Emilia Cioroica, Fraunhofer IESE
Thomas Kuhn, Fraunhofer IESE
Dimitar Dimitrov, Siemens AG

14

Supporting the Creation of Digital Twins for CESs

One important behavioral aspect of collaborative embedded systems (CESs) is their trustworthiness, which can be assessed at runtime by evaluating their software and system components virtually. The key idea behind trust evaluation at runtime is the assessment of system interactions and consideration of an extended set of actors that influence the dynamicity of these systems. In this sense, the behavior of collaborative embedded systems and collaborative system groups (CSGs) is part of a more complex behavior of digital ecosystems that form around the collaborating systems. One way of performing runtime virtual evaluation of such complex behavior is through the implementation of digital twins (DTs). DTs are executable models fed with real-time data that allow behavior to be observed and analyzed in concrete technical situations. The use of digital twins enables goals to be evaluated in holistic scenarios at three different levels: strategic level, tactical level, and operational level, as we present in this chapter.

14.1 Introduction

By considering the actors that interact directly and indirectly with collaborative embedded systems (CESs), the concept of collaborative embedded systems and collaborative system groups (CSGs) extends towards the notion of *digital ecosystems*. Within an ecosystem, actors such as organizations, developers, and users have a multitude of goals, and may act not only in cooperation but also in competition. These dynamics influence the behavior of CESs within CSGs directly and indirectly.

*Collaborative systems
are part of complex
ecosystems*

In [Cioroica et al. 2019], we have defined trust-based digital ecosystems where the trustworthiness of a collaborator is computed rather than being granted by default. In the assessment of a digital ecosystem from the trust perspective, a *trustor* is the user of a service who can trust a *trustee*, who is the provider of the service, to satisfy its needs and expectations linked to a *trustum*, which is the service provided. Consider an example at the level of collaborating systems in the automotive domain: a following vehicle (trustor) uses the coordination commands (trustum) to adapt the speed of a lead vehicle in a platoon (trustee). Similarly, a vehicle that intends to join a platoon (trustor) uses the goal information communicated (communication service is the trustum) by the platoon leader (trustee) to make its decision. The architectural model presented in this chapter supports the creation of digital twins for holistic trust evaluation.

Trust results from reputation computed in multiple verification scenarios. From a safety perspective, the reputation of the leading vehicle must be evaluated to ensure trust in the ecosystem that is built around the platoon. In the model that we introduce in this paper, the quality of service (QoS) provided by a product has an impact on the health of the ecosystem. According to [da Silva et al. 2017], the health of an ecosystem is linked to how well the business develops. For example, wrong or delayed commands lead to string instability within a platoon. String instability is characterized by sudden braking and acceleration, which in turn create an increase in fuel consumption instead of a reduction (business goal). This impact is analyzed by providing a structural hierarchy of the relationships between the quality of service and the business goals of the actors. The computation of trust in a collaborator starts with the evaluation of the *operational goals* of the system. The results are used to evaluate the *strategic goals* of the ecosystem that can be achieved by CESs. If we

return to the context of forming a vehicle platoon, where a system function sends context information that is inaccurate or even intentionally wrong, then the *tactical goal* of the CESs to form an effective vehicle platoon will not be achieved. This has an impact on the *strategic goal* of reducing fuel consumption, with direct impact for the participants in a CSG. However, if this type of behavior is discovered early enough, the vehicle providing the malicious service will not be granted access to the ecosystem. Therefore, successful evaluation of strategic goals relies on proper evaluation of the tactical goals, which in turn relies on the evaluation of operational goals for every system engaged in a collaboration. The hierarchical nature of decision-making based on the main differences and distinctions between three types of decisions—namely strategic decisions, tactical decisions, and operational decisions—is described in [Hollnagel et al. 2003] and [Molen et al. 1988]. In our reference architecture, we use a similar hierarchy to structure the goals within an ecosystem by considering systems, system components, and actors.

14.2 Building Trust through Digital Twin Evaluation

Given the distributed provision of hardware resources and software components, the formation of collaborative systems through runtime activation of system functions requires a runtime evaluation of the hardware–software interaction as well. For this particular situation, [Seaborn and Dullien 2015] have shown that specific hardware–software interaction patterns may be faulty and may lead to serious system failures that manifest into security threats. This would be disastrous for CESs and implicitly for the health of the ecosystem formed around the CSG. A runtime assessment and evaluation of the level of trust in the components of a CES is therefore required.

