Evaluation framework for autonomous systems: the case of Programmable Electronic Medical Systems

Andrea Bombarda, Silvia Bonfanti, Martina De Sanctis, Angelo Gargantini, Patrizio Pelliccione, Elvinia Riccobene, Patrizia Scandurra

Abstract—This paper proposes an evaluation framework for autonomous systems, called LENS. It is an instrument to make an assessment of a system through the lens of abilities related to adaptation and smartness. The assessment can then help engineers understand in which direction it is worth investing to make their system smarter. It also helps to identify possible improvement directions and to plan for concrete activities. Finally, it helps to make a re-assessment when the improvement has been performed in order to check whether the activity plan has been accomplished.
Given the high variability in the various domains in which autonomous systems are and can be used, LENS is defined in abstract terms and instantiated to a specific and important class of medical devices, i.e., Programmable Electronic Medical Systems (PEMS). The instantiation, called LENS_{PEMS}, is validated in terms of *applicability*, i.e., how it is applicable to real PEMS, *generalizability*, i.e., to what extent LENS_{PEMS} is generalizable to the PEMS class of systems, and *usefulness*, i.e., how it is useful in making an assessment and identifying possible directions of improvement towards smartness.

Index Terms—Autonomous systems, Evaluation framework, Programmable Electronic Medical Systems (PEMS).

1 Introduction

In the last years, we have observed active and proficuous research in autonomous and self-adaptive systems (SASs). The Software Engineering of Adaptive and Self-Managing Systems (SEAMS)[1] community produced two roadmaps to summarize the state-of-the-art, for identifying critical challenges for the systematic software engineering of SASs [1], [2]. The second roadmap [2] identified essential topics of self-adaptation, spanning from the design space to processes and to run-time verification and validation. The runtime assurance of SASs has been also targeted through the use of models at runtime (M@RT) [3]. There have also been survey papers aiming at identifying the underlying research gaps and providing a taxonomy of SASs [4], [5]. Finally, a recent book provides a historical perspective of SASs and presents the basic principles, engineering foundations, and applications of SASs [6].

Over the years, SASs are increasingly becoming "smarter" to be able to adapt and learn how to handle and manage new and unexpected events with autonomy. However, the precise meaning of "making a system smarter" is not always obvious, and, more pragmatically, it is not straightforward to decide how to concretely operate to achieve the ambition [7]. Making a system smarter might involve various system's abilities, such as configurability, autonomy, adaptability, perception, cognitive, and interaction with other systems and humans, to mention a few. These abilities can have various levels of importance in specific systems [7]. Then, it is important to understand in which direction it is worth and useful to invest to make systems smarter, and how and how much to improve a system in each specific direction.

In this paper, we aim to provide an instrument to help engineers understand (i) in which direction it is worth investing to make their system smarter, (ii) how to plan for concrete activities, and (iii) how to assess the execution of the plan. The instrument we propose is an evaluation framework for autonomous systems, which focuses on abilities related to adaptation and smartness. An evaluation framework is an instrument to perform an evidence-based assessment of a system under a specific lens (the topic of interest) and to monitor the evolution of the system over time. An example of existing evaluation frameworks is CMMI (Capability Maturity Model Integration),[2] originally developed for the U.S. Department of Defense. It has been used for more than 30 years to help organizations understand their current level of capability and performance and offer a guide to optimizing business results. Another example is the Family Evaluation Framework (FEF) [8],

A. Bombarda is with University of Bergamo, Bergamo, Italy - email: andrea.bombarda@unibg.it
S. Bonfanti is with University of Bergamo, Bergamo, Italy - email: silvia.bonfanti@unibg.it
M. De Sanctis is with Gran Sasso Science Institute (GSSI), L'Aquila, Italy - email: martina.desanctis@gssi.it
A. Gargantini is with University of Bergamo, Bergamo, Italy - email: angelo.gargantini@unibg.it
P. Pelliccione is with Gran Sasso Science Institute (GSSI), L'Aquila, Italy - email: patrizio.pelliccione@gssi.it
E. Riccobene is with Università degli Studi di Milano, Milano, Italy - email: elvinia.riccobene@unimi.it
P. Scandurra is with University of Bergamo, Bergamo, Italy - email: patrizia.scandurra@unibg.it

1. https://www.hpi.uni-potsdam.de/giese/public/selfadapt/seams/

2. https://cmmiinstitute.com/

which has been created for evaluating the performance in software product line engineering of organizations.

The evaluation framework we contribute in this paper is called LENS - evaLuation framEwork for autoNomous Systems - and it can be used for (i) making an assessment of a system under the lens of abilities related to adaptation and smartness, (ii) identifying the possible directions of improvement, and (iii) making a re-assessment when the improvement has been performed. LENS stimulates reasoning to determine which abilities are worth enhancing in a system, and which levels within an ability are suitable and optimal for a system (thus rejecting the idea that higher levels are always better). Then, it will make it possible to plan improvement steps and also define Key Performance Indicators (KPIs) to measure improvements in making systems smarter.

The preliminary idea of LENS has been presented in the short paper published at the ACSOS 2022 conference [7]. In the previous paper, we motivated the need for LENS, we described the process that we followed to create it, and briefly sketched the framework. In this paper, we describe LENS in detail, its instantiation to a specific domain, the tool supporting it, and its validation according to various validation questions. We also provide lessons learned that can support the process of making new customizations of LENS for other domains.

LENS is defined at an abstract level, since there is high variability in the various domains in which autonomous systems are and can be used. We are talking about a family of evaluation frameworks, where LENS is the abstract root and the various instantiations become concrete and ready to be used in practice. We instantiate LENS to a specific and important class of medical devices, and specifically on Programmable Electronic Medical Systems (PEMS) [9]. The concrete instance we obtained is called LENS_{PEMS}. The choice of this precise medical domain was due to the fact that when working on developing an adaptive version of a mechanical ventilator for pneumonia disease, we needed an instrument for evaluating the device in terms of adaptation and smartness and planning the required improvements. However, we were not able to find any instrument with the needed requirements, and this triggered the idea of doing it ourselves. Moreover, considering the lack of similar tools for autonomous systems, in general, in this paper we explain the steps to be performed in order to make an instantiation of LENS for a different class of systems.

We developed LENS and LENS_{PEMS} by exploiting the Multi-annual Robotic Roadmap [10] for robotic systems, since this roadmap identifies various abilities of autonomous robotics and defines levels for each of them. We only considered the abilities that make sense for autonomous systems without motion and manipulation, as these abilities are specific to robots and not inherent to the class of systems that we intended to evaluate. Moreover, note that LENS is an evolving meta-framework, open to inheriting abilities and/or sub-abilities elicited from the LENS's customization to specific classes of systems when those elicited abilities are also suitable for autonomous systems as a whole. We show such LENS's evolution upon LENS_{PEMS} definition, and we envision such kind of advancement due to customization to further classes of concrete autonomous systems.

To show how LENS_{PEMS} is applicable to real PEMS, we use it for evaluating the adaptive abilities of the Mechanical Ventilator Milano (MVM), a mechanical ventilator for COVID-19 [11], [12], [13][3]. It was developed by an international and multidisciplinary network of scientists spread all around the world during the first wave of the COVID-19 pandemic. Evaluating the MVM by means of LENS_{PEMS} has helped us to identify a number of possible improvements and to evaluate the engineering effort (from low to high) required to achieve them. This analysis allows engineers to design the next generation of mechanical ventilators that expose autonomous, adaptive, and learning abilities.

To evaluate the generalizability of LENS_{PEMS}, we identified five additional PEMS (in addition to the MVM) and evaluated changes, extensions, or customizations LENS_{PEMS} would need to become usable for a system belonging to the class of PEMS.

Finally, to assess the usefulness of LENS_{PEMS}, we gathered responses from 26 experts in self-adaptive systems and/or PEMS through a questionnaire we created. Additionally, we conducted 13 interviews to further clarify and supplement the questionnaire responses.

The paper is organized as follows. Section 2 provides background on (i) the Multi-annual robotic roadmap that is used as inspiration to build LENS and LENS_{PEMS} and (ii) the class of PEMS. Section 3 introduces LENS together with the steps that need to be performed to instantiate it to a specific class of systems. Section 4 presents LENS_{PEMS} and the tool supporting it. Section 5 reports the activities we performed to validate LENS_{PEMS}. Finally, Section 6 compares the work with related works, and Section 7 concludes the paper with final remarks and directions for future works.

2 Background

In this section, we provide background information on the Multi-Annual Roadmap for Robotics (Section 2.1) and Programmable Electronic Medical Systems (Section 2.2).

2.1 Multi-Annual Roadmap for Robotics in Europe

In this section, we introduce the Multi-Annual Roadmap for robotics in Europe (MAR) [10] which has been used as an inspiration for the definition of LENS. This roadmap provides a high-level strategic overview of the robotics community and its objectives, and identifies challenges and opportunities available for robotics. Similar roadmaps have been defined for other continents and countries[4]. Even though MAR focuses on robotics, we used it as an inspiration since it is organized into a set of robot

3. https://vexos.com/mvm-ventilator/

4. Australia: https://roboausnet.com.au/robotics-roadmap/, US: https://www.nowpublishers.com/article/DownloadSummary/ROB-066

abilities, similar to how we aimed to structure LENS. Each ability captures one specific aspect of the operation and behavior of a robot system. MAR is focusing on robotics, which is, indeed, a special kind of autonomous system. We then changed and generalized the abilities to autonomous systems in general. The abilities introduced by MAR are:

- *Adaptability*: concerning the actions of the robot to adapt itself to various scenarios, environments, and conditions.
- *Cognitive Ability*: concerning the actions of the robot to interpret a task, human commands, and environment, as well as, work interactively with humans, so as to efficiently and effectively execute the task potentially under uncertainty.
- *Configurability*: concerning the actions of the robot to be (re-)configured or self-(re-)configured to perform a task.
- *Decisional Autonomy*: concerning the actions of the robot to act autonomously (degree of autonomy).
- *Dependability*: to perform its given mission without errors.
- *Interaction Ability*: concerning the actions of the robot to interact both cognitively and physically either with users, operators or other systems around it, including other robots.
- *Perception Ability*: concerning the actions of the robot to perceive its environment.
- *Manipulation Ability*: concerning the actions of the robot to handle objects.
- *Motion Ability*: concerning the actions of the robot to move to specific locations.

Moreover, each ability has a series of ability levels, which provide a progressive characterization of what any robotic system might do. This is another aspect that has been analyzed in depth when building LENS and then its instantiation, LENS_{PEMS}.

2.2 PEMS: Programmable Electronic Medical Systems

Programmable Electronic Medical Systems (PEMS) are systems that do not have Manipulation and moving abilities. Indeed, according to the medical standard IEC 60601-4-1 [9], PEMS are defined as MEE[5] or MES[6] containing one or more systems based on one or more central processing units, including their software interfaces. Therefore, PEMS are not medical robots; they may exhibit a DOA (Degree of Autonomy), but they do not have motion and/or manipulation abilities. On the other side, a medical robot is a PEMS with motion and manipulation abilities (e.g., robotic surgery systems, dog therapy robots, etc.).

5. A MEE (Medical Electrical Equipment) is defined as an electrical equipment having an applied part or transferring energy to or from the patient or detecting such energy transfer to or from the patient and which is: a) provided with not more than one connection to a particular supply mains; and b) intended by its manufacturer to be used: 1) in the diagnosis, treatment, or monitoring of a patient; or 2) for compensation or alleviation of disease, injury, or disability.

6. A MES (Medical Electrical System) is defined as a combination, as specified by its manufacturer, of items of equipment, at least one of which is MEE to be inter-connected by functional connection or by use of a multiple socket-outlet.

3 LENS: evaLuation framEwork for autoNomous Systems

To define LENS, we exploited the various surveys and books in the field of autonomous and self-adaptive systems [1], [2], [4], [5], [6], and we further performed a literature review in evaluation frameworks for system adaptive abilities (see Section 6) to be sure we were not missing relevant papers.

LENS can be formally defined as a tuple $\langle ab_1, \ldots, ab_n \rangle$ of *abilities*, i.e., qualities that an autonomous system owns to perform some given actions. Each ability ab_i can be *primitive* or defined itself as a tuple $\langle ab_{i,1}, \ldots, ab_{i,m_i} \rangle$ of sub-abilities. Each primitive (sub-)ability ab has an associated maximum *level* $MaxLev(ab) > 0$, and for each l from 0 to $MaxLev(ab)$, ab is associated to a tuple $\langle l, name, description \rangle$, where l is the level index; *name* denotes the level of the owned quality; and *description* motivates the quality measure index. Levels are used to measure how much the quality/ability is owned by the autonomous system (the basic level, level 0, usually means that the system does not have the ability).

Since the set of abilities defining LENS is domain-specific, once a given system or class of systems is fixed, it must be customized, as is the case for the LENS_{PEMS} defined in Section 4. Therefore, LENS can be considered a meta-evaluation framework and can be instantiated for any class of autonomous systems upon detecting the set of characterizing abilities and their description levels. It is interesting to highlight that the structure of LENS with its instantiations builds on the idea of engineering product lines and family of products [14], i.e., software systems that share similarities and that can be engineered and built by sharing a set of software assets and a common means of production. It would be interesting in the future to investigate, e.g., precise use of variation, feature models as a meta-modeling language for specializing LENS, using feature models tooling for evaluating instantiations of LENS, and so on.

