A SEMI-AUTOMATED REQUIREMENTS REUSE AND RECYCLING PROCESS FOR AUTONOMOUS TRANSPORTATION SYSTEMS R&D

Damak, Youssef (1,2); Jankovic, Marija (1); Leroy, Yann (1); Chelbi, Karim (2)

1: CentraleSupélec; 2: AKKA Technologies

ABSTRACT

The R&D of Autonomous Transportation Systems (ATS) is hindered by the lack of industrial feedback and client's knowledge about technological possibilities. In addition, because of intellectual properties (IP) issues, technology consulting companies can't directly reuse developed functionalities with different clients. In this context, requirements reuse technics presents a good way to capitalize on their knowledge while avoiding IP issues. However, the literature review on requirements reuse processes doesn't propose methods to the application of reuse processes with little information about the system's operational context. In this paper, we present a semi-automated requirement reuse and recycle process for ATS R&D. The process helps designers' copes with the lack of inputs from the clients. Requirements candidates are retrieved from a database using Natural Language Processing and traceability propagation. It is applied to 3 use cases with inputs less than 5 concepts from the client's needs. The results validate its efficiency through number requirements retrieved and the analysis time consumption

Keywords: Requirements, Autonomous Transportation Systems, Early design phases, Process modelling

Contact:
Damak, Youssef
CentraleSupélec
Laboratoire Génie Industriel
France
youssef.damak@centralesupelec.fr

Cite this article: Damak, Y., Jankovic, M., Leroy, Y., Chelbi, K. (2019) 'A Semi-automated Requirements Reuse and Recycling Process for Autonomous Transportation Systems R&D', in *Proceedings of the 22nd International Conference on Engineering Design (ICED19),* Delft, The Netherlands, 5-8 August 2019. DOI:10.1017/dsi.2019.362

1 INTRODUCTION

Autonomous Transportation Systems (ATS) are perceived as a pillar of future mobility by all the mobility's stakeholder. Hence, Research & Development (R&D) focus on the design and industrialization of ATS is greatly increasing. For instance, carmakers are racing to be the first to industrialize a safe Autonomous Vehicle (AV). In this effervescent context, technology consulting companies help their clients conducting their R&D for ATS. Having a large spectrum of client's industry, the consulting companies face the R&D of various contexts and needs. However, much of the explored solutions overlap, as the different ATS share many characteristics.

Up until now, there have been no industrialization or industrial experimentation of an ATS with an autonomy level 4 or above on the SAE scale. A shown in Damak *et al*. (2018), the transition from the level 2 and 3 to upward levels implies a structuring change on the system's requirements. Therefore, designers face a lack of industrial feedback for R&D of level 4 ATS. For this particular reason, the rich experience and expertise in technology consulting companies are highly needed to accelerate the advances in the design of ATS. However, the companies are bound with intellectual properties issues and cannot directly share and reproduce developments from a client to another. Furthermore, ATS being context sensitive, the designers have to adapt developed and known solutions and manage the context changes impact (Horváth, 2014).

Requirements reuse is a quiet standard activity in the requirements engineering process. In early design stage, it helps designers identify reusable solutions through the analysis of shared requirements and context elements. In addition, the identified differences also indicate the changes needed in known solutions. Besides, Requirements reuse allows for system design feasibility assessment without direct solution reuse. This would avoid intellectual properties issues.

This research work proposes a semi-automated process of reusing and recycling the consulting companies' knowledge about ATS through the reuse and recycling of requirements. We propose the following structure of the paper. The second part reviews the literature of requirements reuse and recycling in requirements engineering. The proposed process is then presented in the third part. A case study in the fourth part is presented to validate its efficiency in the context of ATS design before discussing perspective in the fifth part.

2 LITERATURE REVIEW

There are many definitions of requirements from different standards of different communities such as the INCOSE, ISO, and IEEE. In this study, ATS requirements are expressed by the clients and experts of the domain in an R&D context. They express expected operational and functional properties for the ATS and are not always measurable. In the remaining of the paper, we consider the following definition of a requirement that best corresponds to the context of the study: "the definition of a property of a system that is either needed or wanted by a stakeholder" (Holt *et al*., 2012). The following sub-parts review the literature of requirements elicitation in requirements engineering process, more specifically, Requirements Reuse (RR) strategies and methods.

2.1 Requirement engineering

Requirement Engineering (RE) is a set of engineering activities throughout the design phases that starts with the elicitation of clients' needs and requirements up to the validation & verification of these requirements by the designed system. RE activities are generally defined as follows (Berkovich *et al*., 2011; Jiao and Chen, 2006):

- Stakeholders' requirements elicitation,
- Stakeholders' requirements analysis,
- Requirements specification,
- Requirements change management.

