On Digital Twins for Cloud Continuum Applications

Luiz F. Bittencourt[1][0000−0001−6305−9059],
Kelly R. Braghetto[2][0000−0001−6218−6849], Daniel Cordeiro[2][0000−0003−4971−7355],
and Rizos Sakellariou[3][0000−0002−6104−6649]

[1] Instituto de Computação – Universidade Estadual de Campinas, Brazil
bit@ic.unicamp.br
[2] Universidade de São Paulo, Brazil
{kellyrb, daniel.cordeiro}@usp.br
[3] The University of Manchester, UK
rizos@manchester.ac.uk

Abstract. Digital Twins (DTs) are digital representations of physical objects or processes that can be used for their computer-based analysis. This technique has been used in different fields to analyze, simulate, and optimize various scenarios in real time without interfering with the real twin. Using Serverless Computing as a use case, this paper discusses the underlying costs of using DTs when applied to analyze and optimize the management of computational resources for cloud-continuum applications. We argue that, although feasible, using DTs for Digital Systems can be prohibitively expensive. Defining the ideal DT fidelity for a given application is challenging, as this impacts both system management and performance through heavy monitoring as well as the DT running costs.

Keywords: Digital Twins · Cloud Continuum · Resource Management.

1 Introduction

Creating Digital Twins (DTs) of *physical systems* is a complex task that requires in-depth modeling and mapping of real-world variables and processes into digital variables and digital representation of processes. On the other hand, creating DTs of *computer systems* is straightforward: simply duplicate the whole system (infrastructure, management system, and applications) as well as all requests and tasks, and the DT is ready to use. However, the costs of duplicating a complete digital system can be prohibitively high in many cases.

According to Ahlgren et al. [1], recent developments in DTs have focused primarily on cyber-physical systems, where simulations interact with real-world physical systems. However, there is significant untapped potential in cyber-cyber DTs, which offer complete malleability, meaning that any aspect of the simulated software system can be changed automatically in response to the simulation. This creates a different paradigm, as the distinction between the simulator and the simulated system can blur, with either being able to inform and adapt the other.

Useful implementations of DTs for digital systems should bring high fidelity and real-time reproduction of the real system to support what-if scenarios and uncertainty mitigation. Modeling digital systems into twins with fundamental variables and mathematical models representing processes is challenging. Techniques from computer simulation and emulation can be used, as they help understand the general behavior of computer systems, but DTs should be one step ahead in terms of fidelity and real-time mimicking of a real twin using appropriate inputs monitored from the real system. Thus, the set of variables, their relevance, and the formulation of models that are able to generate a DT of the real digital system should be carefully chosen to enable a cost-efficient yet representative twinning process.

The complexity of building digital twins for digital systems does not come from the modeling and understanding of physical phenomena, but from understanding how simplified the original digital system can be represented by a DT such that uncertainties and costs of the DT do not surpass or are supplanted by the potential gains in resource management to fulfill application requirements. In this sense, we discuss in this paper challenges that must be considered when modeling, building, and running DTs for digital systems focusing on a resource management perspective. To illustrate these challenges, we describe a use case of DTs as a resource manager for serverless computing and applications implemented using the Function as a Service (FaaS) paradigm considering a computing continuum environment. We also discuss research challenges to be addressed to make DTs feasible in this context.

2 Concepts

2.1 Digital Twins

Digital Twins (DTs) were introduced in the early 2000s by Michael Grieves in a product lifecycle management course. Initially considered complex to implement, DTs gained traction with advancements in cloud computing, IoT, and big data, extending from the aerospace industry to manufacturing around 2012. Industry 4.0 developments, data digitization, and embedded sensors were key to this evolution, making realistic virtual testing feasible [10].

