HAL
archives-ouvertes.fr

Reconsidering the Relationship between Cloud Computing and Cloud Manufacturing

Hélène Coullon, Jacques Noyé

▶ **To cite this version:**

Hélène Coullon, Jacques Noyé. Reconsidering the Relationship between Cloud Computing and Cloud Manufacturing. SOHOMA 2017, Oct 2017, Nantes, France. 2017. <hal-01591113>

HAL Id: hal-01591113
https://hal.inria.fr/hal-01591113

Submitted on 20 Sep 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Chapter 1
Reconsidering the Relationship between Cloud Computing and Cloud Manufacturing

Hélène Coullon, Jacques Noyé

Abstract History shows many relations between computer science and manufacturing processes, starting with the initial idea of "digital manufacturing" in the 70's. Since then, advances in computer science have given birth to the *Cloud Computing* (CC) paradigm, where computing resources are seen as a *service* offered to various end-users. Of course, CC has been used as such to improve the IT infrastructure associated to a manufacturing infrastructure, but its principles have also inspired a new manufacturing paradigm *Cloud Manufacturing* (CMfg) with the perspective of many benefits for both the manufacturers and their customers. However, despite the usefulness of CC for CMfg, we advocate that considering CC as a core enabling technology for CMfg, as is often put forth in the literature, is limited and should be reconsidered. This paper presents a new core-enabling vision toward CMfg, called *Cloud Anything* (CA). CA is based on the idea of abstracting low-level resources, beyond computing resources, into a set of core control building blocks providing the grounds on top of which any domain could be "cloudified".

Key words: Cloud Computing, Cloud Manufacturing, Resource Management, IaaS, MES

1.1 Introduction

There is already quite some history of using advances in computer science and information technologies to enhance manufacturing, starting with the initial idea of "digital manufacturing" in the 70's. Since then, three breakthroughs have taken place: the interconnection of computers through the Internet, the interconnection

Hélène Coullon · Jacques Noyé
IMT Atlantique, Inria, LS2N
Nantes, France
e-mail: {helene.coullon,jacques.noye}@imt-atlantique.fr

of the software and the hardware worlds through the provision of sensors, actuators and embedded controllers, and finally the advent of computing as a utility through *Cloud Computing* (CC). Of course, Cloud Computing has been used as such to improve the IT infrastructure associated to a manufacturing infrastructure, but its principles have also inspired a new manufacturing paradigm *Cloud Manufacturing* (CMfg) with the perspective of many benefits for both the manufacturers and their customers in terms of efficiency, cost and flexibility. Making such a paradigm a reality remains, however, an endeavor that requires coordinated progress in many domains.

Whereas the literature shows that CMfg is often addressed through the use of CC services, thus transforming a set of manufacturing processes and functions into Manufacturing-as-a-Service (MaaS), we claim in this paper that CC, despite being useful, does not give access to the software stack at the right level (because of heavy virtualization techniques) to be well suited to the requirements of the lowest layers of CMfg, responsible for packaging the physical manufacturing resources as a service. Instead of considering CC as a core enabling technology for CMfg, we present another vision where common control building blocks, responsible for low-level resource management, could be designed by abstracting away the resource specificities. We call our vision the *Cloud Anything* (CA) model. Thus, the CA model could be used to indifferently build the lowest resource-management-centric layer of CC, *i.e.*, the Infrastructure-as-a-Service layer (IaaS), or the equivalent lowest layer of CMfg.

The rest of the paper is organized as follows. Section 1.2 overviews work related to CMfg. As a foundation for the following sections, Sect. 1.3 compares computing and manufacturing per se. Section 1.4 comes back to their "Cloud" counterparts and compares them. This comparison leads to Sect. 1.5, which presents our Cloud Anything vision (CA) and opens a discussion about its advantages, difficulties and research challenges. Finally, Sect. 1.6 concludes this work and presents perspectives.

1.2 Related Work

As the term Cloud Manufacturing (CMfg) was coined from Cloud Computing (CC), a number of papers have studied how CMfg is related to CC with a look at future trends of CMfg.

