Jurassic Park 2.0: A Reconfigurable Digital Twins Platform for Industrial Internet of Things

Eliseu Pereira, Gil Gonçalves
SYSTEC - Research Center for Systems and Technologies, Faculty of Engineering, University of Porto
Rua Dr. Roberto Frias, 4200-465 Porto, Portugal
Email: {eliseu, gil}@fe.up.pt

***Abstract*—Digital Twin (DT) emerged as an Industry 4.0 concept that reflects the behavior of physical entities and processes in the digital environment providing real-time information and insights. Typically, DTs are deployed on specific scenarios or components, virtualizing services and monitoring process variables. This extremely focused software implementation makes harder the adaptation and replication of DTs for new components with similar characteristics. This work focuses on upgrading an existent Internet of Things (IoT) platform, Jurassic Park, to permit the development of flexible DTs, facilitating the implementation and modification of IEC-61499 compliant Cyber-Physical Systems (CPS). The upgraded Jurassic Park is a web-based platform that manages IEC-61499 compatible devices executing the runtime environment (DINASORE). The platform was validated in 2 different scenarios, 1) in a distributed system composed of different DTs and 2) controlling a gripper of a robotic arm.**

***Index Terms*—Cyber-Physical Systems; Digital Twin; IEC-61499; Industrial Internet of Things; Virtualization.**

I. Introduction

The changeover to Industry 4.0 was introduced in the shop-floor of new information and communication technologies such as the Industrial Internet of Things (IIoT), Cyber-Physical Systems (CPS), among others [1]. The digitalization and virtualization of the physical devices provided to users and operators support for communicating with other devices, tracking errors, optimizing production and many other advantages [2]. Digital Twins (DTs) implement virtual models of physical entities to mirror the geometry and behavior of those entities in the digital domain [3]. With those digital models, real-time monitoring and control of the physical entities can be achieved [4]. Cyber-Physical Production Systems (CPPS) need to be easily reconfigurable since these systems stand out by responding quickly to changes in production and being flexible enough to introduce new functionalities according to the different needs of the production lines. One of the existent standards for the CPPS reconfiguration is the International Electro-technical Commission (IEC) 61499 standard [5], which allows the encapsulation of different functionalities in software modules, the so-called Function Blocks (FBs).

Several DTs proposals have been made recently; however, there is a lack of flexible solutions that are applicable in the production lines [6]. Nowadays, the creation of a DT is a centralized process oriented toward the device intended to be reflected in the digital world. This makes the DT not very flexible. In a complex system with diverse physical entities fulfilling different functionalities, reusing a DT is a critical process and will require increased computational and human effort. This limitation could be solved by using standards for the deployment of reconfigurable CPPS, such as the IEC-61499 standard. However, its integration with the DT concept is still limited and barely explored by the scientific community. Some of the most relevant limitations are the difficulty of managing complex systems with many software modules and the lower flexibility and adaptability of the systems to new tools at the software and hardware level [7].

The main objective of this work was to upgrade an IoT platform [1][8], Jurassic Park, to enable the development of flexible DT applications integrated with the physical devices (executing the DINASORE [9] runtime environment). The DT solution will include 1) a monitoring module, 2) a control module, and 3) a visualization module. The monitoring module is intended to be the processing component of the information that arrives from the physical environment. The main purpose of this module is to understand the behavior of physical entities and store that information in the digital world. The DT control module aims to be the response feature of the DT solution. Through this module, the user can act on the physical entities, thus making the system an analysis component and acting component with the purpose of modifying the system. The general idea behind the DT visualization module is to allow the management of the DTs from the application system. The solution was validated in 2 different scenarios, 1) to monitor and control a distributed CPS, and 2) to control a robotic arm gripper.

The remainder of the paper is structured as follows: in Section II, we present the literature review, Section III details the architecture of the developed solution, Section IV exposes the implementation carried out and Section V presents the main tests and results made to validate the DT solution developed. The discussion of upgraded platform characteristics is detailed in Section VI, and the conclusions and future work are presented in Section VII.

II. Literature Review

Digital Twins (DTs) can adopt different methodologies, concepts, and technologies, providing simulation, monitoring, or optimization capabilities. For that, we initially present the main technologies used as the basis for implementing the DT

TABLE I: Key enabling technologies for DTs

Tools for the Physical World	Tools for DT modeling	Tools for DT data	Tools for DT services	Tools for connections in DTs
VisionPro, Predix, ROS, Matlab, other software	Simulink, Stella, Ansys Twin Builder other software	MySQL, MongoDB, Beacon, other software	Simulink , Labview, Azure IoT, Matlab, other software	Siemens' MindSphere, Predix, other software

platform, and after the related work presents a description of the more relevant DT implementations.

A. Background

Several enabling technologies allow the implementation of CPSs [10], for different purposes, like device management, data collection, workflow orchestration, among others. The DT solution stacks different technologies, having as a basis an environment with essential capabilities, like reconfiguration of software [11], data interoperability, or communication transparency [12].

The DINASORE [9] is a distributed platform that enables the pre-processing of data in edge devices using Python-based FBs. This platform allows the construction of a wide variety of FBs by the user, with different goals, like controlling drivers, integrating sensors, and applying processing techniques, among other functionalities. The DINASORE uses the 4DIAC-IDE as a graphical user interface to orchestrate the FB pipelines deployed in the distributed CPS.

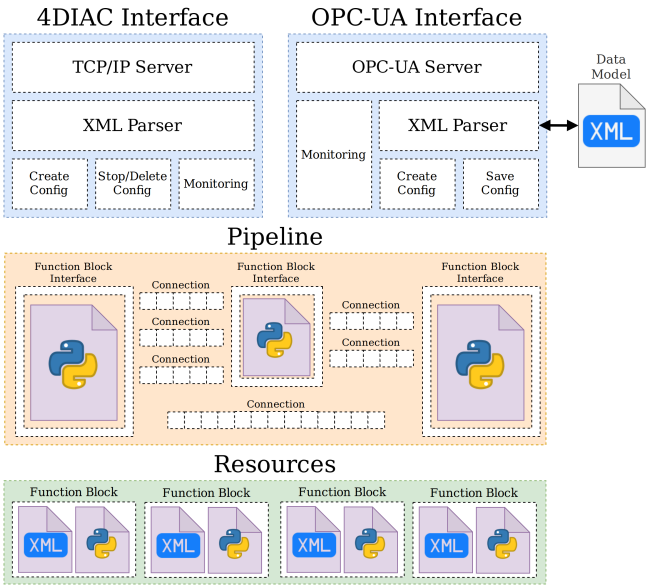


Fig. 1: DINASORE architecture.

