








# Engineering Micro-intelligence at the Edge of CPCS: Design Guidelines

Roberta Calegari<sup>1</sup>  , Giovanni Ciatto<sup>2</sup> , Enrico Denti<sup>1</sup> ,  
and Andrea Omicini<sup>2</sup> 

<sup>1</sup> Dipartimento di Informatica – Scienza e Ingegneria (DISI),  
Alma Mater Studiorum–Università di Bologna, 40136 Bologna, Italy  
{roberta.calegari, enrico.denti}@unibo.it

<sup>2</sup> Dipartimento di Informatica – Scienza e Ingegneria (DISI),  
Alma Mater Studiorum–Università di Bologna, 47521 Cesena, Italy  
{giovanni.ciatto, andrea.omicini}@unibo.it

**Abstract.** The Intelligent Edge computing paradigm is playing a major role in the design and development of Cyber-Physical and Cloud Systems (CPCS), extending the Cloud and overcoming its limitations so as to better address the issues related with the physical dimension of data—and therefore of the data-aware intelligence (such as context-awareness and real-time responses). Despite the proliferation of research works in this area, a well-founded software engineering approach specifically addressing the distribution of intelligence sources between the Edge and the Cloud is still missing. In this paper we propose some general criteria along with a coherent set of *guidelines* to follow in the *design of distributed intelligence* within CPCS, suitably exploiting Edge and Cloud paradigms to effectively enable data intelligence and accounting for both symbolic and sub-symbolic approaches to reasoning. Then, we exploit the notion of *micro-intelligence* as situated intelligence for Edge computing, promoting the idea of *intelligent environment* embodying rational processes meant to complement the cognitive process of individuals in order to reduce their cognitive workload and augment their cognitive capabilities. In order to demonstrate the general applicability of our guidelines, we propose *Situated Logic Programming* (SLP) as the conceptual framework for delivering micro-intelligence in CPCS, and *Logic Programming as a Service* (LPaaS) as its reference architecture and technological embodiment.

**Keywords:** Design guidelines · CPCS · Micro-intelligence · LPaaS · Situated Logic Programming · Edge intelligence

## 1 Introduction

According to the so-called “CPS Revolution”, Cyber-Physical Systems (CPS) are radically changing human needs and expectations, ultimately affecting every aspect of human life through application domains such as smart grids, buildings,

factories, and so on [18, 21, 22]. The steady growth of Cloud services pushes the horizon of Edge computing forward as a new paradigm and promising architecture to address the challenges of Cyber-Physical and Cloud Systems (CPCS). In particular, Edge computing makes it possible to manage CPS devices by directly taking into account their situated nature, as well as to match real-time requirements by avoiding communication bottlenecks between CPS and the Cloud. This is why most CPCS applications are designed around Edge computing principles, decentralising computation towards the edge of the network to lessen the pressure on the Cloud, hence improving user experience through task offloading and by acting in-between sensors and actuators and the Cloud.

Fully addressing CPCS issues and challenges mandates for intelligence, so that intelligent capabilities – such as context-awareness and real-time planning – become crucial for Edge devices in order to fill the gap between the CPS and the Cloud ends. In particular, research efforts on Edge computing demonstrate the benefits of distributing intelligence between the Cloud and the Edge [7, 17, 20]. However, most of the existing research work is basically strict in scope and compartmentalised, so it does not provide for a general software engineering standpoint [1, 13, 20]. As a consequence, criteria, guidelines, and methodologies on how to design and distribute intelligence in CPCS according to the Edge computing paradigm are mostly missing [8].

This is why in this paper we propose some general guidelines for the design of distributed intelligence in CPCS. The proposed design approach accounts for both symbolic and sub-symbolic approaches to reasoning, fruitfully combined to produce intelligent behaviour—as the synergies between the two sorts of techniques make it possible to deal with the diverse requirements of CPCS. Then, we define *micro-intelligence* as the *situated intelligence* of Edge computing, promoting the idea of intelligent environment embodying rational processes meant to complement the cognitive process of individuals in order to reduce their cognitive workload and augment their cognitive capabilities. In particular, we propose Situated Logic Programming (SLP) as the conceptual framework for delivering micro-intelligence in CPCS, and *Logic Programming as a Service* (LPaaS) [5] as its reference architecture and technological embodiment.

