CIRP Annals, Volume 67, Issue 1, 2018, pp. 161-164
DOI:10.1016/j.cirp.2018.04.021

Accepted for publication in CIRP Annals Manufacturing Technology

Resilient architecture for cyber-physical production systems

Tetsuo Tomiyama (1), Florian Moyen

Cranfield University, Cranfield, MK43 0AL, UK

Abstract: Production systems are a typical example of Cyber-Physical Systems (CPS) in which a variety of machines, actuators, sensors and control systems are interwoven to produce products as efficiently as possible. Even though sophisticated condition monitoring systems are deployed, stoppage, breaks, and other types of failures still happen. To avoid catastrophic operational disruptions, desirably the production system itself resiliently and autonomously responses to failures. This paper reports a design method for a resilient architecture of a cyber-physical production system that can deal with disturbances and failures in a discrete-event process. A physical demonstrator was built to demonstrate its reconfiguration capabilities.

Keywords: System architecture, reconfiguration, maintenance

1. Introduction

Minimising downtime of production systems is crucial for any manufacturing enterprise. A variety of approaches and technologies have been developed including advanced monitoring and sensing techniques, better maintenance practices such as TPM (Total Productive Maintenance) [1], and more robust production equipment and machines [2]. The concepts of Cyber-Physical Systems (CPS) [3] [4] and the Internet of Things (IoT) [5] are already part of the production environment and will be more present in the future towards Industry 4.0 [6]. Here, the goal is a system capable of adapting to an ever-changing internal and external environment and able to react to failures without human intervention.

However, despite these efforts, production systems do stop for a variety of reasons. For example, short stoppages of the production line can happen because of jams in part feeders, wrong location and alignment of parts, short supply of parts, and shortage of consumables (such as lubrication oil). Compared with more serious failures (e.g., a failure of a motor bearing), such short stoppages are easier to fix but more frequent and the consequences are relatively expensive [7]. Therefore, the future production system must be 'resilient' against not only heavier serious failures but also these 'lighter' failures.

Currently, the architecture of Cyber-Physical Production Systems (CPPS) is based on fixed connections due to the traditional philosophy inherited from mechatronics (depicted in Figure 1 [8]) in which the controller-sensor-actuator triad will be fixed once designed. However, due to the increased concern about systems' reliability and other through-life performances, the numbers of sensors, actuators and connections in CPS will continue to grow.

Consequently, various reconfigurable CPS architecture types have been proposed, e.g., for embedding redundancy in a sensor network to allow reconfiguration of sensors. This reconfiguration often means replacing part of, e.g., sensor network with an alternative sensor (physical reconfiguration) or generating equivalent information of the failed sensor using information obtained from other (still functioning) sensors based on sensor fusion technologies.

In contrast, this paper presents a new principle and its design methodology for 'resilient architecture' that will permit static and dynamic reconfiguration of the system at both the machine and the network levels [8]. It allows continuous operations of the system even with some subsystems failing or being repaired. In addition, the resilient architecture must accommodate not only serious failures but also lighter stoppages. Figure 2 depicts such dynamic reconfiguration based on 'mesh topology'.

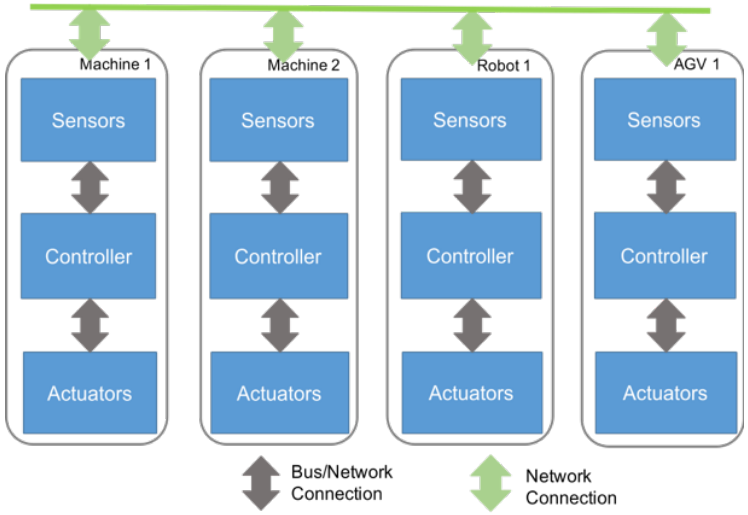

Figure 1. CPS architecture based on fixed triad of traditional mechatronics systems [8]

