Conceptualizing Digital Twins for Decision-Making: Is my Digital Twin Behaving?

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

5th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—Digital Twins (DTs) have emerged as a platform to underpin and drive the decision-making of its Real Twin (RT). The quality of the decision-making provided by the DT depends upon the degree of its fidelity with respect to the RT and will ensure the RT's behaviour is inside the envelope of acceptability (EoA). A DT is seen as an abstraction of its Real Twin (RT), where the decision support is at heart. The paper offers a conceptualisation of DTs that serves as a reference architecture to guide the instantiation of technical solutions. The conceptualisation offered allows stakeholders a common ground to develop technical solutions while focusing on the feedback loop for decision-making. Particularly, the paper focuses on the concept of the *Fidelity of DTs* and the fidelity assessment as a way to quantify and appraise how well the properties mirrored by a DT are aligned with the properties of the Real Twin (RT) to ensure the acceptability of the resultant behaviour. The proposed conceptualisation has been applied to the characterisation of classes of applications. As a proof of concept, the fidelity assessment has been applied to two different substantial cases to evaluate the fitness of a DT to fulfil its role.**

***Index Terms*—Digital Twins, Runtime Models, Fidelity, Decision-Support**

I. Introduction

Internet-connected devices surround us everywhere: from wearable electronics [1], [2] to networked IoT systems used in Agriculture [3], Transportation [4] among other domains (Digital Health, Manufacturing, Banking, Smart Cities [5]). The high number of sensors connected generates large amounts of data. Generated data is replicated to the Digital Twin (DT), e.g. using Machine Learning. The DT is used to provide analysis and to predict possible scenario outcomes. Essentially, DTs are used as decision-support systems [6], [7] to enable the DT to influence the RT by informing and/or issuing control inputs [8]. The quality of the decision-making provided by the DT depends upon the degree of its fidelity [5] with respect to the RT. Given the broad number of domains, a wide range of different purposes and stakeholders that use DT technologies, the terms used can be confusing. Like other initiatives [9]–[13], this paper conceptualises DTs and the Twining Process with its RT that serves as a reference architecture to guide the instantiation of technical solutions. The authors argue that the main goal of a DT is to support decision-making about how the RT would need to react during specific scenarios or, at least, to synthesize insights that would be leveraged by the RT [14]. The DT meets these goals by the use of different techniques such as predictive [8], [12], what-if-analysis [15], [16] and simulations [17], [18].

A concern in the area of research on decision-making is to find techniques that ensure the resultant behaviour of the system is inside of the Envelope of Acceptability (EoA) [19], [20]. Therefore, an issue that has increasingly attracted the attention of researchers is the assessment of the fidelity of the DT with respect to the RT [21], [22], whose behaviour is affected by the recommendations and adaptations dictated by the DT. This paper focuses on assessing the fidelity of DTs, i.e. the accuracy of the properties mirrored by the DT with respect to the RT's properties.

The main contributions of the paper are two-folded:
1) A conceptual framework for DTs that focuses on feedback loop for decision support provided by the DT. The DT mirrors the RT's properties at the abstraction level defined by the models used. The conceptualisation serves as a reference architecture to support communication among stakeholders to decide on technical and design solutions. The conceptualisation enables the use of different techniques for *fidelity* quantification according to the role of the DT and the domain.
2) A technique for quantifying the *fidelity* of the DT based on Logistic Regression.

As a proof of concept, we have applied (1) and (2) to two substantial examples from the networking and IoT domains [23], [24] taken from software artefacts available for experimentation).

The paper is organized as follows: Section II presents the background on decision-making under uncertainty and DTs. Section III presents our proposed Conceptual Framework for DTs for Decision-Support. Section IV describes the assessment of Fidelity of the DTs and experimental evaluations. Section V contrasts the contributions of the paper against related work. Section VI concludes the paper and outlines future research opportunities.

II. Background

This section provides the background for decision-making under uncertainty and Digital Twins as follows:

A. Decision-Making under Uncertainty

The decision-making process is usually modelled as a decision-making agent which performs actions based on the observations from its potentially volatile environment [25], as shown in Fig. 1. The decision-making agent refers to either physical entities such as humans or robots, or it could also be a decision-support software-based system [7]. These decision-making agents can be designed using different approaches based on machine learning (supervised, unsupervised and reinforcement learning) [26], optimization [27], fuzzy inference [28] or simply rule-based techniques.

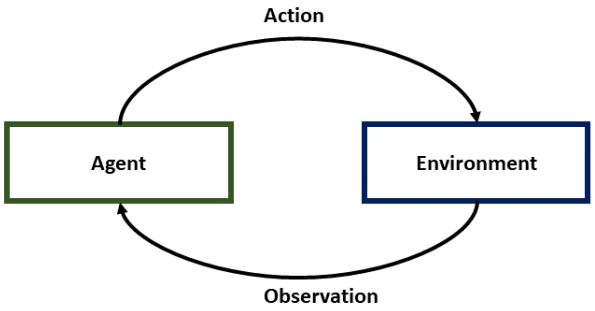

Fig. 1: Agent based Decision-Making Process (from [25])

Decision-making under uncertainty [25], with respect to the role of humans in the decision-making process, can be seen in a broad spectrum, which can be divided into four regions: (a) automated functions collect and aggregate information to support human decisions. (b) automated functions for decision-making can suggest possible courses of action for humans to consider. (c) if automation technologies can be entrusted autonomic systems with making—and acting on—lower-level decisions and humans need to make relatively less frequent, predominantly higher-level decisions, the system will carry out automatically, Finally (d) where the system can be entrusted to fully autonomous decision making that does not require explicit human control. Decision-making with the human-in-the-loop or human-machine teaming [29] is represented by (a), (b), and (c), while (d) entitles autonomous decision-making [30].

As Fig. 1 suggests, the decision-making agent uses feedback control loop. Therefore, the architectural pattern Monitor-Analyze-Plan-Execute over the Knowledge base (MAPE-K) has been proposed [31], [32]. The MAPE-K vision was first introduced by IBM [31]. The feedback control loop has four functional phases: *Monitor*, *Analyze*, *Plan* and *Execute*. All of these phases share common *Knowledge* to support the decision-control mechanism. Hence, this architectural control loop is known as MAPE-K loop in short. The phases of the MAPE-K loop are described as follows:

a) Monitor: During the *Monitor* phase the managing system observes and collects data from the managed system. This collection of data is done using sensors connected to the managed system.

b) Analyze: During the *Analyze* phase the managing system performs inspection and pre-processing of the monitored data to support the *Plan* phase.

c) Plan: Based on the result of *Analyze*, during the *Plan* phase, the decision-making agent selects the actions to be performed based on predictions and optimization.

d) Execute: During the *Execute* phase the decision-making agent performs the planned actions over the managed system.

e) Knowledge: The knowledge required by the managing system to support all phases of the loop, i.e. the data of the managed system and its environment, as well as goals and other relevant states shared by the other MAPE components [33].

Furthermore, according to [31] the MAPE-K vision for decision-support comprises two main components [34], which are i) the managed system that would eventually be the RT and ii) the managing system. The managed system refers to the application logic. On the other hand, the managing system refers to the decision-making agent, which drives the feedback control loop. When there is a substantial degree of autonomous decision-making (such as in (c) and (d) described above), self-adaptation and self-management capabilities are offered to cope with the uncertainty that will be solved at run-time [19], [34], which is the case of Self-Adaptive and Autonomous System (SAS). Uncertainty arises due to different reasons, the stochastic nature of events in the environment, limited sensor capabilities, and difficulties in predicting how the modification of system services will affect the system's behaviour and the system goals [34]. Therefore, based on the observations of its environment, the decision-making agent performs actions to deal with environmental uncertainty and also satisfy the system's objectives [35], [36].

B. Digital Twins

According to [37], a DT of an original real system (i.e the RT) comprises (1) a set of models of the system that mirror the original real system, (2) a set of digital shadows that include a set of data traces collected, aggregated and abstracted for a specific purpose with respect to the original real system, and (3) a set of services to use the models and data with respect to the original real system (i.e. the RT) for a specific purpose.

