prostep IVIP

Recommendation

prostep ivip Recommendation PSI 18 V2.1

ReqIF Recommendation

Comprehensive Collection of Industrial ReqIF Use Cases

Abstract

The ReqIF Requirements Interchange Format [1] enables the exchange of requirement information by transferring XML documents.

This Recommendation provides an overview of the prostep ivip project activities regarding ReqIF. After a short introduction, the interaction of the user-driven Workflow Forum and the vendor-driven Implementor Forum is explained in more detail. Within the ReqIF Workflow Forum project group, use cases for the application of ReqIF in the context of product engineering were specified. These are presented in this document. For the industrial implementation of the use cases a workflow has been defined. To support the workflow the user community agreed on a set of user-defined attributes and their values. These attributes will become part of a ReqIF data exchange.

Disclaimer

prostep ivip Recommendations (PSI Recommendations) are available for anyone to use. Anyone using these recommendations is responsible for ensuring that they are used correctly.

This PSI documentation gives due consideration to the prevailing state-of-the-art at the time of publication. Anyone using PSI documentations must assume responsibility for his or her actions and acts at their own risk. The prostep ivip Association and the parties involved in drawing up the PSI documentation assume no liability whatsoever.

We request that anyone encountering an error or the possibility of an incorrect interpretation when using the PSI documentations contact the prostep ivip Association (psi-issues@prostep.org) so that any errors can be rectified.

Copyright

Table of Content

Figures

Tables

Terms, Definitions

Terms and definitions	Meaning
Check Result	Is a consolidation of relevant Q-check results regarding import or export
Cross section requirement specification	German: Querschnittslastenheft (Q-LH) Example: EE standard
Frozen Requirement Specification	Is a defined and archived version of requirement specifications
MGU Accompanying Document	German: Mitgeltende Unterlage Examples: o laws, international or national standards (e.g. ISO26262) o company specific standards (e.g. VW99000) o cross section requirement specifications
JT Jupiter Tessellation	CAD file format
OPL Open Point List, Offene Punkte Liste	List to discuss acceptance and deviations of requirements
Preview	The preview validates the ReqIFz contents against existing data base in RM-tool and illustrates possible changes
Receiver	The receiver of a Requirements Interchange Package
ReqIF Requirements Interchange Format	The format specified in the OMG standard [1]
ReqIFz Requirements Interchange Package	Is a zipped package, that includes one or more ReqIF files and additional files, e. g. reports, embedded objects, pictures, etc.
Requirement Specification Package	Is a specific set of requirement specifications incl. attributes, links, embedded objects, etc. within a RM-tool
ReqIF Scheme-Check	Is a compliance check against the ReqIF XML scheme
ReqIF Q-Check	Is a quantitative content quality-check, that could be used for preview and validation of the ReqIFz.
Sender	The sender of a Requirements Interchange Package

Abbreviations

Abbreviations	Meaning
CustRE	Customer Requirement Engineer
HIS	Herstellerinitiative Software
OFTP/OFTP2	Odette File Transfer Protocol
Q-Check	Quality-Check
Q-LH	Cross section requirement specification; German: Querschnittslastenheft
OMG	Object Management Group
OLE	Object Linking and Embedding
ReqIF IF	ReqIF Implementor Forum
ReqIF WF	ReqIF Workflow Forum
RM	Requirements Management
RTF	Rich Text Format
SI	System Integrator
SuppRE, SuppRE_1 … n	Supplier Requirement Engineer of Supplier 1 … n
Sub-SuppRE	Supplier Requirements Engineer of a Sub-Supplier
TM	Test Manager
UC	Use Case
XHTML	Extensible Hypertext Markup Language

1 Introduction

The Requirements Interchange Format (ReqIF) was developed under the auspices of the prostep ivip Association and published as an OMG standard in 2011. A ReqIF Implementor Forum (ReqIF IF) was established within the framework of the prostep ivip Association that same year. A great deal has happened since then. The ReqIF IF is one of the Association's many Implementor Forums – which aim to provide system vendors with early support when implementing a standard and ensure that the interfaces exhibit a high level of quality. This approach has also been applied successfully to the ReqIF standard. Leading requirements management tool vendors, system integrators and service providers were involved from the very start, and users made known their technical requirements.

Over the years, a number of rounds of testing were performed to ensure basic compatibility between tools, to test basic functionality, to deal with open issues and to limit the scope for interpretation – measures to ensure that the tools are able to communicate with each other and that requirements specifications, including tables, attributes and links, can be exchanged between requirements management tools and between different software versions with ReqIF without any loss of information. This includes aspects such as

- handling different fonts, including representation in RTF and XHTML
- linking and referencing objects
- embedding images, diagrams, and other representations (OLE objects)

Agreements that go beyond the scope of the standard and which are needed to exchange data efficiently were recorded and published in the ReqIF Implementation Guidelines [2]. The guidelines contain agreements relating to, among other things, naming conventions for file names and system attributes, character encoding, DOORS tables, embedded objects, formatting rules and simplified XHTML content.

The ReqIF Implementor Forum is also ultimately responsible for the maintenance and further development of the standard according to OMG regulations. It should be noted at this point that the standard itself is very stable. The rounds of testing and initial pilots did not reveal any significant weaknesses or deficits and hardly any requests for changes to the ReqIF standard have been received. The recently published version 1.2 [1] contains only editorial changes. The XML schema, which is binding for implementations, remains unchanged.

1.1 Harmonization of ReqIF exchange processes

Restricting possible interpretations and avoiding dialects are just two prerequisites for successfully exchanging requirements. Cross-enterprise requirements management demands more than just the tools understanding each other. The processes by which customers and suppliers deal with requirements must also be examined in this context. This means that the users must be consulted and must come to an agreement. Although, as mentioned earlier, user companies have been actively involved in the ReqIF Implementer Forum in recent years, focus was not placed on harmonizing the processes.

In accordance with the V model approach used by the prostep ivip Association, a decision was made to define use cases, derive requirements relating to the processes, set priorities and draw up an implementation timetable. Implementation would then be performed together with the system vendors in the ReqIF Implementor Forum, i.e. derivation of user stories, definition of the corresponding test cases and ultimately transfer to productive use. The user stories are currently elaborated by the ReqIF WF and will be published at a later date.

Part of this work is also the definition of the requirements exchange process; in particular the status attribute controlled workflow. Starting basis was the "HIS Exchange Process for Requirements" [3]. This process was developed several years ago by the HIS group. Meanwhile, new requirements came up which would need a revision of the HIS specification. As the HIS group is no longer active, the ReqIF Workflow Forum decided to continue this activity and maintain the requirements exchange process.

1.2 Collaboration between the ReqIF Workflow Forum, the Implementor Forum and Benchmark

It was agreed from the start that even after the creation of a user forum, the dialog with vendors of requirements managements systems and system integrators needed to be maintained and that the effective collaboration between users and system vendors that already existed within the framework of the ReqIF Implementer Forum should continue.

In addition, from the vendor perspective, the activities within the ReqIF Implementor Forum can be seen as a kind of preparation for the ReqIF Benchmark. Both ReqIF Workflow Forum and ReqIF Implementor Forum drive the ReqIF Benchmark. It is a neutral activity, which is funded by contributing organization, and not by companies whose products will be tested. Aim is to provide a neutral comparison of up to date ReqIF applications. The benchmark results represent a comprehensive snapshot of ReqIF applications performance and will be published as ReqIF Benchmark [4].

Placing both forums and the benchmark under the same management (industry sponsor) mitigated the risk of the project groups drifting apart.

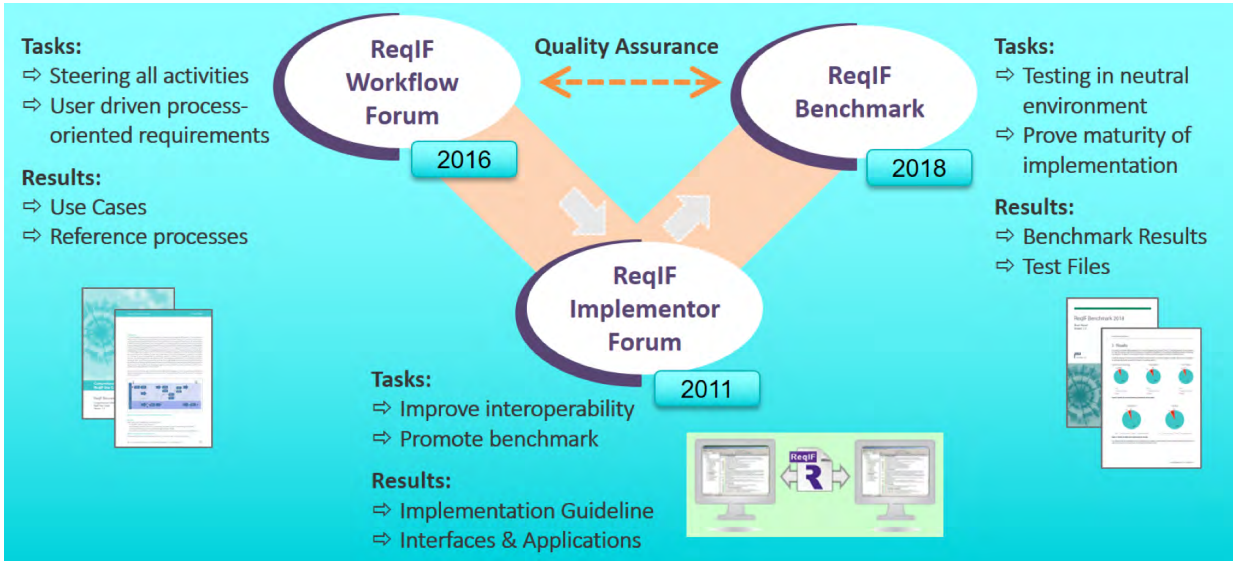


Figure 1: Cooperation between ReqIF forums and benchmark

2 Use Cases

In 2016, the ReqIF Workflow Forum focused on collecting and describing use cases. The ReqIF Workflow Forum looked into the specifications from the HIS Exchange Process, the results of the ReqIF testing activities and the insights gained from pilot projects to define use cases of common interest and for which coordinating and harmonizing the processes make sense. The method tried and tested by the prostep ivip Association and also used with success for JT was applied to describing the use cases. The use cases are specified by means of swim lane diagrams and supplemented with text descriptions. In general, the approach taken in the JT projects served as a blueprint for ReqIF activities.

The industry members in the ReqIF Workflow Forum identified 7 key use cases:

It is, of course, impossible to address all the use cases simultaneously. The definition of appropriate test activities in the Implementer Forum must also be performed gradually, which is why priorities had to be set. The first 5 use cases were highly prioritized and subsequently further detailed. The results have been documented in this publication.

The first two use cases for ReqIF export and import are basic use cases which are included in most of the other use cases. The description of these basic use cases also takes account of the other information needed in addition to the mere exchange of requirements using the ReqIF format. With their test activities and the Implementation Guidelines, the system vendors in the IF have in recent years ensured that the exchange works "per se". This means that the results of the Workflow Forum will not only affect the ReqIF exchange tools but, in particular, the users' requirements management tools and process specifications (e.g. agreeing on common, uniform attributes). From the users' point of view, quality assurance measures such as performing quality checks when ReqIF files are exported or imported are also important. In addition to the actual requirements specifications, it is also necessary that the results of a quality check, and any other relevant information, be exchanged between the process partners.

The use cases are presented in detail in the following chapters. First the structure is explained.

2.1 Structure of the use case descriptions

Each use case has a uniform structure. The use case description is divided into three sub-chapters: "Use Case", "Description" and "Benefit". In the sub-chapter "Use Case", the aim of the use case is described in form of a short sentence. After that the actors are listed. These are the main human and machine entities and their roles. Next the preconditions are explained. This is a description of anything being assumed to have happened before the activities described in the use case description. The sub-chapter "Description" contains a narrative description of the use case or usage scenario, followed by the postconditions where the final state is described. Afterwards a diagram in form of a simple illustration of the use case as sequence or activity diagram is shown. Annex A explains the notation used in the diagrams. Finally, the benefits are described in the sub-chapter "Benefits" in form of a documentation of main benefits concerning the application of this use case for the actors. Any important points that still have to be decided or any open issues, will be discussed at the end of this sub-chapter.

