 doi:10.20944/preprints202104.0648.v1

Concepts and Models of Environment in Self-Adaptive Systems: A Systematic Literature Review

Yong-Jun Shin*, Joon-Young Bae†, Doo-Hwan Bae‡
*‡School of Computing, †Faculty of Engineering
*‡Korea Advanced Institute of Science and Technology (KAIST), †The University of Hong Kong (HKU)
*‡Daejeon, Republic of Korea, †Pokfulam, Hong Kong
*yjshin@se.kaist.ac.kr, †n99joon@hku.hk, ‡bae@se.kaist.ac.kr

***Abstract*—The runtime environment is an important concern for self-adaptive systems (SASs). Although researchers have proposed many approaches for developing SASs that address the issue of uncertain runtime environments, the understanding of these environments varies depending on the objectives, perspectives, and assumptions of the research. Thus, the current understanding of the environment in SAS development is ambiguous and abstract. To make this understanding more concrete, we describe the landscape in this area through a systematic literature review (SLR). We examined 128 primary studies and 14 unique environment models. We investigated concepts of the environment depicted in the primary studies and the proposed environment models based on their ability to aid in understanding. This illustrates the characteristics of the SAS environment, the associated emerging environmental uncertainties, and what is expressed in the existing environment models. This paper makes explicit the implicit understanding about the environment made by the SAS research community and organizes and visualizes them.**

***Index Terms*—self-adaptive systems, environment, concept, model, systematic literature review**

I. Introduction

A self-adaptive system (SAS) adaptively changes its behavior or structure at runtime to achieve its goals and respond to unanticipated situations of the system itself or its operating environment. We refer to these unanticipated situations as uncertainty. This uncertainty can come from imperfect requirements, defective SAS design or implementation, or an unknown runtime environment [P94]. Among the various reasons for uncertainty, the environment is one of the most interesting and challenging entities to address in SAS development. It is difficult to fully anticipate at design time the environment that an SAS will encounter during its operation, and the environments of modern systems are complex and open.

To develop a system that is adaptive to an uncertain environment, various engineering approaches, such as eliciting adaptive requirements from the environment [P25], analyzing SAS design while considering an uncertain environment [P63], testing an SAS implementation with environmental inputs [P17], and updating environmental knowledge for optimal runtime decision-making of an SAS [P66], have been proposed. In this context of active research of an SAS in an uncertain environment, one shortcoming we have noticed is that the meanings of "uncertain environment" and "environmental uncertainty" are inconsistent across different studies. For example, in different papers, an uncertain environment has been described as an environment that changes itself over time, an environment that has been changed by an SAS, an environment that has been misrecognized by sensor noise, and so on.

Although there could be many reasons for this inconsistent understanding of the environment and its uncertainty, what we focus on is the lack of a concrete understanding of the environment of an SAS. In the software engineering community for SASs, we have achieved implicit agreement on the concepts of the environment, its uncertainty, and its effect on SASs, but this implicit agreement has led to ad-hoc interpretations. We believe that the various interpretations of the environment of an SAS are all meaningful in establishing a concrete understanding of it, so we have conducted a systematic literature review (SLR) to gather and analyze them. To detail the landscape in this understanding of the environment, we examined the concepts (how existing studies defined and described it) and models (how existing studies abstracted it) of the environment of SASs. In summary, this SLR tries to answer the following questions:

- How do existing studies describe the environment?
- What are the characteristics of the environment?
- How is the environment abstracted as a model?

The remainder of this paper is organized as follows. Section II introduces the basic concepts of an SAS and the environment. Section III presents our systematic review protocol. Section IV shows the review results for each research question. Section V provides one of the lessons that we have learned through this SLR, and Section VI reveals threats and the validity of our work. Section VIII concludes the paper.

II. Background: SAS and Environment

Some papers that introduce SAS engineering provide a fundamental understanding of SASs and the environment [1]–[3]. Fig. 1 illustrates a conceptual model of an SAS [1]. It

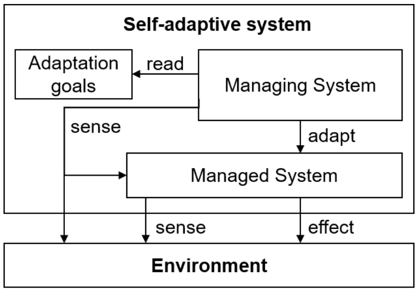

Fig. 1. A conceptual model of SAS [1]

shows the relationships between the SAS and the environment. The environment is an external world where the SAS operates comprising observable physical and virtual entities. Because the environment is regarded as uncertain, the SAS continuously senses it to reliably achieve its adaptation goals. The sensed environmental condition affects the decisions of the SAS, and these decisions can have new effects on the environment. This high-level understanding has been agreed upon in the SAS research community, but the abstract concept of the environment has not been specifically organized. In this paper, we attempt to organize specific concepts and models of the environment empirically.

III. Review Protocol

To conduct an SLR, we designed a review protocol including the review steps and specific inputs and outputs for each step shown in Fig. 2. Designing a review protocol in advance prevents a biased or subjective survey; disclosing it ensures a reproducible review. Based on the goal of this SLR, we specified the research questions, automated search engines, manual search venues, and the search string. The papers searched were evaluated to determine whether they were primary studies[1] under the predefined criteria. The selected primary studies were examined thoroughly. The primary studies also became the sources of cross-reference searching. It was a step in which all the references of the primary studies were exhaustively explored to minimize the possibility of missing important papers. Any newly discovered papers were evaluated by the selection criteria. In particular, we utilized the "snowballing" method[2]. When searching finished, we extracted predefined data items from the primary studies. The extracted data were analyzed, and the analysis results are reported in Section IV. The rest of this section describes the elements of this protocol.

The goal of this SLR is to show the landscape of concepts and models of the environment of SAS in software engineering. To achieve this goal, we specified questions that will be answered, as shown in Table I. Regarding RQ1, to define the environment of SAS, we surveyed how primary studies have explicitly defined the environment. For RQ2, because the

[1]In this case, a primary study refers to a paper subject to review, and the SLR itself is a secondary study [4].

[2]The snowballing cross-reference checking method exhaustively explores all of the backward (cited by the subject paper) and forward (citing the subject paper) references until no additional papers are discovered [5].

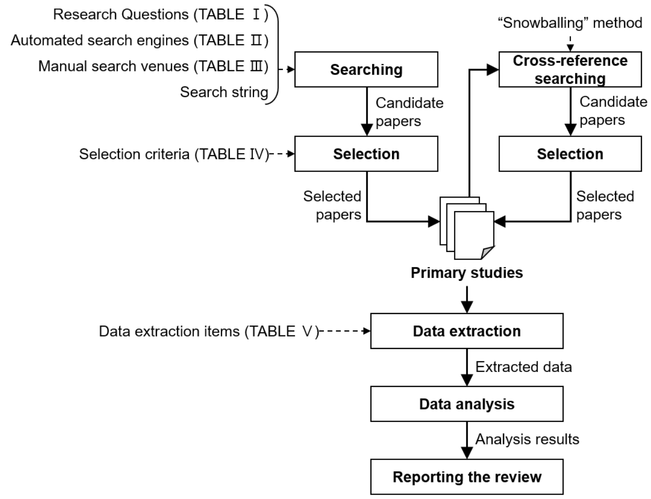

Fig. 2. Overview of the review protocol

TABLE I
Research Questions

Category	ID	RQ
Concepts of the environment of SAS	RQ1	*Definitions of the environment*. How did primary studies explicitly define the "environment" of an SAS?
	RQ2	*Characteristics of the environment*. What characteristics of the environment of an SAS did primary studies mention in describing it?
	RQ3	*Sources of the environmental uncertainty*. What was considered by primary studies to be the source of environmental uncertainty?
Models of the environment of SAS	RQ4	*Modeling of the environment*. Who models, how do they model, when do they model, and why do they model the environment of SAS?
	RQ5	*Application of the environment models*. When and how are the environment models used?
	RQ6	*Expressiveness of the environment models*. How are the characteristics of the environment expressed in the models?

majority of primary studies may not explicitly define the environment and only describe its characteristics, we clarified what characteristics were used to describe the environment. Besides, RQ3 was included because environmental uncertainty is a huge area of interest in software engineering for SAS, but it is an ambiguous term. Therefore, as an approach to reduce this ambiguity, we examined specific causes of the phenomenon called environmental uncertainty in the primary studies. For RQ4, we also selected papers from the primary studies that propose environment models and surveyed these modeling methods. RQ5 looked at the application of the environment models. Finally, in RQ6, we examined the expressiveness of the environment models, especially how the characteristics of the environment were represented in each model.

