Architecting Explainable Service Robots

Marcello M. Bersani[1], Matteo Camilli[1], Livia Lestingi[1], Raffaela Mirandola[1], Matteo Rossi[1], Patrizia Scandurra[2]

[1] Politecnico di Milano, Milan, Italy,
{name}.{surname}@polimi.it
[2] University of Bergamo, Bergamo, Italy,
patrizia.scandurra@unibg.it

Abstract. Service robots entailing a tight collaboration with humans are increasingly widespread in critical domains, such as healthcare and domestic assistance. However, the so-called Human-Machine-Teaming paradigm can be hindered by the black-box nature of service robots, whose autonomous decisions may be confusing or even dangerous for humans. Thus, the explainability for these systems emerges as a crucial property for their acceptance in our society. This paper introduces the concept of explainable service robots and proposes a software architecture to support the engineering of the self-explainability requirements in these collaborating systems by combining formal analysis and interpretable machine learning. We evaluate the proposed architecture using an illustrative example in healthcare. Results show that our proposal supports the explainability of multi-agent Human-Machine-Teaming missions featuring an infinite (dense) space of human-machine uncertain factors, such as diverse physical and physiological characteristics of the agents involved in the teamwork.

Keywords: Human-Machine Teaming, explainability, software architecture, statistical model checking, interpretable ML

1 Introduction

Service robots are being used for a wide range of applications such as telepresence, education, personal care, and assistive medicine [10]. In these applications, humans and robots become "peers" as they share the environment and collaborate to achieve a common goal through coordinated actions. This paradigmatic collaboration is referred to as Human-Machine Teaming [28] (HMT).

Effective teaming results from the ability of team members to coordinate their actions based on mutual *trust*. The level of trust depends on several factors, including dependability aspects and mutual understanding among agents. However, the adoption of complex control policies including Machine learning (ML) techniques often makes robotic agents "opaque", hence difficult for humans to understand [13]. According to Bersani et al., [2, 3], to achieve better trust, robotic agents must exhibit behavior that offers strong assurances, along with

human interpretable *explanations* of the expected collaboration outcome. In particular, human stakeholders need to know the main reasons for phenomena of interest occurring during the teaming, such as dependability issues or excessive fatigue of human agents. The phenomena (or *explananda*) must be understood in terms of interpretable and measurable (changing) factors [2, 3].

Recent studies focus on particular facets of explainability related to the decision-making strategies of the robotic agents [12, 29], while other teaming aspects such as those mentioned above are often neglected. Ultimately, there is still a limited understanding of systematic engineering methods that can generate useful explanations to human stakeholders. Indeed, there exist frameworks that help designers build adaptive HMT-based systems [24] by extending the MAPE-K control loop architectural style with human-related tasks and runtime models to support online teaming monitoring [6]. To the best of our knowledge, there is a lack of design guidelines for service robots realizing *explainable* HMT.

In line with M.A. Köhl et al. [18], we consider explainability as a pivotal requirement. We introduce six different levels of explainability that service robots may achieve during the realization of an HMT. We then propose a software architecture for explainable service robots that supports the (offline) specification and analysis of multi-agent HMT and the (online) generation of explanations for the phenomena of interest. Our solution combines our experience in the domain of service robots, formal verification through Statistical Model Checking [7] (SMC), and interpretable ML [27]. Explanations are generated in a collective manner—i.e., they are produced by multiple cooperating agents that collectively achieve the HMT goals. We evaluate the proposed architecture considering different explainability scenarios occurring in an existing HMT in the healthcare domain. Results show that our proposal supports explainable service robots running HMT missions with infinite (dense) space of factors, such as diverse physical and physiological characteristics of the agents involved in the teamwork.

This paper is organized as follows. In Sec. 2, we provide preliminary concepts and then we introduce an illustrative example in Sec. 3. In Sec. 4, we characterize the notion of explainability and levels of explainability in HMT. In Sec. 5, we describe our architectural solution, while we discuss a scenario-based evaluation in Sec. 6. We discuss related work in Sec. 7 and then draw conclusions in Sec. 8.

2 Preliminaries

Predictive ML models are built by using supervised learning techniques [26] to create a concise representation of the distribution of an outcome y in terms of quantifiable properties, known as *features* (or *explanatory variables*). A data point x is a vector that contains a value x_j for each feature j. A supervised learning algorithm that implements classification or regression is referred to as *classifier* and *regressor*, respectively (more in general, *predictor*). Supervised learning uses a *training set* that includes pre-labeled data points $\langle x, y \rangle$ to "learn" the desired prediction function $\hat{f}$. There exist several popular predictors (either

classifiers or regressors) in supervised ML including, for instance, Decision Trees, Random Forests, and Neural Networks.

Interpretable ML [27] refers to the extraction of relevant knowledge from an ML model concerning existing relations contained in the data or learned by the predictive function. In this context, we refer to *interpretability* (or explainability as introduced by Miller [25]) as the ability of a model to be understood and explained by humans. Some predictive models are designed to have a clear and simple structure, and their predictions are inherently explained (e.g., Linear Regression, Decision Trees). More complex techniques (e.g., Neural Networks, Random Forests) do not explain their predictions and are referred to as *black box* (or *non-interpretable*) models.

The scope of interpretability is either *global* (i.e., holistic model interpretability) or *local* (i.e., interpretability for a single prediction). Global explanations describe the average behavior of a given model. They give a holistic view of the distribution of the target outcome (e.g., class labels) based on the features. Partial Dependence Plot [27] (PDP) is a global model-agnostic method that shows the marginal effect that selected features have on the predicted outcome of a model. Local explanations, such as those produced by Local Interpretable Model-agnostic Explanation [27] (LIME), take into account an individual data point of interest x and examine the prediction $\hat{f}(x)$ to explain possible reasons based on an interpretable surrogate model. The model so built has the local fidelity property, that is, it represents a good approximation of local predictions, but it does not have to be a good global approximation.