2.2 ReqIF for Requirements Export

Use Case

This use case describes the export process with a ReqIF data exchange package. It serves as a basic use case which is included in every other use case having the need to export a ReqIF data exchange package. The aim is to create gapless and coordinated customer/supplier requirement specifications, serving as a technical and valid basis on customer and supplier side. The actors in this use case are the sender and the receiver of a ReqIF data exchange package, which can be located at the customer or supplier site.

The precondition is that a requirement specification package shall be created and reviewed by the relevant department. Optionally, it might be signed and released. Sender and receiver should clarify and agree on exchange scenarios. Both parties have to define relevant exchange documents. Besides that, relevant exchange contents shall be appointed (e.g. project, variants, attributes, attribute types, deletion handling), exchange rules and instructions (technical relevant) shall be agreed and both parties shall agree on a data exchange process.

Description

After creating a set of requirements (initial, reviewed, updated, etc.) within a Requirements Management system, the sender composes a requirement specification package (1). This consists of a specific set of requirement specifications including attributes, links, embedded objects, etc. Thereupon the sender creates a ReqIFz package that includes one or more ReqIF files and additional files (2). Then, a quantitative content Q-check is performed by analyzing the export results and validating the ReqIFz package (3). Finally, the ReqIFz package and additional check results (optional) are sent to the receiver (4). The receiver imports the package and follows the steps within the linked use case “ReqIF for Requirements Import”.

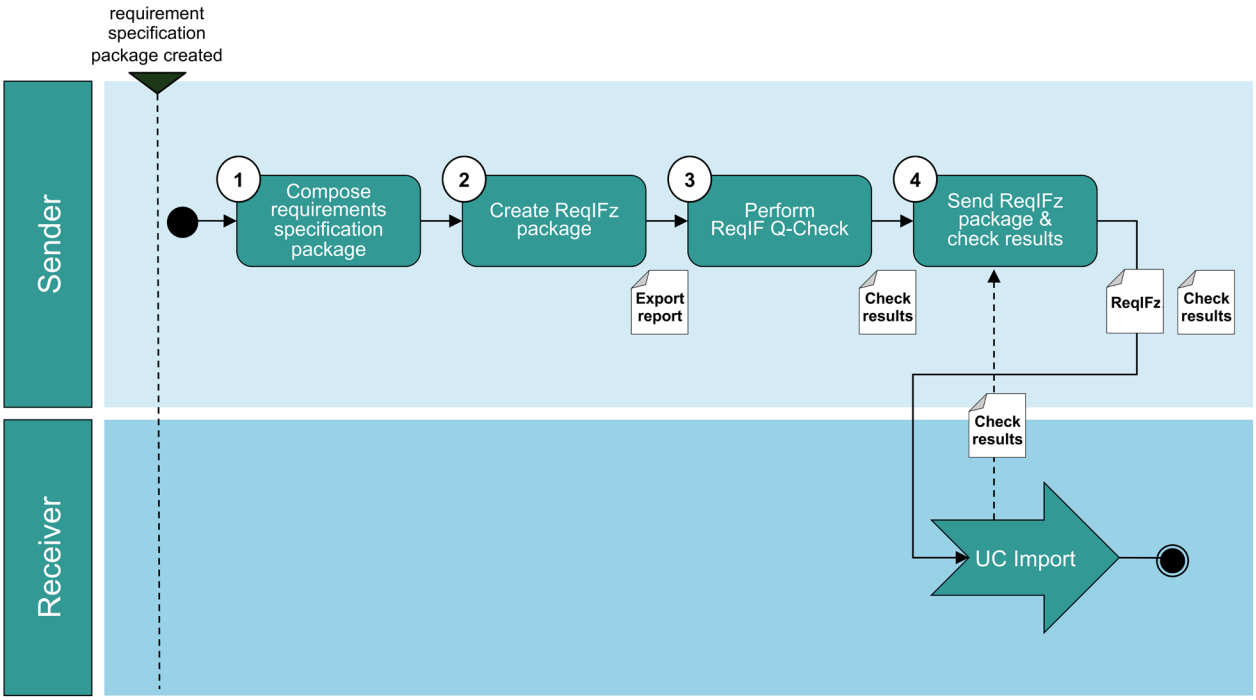


Figure 2: Use case diagram "ReqIF for Requirements Export"

Benefit

Quality control of generated exchange requirements package (see Table 5: Q-Check criteria):

- ReqIF Q-check: validation of ReqIFz package regarding requirement specification inside RM tool. Therefore, the RM tool creates an export report with compilation errors and statistical information about the ReqIF files.

Process improvements

- robustness
- lean and efficient information handling
- consistency of data
- transparency of exchange process

Notes and deduced requirements

Implementation requirements and acceptance criteria are documented in the derived user stories.

2.3 ReqIF for Requirements Import

Use Case

This use case describes the import process with a ReqIF data exchange package. It serves as a basic use case, which is included in every other use case having the need to import a ReqIF data exchange package. The aim is to create gapless and coordinated customer/supplier requirement specifications, serving as a technical and valid basis for the customer and supplier side. The actors in this use case are the sender and the receiver of a ReqIF data exchange package, which can be located at the customer or supplier site.

The precondition is that the relevant department has created a requirement specification package. Sender and receiver should clarify and agree on exchange scenarios. Both parties have to define relevant exchange documents. Besides that, relevant exchange contents shall be appointed (e.g. project, variants, attributes, attributes types, deletion handling), exchange rules and instructions (technical relevant) shall be agreed and both parties shall agree on a data exchange process.

Description

After receiving a ReqIFz package, which includes one or more ReqIF files and additional files, e.g. check results, a compliance check against the ReqIF XML scheme is executed and a preview is generated (1). Within the preview, the receiver validates the ReqIFz contents against the existing data base. Therefore, possible changes can be illustrated. If the validation is not correct, the use case ends. If the validation is correct, the receiver imports the ReqIFz package within his RM system (2). Thereupon he executes a quantitative content Q- check (3) and handovers relevant Q-check results to the sender (4).

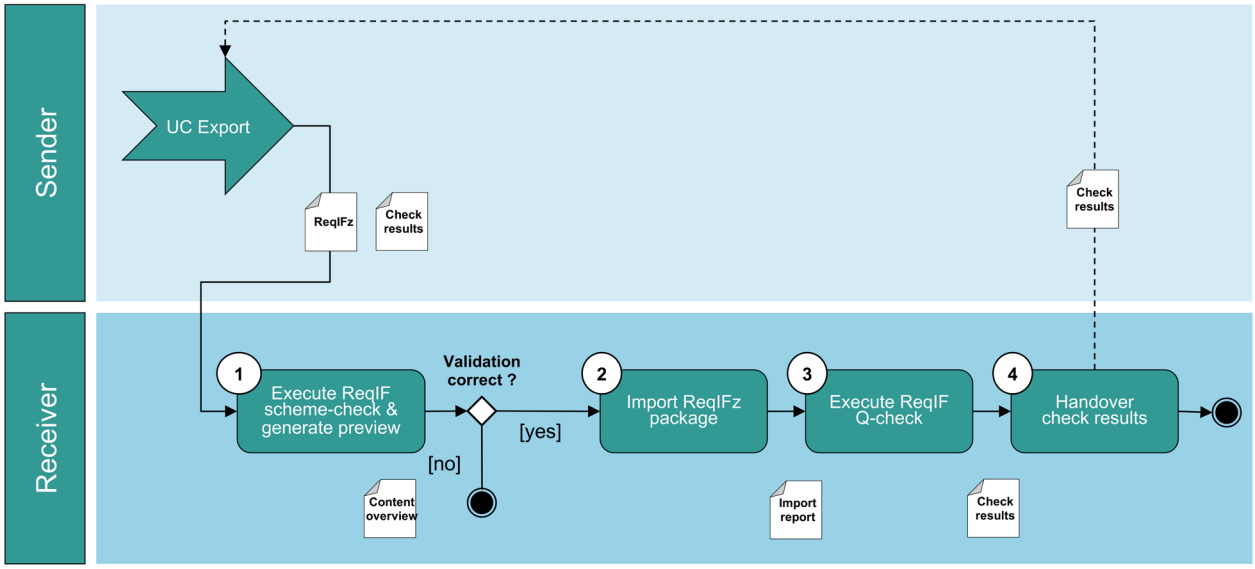


Figure 3: Use case diagram "ReqIF for Requirements Import"

Benefit

Quality control of generated exchange requirements package (see Table 5: Q-Check criteria)

- ReqIF Scheme-check: validation of ReqIFz package regarding ReqIF scheme conformity
- Content preview: statistical information about the content of the ReqIF files and preview of type and attribute definition
- ReqIF Q-check: validation of received ReqIFz package regarding expected requirement specification content. Therefore, the RM tool creates an import report with compilation errors and changes on type and attribute definitions as well as changes in the requirement specification content.

Process improvements

- robustness
- lean and efficient information handling
- consistency of data
- transparency of exchange process

Notes and deduced requirements

Implementation requirements and acceptance criteria are documented in the derived user stories.

2.4 Introduction to Stakeholder Request Clarification (SRC) Use Cases

The aim of the Stakeholder Request Clarification (SRC) use case is the usage of ReqIF for exchanging technical requirement specifications between customers and suppliers from the tender stage. The aim is to create gapless and coordinated customer/supplier requirement specifications, serving as a technical and valid basis for the customer and supplier side. During stakeholder request clarification, requirement specifications from the customer are gradually clarified and voted with potential suppliers.

There exist different scenarios for applying the SRC process – see Figure 4:

- One customer exchanges (different) specifications with one supplier. E.g. in Figure 4, customer C1 exchanges specification Spec3 with supplier S3. In the following, this scenario will be called standard SRC.
- One customer exchanges the same specification with multiple suppliers. E.g. in Figure 4, customer C1 exchanges specification Spec1 with suppliers S1 and S2. This scenario will be called multi-supplier SRC.
- A supplier has sub-suppliers, i.e. changes role within exchange process. E.g. in Figure 4, supplier S1 receives Spec2 from customer C1 and exchanges parts of Spec2 called Spec2' with sub-supplier Sub-S1. This scenario will be called multi-tier SRC.

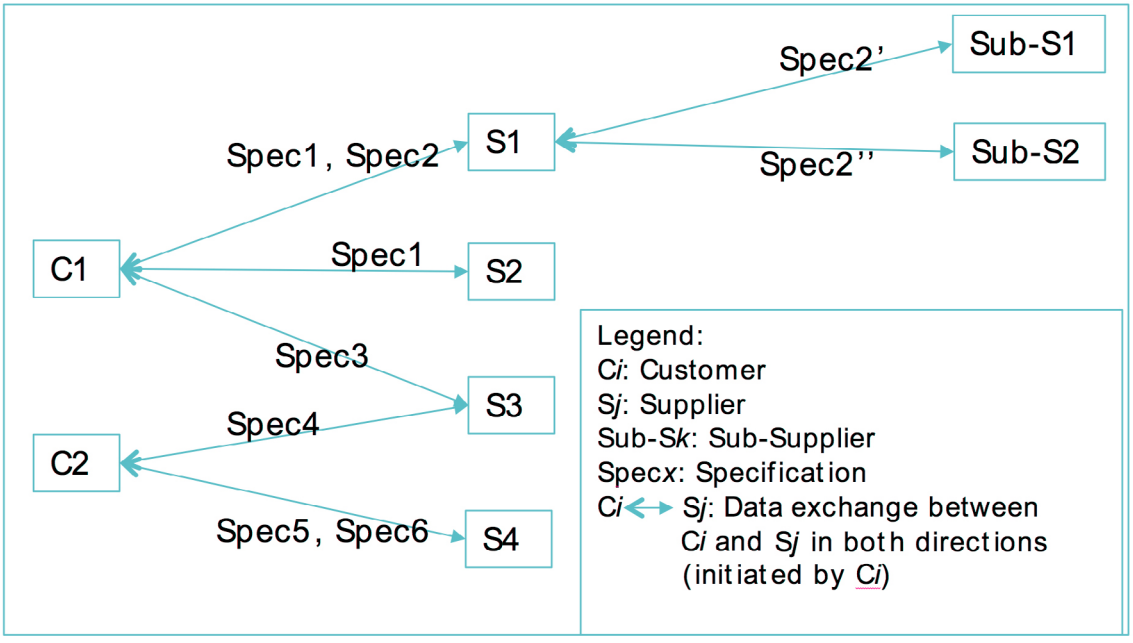


Figure 4: ReqIF Exchange Scenarios

SRC attributes

An essential part of the SRC process is a voting mechanism between customers and suppliers. It is done via status and comment attributes on both sides. In this recommendation, the following "standard" names are proposed for these attributes:

- ReqIF-WF.CustomerStatus
- ReqIF-WF.SupplierStatus
- ReqIF-WF.CustomerComment
- ReqIF-WF.SupplierComment

In the following, these attributes will be called SRC attributes. Details on the attributes and how to use them in the SRC exchange process are described in chapters 3 and 4 and will not be detailed out here.

