Requirements Traceability using SysML Diagrams and BPMN

Corina Abdelahad and Daniel Riesco
Departamento de informática
Universidad Nacional de San Luis
San Luis, Argentina
e-mail: cabdelah@unsl.edu.ar, driesco@unsl.edu.ar

Carlos Kavka
Research and Development Department
ESTECO SPA
Trieste, Italy
e-mail: kavka@esteco.com

***Abstract*— An important activity in systems development is ensuring that all system requirements are met. Model-Based Systems Engineering is a methodology that benefits the documentation of the requirements and decisions that are made during the design process. On the other hand, visualizing different perspectives that focus on different aspects of the system permits to capture all the details of the design while refining the level of detail of the models. SysML is a Systems Modeling Language, which is defined as an extension of the well-known Unified Modeling Language standard. It is based on four pillars, which give the possibility to view a system from four different perspectives, supporting requirements traceability. Requirements traceability refers to the ability to describe and follow the life of a requirement in both a forward and backward direction. This traceability has an important role in Model-Based Systems Engineering. The central aim of this paper is to present a traceability approach that supports decision-making requirements. To carry out this traceability we propose to combine SysML and Business Process Model and Notation and Decision Model and Notation. SysML is used to model some aspects of system, and processes and decision-making activities are defined in terms of BPMN and DMN standards, respectively. This proposal seeks to help engineers to improve their design and enhance traceability starting from requirements, integrating and covering the different views. Our contribution is illustrated by means of a case study.**

Keywords-SysML; BPMN; DMN; requirements traceability.

I. INTRODUCTION

Abstraction is a technique used by engineers to deal with complexity, permitting them to focus only on the information that is considered significant or relevant. To improve the design of requirements, to understand and cover their different views improving maintenance and verification activities, it is necessary to carry out requirements traceability [1]. International Council on Systems Engineering (INCOSE) [2] indicates that "requirements traceability refers to the ability to describe and follow the life of a requirement in both a forward and backward direction along the design stages". Traceability plays an important role as part of any Model-Based Systems Engineering (MBSE) methodology. MBSE is a successful methodology for the design of complex systems, which emphasizes the use of models when performing systems engineering activities [3]. These models, which can be executable or not, are used to describe the structure and the behavior of the systems.

With the evolution of systems engineering, the need for a consistent standard modeling language arose. INCOSE together with the Object Management Group (OMG) [4] defined SysML, a general-purpose modeling language based on UML, which can be used for specifying, analyzing, designing, and verifying complex systems, including hardware, software, information, personnel, procedures, and facilities [5]. SysML is based on four pillars, which give the possibility to view a system from four different perspectives: Requirements, Structure, Behavior and Parametrics, each one of them defined in terms of diagrams [6]. Requirements modeling [7] is implemented in terms of the requirement diagram, which allows for capturing, analyzing and maintaining traceability of requirements in the modeled system. Structure modeling has a block definition diagram as the main diagram, representing structural elements (blocks) with their properties, relationships, and composition. Behavior modeling has different kinds of behavior diagrams like activities diagram, state machine, and sequence diagram. Parametric modeling has a parametric diagram that can be used to identify the system constraints [5]. In SysML, requirements can be related to other requirements, as well as to other model elements via one or more relationships, making possible the traceability of requirements. Furthermore, SysML can be integrated into other tools including spreadsheets and design and simulation software, such as Matlab or Modelica [8], enabling requirements verification.

The specification of business processes also followed the same path requiring standards for its definition. In particular, the Business Process Management Initiative (BPMI) together with the OMG developed the widely used BPMN notation for modeling business processes [9]. BPMN defines an abstract representation for the specification of business processes, which can include human intervention or not. BPMN couples an expressive graphical representation with a rigorous Extensible Markup Language (XML) encoding of processes and the interactions among them, supporting not only modeling activities but also process execution by using appropriate BPMN engines. Several works in the engineering field have shown the value of using BPMN instead of UML activity diagrams. Activity Diagram can be used for business process modeling but BPMN was designed exclusively for modeling business process [9] and OMG adopted BPMN instead of the Activity Diagram (UML AD) as the core standard to create a business modeling framework [10]. BPMN has model elements that, in some cases, do not have a corresponding element in UML 2.0 AD. There are cases when components of the business processes are

modeled using only one symbol in BPMN and using a group of symbols in UML AD [11]. Since many activities within a business process involve decision-making, the OMG defined recently the Notation and Decision Model and Notation (DMN) standard for the elicitation and representation of decision models, effectively separating decision logic and control flow logic in business processes [12]. DMN was designed to be usable alongside the standard BPMN. At present, many companies have adopted BPMN not only because of its popularity, but because it is strongly related to DMN. This standard is already receiving adoption in the industry, with many tools being developed to assist users in modeling, checking, and applying DMN models.

As the main contribution, this work presents an innovative approach to enhance requirements traceability in the context of MBSE, by combining SysML, BPMN and DMN. This approach can help systems engineers to improve the design of requirements, to understand and cover their different views, improving maintenance and verification activities while contributing to refine the level of detail of the models.

The rest of this paper is organized as follows: Section II introduces related work, while Section III summarizes the basic concepts used in this paper. Section IV addresses the proposed approach with a case study presented in Section V. Section VI shows the conclusion.

II. RELATED WORK

Several works in the field of software engineering are related to the concept of requirements traceability using SysML. For example, the authors in [13] show how requirements traceability for mechatronic design can be achieved using MBSE and SysML. SysML is used for linking system requirements to the system elements of the different domain models while guaranteeing the traceability. This paper presents a case study of a mechatronic system in order to show this traceability.

In [14], the authors propose a Model-Based Systems Engineering approach based on SysML. This approach enables the capture and the definition of functional requirements, validate these functional requirements through functional simulation, and verify efficiently the consistency of these functional requirements. The approach is illustrated by means of a case study of an industrial avionics system.

