proceedings

MDPI

Proceedings

Cognitive Distributed Computing and its Impact on Information Technology (IT) as we know it †

Rao Mikkilineni [1,*]

[1] C3DNA; rao@c3dna.com

* Correspondence: rao@c3dna.com; Tel.: +1-408-406-7639

† Presented at the IS4SI 2017 Summit DIGITALISATION FOR A SUSTAINABLE SOCIETY, Gothenburg, Sweden, 12-16 June 2017.

Published: 9 June 2017

Abstract: As the scale of computations become large and as both people and machines demand communication, collaboration and commerce at the speed of light, rapid fluctuations in the demand for computing performance and fluctuations in available resource pools, both make it necessary to respond fast and readjust the computation structures and associated resources so as to not disrupt the user experience or the service transaction. Current Information Technologies from their memory-starved, server-centric, low-bandwidth origins from von Neumann's stored program control implementation of the Turing Machine are evolving with new architectures to meet the demand for scale and speed. In this paper we discuss the evolution of current IT to cognitive IT, where computing processes become self-aware of their resource requirements in real-time and seek to adjust them from a global knowledge of available resource pools and their provisioning processes. This is transforming the current state of the IT as we know it to a cognitive IT.

Keywords: Distributed Computing; Cloud Computing; DIME Computing Model; Information Technologies; Cognitive Computing;

1. Introduction

"Systems manage themselves according to an administrator's goals. New components integrate as effortlessly as a new cell establishes itself in the human body. These ideas are not science fiction, but elements of the grand challenge to create self-managing computing systems." This vision published in January 2003 [1] is I believe finally around the corner now in 2017.

It took almost a decade and a half since Paul Horn in 2001 coined the word autonomic computing to describe a solution to the ever-growing complexity crisis that, still today, threatens to thwart IT's future growth. In the original vision, systems manage themselves in accordance with high-level behavioral specifications from administrators — much as our autonomic nervous system automatically increases our heart and respiratory rates when we exercise. There were four properties identified to recognize autonomic computing - self-provisioning, self-optimizing, self-protecting and self-healing. Ironically, while autonomic systems have proliferated using the autonomic computing model that introduces sensors and controllers into an element and use global knowledge for managing it [2], autonomic computing has not helped to reduce the IT management complexity to date. Figure 1 shows the current state of the art where autonomic computing has been successfully applied to configure, monitor and control business processes and various devices, but managing the resources to meet the fluctuations in computing fuel requirements has fallen short.

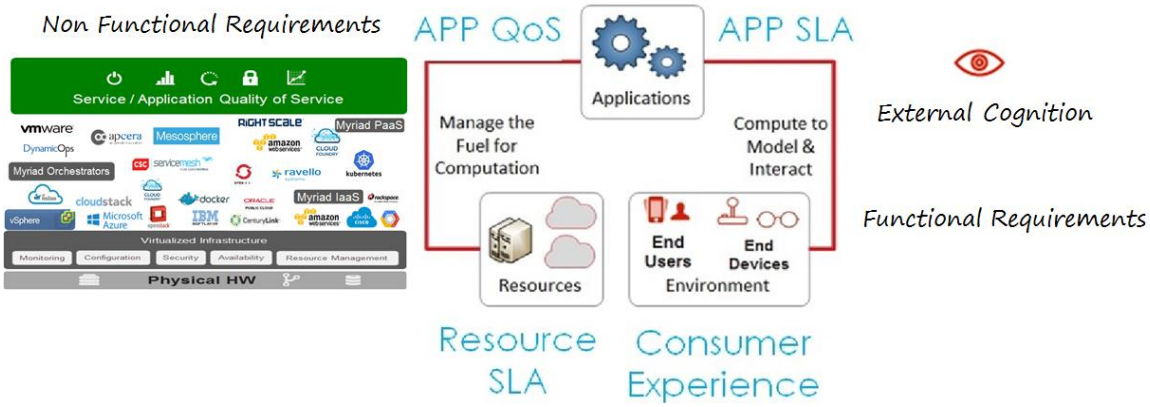

Figure 1: Current State of the Art IT with myriad infrastructure, platforms and orchestrators to support non-functional requirements

Globally distributed applications, by their very nature, consist of hardware and software components with spatial structures executing temporal evolution of computations that are designed to deliver an intent. The intent is captured in the functional requirements in the form of an algorithm that executes with the help of computing, storage and network resources. Current computing paradigm evolved from a decades-old von Neumann's stored program implementation of the Universal Turing machine in which the algorithm is executed using a read --> compute (change the state) --> write cycle, using computing resources consisting of CPU and memory. The efficiency of the algorithm execution depends both on the nature of the algorithm and the resources available to execute it. The availability, performance, security and other attributes of the application are defined in the non-functional requirements and are managed by external agents to match the application need and resource availability. If adequate resources are not available at any point of time, the application non-functional requirements suffer.

