CRitical S**YST**em Engineering **A**cce**L**eration

Requirements Quality Analyzer
D607.021

DOCUMENT INFORMATION

Project	CRYSTAL
Grant Agreement No.	ARTEMIS-2012-1-332830
Deliverable Title	Requirements Quality Analyzer
Deliverable No.	D607.021
Dissemination Level	CO
Nature	P
Document Version	V3.00
Date	2013-11-29
Contact	José Fuentes
Organization	REUSE
Phone	+34912172596
E-Mail	jose.fuentes@reusecompany.com

AUTHORS TABLE

Name	Company	E-Mail
José Fuentes	REUSE	jose.fuentes@reusecompany.com

CHANGE HISTORY

Version	Date	Reason for Change	Pages Affected
1.00	2013-10-30	Initial version	All
2.0	2013-11-07	Internal review	Minor changes along the document
3.0	2013-11-28	External review	Brief introduction to WP607. Clarifications to the architecture figure. Reference to specific versions of some tools. Clarification on some metrics. Typo in the reference to other deliverable. Added more elements to the acronym list

CONTENT

Content of Figures

Content of Tables

1 Introduction

1.1 Role of Deliverable

The goal of this deliverable is describing the features included in the Requirements Quality Analyzer brick (B2.29).

This brick is part of WP607. The main goals of this WP are the following:

- CCC approach for quality requirements:
  - Correctness – Consistency - Completeness
- Requirements reuse
- Integration with IOS
- Support to the development supply chain and shareable content

In order to fulfil with the main objectives of the workpackage, this brick is aimed to customize quality metrics, measure the quality of requirements specifications, and, provide quality reports allowing a quick enhancement of specifications by following the set of recommendations.

1.2 Relationship to other CRYSTAL Documents

This deliverable is related to the rest of deliverables and bricks created in WP607 as well as the main deliverables from WP204 (Electrical Flight Control System – since this use case is primarily based on WP607). The level of relationship is the following:

- D607.011: since this document will describe the further needs and goals, the development and finally the assessment of the brick Requirements Quality Analyzer (B2.29)
- D607.031: since the quality configuration customized in Requirements Quality Analyzer (RQA) is also used by Requirements Authoring Tool (B2.30) in the on-the-fly quality analysis
- D607.041: since part of the analysis performed by relies on ontologies and boilerplates managed in knowledgeMANAGER (kM)
- D204.010: since this deliverable provides industrial needs for requirements-based engineering.

1.3 Structure of this Document

The structure of the document is the following:

- Chapter 2 – Current technical features: first we start with the description of the current state of the brick
- Chapter 3 – Training offered to end-users: this chapter lists the training sessions that have been held related to this brick, a link to the training material is also included
- Chapter 4 – Main goals for the brick during the CRYSTAL project: finally, this chapter summarizes what are the most important goals for the industrial partners related to this WP

2 Current Technical Features

This chapter describes the current technical features of the tool Requirements Quality Analyzer. As a first iteration for this brick/deliverable, the set of features described hereinafter correspond to the status of the tool as it is today in the commercial version (version 4.1), available at http://www.reusecompany.com.

2.1 Description of the Tool

Requirements Quality Analyzer (RQA) belongs to the Requirements Quality Suite (RQS), a set of tools aimed to customize, manage and improve the quality of a set of requirements (see deliverables D607.031 and D607.041 for a more detailed description of the other tools included in the suite).

More specifically, the main goals of RQA are (see section 2.3 for a more detailed description):

- Allowing the customization of quality metrics for the whole suite, so that the suite could provide recommendations to different end-users
- Forcing the re-check of the quality for a set of requirements
- Editing individual requirements by following a set of quality hints
- Generating quality reports:
    - Correctness report: including the quality hints for a set of metrics measured individually, i.e. requirement by requirement
    - Completeness report: this report is based on boilerplates and lists all the boilerplates defined to represent a set of different types of requirements, together with the list of requirements matching any of those boilerplates
    - Consistency report: based on the measurement units used in different requirements
    - Coupling analysis: showing those requirements with a similar semantic graph

2.2 Architecture of the Suite

The picture bellow represents the architecture of the whole Requirements Quality Suite, while the rest of the section describes all the boxes in the architecture and how and why RQA is connected to the other tools.

This picture shows the dependencies among different components as blue arrows. Those components may or may not be installed in the same physical node (a Windows based computer), but all of them must be connected to the same LAN.

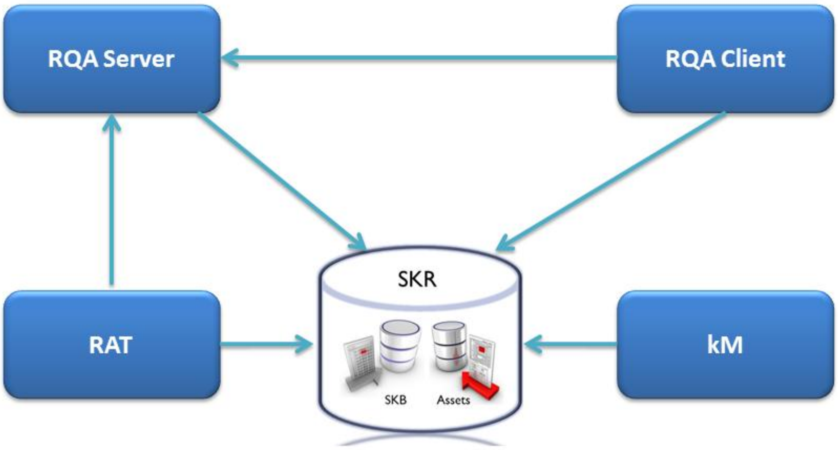


Figure 2-1: RQS architecture

The components of this architecture are the following:

- RQA Server – Requirements Quality Analyzer Server: in charge of the main configuration of the whole suite, e.g. database connection, licensing and low-level database management.
- RQA Client – Requirements Quality Analyzer Client: see other sections of the document to better understand this tool
- RAT – Requirements Authoring Tool: this module allows quality analysis on the fly, that means, when the author is indeed writing the requirements, including recommendations even before the requirements are stored into the Requirements Management System (RMS)
- kM – knowledgeMANAGER: this tool is needed to customize the ontology and boilerplates needed during the quality analysis
- SKR – System Knowledge Repository: this is a relational database (nowadays SQLServer) where we can find two clearly different parts:
    - SKB – System Knowledge Base: represents the main ontology behind all the quality analysis as well as all the information needed to perform Natural Language Processing (NLP) to generate a semantic graph out of a textual requirement
    - Assets: represents the formal representation (mainly as a semantic graph) generated out of every textual requirement

2.3 List of Features

2.3.1 RQS Connectors

RQS is currently connected to some of the most widely used requirements management tools in the market:

- RQA: connected to IBM Rational DOORS (versions 8.x and 9.x), Dassault Systèmes Reqtify (version 2.13), Visure Requirements (version 4.x) and MS Excel (versions 2003, 2007 and 2010)
- RAT: connected as a plugin to IBM Rational DOORS (versions 8.x and 9.x) and MS Excel (versions 2003, 2007 and 2010)

For all of the aforementioned connectors, the corresponding APIs were used, i.e. no interoperable connector has been created yet.

