Alternative Views Regarding Digital Transformation and Requirements Engineering in Embedded and Cyberphysical Systems

DIGITAL TRANSFORMATION (OR simply digitization) is about adopting disruptive technologies to increase productivity, value creation, and social welfare.[1] Digitization is perceived as an increasingly important theme by the software engineering community and certainly deserves further discussion with researchers and practitioners. In a recent article published in *IEEE Software*,[2] Weyer, Daun, and Tenbergen present their views regarding how digital transformation changes requirements engineering in the embedded and cyberphysical domains. They argue that such systems must be functionally safe (e.g., due to correctness) and operate safely, and they mention cybersecurity threats as possible causes of unsafe behavior. Indeed, correctness is a safety premise, but security and privacy are additional requirements that must be embedded and that cyberphysical systems need to satisfy in digitization projects.

The authors state that a specification S is a model for a requirement R, given an environment E; formally, $E, S \vdash R$. However, S can be regarded as a model for R only if a model-based specification approach is adopted. Algebraic and axiomatic techniques are equally suited to digitization and requirements engineering, due to their high abstraction and definitional capabilities. These alternative approaches are often used to capture domain knowledge, elicit customer wishes and needs, and formally represent them with precision.

Concerning requirements implementation, Weyer, Daun, and Tenbergen assert that a program P implements a specification S only if, whenever P is executed on a platform M, P fulfills S. In this case, the system (P, M) satisfies each requirement R supported by S in the context of E; formally $M, P \vdash S$. This formulation does not consider linguistic aspects, but in algebraic and axiomatic settings,[3] E, S, and R must be written in the same language. On the other hand, P and M can be defined in different languages, and even the symbols in S and P denoting the same real-world dimensions can be distinct. In digitization processes, in which lean, or agile, methods are frequently used, requirement and software engineers more and more need to work concurrently and continuously together. They have different cultures and accordingly demand the freedom to adopt the idioms that best suit their needs. Neglecting linguistic aspects also hinders reusability, traceability, and change management, which are prevalent concerns in requirements engineering.

Regarding the connections between design and implementation, M can be considered a realization of E and P a realization of S. In case specifications and programs rely on the same linguistic support, we can write $M, P \Vdash R$ for each requirement R supported by S. Still, the development process may be gradual and have intermediate steps.[3] This possibility reminds us to make more explicit and rigorous the respective development transitions, for example, by adopting logical systems[4] connected through formal language mappings.[5] This approach enforces validity and correctness and enables

Digital Object Identifier 10.1109/MS.2021.3137431
Date of current version: 18 April 2022

effective computer-aided assistance throughout the process.

The elucidation of development steps and the distinction between specification and program satisfaction is crucial because some specifications are not realizable due to the impossibility to satisfy temporal and reliability constraints.[6] Realizability is a vital context-dependent property in embedded and cyberphysical systems, as they aim to solve real problems by relying on hardware and mixed interfaces, e.g., for real-time and feedback control. Realizability is also a relevant concern of requirement engineers, due to their productivity, affordability, and time-to-market commitments, particularly in digital transformation projects in which they are expected to deliver market-disrupting solutions.

The context assumptions and functional/quality guarantees of embedded and cyberphysical systems are related to their open and dynamic nature. However, they do not always lead to an automated system's composability with its environment, which possibly contains elements with concurrent reactive behavior, such as connected Internet of Things devices. In case we adopt a specification discipline that prescribes dynamic interfaces with a few context assumptions and functional/quality commitments,[7] it is possible to ensure the compositionality of designs and the verifiability of emerging progress properties. By strategy, requirement engineers usually formulate broad environment assumptions and minimum viable system obligations to avoid frequent specifications changes.