Setup of clarification process

This recommendation focuses on ReqIF processes from an end-user point of view independent of a specific Requirements Management (RM) tool and its ReqIF connector. In practice, these different RM tools and their ReqIF connectors have different capabilities, e.g. for storing the SRC attributes in the RM database. And these differences matter when exchanging data via ReqIF. Tool vendors need to make sure that the ReqIF processes described here work with any tool combination. In this document, we will deduce some generic requirements on those ReqIF tools. Tool specific usage guidelines need to be documented in a future document.

For a specific RM/ReqIF tool, the following questions need to be answered before using ReqIF for SRC use cases:

- Where will the SRC attributes be stored in the RM tool?
  - o E.g. in the requirements themselves, in objects attached to the requirements, or in attributes related to branches of the requirement structures?
- How will the SRC attributes be named in the RM tool?
  - o E.g. with standard names or with extended standard names that contain customer/supplier specific prefixes or suffixes?
- How will the SRC attributes be named in the ReqIF file?
  - o I.e. will they be named the same as in the customer RM tool or will a mapping take place?
- For the importing ReqIF/RM tool, is it possible to use the same names in the RM tool as in the ReqIF file?
  - o Otherwise, a mapping needs to happen in the supplier ReqIF tool.

The standard SRC, the multi-supplier SRC and the multi-tier SRC will be presented in a simplified way. In the presentation, we assume that each step will be completed before the next step starts, e.g. suppliers vote on customer requirements and send back their feedback before customers perform any changes to the requirements. In practice, some of these steps will be executed in parallel. This will be detailed out in chapter "2.4.5 Parallel working on specifications".

Often, customers want to exchange different variants of a requirement specification with their suppliers. This will be handled in chapter "2.4.4 Variant management".

2.4.1 Standard SRC process

Use Case

The aim of this use case is to describe how to exchange technical requirement specifications between single customers and suppliers using ReqIF. During stakeholder request clarification requirement specifications from the customer are gradually clarified and voted with potential suppliers. There is a narrow and documented voting until the suppliers have confirmed the fulfillment of all requirements. The actors in this use case are the Customer Requirement Engineer (CustRE) and the Supplier Requirement Engineer (SuppRE).

The precondition is that a clear customer requirement specification shall be created, reviewed, signed and released by the relevant customer department. Both parties have to define relevant exchange documents, exchange cycles and determine a desired voting degree. Besides that, relevant exchange contents shall be appointed (e.g. project, variants, attributes, attribute types, deletion handling), exchange rules and instructions (technical relevant) shall be agreed and both parties shall agree on a data exchange process.

Description

The CustRE creates an initial set of requirements within a Requirements Management (RM) system (1). For a defined and archived version of this requirement specification it shall be versioned (2). Based thereon, he exports the customer requirement specification package (UC ReqIF Export). The SuppRE imports this package (UC ReqIF Import). If the requirement specification package is relevant for a voting, the SuppRE reviews and assesses the requirement specification contents and executes a voting (3). If the package isn't relevant for voting, the use case ends. After supplier evaluation, the SuppRE exports his package to the CustRE (UC ReqIF Export). The CustRE imports this package (UC ReqIF Import). After receiving the supplier's data, the CustRE reviews and assesses the responses of the suppliers (4). If necessary, he determines and carries out an actual status report to get an overview of the entire voting state (5). If necessary, he clarifies his stakeholder requests according to all suppliers' feedback. This may result in the rejection of the supplier feedback, in a request for clarification or in a change of requirement specifications (modification, addition or deletion) (6). The CustRE continues, according to the status, with activity #2. If the requirements specification voting is fulfilled, he defines a handling of open issues (7) and proceeds with activity #8. The CustRE freezes the requirement specification for a defined and archived version (8a) and sends a release notification as well as the synchronized requirements (8b) to the SuppRE via Export use case. The SuppRE imports this package again and creates a frozen requirement specification too (9). Finally, a purchasing and sales agreement is to be concluded.

The postcondition of this use case is that the requirement specification package is "frozen", a related baseline as well as a released engineering status is created and stored at both sides. Thereupon the supplier award will take place in cooperation with the procurement.

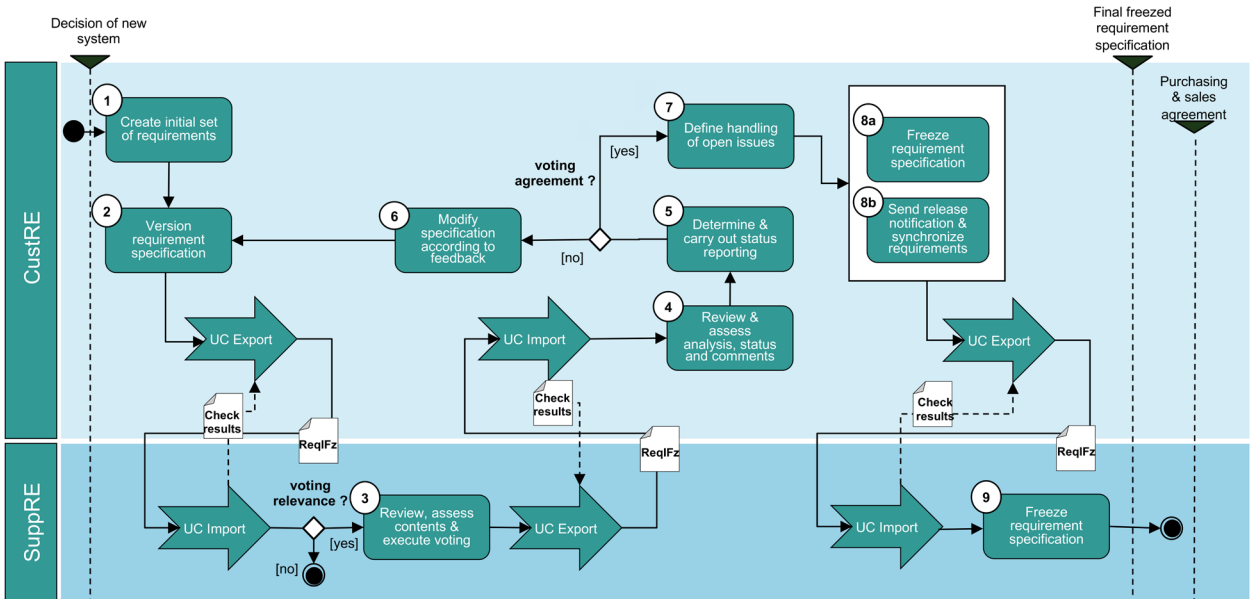


Figure 5: Use case diagram "Standard SRC process"

Benefit

ReqIF can be used for stakeholder request clarification

- Several specifications consisting of multiple atomic requirements can be exchanged
- Interoperability between requirement management tools based on neutral data exchange format ReqIF
- Use and maintenance of one neutral data exchange interface for ReqIF
- Consistent and automated requirement exchange process between engineering partners (Customer/Supplier)

Notes and deduced requirements

ReqIF can be used for stakeholder request clarification

- Depending on capabilities of RM/ReqIF tool, SRC attributes will use standard names or will have additional supplier specific prefixes/suffixes in the exchanged ReqIF file. ReqIF tools have to cope with both situations. In particular, naming conflicts (resulting from data exchanges with different customers) need to be resolved.
- ReqIF tools need to support filtering of attributes when exporting/importing data since some attributes shall not be exposed to the other side during export or updated during import.
- All types/attributes used in the supplier ReqIF file need to be consistent (i.e. same names, IDs, definitions) with the customer ReqIF file, because some ReqIF tools offer manual mapping capabilities that depend on the names used in the ReqIF file.
- The ReqIF tools should support reusable export/import configurations for simplifying export or import of the same (or similar) specification structures multiple times.

2.4.2 Multi-Supplier SRC process

Use Case

The aim of the Multi-Supplier SRC is to describe how a customer can clarify specifications/requirements with multiple suppliers, e.g. during initial quotation phase. For each individual supplier, standard SRC process will be used. The actors in this use case are the Customer Requirement Engineer (CustRE) and the Supplier Requirement Engineers (SuppRE_1, …, SuppRE_n) of the different suppliers.

Preconditions are the same as for standard SRC use case, i.e., customer has agreed with all suppliers on relevant exchange documents, exchange cycles and determined a desired voting degree. Besides that, relevant exchange contents shall be appointed (e.g. project, variants, attributes, attribute types, deletion handling), exchange rules and instructions (technical relevant) shall be agreed and all parties shall agree on a data exchange process.

Description

For each supplier, CustRE creates a separate ReqIF package with a distinct name. The packages will be based on the same versioned requirement specifications and differ essentially only in the values (and possibly the names) of the SRC attributes. CustRE provides supplier-specific ReqIF package to each SuppRE_i (1). Each SuppRE_i imports supplier-specific ReqIF package into supplier RM tool, assesses the requirements provided by the CustRE and adds status and comments according to standard SRC use case. Then he exports updated data containing supplier feedback to ReqIF package and provides this back to CustRE (2). The CustRE imports each supplier-specific ReqIF package into customer RM tool. This will update the supplier attributes. CustRE updates requirements, status, comments according to feedback from SuppRE_i. Status values and comments are managed individually for each SuppRE_i (3). As long as no sufficient agreement has been achieved (at least with one of the suppliers), next roundtrip starts at step 1 again (4). The workflow is completed as in the standard SRC use case, e.g. by freezing the finally agreed requirements specification, performing a final data exchange with (one of) the suppliers and concluding a purchasing and sales agreement (5).

The postcondition of this use case is that the Customer has achieved sufficient agreement with at least one supplier. The same requirements structure will be frozen and baselined on both sides.

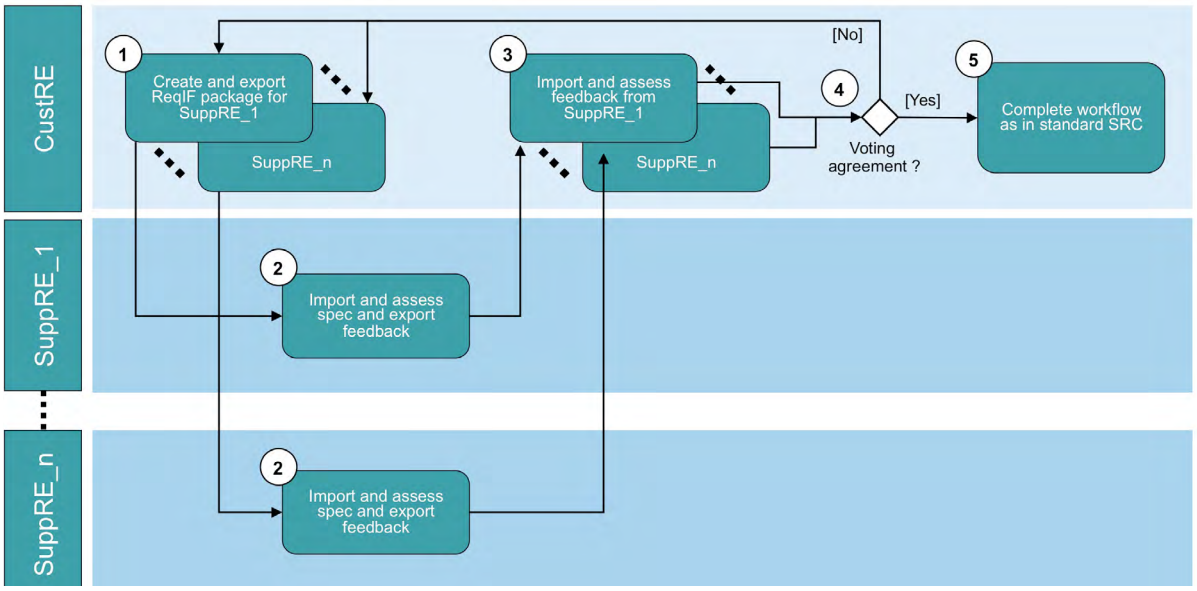


Figure 6: Use case diagram "Multi-Supplier SRC process"

Benefit

- Customer can align requirements with multiple suppliers at the same time during quotation/clarification phase.
- Customer can compare feedback received from multiple suppliers.
- On supplier side, this use case does not differ from normal SRC use case

Notes and deduced requirements

- The exchanges between the customer and the different suppliers do not need to happen at the same time but usually the customer will have an interest in performing data exchanges with the different suppliers at similar points in time. This will reduce the number of different versions that the customer has to manage for the different suppliers.
- It must be possible to apply different export/import configurations to the same requirements structure.