As a starting point for defining all the abilities an autonomous system should have, LENS suggests the 8 abilities listed in Table 1. Some are primitives, e.g., *Configurability*, while others, like *Cognitive*, are defined by a set of sub-abilities. To identify these abilities, we first exploited the Multi-Annual Roadmap for Robotics in Europe (MAR) (see Section 2.1), which has been an important starting point for defining LENS.

When we customized LENS to the PEMS class of systems, we identified the need for adding the sub-abilities *Adaptation trigger* and *Adaptation object* for the *Adaptability* ability, as well as the *Explainability* ability (see Section 5.2 for more details). Since these (sub-)abilities also refer to autonomous systems in general, we considered them as a valid extension of LENS. Therefore, the abilities shown in Table 1 result from the meta-framework evolution after instantiating it to a specific class of systems. We

TABLE 1
LENS abilities

Ability
Configurability
Adaptability
Adaptation trigger
Adaptation object
Dependability
Autonomy
Interaction (Int.)
Human-system Int.
Human-system Int. feedback
System to system Int.
Human-system Int. safety
Perception
General perception
Element recognition
Scene perception
Cognitive
Action
Interpretive
Envisioning
Acquired knowledge
Reasoning
Cognitive human interaction
Explainability

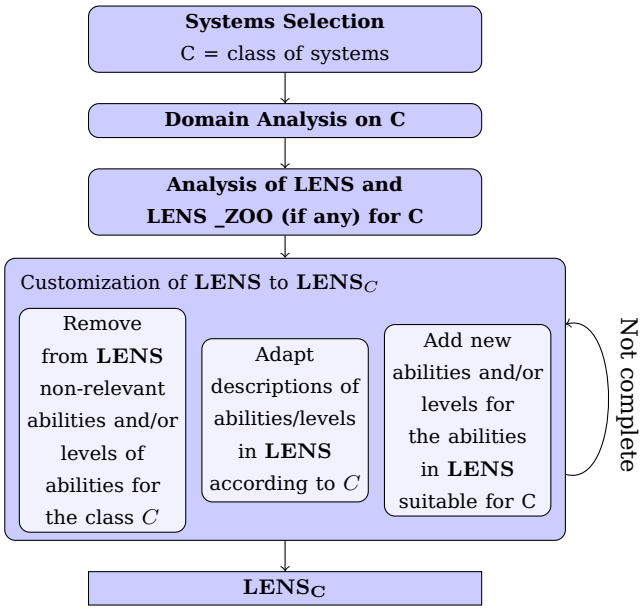


Fig. 1. Reference workflow for customizing LENS

envision such evolution upon customization to further classes of concrete autonomous systems. The openness of LENS enables any appropriate modifications made during customization not to be restricted solely to the particular customization, but to emerge and rise till the meta-framework. As a consequence, any further customization may benefit from previous ones.

In the following, we devise some key learnings and general guidelines on how (i) to customize LENS for a target class of systems, and (ii) to carry out the assessment with the customized framework in practice.

3.1 Customization of LENS: key points and lessons learned

Fig. 1 shows a reference workflow on how a customization LENS$_C$ for a class C of systems can be defined. The starting point for the customization is the characterization of the class C of systems under consideration, and then the analysis of the underlying domain and features of interest. The third step refers to the analysis of the structure (in terms of abilities and sub-abilities) of LENS and of other existing frameworks in LENS_*ZOO* (available in our supplementary material [15]), i.e., frameworks obtained from previous customizations of LENS, for similar classes of systems. Other suitable external frameworks might also be considered for the purpose. According to this preliminary analysis of the domain and diverse systems features, the real customization process can start. It is to be conceived as an iterative execution of abstraction and refinement steps that may imply adding/removing/changing (sub-)abilities and/or levels of abilities, including their descriptions, to make them more suitable for C.

In addition to such a workflow, we here distill some guidance by reporting some key attention points and related lessons learned that we matured from our experience in customizing LENS for the PEMS domain. These key points (KP) include the following:

- *KP0: Characterization of the class of systems*: The starting point in the concretization of the reference workflow for a target domain C is the characterization itself of the class of systems and the analysis of the underlying domain. For this purpose, we found it useful to establish a definition for the class C with the aim of identifying the main elements/features that characterize the target systems and make them different from other classes of similar systems. A common vocabulary of terms starting from that available for LENS and an explanation of their meaning for the domain of C can help in such an analysis. For example, we found it essential to clarify the meaning of the terms *mission* and *task* for the systems under consideration in order to later identify the abilities and sub-abilities of these systems and assess them.
- *KP1: Identification of existing frameworks to analyze*: Once C has been characterized and similar classes of systems have been found, another challenge is the search for a zoo of existing customizations of LENS (if any) for these similar classes. We found it helpful to start from an existing evaluation framework, which in our case was the framework MAR for LENS and LENS for LENS$_{PEMS}$, and then to go ahead with the customization process by abstraction and refinement steps. In the long term, the availability of a documented zoo of diverse customizations of LENS may significantly help with the domain analysis itself.
- *KP2: Customization by examples*: How to reconcile the extension requirements (in terms of abilities and sub-abilities) for C with those of the existing zoo of frameworks? From our experience, it may be useful to start with a

specific instance of C (e.g., the MVM ventilator for PEMS) and then try to generalize to the whole class. This also makes sense because at the beginning one might be interested in using LENS only to evaluate a specific product rather than a broader class. In our case, for example, first LENS_{MVM} was born, and then it became LENS_{PEMS}. Exploiting the similarities between the system variants for the class C could help in such generalization activity.

- *KP3: Customization by abstraction and refinement*: This process is similar to a software design methodology; it is manual and requires some creativity. Refined abilities and sub-abilities can be obtained at first instance as specific to the system example(s) and then generalized and lifted to the entire class of systems. One can consider a spectrum of levels of granularity, e.g., going from coarse-grained to fine-grained. From our experience, we recommend to first remove what is non-necessary (and to be sure of this one has to look at the whole system class), then adapt what remains, and, finally, add what is missing. To identify what is missing, it is necessary to look into specific instances and then generalize to the whole class. The actions taken to address RQ2 during validation serves as an illustration of the execution of this step.
- *KP4: Customization guided by tools*: How to manage the customization process with some means of automation? Customization tools built around tried and trusted practices could be identified and selected to assist in classifying characteristics, possibly overlapping or interrelated, about the target systems. As aforementioned, examples of such tools include those for feature and variability modeling from the area of software product lines engineering [16], for model evolution from model-driven engineering [17], and so on.

3.2 Making an assessment with $LENS_C$

Once the abilities of LENS_C together with their levels and descriptions, for a class C of autonomous systems, are identified, a set of activities are to be performed to evaluate a system in that class. It is important to point out that humans play an important role in the evaluation, and some aspects cannot be completely automated. Part of the evaluation concerns the re-engineering opportunities of the system under evaluation. Indeed, some instruments could be used, e.g., to retrieve important information or to understand the impact of a potential decision. Such instruments are most probably dependent on and specific to the system under evaluation.

In this paper, we will not focus on opportunities to automate this evaluation, and we leave such investigation to future works. The evaluation will be performed by an evaluation committee that has all the needed competencies. It should include experts in the domain able to understand whether it makes sense to automate specific functionalities, engineers able to understand what it is feasible to automate, but also business strategists able to understand whether it is worthwhile to automate specific functionalities. The evaluation of the committee can follow well-known techniques for collecting opinions from the group of experts and reaching a consensus, like the ATAM method for evaluating software architectures [18], or the Delphi method [19].

3.3 Assessment values

When performing the assessment, the evaluation committee should assign to each level one of the following values:

- *Not applicable - white*: this is the default value, and it is assigned when the level is still too low for the application domain, and a higher level must be present for all the systems in that domain.
- *Satisfied - green*: when the level is completely satisfied. We also require the evaluation committee to justify the value.
- *Improvable (low effort) - yellow*: when the system can be improved to (better) satisfy the level. The effort to realize the improvement is low. We also require the evaluation committee to identify the direction of improvements needed to reach that level.
- *Improvable (high effort) - orange*: when the system can be improved to (better) satisfy the level. The effort to realize the improvement is high. We also require the evaluation committee to identify the direction of improvements needed to reach that level.
- *Unable - gray*: when the system is not able to own the ability at that level, cannot be improved to reach this level of ability or the improvement is out of scope. This can be due to several reasons: e.g., the system configuration, its goals, and the lack of other abilities.

3.4 Assessment process

The assessment process is reported in Fig. 2 and explained in the following:

Phase 0 – Setup: Preparation before starting the evaluation. This is an informal step with the main purpose of bringing all important stakeholders on-board.

Phase 1 – Analysis: First, (1) LENS, already customized for a specific class of systems, is presented along with (2) the system under evaluation (SUE), (3) the context in which the SUE is supposed to operate, and (4) the business goals of the SUE, together with its mission and its main tasks. Then, the SUE can be evaluated under each dimension. For each ability, the evaluation committee will do iteratively the following steps:

(5) it assesses the SUE and assigns an assessment value to the ability levels: *Not applicable*, *Satisfied*, *Improvable (low effort)*, *Improvable (high effort)* and *Unable* (see Table 5 for an example of evaluation; some aspects of the evaluation are explained in the following items);

(6) it provides examples explaining the given assessment for *Satisfied* levels (see Table 4 for an example);

(7) it provides recommendations for potential extensions of the SUE for *Improvable (low effort)* and *Improvable (high effort)* levels (see Table 4 for an example).

Phase 2 – Reporting the evaluation results, along with recommendations for potential extensions of the SUE, are presented by the evaluation committee and documented in a report.

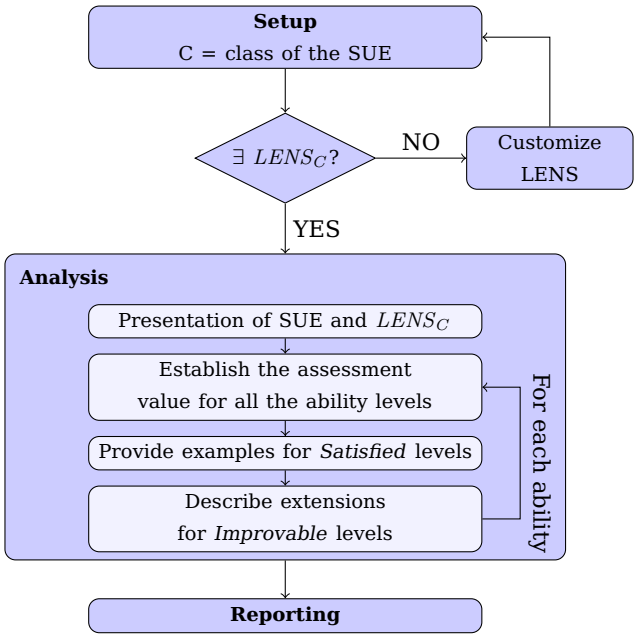


Fig. 2. Assessment process of a SUE with an instance of LENS

Phase 3 – Re-assessment of the SUE, after that all or part of the recommended extensions have been performed. Recommended extensions are delegated to the development team, which possibly differ, in total or in part, from the evaluation committee that carried on the assessment process. This phase essentially consists in a repetition of the assessment process as depicted in Fig. 2, and it can be carried on by the same or a different evaluation committee. The re-assessment of the SUE is particularly useful to verify that the implemented extensions did not degrade one or more abilities previously evaluated as *Satisfied*. In this case, the identification of trade-offs might also be considered.

Table 5 shows what the outcome summary of a potential evaluation would look like. As we anticipated, LENS needs to be instantiated to a domain to be used in practice. However, LENS provides a schema for the evaluation process; the table shows how an overall evaluation will be reported in a summary report as well as filled tables with the details of the evaluation for each ability and level.

An example of an evaluation performed on a concrete system by the instantiation of LENS for PEMS called LENS$_{PEMS}$ is presented in Section 5.1, where we answer the first research question of the validation, i.e., how LENS$_{PEMS}$ is applicable to real PEMS. Specifically, we show the summary of the overall evaluation of the PEMS, and we provide details about the evaluation of two specific abilities, namely the *Adaptability* and *Cognitive* abilities. The overall evaluation of the selected PEMS is available in its entirety in our online supplementary material [15], under the applicability section.

It is important to highlight that the evaluation values do not take into account the return on investment (ROI) for performing a specific improvement of the system. The aim is just to make a rough estimation of the effort to be made to improve the system. We expect that making a proper evaluation of the ROI for a specific improvement requires different profiles and competencies, and it is out of the scope of LENS and LENS$_{PEMS}$. Instead, LENS and LENS$_{PEMS}$ aim at providing the context to enable informed decisions on go/no-go improvements.

4 Instantiation of LENS to the PEMS domain

In this section, we show the instantiation of LENS to a specific class of systems, i.e., medical devices and, specifically, Programmable Electronic Medical Systems (PEMS). The instantiation is called LENS$_{PEMS}$.

4.1 Abilities in LENS$_{PEMS}$

For space reasons, we cannot describe in detail each ability of the LENS$_{PEMS}$ framework and the levels of each ability. We instead focus on two specific abilities that are good representatives of the framework, and we invite the interested reader to refer to the webpage that is associated with the paper for finding details about each ability, together with supplementary material for the entire research made [15].

