European Space Agency
Directorate of Technical and Operational Support

STATEMENT OF WORK AND TECHNICAL REQUIREMENTS

Appendix 1 to AO/1-5184/06/NL/JD

On-Board Model Checking

Reference: TRP EME-019

Issue: 1.1

Date: 27 July 2006

TABLE OF CONTENTS

1 INTRODUCTION

1.1 Scope of document

This document describes the activity to be executed and the deliverables required by the European Space Agency in relation to the research and development activity *On-Board Model Checking*.

This document will be part of the Contract and shall serve as an applicable document throughout the execution of the work, with amendments as agreed at the kick-off meeting, if appropriate. It is organized as follows. Section 2 presents the background and the objectives of the activity. Section 3 presents in more detail the execution of the activity in providing a detailed description of the tasks. Section 4 lists management requirements and deliverables. Section 5 specifies schedule and milestones. To complete this document, Appendices A and B respectively provide the technical background of this study and the tailoring of the ECSS Software Standard (ECSS-E-40 Part 1B).

1.2 Applicable and reference documents

1.2.1 Applicable documents

The following documents contain requirements applicable to the activity. They are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

[E-40 Part 1B] ECSS-E-40 Part 1B – Space engineering – Software – Part 1: Principles and requirements, European Cooperation for Space Standardization (ECSS), ESA Publications, Noordwijk, The Netherlands, 28 November 2003, as tailored per Annex B.
ftp://ftp.estec.esa.nl/pub/wm/wme/ecss/ECSS-E-40Part1B(28Nov2003).pdf

[E-40 Part 2B] ECSS-E-40 Part 2B – Space engineering – Software – Part 2: Document requirements definitions (DRDs), European Cooperation for Space Standardization (ECSS), ESA Publications, Noordwijk, The Netherlands, 31 March 2005.
ftp://ftp.estec.esa.nl/pub/wm/wme/ecss/ECSS-E-40Part2B(31March2005).pdf

1.2.2 Reference documents

The Contractor can consult the following documents as they contain relevant information.

ECSS Standards

[P-001B] ECSS-P001B – Glossary of terms, European Cooperation for Space Standardization (ECSS), ESA Publications, Noordwijk, The Netherlands, 14 July 2004. http://www.ecss.nl/

[E-70-11A] ECSS-E-70-11A – Space engineering – Space segment operability, European Cooperation for Space Standardization (ECSS), ESA Publications, Noordwijk, The Netherlands, 05 Aug. 2005. http://www.ecss.nl/

[Q-30B] ECSS-Q-30B – Space product assurance – Dependability, European Cooperation for Space Standardization (ECSS), ESA Publications, Noordwijk, The Netherlands, 08 Mar. 2002. http://www.ecss.nl/

[Q-80B] ECSS-Q-80B – Space product assurance – Software product assurance, European Cooperation for Space Standardization (ECSS), ESA Publications, Noordwijk, The Netherlands, 10 Oct. 2003. http://www.ecss.nl/

ESA Studies

[MMOPS] M. Woods *et al.*, Mars Mission On-Board Planner and Scheduler (MMOPS) – Summary Report, Issue 1, ESA Contract 17987/03/NL/SFe CCN1, 2006. ftp://ftp.estec.esa.nl/pub/wm/wme/obmc/MMOPS-SUMRPT.pdf

[MUROCO-II] K. Kapellos, Formal Robotic Mission Inspection and Debugging (MUROCO II) – Executive Summary, Issue 1, ESA Contract 17987/03/NL/SFe, 2005. ftp://ftp.estec.esa.nl/pub/wm/wme/obmc/MUROCO-TRA-EXSUM.pdf

[SPAAS] J.-P. Blanquard, Software Product Assurance for Autonomy on-board Spacecraft (SPAAS), Final Report, Issue 1, ESA Contract 14898/01/NL/JA, 2004. ftp://ftp.estec.esa.nl/pub/wm/wme/obmc/SPAAS-FRP.pdf

Other Documents

[AADL] Society for Automotive Engineers (SAE), Architecture Analysis and Design Language (AADL), Standard Document AS-5506, Nov. 2004. http://www.aadl.info/

[Amla05] N. Amla, *et al.*, An Analysis of SAT-Based Model Checking Techniques in an Industrial Environment, In Proc. of 13th Advanced Research Working Conference on Correct Hardware Design and Verification Methods (CHARME 2005), Vol. 3725 of LNCS, pages 254-268, 2005. Springer. http://www.cadence.com/company/cadence_labs/kuehl_CHARME_2005_Analysis.pdf

[Benn05] M.B. Bennett, *et al.*, State-Based Models for Planning and Execution, In Proc. of the Plan Execution Workshop, 15th International Conference on Planning and Scheduling (ICAPS 2005), Monterey, USA, 2005. http://ic.arc.nasa.gov/people/sailesh/icaps2005wksp/ICAPS_Bennettfinal.pdf

[Bra05] G. Brad, *et al.*, Experimental Evaluation of Verification and Validation Tools on Martian Rover Software, In: Formal Methods in Systems Design Journal, Vol. 25, Sept. 2005. http://ase.arc.nasa.gov/visser/fmsdjournal.pdf

[Cima97] A. Cimatti, *et al.*, Planning via Model Checking: A Decision Procedure for AR, In Proc. of the 4th European Conference on Planning (ECP'97), Vol. 1348 of LNAI, pages 130-142, Toulouse, France, Sept. 1997. Springer.
http://sra.itc.it/people/leaf/ecp97.ps.gz

[Cima98] A. Cimatti, *et al.*, Strong Planning in Non-Deterministic Domains via Model Checking, In Proc. of the International Conference on AI Planning Systems (AIPS), pages 36-43. Pittsburgh, USA, 1998.
http://sra.itc.it/people/cimatti/papers/AIPS98.ps.gz

[Fea04] M.S. Feather, *et al.*, Planning for V&V of the Mars Science Laboratory Rover Software, In Proc. of the IEEE Aerospace Conference, Big Sky, USA, Mar. 2004.
http://eis.jpl.nasa.gov/~mfeather/Publications/2004-IEEE-Aero-Feather-Fesq-Ingham-Klein-Nelson.pdf

[Fesq02] L. Fesq, *et al.*, Model-Based Autonomy for the Next Generation of Robotic Spacecraft, In Proc. of the 53rd International Astronautical Congress, International Astronautical Federation (IAC-02), 2002.
http://mers.csail.mit.edu/papers/IAC02_MIT_paper.pdf

[Gat97] E. Gat, On Three-Layer Architectures, In: Artificial Intelligence and Mobile Robots, pages 195-210, 1997. MIT/AAAI Press.
http://robotics.usc.edu/~maja/teaching/cs584/papers/tla.pdf

[Ghal01] M. Ghallab, *et al.*, Architecture and Tools for Autonomy in Space, In Proc. of the 6th International Symposium on Artificial Intelligence and Robotics & Automation in Space, St-Hubert, Canada, 2001.
http://www.laas.fr/~felix/publis/pdf/isairas01.pdf