The 4DIAC-IDE [13] is a tool developed by Eclipse that allows the orchestration of a workflow of FBs, after being deployed in a distributed system. With this tool, it is possible to draw the FBs distributed systems through a graphical interface and then deploy it into the devices connected to the network.

The Jurassic Park [8], in Figure 2, is a web-based application that acts as a centralized repository of FBs, allowing the creation and management of FBs. Jurassic Park comprises two components: the backend and the web application. This application is integrated with the DINASORE and the 4DIAC-IDE, allowing the complete deployment and management of distributed systems through the IEC-61499 standard.

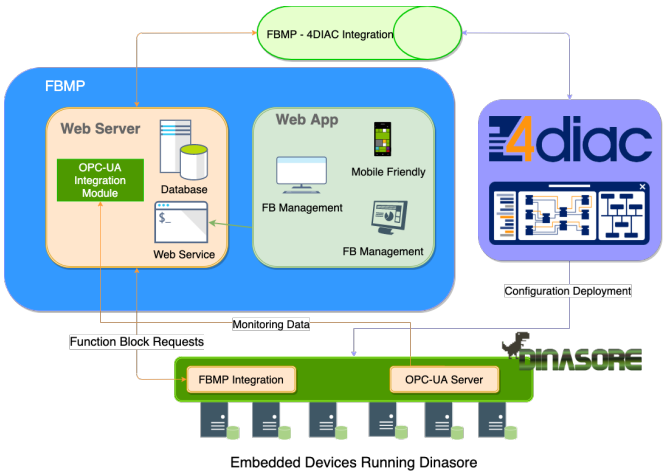
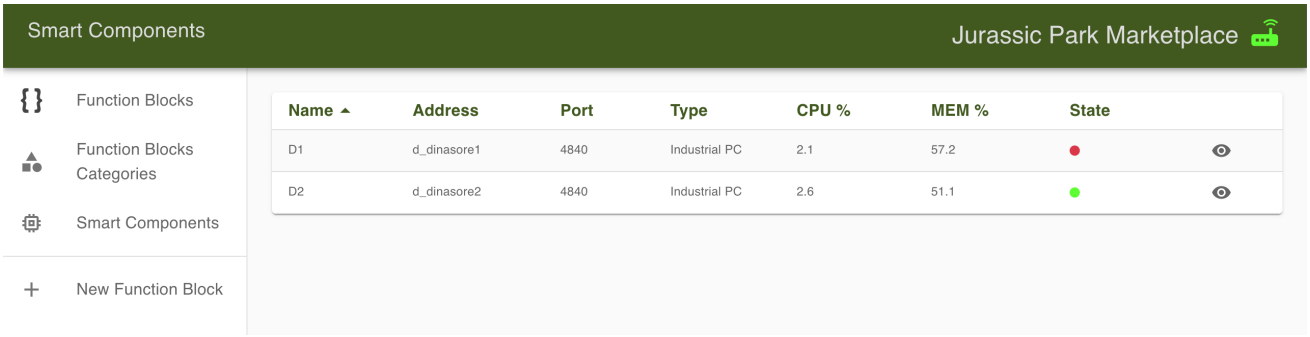


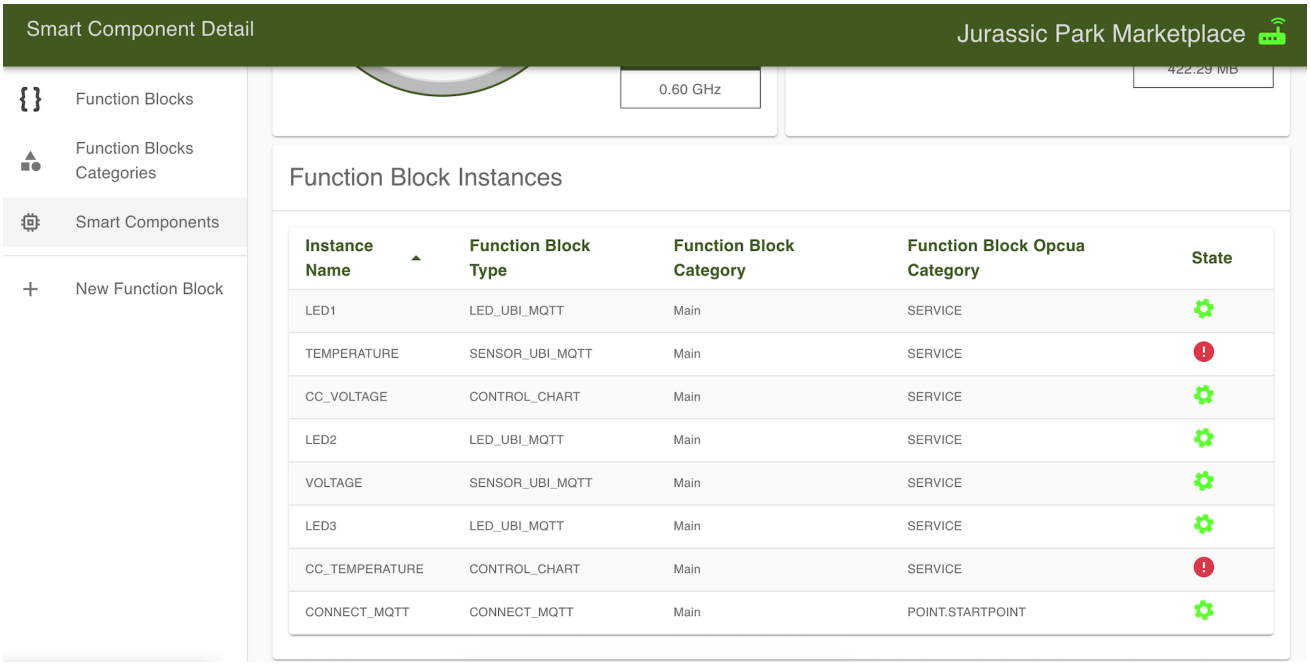
Fig. 2: Original Jurassic Park architecture.