3 Towards explainable HMT

To illustrate our approach, we adopt an example of HMT *mission* in the healthcare domain introduced by Lestingi et al. [22]. The mission features a hospital ward with an analysis room, a waiting room for patients, and a storage room with medical equipment. A service robot assists the patients and the hospital's personnel during daily operations. The robot executes the following sequence of *services* to complete the mission: (*i*) the robot *escorts* a patient from the entrance to the waiting room; (*ii*) the doctor *leads* the robot to a storage room to retrieve the equipment required for the visit; (*iii*) the robot *follows* the doctor to the analysis room while carrying the equipment; and (*iv*) the robot *escorts* the patient from the waiting room to the analysis room set up for the visit.

The example yields a highly dynamic setting in which human agents may indeed behave differently based on their own characteristics. These dynamics can be formally modeled as a Stochastic Hybrid Automata (SHA) network, an automata-based formalism that allows the specification of stochastic behavior and time-dependent physical phenomena through generalized differential equations [8]. The SHA network of our illustrative example includes five automata, together modeling the mission, the behavior of human agents (i.e., the patient and the doctor), the service robot, the physical dynamics of the battery, and

Table 1: HMT factors of our illustrative example.

Factor	Agent	Type	Domain
Free will profile	Patient/Doctor	Categorical	{*focused*, *nominal*, *inattentive*}
Health status	Patient/Doctor	Categorical	{*healthy*, *sick*, *unsteady*}
Age group	Patient/Doctor	Categorical	{*young*, *elderly*}
Walking Speed	Patient/Doctor	Continuous	$[30.0, 100.0]$ cm/s
Initial position x	Doctor	Continuous	$[0.0, 50.0]$ m
Initial position y	Doctor	Continuous	$[0.0, 8.0]$ m
Translational Speed	Robotic Device	Continuous	$[30.0, 100.0]$ cm/s
Battery charge	Robotic Device	Continuous	$[11.1, 12.4]$ V
Maximum Distance	Robot Controller	Continuous	$[5.0, 7.5]$ m
Minimum Distance	Robot Controller	Continuous	$[2.0, 4.5]$ m
Maximum Fatigue	Robot Controller	Continuous	$[0.5, 0.8]$
Minimum Fatigue	Robot Controller	Continuous	$[0.1, 0.4]$
Time Bound (τ)	-	Continuous	$[250, 700]$

the robot controller[3]. Given the SHA network, Statistical Model Checking [7] (SMC) can be used to analyze the HMT mission. For instance, the robot *succeeds* in escorting a human when both are sufficiently close to the destination. This kind of property can be expressed through a logical condition expressed in terms of network elements modeling the successful completion of a certain service provided by the robot. Hence, given a sequence of services, the mission is *complete* when all services in the sequence have been provided. In this case, the robot is *dependable* if it completes the mission within a given time bound, that is, the mission is successful. This is formalized through the Metric Temporal Logic (MTL) property $\psi = \Diamond_{\leq \tau} \bigwedge_i^{\mathsf{N_s}} \gamma_{i,\text{scs}}$, where $\gamma_{i,\text{scs}}$ models the completion of the service i in the sequence $\mathsf{N_s} \in \mathbb{N}$, $\Diamond$ is the "eventually" operator and $\tau \in \mathbb{N}$ is the time bound for the completion of the mission. Uppaal SMC [7] can be used to estimate the probability of ψ holding. In addition, Uppaal can quantify other properties, such as the fatigue of the patients. This quantity can be estimated as the maximum expected value $\mathbb{E}[\leq \tau](\texttt{max} : F_j)$, where F_j is a real-valued variable modeling the physical fatigue of a human subject j in the SHA network.

Achieving explainability in this context is typically challenging since there is a huge (dense) space of uncertain characteristics that can change and collectively affect the mission—hence the phenomena we want to explain. Table 1 lists a number of selected characteristics, hereafter referred to as HMT *factors*, with their intuitive meaning and ranges of values specific to this work. Some factors apply to the agents participating in the mission (e.g., robots or humans), while others apply to software components (e.g., the robot controller). For instance, humans may pay more or less attention to the robot's instructions according to different *free will profile*s representing their inherent attitude. People may walk at different *speed*. Each robot is managed by a controller, which decides when the robot must move or stop based on the fatigue of the patients (*min/max fatigue*) and on protective human-robot distance (*min/max distance*). Ultimately, the

[3] We let the reader refer to [22] for a comprehensive treatment of the model and its accuracy w.r.t. a real-world deployment.

Table 2: Levels of HMT explainability.

Level	Description
L1	**No explanability**: The system ignores any possible explanandum X.
L2	**Recognition of explainability needs**: The system is aware that an explanandum X for stakeholders G exists. Thus, it collects knowledge about the context C either passively or actively, by means that are deliberately designed to increase explainability through exploration.
L3	**Local explainability**: The system provides an explanation E for an explanandum X by considering a specific (punctual) operating context C to make G able to understand how the relevant individual elements of C influence X.
L4	**Global explainability**: The system provides an explanation E for an explanandum X by considering a varying operating context C to make G able to understand the extent to which changes of relevant elements of C influence X on average.
L5	**Collective local explainaibility**: The process of local explainability (L3) is realized by multiple cooperating agents that collectively achieve the mission objectives. Each agent has a partial view of the operating context C whose relevant elements are collected (and possibly analyzed) in a decentralized manner.
L6	**Collective global explainaibility**: The process of global explainability (L4) is realized by multiple cooperating agents that collectively achieve the mission objectives. Each agent has only a partial view of the operating context C whose relevant elements are collected (and possibly analyzed) in a decentralized manner.

