



CENTRE NATIONAL D'ÉTUDES SPATIALES

CNES STANDARDS REFERENCE

RNC

Reference: **RNC-CNES-Q-ST-80-100**

Issue 4

02 June 2008

STANDARD

SPACE PRODUIT ASSURANCE SIMPLIFIED QUALITY ASSURANCE REQUIREMENTS FOR SOFTWARE DEVELOPMENT

APPROVAL STANDARDIZATION OFFICE	BN n°18 du 07/10/05 : ce document a fait l'objet de corrections d'erreurs de traductions (pas de changement du contenu technique). Il se substitue au document initialement introduit dans le RNC. BN n°44 du 08/09/08
APPROVAL OF CDN CHAIRMAN Alain CUQUEL	

DOCUMENT ANALYSIS PAGE

TITLE : SIMPLIFIED QUALITY ASSURANCE REQUIREMENTS FOR SOFTWARE DEVELOPMENT	
KEYWORDS : Software quality - Software development - R&T	
EQUIVALENT STANDARD : Not applicable	
REMARKS : Not applicable	
ABSTRACT: This document defines the simplified Software Quality Assurance requirements which apply to : <ul style="list-style-type: none"> - the development and maintenance of ground information systems and software on-board PME's or in scientific laboratories. - data processing developments linked to Research and Technologies. 	
DOCUMENT STATUS: This document is part of the collection of standards associated with the CNES Standards Reference. It is affiliated to standard "RNC-ECSS-Q-ST-80 Software Product Assurance".	
NUMBER OF PAGES : 43	LANGUAGE : English
Software used / version : Word 2002	
MANAGING DEPARTMENT : General Inspectorate and Quality Directorate (IGQ)	
AUTHOR(S):	DATE : 02 June 2008
Initialement établi par G. GUILLOT/J-F. POBLE repris par J-C. DAMERY	

© CNES 2008

Reproduction strictly reserved for the private use of the copier, not designed for collective use (article 41-2 of Law No. 57-298 of 11 March 1957).

DOCUMENT REVISION SHEET

ISSUE	DATE	PAGES MODIFIED	REMARKS
PR.0	26/08/93		Creation
1.0	21/04/94	i.1; i.2; i.3; 1; 2; 3; 4; 5; 6	Approval by the Technical Committee on reference documentation (CTR) and the Validation Board
2	01/03/00		New codification of documents (was "MPM-53-00-09")
3	08/12/03	All pages	Document restructuring <ul style="list-style-type: none"> ✍ to ensure compliance with standard RNC-ECSS-Q-80A and add DRD's ✍ to broaden the scope of the document to include cases of software development in the area of R&T.
4	02/06/08	All pages	New codification according to ECSS benchmarking (was previous reference "RNC-CNES-Q-80-509")



**SIMPLIFIED QUALITY ASSURANCE
REQUIREMENTS FOR SOFTWARE
DEVELOPMENT**

CONTENTS

1 INTRODUCTION 6

2 SUBJECT 6

3 SCOPE..... 6

4 READING AND APPLICATION GUIDELINES..... 6

4.1 REQUIREMENT ADJUSTMENTS ACCORDING TO THE CONTEXT OF THE PROJECT ... 6

4.2 PRECISIONS ON THE R&T CONTEXT 7

5 ASSOCIATED DOCUMENTS 7

5.1 REFERENCE DOCUMENTS 7

5.2 APPLICABLE DOCUMENTS..... 8

6 ACRONYMS AND ABBREVIATIONS..... 8

7 BASIC PRINCIPLES OF THE METHOD 10

8 ORGANISATION AND PRODUCT ASSURANCE..... 10

9 DESCRIPTION OF THE APPROACH..... 11

9.1 DEVELOPMENT CYCLE.....11

9.1.1 Software specification12

9.1.2 Design.....13

9.1.3 Coding - Unit tests13

9.1.4 Validation14

9.1.5 Acceptance16

9.2 CONFIGURATION MANAGEMENT – MODIFICATION MANAGEMENT16

9.2.1 Principles.....17

9.2.2 Case of UNIX applications18

9.2.3 Configuration description document18

9.3 METHODS, TOOLS, RULES.....18

9.3.1 Principles.....18

9.3.2 Nomenclature rules19

9.3.3 Coding rules.....19

9.4 SPECIAL DEVELOPMENT CONDITIONS.....20



**SIMPLIFIED QUALITY ASSURANCE
REQUIREMENTS FOR SOFTWARE
DEVELOPMENT**

9.4.1	Software dependability and safety	20
9.4.2	Software reused.....	21
9.4.3	Man-machine interface.....	22
9.5	RECOMMENDED DOCUMENTATION IN THE CASE OF A STANDARD SOFTWARE DEVELOPMENT	23
9.6	RECOMMENDED DOCUMENTATION IN THE CASE OF AN R&T SOFTWARE DEVELOPMENT	24
10	ANNEX 1: Document requirement definition.....	25
10.1	APPLICATION PLAN.....	25
10.1.1	Presentation.....	25
10.1.2	Master application plan.....	25
10.2	SPECIFICATION DOCUMENT.....	27
10.2.1	Presentation.....	27
10.2.2	Specification document DRD.....	27
10.3	DESIGN DOCUMENT	29
10.3.1	Presentation.....	29
10.3.2	Design document DRD	29
10.4	VALIDATION DOCUMENT	30
10.4.1	Presentation.....	30
10.4.2	Validation document DRD.....	30
10.5	USER MANUAL.....	33
10.5.1	Presentation.....	33
10.5.2	User manual DRD.....	33
10.6	ACCEPTANCE TEST PLAN	35
10.6.1	Presentation.....	35
10.6.2	Acceptance test plan DRD.....	35
11	ANNEX 2: Index OF RULES.....	38
11.1	CASE OF STANDARD SOFTWARE DEVELOPMENT	38
11.2	CASE OF CRITICAL SOFTWARE DEVELOPMENT.....	40
11.3	CASE OF AN R&T SOFTWARE DEVELOPMENT.....	42

1 INTRODUCTION

This "Simplified Quality Assurance requirements for software development" document is part of standard associated with standard "RNC-ECSS-Q-ST-80 Software Quality Assurance".

It is also included in the "Guidelines for Science Projects" available on the GNS Internet server (Simplified standard guide - <http://gns.cnes.fr>).

2 SUBJECT

The purpose of this document is to present simplified software quality requirements used to obtain a satisfactory software quality level for application software developed as part of some CNES project, by the supplying manufacturers, by the scientific laboratories or in-house in the CNES, as well as those developed in a Research and Technology context (R&T in the remainder of this document).