[Giu99] F. Giunchiglia and P. Traverso, Planning as Model Checking, In Proc. of the 5th European Conference on Planning: Recent Advances in AI Planning, pages 1-20, Sept. 1999.
http://citeseer.ifi.unizh.ch/rd/98805668%2C402497%2C1%2C0.25%2CDownload/http%3AqSqqSqwww.informatik.uni-ulm.deqSqkiqSqbiundoqSqECP-PapersqSqinvited-giunchiglia.ps.gz

[Hayd04] S.C. Hayden, *et al.*, Advanced Diagnostic System on Earth Observing One, In Proc. of AIAA Space 2004, San Diego, USA, 2004.
http://ic.arc.nasa.gov/projects/mba/abstracts/AIAASpace2004.pdf

[Kur98] J. Kurien, *et al.*, Model-Based Autonomy for Robust Mars Operations, In Proc. of the 1st International Conference of the Mars Society, Aug. 1998.
http://mers.csail.mit.edu/papers/mba-mars.pdf

[L2] Livingstone (L2): A Model-Based Reactive Self-Configuring System.
http://ic-www.arc.nasa.gov/projects/L2/doc/

[MBP] Model-Based Planner.
http://sra.itc.it/tools/mbp/

[Musc98] N. Muscettola, HSTS: Integrating Planning and Scheduling, In: Intelligent Scheduling, pages 169-212, 1998. Morgan Kaufmann.
http://www.ri.cmu.edu/pub_files/pub3/muscettola_nicola_1993_1/muscettola_nicola_1993_1.pdf

[ORCCAD] Open Robot Controller Computer Aided Design.
http://sed.inrialpes.fr/Orccad/

[Pec06] C. Pecheur, *et al.*, Formal Verification of Autonomy Models: From Livingstone to SMV. In: Agent Technology from a Formal Perspective, NASA Monographs in Systems and Software Engineering, 2006. Springer.
http://www.info.ucl.ac.be/~pecheur/publi/Livingstone2smv.ps

[Pel97] B. Pell, *et al.*, An autonomous Spacecraft Agent Prototype, In Proc. of the 1st International Conference on Autonomous Agents, pages 253-261, Marina del Rey, USA, Feb. 1997.
http://mers.csail.mit.edu/papers/agents97.pdf

[Pist01] M. Pistore and P. Traverso, Planning as Model-Checking for Extended Goals in Non-Deterministic Domains, In Proc. of the International Joint Conference on Artificial Intelligence (IJCAI-01), pp. 479-484, 2001.
http://sra.itc.it/tr/PT01.pdf

[Sher05] R. Sherwood, *et al.*, Intelligent Systems in Space: The EO-1 Autonomous Sciencecraft Experiment, In Proc. of the 2005 AIAA Infotech@Aerospace Conference, Arlington, USA, Sept. 2005.
http://pearljam.jpl.nasa.gov/sherwood/papers/AIAAInfotechfinal.pdf

[SRA] Automated Reasoning System Department of ITC-IRST.
http://sra.itc.it/

[Stro02] A.W. Stroupe, *et al.*, Technology for Autonomous Space Systems, The Robotics Institute, Carnegie Mellon University, 2002.
http://www.ri.cmu.edu/pub_files/pub3/stroupe_ashley_2001_1/stroupe_ashley_2001_1.pdf

[Vis03] W. Visser, *et al.*, Model Checking Programs, Automated Software Engineering Journal, 10(2), Apr. 2003.
http://ase.arc.nasa.gov/visser/ase00FinalJournal.pdf

[Weld99] D.S. Weld, Recent Advances in AI Planning, AI Magazine, 20(2), 1999.
http://www.cs.washington.edu/homes/weld/papers/pi2.pdf

[Will96] B.C. Williams and P. Pandurang Nayak, A Model-Based Approach to Reactive Self-Configuring Systems, In Proc. of the 13th National Conference on Artificial Intelligence, Portland, USA, Aug. 1996.
http://www.qrg.northwestern.edu/papers/Files/qr-workshops/QR96/Williams_1996_Model-Based_Approach_Self-Configuring_Systems.pdf

2 BACKGROUND AND ACTIVITY OBJECTIVES

2.1 Background

Deep space and remote planetary exploration missions are characterized by severely constrained communication links, which are limited in frequency and data transmission rate. These constraints are not in favour of maintaining adequate real-time communications so that ground operators can receive up-to-date telemetry from a remote spacecraft and react to any potential unforeseen interaction with its environment or equipment failure by sending

telecommands to be executed on-board in real-time. To tackle these constraints, the actual technical solutions proposed are to increase the intelligence on-board in such a way that a system can take autonomous decisions according to unexpected events or anomalous conditions. Changes of environmental conditions and failure of computing resources are examples that could require the adoption of fast and autonomous decisions to avoid situations identified as risky. However, an autonomous decision must be the result of the analysis of a high amount of complex parameters in a short time and should not lead to wrong decisions, i.e. in amplifying a risk or even creating new risks.

In the case of classical (i.e. non-autonomous) systems, telecommands are first executed on ground simulators representing the actual status of the system. Reactions to the telecommands are then analyzed and, if they are as expected, telecommands are considered safe and are sent to the system. This heavy process implies long reaction times that may moreover be increased by poor communication links. This process requires human operators and is clearly inefficient if it can take days before a problem is discovered and many more before corrective telecommands are executed on-board. Even under nominal conditions, the process of planning operations on the ground is manually intensive and time-consuming. Furthermore, without access to live data, decisions may be based on obsolete data, which could endanger the system, even if safing procedures are strictly adhered to. Moreover, autonomy is also a major cost driver since human controlled missions require large earth-based teams and facilities for support.

Providing remote systems with the ability to create their own plans based on up-to-date information and more importantly enabling them to re-plan in response to dynamic events would greatly improve the efficiency of a mission and potentially improve the safety of systems. Ground operators can use the restricted communication link to forward high-level mission objectives, which the on-board system can turn into detailed commands. Execution can be monitored continuously and re-planning invoked when any execution problem occurs.

In applications where a certain amount of autonomy is required (see the ECSS mission execution autonomy levels [E-70-11A]), re-planning is the result of a complex analysis of detailed information. Ensuring that resulting decisions will not endanger the system is also a complex affair. If the problem can be tackled in different ways, it is believed that model-checking techniques applied on models representing the behaviour of a spacecraft could be more than valuable to analyse just in time the effects of decisions before they are actually executed. This approach can moreover be used on each layer of the three-layer autonomy architecture (see Annex A) depending on the level of abstraction of the models.

2.2 Activity objectives

To reach the goal of future missions and increase their scientific return, space systems must be designed in such a way that they can react quickly to the environment (e.g. faulty conditions, mission opportunities) by taking appropriate and fast decisions. This capacity of autonomy can be defined as the ability of taking actions that were previously dedicated to ground control centres. This study has the objective of demonstrating that innovative technology (i.e. model-based autonomy, planning as model-checking) may be used to increase the number of autonomous functions on-board space systems in focusing on on-board planning and health monitoring.