HMT factors yield a possibly dense space $\mathcal{V}$ of elements $\bar{v}$ and, therefore, an infinite set of SHA networks $\mathcal{M}[\bar{v}]$, one for each $\bar{v}$. Hence, explainability by exhaustive exploration of the factor space is unfeasible.

4 Explainability Levels

We define explainability concerns in HMT, building upon the conceptual analysis proposed by M.A. Köhl et al. [18]. In particular, we hereby refer to an explanation E with respect to an explanandum X, a group of stakeholders G, and a context C, as the ability to make any representative of G understand X. Thus, a system is *explainable* if and only if it is able by a means M to produce an explanation E of an explanandum X for a target group G in a certain operating context C. In other words, the system satisfies a given explainability requirement, defined as a tuple $R := \langle X, G, C \rangle$. The means M that produces an explanation E to satisfy R may be part of the system responsible for X or not. When a means M is directly integrated into the system, we consider the system *self-explainable*.

In our view, the context is a composite element that contains factors characterizing relevant phenomena that may affect X according to domain knowledge. Since explanations are directed to G and constructed according to C, it is important that selected factors in C can be interpreted by representatives of G.

The notion of explainability, and in particular the characterization of context C and means M, can be given according to different (increasing) *levels*. We take inspiration from the classification introduced by Camilli et al. [5] that identifies levels of explainability of self-adaptive systems based, in turn, on the guidelines introduced by the roadmap for robotics in Europe [9]. In Table 2, we identify and

describe different levels of explainability in the HMT domain. *Collective* levels L5 and L6 (highlighted in Table 2) are the focus of this work.

To instantiate the abstract notions introduced in Table 2, we exemplify here L5 and L6 with two scenarios occurring in our illustrative HMT mission. The scenarios include multiple mission agents, stakeholders, and different explainability requirements, i.e., explananda (aspects related to mission dependability and patient fatigue), as well as contexts composed of various HMT factors listed in Table 1. We assume that this list captures domain knowledge, and therefore, contains relevant factors of the agents involved in the HMT as described in [21].

SL 5 (Patient Fatigue). The doctor (G) wants to understand the main characteristics of all agents—including the robot(s), the patient, and the doctor himself/herself—that currently affect the fatigue of the patient (X). Understanding the positive/negative impact (E) of these characteristics can suggest to the doctor how to reduce the level of stress of the patient. The context C consists of the HMT factors characterizing the agents involved in the HMT. Furthermore, C does not include factors that cannot be interpreted by the doctor (e.g., controller configuration) who is the main stakeholder in this scenario. An explanation E here may reveal that joint high doctor and high robot speed have a strong negative effect on fatigue only when the patient has unsteady health. It is worth noting that reasoning on the joint effect of factors of multiple agents is possible here because the scenario yields a collective explainability level. A non-collective level (e.g., L3, L4) would lead to short-sighted explanations based on factors of individual agents only (e.g., speed of the doctor without taking into consideration the health status of the patient), ultimately leading to reduced business impact of stakeholder decisions.

SL 6 (Mission Dependability). The system administrator (G) wants to understand what are the important configuration options of the software components (e.g., min and max distance) and how the interactions between them and the other characteristics of the agents affect the likelihood (E) of satisfying the dependability requirements of the mission (X). In this case, context C is composed of all HMT factors, including those concerning the controller configuration that can be interpreted by the system administrator. As an example, the explanation E may suggest to the administrator that the max distance configuration has almost no impact. At the same time, on average there is a linear dependency between max fatigue and the likelihood of mission success.

5 Architectural Solution

Figure 1 illustrates the key components of our architectural solution to realize collective explainability (i.e., L5 and L6) for service robots. The main building blocks are subsystems `Offline Mission Builder`, `Online Mission Monitor`, and the shared `Knowledge` repository.

The component `Specification Manager` assists the modeler in capturing the domain knowledge by creating all necessary artifacts to produce the explanations at runtime. These artifacts are stored in the `Knowledge` subsystem and

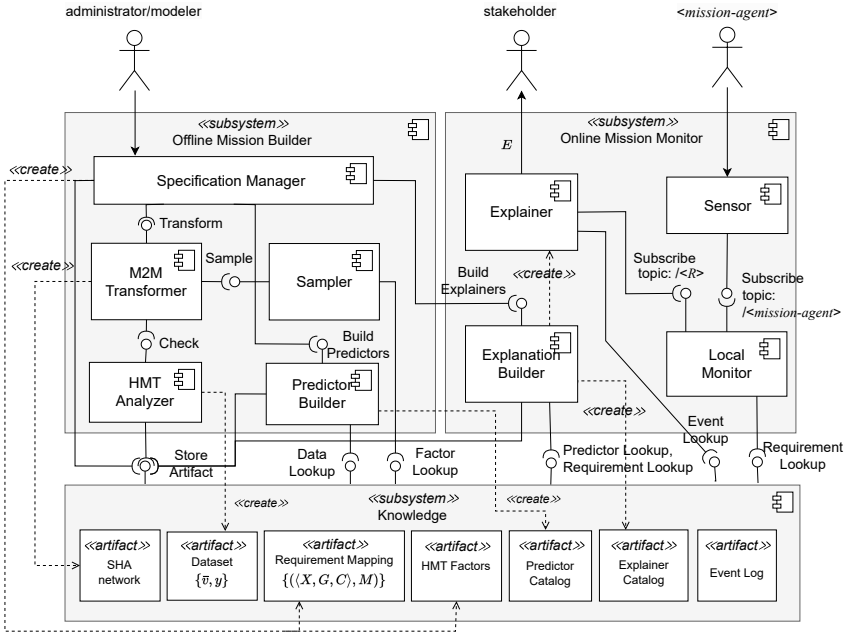


Fig. 1: Component diagram of our software architecture.

include the `SHA network` and `HMT factors` modeling the target mission, and then `Requirement Mapping` that contains a set of tuples $\langle X, G, C\rangle$ mapping to the corresponding means M used to produce explanations for X to stakeholder G, according to the context C. In our current solution, X is an MTL property that can be automatically estimated on SHA networks using statistical model checking, G is a unique identifier associated with the stakeholder, C is a nonempty subset of the HMT factors, while M is a categorical variable that identifies a particular interpretable ML technique (either global or local).