2.3.2 The CCC Approach

RQS is based on the Consistency, Completeness and Correctness (CCC) approach defined in the CESAR project [Allain, 2010].

While Correctness is measured individually, for every single requirement, one by one, Consistency and Completeness are analysed for the whole project or requirements module.

The most developed “C” so far is the Correctness one, while the main goal for the CRYSTAL project is to enhance the Consistency and Completeness analysis of the suite.

2.3.3 Quality Configuration

RQA allows a full customization of the way RQS analyses the quality of a set of requirements. Customizations can be accomplished for different organizations, different teams and even different types of requirements modules.

RQA includes the configuration of:

- Projects or modules to be analysed (here the concepts of project and module are aligned with IBM DOORS)
- Metrics and quality functions for correctness (see section 2.3.4.1 for more detail)
- Special sentences: used in some of the correctness metrics
- Measurement units (magnitudes): used in the consistency analysis based on units
- Domain verbs and domain nouns (ontology): used in some of the correctness metrics
- Coupling tasks: allowing to select those modules whose requirements will be used for the coupling analysis

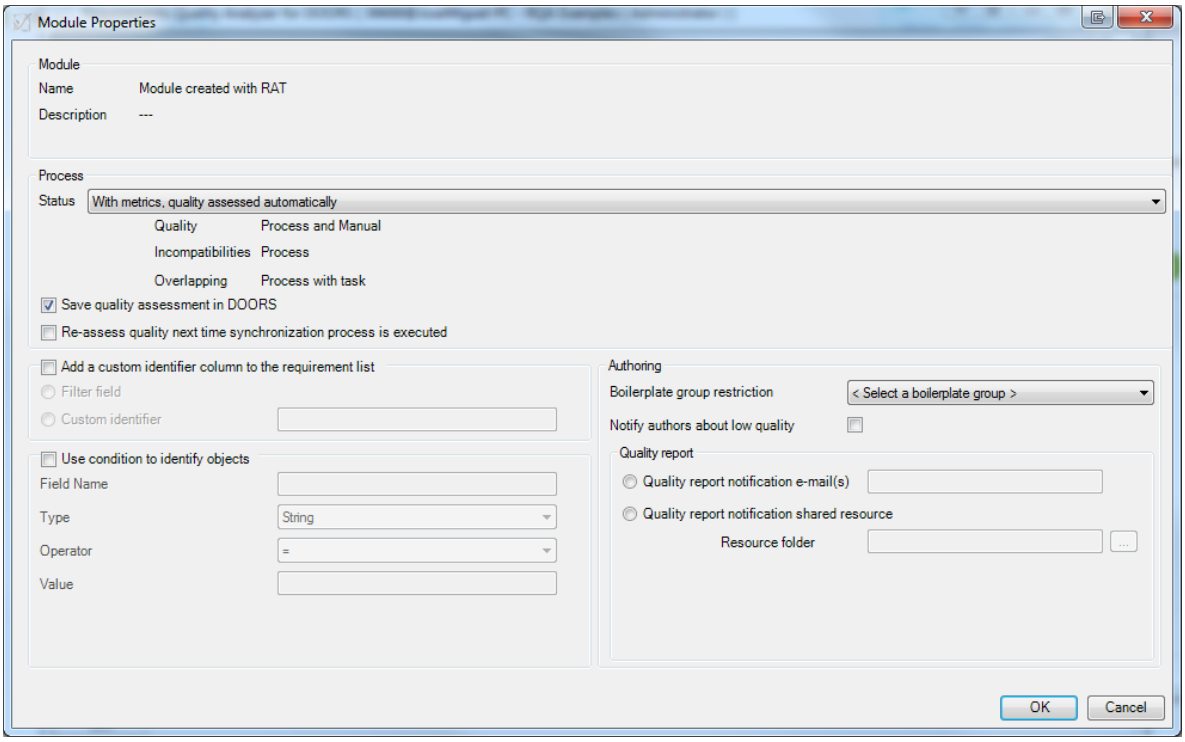
2.3.3.1 Configuration of the Requirements Modules

For every module in the Requirements Management System (RMS), the user may decide whether or not the module should be analysed with RQA. The options are the following:

- Not to be analysed
- To be analysed just manually
- To be analysed both manually and automatically

Once the module has been configured to allow quality analysis, several options allow to (see Figure 2-2):

- Decide whether or not the results of the analysis process should be stored just in the SKR database, or also in the RMS
- Forcing a re-analysis even if the textual content of the requirement has not been changed
- Identify which column is the requirement identifier
- Filter which objects really contain requirements, in order not to analyse other objects with references, descriptions, rationale… (based on a single condition for a field specified during the customization process; e.g. “*id* <> ‘ ‘ “ or “*is_requirement* = *True*”)
- Identify the set of boilerplates to be used in the completeness report (see also Requirements Authoring Tool deliverable D607.031)
- Decide whether or not the author of a requirement must be warned when RQA (Synchronizer) finds a low quality requirement
- Decide whether a quality report shall be issued by mail, stored in a shared repository, or not generated



Module Properties

Module

Name Module created with RAT

Description ---

Process

Status With metrics, quality assessed automatically

Quality Process and Manual

Incompatibilities Process

Overlapping Process with task

☑ Save quality assessment in DOORS

□ Re-assess quality next time synchronization process is executed

□ Add a custom identifier column to the requirement list

○ Filter field

○ Custom identifier

□ Use condition to identify objects

Field Name

Type String

Operator =

Value

Authoring

Boilerplate group restriction < Select a boilerplate group >

Notify authors about low quality □

Quality report

○ Quality report notification e-mail(s)

○ Quality report notification shared resource

Resource folder

Figure 2-2: Module configuration

2.3.4 Correctness Analysis

In this approach, RQA takes every individual requirement, one by one, and gets a series of indicators for every requirement (e.g. text length, readability…). Every indicator is now transformed into a qualitative value thanks to the associated quality function.

