	Number of tools	> 1
OAS01	Number of support requests	N/A
OAS02	Average time of requests	< 2 days
OAS03	Number of issues or bugs sent to AdaCore	N/A

[a] Updated values and new ones are both italicized and marked in bold.

KPI Code	Units	Reference Values for TRL 6/7[a]
OAS04	Average number of days to solve a bug	< 4 days
ORS01	Percentage	< 25 %
DQA01	Number of models	< 3
DQA02	Average number of elements	> 15
DQA03	Hours of engineering effort	N/A
DPI01	Percentage of productivity increase	**≥ 20 %**
DPI02	Reduction of tests	**≥ 20 %**
DQA04	Nesting levels from subsystem	< 15
DQA05	Number of blocks	**≥ 200**
DQA06	Number of nested structures in model bus interfaces	**≥ 1**
DPI03	Error tolerance	0 (no difference)
DPI04	Error tolerance	1e-15
DQA07	MAX code cyclomatic complexity	***≤ 10***
DPI05	Number of nested statements	***≤ 5***
DPI06	Number of statements	***PM[b] ± 10 %***
DSQ01	Percentage (comments/function)	***≤ 30 %***
DPI07	Number of LOC	***PM ± 10 %***
DSQ02	Code branch coverage %	≥ 80%
DSQ03	Code statement coverage % in SIL	≥ 80%
DSQ05	Code statement coverage % in PIL	≥ 80%
DSQ06	SIL test effectiveness	Integration Test: ≥ 90% Verification Test: 100%
DSQ07	PIL test effectiveness	Integration Test: ≥ 90% Verification Test: 100%

Table 7: Reference values for TRL 6/7

Regarding number of statements and LOC (DPI06 and DPI07) we have used reference values relative to the **Previous Metric (PM)** obtained during the development of the original projects. In the case of UPMSat-2, these **PMs** correspond to analysis of the code generated with Embedded Coder.

Currently, there are three N/A target values, that although do not help to stablish the TRL quantitatively, they do allow us to evaluate the quality of the tool and support the obtention of other numerical values. For instance, the "OAS01: number of support requests" helps to evaluate/obtain "OS04: Average number of days to solve a bug" which, in turn, analysis the quality of the technical support provided by AdaCore (QGen's maintainer).

[b] PM: Previous Measurement from the original project that was later adapted for its usage in AURORA as case study. In the case of UPMSat-2 the ACS, PMs include metrics from the code generated with Embedded Coder.

6.3. TRL Calculator

Agencies developed standardized TRL Calculators to determine the achieved TRL for a given technology in specific applications. For instance, a "TRL Calculator" was defined in [RD4] based on the collection of qualitative and quantitative values in the form of spreadsheets and including different weights to represent the importance of different categories. This approach has benefits and disadvantages that must be considered. For instance, in its TRA guide, the U.S. Government Accountability Office [RD8] stated that although TRL Calculators help to simplify and automate the determination of the TRL, the critical evaluation is comprised, and it should include a separate review from the TRA team. Therefore, the approach followed for the QGen's TRL determination included both the **TRL Calculator,** and the **Analysis of Results** as depicted in the Figure 10.

The TRL Calculator required the specification of a formula that returns a single numerical value based on the actual and target KPI measurements. To define such value, the success rate is defined as follows:

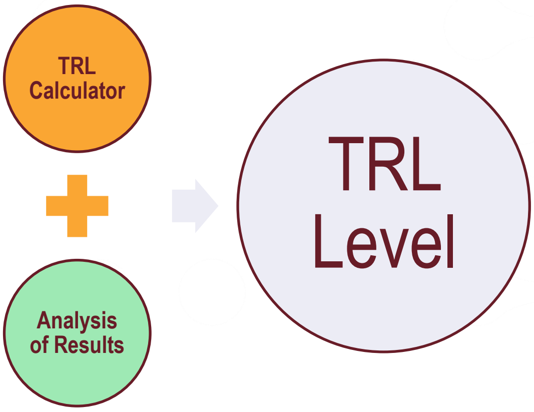

Figure 10 - Strategy for the TRL determination of QGen

$$Success = \frac{Passed\ KPIs}{Passed\ KPIs + Non\ Passed\ KPIs}$$

where:

- $Passed\ KPIs$ represents the total number of KPIs that achieved the reference values.
- Those which did not are denoted as $Non\ Passed\ KPIs$.

$$TRL = \begin{cases} 6 & Success\ Rate \geq 80\ \% \\ 7 & Success\ Rate \geq 90\ \% \end{cases}$$

Figure 11 TRL Calculator

This formula is used as a guidance, and it is limited to TRL-6 and TRL-7 because in the context of this evaluation, it is impossible to claim attainment of TRL-8 or TRL-9 for two reasons: (i) reference value for KPIs is specified for TRL-6 and TRL-7 and (ii) TRL-8 and TRL-9 require validation in the real environment. That is why this formula was considered as a sufficient condition to report the achieved TRL. For instance, there may be situations in which the formula indicates a success rate of 90.2% which corresponds to TRL-7, but since it is close to TRL-6, the evaluators may decide to decrease to TRL-6 based on the documentation of the tool and additional aspects such as the number of unanswered KPIs or the V&V environment.

7. Evaluation Report

7.1. Analysis of Key Data

The following table provides an assessment of the acquired KPI data from both UPMSat-2 and EUCLID case studies as reported in “D6.2 Evidence for the assessment report” RD6. These data are compared with the reference values defined in “Table 7: Reference values for TRL 6/7”.

Most KPIs are project dependent, i.e.: their values differ across projects. However, some KPIs (such as operational indicators) are shared across case studies and their values are common for both UPMSat-2 and EUCLID.

The second column of this table depicts the scope of the KPI; (i) if it is a *case-study dependent KPI*, then results are decomposed in two rows, one per project; on the other hand (ii) if it is a *project-scope KPI*, only one row is presented.

KPI	Scope	Results	Assessment
OIQ01	Common	True	In Target
OIQ02	Common	True	In Target
OIQ03	Common	True	In Target
OIQ04	Common	3	In Target
OMQ01	Common	3	In Target
OAS01	Common	25	In Target
OAS02	Common	7.58 hours	In Target
OAS03	Common	3 minor bugs, 1 major bug, 17 issues	Number of issues or bugs sent to AdaCore KPI: Achieved
OAS04	Common	17.18 days	Out of Target
ORS01	EUCLID	3%	In Target
	UPMSat-2	2.29%	In Target
DQA01	EUCLID	33	In Target
	UPMSat-2	1	Out of Target
DQA02	EUCLID	6005	In Target
	UPMSat-2	619	In Target
DQA03	EUCLID	5630/410 hours	Hours of engineering effort KPI: Achieved
	UPMSat-2	800 hours	Hours of engineering effort KPI: Achieved
DPI01	EUCLID	60%	In Target
	UPMSat-2	N/A	Percentage of productivity increase: Not evaluated

KPI	Scope	Results	Assessment
DPI02	EUCLID	30%	In Target
	UPMSat-2	N/A	Reduction of tests Not evaluated
DQA04	EUCLID	5	In Target
	UPMSat-2	5	In Target
DQA05	EUCLID	1386	In Target
	UPMSat-2	250	In Target
DQA06	EUCLID	4	In Target
	UPMSat-2	0	Out of Target
DPI03	EUCLID	0.0E+00	In Target
	UPMSat-2	N/A	Error tolerance Not evaluated
DPI04	EUCLID	1.0E-15	In Target
	UPMSat-2	99.97 % of effectiveness with 0.0 tolerance for error	In Target
DQA07	EUCLID	7.68	In Target
	UPMSat-2	5.42	In Target
DPI05	EUCLID	6	Out of Target
	UPMSat-2	3	In Target
DPI06	EUCLID	29541	In Target
	UPMSat-2	888	In Target
DSQ01	EUCLID	27.4%	In Target
	UPMSat-2	55.1%	Out of Target
DPI07	EUCLID	28.2	In Target
	UPMSat-2	21.2	In Target
DSQ02	EUCLID	96.14%	In Target
	UPMSat-2	95.2%	In Target
DSQ03	EUCLID	96.11%	In Target
	UPMSat-2	97.9%	In Target
DSQ05	EUCLID	96.11%	In Target
	UPMSat-2	97.9%	In Target
DSQ06	EUCLID	100%	In Target
	UPMSat-2	100%	In Target

