Quality Attributes for Cyber-Physical Systems and Merit Criteria for their Modeling Tools

Armin Moin
Department of Computer Science University of Colorado, Colorado Springs CO, USA
{amoin@uccs.edu}*Corresponding author

Atta Badii
Department of Computer Science University of Reading
Reading, United Kingdom {atta.badii@reading.acuk}

Fatima Bozyigit and Moharram Challenger
Department of Computer Science University of Antwerp and Flanders Make Antwerp, Belgium
{first.last}@uantwerpen.be}

***Abstract*—Cyber-Physical Systems (CPSs) are systems of systems merging the physical world with the virtual world of cyberspace. CPSs are often highly complex and interdisciplinary. Thus, their design, analysis, development, performance refinement, and testing require the consideration of multiple aspects from various domains. This is even more critical for Machine Learning (ML) enabled CPSs, which are increasingly becoming more prevalent. In this study, we take the first step towards collecting a set of quality attributes for smart CPSs. Further, we introduce a set of merit criteria for tools that can be used for designing, modeling, and developing smart CPSs. This is to provide a framework for the comparison and benchmarking of smart CPSs and CPS modeling tools. The framework has been put through the first phase of expert interview-based validation.**

***Index Terms*—cyber-physical systems, modeling, quality at- tributes, merit criteria, machine learning, qualitative research**

I. Introduction

Cyber-Physical Systems (CPSs) are complex systems that connect the physical world with the virtual world of cyberspace [1]. A modern car, an aircraft, a smart grid, a robot in a production line, or a computer system controlling a chemical process at a plant are examples of CPSs. If a CPS possesses any Artificial Intelligence (AI) or cognitive capability, for example, through a Machine Learning (ML) component, it is called a smart (i.e., ML-enabled) CPS. Further, a CPS might be connected to the Internet (Internet of Things, IoT) or be isolated, possibly due to security and/or privacy concerns. Also, CPSs are typically cross-application domains, cross-technologies, and cross-organizations [2]. Therefore, it is vital to have inclusive and objective measures for evaluating, comparing, and benchmarking smart CPSs from various viewpoints.

For instance, a software engineer might be concerned with the modularity of the source code. By contrast, a data scientist might focus on the accuracy, precision, and recall of the underlying ML model, whereas a data engineer might be keen to minimize the footprint (compactness) and latency of the trained ML model such that it could fit in the highly limited main memory of a particular *Tiny ML* device to deliver high- speed edge analytics. Each of these stakeholders would assess and score the system in a very different manner. Figure 1 illustrates a modern car as an example of a CPS. A key Research Question (RQ) is how we can objectively assess and measure the qualities of different CPSs, for example, two cars, thus comparing them from various perspectives. Also, another key RQ we ask is how to compare different CPS modeling tools.