Table 2 reports the *Adaptation trigger* sub-ability of the *Adaptability* ability of LENS$_{PEMS}$ with its levels. *Adaptability* is defined as the ability of the system to adapt itself to different work scenarios, environments, and conditions. Adaptation may take place over long or short time scales. Furthermore, with the Adaptation trigger sub-ability, we focus on the trigger of the adaptation, i.e., the parts of the system or history of collected data that cause an adaptation. Table 3 shows the description of the levels of the *Action* sub-ability of the *Cognitive* ability in LENS$_{PEMS}$. The *Cognitive* ability is defined as the ability to interpret the task and environment such that tasks can be effectively and efficiently executed even where there exists environmental and/or task uncertainty. In this context, the *Action* sub-ability concerns the ability of the system to act purposefully within its environment and the degree to which it is able to carry out actions and plan those actions. These tables might be exploited, by following the process described in Section 3.2, to assess a system according to the two abilities.

4.2 Tool support

The LENS$_{PEMS}$ evaluation framework is presented together with an online evaluation tool[7] that supports engineers in analyzing a PEMS and creating the final report containing the evaluation of the PEMS under analysis and suggestions for making it smarter. In particular, with the LENS$_{PEMS}$ tool, the users can analyze the abilities and assign to each of them the corresponding assessment value. Moreover, a detailed description (e.g., the scenario in which the level is, or can be, reached) for each ability level can be added.

To simplify the evaluation, the tool supports the incremental evaluation process: a JSON file can be exported

7. https://foselab.github.io/LENS4PEMS/

TABLE 2
Adaptability ability - Adaptation trigger sub-ability

Level	Name	Description
0	No Adaptation	The system does not alter its operating behavior in response to experience gained over time.
1	Human-triggered adaptation	The adaptation of the system is triggered by humans.
2	Adaptation triggered by a single part of the system	The adaptation is triggered by individual components, parameters or tasks.
3	Adaptation triggered by various parts of the system	The adaptation is triggered by a set of interconnected or closely coupled parts of the system.
4	Adaptation triggered by collected data, trends on data, history	The adaptation is triggered by analyzing collected data or data history, or by identifying trends on data.

TABLE 3
Cognitive ability - Action sub-ability

Level	Name	Description
0	Defined action	The system executes fully pre-defined actions as a sequence of sub-actions. This sequence can repeat until stopped by an operator or other system event.
1	Decision-based action	The system is able to alter its course of action based on perceptions or system events. It is able to select between a set of pre-defined actions based on its decisional autonomy ability.
2	Sense-driven action	The system is able to modulate its action in proportion to parameters derived from its perceptions. The perceptions are used to drive the selection of pre-defined actions or the parameters of pre-defined actions.
3	Optimized action	The system is able to alter the sub-task sequence it applies to the execution of a task in response to perceptions or a need to optimize a defined task parameter.
4	Knowledge-driven action	The system is able to utilize knowledge gained from perceptions of the environment including elements within it, to inform actions or sequences of action. Knowledge is gained either by accumulation over time or by embedding knowledge from external sources, including user inputs that associate properties with perceptions.
5	Plan-driven action	The system is able to use accumulated information about tasks to inform its plans for action.
6	Dynamic planning	The system is able to monitor its actions and alter its plans in response to its assessment of success.
7	Task action suggestions	The system is able to suggest tasks that contribute to the goals of a specific mission.
8	Mission proposals	The system is able to propose missions that align with high-level objectives.

from the evaluator at any moment and can be loaded at a later time for completing or updating the evaluation of the tool under analysis. When the evaluation is completed, as a result, the evaluator tool produces the LENS evaluation summary and a PDF report containing all the data inserted by the user. The evaluation summary consists of Table 1 filled with colors according to the values assigned to each ability, as described in Section 4.1. Additionally, it reports the complete evaluation for each ability, including the levels with their assessment values and the detailed description of the scenario in which the level is, or can be, reached. An example of a filled table summary is provided in Table 5 reporting the evaluation of a PEMS (the MVM ventilator), while an example of a complete evaluation summary is available in our online supplementary material [15], under the applicability section.

5 Validation

We frame the validation of the LENS$_{PEMS}$ framework into the following three Research Questions (RQs):

- **RQ1** (Applicability): How is LENS$_{PEMS}$ applicable to real PEMS?
- **RQ2** (Generalizability): To what extent is LENS$_{PEMS}$ generalizable to the PEMS class of systems?
- **RQ3** (Usefulness): How is LENS$_{PEMS}$ useful in making an assessment of a PEMS and identifying possible directions of improvement towards smartness?

5.1 RQ1: Applicability of LENS$_{PEMS}$

This section aims to answer RQ1 (Applicability): How is LENS$_{PEMS}$ applicable to real PEMS?

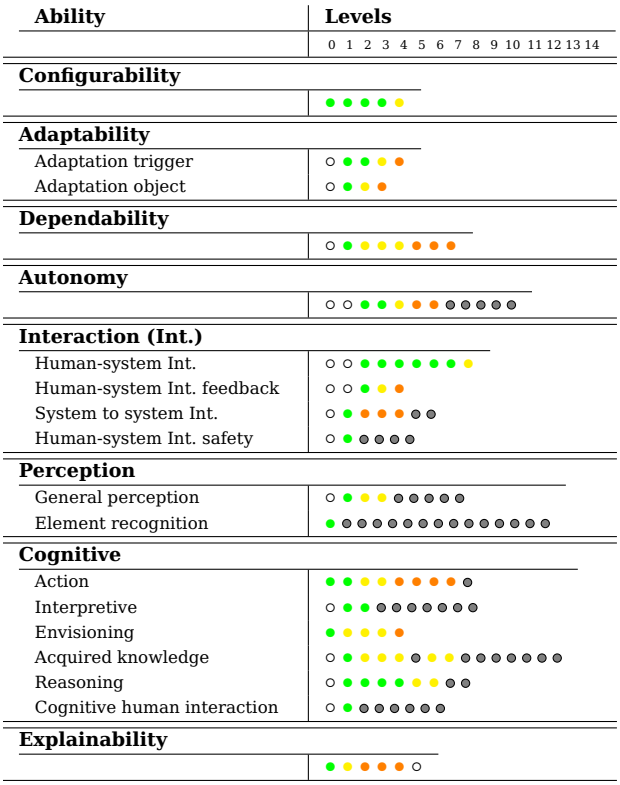
Methodology. To answer RQ1, we used LENS$_{PEMS}$ to evaluate a real PEMS. Specifically, we followed the process described in Section 3.2, and, therefore, for each of the levels of abilities and sub-abilities, we have analyzed the level of satisfaction, improvability, or inability. The PEMS we considered is a mechanical ventilator (MVM), which has been developed during COVID-19 [11]. Most authors of this work have collaborated on the realization of the MVM, therefore, we have access to all data, code, and documentation produced during the MVM development (i.e., around 60 artifacts on requirements engineering, architectural design, testing – unit, integration, and validation –, implementation, documentation, and traceability checking).

The MVM *mission* is to perform patients' ventilation. The mission is pursued through a series of *tasks*: (i) startup (both for hardware and software components) to initialize the ventilator with default parameters, (ii) self-test to ensure that the hardware is fully functional, (iii) alarms management to raise alarms when faulty or dangerous conditions are detected by the software components, (iv) ventilation-on when ventilating the patient, (v) ventilation-off when ventilation is not required, and (vi) safe-mode performed when dangerous situations are

TABLE 4
MVM evaluation on adaptability - Adaptation trigger sub-ability

Level	Evaluation	Details
0	Not applicable	–
1	Satisfied	MVM changes its ventilation mode in response to the patient's breath.
2	Satisfied	MVM raises alarms in case of faulty parts (e.g., tube obstruction, high inspired volume, oxygen level too high) having different priorities (High, Medium, Low).
3	Improvable (low effort)	MVM can be extended with an alarm managing component responsible for aggregating all triggered alarms and prioritizing them using a Fault Tree Analysis (FTA) technique.
4	Improvable (high effort)	By maintaining a history of the alarm system and the patient's state over time, MVM could be equipped with a predictive model to forecast severe situations, for example, when the patient is about to go into apnea status.

TABLE 5
MVM evaluation



detected and the patient must be protected. During ventilation, the MVM can be in two modes: PCV (Pressure Controlled Ventilation) when the patient is not able to start breathing on his own, or PSV (Pressure Support Ventilation) when MVM simply supports the patient's breathing cycles.

MVM has been developed according to the `IEC 62304` standard [12], [13] and it obtained the certification by the FDA (Food and Drug Administration)[8], the Health Canada Authorization[9], and the CE marking. Thanks to these achievements, MVM can be now sold and used not only in the USA, but also in Canada and Europe.

Although MVM has been a successful project and the MVM ventilator has been produced in a large quantity, we have initiated a feasibility study to make MVM smarter and autonomous as much as possible. This work has been carried out in the context of the initial phase of an Italian project, called *MVM-Adapt*[10], on strategic funds for devices and technology to face with COVID-19. The project involved 3 partners, i.e. 3 universities in Italy, for a total of 12 researchers (including 2 physicians). During this phase of the project, we conceived LENS$_{PEMS}$ and made an assessment of MVM to understand its degree of Autonomy and identify all the possible directions in which to improve the autonomy of MVM.

Results of the evaluation of RQ1. Table 5 shows the summary of the LENS$_{PEMS}$ evaluation on MVM.

The evaluation of the current state of the MVM abilities shows that the ventilator is mature enough (see green cells) with respect to *Configurability*, the basic levels of *Perception* (since higher levels are not required by a mechanical ventilator), *Reasoning* (sub-ability of *Cognitive*), and *Human-system interaction*.

According to the MVM mission, some abilities/sub-abilities are not improvable (gray cells), since they are out of the scope of what a mechanical ventilator should do (e.g., *Element recognition*, or *Cognitive human interaction*).

With the opinion and feedback of physicians (that we hired as consultants in the project) and their direct experience in using mechanical ventilators in intensive care units, we identified through LENS$_{PEMS}$ the directions in which the system can be further extended. Some abilities (yellow cells) are feasible with a reasonable effort, while others (orange cells) require a major effort. Specifically, it turned out that MVM can be worthily improved in terms of *Adaptability* and *Autonomy* by endowing MVM with the ASV (Adaptive Support Ventilation) mode (already available in more advanced mechanical ventilators). It consists of adapting the ventilation parameters (mainly pressure and respiratory cycle time) depending on the patient's status. In ASV mode, the ventilator continuously checks if either a patient is able to spontaneously breathe – in that case it simply supports the ventilation – or if the patient needs controlled ventilation.

With the ASV ventilation mode, we would extend the *Cognitive Action* sub-ability (see the yellow cells in Table 6), but also *Autonomy*, *Acquired knowledge*, and *Reasoning* (see the complete documentation online [15] where we make available for download also the report of the analysis, under the applicability section). With the additional availability of a stochastic model of the patient and the involvement of analytics techniques, we could further improve the *Adaptability* and *Cognitive* abilities of the MVM (see the orange cells in Tables 4 and 6), as well as the

8. https://bit.ly/44yrWzH
9. https://bit.ly/3Fg5OPg
10. https://bit.ly/3FRTB3E

TABLE 6
MVM evaluation on Cognitive ability - Action sub-ability

Level	Evaluation	Details
0	Satisfied	Predefined actions are executed in self-test mode, which can be interrupted by the operator.
1	Satisfied	The MVM continuously monitors the patient's breathing and decides to change modes (ventilation algorithms) and set the backup parameters in the transition from PSV to PCV in case of apnea. The MVM is also able to continuously monitor the patient's breathing and decides the inlet valve pressure and outlet valve status.
2	Improvable (low effort)	By introducing the additional ASV ventilation algorithm, a threshold-based parametric adaptation decision is realized by the MVM: when the ventilator is operating in PCV mode, based on the monitoring of patient parameters (respiratory rate and target volume) and the relative distance from the Otis curve, the MVM switches to ASV mode to ensure minimum breathing effort (WOB: work of breathing).
3	Improvable (low effort)	MVM extended with the ASV mode can calculate the optimal breathing pattern that involves the minimum WOB for the patient (it optimizes the inspiratory pressure and the respiratory rate to reach a target value).
4	Improvable (high effort)	The MVM embeds a knowledge base where all the information (e.g., patient state, ventilation failures, etc.) is stored. These historical data are used for data analytics, user profiling, and improving the system's dependability.
5	Improvable (high effort)	MVM is extended with descriptive analytics: it can plan ventilation strategies according to the accumulated knowledge.
6	Improvable (high effort)	MVM is extended with predictive analytics: it can dynamically change the ventilation plan in response to its level of success. For example, in ASV mode, the MVM may monitor the overall life cycle of the ventilation process in terms of a specific flow of activities, and predict how and which parameters to modify to calibrate the ventilator according to the current values of the patient's parameters; in this case, the MVM assists the doctor in the clinical use of the ventilator in ASV mode.
7	Improvable (high effort)	MVM is extended with prescriptive analytics based on a predictive model of the patient (e.g., a stochastic runtime model of the patient). The MVM can provide prescription actions to calibrate the ASV ventilation based on the comparison between the patient's real monitored data and the data prescribed by the patient model.
8	Unable	MVM, as a mechanical ventilator, has only one mission (the lung ventilation).