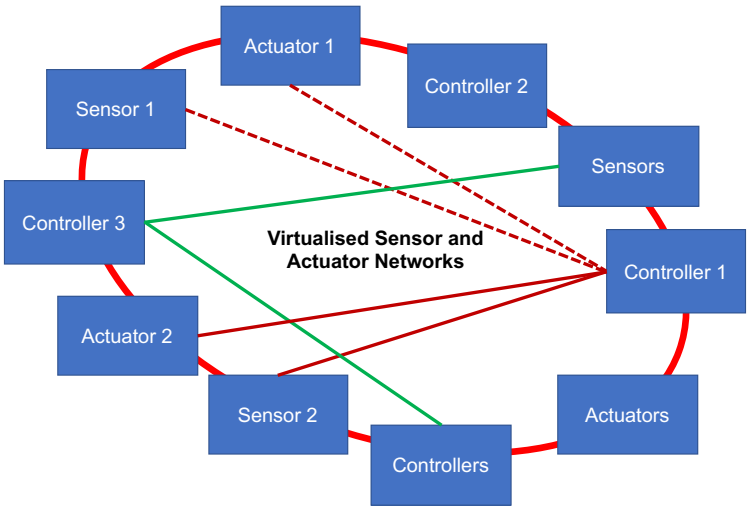

Figure 2. Resilient CPS architecture: reconfigurable mesh topology [8]

The aim of this paper is to establish a resilient CPS architecture that can flexibly reconnect and reconfigure to accommodate failures and other disturbances in a discrete-event process to maintain its functions at an acceptable level. Therefore, the research objectives are:

- Identification of methods for fault-tolerance, redundancy, and reconfiguration to achieve resilience.
- Development of a resilient architecture using the reconfiguration methods.
- Implementation of a physical demonstrator to validate the concept of the resilience architecture.

The rest of the paper is organised as follows. In the next section, relevant concepts to resilient architecture are discussed, including fault-tolerance, redundancy, self-maintenance and reconfiguration. Section 3 proposes resilient CPS architecture. As a proof of concept, a physical demonstrator simulating a production line was developed. Section 4 concludes the paper.

2. Resilient cyber-physical systems

CPS are 'engineered systems that are built from, and depend upon, the seamless integration of computational algorithms and physical components' [4]. CPPS is the application of CPS to the production domain towards Industry 4.0 by permitting the communication between elements of the so-called smart factory [9][10][11].

Traditionally, one of the main expectations to such smart factory is robustness or fault-tolerance, for example, that tolerates disturbance to sensors or actuators, by reconfiguring its state, behaviour, or function. In this paper, we define resilient CPS as a system that can tolerate faults, repair faults, or maintain itself through (preferably) autonomous reconfiguration enabled by redundancy at state, behaviour, or function levels.

Zhang and Van Luttervelt defined resilience as 'the system's capability of leading to success from failure on the system's own – in particular its own infrastructure, substance' and identified four guidelines [12]. However, this statement was made about 'resilient manufacturing networks' but not as technical design principles for CPPS. The following will examine technical aspects of 'resilience'.

2.1. Fault tolerance

The goal of a fault-tolerant or robust system is to accept the fault of an individual component (or components) while maintaining the overall functions at an acceptable level [13]. First, a fault-tolerant system should detect and diagnose the fault. Then, it reacts to a fault [14] by using the following four strategies:

- Reconfiguration
- Controller reconfiguration
- Stop of operation
- Physical repair of components.

Obviously, the third and fourth strategies are not preferred, because of cost and time implications. Instead, this research will focus on the first and second 'reconfiguration' strategies, which are especially interesting to avoid 'short stoppages'.

2.2. Redundancy

In order to reconfigure a system, redundancy is a critical concept [14]. 'Static redundancy' uses parallel modules always active and using the same input, while 'dynamic redundancy' uses a fault management system for cold standby or hot standby. For example, sensor failure is one of the most common failures in production automation. Known strategies for component level redundancy include static redundancy, dynamic redundancy with cold standby or hot standby (such as analytical sensor redundancy), and soft sensors. The most natural actuator redundancy is static redundancy using parallel configuration. The elements are operating in parallel and if one fails, the others used to take over the force of the faulty one; e.g., two motors supporting half load in normal operation are becoming one supporting the whole load.