To collect appropriate primary studies to answer the RQs, we utilized various automated search engines that could help in finding related papers. The selected search engines are listed in Table II. Widely used computer science article search engines were selected, but various multi-disciplinary search engines were also used to exhaustively search for as many related

TABLE II
AUTOMATED SEARCH ENGINES

Disciplinary	Search engine
Computer science and related subjects	IEEE Xplore (http://ieeexplore.ieee.org/)
	ACM Digital Library (http://dl.acm.org/)
	dblp computer science bibliography (https://dblp.org/)
Multi-disciplinary	Web of Science (http://www.webofknowledge.com/)
	SpringerLink (http://link.springer.com/)
	Scopus (http://www.scopus.com/)
	Wiley Online Library (http://onlinelibrary.wiley.com/)
	World Scientific (https://www.worldscientific.com/)
	ScienceDirect (http://www.sciencedirect.com/)

TABLE III
MANUAL SEARCH VENUES

Type	Venue
Journal	ACM Transactions on Software Engineering and Methodology
	ACM Transactions on Autonomous and Adaptive Systems
	IEEE Transactions on Software Engineering
	Journal of Systems and Software
	Information and Software Technology
Conference	Intl. Conference on Software Engineering (ICSE)
	Intl. Symposium on the Foundations of Software Engineering (FSE)
	Intl. Conference on Automated Software Engineering (ASE)
	Intl. Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)
	Intl. Conference on Self-Adaptive and Self-Organizing Systems (SASO)

works as possible. In addition, we conducted a manual search for publications in related journals and conferences, listed in Table III, for added focus in case of unknown errors in the automated search. High-end software engineering and SAS-related venues were selected.

The following search string was used to find related papers:

{(**self-** OR **adapt**) AND (**software** OR **system**) AND (**environment**) AND (**uncertain**)}

Papers handling "software" or "system" with "self-" prefixed properties or "adapt" (as in "adaptive", "adaptiveness", etc.) were searched. The "self-" prefix identifies the most general terms of various adaptive properties [6]. In addition, we searched for studies explicitly referencing the uncertain environment or environmental uncertainties of SAS, which were both caught by our specification of forms of "environment" and "uncertain". This approach was more liberal than searching on the exact phrases, but it provided more papers for manual review, which removed any extraneous ones. One of the candidate keywords was "model", but it was intentionally omitted in order to obtain a larger number of papers for identifying the concepts of the environment. For RQs 4–6, we manually selected papers that proposed environment models after reading all the primary studies. We used the previously identified search string for both the automated and manual search; the search scope included titles, abstracts, and author keywords of the papers.

The searched papers were evaluated using the predefined selection criteria in Table IV. There were both inclusion and exclusion criteria. If a paper satisfied all the inclusion criteria and none of the exclusion criteria, it was selected as a primary study. The inclusion criteria IC4 evaluated whether a paper was appropriate to answer our RQs. Our purpose was to gain a general understanding of the environment of an SAS from papers motivated by the environment and its uncertainty. All the authors of this work read abstracts of the papers (and the introduction if needed) and together judged if the papers were mainly motivated by some characteristics of the uncertain environment of the SAS. Other criteria helped control the discipline focus, quality, and form of the primary studies.

TABLE IV
INCLUSION AND EXCLUSION CRITERIA

Inclusion criteria	
IC1	Papers written in English
IC2	Research papers peer-reviewed and published in conferences, journals, or books
IC3	Papers in the field of computer science
IC4	Papers on the topic of a domain-general software engineering approach for self-adaptive systems motivated from uncertain environment
Exclusion criteria	
EC1	Duplicated papers
EC2	Papers whose contents are not fully accessible
EC3	Papers not in the form of full research papers (i.e., abstracts, tutorials, or reports)
EC4	Collections of studies (i.e., books or proceedings)
EC5	Papers summarizing existing studies or concepts (i.e., overviews, introductions, keynotes, roadmaps, or surveys)

TABLE V
DATA EXTRACTION ITEMS

RQ	Data items
RQ1	Explicit definition of the "environment" of an SAS
RQ2	Expressions explicitly mentioned to describe characteristics of the environment
RQ3	Sources of environmental uncertainty addressed in the primary studies
RQ4	Environment modeling details (modeling time, agent, effort, purpose, formalism, process, etc.)
RQ5	Characteristics of the environment expressed in the models
RQ6	Environment model application details (application time, usage, supportive techniques, etc.)

For each RQ, extracted data items are identified in Table V. Data extraction was conducted manually, and the collected data were analyzed to answer the RQs.

Following our review protocol, we found 128 primary studies. Among the 128 primary studies, we manually found 14 unique models of the environment. We extracted data and analyzed it to answer the six RQs. During all the review steps, to create a reproducible and objective survey, we recorded all the outputs for each step and made all the review data accessible[3]. It includes lists of the searched papers, selection sheets, extracted raw data, and so on. All the data are open to readers, but due to a lack of space, this paper only reports the processed analysis results for each RQ in Section IV.

[3] Access the SLR website for all the review data: https://sites.google.com/se.kaist.ac.kr/sas-environment-slr/

TABLE VI
DEFINITIONS OF ENVIRONMENT

Explicit definition of "environment" of SAS	Ref.
"anything observable by the software system, such as end user input, external hardware devices and sensors, or program instrumentation"	P6
"the physical world or computing elements that are not under control of the system"	P24
"circumstances that interact with or affect the system"	P77

IV. REVIEW RESULTS

A. Concept of the Environment of SAS

RQ1) Definitions of the environment of SAS: To understand the environment of an SAS, we first collected explicit definitions[4]. Three explicit definitions were found and are listed in Table VI. [P6] defines it as external and observable objects. [P24] highlighted the fact that it is not under the direct control of an SAS. In contrast, [P77] defined it as circumstances interacting with the SAS. In paraphrasing the existing definitions, we can say that *the environment of an SAS contains external and observable objects that are not under the control of the SAS but rather interact with it*.

The definitions are acceptable and indicate some key characteristics of the environment, such as *diverse factors*, *externality*, *observability*, and *interaction*. However, we note that few of the selected studies explicitly defined environment, and they varied considerably in terms of the authors' perspectives. This made it difficult to get considerable knowledge about the concept of environment only from the existing definitions. This confirmed the assumptions that drove our motivation for conducting the SLR. Fortunately, the studies without explicit definitions implicitly shared a common understanding about the environment. Therefore, we attempted to gather this understanding in answering RQ2 based on these definitions.

RQ2) Characteristics of the environment of SAS: To establish the concept of the environment, we collected characteristics of the environment of an SAS. Although we had limited definitions, almost all of the primary studies informally described the environment of an SAS of their interests. We searched for all the sentences that included "environment" in the primary studies and collected and categorized the various expressions from the sentences that described the environment in Table VII. The expressions in the primary studies are listed in the second column, and we organized the expressions into five characteristics of the environment of an SAS in the first column. We referred to the features mentioned in the definitions collected in answering RQ1 to make the classifications. Descriptions for each characteristic and the related expressions are also given in the table.

We organized expressions of the environment that were found into five characteristics in Table VII. *Diversity* is represented in how the environment comprises diverse environmental factors. Specification of the environment requires the specification of each environmental factor of interest. *Externality* demonstrates that the environment is outside the SAS boundary. Therefore, the environment is not under the direct control of the SAS. *Observability* is one aspect of the relationship between the environment and the SAS. Although the environment is not within the SAS boundary, it is observable by monitoring elements of the SAS, so the SAS can respond to the environment. *Interactivity* also defines the relationship between the two. The SAS interacts with the environment. The interaction affects both the SAS and the environment. *Uncertainty* is the last characteristic. Because the environment is an external element, it is uncertain. Sometimes the environment itself is dynamic, and an SAS developer may have limited or incorrect information.

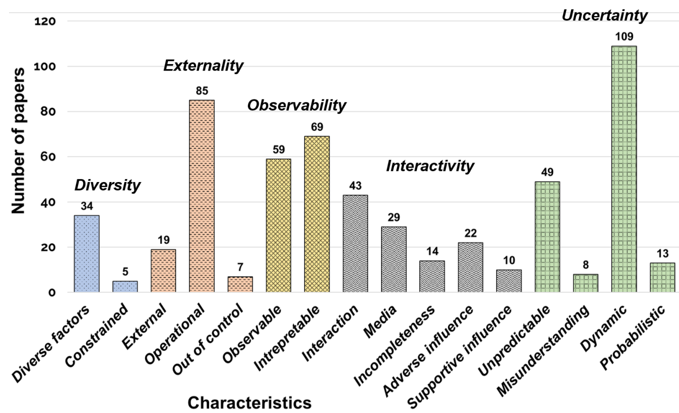

Fig. 3. The number of mentions of characteristics of SAS environment

Fig. 3 shows how many papers mentioned each characteristic of the environment. This indicates what environmental characteristics were relatively familiar to the researchers as expressed in their writing. For example, "dynamic operating environment" was one of the most widely used expressions to describe the environment. The figure shows trends in characteristics mentioned. More important than the trends, however, is that in addressing RQ2, the various characteristics and expressions have been organized to help understand the environment more comprehensively. Although there were not many clear definitions, the SAS research community has established a significant and implicit agreement on the characteristics of the environment of an SAS. Lastly, we have been able to make these agreed upon characteristics explicit and visual.

