electronics

Review

The Role of Mixed Criticality Technology in Industry 4.0

José Simó , Patricia Balbastre *, Juan Francisco Blanes , José-Luis Poza-Luján and Ana Guasque

Instituto de Automática e Informática Industrial (AI2), Universitat Politècncia de València, 46022 Valencia, Spain; jsimo@ai2.upv.es (J.S.); pblanes@ai2.upv.es (J.F.B.); jopolu@ai2.upv.es (J.-L.P.-L.); anguaor@ai2.upv.es (A.G.)
* Correspondence: patricia@ai2.upv.es

Abstract: Embedded systems used in critical systems, such as aeronautics, have undergone continuous evolution in recent years. In this evolution, many of the functionalities offered by these systems have been adapted through the introduction of network services that achieve high levels of interconnectivity. The high availability of access to communications networks has enabled the development of new applications that introduce control functions with higher levels of intelligence and adaptation. In these applications, it is necessary to manage different components of an application according to their levels of criticality. The concept of "Industry 4.0" has recently emerged to describe high levels of automation and flexibility in production. The digitization and extensive use of information technologies has become the key to industrial systems. Due to their growing importance and social impact, industrial systems have become part of the systems that are considered critical. This evolution of industrial systems forces the appearance of new technical requirements for software architectures that enable the consolidation of multiple applications in common hardware platforms—including those of different criticality levels. These enabling technologies, together with use of reference models and standardization facilitate the effective transition to this approach. This article analyses the structure of Industry 4.0 systems providing a comprehensive review of existing techniques. The levels and mechanisms of interaction between components are analyzed while considering the impact that the handling of multiple levels of criticality has on the architecture itself—and on the functionalities of the support middleware. Finally, this paper outcomes some of the challenges from a technological and research point of view that the authors identify as crucial for the successful development of these technologies.

Keywords: embedded control systems; hypervisors; distributed systems; mixed-criticality systems; industry 4.0

check for updates

Citation: Simó, J.; Balbastre, P.; Blanes, J.F.; Poza-Luján, J.-L.; Guasque, A. The Role of Mixed Criticality Technology in Industry 4.0. *Electronics* **2021**, *10*, 226. https://doi.org/10.3390/electronics10030226

Academic Editor: Osvaldo Gervasi
Received: 2 December 2020
Accepted: 15 January 2021
Published: 20 January 2021

1. Introduction

Industry 4.0 expresses a hypothetical fourth stage of the technical-economic evolution of humanity, counting from the First Industrial Revolution. This fourth stage would have started recently, and its development would be projected towards the third decade of the 21st century. Artificial intelligence is pointed out as a central element of this transformation, closely related to the growing accumulation of large amounts of data ("big data"), the use of algorithms to process them, and the massive interconnection of digital systems and devices. From a practical point of view, the evolution towards these new industrial systems is characterized by a large-scale interaction between machines (M2M) and the massive use of data with the aim of achieving flexible production systems, customer-oriented convertible factories, optimization in the use of resources, and circular economy ecosystems.

In this scenario, it is necessary to structure the digitization of means of production so that the management and communication of large amounts of information can coexist with critical automation systems dominated by real-time and security restrictions.

The market for real-time embedded systems has expanded quickly in recent years and is expected to grow for the foreseeable future. This evolution of the technology of embedded systems in terms of computing capacity, connectivity, and performance has been very significant and has enabled the use of more complex control applications that can require more complex design and implementation techniques. Today, all hardware platforms of embedded systems are multicore, and this enables the use of new technologies (e.g., artificial intelligence) [1].

Control activities have traditionally been designed with various tasks that perform the control functions, visualization, and interaction with other devices. All these activities have different levels of criticality due to temporary restrictions or the implication of possible failures and their consequences. The mixed-criticality systems approach seeks to organize in a coherent way the different activities according to their criticality level. Moreover, multi-core computing platforms ideally enable co-hosting applications with different requirements (e.g., high data processing demand and stringent time criticality). Executing non-safety and safety critical applications on a common powerful multi-core processor is of paramount importance in the embedded system market for achieving mixed-criticality systems. This approach enables scheduling a higher number of tasks on a single processor so that the hardware utilization is maximized, while cost, size, weight, and power requirements are reduced. In [2], a survey was presented on mixed criticality in control systems.

In the industrial sector, digital transformation is maintaining and increasing competitiveness. The integration of technologies based on the industrial internet of things (IIOT) and technologies of Industry 4.0 will enable the digital transformation of the latter [3,4]. Intelligent factories, energy systems, and other critical infrastructures are adopting real-time monitoring and analysis capabilities to assess operational efficiency and can incorporate predictive analysis and maintenance to minimize idle time due to system failures. One of the crucial elements in digital transformation is the ability to monitor network control systems in real time. IT solutions attempt to achieve greater efficiency by consolidating common hardware platforms and remote operating capabilities. Recent market trends are forcing industrial manufacturers to look at other software solutions in order to incorporate more non-real-time functionality around crucial real-time tasks in order to accomplish goals, such as cloud connectivity for uploading machine data for server-based predictive maintenance algorithms. Virtualization seems to be a promising path forward, but with the multiple types of virtualization solutions out there, deciding which one to use requires in-depth discussion and thought [5].

Other sectors such as aerospace and defense have used some of these technologies successfully in the design of control systems for avionics components. The integrated modular avionics (IMA) architectures are an evolution of distributed systems to federated systems, and enable integrating multiple applications with different criticality levels in the same hardware platform. The adoption of IMA has resulted in a significant reduction in the cabling, weight, and energy consumption of computer equipment in aircraft, satellites, and drones. The European Space Agency (ESA) has promoted the adaptation of IMA to cover space market needs. The IMA-SP (IMA for Space) project [6] has defined a partitioned architecture with additional services to ARINC-653 standard to deal with new future software developments.

Temporal and space partitioning (TSP) preserve the fault containment properties and "separation of concerns" in development. The functional benefits are related to: The allocation of different criticality/security classes that coexist within the same computer; management of the growth of software functionality; higher degrees of integration as better performing processors becomes available; and easier design for re-use [7].

Objectives and Organization

The main objective of this article is to propose the technology of mixed criticality, hypervisors, and execution isolation, as a key piece in the deployment of industrial digitalization systems, known as Industry 4.0 or industrial internet of things. The difference in requirements between the different levels and components in which the reference architectures are organized leads to an organization based on distributed systems with dedicated hardware elements. The evolution of these systems finds similarities with that experienced by the aeronautical sector (ARINC-653) [8] which, through the use of mixed criticality systems, has managed to optimize the use of hardware resources while maintaining compliance with requirements, certification and very high levels of flexibility.

This paper differs from a traditional review paper in that it aims to bring together the different technologies commented in the previous paragraph. In this sense, we differ from [9] in that we analyze the technologies used to implement mixed-criticality systems that can be useful to implement Industry 4.0 systems.