## 2 Design Guidelines for Distributing Intelligence

Different CPCS come with different requirements in terms of *intelligence*. However, intelligence is often the result of several design choices which effectively fit a given scenario. In this section we motivate and describe a number of guidelines, summarised in Table 1, aimed at steering designers in the complex task of endowing CPCS with intelligence. Columns represent the *design choices* available to software engineers dealing with the issue of spreading intelligence in CPCS, whereas rows represent the *design criteria* they should adopt and assess so as to make informed, well-founded decisions, based on the specific application scenario. Checkboxes represent *suggestions*, reasonable choices—which are by no means intended to be strict.

**Table 1.** Design choices (*columns*) and design criteria (*rows*) to distribute intelligence in CPCS in a principled way. Checkmarks within parentheses represent weak preferences.

<i>intelligence</i>	which		where	who	
	symbolic	sub-symbolic	Cloud	Edge	individual environment e-institution
no requirement	✓		✓		✓
real-time req.		(✓)		✓	
no requirement	✓		✓		✓
fault-tolerance req.				✓	
no requirement		✓	✓		✓
safety critical req.	✓			✓	
big data		✓	✓		✓
small data	✓			✓	
macro scale		✓	✓		✓
micro scale	✓			✓	
commonsense KB			✓		(✓)
contextual KB	✓			✓	
disembodied			(✓)		✓
embodied	✓			✓	
no requirement		✓	✓		✓
situatedness req.	(✓)			✓	

Designers may obviously incur in some cases where the best decision contrast with our guidelines—as it typically happens for design guidelines in SE: as “guidelines” they are meant to *guide* decision making towards the most probable path to follow, and demanding for thorough reasoning when leaving such a path. In fact, it could even be argued that the most important contribution of Table 1 are the *criteria* after which column and row headings are labelled, rather than the mere checkboxes within: to the best of our knowledge, such a detailed and pragmatically-motivated categorisation is novel in the literature on CPCS software engineering.

The design guidelines presented in this paper concern first of all the distribution of the sources of intelligence along three dimensions, conveniently reported in Table 1 (columns):

- *which* sort of intelligent source should be delivered—i.e., which approaches (symbolic vs. sub-symbolic) are to be used for equipping the (portion of the) CPCS with intelligence
- *where* such approaches and techniques should be deployed (Cloud vs. Edge)
- *who* – that is, which part/component of the CPCS – should be exhibiting the designed intelligent behaviour.

With respect to the *who* feature, intelligence can be displayed by the active *individuals* inhabiting the CPCS and performing their own decision making process based on ascribed goals and rules, and their perceptions of the environment.

There, the *environment* is the basic infrastructure enabling sensing, monitoring, and feedback control, while adding intelligence in the form of, for instance, traffic flow forecast, data analytics, self-healing capabilities (e.g., resolution of traffic congestion). Finally, there is another, possibly less “visible” entity which may exhibit intelligence: the *e-institution* [11], that is, the conceptual place where all the norms, goals, constraints which set the boundaries for the overall system behaviour are defined and, possibly, enforced (e.g. through sanctions and penalties).

A few amongst the design criteria reported in the rows of Table 1 are simply extracted from the peculiar features of CPS as devised out by the existing literature<sup>1</sup>—namely, their *real-time* nature, the need for *fault tolerance*, and the fact that CPS are often *safety-critical* systems. A few others stem from the addition of the Cloud to the picture, which enables CPS to reach unprecedented *scale* and provide them with the ability to gather and process a large amount of (*big*) *data*, continuously sent to the Cloud by devices. The remaining criteria are rather novel, and stem from two related aspects: the rise of Edge computing, on the one hand, and the need for more powerful software engineering abstractions, on the other.

Let us now discuss the guidelines in detail—that is, all the checkboxes appearing in Table 1 cells.

