Contributors:
Jon Holt (JDH), ATEGO
Simon Perry (SAP), ATEGO
Finn Overgaard Hansen (FOH), IHA
Stefan Hallerstede (SH), IHA

Editors:
Simon Perry, ATEGO
Jon Holt, ATEGO

Reviewers:
Uwe Schulze, UB
Ana Cavalcanti, UY
André Didier, UFPE
## Document History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Author</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>08.05.2012</td>
<td>J. Holt &amp; S. Perry</td>
<td>Initial version of Annex</td>
</tr>
<tr>
<td>1.1</td>
<td>31.05.2012</td>
<td>S. Perry</td>
<td>Review comments addressed.</td>
</tr>
</tbody>
</table>
Abstract
This Annex to deliverable D21.1 contains an output of the underlying SysML model upon which D21.1 is based.

Schedule:
- full version for review – 10.05.2012
- reviews back by – 24.05.2012
- final version to the EC – 31.05.2012
Table of Contents

1. Introduction ............................................................................................................. 10

2. Requirements View .................................................................................................. 11
   2.1. Source Element View ....................................................................................... 11
   2.2. Requirement Description View ......................................................................... 13
   2.2.1. CMMI ........................................................................................................ 13
   2.2.2. Case Study Requirements ......................................................................... 20
   2.2.3. Disseminator Requirements ..................................................................... 21
   2.2.4. Funder Requirements ............................................................................... 22
   2.2.5. Method Development Requirements ....................................................... 23
   2.2.6. Public Requirements ............................................................................... 24
   2.2.7. Related Projects Requirements ................................................................. 25
   2.2.8. Researcher Requirements ....................................................................... 26
   2.2.9. Reviewers Requirement .......................................................................... 27
   2.2.10. Roadmap User Requirements ................................................................. 28
   2.2.11. Standards Requirements ....................................................................... 29
   2.2.12. Tool Requirements ................................................................................ 30
   2.2.13. Vendor Requirements .......................................................................... 31
   2.2.14. COMPASS Requirements .................................................................. 32
   2.2.15. Description of Work .............................................................................. 32
   2.2.16. ISO 15288 ............................................................................................ 34
   2.2.17. Requirements Engineering for SoS ......................................................... 36
   2.2.18. SoS Guide Requirements ...................................................................... 40

2.3. Traceability View - Needs to Source Elements ..................................................... 44

2.4. Requirement Context Views .................................................................................. 45
   2.4.1. Process owner context ............................................................................ 45
   2.4.2. Requirements engineering process ............................................................ 48
   2.4.3. Requirements management process .......................................................... 53

2.5. Traceability View - Use Cases to Needs ............................................................... 56
   2.5.1. 'Analyse use case' to requirements .......................................................... 56
   2.5.2. 'Be driven by best practice' to requirements ............................................. 57
   2.5.3. 'Control process artefacts' to requirements ............................................. 58
   2.5.4. 'Define V&V criteria' to requirements ..................................................... 59
   2.5.5. 'Demonstrate approach' to requirements ................................................. 60
   2.5.6. 'Engage with stakeholders' to requirements ............................................ 61
   2.5.7. 'Generate process model' to requirements ................................................ 61
   2.5.8. 'Identify use case sources' to requirements ............................................. 62
   2.5.9. 'Manage requirements change' to requirements ...................................... 63
   2.5.10. 'Project Context' to requirements ............................................................ 64
   2.5.11. 'Provide approach for SoS' to requirements ........................................... 65
   2.5.12. 'Provide modelling language' to requirements ....................................... 66
   2.5.13. 'Provide process model...' to requirements ............................................ 67
   2.5.14. 'Provide SoS RE process' to requirements ............................................. 68
   2.5.15. 'Provide SoS requirement process' to requirements ................................ 69
   2.5.16. 'Provide SoS RM process' to requirements ............................................ 70
   2.5.17. 'Provide support' to requirements ........................................................... 71
   2.5.18. 'Provide tools' to requirements ............................................................... 72
   2.5.19. 'Understand context' to requirements ..................................................... 73
   2.5.20. 'Understand SoS context' to requirements ............................................. 74

3. Process Structure View ............................................................................................. 75
3.1. COMPASS - Ontology ........................................................................................................... 75
3.2. Process concepts ...................................................................................................................... 80
3.3. Requirement types .................................................................................................................... 82
3.4. Source Element Types ............................................................................................................ 84
4. Process Content View ................................................................................................................ 86
5. Stakeholder View ....................................................................................................................... 89
6. Information View ........................................................................................................................ 91
7. Process Behaviour View .............................................................................................................. 96
   7.1. PBV - Context Process ............................................................................................................ 96
   7.2. PBV - CS Process Analysis ..................................................................................................... 101
   7.3. PBV - Requirement Control Process ...................................................................................... 105
   7.4. PBV - Requirements Change Process ................................................................................... 108
   7.5. PBV - Requirements Monitor Process .................................................................................. 112
   7.6. PBV - SoS Requirements Development Process .................................................................. 114
   7.7. PBV - Traceability Process ................................................................................................... 117
   7.8. PBV - V&V Definition Process ............................................................................................. 120
8. Process Instance View ................................................................................................................. 123
   8.1. CS context definition scenario ............................................................................................... 123
   8.2. Monitoring - failed control .................................................................................................... 123
   8.3. Monitoring - failed traceability ............................................................................................. 124
   8.4. Monitoring scenario - OK ..................................................................................................... 124
   8.5. Set up scenario - failed CS process analysis ......................................................................... 125
   8.6. Set up scenario - failed requirements control ....................................................................... 126
   8.7. Set up scenario - failed traceability ....................................................................................... 126
   8.8. Set up scenario - OK ............................................................................................................. 127
   8.9. SoS Requirements Engineering scenario ............................................................................. 128
   8.10. Traceability- Scenarios to Use Cases .................................................................................. 128
Figures

Figure 1 COMPASS Source Elements .......................................................... 11
Figure 2 CMMI ......................................................................................... 13
Figure 3 Case Study Requirements .............................................................. 20
Figure 4 Disseminator Requirements ............................................................ 21
Figure 5 Funder Requirements ..................................................................... 22
Figure 6 Method Development Requirements ............................................. 23
Figure 7 Public Requirements ...................................................................... 24
Figure 8 Related Projects Requirements .................................................... 25
Figure 9 Researcher Requirements ............................................................... 26
Figure 10 Reviewers Requirement ............................................................... 27
Figure 11 Roadmap User Requirements ....................................................... 28
Figure 12 Standards Requirements .............................................................. 29
Figure 13 Tool Requirements ...................................................................... 30
Figure 14 Vendor Requirements .................................................................. 31
Figure 15 COMPASS Requirements ............................................................ 32
Figure 16 Description of Work .................................................................... 32
Figure 17 ISO 15288 .................................................................................. 34
Figure 18 Requirements Engineering for SoS ................................................. 36
Figure 19 SoS Guide Requirements ............................................................... 40
Figure 20 Needs - Source Elements .............................................................. 44
Figure 21 Process owner context .................................................................. 45
Figure 22 Requirements engineering process ............................................... 48
Figure 23 Requirements management process ............................................ 53
Figure 24 'Analyse use case' to requirements .............................................. 56
Figure 26 'Be driven by best practice' to requirements ................................ 57
Figure 27 'Control process artefacts' to requirements ................................... 58
Figure 28 'Define V&V criteria' to requirements ........................................... 59
Figure 29 'Demonstrate approach' to requirements ...................................... 60
Figure 30 'Engage with stakeholders' to requirements .................................. 61
Figure 31 'Generate process model' to requirements ...................................... 61
Figure 32 'Identify use case sources' to requirements ................................... 62
Figure 33 'Manage requirements change' to requirements .......................... 63
Figure 34 'Project Context' to requirements .............................................. 64
Figure 35 'Provide approach for SoS' to requirements ................................. 65
Figure 36 'Provide modelling language' to requirements ............................. 66
Figure 37 'Provide process model...' to requirements ................................. 67
Figure 38 'Provide SoS RE process' to requirements ................................... 68
Figure 39 'Provide SoS requirement process' to requirements ..................... 69
Figure 40 'Provide SoS RM process' to requirements ................................... 70
Figure 41 'Provide support' to requirements .............................................. 71
Figure 42 'Provide tools' to requirements ................................................... 72
Figure 43 'Understand context' to requirements .......................................... 73
Figure 44 'Understand SoS context' to requirements .................................... 74
Figure 45 COMPASS - Ontology ................................................................. 75
Figure 46 Process concepts ........................................................................ 80
Tables

Table 1 Traceability from Scenarios to Use Cases ........................................ 129
1. Introduction

This Annex to deliverable D21.1 contains an output of the underlying SysML model upon which D21.1 is based. It contains a section for each of the views that form part of the seven-views approach described in D21.1. The content of this document has been generated from the SysML model produced by Atego as part of the work on D21.1 and held in the Artisan Studio tool. Text that is formatted as blue underlined text thus

Use Case

indicates an underlying entity that is part of the SysML model.
2. Requirements View

2.1. Source Element View

The Source Element View contains all relevant source information that is required to get the requirements right. This view is used primarily as a mechanism to establish traceability in the system and provide links between the requirements and any other aspect of the system.

Any relevant source element can be captured on the Source Element View. Examples include, but are not limited to, conversations, emails, informal documents, formal requirements documents, systems specifications, system designs, processes, existing systems, brainstorming sessions, structured workshops, standards, laws and best-practice models.

**Name: CMMI**

**See:**
- [http://www.sei.cmu.edu/library/abstracts/reports/10tr033.cfm](http://www.sei.cmu.edu/library/abstracts/reports/10tr033.cfm)

**Name: COMPASS Project Description of Work**
Description: Seventh Framework Programme
Annex 1 - Description of Work
Project Full Title: Comprehensive Modelling for Advanced Systems of Systems

Name: COMPASS Requirement Set
Description: This is the spreadsheet that contains the initial requirements set that were captured in the first requirements workshop

Name: DoD System of Systems Guide

Name: ISO 15288

See:
- http://www.iso.org/iso/home.html

Name: Model-based Requirements Engineering
Description: 'Model-based Requirements Engineering' by Jon Holt, Simon Perry and Mike Brownsword. IET Publishing 2011

See:
- http://www.theiet.org/resources/books/computing/mbre.cfm

Name: Pragmatic Guide to Business Process Modelling
Jon Holt. BCS Publishing, 2009

See:
- http://shop.bcs.org/display.asp?K=9781906124120

Name: Requirements Engineering for SoS
Description: 'Requirements Engineering for Systems of Systems'
Grace A. Lewis, Edwin Morris, Patrick place, Soumya Simanta, Dennis B. Smith
Software Engineering Institute, Carnegie Mellon University, Pittsburgh PA, USA
Vancouver, Canada, March 23-26, 2009
2.2. Requirement Description View

2.2.1. CMMI

This diagram shows the Requirement Description View for the Requirements that were abstracted from the CMMI Source Element.

Name: Allocate product component requirements
Description: ‘Allocate the requirements for each product component.’

Refer to the Technical Solution process area for more information about allocation of requirements to products and product components. This specific practice provides information for defining the allocation of requirements but must interact with the specific practices in the Technical Solution process area to establish solutions to which the requirements are allocated.

The requirements for product components of the defined solution include allocation of product performance; design constraints; and fit, form, and function to meet requirements and facilitate production. In cases where a higher level requirement specifies performance that will be the responsibility of two or more product components, the performance must be partitioned for unique allocation to each product component as a derived requirement.’

Name: Analyse and validate requirements
Description: ‘The requirements are analyzed and validated, and a definition of required functionality is developed.’
The specific practices of the Analyze and Validate Requirements specific goal support the development of the requirements in both the Develop Customer Requirements specific goal and the Develop Product Requirements specific goal. The specific practices associated with this specific goal cover analyzing and validating the requirements with respect to the user’s intended environment. Analyses are performed to determine what impact the intended operational environment will have on the ability to satisfy the stakeholders’ needs, expectations, constraints, and interfaces. Considerations, such as feasibility, mission needs, cost constraints, potential market size, and acquisition strategy, must all be taken into account, depending on the product context. A definition of required functionality is also established. All specified usage modes for the product are considered, and a timeline analysis is generated for time critical sequencing of functions.

The objectives of the analyses are to determine candidate requirements for product concepts that will satisfy stakeholder needs, expectations, and constraints; and then to translate these concepts into requirements. In parallel with this activity, the parameters that will be used to evaluate the effectiveness of the product are determined based on customer input and the preliminary product concept. Requirements are validated to increase the probability that the resulting product will perform as intended in the use environment.

**Name: Analyse requirements**

Description: *Analyze requirements to ensure that they are necessary and sufficient.*

In light of the operational concept and scenarios, the requirements for one level of the product hierarchy are analyzed to determine whether they are necessary and sufficient to meet the objectives of higher levels of the product hierarchy. The analyzed requirements then provide the basis for more detailed and precise requirements for lower levels of the product hierarchy. As requirements are defined, their relationship to higher level requirements and the higher level defined functionality must be understood. One of the other actions is the determination of which key requirements will be used to track progress. For instance, the weight of a product or size of a software product may be monitored through development based on its risk.

**Name: Analyse requirements to achieve balance**

Description: *Analyze requirements to balance stakeholder needs and constraints.*

Stakeholder needs and constraints can address cost, schedule, performance, functionality, reusable components, maintainability, or risk.

**Name: Develop customer requirements**

Description: *Stakeholder needs, expectations, constraints, and interfaces are collected and translated into customer requirements.*
The needs of stakeholders (e.g., customers, end users, suppliers, builders, testers, manufacturers, and logistics support personnel) are the basis for determining customer requirements. The stakeholder needs, expectations, constraints, interfaces, operational concepts, and product concepts are analyzed, harmonized, refined, and elaborated for translation into a set of customer requirements. Frequently, stakeholder needs, expectations, constraints, and interfaces are poorly identified or conflicting. Since stakeholder needs, expectations, constraints, and limitations should be clearly identified and understood, an iterative process is used throughout the life of the project to accomplish this objective. To facilitate the required interaction, a surrogate for the end user or customer is frequently involved to represent their needs and help resolve conflicts. The customer relations or marketing part of the organization as well as members of the development team from disciplines such as human engineering or support can be used as surrogates. Environmental, legal, and other constraints should be considered when creating and resolving the set of customer requirements.

**Name: Develop product requirements**
**Description:** 'Customer requirements are refined and elaborated to develop product and product component requirements.'