A number of conceptualizations of DTs have been developed [9]–[11]. According to these conceptual frameworks and the definition presented above, the DT system comprises three main components: 1) the DT, 2) the Physical Twin (PT), and 3) the connections between them that are typically represented through data flows. In [38], a Models and Data Framework, called MODA, is presented. According to the MODA framework, different models exist that have a purpose and play different roles: *Descriptive*, *Predictive* and *Prescriptive* depending on the purpose of the model.

-A *Descriptive* model refers to the representations of current and/or past aspects of the actual system (i.e. the RT) to facilitate analysis and planning.

- A *Predictive* model supports the prediction to allow decision-making.

- A *Prescriptive* role refers to driving the application of the actions on the actual system. Furthermore, the Data in the MODA framework refers to the three kinds of data: input/output data, measured data, and external data for the real system. The data is processed by *Descriptive*, *Predictive*, and *Prescriptive* models in order to provide adaptation actions for the real system [38].

According to [12], based on the MODA framework, a DT is defined as a virtual representation of an Actual System (i.e. the RT) that is continuously updated with data throughout its life-cycle and, at the same time, can interact with and influence the Actual System. The conceptual framework, offered in [12], describes the DT and its components: Data and Models (Descriptive, Predictive and Prescriptive), as shown in Fig. 2. The conceptual framework comprises a DT connected to an Actual System that exists within its environment. The Actual System produces data that are related to different aspects of the system, and the DT captures the data and uses a set of models to carry out different operations/actions on the Actual System. Based on the conceptual framework, a DT consists of a set of models and data associated with the Actual System that enables the creation of a set of services, corresponding to the functionalities provided by the DT. For instance, the DT can use simulation to support decision making for maintenance the Actual System based on availability of resources or new user requirements. The Models and Data components of the DT are described as follows:

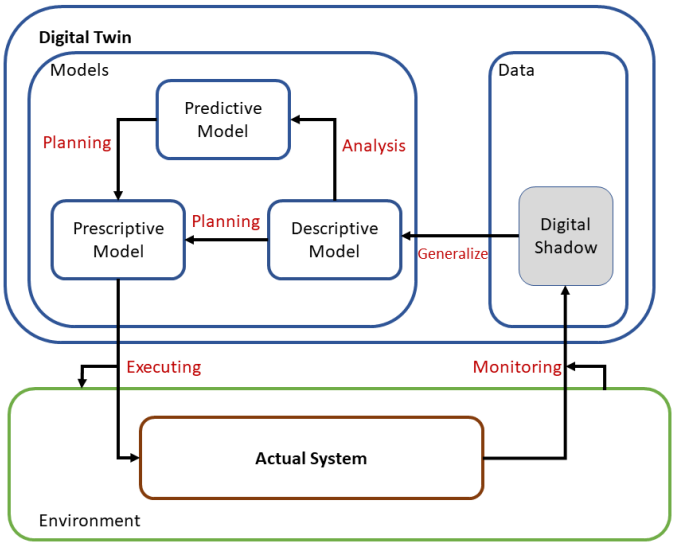

Fig. 2: Digital Twin Framework based on MODA Framework [38] (from [12])

1) ***Models*** *:* The Models in a DT focus on different aspects of a system. The models are of three different types: Descriptive model, Predictive model, and Prescriptive model as shown in Fig. 2. A descriptive model reflects the system or the system's environment in a descriptive manner, representing current or past aspects of the Actual System to support understanding and analysis of the Actual System. The predictive model predicts information that has not been measured, allowing trade-off analysis and supporting decision-making for the Actual System. The prescriptive model is a description of the system to be realized and refers to the prescriptive measures being carried on the Actual System.

2) ***Data*** *:* The data component of the DT presents representation of the current and past data about the Actual System. The Data is used to reflect the different aspects of the Actual System in the models of the DT. Moreover, the DT system also allows the dataflow between the models (descriptive, prescriptive and predictive) to support the reflections offered by the DT for the Actual System. The dataflow consists of the monitoring and generalizing of the data gathered from the Actual System, data flow for analysis and planning and execution of the control mechanisms offered by the DT for the Actual System. The dataflows are presented by the arrows in the Fig. 2. Moreover, the data component also contains Digital Shadows as data structures which offer one-way dataflow between the state of the Actual System and the DT.

In [39], a conceptual, high-level framework for using DT as run-time predictive models for resilience, self-healing and trustworthy autonomy of CPS is presented. The framework integrates the MAPE-K self-healing loop into the DT at multiple levels of the CPS.

Next, we present our conceptualization of DTs for decision-support.

III. Conceptualization of Digital Twins for Decision-Support

This section describes a conceptualization of the DTs for decision support. A DT conceptualisation framework is provided and uses the MAPE-K loop to drive the flow of data and mirror the properties of the RT, therefore, supporting decision-making for the RT. The DT conceptualisation framework for decision-support is presented in Fig. 3. The DT framework comprises two main twined components, which are described as follows:

1) Real Twin (RT): represents the Actual System which is known as the *managed system* in MAPE-K loop. The RT is monitored and is the recipient of the adaptations when they take place.

2) Digital Twin (DT): represents the *managing system*, as in MAPE-K loop. The Knowledge base K of the DT holds the run-time models that support the MAPE feedback control loop i.e. *Monitor*, *Analyze*, *Plan* and *Execute* over *Knowledge* base.

Defining the Twining Process as in Fig. 3 enables stakeholders to explicitly model the co-evolution of the RT and its DT over time. The DT learns about the state of the RT based on the incoming observational data *(M)*. The internal DT run-time models are updated based on the incoming data, to be then evaluated to provide accurate predictive and simulation analysis via the Analysis *(A)* and planning *(P)*. This enables the DT to update the RT accordingly by informing and/or

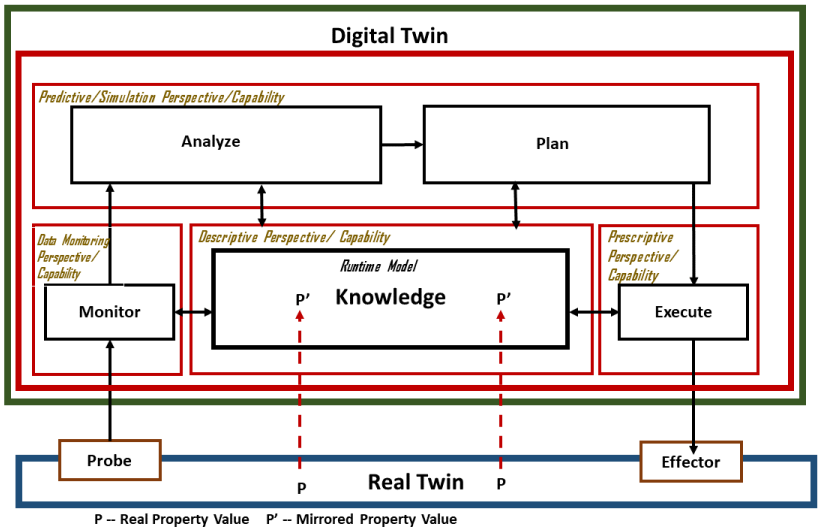

Fig. 3: Conceptual Framework for Digital Twins for Decision Support

executing *(E)* control inputs. This two-way coupled system is modelled as a causally-connected system supported by the run-time models [40], [41], which is defined next:

Definition 1: ***Causally-connected Twining RT / DT***: is supported by the run-time models in the DT, and means that if the RT changes, the run-time models in the K Base should also change accordingly (via the monitored data *(M)*). Vice versa, if the run-time model is updated (due to results from predictive capabilities in *(A)* and *(P)*), then the RT will change accordingly (via the enacting *(E)*).