After this introductory section, the second section establishes the general structure of the system architecture and identifies the levels of interaction, the specific requirements of each level, as well as the technologies associated with each. The third section describes the proposed execution platform and the advantages of isolating the execution. The next section analyses the characteristics of the support middleware and its location in the architecture. To conclude, a set of technological and research challenges are elaborated. Finally, the conclusions of the paper are summarized.

2. Digitalization and Industry 4.0

The transition to Industry 4.0 will depend on the successful adaptation of a set of technologies that enable interconnecting different levels of control and management in an industrial environment.

Figure 1 shows an overview of the system with the various levels and components.

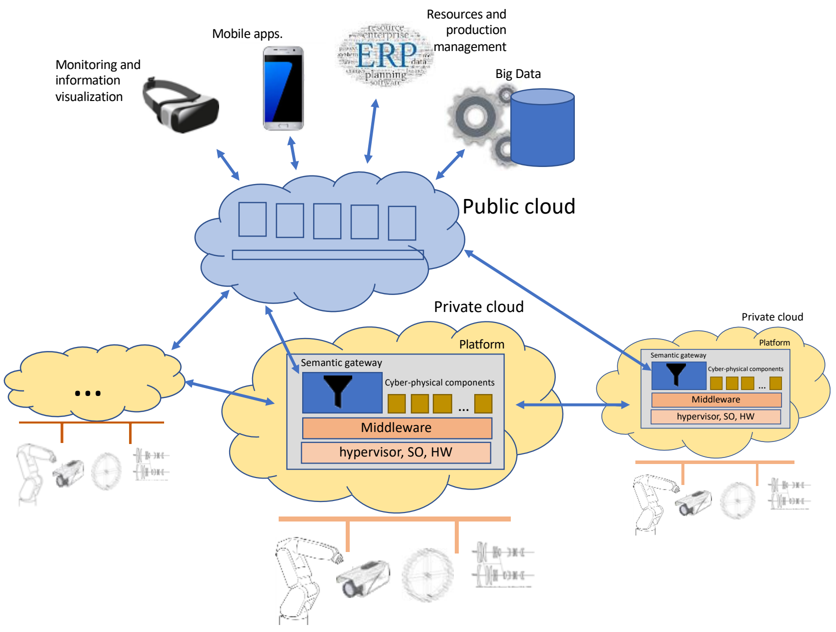

Figure 1. General view of a system for industry digitalization.

The lower level is composed of a set of cyber-physical systems (CPS) each within a private cloud that gathers and combines information from physical systems and provides access to control devices. The CPSs are interconnected through a horizontal communication between private clouds.

The technologies with a strong impact at this level are related to: The control of devices; mechanisms for access and action on devices forming the internet of things; systems based on models to organize and structure information configuring the reference models for industry 4.0; and execution systems that exploit the capacities of the hardware (multicore systems) and organize applications on a common platform in which real time activities interact with physical devices.

The upper level is structured on the cloud and publishes a set of services that aim to provide certain levels of information to the outside and connect this information with services and processes for the management, planning, and visualization of components and devices at other levels.

2.1. Cyber-Physical System Level

The cyber-physical system is one of the key enablers for the realization of Industry 4.0. It refers to an embedded system with real-time functionalities, control, access to devices, communications, high computing capacity, and user interaction. The CPS level groups the components and services for the real time operation of the physical devices that compose the industrial entity (section). The system, guided by an execution platform based on a multicore system, allows the definition of different execution environments that cover the needs for real time, security, confidentiality, and certifiability of the applications that make up the system in which applications with different levels of criticality can coexist. Figure 2 shows a more detailed vision of the CPS level.

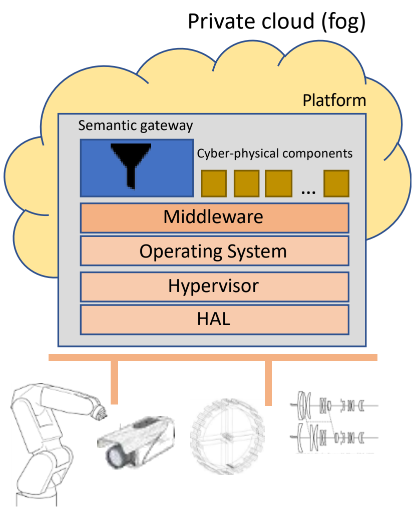

Figure 2. Cyber-physical system level.

At this level, the different CPSs interact with each other through efficient, safe, and predictable horizontal communication. At the same time, these CPSs communicate with the higher level through a vertical communication that enables the development of the strategic and business vision of the company.

The main requirements that are needed at this level can be summarized in the following points.

- Real time capabilities: The embedded system performs control functions by reading sensors and acting on the process with the need to fulfil the time constraints associated with each of the processes or sub-processes that comprise the industrial environment. Real-time activities require real-time operating systems in order to have a predictable behavior and determine system schedulability.
- Support application with different levels of criticality: The evolution from distributed systems to federated systems (in which several applications are executed on the same hardware platform) enables various applications to be executed in isolation with differing time restrictions and levels of certifiability. The ability to handle applications with different levels of criticality on the same platform enables the new multicore hardware platforms to be exploited to the maximum.
- Security and safety: There is an increasing need to securely manage applications. A secure initialization of the platform and verification of integrity is a prerequisite. Secure communications and detection and resistance to cyberattacks are required in operation.

- Application reuse: Applications or components developed in previous projects should be reusable on new embedded computing platforms provided by multi-core processor architectures.
- Communication: At the cyber-physical level, the machines are connected to other machines, material flow management systems, and inspection systems to form a unit that works in a coordinated and highly reconfigurable manner.

The following enabling technologies can be considered basic to the development and operation of this level: Platform virtualization (fault management, temporal, and spatial isolation), industrial internet of things, robotics, cyber-security, fog and edge computing, simulation, and middleware.

As stated in the requirements, many CPS are also mixed-criticality [10]. For example, smart buildings [11] or eHealth IoT systems [12] are works that develop implementations of CPS with mixed criticality systems (MCS) requirements.

2.2. Information Aggregation Level

The machines, robots, and manufacturing resources integrated in Industry 4.0 applications generate an immense volume of communication—both horizontally and vertically.

Vertical integration of automated production is necessary to enable constant process monitoring and integration of additional IoT services. Services such as data analysis, predictive maintenance, or simple access to digital user guides and helpdesks, are customized for the specific machine and integrated directly into the system. Access is granted to all who need it, in a personalized way and, if necessary, through cloud applications. Any external supplier who needs to be integrated into the overall system can obtain specific interfaces for this purpose.

It is important to emphasize that each of the original control tasks of the individual I4.0 components is an essential part of the solution and an important communication task and source of information.