A novel approach to building trust in a software component without executing its behavior in real operation is by evaluating its digital twin at runtime. We have introduced such an approach in [Cioroica et al. 2019]. In the early days of autonomous computing systems, reputation was seen as a good indicator of the level of trust in a system. The authors of [Kephart et al. 2003] propose storing information about a system’s reputation in order to address the need to compute the trustworthiness of potential collaborators. The notion of a digital twin (DT) was initially introduced by NASA [Shafto et al. 2012] as a realistic digital representation of a flying object used in laboratory testing activities. Since then, the notion of DT has also been

Fulfilment of strategic goals of a collaboration relies on the fulfilment of tactical goals which, in turn, relies on the correct implementation of operational goals

adopted in the emerging Industry 4.0 [Rosen et al. 2015] to represent the status of production devices and to enable the forecast of the impacts of change. The reference architecture presented in this chapter enables the creation of digital twins for the whole ecosystem. The digital twin provides a machine-readable representation of the goals of the entities that are part of the ecosystem and supports their trust evaluation through execution of verification scenarios that reflect their dynamic behavior.

Digital twins provide a machine-readable representation of goals

Figure 14-1 depicts an example of a basic classification of system goal types in the supertype-subtype hierarchy. The goals depicted in boxes with a dashed line represent default types that can be reused in any domain. The goals depicted in boxes with a continuous line represent extensions for a specific domain. This classification is supported by evidence showing that, besides its declared well-intended contributions to a collaboration, a system can also have contributions with malicious intent. These intentions can be exposed through malicious behavior caused by malicious faults [Avizienis et al. 2004]. The malicious behavior of a system represents the undeclared competing goals of actors introducing systems and system components on the market.

Fig. 14-1: Classification of Goal Types

The creation of digital twins of the ecosystem and ecosystem participants is enabled by an architecture that contains a description of goals and provides support for the reputation computation in specific verification scenarios. The scenarios describe concrete technical situations in which decisions need to be taken — for example, joining or leaving a platoon. The digital twin of an ecosystem enables information access at runtime and supports a CES in making the decision of whether or not to join a specific platoon. In Figure 14-2, we depict the ecosystem perspective on CESSs. CESSs and CSGs exist within digital ecosystems. In literature, there are two types of digital ecosystems: software ecosystems, formed around software products [Manikas et al. 2013], and smart ecosystems, formed around cyber-physical systems, such as automotive smart ecosystems [Cioroai et al. 2018]. Within an ecosystem, actors can play different roles, such as

manufacturer, distributor, user, subcontractor, etc. and can have a multitude of goals of various types — for example, collaboration, competition, increase in revenue, etc. The system behavior is the asset that enables goal satisfaction.

Fig. 14-2: Ecosystem Perspective on CES

Figure 14-3 and Figure 14-4 show the instantiation of the architecture that enables the creation of digital twins in the automotive and smart grids domains. Given its context-specific operational capacity, an embedded system by itself is meant to operate to achieve dedicated business goals.

Fig. 14-3: Example instantiation in the automotive domain

However, through communication and collaboration with other embedded systems, enhanced functionality can be achieved. Depending on the goal and the role an actor has in an ecosystem, other actors are targeted. The operative part of an ecosystem is formed by the systems that collaborate with each other at runtime in order to fulfill enhanced business goals of different organizations or the same organization. Communication and collaboration are realized through an exchange of data and functions and can be between embedded systems located in the same system or embedded systems located in different systems. In the first case, communication is realized through dedicated communication buses; in the latter case, the communication between embedded systems is realized through Internet communication.

The collaborative goals of systems are influenced by business goals, which in turn depend on risks, such as economic risks. The concepts of risks and goals are related to actors that play certain roles in a collaboration. In Figure 14-5, examples of roles are the user and the provider. A holistic evaluation of embedded systems that collaborate in the field must take all these aspects into account. Figure 14-6 presents an instantiation in the automotive domain.

Fig. 14-4: Example instantiation in the smart grid domain

Fig. 14-5: Business perspective on Collaborative Systems

Fig. 14-6: Instantiation of the business view in the automotive domain

Figure 14-7 depicts the evaluation of trust through computation of reputation. From the point of view of a collaborator, a system is a resource with functional behavior and non-functional properties. Through its behavior or through the service it is providing, the resource influences the reputation of an Original equipment Manufacturer (OEM) that introduces the system on the market. A reputation of a component is a combination of the initial reputation of an OEM, calculated when the system is first introduced on the market, and the runtime reputation computed via a series of algorithms. The computation of runtime reputation is linked to verification scenarios that describe the context in which the resources are evaluated. Verification scenarios are linked to functional and non-functional requirements that reflect the expectations of the user for the system behavior.