2.3. Self-maintenance and reconfiguration based on redundancy

Umeda et al. proposed the concept of self-maintenance based on function-redundancy [15]. This function-redundant type self-maintenance is based on the FBS (Function-Behaviour-State) modelling [16]. In FBS, a state is a snapshot of a system described as a network of entities and parameters with value. A behaviour is a temporal state transition, while a function is a subjective observation of a behaviour represented in the form of 'to do something'. Figure 3 represents the relationship between system's entities *E*, state *S*, behaviour *B*, and function *F*. Figure 4 illustrates a case in which a different state-behaviour configuration (e.g., through software modification) leads to exhibition of a latent function.

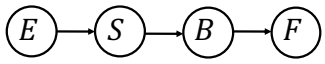

Figure 3. FBS modelling of a system

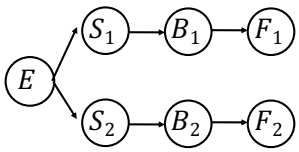

Figure 4. Exhibiting a latent function (originally F_1 but switching to F_2)

Static redundancy (typically parallel system) or dynamic redundancy with cold standby can be introduced as in Figure 5 (a). However, a component level change can lead to a state level change, thereby to a behavioural change. In this case, it is functional redundant type maintenance, because it is using a latent function of a component similar to the

lost function (Figure 5 (b)). This can be contrasted with Figure 4 which exhibits a different latent function with a different behaviour of the same entity.

(a) Static redundant (parallel system) (b) Function-redundant type

Figure 5. Redundant systems

2.4. Dynamic reconfiguration with diagnosis

The purpose of resilient CPS architecture is fault-tolerance or robustness of the system, which allows satisfactory operations even with faults. This needs to be autonomous, which means that fault detection and diagnosis should take place automatically. Therefore, resilient CPPS should allow dynamic reconfiguration combined with diagnosis.

Figure 6 illustrates 'mode transition' for diagnosis, which is a well-known technique in control engineering [17][18]. It begins with the normal mode. Every *x* seconds, a diagnosis process is invoked to test the condition of the system. If the test result is unsatisfactory, the system will move to the KO (Knocked Out) mode in which the system needs to be either reconfigured or stopped. Reconfiguration takes place, depending on the capability of the system, by transiting to a pre-defined reconfigured mode (typically one of those in the previous section) or to one reasoned out through automatic reasoning about the system.

Either way, the system continues to operate in the KO mode for a while and tries to come back to the normal mode. However, this diagnosis process needs to be operative, independently of the system's mode (normal or KO). This means that sensors to diagnose systems conditions need to be separated as much as possible from the systems elements that will be affected by faults or independent from other elements as much as possible. For this purpose, the mesh topology (Figure 2) has an advantage.

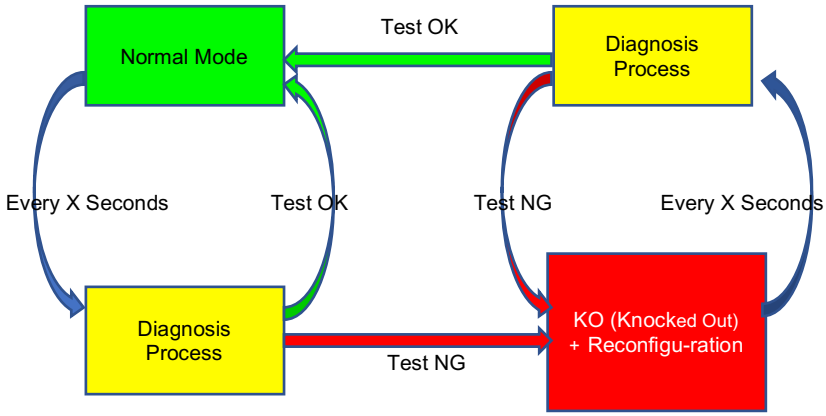

Figure 6. Fault diagnosis models

3. Resilient CPS architecture

A physical demonstrator that models a production system was developed using a Fischertechnik platform [19] and the Arduino open source single-board microprocessor platform [20]. Figure 7 shows the hardware that contains one vacuum gripper robot, (emulated) material processing units, and conveyors for material handling. Workpieces are first stored at the storage station and then transported by the robot to the machining station equipped with drilling and milling machines. Then workpieces are transported by the transport station (conveyor) and then further to another processing station just before going back to the storage station. As this is just for demonstration, workpieces will be circulating within the system.

Each subsystem is controlled by an Arduino CPU, and original sensors and actuators of were re-connected to Arduino based on the network principle (called 'mesh topology') depicted in Figure 2. This allows software-based reconfiguration of some of the sensors and actuators.