3 SCOPE

For certain applications, a relaxation of the quality requirements as stated in "RNC-ECSS-Q-ST-80" has been accepted, in relation with the acceptable risk level; this document applies to software developments intended for these specific applications.

This document also applies to cases of software developments in an R&T context for which the quality level is not of major importance.

It is used:

- ? right from phase A of the project to define the scope of development to future designers,
- ? right from the start of the developments (end of phase B - start of phase C) as a basis for discussions with the software designer in order to finalize the practical provisions.

These provisions are defined during work meetings between CNES and its suppliers; they are formalized in the supplier's proposal, and can be included in an application plan (see EX1).

4 READING AND APPLICATION GUIDELINES

This document is structured according to requirements (in Exn nomenclature) and to recommendations (in Recomj nomenclature) applicable to a software development.

Some requirements are adjusted according to the context of the software development:

- ? Software criticality,
- ? Size of the software,
- ? Costs and schedule requirements,
- ? R&T context of data processing development.

An application context is associated with each requirement.

4.1 REQUIREMENT ADJUSTMENTS ACCORDING TO THE CONTEXT OF THE PROJECT

It is the responsibility of the CNES to reach a decision regarding the adjustments to the requirements, and in this way to determine the package of requirements and of recommendations applicable to a specific

software development, and therefore to ask the supplier in charge of this development to indicate in what way he intends to meet the requirements and the recommendations.

In this document, a classification is suggested to process the two cases of software: “**outside an R&T context**” and “**in an R&T context**”. In the “outside an R&T” context, if the software is slightly more **critical** than the **standard** case, a few additional requirements will be added.

Adjusting the requirements for a project may lead to carry out another classification, such as the application of all or part of the R&T requirements for a very small project.

4.2 PRECISIONS ON THE R&T CONTEXT

It is important to make a distinction between the **R&T software developments** and the **industrialisation** which is carried out for some software on completion of these developments, but outside an R&T context.

In this perspective, the classification of the “R&T” requirements suggested in this document deals with the first phase of R&T development, and therefore does not cover the industrialisation.

To cover the industrialisation phase, the product assurance requirements must be completed in order to include the actions aimed at increasing the product reliability with regard to the following aspects:

- ? Details on the documentation
- ? Test environment (documentation, sets of tests)
- ? Making the code reliable (if need be, full recoding at the industrialisation phase), in terms of interface and input management
- ? Strategy for the management and traceability of errors.

5 ASSOCIATED DOCUMENTS

5.1 REFERENCE DOCUMENTS

- DR1: Software product assurance
RNC-ECSS-Q-ST-80
- DR2: Software general requirements
RNC-ECSS-E-ST-40
- DR3: Contents of a software technical specification document
RNC-CNES-E-HB-40-501*
- DR4: Technical readiness review board
RNC-CNES-Q-ST-20-100
- DR5: Test Review Board
RNC-CNES-Q-ST-20-101
- DR6: Contents of a reused software file
RNC-CNES-Q-HB-80-514
- DR7: Rules for using FORTRAN 77 language
RNC-CNES-Q-HB-80-505*

* Only in French version

- DR8: Rules for using C language
RNC-CNES-Q-HB-80-506*
and its annex for on-board software - RNC-CNES-Q-80-HB-506-A
- DR9: Rules for using Ada language
RNC-CNES-Q-HB-80-504*
and its annex for Ada 95 - RNC-CNES-Q-80-504-A
- DR10: Rules for using C++ language
RNC-CNES-Q-HB-80-513*
- DR11: Rules for using FORTRAN 90 language
RNC-CNES-Q-HB-80-517*
- DR12: Rules for the use of the Ada language in on-board software
RNC-CNES-Q-HB-80-528*
- DR13: Guidelines for the selection and the interpretation of the software complexity measurements
RNC-CNES-Q-HB-80-503*
- DR14: User's manual and operating manuals for a ground computer processing system
RNC-CNES-E-HB-40-503
- DR15: Rules and recommendations to conduct a software product acceptance
RNC-CNES-E-HB-40-506*
- DR16: Software Configuration Description File
RNC-CNES-M-HB-40-516
- DR17: Standard requirements for software configuration management
RNC-CNES-M-ST-40-100
- DR18: Rules for using Java language
RNC-CNES-Q-HB-80-527*
- DR19: Rules for using Wave language
RNC-CNES-Q-HB-80-521*
- DR20: Rules and recommendations for Man-Machine Computer Interface Ergonomics
RNC-CNES-E-HB-40-504
and its appended verification guide: RNC-CNES-E-HB-40-504-A
- DR21 : Rules for using PERL language
RNC-CNES-Q-HB-80-533*
- DR22 : Rules for using IDL language
RNC-CNES-Q-HB-80-534*
- DR23 : Rules for using coding languages
RNC-CNES-Q-HB-80-501*

5.2 APPLICABLE DOCUMENTS

Not applicable.

6 ACRONYMS AND ABBREVIATIONS

- CDF** Configuration Description File
- Crit.** Applicable only to critical software (outside R&T)

* Only in French version

**SIMPLIFIED QUALITY ASSURANCE
REQUIREMENTS FOR SOFTWARE
DEVELOPMENT**

Issue 4
02 June 2008

DRD	Document Requirement Definition
ECSS	European Cooperation for Space Standardization
PTRB	Post Test Review Board
R&T	Research and Technology
RD	Reference document
RNC	CNES Standard Reference
GNS	Simplified standard guide
SSR	Software Specification Review
Std.	Standard application to critical and non-critical software (outside R&T)
TRRB	Technical Readiness Review Board

7 BASIC PRINCIPLES OF THE METHOD

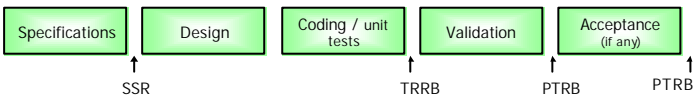
Reference	Description	Outside R&T	R&T context
EX1	Any software development must be conducted according to pre-established rules in order to prove compliance with the requirements of this document.	Std.	x
EX2	Compliance with these requirements is transferred to an application plan (a DRD is given in annex 1 § 10.1), which expresses the supplier's undertaking to take into account the requirements described below.	Std.	

