



The Curious Case of Distributed Systems and Continuous Computing

Samee U. Khan, *North Dakota State University*

Often times, when I teach topics related to computer and network systems, I am asked about the origins of distributed systems. It is a difficult question to answer. Also difficult are answers about the origins of cloud computing or definitions of the grid, cloud computing, or the Internet of Things. This article is certainly not an attempt (or is it?) to stir up the already hot discussion, but is rather an effort to provide some perspective to the core questions resonating with distributed systems. However, it is certainly a curious case of continuously revisiting older concepts with every progress made in distributed systems.

Distributed Computing and Distributed Systems

Off the bat, in my opinion, distributed computing and distributed systems are the same. The “computing” is stressed when the core focus is related to scheduling (of tasks, data, and communications). I personally find it a useless exercise to make a distinction. Otherwise, we should also distinguish between embedded systems and

embedded computing, and cyber-physical systems and cyber-physical computing. If you understand the difference between concurrent computing and parallel processing, you get my point. I can easily see it, from how the term distributed computing emerged.

Between the 1940s and 1980s, meaningful computing was performed on mainframes or minicomputers. Tasks running on these machines would be distributed for processing (or computing) to the various elements (not necessarily processing units) of the computer, coining the term distributed computing. The concept (and subsequently the making and eventually the popularity) of distributed systems might have originated with the invention of ARPANET and IBM 360. I choose IBM 360 (among all other mainframe computers) because it was without a doubt the very first affordable mainframe that could cater to the scales (small to large) and variety (commercial to scientific) of applications. However, we had to wait until the late 1980s when L/WANs were utilized by computers composed of microprocessors, which made distributed

systems a popular field of research and development.

In terms of the core definition of a distributed system, most of us are in agreement that “It is a collection of independent computers connected through a network, which appear to the user as a single computing platform.”¹ The keywords in the definition are “collection of independent computers” and “appearing as a single computing platform.” With this core definition as the starting point, distributed systems have evolved considerably over time. Let’s briefly look at some of these evolutions.

Cluster Computing, Grid Computing

Computer clusters (or cluster computing) are systems that are the closest to the original definition of distributed systems. They also happen to be the most widely used and are found in a variety of scales. The only departing feature from the core definition comes from the fact that computers can be tightly or loosely connected to each other. Consequently, it should be clear that the IBM Sequoia is a good example of a computer cluster with a superfast

network interface connecting super-fast computing elements—a supercomputer. It is difficult to find the exact origins of clusters, and I tend to agree with the assessment of Gregory Pfister that early clusters were not developed by companies but by researchers and computer enthusiasts who really could not get their jobs done given the computing power at the time.¹

When the Internet became mainstream, cluster computing evolved into grid computing. The core difference between cluster and grid computing arises from increased network latency. Consequently, high-performance computing (HPC) or supercomputing may not be expected out of a grid infrastructure. Moreover, grid computing promotes heterogeneity and relies on virtualization techniques. Some examples include the European Grid Infrastructure (www.egi.eu) and distributed.net. A more familiar example would be SETI@Home (evolving into BOINC) rolled out to the general public as volunteer computing; nonetheless, to the end user (SETI project) it is nothing more than grid computing. Other terminologies that emerged during the grid computing era were computational grid and data grid. There has been and continues to be a lot of debate on their uniqueness and also the difference between grid computing, cluster computing, and distributed systems. Nonetheless, they are all distributed systems.

Cloud Computing

Cloud computing, indeed, takes a gigantic step forward in resource orchestration by virtualizing compute nodes, storage, and the network, the latter being the key differentiator from grid computing. As a result, what was not possible previously, when a transition was made from cluster to grid—namely HPC—is now a possibility. There

are many definitions pertaining to cloud computing. I am not going to list or comment on any of them. In my viewpoint, they all have merit. However, one tends to think about how HPC can be possible in the cloud. The popular definitions of “cloud” revolve around the terms “anywhere and anytime,” paving the way for the notion that the computing elements are in a cloud that no one should, may, or can know. In reality, in my opinion, the cloud is nothing but a datacenter that is accessed through a set of APIs. Of course, there are many features and characteristics that are more than what an API would offer, but in a nutshell, this is what the cloud is. Then there are further advancements to the terminologies, such as cloud of clouds, clouds of things, and clouds of systems. Others have spawned the definitions of private, public, and hybrid clouds.