Data and control flows take place in and between the cyber-physical and information integration functional domains. The controls, coordination, and orchestration exercised from each of the functional domains have different granularities and are executed in different time cycles. As it starts in the functional domains, the interactions become coarser, their cycle becomes longer, and the scope of the impact is likely to become larger. Consequently, as information advances in the functional domains, the scope of the information becomes broader and richer, and so new information can be obtained, and new intelligence can emerge in broader contexts. Each functional domain is characterized by the definition of a connectivity model and a computational deployment pattern.

The starting point is the control domain, which is where control tasks are performed, the lowest level information is generated, and time and reliability constraints are more restrictive. The pattern of computational deployment at this level is closely linked to the horizontal connectivity model, and both, working in a coordinated manner, are in charge of the interaction with the physical world composed of machinery and manufacturing resources.

For a satisfactory integration between the cyber-physical level and the information integration level, we propose the development of gateways between levels that make compatible the different quality of service requirements and offer solutions for information management according to the chosen computational deployment pattern—including private cloud and public cloud systems (Figure 1). It is essential to develop information conversion and integration models that, through automatic processing technology, prepare the information according to the context in which it will be used.

Enabling technologies at upper levels include: Big data and analytics, cloud computing, augmented reality, cyber-security monitoring, and deep learning.

3. CPS Platform

Over the last decade, advances in processor technologies have had a strong impact on the architecture of processors with a special emphasis on the processors used in embedded systems. These advances have included the generalization of multicore architectures and support for virtualization at the hardware level.

Virtualization has enabled cloud computing platforms to host applications in the cloud efficiently and on a large scale. Hardware virtualization support in embedded systems has allowed the improvement of hypervisor performance for critical systems offering several isolated execution environments (called partitions with their own operating system and application) on the same hardware platform. Figure 3 shows the architecture of an embedded I4.0 system with hypervisor and partitions.

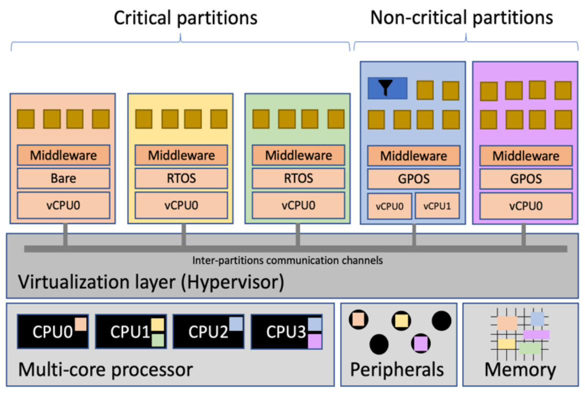

Figure 3. Overview of an embedded I4.0 component in a mixed-criticality context.

Virtualization techniques are the basis for building partitioned software architectures [13]. The virtualization layer is the software layer that virtualizes computing resources. It can be defined and used to manage application or system resources. The hypervisor (also known as a virtual machine monitor (VMM)) implements the virtualization layer of the software and enables several independent operating systems to run their applications on a single hardware platform.

The purpose of the hypervisor is to efficiently virtualize available resources. One of its required features is that it must introduce a low overhead; therefore, the performance of the applications running on the virtualized system must be similar to that of the same applications running on the native system. In the environment of critical systems, the hypervisor is the layer on top of the hardware that has to offer the necessary characteristics for the execution of real-time systems with a high level of integrity and safety.

To run several isolated execution environments, the hypervisor must cover:

- Fault isolation: A fault in one application must not spread to other applications. Any failure must be resolved by the application itself or by the hypervisor.
- Spatial isolation: Applications must run in independent physical memory address spaces. The hypervisor must ensure that the applications cannot access any memory area that has not been specifically assigned to them.
- Temporary isolation: The real-time behavior of an application must be correct independently of the execution of other applications. The allocation of system resources to one application is not influenced by others and can be analyzed independently.

The availability of a platform on which completely independent and isolated partitions such as the one provided by a hypervisor can be executed is the basis for the design and development of mixed criticality systems. Each partition containing an application with its operating system may have a different level of criticality. Criticality level means the level of exigency or safety required by the application that is performing control functions over the processes.

There have been some hypervisors that have been used to implements MCSs since they provide the correct level of isolation to applications of different criticality. The MULTIPARTES and DREAMS architectures [14,15] is one of the approaches for the development of MCS under realistic assumptions.

In [16], RTA-HV is used to provide virtualization support for a multicore hardware platform for the automotive industry. In [17], VOSYS monitor is used as virtualization technology, a multi-core software layer, which allows the co-execution of a safety-critical real-time operating system (RTOS) and a noncritical general purpose operating system (GPOS) on the same hardware ARMv8-A platform. Last release (v2.5.0) is ASIL-C-ISO 26,262 certified [18]. A hypervisor for a mixed criticality on-board satellite software system is discussed by Salazar et al. in [19].

Partitioning is implemented for Components Of The Shelf (COTS) Network On Chip (NoC)-based MultiProcessor System On Chip (MPSoC) for the mixed criticality systems, in the safety critical field in [13]. The proposed technique was developed as a software module to be interred in a certified RTOS for avionics systems.

4. Scheduling in MCS

The first and most popular model for mixed-criticality systems in the real-time community was proposed by Vestal [20]. In his proposal, task criticality level is high (HI) or low (LO). This model assumes that HI tasks have two different computations times: If an overload occurs, a mode change is forced in which LO tasks are dropped, and HI tasks execute with their lowest computation time.

Since Vestal seminal paper, a lot of works addressed real-time scheduling of MCS. The first of them assuming mono processor and independent tasks [21,22]. In multiprocessor systems, the first work to include MCS was [23]. The implementation assumes five levels of criticality and uses static allocation for the higher criticality level and apply different isolation techniques for each level. Regarding task allocation in partitioned multiprocessor systems, traditional bin packing algorithms are also used with MCS. In [24], it was proved that First-Fit with decreasing criticality is the allocation algorithm that obtains best results in homogeneous processors. Regarding schedulability analysis.

A novel approach was proposed in [25] in which a run-time Worst Case Execution Time (WCET) controller monitor the interference caused by LO tasks. It can decide to suspend LO tasks if the interference jeopardizes the execution of HI tasks. They evaluate the proposal on a COTS. A similar approach was proposed in [26] for partitioned systems based on hypervisors.

Vestal's model has been the subject of much controversy [27], mainly because it is not clear why a certification authority would accept two values for the computation time of a task and two different processes for measuring them. For these reason, Vestal model is not being used in industrial applications since industry practice and safety standards provide a different perspective in MCS. This has been discussed in some papers [28–30].

In [31], a mixed-criticality model was proposed in which criticality level is associated to partitions. These can have the following levels of criticality:

- HI: Partitions have to be executed to completion in any condition and temporal constraints of their tasks have to be fulfilled.
- LO: Partitions can be executed in some conditions, depending on the impact caused in the system for not being executed. No temporal constraints are identified but it is expected a bandwidth for them.
    - If, in some cases, they can be dropped, we call them disposable LO partitions (DLO).
    - If they have to be executed in any case, they are called required LO partitions (RLO). However, even if these partitions cannot be dropped, their bandwidth can be reduced if needed

DLO and RLO partitions can be dropped or their bandwidth can be reduced, for example, for energy saving purposes.