In real-world digitization projects, change is the rule. That is why Weyer, Daun, and Tenbergen go beyond the concerns with openness and dynamic behavior by arguing that embedded and cyberphysical systems should be adaptive. Adaptability is achieved by detecting changes in requirements and assumptions made about the real world, thus reacting accordingly. So, each system must keep a model of its own requirements and environment in this scenario, apart from implementing runtime deductive capabilities, which are also required to support self-learning. We can represent the internalized deductive capability by $\Vvdash$ to avoid ambiguity with $\Vdash$ and $\vdash$, symbols adopted in the design and implementation stages. Runtime deduction and introspection are important, say, to preserve, adapt, and optimize system operation in the presence of changes.[5]

Digitization has not inspired radically new software technologies. Instead, it has given rise to new software technology applications, owing to the additional requirements that must be satisfied.[1] In view of the requirements of safety, security, and privacy; linguistic and logical rigor; realizability; composability; and reasoning, the complex processes observed in the development of embedded and cyberphysical systems are challenging to engineers. Due to the identified technical challenges, requirement engineers' skills and capabilities are possibly more important than the underlying methods, techniques, and tools. The human factor is critical because the required competencies for dealing with high abstraction levels, problem solving, and managing complexity are often lacking or insufficient. This situation highlights the need for knowledge transfer and improved education, training, and engagement.

Consequently, it is important to salute Weyer, Daun, and Tenbergen for presenting their extensive views[2] so that we can contrast them here with our own experiences. Indeed, requirement engineers must balance the needs of change and rigor, particularly in connection to software development. In the multifaceted world embraced by digital transformation, their distinct viewpoints, perspectives, and opinions are most welcome.

Carlos Henrique C. Duarte
Rio de Janeiro, Brazil

References

1. C. Ebert and C. H. C. Duarte, "Digital transformation," *IEEE Softw.*, vol. 35, no. 4, pp. 16–21, Jul. 2018, doi: 10.1109/MS.2018.2801537.
2. T. Weyer, M. Daun, and B. Tenbergen, "The changing world and the adapting machine," *IEEE Softw.*, vol. 38, no. 5, pp. 83–91, Sep. 2021, doi: 10.1109/MS.2020.3036287.
3. T. Maibaum and W. Turski, "On what exactly is going on when software is developed step-by-step," in *Proc. 7th Int. Conf. Softw. Eng., ICSE*, IEEE Press, 1984, pp 525–533, doi: 10.5555/800054.802014.
4. J. Meseguer, "General logics," *Stud. Logic Found. Math.*, vol. 129, pp. 275–329, Apr. 2000, doi: 10.1016/S0049-237X(08)70132-0.
5. D. Bouskela *et al.*, "Formal requirements modelling for cyber-physical systems," *Requirements Eng.*, Aug. 2021, doi: 10.1007/s00766-021-00359-z.
6. M. Abadi, L. Lamport, and P. Wolper, "Realizable and unrealizable specifications of reactive systems," in *Proc. 16th Int. Colloquium Automata, Lang. Program. (ICAPL)*, Springer-Verlag, 1989, vol. 372, pp. 1–17.
7. C. H. C. Duarte and T. Maibaum, "A rely-guarantee discipline for open distributed systems design," *Inform. Process. Lett.*, vol. 74, nos. 1–2, pp. 55–63, Apr. 2000, doi: 10.1016/S0020-0190(00)00038-7.

Response From the Authors of "The Changing World and the Adapting Machine" to the Reply From Carlos Henrique C. Duarte

We thank the author for the remarks on our article.[2] We are pleased that our work encourages others to reflect on necessary changes to the job profile of requirements engineers in the development of embedded and cyberphysical systems driven by digital transformation. The remarks regarding extensions of our work's theoretical background seem reasonable yet require more detailed elaboration. We do not share the author's opinion that the theoretical background we use and our deliberations thereof can be applied only in model-based specification. Possibly, the reason for this divergent position lies in a different understanding of the term *model based*. We agree that in the course of the digital transformation, agile approaches and related techniques, such as continuous delivery, have also changed and continue to change the jobs of requirements engineers.

Although we briefly mention these points, they are not the focal subjects of our article. Instead, we show, on the basis of examples from the autonomous driving field, in what way the skills of requirements engineers must develop in directions beyond agility. This includes formal modeling of highly dynamic, partially uncertain, or unknown operating context and formal requirements specification. These skills enable the development of systems that can automatically check the validity of their runtime behavior, e.g., when the context changes, and consequently be able to evaluate available adaptations and reason about new knowledge learned during operation.

Thorsten Weyer, Marian Daun, and Bastian Tenbergen

Digital Object Identifier 10.1109/MS.2022.3162506