Despite the large amount of scientific literature around the theme, there is no consensual definition of a DT and its components. However, recent standardization efforts, including a new ISO standard (ISO/IEC 30173:2023, "Digital twin — Concepts and terminology"), may help address this issue. For the purposes of this work, DTs can be understood as *digital representations of physical objects or processes used for computer-based analysis of their properties.*

Analyzing digital twins typically requires significant computational capacity, which may include diverse computing resources. In a nutshell, the DT's virtual, computer-based representations are fed with data from multiple sensors, and they demand high processing power and rapid networking data transfers to be able to digitally mimic physical objects and their dynamic processes. Therefore,

managing DTs execution in a (potentially distributed) computing system is crucial to achieving acceptable performance that enables timely visualization of the DTs physical objects, processes, and their potential outcomes [2].

The main benefit of having digital twins representing physical systems is to allow simulations, analyses, and optimizations in real time without interfering with the real twin. These simulations help to understand scenarios and support decision-making even before actual events occur in the real system. Analyzing the DT's behavior allows following the real twin more closely to identify and mitigate potential problems and inconsistent states that may derive from the current state and as a consequence of different, sometimes unpredictable, factors.

As virtual representations of physical objects and/or processes, DTs should be able to allow real-time analytics and corrective actions. As such, DTs can be modeled as self-interested agents acting in pursuit of their goals. Multiple types of applications can be represented by DTs to visualize how data input can impact processes and how such impact affects other objects/processes in a chain of digital twins. Therefore, digital twins can be standalone entities, can depend on other digital twins, or can even be part of larger digital twins, organized in a hierarchical structure or interacting in a collaborative model [9].

The level of detail in modeling a physical system for desired outcomes is crucial, as it determines DT's *fidelity*. This fidelity depends on the number, accuracy, and abstraction of parameters exchanged between the real and virtual twins. While high-fidelity models provide detailed and accurate representations, they can be complex, costly, and computationally intensive [6]. Conversely, low-fidelity models are simpler and faster but may fail to capture essential interactions. Balancing fidelity and practicality is essential for a cost-effective DT implementation.

2.2 Resource Management in Serverless Cloud Computing

Managing cloud services presents significant challenges in providing availability, load balancing, auto-scaling, security, and monitoring. These complexities have driven the development of serverless cloud computing. This model abstracts infrastructure management, allowing developers to focus on coding applications without worrying about resource allocation and scaling.

Serverless cloud computing provides both Backend as a Service (BaaS) and Function as a Service (FaaS), with the latter being the most prevalent model. BaaS encompasses services such as storage, messaging, and user management. FaaS allows developers to deploy and execute their code on computing platforms, relying on BaaS services [3]. According to [5], three characteristics mainly differentiate serverless from serverful computing: (i) storage and computation scale independently, with storage provided by a separate cloud service and computation being stateless; (ii) automatic resource allocation, where the cloud automatically provisions the necessary resources for execution of the code provided by users; (iii) billing based on actual resource usage, such as execution time, rather than on the size and number of VMs allocated.

Serverless computing enables developers to break down applications into smaller, manageable functions, allowing for individual scaling of components. On the one hand, this approach can lead to improved efficiency and resource utilization. On the other hand, it introduces challenges related to managing and coordinating numerous functions effectively. The interactions between different functions can be seen as workflows; some providers offer tools to simplify the orchestration of these workflows, such as the AWS Step Functions service [7].

To offer serverless computing in the form of Function as a Service, the cloud provider needs to choose the most suitable computing resource (machine) to run the function code. Also, when functions call each other, forming a workflow, data transfers take place in the computing infrastructure, and delays are introduced between function calls. Moreover, a function in FaaS is often run as a containerized software, which allows a sandboxed environment that can be instantiated on demand and replicated when demand for that function. All these details should be taken into account during the decision-making on where each function should be run in the provider infrastructure, which makes resource management challenging in this context. Digital twins can be used as decision-makers on scaling resources to keep the performance of functions' execution at acceptable quality of service levels.

3 The Case for Digital Twins for Digital Systems

3.1 FaaS characterization and modeling

Let $\mathcal{A} = \{f_1, f_2, ..., f_n\}$ be the set of n serverless functions that compose an application $\mathcal{A}$ designed and implemented considering the FaaS paradigm with an associated Service Level Agreement (SLA). Each function $f_i \in \mathcal{A}$ has an associated invocation pattern p_{f_i}, representing the frequency of calls to f_i in the application and a set of resources utilized per invocation, such as CPU instructions, memory size, and input/output parameters. The SLA of the application contains information about what the user expects, for example, the maximum response time of a submitted request for $\mathcal{A}$.

