CLOUDLIGHTNING: A Framework for a Self-Organising and Self-Managing Heterogeneous Cloud
(Position Paper)

Theo Lynn[1], Huanhuan Xiong[2], Dapeng Dong[2], Bilal Momani[2], George Gravvanis[3], Christos Filelis-Papadopoulos[3], Anne Elster[4], Malik Muhammad Zaki Murtaza Khan[4], Dimitrios Tzovaras[5], Konstantinos Giannoutakis[5], Dana Petcu[6], Marian Neagul[6], Ioan Dragon[6], Perumal Kuppudayar[7], Suryanarayanan Natarajan[7], Michael McGrath[7], Georgi Gaydadjiev[8], Tobias Becker[8], Anna Gourinovitch[1], David Kenny[1] and John Morrison.[2]

[1]*Irish Centre for Cloud Computing and Commerce, DCU, Dublin, Ireland*
[2]*Irish Centre for Cloud Computing and Commerce, UCC, Cork, Ireland*
[3] *Democritus University of Thrace, Xanthi, Greece*
[4] *Norwegian University of Science and Technology, Trondheim, Norway*
[5] *The Centre for Research and Technology, Hellas, Thessaloniki, Greece*
[6] *Institute e-Austria Timisoara and West University of Timisoara,Timisoara, Romania*
[7]*Intel, Leixlip, Ireland*
[8]*Maxeler, London, United Kingdom*

{theo.lynn, anna.gourinovitch, david.kenny}@dcu.ie, {j.morrison, h.xiong,d.dong, b.momani}@cs.ucc.ie, {ggravvan, cpapad}@ee.duth.gr, {elster, malikmk}@idi.ntnu.no,{dimitrios.tzovaras, kgiannou}@iti.gr, petcu@info.uvt.ro, marian.neagul@e-uvt.ro, idragan@ieat.ro, {perumal.kuppuudaiyar, suryanarayanan.natarajan, michael.j.mcgrath}@intel.com, {georgi, tbecker}@maxeler.com

Keywords: Cloud Computing Models, Cloud Infrastructures, Cloud Architecture, Cloud Computing, Cloud Services Self-organisation, Self-Management, Heterogeneous Resources, Resource as a Service, Cloud Orchestration, Data flow engine, Many-integrated cores, MIC, GPU, FPGA, DFE

Abstract: As clouds increase in size and as machines of different types are added to the infrastructure in order to maximize performance and power efficiency, heterogeneous clouds are being created. However, exploiting different architectures poses significant challenges. To efficiently access heterogeneous resources and, at the same time, to exploit these resources to reduce application development effort, to make optimisations easier and to simplify service deployment, requires a re-evaluation of our approach to service delivery. We propose a novel cloud management and delivery architecture based on the principles of self-organisation and self-management that shifts the deployment and optimisation effort from the consumer to the software stack running on the cloud infrastructure. Our goal is to address inefficient use of resources and consequently to deliver savings to the cloud provider and consumer in terms of reduced power consumption and improved service delivery, with hyperscale systems particularly in mind. The framework is general but also endeavours to enable cloud services for high performance computing. Infrastructure-as-a-Service provision is the primary use case, however, we posit that genomics, oil and gas exploration, and ray tracing are three downstream use cases that will benefit from the proposed architecture.

1 INTRODUCTION

Current cloud infrastructures are typically centrally managed and composed of a large number of machines of the same type, made available to the end user using the three standard service models: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). As clouds increase in size and as machines of different types are added to the infrastructure to maximize performance and power efficiency, heterogeneous clouds are being created; posing significant challenges. To efficiently access heterogeneous resources and, at the same time, to exploit these resources to reduce application development effort, to make optimisations easier and to simplify service deployment, requires a re-evaluation of our approach to service delivery. The adoption of heterogeneous computing resources by cloud consumers will allow for improved resource efficiency and hence reduced energy use. Market demand for better resource management at the PaaS and IaaS layers, combined with both demand and adoption of heterogenous resources is rapidly increasing complexity of the cloud ecosystem, which over time will push the boundaries of traditional cloud management techniques.