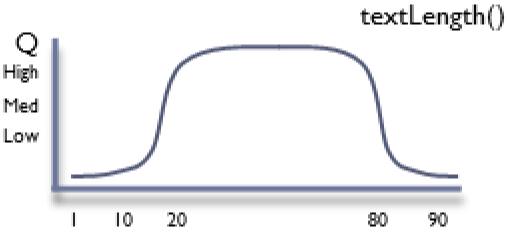
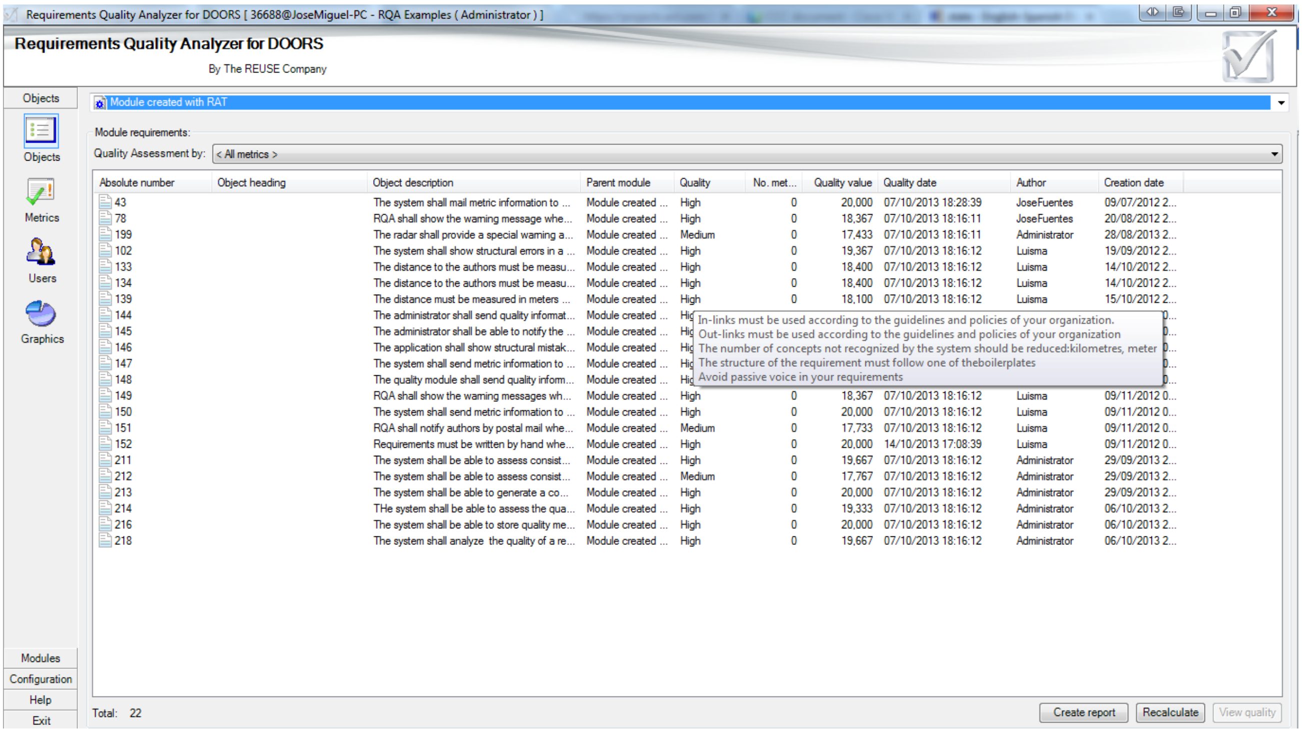


Figure 2-3: Quality functions

During the correctness checking process, every metric rated as medium or low quality will generate a hint that leads the requirement author or reviewer in the best way to get rid of the problem and enhance the quality of the requirement.



Requirements Quality Analyzer for DOORS [36688@JoseMiguel-PC - RQA Examples (Administrator)]

Requirements Quality Analyzer for DOORS

By The REUSE Company

Module created with RAT

Module requirements:

Quality Assessment by: < All metrics >

Absolute number	Object heading	Object description	Parent module	Quality	No. met...	Quality value	Quality date	Author	Creation date
43		The system shall mail metric information to ...	Module created ...	High	0	20,000	07/10/2013 18:28:39	JoseFuentes	09/07/2012 2...
78		RQA shall show the warning message whe...	Module created ...	High	0	18,367	07/10/2013 18:16:11	JoseFuentes	20/08/2012 2...
199		The radar shall provide a special warning a...	Module created ...	Medium	0	17,433	07/10/2013 18:16:11	Administrator	28/08/2013 2...
102		The system shall show structural errors in a ...	Module created ...	High	0	19,367	07/10/2013 18:16:12	Luisma	19/09/2012 2...
133		The distance to the authors must be measu...	Module created ...	High	0	18,400	07/10/2013 18:16:12	Luisma	14/10/2012 2...
134		The distance to the authors must be measu...	Module created ...	High	0	18,400	07/10/2013 18:16:12	Luisma	14/10/2012 2...
139		The distance must be measured in meters ...	Module created ...	High	0	18,100	07/10/2013 18:16:12	Luisma	15/10/2012 2...
144		The administrator shall send quality informat...	Module created ...	Hig					0...
145		The administrator shall be able to notify the ...	Module created ...	Hig					0...
146		The application shall show structural mistak...	Module created ...	Hig					0...
147		The system shall send metric information to ...	Module created ...	Hig					0...
148		The quality module shall send quality inform...	Module created ...	Hig					0...
149		RQA shall show the warning messages wh...	Module created ...	High	0	18,367	07/10/2013 18:16:12	Luisma	09/11/2012 0...
150		The system shall send metric information to ...	Module created ...	High	0	20,000	07/10/2013 18:16:12	Luisma	09/11/2012 0...
151		RQA shall notify authors by postal mail whe...	Module created ...	Medium	0	17,733	07/10/2013 18:16:12	Luisma	09/11/2012 0...
152		Requirements must be written by hand whe...	Module created ...	High	0	20,000	14/10/2013 17:08:39	Luisma	09/11/2012 0...
211		The system shall be able to assess consist...	Module created ...	High	0	19,667	07/10/2013 18:16:12	Administrator	29/09/2013 2...
212		The system shall be able to assess consist...	Module created ...	Medium	0	17,767	07/10/2013 18:16:12	Administrator	29/09/2013 2...
213		The system shall be able to generate a co...	Module created ...	High	0	20,000	07/10/2013 18:16:12	Administrator	29/09/2013 2...
214		THe system shall be able to assess the qua...	Module created ...	High	0	19,333	07/10/2013 18:16:12	Administrator	06/10/2013 2...
216		The system shall be able to store quality me...	Module created ...	High	0	20,000	07/10/2013 18:16:12	Administrator	06/10/2013 2...
218		The system shall analyze the quality of a re...	Module created ...	High	0	19,667	07/10/2013 18:16:12	Administrator	06/10/2013 2...

In-links must be used according to the guidelines and policies of your organization.
Out-links must be used according to the guidelines and policies of your organization
The number of concepts not recognized by the system should be reduced:kilometres, meter
The structure of the requirement must follow one of theboilerplates
Avoid passive voice in your requirements

