2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA) | 978-1-6654-9996-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/ETFA52439.2022.9921725

# ISO23247 Digital Twin Approach for Industrial Grade Radio Frequency Testing Station

Valdemar Leiras*, Sandra Dixe*, Nuno M. C. da Costa*†, L. Filipe Azevedo*,
Paulo Cardoso*, Jaime C. Fonseca*, António H. J. Moreira†‡, João Borges*‡
* *Algoritmi Center, University of Minho, Portugal*
† *2Ai-School of Technology, IPCA, Portugal*
‡ *Polytechnic Institute Cávado and Ave, Portugal*

***Abstract*—The Digital Twin approach has increased in interest in recent years. Without well defined specifications, it is common for different researchers to use different approaches. Digital Twin concept emerged to support Industry 4.0, so it is of utmost importance to specify a best methodology, and tools, for its implementation for industrial use cases. This work proposes a Digital Twin architecture that follows manufacturing-centric standards of an industrial prototype testing station, for a new car infotainment system. Testing is performed by low-cost Software-Defined Radio equipment that aims to replace expensive metrological equipment. Moreover, an environment sensing device is also used to monitor the physical environment around prototype. We present the development of a solution that allows manage hardware processing, real-time monitoring of machine states in the virtual environment, and control of the overall system through a logical sequence supported by its Digital Twin. With the implementation of a standard like ISO23247, helps in the identified problem of having various Digital Twins and improves an interaction with test equipment. All steps made for the development of our architecture approach, as well as some results, are shown and explained for our use case.**

***Index Terms*—Digital Twin, Manufacturing, Big Data, Virtualisation, Simulation, Monitoring**

## I. Introduction

The Digital Twin (DTw) concept was firstly present in 2003 by Michael Grieves [1] [2] [3] [4]. Similar ideas already had been proposed to the industry previously under different conceptual names. Since it emerged, demand for information technology supporting development, maintenance (virtual system), and manufacturing (physical system) has exploded. Primarily been developed in aeronautical fields and more recently becoming a research focus at manufacturing industry.

The emergence of DTws occurs due to the complexity of systems that answer the increased manufacturing of customised products. The concept is an answer to the need to connect already emerged technologies in Industry 4.0 (I4.0), such as Cyber-Physical Systems (CPS), or Internet of Things (IoT), that has enabled the adaptability and flexibility [5]. Nonetheless, this complex network leads to induced disturbances in production and increased costs. To avoid this, it is necessary to promote the interaction between different physical systems, or parts, by creating digital models of physical systems, in a manner that promotes a support for decision making.

The study of new approaches to this concept is increasing due to divergent opinions or related works that are mainly focused on particular use case applications. Several tools/platforms/frameworks are available, originating different approaches for DTw development. Some of these tools focus in 3D modelling, simulation, and predictive algorithms. To achieve DTw standardisation in manufacturing, ISO23247 [6] presents a framework that defines a setup for building DTws in a multi-dimensional/layer model. This article proposes the implementation of a ISO23247 DTw model in a industrial testing station, by establishing the communication between the different layers, link the physical and the digital, and giving a system management to the digital world. This standardisation implementation supports the manufacturing industry implementation of Digital Twins, by being adaptable to different application purposes. The implementation of DTws allows for improved testing procedures and monitoring capabilities, thus improving competitiveness, reducing testing time, resources and costs. The main contributions of the paper are as follows:

- ISO23247 DTw technological implementation methodology;
- DTw based industrial radio frequency (RF) testing procedure.

The rest of this paper is organised as follows. Section 2 presents an overview of the DTw state-of-the-art methodologies, enabling technologies and tools is presented. The technological DTw implementation is described in section 3. Experiments are described in section 4, with their corresponding results. The discussion is presented in section 5. In section 6, this paper is concluded.

## II. Related Work

For Qi et al., DTws are composed with three default parts: physical entity, virtual entity, and connections between them [7]. Other researchers, such as Zheng [8], defend that the constitution parts of a DTw are five: physical entity, virtual entity, data, services, and connections between. Several divergences exists regarding what a DTw should focus on, or in what give more relevance. Some believe that it should focus on modelling and simulation optimisation, while others believe that it should focus on the interconnection of the DTws multiple modules. Moreover, for Singh et al., a DTw does not exist until it has a physical entity, and it is the continuous connection between physical and virtual entities that makes the difference from conventional systems [9]. Attending the state-of-the-art for this radio testing station, related works [10] [11] shows the relevance of the application of DTws on verification, validation and testing systems in manufacturing.

### *A. Data Communication*

Many of the DTws found in the literature are focused on specific phases of an entire product process and work with a huge amount of information from previous processes, whereas I4.0 profits from the huge amount of data that automated systems provide [12]. To Sullivan and is colleagues, without well defined and synchronised data (that completely describes the physical entity) a DTw can not be built, thus data and communication specifications should be the first and more important phase of the building stages of any DTw [13]. Apache Kafka [14] provides a good solution because it is highly customised and scalable. They also promote the idea of a broker that interprets different protocols of communication such as Message Queuing Telemetry Transport (MQTT), Sockets, and Representational State Transfer (REST) [15]. In a PubSub messaging middleware, a single instance can publish data to multiple topics, and there can also be multiple DTw instances publishing data to a common topic [16]. Being the communication protocols a crucial enabler for DTws, a wide variety of standards are available. Moverover, the Open Platform Communication - Unified Architecture (OPC-UA) is a standard machine-to-machine communication protocol for industrial applications [17]. For the data transfer structuring, Vogt et al. [18] proposed known types of data structure and communication protocols that can be used, such as MQTT brokers with JavaScript Object Notation (JSON) files or OPC-UA with Xtensible Markup Language files for automated parsing.