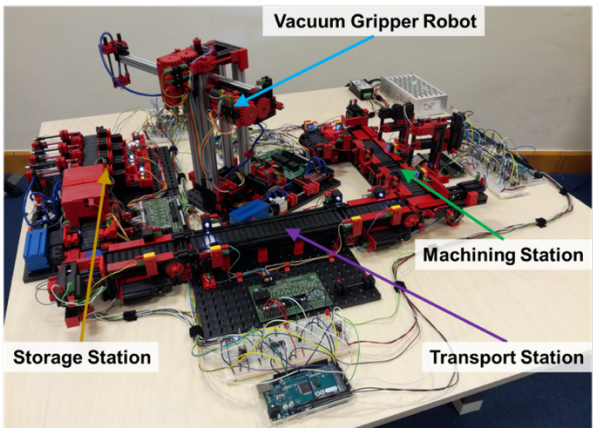

Figure 7. Physical demonstrator

3.1. Fault diagnosis

Each actuator is equipped with a sensor for diagnostic purposes, apart from the sensor for control purpose. Fault diagnosis itself can be executed by a dedicated CPU or by a controller that controls other sensors and actuators (which is the advantage of the mesh topology in Figure 2). The faults which will be handled by reconfiguration need to have a means of detection (e.g., sensors or abnormality in data) independent of failures of the subsystem. Ideally, the faults should be analysed by, e.g., FTA (Fault Tree Analsys) during the design phase and listed with their possible root causes to be avoided or mitigated by reconfiguration.

The machining station in Figure 7 has a drilling machine. The failure of the spindle motor in this implementation can be detected by a vibration sensor placed near the motor (but not necessarily controlled by any other CPU other than the CPU which controls the spindle motor). Figure 8 illustrates the fault diagnosis algorithm for the drilling machine in the Stateflow chart of Simulink. This code will be directly translated to executable code for Arduino.

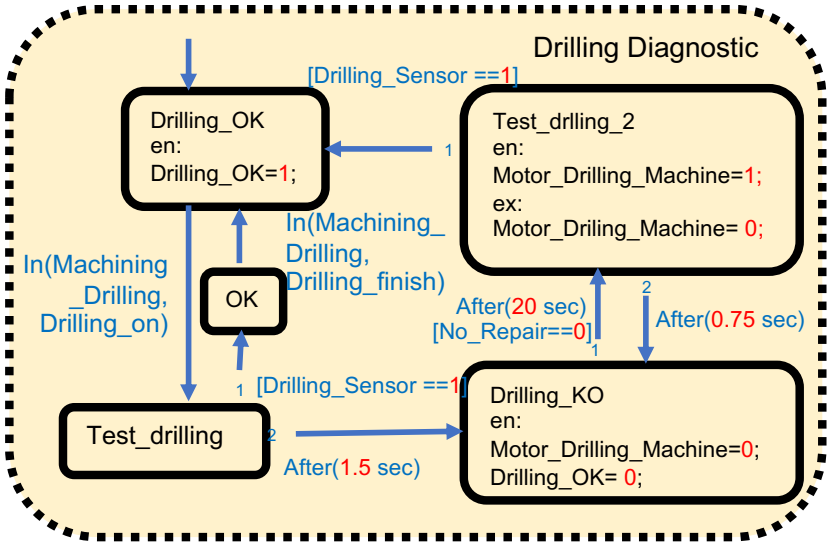

Figure 8. Fault diagnosis applied to a drilling process

3.2. Resilience strategies

So far, we have identified a couple of strategies for resilient CPS architecture as below that should use reconfiguration based on redundancies embedded during the design phase. These redundancies are mainly cold-standby redundancies which are most suitable for production systems.

3.2.1. Mesh topology

Reconfiguration can be better implemented in the mesh topology in Figure 2, which is preferred instead of a star or tree topology, because concentration of load can be avoided and because re-routing of communication paths becomes easier. In addition, this topology is well suited for wireless communication which might develop in the future CPS.

3.2.2. Fault diagnosis capabilities

These capabilities are critical for reconfigurability and embedded in the architecture. The outcome of fault diagnosis should include both plausible causes of the fault and reconfiguration candidates. The mesh topology allows detection and diagnosis of faults by not directly connected CPUs.

3.2.3. Independence of subsystems

The goal of reconfiguration is to maintain the function in case of failure of a subsystem. To do so, each subsystem must be independent or easily separated from other part of the system as much as possible.

