applied sciences

MDPI

Article

QHAR: Q-Holonic-Based ARchitecture for Self-Configuration of Cyber–Physical Production Systems

Douha Macherki [1,2,*], Thierno M. L. Diallo [1,*], Jean-Yves Choley [1], Amir Guizani [2], Maher Barkallah [2] and Mohamed Haddar [2]

[1] Quartz EA7393, Supméca, 3 Rue Fernand Hainaut, 93400 Saint-Ouen, France; jean-yves.choley@isae-supmeca.fr

[2] LA2MP, ENIS, Km 4 Route de la Soukra, Sfax 3038, Tunisia; amir.guizani@live.fr (A.G.); maher.barkallah@enis.tn (M.B.); mohamed.haddar@enis.tn (M.H.)

* Correspondence: douha.macherki@isae-supmeca.fr (D.M.); thierno.diallo@isae-supmeca.fr (T.M.L.D.)

Abstract: Production systems must be able to adapt to increasingly frequent internal and external changes. Cyber-Physical Production Systems (CPPS), thanks to their potential capacity for self-reconfiguration, can cope with this need for adaptation. To implement the self-reconfiguration functionality in economical and safe conditions, CPPS must have appropriate tools and contextualized information. This information can be organized in the form of an architecture. In this paper, after the analysis of several holonic and nonholonic architectures, we propose a holonic architecture that allows for reliable and efficient reconfiguration. We call this architecture QHAR (Q-Holonic-based ARchitecture). QHAR is constructed based on the idea of a Q-holon, which has four dimensions (physical, cyber, human, and energy) and can exchange three flows (energy, data, and materials). It is a generic Holon that can represent any entity or actor of the supply chain. The QHAR is structured in three levels: centralized control level, decentralized control level, and execution level. QHAR implements the principle of an oligarchical control architecture by deploying both hierarchical and heterarchical control approaches. This ensures the overall system performance and reactivity to hazards. The proposed architecture is tested and validated on a case study.

Keywords: CPPS; industry 4.0; architecture; reconfiguration; holon; holonic manufacturing system

check for updates

Citation: Macherki, D.; Diallo, T.M.L.; Choley, J.-Y.; Guizani, A.; Barkallah, M.; Haddar, M. QHAR: Q-Holonic-Based ARchitecture for Self-Configuration of Cyber–Physical Production Systems. *Appl. Sci.* **2021**, *11*, 9013. https://doi.org/10.3390/app11199013

Academic Editor: Juan-Carlos Cano

Received: 30 June 2021
Accepted: 22 September 2021
Published: 28 September 2021

1. Introduction

Thanks to their functionalities, Cyber-Physical Production Systems (CPPS) offer solutions to the many challenges faced in various industries (e.g., increased competition, unpredictability of demand, and increasingly smaller batch sizes). One of the functionalities offered by the CPPS is the ability to adapt to internal and external hazards through self-configuration functionality (total or partial) [1–4]. To realize self-reconfiguration (partial or total) under satisfactory economic and security conditions, we need to have appropriate tools and contextualized information. This information can be organized in the form of an architecture, including all the information about the entities of the production system, the actors of the value chain (e.g., customers, suppliers, subcontractors), and the existing relationships between these entities and actors. The objective of this research work is to define an architecture that allows for the automatic reconfiguration of a production system, according to various needs, internal evolutions of the production system (e.g., breakdowns, absenteeism), and changes coming from the external environment (e.g., modification of customer requests or supplier offers). To propose this self-reconfiguration, the architecture must be sufficiently rich to contain all the necessary information for the reconfiguration and the criteria to choose between two or more alternatives configuration. We must, therefore, possess information not only on the resources of the production system (e.g., machines, operators, computer system), the external entities that the considered enterprise interacts with (e.g., customers, suppliers, subcontractors), and the flows exchanged between the entities of the production system, but also those between the company and the external actors of the value chain.

Several CPPS architectures were proposed [5–8] with a wide variety of objectives, including design, deployment, reconfiguration, and so on. Depending on the purpose of the architecture, the scope and level of detail in the description of entities and actors may differ. For machines, for example, some architectures can go down to the sensor/actuator level. As the objective of this research work is to propose a configuration according to a given need or to facilitate the reconfiguration of the system following a change (internal or external), it is not necessary to go down to this level of detail. We can limit ourselves to the machine level and describe its functionalities, inputs, outputs, and the constraints that need to be taken into account. The level of detail required in a reconfiguration is the level of detail needed for a production system manager/pilot to select the appropriate resources to execute a production plan.

Architectures can decompose the production system differently. Two different approaches can be followed to decompose a production system: according to functions (functional decomposition), or according to physical entities (physical decomposition) [9]. The physical decomposition mode was more widely adapted for reconfiguration purposes [10].

Production entities are hierarchized. Exchanges between these entities can take place between resources belonging to the same level (i.e., horizontal integration), as well as between resources belonging to different levels (i.e., vertical integration). It is, therefore, necessary that the architecture of the production system takes into account the vertical and horizontal integration of its component entities. To define the architecture, it is necessary to describe each entity/actor that composes the system and the different flows exchanged.

In [10], we proposed the Q-Holon, which is capable of representing each of the entities constituting the production system and the actors of the value chain. Using the Q-holon, we can indicate the attributes/properties of the human, physical, cyber, and energy components of an entity/actor, and describe the flows exchanged with other entities and/or actors. We also distinguish three types of flows that each entity/actor can exchange with the others, including information/data flows, material flows, and energy flows. The architecture proposed in this paper is built on the basis of this Q-Holon.

The remainder of this paper is organized as follows: Section 2 discusses some CPPS architectures proposed in the literature. Both nonholonic and holonic architectures are detailed. Section 3 presents the QHAR (Q-Holonic-based Architecture), the proposed architecture. The case study on which the QHAR was tested and validated is presented in Section 4, and Section 5 concludes this paper.

2. Related Work

In this section, we present some Industry 4.0, CPPS, or IIoT (reference or concrete) architectures proposed in the literature, and analyze them based on different criteria. The objective of this analysis is to verify the ability of these architectures to support system self-reconfiguration. The considered analysis criteria were:

a. Goal of the architecture—as the objectives of the proposed architectures may be diverse [11,12], such as design, deployment, reconfiguration, and so on, it is important to identify those that are designed to favor reconfiguration;

b. Resources considered—to support reconfiguration, all production resources involved in the reconfiguration must be taken into account by the architecture (e.g., machinery, raw materials, energy, personnel);

c. Scope of the architecture—the trigger for the reconfiguration may be internal (e.g., breakdown, absence) or external (e.g., change in customer requirements, failure of a supplier or subcontractor). Reconfiguration can impact internal entities (e.g., machines, staff), as well as external actors (e.g., customers, subcontractors). It is, therefore, essential that the scope of the architecture includes those entities and actors that may cause or be affected by reconfiguration;

d. Level of detail—the level of detail of the description of entities/actors must be sufficiently rich to provide the necessary information for a reconfiguration algorithm. However, it should not be too detailed, as this could make it unnecessarily complex;

e. Exchanged flows considered—interaction between entities/actors takes place not only through the exchange of data/information, but also through the exchange of material and energy flows. It is, therefore, essential that these different flows are considered by the architecture;

f. Flow exchange methods/protocols—different exchange methods/protocols exist. The aim here is to identify those that are used or recommended by the different architectures;

g. Mode of decomposition—two main modes of decomposition exist, functional decomposition and physical decomposition [9], the latter being more suitable for reconfiguration processes [10];

h. Type of integration—the proposed architectures generally adopt a hierarchical representation and assign the entities/actors of the system to the different levels. Interactions can take place not only between entities/actors of the same level (horizontal integration), but also between entities/actors belonging to different levels (vertical integration). As reconfiguration can modify both types of integration, a hierarchical architecture must explicitly support them;

The aim here is not to exhaustively describe the state-of-the-art of existing architectures, but to identify the main characteristics of the existing families of architectures to illustrate our comments on their capacity to support self-reconfiguration.

Among the existing architectures, we can distinguish between those that were conceived as a standardization industry reference architecture, which have acquired the status of a national (or even international) standard, and the many other architectures, proposed mainly by academics and which are more or less popular. Among the latter, we can distinguish holonic and non-holonic architectures. Yli–Ojanperä et al. [11] carried out a review of several nonholonic architectures, while a survey of holonic architectures can be found in the work of Derigent et al. [12].

Among standardization architectures for Industry 4.0, the RAMI 4.0 (Reference Architecture Model Industry 4.0) and IIRA (Industrial Internet Reference Architecture) architectures are the most widely known and used [11].

2.1. RAMI 4.0 Architecture

At present, two main reference architectures, RAMI 4.0 and IIRA, are in the process of formalization and standardization.

The RAMI 4.0 (Reference Architecture Model Industry 4.0) architecture [13] was developed by BITCOM (the German Association for IT, Telecommunications, and New Media), VDMA (Mechanical Engineering Industry Association), and ZWEI (German Electrical and Electronic Manufacturers' Association). It is about to become an international standard (IEC PAS 63088) [11]. Its 6 layers—from top to bottom: business layer, functional layer, information layer, communication layer, integration layer, and asset layer—provide an overview of the main aspects of Industry 4.0. It considers two important stages of the product life cycle, the development and production stages. Regarding the level of detail, different hierarchical levels are addressed, from the product level to the extended enterprise level (« Connected word »).

2.2. 5C Architecture

The 5C architecture, proposed by Behrad B. et al. [14], is intended as an approach or methodology for the deployment of Cyber Physical System(CPS) in production systems. It has five levels: the first level is the Smart Connection, which allows for the recovery of data collected by sensors or from an information system (e.g., ERP, MES). This connection can be made using various communication protocols, such as MTConnect. The second level is the Data-to-Information Conversion, which aims to convert data into information that can be used for various decision-making processes, such as prognosis and health management. The third level is the Cyber level, which is defined by the authors as a hub that centralizes the data coming from the different machines [15]. The analysis of these aggregated data can provide further information. The fourth level is the Cognition level, where the authors refer to advanced presentation of the acquired information to enable better decision making. The last level is Configuration level, which involves the application of the decisions made in the previous step to the physical space, in a process of self-adjustment and self-configuration of the machines [16].

External entities, such as suppliers and customers, are not considered. From our point of view, the 5C architecture does not allow for the management of reconfiguration of a production system.