Total: 22

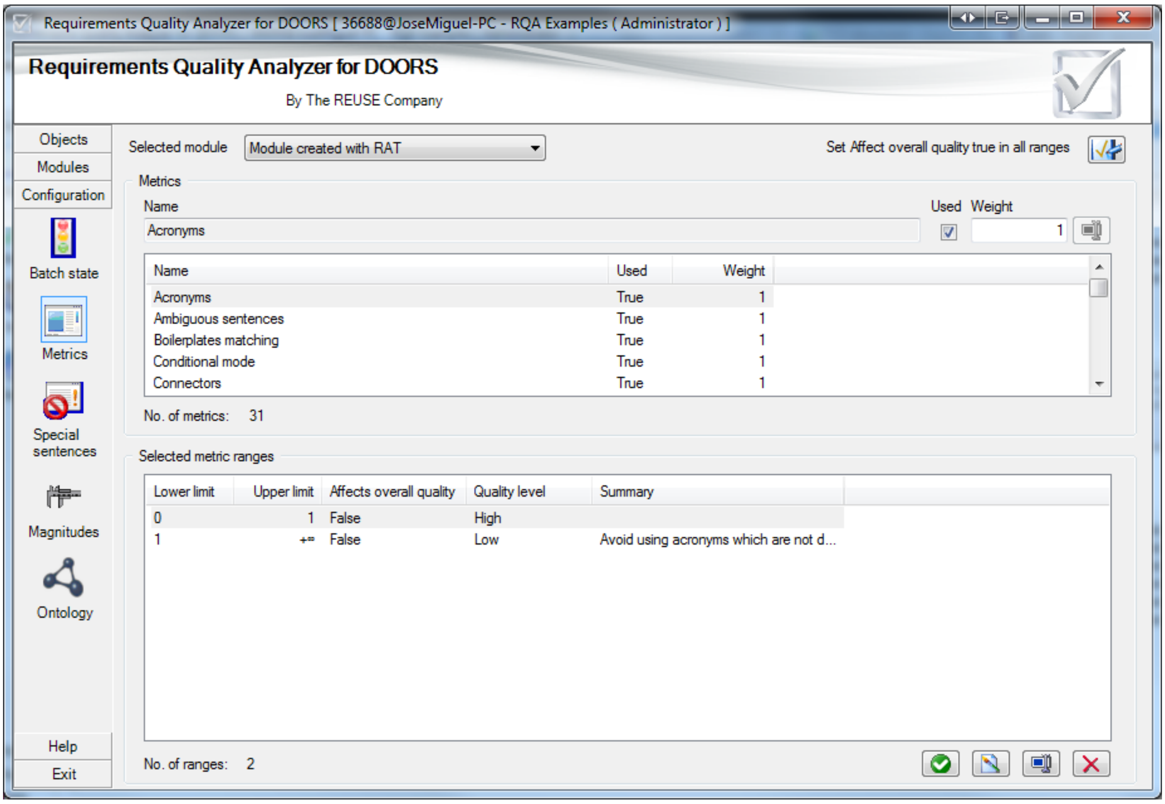
Figure 2-4: Correctness form

2.3.4.1 Quality metrics for correctness

RQA includes more than 30 different metrics that allow to check correctness for individual requirements. Some of these metrics are the following:

- Size: expressed in paragraphs, chars, nouns or verbs. Long requirements will be difficult to understand
- Readability: number of letters between punctuation marks and some other formulas that indicate whether the requirement will be easy to read. Ease to read requirements generates less problems all over the project
- Conditional sentences vs. imperative sentences: avoid “would” and use “shall”, “should” and “will” in the right way
- Active vs. passive voice: avoid using passive voice to increase the readability of the requirement
- Optional sentences: maybe… Optional requirements must be stated by an attribute, never in the body of the requirement
- Ambiguous sentences: fast, user-friendly… Analysts, developers and customers understand ambiguous sentences in different ways
- Subjective sentences: in my opinion, I think that… Don’t show your ideas, but what the system should do
- Implicit sentences: it must be provided by them… Too many pronouns make your requirements difficult to understand
- Abuse of connectors: and, or. Many times connectors reveal different needs enclosed within the same requirement, losing the atomic characteristic
- False friends: customized according to “mother language” of your project
- Negations: no, never… Two or more negations in the same sentence make it difficult to understand
- Speculative sentences: usually, almost always… Make the requirement imprecise
- Design terms: loop, hash… Remember, avoid How, concentrate on What

- Flow terms: while, if, else… Remember avoid How, concentrate on What
- Number of domain nouns and verbs: domain terms and verbs should be involved in the requirement specification, nevertheless, too many different terms in the same requirement often means multiple needs
- Acronyms: avoid those that don’t belong to the domain representation
- Hierarchical levels: don’t complicate your specification with too many indentation levels
- Volatility: if a requirement suffers many changes, you must be very careful with it
- Number of dependencies: the same if your requirement is the source of too many dependences


Requirements Quality Analyzer for DOORS [36688@JoseMiguel-PC - RQA Examples (Administrator)]
Requirements Quality Analyzer for DOORS
By The REUSE Company
Objects
Modules
Configuration
Batch state
Metrics
Special sentences
Magnitudes
Ontology
Help
Exit

Selected module: Module created with RAT

Set Affect overall quality true in all ranges

Metrics

Name: Acronyms Used: ☑ Weight: 1

Name	Used	Weight
Acronyms	True	1
Ambiguous sentences	True	1
Boilerplates matching	True	1
Conditional mode	True	1
Connectors	True	1

No. of metrics: 31

Selected metric ranges

Lower limit	Upper limit	Affects overall quality	Quality level	Summary
0	1	False	High	
1	+∞	False	Low	Avoid using acronyms which are not d…

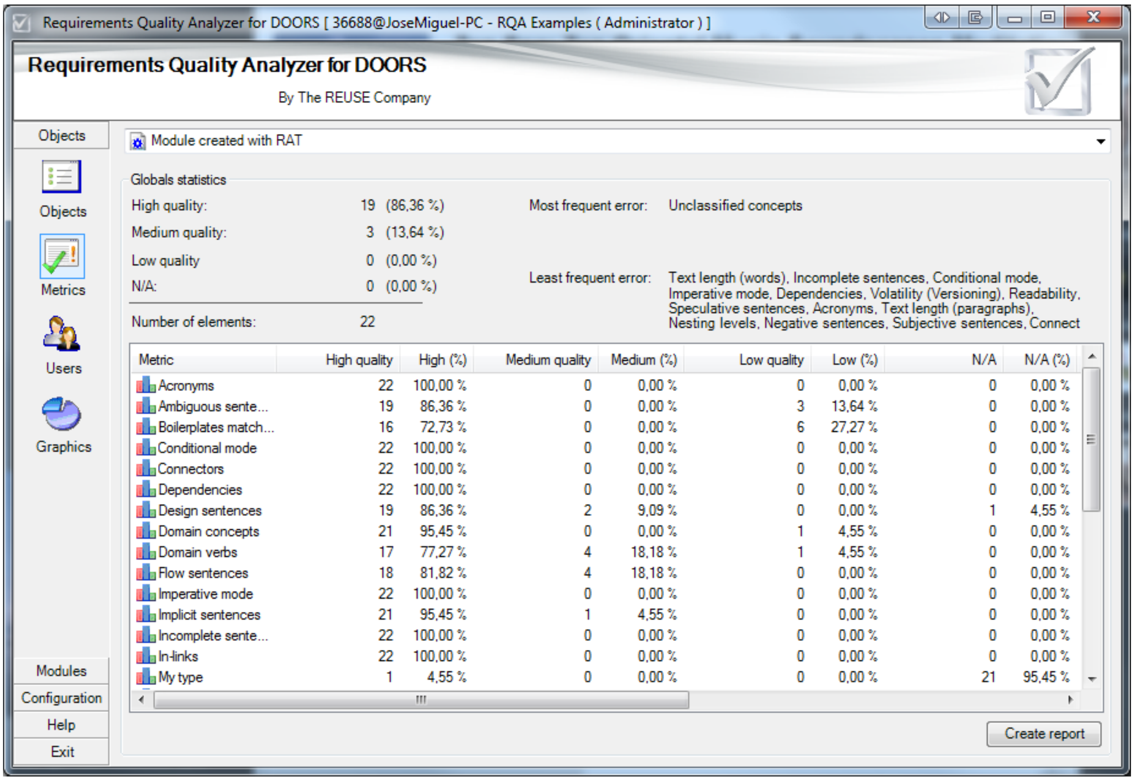
No. of ranges: 2

Figure 2-5: Correctness configuration

2.3.4.2 Correctness Report

RQA is able to create a range of different reports. Together with the screen including all the requirements in the module/project, their quality level and quality hints (see Figure 2-4), other kinds of reports can also be generated.