As they are totally isolated, partitions can be independently certified at the appropriate criticality level according to the application's requirements. In the same line, if an application is modified, the recertification process will only affect the application itself and not the rest of the partitions. This is called incremental certification. Figure 4 shows and example on how incremental certification works. For the sake of simplicity, we assume a mono-core processor.

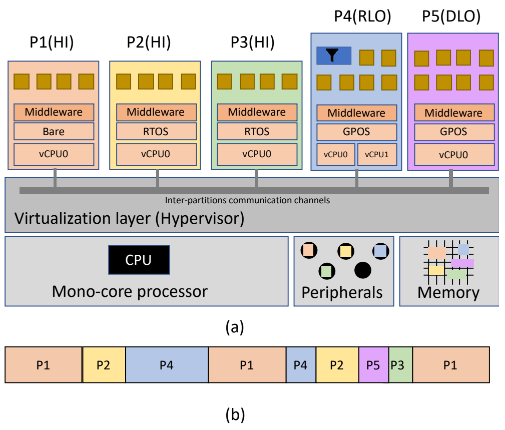

Figure 4. Example of a system with 5 partitions and different criticality levels. (**a**) architecture and (**b**) CPU allocation.

In Figure 4a we can see a partitioned system with five partitions (P1, . . . , P5). P1 to P3 are partitions with the highest level of criticality (HI), while P4 and P5 are LO criticality partitions. In the case of P4, it can be dropped if needed and, as far as P5 is concerned, its bandwidth can be reduced.

Regarding the temporal requirements, Figure 4b shows the CPU allocation of partitions. This allocation has to be statically allocated (off-line) in order to assure the temporal isolation that is mandatory for certification purposes.

Let us suppose there is a change in P2 requirements so that more CPU time is needed for this partition. Be P2′ the old P2 partition with the new requirements. A change in requirements means rebuilding the CPU allocation, resulting in a new schedule for all partitions. However, due to temporal isolation property, P2′ can be allocated in P2 slots without having to change the allocation of the other partitions. Figure 5 shows two possible scenarios for this situation. In Figure 5a, P2′ temporal requirements can be allocated in P2 slots without any modifications. However, let us suppose that these slots are not enough to allocate P2′. Then, we can use criticality of partitions, specifically, we can drop P5 or reduce P4 bandwidth. Figure 5b shows the resulting allocation of the new system where P2′ uses slots from the old P2 and the freed slots due to P4 bandwidth reduction.

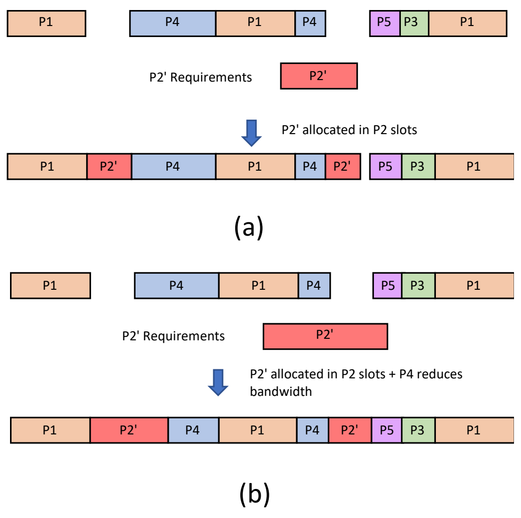
P1 P4 P1 P4 P5 P3 P1
P2' Requirements
P2'
P2' allocated in P2 slots
P1 P2' P4 P1 P4 P2' P5 P3 P1
(a)
P1 P4 P1 P4 P5 P3 P1
P2' Requirements
P2'
P2' allocated in P2 slots + P4 reduces bandwidth
P1 P2' P4 P1 P4 P2' P5 P3 P1
(b)

Figure 5. Two scenarios of changes in P2 requirements: (**a**) P2′ requirements are enough to be allocated in P2 slots. (**b**) P2′ requirements need also that P4 reduces its allocation.

In any case, it avoids having to re-certify the entire system. For this reason, incremental certification has less economic and development costs while avoiding errors due to changes in requirements.

An example of a hypervisor for MCS on multicore platforms is XtratuM. XtratuM [32,33] is an open-source hypervisor that offers the features required for the development of partitioned systems for critical high integrity applications. XtratuM has been successfully used in several satellites and is the basis of several satellite constellations. In addition, XtratuM has been used in several research projects in which application demonstrators have been developed in automotive environments, railways, and wind generation systems (Trujillo et al., 2013). when dealing with multicore systems, the isolation, mainly temporal, can be impacted by shared resources in the system (bus access, memory, etc.). In [2], it was pointed out some of the problems emerged. An alternative is to introduce in the hypervisor control schemes that can control the partition execution. In [34], control schemes inside the hypervisor have been proposed to control the execution of tasks in multicore systems in which shared resources introduce unpredictability in the fulfilment of temporal constraints.

In addition to ensuring the isolation of critical systems, it is also necessary to manage redundancy. Traditionally, distributed control systems have managed redundancy by replicating the hardware platform, as shown in Figure 6a. In this example, a triple redundancy of critical systems is considered necessary. This redundancy requires the deployment of 11 hardware platforms. The power of the current execution platforms enables the use of virtualization as presented in Figure 6b so that only three hardware platforms are required. This configuration offers multiple advantages: Lower cost; less deployment effort; reduction of cabling; and, in general, greater simplicity since part of the complexity managed by hardware is managed by software.

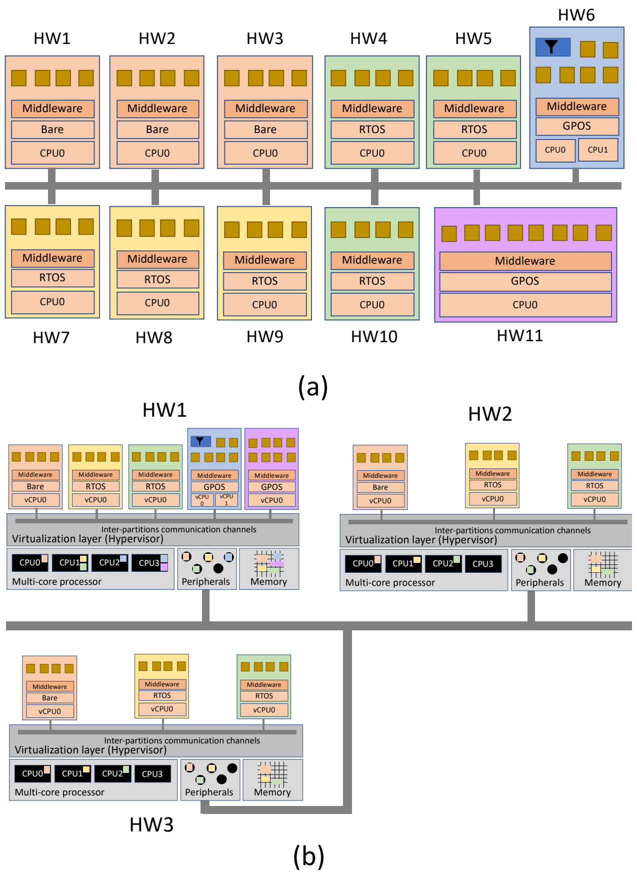

Figure 6. (**a**) Traditional redundancy management. (**b**) Redundancy management in a virtualization context.