Customer requirements are analyzed in conjunction with the development of the operational concept to derive more detailed and precise sets of requirements called “product and product component requirements.” Product and product component requirements address the needs associated with each product lifecycle phase. Derived requirements arise from constraints, consideration of issues implied but not explicitly stated in the customer requirements baseline, and factors introduced by the selected architecture, the design, and the developer’s unique business considerations. The requirements are re-examined with each successive, lower level set of requirements and functional architecture, and the preferred product concept is refined. The requirements are allocated to product functions and product components including objects, people, and processes. The traceability of requirements to functions, objects, tests, issues, or other entities is documented. The allocated requirements and functions are the basis for the synthesis of the technical solution. As internal components are developed, additional interfaces are defined and interface requirements are established.

**Name: Develop the customer requirements**
**Description:** 'Transform stakeholder needs, expectations, constraints, and interfaces into customer requirements.'

The various inputs from the relevant stakeholders must be consolidated, missing information must be obtained, and conflicts must be resolved in documenting the recognized set of customer requirements. The customer requirements may include needs, expectations, and constraints with regard to verification and validation. In some situations, the customer provides a set of requirements to the project, or the requirements exist as an output of a previous project’s activities.
In these situations, the customer requirements could conflict with the relevant stakeholders' needs, expectations, constraints, and interfaces and will need to be transformed into the recognized set of customer requirements after appropriate resolution of conflicts.

Relevant stakeholders representing all phases of the product's lifecycle should include business as well as technical functions. In this way, concepts for all product-related lifecycle processes are considered concurrently with the concepts for the products. Customer requirements result from informed decisions on the business as well as technical effects of their requirements.'

**Name: Elicit needs**
**Description:** 'Elicit stakeholder needs, expectations, constraints, and interfaces for all phases of the product lifecycle'

Eliciting goes beyond collecting requirements by proactively identifying additional requirements not explicitly provided by customers. Additional requirements should address the various product lifecycle activities and their impact on the product.'

**Name: Establish a definition of required functionality**
**Description:** 'Establish and maintain a definition of required functionality.'

The definition of functionality, also referred to as "functional analysis," is the description of what the product is intended to do. The definition of functionality can include actions, sequence, inputs, outputs, or other information that communicates the manner in which the product will be used.

Functional analysis is not the same as structured analysis in software development and does not presume a functionally oriented software design. In object-oriented software design, it relates to defining what are called "services" or "methods." The definition of functions, their logical groupings, and their association with requirements is referred to as a functional architecture.'

**Name: Establish operational concepts and scenarios**
**Description:** 'The requirements are analyzed and validated, and a definition of required functionality is developed.'

The specific practices of the Analyze and Validate Requirements specific goal support the development of the requirements in both the Develop Customer Requirements specific goal and the Develop Product Requirements specific goal. The specific practices associated with this specific goal cover analyzing and validating the requirements with respect to the user’s intended environment. Analyses are performed to determine what impact the intended operational environment will have on the ability to satisfy the stakeholders’ needs, expectations, constraints, and interfaces. Considerations, such as feasibility, mission needs, cost constraints, potential market size, and acquisition strategy, must all be taken into account, depending on the product context. A definition of
required functionality is also established. All specified usage modes for the product are considered, and a timeline analysis is generated for time critical sequencing of functions.

The objectives of the analyses are to determine candidate requirements for product concepts that will satisfy stakeholder needs, expectations, and constraints; and then to translate these concepts into requirements. In parallel with this activity, the parameters that will be used to evaluate the effectiveness of the product are determined based on customer input and the preliminary product concept. Requirements are validated to increase the probability that the resulting product will perform as intended in the use environment.

**Name: Establish product and product component requirements**
**Description:** 'The customer requirements may be expressed in the customer’s terms and may be nontechnical descriptions. The product requirements are the expression of these requirements in technical terms that can be used for design decisions. An example of this translation is found in the first House of Quality Function Deployment, which maps customer desires into technical parameters. For instance, “solid sounding door” might be mapped to size, weight, fit, dampening, and resonant frequencies.

Product and product component requirements address the satisfaction of customer, business, and project objectives and associated attributes, such as effectiveness and affordability. Derived requirements also address the cost and performance of other lifecycle phases (e.g., production, operations, and disposal) to the extent compatible with business objectives. The modification of requirements due to approved requirement changes is covered by the “maintain” function of this specific practice; whereas, the administration of requirement changes is covered by the Requirements Management process area.'

**Name: Identify inconsistencies between project work and requirements**
**Description:** 'Identify inconsistencies between the project plans and work products and the requirements

This specific practice finds the inconsistencies between the requirements and the project plans and work products and initiates the corrective action to fix them'

**Name: Identify interface requirements**
**Description:** 'Identify interface requirements.

Interfaces between functions (or between objects) are identified. Functional interfaces may drive the development of alternative solutions described in the Technical Solution process area.

Refer to the Product Integration process area for more information about the management of interfaces and the integration of products and product components.'
Interface requirements between products or product components identified in the product architecture are defined. They are controlled as part of product and product component integration and are an integral part of the architecture definition.

**Name: Maintain bi-directional traceability of requirements**

Description: *Maintain bidirectional traceability among the requirements and work products.*

The intent of this specific practice is to maintain the bidirectional traceability of requirements for each level of product decomposition. (See the definition of “bidirectional traceability” in the glossary.) When the requirements are managed well, traceability can be established from the source requirement to its lower level requirements and from the lower level requirements back to their source. Such bidirectional traceability helps determine that all source requirements have been completely addressed and that all lower level requirements can be traced to a valid source.

Requirements traceability can also cover the relationships to other entities such as intermediate and final work products, changes in design documentation, and test plans. The traceability can cover horizontal relationships, such as across interfaces, as well as vertical relationships. Traceability is particularly needed in conducting the impact assessment of requirements changes on the project’s activities and work products.

**Name: Manage requirement changes**

Description: *Manage changes to the requirements as they evolve during the project.*

During the project, requirements change for a variety of reasons. As needs change and as work proceeds, additional requirements are derived and changes may have to be made to the existing requirements. It is essential to manage these additions and changes efficiently and effectively. To effectively analyze the impact of the changes, it is necessary that the source of each requirement is known and the rationale for any change is documented. The project manager may, however, want to track appropriate measures of requirements volatility to judge whether new or revised controls are necessary.

**Name: Manage requirements**

Description: *Requirements are managed and inconsistencies with project plans and work products are identified.*

The project maintains a current and approved set of requirements over the life of the project by doing the following:

- Managing all changes to the requirements
- Maintaining the relationships among the requirements, the project plans, and the work products
• Identifying inconsistencies among the requirements, the project plans, and the work products
• Taking corrective action.

**Name: Obtain an Understanding of Requirements**
**Description:** 'Develop an understanding with the requirements providers on the meaning of the requirements.

As the project matures and requirements are derived, all activities or disciplines will receive requirements. To avoid requirements creep, criteria are established to designate appropriate channels, or official sources, from which to receive requirements. The receiving activities conduct analyses of the requirements with the requirements provider to ensure that a compatible, shared understanding is reached on the meaning of the requirements. The result of this analysis and dialog is an agreed-to set of requirements.'

**Name: Obtain commitment to requirements**
**Description:** 'Obtain commitment to the requirements from the project participants.'

Whereas the previous specific practice dealt with reaching an understanding with the requirements providers, this specific practice deals with agreements and commitments among those who have to carry out the activities necessary to implement the requirements.

Requirements evolve throughout the project, especially as described by the specific practices of the Requirements Development process area and the Technical Solution process area. As the requirements evolve, this specific practice ensures that project participants commit to the current, approved requirements and the resulting changes in project plans, activities, and work products.'

**Name: Validate requirements**
**Description:** 'Validate requirements to ensure the resulting product will perform as intended in the user’s environment.'

Requirements validation is performed early in the development effort with end users to gain confidence that the requirements are capable of guiding a development that results in successful final validation. This activity should be integrated with risk management activities. Mature organizations will typically perform requirements validation in a more sophisticated way using multiple techniques and will broaden the basis of the validation to include other stakeholder needs and expectations.

Examples of techniques used for requirements validation include the following:
• Analysis
• Simulations
2.2.2. Case Study Requirements

This diagram shows the Requirement Description View for the Requirements that were abstracted from the COMPASS Requirement Set Source Element.
2.2.3. Disseminator Requirements

This diagram shows the Requirement Description View for the Requirements that were abstracted from the COMPASS Requirement Set Source Element.
2.2.4. Funder Requirements

This diagram shows the Requirement Description View for the Requirements that were abstracted from the COMPASS Requirement Set Source Element.
2.2.5. **Method Development Requirements**

This diagram shows the **Requirement Description View** for the **Requirements** that were abstracted from the **COMPASS Requirement Set Source Element**.
2.2.6. Public Requirements

This diagram shows the Requirement Description View for the Requirements that were abstracted from the COMPASS Requirement Set Source Element.
2.2.7. Related Projects Requirements

This diagram shows the Requirement Description View for the Requirements that were abstracted from the COMPASS Requirement Set Source Element.
2.2.8. Researcher Requirements

This diagram shows the Requirement Description View for the Requirements that were abstracted from the COMPASS Requirement Set Source Element.
2.2.9. Reviewers Requirement

This diagram shows the Requirement Description View for the Requirements that were abstracted from the COMPASS Requirement Set Source Element.
2.2.10. Roadmap User Requirements

This diagram shows the Requirement Description View for the Requirements that were abstracted from the COMPASS Requirement Set Source Element.
2.2.11. Standards Requirements

This diagram shows the Requirement Description View for the Requirements that were abstracted from the COMPASS Requirement Set Source Element.
2.2.12. Tool Requirements

This diagram shows the Requirement Description View for the Requirements that were abstracted from the COMPASS Requirement Set Source Element.
2.2.13. Vendor Requirements

This diagram shows the Requirement Description View for the Requirements that were abstracted from the COMPASS Requirement Set Source Element.
2.2.14. COMPASS Requirements

This diagram shows the Requirement Description View for the Requirements that were abstracted from the COMPASS Project Description of Work Source Element.

2.2.15. Description of Work

This diagram shows the Requirement Description View for the Requirements that were abstracted from the COMPASS Project Description of Work Source Element.

Name: Capture and disseminate in tool
Description: 'These guidelines will be captured and disseminated in Atego Process Director'
**Name: Cover different levels of abstraction**
Description: '... will take into account the need for requirements at different levels, including those for the architectural and constituent systems levels'

'... including problems arising from new and changing requirements, both for the SoS and its constituent systems.'

'Management of requirements throughout the life cycle of a SoS and its constituent systems will be covered.'

**Name: Cover functional and non-functional**
Description: 'This task will investigate the specification of functional and non-functional requirements'

**Name: Develop guidance for requirements**
Description: 'It will address the three crucial areas of requirements specification, systems architecture and system integration'

**Name: Develop guidance for systems architecture**
Description: 'It will address the three crucial areas of requirements specification, systems architecture and system integration'

**Name: Develop guidance for systems integration**
Description: 'It will address the three crucial areas of requirements specification, systems architecture and system integration'

**Name: Develop guidance over-arching approach**
Description: 'These results will then be combined as guidance for an over-arching approach to systems engineering for SoS'

**Name: Develop guidelines for SoS approach**
Description: 'This work package will develop guidelines for the COMPASS approach to SoS development...'

**Name: Manage change**
Description: '... including problems arising from new and changing requirements, both for the SoS and its constituent systems.'

'Changes at the constituent-systems level that impact on the SoS will also be covered by such a process...'

**Name: Manage requirements throughout life cycle**
Description: 'Management of requirements throughout the life cycle of a SoS and its constituent systems will be covered.'
**Name: Meet best practice**
Description: 'The guidelines will incorporate the principles of best existing practice in systems engineering for SoS, such as those recommended by the US DoD'

**Name: Validate guidelines**
Description: 'The guidelines will be validated by being used in practice in the development of the industrial case studies.'

"...validation, using both formal and informal techniques"

### 2.2.16. ISO 15288

This diagram shows the Requirement Description View for the Requirements that were abstracted from the ISO 15288 Source Element

**Name: Analyse and maintain stakeholder requirements**
Description: 'Analyze and maintain stakeholder requirements.'

**Name: Analyse elicited requirements**
Description: 'Analyze the complete set of elicited requirements'

**Name: Confirm stakeholder requirements**
Description: 'Establish with stakeholders that their requirements are expressed correctly'
**Name: Define activity sequences**
Description: 'Define a representative set of activity sequences to identify all required services that correspond to anticipated operational and support scenarios and environments'

**Name: Define solution constraints**
Description: 'Define the constraints on a system solution that are unavoidable consequences of existing agreements, management decisions and technical decisions'

**Name: Define stakeholder requirements**
Description: 'Define stakeholder requirements'

**Name: Elicit requirements**
Description: 'Elicit stakeholder requirements from the identified stakeholders'

**Name: Elicit stakeholder requirements**
Description: 'Elicit stakeholder requirements.'

**Name: Feed back analysed requirements**
Description: 'Feed back the analyzed requirements to applicable stakeholders to ensure that the needs and expectations have been adequately captured and expressed'

**Name: Identify stakeholders**
Description: 'Identify the individual stakeholders or stakeholder classes who have a legitimate interest in the system throughout its life cycle'

**Name: Identify user/system interactions**
Description: 'Identify the interaction between users and the system.'

**Name: Maintain requirements traceability**
Description: 'Maintain stakeholder requirements traceability to the sources of stakeholder need'

**Name: Record stakeholder requirements**
Description: 'Record the stakeholder requirements in a form suitable for requirements management through the life cycle and beyond'

**Name: Resolve problems**
Description: 'Resolve requirements problems'

**Name: Specify critical qualities**
Description: 'Specify health, safety, security, environment and other stakeholder requirements and functions that relate to critical qualities'
2.2.17. Requirements Engineering for SoS

This diagram shows the Requirement Description View for the Requirements Engineering for SoS Source Element.