2.3. 8C Architecture

The 8C architecture [5] is an extension of the 5C architecture, to which it adds three facets: the first is "Coalition", which refers to the different parties involved in the value chain (i.e., production and supply chain). The second one is "Customer", which corresponds to the customer and their role at certain stages of the product life cycle (e.g., design, production, use, after-sales service). The third facet is "Content", which represents product traceability data, such as raw material suppliers, production processes and parameters (e.g., temperature, humidity, vibration), warehouses, production shipment, and after-sales service details.

According to Jian [5], the addition of these three facets ensures horizontal integration (i.e., integration between entities belonging to the same level) in addition to vertical integration. The 8C architecture takes into account both internal and external entities (e.g., suppliers, customers, other supply chain partners).

However, the level of detail and information in this architecture is not sufficient to ensure reconfiguration.

2.4. ACPA4SF Architecture

The ACPA4SF (Anthropocentric Cyber-Physical reference Architecture for Smart Factories) architecture is based on the ACPS (Anthropocentric Cyber-Physical System) reference model [6]. The ACSP Reference Model represents the main entities comprising a CPS and the relationships between them. These entities are physical components (PCs), cyber components (CCs), and human component (HCs).

The ACPA4SF architecture is a reference architecture designed to describe the main entities that constitute intelligent factories and the essential relationships between them. As a reference architecture, it aims to serve as a guide for the development of concrete architectures.

The ACPA4SF architecture includes four types of ACPS: ACPS production system, ACPS product design, ACPS planning and control, and ACPS infrastructure. The ACPS production system represents production resources, such as machines, transport devices, and storage systems. ACPS product design represents all product knowledge from design and manufacturing ranges. The third type, ACPS planning and control, corresponds to planning and control operations. The last type is the ACPS infrastructure, which represents the contextual data and control elements required by the previous ACPS types to operate in the real plant environment (e.g., buildings, rooms, and technological infrastructure).

The ACPA4SF architecture only processes data and information flows exchanged between the different types of ACPS that constitute the production system. It does not explicitly address physical and energy flows. This architecture presents a functional decomposition, where each type of ACPS represents a well-defined function in the production system. Concerning the type of integration, in view of the decomposition of the production system, it can be considered as vertical. Among the enabling technologies to concretely deploy the ACPA4SF architecture, Privu et al. [6] quoted engineering paradigms for distributed manufacturing control systems (agent- and service-oriented computing), semantic interoperability, and human–machine interfaces.

The agent concept or communication protocols (e.g., MTConnect), ontologies, and the Semantic Web were cited as potential solutions to ensure intra- or inter-ACPS interactions.

2.5. 3C Architecture

The 3C architecture, proposed by Ahmadi et al. [17], is based on the ACPS model [6]. This architecture is based on the three components that constitute the ACPS model: physical components, cyber components, and human components. The physical and cyber components are considered to form the machine entity. Two entities are, thus, considered by this architecture: humans and machines. The authors also defined connectors (e.g., ERNI connectors, ERmet 2.0 HM, Modular Jacks & Plugs: RJ45, RJ11, RJ25, and so on) and interface protocols (e.g., Profibus, Profinet, OPC-UA Fieldbus, Ethernet Powerlink) to link the three components.

This architecture does not represent all of the actors in the production system, such as the external components (e.g., suppliers, customers), and it does not take into account all types of flows.

The decomposition of the 3C architecture is structural (depending on the type of component) with horizontal integration. This architecture considers only the internal entities (humans, machines) of the production system.

2.6. PROSA

PROSA (Product-Resource-Order-Staff Architecture), proposed by Van Brussel et al. [7], is one of the most widespread holonic architectures for manufacturing systems [12]. It is the starting point for most other holonic architectures [18], and is composed of three types of Holon: Order Holon, product Holon, and resource Holon (to which the staff Holon can be added). The Product Holon corresponds to information about products and processes, as designed (e.g., product life cycle, user requirements, design, process plans, bill of materials, quality assurance procedures). The resource Holon represents the resources used in the production process (e.g., machines, furnaces, conveyors, raw materials, personnel, space, energy). The Order Holon represents tasks that must be performed (e.g., customer orders, make-to-stock orders, repair orders). The Staff Holon is considered as an external expert that provides advice or information to the three previous Holons. It is a centralized entity for decision making that cannot be decentralized. Centralized schedulers and on-line shop floor controls are examples of Staff Holon.

The four typologies of Holon each represent a set of entities. It is impossible to represent the specificity of each of the corresponding entities with the same Holon. Only the material resources of production are taken into account. The PROSA architecture is characterized by a high degree of self-similarity, which reduces the complexity to integrate new components and enables easy reconfiguration of the system [7].

One of the authors of PROSA, Paul Valckenaers, proposed an evolution of it, referred to as the ARTI (Activity Resource Type Instance) architecture [19,20]. The main changes, compared to that of PROSA, are in the terminology used (generalization of the names of the holons to make them suitable for domains other than manufacturing) and modification of the architecture (i.e., separation of intelligent beings from intelligent agents, distinction between type and instance, resource, and activity). The ARTI architecture embeds a Digital Twin layer, and not all elements of this architecture are represented by holons [12].

2.7. ADACOR

ADACOR (ADAptive holonic COntrol aRchitecture) is a holonic architecture for production systems proposed by Leitão and Restivo [8]. This architecture defines four types of Holon: Product Holon (ProdH), Task Holon (TH), Operation Holon (OpH), and Supervision Holon (SH). The Product, Task, and Operation Holons are equivalent to the Product, Order, and Resource Holons of the PROSA architecture, respectively, while the supervision Holon, differing from the staff Holon of the PROSA architecture, is responsible for the overall coordination and optimization of the production system. The innovative aspect of the ADACOR approach is the supervisor holon which coordinate several operational and supervisor holons, and introduce the global optimisation in decentralised control [8]. To have an adaptive production control, supervisor holon allows to balance the production control between stationary and transient control states; in normal operation (stationary control state), it supervises the activity of the operational holons, whereas when a perturbation appears (transient control state), the operational holons have to find their way without assistance from the supervisor holon [8]. The ADACOR architecture also underwent several evolutions and adaptations. This is the case, for example, for the ADACOR2 architecture proposed by Barbosa et al. [21].

We can also cite other holonic architectures, such as HCBA (Holonic Component Based Architecture) [22], ORCA (dynamic Architecture for an Optimized and Reactive Control) [23], and POLLUX [24].

The main characteristics of each of the main architectures studied previously are summarized in Table 1.

As far as nonholonic architectures are concerned, the different architectures essentially aim to support the design and deployment of CPPS, and mainly concern the production resources. Only the 8C architecture explicitly addresses the other resources in the value chain (e.g., suppliers, customers). With regard to the description of the concerned entities, the information provided is rather generic. The only interaction taken into account between entities is the exchange of data/information, where these exchanges take place through the use of communication protocols. The ACPA4SF architecture also highlights the contribution of the agent concept in the implementation of interactions between entities. The 3C architecture emphasizes the role of connectors in the exchanges between the system entities.

Holonic architectures, on the other hand, attempt to respond to the need to deal with the dynamic aspect of production systems and their environments. The main holonic architectures are PROSA and ADACOR [25]. The description of the entities constituting the system is more detailed, but the description of practical solutions to ensure exchanges is generally not provided. The level of information provided is not sufficient to ensure self-reconfiguration. The Resource Holon of the PROSA architecture and the Operation Holon of the ADACOR architecture, for example, represent not only all the entities that could be used in production (e.g., information system, production machine, personnel, means of transport), but also raw materials and energy. These holons are too generic to take into account all of the parameters necessary for self-reconfiguration. It is difficult in this case, for example, to specify and choose a particular resource among several alternatives when a self-reconfiguration is to take place.

Reference architectures, such as RAMI 4.0, are not suitable for specific applications that require more detailed information. They are intended to give the most general view possible.

The studied architectures are mainly interested in data/information exchanges between entities. However, to carry out self-reconfiguration, all the interactions between the entities must be taken into account. These interactions are, of course, not only carried out through data/information exchange; they can also be done through the exchange of material or energy flows.

Table 1. Characteristics of some nonholonic and holonic CPPS architectures.

Architecture	RAMI 4.0	5C	8C	ACPA4SF	3C	PROSA	ADACOR
Architecture goal	Its aim is to provide a common framework and terminology to all participants involved in Industry 4.0 discussions and activities	It is intended as an approach or methodology for the deployment of CPS in production systems.	Guideline to build cyber-physical systems for smart factories	CPPS description and design	It aims to improve intelligent manufacturing in Industry 4.0	Management of customer order realization and reconfiguration following the addition of a new resource.	Improvement of the manufacturing control systems performance, in terms of the agile reaction to emergence and change
Resources taken into account	Physical and virtual assets	It refers to the resources contributing to the data collection (e.g., sensors, information system).	Considers value chain resources (e.g., supply, production, storage, distribution)	Production resource in the broadest sense (e.g., manufacturing, transport, storage)	Physical Components (PCs), Cyber Components (CCs), and Human Components (HCs).	Material production resources	Production resources (e.g., machines, operators)
Architecture Scope	Internal entities and external actors	Internal entities only. It does not take into account external actors, such as suppliers and customers.	Internal entities (e.g., machines, managers) and external actors (e.g., suppliers, customers). In addition to resources, it is also interested in the product.	Partly internal entities and external actors (customers, by customer order information)	Internal entities (humans, machines).	Partly internal entities and external actors (customers, by customer order information)	Internal entities
Level of detail	From global level (extended enterprise) to product level, but without description of attributes/properties.	Generic presentation. No detailed description.	Generic presentation. No detailed description.	From global level (factory) to component level, but without description of attributes/properties	Generic presentation. No detailed description.	From the global level (factory) to the elementary entity level (task, machine). Attributes and functions of each entity.	From the global level (factory) to the elementary entity level (product, task, machine).
Exchanged flows taken into account	Data and information flows	Data and information flows	Data and information flows	Data and information flows	Data and information flows	Data and information flows	Data and information flows
Flow exchange (protocol) or interaction methods	OPC UA	Communication protocol (MTConnect is given as an example)	Communication protocol (e.g., OPC UA, Ethernet, MQTT)	The agent concept or communication protocols (e.g., MTConnect), ontologies, and the Semantic Web.	Connectors (e.g., ERNI connectors, ERmet 2.0 HM, Modular Jacks and Plugs: RJ45, RJ11, RJ25, and so on) and interface protocols (Profibus, Profinet, OPC-UA Fieldbus, Ethernet Powerlink)	Contract Net Protocol	Ontology
Decomposition mode	Functional and structural	Functional	Functional	Functional	Structural	Functional and structural	Functional and structural
Integration type	Vertical and Horizontal	Vertical	Vertical and Horizontal	Vertical	Horizontal	Vertical and Horizontal	Vertical and Horizontal