The Web Server contains the application and database servers and should run on the machine where the 4DIAC-IDE is running. The Web Server runs a Node Application that exposes an API composed of Web Services to manage the creation, edition, deletion, and retrieval of the FBs. This application also functions as an OPC-UA client connected to the different OPC-UA Servers running on the embedded devices so it can track them in real time and serve this information to the Web App through a TCP WebSocket connection. The Web Server also runs an FTP Application to serve the needed files to the different embedded systems running on the network. The Web Server runs on the machine where the instance of the 4DIAC-IDE is running, as this integration is made through the machine file system. The Backend also comprises an instance of a MySQL Server Database, which stores all the meta-information related to the software modules.

The Web App used by the Jurassic Park administrators and all the users needs to track the devices in real-time. This application allows, in a friendly way, all the operations required to manage the function blocks (creation, edition, deletion, and retrieval of modules). These operations are accomplished by calling the web services exposed by the Web Server, specifically in the Node Application. The CPS devices are running the DINASORE RTE, which integrates with Jurassic Park using a communication bridge implemented in HTTP and FTP.



(a) List of existent DINASOREs.



(b) DINASORE function blocks state.

Fig. 3: Jurassic Park graphical user interface.

B. Related Work

As identified in [14], currently, there is a huge need to provide on-demand manufacturing services through Industrial Internet of Things (IIoT) networks. These services are easily achieved with the implementation of cloud-based manufacturing solutions. The literature review presented in [15] shows a set of cloud solutions available to help in the implementation of Digital Twins (DTs), in particular, the Microsoft Azure IoT solution and the Amazon Web Services (AWS) IoT solution. When large industrial companies began to realize that the implementation and consequent application of DTs could significantly impact the efficiency of their production lines, they began their initiative to develop their tools [16].

In [17], the authors present DT use cases in the industrial environment, where Siemens presents two DT implementations, one for the power system and the other for the wastewater plant. General Electric has not only developed a DT for a wind farm. Still, this implementation has also proved that a wind farm should be operated, developed, and maintained more efficiently. British Petroleum has developed a DT of oil and gas facilities in more restricted areas. The air vehicle manufacturer Airbus has used DT-based solutions to monitor its production lines and to optimize its operations. Going into a more specific example of a DT implementation in the industry context, [18] presents a DT within a hollow glass manufacturing production line. The authors of this implementation argue that this work has allowed them to realize that digital simulations greatly impact how production lines can be optimized since they reflect their behavior in the real world.

The authors in [19] present a DT solution in the commercial greenhouse sector. The article highlights the need for commercial greenhouse production systems to become more energy-efficient while maintaining sustainable production. Therefore,

the DT solution mentioned was proposed to provide a control and monitoring feature of the production flow of a greenhouse production system. The authors of this article also defend two essential ideas. One of them is that, with DT implementations, it is possible to turn the industrial environment into more energy and climate-friendly, leading to reduced costs within the production lines. The other idea is that, nowadays, the industrial environment requires developed generic DT frameworks capable of controlling the processes accurately and responding to changes in the orders of the production lines.

III. Architecture

The DT platform was conceptualized to structure multiple DTs, allowing their monitorization and control. This concept is composed of three main objects, 1) the digital twin, which is at the top of the architecture and is responsible for controlling the functionalities of the physical equipment, 2) the functionality, which describes the DT characteristics, mainly the monitoring system variables and the functions to trigger the execution of specific actions, and 3) the device, that reflects the physical entity in the digital world, like sensors, machines, etc. The DT platform developed is embedded in Jurassic Park, taking advantage of some of its features. Figure 4 shows the DT proposed solution composed of three components, the DT visualization module, the DT monitoring module, and the DT control module.

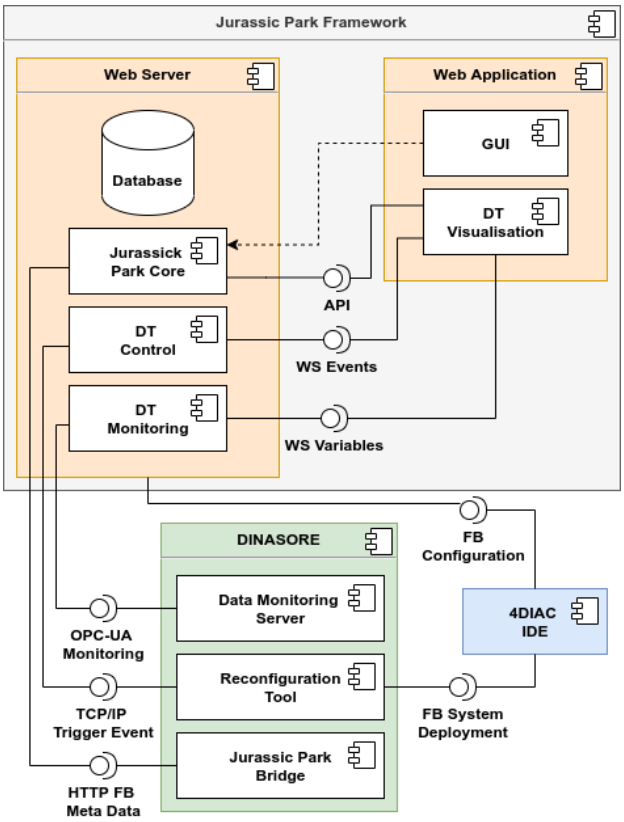


Fig. 4: DT architecture integrated with Jurassic Park and DINASORE.