The service provider should be able to allocate and run each $f_i \in \mathcal{A}$ in the computing continuum infrastructure in such a way that the composed response time of all functions does not violate the application's SLA. Suppose the continuum provider will run all functions of $\mathcal{A}$ in a set $\mathcal{M} = \{m_1, m_2, ..., m_k\}$ of k machines connected by network links $\mathcal{L} = \{l_{m_i,m_j} \mid \forall m_i, m_j \in \mathcal{M}, m_i \neq m_j\}$. Each machine $m_i \in \mathcal{M}$ has an associated computing capacity, modeled as a set of computing characteristics such as CPU, memory, and storage; and each network link $l_{m_i,m_j} \in \mathcal{L}$ connecting machines m_i and m_j has a set of associated network characteristics, such as the available bandwidth and latency. The modeling of each machine and each link can be as detailed as needed to represent the desired *computing model resolution* $\mathcal{R}$. The computing model resolution will impact the digital twin precision and running costs.

Matching the function's requirements with the machine's capacity allows estimating the number of concurrent invocations to a function a given machine

supports without getting overloaded. This kind of estimation is widely used when modeling applications and systems in parallel and distributed computing scheduling algorithms. However, precisely estimating application requirements and machine capacity when running different kinds of workloads in a highly heterogeneous environment as the computing continuum cannot always be achieved. Thus, estimating the actual performance when executing functions is prone to uncertainties, potentially leading to overloaded resources and unpredictable turnaround times if static modeling is used. Using digital twins to understand and manage uncertainties through what-if scenarios can help guarantee functions' performance in the computing continuum.

3.2 Digital twins for serverless computing

A digital twin for managing serverless scenarios will support decision-making on resource allocation to handle requests and keep the quality of service at acceptable levels. Figure 1 overviews the resource management process in a cloud-edge infrastructure that offers Function as a Service through the deployment of containerized services. First, the cloud-edge clients access an application interface to submit requests to be processed in the cloud (step 1). These requests will be fulfilled by running (a set of) containerized functions $\mathcal{A}$, as defined in Section 3.1, which will be invoked by a service composition engine (step 2). Once the functions are invoked, a scheduler should decide which instance of a (potentially) replicated function should actually run the function(s) needed to fulfill the request (step 3). These instances can be deployed in any machine $m_i \in \mathcal{M}$, which can satisfy the functions' resource requirements.

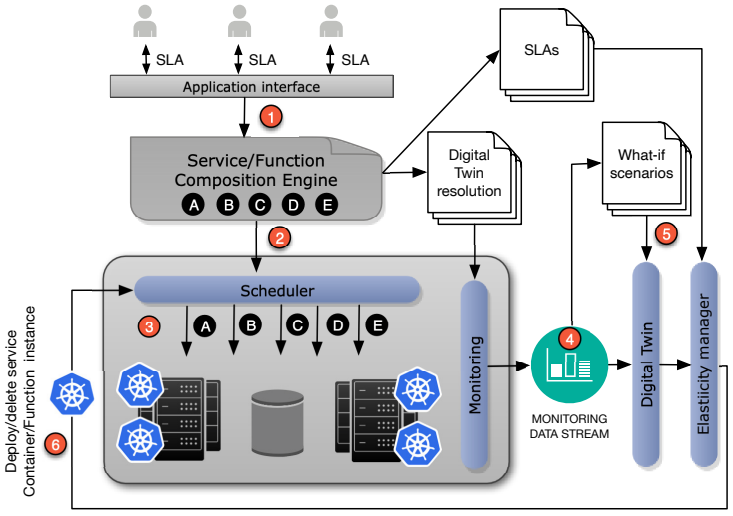

Fig. 1. Digital twins for resource management in FaaS.