- Metric report: showing all the correctness metrics, the number of requirements assessed as high/medium/low quality on that metric, maximum and minimum values, average and standard deviations…


Requirements Quality Analyzer for DOORS [36688@JoseMiguel-PC - RQA Examples (Administrator)]
Requirements Quality Analyzer for DOORS
By The REUSE Company
Objects
Objects
Metrics
Users
Graphics
Modules
Configuration
Help
Exit
Module created with RAT
Create report

Globals statistics

High quality:	19	(86,36 %)
Medium quality:	3	(13,64 %)
Low quality	0	(0,00 %)
N/A:	0	(0,00 %)
Number of elements:	22	

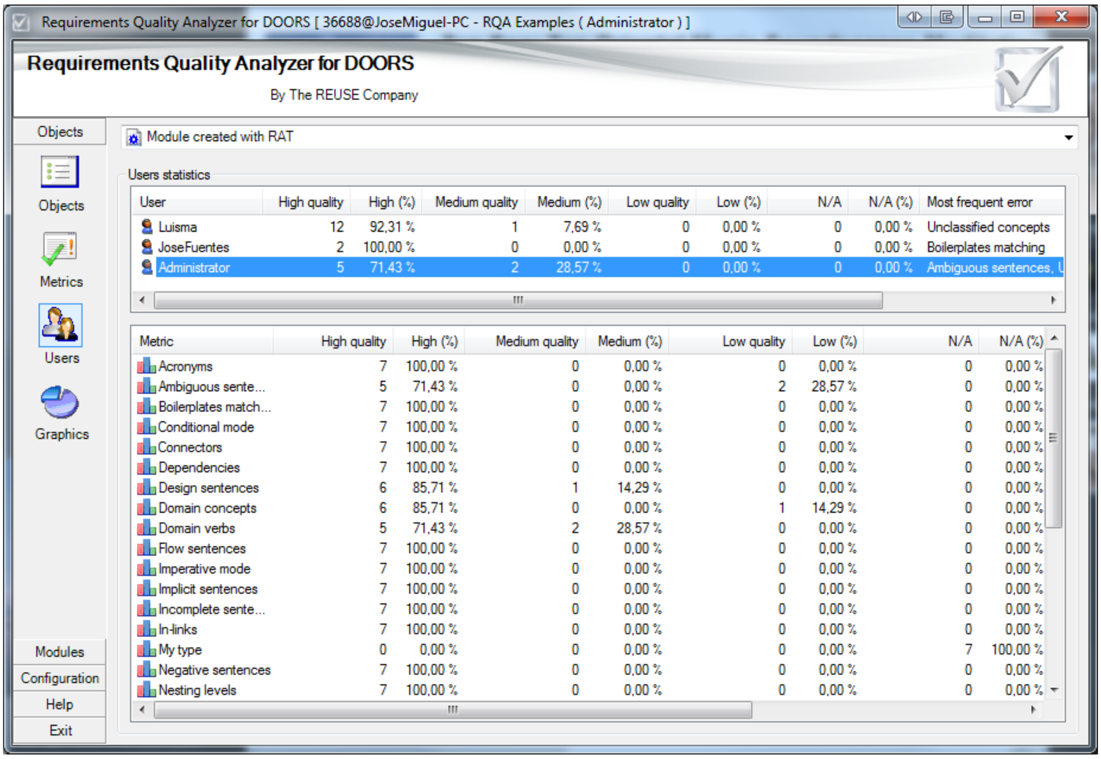
Most frequent error: Unclassified concepts

Least frequent error: Text length (words), Incomplete sentences, Conditional mode, Imperative mode, Dependencies, Volatility (Versioning), Readability, Speculative sentences, Acronyms, Text length (paragraphs), Nesting levels, Negative sentences, Subjective sentences, Connect

Metric	High quality	High (%)	Medium quality	Medium (%)	Low quality	Low (%)	N/A	N/A (%)
Acronyms	22	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
Ambiguous sente...	19	86,36 %	0	0,00 %	3	13,64 %	0	0,00 %
Boilerplates match...	16	72,73 %	0	0,00 %	6	27,27 %	0	0,00 %
Conditional mode	22	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
Connectors	22	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
Dependencies	22	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
Design sentences	19	86,36 %	2	9,09 %	0	0,00 %	1	4,55 %
Domain concepts	21	95,45 %	0	0,00 %	1	4,55 %	0	0,00 %
Domain verbs	17	77,27 %	4	18,18 %	1	4,55 %	0	0,00 %
Flow sentences	18	81,82 %	4	18,18 %	0	0,00 %	0	0,00 %
Imperative mode	22	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
Implicit sentences	21	95,45 %	1	4,55 %	0	0,00 %	0	0,00 %
Incomplete sente...	22	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
In-links	22	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
My type	1	4,55 %	0	0,00 %	0	0,00 %	21	95,45 %

Figure 2-6: Metric report

- Users report: contains the same information than the metric report, but this time representing information author by author:


Requirements Quality Analyzer for DOORS [36688@JoseMiguel-PC - RQA Examples (Administrator)]
Requirements Quality Analyzer for DOORS
By The REUSE Company
Objects
Objects
Metrics
Users
Graphics
Modules
Configuration
Help
Exit
Module created with RAT

Users statistics

User	High quality	High (%)	Medium quality	Medium (%)	Low quality	Low (%)	N/A	N/A (%)	Most frequent error
Luisma	12	92,31 %	1	7,69 %	0	0,00 %	0	0,00 %	Unclassified concepts
JoseFuentes	2	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %	Boilerplates matching
Administrator	5	71,43 %	2	28,57 %	0	0,00 %	0	0,00 %	Ambiguous sentences, U

Metric	High quality	High (%)	Medium quality	Medium (%)	Low quality	Low (%)	N/A	N/A (%)
Acronyms	7	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
Ambiguous sente...	5	71,43 %	0	0,00 %	2	28,57 %	0	0,00 %
Boilerplates match...	7	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
Conditional mode	7	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
Connectors	7	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
Dependencies	7	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
Design sentences	6	85,71 %	1	14,29 %	0	0,00 %	0	0,00 %
Domain concepts	6	85,71 %	0	0,00 %	1	14,29 %	0	0,00 %
Domain verbs	5	71,43 %	2	28,57 %	0	0,00 %	0	0,00 %
Flow sentences	7	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
Imperative mode	7	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
Implicit sentences	7	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
Incomplete sente...	7	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
In-links	7	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
My type	0	0,00 %	0	0,00 %	0	0,00 %	7	100,00 %
Negative sentences	7	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %
Nesting levels	7	100,00 %	0	0,00 %	0	0,00 %	0	0,00 %