KPI	Scope	Results	Assessment
DSQ07	EUCLID	100%	In Target
	UPMSat-2	100%	In Target

Table 8: Acquired KPI data from UPMSat-2 and EUCLID demonstrator

From a total of **51** KPI assessment, **44** have a positive evaluation, **5** have not resulted as expected, and **2** were not evaluated. Thereby, the obtained overall success rate from *TRL Calculator* is equals to **89.80 %**.

Considering separately an evaluation for the 2 demonstrators, the success rate from *TRL Calculator* is:

- EUCLID: 96.67%
- UPMSat-2: 82.14%.

Based on this information, the following sections provide the TRLs considering the UPMSat-2 evaluator, EUCLID technology demonstrator, and the two of them.

7.2. QGen's TRL for UPMSat-2 evaluator

Based on H2020's definition, TRL 7 corresponds to "System prototype demonstrated in operational environment" where an operational environment is understood as "Environment that addresses all of the operational requirements and specifications required of the final system or technology". Examples of operational environments include the test beds for spacecraft vehicles.

In the AURORA project, the UPMSat-2 evaluator did not include hardware equipment but software simulators and emulators of the final operational environment.

H2020's TRL-6 is defined as "Technology demonstrated in a relevant environment" where a relevant environment is understood as a "Testing environment that simulates the key aspects of the operational environment". This definition suites better to the UPMSat-2 evaluator since they are models that simulate the final satellite's vehicle, dynamics, and physical environment. Thereby, if we consider only UPMSat-2 use case, H2020's TRL-7 would be a higher level for the QGen's technology maturity, and it is inferred that:

*The TRL achieved considering only UPMSat-2 information is **TRL 6***

7.3. QGen's TRL for EUCLID evaluator

EUCLID evaluator differs from UPMSat-2 since it includes the operation environment used during the original EUCLID's development and test campaign. Consequently, QGen has been demonstrated and tested in an operational environment rather than a relevant environment. Based on this data it is inferred that:

*The TRL achieved considering only EUCLID information is **TRL 7***

7.4. Overall QGen's TRL

EUCLID's testing facilities allowed to increase the QGen's maturity level to TRL 7 from the original TRL6 which was obtained with the UPMSat-2 evaluator. Consequently, considering both UPMSat-2 and EUCLID evaluators it can be inferred that:

*QGen's TRL is **TRL 7***

On the other hand, if we consider the definition of ECSS for software tools, QGen suites better at TRL 7 and 8 on aspects related to documentation and tool-support since, as it has been demonstrated by some KPIs, AdaCore has provided satisfactory technical support with low response times to solve reported bugs and answer technical questions. In addition, the tool features good technical documentation that contains installation manual, guidelines for tailoring the tool and a complete specification (including unsupported features). However, there is scarce documentation about industrial applications that have successfully used the toolset and regarding the technology evaluator, the tool has not been proven to work in a "flight qualified" space system. Consequently, in the context of space software applications it is inferred that:

QGen does not achieve a TRL of 8 or 9

7.5. Conclusions

This report is inferred from the analysis of the tool in two subsystems used in real projects, namely: ACS from UPMSat-2 and AOCS from EUCLID. The autogenerated code from such models was validated with their testing facilities and making use of QGen's verification and validation capabilities. UPMSat-2 fits int the category of an "relevant environment" (i.e.: models or prototypes of the final operational environment) whereas EUCLID's evaluator correspond to an *operational environment*.

This demonstration exercise has verified the similarity of software incorporating the QGen technology with respect to the actual Euclid application software. Similar levels of performance have been achieved, as described in the Test Cases Report [RD7].