This paper focuses on the activities of requirements elicitation and analysis in ATS R&D. In the requirements analysis activity, Model-Based Requirements Engineering (MBRE) has become a widespread approach (Holt *et al*., 2012; Scherer *et al*., 2017). Requirements modelling consists in representing the requirements attributes, description, identifier, and possible others, and its relation to

other requirements and system's functions. The principle sub-activities of requirements analysis consist of modelling and classifying system requirements and detecting requirement conflicts.
Prior to requirements modelling and analysis, the system's requirements have to be elicited. Requirements elicitation consists of the following activities: research, discover, identify and elaborate client requirements. Zhang (2007) identified in the literature 4 types of requirements elicitation method: conversation, observation, synthetic and analytic methods.
The two first methods, conversation and observation methods have in the case of ATS R&D a low efficiency in eliciting client's requirements. In fact, the technics used in these methods such as interviews, workshops and users observations are hindered by the lack of the client's knowledge about the current technological possibilities in the design of ATS (Curcio *et al*., 2018). As for synthetic technics such as scenarios, storyboarding and prototyping, they are quite efficient and often used in the context of R&D (Sutcliffe, 2003). However, they are not best suited to identify reusable development and solutions, as well as making the best use of previous experiences for the design of emergent systems such as the ATS.
Lastly, analytic technics consist of documentation studies and requirement reuse technics. They are efficient for the development of systems that share similar contexts and characteristics such as modular systems and product lines (Adam and Schmid, 2013). In the context of low knowledge feedback and a need of previous experience, RR technics seem to be better suited to the efficient reuse and recycling of developed solutions.

2.2 Requirements reuse

Requirements Reuse in Requirements Engineering consists in selecting requirements from defined and verified system requirements of a previous project to use them in a new project. It is used to reduce development time & cost and increase the productivity and quality of products. It is particularly useful to help stakeholder rapidly elicit a system's requirements, especially when their knowledge about the current technological possibilities is lacking (Pacheco *et al*., 2015; Toval *et al*., 2002).
RR methods and technics have been greatly explored in the software domain. The literature about software RR shows that all RR processes are based on the selection of requirements candidates from a requirement database (Irshad *et al*., 2018). Moros *et al*. (2008) introduce a method for modelling software requirements *for reuse*, then modelling new software requirements *by reuse*. The requirements for reuse are classified in catalogues while ensuring the traceability and relations between the requirements. In their cases, the catalogues are software functionalities. When such functionalities are needed in new projects, the requirements corresponding to these functionalities are selected for reuse (Moros *et al*., 2008).
Prior to functionality identification, stakeholders' needs are the inputs of design processes and RE processes. Other methods in the literature aim at identifying reusable requirements before the analysis and identification of system's functionalities. For instance Kaiya and Saeki (2006) map the requirement database with domain ontologies. They match concepts expressed in the stakeholders needs with concepts from the domain ontology to identify missing reusable requirements from system's requirements.
Whether the input for RR is system functionality or stakeholder need, direct reuse of requirements is not always possible. When system functionalities are reused in new contexts and integrated to new system functionalities, the description of their requirements must be adapted. This activity is called requirement recycling. It can be defined as keeping the suitable parameters in the base of the requirement description, adapting the other parameters to the new context and integrating the resulting requirement to the new system's requirements (Alexander and Kiedaisch, 2002).
Few propositions have been made for parameter recycling. Knethen *et al*. (2002) propose a systematic process to identify requirement's recycled parameters. For that, they use abstractions of the database requirements in the form of templates. They map these templates with conceptual model of the system. And finally, they use the differences between the former and new systems' conceptual models to deduce the change in requirements parameters. Quite similarly, Alexander and Kiedaisch (2002) map requirements with use cases. Then, in the same fashion, they use the changes is the use cases to deduce the change in requirements parameters.
On the other hand, Toval *et al*. (2002) recommend the inclusion of the stakeholders in the RR process for requirements recycling through parameters analysis and negotiation. The inclusion of stakeholders ensures a better consistency of new parameters elicitation for requirements to be recycled. It also

serves the elicitation and integration of new requirements which is an important activity for the success of an RR process.
By contrast to software systems, ATS functionalities and capabilities implicate heterogeneous interaction and complex interfacing of ATS components. The reuse of an ATS functionality in a different ATS includes the integration of developed components with new system components. And due to the changes in the ATS physical properties and heterogeneous interactions, the reused components may necessitate important adaptation. Besides, as stated previously, RR is important in ATS R&D to overcome the lack of the client's knowledge and his inability to explicitly describing his needs and the ATS operational context. Therefore, relying on functionality reuse for RR is not suited for ATS R&D case and requirement recycling is necessary. For requirement recycling, using system's conceptual models or use cases to deduce parameters modification is also incompatible. Hence, as recommended by Toval *et al*. (2002), including stakeholders in the recycling seem to be better suited. However, to the best of our knowledge, no process in the literature allows efficient RR and recycling with little information about stakeholders' needs and the system's operational context.