2.4.3 Multi-Tier SRC process

Use Case

The aim of the Multi-Tier SRC is to describe how a supplier can clarify customer specifications/requirements with sub-suppliers. Due to the fact that the supplier does not want to perform an evaluation in the customer's SRC process without consulting the sub-suppliers, the supplier passes on the customer specifications to one or more sub-suppliers by reusing the use cases mentioned in the introduction to SRC, e.g. during the initial quotation phase. The actors in this use case are the Customer Requirement Engineer (CustRE), the Supplier Requirement Engineer (SuppRE) and the Sub-Supplier Requirements Engineer (Sub-SuppRE) of the SuppRE.

Preconditions are that the SRC process from the CustRE with the SuppRE has already been started, the preconditions of the standard SRC use case are applicable and the status and comment attributes of the CustRE are not passed to the Sub-SuppRE.

Description

SuppRE starts the SRC process with Sub-SuppRE on second tier and creates a separate ReqIF package with a different name. The package must contain relevant requirements of the customer specifications and, if necessary, additional specifications from the supplier relevant for evaluation (1). After provisioning of the ReqIF package by the SuppRE, the Sub-SuppRE continues according to the SRC use case and provides feedback to the SuppRE (2). SuppRE imports the ReqIF package of the Sub-SuppRE into the supplier's RM tool (3). The SuppRE reacts to feedback from Sub-SuppRE according to the SRC use case. This may result in the rejection of the sub-supplier feedback, in a request for clarification or in a change of the requirements (modification, addition or deletion) of the additional specifications from the supplier, but not in a change of the customer requirement specifications. The SuppRE may share these updates with the Sub-SuppRE in another roundtrip of the SRC process on second tier (4). Due to the interaction of both SRC processes the SuppRE proceeds with the SRC process of the customer (5): He consolidates feedback from all involved parties, executes the overall voting and exports the harmonized specification to the customer. If necessary, CustRE and SuppRE start an additional roundtrip. The above steps may be repeated until a final vote has been given (6). At the latest after the last roundtrip all involved roles define the handling of the open points of both SRC processes (7). After the SuppRE has imported the frozen and synchronized customer requirements, the SuppRE will update the SRC attributes and freeze the customer requirement specification and, if available, the additional specifications from the SuppRE (8).

The postconditions of this use case are described in the Standard SRC process (chapter 2.4.1).

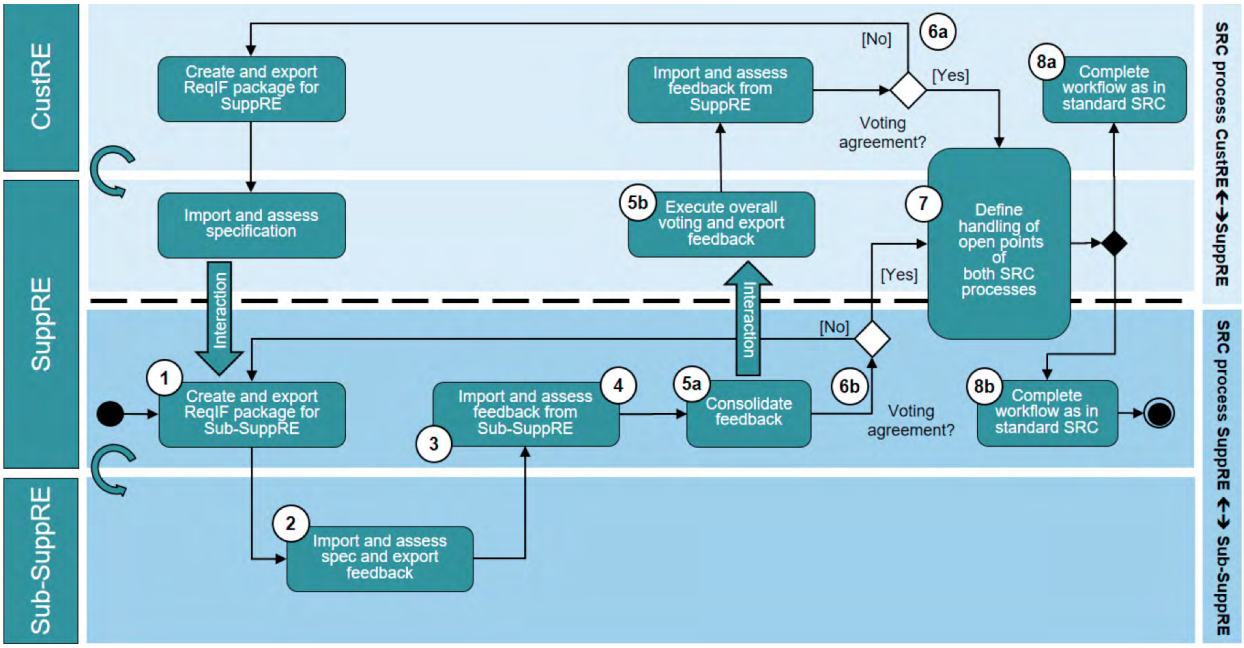


Figure 7: Use case diagram "Multi-Tier SRC process"

Benefit

- In case the customer specification is re-used the SuppRE can easily align the customer requirements with the sub-supplier.
- On sub-supplier side, this use case does not differ from the normal SRC use case.
- All votes, such as agreements, feedbacks that are related to the requirements are documented in the supplier's RM tool.

Notes and deduced requirements

The supplier's ReqIF tool must allow the data exchange of the same requirements with two different parties – the customer and the sub-supplier – in two different roles – in supplier role and in customer role. For each SRC process, separate SRC attributes have to be managed by the supplier's ReqIF tool.

2.4.4 Variant Management

Use Case

The aim of the Variant Management is to describe how information regarding different variants of the product can be exchanged alongside its requirements in a uniform way. Product means the object that is delivered by the supplier to the customer (e.g. rear axle steering). Product variant means an alternative of the product from the customer point of view (e.g. high performance rear axle steering; low performance rear axle steering). Context variant means an alternative context in which a product/product variant is used (e.g. high performance rear axle steering is used in sports car model series and in city car model series). The actors in this use case are the Customer Requirement engineer (CustRE) and the Supplier Requirement engineer (SuppRE).

Precondition is that the customer provides the necessary information about the variants incl. the assignment of requirements to variants to the supplier. This information is included in the requirements export (see section 2.2). For this the customer may use a variant management tool or define a variant attribute for the exchange. The customer may have one attribute for each supplier or one attribute independently from the supplier. Therefore the customer may also use an internal attribute for the product variant. The attribute should be multi-valued enumeration or string. All requirements are assigned to at least one product variant. The standard SRC process is used.

Description

The CustRE filters out product variants for a specific supplier by selecting the respective values of the variant attribute. Then he derives a requirements specification based on the selection of variants for the supplier exchange.

The CustRE is responsible for the coordination of variants. Attributes and workflow are described in the standard SRC process. During SRC all requirements need to be send to the supplier. Exception: A variant is not relevant for a supplier anymore. Regardless of the amount of product variants, only one pair of SRC attributes (as defined in the SRC process) is available for each requirement. All requirements are clarified including the related product variants.

The supplier receives one requirements specification, which pertains to one or more product variants. The SuppRE identifies different product variants in the requirements specification in non-ambiguous way. The SuppRE is able to filter requirements for different variants of the product. Definition of the product variants (name, characteristics, features …) are different from customer and supplier points of view. Therefore, the supplier maps the definition of the product variants from the customer to his definition. If requirements are accepted for one product variant and not for another, the following aspects have to be considered:

- Because reconciliation is done using one reconciliation attribute per requirement, this attribute value is valid for all variants. Therefore, the value "accepted" cannot be set if only some variants are reconciled.
- Solution if there is no common reconciliation across all variants:
  the value "accepted with comment" is set and the variants with need for clarification are documented in the comment attribute.

As the value “accepted with comment” is not recommended as a final status, the CustRE should resolve all requirements, which are evaluated with this value by the SuppRE. For this, the CustRE may change the requirement or create variant-specific requirements.

For all changed or new requirements, the CustRE has to set or update the assignment to the specific variants. A change of the variant assignment has to be handled in the same way as a changed requirement.

The postconditions of this use case are described in the Standard SRC process (chapter 2.4.1).

Benefit

- Customer can identify the variants of a product to which a requirements alignment is valid.
- Supplier can identify the variants of a product across different customers, from whom they have received requirements specifications.
- Requirements that are valid for all product variants need to be clarified only once (benefit for customer and supplier).
- No extra SRC processes are required for each product variant. Instead, the CustRE bundles all variants that pertain to one supplier to one requirements specification for the requirements exchange.

Notes and deduced requirements

If a single product variant is used in different contexts (see term definitions at the beginning of this use case) then two scenarios are possible:

- One SRC process for the single product variant.
- One SRC process for each context variant.
  Remark: there are no context-specific requirements. Therefore, as many SRCs need to be performed as there are context variants.

Assignment of variants to requirements:

- The customer may have one attribute for each supplier to assign variants to requirements. The relevant information of variants have to be included in the attribute for the supplier (supplier-specific variant attribute). The supplier-specific variant attribute has to be included in the export configuration. This option is recommended if enumeration values cannot be chosen for the export configuration.
- Otherwise, if the customer only wants to provide the relevant information of variants to the supplier, the customer has to choose the corresponding enumeration values of the variant attribute for the export configuration.

2.4.5 Parallel working on specifications

Use Case

The aim of this use case is to describe how to control an asynchronous clarification process so that the supplier's evaluation of a requirement matches to the correct revision of the customer's requirement. The use case "Standard SRC process" describes a clarification process on a released specification. In this sequential SRC process, the customer changes a specification only after the import of the Supplier's evaluations. Under this condition, the customer receives always the evaluations to his last exported requirements. In case of simultaneous engineering or agile product development processes, the customer changes the specification during the clarification process. In this situation, the specification on customer and supplier side are not in synchronization.

The precondition is that in the customer's RM-system, a change of a requirement is identifiable by a unique revision information and that this information is exported.

Description

The CustRE releases a set of requirements, creates an initial version (a) of the specification and exports this version to the SuppRE (1). The SuppRE imports the initial version in his RM-system and starts the evaluation. Then he exports the Supplier evaluation attributes and the Customer revision attribute as result back to the CustRE (2). In between, the CustRE adds new, modifies or discards existing requirements as well as restructures the specification in his RM-system. The Customer status is set to [ToEvaluate], if a requirement was modified or added, or to [ProposedToDiscard], if an exchanged requirement should be discarded. The internal revision information is changed, if a requirement was modified. The CustRE receives the preliminary evaluation of the specification (a) and imports the Supplier evaluation attributes and the revision attribute in the new version (b) of his specification (3). The CustRE ignores the Supplier evaluation, if the internal revision of the requirement is different to the imported revision information. After the review of Supplier evaluations, the CustRE exports the changed specification (b) including feedback on Supplier's comments for version (a) (4). The SuppRE imports the changed specification (b) in his RM-system. The import process does not overwrite the content of the Supplier evaluation attributes. In case of added, modified or discarded requirements, the SuppRE identifies the changes and

resets the Supplier evaluation attributes to their initial values (5). If parallel working is ongoing, then the process continues with step (2). If the final specification was imported then the SuppRE finalizes his evaluation and exports the results to the CustRE (6).The CustRE receives and imports the final evaluation of the specification (b) (7).

The descriptions shows only one exchange scenario. However, it is also possible to repeat an export step on CustRE and SuppRE side to send updates. If a revision information of a requirement is not available then the CustRE must tag a changed requirement so that imported evaluations of previous revisions can be discarded.

The postcondition of this use case is that the supplier evaluation is assigned to the correct revision of the requirement. Also a final exchange of synchronized requirements is performed, as described in the Standard SRC process.

