



# Webinar

**Host:** José Fuentes

## Content

- Introduction to TRC and RQS
- Correctness analysis, the first dimension of the requirements quality
- Parameterized correctness metrics
- Custom-code correctness metrics
- Parameterized correctness metrics in RAT
- Live demo
- Q&A



# Introduction

# Introduction



José M. Fuentes



jose.fuentes@reusecompany.com



+34 912 17 25 96



@ReuseCompany



## The REUSE Company (TRC)

### Knowledge Centric Systems Engineering

The REUSE Company is specialized in the application of **Semantic Analysis Technologies** to a wide range of industries (Aerospace, Defense, Automotive, Railway, Energy...)

Our main focus is on System/Software **Traceability, Reuse and Quality**. The integration of tools and technology from The REUSE Company facilitates the representation, analysis and exploitation of knowledge allowing for a knowledge-centric systems engineering approach.

Our mission is to promote system/software and knowledge reuse within any organization, by offering processes, methods, tools and services that make it possible. We offer technology that is fully integrated within the organization's production chain.

## Innovative technologies applied to Systems Engineering

### TRC main Customers

#### Aerospace and Defense



#### Automotive



#### Energy



#### Consulting



#### Banking



#### Health care



#### Other industries



## TRC - Our competences



T<sub>(he)</sub> R<sub>(euse)</sub> Q<sub>(ompany)y</sub>

**Trace + Retrieval + Quality**

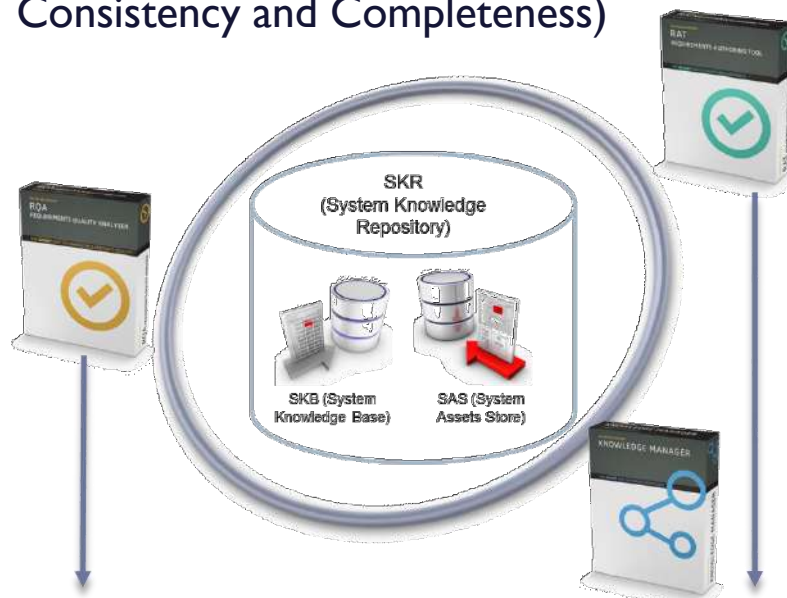
**Towards systematic Reuse**

By means of : **Repositories** containing **Ontologies and Assets**



## RQS – Requirements Quality Suite

- The Requirements Quality Suite (RQS) intends to tackle requirements quality management by offering a set of tools and processes
- Automatic measurement of requirements quality metric
- Support to Requirements Authoring
- RQS models requirements quality metrics using the CCC approach (Correctness, Consistency and Completeness)



- **Requirements Quality Analyzer (RQA):** to setup, check and manage the quality of a requirements specification
- **Requirement Authoring Tool (RAT):** to assist authors while they are creating or editing requirements.
- **Knowledge Manager (KM):** to manage knowledge around a requirements specification: dictionaries, glossaries, concept maps, knowledge models, ontologies, patterns...