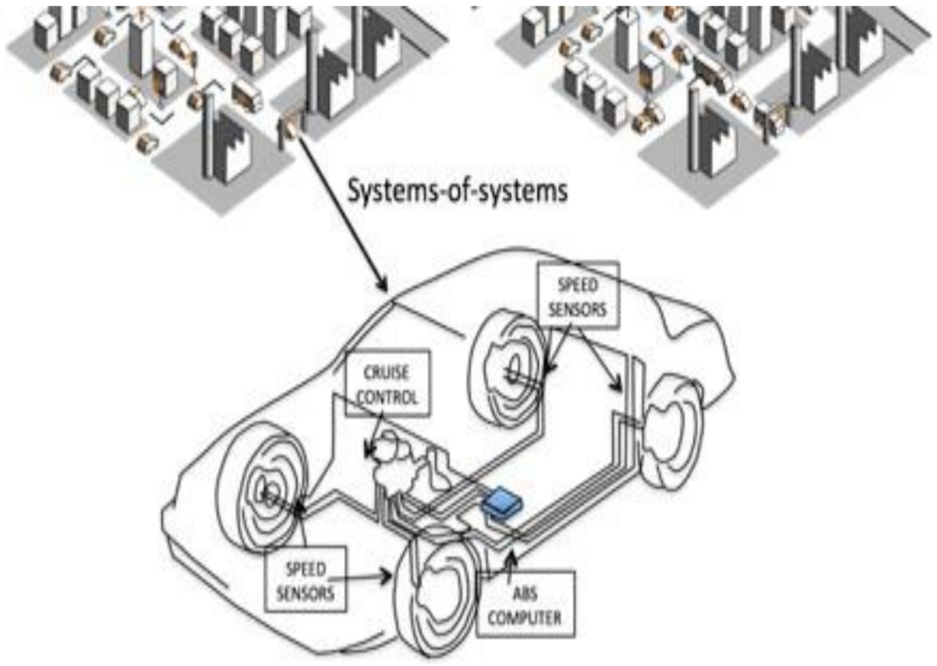

Fig. 1. How to objectively measure the quality of CPSs and compare them?

The contribution of this paper is twofold: i) It proposes a framework of 26 quality attribute groups to support the bench- marking and evaluation of smart CPSs. This includes three subcategories: a) general Systems and Software Engineering (SSE) merit criteria; b) merit criteria related to distributed computing, CPSs, and the IoT; c) merit criteria related to Data Engineering and Analytics (DEA), as well as ML. ii) It proposes a framework of 26 merit criteria groups to help benchmark and evaluate the modeling tools for designing and creating smart CPSs. However, most of the proposed quality attributes and merit criteria are generic, thus being applicable to systems other than CPSs too.

The remainder of this paper is structured as follows: Section II reviews the state of the art. Moreover, Section III elaborates on the proposed quality attributes and merit criteria. Finally, Section IV concludes and suggests future work.

II. Related Work

Recently, several studies have proposed evaluation criteria and metrics for CPSs. Vogel-Heuser and Prieler [3] evaluated selected metrics for the flexibility of Cyber-Physical Production Systems (CPPSs). Moreover, Weyrich et al. [4] proposed an evaluation model for the assessment of CPPSs, which included various identified performance indicators, such as modularity, complexity, maintainability, production efficiency, reconfigurability, automatic planning, automatic adaptation, social interaction, decision support, and usability.

Additionally, related work exists in the literature of Model- Driven Software Engineering (MDSE) regarding evaluation criteria for modeling environments and Domain-Specific Modelling Languages (DSMLs). For instance, Challenger et al. [5] proposed a systematic approach to evaluating DSML environments for Multi-Agent Systems (MAS). Further, Giraldo et al. [6] concentrated on the topic of quality in Model-Driven Engineering (MDE). They elaborated on various facets of quality in MDE and also pointed out a mismatch between the notion of quality in the views of industrial practitioners versus researchers in academia.

III. Quality Attributes and Merit criteria

In this section, we propose two sets of quality at- tributes/merit criteria for CPSs and CPS modeling tools. We have grouped the closely related attributes/criteria in each part (e.g., security and privacy); otherwise, they are listed in an arbitrary order without any intended implication as to ranking. Table I summarizes the demographics of the expert interviewees who contributed to the preliminary validation of the proposed quality attributes/merit criteria framework.

TABLE I

Summary of the demographic information of the interviewees

Type	Breakdown
Sex or gender (12)	female (1), male (11), other (0), no answer (0)
Age group (12)	below 18 (0), 18-24 (0), 25-39 (5), 40-60 (7), 60 plus (0), no answer (0)
Highest degree (12)	Bachelor’s (0), Master’s (2), Ph.D. (10), No academic degree (0), Other (0), no answer (0)
Field of expertise (12)	ML (3), CPSs (3), MDE (2), MDE & CPSs (2), general SSE (2)
Job or occupation (12)	professor (6), researcher (2), industrial practitioner (1), company co-founder (2), project manager (1)