RQ3) Sources of the environmental uncertainty of an SAS: Among the characteristics of an environment, uncertainty is one of the core reasons that a system should be adaptive. However, the use of the term "uncertainty" is typically conceptual and ambiguous, so it can cause inconsistent understanding among various engineers. To tackle ambiguous

[4]An "explicit definition," is a clear definition of the environment present in a paper and indicated by a sentence, such as "environment is defined as ..." or "environment means ..." Although not explicit definitions, expressions used to describe the environment were collected in answering RQ2.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 April 2021 doi:10.20944/preprints202104.0648.v1

TABLE VII
CHARACTERISTICS OF ENVIRONMENT OF SAS AND THE EXPRESSIONS

Organized characteristics	Explicit expressions	Description
Diversity	(Diverse factors) computing/physical (environment element), user (human), system, service, time, factor	The environment consists of diverse (environmental) factors/elements, for example, computing or physical elements, users (humans), or external systems (services).
	(Constrained) constraints	An environmental element has its own constraints.
Externality	(External) external, surrounding	The environment is outside of the SAS boundary.
	(Operational) operation, execution, deployment, runtime	The environment is where the system is deployed, operates, and executes at runtime.
	(Out of control) no control, indirect	An SAS cannot (directly) control its environment.
Observability	(Observable) observable, sense, monitor, measure	The environment is observable by an SAS.
	(Interpretable) parameter, attribute, variable, value, data, input, condition, event, phenomena	An SAS perceives and interprets its environment based on data or values of environmental variables or parameters. The perception is an SAS's environmental input condition or event.
Interactivity	(Interaction) interaction, influence, affect, impact, interface, trigger	The environment interacts with an SAS, so it affects and is affected by the SAS.
	(Media) sensor, effector/actuator	An SAS interacts with the environment through its sensors and effectors (actuators).
	(Incompleteness) error, noise, variation in sensing, signal interference, failure	Interactions between the environment and an SAS can be incomplete or may have failed.
	(Adverse influence) disturbing, unsafe, adverse, disruptive, unfavorable, threat	The environment may adversely affect SAS goal satisfaction.
	(Supportive influence) resource	The environment may be supportively used for SAS goal satisfaction.
Uncertainty	(Unpredictable) uncertain, unforeseen, unexpected, unpredictable	The environment is not fully anticipated at the design time of an SAS.
	(Misunderstanding) unknown, lack of knowledge, missing	Knowledge of the environment may be incomplete. An SAS can encounter an unknown environment. Missing environmental parameters might also be a problem.
	(Dynamic) change, fluctuation, dynamic	The environment dynamically changes its states or behavior.
	(Probabilistic) non-deterministic, probabilistic, stochastic	The environmental knowledge is non-deterministic.

understanding[5], we examined concrete sources that cause environmental uncertainty. In the selected primary studies, we found three papers [P22, P94, P102] whose contributions comprise proposed taxonomies of environmental uncertainty sources. We summarized these taxonomies of sources[3] and reorganized the sources, as presented in Table VIII. The existing source categorizations had overlapping meanings, so we reorganized the sources into two main categories. The first source of environmental uncertainty is limited environmental knowledge. An SAS developer may have limited knowledge about the environment because the environment changes or the environment was not fully identified. A second orthogonal source of environmental uncertainty is incomplete interaction with the environment. Even if the environment is well specified, environmental uncertainty arises if the interaction with the environment is not as expected. These two main sources are further divided into subcategories, and the subcategories follow the classification of the primary studies. Their descriptions and terms are listed in Table VIII.

Three papers proposed taxonomies of sources of environmental uncertainty; the other primary studies proposed specific approaches to handle problems caused by some environmental uncertainty sources. Fig. 4 shows which sources of environmental uncertainty were addressed by these primary studies. In these sources, environmental uncertainty caused

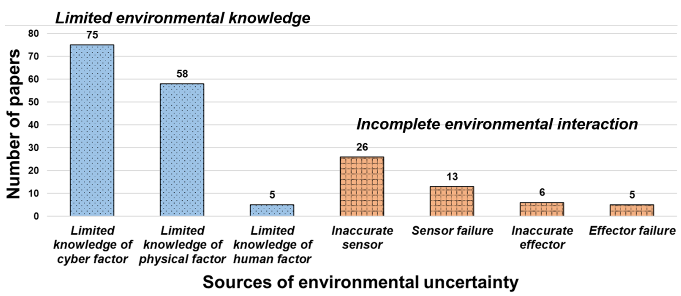

Fig. 4. Frequency of addressing each source of environmental uncertainty

by limited environmental knowledge was addressed more than uncertainty from incomplete interaction. However, limited knowledge about the human environmental factor was rarely addressed compared to the others. With regard to sources of incomplete interaction, the sources related to sensors were relatively familiar to researchers, as evidenced in the writing, more so than effectors. Here, in noting the trends, we must also acknowledge that even if various studies are addressing environmental uncertainty, their use of this term does not necessarily rely on a shared definition. Therefore, researchers need to specifically explain their concerns regarding environmental uncertainty to prevent misinterpretation.

B. Models of the Environment of an SAS

RQ4) Modeling of the environment of an SAS: A model is an abstraction of a subject that represents its important characteristics. We collected 14 unique models that represent the environment of an SAS from the 128 primary studies, and

[5]We also surveyed definitions of "uncertainty" and "environmental uncertainty", but they are not included in this paper due to lack of space. Refer to our website[3].

TABLE VIII
SOURCES OF THE ENVIRONMENTAL UNCERTAINTY

(Reorganized) Source	Sub-category	Existing description (P22, P94, P102)	Existing terms (P22, P94, P102)
1. Uncertainty from limited environmental knowledge	Overall (cyber factor)	"current operating conditions, which are continuously invalidated due to changes" [P22] "the context of execution changes" [P22] "the environment conditions are close to a change" [P102] "Events and conditions in the environment that cannot be anticipated" [P94]	"Uncertainty in the context" [P22,P102], "Uncertainty of parameters in future operation" [P22,P102], "Unpredictable Environment" [P94]
	- Physical factor	"the effect of physical world on the software is a subset of context, which was described in the previous source (uncertainty in the context)" [P22]	"Uncertainty in cyber-physical systems" [P22,P102]
	- Human factor	"the behavior of the crew (human) may be very unpredictable" [P22]	"Uncertainty due to human in the loop" [P22,P102]
2. Uncertainty from incomplete environmental interaction	Inaccurate sensor	"A sensor ... may return a slightly different number every time ..., even if the actual value ... is fixed." [P22] "Random and persistent disturbances that reduce the clarity of a signal" [P94]	"Uncertainty due to noise" [P22,P102], "Sensor noise" [P94]
	Sensor failure	"When a sensor cannot measure or report the value of a property" [P94]	"Sensor failure" [P94]
	Inaccurate effector	"system's ability ... is not only a function of the accuracy of its software, but the precision in the physical steering components (actuator)" [P22] "An adaptation that alters the execution environment in unanticipated ways" [P94]	"Uncertainty in cyber-physical systems" [P22,P102], "effector" [P94]
	Effector failure	"an actuator ... can either fail during an adaptation or ... introduce adverse effects upon the execution environment" [P94]	"effector" [P94]

they are listed in Table IX. If a paper named the model, that is presented in the table; otherwise, a descriptive name was created for the model for the purpose of this review. All the models provide an abstraction of the environment of an SAS, but their representations vary depending on the purpose of the modeling and the authors' perspectives. In addition, based on the authors' purpose, the formalism of the model was decided. Some models followed standardized formalisms, but others were created using the authors' modeling languages or rules. These are summarized in the table. Due to a lack of space, each model is not explained in detail (the reader is directed to the original reference for this information), but the insights obtained from the analysis of the models (modeling process and modeling effort) are shown.

We summarized the modeling processes for each model[3] and noticed common milestones for the modeling of the environment of an SAS. The milestones were as follows:

- Modeling the system boundary and environmental factors
- Modeling the environmental impact on the system goal
- Modeling interfaces of the sys.-env. interactions
- Modeling the variability of the environment

All 14 modeling processes included at least one milestone. The first milestone is identifying the system boundary and enumerating the environmental factors that are outside of the system boundary. The second milestone focuses on the goal of the SAS and models how the environment affects the goal. The third milestone highlights the boundary between the SAS and the environment. It represents how the SAS and the environment utilize their interfaces, such as sensors and actuators. The fourth milestone models the variability of the environment. It expresses the environment that is able to change itself over time or is changed by the SAS. It is not necessary to achieve all the milestones, and they do not need to be achieved in a sequential order. The choice of milestones depends on the modeling purpose.