**Name: Analyse capability dependency**
Description: 'Once the system capabilities have been identified, the next step is to identify dependencies across capabilities. Capability dependencies can exist within a single system or across multiple systems' - Page 6

**Name: Analyse capability re-engineering**
Description: 'Some system capabilities require reengineering before they are ready for external use. Changes to existing capabilities may be expensive or could break the current system. Reengineering of capabilities should be done to maximize their reuse. This can be achieved by identifying requirements for modifications to capabilities such that 1) the existing systems have the least impact due to changes, 2) the capability has the appropriate granularity, and 3) the capabilities have high cohesion and low coupling. These attributes maximize the reusability of the reengineered capabilities' - Page 6

**Name: Analyse goal-capability mapping**
Description: 'Mapping goals from system trees to individual capabilities is a mechanism to identify individual capabilities that are either already available or are currently being developed. Creating a goal-capability mapping has several advantages. First, goals provide a good starting point for identifying capabilities at system level. Second, the capabilities can be identified at the same level of granularity as modeled in the goal tree. Finally, the goal-capability mapping relates the problem domain to the solution domain. It facilitates traceability and
completeness to capability identification. The system capabilities identified in this step will be used to realize SoS interactions’ - Page 6

Name: Analyse the gap
Description: 'Top-down and bottom-up analysis should be followed by a gap analysis. The main goal of this activity is to reconcile outputs from the top-down and bottom-up analysis and to identify gaps in terms of capabilities currently not provided by existing systems and capabilities that require modifications to support SoS goals’ - Page 6

Name: Consider CS and SoS
Description: 'At a high level, the goals of SoS requirements engineering are ... consideration of the perspective of the individual systems as well as the SoS’ - Page 3

Name: Consider current and future needs
Description: 'At a high level, the goals of SoS requirements engineering are ... consideration of SoS current and future needs against individual capabilities’ - Page 3

Name: Identify capability constraints
Description: 'A system capability may have several constraints associated with it. For example, if a capability exists in a legacy system and is implemented using an outdated technology or if the capability cannot be accessed outside a security enclave (e.g., a secure military network). Understanding of these constraints is important from a requirements engineering perspective because they can hinder and in some cases make it impossible to use the capability.’ - Page 6

Name: Identify common goals
Description: '... identification of goals from an SoS requirements perspective allows the identification of systems that can implement or provide similar capabilities. Concepts from commonality analysis can be used to identify common goals across goal trees of multiple participating systems’ - Page 5

Name: Identify conflicting goals
Description: 'It is important to identify and understand these conflicts because individual systems often give higher priority towards meeting their individual goals. Therefore, if an SoS relies heavily on individual systems to provide the necessary capabilities it must find appropriate resolutions for these conflicts with the individual systems.’ - Page 5

Name: Identify individual system capabilities
Description: 'Identification of actual capabilities needs to be performed for existing or planned individual systems. The goal of this activity is to identify coarse-grained functionality and features provided by individual systems in terms of inputs required, outputs expected and the constraints under which the capability works correctly.’ - Page 6
Name: Identify individual system capabilities and constraints
Description: 'SoS capabilities cannot be successfully composed without a proper understanding of the capabilities provided by individual systems. The bottom-up analysis involves identification of capabilities of constituent systems. This activity can be performed in parallel with the top-down activities and should provide valuable information to understand SoS interactions' - Page 6

Name: Identify needs of CS
Description: 'Identification of independent needs of individual systems (current and future) that constitute the SoS.' - Page 3

Name: Identify SoS and Individual System Goals
Description: 'The next step is the identification of goals of the SoS and of the individual participating systems' - Page 4

Name: Identify SoS and system environments
Description: 'Each constituent system, and the SoS itself, is influenced by its environment. For example ... unique decision support rules, governance organizations, and regulatory statutes. These and other aspects of the environment that must be considered include Entities, Influence and Evolution. Understanding the environment helps individual systems and the SoS anticipate future needs and engineer appropriate mechanisms (e.g., architectural) to adapt to these changes in the environment' - Page 4

Name: Identify SoS capabilities
Description: 'Identification of capabilities that are expected from the entire SoS. For example, consider the identification of capabilities and goals of a network of regional healthcare providers forming a Regional Health Information Organization (RHIO).’ - Page 3

Name: Identify SoS context
Description: 'The first step in the requirements engineering of an SoS is to identify its context: the type of SoS and the environments related to the SoS and the constituent systems.' - Page 4

Name: Identify SoS type
Description: 'SoS is provided in the DoD Systems of Systems Engineering Guide. This guide classifies SoS as directed, acknowledged, collaborative, or virtual, in which operational control gradually decreases from first to last. Operational control refers to centralized management of the technical and non-technical aspects of constituent system development, deployment and execution. Operational control is an important attribute from an SoS requirements perspective because it captures the extent to which the requirements for the individual systems are centrally defined by the SoS.' - Page 4
Name: Understand data-centric interactions
Description: 'Another common interaction in an SoS is data-centric, where data from multiple systems is processed (e.g., aggregated, filtered, transformed, and translated) to support SoS needs that cannot be met by any single system.' - Page 5

Name: Understand demands on SoS environment
Description: 'Understanding current and future demands on the SoS environment so that adaptable and robust solutions can be engineered. Multiple stakeholders and variability in operating contexts are some key challenges for satisfaction of this goal.' - Page 4

Name: Understand how CS capabilities can be used
Description: 'Understanding how existing capabilities provided by individual systems can be combined to meet the goals of the SoS' - Page 3

Name: Understand individual CS
Description: 'Understanding how best to engineer individual constituents systems so that they not only meet the requirements of interoperating with existing systems but also are flexible enough to meet future demands with the minimum amount of change' - Page 3

Name: Understand process-centric interactions
Description: 'The first type of interaction represents a process or a workflow supported by the SoS. Process-centric interactions involve flow of data and control across boundaries of multiple participating systems. The processes are implemented by threading capabilities provided by participating systems.' - Page 5

Name: Understand resource-centric interactions
Description: 'Resources controlled by various participating systems can be shared or pooled to provide capabilities to the SoS. Sharing of computing resources, such as CPU cycles and disk space in Grid and Cloud Computing environments, is an example of a resource-centric SoS interaction' - Page 5

Name: Understand SoS Interactions
Description: 'In an SoS, capabilities provided by participating systems interact to provide SoS capabilities. One useful view of an SoS is as a set the interactions that the SoS must support. Additionally, viewing an SoS as a set of interactions makes it easier to engineer individual interactions.' - Page 5
2.2.18. SoS Guide Requirements

This diagram shows the Requirement Description View for the Requirements that were abstracted from the DoD System of Systems Guide Source Element.

**Name: Allow requirements to evolve**
Description: ‘The requirements should evolve so that early experimentation and military utility assessments can be used to enhance the operational community’s understanding of the integrated SoS capability to be developed.’ - Page 74

**Name: Be aware of requirement processes**
Description: ‘Requirements for the constituent systems will typically be managed separately for each system by its systems engineer using their own processes. At a minimum, the SoS systems engineer needs to be informed about these processes.’ - Page 85

**Name: Continue requirements through architecture and design**
Description: ‘Requirements development in an SoS often continues through SoS architecture and design development and implementation, since the architecture in particular will generate requirements for systems in the SoS.’ – Page 74

**Name: Control traceability**
Description: ‘correlation and traceability between the desired capabilities and the configuration of the deployed SoS.’ – Page 86

**Name: Define measures**
Description: ‘SoS requirements need to be defined in terms of measures of outcome and mission measures of effectiveness to derive SoS measures of performance that can then be allocated to individual systems as part of the SoS process across the relevant SoS SE elements.’ - Page 86
Name: Describe requirements by functionality  
Description: '...the requirements should be described in terms of needed functionality and not implementation details' - Page 74

Name: Develop understanding of SoS and CS  
Description: 'In an SoS environment, requirements development requires an understanding of constituent system capabilities, high-level SoS requirements' - Page 74

Name: Ensure requirements activities are on-going  
Description: 'Because a SoS typically evolves over time, requirements may change based on both internal and external factors. As a result, requirements development may be an on-going SoS activity, and the SoS requirements will evolve as well.' - Page 74

Name: Evaluate alternative solutions  
Description: 'so that alternative ways to meet those requirements can be evaluated' - Page 74

Name: Focus requirements management  
Description: '...requirements management may focus on specific functional requirements of the SoS and individual systems...' - Page 86

Name: Identify stakeholders  
Description: 'The stakeholders for an SoS include users and proponents for the SoS, as well as the stakeholders for the constituent systems who may not share the perspective of the SoS.' - Page 74

Name: Identify duplicate requirements  
Description: 'In an SoS context, redundancy across individual systems may be perfectly acceptable, desirable and even necessary when considering the roles that individual systems play apart from the SoS. In some cases, duplicative requirements or functionality across the constituent systems may cause SoS conflicts.' - Page 85

Name: Identify new requirements  
Description: 'In an environment of evolving threats and an evolving concept of operations, a critical aspect of the requirements management activity is the identification and management of new requirements over time,' - Page 86

Name: Involve stakeholders  
Description: 'Building a common understanding of SoS needs and approaches with the SoS and constituent system stakeholders is key to SoS success, but building a stakeholder community takes time.' - Page 74

41
**Name: Manage communications in SoS context**
Description: '...it is also very important to address and manage the communications and data exchange requirements in the context of the SoS' - Page 86

**Name: Manage constituent system requirements**
Description: 'Requirements for the constituent systems will typically be managed separately for each system by its systems engineer using their own processes' - Page 85

**Name: Monitor changes in constituent systems**
Description: 'At a minimum the SoS systems engineer needs to remain cognizant of the changing requirements of the constituent systems' - Page 75

**Name: Recognise redundant requirements**
Description: 'The SoS systems engineer needs to recognize when there are redundant requirements across constituent systems' - Page 85

**Name: Reflect new CS requirements in SoS**
Description: 'Additionally there needs to be a way to ensure that new requirements on systems to meet the SoS needs are reflected in the systems' requirements management processes in a way that is linked to SoS requirements management.' - Page 85

**Name: Requirement management requirement**
Description: 'Requirements Management provides traceability back to user-defined capabilities as documented through the Joint Capabilities Integration and Development System. In evolutionary acquisition, the management of requirements definition and changes to requirements takes on an added dimension of complexity' - Page 85

**Name: Requirements development requirement**
Description: 'the Requirements Development process takes all inputs from relevant stakeholders and translates the inputs into technical requirements' - Page 73

**Name: Resolve conflicts**
Description: 'It may be important to manage and resolve any conflicts, but it may be too costly or disruptive to attempt to back out contentious, redundant requirements or functions.' - Page 86

**Name: Trace SoS requirements**
Description: 'Requirements Management begins with the developed SoS requirements and traces the SoS requirements throughout the process and over time.' - Page 85
Name: Understand constituent systems
Description: 'The major challenge for SoS requirements development is in the complexity of developing requirements for a broad capability within the context of systems that have their own requirements and stakeholders.' - Page 74

Name: Understand interactions between SoS and CS
Description: 'In an SoS environment, requirements development requires an understanding of constituent system capabilities, high-level SoS requirements, and the interactions between the two.' - Page 74

Name: Use available solutions
Description: 'During each iteration of SoS development, the SoS systems engineer reviews requirements and seeks to address them with available solutions, factoring in the requirements and development plans of the systems in the SoS.' - Page 74

Name: Validate requirements using process
Description: 'For new acquisitions, requirements are developed and validated through a formal requirements process' - Page 74
2.3. Traceability View - Needs to Source Elements

The traceability between the Needs and the Source Elements is represented using a traceability view, and example of which is shown here.

The diagram shows how the SysML «trace» relationship is used to establish traceability between two elements.
2.4. Requirement Context Views

2.4.1. Process owner context

![Diagram of Process owner context]

The diagram here shows the context of the process owner, and the main use case is concerned with ‘Provide guidelines for SoS requirements’.

The following use cases are always part of the ‘Provide guidelines for SoS requirements’ use case and, therefore, must be satisfied to meet the requirements of the project:

- ‘Provide tool support’ that states that tool support for the processes is explicitly required and is related to the ‘Tool Vendor’ stakeholder.
- ‘Provide SoS requirements engineering process’, that requires an explicit set of processes related to requirements engineering, and is of interest to the ‘Requirements Engineer’ stakeholder.
- ‘Provide SoS requirements management process’, that requires an explicit set of processes related to requirements management, and is of interest to the ‘Requirements manager’ stakeholder.

The following use cases are constraints on the ‘Provide guidelines for SoS requirements’ use case as they limit the way that the main use case can be realised and, therefore, must be satisfied to meet the requirements of the project:

- ‘Apply to different types of SoS’, that ensures that the guidelines produced are applicable to the four classic types of system of systems (directed,
acknowledged, collaborative and virtual) and is of interest to the ‘Case Study Provider’ stakeholder.

- ‘Use model-based systems engineering’, that ensures that the whole project uses a model-based approach and is of interest to the project ‘Sponsor’ stakeholder.
- ‘Apply across whole life cycle’, that ensures that any guidelines produced can be applied across the system of systems life cycle.
- ‘Comply with best practice’, that ensures that all aspects of the guidelines are traceable back to best practice, whether this is formal standards, best-practice guidelines, publications, etc.

Two of the main use cases are broken down in to more detail on the following diagram.

- ‘Provide SoS requirements engineering process’
- Provide SoS management engineering process’

**Name: ... acknowledged**

Description: The Apply to different types of SoS use case must be able to be applied to ... acknowledged SoS.

An acknowledged SoS, taken from the DoD SoS Guide, is defined as:
'Acknowledged SoS have recognized objectives, a designated manager, and resources for the SoS; however, the constituent systems retain their independent ownership, objectives, funding, and development and sustainment approaches. Changes in the systems are based on collaboration between the SoS and the system.'

**Name: ... collaborative**

Description: The Apply to different types of SoS use case must be able to be applied to ... collaborative SoS.

A collaborative SoS, taken from the DoD SoS Guide, is defined as:
'In collaborative SoS the component systems interact more or less voluntarily to fulfill agreed upon central purposes. The Internet is a collaborative system. The Internet Engineering Task Force works out standards but has no power to enforce them. The central players collectively decide how to provide or deny service, thereby providing some means of enforcing and maintaining standards’

**Name: ... directed**

Description: The Apply to different types of SoS use case must be able to be applied to ... directed SoS.

A directed SoS, taken from the DoD SoS Guide, is defined as:
'Directed SoS are those in which the integrated system-of-systems is built and managed to fulfill specific purposes. It is centrally managed during long-term operation to continue to fulfill those purposes as well as any new ones the
system owners might wish to address. The component systems maintain an
ability to operate independently, but their normal operational mode is
subordinated to the central managed purpose.'

Name: ... guidelines
Description: The processes produced must **Comply with best practice** with
regards to **... guidelines**

Name: ... standards
Description: The processes produced must **Comply with best practice** with
regards to **... standards**

Name: ... virtual
Description: The **Apply to different types of SoS** use case must be able to be
applied to **... virtual** SoS.

A virtual SoS, taken from the **DoD SoS Guide**, is defined as:
'Virtual SoS lack a central management authority and a centrally agreed upon
purpose for the system-of-systems. Large-scale behavior emerges-and may be
desirable-but this type of SoS must rely upon relatively invisible mechanisms to
maintain it.'

Name: Apply across whole life cycle
Description: The **Apply across whole life cycle** constraint means that all the
processes produced as part of **Provide guidelines for SoS requirements** must be
able to be applied across the whole of the systems engineering life cycle.

