Shades of Digital Twinning

Author:
Dr. James J. Hunt
CEO & CTO
aicas GmbH
jjh@aicas.com

IoT Growing Pains: A Digital Twin Perspective

The Internet of Things (IoT) is proliferating with the advent of affordable and accessible processors and sensors enabling more businesses to discover the value in connecting devices to their Information Technology (IT) systems. Sensors for devices are readily available and the cloud has become ubiquitous. Low-cost data storage and analysis for processing vast amounts of data are all merging with new wireless protocols into a powerful force for digital industrial transformation.

The IoT is ushering in the fourth industrial age—Industry 4.0—in an evolution of manufacturing and production from centralized to decentralized and a merger of Operational Technology (OT) with IT. The rise of smart manufacturing is being propelled by connecting ever more powerful devices for factory control directly to the network. As these devices gather and preprocess operational data, the ability to mirror machines and their control in digital models has given rise to the concept of Digital Twins. Pairing of a digital twin and a physical device enables virtual analysis and monitoring to predict and prevent issues before they impact operations.

Defining how IoT devices are developed, tested, deployed and used across operations, as well as the interoperability for updating and improving functions, will provide individual smart manufacturing operations with business-value: better ROI, lower maintenance costs and other operational benefits. IoT is becoming the core driver of process improvement. Upper management may not necessarily care about the architecture of the IoT, but it does need to have the right information at the right time. The architecture is important to ensure that the maximum benefit from IoT is realized.

The focus for IoT and digital twin techniques has been on data collection, visualization and comparison with behavioral models: the twins. These are needed for catching anomalies that indicate imminent failure, but they may also show what factors lead to failure. This information can be used to mitigate failures by changing system behavior. This means changing software; using a device shell on a self-aware system with version and life-cycle management.

An IoT system could be built without standards. Standards, however, are necessary to build interoperable IoT systems. Different aspects of an IoT system have different requirements for standardization. Simulation models could benefit from CAD model standards whereas device proxies need standard protocols to talk with the device. Particularly, industrial systems are often composed of components from diverse sources. Standards will be necessary for them to interact.

Some standards for industrial data exchange exist, such as OPC-UA [1]; but they need updating for security, if nothing else.

[1] https://opcfoundation.org/markets-collaboration/ids/

Standards for software versioning and life-cycle management are few, but OSGi[2] stands out as a well-defined standard modular framework for this. When used with real-time Java, OSGi could also be used for control applications. It is already an element of some gateway and home automation systems. Since communication can be modularized, it also reduces the cost of switching protocols. This might be a good standard for the device shell.

Data exchange standards for CAD models could help produce a digital twin, especially with the correct data reduction methods to remove unnecessary data. Automatically identifying articulation points from a CAD model can be challenging, but this approach speeds up building functional models of the machines being controlled. The more data that can be reused rather than recreated, the better.

The most important places to standardize are protocols for data exchange and device control, as well as CAD models. The protocols are needed for the device shell to communicate with its proxy. CAD models are needed to produce twins that can provide visual feedback as well as to interact with a simulation environment.

In some sense, digital twin is not a new concept. Simulation models have been used for decades. In the past, these models were only used to develop systems but not monitor them. Connectivity brings the ability to compare and contrast what a model would predict with the actual physical behavior of the device. This increases both the value and accuracy of simulation models. But digital twinning is defined more broadly than just a simulation model. Even within simulation, there is more than one way to simulate something.

Devices and their twins are largely recognized as the workhorses that have simplified the creation of an IoT system[3]; however, the definition of digital twin is not very precise. Digital twin has become the popular, widespread catchall for the digital representation of a physical asset. This ambiguity has been compounded by the intertwining of the term "Asset Administration Shell"[4] with that of "Digital Twin." While digital twin has simplified complex parts of the IoT, the myriad interpretations and definitions of very distinct architectures are compounding ambiguity of the term. There is a need to refine the meaning of both terms, or at least subdivide them into more useful categories for reasoning about edge or fog computing. The goal is to provide a better definition for bridging the gap between OT and IT.