### *B. DTw Framework*

Quamsane et al. [16] proposed a unified DTw platform that operates within a Software-Defined Control (SDC) framework, based on Tecnomatix from Siemens, for flexible control reconfiguration of smart manufacturing systems. Moreover, the DTw has a pool that hosts instances of different classes of DTws interconnected via a publisher-subscriber infrastructure. The DTw platform uses historical and real-time data to provide a SDC controller with a centralised overview, thus allowing for prediction and detection of anomalies, monitoring equipment health, or production, in real-time. Other approaches make use of different softwares of the shelf, integrated with each other to reach DTw functionalities. Zubiaga et al. [19] presented an application, where Tecnomatix and Unity were tested along with other tools (e.g. TIA (PLCSIM)+Tecnomatic, Unity+ABB Robot Studio). In the Xu et al. [20] work, is integrated Pro-E, Unity and MATLAB. Inside the virtual reality scope [21], the aggregation with Unity allows to create DTw-based virtual factories for collaborative and coordinated virtual reality (VR) training scenarios. Moreover, FlexSim adds simulation capabilities on top of a 3D environment. With a higher standard for industrialisation, Eclipse Ditto presents greater DTw solutions [22] [23] [24]. Romeo et al. [22], made use of Ditto for the creation of DTws. Moreover, Ditto allows storage of the current state of physical devices, in case of sudden loss of connection. Kamath et al. [23], was capable of mirroring physical assets or devices, providing aspects and services between the different devices, and features. Alternatively Sousa et al. [25] presents an architecture for management of complex systems, such as manufacturing, healthcare, automobiles, and aeroplanes. This architecture of a real-time system uses a Solace broker where data is published (i.e. sensors, databases, etc.) and then connects to Apache Kafka, or directly to Apache Spark software data stream inputs. The processed data appears in Tableau or in the Power BI platform, where it can be analysed and visualised.

## III. Implementation

This work proposes the implementation of a DTw in a test station (shown in Figure 1 b)), which performs industrial validation of car radios (i.e. Devices Under Test [DUT]). Thus, it is required to establish compatibility with the physical system requirements.

### A. Functional Specifications

The testing station main goal is inject RF signals (e.g. Frequency Modulation [FM]), sent by Software-Defined Radio (SDR), into the DUTs in an automated procedure that quantifies the quality of a device. To inject RF signals in the DUTs, an SDR was used (i.e. LimeSDR usb [26]), which is controlled by a Raspberry Pi 4 [27] (Figure 1 a)), the station interprets the feedback from the DUTs according to the test performed and validates, or not, the DUT by presenting a test result. Moreover, DUTs are automatically tested in a physical station, which reads a Quick Response (QR) code of the respective DUT when it enters the station. The station proceeds, through a robotic system, to place the DUT in a free testing-cell to start its DUT test. When the test sequence ends, the robotic system move the DUT from a testing-cell, to the initial position, signals that the test is over. Additionally, the test station provides a virtual environment (Figure 1 c)) in which it is possible to monitor that complete sequence (Figure 1 b)). This virtualisation was implemented in Unity software [28].

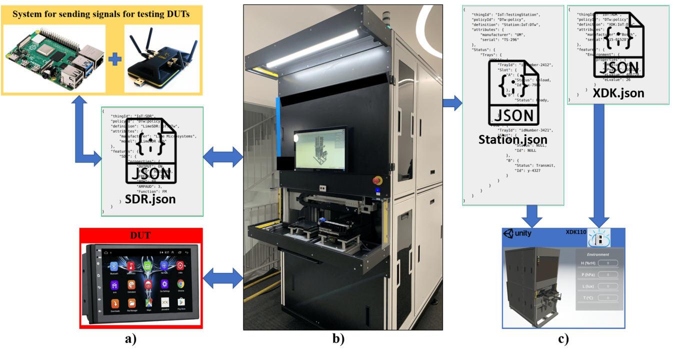


Fig. 1. (a) System to send RF signals for testing DUTs at the station. (b) Physical testing station. (c) Virtualisation of the physical station with information about surrounding environment captured from sensors in real-time.

To monitor this environment surrounding the physical station, an additional XDK110 [29] device, that includes sensors such as temperature, humidity, atmospheric pressure, accelerometer, noise and magnetometer, incorporates and ensures the ideal conditions for testing. All this information is presented in real-time in a developed virtualisation of the system (Figure 1 c)), thus allowing to control the conditions surrounding the station.

### B. Technological Specifications

The testing station is comprised by critical testing components, which guarantee the functional specifications, such as LimeSDR, XDK110 and Virtualisation Model. Moreover, these require compatibility with the DTw proposal, thus we implement an individual interface to each component.