Name: Apply to different types of SoS
Description: The **Apply to different types of SoS** use case describes the need for the
**System of Systems Requirements Processes** to be able to be applied to the
different types of SoS, as defined by the **DoD SoS Guide**, which are:

- directed
- acknowledged
- collaborative
- virtual

The level of operational control gradually decreases from the top of the list to the
bottom.

Name: Comply with best practice
Description: The **Comply with best practice** constraint means that all of the
processes produced by the **Provide guidelines for SoS requirements** must comply
with current best practice.

Name: Provide guidelines for SoS requirements
Description: The **Process owner context** diagram shows the main use cases from
the point of view of the **Process Owner** stakeholder.
**Name: Provide SoS requirements engineering process**
Description: The Provide SoS requirements engineering process use case covers the provision of the set of COMPASS - Process model that considers requirements engineering for system of systems.

**Name: Provide SoS requirements management process**
Description: The Provide SoS requirements management process use case covers the provision of the set of COMPASS - Process model that considers requirements management for system of systems.

**Name: Provide tool support**
Description: All processes must be implemented using automated tools

**Name: Use model-based systems engineering**
Description: The Use model-based systems engineering constraints means that all of the processes produced by Provide guidelines for SoS requirements must be developed using Model-based systems engineering techniques.

### 2.4.2. Requirements engineering process

The diagram here shows the detailed breakdown of the 'Provide SoS requirements engineering process' use case that was first seen in the process owner context.

The main use case has four included use cases:
- 'Identify SoS stakeholders', that requires that all stakeholder roles that are associated with the system of systems, as opposed to the constituent systems, are identified.
• ‘Understand SoS to constituent systems interactions’, that requires that all interaction between the system of systems its associated constituent systems are identified.

• ‘Define V&V criteria’, that requires that verification (it works) and validation (it does what it is supposed to do) criteria are defined. This has to be done in two ways: ‘...using semi-formal scenarios’ (for example sequence diagrams in SysML) and ‘... using formal scenarios’ (mathematically provable).

• ‘Understand SoS context’. Understanding the system of systems context is essential in order to gain a high-level understanding of the needs (goals, requirements and capabilities) of the system of systems. This use case is constrained by two use cases:
  o ‘Consider existing system first’, that ensures that pre-existing solutions are considered before bespoke and new-build solutions.
  o ‘Consider multiple-options’, that ensures that several candidate solution are considered, rather than just a single solution.

Included in both of these constraints is the use case 'Understand constituent system contexts’ that requires that the needs (goals, requirements and capabilities) for the constituent systems are understood.

Two of these use cases are concerned with understanding context and may, therefore, be thought of as types of a generic use case ‘Understand context’, that includes four lower-level use cases:

• 'Identify use case sources’, that requires that the needs that are used as a source for the use cases are identified. These source needs may be requirements, capabilities or goals.

• 'Identify stakeholders’, that requires that all stakeholder roles associated with the context are identified.

• 'Define use cases’, where the use case are identified and defined based on the source needs.

• 'Analyse use cases’, where the use cases are analysed in a number of ways, such as looking for conflicting use cases, complimentary use cases, common use cases, etc. This use case also includes the need to ‘Identify problems’ based on the analyses and then to ‘Resolve problems’ that may have been identified.

The processes that are defined as part of the COMPASS project must be able to satisfy all of these use cases for the system of systems requirements engineering

**Name: ... from capabilities**
Description: **Capabilities** (sp) must be able to be used as a basis for **Use Cases**

**Name: ... from goals**
Description: **Goals** must be able to be used as a basis for **Use Cases**
Name: ... from requirements
Description: Requirements must be able to be used as a basis for Use Cases.

Name: ... using formal scenarios
Description: The verification and validation may use formal scenarios.
In SysML parametric constraints may be used as a start point for using formal methods to provide the formal scenarios.

Name: ... using semi-formal scenarios
Description: The verification and validation may use semi-formal scenarios.
In SysML these scenarios may be realised using sequence diagrams.

Name: Analyse use cases
Description: Each Use Case must be fully understood in terms of its relationships to other Use Cases and Stakeholders.
A number of different analysis techniques may be used.

Name: Consider existing systems first
Description: When considering the constituent systems that make up the system of systems, it is important to consider the use of pre-existing constituent systems before bespoke constituent systems.

Name: Consider multiple options
Description: It is important to consider more than one possible solution for the system of systems. This may be configurations of different sets of constituent systems that go to make up a single solution. These multiple solution options may then be traded off as part of their assessment.

Name: Define use cases
Description: The Use Cases are identified and defined in the form of a context.
These Use Cases will be defined based on a number of Needs (such as Requirements, Goals or Capability) and the relationships between them defined.
Use Cases will also yield some observable results, either directly or indirectly (via other Use Cases) to one or more Stakeholder.
These Use Cases, Stakeholders and their inter-relationships will be captured as part of a context. In SysML this context will be visualised using a SysML use case diagram.

Name: Define V&V criteria
Description: All Use Cases must have verification and validation criteria defined for them, which can be used to validate the Source Elements (Requirements, Goals or Capability) (sp)).
When considering verification, the question is ‘are we building the system right?’ or ‘does it work?’

When considering validation, the question is ‘are we building the right system?’ or ‘does it do what it is supposed to do?’

**Name: Identify problems**
**Description:** Any problems between Use Cases must be identified. These problems may include:
- Conflicts
- Duplication
- Missing requirements
- New requirements
- Gaps between requirements
- Common requirements

This can be identified in a number of ways:
- Internal conflicts may be identified by considering the analysis relationships (‘include’, ‘extend’ and ‘constrain’) between the Use Cases in a specific context
- External conflicts may be identified by considering other contexts for each of the Stakeholders in the specific context. The related Use Cases from the specific and the external contexts may then be assessed for being: complimentary, conflicting, duplicating, etc
- Ensuring that all Use Cases are related, either directly or indirectly, to one or more Stakeholder
- Ensuring that all Stakeholders are directly related to at least one Use Case
- Ensuring that analysis relationships between Use Cases have been considered (‘include’, ‘extend’ and ‘constrain’)
- Ensuring that all constraint-type Use Cases are related to functional Use Cases using the ‘constrain’ relationship.

**Name: Identify SoS stakeholders**
**Description:** The Stakeholders for the specific System of Systems context must be identified.

These Stakeholders will be defined using a Stakeholder view.

In SysML, a Stakeholder View may be realised using a Block Definition diagram, where each block represents a Stakeholder

**Name: Identify stakeholders**
**Description:** The Stakeholders for the specific context must be identified.

These Stakeholders will be defined using a Stakeholder view.

In SysML, a Stakeholder View may be realised using a Block Definition diagram, where each block represents a Stakeholder
Name: Identify use case sources
Description: The Needs that are to be used as a basis for the Use Cases are identified.

The types of Need may include: Requirements, Goals or Capabilitys

Name: Provide SoS requirements engineering process
Description: The Provide SoS requirements engineering process use case covers the provision of the set of COMPASS - Process model that considers requirements engineering for system of systems.

Name: Resolve problems
Description: Any problems that have been identified as a result of the Identify gaps and Identify problems Use Cases must be addressed.

These problems may be resolved by further modelling, may require further information or may require engagement with the customers

Name: Understand constituent system contexts
Description: Identify and define the context for the constituent system.
This is a specialisation of the generic Understand context

Name: Understand context
Description: The Understand context use case is a generic use case that may be applied at any level of abstraction.

Name: Understand SoS context
Description: Identify and define the context for the system of systems.
This is a specialisation of the generic Understand context

Name: Understand SoS to constituent system interactions
Description: The interactions between the SoS and its constituent systems must be identified and defined. This is crucial for the overall understanding on the SoS. This can be achieved in a number of ways, including:
- Understanding the contexts of both the SoS and its CS
- Understanding a number of scenarios that describe the behaviour across the interfaces of both the SoS and its CS
- Analysing the two of these
2.4.3. Requirements management process

The diagram here shows the detailed breakdown of the ‘Provide SoS requirements management process’ that was first introduced in the process owner context.

The main use case has five included use cases, which are:

- ‘Control process artefacts’, that requires that all the artefacts that are produced or consumed as part of the system of systems requirements process (both requirements engineering and requirements management) are identified, managed and controlled. This includes: ‘Configure process artefacts’ to ensure that artefacts can be held under configuration management, ‘Obtain consensus’ that ensures that all relevant stakeholders agree to the requirements artefacts, and ‘Obtain commitment’ that ensures that all stakeholders commit the time and resources that are required to realise the system of systems.

- ‘Manage requirements change’, that requires that any changes to requirements are identified and managed, whether these changes are ‘... for constituent systems’ or ‘... for system of systems’. This requirements change management includes the following use cases:
  - 'Monitor changes to constituent systems’, which ensures that the constituent systems, as well as the system of systems is considered.
  - 'Identify changes to requirements’, that requires that all requirements changes are looked for and identified.
o ‘Evaluate changes’, where the impact of the change on the model is assessed.
o ‘Take action’, where the results of the assessment are considered and appropriate action is decided upon.

- ‘Communicate with stakeholders’, it is essential that all relevant stakeholders are provided with information that is both appropriate for their role and timely.
- ‘Understand constituent systems RM processes’, in order to interact with the constituent systems, in terms of monitoring and identifying requirement, it is essential that their requirements management processes are understood. Once this understanding has been established, it is then possible to ensure that both the constituent systems processes and the system of systems processes can work together.
- ‘Provide traceability’, that requires that the model contains all appropriate traceability paths and mechanism to describe them.

The processes that are defined as part of the COMPASS project must be able to satisfy all of these use cases for the system of systems requirements engineering.

**Name: ... for constituent systems**
Description: Requirement change must be managed for the constituent systems that make up the system of systems.

**Name: ... for system of systems**
Description: Requirement change must be managed for the system of systems that comprises the constituent systems.

**Name: Communicate with stakeholders**
Description: Maintain communication with all relevant stakeholders

**Name: Configure process artefacts**
Description: All artefacts from the COMPASS - Process model must be held under configuration management and version control.

**Name: Control process artefacts**
Description: All artefacts in the COMPASS - Process model must be owned, managed and configured.

**Name: Evaluate changes**
Description: Any changes to the requirement must be evaluated.

This will entail performing some of initial impact analysis on the model to see which elements of the model will be impacted.

Once these elements have been identified, then the level of impact must also be assessed.
Name: **Identify changes to requirements**  
Description: All changes to requirements must be identified.

Name: **Manage requirements change**  
Description: In real projects, it is unrealistic to assume that requirements may be frozen and that they will not change. When requirements do change, then this change must be identified and managed.

Name: **Monitor changes to constituent systems**  
Description: Any changes to the constituent systems that make up the system of systems must be identified.

Name: **Obtain commitment**  
Description: Commitment must be obtained from all relevant stakeholder. This will require these stakeholders to buy-in to the approach that is being taken and commit appropriate resources to the work activities.

Name: **Obtain consensus**  
Description: Agreement must be reached on all artefacts from the COMPASS - Process model from all relevant stakeholders. This will require some kind of review, whether it is a formal review meeting with minutes or an informal agreement between stakeholders.

Name: **Provide SoS requirements management process**  
Description: The Provide SoS requirements management process use case covers the provision of the set of COMPASS - Process model that considers requirements management for system of systems.

Name: **Provide traceability**  
Description: Ensure that the model contains all appropriate traceability paths and mechanism to describe them.

Name: **Take action**  
Description: Any changes to requirements will potentially result in some action that needs to be taken to ensure the continued integrity of the model and, hence the CS or SoS. These actions must be identified, defined and carried out.

Name: **Understand constituent systems RM processes**  
Description: In order to control the use of CS in the CS, it is important to understand the requirements processes that are used by the CS. This enables certain interaction points in the CS processes that can then be used to monitor the CS requirements set.
2.5. Traceability View - Use Cases to Needs

2.5.1. 'Analyse use case' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.2. 'Be driven by best practice' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.3. 'Control process artefacts' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.4. 'Define V&V criteria' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.5. 'Demonstrate approach' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.6. 'Engage with stakeholders' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.

2.5.7. 'Generate process model' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.8. 'Identify use case sources' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.9. ‘Manage requirements change’ to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.10. 'Project Context' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.11. 'Provide approach for SoS' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.12. 'Provide modelling language' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.13. 'Provide process model...' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.14. 'Provide SoS RE process' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.15. 'Provide SoS requirement process' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.16. 'Provide SoS RM process' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.17. 'Provide support' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.18. 'Provide tools' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.19. 'Understand context' to requirements

This diagram shows a Traceability View for traceability between Use Cases and their associated Needs.
2.5.20. 'Understand SoS context' to requirements

This diagram shows a **Traceability View** for traceability between **Use Cases** and their associated **Needs**.
3. Process Structure View

3.1. COMPASS - Ontology

The diagram here covers all of the concepts pertinent to model-based requirements engineering and is used in the definition of the requirements engineering views.

The diagram shows that there is the concept of a **Need** that has three sub-types: **Goal**, **Requirement** and **Capability**. One or more **Need** is elicited from one or more **Source Element**. One or more **Rule** constrains one or more **Need**.

One or more **Use Case** describes the context of each **Need** via a **Context**. There are two types of context shown here: the **System Context** and the ‘Stakeholder Context’, although this list is incomplete. One or more **System Context** represents the need for a ‘System’.

One or more **Scenario** validates one or more **Use Case** and there are two types of **Scenario** - the **Semi-formal Scenario** and the **Formal Scenario**.

**Name: Acknowledged**

**Description:** ‘**Acknowledged System of Systems**’ have recognized objectives, a designated manager, and resources for the **System of Systems**; however, the
Constituent Systems retain their independent ownership, objectives, funding, as well as development and sustainment approaches. Changes in the systems are based on collaboration between the System of Systems and the system.’

Name: Architectural Framework
Description: An Architectural Framework will specify a number of Views and Viewpoints that it is deemed necessary to produce in order to meet the requirements of that framework.

Name: Architecture
Description: An Architecture represents the fundamental organisation of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution over time. An architecture may be modelled from any number of perspectives, or ‘views’. It is important to be able to understand what viewpoint is being taken when any modelling is being performed as the models for different views can vary dramatically.

Name: Capability
Description: A Capability describes the ability to do something in order to deliver stated Goals. A Capability is often realised by a set of Processes.

Name: Collaborative
Description: ‘In Collaborative System of Systems, the component systems interact more or less voluntarily to fulfil agreed-upon central purposes. The Internet is a Collaborative system. The Internet Engineering Task Force works out standards but has no power to enforce them. The central players collectively decide how to provide or deny service, thereby providing some means of enforcing and maintaining standards.’

Name: Constituent System
Description: A Constituent System is a System in its own right that forms part of the overall System of Systems.