*Real-Time Requirement.* Most CPS have strict *real-time* requirements constraining decision making, hence time from analysis of input data down to taking action is typically severely limited. In that case, sub-symbolic approaches to deliver intelligence, deployed at the Edge on individuals and environmental resources, are the most likely to succeed, given that pressing timing constraints make answers from Cloud easily too slow. Moreover, e-institutions are also usually concerned with both enforcing norms governing the space of admissible interactions between system components, and monitoring the long-term evolution of the system so as to guarantee desired properties and detect misbehaviour. Indeed, the Edge computing paradigm is precisely born with the foremost goal of reducing latency of processing and communications. Finally, *application* (*not* training) of a sub-symbolic approach – which may amount at, for instance, traversing a decision tree – is likely to be more performing compared to a symbolic one, which usually implies some form of exhaustive reasoning or joint planning.

*Fault-Tolerance Requirement.* As depicted in the *which* column of Table 1, *fault-tolerance* has little to do with the sort of approach to intelligence. In fact, fault-tolerance – in the sense of ensuring that system faults or miscommunications have the least possible impact on the outcomes of intelligent behaviour – is greatly enhanced by leveraging *decentralised* approaches enabled by the Edge computing paradigm, where individual devices and environmental resources can replicate storage and functionality to achieve greater availability and reliability of both information processing and services provided. Although the Cloud

<sup>1</sup> See <http://cyberphysicalsystems.org> for a quick and nice overview.

is intrinsically a fault-tolerant computational environment – in the sense that services and storage are usually replicated, and a fail-over mechanism is in place – it is also a communication bottleneck as well as a single point of failure.

*Safety-Critical Requirement.* At least a portion of any given CPS is usually a *safety-critical* system. This is the reason why whenever that requirement holds, symbolic approaches deployed at the Edge under responsibility of individual actors of the system are to be preferred. By no means, there, failure in, e.g., sensor devices perceptions or communications between a group of collaborating actors (governed by an e-institution) should lead to catastrophic consequences. For the same reason, the *opaque* nature of sub-symbolic approaches cannot guarantee absence of errors under any given circumstance, whereas symbolic approaches can encode strict safety rules to be applied no matter what.

*Big Data Available?* The next design criterion concerns the nature of the data available for processing with the goal of delivering intelligence: has it *variety*, *velocity*, and *volume*, or, has it not? In the former case, it is desirable to exploit sub-symbolic techniques so as to find relevant patterns buried in data by leveraging the Cloud horsepower. Such data may be well used to perform long-term planning and predictive adaptation at the level of e-institution. Also, it is usually unfeasible to process large amounts of data streams coming into the system at a fast pace at the Edge, and symbolic approaches often suffer from degraded performance as the *knowledge base* increases in size.

*Scale of Interest.* It is worth emphasising the symmetry with the *scale* of interest for delivering intelligence: as big data best suits the Cloud and e-institutions to power sub-symbolic techniques, so does focussing on the macro scale of CPCS as the target of intelligent behaviour. Indeed, as already mentioned, long-term planning is likely to need lots of data, gathered in a considerable time span from many heterogeneous sources, demanding statistical approaches to make sense of it and find valuable insights. On the contrary, symbolic approaches pervasively deployed at the Edge of the CPCS are much more meaningful for small data analysis, so as to leave individuals and intelligent environmental resources perform local inference with the twofold goal of alleviating the computational burden on the Cloud and deliver intelligence on a shorter time horizon, locally.

*Nature of Knowledge.* Here, we intend to focus on the kind of information that artificial intelligence techniques exploit for delivering intelligence: is it general knowledge of a problem domain, of the physical world, of the basics semantics behind everyday objects and their relationships, or, is it specific information which has value only in well-defined situations and w.r.t. precise goals? In other words: is it *commonsense* or *contextual knowledge*?

The distinction is of paramount importance, in practice, and is reflected by the checkmarks in Table 1. First, contextual knowledge is usually acquired *dynamically* during the system operation by the sensor devices displaced at the Edge of the CPCS, and is meant to keep the system informed about the

ever changing situation that individuals and environment resources should deal with. Commonsense knowledge, instead, has a more static and less “situational” nature, since it is meant to capture the innate knowledge that we, as humans, take for granted—an extremely complex task for machines, though. Depending on how critical such information is, it may be stored and exploited either in the Cloud or at the Edge. Going forward, the *who* column in Table 1 deserves special attention: in fact, contextual knowledge has been attributed to the environment and the institutional dimension, whereas commonsense to individuals and the environment. While the environment is an obvious choice for the former – as gathering and processing contextual knowledge is usually a typical sensors’ and actuators’ responsibility –, involving the e-institution abstraction may seem odd, at first. Conversely, the environment is not an obvious choice for commonsense knowledge, as individuals instead possibly are.