5. The Role of the Middleware

Achieving the objectives of Industry 4.0 requires the implementation of large-scale distributed systems. Reconfiguration through digital machine-to-machine interaction involves the implementation of distributed control systems following the line marked by the IEC61499 standard [35]. The information that different distributed control elements exchange for normal operation must be elevated to levels of supervision and analysis to implement the necessary coordination with higher levels of decision and optimization. Various information interactions can be seen in in Figure 1, where information of several and different devices are managed in different levels.

To achieve true interoperability and reconfiguration between elements produced by different manufacturers, it is essential to establish reference models that define the architecture of a digitized industrial system. The association "Industrie 4.0", in Europe [36] proposed the Reference Architecture Model for Industry 4.0 (RAMI) model that provides architectural guides at all levels of an industrial system. The RAMI model is organized in three axes [37] where the main axis, "the factory", represents the classical hierarchical levels of the physical organization of the production system as set out in ISA-95 and, more specifically, the models for integration as defined in IEC 62264. The second axis, "layering", indicates the context in which information is produced or used. This axis represents the logical organization of the industrial system from the product through the means of production to the business processes. The third axis, "the product", indicates the life cycle of the product and highlights the importance of managing production as well as design, development, and possible recycling, and the residual value or environmental impact of a product at the end of its useful life.

From the point of view of software organization, the RAMI reference model (Figure 7) proposes an architecture based on services whose interfaces, information models, and meta-models are driven by standards (IEC 61369, IEC 62,264 IEC CDD) [38,39] that enable their precise interpretation. The basic element of the architecture is the "asset", which represents an object that has value for the company, and its software representation, the "management shell" that organizes the properties and functions of the asset from different points of view. The combination of the asset with its management shell forms the "component" in the context of Industry 4.0. See Figure 8.

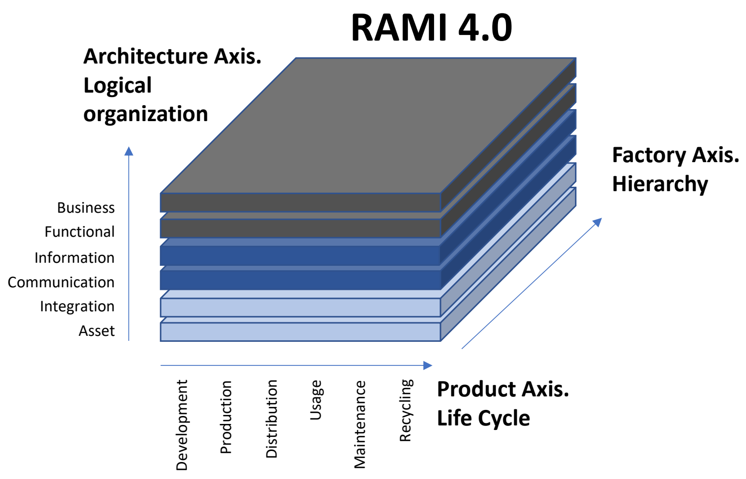

Figure 7. RAMI 4.0 oveview

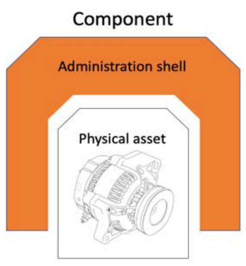

Figure 8. RAMI component.

An I4.0 component offers different interfaces and properties depending on the point of view from which its functionality is managed. For example, a booster pump can be observed from a mechanical point of view (such as size, weight, position of the inlet and outlet terminals, and position and resistance of the anchors), hydrodynamic (including pressure/flow curve, viscosity range, and tolerance to particles or corrosives), or electrical (such as type of motor, power, supply voltage, and protection index). Each of the perspectives that a component must implement must have an associated descriptive model. These models, in an ideal situation, must have the capacity and be sufficiently complete so that, through their automatic processing, the interfaces for M2M interaction and automatic reconfiguration can be generated. Additionally, a component can be formed by other components and establish a hierarchical relationship between them—as well as with the relationships between components at the same level that are configured to form part of a higher component. In this hierarchical relationship, it is necessary to implement the specification of models through the aggregation of sub-models and connections between standardized properties of components.

This summary view of the RAMI reference model gives an idea of the extraordinary complexity of the task of approaching the complete implementation of a middleware system that serves as a platform for the development of I4.0 applications. There are currently experimental prototypes in the field of research [40] and industrial products [41] that, following the ecosystem defined by the manufacturer, enable the development of applications with a many of the properties demanded by current industry (such as modularity, flexibility, digital twin, and data analysis).

In this way of approaching the ideal I4.0 platform, the development of middleware technology is key in the set of enabling technologies. Specifically, a middleware technology should provide infrastructure for the management of the exchange of information between the different components and the control of the execution of the agents or active elements of information processing. This is always within a framework of standardization that enables interoperability, the exhaustive exploitation of model-driven engineering, and considering cybersecurity as a central aspect.

Horizontal interaction enables communication between strongly coupled close control elements and is characterized by a high criticality and the relevance of real time constraints both in communications and in the execution of tasks in the building blocks of distributed control. Vertical interaction consists of the exchange of information between different areas or levels of abstraction in the processing of information.

The middleware for the management of information exchange must ensure the ubiquity of the information and ensure the horizontal and vertical interaction of the components as presented in Figure 9. Middleware must be integrated with the isolation mechanisms provided by partitioned systems that manage the execution of components in a mixed-criticality environment. In other words, middleware must manage the trade-off of having to ensure isolation between components while also ensuring interaction between them through information exchange.