Name: Context
Description: The idea of the Context is fundamental to the requirements framework described in this report. In its simplest form, a Context may be thought of as a ‘point of view’ on the system under development. It is possible to view the Needs of a system from any number of different points of view (contexts), so it is essential that it is well understood from what point of view each Context is taken.

Name: Directed
Description: ‘Directed System of Systems are those in which the integrated System of Systems is built and managed to fulfil specific purposes. It is centrally managed during long-term operation to continue to fulfil those purposes as well as any new ones the system owners might wish to address. The component
systems maintain an ability to operate independently, but their normal operational mode is subordinated to the central managed purpose.'

Name: Formal Scenario  
Description: Formal Scenarios are realised in SysML through parametric constraints and their usages, allowing a more mathematical-based approach to be taken for understanding the Use Cases. The parametric usages are connected together into different networks that allow ‘what if’ analysis and are particularly powerful when considering trade-offs.

Formal Scenarios can be given a semantics using formal methods such as VDM or CSP. In COMPASS we use the Unifying Theories of Programming (UTP) as the underlying formalism in which to express both VDM and CSP. As a consequence, we can model state-based properties in VDM, and behavioural and communication properties in CSP. The formal notations offer some specific advantages for ‘what if’ analysis. Firstly, they are executable so that they can be animated and used in simulations. Secondly, they are verifiable. Using proof, model-checking or testing one can check for correctness criteria in the model and by means of model-based testing also check for correctness criteria in corresponding implementations.

Name: Goal  
Description: A Goal, which defines and describes a desired outcome of a System or System of Systems.

Name: Need  
Description: A Need describes something that can be given meaning by a Use Case.

A good example of this is a Requirement, where a Use Case would be defined as a Requirement that has been put into context.

Name: Ontology  
Description: An Ontology provides a visualisation of all the key concepts, the terminology used to describe them and the inter-relationships between said concepts. The Ontology, however, is much more than just a data dictionary and plays a pivotal role in the definition and use of any rigorous framework.

Name: Requirement  
Description: A Requirement is defined as: ‘... the definition of a property of a system that is either wanted or needed by a stakeholder’

A Requirement is a type of Need.

Name: Rule  
Description: When describing any Need in natural language there is a lot of room for ambiguity and misinterpretation. In order to minimise these problems, the
Rule concept allows the definition of a number of rules that are applied to Need descriptions.

These rules may apply to the Need itself or, more usually, to the properties of a Need. Sometimes these Rules will apply to the way that the wording in a Need description must be applied. Other rules concern the complexity of the text description that is being used to describe the Need.

Name: Scenario
Description: A Scenario is defined as an exploration of a ‘what if’ for a Use Case. Each Use Case will give rise to a number of different situations that may arise when it is being satisfied.

Name: Semi-formal Scenario
Description: Semi-formal Scenarios are realised by a semi-formal notation such as SysML, making use of, for example, sequence diagrams that show interactions between elements in the System. They can also be described informally using text as a set of scenario steps. Often the two are combined. Scenarios will normally be created for each type of context that has been developed. For example, if Stakeholder Context and System Contexts have been developed then both Stakeholder-level Scenarios and System-level Scenarios would be created. The Stakeholder-level scenarios typically treat the system as a black box and analyse the interactions between the Stakeholders and the System. The System-level Scenarios would explore the interactions between system elements within the System.

Name: Source Element
Description: A Source Element describes the type of anything that can be used as the source for a Need. For example, a Book may be used as a source for a Requirement

Name: Stakeholder Context
Description: The Stakeholder Context is a set of points of view that is defined by looking at a set of Needs from the point of view of different Stakeholders, the roles of any person or thing that has an interest in or is affected by a System

Name: System
Description: A System is defined as an interacting combination of elements that work together to achieve a set of Goals and satisfy a set of Needs.

Name: System Context
Description: The System Context is a set of points of view that is based on the level of hierarchy of a System that may itself be broken down further into, for example, subsystems, assemblies and components. When considering such a hierarchy, it is usual to have a number of different types of Needs defined that exist at the various levels in the hierarchy: system requirements for a system, sub-system requirements derived from the system requirements for each sub-system and so on. Each hierarchical level will have one or more Contexts
associated with it that consider the relevant requirements from that point of view, trace back to requirements at the higher level and establish the meaning of the requirements in that context.

**Name: System of Systems**  
Description: A **System of Systems** is a collection of **Constituent System** that pool their resources and capabilities together to create a new, more complex **System** which offers more functionality and performance than simply the sum of the **Constituent System**.

**Name: Use Case**  
Description: A **Use Case** represents something that has been given meaning by being put into context. A good example of this is a **Requirement**, where a **Use Case** would be defined as a **Requirement** that has been put into context.  
This can also apply to a number of **Needs**, such as **Goals** and **Capability**s.

**Name: View**  
Description: A **View** represents a single image of a small part of the overall **Architecture**. Each **Viewpoint** will be made up of one or more **View**.

**Name: Viewpoint**  
Description: A **Viewpoint**, at its simplest, is a collection of **Views** that has some sort of common connection.

**Name: Virtual**  
Description: ‘**Virtual System of Systems** lack a central management authority and a centrally agreed-upon purpose for the **System of Systems**. Large-scale behaviour emerges-and may be desirable-but this type of **System of Systems** must rely upon relatively invisible mechanisms to maintain it.’
3.2. Process concepts

This diagram shows the process structure view for the concepts that relate to process. This is an extension to the core concepts ontology.

**Name: Activity**
Description: An Activity forms part of a Process and represents something that must be done to realise that Process.

An Activity produces and consumes Artefacts and has a Stakeholder that is responsible for its execution. An Activity also uses one or more Resource.

**Name: Artefact**
Description: An Artefact represents something that is produced or consumed by an Activity.

**Name: Competency Profile**
Description: A Competency Profile shows the actual abilities that are possessed by a specific Person. The Competency Profile may be generated at the output of a competency assessment exercise that uses a Competency Scope as its input.

**Name: Competency Scope**
Description: A Competency Scope defines the abilities that are required for a specific Stakeholder.
Name: Gate
Description: A Gate is a special type of review that must be executed before any one Stage may be exited.

A Gate assesses the execution of a Stage.

Name: Life Cycle
Description: The Life Cycle describes the evolution of a System over time. A single System may have any number of Life Cycles associated with it, depending on the context of the system. For example:

- A product Life Cycle
- A project Life Cycle
- An acquisition Life Cycle
- An operational Life Cycle, etc.

These Life Cycles interact with one another via Life Cycle Interaction Points.

Any Life Cycle is made up of one or more Stage.

Name: Life Cycle Interaction Point
Description: A Life Cycle Interaction Point defines a specific point at which one, more than one Life Cycle interacts with another.

Name: Person
Description: A Person is an individual human being. Each Person takes on a number of Stakeholders. Each Person has a Competency Profile associated with it that defines the actual ability of that Person.

A Person is also a type of Resource.

Name: Process
Description: A Process describes an approach to achieving an end. A Process is made up of one or more Activity, one or more Artefact and one or more Stakeholder.

Name: Process Execution Group
Description: A Process Execution Group represents a distinct set of Processes that are executed for a particular reason. These Process Execution Groups may be defined based on function (so there may be a 'component X' Process Execution Group), or by working area (so there may be a 'software' Process Execution Group), amongst others.

Name: Resource
Description: A Resource is anything that is used by an Activity within a Process. Types of Resource include: a Person, a room, etc.
Name: Stage
Description: A Stage represents a discrete time period that describes a specific phase of a Life Cycle. These Stages are typically defined by the context in which the Life Cycle is being used. Before a Stage can be exited for any reason, it must pass through a Gate. A number of Process Execution Group may be executed during each Stage.

Name: Stakeholder
Description: A Stakeholder represents the role of any person, organisation or thing that has an interest in the system of project.

Name: System
Description: A System is defined as an interacting combination of elements that work together to achieve a set of Goals and satisfy a set of Needs.

3.3. Requirement types

This diagram shows the process structure view for the need types. This is an extension to the core concepts ontology.
**Name: Business Requirement**
Description: A **Business Requirement** is used to state the needs or capabilities of a business or organisation. This will include business drivers that impact the entire organisation and all the projects within it. These requirements will be, by necessity, described at a very high level and may, indeed, often be described as **Goals** or one or more **Capability** rather than **Requirements**.

**Name: Capability**
Description: A **Capability** describes the ability to do something in order to deliver stated **Goals**. A **Capability** is often realised by a set of **Processes**.

**Name: Functional Requirement**
Description: A **Functional Requirement** is used to state an aspect of the behaviour of the system, often describing some sort of observable result to stakeholders that are using the system. By their very definition, functional requirements ‘do’ something and result in some sort of function being performed. Functional requirements are usually what are referred to when people misuse the term ‘user requirements’.

**Name: Goal**
Description: A **Goal**, which defines and describes a desired outcome of a **System** or **System of Systems**.

**Name: Need**
Description: A **Need** describes something that can be given meaning by a **Use Case**.

A good example of this is a **Requirement**, where a **Use Case** would be defined as a **Requirement** that has been put into context.

**Name: Non-functional Requirement**
Description: A **Non-functional Requirement** will constrain the way that a **Functional Requirement** may be realised. It should be noted that the term ‘constraint’ is often used rather than **Non-functional Requirement**. This is deliberately not used in this report as the formal term on the ontology because the term ‘constraint’ is one of the key constructs in the SysML notation. To avoid unnecessary confusion the term **Non-functional Requirement** is used.

It is essential to identify and understand the **Non-functional Requirements** that exist in a system. **Non-functional Requirements** are sometimes treated as secondary requirements that are not as important as **Functional Requirements**. Although this may be the case in some instances, overall it is the satisfaction of these **Non-functional Requirements** that will decide whether the project is successful or not.

**Name: Requirement**
Description: A **Requirement** is defined as: ‘... the definition of a property of a system that is either wanted or needed by a stakeholder’
A Requirement is a type of Need.

### 3.4. Source Element Types

This diagram shows the process structure view for the **Source Element Types**. This is an extension to the core concepts ontology.

**Name: Book**
Description: A Book refers to any published work that has an ISBN number. For example: Pragmatic Guide to Business Process Modelling, Model-based Requirements Engineering, etc.

**Name: Paper**
Description: A Paper refers to any academic article that has been published in a journal or at a conference. For example: Requirements Engineering for SoS.

**Name: Presentation**
Description: A Presentation refers to anything that has been published or presented at a conference or any other form of meeting.

**Name: Project Document**
Description: A Project Document refers to any deliverable or input to the COMPASS Project, for example, the COMPASS Project Description of Work.
**Name: Publication**
Description: A **Publication** refers to anything that has been published and is in the public domain.

**Name: Source Element**
Description: A **Source Element** describes the type of anything that can be used as the source for a **Need**.
For example, a **Book** may be used as a source for a **Requirement**

**Name: Standard**
Description: A **Standard** represents any process, technique or approach that has been agreed upon by a number of **Stakeholders**. Typically, these **Standards** will be national and international standards, but may also include best-practice guidelines, proprietary processes, etc. Examples of this include: **ISO 15288**, **DoD System of Systems Guide**, etc.
4. Process Content View

The process content view shows the content of each process, in terms of one or more **Activity** and one or more **Artefact** by representing each **Process** as a single block. The process content view may be thought of as the library of **Processes** that is available for use on the project.

The process content view is realised in SysML by a block diagram, and is very closely related to the process structure view in that it is the process content view that shows the actual activities and artefacts exhibited by each **Process**. Each **Process** has a block to represent it and the process **Artefacts** are represented by properties, whereas each process **Activity** represented by an operation.

**Name: Context Process**

Description: The main aim of the **Context Process** is to define a context based on the **Context Definition View**. This process is a generic one that may be invoked from the **SoS Requirements Engineering** process and may be applied at both the **System of Systems** and the **Constituent Systems** level.

**Name: CS Process Analysis**

Description: The overall aim of the **CS Process Analysis** is to understand the requirement management process of the constituent systems that make up the system of systems.

It is important to monitor the requirements of the constituents so that any changes can be identified and evaluated. In order to do this there needs to be an
understanding of the requirement management process of each of the constituent systems. This will be achieved by modelling each the requirement management process and then mapping to the SOS requirement management process. Once this understanding has been achieved and mapped to the SOS the requirement management process, then a number of control points can be set up that allow requirements changes to be identified periodically.

In the event that the requirement management process of the system of systems and its constituent systems are not compatible, then an exception is raised.

**Name: Requirement Control Process**

Description: The overall aim of the Requirement Control Process process is:

- To ensure that all information contained in the Requirement Model is communicated to the relevant stakeholders
- To ensure that the requirements model is reviewed and that a consensus is achieved between the relevant stakeholders.
- To obtain commitment from the stakeholders that the consensus is the most appropriate way forward and to allocate suitable resources to ensure that the requirements are satisfied.

**Name: Requirements Change Process**

Description: The main aim of the Requirements Change Process is to identify any changes to Requirements, assess the impact and take appropriate actions. This process may be applied at both the System of Systems and the Constituent Systems level and can actually invoke another instance of itself.

**Name: Requirements Monitor Process**

Description: The aim of the Requirements Monitor Process is two-fold:

- To allow requirements from the constituent systems that make up the system of systems to be monitored for change via Control Points.
- To allow requirements form the system of systems to be monitored

Should any change occur in either the system of systems or any of its constituent systems, then the Requirements Change Process will be invoked.

**Name: SoS Requirements Development**

Description: The main aim of the SoS Requirements Development process is to perform most of the requirements engineering at the system of systems level. This involves defining the contexts at system of systems and constituent systems level and identifying the relationships and interactions between them.

This process calls up both the Context Process (at both system of systems and constituent systems levels) and the Verification and Validation Definition Process.

**Name: System of Systems Requirement Process**
Description: This is the high-level abstract Process that defines all the Processes related to systems of systems requirements that are generated as part of the COMPASS Project.

**Name: System of Systems Requirements Engineering Process**
Description: This is the high-level abstract Process that defines all the Processes related to systems of systems requirements engineering that are generated as part of the COMPASS Project.

**Name: System of Systems Requirements Management Process**
Description: This is the high-level abstract Process that defines all the Processes related to systems of systems requirements management that are generated as part of the COMPASS Project.

**Name: Traceability Process**
Description: The overall aim of the Traceability Process is to enable traceability to be set up between any elements in the model. This may be used for the RM but may also trace to any elements on the wider SoS model or its CS models.