In an early paper [20], Tao et al. present a 7-layer architecture of CMfg with a core cloud-service layer. CC is considered as a core enabling technology, to be complemented with IoT technologies together with new service layers (for instance, *Manufacturing-as-a-Service*) catering for resources that are not IT resources. These new layers are pictured as vertical, whereas the cloud services are pictured as horizontal but the precise relationship between the layers is not elucidated. This initial work is refined in [19], which distinguishes the *Internet of Things*, responsible for *service generation*, the *Internet of Services*, based on CC and responsible for *service*

management (including aggregation of lower services to create the manufacturing services evoked above), and the *Internet of Users*, responsible for *service application*, *i.e.*, on-demand use and cooperation.

In [23], Xu makes a broad comparison of CC and CMfg from low-level concerns (*e.g.*, virtualization), up to the application layers for end-users. Xu also introduces a fundamental distinction between *smart industries*, which implies the use of CC (IoT could be brought into the picture, too) inside CMfg, in order to handle the large data sets required to take smart decisions about production lines, and CMfg per se, *i.e.*, applying the Cloud Computing model to manufacturing processes. CMfg services are obtained by virtualizing resources made available to consumers through a Cloud platform. These resources can be tangible (they correspond to physical or basic computational resources) or intangible (they correspond to manufacturing *capabilities* [24]).

In [21], Wu et al. present another interesting survey of CMfg including a state of the art and a strategic vision of CMfg. The state of the art considers the use of CC as a *low-hanging fruit* and quotes the layered architecture of [20] but does not consider the interplay of CC and CMfg further as the research challenges focus on strategic, application-oriented, concerns.

A basic issue with standard, *public* CC services is that these services are located far from the shop floor, under external management, which creates performance as well as security issues. This can be alleviated by considering using *private* CC services. This is what Morariu et al. do [13]. The shop floor as well as part of the Manufacturing Execution System (MES) [6] is virtualized in a private cloud whereas a public cloud is used for high-level application services. As the physical resources are seen as agents, virtualized in the cloud, there is no intelligence left in the physical layer of the architecture. Experiments show the effect of virtualization on the performance of event propagation. A broader discussion of the advantages and disadvantages of relying on public, private, community and hybrid Cloud Manufacturing solutions is available in [9]. The discussion results in the design of a Hybrid Manufacturing Cloud (HMC) infrastructure. However, performance and virtualization issues are not considered. The focus is rather on access control and interoperability issues. These issues are addressed through the use of an ontology and rule-based reasoning with an implementation hosted by a public cloud.

In [7], Kubler et al. are also concerned with interoperability issues but they include IoT in the picture. IoT is considered as another core enabler and key in *product-centric control*. Of course, interoperability requires proper generic and open IoT standards. The issue is then to seamlessly integrate IoT and CC technologies. This dichotomy between a physical layer, handled by IoT technologies, and a virtual layer, handled by CC technologies can be generalized to *Cyber Physical Systems*. In [16], Queiroz et al. consider distributed, collaborative and adaptive process supervision and control in this context. The emphasis is on data analysis, with real-time analysis for monitoring and control at the physical level, and big-data analysis for optimization, planning, and decision making at the virtual level. Interestingly, the paper talks about *smart factories* but does not mention CMfg nor virtualization. It does not address the core of CMfg as defined by Xu.

All the above papers present some interesting viewpoints on CMfg as well as some preliminary ideas on the general architecture of CMfg infrastructures. However, none of them study the specific research challenges of resource management, which is the base of any higher-level solution (services, applications etc.). In this paper, we take the position of [23], spliting Cloud Manufacturing and Smart industries, and we present a new approach to adress the lowest levels of the shop floor and MES functionalities by considering CC as a sibling domain of CMfg rather than a core enabling technology.

1.3 Computing and Manufacturing

Before comparing Cloud Computing and Cloud Manufacturing, let us, in a first step, forget about the cloud. The basic question is then to relate Computing and Manufacturing. Here is a very simple way to look at it. Computations can be described by composing *computable functions*. A basic computation step can be informally written as $out = f(in_1, \ldots, in_k)$, taking k inputs in_i and producing an output *out* as a result of applying the function f to the inputs. This can also be seen as describing a basic manufacturing step, taking as inputs k parts and producing a new part as a result.