8 ORGANISATION AND PRODUCT ASSURANCE

Reference	Description	Outside R&T	R&T context
EX3	The supplier must specify the organisation and the responsibilities of the team in charge of development, and in particular he must identify the Software Product Assurance Manager.	Std.	x
EX4	The supplier must specify the training required to fill in any gaps in the available skills of the development team.	Std.	x
EX5	The supplier must specify the Product Assurance provisions he is setting up, as well as the planning of the corresponding activities.	Std.	x
EX6	The Product Assurance provisions include at least the configuration management and the management of change requests and non-conformances.	Std.	x
EX7	The Product Assurance provisions must be complemented by verification procedures and by the drafting of a quality report.	Std.	
EX8	In cases where the service consists in producing a system , the supplier must include in his proposal an identification of the technical risks, issued from a system analysis, and an identification of their implication with regard to the development of the software.	Std.	x
EX9	In cases where the service consists in producing a sub-system , the supplier must include in his proposal an analysis of the impact of the system risks identified by the CNES.	Std.	x
EX10	The supplier must make sure that his subcontractors implement equivalent quality assurance provisions. To this end, he passes on, to his subcontractors, all or part of the requirements he is subject to.	Std.	x

9 DESCRIPTION OF THE APPROACH

9.1 DEVELOPMENT CYCLE

Reference	Description	Outside R&T	R&T context
EX11	The software development cycle must be defined while taking into account the software technical characteristics and the associated development constraints, such as the selection of the techniques used (database management system, reuse of existing products, man-machine interface generators, etc.) and the risks inherent to the project.	Std.	x
EX12	A product evaluation in coherence with the needs must be done before any software development.	Std.	
EX13	This evaluation is conducted jointly by the supplier and by the CNES.		x
EX14	<p>After this evaluation phase, the software development cycle must include the following phases:</p> <ul style="list-style-type: none"> - Software specification, - design, - coding - unit tests, - validation, - acceptance if required.  <pre> graph LR S[Specifications] --> D[Design] D --> C[Coding / unit tests] C --> V[Validation] V --> A[Acceptance (if any)] SSR[SSR] -.-> S TRRB[TRRB] -.-> C PTRB1[PTRB] -.-> V PTRB2[PTRB] -.-> A </pre>	Std.	
EX15	<p>The development cycle can be iterative: in this case, the iterations are described in the supplier's proposal according to the following criteria:</p> <ul style="list-style-type: none"> ? Objectives, ? Activities planned (which may cover all or parts of the activities planned in the traditional phases of the life cycle (requirements of subclauses 9.1.1 to 9.1.5)), ? Completion criteria. <p>It should be noted that the iterations can be refined as the development progresses; the CNES and the supplier specify the conditions for performance of iteration n+1 at the end of iteration n.</p>		x
EX16	Each phase must end with a technical meeting with the CNES (milestone) to examine the work performed during this phase and the level of preparation of the following phase.	Std.	x
EX17	In the case of software integrated into a piece of equipment , the software participates as a constituent to the equipment-software integration phase, and its development cycle must be consistent with that of the equipment.	Crit.	

9.1.1 Software specification

Reference	Description	Outside R&T	R&T context
EX18	<p>The specification phase activities consist at least in:</p> <ul style="list-style-type: none"> ? identifying the needs expressed by the customer on completion of the latter's needs analysis, and then in translating them in terms of the functions to be fulfilled by the software and of interfaces with the outside and among themselves, ? analysing, according to the needs to be covered, the software which could be reused and assessing the impacts of their reuse upon the development (see § 9.4.2), ? drafting a preliminary version of the validation plan. <p>The output elements are described in the project documentation (§ 9.5 and 9.6).</p>	Std.	x
EX19	<p>The activities of the specification phase are complemented by:</p> <ul style="list-style-type: none"> ? details on the sequence of functions, ? details on the constraints (performance, priorities, storage occupancy), 	Std.	
EX20	<p>The output elements of the specification phase are described in the software specification document (a DRD is given in annex 1 § 10.2).</p>	Std.	
EX21	<p>In the particular case of a development including an MMI, the specification phase must make it possible to design a man-machine interface mock-up (cf. § 9.4.3).</p>	Std.	x
EX22	<p>In the particular case of a development subject to dependability constraints, the specification phase makes it possible to determine the critical parts of the software after a functional analysis (see § 9.4.1).</p>	Crit.	
EX23	<p>This phase ends with a software specification technical meeting (or review) (SSR) to take a decision on the documents originating in this phase and on the possible reuse of software. This technical meeting must take place with the participation of the technical managers, the product assurance managers and/or the users.</p>	Std.	

9.1.2 Design

Reference	Description	Outside R&T	R&T context
EX24	The design activity consists at least in: ? defining the structural breakdown of the application into software constituents, and in detailing each one of them, ? defining the data flow and the interfaces. The output elements are documented in the project documentation (§ 9.5 and 9.6).	Std.	x
EX25	The design activity is complemented by resource estimates (core memory, mass storage, CPU, peripheral equipment...).	Crit.	
EX26	The output elements of the design phase are described in the design document (the DRD of the document is given in annex 1 § 10.3).	Std.	
EX27	The design of the software highlights in different design elements the functions that we wish to reuse in order to be able to validate them independently.	Std.	x

9.1.3 Coding - Unit tests

Reference	Description	Outside R&T	R&T context
EX28	The coding and unit test activities are: ? the coding of the constituents, ? the performance of unit tests on the constituents and of integration tests between constituents.	Std.	x
Recom1	The software constituents are coded in an advanced programming language rather than in assembler. The same language is given preference for the production of the entire software (except for installation scripts).	Std.	x
EX29	The coding complies with the rules stated in subclause 9.3.3.	Std.	x

**SIMPLIFIED QUALITY ASSURANCE
REQUIREMENTS FOR SOFTWARE
DEVELOPMENT**

9.1.4 Validation

Reference	Description	Outside R&T	R&T context
EX30	The activities in the validation phase consist in conducting the test plans in order to check that the software fulfils its specified functions.	Std.	x
EX31	The validation tests must cover all the software specification requirements (a DRD of the validation document is given in annex 1 § 10.4).	Std.	
EX32	In the case of an iterative development , as the validation objectives may vary from one iteration to the next, the validation objectives of iteration n+1 must be defined between the CNES and the supplier at the end of iteration n.		x
EX33	In the case of software integrated into a piece of equipment , the validation must include tests on the identification model of the corresponding equipment.	Crit.	
EX34	The validation tests must be performed on a stable software whose version is registered by configuration management. Any modification of the configuration during the validation phase must be recorded.	Std.	x