**Requirements**

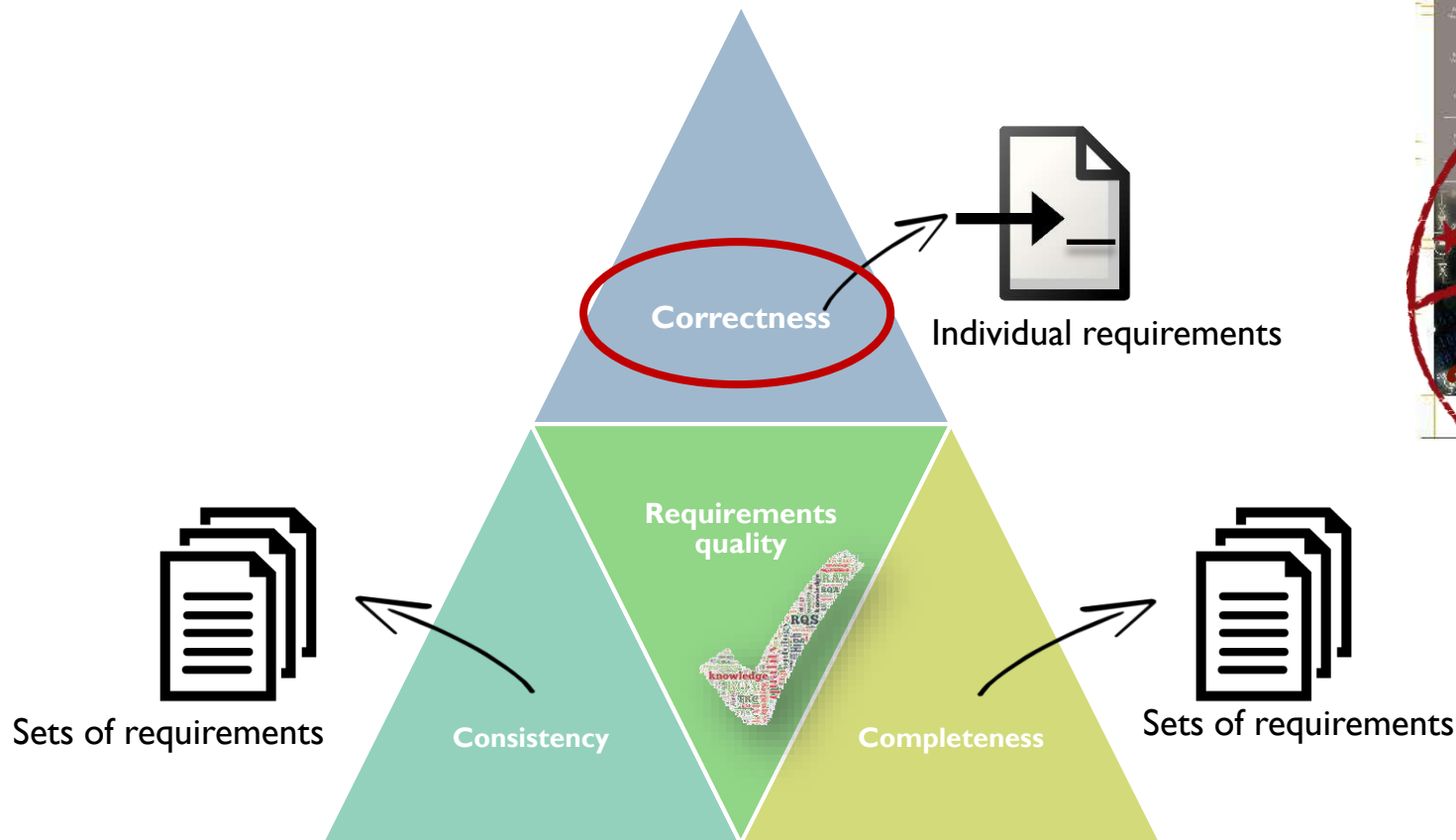
**Quality**

**Correctness  
metrics**



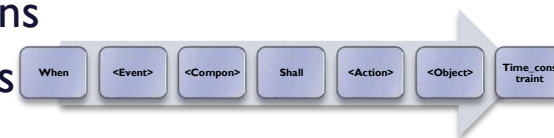
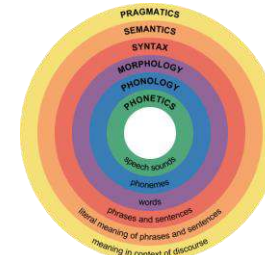
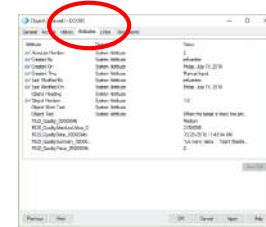
## Requirements quality metrics: CCC Approach

- CCC – Correctness, Consistency and Completeness



## Examples of requirements metrics: Correctness

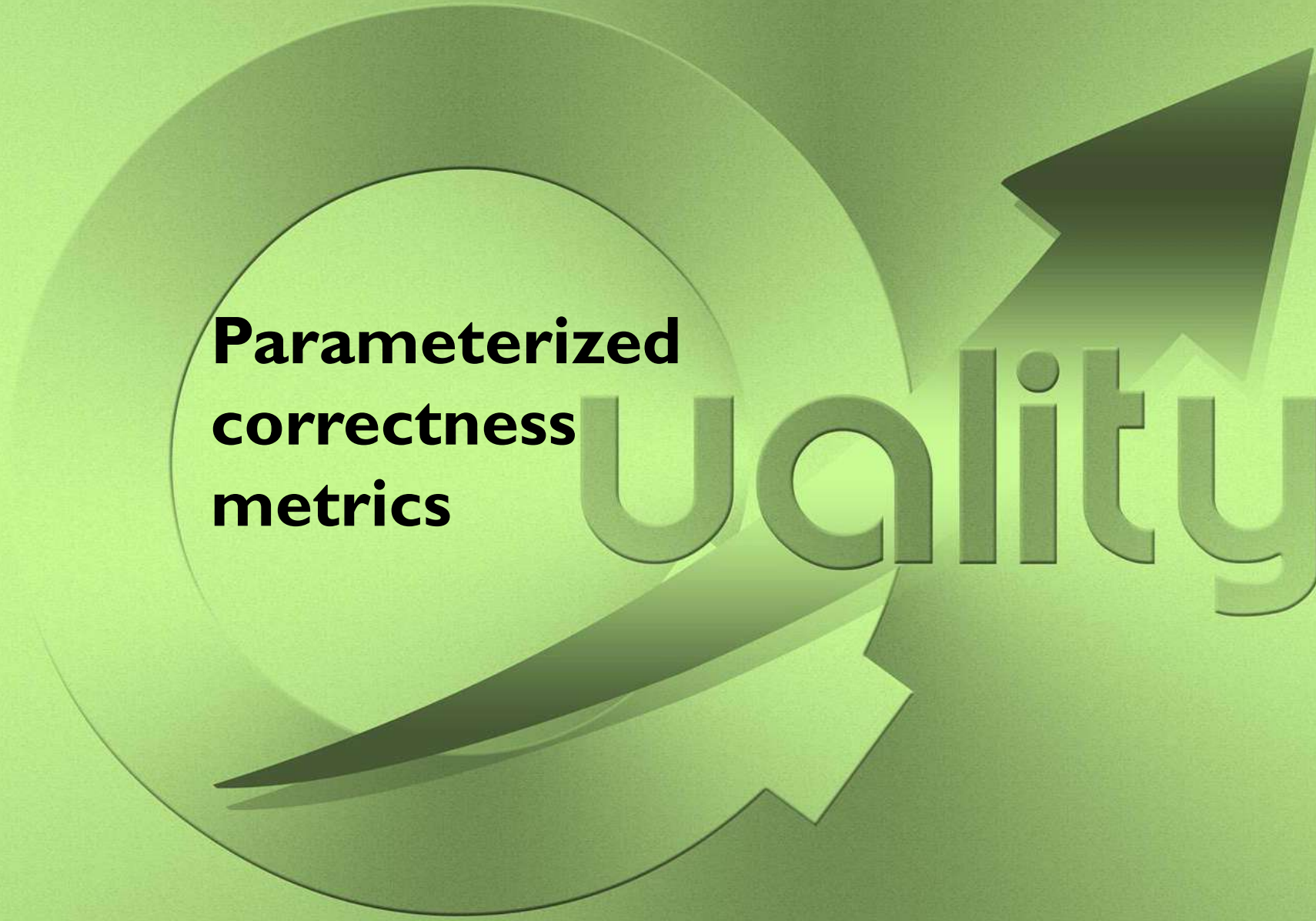
- Metrics based on information coming from the RMS:
  - Attributes, links, versions...
- Metrics based on lists of terms:
  - Forbidden: ambiguous...
  - Restricted: negations, pronouns...
  - Mandatory: 'shall'
- Metrics based on linguistic algorithms:
  - Text length, misspelling....
  - Detection of passive voice, imperative tense...
- Metrics based on the conformance with models:
  - Concepts in your requirements coming from PBS, FBS...
- Metrics based on patterns:
  - Compliance with different types of requirements patterns
  - Detection of specific structures within the requirements



## Correctness metrics

- RQS v15 includes, out-of-the-box +60 metrics
- Two new mechanisms to add more Correctness metrics:
  - **Parameterized metrics:** the behavior can be influenced by a parameter that can be different from one analysis to the other
  - **Custom-coded metrics:** you can create your own library of metrics
- One type of parameterized metric already exists from previous versions:
  - Metrics based on customizable lists of concepts
- What you get with this mechanism:
  - More powerful metrics
  - More flexibility
  - More accurate translation between your guidelines (paper-based) and RQS
  - Different parameterized metrics can be defined at the same time, even of the same type