3. The Q-Holonic-Based ARchitecture (QHAR)

The proposed Q-Holonic-based Architecture (QHAR) considers the physical entities of the production system, the actors of the value chain (e.g., customers, suppliers, subcontractors), and the flows exchanged between entities and actors. To represent these entities, actors, and flows, the Q-Holon is used, which we defined in a previous work [10]. The Q-holon is a generic Holon that can represent any entity or actor, thanks to its four dimensions (physical, cyber, human, and energy). An entity or actor can be modelled by a Q-holon by specifying the attributes and operations of its physical, cyber, human, and energy dimensions. Attributes and operations may depend on one of its dimensions, or a combination of two or more of them. The interactions between Q-Holons are materialized by the exchanged flows. In the first version of the Q-holon, four types of flows were considered: (1) Space (needs for physical adjacency or orientation between two elements); (2) Energy (energy transfer); (3) Data (data/information exchange), and (4) Material (materials exchange). To facilitate understanding and avoid confusion, we propose to consider positioning constraints (Flow #1, Space) not as a flow, but as an attribute. To better describe the spatial requirements and orientation constraints of an entity, it is better to express them in terms of attributes than in terms of flows. Indeed, these aspects constitute intrinsic parameters of the considered entity. Another reason that convinced us of the necessity of this change is the fact that the meaning of the "Space" flow is not homogeneous with the meaning of the three other flows. Indeed, the other flows (Energy, Data, and Material) express the transfer/exchange between two or more entities/actors of an element. This "Space" attribute indicates, if necessary, the layout constraints (location, relative position), required space, load and unload access points, and safety distance to be respected. Therefore, a new version of the Q-holon (see Figure 1) is used to define the QHAR, composed of four dimensions and with the ability to exchange three flows.

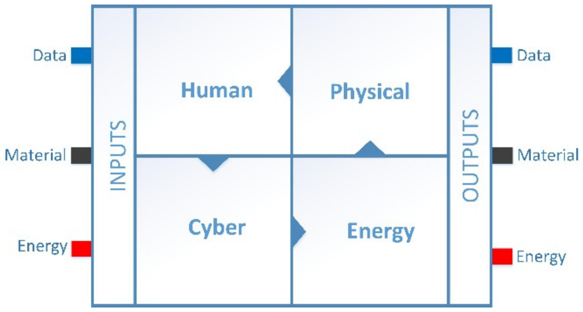

Figure 1. Q-Holon.

The energy flow corresponds to any form of energy that entities/actors can exchange. It can be electrical energy, mechanical energy, thermal energy, and so on. The data flow can concern raw data or information that are more elaborate. The material flow concerns any exchange of tangible elements (e.g., parts, products).

The Q-Holon allows us to deal with the multidimensional character of the CPPS that constitute modern production systems, as well as to consider Humans as full members of the production system. Thanks to these four dimensions and the three exchanged flows, it is possible to model and specify all the necessary aspects, to ensure the implementation and the piloting of the CPPS.

Firstly, we introduce the QHAR in general, then present its main constituents (i.e., entities and actors, along with their attributes and operations) in more detail. Finally, we explain how the different constituents of the system interact with each other.

3.1. General Overview of the Q-Holonic-Based ARchitecture (QHAR)

The QHAR consists of three hierarchical levels (see Figure 2): Centralized control level, Decentralized control level, and Execution level.

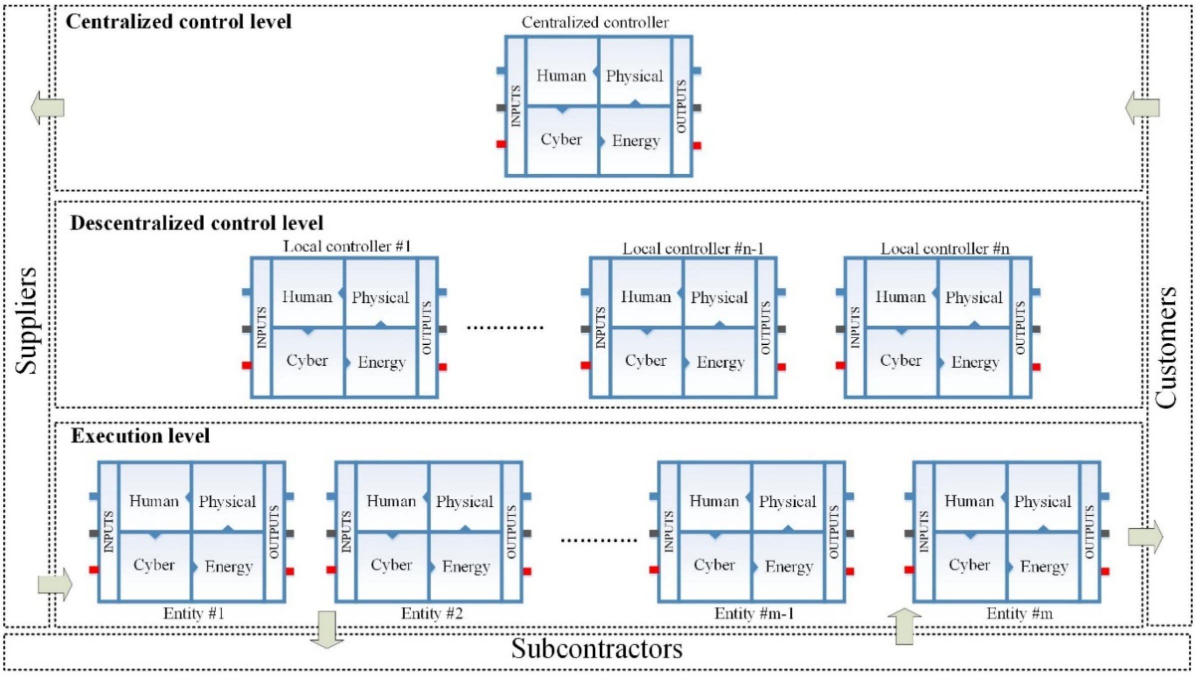

Figure 2. Q-Holonic-based ARchitecture (QHAR).

The hierarchical representation of production systems was adopted for a long time, with the implementation of the IEC 62,264 standard (see Purdue Reference Architecture [26] and CIMOSA Reference Architecture [27] for examples). This hierarchical approach was also adopted in more recent architectures, such as the RAMI 4.0 architecture [13], ADACOR architecture [8], and ARTI architecture [19], even though there was increasing talk of a decentralized architecture for modern production systems (CPPS) [28]. The persistence of the adoption of these hierarchical architectures is due to several reasons. First of all, there are practical reasons: this is, indeed, the dominant way in which production workshops are organized and operated at present. Not all entities in these workshops have the functionalities to operate in a fully decentralized manner. The compromise between autonomy and global optimization and the better performance offered by the oligarchical control architecture, compared to that of other types of control architectures, may also explain the fact that these hierarchical architectures are still being adopted [20,23,29,30]. The number of levels in these hierarchized architectures is very diverse (e.g., 6 levels in the RAMI 4.0 architecture, 5 levels in ADACOR, and 4 levels in the ARTI architecture). For the QHAR, we decided to divide the production system into 3 levels. The lowest level—the execution level—is made up of entities that may be directly involved in carrying out one or more production activities, materials, and finished or semifinished products. As the entities constituting the execution level may not be completely autonomous, other entities are provided for the control and coordination activities that cannot be performed by them. These entities are assigned to the decentralized control level. Finally, the centralized control level is responsible for the overall optimization and interaction with external actors.

3.1.1. Centralized Control Level

This level consists of the centralized controller, which is in charge of control actions that require an overview of not only the internal entities, but also the other actors in the value chain (e.g., customers, suppliers, subcontractors). The actions performed by this controller may involve the establishment of the Master Production Schedule (MPS), the assignment of tasks to internal entities or to subcontractors in charge of carrying them out, verification of the availability of raw materials and other resources (e.g., machines, operators, subcontractors), the establishment of supply or work orders, and so on. To perform these actions, the centralized controller interacts with the customer or the market (e.g., customer order, sales forecast) and possesses information about resources and processes (e.g., raw materials, entities, subcontractors, production routing). The role of the centralized controller can be played by staff, assisted by an ERP-type information system.

The role played by the centralized control level is close to that assigned to the Holon staff in PROSA [7] and to the Holon supervision in ADACOR [8]. The role of the PROSA Holon staff, however, is limited to providing advice or information to the three other Holons (i.e., order Holon, product Holon, and resource Holon). The Holon supervision of ADACOR plays a stronger role (overall coordination and optimization) than that of the PROSA Holon staff. The centralized control level of QHAR, for its part, can have more "active" functions which require an overall view of the system and its environment. These functions may consist of modifying the production plan following a change in a customer's order or subcontracting an operation as it can no longer be carried out internally (e.g., due to breakdowns, workload, and so on), among others.

3.1.2. Decentralized Control Level

Local controllers constitute this level. A local controller is dedicated to a part of the production system entities (one or more). It is in charge of control operations that cannot be carried out by the entity (or entities) for which it is responsible. For example, this may involve giving a start or stop command, local scheduling of assigned tasks, transferring a task assigned to a given entity by the centralized controller to another entity, coordinating with other entities, verification of the correct execution of tasks, sending reports to the centralized controller, and so on. This role of local controller can be performed by an operator, a MES, a SCADA, or similar. This intermediate level is proposed to strengthen the autonomy of the system as a whole, by overcoming the limitations of the individual entities. These limitations may be the inability to diagnose certain faults, to communicate or coordinate with other entities, or otherwise.

3.1.3. Execution Level

This is the operating system, constituted by all the entities involved in the activities of the production process. These activities can be transformation activities (e.g., material removal, machining, chemical processes, deformation), assembly (e.g., welding, riveting), surface treatment, transport, control, storage, and so on. In addition to these activities, these entities also integrate certain control functions (e.g., emergency stop, alarm). The entities corresponding to raw materials and final products also operate in this level.