**SIMPLIFIED QUALITY ASSURANCE
REQUIREMENTS FOR SOFTWARE
DEVELOPMENT**

Reference	Description	Outside R&T	R&T context
EX35	<p>The validation phase is preceded by a technical readiness review board (TRRB) and closed by an end-of-test meeting (Post Test Review Board). After this meeting, all the updated documentation must be accepted by the CNES.</p> <ol style="list-style-type: none"> 1. The purpose of the Technical readiness review board (TRRB) is to authorise the launch of the validation tests. This meeting takes place at the start of the validation phase with both the customer and the supplier. It allows: <ul style="list-style-type: none"> - the checking of the action status, - the checking of the completion and consistency of the test plan and procedures, - the identification of the configuration status and the freezing of the corresponding software release. <p>At the end of this Technical Readiness Review board, if the test start is authorised, all the tests planned are performed.</p> 2. During the tests, the result of each test is recorded in a logbook, along with the non-conformances and modifications which have been identified. <p>As far as possible, the corrective or evolutionary actions associated with the non-conformances and to the modifications identified are only carried out at the end of the validation phase, and will result in a new release of the software.</p> <p>If this is not possible, any change in configuration in the course of the tests is recorded in the logbook.</p> 3. The Post Test Review Board (PTRB) shall take place at the end of the validation phase. It aims at: <ul style="list-style-type: none"> - Presenting a summary of the test results, - Reviewing the status of the non-conformance and residual modifications, - Identifying actions to be led, specifying the closing date and the person(s) responsible for the ones that are still open. - Deciding the acceptance or rejection of the end of the validation phase. <p>In case of rejection or acceptance with reservation(s), the actions to be performed and their deadlines will be explicitly mentioned in the report.</p> 	Std.	

**SIMPLIFIED QUALITY ASSURANCE
REQUIREMENTS FOR SOFTWARE
DEVELOPMENT**

Reference	Description	Outside R&T	R&T context
EX36	<p>The validation phase is framed by 2 meetings. The first permits to identify the configuration state and the freezing of the corresponding software release. The second is to draft:</p> <ul style="list-style-type: none"> ? a summary of the test results, ? a status of the non-conformances and residual modifications, ? the acceptance or rejection of the end of the validation phase. <p>During the tests, the results of each test are recorded, as well as the non-conformances and modifications identified.</p>		x

9.1.5 Acceptance

Reference	Description	Outside R&T	R&T context
EX37	<p>If there is a software acceptance, it must meet the requirements defined above for the software validation phase. The acceptance is based on the acceptance test plan (possibly taken from the software validation document – DRD in annex 1 § 10.6). It takes place after the validation phase has been accepted. It ends with a Post Test Review Board in the course of which the acceptance of the developed software shall be pronounced (or not).</p>	Std.	x
EX38	<p>In the case of software integrated into a piece of equipment, acceptance is carried out on the model used for qualification of the corresponding hardware, or failing this, on the most representative model.</p>	Crit.	

9.2 CONFIGURATION MANAGEMENT – MODIFICATION MANAGEMENT

The software configuration management corresponds to all the manual or automatic activities used to identify, at any time, the elements created, used or modified by the software development process and their relations.

Its main purpose is to store each reference version of an element in order to know precisely and at all times which version is being used, so as to be able to reconstitute any earlier version, if need be.

9.2.1 Principles

Reference	Description	Outside R&T	R&T context
EX39	The software configuration management must identify each element to be managed, in a unique way according to the project nomenclature rules.	Std.	x
EX40	A project team member is appointed to manage the software configuration. His task is to: ? define the elements to be managed, ? define the reference version at any time, ? be aware of the relations between the elements of a given configuration (in order for instance to be able to answer the question: “which validation tests correspond to this software release ?”), ? define when a configuration element is entered into the (archiving) configuration, ? define and apply the delivery conditions for the product.	Std.	x
EX41	For each software release, configuration management must make it possible to: ? identify the documents with their reference and issue/revision, ? save on an identified medium with an unpeelable marking label specifying the name of the application software and its release (version/revision): the product (sources, data, generation procedures), the sets of validation tests, the associated specific test means, if any (simulator, data...), ? know the references: of the back-up medium (tape No., for instance), of the technical facts (non-conformances, corrections, change requests), of the basic software (operating system, emulators, real-time monitors, libraries, compilers, link editors...), of the hardware, of the test means, if any ? restore an archived software, if necessary.	Std.	x

9.2.2 Case of UNIX applications

Reference	Description	Outside R&T	R&T context
Recom 2	In the case of software developed on the UNIX platform, with a portability requirement on several operating systems, it is recommended to use standard generation / installation procedures.	Std.	x

Standard solutions (based on free and freely available products: autoconf, automake, libtool) make it possible to do away with the specific characteristics of the machines (compiler's name, access paths to libraries) during generation, to set the parameters for the product installation directories, and to obtain simple procedures to launch the generation and the installation.

9.2.3 Configuration description document

Reference	Description	Outside R&T	R&T context
EX42	<p>For each version the supplier must draw up a configuration description file (CDF), included in the delivery of the new release of the product. It shall contain at least the following files (see DR16 for a description of the expected contents of these files) :</p> <ul style="list-style-type: none"> ? ddc.txt.....item identification ? fa-couv.txt.....list of covered NCR's ? dm-couv.txt.....list of covered change requests ? etat-fa.txt.....list of open NCR's ? ls-doc.txt.....list of documents (CNES and supplier) associated with the product ? ls-ref.txt.....list of files making up the product 	Std.	x

9.3 METHODS, TOOLS, RULES

9.3.1 Principles

Reference	Description	Outside R&T	R&T context
EX43	<p>The methods and supporting tools (if any) adapted to the various phases of the software life cycle are selected according to the skills of the project team (training plan, past experience...) and to its financial resources.</p> <p>The use of freeware is strongly recommended (within the licence terms and conditions).</p>	Std.	x
EX44	The selection of tools must be justified in the application plan or its equivalent.	Std.	x

**SIMPLIFIED QUALITY ASSURANCE
REQUIREMENTS FOR SOFTWARE
DEVELOPMENT**

Reference	Description	Outside R&T	R&T context
Recom 3	The consistent use of standard tools over the whole development is recommended.	Std.	x

9.3.2 Nomenclature rules

Reference	Description	Outside R&T	R&T context
EX45	Nomenclature rules are defined, in particular for interface variables. These rules are consistent with the standard conventions for the coding language selected.	Std.	x

9.3.3 Coding rules

Reference	Description	Outside R&T	R&T Context
EX46	The interface variables are described in the form of comments in the code.	Std.	x
EX47	The self-documentation rules intrinsic to the programming language (ex: comments that can be used by javadoc for the Java language) must be applied.	Std.	x
EX48	<ul style="list-style-type: none"> ? Each code module includes a comment header containing at least the following headings: <ul style="list-style-type: none"> project name module name release author date role and purpose of the module ? A single entry point and a single exit point for any compilable unit. ? All parts of a code must be accessible (no dead code). ? Each order of conditional jump or break in sequence is commented by explaining the purpose of the disconnection. ? Each function or subroutine must specify its input, output or input/output parameters, with, for each one: <ul style="list-style-type: none"> its name, its meaning, the unit used in case of physical quantity, its degree of accuracy, its value field. 	Std.	x

9.4 SPECIAL DEVELOPMENT CONDITIONS

9.4.1 Software dependability and safety

Reference	Description	Outside R&T	R&T Context
EX49	A functional analysis at system level must be used to identify the critical software modules, taking into account the interaction between the software and its environment.	Crit.	
Recom 4	The supplier must endeavour to design the critical software components as simply as possible in order to facilitate the software dependability and safety analysis and software testing.	Crit.	