3 REQUIREMENTS REUSE & RECYCLING PROCESS

As stated previously, ATS are context sensitive and emergent systems. Currently, designers and engineers have access to very little industrial and knowledge feedback. To implement an efficient reuse of ATS R&D knowledge the authors propose a semi-automated process for reusing and recycling ATS requirements. This process permits the identification of relevant requirements from former ATS R&D projects with little information about the new ATS's operational context. It also includes the clients in the RR process to improve his ATS domain knowledge to improve the process' accuracy.
In this part, we propose a requirements elicitation process through Requirements Reuse and Recycling (RRR) process. The process in Figure 1 is divided into 3 main sub-processes. First and prior to any new ATS R&D project, a database of former projects' system requirements is built. The requirements in the database are modelled *for reuse*. Second, requirement candidates are semi-automatically selected from the database then reused or recycled. Clients are integrated to the reuse and recycling activity. Third, project specific requirements are elicited with the client and integrated to the reused and recycled requirements. Finally, at the end of the whole requirements engineering process, the final ATS requirement is used to update the requirements database. Maintaining the database for future elicitation processes is one of the RRR challenges (Toval *et al*., 2002).

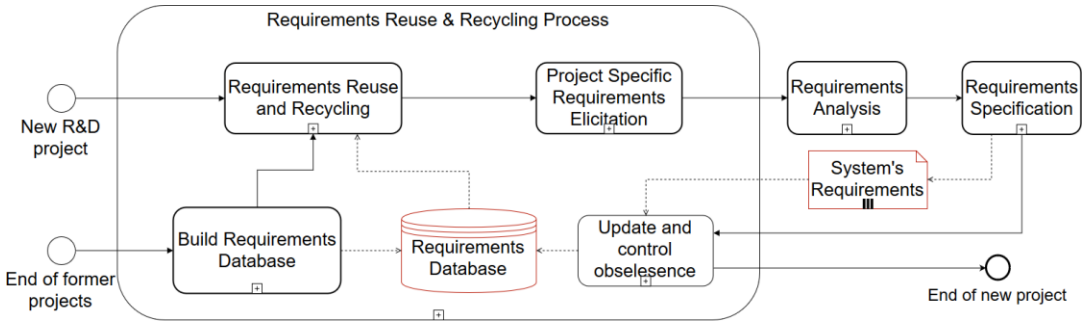

Figure 1: Requirements Reuse and Recycling process

3.1 Requirement database building

To model complex systems requirements *for reuse*, we propose a modelling framework including the requirement types, the relations between requirements and the semantic structure of the requirement. To create the requirements database, we apply this modelling framework on the system requirements of previous projects using graph modelling and processing with python:

3.1.1 Requirement types and hierarchy

The literature contains various types of requirements that differ between domains and uses. The most common types are Operational, Functional, Non-functional and Business Requirements (Holt *et al*., 2012). During our industrial observations of ATS R&D, the following 3 requirement types were recurrently expressed and used by clients, experts and other stakeholders:

- Non-Functional Requirements (NFR): an NFR express a global constraint on the system and the other types of requirements. It expresses "ilities" such as quality, safety, maintainability, etc.
  NFR example: The safety of the platooning_system must always be ensured
- Operational Requirements (OR): an OR expresses a behavioural requirement that the system must satisfy. It shouldn't indicate any functional solution to produce the intended behaviour
  OR example: The platooning_system must start if the START_command is activated from HMI
- Functional Requirements (FR): an FR expresses a function that must be fulfilled by a part of the system. Through the design process, the ATS functions are defined to satisfy ORs and FRs. More detailed FRs can be derived from these functions and their interactions.
  FR example: The communication_network must ensure the transfer of the START_command

During the design process, designers may have to decide between different solution alternatives to satisfy a system requirement. As stated previously, more detailed system requirements can be derived from each solution. This results in several alternatives of requirement sets that satisfy the system requirement. In the database, we model these requirement alternatives using decision gates. As illustrated in Figure 2, the decision gate indicates two alternatives to satisfy OR1: FR1 or FR2. During the design process, designer will have to decide between satisfying FR1 or FR2.