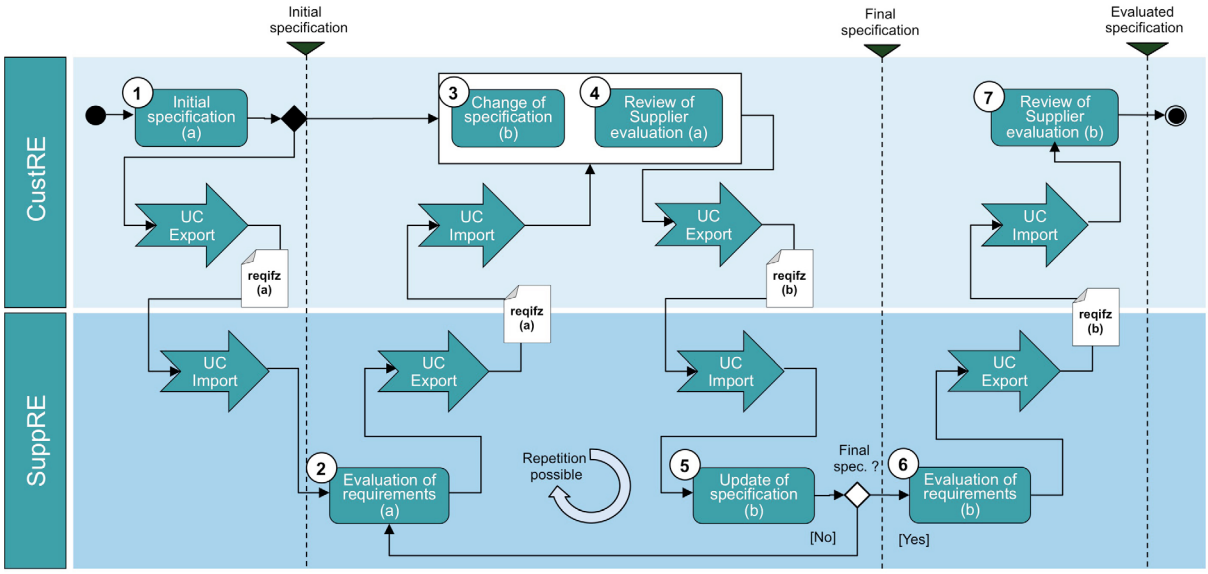


Figure 8: Use case diagram "Parallel working on Specifications"

Benefit

If a requirement is exchanged with a specific revision information then a simultaneous engineering on specification is possible on both sides and the exchange process can be performed in an asynchronous way.

Notes and deduced requirements

For exchange of revision information, a user-defined attribute is needed. The usage of this attribute is described in chapter 3.3.4.

2.5 ReqIF for Testing & Validation

Use Case

During the development process, the specified requirement has to be tested and validated. The aim of this use case is the usage of ReqIF in order to establish bidirectional traceability between requirement specification and test and validation results. The actors in this use case are the Requirements engineer (RE), Test manager (TM), and System integrator (SI).

The precondition is that a decision for the development of the system/component has been made. The RE has the task to specify for his requirements/concept a test specification for deploying his test. For requirement management, test and validation different tools are used.

Description

The RE creates his requirement in the RM tool (1). Then the TM creates the test criteria needed for the creation of the test specification and execution of the test in the RM tool (2). Finally, he exports the requirement specification package (see chapter 2.2). In the test tool, the ReqIF package is imported by the TM (see chapter 2.3). The TM specifies and executes the tests (3) and creates test results (4). The test results are again exported as a ReqIF package (see chapter 2.2) by the TM. In the validation tool, the ReqIF package is imported by the SI (see chapter 2.3). Finally, the SI evaluates the test results (5) and generates a test status (6). The test status is exported (see chapter 2.2) and imported in the RM tool (see chapter 2.3). Steps #5 and #6 are optional. If they are omitted, the results are directly returned to the RM tool. In the RM tool, the test status is connected with the requirements (7).

An alternative to this process could be, that the test deployment is executed by several external service providers.

The postcondition for this use case is that the final test report and status is linked to the original requirement for final documentation. This result can be the trigger for a final approval in the RM tool or development process.

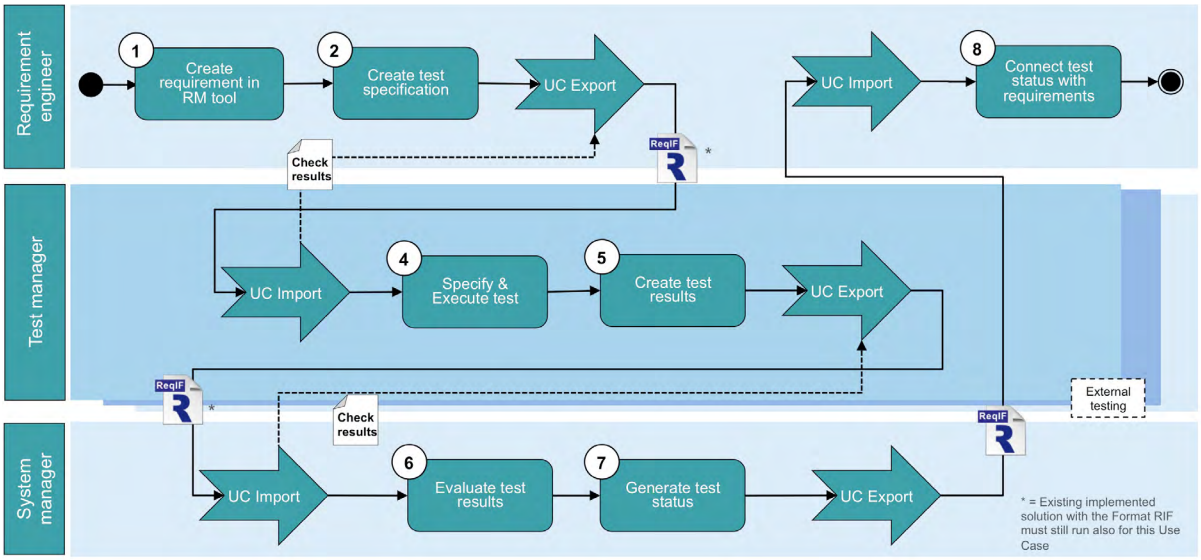


Figure 9: Use case diagram "ReqIF for Testing & Validation"

Benefit

ReqIF can be used for the interoperability with testing and validation

- As the SI is involved in the requirement engineering validation process he can report consolidated results in his steering boards presentations
- Final approval of the specified requirement in the development process (closing-loop in the V-model)
- Consistent and automated requirement exchange process between internal engineering partners (RE/TM/SI)

Notes and deduced requirements

Implementation requirements and acceptance criteria will be documented in the derived user stories.

2.6 Accompanying documents[1] exchange between customer and supplier

Use Case

This use case describes the exchange process of accompanying documents (MGU) between a customer and a supplier. Accompanying documents contain requirements for product development and must be fulfilled by the supplier. The accompanying documents are agreed upon simultaneously to the requirements specification by the customer and supplier. The supplier has to agree to the whole document but deviations to the requirements of the MGU can be documented and agreed upon within separate documents (OPL). The customer's aim is to collect the supplier's complete feedback information on the accompanying documents. The actors in this use case are the Customer requirement engineer (CustRE) and the Supplier requirement engineer (SuppRE).

The precondition is that accompanying documents are stored in MGU platforms. The MGU platform can be realized as a shared platform or as an individual platform. Access to and authorization for the MGU platform and the RM tool is necessary. The exchange of the accompanying documents is not part of this use case. A predefined specific process of how to exchange accompanying documents has to be defined, e.g. via a shared exchange platform or an RM tool (see chapter 2.4).

Description

The MGU relevant for the project are bundled in advance and provided on the MGU platform (0). The CustRE creates the requirement specification with an RM tool. The requirement which refers to an accompanying document in the MGU platform must have a textual reference to the MGU-list. An MGU-list is an additional separate document which belongs to the requirements specification (1).

Afterwards, each Q-LH from the MGU list is linked with the associated Q-LH stored in the MGU platform (2). The MGU list including the links is then exported as a ReqIF package.

The SuppRE imports the ReqIF package (MGU-List) and reviews the relevant MGU per requirement. The links to the contents of the MGU platform from the ReqIF file will be restored during this step (3). The SuppRE creates an OPL, enters open issues into the OPL and links these objects with the corresponding MGU requirements. Within this step, every issue is assigned with a status attribute (cf. chapter 3.3) according to the Stakeholder Request Clarification use case (cf. chapter 2.4) (4). Afterwards, he exports the OPL with links to the MGU platform as a ReqIF file.

The CustRE imports this file and checks if the OPL does contain open issues and whether they can be clarified or agreed upon. The links in the OPL are restored to the MGU platform (5). If there are no open issues or every open point has ReqIF-WF.SupplierStatus [Agreed], the process ends. If there are issues that need to be discussed, these issues will be clarified within a separate meeting between the author of the MGU and a supplier expert (6). The results of this meeting will be a part of future change requests for the requirement specification (1) or result in a change of ReqIF-WF.SupplierStatus within the OPL. In case of a change of ReqIF-WF.SupplierStatus, the OPL is exported and submitted to the supplier again. The process continues with step 4.

Alternatively, the MGU platform can be realized as a shared platform (e.g. Cloud) or as a local MGU copy on the supplier side (e.g. on internal server/portal room or in the RM tool itself).

The postcondition for this use case is that all MGU are agreed upon and deviations are clarified and documented on both the customer and supplier side.

[1] The laws, international, national and company standards are not part of the discussion between customer and supplier.

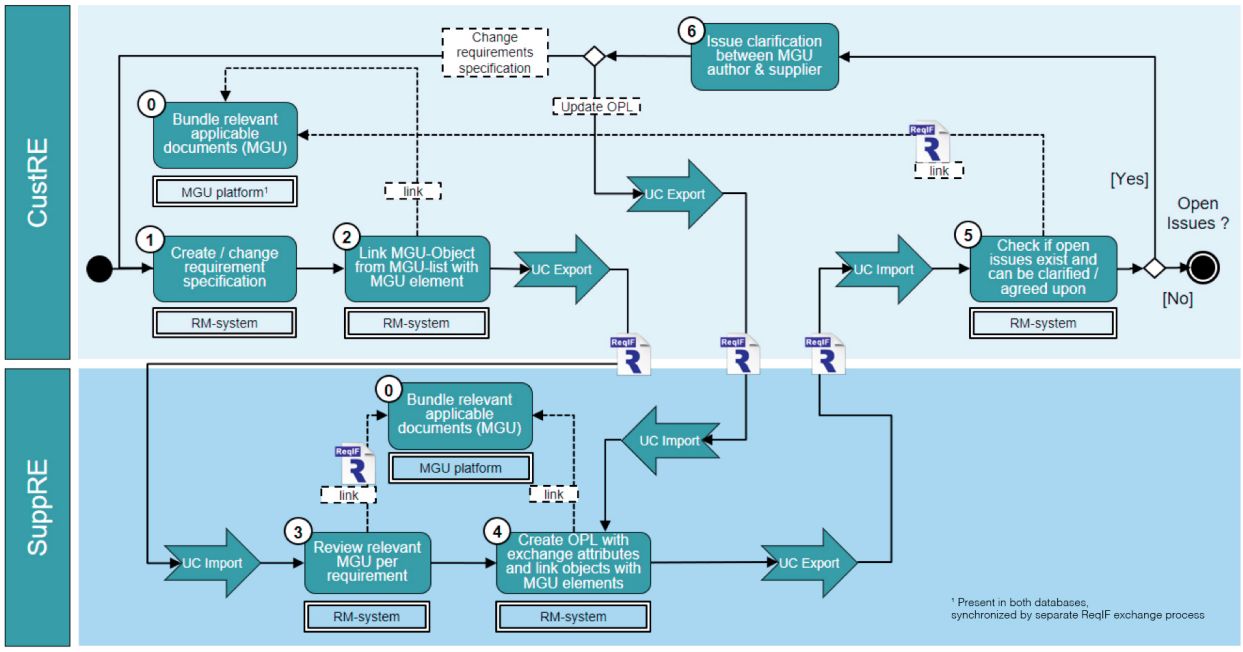


Figure 10: Use case diagram "Accompanying documents exchange between customer and supplier"

Benefit

- Same MGU basis on customer and supplier side will be used for evaluation
- Transparency on MGU changes (versions, change management)
- Efficient management of MGU documents (single source)
- Reuse of evaluation results on supplier side

Notes and deduced requirements

Implementation requirements and acceptance criteria will be documented in the derived user stories.