In [15], a solution for SysML model verification and validation in an industrial context is presented. The authors provide a method and a list of the existing challenges; besides that, they show experimental results. A case study is presented, verification rules are in Object Constraint Language (OCL), while the validation rules are in a formal text format evaluated by a script. The authors mention that the verification of these rules ensures a certain degree of traceability.

In [16], an approach to construct true model-based requirements in SysML is presented. This approach proposes that every requirement can be modeled as an input/output transformation. This proposal uses SysML behavioral and structural models and diagrams, with specific construction rules derived from Wymore's mathematical framework for MBSE and taxonomies of requirements and interfaces. The authors consider that this proposal provides several benefits, including traceability, and improved precision over the use of natural language.

In [17], the authors propose a model-based approach to automotive requirements engineering for the general development of vehicles of passengers. The SysML requirement element is extended, through stereotype, to functional and non-functional requirements. The paper validates the advantages that include classified and modeled requirements graphically, as well as their relationships that are explicitly mapped. This article presents a case study that shows the proposed extension and the performed requirements traceability.

In [18], the authors propose a model-driven requirement engineering approach for the embedded software domain. This approach is based on UML, MARTE and SysML standard notations, which are integrated in order to improve requirements specification and traceability. MARTE is used to allow domain-specific non-functional requirements to improve the software specification and SysML is combined with UML/MARTE models to support requirements management, to follow their changes. The approach is illustrated by means of a case study.

In [19], the authors propose a metamodel, which establishes the traceability links among the requirement model, the solution model and the verification and validation model for embedded system design. This approach enables traceability of requirements by considering heterogeneous languages for modeling and verifying real-time embedded systems. A case study illustrates the approach with the use of languages such as SysML, MARTE, SIMULINK, among others.

However, to the best of our knowledge, no research work about requirements traceability has considered the decision requirement, a kind of requirement that involves decision making. This requirement appears in the decision requirement diagram, which represents human decision making or automated decision making within a process. The main motivation of this work is the need to provide support to decision requirements, by offering adequate tools to the systems engineers that improve the design and handling of these types of requirements. Considering this, the approach presented in this paper is a step forward to support decision requirements, completing the different system engineering views by combining of SysML, BPMN and DMN.

III. BASIC CONCEPTS

This section presents the basic concepts on which the proposed approach is based. Section A describes the role of requirements in system engineering, Section B introduces traceability related concepts in SysML, Section C describes the SysML requirements diagram and its relationships with others diagrams like the: use case diagram, block definition diagram and state machine used in the approach. Section D shows some concepts about DMN and its relationship with BPMN.

A. Requirements

Requirements are the base in system development. They determine what the system has to offer, they can specify a desired feature, property, or behavior of a system, i.e., requirements set out what the system should do and define constraints that it has [20]. The concept of the requirement may also be further classified as [21]:

- Business Requirement. A Business Requirement is used to indicate the needs of a business. This impact on the organization and all the projects within it.
- Functional Requirement. Functional Requirements produce an observable result to someone, or something, that is using the system, i.e., they are the services that the system should provide.
- Non-functional Requirement. A Non-functional Requirement will constrain, or limit in some way, the way in which Functional Requirement may be realized.

B. Traceability in SysML

In [2], INCOSE indicates that "requirements traceability refers to the ability to describe and follow the life of a requirement in both a forward and backward direction along the design stages". Traceability plays an important role as part of any MBSE methodology [3]. MBSE emphasizes the use of models to perform the systems engineering activities, as mentioned before. In fact, "MBSE is the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases" [22].

Modeling with SysML allows good traceability because it defines relationships between requirements and among other modeling elements [23][24]. Figure 1 describes the approach in which SysML accomplishes traceability by means of the 4 pillars presented in Section I. This figure shows the system model as an interconnected set of model elements. The arrows that cross the pillars, as seen in Figure 1, illustrate how the different elements belonging to the different types of diagrams that participate in the pillars are related, supporting requirements traceability.

C. SysML Requirements diagram and its relationships with others diagrams

In SysML, the requirements diagram shows the set of requirements and the relationship between them. A requirement specifies a function that must be satisfied or a condition that a system must achieve. Requirements modeling provides a bridge among different SysML diagrams because a requirement can appear on other diagrams to show its relationship to other modeling elements. The relationships that allow relating requirements with other requirements or with other modeling elements are [5]:

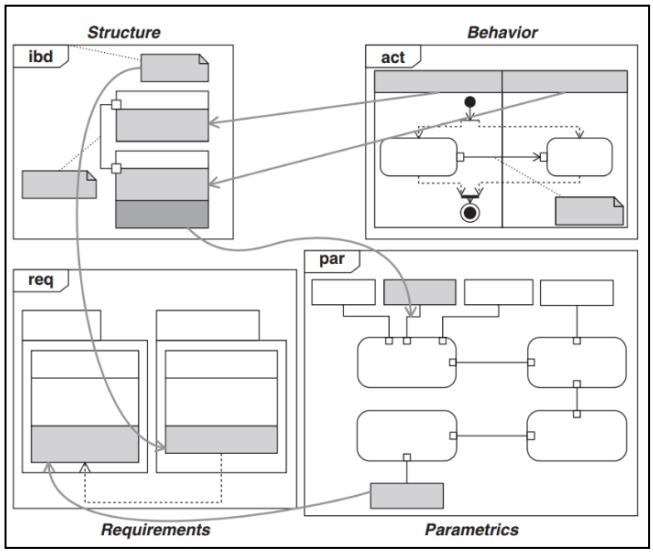

Figure 1. A system model example in SysML where requirements traceability is indicated with the connecting arrows (from [6]).