Of course, the nature of the inputs and outputs is quite different. They belong to the *virtual*, or digital, world in the first case, they are called *data*, and the *physical* one in the second case, we will call them *parts*. Note that the virtual world does not exist per se, but is rather an abstraction of the physical world (*e.g.,* a number is a series of digits, which are themselves abstractions of electrical signals). The nature of the function f is also different. It is a *computable function* in the first case, whose execution relies on a *programmable universal machine*, *e.g.,* a *computer*. In the second case, there is no universal machine to execute the *manufacturing function*. Indeed, an initial step requires either to rely on an existing machine in case of an elementary step or to configure a series of such machines into a proper assembly line. In practice, such a process is not completely absent from computing (it occurs when compiling programs) but can easily be abstracted away. Connecting manufacturing machines and computers results in *hybrid machines*, which can take as input both parts and data and produce as output both parts and data.

All these machines can be connected at different scales through digital and production/transport networks with a huge difference: whereas data can travel at the speed of light, parts can hardly reach the speed of sound. Such a gap also exists when considering *context switching*, *i.e.*, the possibility to interrupt the execution of a computation or a manufacturing step to schedule another one. This is very easy and fast in the virtual world, at different granularity levels, from ultra-lightweight threads to *virtual machines* (VMs), with very few constraints on when this can happen. As a result, it is very easy to share a machine to perform various computations potentially related to various applications on the behalf of various users. This sharing, managed by the Operating System (OS), encompasses hardware and software

resources (Central Processing Units, *i.e.*, CPU, memory, files...). On the other hand, sharing a manufacturing machine is possible with a different latency granularity as building a part or moving a part is a much longer process.

Whereas, in the computing world, an OS is responsible for the automatic management of all resources (CPU, RAM, disk, network, files, etc.) within a computer and their sharing between functions, the automatic management of manufacturing resources, and their sharing between manufacturing orders, has led to the design of Manufacturing Execution Systems (MES) [6].

1.4 Cloud Computing and Cloud Manufacturing

The previous section has compared manufacturing with computing, including MESs and OSes. It appears that many similarities can be found in these concepts. In this section we explore further these similarities by studying CC and CMfg and comparing them.

1.4.1 Cloud Computing

The National Institute of Standards and Technology (NIST) defines Cloud Computing [12] as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction".

This technical model is also closely linked to an economic model whereby users only pay for what they consume, also called the *pay-as-you-go* model. Thanks to this economic model, CC is now a widely spread model used by many companies, public or private institutes, and even by many individual persons all over the world. Actually, this economic model is responsible for an easy, cheap, permanent and unlimited access to computing and storage resources everywhere. For example, for small companies or start-ups, taking advantage of CC resources is cheaper than buying physical resources and paying IT administrators.

A cloud is composed of one or multiple large pools of distributed heterogeneous computers, called *servers*, $s_1, \ldots, s_n$, each one with its own set of resources (CPU, memory, etc.). One pool of servers is often called a data center. To guarantee the viability of the model, the utilization of the pool of resources is optimized and can be reorganized on the fly to be able to answer new incoming user requests. If a computation f can be executed by any of those servers, its execution is specific to the target machine. For this reason, the main enabling technology of CC is *virtualization*, *i.e.*, the ability of creating and managing *virtual machines* (VMs). This mechanism offers a way to decouple a computer (as a set of physical computing resources) from its use, thus enabling easy migration of a computation or a function f to another

computer during its execution. This mechanism enables agility and an efficient way to optimize the utilization of a pool of resources. Moreover, this mechanism enables heterogeneity of computing resources, which is vital for a durable and scalable CC solution.

By using CC a user can focus on its core domain without knowledge of lower-level IT concerns. Most of the time, three different abstraction levels are proposed to the user within a CC solution: (1) *Infrastructure-as-a-Service* (IaaS), where processing, network and storage resources are requested by the user under the form of virtual machines and virtual networks; (2) *Platform-as-a-Service* (PaaS), where users request for complete development platforms, hence leading to the automatic provisioning of virtual machines with an initial operating system and possibly additional specific libraries, frameworks or tools proposed by the cloud provider; and (3) *Software-as-a-Service*, where users request for the use of a full application that implicitly (for the user) leads to the provisioning of a complete software stack onto a set of virtual machines.

One can note that by increasing the abstraction level, the user also may change, from low-level developers (using IaaS), companies developing a cloud-based application by using existing development frameworks (using PaaS), or end-users using an application on their smartphone (thus using SaaS). When a higher abstraction level is defined, it is based on lower ones, the IaaS being the lowest level of CC solutions. In this paper, we focus on the IaaS level of usual CC facilities. Furthermore we claim that a set of common building blocks can be found in both a IaaS system and the lowest layers of a CMfg system by considering a resource as a broader generic notion.