`Offline Mission Builder` uses these artifacts to build a number of alternative predictors in charge of forecasting the explanandum X under changing operating context. Then, `Online Mission Monitor` supervises an ongoing mission taking into account the information in `Requirement Mapping`. In particular, for each tuple $\langle X, G, C\rangle$, it monitors the context C and provides the stakeholder G with explanations for the predicted quantity X using the selected technique M. In the following, we describe the main subsystems in more detail.

Offline Mission Builder. The modeler triggers the offline stage by interacting with `Specification Manager`, a modeling workbench featuring a Domain-Specific Language (DSL) introduced by Lestingi et al. [22]. We use this language to specify the HMT, including the HMT factors, and the explainability requirements. Listing 1.1 shows a small DSL extract describing our illustrative exam-

Listing 1.1: Specification excerpt defining our illustrative HMT.

```
define robots:
    robot Tbot in (2300.0,400.0) type turtlebot3_wafflepi charge 90.0

define humans:
    human patient in (2300.0, 600.0) speed 40.0  is young_sick
     freewill inattentive
    human doctor in (4400.0, 700.0) speed 100.0 is elderly_healthy
     freewill focused

define mission m for Tbot:
    do robot_leader for patient with target waiting_room
    do robot_follower for doctor with target storage_room
    do robot_follower for doctor with target analysis_room
    do robot_leader for patient with target analysis_room
```

ple[4]. The fragment specifies the agents: a robot, a patient, and a doctor, each one with certain physical/physiological characteristics. As anticipated in Sec. 3, the mission is defined by a sequence of services carried out by the robot(s) and defined leveraging pre-defined templates (e.g., "robot leader", or "robot follower") instantiated for the desired agent(s) and for a target location, such as "waiting room"—that is, an alias for a location in the physical space shared by all agents.

The modeler also defines the HMT factors as a set of variables with type and domain (Table 1) and the `Requirement Mapping` by defining all tuples of interest $\langle X, G, C\rangle$ and the corresponding means M. An explanandum X is a quantitative MTL property that can be computed or estimated given the specification of the mission and an assignment to the HMT factors. For instance, in SL5, X is the maximum expected value of the fatigue of the patient. In SL6, X is an MTL property ψ whose probability $P(\psi)$ is the likelihood of mission success. The component in charge of estimating the explananda is the `HMT Analyzer`. Our current solution makes use of UPPAAL SMC given that the HMT is formally specified as an SHA network. To this end, `M2M Transformer` processes the DSL sources to generate an SHA $\mathcal{M}[\bar{v}]$, with $\bar{v}$ a valid value assignment to HMT factors according to their definition. This is carried out by a fully automated model-to-model transformation in which a set of UPPAAL templates corresponding to the elements of the SHA network are customized based on the DSL specification [22]. Our illustrative example reduces to a SHA network with structural complexity equal to $\sim 176 \times 10^3$ (calculated as the product of the number of locations, edges, and the cardinality of state variables' domains).

The objective of the `Sampler` component is to mitigate the uncertainty due to changing HMT factors by enriching `Knowledge` through a stochastic exploration[5] of the factor space rather than exhaustive enumeration (generally unfeasible). `Sampler` produces many assignments $\bar{v}$ to HMT factors. Then, the corresponding

[4] A package with full mission specification, data and sources to replicate our results is available at https://doi.org/10.5281/zenodo.8110691.

[5] Our current implementation relies on uniform random sampling.

SHA network $\mathcal{M}[\bar{v}]$ is generated and analyzed through `HMT Analyzer`. This latter component estimates the explanandum X for each tuple $\langle X, G, C\rangle$.

The analysis, executed for all $\bar{v}$ and all requirements, produces the artifact `Dataset`, which is a set $\{\langle \bar{v}, y\rangle\}$, where y is the value of X for model $\mathcal{M}[\bar{v}]$. For instance, value y in SL5 is the patient fatigue represented as a percentage (i.e., a real value in $[0, 1]$). In SL6, instead, the outcome is the mission success represented by a Boolean value (i.e., a categorical variable in $\{0, 1\}$[6]). When `Dataset` is available, `Predictor Builder` creates a `Predictor` component by training/testing a predictive model (e.g., neural network regressor/classifier) to forecast the explanandum given new HMT value assignments. A predictor is created for each explainability requirement in `Requirement Mapping`, according to the nature of X and context C. In our scenarios, we create regressors for real-value variables and classifiers for categorical variables. Context C determines the subset of HMT factors used for training. Note that we define C based on domain knowledge. Nonetheless, our solution does not prevent engineers from complementing this practice through automated techniques to feature selection.