A. Quality attributes for smart CPSs

We categorized the smart CPS quality attributes into three groups to focus the discussions with experts in the domains most relevant to their respective fields of expertise. These categories are as follows: i) general SSE quality attributes (which also hold for CPSs); ii) Quality attributes related to distributed computing, CPSs, and the IoT; iii) Quality attributes pertaining to DA (including ML) and DE (i.e., DEA). Under each category, a set of quality attributes had already been collected based on insights from previous research (see the respective citations). To minimize the risk of interview- induced biasing of the experts’ responses to questions, before the commencement of each interview, we asked each expert to state which criteria they would use. It is acknowledged that the most significant attribute of every system is the performance metric, indicating the extent of fulfillment of the requirements as specified by the stakeholders.

General SSE quality attributes:

1) **Security and privacy protection:** Regulatory compliance by design (as applicable) [7].
2) **Providing high (adaptive) usability, accessibility, user acceptance, as well as social and environmental acceptability:** This includes sustainability considerations such as energy efficiency, carbon emissions footprint, reusability, recyclability, and re-purposability [7]–[11].
3) **Modularity, maintainability, and evolution support:** Simplification by design and eliminating avoidable complexity [7], [11], [12].
4) **Dependability, reliability, trustworthiness, availability, and robustness of performance** [7], [11], [12].
5) **Efficiency, portability, scalability, and concurrency** [11].
6) **Expressivity, explainability, and transparency:** Avoidance of black box solutions as much as possible to suit the user’s needs.
7) **Cost-effectiveness:** Cost-effective to procure and maintain (the latter may overlap with the maintainability criterion above) [7].
8) ***Learning* capability**: That is, ML-enabled.

Quality attributes related to distributed computing, CPSs, and the IoT:

1) **Safety compliance:** Since CPSs deal with the physical world and may interact with humans, they are often safety-critical [13], [14].
2) **Semantic interoperability of heterogeneous systems and open system design:** As subset of the so-called *cross-** CPS properties [1], [2], [15].
3) **Failure recovery handling and resilience:** Fault- tolerance and graceful recovery from failure (e.g., the capability of an aircraft to fly further beyond the point of a single engine failure and possibly glide for a specific distance beyond the point of failure of all its engines). This is a child class of the so- called *self-** CPS properties, exhibited by reflexive de- sign as a root merit criterion which subsumes others, such as self-adaptive, self-learning, self-optimizing, self- monitoring, self-auditing, self-diagnosing, self-healing, self-repairing, self-accountable, self-expressive, and explanation-giving capability [2], [3], [13], [16].
4) **Scalability and latency:** To satisfy the network throughput requirements and real-time performance as applicable IoT requires scalability at a much higher level.

Quality attributes pertaining to Data Engineering and Analytics (DEA):

1) ***Target-based* metrics**, such as Accuracy, Precision, Re- call (i.e., the true positive rate, also known as sensitivity in the case of binary classification), specificity (i.e., the true negative rate, also known as selectivity in the case of binary classification), and F1-measure can be used in the case of classification. Also, a confusion matrix is often useful for summarizing the performance of a classification algorithm. Moreover, in the case of regression, various error metrics, such as Mean-Squared Error (MSE), can be used. It is noted that the target-based metrics may incorporate the above- mentioned supervised ML performance metrics and/or other application-specific metrics depending on the use context, for example, the unavailability of labeled data and ground truth, such as in clustering tasks for which a range of other performance metrics could be deployed. Some notable examples include the Silhouette Score, (Adjusted) Rand Index, Mutual Information, Calinski- Harabasz Index, Davies-Bouldin Index, and Dunn Index [17], [18].
2) **High-speed performance** in terms of time-to-output (e.g., the latency for making predictions).
3) **Generalization:** Robustness of performance even with certain increased noise levels in the datasets or changes in the data.
4) **Handling uncertainty:** The ability to detect that the data are different from the original data, (adversarial) attack detection, the degree to which the system remains capable of maintaining high Area Under the Curve (AUC) scores for the Receiver Operating Characteristic (ROC) curve, and the quality of calibration of the ML model so that we can interpret the output in terms of a probability.
5) **End-to-end ML:** The capability of the ML model to capture the entire ML pipeline (including the pre- processing and post-processing tasks) and act as a *one- stop-shop*, thus offering the so-called end-to-end ML.
6) **Automated ML (AutoML):** The extent to which the ML modeling pipeline stages are (semi)-automated involving stages, such as data pre-processing (data cleaning and de-noising, dimensionality reduction, sparsity mitigation, re-scaling, standardization, re- sampling, and re-balancing), feature extraction, ML model selection, training, and hyperparameter tuning.
7) **Auto annotation:** Capabilities to process data with variable annotation formats, machine-readable data as well as non-semantic data; (class-) labeled, unlabeled, and partially labeled data for supervised, unsupervised, and semi-supervised ML, respectively.
8) **Ethical and legal compliance by design:** This is another root class merit criterion, particularly exhibited as privacy-preserving by design, dignity-preserving, handling algorithmic bias, inclusiveness by design, security by design (e.g., adversarial attacks resistance), environ- mental acceptability (expressed as energy efficiency and having a low carbon footprint).