9.4.2 Software reused

Reference	Description	Outside R&T	R&T Context
EX50	<p>In cases where the use of existing software (market products, freeware, other products) is considered, the following elements must be taken into account for their selection:</p> <ul style="list-style-type: none"> ? the assessment of the product in relation with the requirements, ? the conditions for acceptance and guarantee (demonstration of proper operation) and/or maintenance, ? the training and use conditions, ? the industrial property and licence constraints (right to use, distribute or modify). 	Std.	x
EX51	<p>For software to be adapted, it is essential, prior to any modification:</p> <ul style="list-style-type: none"> ? to assess the rate of modification, ? to check the status of the existing software documentation in order to ensure that it is consistent and complete for the reused software, ? to analyse the impact of the modifications on the documentation and tests required for validation of the modifications (unit test on the modified constituent, functional test involving the modified constituent, regression testing for validation of the whole software). 	Std.	x
EX52	<p>On completion of the first phase of the project (initial specification phase), the list of software that are being considered for reuse is submitted, with their status and the associated development effort (documentation update, code update, necessary tests) specified, according to the supporting elements above.</p>	Std.	x
EX53	<p>In order to make software maintenance easier, any modification of the code must be carried out in such a way as to preserve the skeleton of the constituent (no “wart”). If this rule cannot be followed, opt for a full reprogramming of the constituent in accordance with structured programming principles.</p>	Std.	x
EX54	<p>The configuration management of the reused software must be set up.</p>	Std.	x
Recom 5	<p>Stand-alone configuration management of the reused software components is recommended. The reused software components are then referenced in the file describing the configuration of the developed product.</p>	Std.	x

9.4.3 Man-machine interface

Reference	Description	Outside R&T	R&T Context
EX55	<p>For software having a man-machine interface, a mock-up of this interface must be set up during the specification phase.</p> <p>This mock-up must show:</p> <ul style="list-style-type: none"> ? the content of the screens, ? the screen sequence dynamics, ? the implementation of the various controls, ? the on-line help features, if any. 	Std.	x
EX56	Future users must participate in the validation of the man-machine interface mock-up, which is to take place in the course of the specification phase.	Std.	
EX57	Future users are involved in the validation of the mock-up. In the case where the mock-up is expanded during iterative cycles, the users must participate in the validation at each step of the cycle.		x
EX58	The users must participate in the software acceptance phase.	Std.	

**SIMPLIFIED QUALITY ASSURANCE
REQUIREMENTS FOR SOFTWARE
DEVELOPMENT**

9.5 RECOMMENDED DOCUMENTATION IN THE CASE OF A STANDARD SOFTWARE DEVELOPMENT

The reference documents (RD) in chapter 5 can be used as a basis for the drafting of some documents required from the supplier.

The formal meetings recommended in the case of a traditional V-shaped life cycle are tabulated below. The times of delivery of the document should be adapted in the case of a different development cycle.

	Kick-off meeting	SSR	End of design	TRRB before validation	PTRB after validation	Acceptance	At each version	At each progress meeting
Application plan	P	D						
Software specification document		D						
Design document			D					
Validation Document		P		C	D			
Configuration description file				C	J	J	J	
User manual		P			C	D		
Acceptance Test plan			P			D		
Schedule	P							J

Key:

- P preliminary version
- C full version
- D final version
- J updated version

Details on the contents of the documents (including DRD) are given in the appendices.

9.6 RECOMMENDED DOCUMENTATION IN THE CASE OF AN R&T SOFTWARE DEVELOPMENT

It is recommended to make a single document in order to streamline the documentation management in this context.

The document must contain the following chapters:

- ? Development context
 - hypotheses made for the development,
 - input data limits.
- ? Specifications
 - software specifications.
- ? Information intended for users
 - user manual (in the case of a library, examples of library function call-ups),
 - reference manual (catalogue of available functions).
- ? Information required for the maintenance of the developed product
 - list of tests used to validate the product acceptance,
 - detailed design documentation (preference is given to self-documentation in the code modules),
 - history of choices made for the development (choices validated during the progress meeting and technical meeting with the customer),
 - configuration description file.

10 ANNEX 1: DOCUMENT REQUIREMENT DEFINITION

10.1 APPLICATION PLAN

10.1.1 Presentation

The plan describes the management provisions which are applied to the project in order to meet the requirements of this document. These management provisions must be defined during work meetings between the customer and the supplier.

10.1.2 Master application plan

GLOSSARY

1. INTRODUCTION

1.1. BRIEF DESCRIPTION OF THE PRODUCT

1.2. ELEMENTS CONCERNED BY THE APPLICATION PLAN

2. APPLICABLE DOCUMENTS

3. REFERENCE DOCUMENTS

4. DEVELOPMENT PLAN

4.1. PROJECT ORGANISATION

4.2. DEVELOPMENT CYCLE

4.2.1. Specification phase

4.2.2. Design phase

4.2.3. Coding - unit test phase

4.2.4. Validation phase

4.2.5. Acceptance phase

4.3. MEETINGS ORGANISATION

4.3.1. Progress meetings

4.3.2. Key points and reviews

4.4. HARDWARE AND SOFTWARE RESOURCES

4.5. SCHEDULE

4.6.

5. PRODUCT ASSURANCE PLAN

5.1. SOFTWARE DEPENDABILITY AND SAFETY

5.2. METHODS, TOOLS AND RULES

5.3. CONFIGURATION MANAGEMENT

5.4. DOCUMENTATION MANAGEMENT

5.5. MODIFICATION MANAGEMENT

5.6. REUSED SOFTWARE

5.7. MAN-MACHINE INTERFACE

10.2 SPECIFICATION DOCUMENT

10.2.1 Presentation

This document describes:

- the functions of the software,
- the way in which the software is to be used,
- the external interfaces,
- the data handled,
- the data flow,
- the enabling conditions, if any,
- the data recovery or backup points, if any, to be provided for in case of contingency,
- the constraints (resources...),
- the conditions of reuse of third party products (COTS, freeware...).