This general objective may be divided into the following sub-objectives:

1. To find out and justify the place of model-checking in on-board space systems in order to increase their degree of autonomy as well as their degree of confidence;
2. To develop a software prototype, called Autonomous Reasoning Engine (ARE), that could be seen as a building block for future space missions;
3. To demonstrate the global approach on a case study and provide experimental results;
4. To conclude on the adequacy of the approach with respect to the peculiarities of the space environment.

More precisely, this study aims at developing a demonstrator and a proof of concept case study for the basic central element of an autonomous spacecraft, which would provide on-board real-time reasoning facilities for the model-based autonomy. The model-based reasoning will employ model-checking techniques for dynamic re-planning, dynamic reconfiguration and fault management. The same ARE will be used for reasoning on different levels of abstraction, implying a unique modelling formalism.

3 ACTIVITY DESCRIPTION

3.1 Work logic

The work to be carried out in this project shall be composed of the tasks mentioned hereafter and detailed in the next sections:

1. Production of the Requirements Baseline (RB)
2. Production of the Technical Specification (TS)
3. Design, Coding, Verification and Validation Against the TS and RB
4. Performance Evaluation and Characterisation of the Approach

This decomposition of tasks reflects the software development process as described in the ECSS Software Standard [E-40 Part 1B]. Figure 1 depicts the different parts and tasks of the study.

Part 1	***Production of the Requirements Baseline*** TASK 11: Synthesis on Autonomy Needs and Potential Solutions TASK 12: Requirements Baseline Elicitation
Part 2	***Production of the Technical Specification*** TASK 21: Software Specification Elicitation TASK 22: Architectural Model Description
Part 3	***Implementation of the Autonomous Reasoning Engine*** TASK 31: Detailed Design, Coding, and Verification (Testing) TASK 32: Validation Against the TS and RB
Part 4	***Performance Evaluation*** TASK 41: Evaluation of the Approach on a Case Study TASK 42: Characterisation of the Approach

Figure 1. Work logic summary

3.2 Production of the Requirements Baseline

This part consists in investigating the state of the art in terms of autonomy for space software systems. This investigation shall be based on innovative techniques (model-checking and model-based autonomy) that can be used to increase the level of autonomy of such systems in maintaining a high level of confidence in such systems to fulfil their mission with limited human interactions.

3.2.1 TASK 11: Synthesis on Autonomy Needs and Potential Solutions

This task is the starting point of the study. It shall provide a detailed overview of the state of the art in terms of on-board software architecture for autonomous systems. It will concentrate on the autonomous spacecraft, including planetary rovers, organized around a three-layer architecture (decision layer, executive control layer, and functional layer), and will analyse in particular the interest of basing the reasoning phase of autonomous decisions and diagnosis on models. It will also propose a selection of technology (based on model-checking) to constitute the core of prototype software, subject of the study, aiming at producing autonomous reasoning (e.g. for diagnosis). This task shall also survey the different formalism on which the Autonomous Reasoning Engine can be based, and select or define one to be used by the engine. It shall also survey and propose a list of possible case studies, from which at least one can be used during the performance evaluation (Task 41).

A milestone meeting (M1) with the Agency will decide on the proposed technology.

Inputs: This statement of work, Contractor's own knowledge, applicable and reference documents (Section 1.2), and other information gathered from any other source.

Outputs: A technical note detailing the state of the art in autonomy software architecture, the technology choice as basis of the study, and possible case studies.

3.2.2 TASK 12: Requirement Baseline Elicitation

This task shall be dedicated to the elicitation of the system requirements related to software for the Autonomous Reasoning Engine. This activity assumes that a high-level avionics architecture has been defined. In order to put the building block in context, a draft AADL (Architecture Analysis and Design Language) [AADL] model of the avionics and software shall be produced, reflecting possible hardware architecture in line with the autonomy objectives, and the other main software blocks.

As the intended use of the building block span over various types of spacecrafts, some requirements shall be expressed in a generic way, with some parameters to be instantiated in the real project. The requirement shall also indicate the selected value for the implementation.

This task shall be concluded by the System Requirements Review (SRR).

Inputs: This statement of work, Contractor's own knowledge, applicable and reference documents (Section 1.2), output of TASK 11 and other information gathered from any other source.

Outputs: The Requirement Baseline.

The avionics and software draft AADL model.

3.3 Production of the Technical Specification

This part will focus on the software specification and the architecture, as in classical software development.

3.3.1 TASK 21: Software Specification Elicitation

This task shall produce the software specification, derived from the requirement baseline.

Inputs: This statement of work, Contractor's own knowledge, applicable and reference documents (Section 1.2), outputs from TASK 11 and TASK 12, and other information gathered from any other source.

Outputs: A specification document detailing the architecture and design choices leading to the selection of existing technologies and software components.

3.3.2 TASK 22: Architectural Model

This task shall produce the architecture. It shall be in the form of a UML model. The genericity defined in the requirements shall be implemented in UML by using its capability of abstraction (e.g. abstract interface, inheritance).

This task shall be concluded by the Preliminary Design Review (PDR).

Inputs: This statement of work, Contractor's own knowledge, applicable and reference documents (Section 1.2), output of TASK 21 and other information gathered from any other source.

Outputs: The UML model of the prototype software

3.4 Implementation of the Autonomous Reasoning Engine

3.4.1 TASK 31: Detailed Design, Coding, and Verification (Testing)

This task shall be devoted to the development of the software building block. This includes:

- The detailed design, as a refinement of the UML model, but without the generic elements. The default values introduced in the specification will be used;
- The coding, in language of choice, traced to the model, possibly using an automatic code generator;
- The unit testing and integration testing (informal).

This task shall be concluded with the Critical Design Review (CDR).

Inputs: This statement of work, Contractor's own knowledge, applicable and reference documents (Section 1.2), output of TASK 21, TASK 22 and other information gathered from any other source.

Outputs: Detailed design, code, and verification test report

3.4.2 TASK 32: Validation against the TS and RB

This task shall be devoted to the validation against the Technical Specification and Requirement Baseline.

This task shall be concluded with the Critical Design Review (CDR).

Inputs: This statement of work, Contractor's own knowledge, applicable and reference documents (Section 1.2), output of TASK 21, TASK 22, TASK 31 and other information gathered from any other source.

Outputs: The final product and associated documentation.

3.5 Performance Evaluation

3.5.1 TASK 41: Evaluation of the Approach on a Case Study

This task shall be devoted to the empirical evaluation of the approach on a case study representative of the space domain (e.g. by software simulation of a planetary rover or orbiting spacecraft). The evaluation should ideally be achieved on two case studies implementing different requirements and constraints.

This task shall be concluded with the Acceptance Review (AR).