- Containment: a relationship, which is used to represent how a compound requirement can be partitioned into a set of simpler requirements (denoted graphically with a circle containing a + symbol).
- «deriveReqt»: a relationship, which describes that a requirement is derived from other requirement.
- «satisfy»: a relationship that describes that a design element satisfies a requirement. Usually, a requirement is satisfied by a block.
- «verify»: a relationship that connects a test case with the requirement that is verified by that test case.
- «refine»: a relationship, which specifies that a model element describes the properties of a requirement in more detail.
- «trace»: a general-purpose relationship between a requirement and any other model element.

The requirements can be related to the use cases through the relationship «refine». On the one hand, a use case can be viewed as functionality and/or capacity. On the other hand, a requirement specifies a capability or condition that must be satisfied, as previously mentioned, therefore, a use case diagram may be used to refine one or more functional requirements. In addition, the requirements are related to the blocks through the relationship «satisfy», as mentioned before. The block definition diagram captures the relation between blocks, such as a block hierarchy. Since the activities can be seen as a block, they can have associations between each other, including composition associations. Activities in block definition diagrams appear as regular blocks, except for the «activity» keyword [5]. Depending on the nature of the block, this can have a behavior associated, in that case, states machine can be used to describe its internal states. The state machine diagram is used to specify a behavior, with a focus on the set of states of a block and the possible transitions between those states in response to

event occurrences, i.e., the state machine diagram presents behavior of an entity, as a block, in terms of its transitions between states triggered by events [25].

SysML enables characterization of any type of requirements for the system, including user, technical or others. A modeler can then define relationships between the specified requirements, providing the opportunity to create traceability among them. There is also an opportunity to create traceability from the logical and structural architecture design to their requirements, one of the most critical activities in systems engineering [26].

D. BPMN and DMN

The OMG provides the DMN notation for modeling decisions, which is not only understandable to stakeholders but it is also designed to be used in conjunction with the BPMN standard notation [12].

DMN provides constructs to both decision requirements and decision logic modeling. For decision requirements modeling, it defines the concept of Decision Requirements Graph (DRG) depicted with the Decision Requirements Diagram (DRD). This latter shows how a set of decisions depends on each other, on input data, and on business knowledge models. A decision element determines an output from the inputs, using decision logic, which may reference one or more business knowledge models. This denotes a function encapsulating business knowledge, e.g., as business rules, a decision table, or an analytic model. A decision table is a representation of decision logic, based on rules that determine the output depending on the inputs [12]. Decision-making modeled in DMN may be mapped to BPMN tasks or activities (Business Rules) within a process modeled with BPMN. The combined use of both thus provides a graphical language for describing decision-making, i.e., the BPMN tasks involving a decision can invoke a DMN decision model.

IV. MBSE AND REQUIREMENTS TRACEABILITY WITH SYSML

In this section, our contribution of traceability of requirements using SysML, BPMN, and DMN is detailed. Section A presents an extension to SysML for BPMN tasks while Section B describes details on the proposed approach for requirements traceability.

A. SysML extensions for BPMN tasks

In order to support the modeling of BPMN tasks in SysML, the element of SysML block diagram must be extended. As noted above, in block definition diagrams, the activities appear as regular blocks with an «activity» stereotype. The stereotypes are one of the extensibility mechanisms of UML, therefore also of SysML, that enable to extend its vocabulary allowing the creation of new kinds that are derived from existing ones but specific to a problem [27]. Stereotypes are shown as text strings surrounded by the symbols "« »" [28]. The stereotypes change or add semantics to a base SysML element.

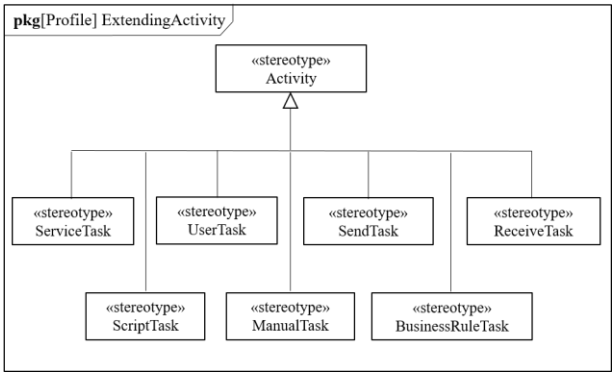

Figure 2. Extension of the SysML «Activity» stereotype.

Figure 2 shows the new types of activities through a generalization in order to support all types of BPMN tasks [5].

This extension consists of the following stereotypes:

- «serviceTask»: represents a task that uses a web service or an automated application.
- «sendTask»: represents a simple task that is designed to send a message to an external participant.
- «receiveTask»: represents a simple task that is designed to wait for a message to arrive from an external participant.
- «userTask»: represents a task where a person performs the task with the assistance of a software application.
- «manualTask»: represents a task that is expected to be performed without the aid of any business process execution engine or any application.
- «scriptTask»: represents a task executed by a business process engine.
- «businessRuleTask»: represents a task that involves decision-making.

The business rule task was defined in BPMN as a placeholder for (business-rule-driven) decisions, being the natural placeholder for a decision task [12].

B. Requirements Traceability using SysML and BPMN-DMN

The interaction between the process and the decision models plays a crucial role because a decision can affect the process behavior or flow [9]. Therefore, it is important that decision-making must be considered as a requirement that should be performed and satisfied.

The approach will be illustrated with an example intended to carry out the traceability of the requirements through forward engineering, mainly focusing on those requirements involved in the decision-making activities, in the blocks that have a behavior related with some of these activities, and in the use cases that refine some of those requirements, with the aim of integrating and covering their different views.