3 User-defined attributes

The exchange of requirements in the Stakeholder Request Clarification use cases (cf. chapter 2.4) is usually performed a multiple number of times. The workflow behind is controlled by a set of well-defined attributes. These attributes are part of a ReqIF exchange package. The set of attributes includes status attributes both for the customer and the supplier, comment fields, and other important information needed to manage the process.

The members of the ReqIF-WF agreed on a set of attributes. This agreement comprises naming conventions, attribute names and allowed values. Chapter 3.1 contains basic thoughts about attribute handling in ReqIF. In chapter 3.2 the user-definedattributes are described. To avoid an overloading of chapter 2 the status attributes and status transitions are described here separately.

3.1 ReqIF conventions and agreements on attributes

Requirement management tools provide out-of-the-box attributes. Some of them are automatically set like a modification date or creator. Others are created by the user of the tool like object heading or text. For the exchange of such system attributes the implementors agreed on a set of standard attributes. These attributes are prefixed with "ReqIF." to distinguish them from user defined attributes. The implementor agreements are documented in the „ReqIF Implementation Guide" [2]. An importing system maps such attributes to their system specific attributes.

Several system attributes are read-only and set automatically by the RM tool, e.g. the author of a requirement. When importing a ReqIF file, these attributes may get overridden and get lost. For example, instead of importing the name of the author of the requirement, the name of the importing person would be stored in the RM system. To denote that the attribute contains system information from the sending system the naming convention "ReqIF.ForeignXXX" will be used, e.g. "ReqIF.ForeignCreatedBy". During import the prefix "ReqIF." will be stripped. After import, the RM system will handle 2 attributes. For the example above this will result in the attributes "CreatedBy" (specifies the importing person), and "ForeignCreatedBy" (specifies the author of the requirement).

User-defined attributes are attributes created by the user of a requirements authoring tool. In contrast to system attributes they are not managed by the tool itself. Nevertheless, in the course of the harmonization of the requirements exchange processes it is advisable to get a common understanding and agreement on such attributes. This ensures that not every user or project uses different names or values. For the attributes defined by the ReqIF Workflow Forum for these purposes the prefix "ReqIF-WF." is used to distinguish them from other user-defined attributes.

In accordance with the conventions of the ReqIF Implementation Guide, the upper camel case notation should be used for attribute names and values.

3.2 Attributes for the ReqIF Use Cases

The following table shows the attributes used for the various use cases defined in this recommendation. It includes ReqIF built-in attributes, system attributes as defined in the "ReqIF Implementation Guide" and agreed-on user-defined attributes. The latter ones are explained in detail in the next chapter. For a description of the built-in and system attributes see [1] and [2].

[1] The laws, international, national and company standards are not part of the discussion between customer and supplier.

Attribute	Description	Attribute type	Remarks
ReqIF.ChapterName	Description of exchange heading	[xhtml]	
ReqIF.Text	Description of exchange information incl. embedded data	[xhtml]	Embedded data: e.g. Images, OLE objects, etc.
ReqIF-WF.Revision	Revision of the exported requirement	[String]	See usage in the attribute description
ReqIF-WF.ProductVariant	Product variant information	[Enum]	See usage in chapter "Variant Management"
ReqIF-WF.CustomerStatus	Voting information from customer view	[Enum]	
ReqIF-WF.SupplierStatus	Voting information from supplier view	[Enum]	
ReqIF-WF.CustomerComment	Voting relevant customer comment	[xhtml]	It should be ensured that both process partners could read the information completely. E.g., avoid OLE objects or attachments, if not supported by all involved systems.
ReqIF-WF.SupplierComment	Voting relevant supplier comment	[xhtml]	It should be ensured that both process partners could read the information completely. E.g., avoid OLE objects or attachments, if not supported by all involved systems. Not necessary, if different ReqIF SPEC-OBJECTS are defined for the different requirements types.
ReqIF-WF.Type	Type of object	[Enum]	Not necessary, if different ReqIF SPEC-OBJECTS are defined for the different requirements types
ReqIF.ForeignID	Unique human-readable identifier on partner side	[String]	
LastChange	Last change status	[Timestamp]	
ReqIF.ForeignModifiedBy	Last Object Editor on client side	[String]	
ReqIF.ForeignDeleted	Marker for explicit deletion information	[Boolean]	See usage in chapter " 4.2.1.2 Explicit deletion of requirements"
ReqIF.ForeignBaseline	Baseline information	[String]	

Table 1: ReqIF attributes for the support of the use cases

3.3 ReqIF user-defined attributes

3.3.1 ReqIF-WF.CustomerStatus

With the attribute ReqIF-WF.CustomerStatus, the customer transmits his status of a requirement to the supplier. The following constraints apply:

- The customer only has write permission. He can change the status. A supplier can only read the ReqIF-WF. CustomerStatus value.

- The initial value of the attribute is [<empty>]. It must be set for requirements (i.e. elements with ReqIF-WF.Type = [Requirement]) before any exchange. The value can remain [<empty>] for headers (elements with ReqIF-WF.Type = [Heading]) and information (elements with ReqIF-WF.Type = [Information]).

The status value should not be used to draw conclusions on the success of a data exchange. I.e., take an empty attribute as an indication that the status has not been transmitted. Empty attributes are always allowed. To ensure that a value has really been exchanged other mechanisms like ID pairs or Q-Checks have to be used. The verification of allowed values and value combinations should be part of a Q-Check.

The status value should also not be used to distinguish whether a requirement under evaluation is new or changed. In both cases, the value [ToEvaluate] should be set. The supplier should recognize by other mechanisms whether a requirement is new or changed.

The following table lists the allowed values for the attribute ReqIF-WF.CustomerStatus.

Attribute	Value	Value Description
ReqIF-WF. CustomerStatus	[<empty>]	The customer does not address the object. Not allowed for objects of ReqIF-WF.Type [Requirement] and is subtypes.
	[NotToEvaluate]	The requirement is relevant, but no evaluation is expected. • Attribute can be set to this value during the first transmission • Attribute can be changed to this value during the assessment process to indicate that a requirement must no longer be evaluated • Supplier can react with SupplierStatus [ToBeClarified] if he has objections to the non-evaluation. Otherwise, the SupplierStatus remains [<empty>] and the workflow ends
	[ToEvaluate]	For the corresponding object an evaluation is expected by the supplier. • Attribute can be set to this value during the first transmission • Attribute can be changed to this value during the assessment process to indicate that a requirement has been modified and must be re-evaluated
	[Accepted]	The supplier evaluation is accepted by the customer.
	[NotAccepted]	The supplier evaluation is not accepted by the customer. • Setting this status terminates the evaluation for this object. The conflict must be solved in the further project course outside of this workflow.
	[ToBeClarified]	The requirement has to be clarified with the supplier. Details in ReqIF-WF.CustomerComment.
	[ProposedToDiscard]	The requirement is marked as invalid and has to be discarded. NOTE: This value is OPTIONAL until the new ReqIF-WF process is well-established in industry.
	[Discarded]	The requirement has been marked explicitly as discarded by the customer.

Table 2: ReqIF-WF.CustomerStatus values

3.3.2 ReqIF-WF.SupplierStatus

With the attribute ReqIF-WF.SupplierStatus, the supplier transmits his status of a requirement to the customer. The following constraints apply:

- The supplier only has write permission. He can change the status. A customer can only read the ReqIF-WF.SupplierStatus value.
- The attribute is [<empty>] for the first Customer -> Supplier exchange.
- A ReqIF-WF.SupplierStatus [<empty>] in the course of a clarification process means that the supplier has not (yet) responded. This is the initial state, as long as the supplier has not performed an evaluation. But it can also be reset to [<empty>] during the assessment process to indicate that there is not yet a response to a ReqIF-WF.CustomerStatus [ToEvaluate] or [ProposedToDiscard].
- A status has to be assigned to each requirement individually. It is not allowed to implicitly set the status for all requirements of a chapter by assigning a status value to the heading.
- The multi-level exchange (Customer -> Supplier -> n-Tier) is not yet addressed in this version of the recommendation.

The following table lists the allowed values for the attribute ReqIF-WF.SupplierStatus.

Attribute	Value	Value Description
ReqIF-WF.SupplierStatus	[<empty>]	Supplier has not yet started the evaluation, or the evaluation is in progress.
	[Agreed]	The supplier accepts the requirement.
	[NotAgreed]	The supplier does not accept the requirement. Details must be declared in ReqIF-WF.SupplierComment.
	[AgreedWithComment]	The supplier accepts the requirement with restrictions. The restrictions are explained by the supplier in the comment attribute ReqIF-WF.SupplierComment.
	[ToBeClarified]	The supplier requires clarification. The supplier must actively clarify the matter with the customer. Details must be declared in ReqIF-WF.SupplierComment.
	[DiscardDeployed]	The supplier confirms that he will not implement the discarded requirement.

Table 3: ReqIF-WF.SupplierStatus values

3.3.3 ReqIF-WF.Type

A classification is needed to be able to distinguish between requirements, headings, and comments. However, the already existing system-attribute ReqIF.Category is used for other purposes by the RM systems and exchange tools and can therefore not be used for the classification of elements.

The user-defined attribute ReqIF-WF.Type will be reserved for this purpose. The following table lists the allowed values.

Attribute	Value	Value Description
ReqIF-WF.Type	[Heading]	The object is a heading.
	[Information]	The object is for information, only.
	[Requirement]	The object is a requirement.

Table 4: ReqIF-WF.Type values

The values of the ReqIF-WF.Type attribute are not restricted to the list above. If the process partners wish to classify requirements in more detail, they can do so. For example, a bilateral agreement may enhance values like [Function], [Logic], or [Behavior].

If the customer converts a requirement to a comment or vice versa, this must be considered in the workflow. That means that the ReqIF-WF.CustomerStatus attribute has to be changed appropriately.

3.3.4 ReqIF-WF.Revision

A revision information is needed to identify the revision of an exchanged requirement.

The ReqIF Implementation Guideline defines ReqIF attributes for a revision or modification of a requirement. These attributes are created during export and added as ReqIF.Foreign attributes to the ReqIF file. Because the content of the imported ReqIF.Foreign attributes is not sent back to the foreign RM-system for comparison of revisions, the customer must define the exchange of requirements revision information individually with his supplier:

- One solution is that the customer inserts a revision information in the attribute ReqIF-WF.Revision and the supplier sends back the same attribute without a change.
- Another solution is that the supplier fills out the attribute ReqIF-WF.Revision with the revision information, he has received from the customer in ReqIF.Foreign attributes.

After import on customer side, the customer can than compare the received revision with his internal revision information. In addition, the customer can use a configuration management system to manage different versions of an exchanged specification, e.g. in different branches.

4 Stakeholder Request Clarification workflow

The workflow is controlled by the status attributes ReqIF-WF.CustomerStatus and ReqIF-WF.SupplierStatus. The workflow diagrams describe the change of status attribute values in response to customer and supplier decisions and the actual status value. Furthermore, the permissible states are documented.

The following diagram shows where the status changes take place in the use case "ReqIF for Stakeholder Request Clarification".