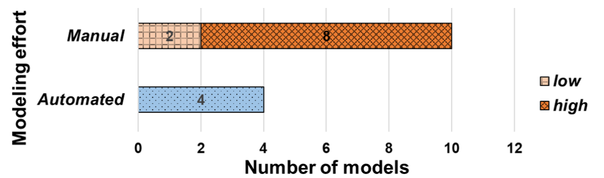

Fig. 5. Modeling efforts of the environment models

In addition, we examined the modeling efforts for each model, and these are summarized in Fig. 5. We divided the modeling efforts into automated and manual modeling. Automated modeling generates environment models automatically through the use of data by their methods. Manual modeling was divided into two cases. The first case (high) is when significant expert-level environment knowledge, such as how that behaves or which environmental conditions are expected, is required. The second case (low) is when modeling can be completed with assistance, such as data, without significant knowledge. Among the 14 models, only four were modeled automatically. The others were manually modeled. It is natural for engineers to manually build models for their purposes. However, the fact that most manual models require significant environmental knowledge suggests that the results of many engineering techniques using environment models can vary, depending on the quality of engineer's knowledge.

RQ5) Application of the environment models: In answering this RQ, we examined how the environment models were used. We summarized the applications of the models in Fig. 6. We categorized the four usages of the models. The first was requirement analysis. Some environment models were used to explicitly identify environmental factors and elicit requirements affected by them. Another application was using the environment models as verification environments to mimic the actual environments of SASs. This was the most common usage. Another way to use the environment model was in generating testing inputs for an SAS. The environment models

 doi:10.20944/preprints202104.0648.v1

TABLE IX
MODELS OF THE ENVIRONMENT OF SAS

ID	Model name	Representation	Modeling purpose / Usage	Formalism	Ref.
M1	Interactive app model (IAM)	Program, program interface, environment, environment interface, uncertainty, configuration	SAS testing environment	-	P17
M2	RELAX-marked conceptual environment model	Environment, environmental factors, sensors, etc.	Uncertainty-aware requirements elicitation	UML class diagram	P25
M3	PRISM stochastic environment game player model	System, environment, sensor model	Uncertainty-aware formal analysis	MDP	P28
M4	Environment model of Tropos4AS	Environmental artifacts, relationships to system agents	Testing environment code generation	UML class diagram	P36
M5	DTMC environment model	Stochastic environmental change	Optimal adaptation decision making	DTMC	P44
M6	Environmental constraint graph	Environmental states and their correlations	Improving model checking validity	Graph	P52
M7	Learning Petri net environment model	Environment states	Formal analysis of SAS behavior and environment	Petri net	P63
M8	Environment domain model	Environment state changes responding to system actions	Runtime behavior model revision	-	P66
M9	Interactive state machine and uncertainty specification	Environmental change, sensor and actuator noise, and environmental constraints	Uncertainty-aware and realistic verification	State machine	P72
M10	Ragnarok uncertainty genome	Numeric information of uncertainty sources	Exploring adverse environmental conditions	-	P82
M11	Game of testing environment model	Environmental state change responding to system actions	Environment model learning for runtime testing	MDP	P97
M12	Contextual variable dependency tree	Contextual variable states and their dependencies	Environment-aware requirements elicitation	Tree	P124
M13	System-environment interaction state model	Interactions between a software system and environment	Optimal adaptation decision making	State machine	P127
M14	Environment configuration variability and reconfiguration model	Environment situation variabilities and reconfiguration process	Environmental condition test case generation	-	P128

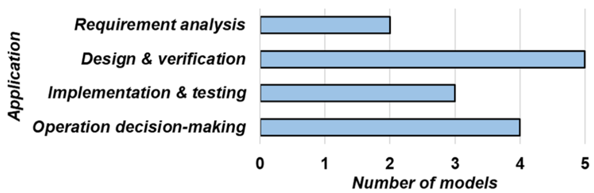

Fig. 6. Application of the environment models

of verification and testing were used to explore failures of an SAS that were triggered by the environment. The last application of the models was for decision-making of an SAS during operation. The environment models were generated or updated during runtime and help an SAS to make optimal decisions in the runtime environment. The usage of each model is also presented in Table IX.

We also summarized techniques that support leveraging the models but do not present them here due to a lack of space (they are available on our website[3]). However, one point that we would like to share here is that a common supportive technique of 11 models was simulation, which was seen as the most fundamental use of environmental models.

RQ6) Expressiveness of the environment models: Finally, we examined how the characteristics of the environment (revealed in answering RQ2) were represented in the models. Fig. 7 shows the analyzed results for each characteristic (the details of each model can be found on our website[3]). For diversity (Fig. 7a), five models required explicit modeling of each environmental factor. They highlight the independence of the factors and can also represent the interaction among the factors. The other models implicitly show that an environmental condition comprises diverse variables. For externality (Fig. 7b), ten environment models were decoupled from the system model. However, the other models were coupled with

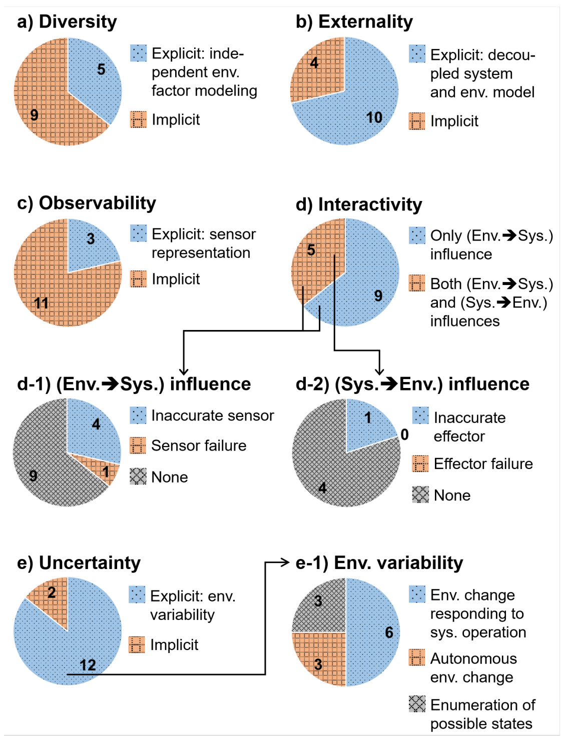

Fig. 7. Representations of the SAS environment characteristics in the models

the system model, and externality was implicitly a part of their modeling process. For observability (Fig. 7c), only three models explicitly described how the environment is monitored by an SAS representing sensor interfaces for environment observation. The others did not show an observation mechanism in the model but just assumed it.

For interactivity (Fig. 7d), all the models illustrated interactions between the environment and the SAS, but the direction of interaction influence can be in either direction. First, environmental conditions can affect the SAS; second, SAS behaviors can affect the environment. Nine models represented only how the environment affects the system. They showed how the SAS goal is affected by or how the SAS reacts to the various environmental conditions. Only five models represented two-way interactions. They modeled how the environment was changed by SAS's behaviors, in addition to the SAS's reaction to the environment. When modeling the interactions, the incompleteness of the environment was also represented in some models. Among the models expressing environmental influence on the SAS (Fig. 7d-1), only four representations of inaccurate sensors, such as sensor noise, and one representation of sensor failure were found. Among the models expressing the SAS behavior's influence on the environment (Fig. 7d-2), only one representation of an effector or actuator possibly being inaccurate was found. This demonstrates that most models so far assume ideal interactions.

With regard to uncertainty (Fig. 7e), although there may be various ways to represent this in the environment, we included how models represented the variability of the environment because most models did this. 12 models explicitly represented the variability of the environment, but two models just assumed the environmental condition can vary over time and not represent it. Among the 12 models (Fig. 7e1), six models represented how the environment responds to the SAS operation, and the other three modeled autonomous changes in the environment over time. They usually specified environmental states and reactive or autonomous state transitions. The other three models represented the variability as an enumeration of possible environmental states. In answering RQ6, we found that every model had a unique expression for the characteristics of the environment depending on perspective, and we have shown the trends of those expressions.

V. Lessons Learned

The analyzed SLR results were described in the previous section; in this section, we present lessons learned through this SLR. First, we found the following four common perspectives for specifying or modeling the environment of an SAS:

- *SAS boundary and external factors*: Identifying a system boundary is essential in defining the environment; identifying the external environmental factors then follows.
- *Environmental impact on the SAS goal*: Understanding how the environment affects the SAS goal is important to make the purpose of adaptation clear.
- *Interface of the SAS-environment interaction*: The interfaces between the environment and the SAS, such as monitored environmental variables, actuating variables of the SAS, or specification of incomplete interaction (e.g., noise or failure) should be identified.
- *Variability of the environment*: Change of an environment over time or by the SAS should be identified for analysis by the SAS in the environment.

These four perspectives will help to sufficiently consider various aspects of the environment throughout the whole development process of the SAS as well as in the modeling.