Inputs: This statement of work, Contractor's own knowledge, applicable and reference documents (Section 1.2), output of TASK 21, TASK 22, TASK 31, TASK 32 and other information gathered from any other source.

Outputs: Technical report presenting the results of the evaluation.

3.5.2 TASK 42: Characterisation of the Approach

This task shall be devoted to the characterisation of the approach in focusing on several parameters (reliability, availability, performances including processing power and memory required) and in concluding on an adequate software and hardware architecture to support such autonomous reasoning engine.

Inputs: This statement of work, Contractor's own knowledge, applicable and reference documents (Section 1.2), output of TASK 21, TASK 22, TASK 31, TASK 32, TASK 41 and other information gathered from any other source.

Outputs: Technical report presenting the characterization of the global approach and the adopted software architecture.

4 MANAGEMENT, REPORTING, MEETINGS AND DELIVERABLES

The standard requirements for management, reporting, meetings and deliverables (Appendix 2 to the Contract) shall apply to the present activity, taking into account the following specific requirements, which shall prevail in case of conflict.

Note: the numbering below refers to the numbering of the said Appendix 2.

4.1 Management

4.2 Reporting

The Contractor shall provide, every month, a progress report covering the work executed. This report shall give a description of progress, reasons for potential slippages and corrective actions, events to be accomplished during the next reporting period, expected dates for major schedule items, and an updated schedule.

The Contractor shall notify the Agency's representatives (Technical Officer and Contracts Officer) of any event likely to cause major delays to the time schedule of the work programme or significantly impact the scope of the work to be performed.

As soon as they become available and always within the time frame agreed by the Agency, the Contractor shall submit for the Agency's approval all technical notes, which are produced during the execution of the Contract. Any technical documentation to be discussed at a meeting with the Agency shall be submitted at least two weeks prior to such a meeting.

Note that all documents mentioned shall also be delivered as an electronic file.

4.3 Meetings

The Agency intends to monitor the execution of the Contract through dedicated meetings: the Kick-Off Meeting, Progress Meetings, and a Final Presentation.

The Kick-Off meeting has to be considered as the first event in the project. It will occur after a negotiation meeting to approve formally the technical baseline.

Progress meetings will take place at a frequency to be determined by the Agency. That frequency could be changed if difficulties occur during the Contract requiring further discussions. It shall be possible to arrange progress meetings at the Contractor's premises when/if required.

The objective of each meeting is to assess the results of the technical efforts, to assess them for completeness, correctness and compliance with the requirements and to verify the achievement of the objectives. Shortcomings, problems, corrective actions, and potential changes will be identified and formally addressed.

A final presentation will be scheduled after formal approval of all deliverables. The Contractor shall make a presentation summarising all the main activities and achievements made during the Contract. All the deliverable items shall be available to ESA within the date of the final presentation.

Additional meetings are not excluded, and either the Agency or the Contractor may request ad hoc meetings.

The Contractor is responsible for the preparation and distribution of minutes of all meetings held in connection with the Contract. The minutes shall clearly identify all agreements made and actions accepted at the meeting together with an update of the action item list and the document list. A draft shall be signed at the end of every meeting.

4.4 Deliverables

The following is applicable to the deliverables. The numbering of the paragraphs refers to the one in the Standard Requirements for Management, Reporting, Meetings and Deliverables (Appendix 2 to the Contract).

Documentation:	Paragraph 4.1.1 (only final report and limited to a maximum of 50 pages) shall apply.
	Paragraph 4.1.2 (only technical data package) shall apply.
Abstract:	Paragraph 4.2 shall apply.
Photographic documentation:	Paragraph 4.3 shall not apply.
Hardware:	Paragraph 4.4 shall not apply.
Computer programmes:	Paragraph 4.3 shall apply. All software and models produced or procured under the Contract shall be delivered in source code.

4.5 Commercial Evaluation

Paragraph 5 of the Standard Requirements for Management, Reporting, Meetings and Deliverables (Appendix 2 to the Contract) shall not apply.

5 SCHEDULE AND MILESTONES

The duration of the activity shall not exceed eighteen months.

The following meetings are foreseen and are given here as indication. The Contractor shall propose a schedule and corresponding milestones.

Meeting	Date	Location
Kick-Off	T0	ESA/ESTEC
Milestone 1	End of TASK 11	Contractor's premises
System Requirement Review	End of TASK 12	Contractor's premises
Preliminary Design Review	End of TASK 22	Contractor's premises
Critical Design Review	End of TASK 31	ESA/ESTEC
Delivery, Installation and Acceptance Review	At completion of TASK 42	ESA/ESTEC
Final Presentation	End of Contract	ESA/ESTEC

In addition, progress meetings shall be organised every two months at the Contractor's premises.

ANNEX A: TECHNICAL BACKGROUND AND REQUIREMENTS

A.1 Acronyms

The following acronyms are used or are relevant in this document.

Acronyms	Description
AADL	Architecture Analysis and Design Language
AR	Acceptance Review
ARE	Autonomous Reasoning Engine
BMC	Bounded Model Checker
CDR	Critical Design Review
COTS	Commercial-Off-The-Shelf
DDF	Design Definition File
DJF	Design Justification File
DRD	Document Requirement Description
ECSS	European Cooperation for Space Standardization
ESA	European Space Agency
ESTEC	European Space research and Technology Centre
FDIR	Fault Detection Identification and Recovery
IPR	Intellectual Property Rights
MF	Maintenance File
MGT	Management File
MMI	Man-Machine Interface
MOTS	Modified-Off-The-Shelf
OP	Operational Documentation
PAF	Product Assurance File
PDR	Preliminary Design Review
QR	Qualification Review
RB	Requirement Baseline
SAT	Propositional Satisfiability
SDP	Software Development Plan
SLC	Software Life Cycle
SRR	System Requirement Review
SUM	Software User Manual
SVTS	Software Validation Testing Specification
TS	Technical Specification
UML	Unified Modelling Language

A.2 Technical background

A.2.1 Background overview

This study takes as basis a generic three-layer hybrid autonomy architecture (e.g. [Stro02], [Gat97], [Ghal01], [ORCCAD]) composed of:

- A *Decision (Deliberative) Layer*, with goal-driven planning and scheduling facilities;
- An *Executive (Execution Control) Layer*, with execution sequencing facilities;
- A *Control (Functional) Layer*, with reactive execution facilities.

Autonomous planning and scheduling activities for the space domain applications have historically been centred on the constraint-based planning approaches combined with heuristic searches on temporal databases for scheduling purposes [Pel97]. Integration of planning and scheduling into a single decision deliberation engine has been studied and applied to various degrees of autonomy on the Hubble Space Telescope and on the Deep Space One spacecraft ([Musc98], [Pel97]). In order to execute the produced plans in a dynamic environment and possible system faults more reactive approaches were evaluated for the Executive and partly for the Control Layers. The most established of these approaches is model-based autonomy [Fesq02]. It provides a potential for integrating the goal-driven operation of a spacecraft with the fault management capabilities, hence facilitating a high level of autonomous operation in the presence of faults and in the partially unknown dynamic conditions of the operational environment. This approach has been implemented in the Livingstone system, which was successfully employed in the Deep Space One spacecraft for the Remote Agent autonomy experiment [Will96]. The second, more advanced version of the Livingstone system, Livingstone 2 [L2], has been used in the Autonomous Sciencecraft Experiment (ASE) on board the Earth Observing One satellite [Hayd04], [Sher05].