Dependability and *Interaction* abilities. Indeed, if the MVM was endowed with a knowledge base, e.g., information on the patients' state and ventilation failures (level 4 in Table 6), it would be able to adapt its ventilation mode based on predefined ventilation strategies (descriptive analytics - level 5 in Table 6), or even dynamically change the ventilation plan in response to its level of success (predictive analytics - level 6 in Table 6).

Moreover, if a predictive model of the patient was available, MVM would be able to calibrate the ventilation parameters (possibly in the advanced ASV mode) based on the comparison between the patient's real monitored data and the data prescribed by the patient model (prescriptive analytics - level 7 in Table 6). These extensions require a higher effort with respect to the yellow ones discussed above, and moreover, the medical community would also show resistance in accepting them as they would employ AI technologies for analytical and decision-making skills [20].

Further details on the evaluation with respect to all the abilities and their levels are provided as part of the online supplementary material [15], which also provides the evaluation summary under the applicability section.

With respect to the applicability of LENS$_{PEMS}$, we can state that the framework's usage for assessing the MVM (by following the process provided in Section 3) shows the ease of use of LENS$_{PEMS}$ and its user-friendliness. This might be because the evaluation committee includes people with the needed competencies and deep knowledge of the SUE. Nevertheless, this should be a pre-requisite to the use of LENS$_{PEMS}$, to guarantee the quality of the evaluation.

RQ1: Applicability

5.2 RQ2: Generalizability of LENS$_{PEMS}$

This section aims to answer RQ2 (Generalizability): To what extent is LENS$_{PEMS}$ generalizable to the PEMS class of systems?

Methodology. To answer this RQ, we collected a number of PEMS and analyzed the following aspects:

- *Fit for purpose*: evaluate whether the current abilities and sub-abilities, together with their levels, (i) are appropriate for evaluating these systems, (ii) need to be slightly changed to better match the needs of the considered PEMS, e.g., adapting some levels or removing or adding some of them, or (iii) abilities and sub-abilities should be removed, or new ones should be added.
- *Extensibility of the tool*: We show how the LENS$_{PEMS}$ tool can be extended with new abilities/sub-abilities and/or levels, or existing ones can be changed.

In our analyses, we have considered a PillBox [21], some models of Insulin Pumps [22], the class of Smart ECG devices [23], [24], a hemodialysis machine [25], and a sterilizer produced by an industrial collaborator[11]. We selected these PEMS since they differ from each other and offer a variety of behaviors, functionalities, interactions with humans, and consequently different levels and needs for autonomy and smartness. Moreover, we identified PEMS that give enough information in terms of published papers, white papers, and/or websites for evaluating them.

In the following, we briefly describe each of them:

- Pillbox [21] is a device that helps patients to follow the prescribed therapy. Pills are organized in compartments based on the doses of medications and, when it is the right time to assume the pill, the

11. https://bit.ly/3Z5zJm6

pillbox notifies the user with sound/light signals or smartphone notifications.

- Insulin pump [22] is a device that administers insulin to diabetic patients. In the last years, insulin pump technology has grown fast: patients are continuously monitored, and the insulin is administrated automatically based on the current blood sugar level and the type of diabetes.
- Smart ECG devices [23], [24] belong to the class of smart health monitoring systems and enable the continuous monitoring of the electrocardiogram (ECG). Smart ECG devices exploit emerging and advanced communication techniques (e.g., wired / wireless communication) to collect and deliver biomedical signals [24], and AI methods for the ECG signal interpretation (e.g., AI algorithms, neural networks) [23]. Specifically, the monitoring can leverage mobile, wearable, and sensor devices [23], integrated into t-shirts, smartphones, and smartwatches.
- Hemodialysis machine [25] is used to clean the blood in case of kidney diseases. The machine withdraws and filters the patient's blood by eliminating wastes and salts, then the blood is returned to the patient.
- The sterilizer for medical devices (e.g., dentist tools), produced by an industrial collaborator, uses moist heat in the form of saturated steam under pressure to destroy all forms of microbial life. It integrates a temperature sensor to inform the user about which is the current temperature inside (e.g., if it is too hot), a smart sensor that tells what the sterilizer needs (e.g., how to optimize drying), and a status sensor that informs the user which is the current status of the sterilization cycle.

All the documents we produced, containing the comments about the generalizability and consequent modifications of LENS_{PEMS} are available online, under the generalizability section [15].

To perform an evaluation of the PEMS and consequently to answer RQ2, we elicited the necessary modifications arising from each of the examined PEMS. Once we confirmed that each modification was suitable for all the PEMS, we used the following symbols to represent the changes and their rationales, as summarized in Table 7:

- $=$ (No change): ability and its levels, when present, are good as they are.
- $\sim$ (To be changed): during the evaluation of the selected PEMS it was found that some changes are needed in LENS_{PEMS}. Possible changes are:
  1) addition of one or more levels;
  2) union of levels;
  3) change in the description and/or name of
     a) the ability;
     b) the sub-ability;
     c) the level;
  4) removal of one or more levels;
  5) split of a level.
- $\times$ (To be removed): the entire ability or sub-ability should be removed;
- ✓ (To be added): the evaluation of the selected PEMS required/suggested the addition of the new ability or sub-ability.

It is important to highlight that in LENS_{PEMS}, when an ability has sub-abilities, the ability has only a name and a description, but it does not contain levels. Consequently, the only changes that are possible for these types of abilities might concern only the name and/or the description. For example, in *Interaction*, we report $=$ since changes in the description of the ability have not been made, while the sub-abilities have been changed in some way. Moreover, in Table 7 we have strike-through the abilities and subabilities that have been changed or removed.

Lastly, we assessed the generalizability of LENS to the PEMS class of systems by considering only five distinct PEMS, as we reached saturation during the refinement of LENS_{PEMS}. More specifically, the majority of modifications were derived from the initial assessment of the first examined PEMS, with the latter ones making minimal contributions to the elicitation process. Nonetheless, they confirmed the appropriateness of the elicited modifications.

Results of the evaluation of RQ2. In the following, we report the outcome of the evaluation for what concerns *Fit for purpose* and *Extensibility of the tool*.

5.2.1 Fit for purpose

The evaluation for generalizability triggered various changes of different granularity in different abilities and sub-abilities. This permitted us to better fit LENS_{PEMS} to the PEMS class of systems. In the following, we summarize the outcome of the fit-for-purpose evaluation:

- *Abilities and sub-abilities that are appropriate for evaluating PEMS and then require no change*: this is the case of the descriptions of the *Dependability*, *Interaction*, and *Perception* abilities.
- *Abilities and sub-abilities that need to be slightly changed to better match the needs of the considered PEMS*: since we revised the terminology to distinguish clearly among environment, physical environment, human environment, and patient, this caused changes in almost every ability, sub-ability, and level. Moreover, changes were mainly triggered by the need of removing ambiguities.
- *Abilities and sub-abilities that need to be changed by adding and/or removing levels*: The only ability that required the addition of a new level is the *Dependability* ability, by adding the level called *prescriptive dependability*. This new level enables making explicit when the system is able to predict that a planned future action may result in a loss of dependability, or that the effect of the partial failure of a component can be mitigated by altering future actions. For what concerns the removal of levels, we have three cases in the *Dependability* and *Autonomy* abilities and the *Cognitive-action* sub-ability. The removal concerned the removal of a level 0 of no dependability, no autonomy, and no action that we considered as confusing

TABLE 7
Evaluation of Generalizability of LENS$_{PEMS}$ to collected PEMS.

Abilities / Subabilities	**Changes**	**Rationale**
Configurability	$\sim_{3.a}$ $\sim_{3.c}$	The ability description and some of the levels' description have been revised such that to remove technicalities and make them more general, thus be applicable to different PEMS.
Adaptability	$\sim_{3}$ $\sim_{4}$	The ability description required a change to accommodate the revised ability, for which we now envisage two sub-abilities, namely *adaptation trigger* and *adaptation object*. The adaptability levels have been removed and reused and adapted in the new *adaptation object* sub-ability.
▷ *Adaptation trigger [NEW]*	✓	Considering that multiple PEMS show some degree of autonomy, we identified a new sub-ability regarding the possibility for PEMS to trigger the need for adaptation.
▷ *Adaptation object [NEW]*	✓	This sub-ability inherits its levels from the previous adaptability ability. Moreover, levels have been adjusted such that to accommodate the identification of the objects of the adaptation and how the system alters its behavior or structure for adaptation purposes.
Dependability	$\sim_{1}$ $\sim_{3.c}$ $\sim_{4}$	Considering that all PEMS are dependable to some degree, we removed the "no dependability" level. Moreover, we added the "prescriptive dependability" level, as emerged from some of the evaluated PEMS.
Autonomy	$\sim_{3.c}$ $\sim_{4}$	Considering that all PEMS exhibit a degree of autonomy, we removed the "no autonomy" level. Moreover, we slightly revised the descriptions of levels to accommodate the revised terminology.
Interaction (Int)	$=$	No changes both in the ability name or description.
▷ *Human-system Int*	$\sim_{3.b}$	We added a description of this sub-ability, which was missing in the previous version of the tool.
▷ *Human-system Int feedback*	$\sim_{3.b}$ $\sim_{3.c}$	We added a description of this sub-ability, which was missing in the previous version of the tool. We further revised some of the levels' names and descriptions to accommodate the revised terminology, to increase the clarity of each level, and to include sound feedback.
▷ *System to system Int*	$\sim_{3.b}$ $\sim_{3.c}$	We added a description of this sub-ability, which was missing in the previous version of the tool. We further revised some of the levels' descriptions to accommodate the revised terminology.
▷ *Human-system Int safety*	$\sim_{3.b}$ $\sim_{3.c}$	We added a description of this sub-ability, which was missing in the previous version of the tool. We further revised some of the levels' descriptions to accommodate the revised terminology, to increase the clarity of each level, and remove ambiguities.
▷ ~~*Human-system Int safety-cont.*~~	×	We removed it for lack of clarity and to remove ambiguities with the other interaction sub-abilities.
Perception	$=$	No changes both in the ability name or description.
▷ ~~*Perception*~~ → *General perception*	$\sim_{3.b}$ $\sim_{3.c}$	We revised the name and description of this sub-ability, and some of the levels' descriptions to accommodate the revised terminology, to increase the clarity of each level, and remove ambiguities.
▷ ~~*Object recognition*~~ →*Element recognition*	$\sim_{3.b}$ $\sim_{3.c}$	We revised the name and description of this sub-ability, and some of the levels' descriptions to accommodate the revised terminology, to increase the clarity of each level, and remove ambiguities.
▷ ~~*Scene perception*~~	×	Given that PEMS do not have motion and manipulation, they do not need to have scene perception capabilities, and general perception (which focuses much on sensing the environment) is enough.
Cognitive	$\sim_{3.a}$	We slightly changed the description of the ability to fit with the revised terminology.
▷ *Action*	$\sim_{3.c}$ $\sim_{4}$	We removed the no action ability level, since PEMS show some level of action on the environment. We slightly revised the other levels' descriptions to fit with the revised terminology.
▷ *Interpretive*	$\sim_{3.b}$ $\sim_{3.c}$	We revised both ability's description and the levels' description to fit with the revised terminology and to better adapt them to the medical domain.
▷ *Envisioning*	$\sim_{3.c}$	We slightly changed the description of some of the levels to fit with the revised terminology.
▷ *Acquired knowledge*	$\sim_{3.b}$ $\sim_{3.c}$	We changed both the description of the ability and the levels' description to fit with the revised terminology.
▷ *Reasoning*	$\sim_{3.c}$	We revised both the levels' names and descriptions to fit with the revised terminology and to better adapt them to the medical domain.
▷ ~~*Human interaction*~~ →*Cognitive human interaction*	$\sim_{3.b}$ $\sim_{3.c}$	We revised the name of the ability and the name and description of one level to remove ambiguities and better fit with the revised terminology.
Explainability [NEW]	✓	Considering that PEMS are medical devices, whose behavior might have often a considerable impact on the patients, we find the explainability ability relevant and applicable to the PEMS class of systems.

since PEMS always exhibit some degree of these abilities. Moreover, in *Adaptability*, we removed all levels since we organized the ability into two sub-abilities, one for the *Adaptation trigger* and one for the *Adaptation object*.

- *Abilities and sub-abilities that should be removed*: we removed two sub-abilities. The *Human-system Interaction Safety - Context* sub-ability of *Interaction* has been removed, since in the context of PEMS it overlaps with the *Human-system Interaction Safety* ability. The *Scene Perception* sub-ability of *Perception* has been removed since PEMS have no motion and manipulation capabilities and therefore the *General perception* is enough (without the need for scene perception).
- *Abilities and sub-abilities that should be added*: We refined the *Adaptability* ability into two sub-abilities, namely *Adaptation object* and *Adaptation trigger*. *Adaptation object* and its levels have been defined by inheriting and refining the levels of the previous *Adaptability* ability. Moreover, we added the *Adaptation trigger* sub-ability, which focuses on the possibility for PEMS to trigger the adaptation. Finally, we added a new ability, namely *Explainability*, since it is getting increasing importance in AI-based and autonomous systems, and it is particularly relevant for PEMS because they involve humans both in the role of users (e.g. medical staff, operators, physicians) and patients.