3.2.4. Parallel system

It is well-known that parallel systems perform better due to their higher reliability. Additionally, they allow continuous operations during maintenance. This feature is especially important to avoid short stoppage. The mesh topology allows massively parallel systems architecture. However, the parallel system concept is expensive in general due to obvious duplications. In such a case, functional redundancy might be a better solution.

3.2.5. Virtual controller

A virtual controller implements one state diagram regarding one function. The mesh topology principle should allow reconfiguration of controllers (i.e., quickly changing a failed controller with another healthy one).

3.3. Reconfiguration demonstration

Table 1 summarises the reconfiguration examples embedded in the demonstrator. These examples realise combinations of the strategies above. Figure 9 depicts the reconfiguration in case of the failure of the main conveyor motor. The demonstrator effectively showed the designed reconfiguration examples. Once a failure was recognised within the inspection cycle depicted in Figure 6, the reconfiguration process listed in Table 1 took place and the alternative reconfigured actions were engaged. These demonstrations are particularly interesting in dealing with short stoppages, because various forms of reconfiguration based on redundancy allowed continuous operations even during one subsystem was not operational or being repaired.

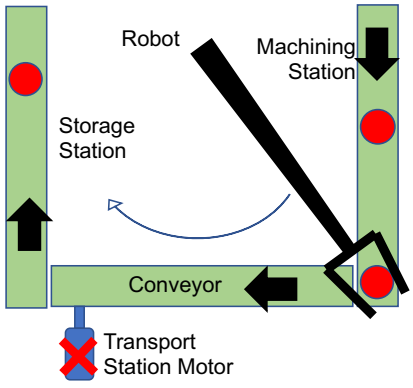

Figure 9. Reconfiguration for the failure of the main conveyor motor

Table 1 Reconfiguration examples

Failure of a machining motor
Sensing: A vibration sensor attached to the motor. **Diagnosis**: The controller responsible for the processing part of the machining station performs the diagnosis of the machining motor. **Reconfiguration**: Diagnosis information is continually transmitted to the controller responsible for the transportation. This will stop sending parts to the machining station and initiate the use of an alternative machining method if available (function-redundancy and stand-by redundancy).
Failure of the main conveyor motor (Figure 9)
Sensing: A vibration sensor attached to the motor or a rotary encoder connected to the conveyor. **Diagnosis and Reconfiguration**: After detecting the main motor failure, its controller requests the vacuum gripper to transport parts so that they can skip the conveyor (function redundancy). The vacuum gripper software is designed to manage by itself the different sequences of transportation, from the storage station to the machining one and from the machining to the storage station (stand-by redundancy through embedded software as in Figure 5).
Failure of the transport station controller
Sensing: A failure of the transport station controller (e.g., timing error or freezing) is electronically detected by the conveyor controller of the machining station. **Diagnosis and Reconfiguration**: Instead of sending the part to the main conveyor, the process requests the vacuum gripper robot to handle the task. The rest is the same as the case above.
Failure of a slider in the storage station
Sensing: When one slider to shoot a part to appropriate storage fails to move, this is detected by the sensor at the slider. **Diagnosis and Reconfiguration:** The controller of the storage station acknowledges this and unless the storage is full, the part is moved to the next available slider (hot stand-by redundancy).

4 Conclusions

This paper discussed 'resilient architecture', which can be designed based on the following three principles. The physical demonstrator demonstrated that the effectiveness of the resilient architecture and the reconfiguration strategies available on this demonstrator.

- Various types of redundancy (particularly, static redundancy, dynamic redundancy with stand-by, and functional redundancy) are useful for reconfiguration.
- The network between controllers, sensors, and actuators should form a mesh topology.

- Subsystems should be separated from other subsystems as much as possible. As the case of the transport station controller shows, thanks to the mesh topology, it is possible to separate sensors from the subsystem they monitor.

The method described in this paper has limitations resulting from the hardware limitations. First, the connection topology was not perfect 'mesh topology', which restricted the kinds of demonstrations available on the demonstrator. Second, it also lacked the capabilities of dynamically reasoning out possible reconfiguration candidates and executing reconfiguration. Despite these limitations, the demonstrator highlighted the possibilities of the resilient architecture. Future work includes the use of, e.g., WiFi IoT chips capable of software-based reconnection. This helps the realisation of 'mesh topology' depicted in Figure 2.

References

[1] Nakajima, S., 1986, TPM – Challenge to the Improvement of Productivity by Small Group Activities, Maintenance Management International, *6*/2:73-83.