In this autonomy setting the Decision, Executive and Control Layers' operation is based on the different formalisms specific to the autonomy approach chosen for the layer. The Decision Layer is built upon a constraint-based formalism with utilisation of the operational and temporal constraints. This constitutes a Constraint Satisfaction Problem, which, in general, is NP-complete and requires combinatorial search approaches in combination with heuristics in order to find a solution within reasonable time limits. The Executive Layer uses a model-based approach utilising an automaton representation of the domain models. The Control (Functional) Layer mostly uses control laws for the feedback control loop algorithms. Capturing the system model in terms of the transition system formalism provides possibility for application of the state exploration techniques (e.g. model checking).

This approach to realisation of on-board autonomy architecture poses additional challenges. Different layers are based on different formalisms, which complicates ensuring consistency between the sets of constraints used by different Layers. Secondly, the planning and scheduling activities of the Decision Layer must take into account the behaviour of the Executive Layer, which is based on different mechanisms [Benn05].

In recent years a broad research has been conducted on expressing the Constraint Satisfaction Problems in terms of transition system formalism and use of model-checking principles and techniques for the planning purposes [Giu99], [Pist01], [Nau04]. This approach can open a possibility of employing the model checking techniques in the Decision, Executive, and possibly Control (Functional) Layers of the on-board autonomy. It can facilitate the alleviation of the multi-formalism matters and allow for a coherent set of models and decision

procedures throughout the on-board autonomy realisation. Related work with regard to the tighter integration of the different autonomy layers based on the state-based modelling has been performed by the Jet Propulsion Laboratory, NASA and resulted in the creation of the Mission Data System (MDS) [Benn05]. Research on the model-based planning is being performed by the Automated Reasoning System Division of ITC-IRST [SRA], where the Model Based Planner has been designed [MBP].

New reasoning mechanisms based on the model checking approaches are also subject of the extensive Research & Development activities being performed [Weld99]. These techniques include model checkers, bounded model checkers (BMCs), and various SAT solvers. BMCs limit the state space explored by the algorithm based on some additional constraints or conditions (e.g. transition probabilities). The SAT solvers perform systematic or stochastic search of a state space with respect to the propositional satisfiability of the requested property. The latest developments in this area show fast, compact, and potentially embeddable techniques [Weld99], [Amla05].

A.2.2 Envisaged solution

Previous ESA activities have addressed some components of these layers:

- The Formal Robotic Mission Inspection and Debugging [MUROCO-II], in which model checking techniques have been applied to the on-ground specification and verification of the Executive Layer functionality;
- The Mars Mission On-Board Planner and Scheduler [MMOPS], in which the on-ground planning is complemented with the on-board Timeline Validation, Control and Repair capability

While these activities have addressed several aspects of on-board autonomy, the proposed activity will consider new approaches to the underlying autonomy principles, basis of the autonomy concepts behind the mentioned architectural autonomy layers, and a unified model-based reasoning approach to the overall on-board autonomy operation.

This activity will be concerned with the development of an Autonomous Reasoning Engine (ARE). The ARE will provide the model-based decision logic and procedures for all the autonomy layers (excluding the feedback control loop algorithms of the Control (Functional) Layer). The ARE will operate on the layer-correspondent domain model. In addition to the model, the sets of the corresponding constraints and current conditions will be provided to the reasoning engine (e.g. general system constraints, current system (health) status, observable status of the environment). On the level of the Decision Layer the ARE will take the mission goals (e.g. and high-level mission scenarios) and will produce a plan (an execution scenario), which in turn will be used as input to the Executive Layer. On the level of the Executive Layer the ARE will search for the execution sequences that satisfy the plan and take into account corresponding constraints and current failure conditions (comparable to the Livingstone system). Possibly similar to the Executive Layer and in a similar way some low-level reasoning can be performed by the ARE at the level of the Control (Functional) Layer. The layered reasoning scheme to produce the system execution commands may be seen as a scenario refinement at each level. Every Layer operates in accordance with the granularity level of its scenario steps. At every step of the corresponding scenario at each level the remainder of the scenario (or plan) is evaluated for executability taking into consideration possible changes in the system status and conditions of the operational environment caused by the execution of the previous step. The reconfiguration or re-planning will be performed on

the corresponding levels as necessary. If a Layer is unable to satisfy the execution of a scenario step it will request a re-planning from the Layer above it. In order to implement such a scheme the domain models will use modelling language suitable for modelling on different abstraction levels correspondent to the levels of the Autonomy Layers and will be based on a formalism suited for the application of the model checking techniques.

The ARE will represent an abstract reasoning engine, independent of the Layer it is used in, making it highly reusable module. The reasoning level will be defined by the model level used. The ARE will implement the (bounded) model checking techniques for its operational purposes. Dependent on the Layer the ARE will be used in possibly different search algorithm approaches will be employed (e.g. bounded (probabilistic) model checking, stochastic SAT solver, systematic SAT solver, etc.). While more demanding and slower algorithms may be employed for the Decision Layer, more reactive and faster algorithms should be used for the lower Layers. The use the different algorithms may be made configurable. Use of the combination of several algorithms for the planning optimisation (Decision Layer) will be evaluated.

The reliability of the ARE operation in presence of the computer memory faults (radiation effects) and approaches for the ARE self-fault tolerance will be evaluated.

ANNEX B: ECSS-E-40 TAILORING

B.1 Introduction

ECSS-E-40 (Space Engineering – Software) has replaced the PSS-05 for the development of new space software, that is software involved in the production of space systems. ECSS-E-40 has the same goal as its PSS-05 predecessor, which is to assist developers in applying good software practices during the development. Compared to PSS-05, however, ECSS-E-40 allows for more flexibility in that:

- The standard encompasses a set of software processes without prescribing any specific life cycle.
- Each software process terminates with reviews that directly tie with those of a satellite development, so that the former explicitly contribute to the progress of the latter.
- Each software process releases descriptive information, not necessarily a set of documents with prescribed table of contents. The Contractor may place and organise the required information in whatever form they may choose to. The Contractor is able to apply their specific development methodology, as long as that satisfied the ECSS-E-40 process requirements.
- The ECSS-E-40 standard requirements must be tailored and adjusted to the specific needs, the costs and risks of the project.

This annex specifically addresses the last item of the above list. The baseline version ECSS-E-40 Part 1B (from 28/11/2003) is made up of several sections, of which only Section 5 and 6 express requirements.