Online Mission Monitor. This subsystem is invoked by `Mission Builder` once `Predictor Catalog` is complete and available in `Knowledge`. The `Explainer Builder` component uses each available predictor to create an `Explainer` component according to M for each tuple in texttttRequirement Mapping. An `Explainer` embeds a global/local model-agnostic interpretable ML technique to make the predicted explananda interpretable by stakeholders. Our current implementation adopts PDP for global explanations and LIME for local explanations (see Sec. 2). Once `Explainer Catalog` is ready, the subsystem initializes a publish-subscribe mechanism to realize the *collective* explainability levels introduced in Sec. 4. In particular, for every requirement R, the corresponding requirement topic `/<R>` is instantiated. Then, the corresponding `Explainer` component subscribes to topic `/<R>`. Finally, for each HMT agent, there is a `Local Monitor` component subscribed to one or more `/<mission-agent>` topics to receive sensor data. Our solution adopts the *Event Sourcing* pattern[7], whereby explanations are determined and possibly reconstructed on demand by storing all messages exchanged over topics `/<R>`. Persisting the messages enables the `Explainer` components to have a complete chronicle of past context changes.

Once the HMT application is deployed, the explanations E are realized at runtime using the publish-subscribe mechanism. Each sensor periodically samples an HMT factor and the associated `Sensor` component publishes the data to the corresponding `/<mission-agent>` topic. The subscriber `Local Monitor` can clean or aggregate raw data received by sensors. Since a `Local Monitor` belongs to an individual agent, collected data represents a subset of the HMT factors, namely a portion C' of one or more contexts. The outcomes of a `Local Monitor` are published to the identified topics `/<R>` and, therefore, received by all `Explainer`s subscribed to them. This mechanism allows each `Explainer` to run a continuous collection of the relevant HMT factors used to build the explana-

[6] Mission success occurs if $P(\psi)$ is greater than a user-defined probability threshold.
[7] https://martinfowler.com/eaaDev/EventSourcing.html

tions E to the stakeholders based on the latest context available. A stakeholder initiates a direct interaction with an `Explainer` component solely at times when an explanation is required. Through Event Sourcing, the `Explainer` components can reconstruct the temporal sequence of explanations over a specific time window by using historical data of the context retrieved from `Event Log`.

6 Evaluation

The evaluation of our approach aims to answer the following research questions:

RQ1: What is the cost of producing accurate predictors in our solution?
RQ2: Is our solution able to support explainability up to level 6?

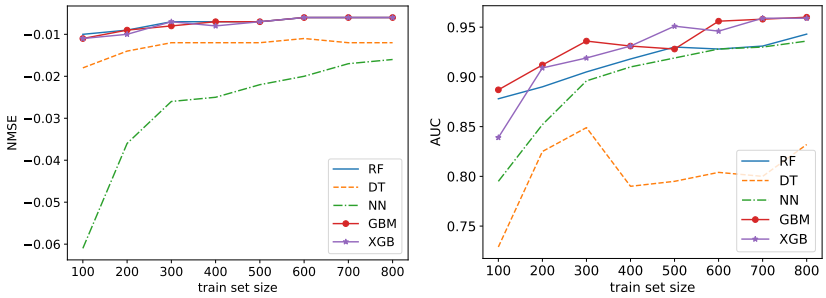
Design of the evaluation. To answer RQ1 and RQ2, we conducted an experimental campaign using our illustrative example. As reported in Sec. 5, the resulting SHA network specifying the mission is not trivial (i.e., structural complexity 10^3). To check the satisfaction of explainability requirements, we adopt a scenario-based assessment considering two selected scenarios: SL5 and SL6 (local and global collective explainability, respectively).

For each scenario, we controlled the HMT factors of interest (see Table 1) collectively composing the context C, and we generated $1k$ unique assignments $\{\bar{v}\}$ using uniform random sampling. For all $\mathcal{M}[\bar{v}]$ with $\bar{v} \in \mathcal{V}$, we used Uppaal to estimate the explanandum X in each scenario Concerning human fatigue, parametrization of the formal model has been carried out considering experiments with real human subjects [15]. Estimates of the explananda are obtained through SMC. The results have been used to create a mapping between $\bar{v}$ and the corresponding real-valued outcome $y \in [0, 1]$ (i.e., patient fatigue) as well as Boolean outcome $y' \in \{0, 1\}$ (i.e., mission failure/success). The two datasets $\{\langle \bar{v}, y \rangle\}$ and $\{\langle \bar{v}, y' \rangle\}$ have been used to feed the offline stage and study the cost of building accurate predictors. Finally, for each scenario and corresponding context C, we collected and analyzed the output E of the `Explainer` components to assess the achievement of the target explainability requirements, that is, whether E is interpretable by G and can help in understanding the explanandum X.

The experimental campaign has been conducted using a commodity hardware machine running Ubuntu OS v22.04 with 64GB RAM and a quad-core Intel x86_64 CPU at 2.1 GHz.

Results RQ1 (cost of producing accurate predictors). To study the cost in terms of execution time, we conducted multiple runs of the offline stage using $\{\langle \bar{v}, y \rangle\}$ and $\{\langle \bar{v}, y' \rangle\}$ for SL5 and SL6, respectively. For both scenarios, we considered five state-of-the-art predictors commonly adopted to address classification and regression problems: Random Forests (RF), Decision Tree (DT), Neural Network (NN), Gradient Boosting Machine (GBM), and eXtreme Gradient Boosting Tree (XGB). We refer the reader to [26] for further details about these techniques.

For each scenario, multiple predictors have been created by varying the size of the training set from 100 to 800 data points to determine the cost of achieving a relatively high and steady accuracy level. Once trained, each predictor has



(a) NMSE over size in SL5. (b) AUC over size in SL6.

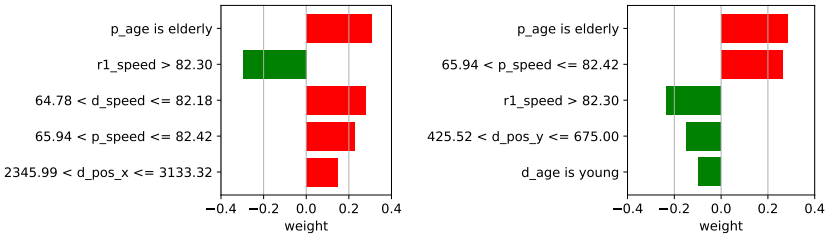
Fig. 2: Model score over training set size.

