



HAL
open science

Digital twins: properties, software frameworks, and application scenarios

Roberto Minerva, Noel Crespi

► **To cite this version:**

Roberto Minerva, Noel Crespi. Digital twins: properties, software frameworks, and application scenarios. IT Professional, 2021, 23 (1), pp.51 - 55. 10.1109/mitp.2020.2982896 . hal-03210180

HAL Id: hal-03210180

<https://hal.science/hal-03210180>

Submitted on 27 Apr 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Digital Twins: Properties, Software Frameworks, and Application Scenarios

Roberto Minerva and Noël Crespi

Telecom SudParis, Institut Polytechnique de Paris

Abstract—The Digital Twin (DT) is an emerging approach that promises to change the way products and systems are made and used. The DT is attracting increasing interest in the Internet of Things community for its potential applications. Despite the hype, this approach requires a more precise definition and characterization in terms of its properties in relationship to software architectures and their platform implementations, as well as a deeper analysis of its potential applications and actual feasibility in several industries. This paper investigates the basic properties that hold for a DT, sketches a software framework and presents two application scenarios. The paper also addresses the business impact of DT by discussing servitization capabilities.

■ **THE DIGITAL TWIN** (DT) was conceived to design, prototype, and operate a virtual copy of a product.¹ It promotes the softwarization of physical objects (POs), into logical components and the development of applications that exploit programmability of the software representation. Originally used by NASA,² it has gained interest in manufacturing.³ Because of its ability to describe POs by means of logical models and actual data, it is increasingly used in the Internet of Things and cyber-physical

systems.⁴ To fully master its potential, there is a need to understand what a DT is from a software perspective, how to support and implement it, how to manage the physical and logical counterparts, and what constraints are implied by the relationship between them. This article addresses the current DT definition, and identifies a set of characterizing properties. Next, a software framework is sketched out to illustrate two application scenarios that highlight technical possibilities and challenges. An analysis of some business benefits is provided. Conclusions summarize challenges and risks of this approach.

Digital Object Identifier 10.1109/MITP.2020.2982896

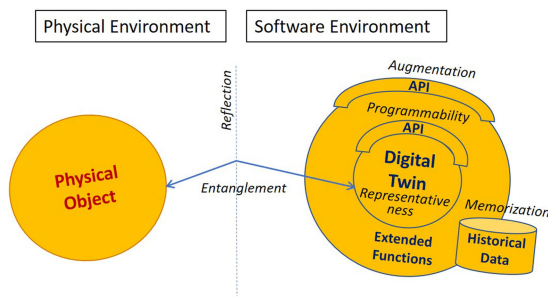


Figure 1. DT representation.

DIGITAL TWIN (DT) AND ITS PROPERTIES

DTs embody software capabilities to describe or simulate relevant features and behavior of POs from design to end-of-life phases.⁵ A DT is a software actionable representation of a PO (or aggregated objects) in a specific environment. Some PO's features could be unrelated to the goals of the software representation. Products or complex objects are the main targets, because mapping one to one atomic objects may be unpractical or useless. DTs in a cyber-physical systems (CPS) domain are characterized⁶ by events they are capable of managing, methods and functionalities they offer, storage capabilities to maintain a historical log of status changes, ability to serve as functional interfaces to applications through application programming interfaces (APIs) and ability to send/receive information. The definition of basic DT properties (see Figure 1) is instrumental to determine requirements and constraints posed to software platforms. The key properties of a DT are given as follows.

- *Representativeness and contextualization.* A DT represents the PO in a specific context, i.e., a DT is a software implementation of a model representing the PO in a specific environment. Feature simplifications or generalizations are possible when they do not affect the behavior in the specific context.
- *Reflection.* A DT mirrors the behavior and the status of the PO. Each change in status, each event faced by the PO is reflected by the DT. Changes that occur to the DT should be reproduced in the PO.
- *Entanglement.* A DT and its PO must be constantly “connected” to instantaneously (i.e., in a time period consistent with requirements

of applications) register any change in status. Entanglement requires an effective, reliable communication between the PO and the DT suitable to the rate of changes.