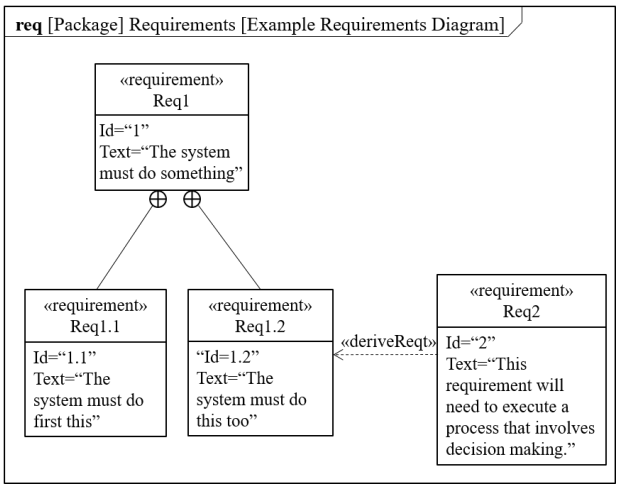

Figure 3. An example of a requirements diagram.

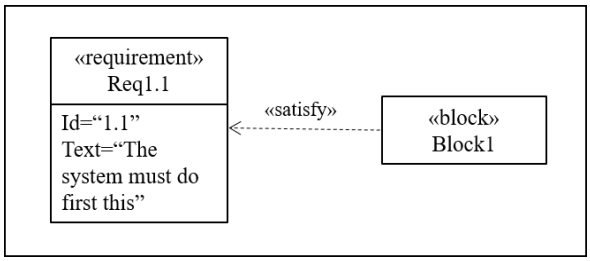

Figure 4. Example of a «satisfy» relationship between an Block and a Requirement.

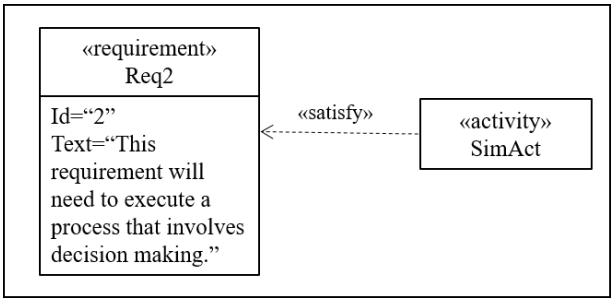

Figure 5. Example of a «satisfy» relationship between an Activity and a Requirement.

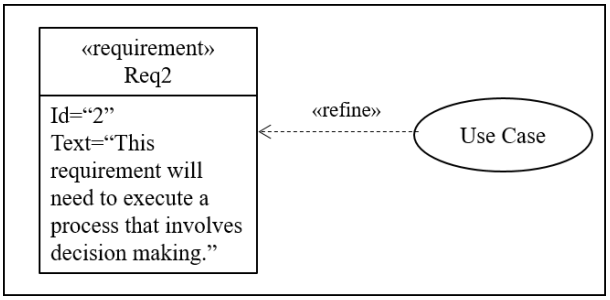

Figure 6. Example of a «refine» relationship between a Use Case and a Requirement

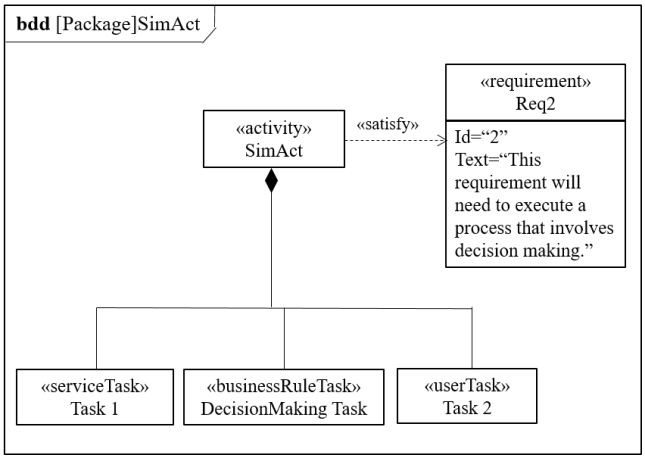

Figure 7. Block definition diagram with activities as blocks.

As it was mentioned before, the SysML requirement diagram has several relationships used to connect requirements. For example, Figure 3 presents a SysML requirements diagram labeled "*Example Requirements Diagram*", which shows the relationship between requirements. In particular, it can be observed that the requirement with *id="2"* has a relationship with the requirement with *id="1.2"* through the «deriveReqt» relation. This relation specifies that the requirement with *id="2"* is derived from the requirement with *id="1.2".*

Requirements can be related to other requirements and to other modeling elements through a specific set of relationships as mentioned before. Relationships between requirements and other modeling elements can appear on various types of diagrams. Figure 4 shows an example of a «satisfy» relationship between a *Block1* block and a *Req1.1* requirement, Figure 5 presents an example of a «satisfy» relationship between a *SimAct* activity and the *Req2* requirement, while Figure 6 shows an example of a «refine» relationship between a *Use Case* and a *Req2* requirement. Both requirements appear in the requirement diagram presented in Figure 3.

The interpretation of the «satisfy» relationship is that the design of the activity depends on the requirement, meaning that if the requirement changes, the design of the activity must be changed. The interpretation of the «refine» relationship is that the Use Case is more concrete than the requirement, i.e., less abstract.