The QHAR also considers the other players in the value chain, including customers, suppliers, and subcontractors.

To explain the operating principle of the QHAR, we consider, for simplicity, a production system that receives a raw material and performs a number of operations on it to obtain a finished product.

The centralized controller establishes a Master Production Schedule, which is a list of products to be manufactured with information on the quantities and the desired delivery date (see Table 2). The centralized controller also determines the raw materials needed to realize the Master Production Schedule and assigns, to each raw material, the Production Routing to obtain the finished product. This Production Routing (see Table 2) lists, in chronological order, all of the operations that a raw material must undergo. For each operation, it indicates the resource in charge of carrying it out and whether it authorizes this one to transfer this operation to another resource in the case of necessity (e.g., due to failure, risk of delay in view of the other tasks to be carried out, capacity). When the transfer of a task is authorized, a list of alternative authorized resources is indicated. The Production

Routing is assigned to the corresponding raw material. The further steps are orchestrated by the raw material. Note that, here, we consider that the raw materials are intelligent; for example, raw materials equipped with RFID (Radio-Frequency IDentification).

Table 2. Master Production Schedule and Production Routing template.

Master Production Schedule			**Production Routing**							
Product	**Quantity to Produce**	**Delivery Date**	**NO**	**Operation**	**Resource**	**Total Time**	**Due Date**	**Alternative Resources**	**Status**
Product 1	Q1	D1							
Product 2	Q2	D2							
...	...	...							
Product n	Qn	Dn							

The raw material solicits the resources indicated for each of the operations in its Production Routing in chronological order. When a resource performs an operation, the status of the task is updated and the resource indicated for the next operation is requested, and so on until the final operation. As shown in Figure 3, if the resource that is indicated to perform an operation is not available, another resource—indicated in the "Alternative resources"—is requested. In all cases, whether operations are realized or not, a report is sent to the centralized controller. Negotiation and coordination between the executing entities can take place directly or through local controllers.

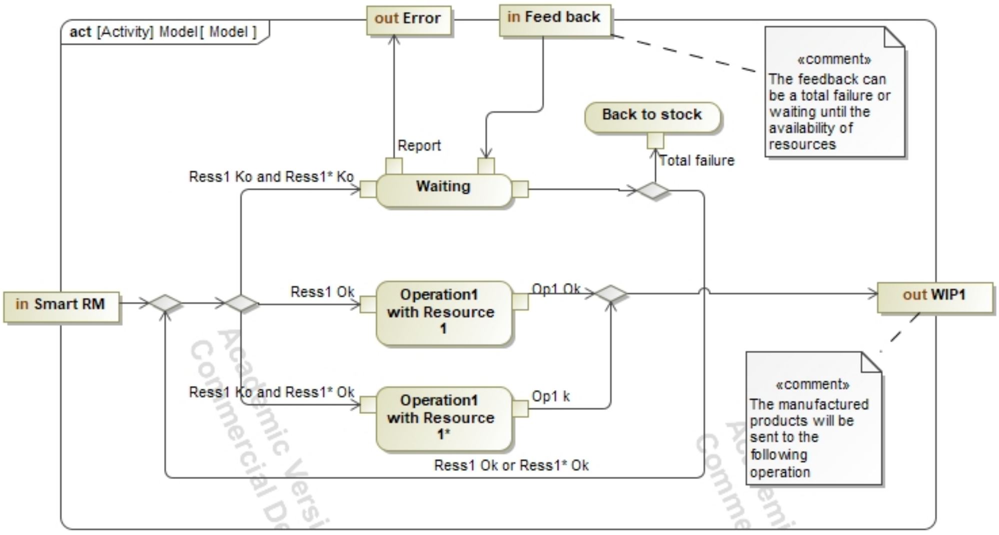
act [Activity] Model[Model]
out Error
in Feed back
«comment» The feedback can be a total failure or waiting until the availability of resources
Back to stock
Report
Total failure
Ress1 Ko and Ress1* Ko
Waiting
in Smart RM
Ress1 Ok
Operation1 with Resource 1
Op1 Ok
out WIP1
«comment» The manufactured products will be sent to the following operation
Ress1 Ko and Ress1* Ok
Operation1 with Resource 1*
Op1 k
Ress1 Ok or Ress1* Ok

Figure 3. Operating principle of QHAR.

The QHAR is an oligarchical architecture. Indeed, the centralized controller ensures the overall performance of the system (hierarchical control). Local controllers and controllers integrated in the execution level entities ensure local optimization and foster reactivity to hazards (heterarchical control).

3.2. Representation of Entities and Actors by Q-Holons

To implement the QHAR, it is necessary to identify the basic constituents of the system that will be represented by Q-Holons, to define the characteristics of each constituent, and to specify the operating rules of the system.

With regard to the main production system studied, any entity that has its own identity, which exists as such and which can play at least one autonomous role in one of the production processes, can be represented by a Q-Holon. It can be a processing machine, a robot, a conveyor, an AGV, an information system, a raw or consumable material, or an operator. Two or more Q-Holons can be combined to form a process segment. Where relevant, the process segment can be represented as a single Q-holon, by combining the attributes and operations of the Q-Holons that compose it. The other actors of the value chain whose operation is not detailed (e.g., customer, supplier, subcontractor) can each be represented by a Q-Holon as well.

The description of the characteristics (attributes, operations) of each Q-holon depends on the nature of the entity/actor—which it represents—and the needs related to the operation of the system.

Thanks to the Q-Holon material flow, the QHAR takes into account the physical flows between entities/actors. These physical flows are absent in other holonic architectures. The cyber dimension and data flow allow for specifying aspects related to interoperability between applications or software. The energy requirement and the possibility of energy recovery from certain entities can be modeled by the energy dimension and the energy flow. This is the case, for example, for certain resources (e.g., machines, AGVs) which consume and/or supply energy. Humans are an integral component of production systems, whose needs and services must be taken into account. A human can be equipped with accessories that can have cyber, physical, and energetic dimensions (e.g., augmented reality glasses, exoskeleton). The equipped operator, who can be qualified as an "Augmented Worker", can therefore be modelled using a Q-holon with the attributes and operations corresponding to these four dimensions. Ergonomic aspects can be taken into account, using attributes such as "Recommended working environment conditions", "Extreme reach zones", and "Maximum capacity".

Below (see Figure 4) are some examples of Q-Holons corresponding to entities belonging to one of the three levels of the QHAR, or representing another actor of the value chain.

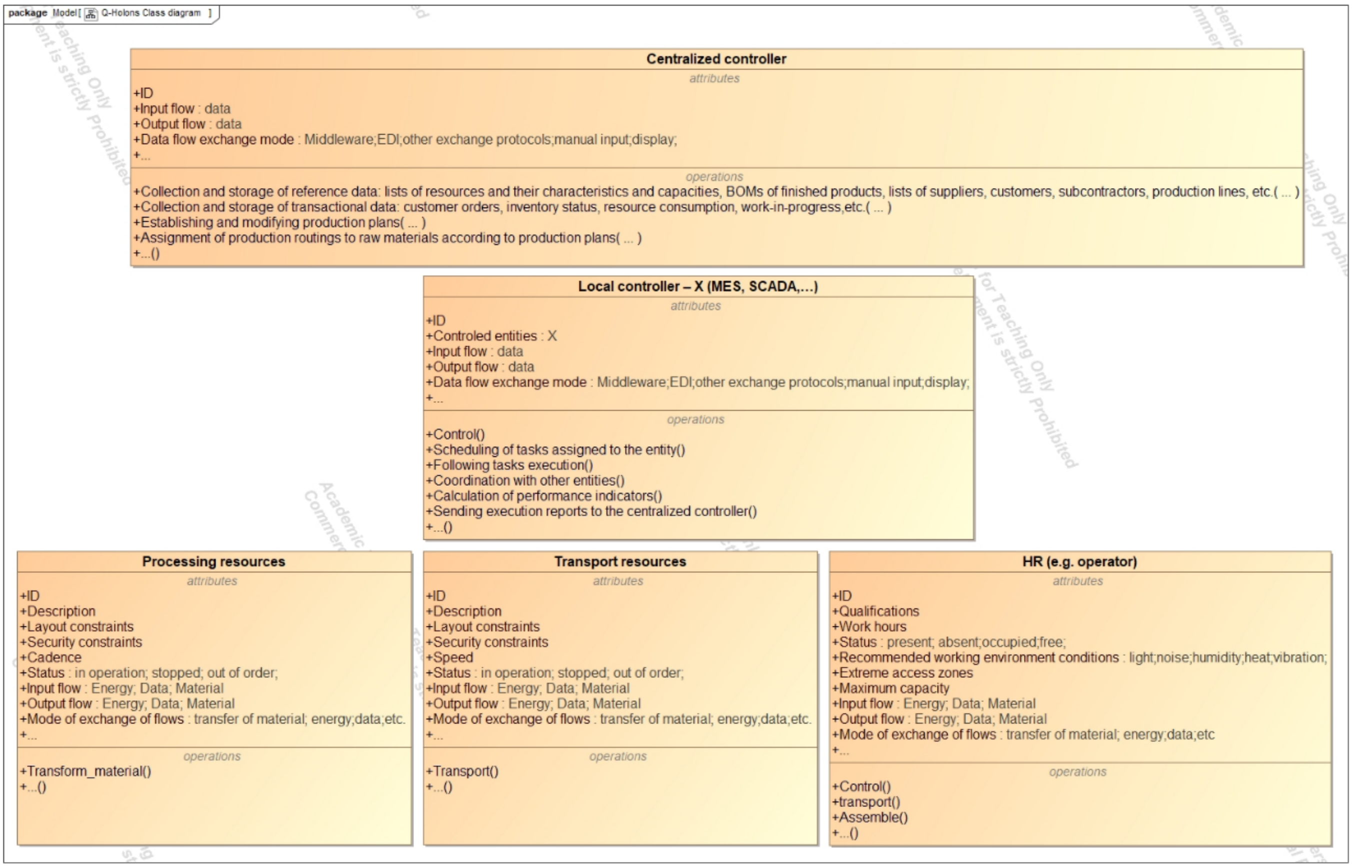
package Model[Q-Holons Class diagram]
Centralized controller
attributes
+ID
+Input flow : data
+Output flow : data
+Data flow exchange mode : Middleware;EDI;other exchange protocols;manual input;display;
+...
operations
+Collection and storage of reference data: lists of resources and their characteristics and capacities, BOMs of finished products, lists of suppliers, customers, subcontractors, production lines, etc.(...)
+Collection and storage of transactional data: customer orders, inventory status, resource consumption, work-in-progress,etc.(...)
+Establishing and modifying production plans(...)
+Assignment of production routings to raw materials according to production plans(...)
+...()
Local controller – X (MES, SCADA,...)
attributes
+ID
+Controled entities : X
+Input flow : data
+Output flow : data
+Data flow exchange mode : Middleware;EDI;other exchange protocols;manual input;display;
+...
operations
+Control()
+Scheduling of tasks assigned to the entity()
+Following tasks execution()
+Coordination with other entities()
+Calculation of performance indicators()
+Sending execution reports to the centralized controller()
+...()
Processing resources
attributes
+ID
+Description
+Layout constraints
+Security constraints
+Cadence
+Status : in operation; stopped; out of order;
+Input flow : Energy; Data; Material
+Output flow : Energy; Data; Material
+Mode of exchange of flows : transfer of material; energy;data;etc.
+...
operations
+Transform_material()
+...()
Transport resources
attributes
+ID
+Description
+Layout constraints
+Security constraints
+Speed
+Status : in operation; stopped; out of order;
+Input flow : Energy; Data; Material
+Output flow : Energy; Data; Material
+Mode of exchange of flows : transfer of material; energy;data;etc.
+...
operations
+Transport()
+...()
HR (e.g. operator)
attributes
+ID
+Qualifications
+Work hours
+Status : present; absent;occupied;free;
+Recommended working environment conditions : light;noise;humidity;heat;vibration;
+Extreme access zones
+Maximum capacity
+Input flow : Energy; Data; Material
+Output flow : Energy; Data; Material
+Mode of exchange of flows : transfer of material; energy;data;etc
+...
operations
+Control()
+transport()
+Assemble()
+...()