*1) LimeSDR:* To perform RF tests in the DUTs, a SDR device was used together with a Raspberry Pi 4. LimeSDR usb implementation is based on GnuRadio [30] software, converted into a Python script, which aimed only at generating FM signals to test DUTs in this physical station. Intrinsically, LimeSDR requires four parameters: (FREQAUD) modulated signal frequency in Hz; (FREQ) carrier frequency in Hz; (POW) power signal gain; and (OUTPUT) to enable/disable the test (i.e. ["ON","OFF"]). Moreover, a generated signal is shown in Figure 2, where (AMPAUD) signal amplitude is an intrinsic gain from the GnuRadio parameters in the LimeSDR device.

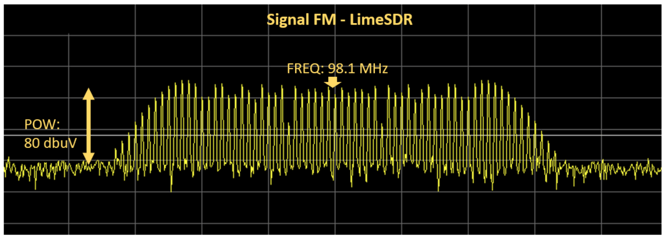


Fig. 2. RF signal generated with a LimeSDR, during a testing procedure, using FREQAUD=1kHz, FREQ=98.1MHz, POW=80dB$\mu$V

To guarantee his communication compatibility with the DTw, a JSON format [18] message was used together with a Solace based publishing topic [25]. An example of the SDR.json structure is:

```
{
    "thingId": "IoT:SDR",
    "policyId": "DTw:policy",
    "definition": "LimeSDR:IoT:DTw",
    "attributes": {
        "manufacturer": "Lime Microsystems",
        "model": "LimeSDR usb"
    },
    "features": {
        "SDR": {
            "properties": {
                "Status": "Transmitting",
                "Function": "FM",
                "OUTPUT": "ON",
                "FREQ": 98100000,
                "FREQAUD": 1000,
                "POW": 80,
                "AMPAUD": 3
            }
        }
    }
}
```

*2) XDK110:* To demonstrate the ease of integration of IoT devices in a DTws-based paradigm, a XDK110

[29] equipment device, developed by Bosch, was chosen. This IoT device was used to monitor environmental variables through XDK.json structure. For the application, this equipment collects multiple sensor features, such as pressure (ePvalue), temperature (eTvalue), humidity (eHvalue), and light sensor (eLvalue), while constantly publishing its information in real-time through the MQTT protocol [18], guaranteeing compatibility with a DTw interface (i.e one must subscribe to the topic to gather information).

*3) Test Station Automation System:* As it was described, the test station performs RF tests in DUTs. These tests are performed in an automated and optimised way, through a set of trays (i.e. defined by POSX), each comprised by two testing slots (i.e. Slot A and B). Moreover, each slot has a process status and a DUT Id described in TestingStation.json structure. In the same way as the XDK110, a token is used to publish, JSON structured messages, real-time information through MQTT protocol.

*4) Virtualisation Model:* The physical virtualisation model requires a constant feed of information from the different test station devices. Thus, JSON structures data, containing the physical system real-time information, are feed through different communication topics, allowing Unity to access and update its information, thus rendering real-time animations of various physical motions and events that arise on the real test station. The stations virtualisation system subscribes to these topics where various devices JSON information is published, namely LimeSDR, XDK110 and the test station automation system. The virtual environment mirrors the physical world through three JSON structure.

### *C. Digital Twin architecture*

This paper proposes a DTw approach for an industrial DUT test station, described above. The goal of creating a DTw is to bring to the digital world the physical one, in a secure way. Accessed with login credentials, this way, and we can guarantee the handling of the physical world remotely. Additionally, the intrinsic characteristics of a DTw are simulation, prediction, monitoring, and virtualisation in real-time. Figure 3 shows the main intrinsic characteristics to a DTw. From them, our case study derives:

- **Physical Entities:** XDK110 [29], Raspberry Pi 4 [27] with LimeSDR [26], and the test station automation system;
- **Virtual Models:** Unity 3D [28] based virtualisation of test station;
- **Data:** MongoDB [31] database, with JSON format messages for communication;
- **Services:** Email and SMS alerts to maintenance teams;
- **Connections:** Solace broker [32] (with MQTT and WebSockets protocols).