In this paper, we present our research agenda and ongoing work for the development of CloudLightning, a new cloud management and delivery model designed on the principles of self-organisation and self-management. We propose a novel architecture for provisioning heterogeneous cloud resources to deliver services, specified by the user, using a bespoke service description language. Service descriptions, provided by prospective cloud consumers, will result in the cloud evolving to deliver the required services. The self-organising behaviour built into, and exhibited by, the cloud infrastructure will result in the formation of a heterogeneous collection of homogeneous resource coalitions; capable of meeting service needs. Typically, the quality, cost and performance of each heterogeneous service will differ. The user will choose from these offerings and appropriate resources will be commissioned to deliver the desired service.

An important objective in creating this system is to remove the burden of low-level service provisioning, optimisation and orchestration from the cloud consumer and to vest them in the collective response of the individual resource elements comprising the cloud infrastructure. A related objective is to locate decisions pertaining to resource usage with the individual resource components, where optimal decisions can be made. Currently, successful cloud service delivery relies heavily on the over-provisioning of resources. Our goal is to address this inefficient use of resources and consequently to deliver savings to the cloud provider and the cloud consumer in terms of reduced power consumption and improved service delivery, with hyperscale systems particularly in mind.

The rest of the paper is organised as follows. Section 2 summarises related work. Section 3 briefly mentions three challenging motivating examples used to guide the architecture design presented in Section 4. The paper concludes with a discussion of work currently in progress.

2 RELATED WORK

The abstraction of hardware architecture from end user applications and indeed from end users and programmers is seen as a key benefit of the cloud computing paradigm (Crago and Walters, 2015). This abstraction allows data centers and service providers to maintain, enhance and expand underlying infrastructure without the absolute necessity of making associated changes in software. As such cloud service providers have been typically able to achieve significant performance improvements in cloud computing architecture performance and scalability by natural improvements in microprocessor capability in line with Moore’s Law (Crago and Walters, 2015). These clouds often make use of homogenous resources; typically identical general purpose microprocessors that are relatively inexpensive. In these systems, over-provisioning as a method for assuring service availability and performance is used extensively. Consequently, servers are largely underutilised, relative to their peak load, typically with frequent idle times resulting in energy inefficiencies. These inefficiencies can be further exacerbated by non-uniform energy consumption by servers (Barroso and Holzle, 2007; Awada et al. 2014). In addition, associated energy costs related to cooling and air conditioning and the need for multiple data conversions in the power distribution system, particularly for backup systems result in a dramatic cost and environmental impact (Awada et al. 2014).

With the explosion in cloud computing usage, infrastructure providers face increased market demand for improved performance at lower costs and at the same time are under increased scrutiny from policymakers relating to their environmental impact. Unsurprisingly, this area has been a fruitful one for academic research. In addition to new cooling methods, data center location, layout and equipment design, a number of solutions have been proposed to address resource management in cloud computing including data voltage and frequency scaling (DVFS), dynamic power management (DPM), virtual machine (VM) consolidation and allocation, task consolidation amongst others (Kilazovich et al. 2010; Lee and Zomaya, 2010; Beloglasov et al. 2012; Awada et al. 2014; Alzamil et al. 2015).

Research suggests that while transistors continue to shrink, concurrent limitations on power density, heat removal and related considerations require a different architecture strategy for improved processor performance by adding identical, general-purpose cores (Esmaeilzadeh et al. 2011; Crago and Walters, 2015). One solution is the heterogeneous cloud. Unlike traditional cloud infrastructure built on the same processor architecture, heterogeneity assumes a cloud that makes use of different specialist hardware devices that can accelerate the completion of specific tasks or can be put in a state where less power is used (or indeed deactivated if possible) when not required; thus maximising both performance and energy efficiency (Scogland et al. 2014). This is particularly relevant to compute-intensive and high performance computing (HPC) workloads. Three such common architectures with relatively high computation/power consumption ratios include graphics processing units (GPUs), many integrated cores (MICs) and data flow engines (DFEs). The programmable nature of GPUs, MICs and DFEs/FPGAs gives them wide range of application use cases and particularly for HPC tasks.