Figure 2-7: Users report

- Graphical report: allowing dynamic charting on the correctness data

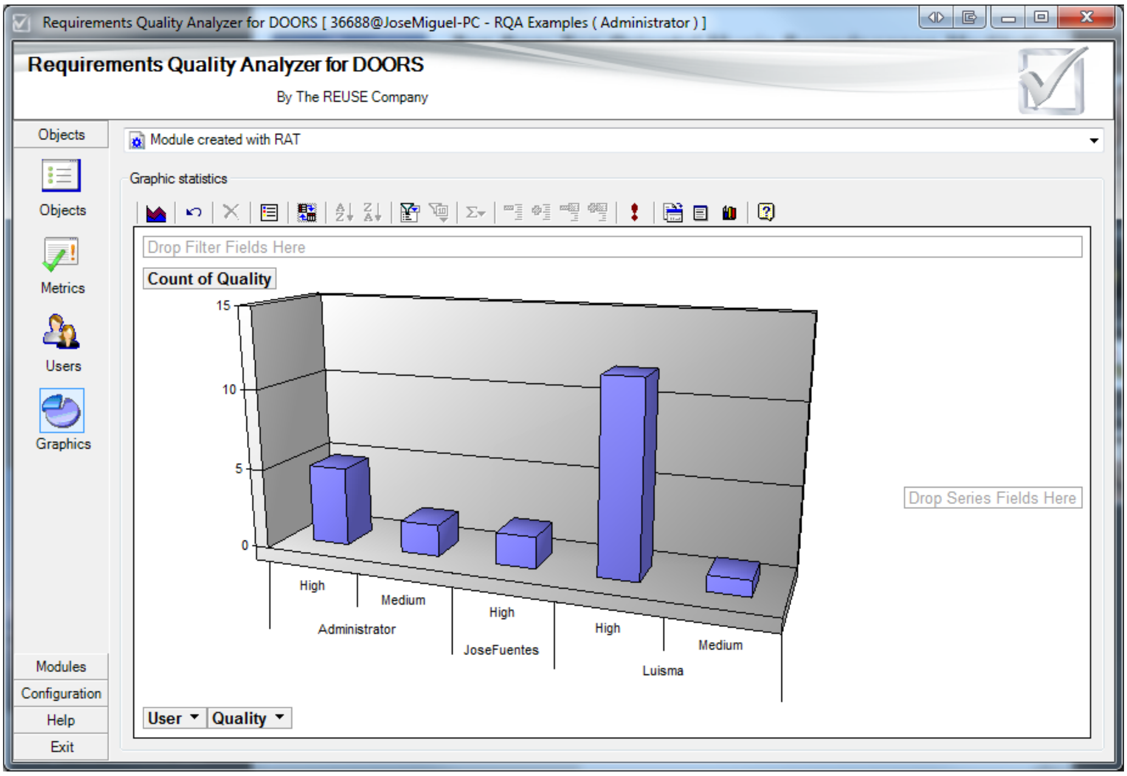
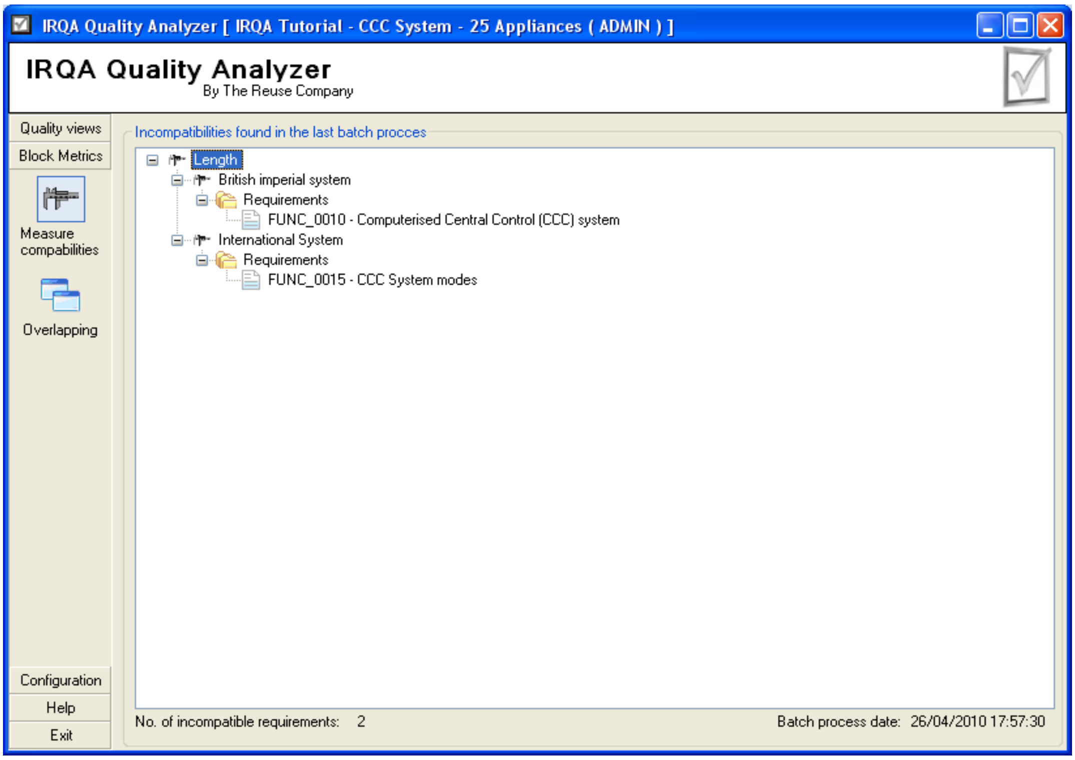


Figure 2-8: Graphical report

- Textual report: including a detailed report that can be printed or exported into the most widely used formats such as PDF, MS Word…

2.3.5 Consistency Analysis

RQA can easily find two or more requirements using inconsistent measurement units. Examples of this lack of consistency could be two requirements where one of them is using *yards* in order to represent the precision an altimeter must take the measurement; and other requirement is using *meters* to represent the minimum distance a target must be in order to be represented in the screen.



Incompatibilities found in the last batch procces

- Length
  - British imperial system
    - Requirements
      - FUNC_0010 - Computerised Central Control (CCC) system
  - International System
    - Requirements
      - FUNC_0015 - CCC System modes

No. of incompatible requirements: 2

Batch process date: 26/04/2010 17:57:30

Figure 2-9: Consistency of units

2.3.6 Completeness Analysis

Based on the proper definition of a taxonomy of requirements types, the user is able to create a set of patterns or boilerplates representing the structure of every type of requirement.