The different perspectives and related capabilities offered by the DT according to the proposed conceptualisation are described next:

- ***Descriptive Perspective***: The DT framework uses run-time models as the Knowledge Artefacts to consistently mirror the RT's properties that describe the current state of the RT. The DT capability, offered by the DT, to maintain the mirrored value representations of the RT constitutes the *Descriptive Perspective* of the DT.

The run-time models also underpin the *Predictive/Simulation* and *Prescriptive* capabilities to carry out the *Analyze*, *Plan* and *Execute* steps.

- ***Predictive/Simulation Perspective***: The *Predictive/Simulation* capability supports the DT to be able to analyse the current state of the RT (via the run-time models in the *Knowledge Base*) and make predictions to plan the adaptations.

- ***Prescriptive Perspective***: the associated capability of the DT enables it to carry out the execution of the selected adaptations on the RT.

When working with a causally-connected Twining RT / DT, simulations and adaptations can be exercised at the DT level (via the run-time model) rather than directly on the RT [40], [42], [43] providing risk assurance. Furthermore, to support the agent-based decision mechanism, which is applied in the phases of Analysis and Planning, different techniques can be used based on the domain problem, e.g. based on machine learning, optimization, planning, reinforcement learning or fuzzy inference [25]. These techniques will allow accurate predictions and simulations using the mirrored values in the DT. Different techniques will state different ways to measure Fidelity, as will be discussed and illustrated in the next section.

The properties of the RT, which the DT mirrors, can be either dynamic or static. Contrary to the static properties, dynamic properties evolve over time, and so are the corresponding mirrored properties by the DT. Based on these types, there are two kinds of DTs: DTs with **Causally-connected twining DTs** and **Shadowing DT**.

Dynamic properties, therefore, are associated with DTs that drive a feedback loop MAPE-K where the RT and the DT have a Causally-connected twining to provide the mirroring (See Definition 1).

As for static properties, they will not change over time and, therefore, won't have a feedback loop associated. They are associated with Shadowing DTs.

Definition 2: ***Shadowing DT***: A shadowing DT mirrors the static properties of the RTs with no causal connection. They lack the full causal connection as they don't perform the execute step of the MAPE-K loop.

Examples of Shadowing DT are those used by experts to gain further understanding and insights about the RT without the need to enact adaptations. Examples include DTs that involve simulations: in the healthcare domain as presented in [44], and bio-medical science [22], where the traces of the protein structures are mirrored and aligned with the real structure. The focus of our proposed conceptual framework in this paper is on the Causally-connected DTs.

A. Run-time Models and Mirrored Property Values

Run-time models provide abstractions of the run-time phenomena [40]. The DT framework that we propose offers mirroring capabilities for the RT according to the run-time models at the level of abstraction that they offer, which is based on the RT's properties modelled and mirrored by the DT. The RT properties to be modelled will depend on the *purpose* and domain problem of the DT. Hence, the DT using its run-time model's *Descriptive* capability maintains the Knowledge in the form of *Mirrored Property Values* for the *Real Properties Values* of the RT, as shown in Fig. 3. Note that the more aligned the *Mirrored Property Values* are to the *Real Property values*, the higher the *fidelity* is. Hence, the assessment of these alignments serves as a key for the assessment of the quality of the DT system.

Next, we present an application example for our proposed conceptual framework for DTs:

B. Application of the Conceptual Framework

Recently, techniques based on Reinforcement Learning (RL) [45] have been used to support decision-making in the case of Self-Adaptive and Autonomous Systems (SAS) [46]–[48]. Hence, for the application of the DT conceptualization framework for decision support, we focus on a DT that makes use of RL for decision-making for self-adaptation. The RL models follow a feedback loop based on the interactions between the agent and the environment, as presented in Fig. 4, which is a specialisation of the agent-based decision-making process shown in Fig. 1. In the case of a SAS, we are referring to a causally connected Twining RT / DT as defined previously.

The agent in the RL process represents the component that interacts with the environment and is responsible for making decisions about the actions [45]. Based on the observations monitored about the current state of the environment, the agent acts to achieve the desired goal. Based on the actions performed by the agent, the agent gets a reward as a feedback signal that indicates how good the action is with respect to achieving the desired goal. This process is continuous using the feedback loop. According to the conceptualisation offered in this paper (see Fig 2) and applied to Fig 4, the Agent is the DT, and the Environment is the RT.

Therefore, according to Fig. 1, during the decision-support mechanism, the RL models maintain *mirrored property values* of the state of the properties of the RT (i.e. the Environment in the RL process) that the DT (i.e. the Agent in the RL process) observes. These *mirrored property values* can be in the form of beliefs maintained by the RL models [49]. The belief is actually the *Knowledge* inferred based on the data collected about the state of the RT, which the RL run-time model maintains. Hence, the maintenance of the mirrored belief values about the state of the RT refers to the *Descriptive Capability* of the DT, which the RL run-time models underpin. Furthermore, the RL model supports the Analysis step of the MAPE-K loop based on the observations and the run-time ***Knowledge*** in the form of the mirrored values. Based on this information, Planning is performed using the RL technique to select the best adaptation action to be applied on the RT. Both Analysis and Planning steps are part of the *Predictive* capability of the DT and applied on the mirrored belief values provided by the RL run-time models. Once the adaptation actions are selected, a policy for the adaptation is formulated using the RL logic, which is then executed in the form of applying an adaptation action on the RT. The policy formulation and execution of the adaptations are performed by the *Prescriptive* capability of the DT. The execution of the adaptation action should then result in the improvement of the state of properties associated with the RT to meet its goal.

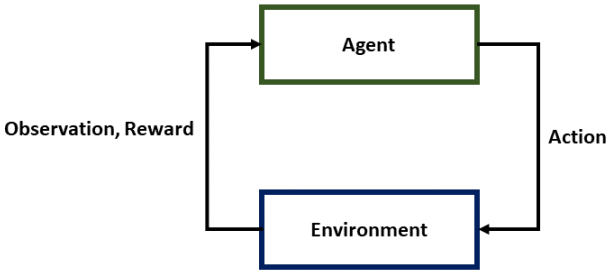

Fig. 4: Reinforcement Learning Model: Feedback Process (from [45])

IV. Assessing Fidelity of the DT

As described before, a DT mirrors the specific properties of the RT at the given level of abstraction, which the model of the RT properties dictates. For example, the mirrored properties to be modelled can be the quality properties of the RT such as performance and reliability [50], [51]. The mirrored properties can also be about the behaviour and operations of the RT [52]. As discussed previously, the assessment of the quality of the DT would depend upon the degree of its fidelity. Hence, the fidelity of the DT can be assessed by evaluating how accurately the *mirrored property values* maintained by the DT mirror the *real property values* of the RT.

In this section, we present an approach to assess the fidelity of the DT based on RL to provide decision support for SASs. The *purpose* of the DT studied is to provide mirroring of the quality properties (such as performance and reliability) of the RT (SAS in this case) [23], [24]. Based on the state of the quality properties, the DT provides decision-support for adaptations to keep the system in the Envelope of Acceptability [53] with respect to satisfying its quality properties.

Next, we present the experimental evaluations for assessing the fidelity of DTs that are equipped with RL models. The goal is to assess the fidelity of the mirrored RT's properties (in the cases studied, they are quality properties such as performance, reliability, or minimisation of energy consumption). We have applied the evaluation to two example cases from networking and Internet of Things (IoT) domains [23], [24]. The authors have been able to use the public software artefacts of the applications [23], [24] for evaluation purposes, which have offered us public repositories for the simulations and data

needed for experimentation. The goal of the experiments is to assess how well the DTs, which use the conceptualisation provided and are equipped with RL runtime models, mirror the quality properties of the RT.