Requirements are provided by the users of services. Based on requirements, verification scenarios are defined in order to evaluate the reputation of resources. The resources provided reflect the goals of the actors during collaboration with other actors. This kind of verification scenario can, for example, evaluate individual goals of vehicles wanting to join platoons for compatibility. Only the vehicles that have compatible routes are granted access to the platoon, and implicitly to the ecosystem. Other verification scenarios can evaluate the expectations with regard to the exchange of services. If, for example, a vehicle requests exchange of information every 100 ms, it should avoid joining a group of vehicles that exchange information every 100 ms. If the internal system functions of a vehicle are activated and checked every 200 ms, joining a platoon that requests information exchange every 100 ms may cause synchronization issues.

Fig. 14-7: Computation of trust based on runtime verification scenarios

14.2.1 Demonstration

In this section, we present scenarios from the automotive and smart grid domains that benefit from the instantiation of digital twins of their ecosystems based on the reference architecture introduced in the previous section.

Automotive Smart Ecosystems

At the entry point for a highway, consider a scenario in which a vehicle (CES) activates a collaboration function (SW component) and corresponding running specifications which are digital twins. The collaboration function enables the vehicle to join or form vehicle platoons (CSG). If the vehicle starts forming a vehicle platoon, it becomes the leader of that platoon (it has the role type “Platoon Leader”). The other vehicles with the same goal (collaborative goal) can be members of the same platoon (assigned the role type “Member”). Besides having the same collaborative goal, the vehicles must have fitting individual goals in order to join beneficial collaborations. For example, only vehicles with the same collaborative goal of being part of vehicle platoons and moving towards similar destinations (Reaching Destination Goal as a subtype of Individual Goal) may be part of the same platoon. When another vehicle approaches an existing platoon, it requests the digital twin of the ecosystem containing the platoon and checks whether its goals fit the goals within the ecosystem. If, for example, the vehicle approaching the platoon has the goal of reaching a destination that is not compatible with the route of the platoon, then it will not join this particular platoon. The collaboration function part of the ecosystem

operates on an ECU (embedded control unit, which is an embedded system). It reads context information such as speed and distance communicated by the vehicle in front. According to our architecture, the process of information reading is an operational goal, which is enabled by the *context information reading* service. The collaboration function sends this information to system functions. The process of sending the information is a *data transfer* service. The system functions are responsible for maintaining the maximum distance between vehicles in a platoon while maintaining the minimum safe distance. According to our architecture, the process of managing the distance is a service associated with the smart agent, via a service assignment with the role type “Provider.” The service has a contract of the contract type “Specification.” The maximum distance is the distance that allows the platoon members to benefit from reduced air friction and implicit reduction of fuel consumption (strategic goal).

If the information provided by a system function is wrong—if, for example, the vehicle in front transmits that the distance is 7 m, but the actual distance is 5 m—then the system function might accelerate. According to our architecture, the acceleration is an operational goal. The vehicle can accelerate until it learns from its own sensors that the minimum safety distance has been violated, and then it will brake immediately. Acceleration followed by instant braking creates string instability in the platoon and implicit higher fuel consumption. In the worst case, this could cause a crash. By using a digital twin of the digital ecosystem instantiated with our reference architecture, a violation event will be recognized before it actually happens. Specifically, the reputation score of the vehicle causing a violation and its associated actors will become negative (based on the output of the reputation computation that compares the observations of the distance properties with the contract). As a result, the vehicle will not be granted access to this ecosystem.

By capturing the system decomposition, our reference architecture forms the basis for instantiating a digital twin of the ecosystem. This allows the identification of failure cases at the system level, thus supporting the replacement of faulty or malicious components. A system function that does not perform according to its specifications can be replaced with an improved version of itself or another system function provided by another organization that is an actor in the ecosystem. A digital ecosystem has specific sets of verification scenarios that compute the reputation of its participants. If the requirements and expectations of the verification scenarios and

of a vehicle that wants to join the ecosystem are compatible, the vehicle is granted access to the ecosystem and it can decide to join it.

Smart Grids

In a smart grid, power can be generated by a large variety of decentralized energy resources (DERs), such as wind turbines or photovoltaic plants, each providing a small fraction of the energy. By integrating a connector box (CES) on a DER, the DER is capable of joining (tactical goal) a virtual power plant (VPP) (digital ecosystem) to sell the energy produced (VPP associated with the business goal). Through the deployment of collaboration functions, connector boxes can become fully autonomous and form coalitions (CSGs) in order to provide flexible quantities of energy (strategic goal) when requested by a distributed system operator (actor assigned by actor assignment to the CSG with the role type “Customer”). When no flexibility of energy production is achieved, sanctions are applied in the form of shutdown of the DER (risk associated with the strategic goal of providing a flexible quantity of energy). Therefore, when a member of a coalition cannot fulfill its commitment, a replacement must be found (tactical goal). In order to find the right replacement, the connector boxes must communicate accurate information about their state (operational goal enabled by broadcasting information regarding its status service). The connector boxes must send their status at least once every 15 minutes (specification of the property of the “status broadcast frequency” property type in the contract of the “status broadcast” service). For example, if one connector box does not communicate its status or does not communicate its status correctly, a broadcast for bids cannot start and the flexibility for providing energy will not be achieved.