**Name: Verification and Validation Definition Process**
Description: The main aim of the Verification and Validation Definition Process is to define a number of scenarios for each use case in a specific context. These scenarios may be either semi-formal (diagram-based) or formal (mathematical-based) and form the basis of the testing of the System of Systems. These scenarios are defined for both verification (it works) and validation (it does what it is supposed to do) for the use cases.
5. Stakeholder View

This diagram is the Stakeholder Context Definition View for the COMPASS Project.

The diagram identifies the stakeholders for the project in the form of blocks.

**Name: Case Study Provider**
Description: The role of the projects partners who will be applying the system of systems requirements processes on case study projects

**Name: Process Automator**
Description: The role of the person who is interested in automating the final process output from the process model.

**Name: Process Modeller**
Description: The role of anyone who is carrying modelling as part of the COMPASS Project.

**Name: Process Owner**
Description: The role of the people who will be the end users of the systems of System of Systems Requirement Processes that are produced as part of the COMPASS Project. This forms the main context for the process definition work on the project.

**Name: Requirement Engineer**
Description: The role of the people who are responsible for aspects of requirements engineering in the System of Systems Requirement Processes.
**Name: Requirement Manager**
Description: The role of the people who are responsible for aspects of requirements management in the system of [System of Systems Requirement Processes](#). It should be noted that this will involve both [System of Systems Requirements Engineering Processes](#) and [System of Systems Requirements Management Processes](#).

**Name: Researcher**
Description: The role of the people who are carrying out research as part of the [COMPASS Project](#).

**Name: Reviewer**
Description: The role of any people who are responsible for the reviewing activities in the system of [System of Systems Requirement Processes](#).
6. Information View

This is the IV - COMPASS Requirements Processes diagram. The information view (IV) identifies the main artefacts and defines their structures and the relationships between.

This information view focusses on the artefacts for the requirements processes in the COMPASS - Process model.

Name: Change Record
Description: The Change Record describes changes that have been made to a Requirement Element as a result of a Change Request.

Name: Change Request
Description: The Change Request is a result of the Requirements Monitor Process and represents the need for a change to be made to a Requirement Element in the Requirement Model. The Change Request starts off the Requirements Change Process, and the results of the Change Request are recorded in the Change Record.

Name: Context Definition View
Description: This view is one of the basic ACRE views. The Context Definition View identifies the points of view that are explored in the Requirement Context View. These points of view, or contexts, may take many forms including stakeholders and levels of hierarchy in a system.
Name: Context Interaction View
Description: The Context Interaction View is intended to provide an overview of the relationships between the contexts of the various Constituent Systems that make up a System of Systems.

Name: Control Point
Description: A Control Point represents the part of the CS Process Model where it is possible to monitor the Process. This is used by the Requirements Monitor Process to detect and changes in the Requirements in the Constituent System that make up the System of Systems.

Name: CS Process Model
Description: The CS Process Model represents a Process Model that abstracts a Constituent System that makes up the System of Systems.

Name: Definition Rule Set View
Description: This view is one of the basic ACRE views. The Definition Rule Set View contains and rules that may have to be applied to each requirement definition. For example, these may be complexity rules in the form of equations of more general text-based rules.

When defining individual requirement descriptions it is often desirable to put constraints on either the values or on measurable aspects of the values. These constraints are enforced by applying one or more rule to the requirement description values. These rules must be defined somewhere and the relevant attributes of the requirement descriptions must be identified. The purpose of this view is to capture this information.

Name: Exception
Description: An Exception is something that happens that is out-of-the-ordinary and results a Process no longer being able to continue.

Name: Process Model
Description: The Process Model abstracts one or more Source Process in the form of a model. For the COMPASS project, all Process Models will be compliant with the 'seven views' approach to process modelling.

Name: Requirement Context View
Description: This view is one of the basic ACRE views.

The Requirement Context View takes the requirements and gives them meaning by looking at them from a specific point of view. This is known as putting the requirements into context and forms the basis of the approach presented in D21.1.

The requirements have only been described so far by defining a number of requirements descriptions in the requirement description view. This is all well and good and an essential part of any requirements exercise, but this is by no
means complete. The problem arises that these requirement descriptions may be interpreted in different ways depending on the viewpoint of the reader of the requirement description. It is essential then that each requirement is looked at from different points of view, or in different contexts. It will also be found that different contexts are concerned with different sets of requirements, all of which will be related together in some way. When a requirement is put into context it is known as a ‘use case’ and by considering these uses case and the relationships between them and other use cases or stakeholders, it is possible to generate a complete point of view, or context.

These contexts may be based on a number of elements, such as stakeholders or levels of hierarchy in a system.

**Name: Requirement Description View**
**Description:** This view is one of the basic ACRE views.

The **Requirement Context View** contains structured descriptions of each requirement. These requirements are considered individually and will usually have a number of attributes, or features, associated with each one.

The main purpose of this view is to describe each individual requirement according to a pre-defined set of attributes. These attributes will vary depending on the process that is being followed, the industry that the work is being carried out in, any standards or best-practice models that may be being used and any other number of factors.

This view is primarily used for managing the requirements of a system and is often the basis of implementation for many of the commercial requirements management tools that are on the market today.

Each requirement description provides a non-contextual description of the requirement. When a requirement is put into context, it is known as a ‘use case’ and, hence, there is a very strong relationship between the requirement descriptions and the use cases from the **Requirement Context Views**.

**Name: Requirement Element**
**Description:** A RE forms part of a **Requirement View** that forms part of the **Requirement Model**. Examples of **Requirement Element** include: **Needs**, **Source Elements**, **Scenarios**, etc.

A change to a **Requirement Element** is indicated by raising a **Change Request** and the results of this change are recorded in a **Change Record**.

**Name: Requirement Model**
**Description:** A **Requirement Model** is a model that represents **Requirements** and that is developed and maintained by the **Process Model**. The **Requirement Model** is made up of one or more **Requirement View**, each of which is made up of one or more **Requirement Element**.
**Name: Requirement View**
Description: A Requirement View represents part of a Requirement Model that is viewed from a specific perspective. Each Requirement View is made up of one or more Requirement Element.

The Requirement Views that are defined as part of the COMPASS Project are: Definition Rule Set View, Source Element View, Context Definition View, Requirement Context View, Requirement Description View, Validation View, Validation Interaction View, Context Interaction View, Test Coverage View

**Name: Review Record**
Description: A Review Record captures the result of a review activity that is carried out on part of the Requirement Model as part of one of the System of Systems Requirement Processes.

**Name: SoS Process Model**
Description: The SoS Process Model represents the Process Model that abstracts the System of Systems that comprises the Constituent Systems.

**Name: Source Element View**
Description: This view is one of the basic ACRE views. The Source Element View contains all relevant source information that is required to get the requirements right. It is essential that the origin of all requirements is known and this is what this view allows us to define. This view is used primarily as a mechanism to establish traceability in the system and provide links between the requirements and any other aspect of the system

**Name: Source Process**
Description: The Source Process represents any Process that is used as part of the COMPASS Project.

**Name: Test Coverage View**
Description: A Test Coverage View shows how much of the Requirement Model is covered by the Validation Views. This view is used to assess how well all the original Needs can be covered by testing based on the Validation Views.

**Name: Traceability View**
Description: Having traceability as an inherent part of the Requirement Model helps to ensure that the traceability can be established correctly and, more importantly, accessed easily and automatically. The COMPASS Project defines a model-based approach to requirements engineering that is designed to address, any Requirement View or Requirement Element appearing on a view can, in theory, be traced one to another. This is shown in the Traceability Views.

**Name: Validation Interaction View**
Description: The Validation Interaction View is intended to provide a combined view of the Scenarios for Use Cases that are involved in the System of Systems.
Name: Validation View
Description: This view is one of the basic ACRE views.

The Validation Views provide the basis for demonstrating that the requirements can be met or complied with in some way. These views can be informal (such as scenarios at various levels of abstraction) or may be formal (such as mathematical-based representation).
7. Process Behaviour View

7.1. PBV - Context Process

The Context Process begins with the Requirement Engineer who identifies a set of needs based on the Source Element View. A set of Requirements is now identified based on the Requirement Description View and then a context is selected.

An initial Requirement Context View is now produced and then it is analysed and any problem that are identified as a result of the analyse are now resolved.

The Reviewer carries out a review and a Review Record is produced. If the outcome of the review is positive, then the Verification and Validation Definition
Process is involved and the resultant Validation View is reviewed and a Review Record is produced. Following a positive outcome of the Validation View, all process artefacts are baselined by the Requirement Manager.

If the outcome of the review is not positive, then the process reverts back to defining the context.

If the outcome of the validation review is not positive, then the process reverts back to defining the validation, and the Verification and Validation Definition Process is re-invoked.

Name: analyse use case
Description: When developing use case diagrams it is important that the Use Cases are understood and well-defined, so it is important to analyse them. There are a number of common patterns that should be looked for as an aid towards the production of good use case diagram:

- **Use Case** at too high a level. One common mistake is to model Use Cases at too high a level - in indication of this is where a Use Case is linked to all external Stakeholder, represented on a use case diagram as ‘actors’. Such a pattern may indicate that the Use Case is at too high a level and that it should be decomposed further, making use of the «include» and «extend» dependencies to link it to more detailed Use Cases. The actors would then be associated with the more detailed Use Cases rather than all being connected to the top-level Use Case.

- Actor at too high a level - an indication of this is where an actor is present that is connected to every Use Case. Such a pattern may indicate that the actor is at too high a level and that it should be decomposed further, or that the diagram has been drawn from the point of view of that actor. If the actor is at too high a level, then it should be decomposed further and replaced on the diagram with the new actors. These actors will then be associated with the relevant Use Cases rather than being associated with all the Use Cases. If the diagram has been drawn from the point of view of that actor, i.e. the use case diagram is drawn for that actor's Context, then the actor should be removed from the diagram. The system boundary should indicate that the diagram is drawn for that actor’s Context.

- Repeated actors. Sometimes a pattern is seen in which two or more actors are connected to the same Use Cases, for example there are two stakeholders that are both connected to the same three use cases. This pattern may indicate that the actors are representing the same stakeholder. Alternatively, it may indicate that instances of stakeholders have been used (check for names of specific people, organisations, standards etc.). Instances should never be used. Remember that a stakeholder represents the role of something that has an interest in the project, not an actual instance involved. Any duplicate actors should be removed from the diagram.

- Something missing. This may be indicated by use cases without actors and actors without use cases. In the case where we have Use Cases that
are not related to anything, there are a number of possible reasons: the Use Case is not needed and should be removed from the diagram, there is an actor (or actors) missing that should be added to the diagram and linked to the Use Case, there is an internal relationship missing; the Use Case should be linked to another Use Case, or there is an external relationship missing; the Use Case should be linked to an existing actor. In the case where an actor is not connected to anything, there are a number of possible reasons: the actor is not needed and should be removed from the diagram, there is a Use Case (or Use Cases) missing that should be added to the diagram and linked to the actor, or there is a relationship missing; the actor should be linked to an existing Use Case.

The suggestions here can be used as part of the analysis of the Use Case.

**Name: baseline**
Description: All process artefacts:
- are recorded
- are held under configuration control
- form part of an overall baseline
- are stored on the project repository

**Name: define context**
Description: The idea of the Context is fundamental to the approach taken in here and, hence, it is very important that the concept if well understood. In its simplest form, a Context may be thought of as a ‘point of view’. It is essential, however, that it is well understood from what point of view each context is taken. It is possible to view the Needs of a system from any number of different points of view, so it is essential that the origins for these points of view are well-defined.

There are many types of context that exist, including:
- **Stakeholder Context**. The stakeholder context is a set of point of views that is defined by looking at a set of requirements from the point of view of different stakeholders. Key to getting this right is being able to identify what the various stakeholders are.
- **System Context**. The second type of context that will be considered is a set of contexts based on a system or, more specifically, the level of hierarchy of a system. This type of context is particularly relevant where we are considering System of Systems as there will be Contexts for both the System of Systems and the Constituent Systems.

The context will be based on the Context Definition View from the previous activity.

This activity will produce a context that:
- Identifies Use Cases based on the Needs and the Constraint Definition View
- Identifies the system boundary
- Identifies stakeholders that are external to this system boundary
- Identifies relationships between the Use Cases and the external stakeholders
- Identifies relationships between Use Cases

This context will take the form of a SysML use case diagram

**Name: define validation**
Description: Once the Requirement Context Views have been created and successfully reviewed, then the validation criteria need to be defined. This is achieved by executing the Verification and Validation Definition Process and then receiving back the appropriate Validation Views for the Requirement Context View.

**Name: elicit requirements**
Description: This activity takes the Source Elements from the Source Element View and then elicits a number of Needs (Requirement, Capability or Goal) from them.

Each Need is described in terms of:
- **UID** - a unique identifier that allows the Need to be located at any point in the project and may also be used for traceability purposes.
- **Name** - a short name that encapsulated the Need at a very high level.
- **Description** - a text description that may be taken from the original Source Element or created from new

These Needs form part of the Requirement Description View and may be structured hierarchically

**Name: identify needs**
Description: The aim of this activity is to identify a set of Source Elements that will be as a basis for the Needs. The Source Elements are identified based on their relevance to the context. The Source Element View is used as an input here as the pool of Source Element that are available.

**Name: resolve problems**
Description: Based on the previous analysis activity, a number or problems may have been identified.
The problems will include, but are not limited to:
- Conflicting use cases
- Repeated use cases
- New use cases identified
- New actors identified
- Use cases at the wrong level of abstraction
The problems are resolved here by updating the model
Name: review context
Description: A context review is carried out. This will involve looking at the Requirement Context View that have been generated and performing a number of checks:
- Syntax checks. These should be carried out by the modellers prior to the review and the results presented at the review. Ideally, these syntax checks should be performed automatically by the tool.
- Assessing complexity of Requirement Context Views. Making sure that the use case diagrams are simple enough to be easily understandable, yet complex enough to convey the context.
- Assess problem resolution. Ensure that any problems that were identified as a result of the analysis have been addressed.
- The outcome of this review will result in one of two possible ways forward:
  - In the event that the review is successful, then the define validation activity is carried out next.
  - In the event that the review is unsuccessful, then the process reverts back to the define context activity.

This activity will result in the production of a Review Record.

Name: review validation
Description: A validation review is carried out. This involves looking at the Validation View that have been returned form the Verification and Validation Definition Process and ensuring that they satisfy the use case in the Requirement Context View.
There are two possible outcomes from this review:
- In the event that the review is successful, then the process ends with the baseline activity.
- In the event that the review is unsuccessful, then the process reverts back to the define validation activity.

This activity results in the production of a Review Record.

Name: select context definition
Description: This activity takes in the Context Definition View and, based on the elements in the views, identifies the source of the context.