- *Persistency.* A DT is always available. Its availability exceeds the actual existence of the PO. A DT could be available before the “creation,” during malfunctioning and crashes, and after the end of life of the PO.
- *Memorization.* A DT stores all the status changes and events occurred to the PO. A DT represents the status of the PO over time and space.
- *Augmentation.* A DT can extend the PO functions and offer them by means of APIs. Augmentation can add new functionalities that the PO does not support or provide access to data in particular formats.
- *Composability.* Physical systems are aggregations of subsystems and components, a DT must support the correlation of different elementary DTs into complex organizations and provide views on the aggregated DT and individual components.
- *Replication.* A DT can be replicated to serve the needs of different applications. Replicas of the same PO must behave consistently, i.e., they cannot have a different status and they cannot exhibit different behaviors.
- *Accountability/Manageability.* Each DT must be manageable; it is possible to determine its status and activities and to optimize its execution in the framework in which it is operating. It must provide information about the usage of the PO by the applications associated with it. Self-management of DTs is a desirable property for large systems.
- *Servitization.* A DT enables the transformation of a product/artifact into a set of functionalities offered to users. This capability transforms products into software-controlled and connected entities accessible “on-demand” by users.
- *Predictability.* A DT has the ability to simulate its behavior over time. Specific instances of DTs can simulate the behavior of PO in a context at a specific time (in the past or in the future).

Some DT properties need contextualization: replicas are used for tracking POs or for simulate

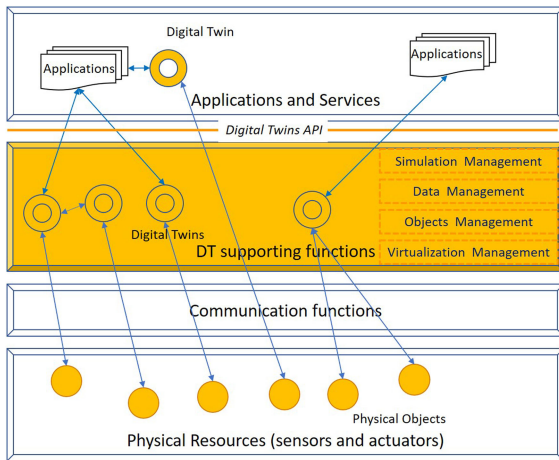


Figure 2. General framework for digital twins.

their behavior. All the replicas of a DT tracking the actual status of a PO should have the same status at the same time t (with time limitations explained for entanglement), DTs used for simulations can diverge depending on the simulation conditions.

TOWARDS A GENERAL SOFTWARE FRAMEWORK

A software framework for DTs implements numerous functionalities, e.g., orchestration of objects, cognition for intelligent optimization, self-management to minimize human intervention. Different software architectures may be applicable.^{7,8} Some adopt layering principles for organizing functionalities.⁹ These architectures must be evaluated to determine how properly they support the DT requirements. Figure 2 represents a generic framework organized in four functional segments. This structure is chosen to illustrate the separation of concerns. IoT devices, communications means (protocols and networks), DT tools and functions will quickly evolve. Layering may prove useful: it is simple enough to be practically used and can allow independent evolution of the different technological layers. Finally, layering is aligned with efforts in standardizing IoT architectures. AIOTI¹⁰ describes an IoT layered architecture that could support the DT approach. However, modularization and linkage of functionalities can prove as practical as layered architectures. Physical resources are devices producing information or

acting on the environment (sensors and actuators). The information (commands, events, requests, and status notifications) exchanged between devices and components of the framework is transported by means of communications functions. The DT supporting functions segment represents the execution environment of DTs and the set of intelligent functionalities that DT instances use for orchestrating their interactions and lifecycles (e.g., instantiation, replication, identification, access control, and self-management), managing collected data, and simulation capabilities. This segment mediates the interaction between applications and physical devices. New functions, mechanisms and tools may be progressively designed and added to better meet DTs requirements. The application and services segment supports the applications that use the DTs interfaces and functions. For customization and execution purposes, DT replicas may be locally executed in this environment.