Figure 4. Object models of some basic entities of QHAR.

3.3. Interaction between Different Production System Components

In this section, we present the interactions between the main internal entities of the production system (see Figure 5). It is a simplified and limited model that shows only data/information flows between internal entities. We start from the step of the production routing assignment to the raw material to be processed, to obtain the finished product of the work order. We consider entity classes (Centralized controller, Raw Materials, Processing Resources, and Transportation Resources) for this explanation. When the QHAR is implemented for a real production system, it is the instances of these classes that are involved in the interactions. Therefore, there could be interactions between instances of the same class. Figure 5 shows an example of interactions between classes.

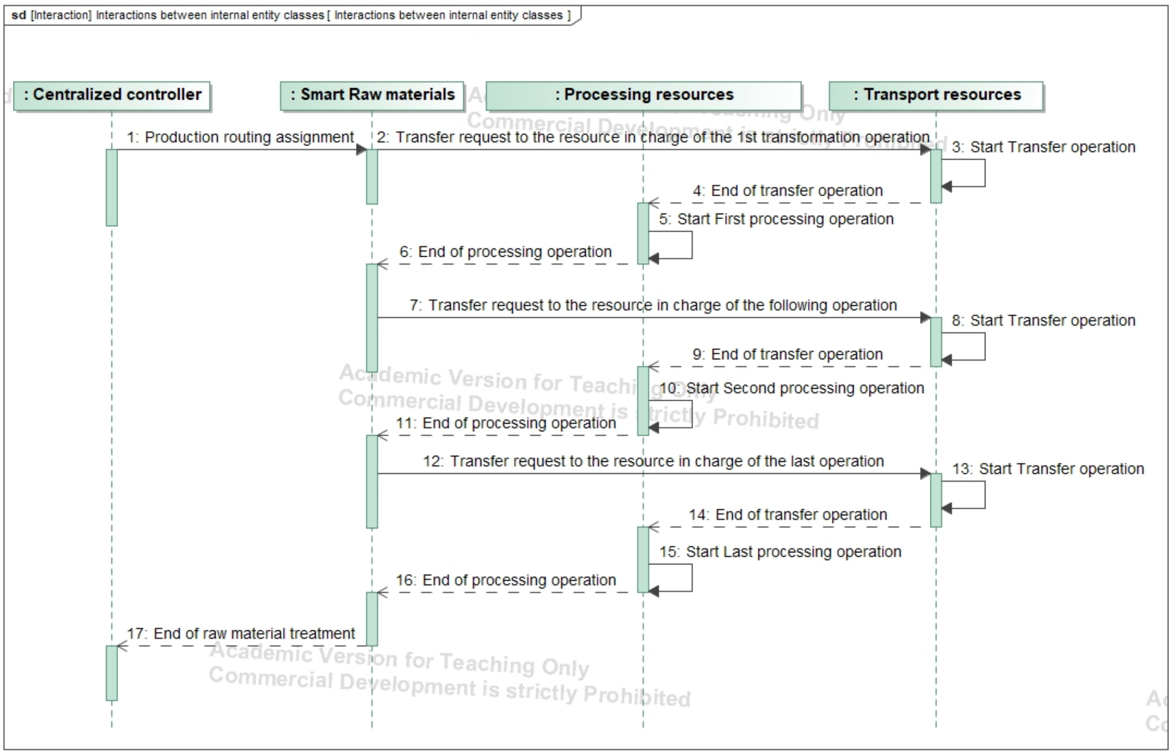
sd [Interaction] Interactions between internal entity classes [Interactions between internal entity classes]
: Centralized controller
: Smart Raw materials
: Processing resources
: Transport resources
1: Production routing assignment
2: Transfer request to the resource in charge of the 1st transformation operation
3: Start Transfer operation
4: End of transfer operation
5: Start First processing operation
6: End of processing operation
7: Transfer request to the resource in charge of the following operation
8: Start Transfer operation
9: End of transfer operation
10: Start Second processing operation
11: End of processing operation
12: Transfer request to the resource in charge of the last operation
13: Start Transfer operation
14: End of transfer operation
15: Start Last processing operation
16: End of processing operation
17: End of raw material treatment

Figure 5. Sequence diagram of interactions between production system's internal entity classes.

Execution-level entities can participate in interactions directly, or through their local controller. The choice between the two options should be carried out on a case-by-case basis, according to the possibilities and the steering policy in force. Here, we indicate only the execution-level entity. In Figure 5, we suppose that the raw material, processing, and transportation resources are available. We use the UML sequence diagram.

When a processing or transport resource indicated by default for an operation is not available (due to, e.g., failure, inability to perform the operation on time), it informs the raw material (directly or indirectly, through its local controller). The latter asks the resources indicated in the list of alternative resources for this operation one by one, until it finds an available resource to perform the operation (see Figure 6).

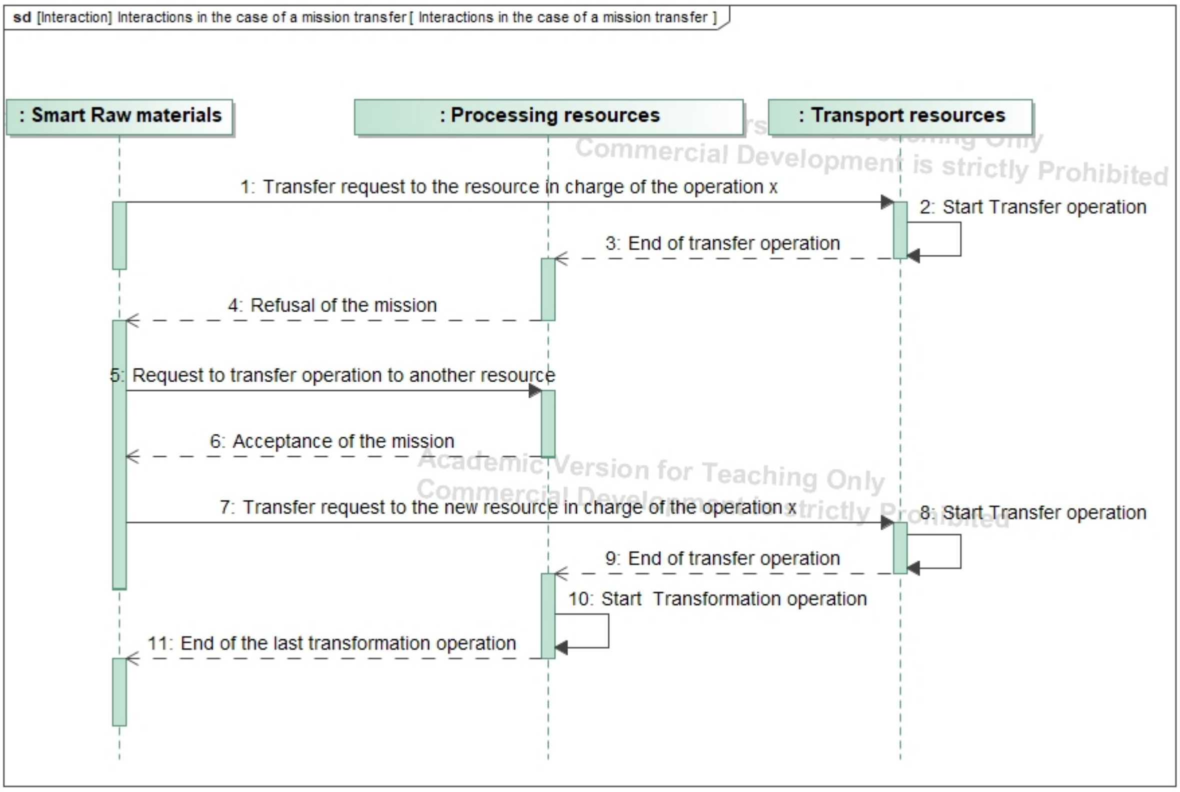
sd [Interaction] Interactions in the case of a mission transfer [Interactions in the case of a mission transfer]
: Smart Raw materials
: Processing resources
: Transport resources
1: Transfer request to the resource in charge of the operation x
2: Start Transfer operation
3: End of transfer operation
4: Refusal of the mission
5: Request to transfer operation to another resource
6: Acceptance of the mission
7: Transfer request to the new resource in charge of the operation x
8: Start Transfer operation
9: End of transfer operation
10: Start Transformation operation
11: End of the last transformation operation

Figure 6. Interaction sequence diagram in case where default resource indicated for an operation is not available.