B.2 Description of the tailoring

B.2.1 Project characteristics

This research & development project concerns the development of a prototype for a future on-board software building block. The qualification as building block will consist in a refinement of the functionalities and a delta testing. Therefore the prototype must be already developed in a proper way to minimize the future effort.

B.2.2 Project risks

Hereafter, Table 1 gives a rough estimation of the magnitude of the most encountered risk in software development. Most specifically for this project, the table states on the estimated risk concerning the potential lack of real-time performances and computers resources.

Table 1. Risk Description

Risk description	Risk magnitude		
	LOW	**MEDIUM**	**HIGH**
1. Complex specification			X
2. Tricky design		X	
3. Reliability critical		X	
4. Safety critical	X		
5. Long term use		X	
6. Supplier's background and maturity			X
7. Potential lack of computers resources (processing time & memory)		X	
8. Potential lack of real-time performances	X		
9. (Effort for providing) assessment of X-bility drivers (e.g. flexibility, modularity, reprogrammability)		X	

B.2.3 Roles

Roles are as described hereafter:

- The customer is **ESA**.
- The Supplier is the **Contractor**.
- The User is **ESA**.
- The maintainer is the **Contractor**, only during the warranty period for corrective maintenance.
- There is **no** operator.

B.2.4 Processes involved

The following software processes are part of this project:

- The system engineering processes related to software for the establishment of the requirement baseline
- The software requirement and architecture engineering process, which describes the software requirements specification, the architectural design and provides the UML model of the reasoning engine
- The software design and implementation engineering process, where the software is detailed designed, coded and validated against the supplier's specification. Note that in this project the validation activity with respect to the RB-TS will be performed in the frame of the software validation and acceptance process;

- The software validation process, where the software is validated against the system requirements document (requirement baseline), delivered to and accepted by the customer.
- The software verification process, at least for the establishment of the RB/TS – SVTS traceability matrices, for the follow of the computer resources in terms of sizing and timing and the validation of the foreseen scheduling model
- The software delivery and acceptance process
- The software management process as described in the draft development plan (including organisation breakdown structure, work breakdown structure, life cycle, development methods and tools, reused software products, documentation to be produced, risk management, milestones, deliveries).

The following software processes are not part of this project:

- The software operation process since no helpdesk is necessary to operate this software;
- The software maintenance process since this Contract covers only the development activities to be performed (except during the warranty period during which some corrective maintenance activities can be requested by the customer).

B.3 Software process mapping to work packages

The software development processes introduced in ECSS-E-40 Part 1B are mapped on the work packages and activities of the Statement of Work in the following way.

Table 2. ECSS-E-40B Software Activities

ECSS-E-40B Software activities	Reference in the Statement of Work
System engineering related to software, Requirement baseline	**TASK 11**: Synthesis on autonomy needs and possible solutions **TASK 12**: Requirement Baseline elicitation
System Requirements Review **(SRR)**	At completion of **TASK 12**
Software Requirements, Top Level Architecture, Technical Specification	**TASK 21**: Software specification elicitation **TASK 22**: Architectural Model
Software Preliminary Design Review **(PDR)**	At completion of **TASK 22**
Design, Code, Unit Tests, Integration Tests, Validation Against the TS	**TASK 31**: Design & implementation with minimal effort of an automated testing prototype tool
Software Critical Design Review **(CDR)**	At completion of **TASK 31 and TASK 32**
Validation Against the Technical Specification	**TASK 32**: Validation against the Technical Specification and the Requirement Baseline
Validation Against the Requirement Baseline	
Software Qualification Review **(QR)**	None - with CDR (RB-TS validation)

ECSS-E-40B Software activities	Reference in the Statement of Work
Delivery and Acceptance	Implicitly included in **TASK 41 and TASK 42**
Software Acceptance Review **(AR)**	At completion of **TASK 41**
Software Operation	None
Software Maintenance	None (except corrective maintenance activities during the warranty period)
Project Management	As described in Section 4 dealing with "Management, reporting, meetings and deliverables"
Verification Activities	RB/TS-SVTS traceability matrices Sizing and timing budget follow up Verification of software requirements through the tasks defined in Part 1

B.4 List of ECSS-E-40 applicable requirements

The following table draws the ECSS-E-40 requirements that are applicable to this project in making a distinction between the middleware layer and the application layer. In some cases, there is no added value in considering both separately, and the applicable requirements are given for the overall product.

Table 3. ECSS-E-40 Part 1B Requirements

List of ECSS-E-40 Part 1B (28 November 2003) requirements		Applicable
5.2	System engineering processes related to software	
5.2.2.1	System requirements specification	Yes
5.2.2.2	System and functional criticality analysis	Yes
5.2.2.3	MMI software mock up requirements	No
5.2.2.4	MMI general requirements and guidelines	No
5.2.3.1a	System design	Yes
5.2.3.1b	System design to system requirements conformance	No
5.2.3.1c	System requirements to system design traceability	No
5.2.3.2a	Software-hardware interface requirements	Yes
5.2.3.2b	Traceability to system partitioning	No
5.2.3.2c	System partition with definition of items (HW, SW, human operation)	No
5.2.3.2d	System configuration items list	Yes
5.2.4.2	Qualification engineering requirements (verification & validation process requirements)	No
5.2.4.3	Software validation requirements at system level	No

List of ECSS-E-40 Part 1B (28 November 2003) requirements		Applicable
5.2.4.4	Requirement baseline verification	No
5.2.4.5	System Requirement Review (SRR)	Yes
5.2.5.1	Identification of observability requirements	Yes
5.2.5.2	Control and data interfaces for system level integration	No
5.2.5.3	Data medium requirements for integration	No
5.2.5.4	System database specification (content and use)	No
5.2.5.5	Identification of development constraints (to support the software integration into the system)	Yes
5.2.5.6	Definition of constraints for software to be reused	No
5.2.5.7	Identification of customer's input for software integration into the system	No
5.3.5.8	Identification of customer's output for software integration into the system	No
5.2.5.9	Planning of supplier support to system integration	No
5.2.6.1	Phasing and management / operational plan	No
5.3.6.2	System requirements definition for software operations	No
5.2.7.1	Software maintenance requirements	No
5.2.7.2	Definition of in-flight capabilities for flight software	No
5.4	Software requirements and architecture engineering process	
5.4.2.1	Establishment and documentation of software requirements / software requirements specification	Yes
5.4.2.1-a	Software requirements – functional and performance	Yes
5.4.2.1-b	Software requirements – quality requirements	No
5.4.2.1-c	Software requirements – security specifications	No
5.4.2.1-d	Software requirements – human factors - ergonomics specifications	No
5.4.2.1-e	Software requirements – data definition and database requirements	No
5.4.2.1-f	Software requirements – interfaces external to the software item	No
5.4.2.2	Definition of functional and performance requirements for in-flight modification	No
5.4.2.3	Identification of requirements unique identifier	Yes
5.4.2.4	Definition of a software logical model	Yes
5.4.2.5	Definition of a behavioural view	Yes
5.4.2.6	MMI software mock-up development	No
5.4.2.6a	MMI specifications	No
5.4.2.6b	MMI specifications or mock-up evaluation report	No
5.4.2.6c	End-users participation in the MMI mock-up evaluation	No
5.4.3.1	Transformation of software requirements into a software architecture	Yes
5.4.3.2	Software design description	Yes
5.4.3.3	Software design documentation	Yes
5.4.3.4	Software architectural design contents	No