What is Digital Twin?

Why may there be confusion? If one looks at definitions by trade and standards organizations, what is being called digital

[2] https://osgi.org/specification/osgi.core/7.0.0/framework.lifecycle.html

[3] https://www.gartner.com/smarterwithgartner/how-digital-twins-simplify-the-iot/

[4] https://www.plattform-i40.de/I40/Redaktion/EN/Downloads/Publikation/vws-in-detail-presentation.pdf?__blob=publicationFile&v=7

twin is far too broad. One can start with a wide definition, but it is critical to uncover what is hidden under these far-reaching brush strokes. There are other concepts needed to express the edge computing ecosystem; parts such as control shell, device proxy, device shadow, visual twin, simulation twin, etc. Of course, it would help to align these terms better with the words used to describe them.

This is not to downplay the digital twin concept. Twinning is central to the value chain of IoT, but oversimplification downplays the complexity of an architecture that needs to be developed for the IoT to be successful. That begins with nomenclature.

There are many definitions for digital twin. Generally the definition surrounds "a digital representation of a physical object or system." Some, such as Oracle, consider it "a software or virtual representation of a physical asset with the objective of making the asset more valuable."[5] Siemens offers "A digital twin is a virtual representation of a physical product or process, used to understand and predict the physical counterpart's performance characteristics."[6]

The definitions may differ, but the concept is the same: "a virtual, digital equivalent to a physical product or the Digital Twin" as defined by Dr. Michael Grieves, when he first coined the name in 2003 for his University of Michigan Executive Course on Product Lifecycle Management.[7]

At that time, the concept was in its infancy. Hardly any data was being collected from physical products, but the core of what was identified was the critical three elements responsible for a flourishing IoT: a physical product, a virtual product and what connects them. It was an important concept to note, as Grieves in a 2014 whitepaper [8] documented that over the course of the ten years since digital twin was born, "the development and maintenance of the virtual product and the design and manufacture of the physical product has exploded."

According to the Gartner Hype Cycle,[9] four years later in 2018,[10] digital twin reached the peak of the IoT hype cycle. Building on Gartner and other industry analyst reports, the ISO and IEC Joint Technical Committee for information technology (ISO/IEC JTC1) put digital twin in the top five category of

[5] https://www.iiconsortium.org/wc-technology.htm

[6] https://www.plm.automation.siemens.com/global/en/our-story/glossary/digital-twin/24465

[7] Grieves, M. Digital Twin: Manufacturing Excellence through Virtual Factory Replication; A White Paper; Michael Grieves, LLC: Melbourne, FL, USA, 2014. [Google Scholar]

[8] M. Grieves, "Digital twin: Manufacturing excellence through virtual factory replication," White paper, 2014 [Online]. https://research.fit.edu/media/site-specific/researchfitedu/camid/documents/1411.0_Digital_Twin_White_Paper_Dr_Grieves.pdf

[9] Gartner Hype Cycle for Emerging Technologies, 2017 & 2018. https://www.gartner.com/smarterwithgartner/top-trends-in-the-gartner-hype-cycle-for-emerging-technologies-2017/

[10] https://www.gartner.com/smarterwithgartner/gartner-top-10-strategic-technology-trends-for-2018/

emerging technologies that urgently need standards, noting that "many industries are evolving rapidly as the digital transformation towards the fourth industrial revolution advances. The challenge to develop relevant standards will be more complicated with many more points to consider than previously."[11]