Once the main requirements have been captured, the elements responsible to satisfy those requirements are modeled through a block definition diagram. As previously mentioned, activities can be seen as a block, except for the «activity» keyword [5]. This later provides a means for representing activity decomposition and allows a requirement to be satisfied with an activity. This activity can then be implemented in terms of the decision requirements diagrams and decision tables DMN. Figure 7 shows an example of decomposition of the *SimAct* activity, which was presented in Figure 5 by using the stereotypes proposed in Section IV-A. The block definition diagram shown in this

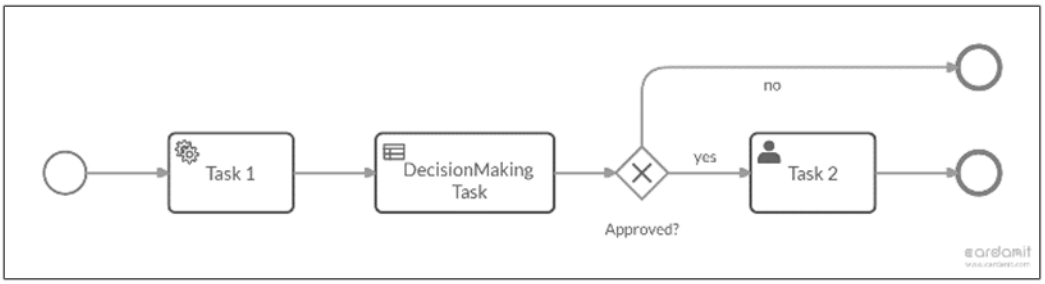

Figure 8. BPMN diagram of SimAct.

figure indicates that the *SimAct* is an activity composed of other activities, including *Task 1*, *DecisionMaking Task* and *Task 2*, all these activities being of BPMN activity types.

Finally, in order to cover the different views of the requirements, a BPMN model is constructed and associated to *SimAct* activity in order to show its behavior, as can be seen in Figure 8. In this figure, the activities that compose *SimAct*, which were modeled in Figure 7, are explicitated in BPMN format.

To conclude, the decision model related to the business rule task is built, since when BPMN and DMN are used, the BPMN tasks (business rule) have a link associated to the decision model, as mentioned in Section III-D. The *DecisionMaking Task* can then be implemented in terms of the associated decision requirements diagrams and decision tables.

V. CASE STUDY

To demonstrate our approach, we conducted a case study that includes the partial modeling of a Biodiesel Distiller and its requirements management. This case study illustrates how to carry out the traceability of the requirements through forward engineering.

Biodiesel is a type of biofuel that is similar to petroleum-based diesel, which can replace fossil fuel diesel. It is a sustainable fuel that is produced from fatty acids derived from animals such as beef fat, pork fat, chicken fat; and vegetable oils such as corn oil and cooking oil like those from restaurants that have already been used and disposed of. These oils are converted to diesel fuel through a chemical process.

Distilled biodiesel is a clean fuel that has been purified through the process of distillation. Biofuel distillation is a method that consists of taking a biofuel and removing particles and impurities within the liquid through an evaporation and condensation process.

The requirements diagram in Figure 9 shows the breakdown of the Biodiesel Distiller's requirements into a hierarchy of more refined requirements. This diagram named "Biodiesel Distiller Requirements Diagram" shows the relationship between its elements. In particular, it can be observed that the requirement *Initial Statement* is partitioned into a set of simpler requirements: *Generate Biodiesel*, *Heat Exchanger*, *Boiler*, *Biodiesel Properties* and *Distill Water*. In addition, two use cases can be seen in the figure: *Distill* and *Change temperature*, one of them refines *Generate Biodiesel* requirement, and the other refines *Heat Exchanger* requirement. In other words, both use cases are more concrete than the requirements, as was indicated in Section IV-B.

As previously mentioned, requirements can be related to other modeling elements through a specific set of relationships as «satisfy» relationship between a block and a requirement. Furthermore, activities can be seen as blocks. As mentioned in Section III-C, depending on its nature, a block can have a behavior associated and this behavior can be modeled using state machines.

Once the main requirements of *Biodiesel Distiller* have been captured, the elements responsible to satisfy them are modeled through a block definition diagram.

Figure 10 illustrates how Machine activity is composed of *Boiler* block, *Water* block, *Generator* activity, which satisfies *Generate Biodiesel* requirement, and *Heat Exchanger* block, which satisfies *Heat Exchanger* requirement, both requirements appear in the requirements diagram presented in Figure 9.

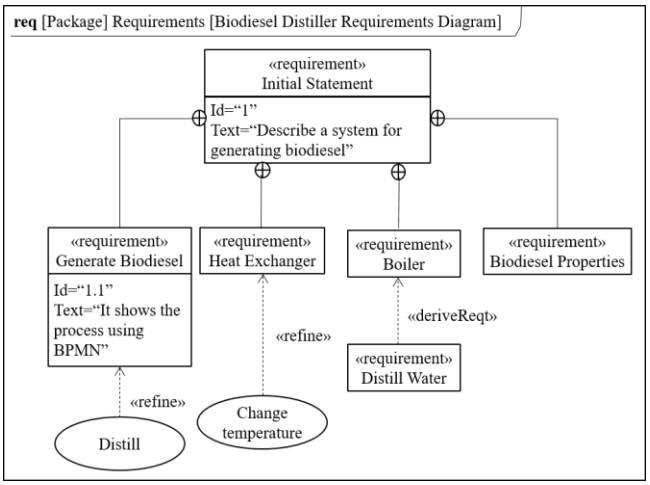

Figure 9. Requirements diagram: *Biodiesel Distiller Requirements Diagram*

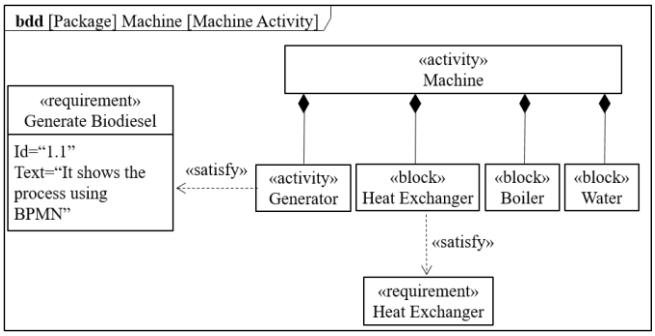

Figure 10. Generator activity satisfies the Generate Biodiesel requirement, and Heat *Exchanger* block satisfies the *Heat Exchanger* requirement