The context represents the needs of the system from a particular point of view (in the form of Use Cases) and, therefore, must have a context source defined.
7.2. PBV - CS Process Analysis

The process begins with the Process Modeller who identifies a Source Process and then model it, producing a CS Process Model. This CS Process Model is then mapped onto the SoS Process Model. This mapping is now evaluated to see whether or not the processes are compatible and, if so, where they may be able...
to interact. In the event that the two process models are not compatible in any way then an Exception is raised and the process is terminated.

In the event that the process models are compatible, then one or more Control Point is set up that allows the process for the Constituent System to be monitored for changes.

If there are more Source Processes, then the process reverts to the process modelling activity, if not there is a review that results in the production of a Review Record. In the case where the outcome of the review is positive, then all the process artefacts are baselined and the process ends.

In the case where the outcome of the review is not positive, then the process reverts back to the process modelling activity.

Name: baseline
Description: All process artefacts:
- are recorded
- are held under configuration control
- form part of an overall baseline
- are stored on the project repository

Name: evaluate
Description: The mapping between the SoS Process Model and the CS Process Model can now be used as a basis for evaluation of the Source Process. The following guidelines should be followed when carrying out the evaluation.
- Process structure view (PSV) mapping. There must be a strong relationship between the two views. If it is not possible to map between all of the elements in the SoS Process Model PSV and the CS Process Model PSV then this is potentially the cause of an Exception.
- Process content view (PCV) mapping. There must be a mapping between all of the requirements management processes in the SoS Process Model and the CS Process Model. This need not necessarily be a one-to-one mapping between the two, but there must be a mapping. If this mapping does not exist then this is potentially the cause of an Exception.
- Information view (IV) mapping. There must be some mapping between the key artefacts in the SoS Process Model and the CS Process Model for the requirements management processes. The IV will provide the basis for the Control Point that are set up later in this process and so are vital. If no mapping exists between the artefacts in the SoS Process Model and CS Process Model then this is potentially the cause of an Exception.
- Stakeholder view (SV) mapping. There is no absolute necessity for a mapping between the stakeholder views. If a mapping exists then this can be useful for raising confidence in the CS Process Model, but the absence of a mapping does not result in an Exception.
- Requirement view (RV) mapping. There is no absolute necessity for a mapping between the stakeholder views. If a mapping exists then this can
be useful for raising confidence in the CS Process Model, but the absence of a mapping does not result in an Exception.

- Process behaviour view (PBV) mapping. There is no absolute necessity for a mapping between the process behaviour views. If a mapping exists then this can be useful for raising confidence in the CS Process Model, but the absence of a mapping does not result in an Exception.
- Process instance view (PIV) mapping. There is no absolute necessity for a mapping between the process instance views. If a mapping exists then this can be useful for raising confidence in the CS Process Model, but the absence of a mapping does not result in an Exception.

These guidelines should be used as a basis for the evaluation, but it will be the ultimate decision of the Requirement Manager where an Exception should be raised or not.

Name: identify CS requirement processes
Description: This activity looks at each of the Constituent Systems that make up the System of Systems and identifies their requirements management process.

The information that describes each of these Source Process is then obtained so that it can be used as a basis for the process modelling in the next activity.

Name: map to SoS processes
Description: The CS Process Model from the previous activity is now mapped onto the SoS Process Model in order that their compatibility can be assessed.

The mapping is carried out by putting the CS Process Model and the SoS Process Model side-by-side and then drawing relationships between the views. This can be done as a paper exercise or, at a preference, using tracing mechanisms within a modelling tools.

This mapping is done per view and also with the elements within each view.

Name: model process
Description: This activity takes the information in the Source Process and creates a CS Process Model based on this.

This process modelling should be carried out in a structured and best-practice manner. In the case of the COMPASS project, the process modelling should be carried out in accordance with the ‘seven views’ approach to process modelling. The notation that is used to perform the modelling will be SysML.

Each of the views will be considered, even in the event that there is no information available to create the view.

The resulting CS Process Model will then be used as a basis of the mapping exercise in the next activity.
Name: raise exception
Description: In the event of a problem occurring that cannot be resolved as part of this process, then an Exception is generated.

An Exception requires further input from Stakeholders and will require other processes to be executed in order to be resolved.

Name: review
Description: The aim of this activity is to consider the following:
- Is the result of the modelling and subsequent evaluation acceptable?
- Are the Control Point appropriate?
- Is the monitoring mechanism for each Control Point feasible and appropriate.

There are two outcomes to this review:
- A ‘fail’, where the process will revert back to the model process activity
- A ‘pass’, where the process continues.

Name: set up control points
Description: The aim of this process is to set up Control Point in the constituent system requirements management process. These Control Points may take three different forms:
- The execution of a specific process. For example, if a ‘requirement change’ process or similar exists in the CS requirements management process set, then the execution of this process means that a requirement has changed. This would be a preferred option as it is the simplest and will always result in the change of a requirement.
- The change of a specific document artefact. For example, the CS may have an artefact named ‘requirement specification’. When this artefact changes, then one of the requirements has potentially changed. This will then require an analysis of the artefact to see if the change is significant and, therefore, is not a preferred option for a Control Point.
- The change of a specific model-based artefact. If the CS has adopted a model-based approach then it will be possible to monitor the change of a specific model elements. For example, it would be possible to monitor the change of a Requirement, Goal or Capability. This is much the preferred option.

These Control Points must be identified along with the mechanism that is used to monitor them. This mechanism may range from an automated report from the Constituent System, to an email, to a phone, and anywhere in between.
7.3. PBV - Requirement Control Process

The process begins with the Requirement Manager who communicates the Requirement Model to a relevant set of Stakeholders. A Stakeholder review is then held, resulting in a Review Record, where the aim is to obtain consensus from the Stakeholders. If a consensus is not obtained, then the SoS Requirements Engineering is invoked. If consensus is obtained, then the next step is to obtain commitment to realise the requirements from the Stakeholders. If commitment is not obtained then the process reverts back to the stakeholder review, otherwise all the process artefacts are baselined.
Name: baseline
Description: All process artefacts:
- are recorded
- are held under configuration control
- form part of an overall baseline
- are stored on the project repository

Name: communicate information
Description: The appropriate part of the Requirement Model is communicated for each Stakeholder. This may be the whole Requirement Model, as set of Requirement Views or a sub set of Requirement Element. This may be done in two ways:
- The Stakeholders may be identified based on the individual requirements that form part of the Requirement Model
- The relevant elements that make up the Requirement Model may be identified based on a set of Stakeholders

It is important that only pertinent information is given to each Stakeholder.

Name: obtain commitment
Description: Once a consensus has been reached, it is important to obtain commitment from the Stakeholders.

This includes the following:
- Ensuring that the Stakeholders agree that the Process Model that has been reviewed and agreed is an appropriate way forward.
- Ensuring that all Stakeholders are willing to commit resources to ensure that the Process Model can be satisfied
- Ensuring that all Stakeholders agree to the current project schedule that will deliver the Process Model

Name: stakeholder review
Description: A stakeholder review is carried out. This involves taking as an input all the relevant information from the Requirement Model that was identified in the previous step.

Depending on the level of detail required by the review, the review may take any number of forms, for example:
- The review may be a simple informal meeting where the outcome is based on dialogue between relevant Stakeholders.
- The review may be carried out remotely by commenting on an artefact before sending back a marked-up version, that can then be used as a basis for discussion.
- The review may be a full formal review with an agenda, minutes, action points, etc.
There are several other possibilities for the format of the review, the ones here are included for illustration purpose only.

The main purpose of this activity is to try to obtain consensus from the Stakeholders. There are two possible outcomes here:

- Consensus is obtained, in which case the process continues
- Consensus is not obtained, in which case the SoS Requirements Engineering is invoked

The overall results of this activity are captured in the Review Record.
7.4. PBV - Requirements Change Process

The process begins with the Requirement Manager who identifies any changes by receiving a Change Request. The next step is to assess whether the change is either impacts the internal system, impacts the external system or has no impact, which means:

- In the case of a change to a System of Systems, the system of systems is considered the ‘internal’ system and the Constituent Systems are considered the ‘external’ systems.
In the case of a change to a Constituent System, the constituent system is considered the ‘internal’ system and the System of Systems is considered the ‘external’ system.

In the case of no impact to either internal or external systems, then the process proceeds immediately to the base lining activity.

Any internal changes may be evaluated as part of this process, but any external changes require a second invocation of the Requirements Change Process. The results of this process may then be evaluated.

In the case of an internal or external impact, then a Change Record is produced, which is then reviewed, resulting in a Review Record. If the outcome of the review is successful, then action is taken and then reviewed, until the review is passed.

If the outcome of the change record review is not positive, then the process reverts back to assessing the change.

Finally, all process artefacts are baselined by the Requirement Manager.

**Name: assess internal/external impact**

**Description:** The aim of this activity is to assess whether the Change Request will have an impact on the Process Model and, if so, whether it will be an internal or external impact.

There are two possible outcomes:
- No impact on the Process Model, therefore continue and take no action
- There will be an impact. This impact may be ‘internal’ or ‘external’ and will lead to further evaluation.

If the Change Request applies to a System of Systems, then:
- ‘internal’ refers to a Change Request that impacts only the System of Systems
- ‘external’ refers to a Change Request that impacts one of its Constituent Systems

If the Change Request applies to a Constituent System, then:
- ‘internal’ refers to a Change Request that impacts only the Constituent System
- ‘external’ refers to a Change Request that impacts either the overarching System of Systems or another Constituent System

**Name: baseline**

**Description:** All process artefacts:
- are recorded
- are held under configuration control
- form part of an overall baseline
are stored on the project repository

**Name: change review**

Description: The aim of this activity is to check that the outcomes of the previous evaluation activities are correct.

There are two possible outcomes for this review:

- The proposed evaluation and subsequent actions are fine, so proceed with taking the action.
- The proposed evaluation and subsequent action are not acceptable, so go back and re-assess the Change Request

The outcome of this activity is recorded in the Change Record.

**Name: evaluate external change(s)**

Description: The aim of this activity is to take the Change Request and to evaluate it with regards to the Requirement Model. This evaluation should take into account the following:

- The impact of the Change Request on the rest of the model must be ascertained. This will involve investigating which other elements of the model will potentially change because of their relationships (either directly or indirectly) to the element identified by the Change Request.
- Assessing the extent of the actions that will need to be carried out to make the change in the Change Request. For example, a change may be as simple as re-wording a description, or may be introducing a brand new requirement that may affect the whole project.
- Assessing the impact of these actions on the project itself, such as the schedule, resources, etc.

This information should be captured as part of the Change Record.

**Name: evaluate internal change(s)**

Description: The aim of this activity is to take the Change Request and to evaluate it with regards to the Requirement Model. This evaluation should take into account the following:

- The impact of the Change Request on the rest of the model must be ascertained. This will involve investigating which other elements of the model will potentially change because of their relationships (either directly or indirectly) to the element identified by the Change Request.
- Assessing the extent of the actions that will need to be carried out to make the change in the Change Request. For example, a change may be as simple as re-wording a description, or may be introducing a brand new requirement that may affect the whole project.
- Assessing the impact of these actions on the project itself, such as the schedule, resources, etc.

This information should be captured as part of the Change Record.
Name: identify change(s)
Description: This activity is concerned with taking any Change Request that have occurred and relating them to the Requirement Model. The aim here is to simply identify which part of the model the Change Request applies to and not to perform and assessment on the Change Request.

Name: resolution review
Description: The aim of this activity is to check that the actions taken in response to the Change Request have been carried out correctly. There are two possible outcomes to this:
- The actions taken were sufficient, therefore go back and do them again
- The actions taken were sufficient and the Change Request has been completely satisfied

The outcomes of this activity are recorded in the Change Record.

Name: take action
Description: The activity involves taking whatever actions were decided upon and approved by the previous activity.

Note that these actions can potentially differ wildly, from a simple text change, to invoking a whole new iteration of processes.
7.5. PBV - Requirements Monitor Process

This process is quite straightforward and begins with the Requirement Manager who monitors both the System of Systems Requirement Model and the Constituent Systems Control Points for changes. When a change occurs the Requirements Change Process is invoked, otherwise the process continues monitoring.

Name: monitor CS control points
Description: The requirements that are part of the Constituent Systems will be monitored for any changes. This is achieved by monitoring the Control Points that have been set up. Should any change be detected, then the Requirements
Change Process must be invoked, regardless of how simple or insignificant the change may first appear.

**Name: monitor SoS requirements**
Description: The requirements that are part of the System of Systems will be monitored for any changes. Should any change be detected, then the Requirements Change Process must be invoked, regardless of how simple or insignificant the change may first appear.
7.6. PBV - SoS Requirements Development Process

The diagram here shows the process behaviour view for the SoS Requirements Development process using a SysML activity diagram.

The process begins with the ‘Requirement Engineer’ who identifies the Source Element View and then identifies both the System of Systems and the Constituent System contexts. Once the System Contexts have been identified, then the contexts can be defined. The Context Process is then invoked at the system of system level and a Requirement Model is returned.

The next step is to define the context for each of the Constituent Systems by invoking the Context Process again, this time at the Constituent System level. A Requirement Model is then returned for each of the Constituent Systems.

Once complete, the interactions between the System of Systems and the Constituent Systems can be identified by comparing and analysing the
Requirement Models. This results in the production of the Validation Interaction View and the Context Interaction View.

These views are now reviewed and a Review Record produced and, upon a positive review outcome, all process artefacts can now be baselined by the Requirement Manager.

If the review outcome is not positive, then the process returns to identifying the interactions between the System of Systems and the Constituent Systems.

**Name: baseline**

**Description:** All process artefacts:
- are recorded
- are held under configuration control
- form part of an overall baseline
- are stored on the project repository

**Name: identify interactions between SoS and CS**

**Description:** This activity identifies the interactions between the System of Systems and its Constituent Systems.

The System Context for the System of Systems and its Constituent System are compared and then the interactions between them are identified. This can be achieved in a number of ways, including:
- Considering the Requirement Context Views for the System of Systems and the Constituent System and identifying the relationships between them and creating the Context Interaction View
- Based on these interactions that have been identified, the Validation Views for the System of Systems and the Constituent System can now be analysed and the Validation Interaction View generated.

These two views are then used as part of the review activity.

**Name: identify source elements**

**Description:** This activity identifies the Source Element View that will be used as a basis for the System of Systems Requirement Process

**Name: identify SoS constituent system contexts**

**Description:** This activity identifies all the Constituent Systems that form part of the System of Systems and that will require an understanding of their System Context.

This results in the creation of a System Context Definition View that forms part of the overall Context Definition View.

This activity is carried out at the same time as the identify SoS stakeholder contexts activity.
Name: identify SoS stakeholder contexts
Description: This activity identifies all the Stakeholder that are relevant for the System of Systems.