*Embodied Computation?* Another design criterion is the notion of *embodiment*—namely, the feature of a computation of being strictly bound to the *physical nature* of its hosting device. Embodied computations are better suited for Edge computing and should be responsibility of individuals and environmental resources. Those are in fact the devices and physical components usually equipped with sensing and actuating capabilities, hence whose computations are inevitably bound to the available equipment. Disembodied computations are instead the typical use case for the abstraction of e-institutions, which are meant to encapsulate all the norms that rule interaction and behaviour of individuals. As such, disembodied computations are typically hosted on the Cloud, but nothing goes strong against executing them also at the Edge—after all, they are just ordinary computations. Finally, since the embodiment of computations has little to do with the technical approach to deliver intelligence, this criterion does not impact the choice of whether to exploit either symbolic or sub-symbolic models.

*Situatedness Requirement.* Even if no preference is made explicit as regards the “Which” of intelligence, there exist arguments in favour of a stronger prevalence of symbolic approaches when situatedness *is required*. Intended as the property of computations to heavily depend on environmental conditions, situatedness is a key requirement for delivering intelligence in this context. Since, by definition, it may correspond to rare (if not unique) occurrences of events out of the normal system operations, few examples may exist for training a sub-symbolic model to detect and classify them: this is why a symbolic approach may be well suited to explicitly handling such exceptional conditions.

Finally, w.r.t. the last two criteria, we mean to emphasise that embodied vs. disembodied, situated vs. non-situated computations should be viewed and dealt with as complementary facets, to be exploited in synergy so as to better tackle real-world problems with the proper “degree of situatedness” [15].

### 3 Micro-intelligence in CPCS with LPaaS

In order to demonstrate the feasibility of our guidelines – that is, the actual applicability in the architectural design phase –, in this Section we present the micro-intelligence approach and LPaaS as its reference architecture and technology—which can be used in the detailed design and development stages to implement the guidelines. More precisely, the *micro-intelligence* approach [2] is exploited as a way to reify the guidelines in the LPaaS technology, thus completing the tile of intelligence in CPCS by acting synergically with sub-symbolic techniques.

In a nutshell, micro-intelligence is about scattering small chunks of machine intelligence all over a distributed, situated system, capable of enabling the individual intelligence of any kind of devices [16]: the main idea is that (micro-) intelligence can be encapsulated in devices of any sort, making them both smart and capable to work together in groups, aggregates, societies. Accordingly, the reference scenario assumes that (i) knowledge is locally scattered in the distributed environment, hence its situated nature; (ii) inference capabilities are admissible and available over this knowledge, with the goal of extending the local knowledge through induction, deduction, abduction, and the like.

Philosophically, micro-intelligence can be interpreted as the *externalised rationality* of cognitive agents (individual), complementing their own in the sense of Clark and Chambers' *active externalism* [9], and under the perspective of Hutchins' *distributed cognition* [12]. In that context, *external* means that it does not strictly belong to individuals—in fact, it is a process independently possibly executed by another entity (for instance the environments, the infrastructure) to whom the individual is (possibly, temporarily) coupled. It is also rational because it is supposed to convey a sound inference process. It complements individuals' own cognitive process because, by embodying situated knowledge about the local environment along with the inference processes admissible therein, augments the cognitive capabilities of agents that can be unaware of the knowledge embodied in the environment.