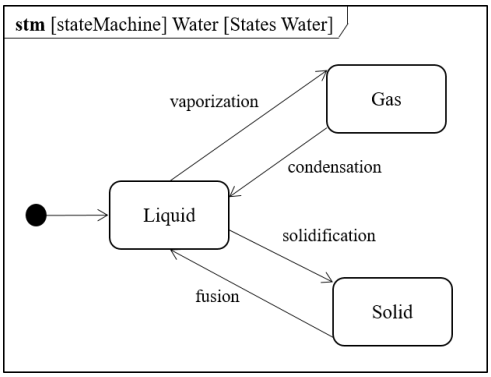

Figure 11. States machine diagram: *States Water*

Continuing with the approach, the state machine diagram in Figure 11 shows possible states of Water block presented in Figure 10, which correspond to the states of water during the distillation process shown in Figure 14. Figure 12 illustrates the decomposition of the *Generator* activity by using the stereotypes proposed in Section IV-A. The block definition diagram shown in this figure indicates that the *Generator* is an activity composed of other activities such as: *Choose method* business rule, which involves decision-making, *Prepare reactors* activity, *Notify* user task, *Heat Water* service task, *Separate materials* service task, *Decant* activity and *Washing by decanting* service task, all these activities being of BPMN activity types.

Following the approach presented in this work, a BPMN model is constructed in order to cover the different views of the requirements. This model shows the process, which is carried out to generate biodiesel associated with the *Generator* activity. Its behavior can be observed in Figure 14. Note that the states of water presented in Figure 11 participate in several of the activities of the BPMN model such as: *Decant, Separate materials, Heat Water and Washing by decanting.*

The decision model related to the business rule task is built once the BPMN model is generated. To prepare the reactors, the type of method to be used must be known and this depends on the type of material that will be used for the generation of biodiesel. The materials can be beef fat, pork

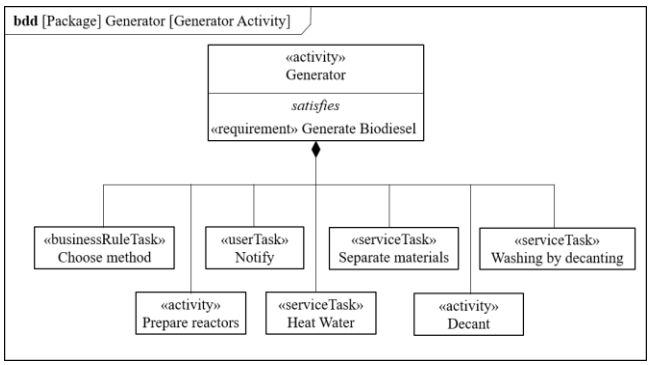

Figure 12. Decomposition of the *Generator* activity.

U	material	Method
	Text [beef, pork, chicken, vegetable]	Text [M1,M2,M3,M4]
1	beef	M1
2	pork	M2
3	chicken	M3
4	vegetable	M4

Figure 13. Decision table to *Choose method*.

fat, chicken fat, and vegetable fat. In the case of study, this decision making is shown in terms of the decision table as shown in Figure 13.

As it was mentioned before, traceability refers to the capability to describe and follow the life of a requirement in both a forward and backward direction along the design stages. Traceability plays an important role in the MBSE methodology. To conclude the study case, and in order to show the requirements traceability presented in this article, Figure 15 shows our approach using three of the four pillars of SysML presented in Section I. This figure shows, through direct engineering, how traceability is carried out, showing how all the models and their elements presented in this section are connected to each other.

As it was mentioned before, the multiple cross-cutting relationships between the model's elements enable systems engineers to view several different perspectives that focus on different aspects of the system.

The arrows that cross the pillars structure, behavior, and requirements, as seen in Figure 15, illustrate how the different elements belonging to the different types of models that participate in these pillars are interconnected, supporting in this way requirements traceability.

Figure 15 shows the different diagrams presented in this study case: requirement diagram, block definition diagram, state machine diagram, BPMN diagram, and decision table, respectively.

The Requirements pillar contains the requirements model that was initially built for the development of the case study.

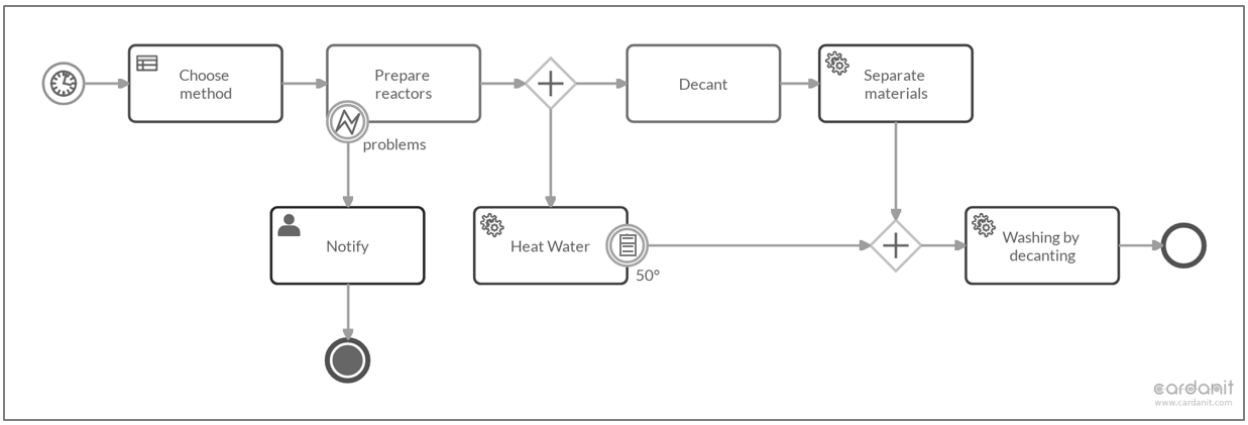

Figure 14. BPMN model of *Generator*.