9) **Explainability:** This distinguishes explainable by- design ML models, such as PGMs, from those ML models that are not explainable by design, such as ANNs.
10) **Scalable batch data processing or offline learning:** Coping with large or very large datasets in the order of 10-100 thousand, or 100 thousand to 1 million instances.
11) **Efficient stream processing or online learning:** Being capable of efficient stream processing or online learning (i.e., dealing with unbounded datasets); and possibly ad- dressing continual or lifelong learning needs; capability to auto-evolve new models.
12) **Supporting transfer learning.**
13) **Disk space and memory usage efficiency:** In particular, in the case of resource-constrained devices, such as *TinyML* platforms [19].
14) **MLOps-enabled**: Support for integrative MLOps- DevOps (MLOps is the application of DevOps from SE to the ML domain).

B. Merit criteria for CPS modeling tools

1. **Domain-specific or domain-agnostic:** If domain- specific, then whether the domain of focus is a problem (use case) domain (e.g., healthcare) or a solution domain (e.g., cloud computing)? Domain-specific tools can often support a higher automation level and generate a higher quality code on average [20]. Moreover, if the focus domain is a problem domain, the user of the tool should be a domain expert in that vertical (application) domain. In contrast, if the domain of focus is a solution domain, the user of the tool should be a practitioner familiar with that area.
2. **Suitability for the application domain:** The expresseness of the modeling language deployed by the tool, the level of abstract syntax completeness and appropriateness (e.g., the meta-model containing the necessary concepts, relationships, and semantics) [5].
3. **Syntax usability:** (i.e., practitioner-friendly syntax) The adequacy and suitability of the vocabulary in the case of textual concrete syntax or the diagrams in the case of graphical concrete syntax.
4. **Modeling and development support:** Model management (e.g., versioning support), traceability, debuggability, documentation support, and testing support.
5. The extent of **support for multiple architecture view- points** (for various stakeholders) [6].
6. The extent of **support for collaborative modeling**.
7. The extent of **support for fully automated code generation:** End-to-end complete solution generation vs. only skeleton generation [5].
8. **Supported target platforms:** Hardware architectures, operating systems, programming languages, and APIs supported for code generation.
9. The extent of **support for model checking and formal verification**.
10. The extent of **support for simulation at the design time**.
11. **Portability and support for web-based access** through web browsers.
12. **Interoperability** with other tools, APIs, and standards.
13. **Supported ML libraries,** frameworks, methods, algorithms, and techniques for code generation of ML components.
14. The extent of **support for ML techniques for non-i.i.d[1] data, such as sequential data (**e.g., time series or DNA data).
15. **Certifiably and compliant with code generation:** Assured standard compliance of generated code (e.g., as required for safety-critical use contexts).
16. **Extensibility and adaptability** of the modeling language, including the model transformations. Also, sup- port for legacy systems.
17. **Open-source availability:** For example, permissive li- cense (such as Apache, MIT, or BSD).
18. The extent of **technical support:** As may be available
19. **User support tutorials:** The extent to which tutorials and examples are available for the users (and developers) as well as their effectiveness towards development efficiency.