As discussed in Section III-B, the RL models maintain a proxy model of the properties as a belief, which os the *mirrored property value* of the real values of the property in the RT. The state of the RT is represented by the values (state) of the quality properties. Our research question is formulated as follows:

"*How accurate are the mirrored property values (using the RL model for the DT) with respect to the RT ?*"

With the experiments performed, the goal is to check how well the mirrored values of quality properties are aligned with the actual value of the quality properties of the RT over time. Here, the fidelity of the DT refers to the extent to which the mirrored values are a faithful representation of the real state of the quality properties for the SAS. This relationship between real observations related to the quality properties and the mirrored values is assessed using Logistic Regression [54]. Logistic Regressionis applied to evaluate the decision-support for adaptations offered by the DT for the case studies [23], [24]. It considers both the *mirrored values* maintained by the RL model and the network parameters' values (representing the *real property values*) coming from the network environments [23], [24]. By using Logistic Regression, we want to check how good the *mirrored values* in RL models are in reflecting the state of the quality properties for the RT based on these values. In the DT, the state of the quality property is considered as satisfied (i.e. True) if it meets the required acceptability threshold and vice versa. Due to the binary nature of the state (i.e. True or False) of the quality properties [48], [55], Binary Logistic Regression [54] has been selected for evaluation purposes. It helps study the fidelity of DT to reflect the states of the quality properties using simple binary classification. Moreover, the Softmax function [56] is applied during Logistic Regression to determine the extent to which a particular *mirrored value* of the DT belongs to a particular quality property state class (True or False).

Next, we provide the hypotheses for the experiments:

Hypotheses:

The hypotheses for this experiments are as follows:

H_0: "During the decision-making process for SASs, the state for the quality properties are not accurately mirrored by the property values maintained by DT."

H_a: "During the decision-making process for SASs, the state for the quality properties are accurately mirrored by the property values maintained by the DT."

TABLE I: RDM Case: DataSet Format

Bandwidth Consumption	Mirrored Value	Actual state
2376	0.88656982	True
4050	0.904919218	False
2465	0.878517531	True
2320	0.902641707	True
1080	0.906088102	True

The next subsections provide experiments for the two example cases.

A. Example Case 1–Remote Mirroring Network

The first example case is based on the Remote Mirroring application [23]. For the purpose of evaluations, we have used an opensource artefact of RDMSim [23] to provide simulations of the remote mirroring network. Based on the specifications on the artefact, we consider the three quality properties: Minimization of Operational Costs (MC), Maximization of Reliability (MR) and Maximization of Performance (MP). The network parameters to measure the MC, MR and MP are bandwidth consumption, active links and writing time for data respectively. This networking application represents the RT. The DT is based on the RL model to provide *Data Monitoring*, *Descriptive*, *Predictive* and *Prescriptive Capabilities*. The Logistic Regression model is used to assess the fidelity of the mirrored values by the RL model. It considers both mirrored values maintained by the RL for the quality properties and the values of the network parameters coming from the network (as the real quality properties values) to assess the fidelity. By using Logistic Regression, we want to check when both the features are considered, to what extent they represent a specific state of a quality property.

B. Experimental Setup

The experimental setup for the Logistic Regression model is as follows:

Experimental DataSet: The dataset for the experiments is of the format presented in Table I. The input and output features are presented as follows:

Input Features: 1) Value of the Network Parameter such as Bandwidth Consumption, Time to Write Data and Active Links generated by the network. These values represent the RT's quality property values. 2) Mirrored value for the state of the quality property.

Output Variable : Actual State for quality property having a value for True when a quality property is satisfied and False otherwise.

For the experiments, the training data set consists of results of the decision-support for adaptive control offered by the DT empowered by RL runtime model. The results comprise of 4 simulation runs of *RDMSim*. Each run was executed for 500 simulation time steps. These simulation runs represent an execution trace of the *real property values* and the *mirrored values* over time. Based on the execution trace, we assess how well the the *mirrored property values* are aligned to the *real property values* (i.e. network parameter values in our case). As a result, the training set is composed of 2000

TABLE II: RDM Case: Learning Rate Accuracy Score

Learning Rate	Accuracy Score MC	Accuracy Score MR	Accuracy Score MP
0.0001	0.96241	0.98496	0.93985
0.0003	0.97995	0.99749	0.96742
0.0005	0.98496	0.99749	0.97494
0.0007	0.98496	0.99749	0.98496
0.0009	0.98747	0.99749	0.98747
0.001	0.98747	0.99749	0.98747
0.003	0.99248	0.99749	0.99499
0.005	0.99749	0.99749	0.99499
0.007	0.99749	0.99749	0.99749
0.009	0.99749	0.99749	0.99749
0.01	0.99749	1.0	0.99749
0.03	1.0	1.0	1.0
0.05	0.99749	1.0	1.0
0.07	0.99749	1.0	1.0
0.09	0.99749	1.0	1.0

training examples. For the purpose of tuning of the parameters of the Logistic Regression model, we have used Stochastic Gradient Descent (SGD) Algorithm. To tune the parameters, we have applied cross-validation by splitting the training set into 80/20 proportions. After tuning of the parameters using cross-validation, we have executed results for the test set that comprises the newly generated execution trace comprising 500 simulation time steps. The data set for the experiments is available at [57].

C. Experiments

The experiments have been executed to evaluate results for different scenarios provided by *RDMSim* that represent different dynamic environments for an RDM network, Due to the limitation of space, in this paper, we present the evaluation of one of the dynamic scenario for the *RDMSim* network known as Scenario 1 that represents the dynamic situations of unexpected packet loss during the execution of MST topology. Evaluation results for the rest of the scenarios are reported in [57]. The tuning of the parameters for the Logistic Regression and experimental evaluations are discussed as follows:

***Parameter Tuning**:* We have executed SGD for 10,000 iterations with different learning rates for the results computed for each quality property separately. On the basis of the accuracy scores presented in Table II, we have selected the learning rate of 0.03 for the model to evaluate the test set.

***Classification Results**:* The classification results for the quality properties using logistic regression are presented in Tables III, IV and V. The results show an accuracy score of 0.986 for MC, 0.998 for MR and 1.0 for MP with a precision of 1.0 for MC and MP, and 0.9967 for MR. The F1 scores for MC, MR and MP are 0.9889, 0.9983 and 1.0 respectively.

***Classification Results for MC**:* For the execution trace of 500 simulation time steps, under Scenario S_1 of RDMSim, the model correctly classifies the state of MC with the exception of 7 simulation time steps, as shown in Fig. 5. For these 7 simulation time steps, the actual state for MC was True but it was predicted as False.

Moreover, based on the results presented in Table III, we can deduce the extent to which MC can be considered as satisfied, when both the bandwidth consumption and mirrored value are considered. For example, at time step 482, the bandwidth consumption is 5550 GBps and the *mirrored value* is 0.7255. The state predicted by the Logistic Regression technique is also False with the 1.00000000e+00 probability of being False. It means that given the input values, there is 100 percent chance of MC not being satisfied i.e. having state of False.

***Classification Results for MR**:* For the whole execution trace comprising 500 simulation time steps comprising the test set, the Logistic Regression technique correctly classifies the state of MR with an exception of only 1 simulation time step, as shown in Fig. 5. For this 1 simulation time step, the actual state for MR was False but it was predicted as True.

Moreover, based on the results presented in Table IV, we can deduce the extent of the state of MR, when both the active links and mirrored value are considered. For example, at time step 482, the active links are 222 and the mirrored value is 0.95869. The output state predicted by the Logistic Regression technique is True similar to the actual state with the 1.000e+00 probability of being True. It means that given the input values, there is 100 percent chance of MR being satisfied i.e. having state of True.

***Classification Results for MP**:* For the execution trace comprising 500 simulation time steps in the test set, the Logistic Regression technique correctly classifies the state of MP as shown in Fig. 5.

Moreover, based on the results presented in Table V, we can deduce the extent of the state of MP, when both the writing time and *mirrored value* are considered. For example, at time step 482, the writing time is 2220 ms and the *mirrored value* by the DT is 0.7165. The output state predicted by the Logistic Regression is True similar to the actual state with the 0.99878 probability of being True. It means that given the input values, there is around 99 percent chance of MP to be satisfied i.e. having state of True.