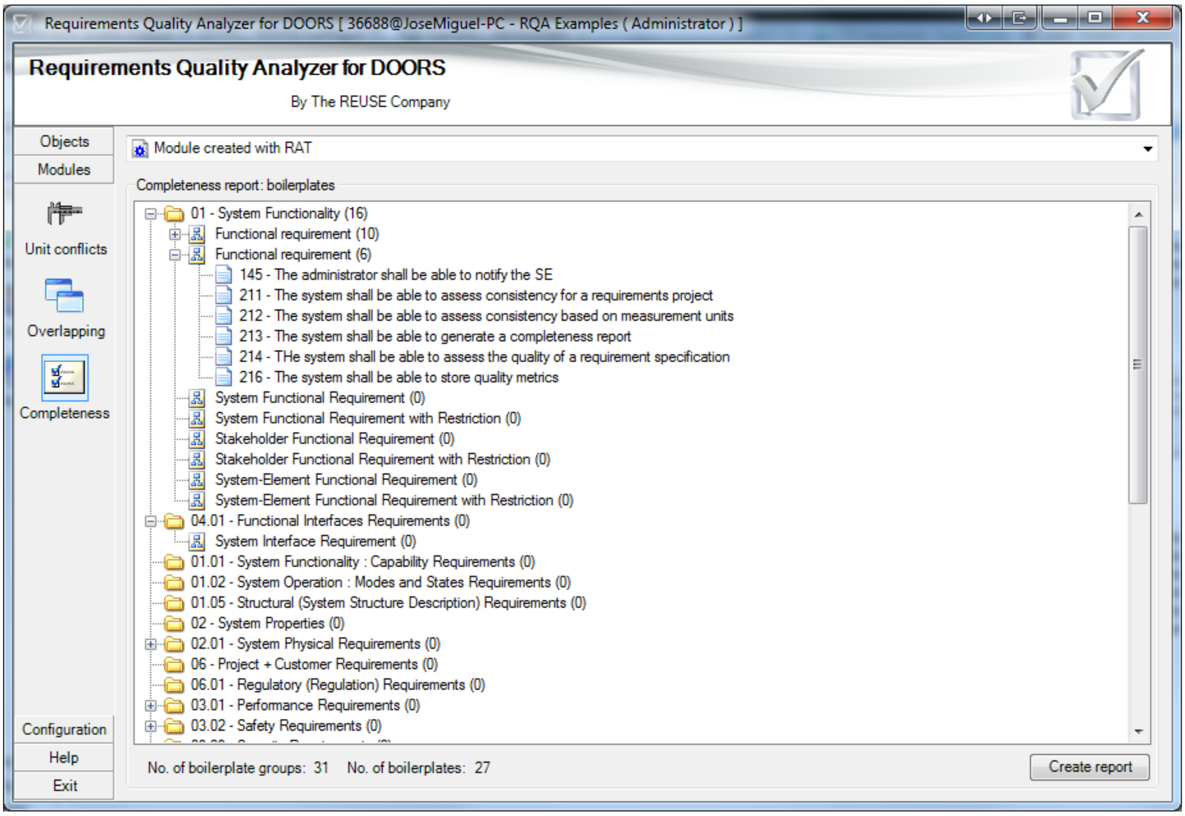
For example, the following requirement:

While in landing mode, in case the button x is pressed, the emergency engine must start

Will match in a boilerplate with the following structure:

While + in + **<mode>** + , + in case + the + **<component>** + to_be + **<trigger>** + , + the + **<component>** + must + **<action>**

By doing that, and once a proper name is given to every different boilerplate, the completeness report will show how many requirements match with any of the different boilerplates available for the project.



Requirements Quality Analyzer for DOORS [36688@JoseMiguel-PC - RQA Examples (Administrator)]
Requirements Quality Analyzer for DOORS
By The REUSE Company
Objects
Modules
Unit conflicts
Overlapping
Completeness
Configuration
Help
Exit
Create report

Module created with RAT

Completeness report: boilerplates

- 01 - System Functionality (16)
  - Functional requirement (10)
  - Functional requirement (6)
    - 145 - The administrator shall be able to notify the SE
    - 211 - The system shall be able to assess consistency for a requirements project
    - 212 - The system shall be able to assess consistency based on measurement units
    - 213 - The system shall be able to generate a completeness report
    - 214 - THe system shall be able to assess the quality of a requirement specification
    - 216 - The system shall be able to store quality metrics
  - System Functional Requirement (0)
  - System Functional Requirement with Restriction (0)
  - Stakeholder Functional Requirement (0)
  - Stakeholder Functional Requirement with Restriction (0)
  - System-Element Functional Requirement (0)
  - System-Element Functional Requirement with Restriction (0)
- 04.01 - Functional Interfaces Requirements (0)
  - System Interface Requirement (0)
- 01.01 - System Functionality : Capability Requirements (0)
- 01.02 - System Operation : Modes and States Requirements (0)
- 01.05 - Structural (System Structure Description) Requirements (0)
- 02 - System Properties (0)
- 02.01 - System Physical Requirements (0)
- 06 - Project + Customer Requirements (0)
- 06.01 - Regulatory (Regulation) Requirements (0)
- 03.01 - Performance Requirements (0)
- 03.02 - Safety Requirements (0)

No. of boilerplate groups: 31 No. of boilerplates: 27

Figure 2-10: Completeness report

2.3.7 Coupling Analysis

Redundancy and inconsistency are two big issues in the requirements engineering process. Thanks to RQA, different requirements, with very different wording, could be matched in case they could share a suspicious similar meaning.

To do that, every requirement is transformed into its semantic representation. Such a representation is a semantic graph.

Thus, two apparently different requirements could be easily identified, allowing the author/reviewer to take the proper steps with both requirements. An example of such a semantic analysis is depicted in the following picture, where two requirements are eventually represented with the same semantic graph.