4. Case study

4.1. General Presentation

To test and validate the QHAR, we applied it to a manufacturing workshop, consisting of (see Figure 7):

- Three workstations:
  - ○ A CNC–machining center, capable of machining different parts and driven by an operator;
  - ○ a drilling station, held by an operator;
  - ○ an assembly station, driven by an operator assisted by a cobot, and
- an AGV equipped with a robot arm.

Each of the three workstations is driven by an operator. An operator may provide assistance to another workstation than their own, if necessary, and may also be involved in the transportation of parts and products. A person is responsible for the overall management of the workshop. They are equipped with a planning tool (ERP), as well as a tool for launching and monitoring production orders (MES).

The workshop is supplied with energy by a public network, and may also use locally produced energy. The parts processed in this workshop do not all have the same production routing; hence, each part is associated with information of its production routing and handles the execution of operations of this routing by negotiating with the manufacturing or transportation resources. The production routing information is updated regularly. For reasons of simplicity, external actors (e.g., customers, suppliers, subcontractors) were not included in the modeling.

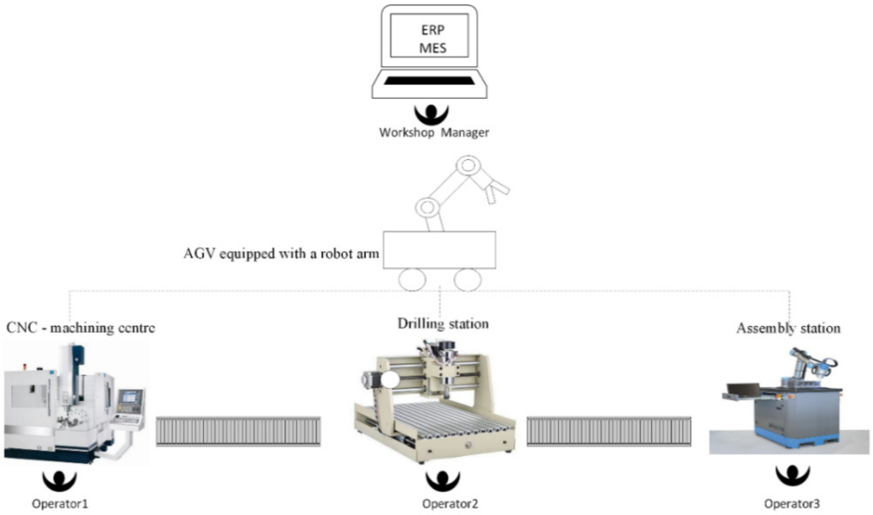

Figure 7. Manufacturing workshop taken as a case study.

4.2. Modeling of Workshop Entities by Q-Holons and Their Assignment to the Three Levels of the QHAR

A Q-Holon allows us to represent not only the characteristics of an entity (or an actor), but also the flows that it can exchange with the other entities. The Q-Holon's characteristics are described by the attributes/properties of the Q-Holons four dimensions (physical, cyber, human, and energy) and the eventual operations that it can perform. With regard to flows, they can be of three types: energy, material, and data/information.

The workshop manager, assisted by an ERP system, plays the role of the centralized controller. This role is modeled by the Q-Holon "Centralized controller". This same workshop manager is assisted by an MES to perform the "Decentralized controller" role; they are also represented by a Q-Holon. The machining center, conveyors 1 and 2, the drilling station, the assembly station, and the AGV are modeled by a Q-Holon each, all belonging to the Execution level.

We could have integrated the operator-drivers of the different workstations into the Q-Holons that represent these stations. However, as the operators could carry out other tasks in addition to the driving of workstations, we decided to model them by autonomous Q-Holons. The public energy distribution network is also modeled by a Q-Holon.

We recommend the use of two types of SysML diagrams to represent all aspects of a Q-Holon-based model: a BDD diagram and an IBD diagram. The BDD diagram, as shown in Figure 8, represents the attributes/properties and operations of the Q-Holons, while the flows exchanged between the Q-Holons are described by the IBD diagram (see Figure 9).

Figure 10 shows an example of the flows exchanged between two entities of the Execution Level; namely, the Assembly center block. This block is constituted by a cobot associated with an operator, each of which is characterized by the required spatial attribute which represents the safety space necessary to avoid collisions between the two entities. Both the cobot and operator receive information from the local control center (i.e., production rules and operational orders), as well as the parts that need to be assembled. The energy used here can be supplied by the public energy network or locally produced by the cobot, if possible; this is modeled by the in/out (in and out) energy ports. Additionally, the cobot exchanges information with the operator through a human–machine interface. As output, assembled products are transferred and real-time reports, operational responses, and cobot- and operator-specific data are sent to the local control center.

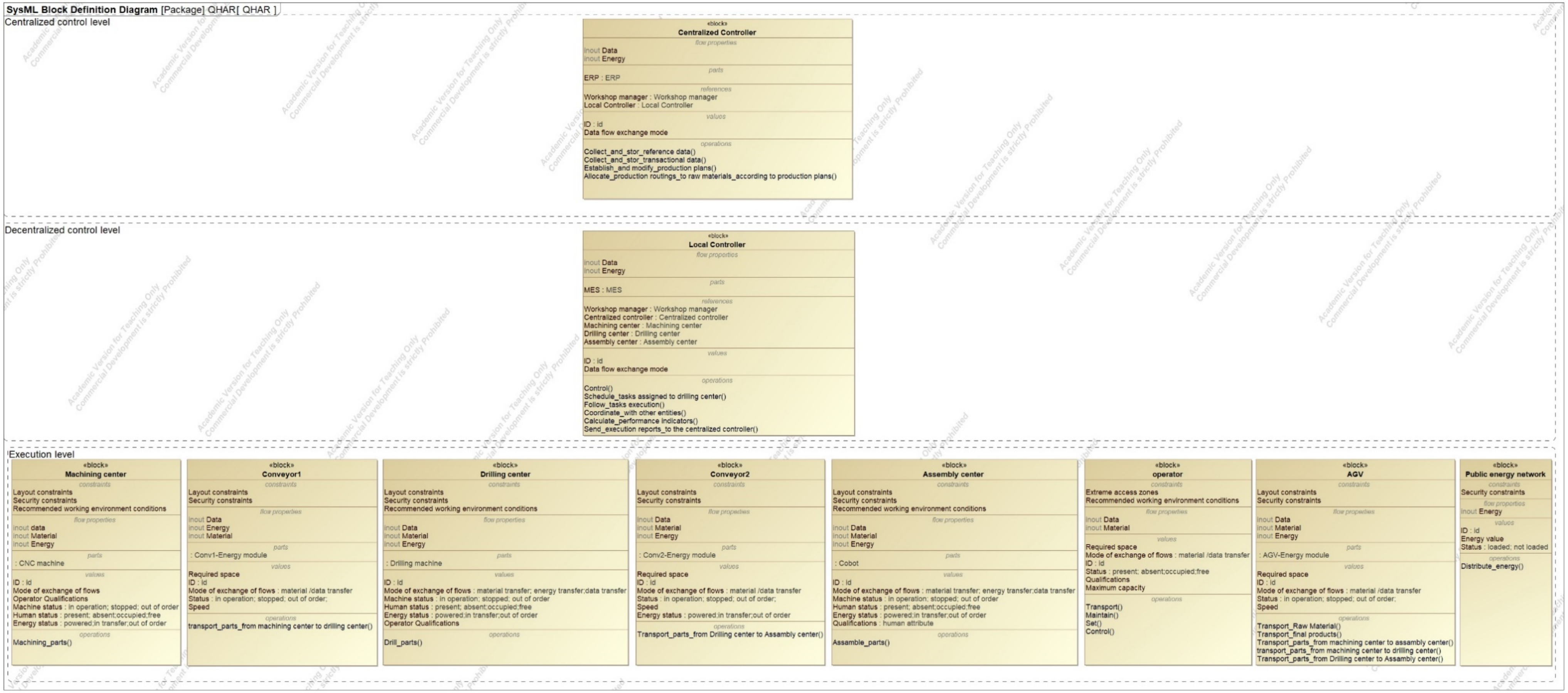
SysML Block Definition Diagram [Package] QHAR[QHAR]
Centralized control level
«block»
Centralized Controller
flow properties
inout Data
inout Energy
parts
ERP : ERP
references
Workshop manager : Workshop manager
Local Controller : Local Controller
values
ID : id
Data flow exchange mode
operations
Collect_and_stor_reference data()
Collect_and_stor_transactional data()
Establish_and modify_production plans()
Allocate_production routings_to raw materials_according to production plans()
Decentralized control level
«block»
Local Controller
flow properties
inout Data
inout Energy
parts
MES : MES
references
Workshop manager : Workshop manager
Centralized controller : Centralized controller
Machining center : Machining center
Drilling center : Drilling center
Assembly center : Assembly center
values
ID : id
Data flow exchange mode
operations
Control()
Schedule_tasks assigned to drilling center()
Follow_tasks execution()
Coordinate_with other entities()
Calculate_performance indicators()
Send_execution reports_to the centralized controller()
Execution level
«block»
Machining center
constraints
Layout constraints
Security constraints
Recommended working environment conditions
flow properties
inout data
inout Material
inout Energy
parts
: CNC machine
values
ID : id
Mode of exchange of flows
Operator Qualifications
Machine status : in operation; stopped; out of order
Human status : present; absent;occupied;free
Energy status : powered;in transfer;out of order
operations
Machining_parts()
«block»
Conveyor1
constraints
Layout constraints
Security constraints
flow properties
inout Data
inout Energy
inout Material
parts
: Conv1-Energy module
values
Required space
ID : id
Mode of exchange of flows : material /data transfer
Status : in operation; stopped; out of order;
Speed
operations
transport_parts_from machining center to drilling center()
«block»
Drilling center
constraints
Layout constraints
Security constraints
Recommended working environment conditions
flow properties
inout Data
inout Material
inout Energy
parts
: Drilling machine
values
ID : id
Mode of exchange of flows : material transfer; energy transfer;data transfer
Machine status : in operation; stopped; out of order
Human status : present; absent;occupied;free
Energy status : powered;in transfer;out of order
Operator Qualifications
operations
Drill_parts()
«block»
Conveyor2
constraints
Layout constraints
Security constraints
flow properties
inout Data
inout Material
inout Energy
parts
: Conv2-Energy module
values
Required space
ID : id
Mode of exchange of flows : material /data transfer
Status : in operation; stopped; out of order;
Speed
Energy status : powered;in transfer;out of order
operations
Transport_parts_from Drilling center to Assambly center()
«block»
Assembly center
constraints
Layout constraints
Security constraints
Recommended working environment conditions
flow properties
inout Data
inout Material
inout Energy
parts
: Cobot
values
ID : id
Mode of exchange of flows : material transfer; energy transfer;data transfer
Machine status : in operation; stopped; out of order
Human status : present; absent;occupied;free
Energy status : powered;in transfer;out of order
Qualifications : human attribute
operations
Assamble_parts()
«block»
operator
constraints
Extreme access zones
Recommended working environment conditions
flow properties
inout Data
inout Material
values
Required space
Mode of exchange of flows : material /data transfer
ID : id
Status : present; absent;occupied;free
Qualifications
Maximum capacity
operations
Transport()
Maintain()
Set()
Control()
«block»
AGV
constraints
Layout constraints
Security constraints
flow properties
inout Data
inout Material
inout Energy
parts
: AGV-Energy module
values
Required space
ID : id
Mode of exchange of flows : material /data transfer
Status : in operation; stopped; out of order;
Speed
operations
Transport_Raw Material()
Transport_final products()
Transport_parts_from machining center to assambly center()
transport_parts_from machining center to drilling center()
Transport_parts_from Drilling center to Assambly center()
«block»
Public energy network
constraints
Security constraints
flow properties
inout Energy
values
ID : id
Energy value
Status : loaded; not loaded
operations
Distribute_energy()