The DT visualization component (presented in Section IV-C) allows the interaction between the DT platform and the user. This component is incorporated into the web application of Jurassic Park. The main functionalities of this visualization component focus on the management of the DTs. Additionally, this component will be responsible for triggering the monitoring and control requests from the DTs. The DT monitoring module (presented in Section IV-D) is intended to serve as a processing tool for the information that arrives from the physical entity to the digital one and serves the information to the user through the graphical interface. The component is incorporated into the web server of Jurassic Park using the Open Platform Communications - Unified Architecture (OPC-UA) communication protocol as a data source to collect information from the physical world. The monitoring module collects data from the FBs variables and events running on the DINASOREs connected to the platform, allowing the user to filter the ones to monitor. The DT control component (presented in Section IV-E) permits triggering functionalities over the physical entities connected to the platform. The component uses the variables and events from the FBs on the devices connected to the platform and assigns them a trigger feature. In this way, the user will have the opportunity to manipulate the events of the FBs. As in the case of the monitoring component, the control component is also incorporated into the web server.

The backend component (webserver) is a complex module that uses several technologies to consolidate the integration of all the services it supports. This component is responsible for the HTTP API that manages the FBs and the real-time communication of the devices connected to the platform and the database. One of the most relevant technologies used in the backend is the Node.js framework, which integrates several backend components such as the API, the real-time communication with the web application, and the real-time communication with the DINASOREs, through the OPC-UA communication protocol. The backend component uses MySQL as database. The backend also includes a File Transfer Protocol (FTP) server that allows the download of FBs by the DINASOREs.

The frontend component (web application) is the platform's graphical user interface, and it was developed using React. This component is responsible for the page components (buttons, boxes, tables, among other User Interface (UI) elements), which are customized according to the existent resources in the CPS. The automatic update of the information is performed using the Socket IO library, which supports real-time communication with the backend. It avoids the need to refresh the web page to update the information.

IV. Implementation

The upgrade of the Jurassic Park platform followed an agile methodology of development using git that enabled the usage of stable versions for deployments while the platform was upgraded. As follows, this section presents the main Jurassic Park components as well as the top upgrades, mainly the Digital Twin features.

A. Jurassic Park Backend

As mentioned before, Jurassic Park's backend is divided into several components, described in Figure 5. The API component uses a JavaScript package to handle and correctly route the HTTP requests: Express. This component is a composition of other components called API modules. Each of these modules handles a list of requests, and the set of all modules forms the full HTTP Rest API. The following items list the available API operations:

- **GET /function-block/**: Returns a list of the FBs available in Jurassic Park.
- **GET /function-block/:type**: Endpoint responsible for giving the information of the FBs to the DINASOREs connected.
- **POST /function-block/**: Endpoint responsible for creating a new FB in Jurassic Park.
- **PUT /function-block/:id**: Endpoint responsible for editing a specific FB identified by the FB id.
- **DELETE /function-block/:id**: Endpoint responsible for deleting a specific FB identified by the FB id.
- **GET /function-block-category/**: Returns a list of the FB categories available in the platform.
- **POST /function-block-category/**: Endpoint responsible for creating a new FB category in Jurassic Park.
- **PUT /function-block-category/:id**: Endpoint responsible for editing a specific FB identified by the FB category id.
- **DELETE /function-block-category/:id**: Endpoint responsible for deleting a specific FB category identified by the FB category id.
- **POST /smart-component/**: Endpoint responsible by the DINASORE to announce the connection to Jurassic Park.

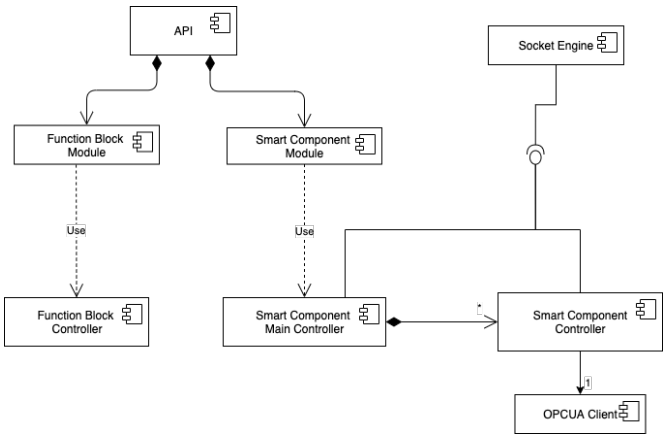


Fig. 5: Jurassic Park backend components diagram.

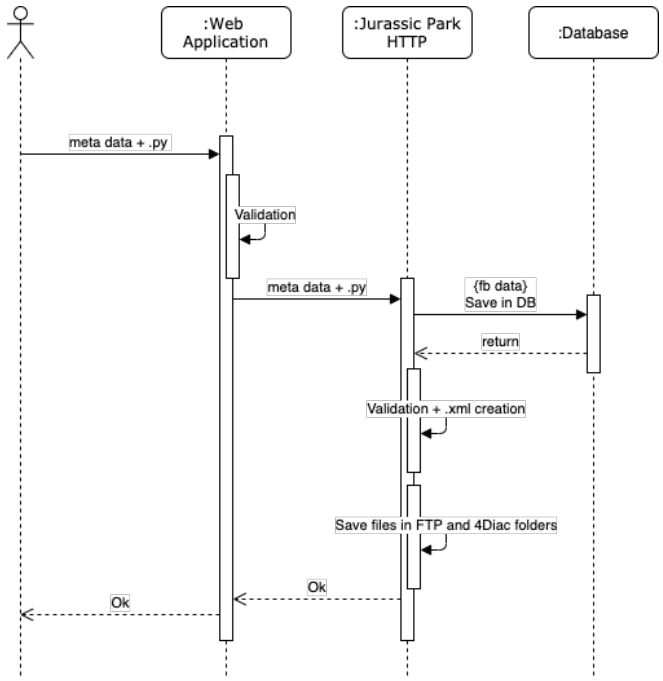
To establish real-time communication between the Web Application running in a browser and the Jurassic Park backend, there must be a permanent communication channel between these two entities. To achieve this, the approach was to use WebSocket technology. The Socket Engine component provides an interface used by both the Smart Component Main Controller and the Smart Component Controller to send data to the connected sockets functioning as a bridge between the controllers and the client connected and running the Web Application on a browser.