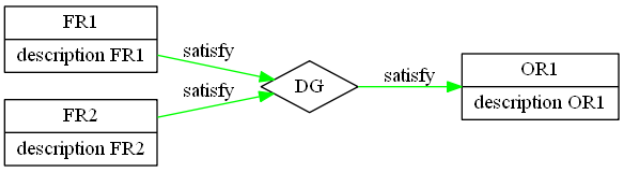

Figure 2: Decision gates in the requirements database

3.1.2 Relations between requirements

In the 3.2, we detail the identification of requirement candidates for reuse or recycling. The links between requirements is used for one of the ways to do so. According to Winkler and von Pilgrim, the definition of the traceability links depends on the analyst's interpretation (Winkler and von Pilgrim, 2010). Same as for requirements types, we identified the most relevant and used requirements links during our industrial observation of ATS R&D projects. In the database we model the following links:

- "Refine" link: the requirements R1.1 and R1.2 refine R1 when R1, being complex, is broken down into smaller and more manageable requirements R1.1 and R1.2.
- "Satisfy" link: The requirement R2 satisfies R1 when R2 is a lower level requirement that has been defined for the system to satisfy R1.
- "Evolve to" link: The requirement R1 evolve to R2 when R2 is a newer version of R1

The traceability links are also deeply connected to the hierarchical structure of a requirements system. According to these definitions, the "Refine" and "Satisfy" define the requirements hierarchy. We can also notice that the "Refine" link conserves the requirement type, while the level and type transition of requirements are modelled through the "Satisfy" link. As an example, an FR can "satisfy" an OR while an NFR can be satisfied by both ORs and FRs. This hierarchical structure verifies the key K2.

3.1.3 A formal semantic structure

The use of a formal semantic structure facilitates the identification of requirements candidates with the little information obtained from the clients. It also conserves requirements consistency while recycling their parameters for database update (cf. 3.2). In addition, it reduces the expression ambiguity of the requirement's description. To obtain the parameters of the formal semantic structure that would capture the complex aspects of ATS, we based it on a proposal for formal modelling of CPS properties (Garro *et al.*, 2016):

- **What** is to be satisfied;
- **When** in time that is to be satisfied;
- **Which** entity in the system that is to be satisfied;
  - <source object>: the entity that satisfies the requirement
  - <Target object>: the entity that is the target of the requirement
- **How well** that is to be satisfied

The requirement must express: an entity (**Which** <source object>), that must, have to, could, or should satisfy something (**what)**. In some complex requirements, this action is applied on/to a second entity (**which** <target object >) and can be satisfied only in a certain condition (**when**) with a certain quality (**how well**). Hence, the requirement description structure is as follows:

Which <source object> <must/have to/could/should> **what**
[<on/to> **which** <target object >], [**when** and **how well**]

As an example, the following requirement illustrates the different structure's elements:

The follower_vehicle (**which** <source object>) must send a warning_message (**what**) to the fleet_management_system (**which** <target object >), if a failure is detected (**when**)

3.2 Requirements reuse & recycling

In part 2, Requirements candidates' identification for reusing and recycling was conducted through functionalities, use cases and domain ontologies. In The case of ATS R&D, the lack of information on the client's need and the system's operational context prevent the use of the two first. When browsing the database, the analysis of the requirement's parameters is the only way of identifying potential requirements candidates. In our proposed process, we identify requirements candidates by matching a few key concepts expressed by the stakeholders with the parameters from the formal structure of requirements' descriptions. This step is automated using Natural Language Processing (NLP) and matching concepts. The used technic is out of the scope of this paper and in the remaining of the paper, we consider that we matched key concepts from stakeholders with some of the requirements' parameters. The requirements containing these parameters are the first set of requirements candidates to be selected. From this set, we use their traceability links to other requirements in the database to retrieve more requirements candidate. Further details about this step come in the next paragraph.

The results of the literature review showed that including stakeholders in the requirement reuse and recycle process is more adapted to ATS R&D. However, if the overall candidates' selection results in a considerable number of candidates, it would be complicated and time consuming for the clients and requirements engineers to process each one of them. We propose in this process to prioritize the set of requirements candidates that are proposed to them. With these successive sets, the engineers select with the clients what requirements are relevant for the project. The first set of candidates retrieved through NLP is the most relevant set of requirements to be processed. Its requirements have direct semantic links to the clients need. We classify them as requirements of the category C1.

One by one, the C1 requirements are automatically proposed to the engineers and clients, and they manually chose to reuse, recycle or discard it. If they chose to recycle the requirements, they arbitrarily change the parameters of the requirement's semantic structure. The structure is conserved and its parameters are updated. After processing all C1's requirements, the remaining ones are used to retrieve more candidates through their traceability link.