The actual exploitation of the DT as a general-purpose approach occurs if some conditions are met: the availability of a software framework capable of fulfilling the properties of DTs; the ability to create new applications with better functionalities for the user; and the ability to introduce new business models or to improve existing ones. These issues are analyzed in the following sections.

APPLICATION SCENARIOS

DT can be used to implement several scenarios. In this article, two application scenarios, based on properties of DTs and software framework, illustrate possibilities and issues.

Scenario 1: Virtual Object

This example describes how to create the DT of an object. A sensor is chosen, but different (aggregations of) objects may be considered. The needed design tasks are given as follows.

Physical object model: An abstract computable description of the data and behavior of the PO that the DT implements. The sensor status and the measurements it takes in the environment are relevant aspects to consider and to model.

API definition: an interface for requesting the functionalities offered by the DT. Applications use it to get data or to give commands. Data formats and interaction mechanisms should be defined.

Entanglement model: a model governing how PO and DT exchange data and status updates. The framework must offer mechanisms and protocols to support communication. Often the data flow is unidirectional from PO to DT, while it is bidirectional between the DT and applications (data, events, and commands). Commands from the applications to DT are used to govern the DT behavior.

A single PO can be replicated by several DTs. This occurrence accommodates specific requirements of applications: providing special functions, or permit access to the data with different rates, or segmenting and securing the computational space. Replicas can be instantiated in the framework and others in the application space. This option offers flexibility in deployment. Replication of DTs poses interesting issues. When the status of the PO changes, all the replicas need to change status. The propagation of information can be achieved with different mechanisms; PubSub is used in IoT implementations¹¹ and it suits the case of unidirectional communication (where the PO is a publisher and the replicas are subscribers). If PO status can be modified or it can be instructed (an actuator), the problem of synchronization and consistency of information between the PO and different replicas emerges. This issue can be solved by introducing transaction management functions into the framework.

Scenario 2: Digital Twins in Smart City

Cities are complex systems¹² difficult to control and program. A prerequisite for a smart city is having “measurable” attributes. Sensors and actuators are deployed to capture data that describe its processes. This make a city quantifiable and assessable. Once reliable data are available, artificial intelligence and machine learning, AI/ML, techniques determine patterns and behavioral models of urban systems.¹³ The city becomes “understandable,” i.e., it is possible to predict events and their associated causes by means of data analysis. DTs are a step toward the programmability of a city: the digital representation of resources makes it possible to design applications that, by interacting with DTs, can affect the behavior of “things” and optimize city resources and their behavior. Several experiences introduce the DT concept in smart

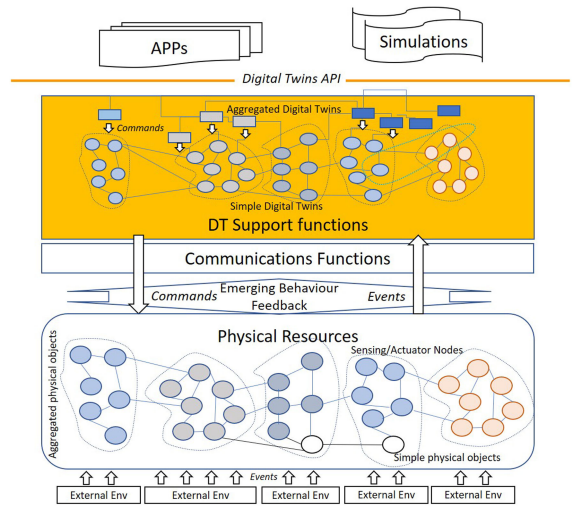


Figure 3. Programmability of complex systems with digital twins.

cities.^{14,15} The DT application introduces new capabilities and has an impact on how to govern the complexity of urban systems. For example, the adoption of the DT approach is improving the usage of building information modeling, e.g., such as given by Delbrügger *et al.*¹⁶