The plan for implementing functional requirements is detailed in the system design and implemented as software. The plan for implementing non-functional requirements is detailed in the system deployment and operations architecture. These requirements include availability, reliability, performance, security, scalability, compliance and efficiency at run-time. Traditional IT has developed various efficiencies in designing the functional requirements and implementing them with well-defined processes that span from requirements definition to software implementation, quality assurance and software deployment in production. On the other hand, providing the infrastructure to develop and deploy the software has evolved into a plethora of infrastructure management systems with ever increasing complexity as shown on the left hand side of the picture.

2. Need for a New Architecture for Implementing Workflow Non-functional Requirements

It is important to note that current state of the art approaches lock the application in an operating system or a container and provision and orchestrate the container or the virtual machine to address the non-functional requirements. In order to address fluctuations, the infrastructure has to be reconfigured which requires complex plethora of server, storage and network infrastructure reconfiguration in distributed environments. The resulting complexity and tool fatigue is forcing enterprises to look for solutions that decouple application from the infrastructure and infuse the ability in the application to be provisioned and orchestrated on any infrastructure independent of infrastructure management systems except for provisioning. What the enterprise line of business owners are looking for is end-to-end service transaction and business process evolution visibility and control across any infrastructure anywhere to deliver their business services with assured quality.

There is a reason for the increased complexity with scale and resiliency demand. According to Cockshott et al [3] “the key property of general-purpose computer is that they are general purpose. We can use them to deterministically model any physical system, of which they are not themselves a part, to an arbitrary degree of accuracy. Their logical limits arise when we try to get them to model a

part of the world that includes themselves." They point to a fundamental limitation of current Turing machine implementations of computations using the serial von Neumann stored program control computing model. The universal Turing machine allows a sequence of connected Turing machines synchronously to model a physical system as a description specified by a third party (the modeler). The context, constraints, connections, communication abstractions and control of various aspects (specifying the relationship between the computer acting as the observer and the computed acting as the observed) during the execution of the model cannot be also included in the same description of the model because of Gödel's theorems of incompleteness and decidability.

According to Samad and Cofer [4] there are two theoretical limitations of formal systems that may inhibit the implementation of autonomous systems. "First, we know that all digital computing machines are "Turing-equivalent." They differ in processing speeds, implementation technology, input/output media, etc., but they are all (given unlimited memory and computing time) capable of exactly the same calculations. More importantly, there are some problems that no digital computer can solve. The best known example is the halting problem; we know that it is impossible to realize a computer program that will take as input another, arbitrary, computer program and determine whether or not the program is guaranteed to always terminate. Second, by Gödel's proof, we know that in any mathematical system of at least a minimal power there are truths that cannot be proven. The fact that we humans can demonstrate the incompleteness of a mathematical system has led to the claims that Gödel's proof does not apply to humans." An important implication of Gödel's incompleteness theorem is that it is not possible to have a finite description with the description itself as the proper part. In other words, it is not possible to read yourself or process yourself as a process. In short, Gödel's theorems prohibit "self-reflection" in Turing machines. Louis Barrett highlights [5] the difference between Turing machines implemented using von Neumann architecture and biological systems. "Although the computer analogy built on von Neumann architecture has been useful in a number of ways, and there is also no doubt that work in classic artificial intelligence (or, as it is often known, Good Old Fashioned AI: GOFAI) has had its successes, these have been somewhat limited, at least from our perspective here as students of cognitive evolution." She argues that the Turing machines based on algorithmic symbolic manipulation using von Neumann architecture, gravitate toward those aspects of cognition, like natural language, formal reasoning, planning, mathematics and playing chess, in which the processing of abstract symbols in a logical fashion and leaves out other aspects of cognition that deal with producing adoptive behavior in a changeable environment. Unlike the approach where perception, cognition and action are clearly separated, she suggests that the dynamic coupling between various elements of the system, where each change in one element continually influences every other element's direction of change has to be accounted for in any computational model that includes system's sensory and motor functions along with analysis. To be fair, such couplings in the observed can be modeled and managed using a Turing machine network and the Turing network itself can be managed and controlled by another serial Turing network. What is not possible is the tight integration of the models of the observer and the observed with a description of the "self" using parallelism and signaling that are the norm and not an exception in biology.

3. New Computing Model Infusing Cognition into Computing Workloads

The DIME network architecture presented in the Turing Centenary Conference [6 - 8] infuses cognition into the Turing machine to create a managed Turing machine and introduces autonomic management of applications in managing their resources. Figure 2 shows the new approach where managed Turing machine computing model is applied to infuse cognition into application workload management across distributed infrastructures.