Second, we identified some research challenges and limitations of the existing SAS's environment modeling, as follows:

- *Limited consideration of various characteristics of environment*: Few papers systematically identified the characteristics of the SAS environment prior to this work, so the various characteristics of environments were not often explicitly expressed. Future modeling should reflect diverse characteristics and perspectives of the SAS environment.
- *Limited consideration of various sources of environmental uncertainty*: Although there are various sources of environmental uncertainty, existing models do not represent them comprehensively. Future research should also address complex environmental uncertainty in which various sources are combined.
- *Considerable manual effort and domain knowledge required for modeling*: Adaptions based on environment models are increasing, but they still rely on manual models and domain knowledge. For effective use of the environment model, more research on automated or data-driven model generation is needed.

To overcome the limitations, this SLR provides background knowledge about the environment of SASs.

VI. Threats to Validity

Every survey paper must deal with the threat of poor representativeness in the primary studies that have been chosen. To reduce this threat, we designed a systematic review protocol (shown in Section III), so that we could exhaustively search for related papers as much as possible and select primary studies objectively. In addition, having and presenting a review protocol makes our method reproducible by readers. Another threat is the possibility of biased analysis of the collected data. To reduce this threat, we tried to utilize the existing expressions or terms and mostly reorganize them for the analysis, as shown in Tables VII and VIII. Nevertheless, because the authors' viewpoints can inevitably be reflected in the interpretation of the data, all raw data extracted from the primary studies are disclosed[3] so that anyone can re-examine it and our conclusions.

VII. Related Surveys on SAS

Several systematic literature reviews describe the landscape of research on SAS engineering. Weyns et al. identified tradeoffs of architectural self-adaptation and proposed research directions [7]. Yang et al. *et al.* [8] and Sucipto et al. [9] investigated requirement engineering approaches for SASs. Muccini et al. analyzed the state of the art of architectural

 doi:10.20944/preprints202104.0648.v1

adaptation approaches for cyber-physical systems [10]. Da Silva et al. investigated UML-based modeling languages for SASs [11]. Recently, applications of machine learning in SASs were surveyed by Saputri et al. [12]. So far, there has been no systematic survey to organize the concepts and models of the environment of an SAS; and as far as we know, this is the first SLR to address them.

VIII. Conclusion

In this paper, to examine the landscape of understanding regarding the environment of SASs, we conducted an SLR and examined how primary studies described the concepts of the environment; in addition, we investigated how the studies represented the environment as models. Following a systematic review protocol, we collected 128 primary studies and 14 unique environment models among the primary studies. Through the SLR, we organized five characteristics of the environment of an SAS (diversity, externality, observability, interactivity, and uncertainty) and two main sources of environmental uncertainty (limited environmental knowledge and incomplete environmental interaction). Based on them, we analyzed trends in what aspects of an uncertain SAS environment have been addressed by the selected studies. It was shown that the characteristics of the environment were variously represented in the environment models for each study. This paper provided a view of the current understanding of the environment to which an SAS should adapt, along with evidence from the primary studies. This will guide future research by providing knowledge of the environment to be considered in SAS development and environment modeling. In addition, we provided a perspective that enables environment specification from diverse aspects.

References

[1] D. Weyns, "Software engineering of self-adaptive systems: an organised tour and future challenges," *Chapter in Handbook of Software Engineering*, 2017.

[2] 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*. Springer, 2013, pp. 1–32.

[3] D. Weyns, *An Introduction to Self-adaptive Systems: A Contemporary Software Engineering Perspective*. John Wiley & Sons, 2020.

[4] D. Budgen and P. Brereton, "Performing systematic literature reviews in software engineering," in *Proceedings of the 28th international conference on Software engineering*, 2006, pp. 1051–1052.

[5] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, 2014, pp. 1–10.

[6] M. Salehie and L. Tahvildari, "Self-adaptive software: Landscape and research challenges," *ACM transactions on autonomous and adaptive systems (TAAS)*, vol. 4, no. 2, pp. 1–42, 2009.

[7] D. Weyns and T. Ahmad, "Claims and evidence for architecture-based self-adaptation: A systematic literature review," in *European Conference on Software Architecture*. Springer, 2013, pp. 249–265.

[8] Z. Yang, Z. Li, Z. Jin, and Y. Chen, "A systematic literature review of requirements modeling and analysis for self-adaptive systems," in *International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer, 2014, pp. 55–71.

[9] S. Sucipto and R. S. Wahono, "A systematic literature review of requirements engineering for self-adaptive systems," *Journal of Software Engineering*, vol. 1, no. 1, pp. 17–27, 2015.

[10] H. Muccini, M. Sharaf, and D. Weyns, "Self-adaptation for cyber-physical systems: a systematic literature review," in *Proceedings of the 11th international symposium on software engineering for adaptive and self-managing systems*, 2016, pp. 75–81.

[11] J. P. S. da Silva, M. Ecar, M. S. Pimenta, G. T. Guedes, L. P. Franz, and L. Marchezan, "A systematic literature review of uml-based domain-specific modeling languages for self-adaptive systems," in *Proceedings of the 13th International Conference on Software Engineering for Adaptive and Self-Managing Systems*, 2018, pp. 87–93.

[12] T. R. D. Saputri and S.-W. Lee, "The application of machine learning in self-adaptive systems: A systematic literature review," *IEEE Access*, vol. 8, pp. 205 948–205 967, 2020.

Primary Studies

[P1] L. Zou, H. Yang, and L. Xu, "An approach to controlling context by combining game theory and control theory for self-adaptive software in creative computing," in *2016 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)*. IEEE, 2016, pp. 311–317.

[P2] R. Lin, B. Chen, Y. Xie, X. Peng, and W. Zhao, "Learning-based multi-controller coordination for self-optimization," in *2012 IEEE 36th Annual Computer Software and Applications Conference Workshops*. IEEE, 2012, pp. 164–169.

[P3] L. Shen, X. Peng, and W. Zhao, "Quality-driven self-adaptation: Bridging the gap between requirements and runtime architecture by design decision," in *2012 IEEE 36th Annual Computer Software and Applications Conference*. IEEE, 2012, pp. 185–194.

[P4] T. Zhao, "The generation and evolution of adaptation rules in requirements driven self-adaptive systems," in *2016 IEEE 24th International Requirements Engineering Conference (RE)*. IEEE, 2016, pp. 456–461.

[P5] M. Camilli, A. Gargantini, and P. Scandurra, "Specifying and verifying real-time self-adaptive systems," in *2015 IEEE 26th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2015, pp. 303–313.

[P6] S. Monpratarnchai and T. Tetsuo, "Applying adaptive role-based model to self-adaptive system constructing problems: a case study," in *2011 Eighth IEEE International Conference and Workshops on Engineering of Autonomic and Autonomous Systems*. IEEE, 2011, pp. 69–78.

[P7] M. Camilli, C. Bellettini, A. Gargantini, and P. Scandurra, "Online model-based testing under uncertainty," in *2018 IEEE 29th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2018, pp. 36–46.

[P8] V. Klös, T. Göthel, and S. Glesner, "Formal models for analysing dynamic adaptation behaviour in real-time systems," in *2016 IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS* W)*. IEEE, 2016, pp. 106–111.

[P9] A. Vialon, K. Tei, and S. Aknine, "Soft-goal approximation context awareness of goal-driven self-adaptive systems," in *2017 IEEE International Conference on Autonomic Computing (ICAC)*. IEEE, 2017, pp. 233–238.

[P10] M. Tamersoy, E. E. Ekinci, R. C. Erdur, and O. Dikenelli, "A requirements model for adaptive multi-organizational systems," in *2017 IEEE 11th International Conference on Self-Adaptive and Self-Organizing Systems (SASO)*. IEEE, 2017, pp. 41–50.

[P11] A. Filieri, H. Hoffmann, and M. Maggio, "Automated design of self-adaptive software with control-theoretical formal guarantees," in *Proceedings of the 36th International Conference on Software Engineering*, 2014, pp. 299–310.

[P12] Y. Liu, W. Zhang, and W. Jiao, "A generative genetic algorithm for evolving adaptation rules of software systems," in *Proceedings of the 8th Asia-Pacific Symposium on Internetware*, 2016, pp. 103–107.

[P13] Y. Liu, D. Bai, and W. Jiao, "Generating adaptation rules of software systems: A method based on genetic algorithm," in *Proceedings of the 2018 10th International Conference on Machine Learning and Computing*, 2018, pp. 347–356.

[P14] G. S. Rodrigues, F. P. Guimarães, G. N. Rodrigues, A. Knauss, J. P. C. de Araújo, H. Andrade, and R. Ali, "Goald: A goal-driven deployment framework for dynamic and heterogeneous computing environments," *Information and software technology*, vol. 111, pp. 159–176, 2019.

[P15] M. Camilli, A. Gargantini, and P. Scandurra, "Zone-based formal specification and timing analysis of real-time self-adaptive systems," *Science of Computer Programming*, vol. 159, pp. 28–57, 2018.