[1]The i.i.d. acronym stands for independent and identically distributed

20. **Performance leap:** The extent to which the deployment of the modeling tool enables performance efficiency gain for software designers and developers [21].
21. **Technology Readiness Level (TRL):** Technical maturity of the tool.
22. **Code generation quality:** This includes a range of merit criteria sub-classes, such as functional correctness, efficiency, modularity, and elegance of the generated code [5], [21].
23. **Report generation:** For example, this includes rendering plots and visualization of data to support data analytics.
24. **Support for systems integration and networking:** For example, support for selection of network topology or setting up logical connections (e.g., Virtual Private Networks, VPNs.
25. **Runtime support:** This includes support for asset monitoring and management, for example, support for Over Air Programming ("OTA") for sensors, and MOD- ELS@RUNTIME [22].
26. **DevOps support:** The capability to support DevOps pipelines, for example, Continuous Integration (CI) setup.

IV. Conclusion and future work

In this position paper, we have proposed a framework of quality attributes and merit criteria that support the capability assessment of smart CPSs and CPS modeling tools. The framework comprises two sets that include 52 quality attributes and merit criteria groups for the mentioned systems and tools. We have conducted a literature review and interviewed 12 experts. Our work has taken the initial steps toward developing a comprehensive benchmarking framework for smart CPSs and their modeling tools.

In the future, we plan to prioritize the proposed attributes and criteria, reorganize the grouping, and possibly extend the sets. Additionally, more concrete and measurable metrics and indicators for each attribute/criterion should be determined as part of future work. In some cases, such as the ML target- based metrics, this has already been achieved and elaborated. However, the quantification of some other attributes/criteria remains a key challenge. Additionally, their interrelations and possible trade-offs should be studied in depth. Finally, multiple case studies and further expert interviews will be necessary to validate the final framework.

Acknowledgment

The authors would like to thank the interview partners: Vahid Akram, Ankica Barisic, Alican Bozyigit, Bertrand Char- through an open-source community or a company.

References

[1] E. Geisberger and M. Broy, Eds., *Living in a networked world. Integrated research agenda Cyber-Physical Systems (agendaCPS)*, ser. acatech STUDY. Munich, Germany: Herbert Utz Verlag, 2014.
[2] B. Schaetz, "The role of models in engineering of cyber-physical systems – challenges and possibilities," in *CPS20: CPS 20 years from now - visions and challenges*, ser. CPS Week, 2014.
[3] B. Vogel-Heuser and J. Prieler, "Evaluation of selected metrics for flexibility of cyber physical production systems," in *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, 2017, pp. 701–708.
[4] M. Weyrich, M. Klein, J.-P. Schmidt, N. Jazdi, K. D. Bettenhausen, F. Buschmann, C. Rubner, M. Pirker, and K. Wurm, *Evaluation Model for Assessment of Cyber-Physical Production Systems*. Cham: Springer International Publishing, 2017, pp. 169–199. [Online]. Available: https://doi.org/10.1007/978-3-319-42559-7 7 –
[5] M. Challenger, G. Kardas, and B. Tekinerdogan, "A systematic ap- proach to evaluating domain-specific modeling language environments for multi-agent systems," *Software Qual J*, vol. 24, pp. 755–795, 2016.

[6] F. D. Giraldo, S. Espan˜a, O´ scar Pastor, and W. J. Giraldo, “Considerations about quality in model-driven engineering,” *Software Qual J*, vol. 26, p. 685–750, 2018.