A IaaS is responsible for the automatic (or semi-automatic) management and optimization of a pool of computing, storage and network resources (a data center) within a CC infrastructure. A IaaS is usually structured as a *MAPE loop* often used in computer science [4, 11]. A MAPE loop is a perpetual loop composed of four coarse-grain steps: (1) *monitor*, to get information from the infrastructure and external inputs; (2) *analyze*, to analyze monitoring information and take decisions; (3) *plan*, to schedule decisions taken during the analysis; and (4) *execute*, to execute the plan. A MAPE loop is the basis of any autonomic system. For example, within OpenStack, the de-facto open-source solution to address the IaaS level of the CC paradigm, the Telemetry project[1] represents the Monitoring step of MAPE, the Nova project[2] represents the Planning step of MAPE, etc. Thus, a IaaS system can be defined as an operating system to handle CC infrastructures, *i.e.*, pools of on-demand distributed computing, storage and network resources.

[1] `https://wiki.openstack.org/wiki/Telemetry`

[2] `https://wiki.openstack.org/wiki/Nova`

1.4.2 Cloud Manufacturing

Inspired by CC, which transforms a pool of IT resources into a set of on-demand services, the vision of [23] is that Cloud Manufacturing transforms a pool of manufacturing resources and capabilities into on-demand manufacturing services. Thus, in [23], Cloud Manufacturing is defined as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable manufacturing resources (*e.g.*, manufacturing software tools, manufacturing equipment, and manufacturing capabilities) that can be rapidly provisioned and released with minimal management effort or service provider interaction".

CMfg can be, at least, as complex as CC, *i.e.*, composed of multiple layers of services inherited from MESs and also composed of an economic model. In this paper, we restrict our vision to the lowest level of manufacturing and computing systems: the resource management and its optimization. Thus, in this paper, if for CC we focus on the IaaS layer, *i.e.*, an operating system for distributed on-demand computing resources, for CMfg, we focus on the transformation of the lowest levels of MESs to a management system for distributed on-demand manufacturing resources.

For this reason, this paper separates CMfg from CC as a start, even if both of them are related at some point. On the one hand, we consider that taking advantage of a combination of CC, Manufacturing systems and IoT techniques (getting information from small sensors), to smartly take decisions regarding production, products and related services, is not the definition of CMfg but is more related to the concept of *Smart Industry* [16]. Thus, on the other hand, we consider that CMfg refers to enabling on-demand access to manufacturing resources everywhere, which does not necessarily include the use of CC but is related to it.

As mentioned in Sect. 1.2, most existing CMfg initiatives suggest using CC facilities to host low- (physical layer) and high-level functionalities of MESs, as well as higher business layer functionalities [7, 13, 17]. By using CC facilities, manufacturing industries get rid of the burden of investing and maintaining IT resources locally, and also increase the quality of service regarding fault tolerance and security for cloud-hosted functionalities.

In [13, 17] private CC solutions are used to deploy low-level functions responsible for the physical layer in MESs. A private CC offer a proximity compared to public CC that leads to lower latencies and facilitates security and data privacy management (within a restricted geographical region). However, renting private CC facilities is more expensive than renting public facilities, and private resources are limited.

The authors of [17] objectively claim that by using CC the virtual machines used will always be available, because of the High Availability (HA) of CC, but this does not guarantee that the applications deployed within the VMs will automatically inherit the same availability. This illustrates that, in order to offer a complete CMfg solution, as defined in [23], with equivalent properties to those offered by CC, the same kind of development efforts have to be made, and that, using CC to deploy MES functionalities in VMs does not make CMfg.

This has led us to devise a new approach to CMfg.

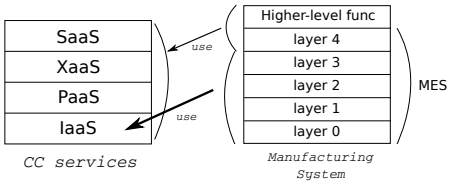

Fig. 1.1: Usual CMfg approach using CC to deploy all MES and business functions

1.5 Cloud Anything

This section presents our new approach to handling resources in CMfg and, more generally, to build the lowest layers (such as a IaaS) of any Cloud system. In a first step, a centralized setting is assumed. In a second step, distribution aspects are discussed based on the emerging new paradigms for geo-distributed Clouds.