The next challenge is to determine which set of requirements has the highest priority to be processed. A requirement can be linked to other requirements with the "satisfy" or "refine" links. It can satisfy/refine or be satisfied/refined by other requirements. At this point, the category C1 contains the first set of reused/recycled requirements. The requirements must be realised by the system. It is then logical to analyse what are the requirements that satisfy or refine this set of requirements. Hence, the next set of candidates is composed of the requirements that satisfy or refine C1 requirements. They form the category C2. We call the process to retrieve C2 requirements *backward traces propagation*. On the other hand, it is also important to know the purpose of the reused/recycled requirements. What they satisfy or refine represent the reasons why such requirements exist in the first place. Hence, the third set of candidates is composed of the requirements that are satisfied or refined by C1 requirements. They form the category C3. We call the process to retrieve C3 requirements *frontward traces propagation*. C2 and C3 requirements are then successively processed to be either reused, recycled or discarder.

Following this reasoning, we propose a tree structure for the requirements candidates' categories. As illustrated in Figure 3, the root of the tree is the category C1. Each node of the tree has two child nodes: the left child node represents its sub-C2 category and is filled through *backward traces propagation* on its parent nodes requirements. The right child node represents its sub-C3 category and is filled through *frontward traces propagation* on its parent nodes requirements. The generation of categories stops when no more requirements candidates are detected in the database.

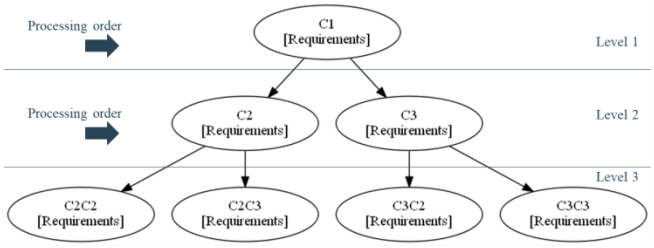

Figure 3: Candidates Category Tree

Figure 4 illustrate an example of the filling of the requirements category tree from a small database of 16 requirements. In his example, FR1 is detected through NLP. For the sake of the example FR8 was supposed to be discarded by the client when proposed. Through this example, we can notice several characteristics of this process. First, all the requirements linked to FR8 where not proposed to be reused or recycled. In addition, FR16 was isolated, thus couldn't be reached by the traceability propagation process. Besides, the decision gate between FR4 and FR5 was conserved to warn the requirements engineer. Later in the project, a decision must be taken to satisfy either FR4 or FR5. Finally, we can notice that once a requirement has been processed in a category, it does never appear in lower level categories, even if it has been discarded.

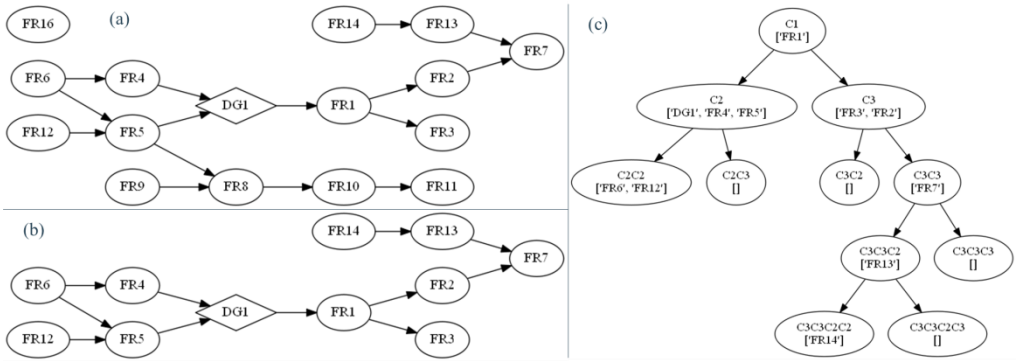

Figure 4: Example of requirements candidates' selection: (a) Requirements database graph; (b) reused/recycled requirements graph; (c) candidates category tree

3.3 New requirements elicitation and requirements integration

The requirements reused and recycled in the previous step are a base for discussion with the client. In fact, during the previous step, the client's ATS domain knowledge should have increased through the analysis and negotiation of reused and recycled requirements. With a better understanding of the technological possibilities, the client is more capable to express his requirements. Hence, using reused and recycled requirements as a base, requirements engineers manually elicit with the client missing ones to complete the system's requirements. In this task, they are free to use any elicitation technic to complete and clarify the requirements.

Newly elicited requirements are integrated to the reused and recycled requirements through the traceability links analysis. The global requirements engineering process continues with the result of this sub-process. Requirements feasibility and analysis, conflict management and requirements specification are conducted until the final system's requirements are clearly defined.