Cyber-Physical Systems

A good number of us are confused with what exactly “cloud” is, and while we are still resolving the issue, along comes cyber-physical systems (CPS). Now, CPS is quite interesting because it seems to be a merger of embedded systems and (wireless) sensor (and actuator) networks. I say this because an embedded system is a composition of standalone computing elements. A (wireless) sensor (and actuator) network has some sort of self-* capabilities to observe the environment and then cooperatively send the information to a central (or sink) node, which relays the information to the main computational element (could be a standalone computer or a cluster). (The * represents terminologies, such as healing, repairing, regenerating, optimizing, and stabilizing. In the 2000s the asterisk was also fondly called autonomic computing. All of these terms have their roots in the classical feedback control loop

theory.) Of course, we can call a home automation system a CPS, but I suspect the term was coined to provide solutions for large-scale systems, such as power grids, and border security. Nonetheless, if we think about what CPS is meant for and what are the building blocks, it is a distributed system.

A CPS on a gigantic scale (smart cities, intelligent transportation systems, and so on) would be the Internet of Things (IoT), which might even be an already old term as we now have the Internet of Everything (IoE). I personally do not understand the difference, but there could be one. A popular definition for the IoT/IoE is “a collection of physical entities, called ‘things,’ that are embedded with sensors, actuators, software, and network connectivity, which enables these things to collect and exchange data” (www.itu.int/en/ITU-T/gsi/iot/Pages/default.aspx). But wait a second—isn’t this what pervasive computing used to be at the turn of the century? And pervasive computing was also known as ubiquitous computing, ambient intelligence, ambient media, and “everyware.” Other names for the IoT/IoE are physical computing and haptic computing. In fact, what I should not do is say “other names.” They all focus on different aspects; however, the turnaround time in the creation of newer terms is so short that one wonders if all these are necessary and can avoid being called “buzzwords.”

Fog Computing, Edge Computing, Cloudlet

We also have some brand new major (popular) terms, as well, the first being fog computing, which is a paradigm that advocates that the storage, processing, management, communication, and control be performed at the (near) client-side computing elements to avoid utilizing datacenters and the backbone network. The second term is edge computing, which advocates that

In This Issue: IT Governance and Management

IT governance concerns the allocation of resources, implementation of technology, integration approaches, and overall “information hygiene” of an organization. It can be driven by privacy, security, and regulatory imperatives as well as competitive pressures, rising customer expectations, and marketplace deployment of new capabilities. The challenge of effective governance is that it requires buy-in from and participation of business and IT stakeholders and strong leadership in each of these camps.

In the first article in this special issue, “Metrics-Driven Information Governance,” Seth Earley posits that information governance does not receive the needed attention and resources from organizational leadership. He then goes on to examine how, by linking information governance efforts to data and process metrics frameworks, the value of these efforts can be made more evident to stakeholders to attract their buy-in. Information governance can be implemented in various ways, both general and specific, and organizations must determine whether their governance practices are having a positive impact. If they aren’t, the resources invested are being wasted, and stakeholders will likely lose interest. Implementing a metrics-driven approach to information governance can have far-reaching effects.

Next, in “A Framework for Information Security Governance and Management,” authors Marian Cary, Karen Renaud, Stephen McLaughlin, and Conor O’Brien discuss how daily changes in the information security landscape mean that organizations face unprecedented challenges in securing their information

systems, and tend to respond to problems as they occur. This, however, makes it difficult to act strategically. The authors present a capability maturity framework that supports organizations in assessing their current maturity state with respect to information security governance and management, and proactively identifying problem areas that need to be addressed. The framework addresses technical, process, and human aspects of information security and provides guidelines organizations can use to implement effective information security governance and management processes.

While these two practical approaches deal with addressing IT governance challenges today, in “Traveling Technology Governance,” Stephen Andriole examines IT governance’s history, and how it has moved from centralized control to something much more complex that significantly expands the number of players involved in the governance process. He then looks to the future, and what is likely to be a dramatic shift in the technology governance landscape.

Finally, this issue’s Spotlight department presents interviews with three past and current CIOs from the business, university, and public transportation sectors. These professionals “in the trenches” present their thoughts about several aspects of the governance process, describe their experiences and the challenges they’ve faced, and speculate on the future of technology governance.