1.5.1 Abstraction of Resources in Low-level Cloud Models

As explained in Sect. 1.4 usual CMfg approaches use CC facilities to deploy various kinds of low- and high-level functionalities of MES systems and business functions. A major advantage of this approach is to externalize the deployment and management of these control functions. Thus, the IT management is not handled by the manufacturing enterprise, and CC operators offer guarantees on fault tolerance, security, high availability, etc. This usual approach is represented in Fig. 1.1. In this figure, the 5-layer ISA-95.03 specification of MES as defined in [17] is used. Most of the time the lowest layers, the ones responsible for resource management, are placed in IaaS-provided VMs. Higher layers can also be handled by CC services. The figure does not make any assumption on the type of the infrastructure (public, private...).

However, deploying applications and functions on a CC infrastructure is not sufficient to inherit all the characteristics of the CC paradigm [17]. A CC facility offers a precise type of service, *i.e.*, an on-demand infrastructure, platform or application. When using the IaaS level, the CC operator guarantees high availability, scalability, fault tolerance and security onto rented virtual machines, but cannot guarantee that the same properties will apply to the application hosted on VMs. This is why we think that implementing manufacturing functions using CC, despite being useful, does not lead by itself to CMfg. To offer manufacturing as a service, the same efforts that have been undertaken to implement CC must also be undertaken, with their own specificities, to implement CMfg.

In addition to this, we have noticed many conceptual similarities between IaaS and MES systems. In both cases a pool of distributed resources has to be managed according to a set of requests, computing requests in one case and production or-

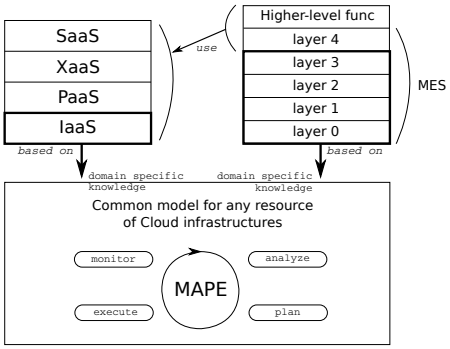

Fig. 1.2: New approach to CMfg, the Cloud Anything model

ders in the other. Hence, in both cases the concept of resource is used and common control building blocks can be found. For example, building blocks responsible for receiving requests, for computing scheduling optimizations, for receiving monitoring information from sensors, for quality management, etc., can be found in both IaaS and MES systems. For this reason, we propose another way to reach CMfg.

Instead of deploying MES functions in a CC infrastructure using IaaS, we propose to build a common set of cloud control building blocks that, in a first step, abstract the concept of a resource, but, in a second step, can be specialized to a specific domain using meta information specific to the domain. Depending on various factors linked to the domain as well as technological constraints, this specialization could happen at modeling, implementation, or execution time. This idea is depicted in Fig. 1.2 where it is applied to both CC and CMfg. As, in both cases, an autonomic system is built, we assume that the set of common building blocks are organized around a MAPE loop. However, such an organization is quite generic. It should be refined in order to integrate some other important aspects of resource management, *e.g.*, an economic model, required to control resource consumption and manage billing information. We call this model *Cloud Anything* (CA) as it could be applied to various domains, beyond CC and CMfg.

This model allows using CC facilities to get more computing resources if needed, as implied in the figure by the thin arrow. Still, the lowest layers of MESs do not need to be deployed using IaaS when high performance is required. IaaS is a very high-level on-demand computation service, resulting in high overheads (due to data transfers through the network, distant data centers, and complex software stacks for VM management). Our proposal offers a generic way to build a Cloud Anything infrastructure, decoupled from the underlying resources. Moreover, it offers the possibility to handle both CC and CMfg within a single operator with minor overheads.

On the other hand, what happens if resources embed computing resources, *e.g.*, a computer unit responsible for hosting control? Does this computing resource, used within the IaaS-like implementation layer, require specific administration without

any support from the infrastructure? Such a situation already happens in CC where deploying a IaaS layer requires computing resources. Many solutions are appearing to address such a situation. These solutions consist in the automation of the deployment of IaaS components onto distributed computers [3] (Juju[3], Kubernetes[4] [18], TripleO[5], etc.). The weird and touchy part of deployment automation is that such a system is close to a IaaS system itself (as a IaaS is responsible for the deployment and management of virtual machines), but the system can be much lighter than a IaaS as all Cloud-related parts of a IaaS do not need to be handled. Moreover, such solutions can work without virtual machines (namely bare-metal resources), or can use lighter virtualization techniques such as containers[6] [5] or uni-kernels [22].