10.2.2 Specification document DRD

GLOSSARY

1. SUBJECT

2. APPLICABLE DOCUMENTS

3. REFERENCE DOCUMENTS

4. FUNCTIONAL SPECIFICATIONS

4.1. PRESENTATION OF THE MISSION OF THE SOFTWARE

4.2. SPECIFICATIONS OF THE <I> FUNCTION

4.3. SPECIFICATIONS OF THE <J> FUNCTION

5. OPERATIONAL SPECIFICATIONS

5.1. DESCRIPTION OF THE REQUIRED ENVIRONMENT

5.1.1. Hardware environment

5.1.2. Software environment

5.2. INITIALIZATION / STOP CONDITIONS / RESUME CONDITIONS

5.3. CONDITIONS OF ENABLING FUNCTION

6. PERFORMANCE SPECIFICATIONS

7. EXTERNAL INTERFACE SPECIFICATIONS

7.1. INPUT DATA

7.2. OUTPUT DATA

8. INTERNAL INTERFACE SPECIFICATIONS

8.1. DESCRIPTION OF THE <I> DATUM

8.2. DESCRIPTION OF THE <J> DATUM

9. EVALUATION OF THE REUSED SOFTWARE

9.1. <A> SOFTWARE

9.1.1. Presentation of the software

9.1.2. Advantages of reusing the software

9.1.3. Modifications envisaged on the software

On the product, its documentation and its tests

10. CONFORMANCE WITH THE REQUIREMENT SPECIFICATION

In the form of a cover matrix if the specification is structured as requirements or subclauses.

10.3 DESIGN DOCUMENT

10.3.1 Presentation

This document describes the solution selected to meet the software specifications:

- software architecture with a breakdown into constituents (call-up charts, data flow),
- the synchronisations between constituents, if any,
- the strategy for processing errors and exceptions.

A description of each constituent appears in the form of comments in the code. It includes:

- its role from a functional point of view,
- its main interfaces (files, parameters, messages...),
- its “links” with other constituents (links for call-up, dependency, service, inheritances, genericity...),
- its enabling conditions, if any,
- its pseudo-code.

10.3.2 Design document DRD

GLOSSARY

1. SUBJECT

2. APPLICABLE DOCUMENTS

3. REFERENCE DOCUMENTS

4. DESIGN APPROACH

4.1. DESCRIPTION OF THE DESIGN APPROACH

4.2. CONSTRAINTS

4.3. ERROR MANAGEMENT STRATEGY

5. SOFTWARE ARCHITECTURE

5.1. PRESENTATION OF THE ARCHITECTURE PRINCIPLES

5.2. DESIGN ORGANISATION

Design diagrams

6. DATA DESIGN

6.1. DATA TREE STRUCTURE ON DISK

6.2. DATA FLOW

10.4 VALIDATION DOCUMENT

10.4.1 Presentation

At the end of the software specification phase, a first version of this document describes:

- the organisation, the planning and the sequence of the tests,
- the resources required, particularly the test means, specifying the limits of their representativeness in relation to the actual environment:
 - physical domain limits,
 - performance limits,
 - limits in terms of functions,
 - limits in terms of controllability and observability,
- the validation plan (description of the software components to be tested from a functional point of view, list of tests in terms of objectives and resources required).

This document is completed before the start of the software validation phase in order to describe the test procedures for each validation tests. These procedures include a description of the test implementation, of the actions to be performed (before, during or after the test), of the resources to be set up and of the expected results.

At the end of the software validation phase, the final document is to be completed with test reports which will describe the results obtained with the references of the non-conformances detected, if any.

10.4.2 Validation document DRD

GLOSSARY

- 1. INTRODUCTION**
- 2. APPLICABLE DOCUMENTS**
- 3. REFERENCE DOCUMENTS**
- 4. REMINDER OF THE FUNCTIONS OF THE PRODUCT**
- 5. PRODUCT ARCHITECTURE**
- 6. VALIDATION ORGANISATION**
- 6.1. ENVIRONMENT**

6.1.1. CONFIGURATION AND SITE

6.1.2. TEST MEANS

6.2. SCHEDULE

6.3. TEST SEQUENCE LOGIC

7. VALIDATION PLAN

7.1. SOFTWARE TO BE TESTED

7.2. INSTALLATION - PARAMETER SETTINGS

7.3. FUNCTIONAL TESTS

7.4. CONSTRAINTS

7.4.1. ENVIRONMENTAL CONSTRAINTS

7.4.2. PERFORMANCE

8. DESCRIPTION OF TESTS

Project XXXXX		
Test No: xxxxxxxx		
Test: XXXXX	Date: 21/04/2003	Test release: 1.0
Type: Functional [<input type="checkbox"/>]	Contingency [<input type="checkbox"/>]	Performance [<input type="checkbox"/>]
PURPOSE OF THE TEST:		
PRELIMINARY TESTS:		
TEST DATA:		
EXPECTED RESULTS:		
IMPLEMENTATION PROCEDURE:		
OBTAINED RESULTS:		
Non-conformance reference:		
Comments:		

10.5 USER MANUAL

10.5.1 Presentation

The supply of this document depends on the type of software. It is necessary for software with a man-machine interface.

This document describes:

- the different types of possible users,
- the means of interaction (keyboard, mouse, function keys, scroll bars, buttons, ...), their meaning and their principles of use,
- the menus and screen grids: type, meaning of the various fields, verifications performed and associated actions,
- the messages issued by the software with an indication of the issuing function and the associated corrective actions,
- the screen sequence,
- the format of hard copies.

The organisation and the contents of this document are to be submitted to future users for approval.

This document is initialised right from the software specification phase, and is completed during the development for a complete version to be delivered at the end of the validation of the software.

10.5.2 User manual DRD

GLOSSARY

1. INTRODUCTION

1.1. OBJECTIVES

1.2. USING THE DOCUMENT

1.3. WRITING CONVENTIONS

2. GENERAL TECHNICAL CONCEPTS

2.1. GENERAL PRESENTATION OF THE PRODUCT

2.2. PRESENTATION OF THE FUNCTIONS AND OF THE PARTICIPANTS

2.3. SPECIFIC TECHNICAL FEATURES

2.4. THE DATABASE

3. SYSTEM ENVIRONMENT

3.1. ORGANISATION

3.2. RESOURCES

3.3. ACCESS RIGHTS

4. STARTING THE APPLICATION

5. DESCRIPTION OF THE MMI

6. GENERAL UTILITIES

6.1. USER ADMINISTRATION

6.2. SYSTEM ADMINISTRATION

6.3. THE LOG BOOK

7. DETAIL OF THE FUNCTIONS

7.1. < NAME OF FUNCTION 1 >

7.1.1. Operational description

7.1.2. Description of the environment

7.1.3. Starting the function

7.1.4. Dynamics of the function

7.1.5. The services

For each service are given its description, its conditions of use, its implementation, examples and error messages.