The heterogeneous cloud is still at a nascent stage, however, larger cloud infrastructure providers are offering commercial services e.g., Amazon Web Services offers a variety of GPU services. As demand for better processor, price and power performance increases, it is anticipated that larger infrastructure providers will need to cater for several of these processor types and specifically for the emerging HPC public cloud market (IDC, 2014a). However, integrating and managing these different architectures independently and with an existing general purpose cloud architecture is not without challenges; not least of which is access to a pool of qualified engineers with deep IT knowledge. The adoption of heterogeneous resources will dramatically increase the complexity of an already complex cloud ecosystem. We present self-organisation and self-management as powerful techniques for addressing this complexity.

Self-organisation is a powerful technique for addressing complexity and borrows heavily from the natural sciences and the study of natural systems (Gell-Mann, 1988; Schuster, 2007). In the computing context, Heylighen and Gershenson (2003) define organisations as “structure with function” and self-organisation as a functional structure that appears and maintains spontaneously. Control is distributed over all of the large number of participating components in the system. Of self-organisation, Alan Turing (1952) once observed that “global order arises from local interactions”. In this context, global order is achieved through propagation and adaptation. Components in a self-organising system are mutually dependent and typically only interact with nearby components. However, the system is dynamic and therefore the components can change state to meet mutually preferable, satisfactory or stable states (Heylighen and Gershenson, 2003). As they meet these states, they adapt and achieve “fit” and this propagation of “fit” results in system growth. Structural complexity is driven by increasing the interchangeability and individuality of components capable of achieving fit. As more and more components adapt and become assimilated within the system, complexity increases to incorporate the individual characteristics of components. Growth only stops when resources have been exhausted and self-maintenance is the *de facto* purpose of the system. As such, self-organising systems are defined by their robustness, flexibility, and adaptivity (Heylighen, 2001).

Self-management has been posited as a solution to complexity in IT infrastructure development generally and cloud computing specifically (Kramer and Magee, 2011; Puviani and Frei, 2013). It has its roots in autonomic computing; these systems are designed to react to internal and external observations without human intervention to overcome the management complexity of computer systems (Zhang et al, 2010). As such, self-managing systems are described in terms of four aspects of self-management, namely, self-configuration, self-optimisation, self-protection and self-healing (Parashar and Hariri, 2005). In line with autonomic computing, use of control or feedback loops such as MAPE-K is regularly referenced in self-management literature (Kephardt at al. 2007).

The application of self-organising and self-management principles to cloud computing is at an early stage. Zhang et al. (2010) posit that cloud computing systems are inherently self-organising and, while they exhibit autonomic features, are not self-managing as they do not have reducing complexity as a goal. Marinescu et al. (2013) argue that cloud computing represents a complex system and therefore self-organisation is an appropriate technique to address this complexity. They propose an auction-driven self-organising cloud delivery model based on the tenets of autonomy of individual components, self-awareness, and intelligent behaviour of individual components. They simulate a new model of a complex heterogeneous system with a very large number of components and with many interaction channels among them. Preliminary results suggest that the self-organising architecture was scalable and the bidding mechanisms and coalition formation algorithms are feasible at scale.

Extending work on self-manageable cloud services by Brandic (2009) at an individual node level, Puviani and Frei (2013) propose self-management as a solution for managing complexity in cloud computing at a system level. They propose a catalogue of adaptation patterns based on requirements, context and expected behaviour. These patterns are classified according to the service components and autonomic managers. Control loops following the MAPE-K approach enact adaptation. In their approach, each service component is autonomous and autonomic and has its own autonomic manager that monitors itself and the environment. The service is aware of changes in the environment including new and disappearing components and adapts on a negotiated basis with other components to meet system objectives. Initial simulations suggest that self-management is a promising approach but like Marinescu et al. (2013) further research in to a real implementation is necessary to reach final conclusions.