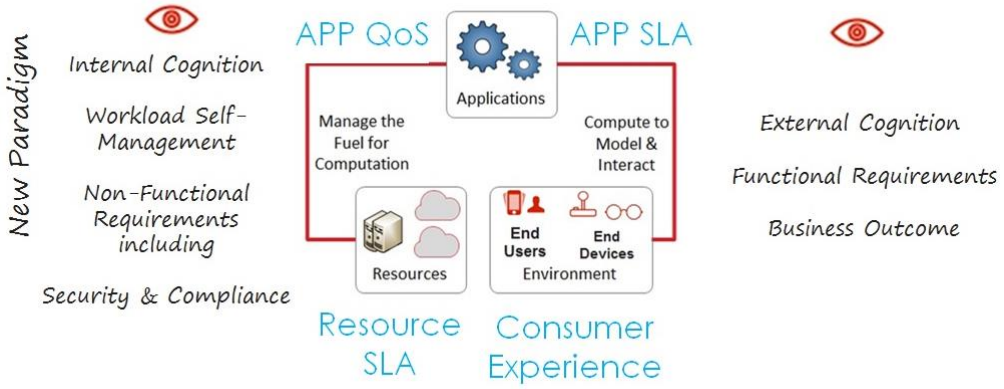

Figure 2: External cognition is complemented by internal cognition that allows both functional and non-functional requirements to be fulfilled by autonomic computing and self-management.

The new approach (Figure 2) provides the following functions to implement cognitive processes using distributed computing machines to execute an intent involving the reciprocal influence of "bottom-up" and "top-down" processes:

1. Managing the "Life" of a Cognitive Process – Policy based resource assurance for system-wide process execution: Each instance provides self-management of resources based on local policies and an ability to load and execute a process using local operating system.
2. Instantiating a distributed process flow with scale-invariant composition and management abstractions: Each instance has a unique identity upon instantiation. A regulator allows provisioning, provides heart-beat, security, performance and account management for each process it executes. A signaling scheme (providing addressing, alerting, mediation and supervision) to facilitate interaction with other instances. The signaling allows same regulation features as one instance for a group of instances (application subnetwork and network level).
3. Supporting [9] embodied, embedded, enacted and extended process flows using a DIME network with a system-wide awareness: The DIME network provides a natural framework for implementing reliable managed cognitive processes that are embodied (partly constituted by distributed and specialized structures and processes). Individual processes at the leaf level may be embedded (designed to function only in tandem with system's environment. By interacting with the environment through embedded processes, the DIME network supports cognitive processes that are enacted. The distributed nature and interaction with environment provides the execution of extended cognitive processes using the recursive composition capability of the DIME network.
4. Dynamic process Management: The run-time monitoring of both resource utilization and process execution along with signaling capabilities allows policy based intervention in each instance while computation is in progress using its Turing O-machine like behavior of each instance described in the Turing centenary conference proceedings.

4. Conclusion

In this symposium, we have presentations on the recent advances in our understanding of the new computing models, an implementation that takes the workload management beyond the current state of the art to reduce complexity and a theoretical framework behind the new approach. The new approach puts the safety and survival of application workloads and groups of applications delivering a service transaction first and provides the information relevant to sectionalize, isolate, diagnose and fix the infrastructure at leisure. Finally, I believe that Paul Horn's vision is just around the corner.

References

1. J.O. Kephart and D. Chess, "The Vision of Autonomic Computing," Computer, vol. 36, no. 1, 2003, pp. 41–50.
2. Huebscher, M. C. And mccann, J. A. "A survey of Autonomic Computing—Degrees, Models, and Applications" ACM Computing Surveys, Vol. 40, No. 3, Article 7, August 2008.

3. Cockshott P., mackenzie L. M., and Michaelson, G, (2012) Computation and its Limits, Oxford University Press, Oxford.
4. Samad, T., Cofer, T., Autonomy and Automation: Trends, Technologies, In Gani, R., Jørgensen, S. B., (Ed.) Tools in European Symposium on Computer Aided Process Engineering volume 11, Amsterdam, Netherlands: Elsevier Science B. V., (2001)
5. L. Barrett, "Beyond the Brain," Princeton University Press, Princeton, 2011.
6. Mikkilineni, R.; Comparini, A.; Morana, G. The Turing O-Machine and the DIME Network Architecture: Injecting the Architecture Resiliency into Distributed Computing, Proc. The Turing Centenary Conference, Turing-100, Alan Turing Centenary, EasyChair Proc. In Computing, epic vol. 10 (ed. A. Voronkov), Manchester, UK, June 2012, 239-251.
7. R. Mikkilineni, "Going beyond Computation and Its Limits: Injecting Cognition into Computing." Applied Mathematics 3 (2012): 1826.
8. Burgin, M., Mikkilineni, R., and Morana, G. (2016) Intelligent Organization of Semantic Networks, DIME Network Architecture and Grid Automata, International Journal of Embedded Systems (IJES), Vol. 8, No. 4.
9. Rowlands, M. (2010). The New Science of the Mind, The MIT Press: Cambridge