7.1.6. Error messages and recoveries

7.1.7. Contingencies

7.2. < NAME OF FUNCTION 2 >

7.3. ...

8. GENERAL INDEX

10.6 ACCEPTANCE TEST PLAN

10.6.1 Presentation

The acceptance tests described in this document are taken from the list of tests detailed in the software validation document. The acceptance test plan must be approved by the CNES before the start of the acceptance.

10.6.2 Acceptance test plan DRD

GLOSSARY

1. INTRODUCTION

2. REMINDER ON THE FUNCTIONS OF THE PRODUCT

3. PRODUCT ARCHITECTURE

4. ACCEPTANCE COVERAGE

4.1. SUPPLIES TO BE ACCEPTED

4.1.1. SOFTWARE

List of specific software with their reference and release

List of standard (or off-the-shelf) software, with their reference and release

4.1.2. DOCUMENTS

List of the documents with their reference and release

4.1.3. HARDWARE

List of hardware with their reference and release

4.2. CONTEXT OF THE ACCEPTANCE

Specify for instance in what way the acceptance is partial (functions, tests, ...)

5. ACCEPTANCE TESTS

5.1. INSTALLATION - PARAMETER SETTINGS

5.2. FUNCTIONAL TESTS

5.3. CONSTRAINTS

5.3.1. ENVIRONMENTAL CONSTRAINTS

5.3.2. PERFORMANCE

6. TEST SEQUENCE LOGIC

7. ACCEPTANCE ORGANISATION

7.1. ENVIRONMENT

7.1.1. CONFIGURATION AND SITE

7.1.2. TEST MEANS

7.2. SCHEDULE

7.3. PARTICIPANTS

7.4. ACCEPTANCE RESULTS

8. ANNEX 1 - REQUIREMENT COVERAGE MATRIX

9. ANNEX 2 - INCLUDED TECHNICAL FACTS

10. ANNEX 3 - TECHNICAL FACTS NOT COVERED

11. TYPICAL WORKSHEET

Project XXXXX		
Acceptance Test No: xxxxxxxx		
Test: XXXXX	Date:	Test release: 1.0
Type: Functional []	Contingency []	Performance []
PURPOSE OF THE TEST:		
PRELIMINARY TESTS:		
TEST DATA:		
EXPECTED RESULTS:		
IMPLEMENTATION PROCEDURE:		
RESULTS:	TEST	ACCEPTED [] REJECTED []
Non-conformance reference:		
Comments:		

11 ANNEX 2: INDEX OF RULES

11.1 CASE OF STANDARD SOFTWARE DEVELOPMENT

Rule Code	Heading	Topic	Page
EX1	Basic principles of the method	Pre-established rules	10
EX2	Basic principles of the method	Application plan	10
EX3	Organisation and product assurance	Organisation and responsibilities	10
EX4	Organisation and product assurance	Training	10
EX5	Organisation and product assurance	Product Assurance Provision	10
EX6	Organisation and product assurance	Minimum Product Assurance	10
EX7	Organisation and product assurance	Extra Product Assurance	10
EX8	Organisation and product assurance	Identification of technical risks	10
EX9	Organisation and product assurance	System risk impact analysis	10
EX10	Organisation and product assurance	Subcontractors	10
EX11	Development cycle	Definition of the development cycle	11
EX12	Development cycle	Requirement analysis	11
EX14	Development cycle	Phases of the development cycle	11
EX16	Development cycle	End-of-phase milestone	11
EX18	Software specification	Minimum specification activities	12
EX19	Software specification	Complementary specification activities	12
EX20	Software specification	Software specification document	12
EX21	Software specification	MMI mock-up	12
EX23	Software specification	Software specification review (SSR)	12
EX24	Design	Minimum design activities	13
EX26	Design	Software design document	13
EX27	Design	Design of reused functions	13
EX28	Coding - Unit tests	Coding and unit test activities	13
Recom1	Coding - Unit tests	Programming language	13
EX29	Coding - Unit tests	Adherence to coding rules	13
EX30	Validation	Validation activities	14
EX31	Validation	Coverage of validation tests	14
EX34	Validation	Stability of software release	14
EX35	Validation	Technical readiness review board - Post Test Review Board	15
EX37	Acceptance	Acceptance phase	16
EX39	Configuration management - Principles	Identification of the elements	16

**SIMPLIFIED QUALITY ASSURANCE
REQUIREMENTS FOR SOFTWARE
DEVELOPMENT**

Rule Code	Heading	Topic	Page
EX40	Configuration management - Principles	Configuration management supervisor	16
EX41	Configuration management - Principles	Configuration management activities	16
Recom2	Configuration management - UNIX	Standard generation / installation procedures	18
EX42	Configuration management - CDF	Configuration description file	18
EX43	Methods, tools, rules - Principles	Selection of methods and tools	18
EX44	Methods, tools, rules - Principles	Justification of tool selection	18
Recom3	Methods, tools, rules - Principles	Use of standard tools	19
EX45	Methods, tools, rules - Nomenclature	Nomenclature rules	19
EX46	Methods, tools, rules - Coding rules	Interface variables	19
EX47	Methods, tools, rules - Coding rules	Self-documentation	19
EX48	Methods, tools, rules - Coding rules	Essential coding rules	19
EX50	Resused software	Selection criteria for the use of existing software	21
EX51	Resused software	Evaluation of software to be adapted	21
EX52	Resused software	Condition of reused software	21
EX53	Resused software	Code modification rules	21
EX54	Resused software	Configuration management of reused software	21
Recom5	Resused software	Stand-alone configuration management of reused software components	21
EX55	Man-machine interface	MMI mock-up	21
EX56	Man-machine interface	Validation of the MMI mock-up	21
EX58	Man-machine interface	User participation in acceptance	21