3.4 Database update

As stated in the beginning of part 3, the requirement database maintenance for future elicitation processes is one of the RRR challenges (Toval *et al.*, 2002). The final sub-process's aim is the update of the database. Throughout the ATS R&D project, the requirement's system evolves drastically. The final version of the system' requirement is used to update the requirements database. For the update, the following steps are automatically applied to the database:

- Recycled requirements are added to the database with an "evolve to" link from the older version to this new version,
- New elicited requirements are added to the requirements database with their corresponding links to the recycled requirement,

- A conflicts analysis is conducted to avoid integrating conflicting requirements to the database. Each new requirement is manually checked with the requirements that satisfy the same upper level requirements. If conflict is detected between two requirements, a decision gate is included.

4 CASE STUDY

To validate our process, we tested this RR process with the autonomous systems team of AKKA Technologies. We built the requirements database from a former project of platooning system design. The platooning system is an ATS composed of a lead vehicle, driven by an operator, and followed by a number of autonomous follower vehicles. The team is developing a platooning system in collaboration with other companies for the integration on a test platform and the development of communication network between the vehicles. From a part of this project, we built a database with 2 NFR, 22 OR and 76 FR and one decision gate. The database's requirement graph is illustrated in Figure 5. The red links represent "refine" links while the green links represent "satisfy" links.

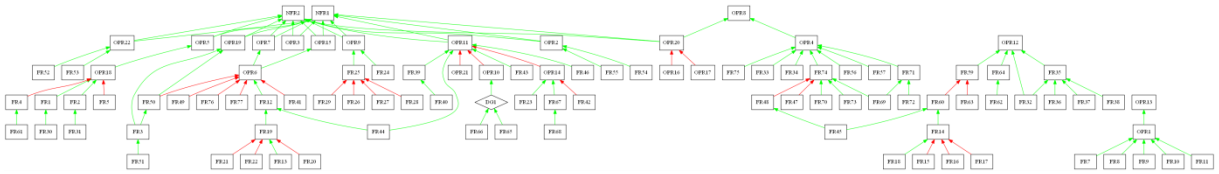

Figure 5: Sample of the platooning system's requirements database graph

3 tests were conducted to validate our process. For each case, around an hour was taken for the reuse, recycling, and discard analysis. The results are summarised in Table 1:

- Use case 1: A new client asks for the R&D of a new platooning system. Some aspects of the operational context were different from the former platooning project, but we used 3 key concepts expressed by the client: "follower_vehicle", "lead_vehicle", and "platoning_system". In this use case, we could retrieve all the requirements from the database. Interestingly, the category C1 contained half of the database. Even for a new project very close to the former one, we could eliminate half of the database from the initial discussion to avoid overwhelming the client.
- Use case 2: A new client asks for the R&D to automate a vehicle. In its operational context, it should be able to handle an intersection. Hence we used the following 3 key concepts for candidates' selection: "intersection", "vehicle", and "autonomous".
- Use case 3: The use case 2 client added new information to the operational context. The vehicle must activate an emergency stop if it encounters an obstacle on its way. Hence, we added two more key concepts and used the following concepts for candidates' selection: "intersection", "vehicle", "autonomous", "obstacle", and "emergency_stop"

Table 1: Use cases results

	Use case 1	Use case 2	Use case3
Number of C1 requirements	51	9	11
Number of C2 and children requirements	14	10	12
Number of C3 and children requirements	35	25	22
Total of reused/recycled requirements	100	44	45

The results of Table1 show that in an hour and with as little inputs as 3 concepts, we could reuse and recycle up to 44 requirements from a database of 100 requirements. The categorization of the candidates helped the engineers handle the important number of requirements and progressively improved their understanding of the clients' needs. However, we can also notice from the table that using 2 more concepts, in this case, didn't drastically change the number of reused/recycled requirements. Although the number didn't change, Figure 6 shows that the categories, and the process by extension, were impacted. We conclude that more information changes the priority of the system's requirements.