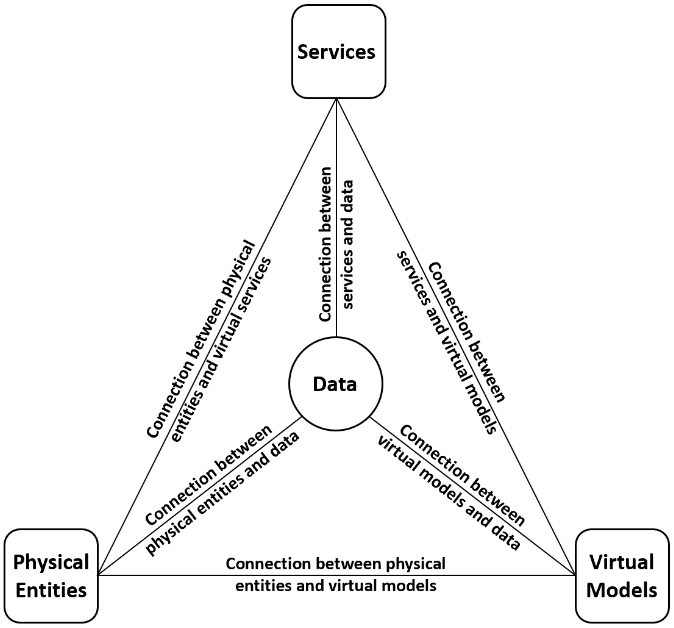

Services
Connection between services and data
Connection between physical entities and virtual services
Connection between services and virtual models
Data
Connection between physical entities and data
Connection between virtual models and data
Physical Entities
Connection between physical entities and virtual models
Virtual Models

Fig. 3. Basic architecture of a Digital Twin, main constituent parts.

Our proposal of DTw, aggregates all the derived characteristics of the test station, based on ISO23247 standard, (as shown in Figure 4). This standard was designed to be flexible and adaptable to the intended case study, so in our case, we followed this entire standard layers, except for the station virtualisation module, presented in Unity, which was directly connected to the Data Collection communication layer (i.e. Solace). The Cross-System Entity block is transversal to all the layers of our DTw, and it is from here that the user validates access to any state through his credentials. By following this approach, is promoted a better overall linkage of different DTws, supported by a framework that guides us in a global overview of a complex system. Easy to manage and to define, supports the flow of data through the different layers.

Both the physical layer, physical entities (i.e. test station, XDK110 and LimeSDR usb together with Raspberry Pi 4) represented in the bellow block of Figure 4 (i.e. Observable Manufacturing Elements), and the communication layer (i.e. Data Collection and Device Control Entity) are connected through MQTT protocol. The station virtualisation layer (i.e. Unity) is connected to the Data Collection and Device Control Entity communication layer, through WebSocket protocol. This is a standard protocol accepted by both Unity and Solace, which allows interpretation compatibility with

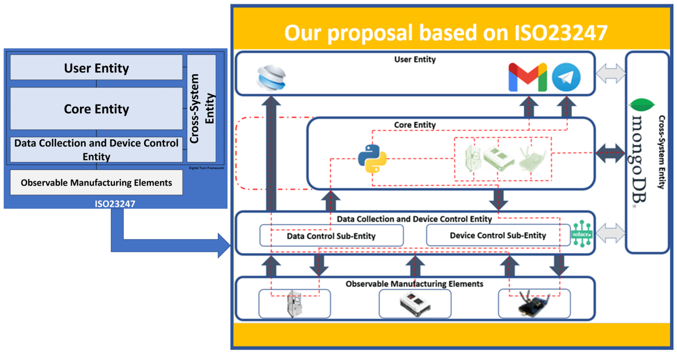

User Entity
Core Entity
Data Collection and Device Control Entity
Cross-System Entity
Digital Twin Framework
Observable Manufacturing Elements
ISO23247
Our proposal based on ISO23247
User Entity
Core Entity
Data Collection and Device Control Entity
Data Control Sub-Entity
Device Control Sub-Entity
solace
Observable Manufacturing Elements
mongoDB
Cross-System Entity

Fig. 4. Digital Twin proposal for our case study based on the ISO23247.

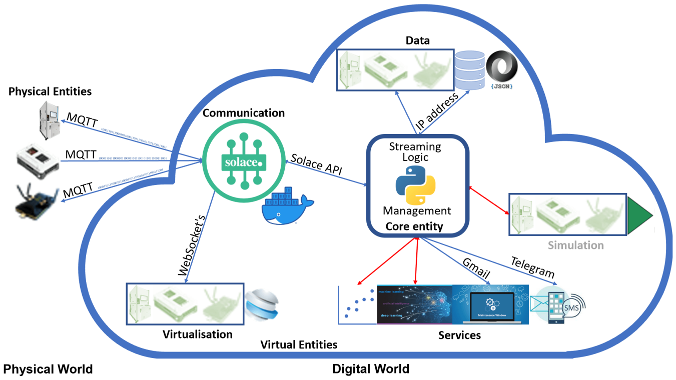


Fig. 5. Digital Twin proposal for our case study architecture.

the MQTT protocol. The Core Entity layer, which has the functionality to manage DTws, that is, our physical devices (i.e. test station, XDK110 and LimeSDR usb together with Raspberry Pi 4) is managed by Python scripts (one for each physical device) that stream data from the physical devices and update the respective information, in real-time, to the respective JSON file. Moreover, that transmit the information between the different modules of the system. In terms of user services (User Entity layer), email alerts and messages (i.e. Gmail and Telegram) were used to send alerts to maintenance teams when necessary. All the information is stored in a database (i.e. MongoDB) which resides in the Cross-System Entity Layer, which can be used for other possible DTw modules such as prediction and simulation. The red arrows refer to DTw modules (i.e. simulation, statistical, Artificial Intelligence) that we intend to implement in a future application. All the information that runs through our DTw approach is formatted and structured in JSON format whenever any change occurs in the physical devices.