When a smart agent inside a connector box wants to take part in a collaboration, it must compute the level of trust in the ecosystem that forms around the collaborating systems. This can be achieved by querying the digital twin of the ecosystem, which provides information about the goals of different DERs together with their behavior evaluation in various verification scenarios and their associated reputations.

14.3 Conclusion

In this chapter, we presented a reference architecture that enables automatic computation of trust in ecosystems and ecosystem

components. The reference architecture captures the main concepts and relationships within an ecosystem and can be used to instantiate digital twins. The reference architecture was developed to be flexible and be customizable in various application domains. We showed the expressiveness and reusability of the architecture by providing examples of its instantiation in scenarios from both the automotive and energy domains. Currently, the reference architecture provides the high-level logical view of ecosystems. In future work, we aim to extend the reference architecture with additional views such as the following: a use case view to capture the key usage scenarios; an interaction view to explicitly model the processes and interactions within the domain; and a deployment view to capture the implementation decisions for systems based on the reference architecture. Additionally, because trust evaluation requires detailed analysis of goals, ongoing work is directed towards detailing the goal classification for trust computation.

14.4 Literature

- [Avizienis et al. 2004] A. Avizienis, J. C. Laprie, B. Randell, C. Landwehr: Basic Concepts and Taxonomy of Dependable and Secure Computing. In: IEEE transactions on dependable and secure computing, Vol.1, 2004, pp. 11–33.
- [Cioroica et al. 2018] E. Cioroica, T. Kuhn, T. Bauer: Prototyping Automotive Smart Ecosystems. In: 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W). IEEE, 2018, pp 255–262.
- [Cioroica et al. 2019] E. Cioroica, S. Chren, B. Buhnova, T. Kuhn, D. Dimitrov: Towards Creation of a Reference Architecture for Trust-Based Digital Ecosystems. In: Proceedings of the 13th European Conference on Software Architecture - Volume 2. ACM, 2019, pp. 273–276.
- [Cioroica et al. 2019] E. Cioroica, T. Kuhn, B. Buhnova: (Do Not) Trust in Ecosystems, In: Proceedings of the 41st International Conference on Software Engineering, 2019.
- [da Silva et al. 2017] S. da Silva Amorim, F. S. S. Neto, J. D. McGregor, E. S. de Almeida, C. von Flach G Chavez: How Has the Health of Software Ecosystems Been Evaluated?: A Systematic Review. In: Proceedings of the 31st Brazilian Symposium on Software Engineering. ACM, 2017, pp. 14–23.
- [Hollnagel et al. 2003] E. Hollnagel, A. N’abo, I. V. Lau: A Systemic Model for Driver-in-Control, 2003.
- [Kephart et al. 2003] J.O. Kephart, D.M. Chess: The Vision of Autonomic Computing. In: Computer 2003, pp 41–50.
- [Manikas et al. 2013] K. Manikas, K. M. Hansen: Software Ecosystems – A Systematic Literature Review. In: Journal of Systems and Software Vol. 5, 2013, pp: 1294–1306.
- [Molen et al. 1988] H. H. Van der Molen, A. M. Bötticher: A Hierarchical Risk Model for Traffic Participants. In: Ergonomics, vol. 31, no. 4, 1988, pp. 537–555.

- [Rosen et al. 2015] R. Rosen, G. von Wichert, G. Lo, K. D. Bettenhausen: About the Importance of Autonomy and Digital Twins for the Future of Manufacturing. In: IFAC-PapersOnLine48, Vol.3, 2015, pp. 567–572.
- [Seaborn and Dullien, 2015] M. Seaborn, T. Dullien: Exploiting the DRAM Rowhammer Bug to Gain Kernel Privileges, Black Hat15, 2015.
- [Shafto et al. 2012] M. Shafto, M. Conroy, R. Doyle, E. Glaessgen, C. Kemp, J. LeMoigne, L. Wang: Modeling, Simulation, Information Technology and Processing Roadmap. In: National Aeronautics and Space Administration, 2012.

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.