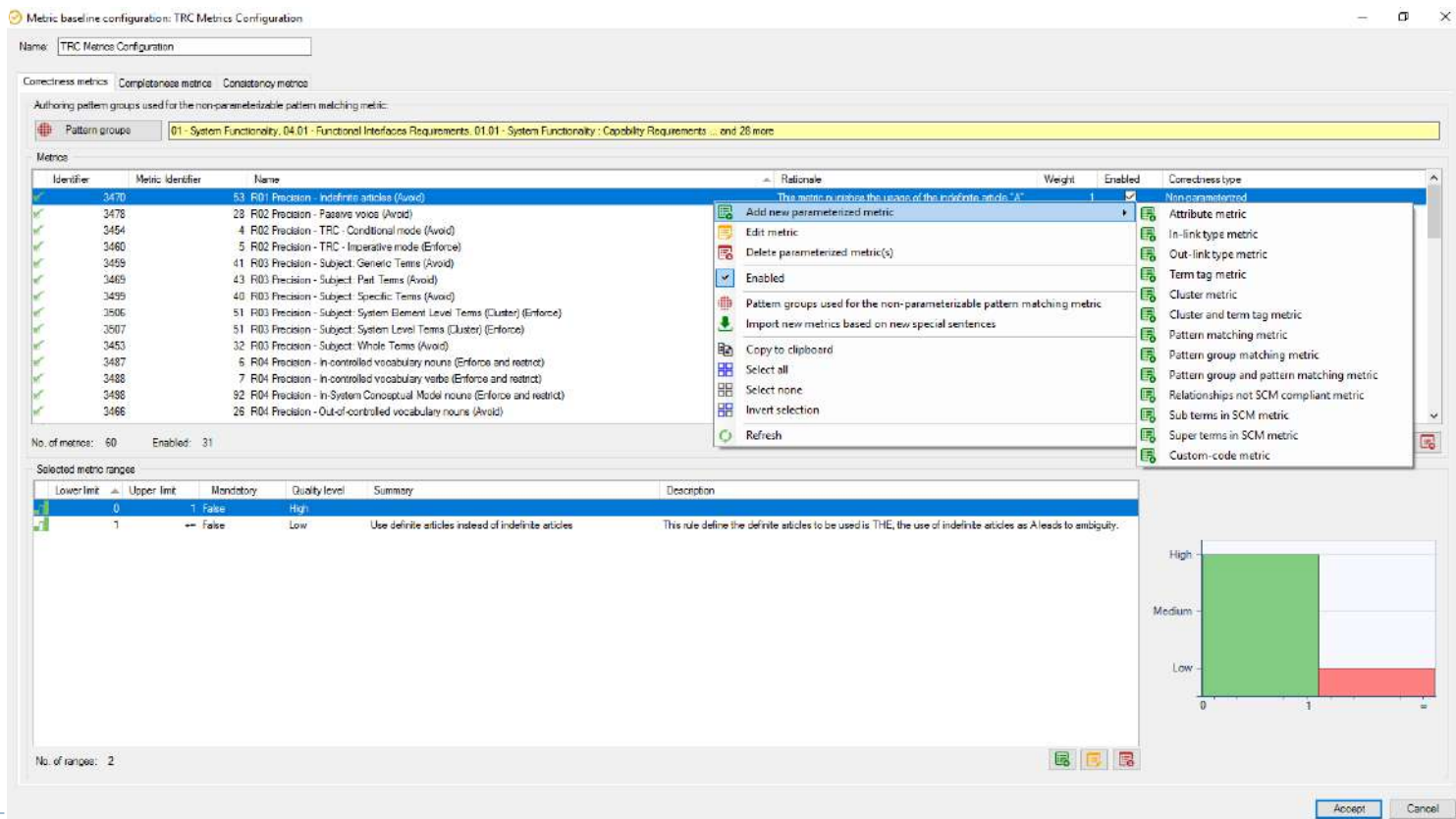


**Parameterized  
correctness  
metrics**

Quality

## Parameterized correctness metrics

- RQS includes a set of *built-in* metrics: out-of-the-box
- Plus the possibility of instantiating a number of parameterized metrics:



The screenshot shows the 'Metric baseline configuration: TRC Metrics Configuration' window. The 'Correctness metrics' tab is active. A list of metrics is displayed with columns for Identifier, Metric Identifier, Name, Rationale, Weight, Enabled, and Correctness type. A context menu is open over the list, showing options like 'Add new parameterized metric', 'Edit metric', 'Delete parameterized metric(s)', 'Enabled', 'Pattern groups used for the non-parameterizable pattern matching metric', 'Import new metrics based on new special sentences', 'Copy to clipboard', 'Select all', 'Select none', 'Invert selection', and 'Refresh'. The 'Correctness type' column shows various metric types such as 'Non-parameterized', 'Attribute metric', 'In-link type metric', 'Out-link type metric', 'Term tag metric', 'Cluster metric', 'Cluster and term tag metric', 'Pattern matching metric', 'Pattern group matching metric', 'Pattern group and pattern matching metric', 'Relationships not SCM compliant metric', 'Sub terms in SCM metric', 'Super terms in SCM metric', and 'Custom-code metric'.

Identifier	Metric Identifier	Name	Rationale	Weight	Enabled	Correctness type
3470	53	R01 Precision - Indefinite articles (Avoid)	The metric punishes the usage of the indefinite article "A"	1	✓	Non-parameterized
3478	28	R02 Precision - Passive voice (Avoid)				Attribute metric
3454	4	R02 Precision - TRC - Conditional mode (Avoid)				In-link type metric
3460	5	R02 Precision - TRC - Imperative mode (Enforce)				Out-link type metric
3459	41	R03 Precision - Subject: Generic Terms (Avoid)				Term tag metric
3465	43	R03 Precision - Subject: Part Terms (Avoid)				Cluster metric
3499	40	R03 Precision - Subject: Specific Terms (Avoid)				Cluster and term tag metric
3506	51	R03 Precision - Subject: System Element Level Terms (Cluster) (Enforce)				Pattern matching metric
3507	51	R03 Precision - Subject: System Level Terms (Cluster) (Enforce)				Pattern group matching metric
3453	32	R03 Precision - Subject: Whole Terms (Avoid)				Pattern group and pattern matching metric
3487	6	R04 Precision - In-controlled vocabulary nouns (Enforce and restrict)				Relationships not SCM compliant metric
3488	7	R04 Precision - In-controlled vocabulary verbs (Enforce and restrict)				Sub terms in SCM metric
3488	92	R04 Precision - In-System Conceptual Model nouns (Enforce and restrict)				Super terms in SCM metric
3466	28	R04 Precision - Out-of-controlled vocabulary nouns (Avoid)				Custom-code metric

No. of metrics: 60    Enabled: 31

Selected metric ranges

Lower limit	Upper limit	Mandatory	Quality level	Summary	Description
0	1	False	High		
1		False	Low	Use definite articles instead of indefinite articles	This rule define the definite articles to be used is THE, the use of indefinite articles as A leads to ambiguity.