We mention that we did not find the need for unifying or splitting existing levels, and, therefore, we never used

```
{   "abilityName": "Explainability",
    "abilityDescription": "The explainability is the
     ability of the system to [...]",
    "hasSubAbilities": false,
    "abilityLevels": [ {
            "level": 0,
            "levelName": "No explainability",
            "levelDescription": "The system does not
     explain its operating behavior while it adapts itself
     ."
        },
        [...]
    ]
}
```

Listing 1. Excerpt of the JSON file describing the abilities used by the LENS$_{PEMS}$ tool

the symbols "$\sim_2$" and "$\sim_5$" in Table 7.

Note that the MVM evaluation reported in Section 5.1 is the result of the device re-evaluation after the LENS$_{PEMS}$ improvements due to generalizability analysis. With respect to the previous evaluation, small changes, reflecting modifications shown in Table 7, involve *Configurability*, *Human-System Interaction Safety*, and *Scene perception*; deeper changes concern *Adaptability* since the reviewed version of LENS$_{PEMS}$ now distinguishes between trigger and object of adaptation.

5.2.2 Extensibility of the tool

The need for extending or changing LENS$_{PEMS}$ gave us the opportunity to experiment with the extensibility of the tool. In this subsection, we report our experience in extending the LENS$_{PEMS}$ tool. In particular, we here explain how the process of extending the tool to support the new *Explainability* ability has been carried out.

The first activity needed for extending the LENS$_{PEMS}$ tool is the update of the JSON file containing the description of the abilities and their levels[12]. Performing it is a very straightforward operation, and it is necessary whenever a new ability is included in the evaluation framework of LENS$_{PEMS}$, or if there are changes to the existing abilities. In the case of the *Explainability*, the JSON file has been modified by adding the description of the ability, its properties, the levels, and their description, as shown in the excerpt reported in Listing 1.

The last activity consists of modifying the HTML webpage we are currently hosting on GitHub pages[13] in order to set the correct buttons and tabs for each ability in LENS$_{PEMS}$. This is only required when new abilities are added or removed, but not when additional levels are appended or updates in the descriptions or levels are performed. In the case of the *Explainability*, the needed update is very simple, since only two lines have to be added to the HTML file, as shown in Listing 2.

12. https://github.com/foselab/LENS4PEMS/blob/main/docs/abilities.json

13. https://github.com/foselab/LENS4PEMS/blob/main/docs/index.html

```
[...]
<button class="tablinks" onclick="openAbility(event, '
    Explainability')">Explainability</button>
[...]
<div id="Explainability" class="tabcontent"></div>
[...]
```

Listing 2. Excerpt of the HTML file of the LENS$_{PEMS}$ tool

With respect to the fit for purpose of LENS$_{PEMS}$ to the PEMS class of systems, we can state that LENS$_{PEMS}$ was overall appropriate for evaluating these systems. Indeed, after the analysis and evaluation of 5 different types of PEMS, the majority of elicited changes referred to the descriptions of abilities and levels, mainly due to a terminology revision. Only a few abilities, sub-abilities, or levels have been added or removed. Moreover, the tool shows suitability for extensions, by easily supporting adjustments.

RQ2: Generalizability

5.3 RQ3: Usefulness of LENS$_{PEMS}$

This section aims to answer RQ3 (Usefulness): How is LENS$_{PEMS}$ useful in making an assessment of a PEMS and identifying possible directions of improvement toward smartness?

Methodology. We answered this question by following a mixed research methodology, including answers to a questionnaire and interviews. The questionnaire is structured as follows:

- Introduction to LENS and LENS$_{PEMS}$ through text and a short video explaining their usage;
- Introductory questions to collect demographic information and experience in projects for developing/studying/using PEMS (Q1 and Q2);
- Questions to assess the usefulness of LENS and LENS$_{PEMS}$ (from Q3 to Q7);
- Questions to check the availability of the person to test LENS$_{PEMS}$ on a PEM system or for a follow-up interview (Q8).

The questionnaire, the anonymized responses, and the transcription of interviews are available online, under the usefulness section [15]. To mitigate the potential involvement of non-experts, we specifically contacted people with expertise in adaptive systems and/or PEMS. Furthermore, we made sure to avoid people who might have conflicts of interest with the authors or with the research outcomes. This is further discussed in the threat to validity section (Section 5.4). Table 8 shows anonymized information about the participants in the questionnaire and interviews, together with information on the type of organization where they work (university, research center, or industry), the knowledge they have in PEMS and/or their experience in PEMS projects. Overall, 6 out of the 26 participants come from industry. Specifically, 4 of them work in a company producing PEMS. The remaining 2 work in companies dealing also with self-adaptive systems.

TABLE 8
Participants to the questionnaire and interview

ID	Type of organiz.	Experience in PEMS (years) / # of PEMS in which they worked	Quest. / Interview
P1	University	2-5 years / < 2 PEMS	Q
P2	University	<2 years / < 2 PEMS	Q
P3	University	<2 years / < 2 PEMS	Q and I
P4	University	<2 years / < 2 PEMS	Q and I
P5	University	<2 years / < 2 PEMS	Q and I
P6	University	<2 years / < 2 PEMS	Q and I
P7	University	<2 years / < 2 PEMS	Q
P8	University	<2 years / < 2 PEMS	Q
P9	University	<2 years / <2 PEMS (*)	Q and I
P10	University	<2 years / < 2 PEMS	Q
P11	University	<2 years / < 2 PEMS	Q
P12	Industry	<2 years / <2 PEMS (*)	Q and I
P13	Research center	<2 years / < 2 PEMS	Q
P14	University	<2 years / < 2 PEMS	Q and I
P15	University	<2 years / < 2 PEMS	Q
P16	University	2-5 years / < 2 PEMS	Q and I
P17	University	<2 years / < 2 PEMS	Q and I
P18	University	5-8 years / 2-5 PEMS	Q
P19	University	<2 years / < 2 PEMS	Q
P20	University	>8 years / 2-5 PEMS	Q
P21	University	<2 years / < 2 PEMS	Q
P22	Industry	>8 years / 2-5 PEMS (*)	Q and I
P23	Industry	<2 years / < 2 PEMS	Q
P24	Industry	2-5 years / 2-5 PEMS (*)	Q and I
P25	Industry	<2 years / < 2 PEMS	Q and I
P26	Industry	2-5 years / 2-5 PEMS	Q and I

Among the other 20 participants coming from universities, the majority of them have experience in self-adaptive systems, while 4 of them declare more than 2 years of experience in PEMS development, with P18 and P20 stating more than 5 and 8 years, respectively. Concerning the experience in projects for developing/studying/using PEMS, the majority of participants declare less than 2 years of experience. Thanks to the interviews, we find out that 5 of them (P4, P5, P6, P9, P25) declare one year of experience in PEMS projects. According to the number of PEMS projects that participants have worked on, the majority of them declare less than 2 projects, while 4 of them have experience ranging from 2 to 5 years. The interviews helped participants to better understand and clarify their effective experience in PEMS projects. Indeed, some of them, highlighted with the (*) in Table 8, revised their experience during the interview discussion (due to question misunderstanding), by increasing (P22) or decreasing (P9, P12) both the years of experience with PEMS projects and the number of PEMS projects they worked on. P24, instead, decreased the years of experience and increased the number of PEMS projects.

Results of the evaluation of RQ3. The summary of the agreements of all participants with the questions regarding the usefulness of LENS$_{PEMS}$ is reported in Fig. 3. Overall, as shown for Q3 (*It is a good idea to perform the evaluation in terms of abilities (e.g. Adaptability, Autonomy, etc.) and levels*) in Fig. 3, the participants are of the opinion that assessing according to abilities and levels is a beneficial approach. Through the interviews, we collected further feedback on this question. Specifically, some of the participants (P5, P22, P24, P25, P26) highlight that the abilities of LENS$_{PEMS}$ could be not general enough for different systems in the same class. In this regard, we generalized the framework, as explained in Section 5.2. This is a confirmation of how typically evaluation frameworks are built; for instance, CMMI[14] and FEF [8] have the same structure.

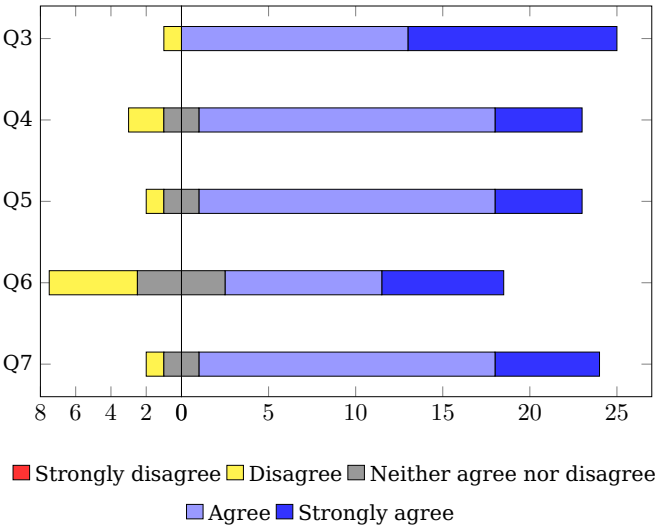


Fig. 3. Agreement with the questions regarding the usefulness of LENS$_{PEMS}$

To the question Q4 (*It is useful to enable an evaluation for each level with values "Not applicable", "Satisfied", and "Unable", together with an evaluation of how costly is to perform an extension "Improvable (low effort)" and "Improvable (high effort)"*) the participants are overall positive as reported in Fig. 3, with only two disagree (P11 and P18).

Among the 15 agrees, P4 and P22 suggested some rewriting and polishing to remove ambiguities (we fixed the recommendations), P10 highlights that the evaluation of the cost could be hard for certain abilities, and consequently a method like Delphi [19] might help. This is indeed in line with the process we defined in Section 3 to evaluate with LENS. Probably this aspect was not made clear in the questionnaire since also P13 added comments on how to decide how much effort is needed for an improvement, i.e., "Improvable (low effort)" and "Improvable (high effort)", or how one decides the evaluation result for each level; we used this recommendation to better explain the process to perform an evaluation with LENS in Section 3. P21 (in the group of neither agree nor disagree) highlights that we can make clear how to scale LENS$_{PEMS}$ since, for instance, the cost is just one factor and there could be other impediments like strategic, regulatory, market positioning, expected ROI, etc. We explained in the paper how to extend LENS$_{PEMS}$ and also how to extend the tool support (see the evaluation in Section 5.2.2).

14. https://cmmiinstitute.com/

During the interviews, we further analyzed the opinions of the participants about Q4. Most of the interviewed participants considered that our formulation is reasonable, and some of them gave us interesting points we describe in the following. In particular, P14 and P4 (who both agree with the question), found it difficult to recognize when a level can be assessed as "unable" or "not applicable". To solve this issue, we now better explain the difference between the two assessment values in Sect. 3.

Moreover, some of the interviewed participants (P5, P14, and P26) highlighted that deciding whether a level can be assessed as improbable with low or high effort is, sometimes, difficult and may require considering also the market request. This is the reason why we envision the assessment with LENS$_{PEMS}$ being performed by an evaluation committee composed of domain experts, who know the market and can better estimate the effort required for an improvement. This type of evaluation performed by an evaluation committee that is required to follow a defined process is also used in other contexts, like the ATAM method to evaluate software architectures [18].

Finally, P4 suggested enforcing that level l_{i+1} can be assessed as "satisfied" only when level l_i is satisfied. This could require the introduction of multiple axes. This is a good point, and we will consider that in future developments.

Concerning Q5 (*I believe that LENS$_{PEMS}$ can be useful to assess the current level of adaptation and smartness of a PEMS*), the consensus among participants is that LENS$_{PEMS}$ can indeed be valuable for such an assessment. P21 explains that the score of neither agree nor disagree is caused by her/his lack of knowledge in PEMS: "It is quite difficult for me to assess this correctly". P5 says that, indeed, the tool is capable of assessing the level of adaptation and smartness and suggesting improvements in those areas. P10 highlights that to properly answer the question, LENS$_{PEMS}$ should be tried in practice; however, she/he does not see issues about the conceptual soundness of the approach. P13 points out that the abilities are related to the adaptation and smartness of a PEMS. The only concern is how to ensure the correctness and accuracy of the evaluation, since we need to do it manually. We argue that these types of evaluations can only be made manually, and the definition of the evaluation process solves this issue (see Section 3).

Additionally, further comments come from the interviews. P5 highlights that sometimes it might be appropriate to also measure on a quantitative basis. This comment is in line with the opinion of P12 stating that LENS$_{PEMS}$ gives an overview of how the system is classified, but it might not be enough for quantification. In this regard, we argued already that LENS$_{PEMS}$ is open to extensions, and it can be used in combination with other tools or methods for quantitative analysis. P6 confirms that having an evaluation committee (composed of experts) mitigates the subjectiveness of the evaluation, while P16 states that the use of the tool is intuitive. P4, instead, argues that using LENS would be great for companies working on adaptive devices since it gives a view that can aid in understanding how the devices will react in different scenarios. Lastly, according to P14 no competing frameworks exist, thus it is good to push the development of LENS, while P22 suggests adding other factors beyond smartness.