To get back to our proposal depicted in Fig. 1.2, many research challenges, in addition to more technical issues, arise from abstracting resources from a set of resource management building blocks, thus from externalizing any specific knowledge from the type of resource. However, such work could be a great advance to easily move toward various kinds of Cloud infrastructures.

1.5.2 Advances in Geo-distributed Clouds

Fog and Edge Computing [1, 10, 15] are two rather new CC paradigms. They extend the centralized CC concept with a massively geo-distributed set of micro Clouds deployed on strategic points inside the highly-connected Internet network and into small intelligent objects. Hence, CC is extended with Fog (core network micro data centers) and Edge (smart objects used as small CC resources) facilities, which are closer to data sources and end-users. Both paradigms can be classified as massively geo-distributed utility computing.

Many advantages can be gained from the Fog and the Edge Computing paradigms. First, one outcome and drawback of centralized CC is the huge amount of energy consumed by the infrastructure itself, but also by cooling facilities [14]. Thus, by limiting the size of centralized CC and by adding many small Clouds into the highly connected infrastructure, the energy consumption can be more easily handled. Second, like private CC, these paradigms can improve latency and bandwidth issues, compared to centralized distant CC, and can help controlling security or data privacy.

However, research challenges also arise from the design of Fog and Edge infrastructures, particularly when dealing with massively geo-distributed control building blocks of IaaSes for very heterogeneous resources [3, 8].

Interestingly, CMfg also needs to be designed as a geo-distributed Cloud managing very heterogeneous kinds of resources and gathering multiple sites of manufac-

[3] `https://jujucharms.com/`

[4] `http://kubernetes.io/`

[5] `https://wiki.openstack.org/wiki/TripleO`

[6] `https://www.docker.com/`

turing to aggregate enough resources for elasticity, high availability, etc. The work on distributed MESs and their design as multi-agent holonic systems (see, for instance, [13, 16, 17]) is very relevant in this context, with some shared research challenges appearing when designing a massively distributed IaaS and a cloud-based holonic MES. In other words, considering our proposal, the set of generic building blocks could also be designed such that each block can be distributed, thus making it possible to manage geo-distributed complex Cloud Anything infrastructures. Moreover, at higher levels, including Smart industries, where CC is used to conduct heavy computations, upgrading to Fog and Edge CC seems to be a very good choice to get lower latencies and stronger security properties. There is a potential for holonic systems to inspire IaaSes under design for Fog and Edge computing as well as the other way around. This, in addition to our proposal, represents a great opportunity for collaborative work.

1.6 Conclusion

This paper has explored the relationship between computing and manufacturing and between Cloud Computing and Cloud Manufacturing, from a Computer Science perspective, with a focus on the lower service-level of resource management.

This has suggested considering the design of a generic resource management layer compatible with a wide range of resources and generalizing the IaaS layer of Cloud Computing beyond computing resources. This is a first step toward a generalization of the concept of Cloud, enabling "Cloudification" of various domains including Manufacturing. This also makes it possible to handle both Cloud Computing and Cloud Manufacturing within a single operator, with minor overheads compared to usual approaches. The design of such a generic layer is challenging but could benefit from existing results and on-going work on both Cloud Computing and Cloud Manufacturing with synergistic effects.

The next step would be to identify the common control building blocks of resource management for Clouds (Computing and Manufacturing) and explore how to implement our proposal on an appropriate subset in order to produce a proof of concept, with a particular attention paid to a distributed (or holonic) design.

References

1. F. Bonomi, R. Milito, J. Zhu, and S. Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, MCC '12, pages 13–16. ACM Press, 2012.
2. T. Borangiu, D. Trentesaux, A. Thomas, P. Leitão, and J. B. Oliveira, editors. *Service Orientation in Holonic and Multi-Agent Manufacturing : Proceedings of SOHOMA 2016*. Springer International Publishing, 2017.