[2] Takata, S., Kimura, F., van Houten, F.J.A.M., Westkämper, E., Shpitalni, M., Ceglarek, D., Lee, J., 2004, Maintenance: Changing role in life cycle management, CIRP Annals - Manufacturing Technology, *53*/2:643–655.

[3] Lee, E.A., 2008, Cyber physical systems: Design challenges, Proc. of 11th IEEE Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing, ISORC 2008, pp. 363–369.

[4] National Science Foundation, 2017, Cyber-Physical Systems (CPS), available at: https://www.nsf.gov/pubs/2017/nsf17529/nsf17529.htm (accessed: 2 June 2017).

[5] Gershenfeld, N., Krikorian, R., Cohen, D., 2004, The Internet of Things, Scientific American, 291/4:76–81.

[6] Kagermann, H., Wahlster, W., Helbig, J. (eds.), 2013, Recommendations for implementing the strategic initiative Industrie 4.0: Final report of the Industrie 4.0 Working Group, available at: http://www.acatech.de/fileadmin/user_upload/Baumstruktur_nach_Website/Acatech/root/de/Material_fuer_Sonderseiten/Industrie_4.0/Final_report__Industrie_4.0_accessible.pdf (accessed: 20 January 2018)

[7] Alsyouf, I., 2007, The role of maintenance in improving companies' productivity and profitability, International Journal of Production Economics, 105/1:70–78.

[8] Hehenberger, P., Vogel-Heuser, B., Bradley, D., Eynard, B., Tomiyama, T., Achiche, S., 2016, Design, modelling, simulation and integration of cyber physical systems: Methods and applications, Computers in Industry, 82:273–289.

[9] Teti, R., Jemielniak, K., O'Donnell, G., Dornfeld, D., 2010, Advanced monitoring of machining operations, CIRP Annals - Manufacturing Technology, 59/2:717–739.

[10] Monostori, L., Kádár, B., Bauernhansl, T., Kondoh, S., Kumara, S., Reinhart, G., Sauer, O., Schuh, G., Sihn, W., Ueda, K., 2016, Cyber-physical systems in manufacturing, CIRP Annals - Manufacturing Technology, 65/2:621–641.

[11] Wang, L., Törngren, M., Onori, M., 2015, Current status and advancement of cyber-physical systems in manufacturing, Journal of Manufacturing Systems, 37:517–527.

[12] Zhang, W.J., van Luttervelt, C.A., 2011, Toward a resilient manufacturing system, CIRP Annals - Manufacturing Technology, 60/1:469–472.

[13] Umeda, Y., Tomiyama, T., Yoshikawa, H., Shimomura, Y., 1994, Using Functional Maintenance to Improve Fault Tolerance, IEEE Expert-Intelligent Systems and Their Applications, 9/3:25–31.

[14] Muenchhof, M., Beck, M., Isermann, R., 2009, Fault Tolerant Actuators and Drives – Structures, Fault Detection Principles and Applications, IFAC Proceedings Volumes. IFAC, 42/8:1294–1305.

[15] Umeda, Y., Tomiyama, T., Yoshikawa, H., 1992, Design methodology for a self-maintenance machine based on functional redundancy. ASME, Design Engineering Division DE Vol. 42, pp. 317-324.

[16] Umeda, Y., Ishii, M., Yoshioka, M., Shimomura, Y., Tomiyama, T., 1996, Supporting conceptual design based on the function-behavior-state modeller, Artificial Intelligence for Engineering, Design, Analysis and Manufacturing, 10/4:275–288.

[17] Sampath, M., Sengupta, R., Lafortune, S., Sinnamohideen, K., Teneketzis, D. C., 1996, Failure diagnosis using discrete-event models, IEEE Transactions on Control Systems Technology, 4/2:105–124.

[18] Zhang, Y., Jiang, J., 2008, Bibliographical review on reconfigurable fault-tolerant control systems, Annual Reviews in Control, 32/2:229–252.

[19] Fischertechnik GmbH, 2017, Industry–Training models, available at: https://www.fischertechnik.de/en/products/teaching/training-models (accessed: 20 January 2018).

[20] Arduino, 2017, Arduino Mega 2560 Rev3, available at: https://store.arduino.cc/arduino-mega-2560-rev3, (accessed: 20 January 2018).

Resilient architecture for cyber-physical production systems

Tomiyama, Tetsuo

2018-05-24

Tomiyama T, Moyen F. (2019) Resilient architecture for cyber-physical production systems.
CIRP Annals, Volume 67, Issue 1, 2018, pp. 161-164
http://dx.doi.org/10.1016/j.cirp.2018.04.021