DTs are programmable objects that can orchestrate, on request of applications, POs like actuators toward the desired behavior. DTs can detect malfunctioning in POs and minimize their impact on other resources. DTs, exploiting advancement in storage technology, can store large datasets and hence supporting the continuous “learning” of city behavior. They offer the possibility to evaluate new policies by simulating future situations. DTs could be utilized to solve a common issue of smart cities: the siloing of application domains (e.g., transportation, energy, and so on). DTs offer interfaces through which different applications can access data, behavior, and status corresponding to specific resources. DT composability enables the creation of object aggregations that support abstract views of larger phenomena. The framework must scale up from few resources to a multitude. Figure 3 represents a DT platform able to capture the behavior of a complex system. Some applications are designed to monitor the situation for applying policies on specific issues (traffic, transportation, etc.). When issues arise, these applications implement mechanisms to minimize impacts on users and optimize

291 resource usage. Other applications are simula- 338
292 tions executed to understand how the “smart 339
293 city” could react to changes in policies of 340
294 resources’ usage. Benefits of applying the DT 341
295 concept to the “smart city” are many. These 342
296 advantages depend upon the introduction of 343
297 higher programmability levels, a strong integra- 344
298 tion with AI/ML capabilities, the elimination of 345
299 silo constraints, and the possibility to simulate 346
300 the functioning of a city. 347

301 BUSINESS AS USUAL? 348

302 The DT concept has business implications. It 350
303 changes the nature of POs and products. DT 351
304 properties facilitate the adoption of servitiza- 352
305 tion, i.e., the ability to transform a physical prod- 353
306 uct into a set of services offered to the user.^{17,18} 354
307 Servitization nurtures customization of function- 355
308 alities, making products more aligned to the life- 356
309 styles and needs of people and it introduces 357
310 additional innovations in the service area.¹⁹ 358
311 Products could even be modified, improved, and 359
312 customized by the user. New functions can be 360
313 added to products if users’ needs change. The 361
314 customer relationship process could be trans- 362
315 formed.²⁰ Customers are “entangled” with prod- 363
316 ucts. If malfunctions emerge in a product, they 364
317 can be predicted before they actually occur and 365
318 the impact on users could be mitigated. Mainte- 366
319 nance can be improved by focusing more on pre- 367
320 venting problems than on solving them. The 368
321 products’ value chain could be largely modified. 369
322 As the mobile phone industry showed, a rich 370
323 application layer has a tremendous impact on 371
324 products and it can determine success in a 372
325 global market. Customization, and the actors 373
326 supporting it, could become an important com- 374
327 ponent of the entire value chain of new digital 375
328 markets. 376

329 CONCLUSIONS 377

330 The DT approach promises to have impact. It 378
331 may provide solutions in different sectors (smart 379
332 cities, e-health, e.g., virtual patient, automotive, 380
333 cultural heritage), supporting the provision of 381
334 servitized products, and programmability of 382
335 complex systems. It fosters new ways of doing 383
336 business by transforming products into sets of 384
337 customizable services (an autonomous car and a 385

smart building). A precise definition of the DT 338
can help in better understanding requirements 339
and impacts on IoT software platforms. DTs 340
should be evaluated for their ability to support 341
and deliver the expected benefits. Further inves- 342
tigation on how DTs can enable an ecosystem of 343
applications that are feasible, marketable, and 344
sustainable by the industry is needed. There are 345
two major concerns: feasibility, i.e., the complex- 346
ity and technical challenges associated with the 347
creation of scalable, secure DT frameworks are 348
such that the construction of a viable platform 349
remains to be proven in-the-large; and user own- 350
ership, i.e., the possibility to frame users into a 351
system of products and applications that limit 352
privacy. The DT approach needs validation 353
before it will be widely accepted and used. It 354
already is a powerful means to control, govern, 355
and program the lifecycle of objects, physical 356
resources, and products, for providing better 357
services to end-users. It is a challenge to acade- 358
mia and industry to find relevant applications 359
for it. 360