TABLE III: RDM Case: Example Classification Results for MC for time steps 482-488

Step	Bandwidth Consumed	Mirrored Value	Actual State	Predicted State	Probability per class
482	5550	0.7255	False	False	[1.000e+00 4.2838e-10]
483	4833	0.7834	False	False	[9.9999e-01 1.04155e-06]
484	4725	0.7918	False	False	[9.9999e-01 3.3652e-06]
485	2625	0.8642	True	True	[5.3886e-05 9.9995e-01]
486	1944	0.9024	True	True	[3.5155e-08 9.9999e-01]
487	5175	0.8672	False	False	[9.9999e-01 4.0101e-08]
488	3525	0.8732	True	True	[0.4058 0.5942]

TABLE IV: RDM Case: Example Classification Results for MR for time steps 482-488

Step	Active Links	Mirrored Value	Actual State	Predicted State	Probability per class
482	222	0.95869	True	True	[2.3038e-17 1.0000e+00]
483	179	0.96449	True	True	[2.5793e-11 1.000e+00]
484	225	0.95236	True	True	[8.707e-18 1.000e+00]
485	125	0.8378	True	True	[9.898e-04 9.9901e-01]
486	81	0.766	False	False	[9.9934e-01 6.62175e-04]
487	225	0.9029	True	True	[8.61892e-18 1.000e+00]
488	141	0.8333	True	True	[5.5584e-06 9.9999e-01]

D. Example Case 2- Internet of Things Network:

The second example case is based on an IoT Network [24]. The example case is based on an opensource artefact

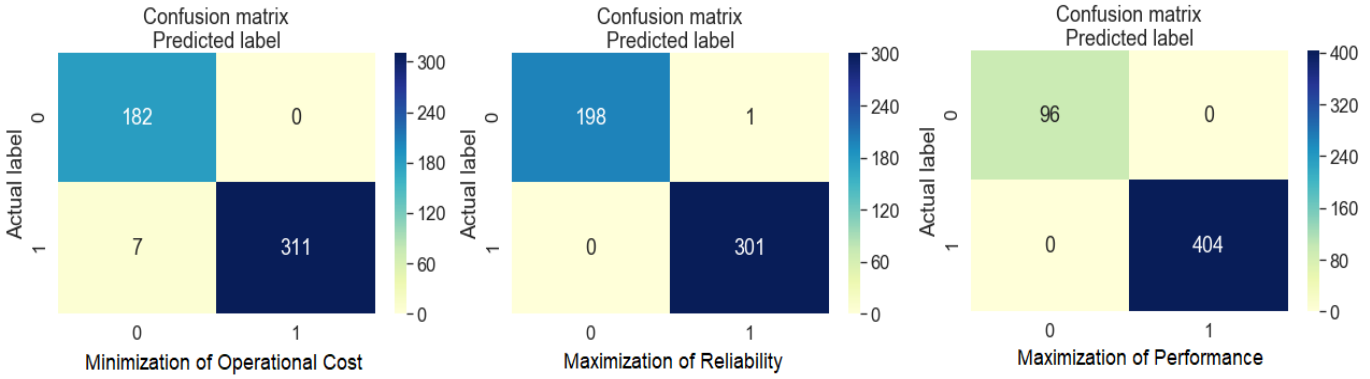

Fig. 5: Example Case 1: Confusion Matrix for Classification of state of quality properties

TABLE V: RDM Case: Example Classification Results for MP for time steps 482-488

Step	Writing Time	Mirrored Value	Actual State	Predicted State	Probability per class
482	2220	0.7165	True	True	[0.00122 0.99878]
483	2148	0.6997	True	True	[4.489e-04 9.9955e-01]
484	2475	0.6997	True	True	[0.047989 0.95201]
485	1375	0.8081	True	True	[4.86432e-09 9.9999e-01]
486	1215	0.9101	True	True	[3.7309e-10 1.000e+00]
487	2925	0.8498	False	False	[0.95808 0.041922]
488	2820	0.8231	False	False	[0.84301 0.15699]

TABLE VI: IoT Case: DataSet Format

Total Energy Consumption	Mirrored Value	Actual state
27.713032	0.882122473	False
27.166639	0.838070816	False
26.961975	0.831177457	False
14.415679	0.882122473	True
15.693936	0.838070816	True

DELTA-IoT [24] that simulates an IoT network for a smart campus. The DELTA-IoT network represents a multi-hop network comprising 14 nodes that communicate with each other and relay information to the central gateway. Based on the artefact's specification, we consider the quality properties of Minimization of Energy Consumption (MinEC) and Minimization of Packet Loss (MinPL).

DataSet: The dataset format for the experiments is presented in Table VI. The input and output features are presented as follows:

Input Features: 1) Value of the Network Parameter such as total energy consumption or total packets lost at a particular simulation time step. These values represent the RT's quality property values. 2) Mirrored Value for the state of quality property.

Output Variable : Actual state having a value of True when an quality property is satisfied and False otherwise.

For the experiments, the training dataset consists of results of the decision-support for the adaptations of the IoT network offered by the DT empowered by RL runtime model. Each run was executed for 50 simulation time steps. During each time step, local adaptation decisions were taken for each of the network nodes (sensors) individually. As a result, the training set is composed of 2800 training examples. For the purpose of tuning of the parameters of the Logistic Regression, the Stochastic Gradient Descent (SGD) Algorithm [56] is used. To tune the parameters, cross-validation has been applied by splitting the training set into 80/20 proportions. During cross-validation, the 80 percent of the training set data is used as training set and remaining 20 percent is used as cross-validation set. After tuning of the parameters using cross-validation, the results for the test set comprising the newly generated execution trace of 50 simulation time steps were collected. The data set is available at [57].

E. Experiments

Similar to Example Case 1, the experiments have been performed to assess the fidelity of *mirrored values* of the DT using RL model to mirror the state of quality properties of the IoT network (i.e. RT in this case). The parameter tuning and the classification results using Logistic Regression are described as follows:

Parameter Tuning: For the purpose of tuning of the parameters for the Logistic Regression model, the SGD algorithm is executed for 10,000 iterations with different learning rates for each quality property separately. On the basis of the accuracy scores presented in Table VII, the learning rate of 0.009 is selected for the Logistic Regression model to evaluate the test set.

Classification Results: The classification results for the quality properties using Logistic Regression are presented in Tables VIII and IX. The results show an accuracy score of 1.0 for MinEC and 0.9943 for MinPL with a precision of 1.0 for both MinEC and MinPL. The F1 scores for the classification results of MinEC and MinPL are 1.0 and 0.9963, respectively. The classification results for the quality properties are discussed as follows:

Classification Results for MinEC: For the execution trace comprising 700 test examples, collected as a result of 50 simulation time steps, the Logistic Regression technique correctly classifies the state of MinEC as shown in Fig. 6.

TABLE VII: IoT Case: Learning Rate Accuracy Score

Learning Rate	Accuracy Score MinEC	Accuracy Score MinPL
0.0001	1.0	0.83542
0.0003	1.0	0.92308
0.0005	1.0	0.94097
0.0007	1.0	0.96422
0.0009	1.0	0.98032
0.001	1.0	0.98032
0.003	1.0	0.98032
0.005	1.0	0.98389
0.007	1.0	0.99821
0.009	**1.0**	**1.0**
0.01	1.0	1.0
0.03	1.0	1.0
0.05	1.0	1.0
0.07	1.0	1.0
0.09	1.0	1.0

TABLE VIII: IoT Case: Example Classification Results for MinEC

Test Example	Energy Consumed	Mirrored Value	Actual State	Predicted State	Probability per class
634	11.31384	0.83118	True	True	[6.689e-04 9.9933e-01]
635	10.3967	0.88212	True	True	[1.703e-04 9.998e-01]
636	13.24697	0.83807	True	True	[0.0028 0.9971]
637	16.28405	0.83118	True	True	[0.0338 0.9662]
638	11.86322	0.88212	True	True	[5.469e-04 9.994e-01]

TABLE IX: IoT Case: Example Classification Results for MinPL

Test Example	Packets Lost	Mirrored Value	Actual State	Predicted State	Probability per class
440	0.08333	0.9261	True	True	[2.2177e-05 9.999e-01]
441	0.0615	0.9797	True	True	[1.6867e-06 9.9999e-01]
442	0.1428	0.9812	True	True	[0.0031 0.9968]
443	0.2	0.9261	True	False	[0.527 0.4729]
444	0.275	0.9797	False	False	[0.9985 0.00148]

Moreover, based on the results presented in Table VIII, we can deduce the extent to which the state of MinEC can be considered as satisfied (i.e. True), when both the energy consumption and *mirrored value* are considered. For example, for test example 636, the energy consumed is 13.24697 coulombs and the *mirrored value* is 0.83807. Similar to the actual state, the state predicted by the model is True with 0.9971 probability. Hence, given the input values, there is around 99 percent chance of MinEC being satisfied i.e. having state as True.

b) Classification Results for MinPL: For the execution trace comprising 700 test examples collected as a result of 50 simulation time steps, the Logistic Regression technique correctly classifies the state of MinPL with an exception of 4 test examples, as shown in Fig. 6. For these test examples, the actual state for MinPL was True but it was predicted as False.