## IV. Experiment

To prove the functionality of our DTw approach, and after implementing the architecture shown in Figure 5, we proceeded to test a DUT from start to finish in the physical station.

The whole process will be detailed in the points below:

1) When a DUT is placed in insertion/removal position at physical station, a sensor detects a DUT, activating a QR code reading camera, while triggering the update of the physical station status variable on his DTw.
2) The DTw at physical station checks which position of the station is free to receive a DUT, where it can be tested, and updates this variable instructing physical station to move the DUT to that free position.
3) Once the DUT has been placed in its test location, station informs Core Entity layer of the presence of a DUT. The Core Entity accesses testing configurations and sends desired parameters for those tests, to the LimeSDR system, where it injects that desired RF signals into DUT.
4) When that DUT has completed its test it informs the test station DTw that tests has been completed and that it should be removed from the stations test position. The DTw instructs the physical station to remove that DUT.
5) That DUT returns to initial position (i.e. insertion/removal), at which time an alert is issued with test results (i.e. positive or negative).
6) Additionally the XDK110 systems constituent sensors provides DTw with real-time information on the stations ambient conditions. This entire process takes place continuously in parallel to other processes that happens in the physical station.

The experiment described above is represented graphically in Figure 6. Whenever any event occurs as described in the previous points, these are updated or saved in a database.

Figure 7 shows the interaction between DTW and the LimeSDR testing system.

Finally, a laboratory testing procedure is presented in Figure 8. There it is possible to observe the injected signals from LimeSDR to a DUT.

## V. Discussion

This paper proposes the use of DTw for an application in an industrial test station according to ISO23247 (show in Figure 4). For this purpose, a prototype was developed that aimed to control and monitor an entire testing process of a DUT before it is placed on the market (show as in Figure 1). With the system and architecture presented in this article, it is possible to control a transmission of RF signals into DUTs, during an automated testing process.

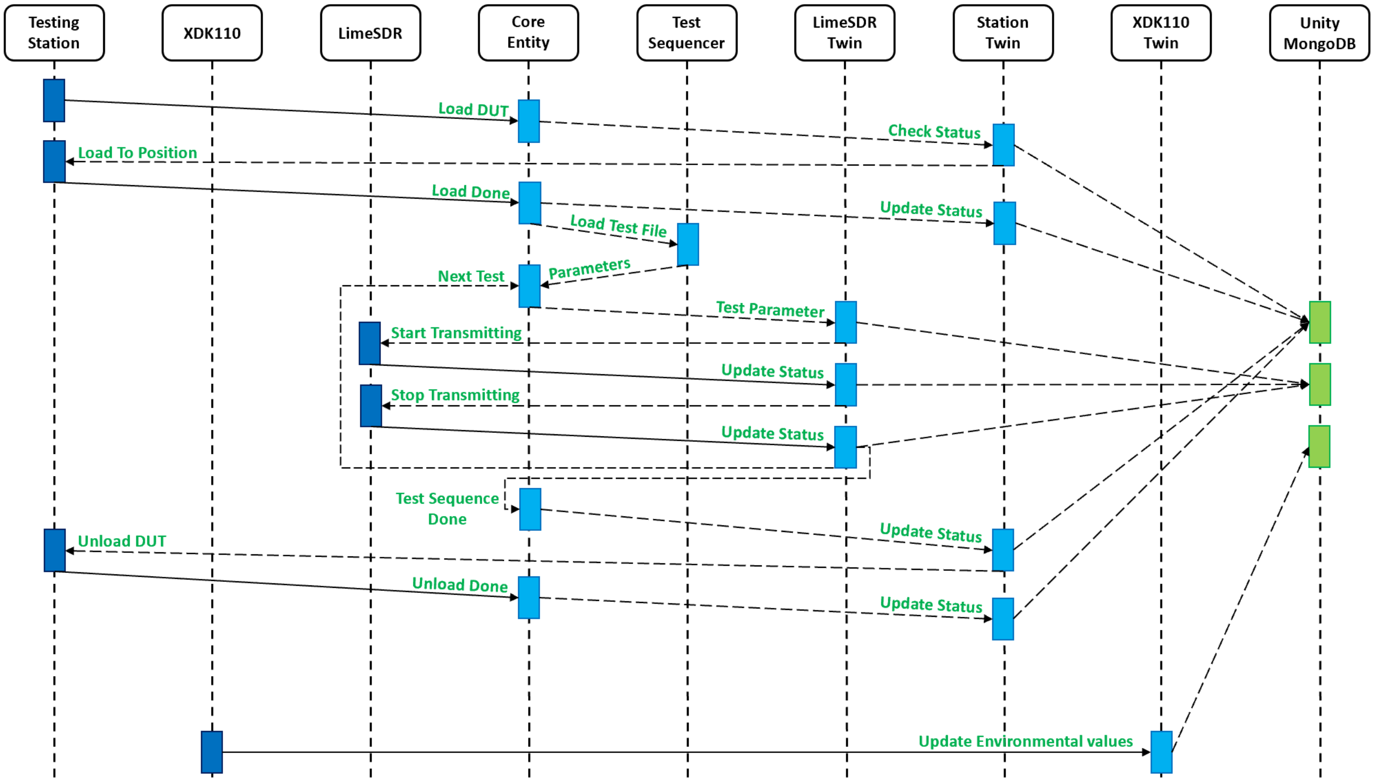
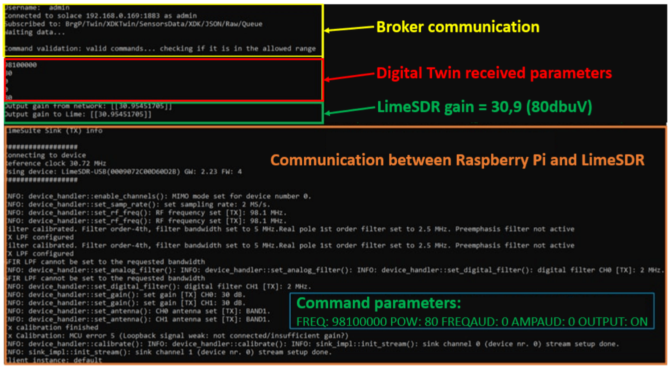