The Function Block Controller and the Smart Component Controller were developed to control a specific application area. The first one controls the CRUD Operations of the FBs and FB Categories, and the second controller, a more complex one, controls the DINASORE state at any given moment. The Function Block Controller was developed as a singleton class with methods to create, update, delete and retrieve function blocks and categories. The Smart Component Controller is a singleton class like the Function Block Controller. However, this controller will contain a list of Smart Component Controllers (sub-components), one for each connected smart component device.

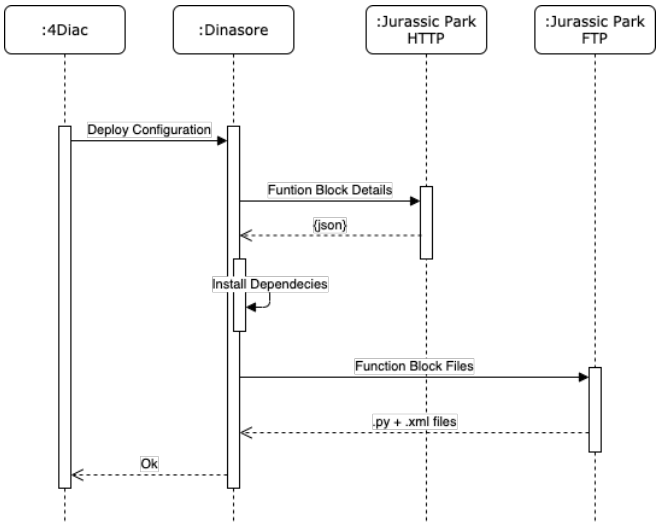
B. Jurassic Park Frontend

As previously mentioned, the Jurassic Park frontend uses React to build the web application. It is a JavaScript library developed by Facebook that switches from the imperative programming style to a declarative one. This way, the application becomes much more modular, allowing the reusage of some components so they can act as a sort of template. The web application structure has been divided into the following categories; each category is a folder containing other subfolders or the actual implementation file in React.

- **Components**: This category contains the basic UI components of the application; these components were subdivided into the Function Block folder, with all the UI elements to manage the Function Blocks, the Smart Components folder, to monitor the Smart Components connected to the network, the Function Block Categories folder, containing the UI elements to allow the users to manage the Function Block Categories and the templates folder, containing the UI elements meant to be reused in the other components like charts, navigators and tables.
- **Services**: This category of files was created to accommodate the services needed to communicate with external system components, in this case, with the Jurassic Park backend. Inside this category, there are two subcategories; the HTTP module, with all the functions to fetch and send data to the backend; and the Socket module, with functions to communicate via TCP sockets with the backend, uses Socket IO external package.
- **State**: The paradigm of declarative programming and React, in particular, is straightforward and based on a UI change in reaction to an internal change in the component state. This state can be a simple variable like a string, number, array, or even a set of these. Once changed, this state triggers an update in all the UI elements dependent on that state. However, for complex applications with more than one component, there should be a place to centralize all the state variables used by different components. Therefore, the functions built inside this module can extract or change the state, so any component that needs to change this centralized state can only do that



(a) Function block creation sequence diagram.



(b) Function block download sequence diagram.

Fig. 6: Sequence diagrams for the creation (a) and download (b) of function blocks.

by using this API. For example, the Function Block categories fetched from Jurassic Park backend should be centralized here so the Function Block components and the Function Block Categories components can use them to build their respective UI.

Using these categories as a basis, Jurassic Park includes web pages used for interacting with the user, through a browser, on a desktop, or mobile device. The main pages of the application are in the following list, where the first three enable the CRUD operations.

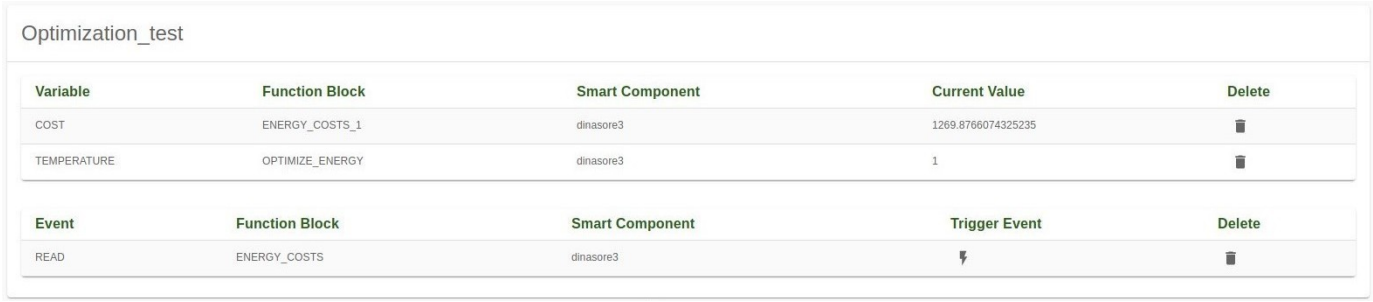
- **Function Block List**: The first page is the list of all the FBs available in the marketplace. In this list, it is possible to see all the main characteristics of each FB, including its type, description, category, and general category.
- **New Function Block**: The second page uses a form to create a new FB. This form contains all the fields for creating a new FB, including all the mechanisms to ensure the correct creation of the FB, ensuring that it follows the correct structure. Those validations include verifying that there can't be more than one FB with the same name or that the user uploaded the FB Python file.
- **Edit Function Block**: The third page allows the edition of a FB. In this page, the FB form had already been built, so it was just a matter of reusing it. The difference is how it is loaded when the form is loaded; the data of an already existing FB is passed on.
- **Smart Component List**: This page lists the DINASOREs connected to the network. This list shows the information on each device that has already established a connection to Jurassic Park. Hence, a DINASORE can be connected or disconnected, as the green or red light indicates in Figure 3a. The list also shows, for each component, its name, address, port, CPU percentage current usage, and memory used.
- **Smart Component Detail**: This page shows the monitored information of a DINASORE in Figure 3b. The monitored information includes the FB executing in that DINASORE, detailing each FB state, where the green icon means the FB is running correctly, and when it goes to red, it stops working. Like the previous component, this page is dynamically updated, making it a live dashboard to monitor the DINASORE.
- **Function Block Category**: This page lists the available FB categories. For each category created, the application shows the category's name and a list of FBs in that category. The categories managed here are available in the FB form.