List of ECSS-E-40 Part 1B (28 November 2003) requirements		Applicable
5.4.3.5	Software design method	No
5.4.3.6	Selection of a computational model for real time software	No
5.4.3.6a	Computational model	No
5.4.3.6b	Scheduling report	No
5.4.3.7	Description of software dynamic behaviour	No
5.4.3.8	Development and documentation of the software interfaces	Yes
5.4.3.9	Definition of methods and tools for software to be reused	No
5.4.3.10	Evaluation of potential reuse of software	No
5.4.3.11	Evaluation of reuse of pre-developed software	No
5.4.3.12	Analysis of potential reusability	No
5.4.3.13	Definition and documentation of the software integration requirements and plan	No
5.4.314	Conducting a Preliminary Design Review (PDR)	Yes
5.5	Software design and implementation engineering process	
5.5.2.1	Detailed design of each software components	Yes
5.5.2.2	Development and documentation of the software interface detailed design	Yes
5.5.2.3	Production of software items physical model	No
5.5.2.4	Utilization of method for software static design	No
5.5.2.5	Description of the software dynamic aspects of physical model for real-time software	No
5.5.2.5a	Dynamic physical model	No
5.5.2.5b	Scheduling simulation	No
5.5.2.6	Utilization of description techniques for the software behaviour	No
5.5.2.7	Determination of design methods consistency for real-time software	No
5.5.2.8	Development and documentation of the software user manual	Yes
5.5.2.9	Definition and documentation of the software unit test requirements and plan	No
5.5.2.10	Updating of the software integration requirements and plan	No
5.5.2.11	Conducting a Detailed Design Review (DDR) for flight software	No
5.5.3.1	Development and documentation of the software units, test procedures and test data	Yes
5.5.3.2	Software unit testing	No
5.5.3.3	Software user manual updating	Yes
5.5.3.4	Updating of the software integration test requirements and plan	No
5.5.4.1	Software integration test plan development	No
5.5.4.2	Software units and software components integration and testing	No
5.5.4.3	Software user manual updating	Yes

List of ECSS-E-40 Part 1B (28 November 2003) requirements		Applicable
5.6	Software validation process	
5.6.2.1	Determination of the validation effort	No
5.6.2.2	Establishment of a validation process	No
5.6.2.2a	Software validation plan – validation process identification	No
5.6.2.2b	Software validation plan – methods and tools	No
5.6.2.3	Selection of an ISVV organization	No
5.6.2.3a	Independent software validation plan – organization selection	No
5.6.2.3b	Independent software validation plan – level of independence	No
5.6.2.4	Development and documentation of a validation plan	No
5.6.3.1	Development and documentation of a software validation testing specification (SVTS) with respect to TS	Yes
5.6.3.2	Conducting the validation with respect to TS (combined with RB)	Yes
5.6.3.3	Updating the software user manual	Yes
5.6.3.4	Test Readiness Review (TRR)	No
5.6.3.5	Conducting a Critical Design Review (CDR)	Yes
5.6.4.1	Development and documentation of a software validation testing specification (SVTS) with respect to RB	Yes
5.6.4.2	Conducting the validation with respect to RB (combined with TS)	Yes
5.6.4.3	Updating the software user manual	Yes
5.6.4.4	Test Readiness Review (TRR)	No
5.6.4.5	Conducting a Qualification Review (QR)	No
5.7	Software delivery and acceptance process	
5.7.2.1	Preparation of the software product	Yes
5.7.2.2	Supplier's provision of training and support	No
5.7.2.3	Installation planning	No
5.7.2.4	Installation activities reporting	No
5.7.3.1	Acceptance test planning	Yes
5.7.3.2	Acceptance test execution	Yes
5.7.3.3	Executable code generation and installation	Yes
5.7.3.4a	Supplier's support to customer's acceptance	Yes
5.7.3.4b	Links with Software Product Assurance	No
5.7.3.4c	Acceptance testing documentation	Yes
5.7.3.5	Evaluation of acceptance testing	Yes
5.7.3.6	Conducting an Acceptance Review (AR)	Yes
5.8	Software verification process	
5.8.2.1	Determination of the verification effort for the project	No
5.8.2.2	Establishment of the verification process, methods and tools	No

List of ECSS-E-40 Part 1B (28 November 2003) requirements		Applicable
5.8.2.3	Selection of the organization responsible for conducting the verification	No
5.8.2.4	Development and documentation of a verification plan covering the software verification activities	No
5.8.3.1	Verification of software requirements	No
5.8.3.2	Verification of software architectural design	No
5.8.3.3	Verification of software detailed design	No
5.8.3.4	Verification of code	No
5.8.3.5	Verification of software integration	No
5.8.3.6	Verification of software documentation	No
5.8.3.7	Verification of test specifications	Yes
5.8.3.8	Verification of software validation with respect to TS and RB	No
5.8.3.9	Evaluation of validation: complementary system level validation	No
5.8.3.10a	Problem and non conformance handling / identification of problems during software verification process, software validation process and CDR	Yes
5.8.3.10b	Problem and non conformance handling / customer's visibility of problems detected during the verification activities	Yes
5.8.3.11a	Schedulability analysis as support for verification of software requirements and architectural design	Yes
5.8.3.11b	Schedulability analysis as support for verification of software detailed design	Yes
5.8.3.11c	Schedulability analysis as support for verification of software coding and testing	Yes
5.8.3.12a	Technical budget management: as support for verification of software requirements & architectural design / sizing (memory) and timing (CPU load) estimation	Yes
5.8.3.12b	Technical budget management: as support for verification of software detailed design/ sizing (memory) and timing (CPU utilization in WCET) estimation refinement	Yes
5.8.3.12c	Technical budget management: as support for verification of software coding and testing / sizing (memory) and timing (CPU utilization in WCET) calculation	Yes
5.8.3.13a	Behavioural modelling verification as support for verification of software requirements and architectural design – verification of the behavioural view of the logical model	No
5.8.3.13b	Behavioural modelling verification as support for verification of software detailed design / modelling the software behaviour and verifying by means of the techniques used for its description	No
5.8.3.14	Verification of design: feasibility of testing / availability of appropriate verification points, assertions, capability of fault injection	No
5.9	Software operation process	
5.9.2.1	Operational plans and standards development	No
5.9.2.2	Problem handling procedures definition	No
5.9.2.3	Operational testing definition	No