[P16] N. Ferry, F. Chauvel, H. Song, and A. Solberg, "Towards meta-adaptation of dynamic adaptive systems with models@ runtime." in *MODELSWARD*, 2017, pp. 503–508.

[P17] Y. Qin, C. Xu, P. Yu, and J. Lu, "Sit: Sampling-based interactive testing for self-adaptive apps," *Journal of Systems and Software*, vol. 120, pp. 70–88, 2016.

[P18] J. M. Franco, F. Correia, R. Barbosa, M. Zenha-Rela, B. Schmerl, and D. Garlan, "Improving self-adaptation planning through software architecture-based stochastic modeling," *Journal of Systems and software*, vol. 115, pp. 42–60, 2016.

[P19] A. M. Madni and M. Sievers, "Combining formal and probabilistic modeling in resilient systems design," *Procedia Computer Science*, vol. 153, pp. 343–351, 2019.

[P20] S. Andrade and R. Macêdo, "Principled eliciting and evaluation of trade-offs when designing self-adaptive systems architectures," in *Managing Trade-Offs in Adaptable Software Architectures*. Elsevier, 2017, pp. 181–202.

[P21] Q.-L. Yang, J. Lv, X.-P. Tao, X.-X. Ma, J.-C. Xing, and W. Song, "Fuzzy self-adaptation of mission-critical software under uncertainty," *Journal of Computer Science and Technology*, vol. 28, no. 1, pp. 165–187, 2013.

[P22] N. Esfahani and S. Malek, "Uncertainty in self-adaptive software systems," in *Software Engineering for Self-Adaptive Systems II*. Springer, 2013, pp. 214–238.

[P23] J. Andersson, L. Baresi, N. Bencomo, R. De Lemos, A. Gorla, P. Inverardi, and T. Vogel, "Software engineering processes for self-adaptive systems," in *Software Engineering for Self-Adaptive Systems II*. Springer, 2013, pp. 51–75.

[P24] J. Van Der Donckt, D. Weyns, M. U. Iftikhar, and S. S. Buttar, "Effective decision making in self-adaptive systems using cost-benefit analysis at runtime and online learning of adaptation spaces," in *International Conference on Evaluation of Novel Approaches to Software Engineering*. Springer, 2018, pp. 373–403.

[P25] J. Whittle, P. Sawyer, N. Bencomo, B. H. Cheng, and J.-M. Bruel, "Relax: a language to address uncertainty in self-adaptive systems requirement," *Requirements Engineering*, vol. 15, no. 2, pp. 177–196, 2010.

[P26] K. Welsh and P. Sawyer, "Understanding the scope of uncertainty in dynamically adaptive systems," in *International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer, 2010, pp. 2–16.

[P27] M. Dragone, "Building self-adaptive software systems with component, services & agents technologies: Self-osgi," in *International Conference on Agents and Artificial Intelligence*. Springer, 2012, pp. 300–316.

[P28] J. Cámara, W. Peng, D. Garlan, and B. Schmerl, "Reasoning about sensing uncertainty in decision-making for self-adaptation," in *International Conference on Software Engineering and Formal Methods*. Springer, 2017, pp. 523–540.

[P29] A. Filieri, C. Ghezzi, and G. Tamburrelli, "A formal approach to adaptive software: continuous assurance of non-functional requirements," *Formal Aspects of Computing*, vol. 24, no. 2, pp. 163–186, 2012.

[P30] S. Paal, R. Kammüller, and B. Freisleben, "Crosslets: Self-managing application deployment in a cross-platform operating environment," in *International Working Conference on Component Deployment*. Springer, 2005, pp. 52–66.

[P31] G. H. Alférez and V. Pelechano, "Dynamic evolution of context-aware systems with models at runtime," in *International Conference on Model Driven Engineering Languages and Systems*. Springer, 2012, pp. 70–86.

[P32] E. Vassev and M. Hinchey, "The knowlang approach to self-adaptation," in *Software, Services, and Systems*. Springer, 2015, pp. 676–692.

[P33] W. Qian, X. Peng, B. Chen, J. Mylopoulos, H. Wang, and W. Zhao, "Rationalism with a dose of empiricism: combining goal reasoning and case-based reasoning for self-adaptive software systems," *Requirements Engineering*, vol. 20, no. 3, pp. 233–252, 2015.

[P34] C. Hernández, J. L. Fernández, G. Sánchez-Escribano, J. Bermejo-Alonso, and R. Sanz, "Model-based metacontrol for self-adaptation," in *International Conference on Intelligent Robotics and Applications*. Springer, 2015, pp. 643–654.

[P35] K. Welsh, N. Bencomo, P. Sawyer, and J. Whittle, "Self-explanation in adaptive systems based on runtime goal-based models," in *Transactions on Computational Collective Intelligence XVI*. Springer, 2014, pp. 122–145.

[P36] M. Morandini, L. Penserini, A. Perini, and A. Marchetto, "Engineering requirements for adaptive systems," *Requirements Engineering*, vol. 22, no. 1, pp. 77–103, 2017.

[P37] A. J. Ramirez, B. H. Cheng, N. Bencomo, and P. Sawyer, "Relaxing claims: Coping with uncertainty while evaluating assumptions at run time," in *International Conference on Model Driven Engineering Languages and Systems*. Springer, 2012, pp. 53–69.

[P38] E. M. Fredericks, A. J. Ramirez, and B. H. Cheng, "Validating code-level behavior of dynamic adaptive systems in the face of uncertainty," in *International Symposium on Search Based Software Engineering*. Springer, 2013, pp. 81–95.

[P39] V. E. S. Souza, A. Lapouchnian, and J. Mylopoulos, "Requirements-driven qualitative adaptation," in *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer, 2012, pp. 342–361.

[P40] E. M. Fredericks, B. DeVries, and B. H. Cheng, "Autorelax: automatically relaxing a goal model to address uncertainty," *Empirical Software Engineering*, vol. 19, no. 5, pp. 1466–1501, 2014.

[P41] T. Chen, K. Li, R. Bahsoon, and X. Yao, "Femosaa: Feature-guided and knee-driven multi-objective optimization for self-adaptive software," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 27, no. 2, pp. 1–50, 2018.

[P42] S. Shevtsov, D. Weyns, and M. Maggio, "Simca* a control-theoretic approach to handle uncertainty in self-adaptive systems with guarantees," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 13, no. 4, pp. 1–34, 2019.

[P43] K. Angelopoulos, A. V. Papadopoulos, V. E. S. Souza, and J. Mylopoulos, "Engineering self-adaptive software systems: From requirements to model predictive control," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 13, no. 1, pp. 1–27, 2018.

[P44] G. A. Moreno, J. Cámara, D. Garlan, and B. Schmerl, "Flexible and efficient decision-making for proactive latency-aware self-adaptation," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 13, no. 1, pp. 1–36, 2018.

[P45] R. D. Nicola, M. Loreti, R. Pugliese, and F. Tiezzi, "A formal approach to autonomic systems programming: The scel language," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 9, no. 2, pp. 1–29, 2014.

[P46] R. Calinescu, D. Weyns, S. Gerasimou, M. U. Iftikhar, I. Habli, and T. Kelly, "Engineering trustworthy self-adaptive software with dynamic assurance cases," *IEEE Transactions on Software Engineering*, vol. 44, no. 11, pp. 1039–1069, 2017.

[P47] A. Filieri, G. Tamburrelli, and C. Ghezzi, "Supporting self-adaptation via quantitative verification and sensitivity analysis at run time," *IEEE Transactions on Software Engineering*, vol. 42, no. 1, pp. 75–99, 2015.

[P48] L. Rosa, L. Rodrigues, A. Lopes, M. Hiltunen, and R. Schlichting, "Self-management of adaptable component-based applications," *IEEE Transactions on Software Engineering*, vol. 39, no. 3, pp. 403–421, 2012.

[P49] D. Cooray, E. Kouroshfar, S. Malek, and R. Roshandel, "Proactive self-adaptation for improving the reliability of mission-critical, embedded, and mobile software," *IEEE Transactions on Software Engineering*, vol. 39, no. 12, pp. 1714–1735, 2013.

[P50] W. Yang, C. Xu, M. Pan, C. Cao, X. Ma, and J. Lu, "Efficient validation of self-adaptive applications by counterexample probability maximization," *Journal of Systems and Software*, vol. 138, pp. 82–99, 2018.

[P51] M. Ahmad, N. Belloir, and J.-M. Bruel, "Modeling and verification of functional and non-functional requirements of ambient self-adaptive systems," *Journal of Systems and Software*, vol. 107, pp. 50–70, 2015.

[P52] Y. Liu, C. Xu, and S. C. Cheung, "Afchecker: Effective model checking for context-aware adaptive applications," *Journal of Systems and Software*, vol. 86, no. 3, pp. 854–867, 2013.