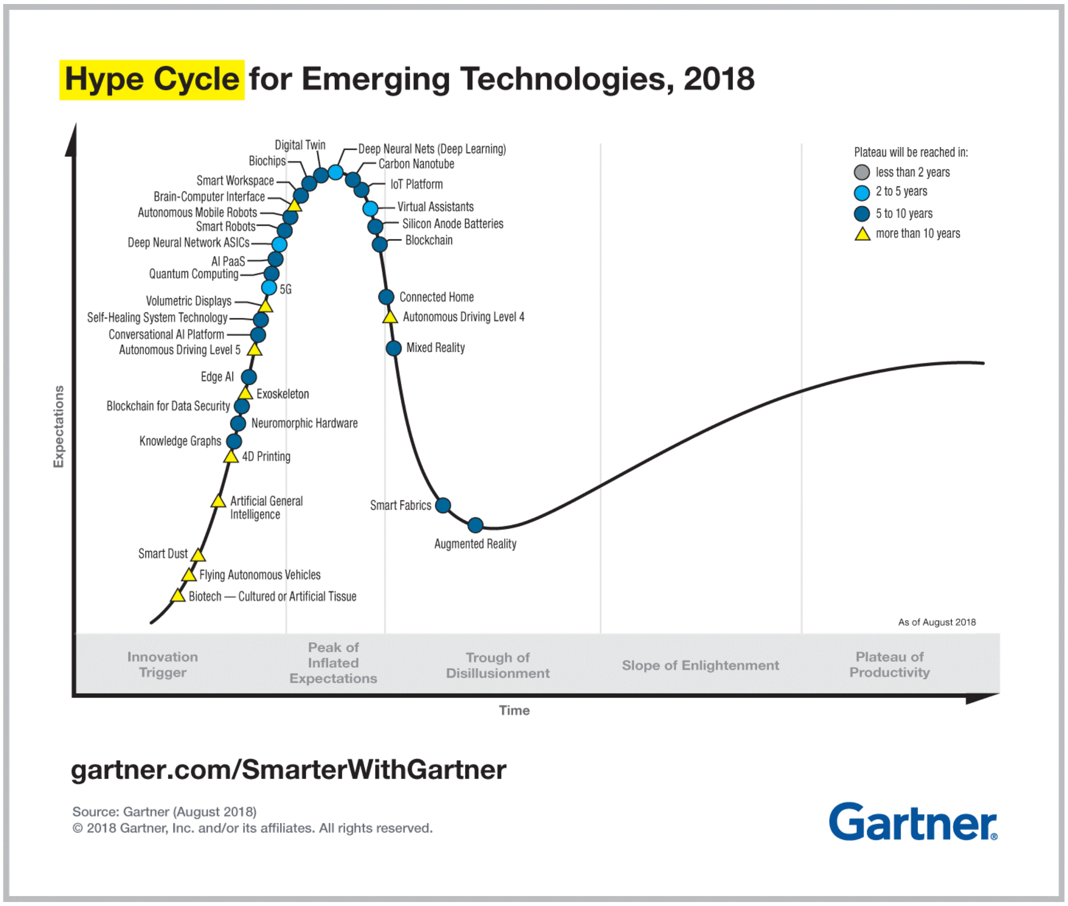

Figure 1: Hype Cycle for Emerging Technologies, 2018

Emerging Standards and the Asset Administration Shell

The German Plattform Industrie 4.0 selected the unfortunate term, Asset Administration Shell,[12] as their name for any digital model of a physical device used in smart manufacturing, since the term shell has been used in IT for a command line interpreter for controlling an operating system for more than 40 years. Asset Administration Shell conjures up the idea of a way of controlling edge devices. Nonetheless, its use in IEC PAS

[11] https://jtc1info.org/technology/jeti/

[12] https://www.arcweb.com/blog/concepts-applications-i40-asset-administration-shell

63088,[13] a multinational and multi-organizational effort to define a reference architecture model for smart factories, uses Asset Administration Shell to name digital models of devices. Plattform Industrie 4.0 is collaborating with the Industrial Internet Consortium, where Asset Administration Shell is seen as a synonym for Digital Twin.

In 2019, the IEEE Standards Association took the issue to task as well with its "P2806—System Architecture of Digital Representation for Physical Objects in Factory Environments."[14] The goal is to define "the system architecture of digital representation for physical objects in factory environments. The system architecture describes the objective, important components, required data resources and basic establishing procedure of digital representation in factory environments."