C. Digital Twin GUI

The DT visualization component allows the user to create and manipulate DTs. The platform allows users to group the devices they want to monitor and control into a category called DT. Subsequently, they can associate that DT with a general functionality. Therefore, to describe the page elements in a more user-focused way, we list below the different pages implemented to fulfill the requirements of the DT component.

The new DT page allows the user to add new DTs with a specific name and to associate them with respective DINASOREs. The page has a menu for creating a new DT; in this menu, the user has access to all the devices communicating

(a) DT monitoring web page.



(b) Functionality details web page.

Fig. 7: Digital Twin GUI, including the monitoring (a) and functionalities details (b) web pages.

with Jurassic Park in real-time. Therefore, to create a new DT, the user only has to complete the field to insert the name of the new DT, open the list of available devices, and select those he wants to associate with. It should be noted that the user can choose more than one DINASORE. Additionally, the user can only create the new DT if he has chosen at least one DINASORE.

The Digital Twin monitoring page, in Figure 7a, performs all the management of the DT. On this page, the user can observe the currently active functionalities with the respective DT. Besides that, on this page, the user can create other functionalities and associate a DT capable of observing the variables and/or events of interest. Looking first at the feature of the page concerning the creation of a new functionality, the user can choose the name of the functionality they want to add to the monitoring platform. After choosing the name, the user will have to choose one DT from the range of available DTs previously created on the New Digital Twin page, which they want to associate with the new functionality. Through this page, the user will have at his disposal a table, listing all the functionalities currently active and a set of features. The *Details* feature allows the user to view in detail the information collected from the variables and events currently being monitored. The *Add Details* feature enables users to choose the variables and events they want to monitor on a given FB. With the *Edit* feature, the user can edit the name of the functionality. In addition to being able to edit and manipulate the variables and events that each functionality will monitor, the user can also delete the available functionalities they want with the *Delete* feature.

The interface redirects to a new page whenever the user presses the functionality details button. The functionality details page, in Figure 7b, has two different tables, one for monitoring variables and another for triggering events. Note that the page only displays the variables and events that have been previously selected on the DT monitoring page. Thus, the user has information organized either by variables or by events at their disposal. Regarding the variables monitoring, the user can observe in the table some information about the variable, such as the variable name, the FB where the variable is being monitored, the device that is allocating, and the variable's current value.

Additionally, the user can delete the variables they no longer wish to monitor by clicking on the delete button in each row of the table. The event monitoring table does not differ much from the variable monitoring table. It is also possible to see the name of each event selected by the user, the associated FB and the DINASORE where it is being mapped. However, unlike the variable monitoring table, the event table has a button that allows the user to trigger an action on the associated event. When the user presses the trigger event button, Jurassic Park automatically sends a request to the server to execute the event on the associated FB. The user also has a button on the event table that allows them to delete events they no longer wish to observe.

D. Digital Twin Monitoring

The monitoring component of the DT platform is responsible for requesting and collecting information on each DT,

which is then made available in the DT visualization component. For that, the user should create the different DTs and functionalities and define the monitored devices, variables, and events. With the previous objects created, the following integration is to automatically receive real-time feedback on the values of the monitored variables. Initially, the request is sent via WebSockets from the web application, where a persistent listener waits for feedback from the monitored variables. On the server side, the DINASORES communicate using an OPC-UA client. When the monitored variables present a change, the OPC-UA client automatically identifies it. Then the data is sent back to the backend controller, which ensures the notification of the new value to the web application, waiting for the information through the listener function. When the user no longer wants to observe the variables, an event to cancel the subscriptions is sent to the OPC-UA client.

E. Digital Twin Control

The DT control component allows users to trigger events in certain FBs running on specific DINASOREs. After interacting with the UI element, a message containing the triggered event's information is sent to the backend component. Having the information regarding the event in the backend, a function sends the action to the device in question, using Transmission Control Protocol/Internet Protocol (TCP/IP) sockets. For that, the function uses as input the event information, in particular the FB name, event name, device IP address, and port. When the DINASORE receives the message, it pushes an event on the specified FB and executes the triggered functionality.

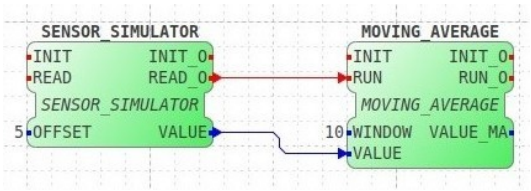
V. Experiments and Results

The experiments performed to validate the DT component of the platform focused on two use cases that allow testing of the different implemented features. The experimented use cases are 1) the monitorization of simulated sensors and the optimization of energy in a distributed CPS and 2) the manipulation of a robotic arm gripper.

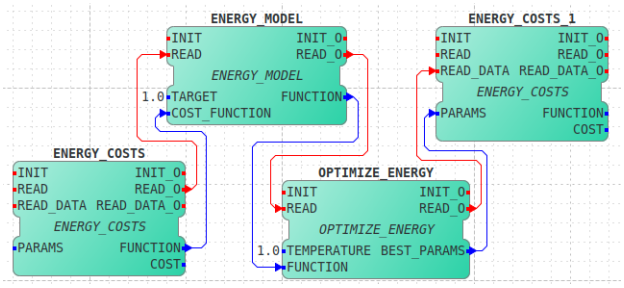
A. Monitoring and optimizing a distributed CPS

The first validation scenario is a distributed system with different devices connected to the platform. This experiment consisted of connecting a set of raspberry pis to the platform, which are responsible for executing two different FBs pipelines/workflows, one for sensing purposes and the other for energy optimization. This set of FBs pipelines allows the monitoring of variables and trigger of events in a distributed system, allowing the validation of the previously explained monitoring and controlling features of the DT platform.

The first raspberry pi was used to validate the variable monitoring component, using a small FBs workflow that simulates a sensing system, present in Figure 8a. The FBs composing the workflow are 1) the SENSOR_SIMULATOR, responsible for generating a random value to simulate the value of a sensor, and 2) the MOVING_AVERAGE, which calculates the average of the last N values. After the deployment of the workflow, the next step was to create a DT for the sensing



(a) Sensor simulation FBs workflow.