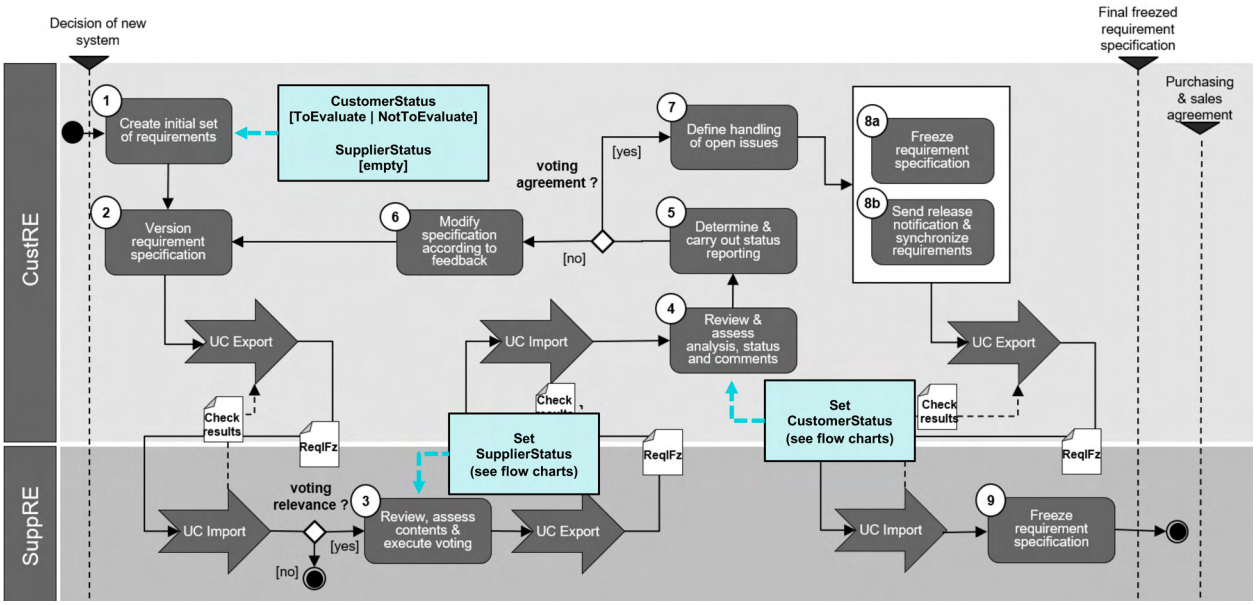


Figure 11: UC "ReqIF for Stakeholder Request Clarification" status values

Annex B contains the detailed diagrams.

The following figure provides a static view on the attribute value combinations for ReqIF-WF.CustomerStatus and ReqIF-WF.SupplierStatus and indicates whether a combination is not possible, is a transition state, or a final state. For example, ReqIF-WF.CustomerStatus=[ToEvaluate] and ReqIF-WF.SupplierStatus=[Agreed] is an allowed transition state.

Supplier Status \ Customer Status	<empty>	NotTo Evaluate	To Evaluate	Accepted	ToBe Clarified	Not Accepted	Proposed ToDiscard	Discarded
<empty>	1)							
Agreed								
ToBeClarified								
AgreedWithComment				2)		2)		
NotAgreed						2)		
DiscardDeployed								

Transition state
Invalid transition
Final state

1) Can be empty, if no voting is required (i.e., for ReqIF-WF.Type = [Heading, Information])
2) Is a "Final state" only, if this is defined by the Customer

Figure 12: Allowed states

Annex C shows, how the status model can be used in different project phases, e.g., quotation or engineering.

4.1 Conventions

4.1.1 Handling of requirements that a supplier cannot fully meet

Sometimes a requirement cannot be fully met by a supplier. The preferred workflow for this situation is that the supplier enforces a clarification and the customer changes the requirement:

- The supplier sets ReqIF-WF.SupplierStatus to [ToBeClarified] and describes the deviation request in the ReqIF-WF.SupplierComment.
- The customer modifies the requirement and requests a re-evaluation by setting the ReqIF-WF.CustomerStatus to [ToEvaluate].
- The supplier sets ReqIF-WF.SupplierStatus to [Agreed], Customer terminates with ReqIF-WF.CustomerStatus [Accepted].

There are situations where the customer is not able or not willing to change the requirement. In this case an alternative approach has been agreed to:

- The supplier sets ReqIF-WF.SupplierStatus to [AgreedWithComment] and documents the deviation in ReqIF-WF.SupplierComment.
- The customer terminates with ReqIF-WF.CustomerStatus [Accepted].
- The binding deviation should not have to be extracted from the complete comment conversation but should be summarized as the last entry in the ReqIF-WF.SupplierComment. The customer might document the deviation in his ReqIF-WF.CustomerComment, additionally.

4.1.2 Handling of sub elements

Requirements may be split into sub elements. This happens for example for the pre- and postconditions of a use case. The customer usually expects one evaluation for the main item only. To handle this situation, the following workflow was agreed on:

- All elements are classified as [Requirement] (or one of the specializations).
- The ReqIF-WF.CustomerStatus is set to [ToBeEvaluated] for the main element only.
- For the sub elements, the status is set to [NotToEvaluate].
- The supplier sets the ReqIF-WF.SupplierStatus for the main element only.
- For the sub-elements, ReqIF-WF.SupplierStatus remains [<empty>].

4.2 Partial update of an existing specification

To update an existing specification, all requirements must be part of the ReqIF file. If a requirement is missing in the ReqIF file, then it can be interpreted as a “partial update” of the specification or an “implicit deletion” of requirements. Because the interpretation is undetermined, the Sender must announce the intention clearly.

In an exchange scenario, use cases exist in which an existing specification is updated in the RM-system by a ReqIF file in which some requirements are missing:

a) The Customer has added new requirements in his module and updates his module with Supplier evaluations without the newly added requirements
b) The Customer exports fewer requirements than before and the Supplier updates his previously imported module

The importing tool must be able to handle both cases in a different way. In case a), the differences have to be ignored and in case b), the specification must be updated.

4.2.1 Handling of deleted requirements

In general, a deletion of requirements should be avoided in a SRC process and it is recommended that an invalid requirement of an exchanged specification should be discarded only by a discarding workflow.

However, in some use cases a deletion is intended e.g. to clean up a specification. After the exchange of deletion information, the Sender and Receiver can agree to exclude the deleted requirements from further exchange.

Possible deletion scenarios are described in the following sections.

4.2.1.1 Implicit deletion of requirements

An "implicit deletion" process makes sense only, if the importing tool identifies the differences between the ReqIF file and the target specification in the RM-system. The "implicitly deleted" requirements must be identifiable in the specification, so that the Receiver can perform post-processing steps on the discarded requirements after the import.

4.2.1.2 Explicit deletion of requirements

Depending on the RM-system a requirement can be tagged as deleted, but not removed from the specification. In this case, the requirement is exported by the RM-system with an additional ReqIF system attribute "ReqIF.ForeignDeleted". The value "true" is an "explicit deletion" information and can be used to identify discarded requirements.

5 Final remarks and outlook

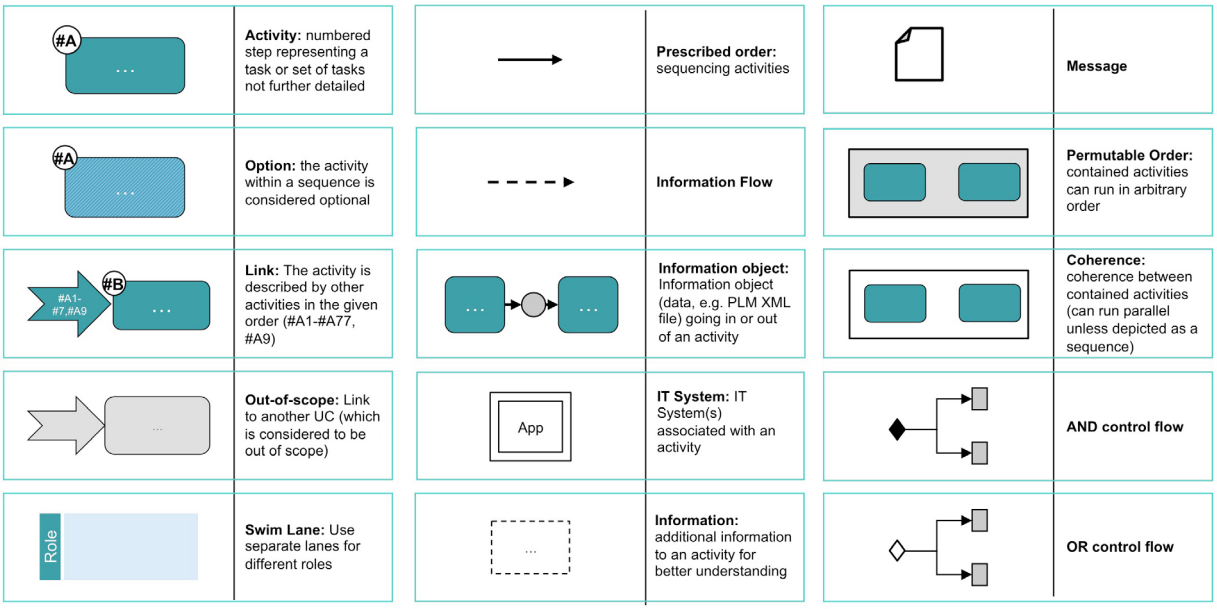
The objective of this publication is to make the elaborated use cases available to a broader public and to start a dialogue with the implementors. The authors are aware that further details have to be elaborated to implement the use cases. These results will be published in a future edition or amendment to this recommendation. Moreover, additional use cases may be addressed.

The quality criteria and the extent of such information have yet to be worked out and specified. This task must also be closely coordinated with the system vendors. The question as to where such checks should be performed also needs to be clarified. A quality check can be part of an exchange tool, but it can also be performed in the RM system or in a separate tool. The use cases may also give rise to additional requirements relating to the ReqIF standard or the Implementation Guidelines.

The use cases set the basis for the ReqIF Benchmarks which have been conducted since 2018. The scope addressed aspects like core exchange functionalities, conversion quality, performance, usability, and Q-Check capabilities.

Annex A: Glossary Swim Lane diagrams

The notation used in the use case diagrams is described below:



Annex B: Workflow diagrams

The status graph in the following tables shows reasonable status transitions.