Some architectures assume the standard definitions for an Administration Shell are sufficient. Administration Shell submodels are being produced relating to functions, such as safety, or process capabilities, such as energy efficiency. However, the emerging and expanding portfolio of subtly differentiated submodels further add to the confusion. Each person has a clear picture of what the different parts are; however, the industry needs standards and interoperability, as well as a defined vocabulary. Administration shells and digital twins are not quite the same thing and do not take into account that IoT architecture requires consideration of code running on the edge, code in the cloud representing the current state of the edge and code in the cloud modeling the states the edge may take. Having well defined terms for these various components would help build better IoT systems. After all, collecting data is just half the job; ultimately, the behavior of edge devices will need to evolve as well.

THE DEFINITIONS AND FEATURES: SHELL, PROXY, TWIN

So what is a twin? The digital twin is not a proxy for a device, and it is not a command interface or shell for a device. A twin is a means of visualizing a device and emulating what could happen operationally. The architecture must enable the user to investigate "what if" questions informed by actual data from running systems. This is more than the traditional simulation models used for product development. Rather, twinning is a means of refining these models and extending them from the design and development phase through deployment and service phases of the product's life cycle. Thus, digital twinning can be extended with the following definitions:

- Device Shell—a command interface for code running on a physical asset on the edge;
- Device Proxy—a virtual representation of the current state of a device; and
- Device Twin—virtual representation of the device for simulation and visualization.

[13] https://standardsdevelopment.bsigroup.com/projects/2016-03114

[14] https://standards.ieee.org/project/2806.html

The Device Shell

The device shell provides a means of managing the life-cycle of the code running on an edge device. With it, one can install new software and upgrade existing software, as well as control what software is running at any given time. One can also monitor the current state of the system and know what versions of each component are running on the system. Version compatibility checking and admission control for new software are essential parts of these services. Other services, such as system logging, watchdog management and intrusion detection, may also be provided. Secure system identity is also a key part of the device shell. Remote administration can be provided over a remote device shell.

The Device Proxy

The device proxy bridges the gap between the device shell and its device twins. There are two aspects to this: a stand-in for a non-connected device and a latest device state keeper, sometimes known as a device shadow, for covering the time when a connected device is not reachable—either because it is off or because of sporadic connectivity. Additionally, it can enable processes such as scheduling updates, and sending out update code modules.

The Device Twin

The device twin provides device visual and physical simulation. Whether visual or physical, a twin is a model and thus an approximation of the physical device. By its very nature, it is incomplete, but it should be fully accurate for the attribute under observation. For instance, a visualization model need not have all the CAD information used to build the model. In fact, it is desirable to have a simplified version thereof so the visual simulation can keep pace with the actual device. Any information about aspects of the model that are too small to see at the rendering resolution are irrelevant for the visual simulation and should rather be left out. A different model might be used for modeling the effect of wear and tear on parts of the device. Adding visual information would slow down the simulation with no benefit. For this reason, several twins may be used for a single physical device depending on what aspect is important for that simulation.

It would be useful to have methods for distilling various models from the original CAD data for a device, or rather a machine. Simulation using the full CAD data, where possible, would run quite slowly. Particularly, simulations that look at the interaction of machines are vulnerable to a problem of too much data. Common data exchange formats would be helpful, as well.