3 POTENTIAL IAAS AND HPC USE CASES

Industry research suggests that a significant feature of future HPC purchases will emphasise GPUs, Intel MICs/Xeon Phi processors and FPGAs (Intersect360 Research, 2015; IDC, 2014b). The top attributes of future HPC systems from a purchasing perspective were price, total cost of ownership, performance and hardware/software compatibility with existing HPC systems (IDC, 2014b).

The primary use case for CloudLightning is Infrastructure-as-a-Service (IaaS) provision, however, three preliminary HPC use cases have been identified as suitable to validate CloudLightning, namely (i) genomics, (ii) oil and case exploration, and (iii) ray tracing. IaaS provision may include public cloud service providers and/or companies providing, configuring or using private and hybrid clouds. In each case, CloudLightning is anticipated to offer better performance and greater energy efficiency. By exploiting heterogeneous computing technologies, we anticipate significantly improved performance/cost and performance/watt but also enabling computation to be hosted at large-scale in the cloud, making large-scale compute-intensive application and by-products accessible and practical from a cost and time perspective for a wider group of stakeholders. In each use case, relatively small efficiency and accuracy gains can result in competitive advantage for industry.

4 PROPOSED ARCHITECTURE

Figure 1 depicts the high level architecture for the proposed self-organising self-managing heterogeneous cloud. The life-cycle of a proposed cloud service is initiated by an Enterprise Application Developer ("EAD") by submitting a Blueprint to a Gateway Service. A Blueprint is a graph representing a workflow of Services collectively composed to automate a particular task or business process.

The Gateway Service is a front-facing component of the CloudLightning system, abstracting the inner workings of the backend components and providing a unified access interface to the CloudLightning system. The Gateway Service provides a graphical user interface for EADs. As well as providing a workflow and an enabling process for Blueprint development, it also sends requests for resource options capable of running the Blueprint, receives resource deployment option(s), presents options to the EAD and ultimately enables the EAD to select and deploy the Blueprint to the chosen resources automatically.

The Gateway Service communicates to a self-contained self-organising self-managing system, which we define as a "Cell". We envisage a Cell being typically associated with one geographical region and the CloudLightning system as a whole being composed of multiple Cells.