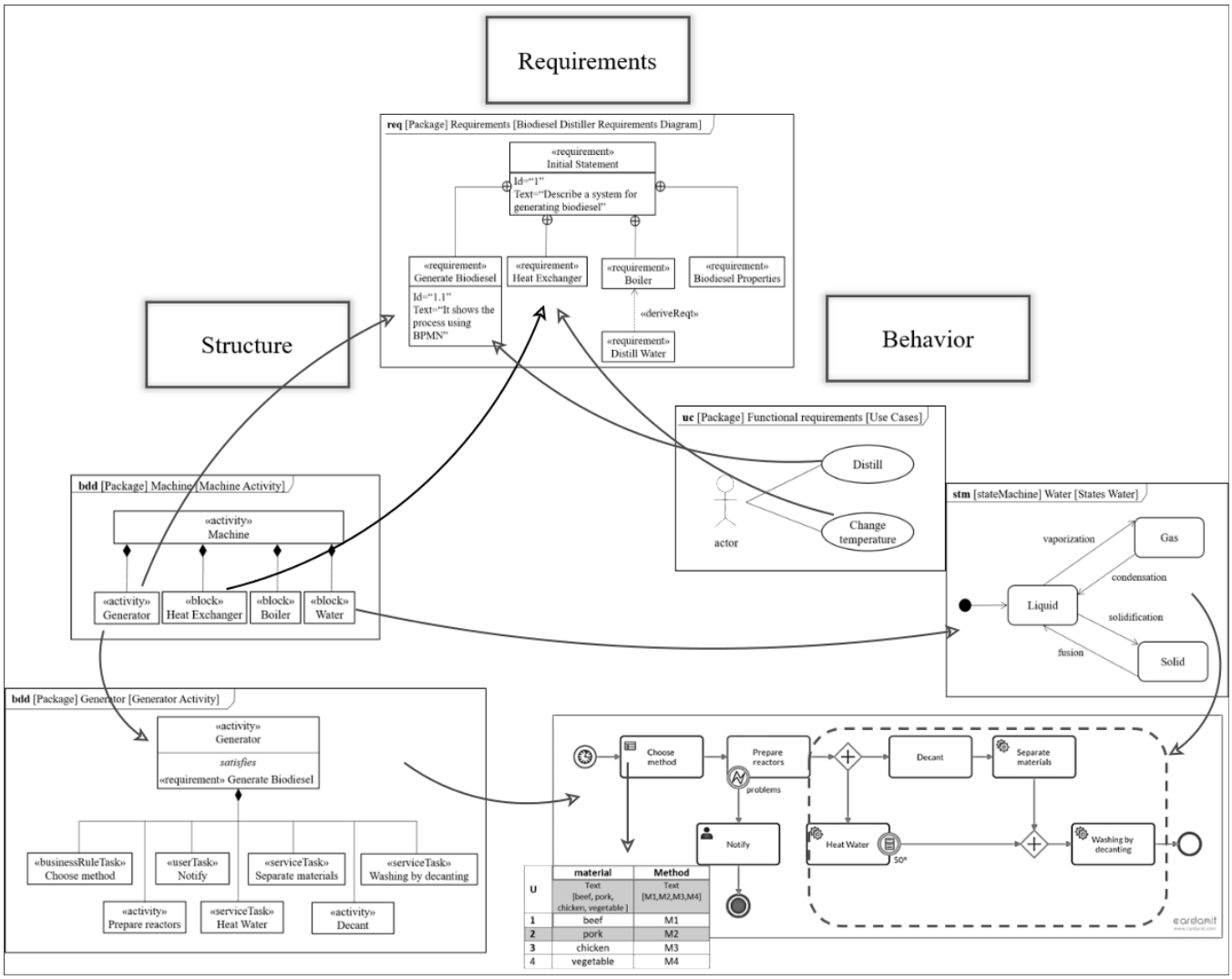

Figure 15. Traceability study case.

The Structure pillar contains two block definition diagrams: the *Machine Activity* block definition diagram and the *Generator Activity* block definition diagram. The *Machine Activity* block definition diagram has four relations. One of them shows the relationship between *Heat Exchanger* block and *Heat Exchanger* requirement through a *satisfy* relationship, as mentioned before. The other relation illustrates the relationship between *Generator Activity* and *Generate Biodiesel* requirement through a *satisfy* relationship, also this diagram has another relation toward *Generator Activity* block definition diagram, which shows how this activity is decomposed by using the extension proposed in this article. The fourth relation shows how *Water* block is related with states machine diagram in Behavior pillar. The *Generator Activity* block definition diagram has a relationship with BPMN model because of model illustrates the process, which is carried out to generate biodiesel associated with the *Generator* activity.

The Behavior pillar contains three diagrams. A use cases diagram shows the use cases that refine some requirements in the requirement diagram presented, in other words, *Distill* use case refines *Generate Biodiesel* requirement, and *Change temperature* use case refines *Heat Exchanger* requirement. The other diagram is *States Water* state machine diagram, which is used to describe the possible states of *Water* block in *Machine Activity* block definition diagram being these possible states used in some activities in BPMN model, as mentioned before. These activities are grouped in the model (rounded corner rectangle with a solid dashed line). Finally, this pillar has a BPMN model, which shows the process to generate biodiesel as can be observed in Figure 15 and mentioned before. In addition, the same figure shows the decision model related to the Choose method business rule task.