Figure 8. Attributes/properties and operations of Q-holons modeling entities of workshop.

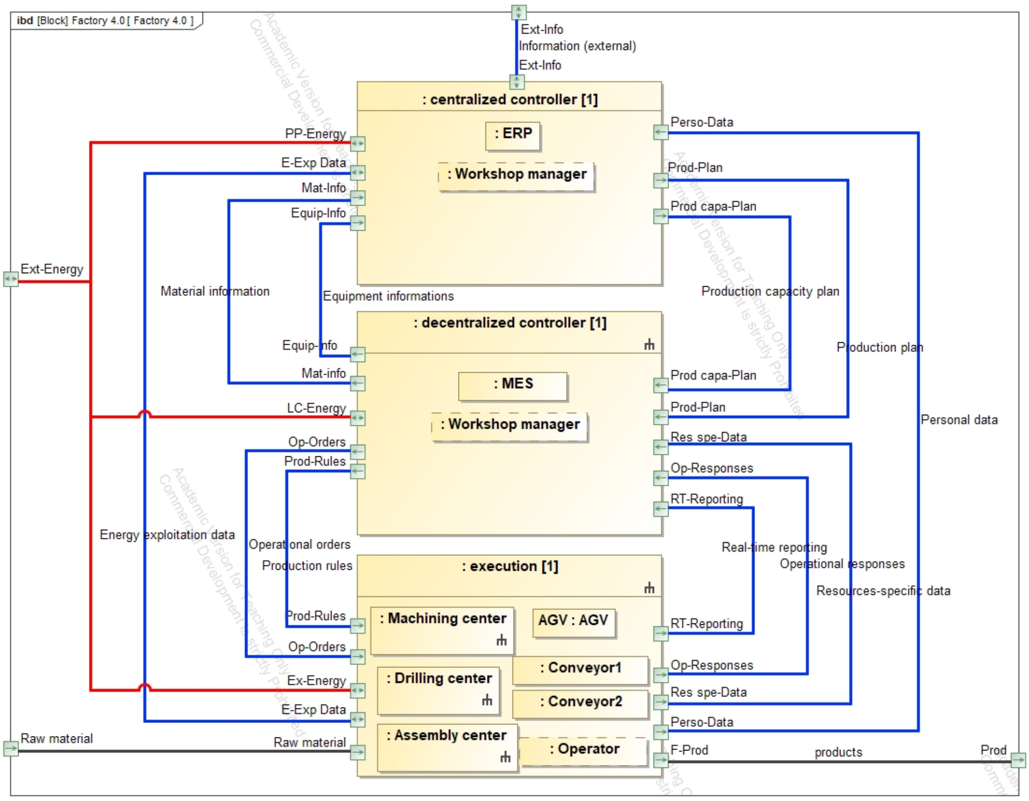

Figure 9. IBD Diagram.

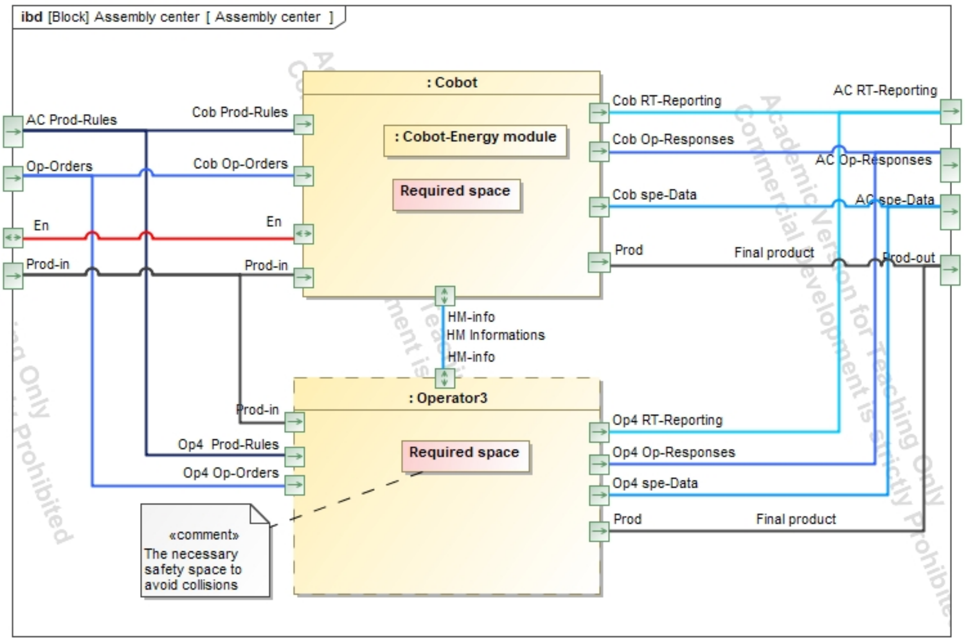

Figure 10. Assembly center IBD diagram.

The aim of this case study was to test and verify the ability of the proposed architecture to allow self-reconfiguration. For this purpose, the case study had to be representative of existing production workshops. This representativeness must be verified, in terms of the entities that make it up as well as the operations carried out. Thus, there are classic entities, such as CNC machines and drilling machines, as well as more recent equipment, such as AGVs and cobots. In terms of operations, machining, assembly, and transport operations are carried out. Care has also been taken to simplify the case, to make it easier to understand without impacting its representativeness.

Through this case study, we can see that the Q-Holon concept, thanks to its four dimensions and three types of flows, makes it possible to model the entities/actors of the production system and potentially allows for automatic or semiautomatic reconfiguration. Indeed, thanks to the use of Q-Holons, each entity/actor is represented by its attributes/properties, its operations, and the flows it can exchange with the other entities/actors. An internal or external change in the production system requiring reconfiguration will modify one or more of the Q-Holon parameters (attributes/properties, operations, flows). Reconfiguration, therefore, must consist of selecting and connecting the entities/actors whose parameter values respect the constraints and needs of the new context. This search for the new configuration can be carried out using an automatic reconfiguration algorithm, based on the information provided by the Q-Holon-based model of the production system.

4.3. How Could the Implementation of the Proposed Architecture, QHAR, Support Self-Reconfiguration?

To perform self-reconfiguration, three main actions must be performed autonomously [2,31–33]:

1. **Detection of the need for reconfiguration**—this is the preliminary phase of the actual reconfiguration process. With regard to faults, several advances were made in the implementation of self-diagnostic functions for certain faults [34,35]. This generally concerns foreseen defaults [31]. For unforeseen defects or external disturbances, further progress is still needed [31]. QHAR could be used to help diagnose such unforeseen faults. Indeed, the proper functioning of an entity can be characterized by the values of the attributes/properties of its four possible dimensions (human, cyber, energy, physical), those of its three possible incoming and outgoing flows (data, physical, energy), and by the operations it is supposed to perform. The failure of any of these parameters might be a sign of failure of the entity in question. This failure may be noted by the entity itself, its local controller, or the central controller. The monitoring of all these parameters by the entity itself, its local controller, and/or the centralized controller can ensure that practically all potential defects can be detected.
2. As far as external disturbances are concerned, they are detected by the centralized controller. In the case of a change in customer demand, for example, the centralized controller ascertains whether the current installation is capable of handling the new order. If it is not guaranteed, this may indicate the need to change the current configuration.
3. **Determination of the entities needed for the new configuration**—the need for reconfiguration detected earlier usually involves a change in the composition of the production system. This involves choosing the most appropriate entities for the current need. This choice may require decision support tools/methods and the definition of selection criteria. The first selection criteria are those relating to the operational capabilities and properties of the entities, where the operational capabilities and properties of each entity can be defined using the Q-Holon that represents it.
4. **Definition of a new layout**—this involves determining the location of the entities and the interconnections to be established between them. Thanks to the attributes/properties of the entities—in particular, the physical/spatial dimension—the best location can be defined for each entity. The interaction between entities can be characterized by the flows (material, data, and energy) they exchange. Thus, entities that exchange flows may be permanently or temporarily interconnected.

5. Conclusions

Adaptation to internal and external change—or self-reconfiguration—is one of the potential functionalities that CPPS can offer. However, this functionality requires a suitable architecture to be implemented. In this paper, several holonic and nonholonic architectures were analyzed and a new holonic architecture was proposed. The Q-Holonic-based ARchitecture (QHAR), the proposed architecture, is based on the idea of a Q-Holon, an enriched Holon with four dimensions (physical, cyber, human, energy), which is used to represent each of the entities and actors that compose the value chain.