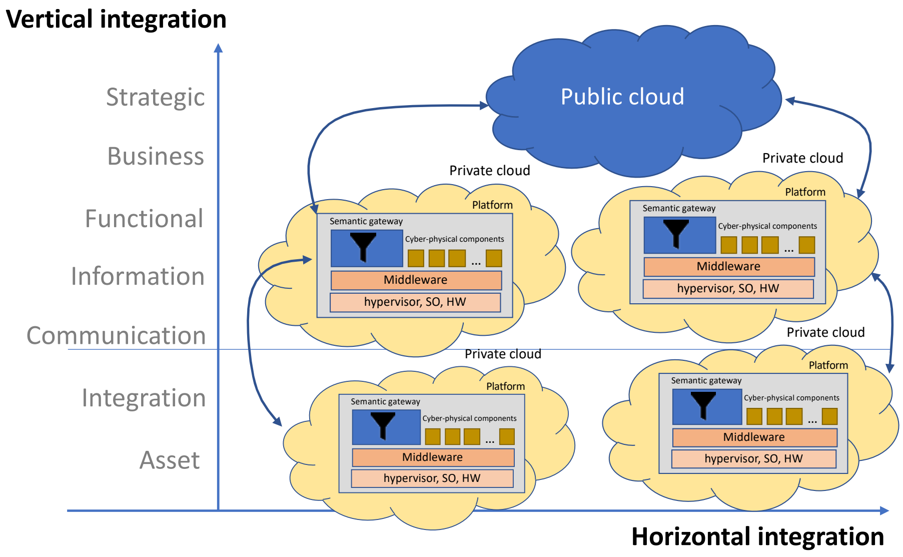

Figure 9. Horizontal and vertical components integration.

The service-based interaction proposed by the RAMI reference model is suitable for vertical interactions between components located at high levels of the architecture and close to the public cloud [42]. The current RAMI implementations [43] use some form of REpresentational State Transfer (REST) web services as infrastructure. This technology is not suitable for critical horizontal interactions, with time constraints typical of distributed control systems and whose communication model consists of the propagation and processing of events. As a consequence, the communications middleware must support both synchronous client/server Service-Oriented Architecture (SOA) models, interactions between distributed objects under real time restrictions such as RT-CORBA, and asynchronous models focused on publication/subscription data, such as DDS [44]. In any case, regardless of the interaction model used, the information must be characterized in semantic terms to support automatic reconfiguration by establishing connections between component properties. In Figure 9 we can see a component showed in Figure 2, the "semantic gateway", whose purpose is the monitoring of horizontal communications and their vertical propagation. These gateways work by configuring information retainers, aggregators, and converters, offering new information synchronously through services, or by publishing asynchronously according to the quality of service specifications required at higher levels.

In short, the capabilities of the communications middleware have a strong impact on the characteristics that a given industrial digitization implementation can achieve. The most important functionalities that the communications middleware must provide are:

- Standardization through reference models (RAMI) and open protocols and compatibility mechanisms with legacy systems such as IEC16499.
- Support of mixed communication paradigms: SOA and DCPS.
- Processing, conversion, and transmission of information directed by semantic models.
- Support for reliable and isolated execution of components.
- Monitoring, recording, and encryption of information exchanged between secure execution environments. Isolation control.
- Distribution of the execution of components directed by criticality models.

6. Challenges

Both technological fields (MCS and I4.0) involve a significant number of areas of interest, each of which offers a set of specific challenges from both a technological and a research point of view.

In this paper we have focused on the most significant techniques that bring together both themes. Table 1 summarizes all the references of this paper organized by topics.

We have structured these topics in: Embedded systems or platforms, cyber-physical systems, I4.0, mixed criticality systems, and execution environment (operating systems and hypervisors).

Table 1. References organized by topics.

Topic	Embedded Systems/PLATFORMS	CPS	I4.0	MCS	Run-Time Environment
Embedded systems/Platforms		1, 11, 12, 35	38, 39, 40, 42, 43	2, 11, 12, 13, 14, 16, 18, 19, 25, 27, 28, 29, 30, 34, 44	6, 7, 8, 13,14, 15, 16, 17, 18, 19, 34
CPS			9	11, 12	10
I4.0			36, 37, 3, 4, 41, 43		5
MCS				20, 21, 22, 23, 24	8, 13, 14, 17, 23, 26, 31, 32, 33
Run-time environment					

As we can see in the table, we find many references to each topic individually but very few, if any, that bring together two or more topics. It is significant that we have not found any references that combine I4.0 with MCS.

Next, we analyze the challenges that, in our opinion, are more relevant for the integration of both fields of technology. These challenges are addressed under three main lines: Execution platforms, application integration, and model engineering.

6.1. Execution Platforms

In complex systems with multiple applications containing critical components, it is necessary to isolate the different applications in order to strongly restrict the interaction between them. This isolation allows failures that may occur in a system component or application to be isolated and not to propagate to other components or applications. On the basis of a strong isolation, it is possible to establish the foundation for the execution of applications with different levels of criticality on the same execution platform. The need to take full advantage of the functionalities offered by virtualization techniques (hypervisors) for the development of complex applications in an Industry 4.0 environment is a major challenge for the industry.

The massive use of multi-core computing platforms adds complexity to the system but the advantages it offers exceed the disadvantages. The increase in computing capacity combined with the use of a hypervisor layer allows migration of applications developed for single-core to partitions on top of the hypervisor with its own operating system. Models that allow the migration of these applications to multicore partitioned systems together with the analysis and configuration tools of the new environments in which the partitions have different levels of criticality are crucial for the development of this type of systems.

In addition to the use of the multicore system, it is essential to extract the maximum performance from the hardware. In this sense, it is crucial to conduct an optimal scheduling of the applications running on the multicore system. Scheduling techniques in a multicore system with the specifications of the partitions with their internal real time tasks requires a review of the techniques and tools of analysis of this type of system.

6.2. Application Integration

As previously mentioned, mixed criticality systems involve different applications (or partitions in the terminology of partitioned systems) with different levels of requirements from the point of view of safety, reliability and security. The hypervisor layer should allow to identify the criticality of the different partitions and to handle it appropriately. In Industry 4.0, common services to applications (such as middleware) coexist and, therefore, they must be integrated in the criticality model. The need to independently certify applications and those common services requires the development of specific techniques for the specification, verification, and validation of all components independently.

While the use of integration of applications with different levels of criticality is a recognized topic, there is limited analysis of the use of devices and their integration. In avionics a significant number of functions depend on sensors, actuators, and external information. In the processes of the Industry 4.0 this need to integrate sensors and actuators and information from other devices (robots, machine tools, etc.) of the industry is fundamental. This information is typically acquired through processors using different mechanisms from the traditional ones (memory subsystems, etc.) or as specialized interfaces. The integration of these interfaces associated with applications with levels of criticality and in a multicore environment require a deep analysis and restructuring.

6.3. Model Engineering

The development of applications in a complex environment requires modeling tools based on a reference architecture. The complete vision of the set of applications with different level of criticality, the deployment on a set of hardware platforms with diverse execution environments (partitioned with hypervisor, traditional, small cyber-physical systems) cannot be approached without a model-driven engineering (MDE) process in the set of tools for analysis and deployment in the factory.

The design of the system architecture and the corresponding assumptions and decisions have a great impact on aspects such as the use of time-restricted design that controls access to shared resources to ensure freedom of interference of mixed critical applications (as required by security standards) through temporal and spatial segregation, or the management of failure scenarios considered at design time (i.e., reconfiguration and degradation in case of hardware failure).