This results in the creation of a Stakeholder Context Definition View that forms part of the overall Context Definition View.

This activity is carried out at the same time as the identify SoS constituent system contexts activity.

Name: invoke 'context' process for CS
Description: The System Context for the Constituent System is generated by invoking the Context Process that returns a Requirement Model.

Name: invoke 'context' process for SoS
Description: The System Context for the System of Systems is generated by invoking the Context Process that returns a Requirement Model.

Name: review
Description: A review of the System Context for the Constituent Systems and the System of Systems is carried out.

The main aim behind this review is to:
- Ensure that the System of Systems has a context defined.
- Ensure that all the Constituent System that form part of the System of Systems have their System Context defined.
- Check that each of the System Contexts is readable, understandable and accurately reflects the needs of the underlying Requirements.
- Ensure that all the relationships between the System Contexts (between the System of Systems and the Constituent Systems) have been identified.

There are three possible outcomes from this review:
- In the event that the review is successful, then the process ends with the baseline activity.
- In the event that the review is unsuccessful because of the interactions between the System of Systems and the Constituent System, then the processes reverts back to the identify interactions between SoS and CS activity.
- In the event that the review is unsuccessful because the System Contexts are lacking, then the process reverts back to the start of the process.

This activity results in the generation of a Review Record.

Name: select constituent systems
Description: A Constituent System is selected, based on the System Context Definition View, ready for its System Context to be defined.
7.7. PBV - Traceability Process

The process begins with the **Requirement Manager** who identifies traceable elements from the **Process Model** and then identifies their traceability paths. This is then verified with the model, and there are three possible outcomes:

- There are no problems, but there are more elements that need to be traced, in which case the process reverts back to identifying traceable elements.
- There are no problems and no more elements, in which case the traceability is set up with a **Traceability View** and then all the process artefacts are baselined.
• There are problems, in which an Exception is raised and the process ends.

Name: baseline
Description: All process artefacts:
• are recorded
• are held under configuration control
• form part of an overall baseline
• are stored on the project repository

Name: identify traceability paths
Description: Once the traceability elements have been identified, it is then possible to identify paths to other traceability elements that may be used for the traceability. In order for a traceability path to exist, there must be a relationship between two elements on the COMPASS - Ontology or information view from the relevant Process Model.

For example, form the IV - COMPASS - Framework there is a relationship between Context Definition View and Requirement Context View in the association ‘defines context for’. This association may be used as a base for a traceability path between these two views.

Name: identify traceable elements
Description: The aim of this activity is to look for elements in the model that need to traced to or from. These will be identified based on the COMPASS ontology and information views from the relevant Process Model where all possible traceable will be able to be identified.

For example, any model element on the ontology may be traced to or from, any artefact on any information view may be traced to or from.

Name: raise exception
Description: In the event of a problem occurring that cannot be resolved as part of this process, then an Exception is generated.

An Exception requires further input from Stakeholders and will require other processes to be executed in order to be resolved.

Name: set up traceability
Description: The relevant TV are now set up based on the previous activities. This may take a number of forms, for example, drawing up specific traceability diagrams in SysML, drawing up a table, drawing up a matrix, etc.

Name: verify with model
Description: This activity will verify the traceability paths. This is achieved by checking that the traceable elements and their paths are consistent with the existing relationships in the information views in the Process Model.

There are three possible outcomes to this:
• There is a problem with the traceability, therefore an Exception is raised
• There is no problem, but there are more elements to be traced between, therefore go back and identify some more traceable elements.
• There is no problem and no more elements, therefore proceed and set up the traceability on the TVs

The outcomes of this activity will ultimately be captured as part of the traceability view.
7.8. PBV - V&V Definition Process

The process begins with the **Requirement Engineer** who, first of all, selects a context based on the **System of Systems Context Definition View**, and then selecting a single use case from the **Requirement Context View**.

Based on a high-level assessment of the type of verification and/or validation required, the level of rigour is defined and then the scenarios (both semi-formal and formal) are defined, producing one or more **Validation View**.

These artefacts are now reviewed and a **Review Record** is produced. If the outcome of the review is positive, then the **Validation Views** are traced back onto the model and a **Test Coverage View** is produced. This is then assessed to ensure that it covers all relevant parts of the model. Following a positive outcome to the assessment, all the process artefacts are baselined by the **Requirement Manager**.

If the outcome of the review of the **Validation Views** is not positive, then the process reverts back to defining the level of rigour, and continues as previously. If the outcome of the test coverage assessment is not positive, then the process reverts back to selecting a use case and continues as before.

**Name:** baseline

**Description:** All process artefacts:
- are recorded
- are held under configuration control
• form part of an overall baseline
• are stored on the project repository

Name: define formal scenarios
Description: A number of scenarios for the selected Use Case are defined in the form of Constraint Validation Views.

These formal scenarios are shown in SysML using parametric constraints which may then be associated with external, formal and rigorous techniques for modelling scenarios.

These scenarios then form part of the overall Validation View set.

Name: define level of rigour
Description: This is a straightforward decision as to which level of rigour is necessary to validate the Use Case. The level of rigour can be: formal, semi-formal or both.

Name: define semi-formal scenarios
Description: A number of scenarios for the selected Use Case are defined in the form of Stakeholder Scenario Views and System Scenario Views.

These formal scenarios are shown in SysML using sequence diagrams. These scenarios then form part of the overall Validation View set.

Name: review
Description: A validation review is carried out. The main aim of this activity is to ensure that the Validation View that have been created present an accurate and feasible demonstration of satisfying the associated use case.

There are two possible outcomes to this review:
• In the event that the review is successful, then process continues with the trace to model activity
• In the event that the review is unsuccessful, then the process reverts back to the define level of rigour activity.

This process results in the creation of a Review Record

Name: review coverage
Description: This activity performs a review of the Test Coverage View.

The aim of this activity is to ensure that all of the Use Cases in the Requirement Context Views have Validation View defined for them. This is achieved by assessing the Test Coverage View.

There are two possible outcomes from this assessment:
- In the event that the coverage is sufficient, then the process ends with the baseline activity.
- In the event that the coverage is insufficient, then the process reverts back to the select use case activity.

This activity results in the production of a Review Record.

**Name: select context**
Description: Based on the Context Definition View, the appropriate context is identified.

This is carried out as more of a checking activity to ensure that the Requirement Context View that is received is correct.

**Name: select use case**
Description: Once the appropriate context has been identified and the Requirement Context View selected, then a Use Case can be selected that will have its Validation View created.

**Name: trace to model**
Description: The reviewed Validation Views are now applied back to the model. The main aim of this activity is to ensure that all of the Use Cases have a number of Validation Views associated with them.

This is not as straightforward as it may seem as there will not necessarily be a number of Validation View associated with each and every Use Case. This may be for a number of reasons, including:

- Use Cases that are included as part of other Use Cases. If scenarios are associated with a high-level Use Case, then it is possible that all of the lower-level Use Cases can be satisfied. Conversely, if all the lower-level Use Cases can be satisfied, then it is possible that the higher-level UCs can be satisfied.
- Use Cases that are generalisations of other Use Cases. If scenarios are associated with a specialised Use Case, then it is possible that they are also satisfy the parent Use Case. Conversely, if scenarios are defined for a parent Use Case, then it is possible that they also satisfy the child Use Cases.

This activity results in a Test Coverage View that indicates which elements of the Requirement Context Views have associated Validation View, either directly or indirectly.
8. Process Instance View

8.1. CS context definition scenario

This scenario shows the situation where a context where a constituent systems needs to be defined.

The SoS Requirements Engineering process calls from the Context Process which then calls the Verification and Validation Definition Process. Control is then passed back to the SoS Requirements Engineering process.

This scenario validates the following use cases:
- Understand constituent system contexts
- Understand context
- Define V&V criteria

8.2. Monitoring - failed control

This scenario shows the situation where requirements are being monitored for changes. In this scenario, the control fails.

The start point is that the Set up scenario - OK scenario is executed which ends in the requirements being monitored for changes. When a change is detected, the
SoS Requirements Engineering process is called up and from there the Traceability Process and the Requirement Control Process.

A problem occurs and the SoS Requirements Engineering process is called up again.

This scenario validates the following use cases:
- Manage requirements change

### 8.3. Monitoring - failed traceability

This scenario shows the situation where requirements are being monitored for changes. In this scenario, the traceability fails.

The start point is that the Set up scenario - OK scenario is executed which ends in the requirements being monitored for changes. When a change is detected, the SOS process is called up and from there the Traceability Process.

A problem occurs and an exception is raised.

This scenario validates the following use cases:
- Manage requirements change

### 8.4. Monitoring scenario - OK

This scenario shows the situation where requirements are being monitored for changes. In this scenario, the traceability fails.

The start point is that the Set up scenario - OK scenario is executed which ends in the requirements being monitored for changes. When a change is detected, the SOS process is called up and from there the Traceability Process.

A problem occurs and an exception is raised.

This scenario validates the following use cases:
- Manage requirements change
This scenario shows the situation where requirements are being monitored for changes. In this scenario, everything goes Ok and according to plan.

The start point is that the Set up scenario - OK scenario is executed which ends in the requirements being monitored for changes. When a change is detected, the SoS Requirements Engineering process is called up and from there the Traceability Process and the Requirement Control Process.

Once this is complete, control returns to the Requirements Change Process and the monitoring continues.

This scenario validates the following use cases:

- Manage requirements change

### 8.5. Set up scenario - failed CS process analysis

![Diagram](Figure 63 Set up scenario - failed CS process analysis)

This scenario shows the generic situation where the whole requirements activity is set up at the beginning of the project. In this scenario, there is a failure during the analysis of the CS processes.

The start point is that the SoS Requirements Engineering scenario is executed which ends in the Traceability Process being called up. From here, the Requirement Control Process is called up and then the CS Process Analysis. A problem occurs during the process and an exception is raised.

This scenario validates the following use cases:

- Control process artefacts
- Communicate with stakeholders
- Provide traceability
- Understand constituent systems RM processes
8.6. Set up scenario - failed requirements control

This scenario shows the generic situation where the whole requirements activity is set up at the beginning of the project. In this scenario, there is a failure during the control of the requirements.

The start point is that the **SoS Requirements Engineering scenario** is executed which ends in the **Traceability Process** being called up. From here, the **Requirement Control Process** is called up and a problem occurs. The **SoS Requirements Engineering** is then called up.

This scenario validates the following use cases:

- Control process artefacts
- Communicate with stakeholders
- Provide traceability

8.7. Set up scenario - failed traceability

This scenario shows the generic situation where the whole requirements activity is set up at the beginning of the project. In this scenario, there is a failure during the control of the requirements.

The start point is that the **SoS Requirements Engineering scenario** is executed which ends in the **Traceability Process** being called up. From here, the **Requirement Control Process** is called up and a problem occurs. The **SoS Requirements Engineering** is then called up.

This scenario validates the following use cases:

- Control process artefacts
- Communicate with stakeholders
- Provide traceability
This scenario shows the generic situation where the whole requirements activity is set up at the beginning of the project. In this scenario, there is a failure during the traceability of the requirements.

The start point is that the SoS Requirements Engineering scenario is executed which ends in the Traceability Process being called up and a problem occurs. An exception is then raised.

This scenario validates the following use cases:

- **Provide traceability**

### 8.8. Set up scenario - OK

Description: This scenario shows the generic situation where the whole requirements activity is set up at the beginning of the project. In this scenario, everything goes OK and according to plan.

The start point is that the SoS Requirements Engineering scenario is executed which ends in the Traceability Process being called up. From here, the Requirement Control Process is called up and then the CS Process Analysis. After the CS processes have been analysed, the Requirements Monitor Process is called up.

This scenario validates the following use cases:

- **Control process artefacts**
- **Communicate with stakeholders**
- **Understand constituent systems RM processes**
- **Provide traceability**
8.9. SoS Requirements Engineering scenario

This scenario shows the generic high-level situation that describes the whole of the SoS requirements engineering.

The SoS Requirements Engineering process is called and, from there, the Context Process is called to define the systems of systems context. This then calls the Verification and Validation Definition Process and then control returns to the SoS Requirements Engineering process.

For each of the constituent systems that make up the system of systems, the CS context definition scenario is then executed.

When the SoS Requirements Engineering process is complete, the Traceability Process is called up.

This scenario validates the following use cases:
- Provide SoS requirements engineering process
- Identify SoS stakeholders
- Understand SoS context
- Define V&V criteria
- Understand SoS to constituent system interactions

8.10. Traceability- Scenarios to Use Cases

The table below shows the traceability between the scenarios described above and the use cases that they validate.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Satisfied use cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS context definition scenario</td>
<td>Understand constituent system contexts, Understand context, Define V&amp;V criteria</td>
</tr>
<tr>
<td>Monitoring - failed control</td>
<td>Manage requirements change</td>
</tr>
<tr>
<td>Monitoring - failed traceability</td>
<td>Manage requirements change</td>
</tr>
<tr>
<td>Monitoring scenario - OK</td>
<td>Manage requirements change</td>
</tr>
<tr>
<td>Scenario</td>
<td>Satisfied use cases</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Set up scenario - failed CS process analysis</td>
<td>Control process artefacts</td>
</tr>
<tr>
<td></td>
<td>Communicate with stakeholders</td>
</tr>
<tr>
<td></td>
<td>Provide traceability</td>
</tr>
<tr>
<td></td>
<td>Understand constituent systems RM processes</td>
</tr>
<tr>
<td>Set up scenario - failed requirements control</td>
<td>Control process artefacts</td>
</tr>
<tr>
<td></td>
<td>Communicate with stakeholders</td>
</tr>
<tr>
<td></td>
<td>Provide traceability</td>
</tr>
<tr>
<td>Set up scenario - failed traceability</td>
<td>Provide traceability</td>
</tr>
<tr>
<td>Set up scenario - OK</td>
<td>Control process artefacts</td>
</tr>
<tr>
<td></td>
<td>Communicate with stakeholders</td>
</tr>
<tr>
<td></td>
<td>Understand constituent systems RM processes</td>
</tr>
<tr>
<td></td>
<td>Provide traceability</td>
</tr>
<tr>
<td>SoS Requirements Engineering scenario</td>
<td>Provide SoS requirements engineering process</td>
</tr>
<tr>
<td></td>
<td>Identify SoS stakeholders</td>
</tr>
<tr>
<td></td>
<td>Understand SoS context</td>
</tr>
<tr>
<td></td>
<td>Define V&amp;V criteria</td>
</tr>
<tr>
<td></td>
<td>Understand SoS to constituent system interactions</td>
</tr>
</tbody>
</table>

Table 1 Traceability from Scenarios to Use Cases