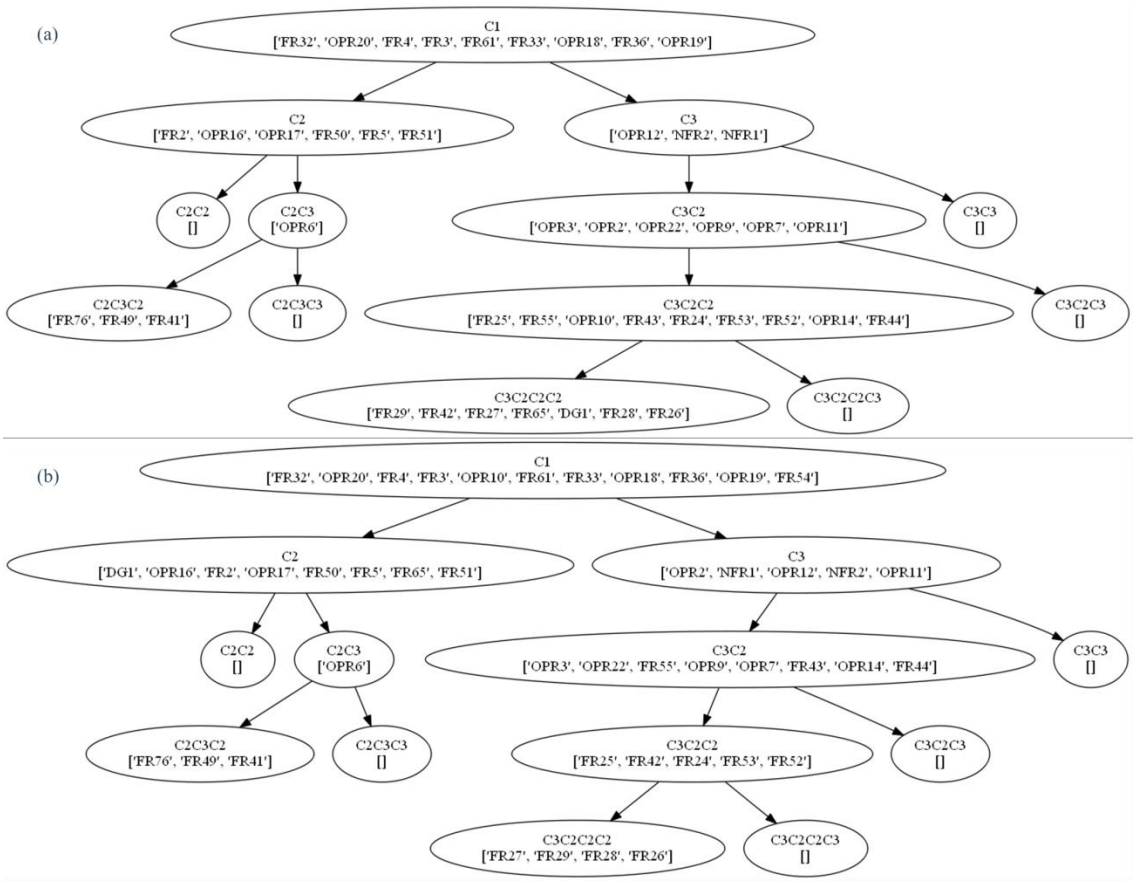

Figure 6: Reused/recycled requirements categories: (a) use case 1; (b) use case 2

5 CONCLUSION

Requirements elicitation in ATS R&D presents many challenges. The lack of industrial feedback, and client's knowledge about technological possibilities prevent the usual elicitation methods. to accelerate this process, technology consulting companies can reuse former knowledge on ATS from different industry. However, direct reuse of functionalities and technical solutions is hinder by complex integration and intellectual property issues. Nonetheless, requirements reuse and recycling seems to be on the best ways of reusing former knowledge while avoiding these issues. Yet, to the best of our knowledge, the literature on requirements reuse doesn't cover the lack of inputs on the new operational context as well as the structural complexity of ATS.

In this paper, we propose a requirement reusing and recycling process that deals with the mentioned challenges. Through a formal semantic modelling of the requirements and a structured database, we use NLP to identify requirements candidate to be reused. We prioritize the candidate and classify them in categories. The prioritization allows requirements engineer to implicate the stakeholders in the reuse, recycling or discarding analysis of requirements candidates. By prioritizing the candidates, stakeholders' knowledge about the system increase progressively. Finally, the maintenance of the database is ensured through tracing the recycled requirements to the original ones. The formal semantic structure also helps conserving the database consistency while requirements parameters are recycled. We validated this approach with a comparative study on 3 use cases. The results show the efficiency in the number of reused/recycled requirements and the time of processing. With little information about the operation context, we could reuse and recycle more than 40% of the requirements database.

Although efficient, this process focusses on the recycling of requirements parameters. In our proposition, we do not consider the recycling of attributes such as requirements maturity, criticality, priority, etc. In addition, we do not control the obsolescence of requirements in the database. In future work, we should focus on improving these elements. Besides, the process should be tested and validated on a bigger scale and with a database combining several project.

REFERENCES

Adam, S. and Schmid, K., (2013). *Effective Requirements Elicitation in Product Line Application Engineering – An Experiment, in: Requirements Engineering: Foundation for Software Quality*. Springer Berlin Heidelberg, pp. 362–378. https://doi.org/10.1007/978-3-642-37422-7_26

Alexander, I. and Kiedaisch, F. (2002), "Towards recyclable system requirements", *Proceedings Ninth Annual IEEE International Conference and Workshop on the Engineering of Computer-Based Systems*, pp. 9–16. https://doi.org/10.1109/ECBS.2002.999817.