Concerning Q6 (*I believe that LENS$_{PEMS}$ can be useful for understanding in which ability it is worth to invest in order to make a PEMS smarter*), we also have a general agreement with 16 participants agreeing or strongly agreeing (Q6 in Fig. 3). This is how evaluation frameworks, like CMMI and FEF [8], are often used; therefore, the general agreement is in line with the literature.

P10 highlights that, under the assumption that scores are produced by engineers who are also domain experts, she/he strongly agrees. However, it can be difficult to find these assessors. Indeed, this is our assumption since the assessment should be done by experts, as explained in Section 3. However, what P10 is probably missing is that a team makes the evaluation, and the team should have the required knowledge, more than each individual.

P17 explains that while she/he agrees that the tool can be used to identify possible opportunities for investment to improve the abilities of the system, it would be an opportunity to put a front focus on quantifying the benefits or "level of need" of uplifting the abilities into higher levels in terms of improvement required (as detailed in the interview). Indeed, this is a good point, and we will consider that in future developments.

In the interview, P6 suggested including a score for each ability to better decide in which direction the device has to be improved. Among the participants answering neither agree nor disagree, P4 explains that LENS$_{PEMS}$ is very nice to spot the weakest points in the system; nevertheless, it is not obvious how this information can be used to prioritize development. Moreover, in the interview, he/she specifies that investments also need to take financial goals into account (as suggested by P14 during the interview), and sometimes having a smarter medical device implies more effort (which they are not always willing to do) to get the device certified. This observation is related to the return on investment for performing a specific improvement of the system, which is, as explained in Section 3, out of the scope of this paper.

P13 explains that there are many abilities related to the smartness of a PEMS (e.g., *Autonomy*, *Perception*, and *Cognitive*). However, it is not clear the importance of each ability; therefore, when multiple abilities are evaluated as requiring improvement, it becomes hard to decide which ability is the most worth investing in. Also for this question, P21 explains that she/he is not a PEMS expert, and then it is quite difficult to assess this correctly.

Finally, among the 5 participants disagreeing, P22 points out that some features may be interesting for research purposes but not worth investing in because the market does not require them, since it is not given that the smartness of a PEMS is the only driving factor for product development. This statement has been confirmed in the interview by P26 and P25, while P22 has also specified that

companies can be more focused on making the device safer instead of smarter. We agree with this observation, and this is why we propose to use LENS$_{PEMS}$ only for what concerns the smartness of the system, while, for other properties, other evaluation frameworks may be used.

Moreover, as we introduced in Section 3, LENS$_{PEMS}$ does not take into account the ROI, but only aims at giving possible improvement directions, while their impact should be evaluated with other tools or methods. Moreover, P24 pointed out that the improvements to make the system smarter depend on the stakeholders included in the assessment phase.

Concerning Q7 (*I believe that LENS$_{PEMS}$ can be useful for making a re-assessment when the improvement has been performed on a PEMS*), P9 provides a general comment since the evaluation is made manually, it can be subjective. Unfortunately, these evaluations, like ATAM [18] or Delphi [19], can only be made by humans and there is no testing or trustable autonomous validation that can substitute humans. P13 would like to have a re-assessment example in the video to better understand. Moreover, she/he would like to understand when "Improvable (high effort)" can be re-assessed as "Improvable (low effort)" and in which cases "Improvable (high effort)" can be re-assessed as "Satisfied". This is a good point, and we will come back to it in future works.

Similarly to the previous question of the questionnaire, P10 highlights that also for this type of evaluation, there is the assumption that scores are produced by experts. 2 participants declare that they neither agree nor disagree (P12 and P21). P12 highlights that she/he would need some extensive use to answer more precisely. Similarly, P21 explains that she/he is not a PEMS expert, and then it is quite difficult to assess this correctly. Again, P11 shows her/his disagreement, and, also for this question, there are no explanations or possibilities for a follow-up investigation.

Additionally, according to interviews, P14 argues that without the framework people would not even have a way of understanding if they were doing something in some direction. Then, she/he adds that metrics are needed to avoid losing details in the evaluation. This last comment is in line with the opinions of P5 and P12. Specifically, P5 states that it would be useful to have requirements, and trade-off analysis to establish a compromise between the system's qualities before and after a change. P12 suggests enabling comparisons with other views of the system, such as the system's requirements.

P6 and P9 argue that there might be subjectiveness issues in the re-assessment. We already discussed this point about Q5. Whereas, P14, P16, P17, P3, P9, P4, P6, P25, P26, P24, and P22 somehow agree about the coherence of using LENS$_{PEMS}$ for re-assessing a previously assessed system, supporting comparison and enabling an iterative process. Lastly, P16 provides an interesting suggestion on how the tool can support interaction with stakeholders. Specifically, she/he says that "the interface of the tool is simple, you can see how stakeholders react to a change, compare the level of satisfaction or how they perceive the changes (e.g., all satisfied, not all satisfied, or when there is uncertainty)".

Concerning Q8 (*Do you have a PEMS and/or are you interested in using LENS$_{PEMS}$ for evaluating your PEMS?*), 15 out of 26 participants answered positively. In particular, during the interviews, P17, who answered negatively to the question in the questionnaire, confirmed that he does not have a PEMS, but he will definitely be interested in using LENS$_{PEMS}$ if he has a PEMS. Moreover, other interviewed participants (P17, P12, P4) declared to be interested in using LENS also in other classes of systems. We will follow up with them to further validate in practice LENS$_{PEMS}$.

Overall, we can conclude that the usefulness evaluation of LENS and LENS$_{PEMS}$ is positive:

- The structure of LENS in abilities and levels is positively evaluated and considered valuable to drive the evaluation.
- LENS$_{PEMS}$ is evaluated as a good and useful instrument to explore the adaptation and smartness space for the system's abilities and provide more actionable insights to the product engineer to reason to what extent it is worth investing in. It is also considered a valid tool for the re-assessment when the improvement has been actuated on a PEMS, although they suggested us some strengthening means for such an intent.

The interviews also provided good observations and suggestions for further improving LENS and LENS$_{PEMS}$ in the future.

RQ3: Usefulness

5.4 Threats to validity

In this section, we describe the main threats to validity according to the scheme proposed in [26] and elaborate on mitigation strategies.

5.4.1 Internal validity

Internal validity is a concern that arises when the design of a study may compromise the accuracy of the results [26]. To minimize this risk, we designed and constructed LENS and LENS$_{PEMS}$ by deeply studying the literature, the Multi-Annual Robotics Roadmap [10], experimented through an evaluation of the MVM, checked on other PEMS, and evaluated with experts in autonomous systems and/or PEMS. Concerning the evaluations, we followed recommendations and best practices in designing them. The interviewed experts participated voluntarily, and confidentiality was emphasized in the interviews to encourage them to respond to the interview questions in the most truthful way.

5.4.2 Construct validity

Construct validity is a concern that arises when the connection between theory and observation may be compromised [26]. Since some of the co-authors have been working on the certification of MVM, this deep knowledge of MVM might have influenced the construction of LENS$_{PEMS}$

and made it specific for MVM, thus compromising its generalizability. This is mitigated by the fact that some of the co-authors have been working in various PEMS, and other co-authors have not been involved in the MVM work. Moreover, we dedicated a research question, i.e., RQ2, to the generalizability of LENS_{PEMS} so as to check to what extent LENS_{PEMS} is generalizable to the PEMS class of systems.

The class of PEMS has influenced both the construction of LENS, besides of LENS_{PEMS}, and its validation. This might lead to over fit LENS to this specific domain. However, we implicitly analyzed also the robotics domain because of the influence of the Multi-Annual Roadmap for Robotics in Europe (MAR) in the construction of LENS. Moreover, as explained in Section 3, to define LENS, (i) we exploited the various surveys and books in the field of autonomous and self-adaptive, (ii) we performed a literature review in evaluation frameworks for system adaptive abilities to be sure we were not missing relevant papers (see Section 6), and (iii) we exploited our experience in other domains to mitigate this risk.

Concerning the questionnaire and interviews, both the questions and interview guide have been defined by researchers who have prior experience in conducting qualitative research within the software engineering domain. We performed the interviews with at least two coauthors participating to minimize the risk of misunderstanding the comments of the experts. We also made recordings, when allowed by the interviewees, took notes during the interviews, made transcripts, and discussed internally until we reached an agreement before including the message from the interviews in the paper.

5.4.3 External validity

External validity is a concern that arises when the results and outcomes of a study may not be generalizable to a wider population [26]. Concerning RQ3, the research question about usefulness, we selected the participants of the questionnaire and interviews with knowledge of autonomous systems and PEMS (in fact, we have no question about their knowledge of these systems).

In this way, we minimize the risk of having useless judgments. To minimize the risk of having biased answers, we avoided experts who could have conflicts of interest with the authors of this work or with LENS.

5.4.4 Conclusion validity

Conclusion validity is a concern that arises when the relationship between the extracted data and the obtained findings may compromise the credibility of the conclusions drawn [26]. Concerning RQ1, we validated the applicability of LENS_{PEMS} by applying it to MVM, which is a certified and real product. Part of the co-authors of this publication have been working on the MVM re-engineering activities and on the software certification. This guarantees that we have enough knowledge of MVM, as well as access to the needed documentation, code, etc. to perform a good evaluation.

Concerning RQ2, the selection of the PEMS used for the evaluation is subject to a selection bias that may impact the external validity of our results, as it influences their generalizability to PEMS not considered. The selection of the PEMS used for the evaluation that is used for answering our RQ2 is also a threat to external validity, since it influences the extent to which our results can be generalized. To mitigate these threats, we considered PEMS that differ from each other in terms of purpose, characteristics, and criticality and cover a large set of functionalities, different types of interaction with humans, and different types of autonomy and smartness.

Concerning RQ3, asking subjects about the usefulness of LENS and LENS_{PEMS} might lead to the hypothesis-guessing phenomenon [26]. To mitigate this risk, we tried to ask questions about the usefulness of specific features rather than the overall framework, and we tried to avoid making bold conclusions, but instead, we focused on the comments, critics, and suggestions for improvement.

Moreover, to mitigate the conclusion validity threats, we documented every step of our research and provided a public replication package and supplementary material to ensure transparency and replicability.

Another threat to the conclusion validity is the need of a regression assessment of the core SUE functionalities after new ones suggested by the proposed framework have been effectively engineered. The new smart features could in fact introduce unintended side effects and invalidate important properties (like safety) of the system. However, we assume that this form of regression assessment is carried out systematically (e.g., via regression testing) after each significant system change as part of the system development process.

6 Related works

In the last years, we observed active and applicable research in autonomous and self-adaptive systems (SASs). The SEAMS community produced two roadmaps to summarize the state-of-the-art, for identifying critical challenges for the systematic software engineering of SASs [1], [2].

However, to the best of our knowledge, no existing paper provides an evaluation framework to assess abilities related to adaptation and to provide guidance to developers and engineers to make a system smarter, in the spirit of making it more autonomous. Existing taxonomies, e.g., [4], [5], identify concepts behind the adaptation, but the taxonomies cannot be used as evaluation frameworks. In the literature, we can find works focusing on a specific system ability, such as adaptability, providing metrics to measure them [27], [28], [29], [30], [31]. Specifically, in Table 9 we collected the metrics provided by the reviewed works, by grouping them in macro categories. For each work, we report the number of provided metrics and the category they belong to. It can be observed that the majority of them contribute with *Architectural Adaptability* metrics. Perez-Palacin et al. [27] aim to support software architects to guide the system adaptation to fulfill systems' quality requirements. To do so,

they provide metrics able to quantify and evaluate software adaptability at the architectural level. In a successive work [31], the same authors present a more extensive set of architectural metrics that can be used for the evaluation of the system adaptability. They further analyze the relationships between adaptability and quality, by means of the defined metrics. Subramanian et al. [29] also target adaptability at the architectural level, with the objective of exploiting the measured architectural adaptation to determine the adaptability of the final software system. Raibulet et al. [30], in addition to architectural adaptability metrics, further define a set of *Dynamic Adaptivity* metrics, to address the evaluation of dynamic adaptivity non-functional requirement. *Architectural Resilience* metrics are, instead, the core of the work by Cámara et al. [28], where 13 metrics are presented, specifically used to show the evidence of the beneficial impact of architecture-based self-adaptation on resilience with respect to different approaches, such as those relying on code-based adaptation. Lastly, Tomforde and Goller in their recent works [32], [33] provide metrics for measuring the *Configuration & Adaptation Coherence* and *Global & Avg. Parameter Spectrum Usage*, respectively, as we will better discuss later on in this section.

TABLE 9
Metrics for self-adaptive systems.

Category of Metrics (tot. # of metrics)	Perez-Palacin et al. [27]	Cámara et al. [28]	Subramanian et al. [29]	Raibulet et al. [30]	Perez-Palacin et al. [31]	Tomforde et al. [32]	Goller et al. [33]
Architectural Adaptability (**16**)	6		3	2	5		
Architectural Resilience (**13**)		13					
Dynamic Adaptivity (**10**)				10			
Global & Avg. Parameter Spectrum Usage (**2**)							2
Configuration & Adaptation Coherence (**2**)						2	

Being focused on a few specific metrics, referring only to one (or a few) abilities of the system, these works miss a holistic view, and, moreover, metrics are defined for specific development phases, e.g., architecture development. Furthermore, the operation of the system is only marginally considered. As expected, adaptability is the most assessed system ability by means of metrics. Also, in LENS and LENS$_{PEMS}$ adaptability plays a central role and it is organized into two sub-abilities, namely adaptation trigger and adaptation object. However, for a wider evaluation of the system's smartness and autonomy, the system adaptation must be considered in synergy with a variety of multiple abilities the system should possess. For these reasons, we think that existing metrics-based evaluation frameworks are not suitable for a holistic evaluation of smart and autonomous systems. However, in the context of our work, we do not exclude that metrics might be exploited in specific customizations of LENS, such as LENS$_{PEMS}$, for measuring a given level under a given ability.