Moreover, based on the results presented in Table IX, we can deduce the extent of the state of MinPL, when both the percentage packets lost and *mirrored value* are considered. For example, for test example 441, the packet loss is 0.0615 and the *mirrored value* is 0.9797. The output state predicted by the Logistic Regression is True, similar to the actual state, with 9.9999e-01 probability of being True. It means that given the input values, there is around 99 percent chance of MinPL being satisfied i.e. having the state as True.

Summary of Findings

In summary, the results for both cases show an overall accuracy score of aprox 99 per cent in predicting the states of the quality properties. The predictions are based on the *mirrored values* and observations for network parameters (representing *real property values*). Therefore, it satisfies the hypotheses for the experiments by supporting that there is an acceptable level of DT fidelity when mirroring the state of real quality properties (which has been estimated to be 99 percent). Hence, the hypothesis $\boldsymbol{H}_0$ is rejected and satisfies $\boldsymbol{H}_a$.

Based on the results, we can deduce that the *mirrored values* maintained by the DT equipped with RL model offer a faithful representation of the *real property values* of the RT. The state of the quality properties of the RTs (RDM and IoT networks) are considered accurately mirrored by the DT. The DT framework that we provide focuses on the dynamic properties of the RT, and the use of Logistic Regression offers a quantitative technique for measuring the fidelity of the Causally-connected Twining DTs. Based on the results, we consider that the DT has high fidelity, which has been quantified as 99% and which can inform about the quality of the behaviour with respect to the EoA.

V. Related Work

This section discusses the related work. The focus is on discussing the state-of-the-art approaches used for measuring fidelity of DTs and decision-making support offered by the DTs.

A. Measuring Fidelity of DTs

Efforts have been made to assess the fidelity of DTs [22], [58]–[60]. The existing approaches for measuring fidelity mostly focus on setting the level of fidelity with respect to certain characteristics of the RT to match the intended purpose of the DT. A number of approaches have been developed that propose the development of DTs with distinct abstraction and resolution levels and therefore, they result in a hierarchy of multi-fidelity DTs. In [22], the concept of conformance is used to assess the degree of fidelity by evaluating the similarity between the DT and the RT system. In [59], the correlation between the cyber physical systems (CPS) and their DTs is explored from different perspectives i.e. including their origin, development, engineering practices, cyber–physical mapping, and core elements. However, they don't focus on measuring the fidelity of the DT for cyber-physical systems. Moreover, the concept of multi-fidelity DTs [60] has been presented to evaluate the accuracy of DT and deal with the complexity involved in testing the different features of the CPS. In contrast, we measure the fidelity of the DT by evaluating the alignment of the trace of the state of the dynamic properties of the RT over time (e.g. the state of the quality properties in case of SAS) by using Logistic Regression.

Approaches based on techniques such as X-in-the-Loop simulation-based testing levels [61], [62] have been used to test

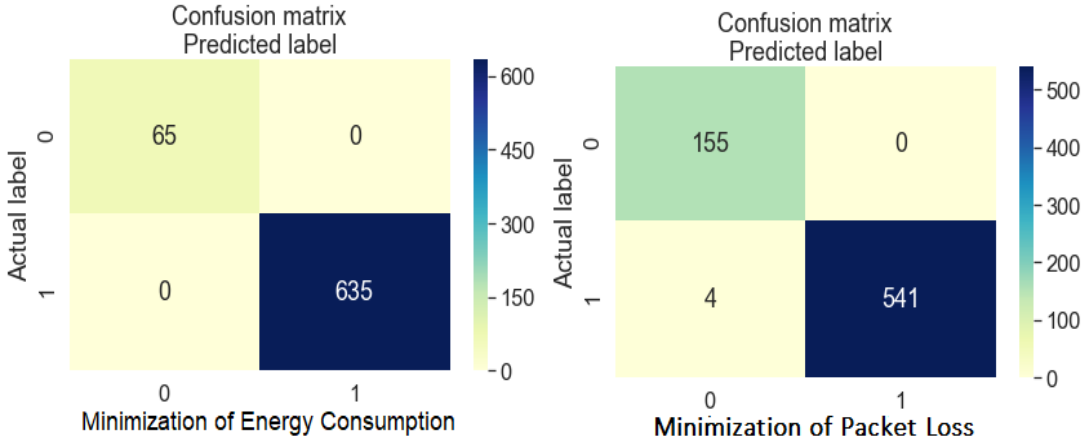

Fig. 6: Example Case 2: Confusion Matrix for Classification of state of Quality Properties

DTs. However, these simulation based approaches are quite time consuming and have high complexity levels. Furthermore, in [63], an approach based on Generative Adversarial Network (GAN) is presented to detect anamolies of the CPS with DT that is represented as a Timed Automaton Machine (TAM). In contrast, we promote the use of the machine learning technique to assess how accurately the states of the properties of the RT are mirrored by the DT equipped with the RL run-time model.

B. Support for Decision Making

The DTs have been used to provide simulations to support the decisions for smart manufacturing [59] of the systems cyber physical systems such as aerospace, automotive, and construction and health care [12], [58]. In [39], the DTs are presented as run-time predictive models to support decisions for CPS for the purpose of making CPS resilient and trustworthy. In [17], DTs are used to provide simulation-based decision support for in-house logistics planning for the overall shop-floor operations. The presented DT comprises two components: the virtual system providing simulations and the decision-support system. In [64], a DT framework to support decision-making for re-scheduling of a cyber-physical production system is presented. The DT makes use of fuzzy inference system to support decision-making using the state or conditions of different assets and the production rate of the whole system. The DT system comprises of several DTs to specify different physical assets and autonomous decision-making and they further communicate with a global DT to carry out production scheduling optimization. In comparison, our conceptual framework for DT for SAS provides decision-support using the different capabilities over the runtime model. The focus of our conceptual framework is on the dynamic properties of the RTs and offers Causally-connected Twining DTs.

VI. Conclusion and Future Work

In this paper, we have presented a conceptual framework for DTs. We have focused on its application on SASs. The conceptual framework is based on the MAPE-K loop. The framework uses run-time models to support the steps of the MAPE-K feedback loop by offering four different capabilities: *Data Monitoring*, *Descriptive*, *Predictive* and *Prescriptive* that will act upon the run-time model. The focus of our conceptual framework is on causally-connected twining DTs that deal with the dynamic properties of the DT. As a proof of concept, we have provided an example application of DT that uses the RL based run-time models to provide decision-support for adaptations for a SAS (which is the RT) to keep the SAS in the Envelope of Acceptable Behaviour with respect to its properties. We have shown how mirrored values support the quantification and measurement of the fidelity of DTs. We have also evaluated the fidelity of the example application provided by using DT for two example cases from networking domains, including RDM and IoT networks (which are open-source software artefacts available for evaluating self-adaptive techniques). We have shown how Logistic Regression can be used to evaluate how well the DT, based on RL runtime models, mirrors the quality properties of the RT. The results have shown an accuracy score of 99 per cent showing high fidelity.