Table 3: Cost of verification and predictor train/test.

size	SMC (sec.)	train/test regressors (sec.)					train/test classifiers (sec.)				
		RF	DT	NN	GBM	XGB	RF	DT	NN	GBM	XGB
100	107805.50	0.33	**0.01**	0.30	0.23	1.91	0.16	**0.01**	0.56	0.45	1.62
200	177798.85	0.35	**0.02**	0.75	0.33	1.88	0.48	**0.02**	0.78	0.50	1.71
300	282922.82	0.46	**0.03**	0.63	0.49	1.94	0.50	**0.03**	1.17	0.81	1.78
400	373003.94	0.47	**0.02**	0.83	0.59	2.02	0.52	**0.04**	1.40	1.02	1.47
500	479403.91	0.51	**0.02**	0.81	0.72	2.09	1.86	**0.19**	1.73	1.14	1.52
600	563768.23	0.56	**0.02**	0.88	0.86	2.21	2.03	**0.05**	2.05	1.40	1.37
700	652054.33	0.54	**0.03**	0.97	0.97	2.15	2.07	**0.01**	2.65	1.47	1.37
800	756494.90	0.62	**0.03**	1.08	1.15	2.08	2.03	**0.02**	2.47	1.60	1.34

been tested using the same test set composed of 200 data points that do not belong to the training set. To measure the accuracy of regressors, in SL5, we adopt the Negative Mean Squared Error (NMSE), which is a negative value that increases to zero as the error decreases. In SL6, we adopt the Area Under the receiver operator characteristic Curve (AUC) to measure the accuracy as the discriminatory power of the classifiers [11]. The AUC ranges between 0 (worst), 0.5 (no better than random guessing), and 1 (best).

Figure 2 shows the accuracy of the predictors obtained in SL5 (Fig. 2a) and SL6 (Fig. 2b) using training sets of increasing size. We can observe that the accuracy generally increases as the size of the training set does. In both scenarios, the best predictors, stabilizes around size 600. Table 3 shows the cost considering verification and creation of predictors. Each row shows the cost of generating the dataset of a certain size as well as the train/test cost per each individual model. The most time-consuming part of the offline stage is due to the SMC being repeatedly executed for each data point. Around 6.5 days are necessary to collect 600 data points ($\sim$ 10 mins per run) and produce the predictors with the highest accuracy. The time required to train and test the predictors is always negligible compared to SMC. DT yields the lowest execution time (boldface). To further assess the accuracy of predicting the target explanandum X, we adopt 10-fold cross-validation using 600 points for the training set (as discussed above)



(a) 79% patient fatigue explanation. (b) 43% patient fatigue explanation.

Fig. 3: Explanations for SL5 using a LIME `Explainer` component.

and the remaining 400 points as the test set. After cross-validation, we created a rank of the predictive models using the non-parametric Scott-Knott Effect Size Difference (ESD) test [30]. Namely, we partitioned the set of AUC/NMSE values into distinct groups with a non-negligible difference. Consistently with the data in Fig. 2, RF is one of the first-rank regressors, and in particular it is the one that predicts the fatigue of the patient (SL5) with the highest median NMSE, equal to −0.01. Also, GBM is the first-rank classifier that predicts (un)successful missions (SL6) with the highest median AUC, equal to 0.96.

RQ1 Summary. The most expensive part of the process is offline SMC. In our scenarios around 6.5 days are necessary to collect 600 data points and achieve high and steady accuracy. The time required by training/testing is negligible (less than 3 sec.). RF is the best regressor in SL5 (median NMSE −0.01). GBM is the best classifier in SL6 (median AUC 0.96).

Results RQ2 (satisfaction of explainability requirements). To answer this question we executed the online stage and we collected the results produced by the `Explainer` components in our two selected scenarios. Then, we carried out a qualitative assessment of the explanations to determine the extent to which the target explanandum can be understood by stakeholders.

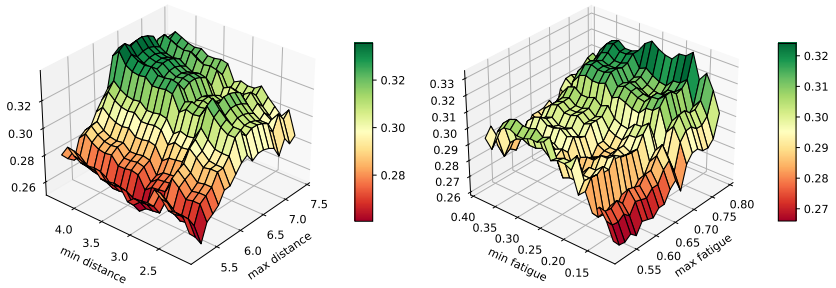
Concerning SL5, all the HMT factors that can be interpreted by the doctor (i.e., all factors except for those affecting the robot controller) are collected and then dispatched to the local `Explainer` component paired with the best `Predictor` trained to forecast the patient fatigue (i.e., RF regressor according to RQ1). Then, we adopt a LIME `Explainer` to build an on-the-fly (interpretable) local surrogate model that, given a snapshot of the context, predicts the fatigue and explains the contribution of the factors. Figure 3a shows a LIME explanation for a mission run where the patient fatigue is relatively high (79%). The plot shows the relative importance of the top 5 HMT factors and illustrates whether each value contributes to an increase or decrease in the expected fatigue level. For instance, the *elderly* age group has the highest positive weight (0.31) and, therefore, represents the main root cause of high fatigue levels. The robot speed being greater than 82.3 cm/s has the lowest negative weight (−0.29). According

to Fig. 3a, the doctor can see there are some factors under his/her own control that have a high positive contribution: a relatively high walking speed (between 65.9 and 82.4 cm/s) and the initial position x (between 2345.9 and 3133.3). The doctor can indeed inspect these values and change them to understand the extent to which these changes impact the target explanandum. Figure 3b shows the LIME explanation for a new assignment where these two latter factors have been changed to decrease the expected fatigue. We can see that the new assignment of doctor speed and position reduces their overall impact since they are not in the top 5 factors anymore. Under the new assignment, the doctor can see that the expected fatigue level in SL5 decreases from 79% to 43%.