No. of ranges: 2

High  
Medium  
Low

0 1 2

Accept Cancel

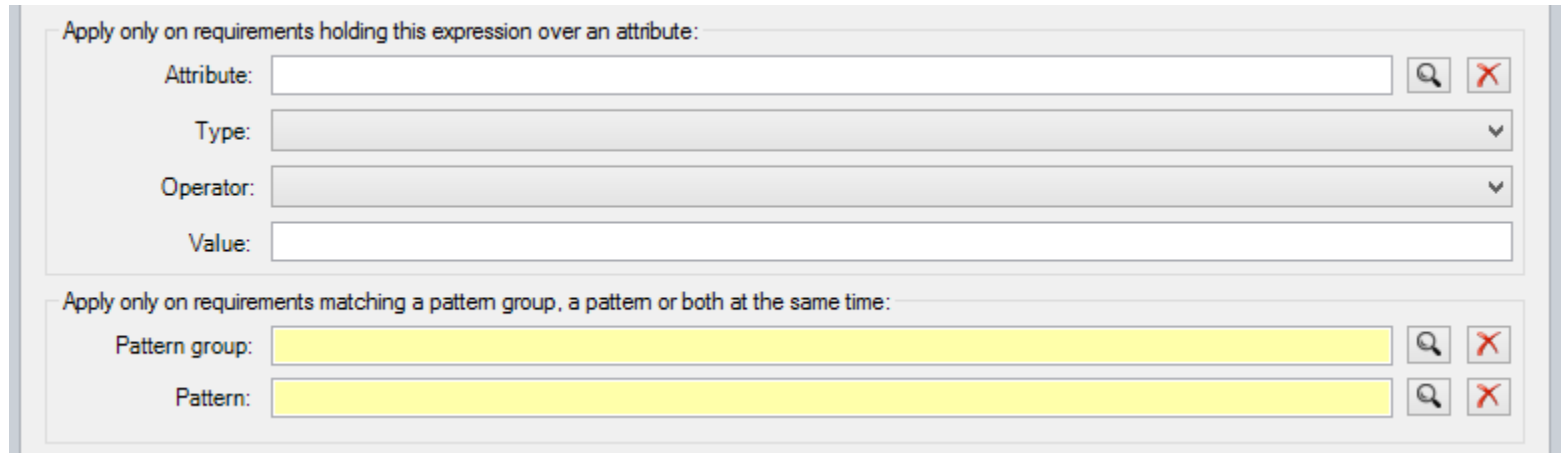


## Parameterized correctness metrics

- As the rest of the metrics, the parameterized metrics:
  - Are seamlessly integrated with other metrics
  - Return a numerical value
  - Use quality functions to transform this value into a high-medium-low scale
  - Use weight to compute the overall quality level of the requirement

## Parameterized correctness metrics

- Unlike the other OOTB metrics:
  - Can include an additional condition to filter out which requirements will not be assessed with a specific metric. Filtering is based on *attributes* or *patterns*

A screenshot of a software interface for filtering requirements. It contains two main sections. The first section is titled 'Apply only on requirements holding this expression over an attribute:' and includes four input fields: 'Attribute:', 'Type:', 'Operator:', and 'Value:'. Each field has a search icon and a red 'X' icon to its right. The second section is titled 'Apply only on requirements matching a pattern group, a pattern or both at the same time:' and includes two input fields: 'Pattern group:' and 'Pattern:'. These fields are highlighted in yellow and also have search and red 'X' icons to their right.

- Some might not be available at metric baseline level, but rather at module specific level: e.g. attributes and links
- Some are not available for some *not real RMS* (e.g. Excel, XML)

## Parameterized correctness metrics. Returned value

- The quantitative result is computed thanks to a quality function, as in any other OOTB metric. E.g. for *Attribute expression matching*:
  - $[0, 1)$  : the expression was not matched → Low quality
  - $[1, \infty)$  : the expression was matched → High quality

Selected metric ranges

Lower limit	Upper limit	Mandatory	Quality level	Summary	Description
0	1	False	Low	The structure of the requireme...	A style guide is important to ensure that requirements are written in a consistent m...
1	+	False	High		

No. of ranges: 2

Modify metric range

Range information:

Quality level: High

☒ Range: 1 ☐ Set as  $-\infty$

Lower limit included ( $\geq$ ), except when it is preceded by a point ( $>$ )

Upper limit:  ☒ Set as  $+\infty$

Upper limit excluded ( $<$ )

☐ Point:

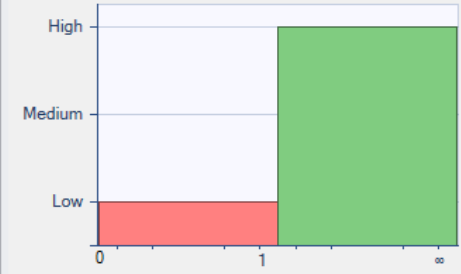
Point value:

Summary: Ambiguous sentences must be avoided

Description: Ambiguous sentences make the requirement difficult to understand, and can provoke other stakeholders to understand something different than the idea initially planned by the author of the requirement.

Mandatory: ☐

Accept
Cancel



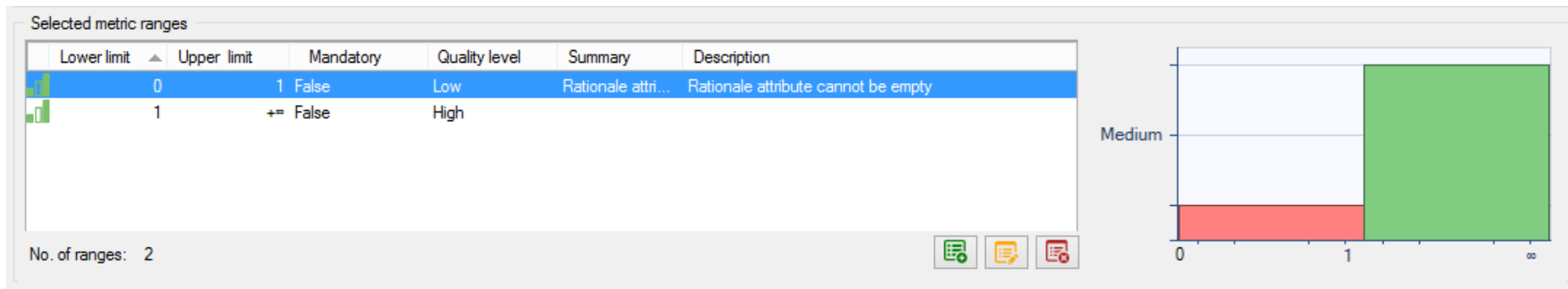
## Special sentences

- Some lists are included as OOTB metrics
- These OOTB lists can be edited as needed
- New lists can also be created and managed, related to new metrics

[illegible]

## Parameterized Attribute metric

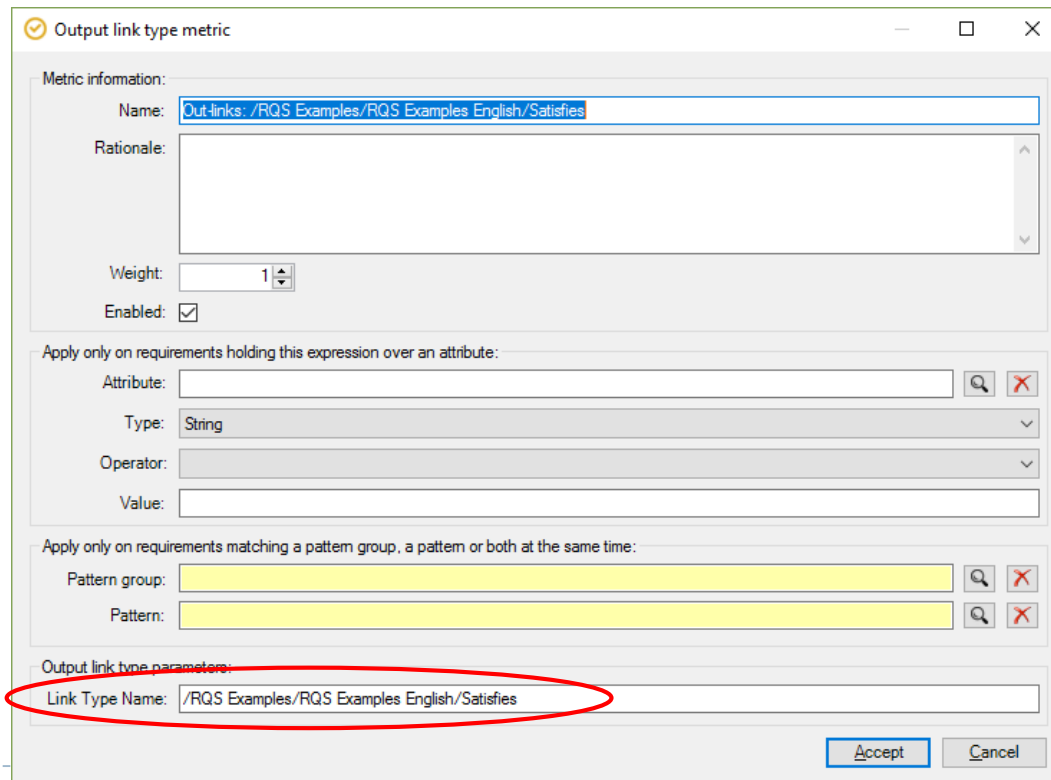
- Evaluation of the value of an attribute, looking for matching with an expression
  - Numbers:  $>$ ,  $>=$ ,  $<$ ,  $<=$ ,  $=$
  - Strings:  $=$ ,  $\neq$ , *regular expression* ([link](#) to an example based on reg expressions)
  - Boolean:  $=$ ,  $\neq$
- Possible return values:
  - Value = 1: the expression was matched  $\rightarrow$  high quality
  - Value = 0: the expression was not matched  $\rightarrow$  low quality
  - Value = 0: when the attribute doesn't exist