One of the main characteristics that differentiate digital twins from standard simulation and emulation is the need for real-time feedback through the adoption of what-if scenarios based on instantaneous information from the monitored system. We define the *digital twin resolution* as the main driver of the digital twin fidelity when running the what-if scenarios of a digital system. The digital twin resolution $\mathcal{R}$ should define the granularity of the monitoring and how frequently the twin evaluation process is performed using the collected data from the real system. In our of FaaS use case, we illustrate this resolution as a configuration derived from the function invocation patterns p_{f_i} and their Service Level Agreements (SLAs), which are used as input in the infrastructure monitoring services that will feed what-if scenarios and the digital twin with a data stream representing the current status of the whole system (step 4). A function invocation pattern p_{f_i} can determine how frequently monitoring is needed at different points in time, and how frequently the twin should be used to assess performance and what-if scenarios to handle potential performance uncertainties. Note that the monitored data (CPU, I/O, memory utilization, application profiling, etc.) and monitoring frequency are a direct result of the defined DT model resolution $\mathcal{R}$, impacting the data stream flow and, consequently, the digital twin precision, computing costs, and delays. Moreover, the monitored system state allows what-if scenarios to be generated and inputted into the digital twin (step 5). Based on the observed DT behavior, an elasticity component will be able to make decisions on increasing/decreasing the amount of computing resources (e.g., replicas) dedicated to a set of functions based on their SLAs (step 6).

3.3 Challenges

In general, modeling and running digital twins has several challenges [10]. In this paper, we focus on challenges related to twinning digital systems for resource management with a holistic approach, taking an illustrative use case of serverless computing implementing FaaS. We are motivated by recent research that presents encouraging results for using DTs in some form of resource management for specific applications [4,11,12,8]. In this section, we highlight the challenges of using DTs for resource management for computing continuum systems.

The Digital Twin resolution defines how precisely the digital twin model represents the twinned digital system and is dependent on the characteristics of the modeled system. The resolution of a DT can be a direct consequence of how the DT of a digital system is modeled (i.e., the model resolution $\mathcal{R}$), e.g., as a set of mathematical representations of the real system or by a full replica of the system running every request as a duplicate. In the first case, the resolution can be determined by the complexity of the modeling utilized to represent the system's processes and also by the amount of monitoring performed in the system and the frequency of the inputs provided to the digital twin. Defining the ideal resolution for a given application is challenging, as this impacts both the system management and performance through heavier monitoring and the digital twin running costs. In addition, dynamically controlling the DT resolution can be effective in handling unexpected scenarios where more frequent analyses are

needed, but this adaptive behavior also brings challenges in terms of modeling and system adaptation to varying DT resolutions.

Automating the definition of what-if scenarios to serve as input for the digital twin is also challenging. A set of pre-defined what-if scenarios can be loaded into the system with the most frequently expected events, such as a workload increase or decrease. In our illustrative FaaS use case, adding and removing service/function replicas can be explored as what-if scenarios. This way, the DT can recommend system configuration changes based on how it would behave, improving applications' QoS. For example, what is the impact on an application if the system reduces the number of replicas of a given function/application component? From a resource management perspective, this answer is important to reduce costs, improve system utilization, and also guarantee QoS, but if this is done in a trial and error approach in production, the system can get overloaded, and the application can be severely impacted until the misbehavior can be detected and corrected.

Even though DTs can help in resource management, managing the DT execution itself can be challenging. Keeping DT response times acceptable to provide timely responses when running (potentially multiple) what-if scenarios concurrently should be attached to the DT's resolution: is it more effective, for a given application/infrastructure, to run multiple what-if scenarios in low DT resolutions, or is it better to focus on a few frequently expected scenarios with higher DT resolutions?

Finally, incorporating the DT costs into business models in a way that improves resource management and the system customers observe concrete advantages can also be challenging. Offering DTs as an improved resource management service can be attractive but challenging to design, implement, and price.

4 Conclusion

DTs can simulate different scenarios and workloads, helping to optimize resource allocation in cloud environments. By monitoring and analyzing how resources such as CPU, memory, or network are being used, DTs can suggest optimal configurations and scaling strategies to improve performance and efficiency. DTs can also facilitate dynamic reconfiguration of computer systems, automatically adapting to varying loads and resource availability, ensuring SLAs are met.