These articles together will help you understand the intricacies of IT governance and management and (re)define and implement an organization’s governance policies and practices better.

processing (or computing) must be done as far away from the core Internet (that includes datacenters, the backbone network, and tier-1 routers) as possible. Computing needs data storage, some data is interdependent, so you need communication, and, of course, this data needs to be generated (or sensed) to be utilized for some meaningful outcome. Do fog and edge computing sound similar? I can also think of an olden days distributed system to which we can relate fog/edge computing—the content distribution network (CDN). A few other terms related to fog/edge computing are mobile edge computing and cloudlet.

All Are Distributed Systems, Anyway

Of course, there are many other distributed systems I have skipped, such as peer-to-peer computing, utility computing, on-demand computing, mobile computing, mobile cloud, crowd computing, and I am sure a dozen more that I’m missing. The fact of the matter is that all are distributed systems. They all strive to fulfill the four fundamental goals of distributed systems: resource availability, distribution transparency, openness, and scalability.² They are all making human lives better by solving important scientific problems, and


are part of our everyday lives. Internet, weather prediction, and social media are a few examples that couldn’t exist if there were no distributed systems.

The challenges faced by all of these systems have their roots embedded in the four fundamental goals of distributed systems. It should be understood that some might emphasize one feature over another, but it does not make one less of a distributed system than the other. It is good to study them—no, it is great to study them—for a reason: the continuous computing scenario that I envisage.

Continuous Computing

I think in the very near future, we will have a new paradigm: continuous computing, in which the task at hand will autonomously scale up and scale down a given computing platform. Imagine that you are watching a movie in a 3D theater and you forgot to buy popcorn; you get up, and as you walk out, you have the movie playing on your smartphone in 2D. On the way to the popcorn stand, you pass an electronic display showcasing the movies being played in the theater, and at that very instant, your favorite scene of the movie starts playing. With a right swipe, you send the movie to the electronic display. All devices, based on their capabilities, seamlessly and continuously play (compute) the movie (task). Is it edge computing or is it fog com-

puting? I do not know; I will let you define “continuous computing.”

What I do know is that it is a distributed system, and it does have a great future. 

References

1. A.S. Tanenbaum and M. van Steen, *Distributed Systems: Principles and Paradigms*, Pearson, 2nd ed., 2006.
2. G. Pfister, *In Search of Clusters*, Prentice Hall, 2nd ed., 1997.

Samee U. Khan is an associate professor of electrical and computer engineering at the North Dakota State University. His research interests include optimization, robustness, and the security of cloud, grid, cluster, and big data computing, social networks, wired and wireless networks, power

systems, smart grids, and optical networks. Khan's work has appeared in more than 300 publications and he is on the editorial boards of leading journals, including IEEE Access, IEEE Cloud Computing, IEEE Communications Surveys and Tutorials, and IT Professional. He is a fellow of the Institution of Engineering and Technology and the British Computer Society, an ACM distinguished speaker, and an IEEE distinguished lecturer. Contact him at samee.khan@ndsu.edu.

 Selected CS articles and columns are available for free at <http://ComputingNow.computer.org>.

ACM - IEEE CS ECKERT-MAUCHLY AWARD

Call for Award Nominations • Deadline: 30 March 2016 • www.computer.org/awards • awards.acm.org

ACM and the IEEE Computer Society co-sponsor the **Eckert-Mauchly Award**, which was initiated in 1979. The award is known as the **computer architecture community's most prestigious award**.

The award recognizes outstanding contributions to computer and digital systems architecture. **It comes with a certificate and a \$5,000 prize.**

The award was named for John Presper Eckert and John William Mauchly, who collaborated on the design and construction of the Electronic Numerical Integrator and Computer (ENIAC), the first large-scale electronic computing machine, which was completed in 1947.

TO BE PRESENTED AT



ISCA 2016

The 43rd International Symposium on Computer Architecture

Nomination Guidelines:

- Open to all. Anyone may nominate.
- Self-nominations are not accepted.
- This award requires 3 endorsements.

Questions? Write to IEEE Computer Society Awards Administrator at awards@computer.org or the ACM Awards Committee Liaison at acm-awards@acm.org