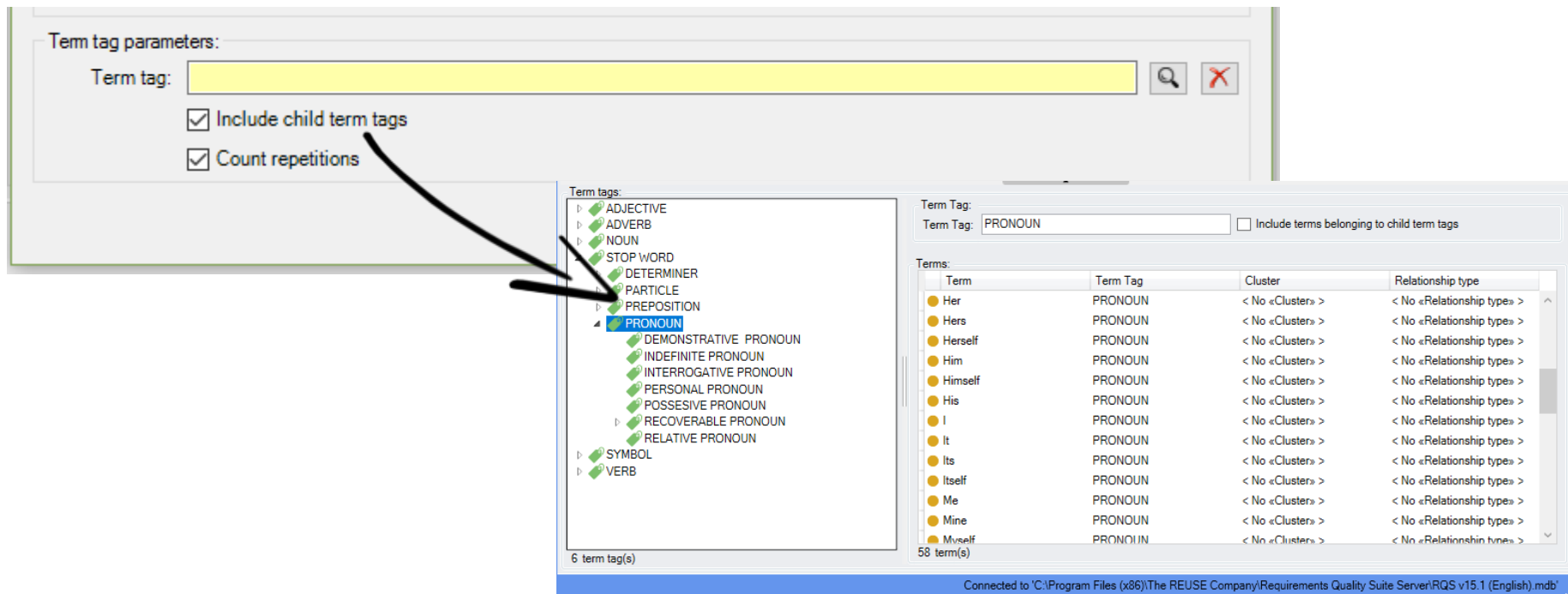
## Parameterized in/out-link type metric

- Returns the number of links (*in* or *out*)
- Unlike the OOTB metric, the parameter represents a specific link module
  - N.B.: for DOORS, provide the **full path** of the Link Module

A screenshot of a software dialog box titled 'Output link type metric'. The dialog has a title bar with standard window controls. It contains several sections: 'Metric information' with fields for 'Name' (containing '/RQS Examples/RQS Examples English/Satisfies'), 'Rationale' (a large text area), 'Weight' (a spinner set to 1), and 'Enabled' (checked). Below this is a section 'Apply only on requirements holding this expression over an attribute:' with fields for 'Attribute', 'Type' (set to 'String'), 'Operator', and 'Value'. Another section 'Apply only on requirements matching a pattern group, a pattern or both at the same time:' has fields for 'Pattern group' and 'Pattern'. At the bottom, a section 'Output link type parameters:' contains a field 'Link Type Name' with the value '/RQS Examples/RQS Examples English/Satisfies', which is circled in red. 'Accept' and 'Cancel' buttons are at the bottom right.

## Parameterized term tag metric

- This metric counts how many terms of a given syntactic tag appear in a specific requirement
- Example: let's detect the number of pronouns to forgive the use of pronouns in a requirement



The screenshot shows the 'Term tag parameters' dialog box with the 'Term tag' field empty. Below it, two checkboxes are checked: 'Include child term tags' and 'Count repetitions'. A black arrow points from the 'Include child term tags' checkbox to the 'PRONOUN' tag in the 'Term tags' list. The 'Term tags' list is expanded, showing a hierarchy of tags: ADJECTIVE, ADVERB, NOUN, STOP WORD, DETERMINER, PARTICLE, PREPOSITION, PRONOUN (selected), DEMONSTRATIVE PRONOUN, INDEFINITE PRONOUN, INTERROGATIVE PRONOUN, PERSONAL PRONOUN, POSSESSIVE PRONOUN, RECOVERABLE PRONOUN, RELATIVE PRONOUN, SYMBOL, and VERB. The 'Terms' table below shows a list of terms and their corresponding tags, clusters, and relationship types.