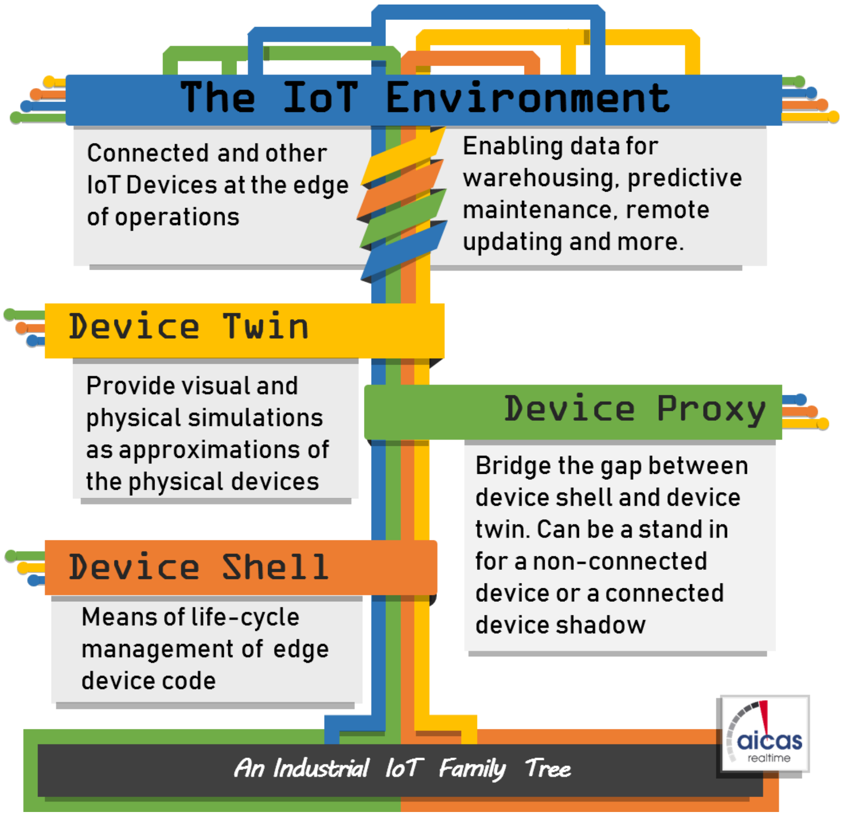

Figure 2: The IoT Family Tree

Why Rock the Boat?

The importance of distinguishing between a device twin and a device proxy is that they have different roles in the overall IoT architecture. They are both digital models, but the proxy is more of a data model whereas a twin is more of a behavior model of a device. It is useful to have a proxy in the cloud, especially a shadow, for each device in the field; but it may not be necessary to be able to simulate or visualize each and every device in a factory. Furthermore, merely collecting data is insufficient for obtaining the full value of IoT. Behavior must also change based on the data. This means that software life-cycle management, version control and version compatibility are also important. Hence, a device shell is also needed.

Data mining and cloud services have had a huge impact on IT systems. Connecting edge devices with the cloud promises to bring similar advantages to OT systems. Digital twinning is an important part of this story, but device proxies and device shells are also important. Having full control of device software is essential to close the loop of collecting data, learning from data and updating behavior. This is where the device shell comes into play. Full updates cannot be done in a running system. Updates must be modular so that the system interruption can be held to a minimum.

The main difference between IT systems and OT systems is that OT systems are time critical whereas IT systems are not. The techniques used to manage IT systems must be modified to apply to OT systems. This is one reason why Programmable Logic Controller (PLC) languages are used for OT software but are unheard of in IT systems. Connectivity must work with such systems.

Interoperability is the cornerstone of Industry 4.0. The rapid advance of the IoT has created disparate terms and definitions around monitoring, diagnostics and data analytics that may inhibit the required interoperability. Universal definitions ease the task of bringing together systems from various manufacturers and suppliers.

This is especially true when the requisite security aspects are taken into consideration. For instance, a device shell must be able to authenticate itself to its proxy so that the IT system can be sure it is monitoring and controlling the correct devices. One must also be able to ensure that only the desired software is run on any given device. Only then can the full advantages of automatic software deployment and updates become available without introducing new problems into the factory.

The Business Case for Standards: It is not about the data

Digital Twin should not be a catchall for any digital model of a physical system. By distinguishing between device shell, device proxy and device twin, one can focus on the necessary standards and protocols for each type of interaction. Nebulously defined aspects of the IoT will stunt its growth while uniform identifications will enable accurate IoT virtualization. Now is the time to classify these capabilities to foster adoption and easier integration of applications, and to set expectations for how these can be updated and exchanged.

➢ Return to IIC Journal of Innovation landing page for more articles and past editions