TABLE 10
Inclusion and Exclusion Criteria.

Inclusion criteria
1. Peer-reviewed papers published in journals, conferences, and workshops.
2. Papers presenting an evaluation framework.
Exclusion criteria
1. Papers not written in English.
2. Short papers, posters and tutorials (< 3 pages).
3. Conference Proceedings.

Following famous evaluation frameworks, like CMMI, an evaluation framework needs to cover various abilities together with a clear identification of various levels for each of these abilities.

To further investigate the state of the art in evaluation frameworks for system adaptive abilities, and be sure not to omit existing relevant evaluation frameworks, we systematically analyzed related literature. Specifically, we searched for suitable publications in the *IEEE Xplore*, *ACM*, and *Scopus* digital libraries. As search string, we used the following:

> *(Adaptive System(s)* ***OR*** *Autonomous System(s))* ***AND*** *(Evaluation Framework* ***OR*** *Measurement Framework* ***OR*** *Assessment Framework).*

For this work, the search extends a previous work [7] to further incorporate the most recent works. Specifically, the search was performed by considering publications' title, abstract, and keywords, in the time period from 2012 to (July) 2023. As a result, we got 46 papers from Scopus, 8 papers from IEEE Xplore, and 2 papers from ACM. However, the subsequent screening of publications and duplicates removal showed that the set of papers from Scopus already included papers obtained by querying the other libraries. Thus, the evaluation has been performed on a total of **46** unique papers. The inclusion and exclusion criteria we defined to identify the set of potentially relevant papers are given in Table 10. Details on the SLR can be found in the replication package [15].

Eventually, only a few papers (i.e., [32], [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], [43]) fulfilled the criteria and were evaluated. Hartsell et al. [34] present *ReSonAte*, a dynamic risk estimation and assessment framework for autonomous systems. The probabilities of unsafe conditions or failures are computed from runtime observations about the state of the system and environment, besides safety requirements, design time assumptions, and past failures. Similarly, according to the work by Le et al. [35], IoT data are dynamically exploited to support the risk awareness of Connected and Autonomous Vehicles (CAV), thus enabling faster reactions and better decision-making for a safer mobility [35]. The proposed framework exploits multiple risk profiles to support users in better understanding risks and appropriately adapting to various situations. Mundt et al. [42] also refer to autonomous

driving systems. The authors present *KnowGo*, a dynamic risk assessment framework. It makes use of a risk prediction architecture enabling dynamic reconfiguration in terms of risk criterion, risk model selection, and level of automation, to manage dynamic changes in the operational environment. The frameworks proposed in [34], [35], [42] clearly show that, due to its criticality, risk estimation and assessment require dynamic runtime techniques, thus capturing the failures that systems face at runtime. To this aim, they are specifically intended for evaluating systems during their operation and under the lens of one or a few critical properties, such as the risk of failures. These frameworks do not consider the system as a whole, nor other systems' abilities different from the risk estimation and assessment ability. Given the holistic view of LENS and its objective to support the evaluation of (classes of) autonomous systems all-round, the risk's estimation and assessment is only one ability, among many others, whose presence (and extent in terms of levels) can be considered and assessed, given the specific class of systems under analysis. Differently from [34] and [35], where risk refers to the probability and severity of undesirable events, Smith et al. [36] consider risk as a barrier to the implementation of autonomous systems or as consequences of the use of such systems. The authors provide a risk assessment framework, which is intended to evaluate the different levels of autonomy a system can show, and the risk faced when implementing each of these levels. In other words, the framework aims to assess whether the current state of automation technology will support the envisaged level of autonomy. It also provides an initial consideration of the level of risk associated with the implementation of a specific level of automation. Interestingly, Smith et al. followed a similar approach to ours in their work. Specifically, they structured the proposed *statement of intent of automation* on the basis of a description loosely based on the Autonomy Levels developed by the Society of Automotive Engineers (SAE) [44], adapted to make it applicable to autonomous systems across a variety of domains. Moreover, similarly to LENS, the *statement of intent of automation* is presented in terms of levels, e.g., no automation, assistance, partial automation, and so on, with a level number, a name and a description. It is then combined with *capability of technology* levels. Although rather limited to the assessment of risk in the implementation of autonomous systems, this framework shows the emerging need for practitioners for a tool to support the evaluation of systems' abilities to guide decision-making.

Diaconeasa et al. [37] propose a model-based resilience assessment framework that exploits a resilience ontology to guarantee transparent and up-to-date modeling and quantification of the system risk and reliability metrics. The main objective of this work is to overcome the usually weak integration of reliability analysis in the model-based software engineering process. As opposed to LENS, it is specifically tied to metrics for reliability, enabling the use of a range of resilience mechanisms in the design and operation of a system.

Shimizu et al. [38] present an evaluation framework of the performance limitations of autonomous systems that combines safety analysis and sensor attack simulation. Feth et al. [39] propose a conceptual framework, i.e., a metamodel, to support early design decisions for systematically deriving Safety Supervisors (SSV) for autonomous systems. From an engineering perspective, the framework allows one to arrive at an evidence-based decision about which algorithms to choose for the further development of a safety monitor, by conducting what-if analyses and comparing different meaningful combinations of available solutions. Both frameworks focus on system safety. The framework in [38] essentially aims to evaluate how the system safety is affected by previously identified sensor attacks scenarios. The framework in [39], instead, is more directed towards runtime safety monitoring. These works suggest that evaluating systems safety, especially for safety-critical systems, can address further system improvements during future development. On the one hand, a framework like LENS can easily support the evaluation of safety-related abilities and/or levels, i.e., by including them under the dependability ability and/or by further enabling the reasoning on how to improve the system safety, if needed. On the other hand, LENS is open to the evaluation of other systems' properties specifically tied to the target class of systems.

Vuorimaa et al. [40] target the need for development organizations of capabilities, processes and tools required to achieve the needed readiness for designing Autonomous Machine Systems (AMS). To this aim, by means of semi-structured interviews and based on the literature, the authors propose a set of organizational factors that organizations should exhibit to show a sufficient readiness in designing AMS. Examples of readiness elements are design practices and competences, digital design tools and practices, partnerships and ecosystems. Differently from LENS, the target of the evaluation framework presented in this work is the organization developing an AMS and not the AMS itself.

Dong et al. [41] propose an assessment framework for dependability properties (safety, resilience, robustness, detection, and recovery) of Deep Reinforcement Learning-driven Robotics and Autonomous Systems. Specifically, the dynamics of risk/failures of systems in an uncertain environment are modeled as a Discrete-Time Markov Chain (DTMC). Temporal logic is used for defining dependability properties that are then verified on the DTMC through Probabilistic Model Checking. In this work, similarly to LENS, the aim of the authors is that of providing an assessment framework for dependability able to address the target systems in a holistic way, namely by considering all dependability properties. In addition to this, the objective of LENS and its instances is that of providing a holistic view for the assessment of multiple properties of (specific classes of) autonomous systems, besides dependability.

The studies by Tomforde and Goller [32], [33] extend an existing measurement framework for the properties of adaptive systems, by defining new metrics. Metrics

measuring the quantification of parameters utilization, under dynamic conditions. The overall idea consists of quantifying the change in parameter configurations, thus detecting unexpected events [33]. Metrics measuring the coherence of configurations of self-adaptive systems, by collecting and analyzing the configuration decisions of all the autonomous subsystems of a self-adaptive system [32]. These metrics-based works contributed to Table 9. As discussed above, they exhibit the same limitations of metrics-based evaluation frameworks. Al-Tahir et al. [43] apply knowledge from control system theory to define an assessment framework for co-adaptive human-machine interfaces. The approach enables a better understanding of the dynamics of co-adaptive myoelectric human-machine systems. Specifically, the proposed approach exploits Poincaré maps, to identify learning effects, oscillations and uncertainty in performance. However, the proposed framework is rather tied to myoelectric systems. This makes it difficult to use it for different types of adaptive systems.

In Table 11, we schematically outline the discussed related works about evaluation frameworks for adaptive and autonomous systems, by organizing them according to the systems' properties that each evaluation framework focuses on. It can be observed that all of them target only

TABLE 11
Evaluation frameworks for adaptive and autonomous systems.

Approaches	Risk	Resilience	Safety	Org. Readiness	Dependability	Adaptation
Hartsell et al. [34] (2021)	✓					
Le et al. [35] (2018)	✓					
Smith et al. [36] (2018)	✓					
Mundt et al. [42] (2022)	✓					
Diaconeasa et al. [37] (2019)		✓				
Shimizu et al. [38] (2021)			✓			
Feth et al. [39] (2017)			✓			
Vuorimaa et al. [40] (2021)				✓		
Dong et al. [41] (2022)					✓	
Goller et al. [33] (2022)						✓
Tomforde et al. [32] (2021)						✓
Al-Tahir et al. [43] (2022)						✓

one specific system ability or property, or a set of properties all referring to the same umbrella ability, such as the work by Dong et al. [41] that considers multiple dependability properties. On the contrary, LENS and its instances, such as LENS$_{PEMS}$, provide a broader coverage of the abilities assessment of (classes of) autonomous systems.

In conclusion, to the best of our knowledge, there exists no generic framework for the assessment of adaptive abilities that is applicable to a broad spectrum of autonomous systems and, specifically, to PEMS, i.e., the class of medical devices, which we focus on.

7 Conclusions

This paper contributes a meta-evaluation framework for autonomous systems, called LENS, which enables the assessment of systems under the lens of abilities related to adaptation and smartness. Since the domains in which autonomous systems can be used are various and systems differ among them in terms of purpose, characteristics, abilities, and type of interaction with humans, we defined LENS at an abstract level. Instantiating LENS to a specific domain of autonomous systems allows for the definition of concrete evaluation frameworks that engineers can use to understand in which direction is worth investing to make their system smarter. In this paper, we specialized LENS to the domain of Programmable Electronic Medical Systems (PEMS) and we obtained LENS$_{PEMS}$. The evaluation framework is supported by a tool that guides engineers during the evaluation.

We evaluated LENS$_{PEMS}$ according to three main aspects: (i) *applicability* - how it is applicable to real PEMS, (ii) *generalizability* - to what extent it is generalizable to the PEMS class of systems, and (iii) *usefulness* - how it is useful in making an assessment of a PEMS and identifying possible directions of improvement towards smartness. The results of the evaluations are convincing and promising, and we make also available all the data for transparency and replicability purposes [15]. We also make available on the website of LENS the information that is required to guide interested researchers to replicate the instantiation of LENS for a different class of systems.

As future work, we plan to test LENS$_{PEMS}$ with other PEMS, also in collaboration with some of the experts interviewed during the validation, and who showed their interest in trying LENS$_{PEMS}$ in practice, to evaluate their PEMS. We further plan to perform instantiations of LENS to other classes of systems, such as autonomous guidance or satellite systems. Moreover, we plan to compare the use of LENS's instances with certification standards of the SUE to evaluate the feasibility of possible improvements suggested by the evaluation framework.

Acknowledgements

This work has been partially funded by (a) the European Union - NextGenerationEU under the Italian Ministry of University and Research (MUR) National Innovation Ecosystem (i) grant ECS00000041 - VITALITY – CUP: D13C21000430001 and (ii) PNRR Missione 4 Componente 2 Investimento 1.3, grant PE0000020 – CHANGES – CUP: D53C22002560006; (b) the MUR (Italy) Department of Excellence 2023 - 2027 for GSSI; (c) the MUR FISR 2020 IP_05310 MVM-Adapt; (d) the PRIN project P2022RSW5W - RoboChor: Robot Choreography; (e) the PRIN project 2022JKA4SL - HALO: etHical-aware AdjustabLe autOnomous systems.

The work of P. Pelliccione was also partially supported by the Centre of EXcellence on Connected, Geo-Localized and Cybersecure Vehicles (EX-Emerge), funded by the Italian

Government under CIPE resolution n. 70/2017 (Aug. 7, 2017).

The work of A. Bombarda was partially supported by PNRR - ANTHEM (AdvaNced Technologies for Human-centrEd Medicine) - Grant PNC0000003 – CUP: B53C22006700001 - Spoke 1 - Pilot 1.4.

References

[1] B. H. C. Cheng *et al.*, *Software Engineering for Self-Adaptive Systems: A Research Roadmap*. Springer, 2009.

[2] R. De Lemos *et al.*, *Software Engineering for Self-Adaptive Systems: A Second Research Roadmap*. Springer, 2013.

[3] B. H. C. Cheng *et al.*, *Using Models at Runtime to Address Assurance for Self-Adaptive Systems*. Springer, 2014.

[4] M. Salehie and L. Tahvildari, "Self-adaptive software: Landscape and research challenges," *ACM Trans. Auton. Adapt. Syst.*, vol. 4, no. 2, 2009.