Moving from the widely-accepted consideration that pervasive and distributed systems have no global state (intended as a single, coherent knowledge base), but are rather composed of local, fragmented knowledge chunks, Situated Logic Programming (SLP) [3] introduces a possible framework for the embodiment of the micro-intelligence vision: there, multiple logical theories, scattered in the environment, can co-exist to represent the local, possibly partial knowledge base (KB). In this perspective, SLP can be seen as an extension of LP where *each logic theory is situated* in space, time, and (possibly) w.r.t. a specific environmental resource. LPaaS [5] can be seen as the natural instantiation of the SLP idea [5], designed according to the architectural style and principles of the Service-Oriented Architecture (SOA) paradigm [10] and, in particular, of the microservice vision [6]. As such, LPaaS constitutes a suitable reference architecture for delivering micro-intelligence to the CPCS edge [4], enabling situated reasoning via the explicit definition of the spatio-temporal structure of the environment where situated entities act and interact: by doing so, it suitably re-interprets the notion of distribution of LP accordingly to the SLP framework.

It is worth noting that its SOA nature further emphasises the role of *situatedness*, already brought along by distribution in itself—developing the idea of LP as a situated service while promoting key features such as encapsulation, statelessness, and locality. Concretely, LPaaS features distributed service instances (servers) scattered in a CPCS environment, each exposing its functionalities concurrently to multiple client agents, via suitable interfaces: its implementation is freely available at [14].

### 3.1 Symbolic Micro-intelligence at the Edge

In this section we discuss why the micro-intelligence vision and its incarnation in LPaaS and technology can be seen as the reification of the above guidelines for spreading intelligence in compliance with the Edge computing paradigm.

*Scale of Interest.* LPaaS is conceived for delivering intelligence at the *micro* scale of the system, at the connected (intelligent) things level—as suggested in [19], where the need for different scales of intelligence is highlighted. There, intelligence at the Edge is provided by gathering information and inference processes closer to the devices, by enabling local (symbolic) reasoning to complement global sub-symbolic reasoning (usually in the cloud).

*Big Data or Small Data.* Micro-intelligence proposes to synergistically exploit symbolic and sub-symbolic approaches, so as to better cope with the different requirements arising when, for instance, dealing with big data or small data analytics. Indeed, reasoning over symbolic knowledge bases allows consistency checking (i.e., detecting contradictions between facts or statements), classification (i.e., generating taxonomies), and other forms of deductive inference (i.e., revealing new, implicit knowledge given a set of facts).

Thus, we envision intelligent CPCS mitigating some of the issues experienced in sub-symbolic approaches by adding symbolic techniques to the picture, ultimately enabling novel forms of distributed and local reasoning. For instance, machine learning algorithms could generate the knowledge to be scattered across the Edge of the network, containing general information about the domain: then, such a knowledge could be refined by local constraints (e.g. specific spatio-temporal data). An LPaaS service can then reason over such knowledge to, e.g., guarantee consistency, or, infer novel information.

*Fault-Tolerance Requirement.* By isolating the failures of individual microservices, the LPaaS architecture helps achieving fault tolerance: since services can fail at any time, it is of paramount importance both that failures are quickly detected and, mostly, that services are automatically and quickly restored. “Fail-over” could be provided by the life-cycle management of the service itself, ensuring that a failed inference process is taken over by another LPaaS service.



*Safety-Critical Requirement.* Delivering the LPaaS services at the Edge delegates the individual system actors to take autonomous decisions, preventing possible failures in communication from leading to catastrophic consequences: hence, it helps supporting safety-critical applications. Although communication problems could arise because of the service architecture, such issues can be limited by embedding the LPaaS service in both individual agents and the environment, as well as by exploiting it as a library.

*Nature of Knowledge.* Being inherently rooted in the notion of situatedness, micro-intelligence is a natural choice for the design and implementation of contextual knowledge and reasoning: in fact, the LPaaS resolution process is intrinsically bound to the general computational context, both in terms of spatio-temporal context and environmental resources. This is why the LPaaS knowledge base contains (those) specific rules whose validity is bound in space and time (the “context”). Along this line, reasoning is delegated to components embedding the situated knowledge, which are the only capable of timely recognising exceptional situations: the inference capabilities enable the enactment of specific countermeasures, possibly taking real-time requirements into account. New facts and rules can also be acquired during the service lifetime, keeping the system up-to-date about the ever changing situation that individuals and environment resources are supposed to deal with. New rules can be inferred, as well.