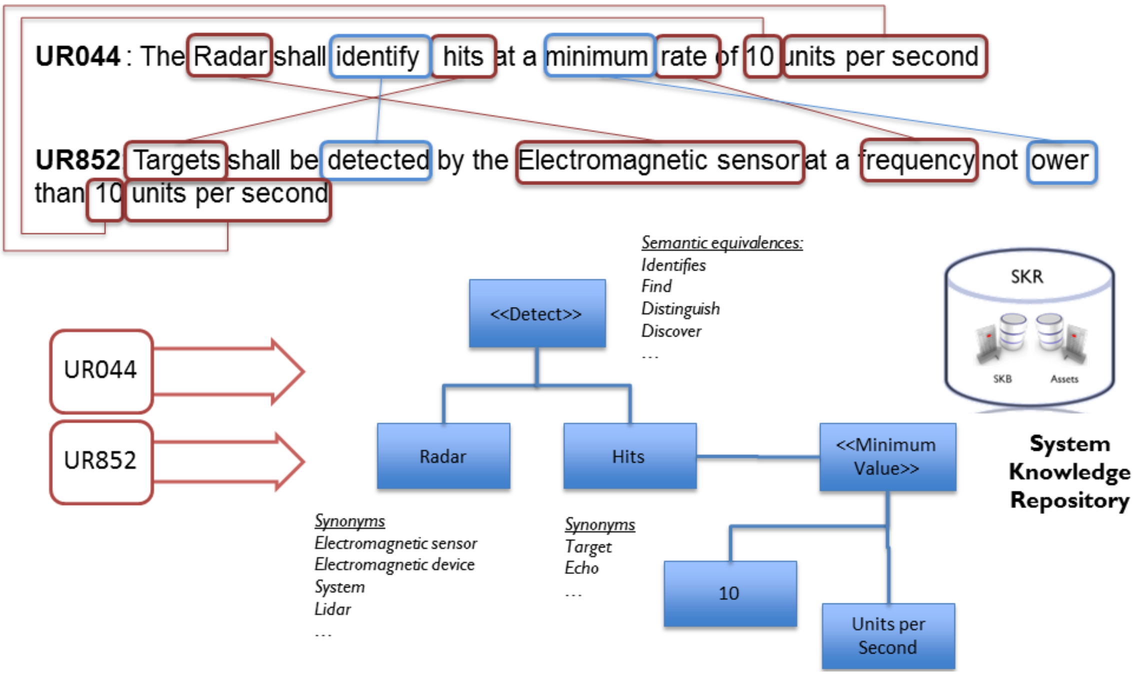

UR044 : The Radar shall identify hits at a minimum rate of 10 units per second
UR852 Targets shall be detected by the Electromagnetic sensor at a frequency not ower than 10 units per second
UR044
UR852
<<Detect>>
Semantic equivalences:
Identifies
Find
Distinguish
Discover
…
Radar
Hits
<<Minimum Value>>
10
Units per Second
Synonyms
Electromagnetic sensor
Electromagnetic device
System
Lidar
…
Synonyms
Target
Echo
…
SKR
SKB
Assets
System Knowledge Repository

Figure 2-11: Coupling analysis

In order to get such a result, the ontology (see knowledgeMANAGER deliverables D607.041) must be populated with enough information to allow the tool to identify the knowledge behind both requirements as similar. Examples of this kind of knowledge in the ontology could be the following:

- Radar *is_a_kind_of* Electromagnetic sensor
- To detect and To identify both have the same semantics (meaning)
- Two different boilerplates have been represented both with a different grammar (structure) but both with a similar formalization to represent the rate > 10 units per second

3 Training Offered to Partners

During the first few months of the project, several training sessions have been scheduled to:

- train industrial partners on how to use RQA
- train other technical partners involved in WP607 on the fundamentals and details about the semantic approach followed by RQS

The training material used for both sessions is available in the CRYSTAL repository:

- Training for end users (14 October 2013): https://projects.avl.com/11/0154/Data%20Exchange/Forms/AllItems.aspx?RootFolder=%2f11%2f0154%2fData%20Exchange%2f001_MEETINGS%2f011_SP6_Meetings%2fWP6_7%2fMeetings%2f2013-10-14%20RQS%20Training&FolderCTID=&View=%7bA036B3F1-CA9C-4631-A46F-C55BDA6D5C01%7d
- Training for technical partners (16 September 2013): https://projects.avl.com/11/0154/Data%20Exchange/Forms/AllItems.aspx?RootFolder=%2f11%2f0154%2fData%20Exchange%2f001_MEETINGS%2f011_SP6_Meetings%2fWP6_7%2fMeetings%2f2013-09-16_RBE%20Training%20about%20the%20tool%20bricks%20%28Madrid%29%2fkM%20Documentation&FolderCTID=&View=%7bA036B3F1-CA9C-4631-A46F-C55BDA6D5C01%7d

4 Main Enhancement Goals

The set of features described so far has been perceived as very valuable for the industrial partners related to WP607; nevertheless, all the partners involved in that workpackage are working on envisaging a set of new features to improve the tool. Those improvements are pretty much focussed on CCC, ontology, customization and interoperability needs and will be addressed in the next versions of the RQA brick:

CCC Approach:

- Support to a set of new techniques for consistency checking, e.g.:
  - Identification of sets of potentially inconsistent requirements based on the knowledge of the used concepts and the structure of the system
    - Formal consistency analysis based on these sets which guarantees a consistency between the requirements
    - Identification of similar duplicate or possible redundant requirements
  - Consistency between the requirements specification and the related SysML models:
    - E.g. the possible transitions in a state machine described in a specification can be checked about those transitions shown in the corresponding SysML state machine.
- Support to a set of new techniques for completeness checking, e.g.:
  - Structural completeness questions addressable by exploiting knowledge of the system structure, e.g:
    - Each interface is addressed in a requirement
    - All instances of environmental conditions are addressed by the requirements
    - Hazards have been addressed in requirements
  - Range Completeness for interface variables, e.g.:
    - Check for certain variables (e.g. those that are used for conditional statements) whether the whole range of the domain is covered by the requirements. This metric will be based on the value of certain attributes from a PBS (*max_value* and *min_value* for a specific attribute of a component)

- New correctness metrics:
  - Deprecated concepts
  - Use of not preferred concepts (synonyms)
  - Use of concepts identified as ambiguous because of their list of more specific concept in the ontology
- Enhance the current in and out-links metric with nominal links where the user could identify the name and direction of the link to quantify

Support for formal requirements:

- Boilerplates with formal semantics to enable formal CCC analysis techniques
- Metrics based on the matching of boilerplates for the whole requirement or part of it. E.g. whether or not the requirement is matching with at least one boilerplate; or even the detection of smaller patterns such as one defined for detecting wrong requirements such as *The system must do action1, action2, action3 and action4 on object1 and object2*

Contracts:

Description of guaranteed properties with explicitly stated assumptions on the context in which a component is embedded, e.g.;

- The guarantee of a component can only restrict the outputs of the component, not the inputs.

- Virtual integration analysis

Customization:

- Customized metrics: allowing the end-user to write the code for their own metrics
- Pre and post-analysis code: this represents a way for the end-users to write customized code to be executed at different particular moments while the quality analysis:
  - Before the analysis: the code will be able to change any of the attributes of the requirement
  - After the analysis: RQA will provide information related to the result of the analysis so that the proper actions could be taken

Interoperability:

- Having in mind that the main goal of this brick, in terms of integration with RM tools, is to integrate RQS with IBM DOORS; an integration in the overall tool chain through the IOS of the CRYSTAL RTP is conceived
- Integration with modelling tools through IOS to check correctness, completeness and consistency
- Enhance collaborative work with RQA along the supply chain, including shared access, locking and conflict resolution

5 Terms, Abbreviations and Definitions

CCC	Correctness, Completeness and Consistency
CESAR	Cost-efficient methods and processes for safety relevant embedded systems
CO	Confidential, only for members of the consortium (including the JU).
CRYSTAL	**CR**itical S**YST**em Engineering **A**cce**L**eration
D	Demonstrator
IOS	Interoperability Specification
kM	knowledgeMANAGER
LAN	Local Area Network
Layout	The arrangement of visual elements in the different screens of the tool
NLP	Natural Language Processing
O	Other
P	Prototype
PBS	Product Breakdown Structure
PP	Restricted to other program participants (including the JU).
PU	Public
R	Report
RAT	Requirements Authoring Tool
RE	Restricted to a group specified by the consortium (including the JU).
RMS	Requirements Management System
RQA	Requirements Quality Analyzer
RQS	Requirements Quality Suite
RTP	Reference Technology Platform
SKB	System Knowledge Base
SKR	System Knowledge Repository
SP	Subproject
UC	Use case
WP	Work Package

Table 5-1: Terms, abbreviations and definitions

6 References

[Allain, 2010]	G. Allain et al; *Completeness/Consistency/Correctness*; CESAR project deliverable, October 2010 (D_SP2_R3.2_M2 Vol4)
[Hull et al, 2009]	Requirements Engineering
[INCOSE, 2012]	Guide for Writing Requirements