In summary, the evaluation report has determined that the overall TRL determined for the QGen toolset is **TRL-7**:

- Considering UPMSat-2 data, QGen has achieved a TRL 6 per H2020 and ECSS definitions.
- Considering EUCLID data, QGen has achieved a TRL 7 per H2020 and ECSS definitions.
- Considering both UPMSat-2 and EUCLID evaluators, QGen has achieved a TRL 7 per H2020 and ECSS definitions.

The TRA process has also allowed us to demonstrate the viability of a reduction in the efforts and planning of the software life cycle at incorporating Autocoding technologies in the process:

- A general increase in productivity: reduction of development effort by 60%.
- A reduction of testing campaign: reduction in testing time by 30%.

We could not however quantify relevant results about the objective of the improvement of the integration phase (time and costs) by applying the Interoperability capability as resulting from WP5 on Component-Based Interface (CBI) technologies. This objective could be reached considering CBI implementation in a long-term approach (as technology consolidates).

We should consider that since the TRA was conducted for EUCLID's ADCS and UPMSat-2's ACS, the results and conclusions from this evaluation are limited to those use cases. If multiple use cases were considered other capabilities could have been tested. For instance, the UPMSat-2 ACS only included one Simulink model that could be transformed into source code by the tool. This affected operational KPIs since, during its integration into the TASTE toolset, the model could only be treated as a single passive component impeding the testing of required interfaces functionality for QGen and Simulink functions.

Finally, to consider that the viability of the QGen technology can be further developed to get an increase of TRL:

- QGen's TRL may be increased when the tool gets finally qualified at Tool Qualification Level 1 (TQL-1).
- QGen's TRL may be increased to, at least, TRL-8 if QGen had a longer record applicability in the industry.

8. Lists

8.1. List of Tables

8.2. List of Figures

Annex A: TRL Levels Refined for Software

This annex contains the complete definition of different TRL levels refined for software tools that were found/used for the QGen TRL assessment report.

Table 9: ESA's Software TRLs definition

TRL	Definition	Software Description
1	Basic principles observed and reported.	MATHEMATICAL FORMULATION • What: Expression of a problem and of a **concept of a solution**. • Where: There is no SW, then, there is no validation nor verification. • Doc.: Detailed mathematical formulation description. Publication of research results. • E.g.: the algorithm related to parsing source code to generate machine code exists.
2	Technology concept and/or application formulated.	Individual ALGORITHMs or functions are prototyped. • What: Practical application is identified. A concrete specification of a part of the problem is stablished. • Where: The prototype algorithms are tested with synthetic data in not necessarily representative final target. • Doc.: Algorithm implementation documented. Results documented. • E.g.: There is a set of prototypes that reads a selection of the source code syntax and generates machine code using part of the ISA.
3	Analytical and experimental critical function and/or characteristic proof-of-concept.	PROTOTYPE of the main functionalities of the integrated system. • What: Preliminary solution to specific needs. • Where: Some functionalities are tested in a representative of the final target such as a simulated laboratory environment. • Doc.: Architectural design of important functions is documented. Depending on size and complexity of the implementation. • E.g.: The architecture of the compiler is defined, and the complete source code syntax and semantics is covered.
4	Component and/or breadboard function verification in laboratory environment.	ALPHA version. - What: preliminary release of not-mature software version, distributed to a community at an early stage of the software development life cycle that implements the **main** functionality of the software. - Where: Alpha version of the software is tested internally by the developers in a representative of the final target in a representative simulated laboratory environment. - Doc.: TRL3 docs + user manual + design file.