As part of future work, we want to explore further the assessment of the fidelity of the DTs based on the dimension of frequency of the update related to the MAPE-K loop. In this case, we would be assessing if the DT is lagging and the effect of that, for example. We also plan to explore different ways based on machine learning and optimization for measuring the fidelity of the DTs for the specific case of SAS. Also, as the results are encouraging, we want to further study how the techniques proposed can be used to conclude and quantify the level of Fidelity of other DTs.

Acknowledgements Acknowledgements here.

References

[1] B. R. Barricelli, E. Casiraghi, J. Gliozzo, A. Petrini, and S. Valtolina, "Human digital twin for fitness management," *IEEE Access*, vol. 8, pp. 26 637–26 664, 2020.

[2] X. Liu, L. Zheng, Y. Wang, W. Yang, Z. Jiang, B. Wang, F. Tao, and Y. Li, "Human-centric collaborative assembly system for large-scale space deployable mechanism driven by digital twins and wearable ar devices," *Journal of Manufacturing Systems*, vol. 65, 11 2022.

[3] A. Ghandar, A. Ahmed, S. Zulfiqar, Z. Hua, M. Hanai, and G. Theodoropoulos, "A decision support system for urban agriculture using digital twin: A case study with aquaponics," *IEEE Access*, vol. 9, pp. 35 691–35 708, 2021.

[4] J. Wu, X. Wang, Y. Dang, and Z. Lv, "Digital twins and artificial intelligence in transportation infrastructure: Classification, application, and future research directions," *Computers and Electrical Engineering*, vol. 101, p. 107983, 2022.

[5] D. Jones, C. Snider, A. Nassehi, J. Yon, and B. Hicks, "Characterising the digital twin: A systematic literature review," *CIRP Journal of Manufacturing Science and Technology*, vol. 29, pp. 36–52, 2020.

[6] C. Zhou, J. Xu, E. Miller-Hooks, W. Zhou, C.-H. Chen, L. H. Lee, E. P. Chew, and H. Li, "Analytics with digital-twinning: A decision support system for maintaining a resilient port," *Decision Support Systems*, vol. 143, p. 113496, 2021.

[7] G. M. Marakas, *Decision support systems in the 21st century*. Prentice Hall Upper Saddle River, 2003, vol. 134.

[8] M. G. Kapteyn, J. V. R. Pretorius, and K. E. Willcox, "A Probabilistic Graphical Model Foundation for Enabling Predictive Digital Twins at Scale," *arXiv e-prints*, p. arXiv:2012.05841, Dec. 2020.

[9] I. Onaji, D. Tiwari, P. Soulatiantork, B. Song, and A. Tiwari, "Digital twin in manufacturing: conceptual framework and case studies," *International journal of computer integrated manufacturing*, vol. 35, no. 8, pp. 831–858, 2022.

[10] L. Barth, M. Ehrat, R. Fuchs, and J. Haarmann, "Systematization of digital twins: Ontology and conceptual framework," in *Proceedings of the 3rd International Conference on Information Science and Systems*, 2020, pp. 13–23.

[11] M. Grieves and J. Vickers, "Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems," *Transdisciplinary perspectives on complex systems: New findings and approaches*, pp. 85–113, 2017.

[12] R. Eramo, F. Bordeleau, B. Combemale, M. van Den Brand, M. Wimmer, and A. Wortmann, "Conceptualizing digital twins," *IEEE Software*, vol. 39, no. 2, pp. 39–46, 2021.

[13] A. D. Benedictis, F. Flammini, N. Mazzocca, A. Somma, and F. Vitale, "Digital twins for anomaly detection in the industrial internet of things: Conceptual architecture and proof-of-concept," *IEEE Transactions on Industrial Informatics*, pp. 1–11, 2023.

[14] L. F. Rivera, M. Jiménez, N. M. Villegas, G. Tamura, and H. A. Müller, "Toward autonomic, software-intensive digital twin systems," *IEEE Software*, vol. 39, no. 2, pp. 20–26, 2022.

[15] A. El Saddik, H. Badawi, R. A. M. Velazquez, F. Laamarti, R. G. Diaz, N. Bagaria, and J. S. Arteaga-Falconi, "Dtwins: a digital twins ecosystem for health and well-being," *IEEE COMSOC MMTC Commun. Front*, vol. 14, no. 2, pp. 39–43, 2019.

[16] W. Kuehn, "Digital twins for decision making in complex production and logistic enterprises," *International Journal of Design & Nature and Ecodynamics*, vol. 13, no. 3, pp. 260–271, 2018.

[17] F. Coelho, S. Relvas, and A. Barbosa-Póvoa, "Simulation-based decision support tool for in-house logistics: the basis for a digital twin," *Computers & Industrial Engineering*, vol. 153, p. 107094, 2021.

[18] V. Kulkarni, S. Barat, T. Clark, and B. S. Barn, "Digital twin as an aid for decision-making in the face of uncertainty," in *2022 Winter Simulation Conference (WSC)*, 2022, pp. 1371–1385.

[19] R. De Lemos, H. Giese, H. A. Müller, M. Shaw, J. Andersson, M. Litoiu, B. Schmerl, G. Tamura, N. M. Villegas, T. Vogel *et al.*, "Software engineering for self-adaptive systems: A second research roadmap," in *Software Engineering for Self-Adaptive Systems II: International Seminar, Dagstuhl Castle, Germany, October 24-29, 2010 Revised Selected and Invited Papers*. Springer, 2013, pp. 1–32.

[20] M. Rinard, C. Cadar, and H. H. Nguyen, "Exploring the acceptability envelope," in *Companion to the 20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications*, ser. OOPSLA '05. New York, NY, USA: Association for Computing Machinery, 2005, p. 21–30. [Online]. Available: https://doi.org/10.1145/1094855.1094866

[21] D. C. Gross *et al.*, "Report from the fidelity implementation study group," in *Fall Simulation Interoperability Workshop Papers*, 1999.

[22] P. Muñoz, "Measuring the fidelity of digital twin systems," in *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, 2022, pp. 182–188.

[23] H. Samin, L. H. G. Paucar, N. Bencomo, C. M. C. Hurtado, and E. M. Fredericks, "RDMSim: An Exemplar for Evaluation and Comparison of Decision-Making Techniques for Self-Adaptation," in *16th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), Artefact Track*, 2021.

[24] M. U. Iftikhar, G. S. Ramachandran, P. Bollansee, D. Weyns, and D. Hughes, "DeltaIoT: A Self-Adaptive Internet of Things Exemplar," in *2017 IEEE/ACM 12th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. Buenos Aires, Argentina: IEEE, May 2017, pp. 76–82.

[25] M. J. Kochenderfer, *Decision Making Under Uncertainty: Theory and Application*. Massachusetts London, England: The MIT Press Cambridge, 2015.

[26] M. Ale Ebrahim Dehkordi, J. Lechner, A. Ghorbani, I. Nikolic, E. Chappin, and P. Herder, "Using machine learning for agent specifications in agent-based models and simulations: A critical review and guidelines," *Journal of Artificial Societies and Social Simulation*, vol. 26, no. 1, p. 9, 2023.

[27] M. Gavalec, J. Ramík, and K. Zimmermann, *Decision Making and Optimization*. Springer Cham, 2015.

[28] O. Hussain, T. Dillon, E. Chang, and F. Hussain, "A fuzzy inference model for risk based informed decision-making in e-business," in *2009 International Conference on Network-Based Information Systems*, 2009, pp. 519–524.

[29] J. Cleland-Huang, A. Agrawal, M. Vierhauser, M. Murphy, and M. Prieto, "Extending mape-k to support human-machine teaming," in *Proceedings of the 17th Symposium on Software Engineering for Adaptive and Self-Managing Systems*, ser. SEAMS '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 120–131.