■ REFERENCES 361

- 362 1. M. Grieves, *Product Lifecycle Management: Driving* 363
the Next Generation of Lean Thinking. New York, NY 364
USA: McGraw-Hill, 2006. 365
- 366 2. R. Rosen, G. Von Wichert, G. Lo, and 367
K. D. Bettenhausen, “About the importance of 368
autonomy and digital twins for the future of 369
manufacturing,” *IFAC-PapersOnLine*, vol. 48, no. 3, 370
pp. 567–72, Jan. 1, 2015. 371
- 372 3. F. Tao, J. Cheng, Q. Qi, M. Zhang, H. Zhang, and 373
F. Sui, “Digital twin-driven product design, 374
manufacturing and service with big data,” *Int. J. Adv.* 375
Manuf. Technol., vol. 94, no. 9–12, pp. 3563–3576, 376
Feb. 1, 2018. 377
- 378 4. E. Negri, L. Fumagalli, and M. Macchi, “A review of the 379
roles of digital twin in CPS-based production 380
systems,” *Procedia Manuf.*, vol. 11, pp. 939–948, 381
2017. 382
- 383 5. S. Haag and R. Anderl, “Digital twin—proof of concept,” 384
Manuf. Lett., vol. 15, pp. 64–66, 2018. 385
- 386 6. C. Steinmetz, A. Rettberg, F. G. C. Ribeiro, 387
G. Schroeder, and C. E. Pereira, “Internet of Things 388
ontology for digital twin in cyber physical systems,” in 389
Proc. VIII Brazilian Symp. Comput. Syst. Eng., 2018, 390
pp. 154–159. 391

- 386 7. V. Souza, R. Cruz, W. Silva, S. Lins, and V. Lucena, "A
387 digital twin architecture based on the industrial
388 Internet of Things technologies," in *Proc. IEEE Int.*
389 *Conf. Consum. Electron.*, 2019, pp. 1–2.
- 390 8. S. Malakuti and S. Grüner, "Architectural aspects of
391 digital twins in IIoT systems," in *Proc. 12th Eur. Conf.*
392 *Softw. Architecture: Companion*, Sep. 24, 2018,
393 pp. 1–2.
- 394 9. S. Malakuti, J. Schmitt, M. Platenius-Mohr, S. Grüner,
395 R. Gitzel, and P. Bihani, "A four-layer architecture
396 pattern for constructing and managing digital twins," in
397 *Proc. Eur. Conf. Softw. Architecture*, Sep. 9, 2019,
398 pp. 231–246.
- 399 10. AIOTI - WG3 - IoT Standardization "High level
400 architecture," Jun. 2018, Accessed on: Jan. 31, 2020.
401 [Online]. Available: [https://aioti.eu/wp-content/
402 uploads/2018/06/AIOTI-HLA-R4.0.7.1-Final.pdf](https://aioti.eu/wp-content/uploads/2018/06/AIOTI-HLA-R4.0.7.1-Final.pdf)
- 403 11. D. Happ, N. Karowski, T. Menzel, V. Handziski, and
404 A. Wolisz, "Meeting IoT platform requirements with
405 open pub-sub solutions," *Ann. Telecommun.*, vol. 72,
406 no. 1/2, pp. 41–52, Feb. 2017.
- 407 12. M. Batty, "Complexity in city systems: Understanding,
408 evolution, and design," *A Planner's Encounter With*
409 *Complexity*. G. de Roo and E. Silva Eds. 2007,
410 pp. 99–122.
- 411 13. S. Khan, D. Paul, P. Momtahan, and M. Aloqaily,
412 "Artificial intelligence framework for smart city
413 microgrids: State of the art, challenges, and
414 opportunities," in *Proc. 3rd Int. Conf. Fog Mobile Edge*
415 *Comput.*, 2018, pp. 283–288.
- 416 14. C. Perera, A. Zaslavsky, P. Christen, and
417 D. Georgakopoulos, "Sensing as a service model for
418 smart cities supported by Internet of Things," *Trans.*
419 *Emerg. Telecommun. Technol.*, vol. 25, no. 1,
420 pp. 81–93, 2014.
- 421 15. G. Garfield-Bennett, "Virtual Jersey becomes a
422 reality," *Jersey Evening Post*, Feb. 8, 2019, Accessed
423 on: Jan. 31, 2020. [Online]. Available: [https://
424 jerseyeveningpost.com/news/business/2019/02/08/
425 virtual-jersey-becomes-a-reality/](https://jerseyeveningpost.com/news/business/2019/02/08/virtual-jersey-becomes-a-reality/)
- 426 16. T. Delbrügger, L. T. Lenz, D. Losch, and J. Romann, "A
427 navigation framework for digital twins of factories
428 based on building information modeling," in *Proc.*
429 *22nd IEEE Int. Conf. Emerg. Technol. Factory Autom.*,
430 2017, pp. 1–4.
- 431 17. C. Kowalkowski, H. Gebauer, B. Kamp, and G. Parry,
432 "Servitization and deservitization: Overview, concepts,
433 and definitions," *Ind. Marketing Manage.*, vol. 60,
434 pp. 4–10, 2017.
18. Q. Qi, F. Tao, Y. Zuo, and D. Zhao, "Digital twin service
435 towards smart manufacturing," *Procedia CIRP*, vol. 72,
436 pp. 237–242, Jan. 1, 2018. 437
19. P. Zheng, T. J. Lin, C. H. Chen, and X. Xu, "A
438 systematic design approach for service innovation of
439 smart product-service systems," *J. Cleaner Prod.*,
440 vol. 201, pp. 657–667, Nov. 10, 2018. 441
20. R. N. Bolton *et al.*, "Customer experience challenges:
442 Bringing together digital, physical and social realms,"
443 *J. Service Manage.*, vol. 29, pp. 776–808,
444 Oct. 15, 2018. 445