List of ECSS-E-40 Part 1B (28 November 2003) requirements		Applicable
5.9.3.1	Operational testing execution	No
5.9.3.2	Software operational requirements demonstration	No
5.9.4	Software operation	No
5.9.5.1	User's assistance	No
5.9.5.2	Handling of user's requests	No
5.9.5.3	Provisions of work-around solutions	No
5.10	Software maintenance process	
5.10.2.1	Software maintenance process planning	No
5.10.2.2	Software maintenance process: procedures, methods and standards	No
5.10.2.3	Problem reporting and handling	No
5.10.2.4	Implementation of configuration management process	No
5.10.2.5	Long term maintenance for flight software	No
5.10.3.1	Problem analysis	No
5.10.3.2	Problem verification	Yes
5.10.3.3	Development of options for modifications	Yes
5.10.3.4	Documentation of problem, analysis and implementation	No
5.10.3.5	Customer approval of selected modifications options	Yes
5.10.4.1	Analysis and documentation of product modification	No
5.10.4.2	Documentation of software product changes	Yes
5.10.4.3	Invoking of software engineering process for modification implementation	Yes
5.10.5	Conducting maintenance review	No
5.10.6.1	Applicability of this standard to software migration	No
5.10.6.2	Migration planning and execution	No
5.10.6.3	Contribution to the migration plan	No
5.10.6.4	Preparation for migration	No
5.10.6.5	Notification of transition to migrated system	No
5.10.6.6	Post-operation review	No
5.10.6.7	Maintenance and accessibility of data of former system	No
5.10.7.1	Retirement planning	No
5.10.7.2	Notification to the operator of retirement	No
5.10.7.3	Identification of requirements for software retirement	No
5.10.7.4	Maintenance and accessibility to data of the retired product	No
5.3	Software management process	
5.3.2.1	Definition of software life cycle phases / definition of SLC phases included in the SDP	Yes
5.3.2.2a	Software life cycle identification / project SLC definition in SDP	Yes
5.3.2.2b	Software life cycle identification / definition of software development,	Yes

List of ECSS-E-40 Part 1B (28 November 2003) requirements		Applicable
	operations, and maintenance techniques + identification of projects risks in SDP	
5.3.2.2c	Software life cycle identification / definition of SLC in line with the software and system level processes in SDP	Yes
5.3.2.3	Identification of inputs and outputs associated to each phases / review plan | milestones	Yes
5.3.2.4	Identification of documentation relevant to each milestone / outputs from the milestones	Yes
5.3.2.5	Identification of interface between the development and the maintenance processes / elements of the maintenance plan	Yes
5.3.2.6	Requirements baseline at the SRR / customer approval of the requirements baseline	No
5.3.2.7	Software technical specification phase	Yes
5.3.2.8	Preliminary Design Review (PDR) / customer approval of technical specification and software architecture	Yes
5.3.2.9	Detailed Design Review (DDR) for flight software	No
5.3.2.10a	Critical Design Review (CDR) / milestone report	Yes
5.3.2.10b	Critical Design Review (CDR) / completeness of the software validation activities	Yes
5.3.2.11	Software verification and validation process / activities phasing in SDP	Yes
5.3.2.12a	Qualification Review (QR) / milestone report	No
5.3.2.12b	Qualification Review (QR) / review of summary of tests reports and SUM + verification of the software documentation consistency / customer's approval of qualified state	No
5.3.2.13	Acceptance Review (AR) / customer's approval of accepted state	Yes
5.3.2.14	Validation activities phasing with respect to AR / phasing of the activities of the software validation with respect to the RB in the SDP	No
5.3.2.15	Software procurement process implementation	Yes
5.3.3.2	Support to software reviews / milestone review reports	Yes
5.3.3.3a	Technical reviews / reports	Yes
5.3.3.3b	Technical reviews / plans for each SWP within its defined SLC	Yes
5.3.4.1	Interface definition / interface requirement document	Yes
5.3.4.2	Interface management procedures	No
5.3.5.1	Technical budget and margin philosophy	Yes
5.3.5.2	Technical budget and margin status at each milestone	Yes

B.5 Documentation

The ECSS software standards are completed with some Document Requirement Descriptions (DRDs), describing the most important software documents. The DRD list is a subset of the exhaustive list of documents to be produced in order to cover all the work output required by the standards.

The expected output of the requirements resulting of this tailoring can be placed in the following DRDs:

Table 4. Document Requirement Description

Destination folder	Document item	Applicable	Output from
RB	(Software) system specification	Yes	TASKS 11 & 12
TS	Software requirements specification	Yes	TASKS 21 & 22
TS	Software interface control document	Yes	TASKS 21 & 22
DDF	Software design document – software architecture	Yes	TASK 31
DDF	Software design document – software components design	Yes	TASK 31
DDF	Software source code	Yes	TASK 31
DDF	Software configuration file	Yes	TASK 41
DDF	Software release document	Yes	TASK 41
DDF	Training material	No	
DJF	Software reuse file	If any	
DJF	Procured software component list (ECSS-Q-80 output)	No	
DJF	Software verification plan	No	
DJF	Software validation plan	No	
DJF	Independent software verification and validation plan	No	
DJF	Software units/integration test plan	No	
DJF	Software validation testing specification with respect to RB-TS	Yes	TASK 32
DJF	(Analyses and inspection) verification report with respect to RB-TS	Yes	TASKS 32 & 41
DJF	Software traceability matrices	Yes	TASK 32
DJF	Software acceptance test plan	Yes	TASKS 32 & 41
DJF	Software requirements verification report	No	
DJF	Software architectural design verification report	No	
DJF	Software detailed design verification report	No	
DJF	Software code verification report	No	
DJF	Software documentation verification report	No	
DJF	Software unit/integration test report	No	
DJF	Software validation test report with respect to RB-TS	Yes	TASKS 32 & 41
DJF	Validation evaluation report with respect to TS	No	
DJF	Validation evaluation report with respect to RB	No	
DJF	Software design and test evaluation report	No	
DJF	Acceptance test report	Yes	TASKS 32 & 41
DJF	Installation plan	No	

Destination folder	Document item	Applicable	Output from
DJF	Installation report	No	
DJF	Software budget report	Yes	TASK 41
DJF	Software acceptance data package	Yes	TASK 41
DJF	Schedulability analyses	Yes	TASK 41
DJF	Numerical accuracy analyses (ECSS-Q-80 output)	No	
DJF	Software behaviour verification	No	
DJF	Testing feasibility report	No	
DJF	Problems and non-conformance report	Yes	
DJF	Milestones report	Yes	PDR, CDR, AR
MF	Problem report and non-conformance report / modification analysis report / problem analysis report	Yes	
MF	Migration plan	No	
MF	Retirement plan	No	
MGT	Software development plan (at proposal only)	Yes	At proposal
PAF	Compliance matrix to the applicable software (ECSS-Q-80 output)	No	
PAF	Software product assurance requirements for suppliers (ECSS-Q-80 output))	No	
PAF	Audit plan (ECSS-Q-80 output)	No	
PAF	Software process assessment plan (ECSS-Q-80 output)	No	