[5] C. Krupitzer *et al.*, "A survey on engineering approaches for self-adaptive systems," *Pervasive Mob. Comput.*, vol. 17, 2015.

[6] D. Weyns, *An Introduction to Self-Adaptive Systems: A Contemporary Software Engineering Perspective*. John Wiley & Sons, Ltd, 2021.

[7] A. Bombarda, S. Bonfanti, M. De Sanctis, A. Gargantini, P. Pelliccione, E. Riccobene, and P. Scandurra, "Towards an evaluation framework for autonomous systems," in *2022 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*, 2022, pp. 43–48.

[8] F. van der Linden, K. Schmid, and E. Rommes, *The Family Evaluation Framework*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 79–108. [Online]. Available: https://doi.org/10.1007/978-3-540-71437-8_6

[9] *IEC 60601-1:2005 Medical electrical equipment - Part 1: General requirements for basic safety and essential performance*, https://www.iso.org/standard/70755.html, International Electrotechnical Commission Std.

[10] EU, "Robotics 2020 Multi-Annual Roadmap For Robotic in Europe," https://old.eu-robotics.net/cms/upload/topic_groups/H2020_Robotics_Multi-Annual_Roadmap_ICT-2017B.pdf, 2016.

[11] A. Abba *et al.*, "The novel mechanical ventilator milano for the COVID-19 pandemic," *Physics of Fluids*, vol. 33, 2021.

[12] A. Bombarda, S. Bonfanti, C. Galbiati, A. Gargantini, P. Pelliccione, E. Riccobene, and M. Wada, "Lessons learned from the development of a mechanical ventilator for COVID-19," in *ISSRE2021*. IEEE, 2021.

[13] ——, "Guidelines for the development of a critical software under emergency," *Information and Software Technology*, vol. 152, p. 107061, Dec. 2022. [Online]. Available: https://doi.org/10.1016/j.infsof.2022.107061

[14] F. van der Linden, K. Schmid, and E. Rommes, *Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering*. Springer Berlin Heidelberg, 2010. [Online]. Available: https://books.google.it/books?id=3-ebcQAACAAJ

[15] A. Bombarda *et al.*, "Supplementary material of the work titled "Evaluation Framework for Autonomous Systems: the case of Programmable Electronic Medical Systems"," https://foselab.github.io/LENS4PEMS/additional_material/.

[16] J. M. Horcas, M. Pinto, and L. Fuentes, "Empirical analysis of the tool support for software product lines," *Softw. Syst. Model.*, vol. 22, no. 1, pp. 377–414, 2023. [Online]. Available: https://doi.org/10.1007/s10270-022-01011-2

[17] R. F. Paige, N. Matragkas, and L. M. Rose, "Evolving models in model-driven engineering: State-of-the-art and future challenges," *Journal of Systems and Software*, vol. 111, pp. 272–280, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121215001909

[18] R. Kazman, M. Klein, and P. Clements, "Atam: Method for architecture evaluation," Carnegie-Mellon Univ Pittsburgh PA Software Engineering Inst, Tech. Rep., 2000.

[19] D. Beiderbeck, N. Frevel, H. A. von der Gracht, S. L. Schmidt, and V. M. Schweitzer, "Preparing, conducting, and analyzing delphi surveys: Cross-disciplinary practices, new directions, and advancements," *MethodsX*, vol. 8, p. 101401, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2215016121001941

[20] E. H. Shortliffe and M. J. Sepúlveda, "Clinical Decision Support in the Era of Artificial Intelligence," *JAMA*, vol. 320, no. 21, pp. 2199–2200, 12 2018.

[21] A. Bombarda, S. Bonfanti, and A. Gargantini, "Developing medical devices from abstract state machines to embedded systems: A smart pill box case study," in *Software Technology: Methods and Tools*. Springer International Publishing, 2019, pp. 89–103. [Online]. Available: https://doi.org/10.1007/978-3-030-29852-4_7

[22] C. Berget, L. H. Messer, and G. P. Forlenza, "A clinical overview of insulin pump therapy for the management of diabetes: Past, present, and future of intensive therapy," *Diabetes Spectrum*, vol. 32, no. 3, pp. 194–204, Aug. 2019. [Online]. Available: https://doi.org/10.2337/ds18-0091

[23] M. A. Serhani, H. T. E. Kassabi, H. Ismail, and A. N. Navaz, "ECG monitoring systems: Review, architecture, processes, and key challenges," *Sensors*, vol. 20, no. 6, p. 1796, Mar. 2020. [Online]. Available: https://doi.org/10.3390/s20061796

[24] M. M. Baig, H. Gholamhosseini, and M. J. Connolly, "A comprehensive survey of wearable and wireless ECG monitoring systems for older adults," *Medical & Biological Engineering & Computing*, vol. 51, no. 5, pp. 485–495, Jan. 2013. [Online]. Available: https://doi.org/10.1007/s11517-012-1021-6

[25] A. Mashkoor, "The hemodialysis machine case study," in *Abstract State Machines, Alloy, B, TLA, VDM, and Z*, M. Butler, K.-D. Schewe, A. Mashkoor, and M. Biro, Eds. Cham: Springer International Publishing, 2016, pp. 329–343.

[26] C. Wohlin, P. Runeson, M. Höst, M. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*, ser. Computer Science, 2012.

[27] D. Perez-Palacin *et al.*, "Software architecture adaptability metrics for qos-based self-adaptation," in *Proceedings of the Joint ACM SIGSOFT Conference – QoSA and ACM SIGSOFT Symposium – ISARCS on Quality of Software Architectures – QoSA and Architecting Critical Systems – ISARCS*, 2011.

[28] J. Cámara *et al.*, "Empirical resilience evaluation of an architecture-based self-adaptive software system," in *Proceedings of the 10th International ACM Sigsoft Conference on Quality of Software Architectures*, ser. QoSA '14, 2014.

[29] N. Subramanian and L. Chung, "Metrics for software adaptability," *Proc. Software Quality Man. (SQM 2001)*, vol. 158, 2001.

[30] C. Raibulet and L. Masciadri, "Evaluation of dynamic adaptivity through metrics: an achievable target?" in *2009 Joint Working IEEE/IFIP Conference on Software Architecture European Conference on Software Architecture*, 2009.

[31] D. Perez-Palacin *et al.*, "On the relationships between qos and software adaptability at the architectural level," *J. Syst. Softw.*, vol. 87, jan 2014.

[32] S. Tomforde and M. Goller, "Beyond homeostasis: A novel approach for assessing the stability and coherence of self-adaptive systems," in *IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress, DASC/PiCom/CBDCom/CyberSciTech 2021, Canada, October 25-28, 2021*. IEEE, 2021, pp. 10–17. [Online]. Available: https://doi.org/10.1109/DASC-PICom-CBDCom-CyberSciTech52372.2021.00018

[33] M. Goller and S. Tomforde, "Runtime assessment of the parameter utilisation in adaptive systems," in *2022 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events, PerCom 2022 Workshops, Pisa, Italy, March 21-25, 2022*. IEEE, 2022, pp. 200–205. [Online]. Available: https://doi.org/10.1109/PerComWorkshops53856.2022.9767230

[34] C. Hartsell *et al.*, "Resonate: A runtime risk assessment framework for autonomous systems," in *SEAMS@ICSE 2021*, 2021.

[35] A. Le, C. Maple, and T. Watson, "A profile-driven dynamic risk assessment framework for connected and autonomous vehicles," in *Living in the IoT: Cybersecurity of the IoT*, 2018.
[36] K. T. Smith, L. M. Coventry, and R. GreenSmith, "An initial generic assessment framework for the consideration of risk in the implementation of autonomous systems," in *IFIP WG 13.6 Working Conference, HWID on Human Work Interaction Design. Designing Engaging Automation*, 2018.
[37] M. A. Diaconeasa, A. Mosleh, A. Morozov, and A. T. Tai, "Model-Based Resilience Assessment Framework for Autonomous Systems," ser. ASME International Mechanical Engineering Congress and Exposition, 2019.
[38] K. Shimizu, D. Suzuki, R. Muramatsu, H. Mori, T. Nagatsuka, and T. Matsumoto, "Evaluation framework for performance limitation of autonomous systems under sensor attack," in *Computer Safety, Reliability, and Security*. Springer, 2021.
[39] P. Feth, D. Schneider, and R. Adler, "A conceptual safety supervisor definition and evaluation framework for autonomous systems," in *Computer Safety, Reliability, and Security*, 2017.
[40] V. Vuorimaa *et al.*, "Factors affecting the organizational readiness to design autonomous machine systems: Towards an evaluation framework," in *Int. Systems and Applications*. Springer, 2021.
[41] Y. Dong, X. Zhao, and X. Huang, "Dependability analysis of deep reinforcement learning based robotics and autonomous systems through probabilistic model checking," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2022, Kyoto, Japan, October 23-27, 2022*. IEEE, 2022, pp. 5171–5178. [Online]. Available: https://doi.org/10.1109/IROS47612.2022.9981794
[42] P. Mundt, I. Kumara, W. van den Heuvel, D. A. Tamburri, and A. S. Andreou, "Knowgo: An adaptive learning-based multi-model framework for dynamic automotive risk assessment," in *Business Modeling and Software Design - 12th International Symposium, BMSD 2022, Fribourg, Switzerland, June 27-29, 2022, Proceedings*, ser. Lecture Notes in Business Information Processing, B. Shishkov, Ed., vol. 453. Springer, 2022, pp. 268–278. [Online]. Available: https://doi.org/10.1007/978-3-031-11510-3_18
[43] I. Al-Tahir, J. W. Sensinger, and E. J. Scheme, "A better framework for the assessment of performance and stability of co-adaptive myoelectric systems," in *International Conference on Rehabilitation Robotics, ICORR 2022, Rotterdam, Netherlands, July 25-29, 2022*. IEEE, 2022, pp. 1–5. [Online]. Available: https://doi.org/10.1109/ICORR55369.2022.9896541
[44] S. International, "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles - j3016_202104."

Andrea Bombarda is a Research Associate in the FOSELab (Formal Methods and Software Engineering Laboratory) at the University of Bergamo (Italy). His research topics are mainly in the context of quality assurance for medical software and systems, by applying rigorous methods and suitable software engineering processes, aiming at improving the effectiveness and the rapidness of the medical software certification process. He received his PhD in Engineering and Applied Sciences from the University of Bergamo (Italy). More information is available at https://cs.unibg.it/bombarda/.

Silvia Bonfanti is a Research Associate at the University of Bergamo (Italy) and she is part of FOSELab (Formal Methods and Software Engineering Laboratory). She received her PhD in Engineering and Applied Science from the University of Bergamo, in collaboration with SCCH (Software Competence Center Hagenberg). Her research interests include software engineering, software testing, formal methods, and medical software certification.More information is available at https://cs.unibg.it/bonfanti/.

Martina De Sanctis is Assistant Professor at the Computer Science department of the Gran Sasso Science Institute (GSSI), in L'Aquila, (Italy). Her research interests include behavioral and architectural adaptation of smart systems and applications, collective aspects and modeling of multi-agent systems, and their application to several domains, e.g., mobility, smart cities, IoT, and eHealth. She received a Ph.D. in Computer Science at the Doctoral School in Information and Communication Technology in 2018, from the University of Trento and the Fondazione Bruno Kessler (FBK) in Trento (Italy). More information is available at https://martinadesanctis.bitbucket.io/index.html

Angelo Gargantini is a full professor in Computer Science and Engineering at the University of Bergamo (Italy), and he is the director of the FOSELab (Formal Methods and Software Engineering Laboratory). He received his PhD in computer engineering from the Politecnico of Milan. Before joining the University of Bergamo, he has worked for the Politecnico of Milan, the Naval Research Laboratory in Washington DC, and the University of Catania. His research focuses on automated testing techniques, model-based testing, mutation testing, and the application of formal methods in software validation and verification. More information is available at https://cs.unibg.it/gargantini/.

Patrizio Pelliccione is a Professor in Computer Science and Director of the computer science area at Gran Sasso Science Institute (GSSI, Italy). He is also an adjunct professor at the University of Bergen, Norway. His research topics are mainly in software engineering, software architecture modeling and verification, autonomous systems, and formal methods. He received his Ph.D. in computer science from the University of L'Aquila (Italy). Thereafter, he worked as a senior researcher at the University of Luxembourg in Luxembourg, then assistant professor at the University of L'Aquila in Italy, then Associate Professor at both Chalmers | University of Gothenburg in Sweden and University of L'Aquila. More information is available at http://www.patriziopelliccione.com.

Elvinia Riccobene is full professor in Computer Science at the Computer Science Department of the University of Milan (Italy), and she is the director of the FALSE Lab (Formal Methods and Software Engineering Laboratory). She received laurea and PhD degree in Mathematics. Her research interests include formal methods, with particular expertise in the Abstract State Machines, integration between formal modeling and model-driven engineering, model analyses techniques for software systems. More information is available at https://homes.di.unimi.it/riccobene/.

Patrizia Scandurra is Associate Professor at the University of Bergamo (Italy). She obtained her PhD in Computer Science (2006) at the University of Catania. Her research interests are mainly in the area of software engineering, formal methods, software architecture, and self-adaptive and autonomous systems. She is member of the FOSELab (Formal Methods and Software Engineering Laboratory). More information is available at https://cs.unibg.it/scandurra/.