Concerning SL6, all HMT factors (including those affecting the robot controller) are collected and dispatched to the global `Explainer` component paired with the best `Predictor` trained to forecast the mission success (i.e., GBM classifier according to RQ1). In this scenario, we adopt a PDP `Explainer` that builds explanations to understand the marginal effect of selected HMT factors on the expected probability of success. Figure 4 shows two selected PDP explanations illustrating the joint effect of *min*/*max* distance (Fig. 4a) and *min*/*max* fatigue (Fig. 4b). These factors represent system configuration options that affect the decisions of the robot (e.g., the robot stops and waits for the human when the distance is higher than the *max* value). The administrator can inspect the plots using a causal interpretation since, in this case, we explicitly model the probability of success as a function of the HMT factors. As an example, Fig. 4b shows an almost linear dependency between max fatigue and probability of success, while min fatigue affects the success with a concave function. The administrator can thus (re)configure the robot controller by selecting the ranges that maximize the expected success: max fatigue 80% and min fatigue between 25% and 35%.

RQ2 Summary. Considering our two selected scenarios, HMT factors have been collected from multiple mission agents and dispatched to the corresponding local/global `Explainer` components. In both scenarios, we illustrate the achievement of the desired explainability level by showing how the stakeholders can interpret the explanations and take decisions to influence the explanandum by changing relevant aspects of the operating context.

Threats to validity. We limited *construct validity* threats by assessing the metrics adopted in our experiments before using them. Both AUC and NMSE are widely suggested to evaluate predictive models [20]. We also use a mainstream measure of the cost in terms of execution time required by the main stages of our approach. *Conclusion validity* threats have been mitigated by reducing the possibility of overfitting on the test set by applying 10-fold cross-validation [31]. Conclusions are partially based on a qualitative assessment carried out by the authors rather than the stakeholders involved in SL5 and SL6. Comprehensive understanding of the quality of the explanations from the point of view of real stakeholders requires further investigation. We addressed *internal validity* threats by creating a testbed with fine-grained access to HMT factors to increase internal validity compared to observations without manipulation. We



(a) Joint min/max distance (b) Joint min/max fatigue

Fig. 4: Explanations for SL6 using a PDP `Explainer` component.

also adopted stratified sampling to reduce the risk of obtaining underrepresented HMT factors while building the predictors. *External validity* threats exist since our experiments consider a single case study. We limited these threats by considering an example described by existing literature as indicative of the characteristics of other HMT systems. The generalization of our findings to other domains requires additional experiments.

7 Related work

In recent years, explainability—i.e., the ability to provide a human with understandable explanations of the results produced by AI and ML algorithms—has become a key aspect of designing tools based on these techniques [1], especially in critical areas such as healthcare [33]. As such, it is attracting a growing interest in the Software Engineering community [32], as witnessed by explainable analytical models for predictions and decision-making [32], explainable counterexamples [14], and explainable quality attribute trade-offs in software architecture selection [4]. In the area of self-adaptive systems, there are preliminary approaches that aim at embedding explainability in software applications [16,17] and providing a more general approach to the construction of human-understandable explanations for successful adaptation in robotic scenarios [5]. The role of humans in self-adaptive systems has been mainly classified into "humans-out-of-the-loop" (if humans cannot change the system's behavior/outcome), and "humans-on/in-the-loop" (if they act as external controllers and supervisors [19,24], or as input providers for the system [23]). To facilitate the understanding of the system operation through explanations, humans-on/in-the-loop have been modeled using stochastic models, which undergo model checking [4,23]. Stochastic models have been applied to develop service robotic applications for which formal guarantees on the feasibility of the collaborative scenarios are obtained through SMC [22]. In such applications, the integration of explainability techniques allows both the designers of robotic scenarios and the humans involved in the interaction

with robots to understand the reasons why collaboration can fail or successfully complete [2, 3]. Although these works show an effective combination of ML, explainability techniques, and formal methods, they lack a detailed investigation of the architectural aspects involved.

8 Conclusion and future work

We addressed the problem of providing meaningful explanations in multi-agent HMT applications to foster trust by introducing six levels of explainability and presenting an architectural solution capable of providing stakeholders with human interpretable explanations based on user-specified explainability requirements. Our evaluation shows that the proposed architectural solution supports explainability up to level six. We plan to extend our solution with other factor sampling strategies based on metaheuristic optimization, in order to push the exploration of the factor space toward specific conditions of interest. We also plan to validate the approach with human participants by presenting real stakeholders with the produced explanations and having them assess their quality.