VI. CONCLUSION

Traceability plays an important role in any Model-Based Systems Engineering methodology. This methodology emphasizes the use of models to perform the systems engineering activities. The objective of this work has been to carry out requirements traceability understanding that requirements traceability is the capability to follow the life-cycle of the requirement. SysML is a general-purpose modeling language, based on UML, which enables traceability because it defines relationships between requirements and other modeling elements. The combination of SysML and BPMN-DMN is attractive and is a step forward that enhances the modeling of the different views of the system to be built including decision requirements. The proposed approach uses the definition of new stereotypes in SysML to support all types of BPMN tasks.

This proposal seeks to integrate and cover the different views of all requirements, helping systems engineers to improve the design of them.

The approach was illustrated through a case study to show how traceability of requirements can be performed.

In future work, we consider analyzing the link among the DMN decision requirements diagram, SysML requirements diagram, and use cases diagram. In addition, we consider working with parametric diagrams in order to complete the pillars of SysML.

ACKNOWLEDGMENT

The authors thank the reviewers of the ICSEA'19 conference and the IARIA Journal for the very useful comments that have contributed to enhance both the original and the extended versions of the paper.

REFERENCES

[1] C. Abdelahad, D. Riesco, and C. Kavka, "A SysML-based Approach to Requirements Traceability using BPMN and DMN," ICSEA, pp. 210-216, 2019.

[2] INCOSE https://www.incose.org/ [retrieved: December, 2020].

[3] J. Jacobs and A. C. Simpson, "Towards a process algebra framework for supporting behavioural consistency and requirements traceability in SysML," in Proceedings of the 15th International Conference on Formal Engineering Methods (ICFEM 2013), Lecture Notes in Computer Science, Springer, vol. 8144, pp. 265-280, 2013.

[4] OMG https://www.omg.org/ [retrieved: December, 2020].

[5] SysML https://www.omg.org/spec/SysML/1.6/PDF [retrieved: December, 2020].

[6] S. Friedenthal, A. Moore, and R. Steiner, "A Practical Guide to SysML: The Systems Modeling Language," Elsevier, 2014.

[7] P. Spoletini and A. Ferrari, "Requirements elicitation: a look at the future through the lenses of the past," In 2017 IEEE 25th International Requirements Engineering Conference (RE), IEEE, pp. 476-477, 2017.

[8] Modelica https://www.modelica.org/ [retrieved: December, 2020].

[9] Business process model and notation https://www.omg.org/spec/BPMN/2.0.2/PDF [retrieved: December, 2020].

[10] D. Q. Birkmeier, S. Klöckner, and S. Overhage, "An Empirical Comparison of the Usability of BPMN and UML Activity Diagrams for Business Users," ECIS 2010 Proceedings 51, 2010.

[11] C. V. Geambaşu, "BPMN vs UML activity diagram for business process modeling," Accounting and Management Information Systems vol. 11, no. 4, pp. 934-945, 2012.

[12] Decision Model and Notation https://www.omg.org/spec/DMN/1.2/ [retrieved: December, 2020].

[13] E. J. Vidal and E. R. Villota, "SysML as a Tool for Requirements Traceability in Mechatronic Design," In Proceedings of the 2018 4th International Conference on Mechatronics and Robotics Engineering, ACM, pp. 146-152, 2018.

[14] Z. H. U. Shaofan, T. A. N. G. Jian, J. M. Gauthier, and R. Faudou, "A formal approach using SysML for capturing functional requirements in avionics domain," Chinese Journal of Aeronautics, vol. 32, no. 12, pp. 2717-2726, 2019.

[15] R. Baduel, M. Chami, J. M. Bruel, and I. Ober, "SysML Models Verification and Validation in an Industrial Context: Challenges and Experimentation," In European Conference on Modelling Foundations and Applications. Springer, Cham, pp. 132-146, 2018.

[16] A. Salado and P. Wach, "Constructing True Model-Based Requirements in SysML," Systems, vol. 7, no. 2, 21 pages, 2019.

[17] K. Gruber, J. Huemer, A. Zimmermann, and R. Maschotta, “Integrated description of functional and non-functional requirements for automotive systems design using SysML,” 2017 7th IEEE Int. Conf. on System Engineering and Technology (ICSET), pp. 27-31, 2017.

[18] M. R. S. Marques, E. Siegert, and L. Brisolara, “Integrating UML, MARTE and SysML to improve requirements specification and traceability in the embedded domain,” In 2014 12th IEEE International Conference on Industrial Informatics (INDIN), IEEE, pp. 176-181, 2014.

[19] H. Dubois, M. A. Peraldi-Frati, and F. Lakhal, “A model for requirements traceability in a heterogeneous model-based design process: Application to automotive embedded systems,” In 2010 15th IEEE International Conference on Engineering of Complex Computer Systems, IEEE, pp. 233-242, 2010.

[20] I. Sommerville, “Software Engineering,” Seventh Edition, Pearson Education, 2004.

[21] J. Holt and S. Perry, “SysML for systems engineering,” vol. 7, IET, 2008.

[22] T. Weilkiens, “Systems engineering with SysML/UML: modeling, analysis, design,” Elsevier, 2011.

[23] O. C. Z. Gotel and A. C. W. Finkelstein, “An Analysis of the Requirements Traceability Problem,” Proc. IEEE Int. Conf. on Requirements Engineering, pp. 94-101, 1994.

[24] K. Hampson, “Technical evaluation of the systems modeling language (SysML),” Procedia Computer Science, vol. 44, pp. 403-412, 2015.

[25] L. Delligatti, “SysML distilled: A brief guide to the systems modeling language,” Addison-Wesley, 2013.

[26] http://www.omgsysml.org/INCOSE-OMGSysML-Tutorial-Final-090901.pdf [retrieved: December, 2020].

[27] UML 2.4 “Infrastructure Specification” https://www.omg.org/spec/UML/2.4.1/ [retrieved: December, 2020].

[28] G. Booch, J. Rumbaugh, and I. Jacobson, “The Unified Modeling Language User Guide,” Addison-Wesley, 1999.