The challenge of employing DTs to manage cloud-continuum resources for executing serverless applications has analogies with DT-assisted dynamic production scheduling in manufacturing, a topic extensively studied [8]. Dynamic production scheduling optimizes resource allocation and task sequencing over time to achieve various goals, such as minimizing makespan, cost, or the number of tardy jobs. It enhances the responsiveness of manufacturing systems by allowing schedule rearrangements to mitigate disruptions.

However, scheduling serverless applications in cloud platforms presents specific challenges. These include large-scale (in terms of both the number of resources and workload), highly heterogeneous resources, varying pricing and busi-

ness models, and real-time requirements. These factors complicate modeling and simulation, increasing both complexity and costs, which can render high-resolution modeling impractical. Yet, the potential for using Digital Twins to optimize resource management in computer systems is vast. However, more research is needed to develop modeling strategies and DT architectures that effectively balance the costs and benefits of DT implementation.

Acknowledgments. Grants #2019/26702-8, #2023/00702-7 (in collaboration with the University of Manchester), and #2024/01115-0, São Paulo Research Foundation (FAPESP), CAPES (grant 88887.954253/2024-00) and CNPq (grants 405940/2022-0 and 421787/2022-8) partially fund this work.

Disclosure of Interests. The authors have no competing interests to declare.

References

1. Ahlgren, J., Bojarczuk, K., Drossopoulou, S., Dvortsova, I., George, J., Gucevska, N., Harman, M., Lomeli, M., Lucas, S.M., Meijer, E., et al.: Facebook's cyber–cyber and cyber–physical digital twins. In: Proceedings of the 25th International Conference on Evaluation and Assessment in Software Engineering. pp. 1–9 (2021)
2. Bellavista, P., Bicocchi, N., Fogli, M., Giannelli, C., Mamei, M., Picone, M.: Exploiting microservices and serverless for digital twins in the cloud-to-edge continuum. Future Generation Computer Systems **157**, 275–287 (2024)
3. Hassan, H.B., Barakat, S.A., Sarhan, Q.I.: Survey on serverless computing. Journal of Cloud Computing **10**, 1–29 (2021)
4. Jeremiah, S.R., Yang, L.T., Park, J.H.: Digital twin-assisted resource allocation framework based on edge collaboration for vehicular edge computing. Future Generation Computer Systems **150**, 243–254 (2024)
5. Jonas, E., Schleier-Smith, J., Sreekanti, V., Tsai, C.C., Khandelwal, A., Pu, Q., Shankar, V., Carreira, J., Krauth, K., Yadwadkar, N., et al.: Cloud programming simplified: A Berkeley view on serverless computing. arXiv preprint arXiv:1902.03383 (2019)
6. Kshetri, N.: The economics of digital twins. Computer **54**(4), 86–90 (2021)
7. McGrath, G., Brenner, P.R.: Serverless computing: Design, implementation, and performance. In: 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW). pp. 405–410. IEEE (2017)
8. Ouahabi, N., Chebak, A., Kamach, O., Laayati, O., Zegrari, M.: Leveraging digital twin into dynamic production scheduling: A review. Robotics and Computer-Integrated Manufacturing **89**, 102778 (2024)
9. Segovia, M., Garcia-Alfaro, J.: Design, modeling and implementation of digital twins. Sensors **22**(14), 5396 (2022)
10. Sharma, A., Kosasih, E., Zhang, J., Brintrup, A., Calinescu, A.: Digital twins: State of the art theory and practice, challenges, and open research questions. Journal of Industrial Information Integration **30**, 100383 (2022)
11. Yu, H., Han, S., Yang, D., Wang, Z., Feng, W.: Job shop scheduling based on digital twin technology: A survey and an intelligent platform. Complexity **2021**(1), 8823273 (2021)
12. Zhao, J., Xiong, X., Chen, Y.: Design and implementation of a cloud-network resource management system based on digital twin. In: 2023 IEEE 3rd International Conference on Digital Twins and Parallel Intelligence (DTPI). pp. 1–5 (2023)