Berkovich, M., Leimeister, J.M. and Krcmar, H. (2011), "Requirements Engineering for Product Service Systems", *Bus. Inf. Syst. Eng*, Vol. 3, pp. 369–380. https://doi.org/10.1007/s12599-011-0192-2.

Curcio, K., Navarro, T., Malucelli, A. and Reinehr, S. (2018), "Requirements engineering: A systematic mapping study in agile software development", *J. Syst. Softw*, Vol. 139, pp. 32–50. https://doi.org/10.1016/j.jss.2018.01.036.

Damak, Y., Jankovic, M., Leroy, Y. and Yannou, B. (2018), "Analysis of Safety Requirements Evolution in the Transition of Land Transportation Systems Toward Autonomy", *Presented at the 15th International Design Conference*, pp. 2845–2854. https://doi.org/10.21278/idc.2018.0448.

Garro, A., Tundis, A., Bouskela, D., Jardin, A., Thuy, N., Otter, M., Buffoni, L., Fritzson, P., Sjölund, M., Schamai, W. and Olsson, H. (2016), "On formal cyber physical system properties modeling: A new temporal logic language and a Modelica-based solution", *2016 IEEE International Symposium on Systems Engineering (ISSE)*, pp. 1–8. https://doi.org/10.1109/SysEng.2016.7753137.

Holt, J., Perry, S.A. and Brownsword, M. (2012), *Model-based requirements engineering*, IET.

Horváth, I. (2014), "What the Design Theory of Social-Cyber-Physical Systems Must Describe, Explain and Predict?", In: A. Chakrabarti and L.T.M. Blessing, (Ed.), *An Anthology of Theories and Models of Design: Philosophy, Approaches and Empirical Explorations*, Springer London, London, pp. 99–120. https://doi.org/10.1007/978-1-4471-6338-1_5.

Irshad, M., Petersen, K. and Poulding, S. (2018), "A systematic literature review of software requirements reuse approaches", *Inf. Softw. Technol*, Vol. 93, pp. 223–245. https://doi.org/10.1016/j.infsof.2017.09.009.

Jiao, J.R. and Chen, C.-H. (2006), "Customer Requirement Management in Product Development: A Review of Research Issues", *Concurr. Eng*, Vol. 14, pp. 173–185. https://doi.org/10.1177/1063293X06068357.

Kaiya, H. and Saeki, M. (2006), "Using Domain Ontology as Domain Knowledge for Requirements Elicitation", *14th IEEE International Requirements Engineering Conference (RE'06)*, pp. 189–198. https://doi.org/10.1109/RE.2006.72.

Knethen, A.v., Paech, B., Kiedaisch, F. and Houdek, F. (2002), "Systematic requirements recycling through abstraction and traceability", *Proceedings IEEE Joint International Conference on Requirements Engineering*, pp. 273–281. https://doi.org/10.1109/ICRE.2002.1048538.

Moros, B., Vicente-Chicote, C. and Toval, A. (2008), "REMM-Studio + : Modeling Variability to Enable Requirements Reuse", *Conceptual Modeling - ER 2008*, Springer Berlin Heidelberg, pp. 530–531. https://doi.org/10.1007%2F978-3-540-87877-3_46.

Pacheco, C.L., Garcia, I.A., Calvo-Manzano, J.A. and Arcilla, M. (2015), "A proposed model for reuse of software requirements in requirements catalog", *J. Softw. Evol. Process*, Vol. 27, pp. 1–21. https://doi.org/10.1002/smr.1698.

Scherer, H., Albers, A. and Bursac, N. (2017), "Model Based Requirements Engineering for the Development of Modular Kits", *Complex Syst. Eng. Dev. Proc. 27th CIRP Des. Conf*, Cranfield Univ. UK, 10th – 12th May 2017 60, pp. 145–150. https://doi.org/10.1016/j.procir.2017.01.032.

Sutcliffe, A. (2003), "Scenario-based requirements engineering", *11th IEEE International Requirements Engineering Conference*, 2003, IEEE, pp. 320–329. https://doi.org/10.1109/ICRE.2003.1232776.

Toval, A., Nicolás, J., Moros, B. and García, F. (2002), "Requirements Reuse for Improving Information Systems Security: A Practitioner's Approach", *Requir. Eng*, Vol. 6, pp. 205–219. https://doi.org/10.1007/PL00010360.

Winkler, S. and von Pilgrim, J. (2010), "A survey of traceability in requirements engineering and model-driven development", *Softw. Syst. Model*, Vol. 9, pp. 529–565. https://doi.org/10.1007/s10270-009-0145-0.

Zhang, Z. (2007), "Effective requirements development-A comparison of requirements elicitation techniques, in: Software Quality Management XV: Software Quality in the Knowledge Society (SQM2007)", *British Computer Society*, pp. 225–240.