[P53] T. Patikirikorala, A. Colman, J. Han, and L. Wang, "An evaluation of multi-model self-managing control schemes for adaptive performance management of software systems," *Journal of Systems and Software*, vol. 85, no. 12, pp. 2678–2696, 2012.

[P54] M. Amoui, M. Derakhshanmanesh, J. Ebert, and L. Tahvildari, "Achieving dynamic adaptation via management and interpretation of runtime models," *Journal of Systems and Software*, vol. 85, no. 12, pp. 2720–2737, 2012.

[P55] C. Xu, S. C. Cheung, X. Ma, C. Cao, and J. Lu, "Adam: Identifying defects in context-aware adaptation," *Journal of Systems and Software*, vol. 85, no. 12, pp. 2812–2828, 2012.

[P56] M. Salifu, Y. Yu, A. K. Bandara, and B. Nuseibeh, "Analysing monitoring and switching problems for adaptive systems," *Journal of Systems and Software*, vol. 85, no. 12, pp. 2829–2839, 2012.

[P57] N. Gui, V. De Florio, H. Sun, and C. Blondia, "Toward architecture-based context-aware deployment and adaptation," *Journal of Systems and Software*, vol. 84, no. 2, pp. 185–197, 2011.

[P58] T. Chaari, D. Ejigu, F. Laforest, and V.-M. Scuturici, "A comprehensive approach to model and use context for adapting applications in pervasive environments," *Journal of Systems and Software*, vol. 80, no. 12, pp. 1973–1992, 2007.

[P59] D. Han, Q. Yang, J. Xing, J. Li, and H. Wang, "Fame: A uml-based framework for modeling fuzzy self-adaptive software," *Information and Software Technology*, vol. 76, pp. 118–134, 2016.

[P60] G. Chatzikonstantinou and K. Kontogiannis, "Run-time requirements verification for reconfigurable systems," *Information and Software Technology*, vol. 75, pp. 105–121, 2016.

[P61] A. Knauss, D. Damian, X. Franch, A. Rook, H. A. Müller, and A. Thomo, "Acon: A learning-based approach to deal with uncertainty in contextual requirements at runtime," *Information and software technology*, vol. 70, pp. 85–99, 2016.

[P62] G. Tziallas and B. Theodoulidis, "A controller synthesis algorithm for building self-adaptive software," *Information and Software Technology*, vol. 46, no. 11, pp. 719–727, 2004.

[P63] Z. Ding, Y. Zhou, and M. Zhou, "Modeling self-adaptive software systems with learning petri nets," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 4, pp. 483–498, 2015.

[P64] N. D'Ippolito, V. Braberman, J. Kramer, J. Magee, D. Sykes, and S. Uchitel, "Hope for the best, prepare for the worst: multi-tier control for adaptive systems," in *Proceedings of the 36th International Conference on Software Engineering*, 2014, pp. 688–699.

[P65] C. Ghezzi, L. S. Pinto, P. Spoletini, and G. Tamburrelli, "Managing non-functional uncertainty via model-driven adaptivity," in *2013 35th International Conference on Software Engineering (ICSE)*. IEEE, 2013, pp. 33–42.

[P66] D. Sykes, D. Corapi, J. Magee, J. Kramer, A. Russo, and K. Inoue, "Learning revised models for planning in adaptive systems," in *2013 35th International Conference on Software Engineering (ICSE)*. IEEE, 2013, pp. 63–71.

[P67] G. A. Moreno, J. Cámara, D. Garlan, and B. Schmerl, "Proactive self-adaptation under uncertainty: a probabilistic model checking approach," in *Proceedings of the 2015 10th joint meeting on foundations of software engineering*, 2015, pp. 1–12.

[P68] A. Filieri, H. Hoffmann, and M. Maggio, "Automated multi-objective control for self-adaptive software design," in *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, 2015, pp. 13–24.

[P69] S. Shevtsov and D. Weyns, "Keep it simplex: Satisfying multiple goals with guarantees in control-based self-adaptive systems," in *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 2016, pp. 229–241.

[P70] A. Elkhodary, N. Esfahani, and S. Malek, "Fusion: a framework for engineering self-tuning self-adaptive software systems," in *Proceedings of the eighteenth ACM SIGSOFT international symposium on Foundations of software engineering*, 2010, pp. 7–16.

[P71] M. Sama, D. S. Rosenblum, Z. Wang, and S. Elbaum, "Model-based fault detection in context-aware adaptive applications," in *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering*, 2008, pp. 261–271.

[P72] W. Yang, C. Xu, Y. Liu, C. Cao, X. Ma, and J. Lu, "Verifying self-adaptive applications suffering uncertainty," in *Proceedings of the 29th ACM/IEEE international conference on Automated software engineering*, 2014, pp. 199–210.

[P73] C. Xu, W. Yang, X. Ma, C. Cao, and J. Lü, "Environment rematching: toward dependability improvement for self-adaptive applications," in *2013 28th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 2013, pp. 592–597.

[P74] A. Filieri, C. Ghezzi, A. Leva, and M. Maggio, "Self-adaptive software meets control theory: A preliminary approach supporting reliability requirements," in *2011 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011)*. IEEE, 2011, pp. 283–292.

[P75] A. J. Ramirez, A. C. Jensen, B. H. Cheng, and D. B. Knoester, "Automatically exploring how uncertainty impacts behavior of dynamically adaptive systems," in *2011 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011)*. IEEE, 2011, pp. 568–571.

[P76] H. Tajalli, J. Garcia, G. Edwards, and N. Medvidovic, "Plasma: a plan-based layered architecture for software model-driven adaptation," in *Proceedings of the IEEE/ACM international conference on Automated software engineering*, 2010, pp. 467–476.

[P77] G. F. Solano, R. D. Caldas, G. N. Rodrigues, T. Vogel, and P. Pelliccione, "Taming uncertainty in the assurance process of self-adaptive systems: a goal-oriented approach," in *2019 IEEE/ACM 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, 2019, pp. 89–99.

[P78] 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 *2018 IEEE/ACM 13th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, 2018, pp. 19–25.

[P79] G. A. Moreno, J. Cámara, D. Garlan, and M. Klein, "Uncertainty reduction in self-adaptive systems," in *Proceedings of the 13th International Conference on Software Engineering for Adaptive and Self-Managing Systems*, 2018, pp. 51–57.

[P80] A. Cailliau and A. V. Lamsweerde, "Runtime monitoring and resolution of probabilistic obstacles to system goals," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 14, no. 1, pp. 1–40, 2019.

[P81] P. Weisenburger, M. Luthra, B. Koldehofe, and G. Salvaneschi, "Quality-aware runtime adaptation in complex event processing," in *2017 IEEE/ACM 12th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, 2017, pp. 140–151.

[P82] E. M. Fredericks, "Automatically hardening a self-adaptive system against uncertainty," in *2016 IEEE/ACM 11th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, 2016, pp. 16–27.

[P83] K. Angelopoulos, A. V. Papadopoulos, V. E. S. Souza, and J. Mylopoulos, "Model predictive control for software systems with cobra," in *2016 IEEE/ACM 11th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, 2016, pp. 35–46.

[P84] L. Nahabedian, V. Braberman, N. D'Ippolito, S. Honiden, J. Kramer, K. Tei, and S. Uchitel, "Assured and correct dynamic update of controllers," in *2016 IEEE/ACM 11th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, 2016, pp. 96–107.

[P85] A. M. Sharifloo, A. Metzger, C. Quinton, L. Baresi, and K. Pohl, "Learning and evolution in dynamic software product lines," in *2016 IEEE/ACM 11th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, 2016, pp. 158–164.

[P86] S. Hassan, N. Bencomo, and R. Bahsoon, "Minimizing nasty surprises with better informed decision-making in self-adaptive systems," in *2015 IEEE/ACM 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*. IEEE, 2015, pp. 134–145.

[P87] E. M. Fredericks and B. H. Cheng, "Automated generation of adaptive test plans for self-adaptive systems," in *2015 IEEE/ACM 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*. IEEE, 2015, pp. 157–167.

[P88] E. M. Fredericks, B. DeVries, and B. H. Cheng, "Towards run-time adaptation of test cases for self-adaptive systems in the face of uncertainty," in *Proceedings of the 9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, 2014, pp. 17–26.

[P89] D. F. Mendonça, R. Ali, and G. N. Rodrigues, "Modelling and analysing contextual failures for dependability requirements," in *Proceedings of the 9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, 2014, pp. 55–64.

[P90] M. U. Iftikhar and D. Weyns, "Activforms: Active formal models for self-adaptation," in *Proceedings of the 9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, 2014, pp. 125–134.

[P91] J. Cámara, G. A. Moreno, and D. Garlan, "Stochastic game analysis and latency awareness for proactive self-adaptation," in *Proceedings of the 9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, 2014, pp. 155–164.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 April 2021 doi:10.20944/preprints202104.0648.v1

[P92] E. M. Fredericks, A. J. Ramirez, and B. H. Cheng, "Towards run-time testing of dynamic adaptive systems," in *2013 8th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, 2013, pp. 169–174.