Fig. 6. Experiment testing execution sequence flowchart.


Broker communication
Digital Twin received parameters
LimeSDR gain = 30,9 (80dbuV)
Communication between Raspberry Pi and LimeSDR

```
Jsername:  admin
Connected to solace 192.168.0.169:1883 as admin
Subscribed to: BrgP/Twin/XDKTwin/SensorsData/XDK/JSON/Raw/Queue
Waiting data...

Command validation: valid commands... checking if it is in the allowed range
98100000
30
0
0
30
Output gain from network: [[30.95451705]]
Output gain to Lime: [[30.95451705]]
LimeSuite Sink (TX) info

##################
Connecting to device
Reference clock 30.72 MHz
Using device: LimeSDR-USB(0009072C00D60D2B) GW: 2.23 FW: 4
##################

INFO: device_handler::enable_channels(): MIMO mode set for device number 0.
INFO: device_handler::set_samp_rate(): set sampling rate: 2 MS/s.
INFO: device_handler::set_rf_freq(): RF frequency set [TX]: 98.1 MHz.
INFO: device_handler::set_rf_freq(): RF frequency set [TX]: 98.1 MHz.
Filter calibrated. Filter order-4th, filter bandwidth set to 5 MHz.Real pole 1st order filter set to 2.5 MHz. Preemphasis filter not active
TX LPF configured
Filter calibrated. Filter order-4th, filter bandwidth set to 5 MHz.Real pole 1st order filter set to 2.5 MHz. Preemphasis filter not active
TX LPF configured
GFIR LPF cannot be set to the requested bandwidth
INFO: device_handler::set_analog_filter(): INFO: device_handler::set_analog_filter(): INFO: device_handler::set_digital_filter(): digital filter CH0 [TX]: 2 MHz.
GFIR LPF cannot be set to the requested bandwidth
INFO: device_handler::set_digital_filter(): digital filter CH1 [TX]: 2 MHz.
INFO: device_handler::set_gain(): set gain [TX] CH0: 30 dB.
INFO: device_handler::set_gain(): set gain [TX] CH1: 30 dB.
INFO: device_handler::set_antenna(): CH0 antenna set [TX]: BAND1.
INFO: device_handler::set_antenna(): CH1 antenna set [TX]: BAND1.
Tx calibration finished
Tx Calibration: MCU error 5 (Loopback signal weak: not connected/insufficient gain?)
INFO: device_handler::calibrate(): INFO: device_handler::calibrate(): INFO: sink_impl::init_stream(): sink channel 0 (device nr. 0) stream setup done.
INFO: sink_impl::init_stream(): sink channel 1 (device nr. 0) stream setup done.
Client instance: default
```

Command parameters:
FREQ: 98100000 POW: 80 FREQAUD: 0 AMPAUD: 0 OUTPUT: ON

Fig. 7. DTw and LimeSDR system-wide communication results.

The communication and interaction between different parts of our DTw (i.e. Services, Physical Entities, Virtual Models, Data and inherent connections), happens in real-time and remotely (see Figure 7 and 8). This MQTT protocol was selected as it is one of the most widely used in industry to connect IoT devices. The communication server chosen to establish communication between layers was Solace, do to it compatibility with various communication protocols, including MQTT. This is a standard protocol accepted by both Unity and Solace, which allows interpretation compatibility with the MQTT. Additionally, virtualisation, monitoring and remote control capabilities are added to this test station, through DTws, together with an automated testing procedure. Moreover, allowing to test several devices simultaneously, even in different tests, though resource sharing, thus allowing a reduction in costs, space and human intervention. We can communicate and manage an entire testing process of a DUT in $5ms$, which means that we can produce a signal from