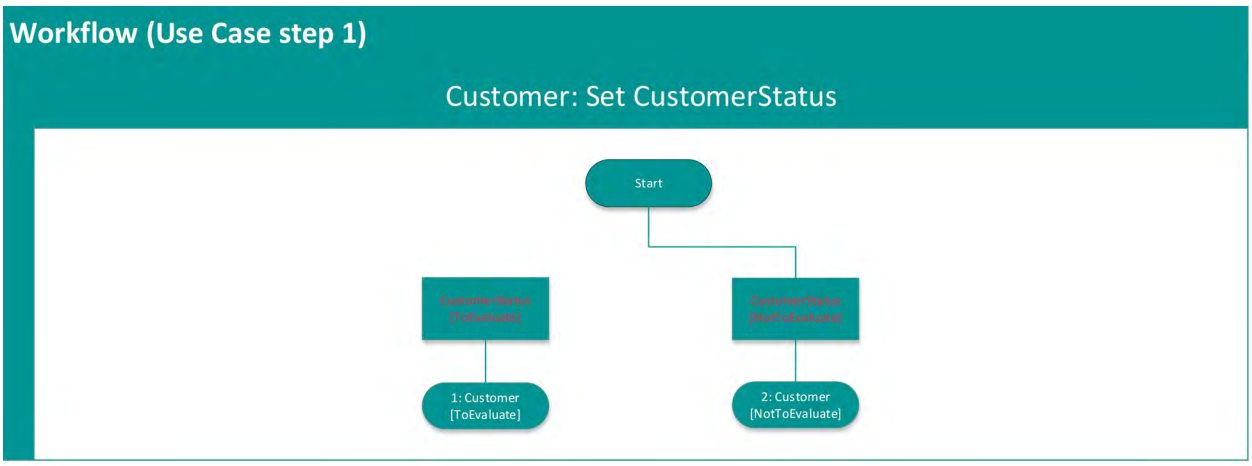


Figure 13: Workflow: Start

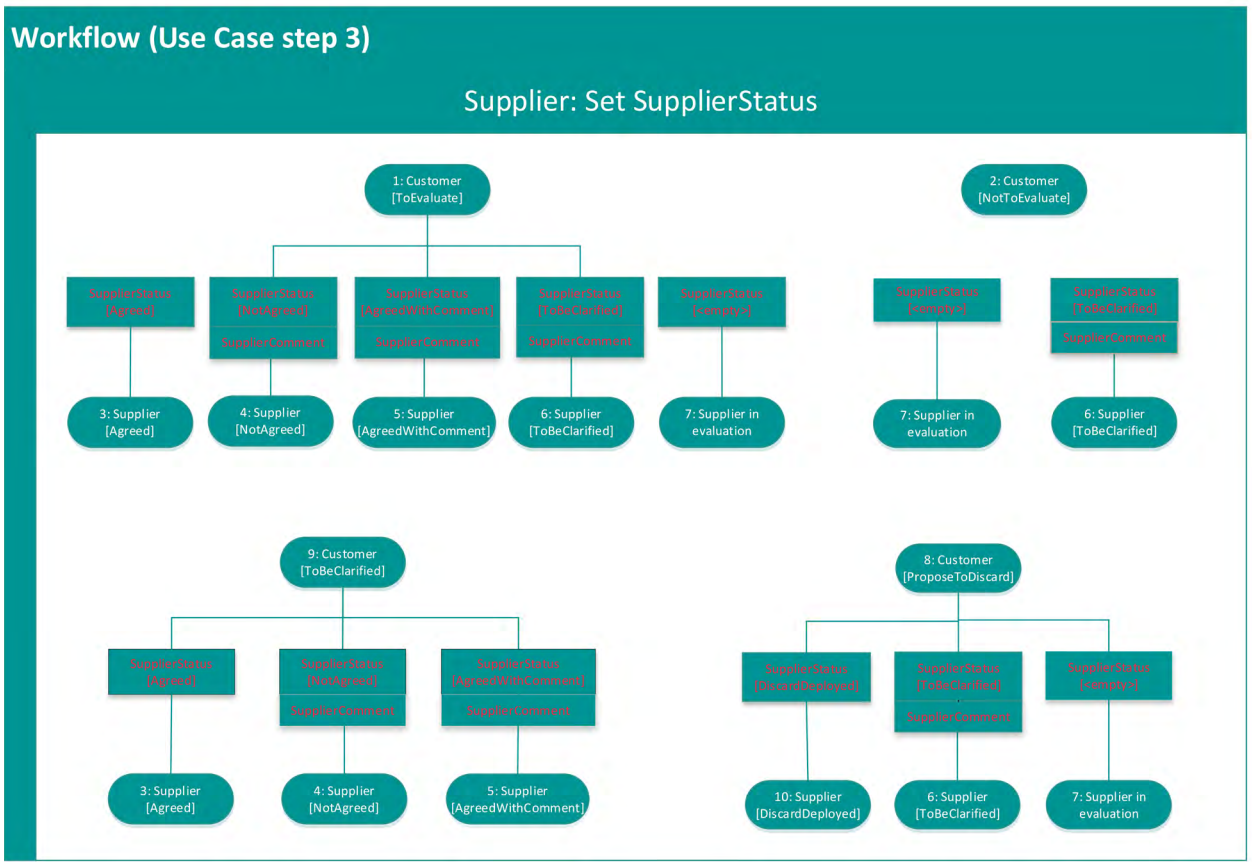


Figure 14: Workflow: Supplier evaluates

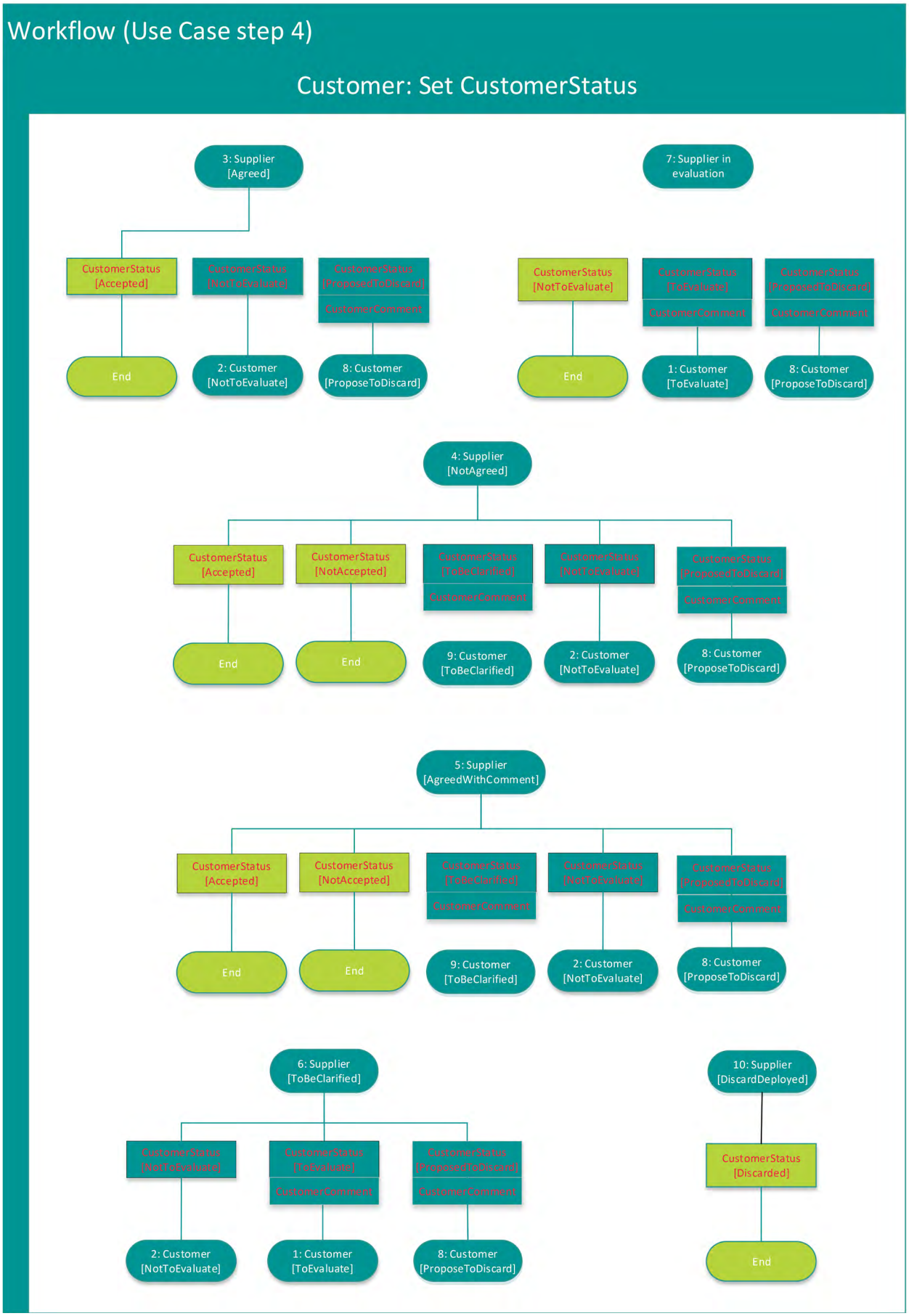


Figure 15: Workflow: Customer evaluates

Annex C: Examples of status usage

In a quotation phase of a project the specifications are exchanged and evaluated. The intention is to get a generic feedback, perhaps from different suppliers, about the realization of the requirements.

Therefore, various evaluations of the supplier are possible, to show critical requirements or requirements which cannot fulfilled completely and must be changed or defined in detail later. Normally the requirements are not adapted or discarded by the customer and disagreements will be clarified later.

Quatation Phase: Requirements are not completely agreed, but will not be discarded

Supplier Status \ Customer Status	<empty>	NotTo Evaluate	To Evaluate	Accepted	ToBe Clarified	Not Accepted	Proposed ToDiscard	Discarded
<empty>	1)							
Agreed								
ToBeClarified								
AgreedWithComment				2)		2)		
NotAgreed						2)		
DiscardDeployed								

Transition state
Invalid transition
Final state

1) Can be empty, if no voting is required (i.e., for ReqIF-WF.Type = [Heading, Information])
2) Is a “Final state” only, if this is defined by the Customer

Figure 16 Quotation Phase: Requirements are not completely agreed, but will not be discarded

In an engineering phase of a project, all valid requirements must be implemented until product release. Therefore, requirements are changed or discarded by the customer and final agreed by the supplier.

Enginering Phase: Requirements must be completely agreed or can be discarded

Supplier Status \ Customer Status	<empty>	NotTo Evaluate	To Evaluate	Accepted	ToBe Clarified	Not Accepted	Proposed ToDiscard	Discarded
<empty>	1)							
Agreed								
ToBeClarified								
AgreedWithComment				2)		2)		
NotAgreed						2)		
DiscardDeployed								

Transition state
Invalid transition
Final state

1) Can be empty, if no voting is required (i.e., for ReqIF-WF.Type = [Heading, Information])
2) Is a “Final state” only, if this is defined by the Customer

Figure 17 Engineering Phase: Requirements must be completely agreed or can be discarded

Annex D: Quality Checks

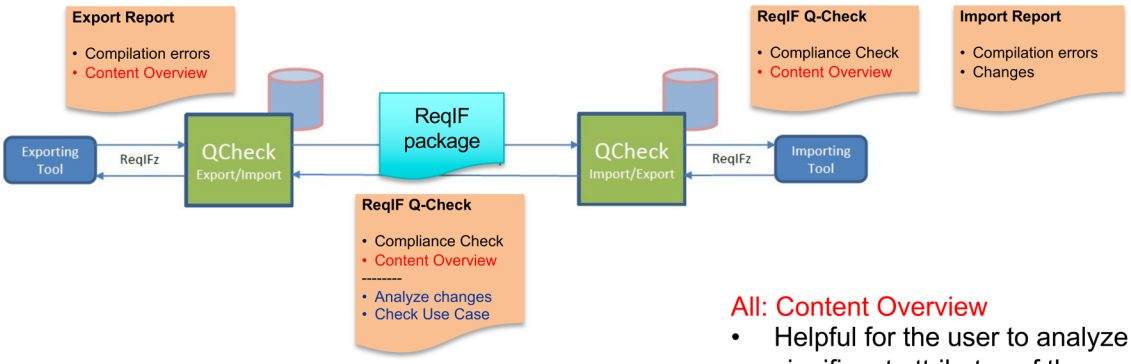
Quality Checks are performed after different step during export and import (see chapter 2.2 and 2.3). The RM-System (or connected exchange tools) shall support the user by adequate reports or dialogs. The following tables shows an overview about relevant check criteria.

Check	Scope	Attribute type
Compilation error report	Information per specification of the package	Hints about data type mapping
	Information per object of a specification	Hints about XHTML compilation problems, e.g. formatting, tables, fonts
		Hints about information loss in attributes, e.g compilation problems, data base restrictions (e.g. colors)
Compliance Check	ReqIF scheme-check	
	XTHML code check per object	Invalid ReqIF XHTML code, e.g. usage of wrong fonts and special characters
	Check of embedded objects (e.g. files)	Hints about missing files
Content Overview	Information about the package	Properties (title, tool, ID, comment …)
		Number of specifications
		Number of objects (not part of a specification)
	Information per specification	Properties (name, ID, type)
		Number of objects
		Number of relations
	Information about data types	Properties (name, ID, base type, literals)
	Information about spec types	Specification types (name, ID, attributes)
		Object types (name, ID, attributes)
		Spec-attributes (tame, type)
Change Analysis	Differences in significant attributes	See "Content Overview"
	Information per specification	Number of created objects
		Number of missing objects
		List of created links
		List of removed links
Use Case Check	Work Flow	Check of attributes and types
	Use Case	Statistics about attribute status
		Invalid status pairs of attributes
	Restrictions, not supported by every tool	Warning, if specific XHTML code is used,(e.g. colors, numbered list, tables
		Warning, if hierarchy is not supported
		Warning, if different fonts are used

Table 5: Q-Check criteria

Q-Checks can be supported by external tools [ASARO Q-Checker]. External tools can analyze the ReqIF package, but cannot identify compilation errors during export and import. This is the task of the RM-System. External tools can validate the XHTLM scheme and can support in analyzing the content of the ReqIF package as well as checking of use case specific configurations.

- Support of ReqIF Q-Checks within external tool:



- Additional features of an external tool:
  - Analyze changes between two ReqIF files
  - Check conditions for a specific Use Case

Figure 18: Support of ReqIF Q-Checks with external tool

Annex E: References

[1] OMG, "ReqIF specification: OMG Requirements Interchange Format (ReqIF), Version 1.2," 2016. [Online]. Available: http://www.omg.org/spec/ReqIF/1.2.

[2] prostep ivip, "ReqIF Implementation Guide, Version 1.6," 2019. [Online]. Available: http://www.prostep.org/mediathek/.

[3] HIS_AK-RE, "HIS Exchange Process for Requirements," 2012.

[4] prostep ivip, "ReqIF Benchmark," 2018ff. [Online]. Available: http:/www.prostep.org/mediathek/.

prostep IVIP

prostep ivip association

Dolivostraße 11
64293 Darmstadt
Germany

Phone +49-6151-9287336
Fax +49-6151-9287326
psev@prostep.com
www.prostep.org

ISBN 978-3-9820795-54
Version 2.1, 2022
PSI 18