[P93] J. Cámara and R. De Lemos, "Evaluation of resilience in self-adaptive systems using probabilistic model-checking," in *2012 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, 2012, pp. 53–62.

[P94] A. J. Ramirez, A. C. Jensen, and B. H. Cheng, "A taxonomy of uncertainty for dynamically adaptive systems," in *2012 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, 2012, pp. 99–108.

[P95] T. Patikirikorala, A. Colman, J. Han, and L. Wang, "A multi-model framework to implement self-managing control systems for qos management," in *Proceedings of the 6th international symposium on software engineering for adaptive and self-managing systems*, 2011, pp. 218–227.

[P96] D. Sykes, W. Heaven, J. Magee, and J. Kramer, "From goals to components: a combined approach to self-management," in *Proceedings of the 2008 international workshop on Software engineering for adaptive and self-managing systems*, 2008, pp. 1–8.

[P97] A. Reichstaller and A. Knapp, "Risk-based testing of self-adaptive systems using run-time predictions," in *2018 IEEE 12th International Conference on Self-Adaptive and Self-Organizing Systems (SASO)*. IEEE, 2018, pp. 80–89.

[P98] S. Vansyckel, D. Schäfer, G. Schiele, and C. Becker, "Configuration management for proactive adaptation in pervasive environments," in *2013 IEEE 7th International Conference on Self-Adaptive and Self-Organizing Systems*. IEEE, 2013, pp. 131–140.

[P99] N. Gui and V. De Florio, "Towards meta-adaptation support with reusable and composable adaptation components," in *2012 IEEE Sixth International Conference on Self-Adaptive and Self-Organizing Systems*. IEEE, 2012, pp. 49–58.

[P100] H. Nakagawa, A. Ohsuga, and S. Honiden, "Towards dynamic evolution of self-adaptive systems based on dynamic updating of control loops," in *2012 IEEE Sixth International Conference on Self-Adaptive and Self-Organizing Systems*. IEEE, 2012, pp. 59–68.

[P101] D. Kim and S. Park, "Reinforcement learning-based dynamic adaptation planning method for architecture-based self-managed software," in *2009 ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems*. IEEE, 2009, pp. 76–85.

[P102] D. Perez-Palacin and R. Mirandola, "Uncertainties in the modeling of self-adaptive systems: A taxonomy and an example of availability evaluation," in *Proceedings of the 5th ACM/SPEC international conference on Performance engineering*, 2014, pp. 3–14.

[P103] J. Cámara, A. Lopes, D. Garlan, and B. Schmerl, "Adaptation impact and environment models for architecture-based self-adaptive systems," *Science of Computer Programming*, vol. 127, pp. 50–75, 2016.

[P104] Z. Liang and Y. Qin, "Generating environmental models for testing self-adaptive systems," in *Proceedings of the 11th Asia-Pacific Symposium on Internetware*, 2019, pp. 1–6.

[P105] J. C. Moreno, A. Lopes, D. Garlan, and B. Schmerl, "Impact models for architecture-based self-adaptive systems," in *International Conference on Formal Aspects of Component Software*. Springer, 2014, pp. 89–107.

[P106] A. Palm, A. Metzger, and K. Pohl, "Online reinforcement learning for self-adaptive information systems," in *International Conference on Advanced Information Systems Engineering*. Springer, 2020, pp. 169–184.

[P107] H. N. Ho and E. Lee, "Model-based reinforcement learning approach for planning in self-adaptive software system," in *Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication*, 2015, pp. 1–8.

[P108] W. Cheng, Q. Li, L. Wang, and L. He, "Handling uncertainty online for self-adaptive systems," in *2018 5th International Conference on Soft Computing & Machine Intelligence (ISCMI)*. IEEE, 2018, pp. 135–139.

[P109] B. H. Cheng, P. Sawyer, N. Bencomo, and J. Whittle, "A goal-based modeling approach to develop requirements of an adaptive system with environmental uncertainty," in *International Conference on Model Driven Engineering Languages and Systems*. Springer, 2009, pp. 468–483.

[P110] J. Whittle, P. Sawyer, N. Bencomo, B. H. Cheng, and J.-M. Bruel, "Relax: Incorporating uncertainty into the specification of self-adaptive systems," in *2009 17th IEEE International Requirements Engineering Conference*. IEEE, 2009, pp. 79–88.

[P111] H. J. Goldsby and B. H. Cheng, "Automatically generating behavioral models of adaptive systems to address uncertainty," in *International Conference on Model Driven Engineering Languages and Systems*. Springer, 2008, pp. 568–583.

[P112] N. Esfahani, A. Elkhodary, and S. Malek, "A learning-based framework for engineering feature-oriented self-adaptive software systems," *IEEE transactions on software engineering*, vol. 39, no. 11, pp. 1467–1493, 2013.

[P113] N. Esfahani, E. Kouroshfar, and S. Malek, "Taming uncertainty in self-adaptive software," in *Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering*, 2011, pp. 234–244.

[P114] G. A. Moreno, J. Cámara, D. Garlan, and B. Schmerl, "Efficient decision-making under uncertainty for proactive self-adaptation," in *2016 IEEE International Conference on Autonomic Computing (ICAC)*. IEEE, 2016, pp. 147–156.

[P115] J. Cámara, D. Garlan, B. Schmerl, and A. Pandey, "Optimal planning for architecture-based self-adaptation via model checking of stochastic games," in *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, 2015, pp. 428–435.

[P116] L. Zhang, C. Xu, X. Ma, T. Gu, X. Hong, C. Cao, and J. Lu, "Resynchronizing model-based self-adaptive systems with environments," in *2012 19th Asia-Pacific Software Engineering Conference*, vol. 1. IEEE, 2012, pp. 184–193.

[P117] E. Zavala, X. Franch, J. Marco, A. Knauss, and D. Damian, "Sacre: Supporting contextual requirements' adaptation in modern self-adaptive systems in the presence of uncertainty at runtime," *Expert Systems with Applications*, vol. 98, pp. 166–188, 2018.

[P118] M. Tanabe, K. Tei, Y. Fukazawa, and S. Honiden, "Learning environment model at runtime for self-adaptive systems," in *Proceedings of the Symposium on Applied Computing*, 2017, pp. 1198–1204.

[P119] S.-W. Cheng, V. V. Poladian, D. Garlan, and B. Schmerl, "Improving architecture-based self-adaptation through resource prediction," in *Software Engineering for Self-Adaptive Systems*. Springer, 2009, pp. 71–88.

[P120] B. Eberhardinger, H. Seebach, A. Knapp, and W. Reif, "Towards testing self-organizing, adaptive systems," in *IFIP International Conference on Testing Software and Systems*. Springer, 2014, pp. 180–185.

[P121] E. M. Fredericks, "An empirical analysis of the mutation operator for run-time adaptive testing in self-adaptive systems," in *2018 IEEE/ACM 11th International Workshop on Search-Based Software Testing (SBST)*. IEEE, 2018, pp. 59–66.

[P122] J. Cámara, W. Peng, D. Garlan, and B. Schmerl, "Reasoning about sensing uncertainty and its reduction in decision-making for self-adaptation," *Science of Computer Programming*, vol. 167, pp. 51–69, 2018.

[P123] C. Ghezzi and A. M. Sharifloo, "Dealing with non-functional requirements for adaptive systems via dynamic software product-lines," in *Software Engineering for Self-Adaptive Systems II*. Springer, 2013, pp. 191–213.

[P124] A. Rodrigues, G. N. Rodrigues, A. Knauss, R. Ali, and H. Andrade, "Enhancing context specifications for dependable adaptive systems: A data mining approach," *Information and software technology*, vol. 112, pp. 115–131, 2019.

[P125] K. Welsh and P. Sawyer, "Managing testing complexity in dynamically adaptive systems: A model-driven approach," in *2010 Third International Conference on Software Testing, Verification, and Validation Workshops*. IEEE, 2010, pp. 290–298.

[P126] P. Inverardi and M. Mori, "Feature oriented evolutions for context-aware adaptive systems," in *Proceedings of the Joint ERCIM Workshop on Software Evolution (EVOL) and International Workshop on Principles of Software Evolution (IWPSE)*, 2010, pp. 93–97.

[P127] K. Aizawa, K. Tei, and S. Honiden, "Analysis space reduction with state merging for ensuring safety properties of self-adaptive systems," in *2019 IEEE (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*. IEEE, 2019, pp. 1363–1370.

[P128] G. Püschel, C. Piechnick, S. Götz, C. Seidl, S. Richly, T. Schlegel, and U. Aßmann, "A combined simulation and test case generation strategy for self-adaptive systems," *Journal On Advances in Software*, vol. 7, no. 3&4, pp. 686–696, 2014.