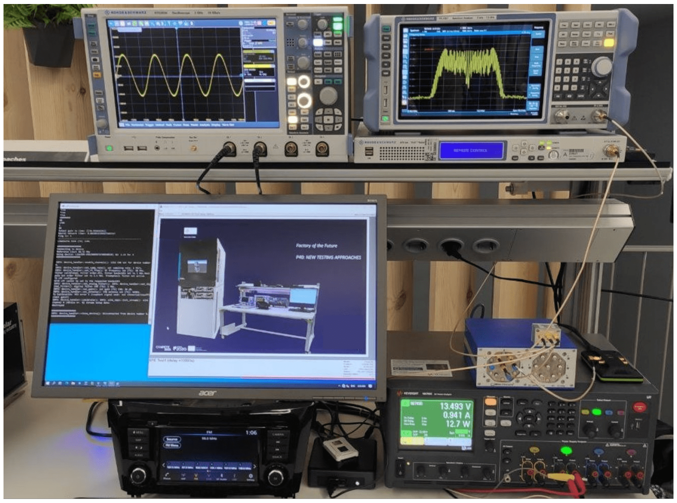


Fig. 8. Laboratory demonstration of testing procedure.

the SDR in less than $5ms$, which guarantees real-time requirements (as show in Figure 7).

The virtual model made in Unity software that subscribes directly to the broker, updates is status and animate the interface with an user, providing a real-time animation showing movements and other variables that was pretended to visualise.

The time between failures is incremented due this constant monitoring, and it reduces the number of DUTs that are failed in tests that are made. That reduction of failures makes that manual processes that are necessary to make extra tests to verify why DUT fails on tests. The control of this station also eliminates extra need of metrological hardware/software to verify the correct functioning of that testing station metrological systems, by making that with his proper systems.

The ISO23247 framework enables enormous versatility in building DTws, by giving the possibility to provide personal and dedicated applications. In our case that was verified through a direct connection of the virtualisation module to the communication layer, and having absence of other inherent modules of the DTws. The implementation of this standard framework is needed in future approaches, where all DTws could be at the cloud and available everywhere to everyone that have access credentials to that. Some of the modules of a DTw construction are not well defined in this implementation, but it was pretended to follow the presented setup, in order to attend to some requirements proposed to this proposed prototype construction. This use case implementation proves that ISO23247 shows that in fact, the standard framework could be implemented different applications due to its versatility.

## VI. Conclusions

Regarding the beginning of this work it was proposed a DTw based on the ISO23247 attending to the needs of a testing station system, and it was been achieved through the digitisation and real-time control of the physical system. A DTw can be seen has a manager that know all about everything that occurs in the real world, and take some steps to address future problems, by using already known technologies (IoT, CPS, VR, ...), and creating a linkage between several modules or layers. In this approach, the use case was focused on the linkage of several devices or things to a virtual monitoring environment, and the removal processing weight from the testing station hardware, into software. The monitoring of this station environment that was on focus, allows to probe the concept of a real-time monitoring system, in a visual model.

On the other hand, with this testing concept, it was possible to increase time between failures due to the reduction in the number of manual processes, and to reduce costs and complexity of the test station with regard to his need for equipment, through the sharing of resources to make tests such as: measuring instruments and signal generators. This new station concept, promoted the development of DTws that allow the implementation of monitoring, control and self-diagnosis functions of a testing station system.

Finally, this new test approaches developed and implemented within the scope of this project were all integrated into the same concept which ends up with a new test station based on a station concept, with the digital pairs of various elements captured by their respective DTws, whose equipment for generating RF signals is made through the use of low-cost SDRs. Leading us to improve a DTw architecture with free tools that follows the DTw framework ISO23247 like in the case of Eclipse Ditto. This makes an open framework that can be applied in generic use cases.

## Acknowledgments

This work has been supported by FCT - Fundação para a Ciência e Tecnologia within the R&D Units Project Scope: UIDB/00319/2020.

The author would also like to acknowledge FCT for the attributed Doctoral grant PD/BDE/150500/2019.

## References

[1] M. Grieves, “Digital twin : Manufacturing excellence through virtual factory replication this paper introduces the concept of a a whitepaper by dr . michael grieves,” *White Paper*, 2014.

[2] F. Tao, H. Zhang, A. Liu, and A. Y. Nee, “Digital twin in industry: State-of-the-art,” *IEEE Transactions on Industrial Informatics*, vol. 15, pp. 2405–2415, 2019.

[3] K. Y. H. Lim, P. Zheng, and C. H. Chen, “A state-of-the-art survey of digital twin: techniques, engineering product lifecycle management and business innovation perspectives,” *Journal of Intelligent Manufacturing*, vol. 31, pp. 1313–1337, 2020.

[4] S. Haag and R. Anderl, “Digital twin – proof of concept,” *Manufacturing Letters*, vol. 15, pp. 64–66, 2018. [Online]. Available: https://doi.org/10.1016/j.mfglet.2018.02.006

[5] L. Monostori, B. Kádár, T. Bauernhansl, S. Kondoh, S. Kumara, G. Reinhart, O. Sauer, G. Schuh, W. Sihn, and K. Ueda, “Cyber-physical systems in manufacturing,” *CIRP Annals*, vol. 65, no. 2, pp. 621–641, 2016.