In order to achieve a more efficient development (e.g., to shorten the time to market and reduce the cost of certification), the use of possibly unspecified models that are collaboratively edited by engineers from different system domains working simultaneously on artefacts from different design phases (e.g., requirements specification and design models) would be interesting. This approach involves continuously checking the design constraints in order to draw parallels with the development process and to load the feedback on the design in the early stages.

Future MDE processes for MCS should allow for the use of non-specified artefacts as a preliminary basis for subsequent stages of the development process. For example, this would allow work to begin on system design until the requirements model has been completed. As also an incomplete requirements model defines the limitations, validation can be carried out immediately while the design model is being created. This approach allows efficient exploration of "state-of-the-art solutions", and exploits all the information about the system that has been modeled so far. An important requirement is the availability of tools that not only automatically derive and update refined artefacts (e.g., based on design rules), but also perform continuous validation and verification of the as yet unspecified system. This anticipation of validation and verification activities provides immediate feedback to the system engineers and helps to reduce the round-trip time during the design phase.

7. Conclusions

The impact of the digitization of industrial systems has opened a range of new opportunities for automation and adaptation of production processes. Some analysts consider that we are facing a new industrial revolution and the term "Industry 4.0" has been established to refer to the set of technologies and procedures related to the exploitation of production systems through the massive exchange and analysis of information. The advantages of digitization are associated with new problems to be solved, mainly in the areas of security, reliability, and control of information flows. Considering the specific requirements of I4.0 systems, this article has analyzed the current context of the technological development of mixed criticality systems. In this paper we propose the use of hypervisors as a possible solution for the execution of automation processes in a reliable way, and analyze the characteristics of a middleware platform that enables the integration of different execution supports. An architectural proposal for the I4.0 system has been drafted that is compatible with the ideas presented and with the reference models established by the international consortia. Finally, some technological and research challenges have been envisioned.

Author Contributions: J.S.: Methodology; Resources; Writing—Original draft; P.B.: Conceptualization; Supervision; Writing—Review and Editing; A.G.: Investigation; Visualization; Writing—Original draft; J.F.B.: Investigation; Visualization; Writing—Original draft; J.-L.P.-L.: Investigation; Visualization; Writing—Original draft. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Spanish Science and Innovation Ministry MICINN: CICYT project PRECON-I4: "Predictable and dependable computer systems for Industry 4.0" TIN2017-86520-C3-1-R.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. ARTEMIS. Embedded Intelligence: Trends and Challenges, a Study by Advancy, Commissioned by ARTEMIS Industry Association. 2019. Available online: https://artemis-ia.eu/publication/download/advancy-report.pdf (accessed on 30 November 2020).
2. Burns, A.; Davis, R. Mixed criticality systems–A review. In *Technical Report*; Department of Computer Science, University of York: York, UK, 2013.
3. IIC. The Industrial Internet of Things Volume G1: Reference Architecture. 2019. Available online: https://www.iiconsortium.org/IIRA.htm (accessed on 30 November 2020).
4. Kagermann, H.; Helbig, J.; Hellinger, A.; Wahlster, W. Recommendations for implementing the strategic initiative INDUSTRIE 4.0. In *Final Report of the Industrie 4.0 Working Group*; National Academy of Science and Engineering: München, Germany, 2013.
5. Kou, E. Achieving the Industrial Internet of Things through Virtualization. White Paper. Available online: https://www.5gtechnologyworld.com/achieving-the-industrial-internet-of-things-through-virtualization/ (accessed on 30 November 2020).
6. IMA-SP. IMA-SP. IMA-SP Integrated Modular Avionics for Space. In *ESA Project*; Coordinator: Astrium SAS; Contract ESTEC 4000100764, European Space Agency (ESA): Noordwijk, The Netherlands, 2011.
7. Windsor, J.; Hjortnaes, K. Time and space partitioning in spacecraft avionics. In *Space Mission Challenges for Information Technology, Proceeding of the Third IEEE International Conference on Space Mission Challenges for Information Technology, Pasadena, CA, USA, 19–23 July 2009*; IEEE: Piscataway Township, NJ, USA, 2009; pp. 13–20.
8. Airlines Electronic Engineering Committee (AEEC). *Avionics Application Software Standard Interface: ARINC Specification 653 Part 1*; Aeronautical Radio Inc.: Annapolis, MD, USA, 2013.
9. Alcácer, V.; Cruz-Machado, V. Scanning the Industry 4.0: A Literature Review on Technologies for Manufacturing Systems. *Eng. Sci. Technol.* **2019**, *22*, 899–919. [CrossRef]
10. Biondi, A.; Marinoni, M.; Buttazzo, G.; Scordino, C.; Gai, P. Challenges in virtualizing safety-critical cyber-physical systems. In Proceedings of the Embedded World Conference, Nuremberg, Germany, 27 February–1 March 2018; pp. 1–5.
11. Dimopoulos, A.C.; Bravos, G.; Dimitrakopoulos, G.; Nikolaidou, M.; Nikolopoulos, V.; Anagnostopoulos, D. A multi-core context-aware management architecture for mixed- criticality smart building applications. In Proceedings of the System of Systems Engineering Conference (SoSE), Kongsberg, Norway, 12–16 June 2016; pp. 1–6.
12. Kotronis, C.; Nikolaidou, M.; Dimitrakopoulos, G.; Anagnostopoulos, D.; Amira, A.; Ben-saali, F. A model-based approach for managing criticality requirements in e-health iot systems. In Proceedings of the 13th Annual Conference on System of Systems Engineering (SoSE), Paris, France, 19–22 June 2018; pp. 60–67.
13. Avramenko, S.; Violante, M. RTOS solution for noc-based COTS MPSoC usage in mixed- criticality systems. *J. Electron. Test.* **2019**, *35*, 29–44. [CrossRef]
14. Trujillo, S.; Crespo, A.; Alonso, A. Multi-PARTES: Multicore virtualization for mixed-criticality systems. In Proceedings of the Euromicro Conference on Digital System Design, DSD, Los Alamitos, CA, USA, 4–6 September 2013; pp. 260–265.
15. Obermaisser, R.; Perez, J. *Distributed Real-Time Architecture for Mixed-Criticality Systems (English Edition)*; CRC Press: Boca Raton, FL, USA, 2018.
16. Evripidou, C.; Burns, A. Scheduling for mixed-criticality hypervisor systems in the auto- motive domain. In Proceedings of the WMC 2016 4th International Workshop on Mixed Criticality Systems, Porto, Portugal, 29 November 2016.