(b) Energy optimization FBs workflow.

(c) Physical scenario, including a wireless router and 2 raspberry pis.

Fig. 8: Distributed CPS composition with a sensorization (a) and optimization (b) workflows and the physical scenario (c).

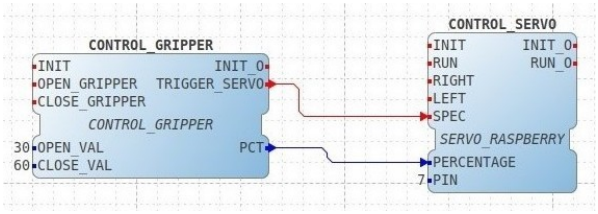
component associated with the raspberry pi that executes the DINASORE. Then, we created a functionality that includes the following monitoring variables, 1) the variable VALUE (variable containing the simulated sensor data) from the SENSOR_SIMULATOR FB and, 2) the variable VALUE_MA (variable containing the moving average which is calculated) from the MOVING_AVERAGE FB.

The second raspberry pi hosts a set of FBs that optimize the energy consumption of a process, present in Figure 8b. This FB pipeline allows the validation of the event-triggering process through the DT platform and, consequently, the monitoring of variables linked to that event. The FBs composing the workflow are 1) the ENERGY_COSTS, which specifies the function for energy costs regarding velocity and power, 2) the ENERGY_MODEL, which is the model that allocates the energy consumption, using as input the energy costs, and 3) the OPTIMIZE_ENERGY, which optimizes the given function, through the use of the Dual Annealing algorithm. The DT associated with the optimization workflow includes

functionalities that contain the following variables and events: 1) the COST variable (optimized energy costs) present in the ENERGY_COSTS_1 FB, and 2) the READ event (triggering optimization) of the ENERGY_COSTS FB. With this variable and event combination, we can trigger an event via the platform, sent to the raspberry pi, that executes the optimization.

B. Creating a Digital Twin for a Gripper

The second scenario focuses on creating a DT for a robotic arm gripper, validating the physical component's control and monitoring features. The component used was a 3D-printed gripper using a raspberry pi as a controlling system for the servo motor. This experiment consisted in controlling the gripper through the developed DT platform by manipulating the event trigger to make the gripper arm open and close according to the event, as shown in Figure 9b and 9c. The FB workflow that allows the gripper's control, present in Figure 9a, is composed of the FBs: 1) the CONTROL_GRIPPER, responsible for identifying whether there is a request to open or close the gripper, depending on which, the FB sends the corresponding percentage to the output (PCT), and 2) the CONTROL_SERVO, which receives as input the percentage and then updates the general-purpose input/output (GPIO) that controls the servo motor with the corresponding value.



(a) FBs workflow to control the gripper.

(b) Gripper open.

(c) Gripper closed.

Fig. 9: Results obtained with the gripper scenario, including the FBs workflow (a) and the physical gripper (b) and (c).

The DT created intends to monitor and control the gripper manipulation. For this use case, two distinct events and a variable (to evaluate whether the events are being correctly triggered) have been added to the functionality associated with the DT. The selected events to control the gripper were the OPEN_GRIPPER and CLOSE_GRIPPER events. To check whether the control on the DT platform is functional, the gripper would have to open when the user pressed the release button. If the close button were pressed, the gripper would have to execute the closing movement. The execution of both movements was validated visually and also through the monitoring variable (percentage), which confirms if the gripper moves according to the monitored percentage.

VI. Discussion

The discussion of the results obtained in the different scenarios will focus on the verification of the upgraded Jurassic Park platform in terms of compliance with IoT requirements [2]. The IoT requirements enable developers and researchers to validate if the IoT solutions comply with current standards and align with modern CPS's goals. The upgraded Jurassic Park platform passed through the evaluation in 5 different requirements, i.e., scalability, reliability, interoperability, timing, and reconfigurability.

- **Scalability**: The platform presents scalability in the number of devices connected, DT representations, monitored variables, and CPPS functionalities.
- **Reliability**: In terms of reliability, the platform was validated in different scenarios, replicating conditions from a real CPS, like distributed communication, execution of devices with low computational resources, and actuation into physical assets.
- **Interoperability**: The interoperability of a CPS enables transparent and fluid communication across distributed devices. The validation scenario with distributed devices (2 raspberry pis) enabled the validation of that requirement.
- **Timing**: During the platform execution, the response times of the variables monitorization and functionalities triggering are near real-time, i.e., approximately between 1 and 2 seconds of response time.
- **Reconfigurability**: As described before, the upgraded Jurassic Park platform enables the quick and easy modification of any parameter of the CPPS, e.g., the monitored FBs or the devices connected to the platform. The GUI has particular pages for the modification

With these findings, the upgraded Jurassic Park platform presents a large set of characteristics and enablers to their deployment in a real CPS. Those characteristics and achieved requirements prove not only the capacity for the platform to execute in a CPS but also facilitate the day-to-day life of the CPS managers and industry operators because it makes easier the traceability of the processes and makes the systems more transparent to humans.

VII. Conclusion

In conclusion, the main objectives the Jurassic Park 2.0 were to upgrade it by implementing a flexible and reconfigurable DT solution capable of increasing the monitoring capacity and enabling the remote control of a CPS. The components developed and upgraded, mainly the DT visualization component, the DT monitoring component, and the DT control component, permitted the CPS to accomplish the described attributes, like DT reconfiguration. The solution was integrated with a mature stack of technologies, including DINASORE. The experiments

were performed to support the usability and flexibility of the web-based platform. Finally, it is essential to highlight that one of the most significant contributions of this project was to develop a platform that, given its flexibility and configurability, can be easily integrated into the industrial sector and supports the IEC-61499 standard.

As future work, one of the main goals will be the storage of data generated by the DT variables, considering data volume constraints in terms of storage and data flow/rate. On top of this unstructured database, it will be possible to implement predictive algorithms to forecast and optimize [20][21] the behavior of DTs. Additionally, the entire platform will be integrated into an industrial scenario composed of different machines, including the integration of sensors, simulation of processes, and optimization of resources.