[7] K. Lochmann and A. Goeb, “A unifying model for software quality,” ser. WoSQ ’11. New York, NY, USA: Association for Computing Machinery, 2011, p. 3–10. [Online]. Available: https://doi.org/10.1145/2024587.2024591

[8] A. Badii and D. L. Fuschi, “User-Intimate Requirements Hierarchy Resolution Framework (UI-REF) in work-flow design for 3D media pro- duction & distribution,” in *4th International Conference on Automated Solutions for Cross Media Content and Multi-Channel Distribution (AXMEDIS)*, 11 2008.

[9] A. Badii, D. Fuschi, A. Khan, and A. Adetoye, “Accessibility-by-design: A framework for delivery-context-aware personalised media content re- purposing,” in *HCI and Usability for e-Inclusion*, A. Holzinger and K. Miesenberger, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 209–226.

[10] S. Naumann, E. Kern, M. Dick, and T. Johann, “Sustainable software engineering: Process and quality models, life cycle, and social aspects,” in *ICT Innovations for Sustainability*, L. M. Hilty and B. Aebischer, Eds. Cham: Springer International Publishing, 2015, pp. 191–205.

[11] M. Wolski, B. Walter, S. Kupin´ski, and J. Chojnacki, “Software quality model for a research-driven organization—an experience report,” *Journal of Software: Evolution and Process*, vol. 30, no. 5, p. e1911, 2018. [Online]. Available: https://onlinelibrary.wiley.com/doi/ abs/10.1002/smr.1911

[12] P. Nistala, K. V. Nori, and R. Reddy, “Software quality models: A systematic mapping study,” in 2019 IEEE/ACM International Conference on Software and System Processes (ICSSP), 2019, pp. 125–134.

[13] A. Badii, A. Khan, R. Raval, H. Oudi, R. Ayora, W. Khan,

[14] A. Jaidi, and N. Viswanathan, “Situation assessment through multi- modal sensing of dynamic environments to support cognitive robot control,” Facta Universitatis, Series: Mechanical Engineering, vol. 12, no. 3, pp. 251–260, 2014. [Online]. Available: http://casopisi.junis.ni.ac.rs/index.php/FUMechEng/article/view/583

[15] G. Sabaliauskaite and A. P. Mathur, “Aligning cyber-physical system safety and security,” in Complex Systems Design & Management Asia, M.-A. Cardin, D. Krob, P. C. Lui, Y. H. Tan, and K. Wood, Eds. Cham: Springer International Publishing, 2015, pp. 41–53.

[16] I. Kunold, H. Wo¨hrle, M. Kuller, N. Karaoglan, F. Kohlmorgen, and

[17] J. Bauer, “Semantic interoperability in cyber-physical systems,” in 2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), vol. 2, 2019, pp. 797–801.

[18] S. Karnouskos, L. Ribeiro, P. Leita˜o, A. Lu¨der, and B. Vogel-Heuser, “Key directions for industrial agent based cyber-physical production systems,” in 2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS), 2019, pp. 17–22.

[19] “Performance Metrics in Machine Learning,” https://towardsdatascience.com/performance- metrics-in-machine- learning-part-3-clustering-d69550662dc6, 2021, accessed: 2023-01-07.

[20] J. C. Dunn, “Well-separated clusters and optimal fuzzy partitions,”Journal of Cybernetics, vol. 4, no. 1, pp. 95–104, 1974.

[21] P. Warden and D. Situnayake, TinyML. USA: O’Reilly Media, Inc., 2019.

[22] S. Kelly and J.-P. Tolvanen, Domain-Specific Modeling: Enabling Full Code Generation, 1st ed. Wiley, 2008.

[23] F. Santos, I. Nunes, and A. L. Bazzan, “Quantitatively assessing the benefits of model-driven development in agent-based modeling and simulation,” Simulation Modelling Practice and Theory, vol. 104,

[24] p. 102126, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1569190X20300654

[25] F. Fouquet, B. Morin, F. Fleurey, O. Barais, N. Plouzeau, and J.-M. Jezequel, “A dynamic component model for cyber physical systems,” in Proc. of the 15th ACM SIGSOFT Symp. on Component Based Software Engineering. NY, USA: ACM, 2012, p. 135–144.

****