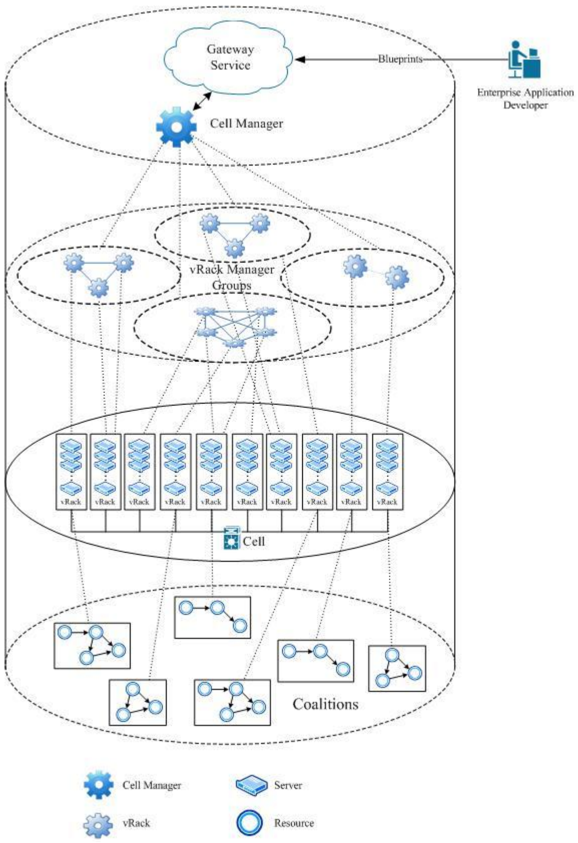

Figure 1 CloudLightning Architecture

Each Cell is composed of resource fabric comprising heterogeneous compute, storage and network resources organised in to groups, each of which is called a vRack. A vRack Manager is used to maintain information about groups of servers with the same type of physical resources and manages a logical group of these resources which is called a Coalition. Coalitions can be formed either statically or dynamically. Each vRack is self-managed. Local decisions are made by the vRack Manager, to decide on how coalitions are optimally formed (or identified) to deliver the service - reflecting self-optimisation at the vRack Manager Level. Self-healing and Self-protection may be addressed by feedback loops and fault tolerance, respectively. A vRack Manager (or vRack Manager Group) can deliver one type of the heterogeneous options for implementing a service.

vRack Managers cooperate to form a vRack Manager Group. vRacks Managers in the same Group are capable of self-organising to meet the objective of efficient power consumption based on their local knowledge of underlying resource utilisations. Like Puviani and Frei (2013), each vRack Manager (or vRack Manager Group) autonomously monitors itself and the environment. Similarly, vRack Managers are aware of changes in the environment including new and disappearing resources and adapts on a negotiated basis with other vRack Managers within the same vRack Manager Group to meet system objectives.

Back at the Cell-level, having received a request from the Gateway Service for resource options capable of running the Blueprint, a resource discovery service propagates this request to appropriate vRack Manager Groups. The Cell Manager has the knowledge of the resource capacity in each vRack Manager Group and therefore can immediately communicate back to the Gateway Service informing it of the various delivery options available. These options can be agreed upon, using a Resource Selection Service, or selected automatically without the need to reserve the underlying resources. Having been agreed, the resources necessary for the service implementation can be commissioned by the appropriate vRack Manager (or vRack Manager Groups). This commissioning process involves either the identification of pre-defined, statically created, coalitions or by dynamically creating new coalitions in consultation with the Coalition Formation Services.

6 CONCLUSIONS

This paper proposes a novel cloud management and delivery architecture based on the principles of self-organisation and self-management that addresses the challenges of complexity introduced by heterogeneous resources. This approach can be beneficial by shifting the deployment and optimisation effort from the consumer to the software stack running on the cloud infrastructure and reducing power consumption and improving service delivery. This paper represents work in process. Our ambition is to provide a live implementation of a self-organising and self-managing heterogeneous cloud that can be used to validate the hypothesised efficiencies. Performance metrics, such as reliability, cost and power consumption, will be used to compare the self-organising and self-managing approach to the current state of the art in cloud delivery. We present three HPC use cases where the proposed architecture can have a significant positive impact on power consumption and service delivery.

End user surveys of the HPC community have repeatedly highlighted the growing complexity of the domain and the need for “ease of everything” in the management of HPC (IDC, 2014b). This complexity is mirrored in the wider cloud computing context as more specialist computing technologies are introduced in to the marketplace and form important components in new service provision. Being powerful techniques for addressing complexity, self organisation and self-management may present a solution to this complexity by shifting the deployment and optimisation effort from the consumer to the software stack and in the process hiding this complexity from the consumer.

ACKNOWLEDGEMENTS

This work is partially funded by the European Union’s Horizon 2020 Research and Innovation Programme through the CloudLightning project (http://www.cloudlightning.eu) under Grant Agreement Number 643946.

REFERENCES

Alzamil, I., Djemame, K., Armstrong, D., and Kavanagh, R. 2015. Energy-Aware Profiling for Cloud Computing Environments. Electronic Notes in Theoretical Computer Science, 318, 91-108.

Awada, U., Li, K., and Shen, Y. 2014. Energy Consumption in Cloud Computing Data Centers. International Journal of Cloud Computing and Services Science (IJ-CLOSER), 3(3), 145-162.

Barroso, L. A., and Hölzle, U. 2007. The case for energy-proportional computing. Computer, (12), 33-37.