11.2 CASE OF CRITICAL SOFTWARE DEVELOPPEMENT

Rule Code	Heading	Topic	Page
EX1	Basic principles of the method	Pre-established rules	10
EX2	Basic principles of the method	Application plan	10
EX3	Organisation and product assurance	Organisation and responsibilities	10
EX4	Organisation and product assurance	Training	10
EX5	Organisation and product assurance	Product Assurance Provision	10
EX6	Organisation and product assurance	Minimum Product Assurance	10
EX7	Organisation and product assurance	Extra Product Assurance	10
EX8	Organisation and product assurance	Identification of technical risks	10
EX9	Organisation and product assurance	System risk impact analysis	10
EX10	Organisation and product assurance	Subcontractors	10
EX11	Development cycle	Definition of the development cycle	11
EX12	Development cycle	Requirement analysis	11
EX14	Development cycle	Phases of the development cycle	11
EX16	Development cycle	End-of-phase milestone	11
EX17	Development cycle	Software integrated into an equipment	11
EX18	Software specification	Minimum specification activities	12
EX19	Software specification	Complementary specification activities	12
EX20	Software specification	Software specification document	12
EX21	Software specification	MMI mock-up	12
EX22	Software specification	Determination of critical parts	12
EX23	Software specification	Software specification review (SSR)	12
EX24	Design	Minimum design activities	13
EX25	Design	Complementary design activities	13
EX26	Design	Software design document	13
EX27	Design	Design of reused functions	13
EX28	Coding - Unit tests	Coding and unit test activities	13
Recom1	Coding - Unit tests	Programming language	13
EX29	Coding - Unit tests	Adherence to coding rules	13
EX30	Validation	Validation activities	14
EX31	Validation	Coverage of validation tests	14
EX33	Validation	Software integrated into an equipment	14
EX34	Validation	Stability of software release	14

**SIMPLIFIED QUALITY ASSURANCE
REQUIREMENTS FOR SOFTWARE
DEVELOPMENT**

Rule Code	Heading	Topic	Page
EX35	Validation	Technical readiness review board – Post Test Review Board	15
EX37	Acceptance	Acceptance phase	16
EX38	Acceptance	Software integrated into an equipment	16
EX39	Configuration management - Principles	Identification of the elements	16
EX40	Configuration management - Principles	Configuration management supervisor	16
EX41	Configuration management - Principles	Configuration management activities	16
Recom2	Configuration management - UNIX	Standard generation / installation procedures	18
EX42	Configuration management - CDF	Configuration description file	18
EX43	Methods, tools, rules - Principles	Selection of methods and tools	18
EX44	Methods, tools, rules - Principles	Justification of tool selection	18
Recom3	Methods, tools, rules - Principles	Use of standard tools	19
EX45	Methods, tools, rules - Nomenclature	Nomenclature rules	19
EX46	Methods, tools, rules - Coding rules	Interface variables	19
EX47	Methods, tools, rules - Coding rules	Self-documentation	19
EX48	Methods, tools, rules - Coding rules	Essential coding rules	19
EX49	Dependability	Functional analysis	20
Recom4	Dependability	Design of critical software components	20
EX50	Resused software	Selection criteria for the use of existing software	21
EX51	Resused software	Evaluation of software to be adapted	21
EX52	Resused software	Condition of reused software	21
EX53	Resused software	Code modification rules	21
EX54	Resused software	Configuration management of reused software	21
Recom5	Resused software	Stand-alone configuration management of reused software components	21
EX55	Man-machine interface	MMI mock-up	24
EX56	Man-machine interface	Validation of the MMI mock-up	24
EX58	Man-machine interface	User participation in acceptance	24

11.3 CASE OF AN R&T SOFTWARE DEVELOPMENT

Rule Code	Heading	Topic	Page
EX1	Basic principles of the method	Pre-established rules	10
EX3	Organisation and product assurance	Organisation and responsibilities	10
EX4	Organisation and product assurance	Training	10
EX5	Organisation and product assurance	Product Assurance Provisions	10
EX6	Organisation and product assurance	Minimum Product Assurance	10
EX8	Organisation and product assurance	Identification of technical risks	10
EX9	Organisation and product assurance	System risk impact analysis	10
EX10	Organisation and product assurance	Subcontractors	10
EX11	Development cycle	Definition of the development cycle	11
EX13	Development cycle	Requirement analysis	11
EX15	Development cycle	Iteration criteria	11
EX16	Development cycle	End-of-phase milestone	11
EX18	Software specification	Minimum specifications activities	12
EX21	Software specification	MMI mock-up	12
EX24	Design	Minimum design activities	13
EX27	Design	Design of reused functions	13
EX28	Coding - Unit tests	Coding and unit test activities	13
Recom1	Coding - Unit tests	Programming language	13
EX29	Coding - Unit tests	Adherence to coding rules	13
EX30	Validation	Validation activities	14
EX32	Validation	Validation in iterative development	14
EX34	Validation	Software release stability	14
EX36	Validation	Start- and end-of-validation meetings	16
EX37	Acceptance	Acceptance phase	16
EX39	Configuration management - Principles	Identification of the elements	16
EX40	Configuration management - Principles	Configuration management supervisor	16
EX41	Configuration management - Principles	Configuration management activities	16
Recom2	Configuration management - UNIX	Standard generation / installation procedures	18
EX42	Configuration management - CDF	Configuration description file	18
EX43	Methods, tools, rules - Principles	Selection of methods and tools	18
EX44	Methods, tools, rules - Principles	Justification of tool selection	18
Recom3	Methods, tools, rules - Principles	Use of standard tools	19

**SIMPLIFIED QUALITY ASSURANCE
REQUIREMENTS FOR SOFTWARE
DEVELOPMENT**

Rule Code	Heading	Topic	Page
EX45	Methods, tools, rules - Nomenclature	Nomenclature rules	19
EX46	Methods, tools, rules - Coding rules	Interface variables	19
EX47	Methods, tools, rules - Coding rules	Self-documentation	19
EX48	Methods, tools, rules - Coding rules	Essential coding rules	19
EX50	Resused software	Selection criteria for the use of existing software	21
EX51	Resused software	Evaluation of software to be adapted	21
EX52	Resused software	Condition of reused software	21
EX53	Resused software	Code modification rules	21
EX54	Resused software	Configuration management of reused software	21
Recom5	Resused software	Stand-alone configuration management of reused software components	21
EX55	Man-machine interface	MMI mock-up	23
EX57	Man-machine interface	Validation of the MMI mock-up	23



REFERENTIEL NORMATIF REALISE PAR :
Centre National d'Études Spatiales
Inspection Générale Direction de la Qualité
18 Avenue Edouard Belin
31401 TOULOUSE CEDEX 9
Tel: 05 61 27 31 31 - Fax: 05 61 28 28 49

CENTRE NATIONAL D'ÉTUDES SPATIALES

Head Office: 2 pl. Maurice Quentin 75039 Paris cedex 01 / Tel. (33) 01 44 76 75 00 / Fax: 01 44 46 76 76
RCS Paris B 775 665 912 / Siret: 775 665 912 00082 / Code APE 731Z