Roberto Minerva is currently an Associate Pro- 446
fessor within the Service Architecture Lab, Institut 447
Mines Telecom–Telecom Sud Paris, Institute Poly- 448
technique de Paris, Paris, France. He received the 449
M.S. degree in computer science from Bari Univer- 450
sity, Bari, Italy, in 1987, and the Doctoral degree in 451
computer science and telecommunications from 452
UPMC (Paris-Sorbonne University) in 2013. From 453
1987 to 2016, he was a Researcher then a responsi- 454
ble of the Service Architectures and Network Intelli- 455
gence area within Telecom Italia Research Center. 456
From 2016 to 2018, he was the Technical Project 457
Leader of SoftFIRE, a European Project devoted to 458
the experimentation of NFV, SDN, and edge comput- 459
ing. He has been the Chairperson of the IEEE IoT Ini- 460
tiative in the period 2014–2016. He is a Senior 461
Member of IEEE. Contact him at roberto.minerva@te- 462
lecom-sudparis.eu. 463

Noël Crespi joined the Institut Mines-Telecom 464
SudParis, in 2002, and is currently a Professor and 465
Program Director with the Institut Polytechnique de 466
Paris, leading the Service Architecture Laboratory. 467
He is also an Adjunct Professor with KAIST 468
(South Korea), an Affiliate Professor with Concordia 469
University (Canada), and a Guest Researcher with 470
the University of Goettingen (Germany). His current 471
research interests include data analytics, the Internet 472
of Things, and softwarization. He received the mas- 473
ter's degrees from the Universities of Orsay, Paris 11, 474
France, and Kent, Canterbury, U.K., a diplome 475
d'ingénieur from Telecom ParisTech, Paris, France, 476
and the Ph.D. and Habilitation degrees from UPMC 477
(Paris-Sorbonne University). From 1993 to 2002, he 478
worked in the industrial research of telecom organi- 479
zations in the creation of new services and 480
in standardization. He is a Senior Member of IEEE. 481
Contact him at noel.crespi@mines-telecom.fr. 482