3. H. Coullon, C. Pérez, and D. Pertin. Production deployment tools for IaaSes: an overall model and survey. In *IEEE International Conference on Future Internet of Things and Cloud (FiCloud) 2017*, Prague, Czech Republic, Aug. 2017.
4. M. C. Huebscher and J. A. McCann. A survey of autonomic computing – degrees, models, and applications. *ACM Computing Surveys*, 40(3):7:1–7:28, Aug. 2008.
5. A. M. Joy. Performance comparison between linux containers and virtual machines. In *2015 International Conference on Advances in Computer Engineering and Applications*, pages 342–346, Mar. 2015.
6. J. Kletti. *Manufacturing Execution System - MES*. Springer Publishing Company, Incorporated, 1st edition, 2010.
7. S. Kubler, J. Holmström, K. Främling, and P. Turkama. Technological theory of cloud manufacturing. In *Service Orientation in Holonic and Multi-Agent Manufacturing - Proceedings of SOHOMA 2015, Cambridge, UK, November 5-6, 2015*, pages 267–276, 2015.
8. A. Lebre, J. Pastor, A. Simonet, and F. Desprez. Revising OpenStack to operate fog/edge computing infrastructures. In *IEEE International Conference on Cloud Engineering*, Vancouver, Canada, Apr. 2017.
9. Y. Lu, X. Xu, and J. Xu. Development of a hybrid manufacturing cloud. *Journal of Manufacturing Systems*, 33(4):551 – 566, 2014.
10. R. Mahmud and R. Buyya. Fog computing: A taxonomy, survey and future directions. *CoRR*, abs/1611.05539, 2016.
11. M. Maurer, I. Breskovic, V. C. Emeakaroha, and I. Brandic. Revealing the MAPE loop for the autonomic management of cloud infrastructures. In *2011 IEEE Symposium on Computers and Communications (ISCC)*, pages 147–152, June 2011.
12. P. M. Mell and T. Grance. The NIST definition of cloud computing. Special Publication 800-145, National Institute of Standards & Technology, Gaithersburg, MD, United States, 2011.
13. O. Morariu, T. Borangiu, and C. Morariu. From service oriented to cloud powered manufacturing systems. In *Proceedings of the 2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing*, UCC '13, pages 83–90, Washington, DC, USA, 2013. IEEE Computer Society.
14. A.-C. Orgerie, M. D. d. Assuncao, and L. Lefevre. A survey on techniques for improving the energy efficiency of large-scale distributed systems. *ACM Computing Surveys*, 46(4):47:1–47:31, Mar. 2014.
15. C. Perera, Y. Qin, J. C. Estrella, S. Reiff-Marganiec, and A. V. Vasilakos. Fog computing for sustainable smart cities: A survey. *ACM Computing Surveys*, 50(3):32:1–32:43, June 2017.
16. J. Queiroz, P. Leitão, and E. Oliveira. *Industrial Cyber Physical Systems Supported by Distributed Advanced Data Analytics*, pages 47–59. In Borangiu et al. [2], 2017.
17. S. Răileanu, F. Anton, and T. Borangiu. *High Availability Cloud Manufacturing System Integrating Distributed MES Agents*, pages 11–23. In Borangiu et al. [2], 2017.
18. D. K. Rensin. *Kubernetes - Scheduling the Future at Cloud Scale*. O'Reilly, June 2015.
19. F. Tao, Y. Cheng, L. D. Xu, L. Zhang, and B. H. Li. CCIoT-CMfg: Cloud computing and internet of things-based cloud manufacturing service system. *IEEE Transactions on Industrial Informatics*, 10(2):1435–1442, May 2014.
20. F. Tao, L. Zhang, V. C. Venkatesh, Y. Luo, and Y. Cheng. Cloud manufacturing: a computing and service-oriented manufacturing model. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 225(10):1969–1976, 2011.
21. D. Wu, M. J. Greer, D. W. Rosen, and D. Schaefer. Cloud manufacturing: Strategic vision and state-of-the-art. *Journal of Manufacturing Systems*, 32(4):564 – 579, 2013.
22. B. Xavier, T. Ferreto, and L. Jersak. Time provisioning evaluation of KVM, Docker and unikernels in a cloud platform. In *2016 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, pages 277–280, May 2016.
23. X. Xu. From cloud computing to cloud manufacturing. *Robotics and Computer-Integrated Manufacturing*, 28(1):75 – 86, 2012.
24. L. Zhang, Y. Luo, F. Tao, B. H. Li, L. Ren, X. Zhang, H. Guo, Y. Cheng, A. Hu, and Y. Liu. Cloud manufacturing: a new manufacturing paradigm. *Enterprise Information Systems*, 8(2):167–187, 2014.