The QHAR considers the internal entities of a production system, as well as other players in the value chain, such as customers, suppliers, and subcontractors. It consists of three levels: Centralized control level, Decentralized control level, and Execution level. The composition and roles of each of these levels were presented. The QHAR considers all of the types of flows that can exist between the entities and actors of the value chain; namely, data/information flows, material flows, and energy flows. Some modes of interaction between system constituents were also described, using a SysML sequence diagram. QHAR is a concrete architecture, providing all the necessary information for a self-reconfiguration algorithm.

The proposed architecture was applied to a case study to demonstrate its relevance and feasibility. This case study allowed us to illustrate how the proposed architecture, QHAR, could support a self-reconfiguration algorithm. Care was taken to ensure that it was as representative as possible of current production systems, in terms of the entities that make it up and the operations carried out. However, external actors (e.g., customers, suppliers, subcontractors) and their interactions with other entities were not represented.

Self-reconfiguration is a very valuable potential feature of CPPS that is not yet fully implemented [31]. The successful deployment of the QHAR can contribute to the effective implementation of the self-reconfiguration capability of CPPS.

Author Contributions: Conceptualization D.M., T.M.L.D. and J.-Y.C.; methodology, D.M., T.M.L.D. and J.-Y.C.; validation, T.M.L.D. and J.-Y.C.; formal analysis, T.M.L.D. and J.-Y.C.; investigation, D.M.; resources, D.M.; writing—original draft preparation, D.M. and T.M.L.D.; writing—review and editing, D.M., T.M.L.D. and A.G.; supervision, T.M.L.D., J.-Y.C., A.G. and M.B.; project administration, J.-Y.C. and M.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding. The APC was funded by ISAE-Supméca.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bortolini, M.; Galizia, F.G.; Mora, C. Reconfigurable manufacturing systems: Literature review and research trend. *J. Manuf. Syst.* **2018**, *49*, 93–106. [CrossRef]
2. Garbie, H. A methodology for the reconfiguration process in manufacturing systems. *J. Manuf. Technol. Manag.* **2014**, *25*, 891–915. [CrossRef]
3. Huang, S.; Wang, G.; Shang, X.; Yan, Y. Reconfiguration point decision method based on dynamic complexity for reconfigurable manufacturing system (RMS). *J. Intell. Manuf.* **2018**, *29*, 1031–1043. [CrossRef]
4. Järvenpää, E.; Siltala, N.; Lanz, M. Formal resource and capability descriptions supporting rapid reconfiguration of assembly systems. In Proceedings of the 2016 IEEE International Symposium on Assembly and Manufacturing, Fort Worth, TX, USA, 21–22 August 2016; pp. 120–125.
5. Jiang, J.-R. An improved Cyber-Physical Systems architecture for Industry 4.0 smart factories. In Proceedings of the 2017 International Conference on Applied System Innovation (ICASI), Sapporo, Japan, 13–17 May 2017; pp. 918–920. [CrossRef]
6. Pirvu, B.-C.; Zamfirescu, C.-B.; Gorecky, D. Engineering insights from an anthropocentric cyber-physical system: A case study for an assembly station. *Mechatronics* **2016**, *34*, 147–159. [CrossRef]
7. Van Brussel, H.; Wyns, J.; Valckenaers, P.; Bongaerts, L.; Peeters, P. Reference architecture for holonic manufacturing systems: PROSA. *Comput. Ind.* **1998**, *37*, 255–274. [CrossRef]

8. Leitão, P.; Restivo, F. ADACOR: A holonic architecture for agile and adaptive manufacturing control. *Comput. Ind.* **2006**, *57*, 121–130. [CrossRef]
9. Shen, W.; Norrie, D.H. Agent-Based Systems for Intelligent Manufacturing: A State-of-the-Art Survey. *Knowl. Inf. Syst.* **1999**, *1*, 129–156. [CrossRef]
10. Macherki, D.; Diallo, T.M.; Choley, J.-Y.; Guizani, A.; Barkallah, M.; Haddar, M. The Q-Holon: A quadridimensional holon to design and operate an adaptive and scalable architecture for CPPS. In Proceedings of the 2020 21st International Conference on Research and Education in Mechatronics (REM), Cracow, Poland, 9–11 December 2020; pp. 1–6.
11. Yli-Ojanperä, M.; Sierla, S.; Papakonstantinou, N.; Vyatkin, V. Adapting an agile manufacturing concept to the reference architecture model industry 4.0: A survey and case study. *J. Ind. Inf. Integr.* **2019**, *15*, 147–160. [CrossRef]
12. Derigent, W.; Cardin, O.; Trentesaux, D. Industry 4.0: Contributions of holonic manufacturing control architectures and future challenges. *J. Intell. Manuf.* **2020**, *32*, 1797–1818. [CrossRef]
13. Zezulka, F.; Marcon, P.; Vesely, I.; Sajdl, O. Industry 4.0—An Introduction in the phenomenon. *IFAC-PapersOnLine* **2016**, *49*, 8–12. [CrossRef]
14. Bagheri, B.; Yang, S.; Kao, H.-A.; Lee, J. Cyber-physical Systems Architecture for Self-Aware Machines in Industry 4.0 Environment. *IFAC-PapersOnLine* **2015**, *48*, 1622–1627. [CrossRef]
15. Steindl, G.; Stagl, M.; Kasper, L.; Kastner, W.; Hofmann, R. Generic Digital Twin Architecture for Industrial Energy Systems. *Appl. Sci.* **2020**, *10*, 8903. [CrossRef]
16. Bagheri, B.; Rezapoor, M.; Lee, J. A unified data security framework for federated prognostics and health management in smart manufacturing. *Manuf. Lett.* **2020**, *24*, 136–139. [CrossRef]
17. Ahmadi, A.; Sodhro, A.H.; Cherifi, C.; Cheutet, V.; Ouzrout, Y. Evolution of 3C Cyber-Physical Systems Architecture for Industry 4.0. In *Service Orientation in Holonic and Multi-Agent Manufacturing*; Borangiu, T., Trentesaux, D., Thomas, A., Cavalieri, S., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 448–459. [CrossRef]
18. Cardin, O.; Derigent, W.; Trentesaux, D. Evolution of holonic control architectures towards Industry 4.0: A short overview. *IFAC-PapersOnLine* **2018**, *51*, 1243–1248. [CrossRef]
19. Valckenaers, P. ARTI Reference Architecture-PROSA revisited, Service Orientation in Holonic and Multi-Agent Manufacturing, Service Orientation in Holonic and Multi-Agent Manufacturing. *Stud. Comput. Intell.* **2019**, *803*, 119.
20. Valckenaers, P. Perspective on holonic manufacturing systems: PROSA becomes ARTI. *Comput. Ind.* **2020**, *120*, 103226. [CrossRef]
21. Barbosa, J.; Leitão, P.; Adam, E.; Trentesaux, D. Dynamic self-organization in holonic multi-agent manufacturing systems: The ADACOR evolution. *Comput. Ind.* **2015**, *66*, 99–111. [CrossRef]
22. Chirn, J.-L.; McFarlane, D.C. A holonic component-based approach to reconfigurable manufacturing control architecture. In Proceedings of the 11th International Workshop on Database and Expert Systems Applications, London, UK, 4–8 September 2000; pp. 219–223.
23. Pach, C.; Berger, T.; Bonte, T.; Trentesaux, D. ORCA-FMS: A dynamic architecture for the optimized and reactive control of flexible manufacturing scheduling. *Comput. Ind.* **2014**, *65*, 706–720. [CrossRef]
24. Jimenez, J.-F.; Bekrar, A.; Zambrano-Rey, G.; Trentesaux, D.; Leitão, P. Pollux: A dynamic hybrid control architecture for flexible job shop systems. *Int. J. Prod. Res.* **2017**, *55*, 4229–4247. [CrossRef]
25. Cruz Salazar, L.A.; Ryashentseva, D.; Lüder, A.; Vogel-Heuser, B. Cyber-physical production systems architecture based on multi-agent's design pattern—Comparison of selected approaches mapping four agent patterns. *Int. J. Adv. Manuf. Technol.* **2019**, *105*, 4005–4034. [CrossRef]
26. Williams, T.J. The Purdue Enterprise Reference Architecture. *IFAC Proc. Vol.* **1993**, *26*, 559–564. [CrossRef]
27. Kosanke, K. CIMOSA—Overview and status. *Comput. Ind.* **1995**, *27*, 101–109. [CrossRef]
28. Jeschke, S.; Brecher, C.; Meisen, T.; Özdemir, D.; Eschert, T. Industrial internof things and cyber manufacturing systems. In *Industrial Internof Things*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 3–19.
29. Brennan, R.W.; Balasubramanian, S.; Norrie, D.H. Dynamic control architecture for metamorphic control of advanced manufacturing systems. In *Architectures, Networks, and Intelligent Systems for Manufacturing Integration*; Division of Manufacturing Engineering, The University of Calgary 2500 University Dr. NW: Calgary, AB, Canada, 1997. [CrossRef]
30. Meissner, H.; Ilsen, R.; Aurich, J.C. Analysis of Control Architectures in the Context of Industry 4.0. *Procedia CIRP* **2017**, *62*, 165–169. [CrossRef]
31. Balzereit, K.; Niggemann, O. Automated reconfiguration of cyber-physical production systems using satisfiability modulo theories. In Proceedings of the 2020 IEEE Conference on Industrial Cyberphysical Systems (ICPS), Tampere, Finland, 10–12 June 2020; Volume 1, pp. 461–468.
32. Müller, T.; Walth, S.; Jazdi, N.; Weyrich, M. Identification of Reconfiguration Demand and Generation of Alternative Configurations for Cyber-Physical Production Systems. In *Advances in Automotive Production Technology—Theory and Application*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 63–70.
33. Müller, T.; Lindemann, B.; Jung, T.; Jazdi, N.; Weyrich, M. Enhancing an Intelligent Digital Twin with a Self-organized Reconfiguration Management based on Adaptive Process Models. *arXiv* **2021**. [CrossRef]

34. Epureanu, B.I.; Li, X.; Nassehi, A.; Koren, Y. Self-repair of smart manufacturing systems by deep reinforcement learning. *CIRP Ann.* **2020**, *69*, 421–424. [CrossRef]
35. Xu, P.; Xu, S.; Yin, H. Application of self-organizing competitive neural network in fault diagnosis of suck rod pumping system. *J. Pet. Sci. Eng.* **2007**, *58*, 43–48. [CrossRef]