17. Lucas, K.P.; Chappuis, M.; Paolino Dagieu, N.; Raho, D. VOSYS monitor, a low latency monitor layer for mixed-criticality systems on ARMv8-A. In *Leibniz International Proceedings in Informatics (LIPIcs)*; Bertogna, M., Ed.; Schloss Dagstuhl Leibniz Zentrum fuer Informatik: Wadern, Germany, 2017; pp. 6:1–6:18.
18. Virtual Open Systems Newsletter. Available online: http://www.virtualopensystems.com/en/company/news/newsletter-2018-09/ (accessed on 30 November 2020).
19. Alonso, A.; de la Puente, J.A.; Zamorano, J.; de Miguel, M.A.; Salazar, E.; Garrido, J. Safety concept for a mixed criticality on-board software system. *IFAC-Papers Online* **2015**, *48*, 240–245. [CrossRef]
20. Vestal, S. Preemptive Scheduling of Multi-criticality Systems with Varying Degrees of Execution Time Assurance. In Proceedings of the 28th IEEE International Real-Time Systems Symposium, Tucson, AZ, USA, 3–6 December 2007; IEEE: Tucson, AZ, USA, 2007; pp. 239–243.
21. Baruah, S.K. *Mixed Criticality Schedulability Analysis is Highly Intractable*; Technical Report; University of North Carolina: Chapel Hill, FL, USA, 2009.
22. Baruah, S.K.; Easwaran, A.; Guo, Z. Mixed-criticality scheduling to minimize makespan. In Proceedings of the 36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, Chennai, India, 15 July 2016.
23. Anderson, J.H.; Baruah, S.K.; Brandenburg, B.B. Multicore operating-system support for mixed criticality. In Proceedings of the Workshop on Mixed Criticality: Roadmap to Evolving UAV Certification, San Francisco, CA, USA, 16 April 2009.
24. Kelly, O.R.; Aydin, H.; Zhao, B. On partitioned scheduling of fixed-priority mixed- criticality task sets. In Proceedings of the IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications, Changsha, China, 16–18 November 2011; pp. 1051–1059.
25. Kritikakou, A.; Pagetti, C.; Rochange, C.; Roy, M.; Faugre, M.; Girbal, S.; Prez, D.G. Distributed run-time WCET controller for concurrent critical tasks in mixed-critical systems. In Proceedings of the 22nd International Conference on Real-Time Networks and Systems, Versailles, France, 14 October 2014.
26. Crespo, A.; Balbastre, P.; Simó, J.; Coronel, J.; Gracia Pérez, D.; Bonnot, P. Hypervisor-Based Multicore Feedback Control of Mixed-Criticality Systems. *IEEE Access* **2018**, *6*, 50627–50640. [CrossRef]
27. Ernst, R.; Di Natale, M. Mixed Criticality Systems—A History of Misconceptions? *IEEE Des. Test* **2016**, *33*, 65–74. [CrossRef]
28. Paulitsch, M.; Duarte, O.M.; Karray, H.; Mueller, K.; Muench, D.; Nowotsch, J. Mixed-criticality embedded systems–a balance ensuring partitioning and performance. In Proceedings of the Euromicro Conference on Digital System Design (DSD), Madeira, Portugal, NJ, USA, 26–28 August 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 453–461.
29. Wilhelm, R. Mixed Feelings About Mixed Criticality (Invited Paper). In Proceedings of the 18th International Workshop on Worst-Case Execution Time Analysis (WCET), Dagstuhl, Germany, 3 July 2018; pp. 1:1–1:9.
30. Esper, A.; Neilissen, G.; Neils, V.; Tovar, E. How realistic is the mixed-criticality real-time system model? In Proceedings of the 23rd International Conference on Real-Time Networks and Systems (RTNS 2015), Lille, France, 4–6 November 2015; pp. 139–148.
31. Guasque, A.; Balbastre, P.; Crespo, A.; Peiró, S. Energy efficient partition allocation in mixed-criticality systems. *PLoS ONE* **2019**, *14*. [CrossRef]
32. Masmano, M.; Ripoll, I.; Crespo, A.; Metge, J. Xtratum: A hypervisor for safety critical embedded systems. In Proceedings of the Eleventh Real-Time Linux Workshop, Dresden, Germany, 28–30 September 2009.
33. Masmano, M.; Ripoll, I.; Peiró, S.; Crespo, A. Xtratum for leon3: An open source hypervisor for high integrity systems. In Proceedings of the European Conference on Embedded Real Time Software and Systems, Toulouse, France, 29–31 January 2010; ERTS2: Toulouse, France, 2010.
34. Crespo, A.; Alonso, A.; Marcos, M.; Puente, J.A.; Balbastre, P. Mixed criticality in control systems. In Proceedings of the 19th IFAC World Congress, Cape Town, South Africa, 24–29 August 2014.
35. Yang, C.; Vyatkin, V.; Pang, C. Model-Driven Development of Control Software for Distributed Automation: A Survey and an Approach. *IEEE Trans. Syst. Man Cybern. Syst.* **2014**, *44*, 292–305. [CrossRef]
36. Frysak, J.; Kaar, C.; Stary, C. Benefits and pitfalls applying RAMI4.0. In Proceedings of the 2018 IEEE Industrial Cyber-Physical Systems (ICPS), St. Petersburg, Russia, 15–18 May 2018.
37. Hermann, M.; Otto, B.; Pentek, T. Design Principles for Industrie 4.0 Scenarios: A Literature Review. In Proceedings of the Hawaii International Conference on System Sciences, Koloa, HI, USA, 5–8 January 2016; HICSS: Koloa, HI, USA, 2016.
38. IEC1 IEC62264-1. Enterprise-control System Integration–Part1: Models and Terminology. 2013. Available online: http://www.iso.org/iso/catalogue_detail.htm (accessed on 30 November 2020).
39. IEC2 IEC61512-1. BatchControl-Part1: Model sand Terminology. 1997. Available online: http://www.en-standard.eu/iec-61512-1-1997-batch-control-part-1-models-and-terminology (accessed on 30 November 2020).
40. Poza-Lujan, J.; Posadas, J.; Simó, J.; Blanes, J.F. Smart Resources for Smart Cities: Distributed Architecture to Integrate Sensor Information. *Sensors (Basel)* **2019**, *20*, 112. [CrossRef] [PubMed]
41. Wiesner, S.; Thoben, K.D. Requirements for models, methods and tools supporting servitisation of products in manufacturing service ecosystems. *Int. J. Comput. Integr. Manuf.* **2016**, 1–12. [CrossRef]
42. Belman-Lopez, C.E.; Jiménez-García, J.A.; Hernández-González, S. Comprehensive analysis of design principles in the context of Industry 4.0. *Revista Iberoamericana de Automática e Informática industrial* **2020**, *17*, 432–447. [CrossRef]
43. Wang, X.; Ong, S.; Nee, A. A comprehensive survey of ubiquitous manufacturing research. *Int. J. Prod. Res.* **2017**, 604–628. [CrossRef]
44. Tijero, H.P.; Gutíerrez, J.J. Criticality distributed systems through the DDS standard. *RIAI J.* **2018**, *15*, 439–447.