[6] D. M. W. G. (WG15), “Digital twin framework iso 23247.” [Online]. Available: https://www.ap238.org/iso23247/

[7] Q. Qi, F. Tao, Y. Zuo, and D. Zhao, “Digital twin service towards smart manufacturing,” *Procedia CIRP*, vol. 72, pp. 237–242, 2018. [Online]. Available: https://doi.org/10.1016/j.procir.2018.03.103

[8] X. Zheng, F. Psarommatis, P. Petrali, C. Turrin, J. Lu, and D. Kiritsis, “A quality-oriented digital twin modelling method for manufacturing processes based on a multi-agent architecture,” vol. 51. Elsevier B.V., 2020, pp. 309–315.

[9] S. Singh, M. Weeber, and K. P. Birke, “Advancing digital twin implementation: A toolbox for modelling and simulation,” vol. 99. Elsevier B.V., 2021, pp. 567–572.

[10] T. U. Wien, I. of Electrical, E. Engineers, and I. I. E. Society, “Proceedings, 2020 25th ieee international conference on emerging technologies and factory automation (etfa) : Vienna, austria - hybrid, 08-11 september, 2020,” 2020.

[11] E. Negri, L. Fumagalli, and M. Macchi, "A review of the roles of digital twin in cps-based production systems," *Procedia Manufacturing*, vol. 11, pp. 939–948, 2017. [Online]. Available: http://dx.doi.org/10.1016/j.promfg.2017.07.198

[12] L. Thames and D. Schaefer, "Software-defined cloud manufacturing for industry 4.0," *Procedia CIRP*, vol. 52, pp. 12–17, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.procir.2016.07.041

[13] J. O'Sullivan, D. O'Sullivan, and K. Bruton, "A case-study in the introduction of a digital twin in a large-scale smart manufacturing facility," vol. 51. Elsevier B.V., 2020, pp. 1523–1530.

[14] Apache, "Apache kafka," 2022. [Online]. Available: https://kafka.apache.org/

[15] A. Rebmann, S. Knoch, A. Emrich, P. Fettke, and P. Loos, "A multi-sensor approach for digital twins of manual assembly and commissioning," vol. 51. Elsevier B.V., 2020, pp. 549–556.

[16] Y. Qamsane, C.-Y. Chen, E. C. Balta, B.-C. Kao, S. Mohan, J. Moyne, D. Tilbury, and K. Barton, "A unified digital twin framework for real-time monitoring and evaluation of smart manufacturing systems," 2019.

[17] M. Perno, L. Hvam, and A. Haug, "Implementation of digital twins in the process industry: A systematic literature review of enablers and barriers," *Computers in Industry*, vol. 134, 1 2022.

[18] A. Vogt, R. K. Müller, T. Kampa, R. Stark, and D. Großmann, "Concept and architecture for information exchange between digital twins of the product (cps) and the production system (cpps)," *Procedia CIRP*, vol. 104, pp. 1292–1297, 2021.

[19] D. Guerra-Zubiaga, V. Kuts, K. Mahmood, A. Bondar, N. Nasajpour-Esfahani, and T. Otto, "An approach to develop a digital twin for industry 4.0 systems: manufacturing automation case studies," *International Journal of Computer Integrated Manufacturing*, vol. 34, pp. 933–949, 2021.

[20] "Application and research on digital twin in electronic cam servo motion control system," *International Journal of Advanced Manufacturing Technology*, 2021. [Online]. Available: https://doi.org/10.1007/s00170-020-06553-7

[21] E. Yildiz, C. Møller, and A. Bilberg, "Virtual factory: Digital twin based integrated factory simulations," vol. 93. Elsevier B.V., 2020, pp. 216–221.

[22] L. Romer, S. E. Jeroschewski, and J. Kristan, "Leveraging eclipse iot in the arrowhead framework," 2020.

[23] V. Kamath, J. Morgan, and M. I. Ali, "Industrial iot and digital twins for a smart factory : An open source toolkit for application design and benchmarking." Institute of Electrical and Electronics Engineers Inc., 6 2020.

[24] B. Mishra and A. Kertesz, "The use of mqtt in m2m and iot systems: A survey," *IEEE Access*, vol. 8, pp. 201 071–201 086, 2020.

[25] R. Sousa, R. Miranda, A. Moreira, C. Alves, N. Lori, and J. Machado, "Software tools for conducting real-time information processing and visualization in industry: An up-to-date review," *Applied Sciences (Switzerland)*, vol. 11, 6 2021.

[26] M. RF, "Limesdr-usb," *https://myriadrf.org/*, 4 2022. [Online]. Available: https://wiki.myriadrf.org/LimeSDR-USB

[27] R. P. Foundation, "Raspberry pi 4 model b," *Raspberry Pi 4*, 4 2022. [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/

[28] U. Technologies, "Unity," *Our digital humans are evolving.*, 2022. [Online]. Available: https://unity.com/

[29] R. B. GmbH, "Xdk 110," *XDK - The Sensor X-perience*, 4 2022. [Online]. Available: https://developer.bosch.com/products-and-services/sdks/xdk/develop/c/technical-information

[30] G. R. project, "Gnuradio," 2022. [Online]. Available: https://www.gnuradio.org/

[31] I. MongoDB, "Mongodb," *Database*, 2022. [Online]. Available: https://www.mongodb.com/

[32] Solace, "Powering real-time event-driven," *Solace Broker*, 2022. [Online]. Available: https://solace.com/