*Embodied Computation.* Micro-intelligence lays its roots in the IoT world and its inner physical nature: accordingly, the LPaaS service comes with an *ad hoc* API for dealing with data collected by sensors and with streams of data (and therefore of solutions). As a result, LPaaS turns out to be an effective choice for capturing the *embodiment* feature of intelligence.

*Situatedness Requirement.* Stemming directly from the two previous considerations, LPaaS can well be regarded as an effective embodiment of the Situated Logic Programming paradigm. Indeed, the situated nature of the service is twofold: first, the LP inference process is itself situated in the spatio-temporal context, thus affecting solutions in relation to both the place where the service is physical located, and the time of the query; second, extra solve operations are provided for dealing with streams of solutions and with timed requests, while a dedicated API supports the exploration of services in a neighbourhood (the reader is referred to [3] for a thorough discussion).

## 4 Conclusion

The CPS revolution is characterised by decentralisation – from the Cloud towards the Edge – and by the need of exploiting low-level devices for the decision-making process in order to deal with issues including distributed processing, low latency, fault tolerance, better scalability and situated deliberation. However, there is still no general, well-founded software engineering approach

specifically addressing the issues of intelligent Edge computing for CPCS. Along this line, in this paper we define a some general design criteria along with a coherent set of design guidelines for distributing intelligence in CPCS. Such guidelines could lay the foundation for the definition of a full-fledged methodology for intelligent CPCS. To reify the guidelines in concrete architectures and technologies, following the insights from distributed cognition, we introduce the concept of *micro-intelligence* as the way to distribute chunks of symbolic intelligence at the Edge of CPCS. We exploit Situated Logic Programming (SLP) as the reference framework for micro-intelligence, empowering the Edge with knowledge and inference capabilities of computational logic, and show how LPaaS can straightforwardly work as the reference architecture as well as a potential technological embodiment.

## References

1. Ananthanarayanan, G., et al.: Real-time video analytics: the killer app for edge computing. *IEEE Comput.* **50**(10), 58–67 (2017). <http://ieeexplore.ieee.org/document/8057318>
2. Calegari, R.: Micro-intelligence for the IoT: logic-based models and technologies. Ph.D. thesis, Alma Mater Studiorum-Università di Bologna, Bologna, Italy (2018). <http://amsdottorato.unibo.it/8521>
3. Calegari, R., Ciatto, G., Mariani, S., Denti, E., Omicini, A.: Logic programming in space-time: the case of situatedness in LPaaS. In: Cossentino, M., Sabatucci, L., Seidita, V. (eds.) 19th Workshop “From Objects to Agents” (WOA 2018), CEUR Workshop Proceedings, vol. 2215, pp. 63–68. Sun SITE Central Europe, RWTH Aachen University, June 2018. <http://ceur-ws.org/Vol-2215/paper%5F11.pdf>
4. Calegari, R., Ciatto, G., Mariani, S., Denti, E., Omicini, A.: LPaaS as micro-intelligence: enhancing IoT with symbolic reasoning. *Big Data Cogn. Comput.* **2**(3) (2018). <http://www.mdpi.com/2504-2289/2/3/23>
5. Calegari, R., Denti, E., Mariani, S., Omicini, A.: Logic programming as a service. *Theor. Pract. Logic Program.* **18**(3–4), 1–28 (2018). <https://doi.org/10.1017/S1471068418000364>
6. Calegari, R., Denti, E., Mariani, S., Omicini, A.: Logic programming as a service in multi-agent systems for the Internet of Things. *Int. J. Grid Util. Comput.* **10**(4), 344–360 (2019). <https://doi.org/10.1504/IJGUC.2019.10022135>
7. Chen, M., Li, W., Fortino, G., Hao, Y., Hu, L., Humar, I.: A dynamic service migration mechanism in edge cognitive computing. *ACM Trans. Internet Technol.* **19**(2) (2019). <http://dl.acm.org/citation.cfm?id=3239565>
8. Cicirelli, F., Guerrieri, A., Mercuri, A., Spezzano, G., Vinci, A.: ITEMa: a methodological approach for cognitive edge computing IoT ecosystems. *Future Gener. Comput. Syst.* **92**, 189–197 (2019). <http://www.sciencedirect.com/science/article/pii/S0167739X17330224>
9. Clark, A., Chalmers, D.J.: The extended mind. *Analysis* **58**(1), 7–19 (1998). <http://www.jstor.org/stable/3328150>
10. Erl, T.: *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall/Pearson Education International, Upper Saddle River (2005). <http://dl.acm.org/citation.cfm?id=1088876>