TRL	Definition	Software Description
		- E.g.: The Alpha version of the compiler has a primitive GUI, generates non optimized machine code, and the execution time is slow. It is validated using typical examples of source code.
5	Component and/or breadboard critical function verification in a relevant environment.	BETA version. - What: preliminary release of not-mature software version, distributed to a community at an early stage of the software development life cycle, that implements the **complete** functionality of the software. - Where: Beta version of the software is tested internally by the developers in an end-to-end representative environment **including the real target**. - Doc.: Full doc. according to the applicable software engineering and quality standards. It includes test reports + application examples. - E.g.: The Beta version of the compilers has optimized the machine code generation, the performance, and the ergonomics of the GUI. A reference test suite of source code has been established to validate the compiler and the generated object code runs on the hardware processor.
6	Model demonstrating the critical functions of the element in a relevant environment.	Product RELEASE - What: the SW is ready for use in an operational or production context, including user support. - Where: Same as TRL 5 + all use cases are verified and validated. - Doc.: Full doc. according to the applicable software engineering and quality standards for a software product. - E.g.: The compiler is a Product with good documentation and acceptable performances. It produces error messages which are complete and user friendly. The support is organized as well as the product packaging and delivery.
7	Model demonstrating the element performance for the operational environment.	Early adopter version. - What: It corresponds to full validation on a representative pilot case. - Where: The SW was successfully validated **in a pilot case**, **representative** of the intended project application. - Doc.: TRL 6 doc + tailoring of the software tool + lessons learnt report + qualification tests. - E.g.: The compiler is delivered to early adopters for extensive testing. Then, the compiler robustness is improved following user feedbacks.
8	Actual system completed and accepted for flight ("flight qualified").	General product. - What: Stable version of the software available for the market. It corresponds to the readiness for the full deployment in operation.

TRL	Definition	Software Description
		- Where: The tool was successfully applied **in an operational project** but has **not** yet been **validated against the in- flight experience.** - Doc.: Full doc + spec + design + justification + V&V + users + installation manuals + sw problem reports + non-compliances + qualification files + lessons learnt report. - E.g.: The compiler is deployed to the complete user community, and it has been accepted to be used in a real space mission.
9	Actual system "flight proven" through successful mission operations.	Live product. - What: Stable version of the software available for the market in full business plan conditions. It corresponds to successful operations and performance achievement in the application. - Where: The tool was successfully validated in **one or several space missions**, including exploitation of in-orbit data. All anomalies encountered were analysed and resolved. - Doc.: TRL 8 docs + updates + track record of application in space projects. - E.g.: the compiler is deployed to the complete user community, and it has been used successfully in a spacecraft that has been launched and is fully operational.

Table 10: NASA's Software TRLs definition

TRL	Definition	Software Description
1	Basic Principles observed and reported.	Scientific knowledge generated underpinning basic properties of software architecture and mathematical formulation
2	Technology concept and/or application formulated	Practical application is identified but is speculative, no experimental proof or detailed analysis is available to support the conjecture. Basic properties of algorithms, representations and concepts defined. Basic principles coded. Experiments performed with synthetic data
3	Analytical and experimental critical function and/or characteristic proof of concept.	Development of limited functionality to validate critical properties and predictions using non-integrated software components.
4	Component and/or breadboard validation in laboratory environment	Key, functionally critical, software components are integrated, and functionally validated, to establish interoperability and begin architecture development. Relevant Environments defined and performance in this environment predicted
5	Component and/or breadboard validation in relevant environment	End-to-end software elements implemented and interfaced with existing systems/simulations conforming to target environment. End-to-end software system, tested in relevant environment, meeting predicted performance. Operational environment performance predicted. Prototype implementations developed.
6	System/sub-system model or prototype demonstration in an operational environment	Prototype implementations of the software demonstrated on full-scale realistic problems. Partially integrate with existing hardware/software systems. Limited documentation available. Engineering feasibility fully demonstrated.
7	System prototype demonstration in operational environment	Prototype software exists having all key functionality available for demonstration and test. Well integrated with operational hardware/software systems demonstrating operational feasibility. Most software bugs removed. Limited documentation available.
8	Actual system completed and "flight qualified" through test and demonstration.	All software has been thoroughly debugged and fully integrated with all operational hardware and software systems. All user documentation, training documentation, and maintenance documentation completed. All functionality successfully demonstrated in simulated operational scenarios. Verification and Validation (V&V) completed.
9	Actual system flight proven through successful mission operation	All software has been thoroughly debugged and fully integrated with all operational hardware/software systems. All documentation has been completed. Sustaining software engineering support is in place. System has been successfully operated in the operational environment.

Aurora

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101004291