ACKNOWLEDGMENT

INDTECH 4.0 - New technologies for intelligent manufacturing. Support on behalf of IS for Technological Research and Development (SI a Investigacao e Desenvolvimento Tecnologico). POCI-01-0247-FEDER-026653

The authors thank the students involved for helping in the development of the platform, in particular Maria Arieiro.

REFERENCES

[1] E. Pereira, M. Arieiro, and G. Gonçalves, "Reconfigurable digital twins for an industrial internet of things platform," in *INTELLI 2022, The Eleventh International Conference on Intelligent Systems and Applications*, 2022, pp. 30–35.

[2] L. Antão, R. Pinto, J. Reis, and G. Gonçalves, "Requirements for testing and validating the industrial internet of things," in *2018 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2018, pp. 110–115.

[3] C. Cimino, E. Negri, and L. Fumagalli, "Review of digital twin applications in manufacturing," *Computers in Industry*, vol. 113, p. 103130, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166361519304385

[4] K. Ding, F. T. S. Chan, X. Zhang, G. Zhou, and F. Zhang, "Defining a Digital Twin-based Cyber-Physical Production System for autonomous manufacturing in smart shop floors," *International Journal of Production Research*, vol. 57, no. 20, pp. 6315–6334, 2019.

[5] K. Thramboulidis, "Iec 61499 in factory automation," in *Advances in Computer, Information, and Systems Sciences, and Engineering*, K. Elleithy, T. Sobh, A. Mahmood, M. Iskander, and M. Karim, Eds. Dordrecht: Springer Netherlands, 2006, pp. 115–124.

[6] Y. Fan, J. Yang, J. Chen, P. Hu, X. Wang, J. Xu, and B. Zhou, "A digital-twin visualized architecture for flexible manufacturing system," *Journal of Manufacturing Systems*, vol. 60, pp. 176–201, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S027861252100114X

[7] G. Lyu and R. W. Brennan, "Towards IEC 61499-Based Distributed Intelligent Automation: A Literature Review," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 4, pp. 2295–2306, 2021.

[8] J. Pedro Furriel, E. Pereira, J. Reis, and G. Gonçalves, "Jurassic park - a centralized software modules repository for iot devices," in *2021 10th Mediterranean Conference on Embedded Computing (MECO)*, 2021, pp. 1–4.

[9] E. Pereira, J. Reis, and G. Gonçalves, "Dinasore: A dynamic intelligent reconfiguration tool for cyber-physical production systems," in *Eclipse Conference on Security, Artificial Intelligence, and Modeling for the Next Generation Internet of Things (Eclipse SAM IoT)*, 2020, pp. 63–71.

[10] Q. Qi, F. Tao, T. Hu, N. Anwer, A. Liu, Y. Wei, L. Wang, and A. Nee, "Enabling technologies and tools for digital twin," *Journal of Manufacturing Systems*, vol. 58, pp. 3–21, 2021, digital Twin towards Smart Manufacturing and Industry 4.0.

[11] C. Zhang, W. Xu, J. Liu, Z. Liu, Z. Zhou, and D. T. Pham, "A reconfigurable modeling approach for digital twin-based manufacturing system," *Procedia CIRP*, vol. 83, pp. 118–125, 2019, 11th CIRP Conference on Industrial Product-Service Systems. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212827119304469

[12] E. Pereira, R. Pinto, J. Reis, and G. Gonçalves, "Mqtt-rd: A mqtt based resource discovery for machine to machine communication," in *Proceedings of the 4th International Conference on Internet of Things, Big Data and Security - IoTBDS,*, INSTICC. SciTePress, 2019, pp. 115–124.

[13] A. Zoitl, T. Strasser, and A. Valentini, "Open source initiatives as basis for the establishment of new technologies in industrial automation: 4diac a case study," in *2010 IEEE International Symposium on Industrial Electronics*, 2010, pp. 3817–3819.

[14] Y. Lu and X. Xu, "Cloud-based manufacturing equipment and big data analytics to enable on-demand manufacturing services," *Robotics and Computer-Integrated Manufacturing*, vol. 57, pp. 92–102, 2019.

[15] T. Catarci, D. Firmani, F. Leotta, F. Mandreoli, M. Mecella, and F. Sapio, "A conceptual architecture and model for smart manufacturing relying on service-based digital twins," in *Proceedings - 2019 IEEE International Conference on Web Services, ICWS 2019 - Part of the 2019 IEEE World Congress on Services*. IEEE, Piscataway, NJ, USA, 2019, pp. 229–236.

[16] A. Fuller, Z. Fan, C. Day, and C. Barlow, "Digital Twin: Enabling Technologies, Challenges and Open Research," *IEEE Access*, vol. 8, pp. 108 952–108 971, 2020.

[17] 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, no. 4, pp. 2405–2415, 2019.

[18] H. Zhang, Q. Liu, X. Chen, D. Zhang, and J. Leng, "A Digital Twin-Based Approach for Designing and Multi-Objective Optimization of Hollow Glass Production Line," *IEEE Access*, vol. 5, pp. 26 901–26 911, 2017.

[19] D. Anthony Howard, Z. Ma, J. Mazanti Aaslyng, and B. Nørregaard Jørgensen, "Data architecture for digital twin of commercial greenhouse production," in *2020 RIVF International Conference on Computing and Communication Technologies (RIVF)*, 2020, pp. 1–7.

[20] E. Pereira, J. Reis, G. Gonçalves, L. P. Reis, and A. P. Rocha, "Dutch auction based approach for task/resource allocation," in *Innovations in Mechatronics Engineering*, J. Machado, F. Soares, J. Trojanowska, and S. Yildirim, Eds. Cham: Springer International Publishing, 2022, pp. 322–333.

[21] R. Rossini, G. Prato, D. Conzon, C. Pastrone, E. Pereira, J. Reis, G. Gonçalves, D. Henriques, A. R. Santiago, and A. Ferreira, "Ai environment for predictive maintenance in a manufacturing scenario," in *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2021, pp. 1–8.