11. Esteva, M., de la Cruz, D.D.L., Rosell, B., Arcos, J.L.A., Rodríguez-Aguilar, J.A., Cuní, G.: Engineering open multi-agent systems as electronic institutions. In: 19th National Conference on Artificial Intelligence (AAAI 2004), pp. 1010–1011. AAAI Press (2004). <http://dl.acm.org/citation.cfm?id=1597303>
12. Hollan, J., Hutchins, E., Kirsh, D.: Distributed cognition: toward a new foundation for human-computer interaction research. *ACM Trans. Comput.-Hum. Interact.* **7**(2), 174–196 (2000). <http://dl.acm.org/citation.cfm?id=353487>
13. Hu, L., Miao, Y., Wu, G., Hassan, M.M., Humar, I.: iRobot-factory: an intelligent robot factory based on cognitive manufacturing and edge computing. *Future Gener. Comput. Syst.* **90**, 569–577 (2019). <http://www.sciencedirect.com/science/article/pii/S0167739X1831183X>
14. Logic Programming as a Service (LPaaS) (2018). <http://lpaas.apice.unibo.it/>
15. Mariani, S., Omicini, A.: TuCSoN on cloud: an event-driven architecture for embodied/disembodied coordination. In: Aversa, R., Kołodziej, J., Zhang, J., Amato, F., Fortino, G. (eds.) ICA3PP 2013. LNCS, vol. 8286, pp. 285–294. Springer, Cham (2013). [https://doi.org/10.1007/978-3-319-03889-6\\_33](https://doi.org/10.1007/978-3-319-03889-6_33)
16. Omicini, A., Calegari, R.: Injecting (micro)intelligence in the IoT: logic-based approaches for (M)MAS. In: Lin, D., Ishida, T., Zambonelli, F., Noda, I. (eds.) MMAS 2018. LNCS (LNAI), vol. 11422, pp. 21–35. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-20937-7\\_2](https://doi.org/10.1007/978-3-030-20937-7_2)
17. Pace, P., Aloï, G., Gravina, R., Caliciuri, G., Fortino, G., Liotta, A.: An edge-based architecture to support efficient applications for healthcare Industry 4.0. *IEEE Trans. Ind. Informatics* **15**(1), 481–489 (2019). <http://ieeexplore.ieee.org/document/8370750/>
18. Rauch, E., Linder, C., Dallasega, P.: Anthropocentric perspective of production before and within industry 4.0. *Comput. Ind. Eng.* (In press). <http://www.sciencedirect.com/science/article/pii/S0360835219300233>
19. Rosenberg, D., Boehm, B., Wang, B., Qi, K.: Rapid, evolutionary, reliable, scalable system and software development: the resilient agile process. In: 2017 International Conference on Software and System Process (ICSSP 2017), pp. 60–69. ACM (2017). <http://dl.acm.org/citation.cfm?id=3084107>
20. Tang, B., Chen, Z., Hefferman, G., Pei, S., Wei, T., He, H., Yang, Q.: Incorporating intelligence in fog computing for big data analysis in smart cities. *IEEE Trans. Ind. Inf.* **13**(5), 2140–2150 (2017). <http://ieeexplore.ieee.org/document/7874167/>
21. Um, J.-S.: Futurology and future prospect of drone CPS. *Drones as Cyber-Physical Systems*, pp. 257–274. Springer, Singapore (2019). [https://doi.org/10.1007/978-981-13-3741-3\\_8](https://doi.org/10.1007/978-981-13-3741-3_8)
22. Waschull, S., Bokhorst, J., Molleman, E., Wortmann, J.: Work design in future industrial production: transforming towards cyber-physical systems. *Comput. Ind. Eng.* (In press). <http://www.sciencedirect.com/science/article/pii/S0360835219300683>