[30] R. de Lemos, "Human in the loop: What is the point of no return?" in *Proceedings of the IEEE/ACM 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, ser. SEAMS '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 165–166. [Online]. Available: https://doi.org/10.1145/3387939.3391597

[31] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[32] E. Rutten, N. Marchand, and D. Simon, "Feedback control as mape-k loop in autonomic computing," in *Software Engineering for Self-Adaptive Systems III. Assurances: International Seminar, Dagstuhl Castle, Germany, December 15-19, 2013, Revised Selected and Invited Papers*. Springer, 2017, pp. 349–373.

[33] P. Arcaini, E. Riccobene, and P. Scandurra, "Modeling and analyzing mape-k feedback loops for self-adaptation," in *2015 IEEE/ACM 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, 2015, pp. 13–23.

[34] B. H. Cheng, H. Giese, P. Inverardi, J. Magee, R. de Lemos, J. Andersson, B. Becker, N. Bencomo, Y. Brun, B. Cukic *et al.*, "08031–software engineering for self-adaptive systems: A research road map," in *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2008.

[35] S. M. Hezavehi, D. Weyns, P. Avgeriou, R. Calinescu, R. Mirandola, and D. Perez-Palacin, "Uncertainty in self-adaptive systems: A research community perspective," 2021.

[36] N. Esfahani and S. Malek, "Uncertainty in self-adaptive software systems," vol. 7475, 01 2013.

[37] J. Gray and B. Rumpe, "Modeling of, for, and with digital twins," *Software and Systems Modeling*, vol. 21, no. 5, pp. 1685–1686, 2022.

[38] B. Combemale, J. Kienzle, G. Mussbacher, H. Ali, D. Amyot, M. Bagherzadeh, E. Batot, N. Bencomo, B. Benni, J.-M. Bruel *et al.*, "A hitchhiker's guide to model-driven engineering for data-centric systems," *IEEE Software*, vol. 38, no. 4, pp. 71–84, 2020.

[39] F. Flammini, "Digital twins as run-time predictive models for the resilience of cyber-physical systems: a conceptual framework," *Philosophical Transactions of the Royal Society A*, vol. 379, no. 2207, p. 20200369, 2021.

[40] G. Blair, N. Bencomo, and R. B. France, "Models@ run. time," *Computer*, vol. 42, no. 10, pp. 22–27, 2009.

[41] N. Bencomo, S. Götz, and H. Song, "Models@run.time: a guided tour of the state of the art and research challenges," *Softw. Syst.*

Model., vol. 18, no. 5, pp. 3049–3082, 2019. [Online]. Available: https://doi.org/10.1007/s10270-018-00712-x

[42] N. Bencomo and L. H. Garcia Paucar, "Ram: Causally-connected and requirements-aware runtime models using bayesian learning," in *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems (MODELS)*, 2019, pp. 216–226.

[43] B. Morin, O. Barais, G. Nain, and J. Jézéquel, "Taming dynamically adaptive systems using models and aspects," in *31st International Conference on Software Engineering, ICSE 2009, May 16-24, 2009, Vancouver, Canada, Proceedings*. IEEE, 2009, pp. 122–132. [Online]. Available: https://doi.org/10.1109/ICSE.2009.5070514

[44] J. Corral-Acero, F. Margara, M. Marciniak, C. Rodero, F. Loncaric, Y. Feng, A. Gilbert, J. F. Fernandes, H. A. Bukhari, A. Wajdan *et al.*, "The 'digital twin'to enable the vision of precision cardiology," *European heart journal*, vol. 41, no. 48, pp. 4556–4564, 2020.

[45] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[46] F. Feit, A. Metzger, and K. Pohl, "Explaining online reinforcement learning decisions of self-adaptive systems," in *2022 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS)*. IEEE, 2022, pp. 51–60.

[47] H. Samin, L. Garcia Paucar, B. Nelly, and P. Sawyer, "Towards priority-awareness in autonomous intelligent systems," in *36th ACM/SIGAPP Symposium On Applied Computing (SAC)*. ACM, 2021. [Online]. Available: https://www.overleaf.com/read/cqqcxjhkbzdv

[48] L. H. G. Paucar and N. Bencomo, "RE-STORM: mapping the decision-making problem and non-functional requirements trade-off to partially observable markov decision processes," in *Proceedings of the 13th International Conference on Software Engineering for Adaptive and Self-Managing Systems - SEAMS '18*. Gothenburg, Sweden: ACM Press, 2018, pp. 19–25.

[49] D. M. Roijers, S. Whiteson, and F. A. Oliehoek, "Point-based planning for multi-objective pomdps," in *Proceedings of the twenty-fourth international joint conference on artificial intelligence (IJCAI)*, vol. 2015. AAAI Press, 2015, pp. 1666–1672.

[50] K. M. Bowers, E. M. Fredericks, R. H. Hariri, and B. H. Cheng, "Providentia: Using search-based heuristics to optimize satisficement and competing concerns between functional and non-functional objectives in self-adaptive systems," *Journal of Systems and Software*, vol. 162, p. 110497, 2020.

[51] K. M. Bowers, E. M. Fredericks, and B. H. Cheng, "Automated optimization of weighted non-functional objectives in self-adaptive systems," in *Search-Based Software Engineering: 10th International Symposium, SSBSE 2018, Montpellier, France, September 8-9, 2018, Proceedings 10*. Springer, 2018, pp. 182–197.

[52] B. Schleich, N. Anwer, L. Mathieu, and S. Wartzack, "Shaping the digital twin for design and production engineering," *CIRP annals*, vol. 66, no. 1, pp. 141–144, 2017.

[53] M. Rinard, C. Cadar, and H. H. Nguyen, "Exploring the acceptability envelope," in *Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, 2005, pp. 21–30.

[54] S. Menard, *Applied logistic regression analysis*. Sage, 2002, vol. 106.

[55] H. Samin, N. Bencomo, and P. Sawyer, "Decision-making under uncertainty: be aware of your priorities," *Software and Systems Modeling*, vol. 21, no. 6, pp. 2213–2242, 2022.

[56] E. Bisong, "Logistic regression," in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*. Springer, 2019, pp. 243–250.

[57] xyz, "Logistic regression dataset and experiments," Tech. Rep. v0.9, 2023. [Online]. Available: https://github.com/fidelitydt/LogisticRegression.git

[58] F. Tao, J. Cheng, Q. Qi, M. Zhang, H. Zhang, and F. Sui, "Digital twin-driven product design, manufacturing and service with big data," *The International Journal of Advanced Manufacturing Technology*, vol. 94, pp. 3563–3576, 2018.

[59] F. Tao, Q. Qi, L. Wang, and A. Nee, "Digital twins and cyber–physical systems toward smart manufacturing and industry 4.0: Correlation and comparison," *Engineering*, vol. 5, no. 4, pp. 653–661, 2019.

[60] A. Arrieta, "Multi-fidelity digital twins: a means for better cyber-physical systems testing?" *arXiv preprint arXiv:2101.05697*, 2021.

[61] M. Borg, R. B. Abdessalem, S. Nejati, F.-X. Jegeden, and D. Shin, "Digital twins are not monozygotic–cross-replicating adas testing in two industry-grade automotive simulators," in *2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST)*. IEEE, 2021, pp. 383–393.

[62] C. Menghi, S. Nejati, L. Briand, and Y. I. Parache, "Approximation-refinement testing of compute-intensive cyber-physical models: An approach based on system identification," in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, 2020, pp. 372–384.

[63] Q. Xu, S. Ali, and T. Yue, "Digital twin-based anomaly detection in cyber-physical systems," in *2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST)*. IEEE, 2021, pp. 205–216.

[64] A. Villalonga, E. Negri, G. Biscardo, F. Castano, R. E. Haber, L. Fumagalli, and M. Macchi, "A decision-making framework for dynamic scheduling of cyber-physical production systems based on digital twins," *Annual Reviews in Control*, vol. 51, pp. 357–373, 2021.