Term	Term Tag	Cluster	Relationship type
Her	PRONOUN	< No «Cluster» >	< No «Relationship type» >
Hers	PRONOUN	< No «Cluster» >	< No «Relationship type» >
Herself	PRONOUN	< No «Cluster» >	< No «Relationship type» >
Him	PRONOUN	< No «Cluster» >	< No «Relationship type» >
Himself	PRONOUN	< No «Cluster» >	< No «Relationship type» >
His	PRONOUN	< No «Cluster» >	< No «Relationship type» >
I	PRONOUN	< No «Cluster» >	< No «Relationship type» >
It	PRONOUN	< No «Cluster» >	< No «Relationship type» >
Its	PRONOUN	< No «Cluster» >	< No «Relationship type» >
Itself	PRONOUN	< No «Cluster» >	< No «Relationship type» >
Me	PRONOUN	< No «Cluster» >	< No «Relationship type» >
Mine	PRONOUN	< No «Cluster» >	< No «Relationship type» >
Myself	PRONOUN	< No «Cluster» >	< No «Relationship type» >

6 term tag(s)  
58 term(s)

Connected to 'C:\Program Files (x86)\The REUSE Company\Requirements Quality Suite Server\RQS v15.1 (English).mdb'

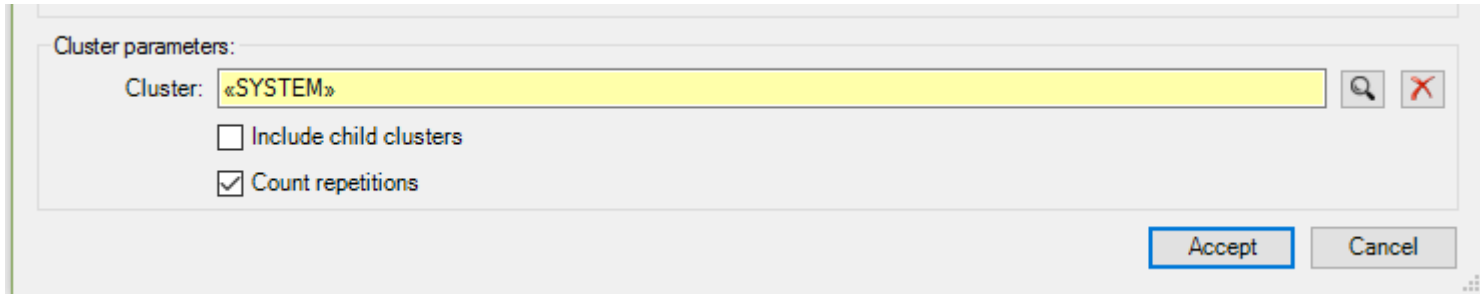
## Parameterized term tag metric

- Unlike the *special list based* metrics, this one uses semantic algorithms to disambiguate
- Example:
  - The IT Department shall generate a monthly report, it has to be sent to them



## Parameterized semantic cluster metric

- Measures the number of terms, belonging to a specific semantic cluster (the parameter), that appears in a requirement
- Example: let's force every system requirement to involve –at least- the name of one system



Cluster parameters:

Cluster: «SYSTEM»

☐ Include child clusters

☒ Count repetitions

Accept Cancel

- Depending on how the quality function is created, RQA can detect when a cluster element is expected but not found, or when it's not expected but found. In this example: OK for System, KO for System Elements

Selected metric ranges

	Lower limit	Upper limit	Mandatory	Quality level	Summary	Description
	0	1	False	Low	A System Re...	A System Requirement mus...
	1	++	False	High		

No. of ranges: 2

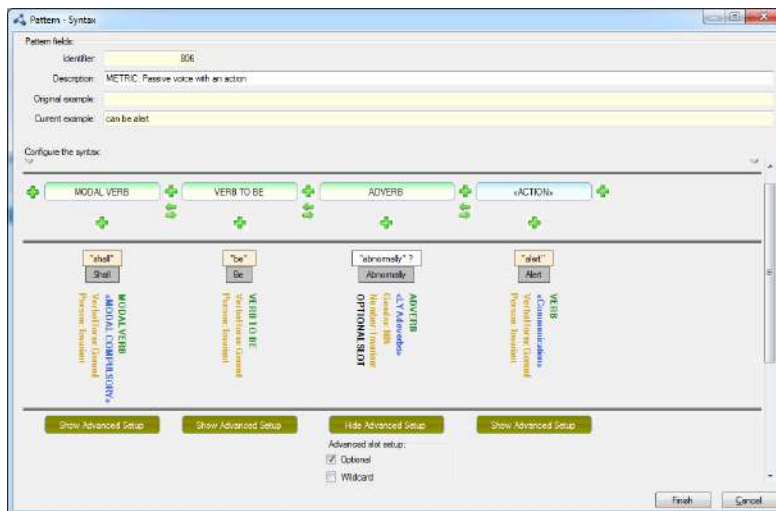
Selected metric ranges

	Lower limit	Upper limit	Mandatory	Quality level	Summary	Description
	0	1	False	High		
	1	++	False	Low	System elem...	The details of the system d...

No. of ranges: 2

## Parameterized pattern matching metric

- Receives a specific pattern as the parameter (watch this [Webinar](#) for more info about patterns) and counts how many times (if any) the pattern appears in the requirement
- N.B.:** Only the selected pattern is checked for a possible match, regardless the weight of the rest of the patterns
- Example: let's look for passive voice using patterns in order to avoid passive voice in a requirement



	Lower limit	Upper limit	Mandatory	Quality level	Summary	Description
	0	1	False	High		
	1	++	True	Low	Avoid passive voice	Avoid passive voice in your requirements. An e...

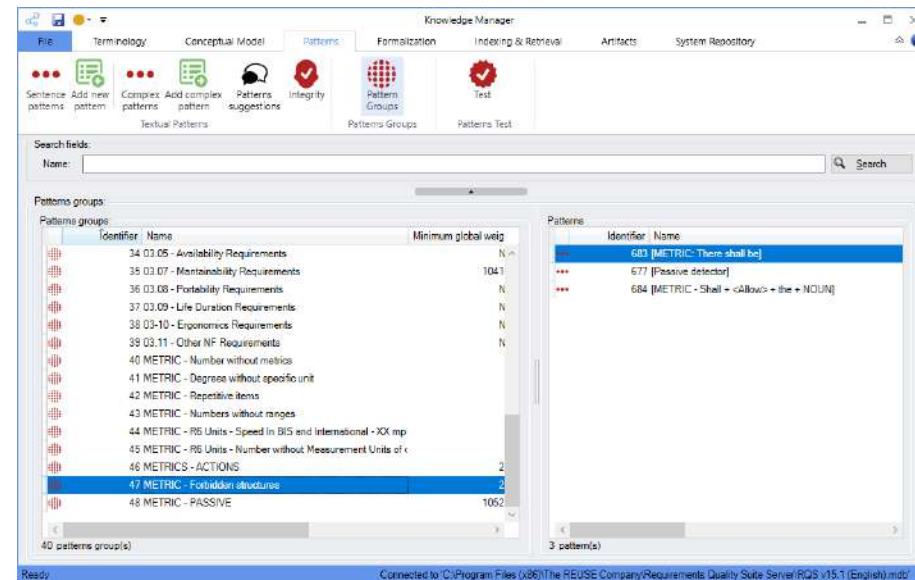
No. of ranges: 2





## Parameterized pattern group matching metric

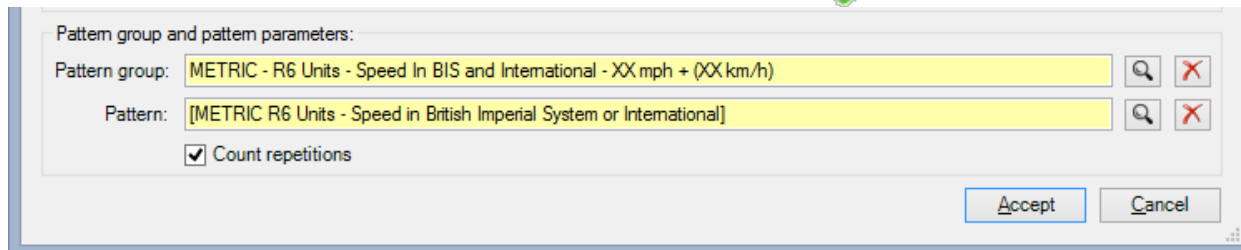
- Computes the number of times any of the patterns belonging to the selected pattern group (the parameter) is matched in a requirement
  - The weight of other patterns not belonging to the group are not interfering in this metric
- Example: let's try to identify a set of wrongly written requirements so that RQA/RAT can easily identify them. In this example, the following types of requirements will be identified:

- “There shall be ....”: every requirement must clearly state the subject of the action
- “... shall allow + NOUN” for example, “The system shall allow the activation...”. The action must be a verb instead of a noun
- Passive structures



## Parameterized pattern group and pattern matching metric

- This metrics takes a pattern group as a parameter (par#1) and a specific pattern into this group as the second parameter (par#2)
  - The weight of other patterns not belonging to this group are **not** interfering
  - But all the patterns that do belong to the group will “compete” with the rest by using their weights
- Example:
  - Let`s detect if every time a speed is stated in *mph*, the corresponding speed in *km/h* is also stated
  - E.g. 1: “When the speed of the car is below 5 mph, the driver shall...” 
  - E.g. 2: “When the speed of the car is below 5 mph (8 km/h), the driver shall...” 

A screenshot of a software dialog box titled 'Pattern group and pattern parameters:'. It contains two input fields: 'Pattern group:' with the text 'METRIC - R6 Units - Speed In BIS and International - XX mph + (XX km/h)' and 'Pattern:' with the text '[METRIC R6 Units - Speed in British Imperial System or International]'. Both fields have search and delete icons to their right. Below the fields is a checkbox labeled 'Count repetitions' which is checked. At the bottom right are 'Accept' and 'Cancel' buttons.

Pattern group and pattern parameters:

Pattern group: METRIC - R6 Units - Speed In BIS and International - XX mph + (XX km/h)

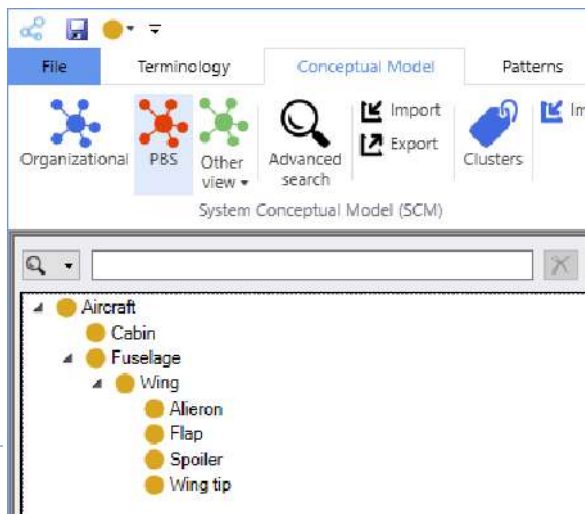
Pattern: [METRIC R6 Units - Speed in British Imperial System or International]

☒ Count repetitions

Accept Cancel

## Parameterized RSHP not in SCM

- Takes a specific type of relationship (a view in the SCM) as parameter
  - In the requirement, it takes all the relationships, with the selected type, that are generated (formalized based on requirements patterns)
  - For each couple of related terms in the requirement, it checks whether those related terms are **not** related in the SCM
- Example:
  - Let's check the PBS to avoid *bad* requirements such as: “The cabin shall have wings”, or even “The Aircraft shall have flaps”,



Apply only on requirements matching a pattern group, a pattern or both at the same time:

Pattern group:

Pattern:

Select a SCM view and a maximum no. of relationship levels to check the specification relationships:

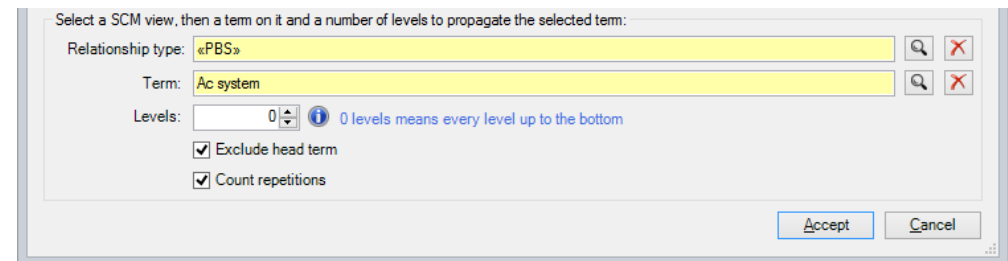
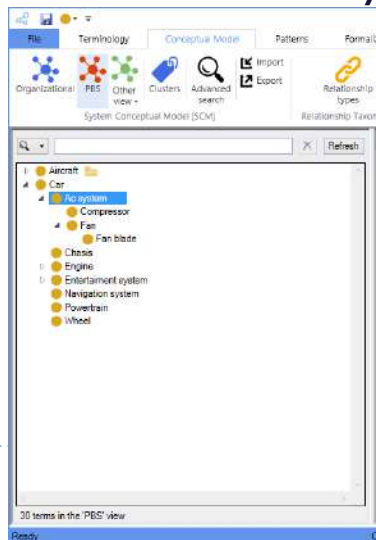
Relationship type:

Max. relationships:  0 means that any number of relationships in the selected SCM View is allowed to relate terms.  
1 means that terms that are related directly in the selected SCM View are only allowed.  
More than 1 means that as many relationships in the selected SCM View as indicated by this parameter are allowed in between terms.

☐ Count duplicated relationships as new instances

## Parameterized Super/Sub-terms in SCM

- Given a specific view of the SCM (e.g. a PBS structure)
  - Counts the number of terms under a specific point in the SCM
  - Note that the opposite (super-terms) can also be checked
- Example:
  - Let's assume a requirements specification about a specific item in the PBS
  - The metric detects (and forbids) any more detailed sub-element in the PBS structure to avoid early architectural details





```
string sInput;  
int iLength, iN;  
double dblTemp;  
bool again = true;
```

```
while (again) {  
    iN = -1;  
    again = false;  
    getline(cin, sInput);  
    system("cls");  
    sInput.erase(sInput.begin(), sInput.end());  
    iLength = sInput.length();  
    if (iLength < 4) {  
        again = true;  
        continue;  
    } else if (sInput[iLength - 1] != '.') {  
        again = true;  
        continue;  
    } while (++iN < iLength) {  
        if (isdigit(sInput[iN])) {  
            continue;  
        } else if (iN == (iLength - 3)) {  
            continue;  
        }  
    }
```