References

1. Angelov, P.P., Soares, E.A., Jiang, R., Arnold, N.I., Atkinson, P.M.: Explainable artificial intelligence: an analytical review. WIREs Data Mining and Knowledge Discovery **11**(5), e1424 (2021)
2. Bersani, M.M., Camilli, M., Lestingi, L., Mirandola, R., Rossi, M.: Explainable human-machine teaming using model checking and interpretable machine learning. In: Intl. Conf. on Formal Methods in Software Engineering. pp. 18–28. IEEE (2023)
3. Bersani, M.M., Camilli, M., Lestingi, L., Mirandola, R., Rossi, M., Scandurra, P.: Towards better trust in human-machine teaming through explainable dependability. In: ICSA Companion. pp. 86–90. IEEE (2023)
4. Cámara, J., Silva, M., Garlan, D., Schmerl, B.R.: Explaining architectural design tradeoff spaces: A machine learning approach. In: ECSA. LNCS, vol. 12857, pp. 49–65. Springer (2021)
5. Camilli, M., Mirandola, R., Scandurra, P.: XSA: Explainable self-adaptation. In: Intl. Conf. on Automated Software Engineering. ASE'22, ACM (2023)
6. Cleland-Huang, J., Agrawal, A., Vierhauser, M., Murphy, M., Prieto, M.: Extending MAPE-K to Support Human-Machine Teaming. In: SEAMS. p. 120–131. ACM (2022)
7. David, A., Larsen, K.G., Legay, A., Mikučionis, M., Poulsen, D.B.: Uppaal SMC tutorial. STTT **17**(4), 397–415 (2015)
8. David, A., Larsen, K.G., Legay, A., Mikučionis, M., Poulsen, D.B., Van Vliet, J., Wang, Z.: Statistical model checking for networks of priced timed automata. In: FORMATS. pp. 80–96. Springer (2011)
9. EU: Robotics 2020 Multi-Annual Roadmap For Robotic in Europe. https://www.eu-robotics.net/sparc/upload/about/files/H2020-Robotics-Multi-Annual-Roadmap-ICT-2016.pdf (2016)
10. García, S., Strüber, D., Brugali, D., Berger, T., Pelliccione, P.: Robotics software engineering: A perspective from the service robotics domain. p. 593–604. ESEC/FSE 2020, ACM (2020)

11. Hanley, J.A., McNeil, B.J.: The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology **143**(1), 29–36 (1982)
12. Hayes, B., Shah, J.A.: Improving robot controller transparency through autonomous policy explanation. In: HRI. pp. 303–312. IEEE (2017)
13. Jovanović, M., Schmitz, M.: Explainability as a user requirement for artificial intelligence systems. Computer **55**(2), 90–94 (2022)
14. Kaleeswaran, A.P., Nordmann, A., Vogel, T., Grunske, L.: A systematic literature review on counterexample explanation. Inf. Softw. Technol. **145**, 106800 (2022)
15. Kang, H.G., Dingwell, J.B.: Differential changes with age in multiscale entropy of electromyography signals from leg muscles during treadmill walking. PloS one **11**(8), e0162034 (2016)
16. Khalid, N., Qureshi, N.A.: Towards self-explainable adaptive systems (SEAS): A requirements driven approach. In: Joint Proceedings of REFSQ. CEUR Workshop Proceedings, vol. 2857. CEUR-WS.org (2021)
17. Kordts, B., Kopetz, J.P., Schrader, A.: A framework for self-explaining systems in the context of intensive care. In: ACSOS. pp. 138–144. IEEE (2021)
18. Köhl, M.A., Baum, K., Langer, M., Oster, D., Speith, T., Bohlender, D.: Explainability as a non-functional requirement. In: RE. pp. 363–368. IEEE (2019)
19. de Lemos, R.: Human in the loop: What is the point of no return? In: SEAMS. p. 165–166. ACM (2020)
20. Lessmann, S., Baesens, B., Mues, C., Pietsch, S.: Benchmarking classification models for software defect prediction: A proposed framework and novel findings. IEEE Trans. on Software Engineering **34**(4), 485–496 (2008)
21. Lestingi, L., Askarpour, M., Bersani, M.M., Rossi, M.: A deployment framework for formally verified human-robot interactions. IEEE Access **9**, 136616–136635 (2021)
22. Lestingi, L., Zerla, D., Bersani, M.M., Rossi, M.: Specification, stochastic modeling and analysis of interactive service robotic applications. Robotics and Autonomous Systems **163** (2023)
23. Li, N., Cámara, J., Garlan, D., Schmerl, B.R., Jin, Z.: Hey! preparing humans to do tasks in self-adaptive systems. In: SEAMS. pp. 48–58. IEEE (2021)
24. Madni, A.M., Madni, C.C.: Architectural framework for exploring adaptive human-machine teaming options in simulated dynamic environments. Systems **6**(4) (2018)
25. Miller, T.: Explanation in artificial intelligence: Insights from the social sciences. Artificial intelligence **267**, 1–38 (2019)
26. Mitchell, T.M.: Machine Learning. McGraw-Hill, Inc., USA, 1 edn. (1997)
27. Molnar, C.: Interpretable Machine Learning. 2 edn. (2022), https://christophm.github.io/interpretable-ml-book
28. Ozkaya, I.: The behavioral science of software engineering and human–machine teaming. IEEE Software **37**(6), 3–6 (2020)
29. Paleja, R., Ghuy, M., Ranawaka Arachchige, N., Jensen, R., Gombolay, M.: The utility of explainable AI in ad hoc human-machine teaming. In: NEURIPS. vol. 34, pp. 610–623. Curran Associates, Inc. (2021)
30. Scott, A.J., Knott, M.: A cluster analysis method for grouping means in the analysis of variance. Biometrics **30**(3), 507–512 (1974)
31. Stone, M.: Cross-validatory choice and assessment of statistical predictions. Journal of the Royal Statistical Society: Series B (Methodological) **36**(2), 111–133 (1974)
32. Tantithamthavorn, C.K., Jiarpakdee, J.: Explainable AI for software engineering. In: ASE. pp. 1–2. ACM (2021)
33. Tjoa, E., Guan, C.: A survey on explainable artificial intelligence (XAI): Toward medical XAI. IEEE Transactions on Neural Networks and Learning Systems **32**(11), 4793–4813 (2021)