Beloglazov, A., Abawajy, J., and Buyya, R. 2012. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future generation computer systems, 28(5), 755-768.

Brandic, I. 2009. “Towards self-manageable cloud services” in 33rd Annual IEEE Conference on Computer Software and Applications Conference 2009 (COMSAC ’09), vol. 2, IEEE, pp. 128-133.

Crago, S. P., & Walters, J. P. 2015. Heterogeneous Cloud Computing: The Way Forward. Computer, 48(1), 59-61.

Esmaeilzadeh, H., Blem, E., Amant, R. S., Sankaralingam, K., & Burger, D. 2011. Dark silicon and the end of multicore scaling. In Computer Architecture (ISCA), 2011 38th Annual International Symposium on (pp. 365-376). IEEE.

Gell-Mann, M. 1988. Simplicity and complexity in the description of nature. Engineering and Science, 57(3), 2-9.

Herrmann, K., Mühl, G., and Geihs, K. 2005. Self management: the solution to complexity or just another problem?. Distributed Systems Online, IEEE, 6(1).

Heylighen, F. 2001. The science of self-organisation and adaptivity. The encyclopedia of life support systems, 5(3), 253-280.

Heylighen, F., & Gershenson, C. 2003. The meaning of self-organisation in computing. IEEE Intelligent Systems, 18(4).

Intersect360 Research, 2015. Top Six Predictions for HPC in 2015. Special Report. February 2015. California, USA.

IDC, 2014a. Worldwide Broader HPC 2014–2018 Forecast: Servers, Storage, Software, Middleware, and Services. IDC. Massachusetts, USA.

IDC, 2014b. Market Analysis Perspective: Worldwide HPC, 2014 — Directions, Trends, and Customer Requirements. IDC. Massachusetts, USA.

Kephart, J., Kephart, J., Chess, D., Boutilier, C., Das, R., Kephart, J. O., and Walsh, W. E. 2007. An architectural blueprint for autonomic computing. IEEE internet computing, 18(21).

Kliazovich, D., Bouvry, P., & Khan, S. U. 2013. DENS: data center energy-efficient network-aware scheduling. Cluster computing, 16(1), 65-75.

Kim. W. 2009. Cloud computing: Today and Tomorrow. Journal of Object Technology. 8(1), 65-72.

Kramer, J., and Magee, J. 2007. Self-managed systems: an architectural challenge. In Future of Software Engineering, 2007. FOSE'07 (pp. 259-268). IEEE.

Lee, Y. C., & Zomaya, A. Y. 2012. Energy efficient utilization of resources in cloud computing systems. The Journal of Supercomputing, 60(2), 268-280.

Marinescu, D. C., Paya, A., Morrison, J. P., and Healy, P. 2013. An auction-driven self-organising cloud delivery model. arXiv preprint arXiv:1312.2998.

Parashar, M., and Hariri, S. 2005. Autonomic computing: An overview. In Unconventional Programming Paradigms (pp. 257-269). Springer Berlin Heidelberg.

Puviani, M. and Frei. R. 2013. Self-Management for cloud computing, in Science and Information Conference 2013, London, UK.

Schuster, P. 2007. Nonlinear dynamics from physics to biology. Complexity, 12(4), 9-11.

Scogland, T. R., Steffen, C. P., Wilde, T., Parent, F., Coghlan, S., Bates, N., Feng, W.C. & Strohmaier, E. 2014. A power-measurement methodology for large-scale, high-performance computing. In Proceedings of the 5th ACM/SPEC international conference on Performance engineering (pp. 149-159). ACM.

Turing, A. M. 1952. The chemical basis of morphogenesis. Philosophical Transactions of the Royal Society of London B: Biological Sciences, 237(641), 37-72.

Zhang, Q., Cheng, L., and Boutaba, R. 2010. Cloud computing: state-of-the-art and research challenges. Journal of internet services and applications, 1(1), 7-18.