# Custom code

## Metrics



## Parameterized Custom – Code metrics

- This extensibility mechanism allows you to code your own metrics
- Use .NET Framework 4.0 to create your library
- 5 different interfaces. The most common:
  - TYPE 1. Custom function only taking into account the requirement URI
    - `float custom_Function_Type1 (string requirementUri)`
  - TYPE 2. Custom function taking into account also the requirement text
    - `float custom_Function_Type2 ( string requirementUri, string requirementText)`
  - TYPE 3. Custom function taking into account the requirement text and the out-of-the box metric evaluations
    - `float custom_Function_Type3(string requirementUri, string requirementText, RequirementEvaluation[] metricIdsAndEvaluations)`

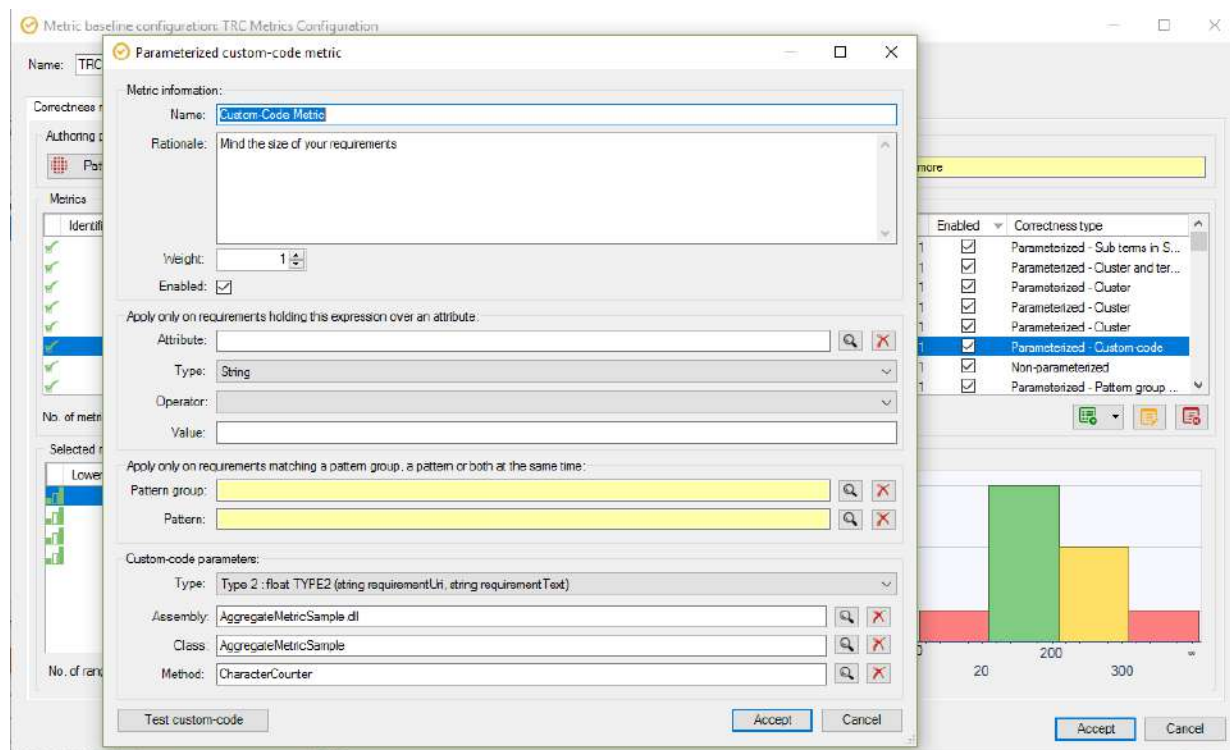


## Parameterized Custom – Code metrics

### ➤ Example of type 2:

0 references

```
public static double CharacterCounter(string requirementUri, string requirementText) {
    int counter;
    counter = requirementText.Length;
    return counter;
} // CharacterCounter
```



## Parameterized Custom – Code metrics

### ➤ Example of type 3:

0 references

```
public static double TraceabilityChecker(string requirementUri, string requirementText, IRequirementMetric[] requirementEvaluations) {  
    double result;  
    result = 0f;  
    if (requirementEvaluations != null) {  
        IRequirementMetric OutlinksDerives, InlinksVerifies;  
        OutlinksDerives = (from IRequirementMetric requirementEvaluation in requirementEvaluations  
            where requirementEvaluation.MetricId == 44 && requirementEvaluation.MetricPerTypeId == 2906  
            select requirementEvaluation).FirstOrDefault();  
        InlinksVerifies = (from IRequirementMetric requirementEvaluation in requirementEvaluations  
            where requirementEvaluation.MetricId == 44 && requirementEvaluation.MetricPerTypeId == 2876  
            select requirementEvaluation).FirstOrDefault();  
  
        if (InlinksVerifies.AbsoluteScoring == 1 && OutlinksDerives.AbsoluteScoring > 0) {  
            result = 1; // High quality  
        } else {  
            result = 0; // Low quality  
        }  
    }  
    return result;  
} // TraceabilityChecker
```

## Live demo

Requirements Quality Analyzer

File Quality Control Project configuration Quality Assurance

Module selector: STRS

Requirements: Simple view Quality view Full view

Correctness Users Charts Metrics Metrics Suggestions Knowledge base

Requirements:

	ID	Text	Correctness	Score	Mand..	Correctness qualit..	Consistency	Issues
<input type="checkbox"/>	StR1	The altitude resolution is equal to or less than	★ ★ ★	2.85	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR2	The pressure altitude from an approved s... controller	★ ★ ★	2.85	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR3	The system shall warn the air traffic con...	★ ★ ★	2.85	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR4	The pilot shall be able to light the inter...	★ ★ ★	3.57	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR5	The engine shall provide enough powe...	★ ★ ★	2.85	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR6	The air traffic controller shall be warn...	★ ★ ★	3.57	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR7	The dashboard shall warn the pilot abo...	★ ★ ★	3.57	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR10	There shall be a button in the dashboa...	★ ★ ★	4.28	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR11	The aircraft should quickly allow the rem...	★ ★ ★	2.85	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR43	The maximum speed of the aircraft shall be 90...	★ ★ ★	3.33	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR13	The maximum speed of the aircraft shall be 900 mph (1448 km/h)	★ ★ ★	2.66	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR14	The maximum speed of the aircraft shall be 900 mph (1600 km/h)	★ ★ ★	2.00	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR31	TBD	★ ★ ★	20.00	1	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR32	When the speed of the car is above 5 mph (8 Km/h) the passengers shall not be allowed to o...	★ ★ ★	2.00	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR34	The fire alarm shall be activated when the temperature is over 100 degrees	★ ★ ★	2.85	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR33	The aircraft shall be white	★ ★ ★	2.85	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR35	The aircraft shall have flaps	★ ★ ★	4.28	0	21/03/2017 12:51:...	★ ★ ★	N/A
<input type="checkbox"/>	StR36	The aircraft shall have 2 wings	★ ★ ★	4.66	0	21/03/2017 12:51:...	★ ★ ★	N/A

Total requirements: 22

Reports Assess CCC for the whole specification View quality details

RMS Repository: 36677@localhost; Project: Webinar examples RMS User: jmfuentes Connected to 'C:\Program Files (x86)\The REUSE Company\Requirements Quality Suite Server\RQS v15.1 (English).mdb'

## Questions & Answers



