

# Performance Engineering for Cloud Computing

John Murphy

Lero – The Irish Software Engineering Research Centre  
School of Computer Science and Informatics, University College Dublin, Ireland  
J.Murphy@ucd.ie

**Abstract.** Cloud computing potentially solves some of the major challenges in the engineering of large software systems. With the promise of infinite capacity coupled with the ability to scale at the same speed as the traffic changes, it may appear that performance engineering will become redundant. Organizations might believe that there is no need to plan for the future, to optimize applications, or to worry about efficient operation. This paper argues that cloud computing is an area where performance engineering must be applied and customized. It will not be possible to “cloud wash” performance engineering by just applying previous methods. Rather it is essential to both understand the differences between the cloud and previous systems, and the applicability of proposed performance engineering methods.

**Keywords:** Performance Engineering, Software Engineering, Cloud Computing

## 1 Evolution of Performance Engineering

Performance engineering is typically a collection of techniques that help manage how well a system will operate or is operating. Performance engineering covers the full life cycle of a system, from choosing a candidate list of technologies, to the high level design, detailed design, modeling, unit testing, system testing, pre-production testing, live monitoring, capacity planning, upgrading and migration of the system. One of the major areas that performance engineering has been applied to is telecommunication systems, where there has been considerable research output in the last century since Erlang’s seminal work [1]. This mathematical treatment of the topic led to the development of teletraffic theory and was published in journals (e.g. The Post Office Electrical Engineers Journal<sup>1</sup> 1908-82, Bell System Technical Journal<sup>2</sup> 1922-83, and more recently Performance Evaluation<sup>3</sup> 1981 on) and conferences (e.g. the major conference is the International Teletraffic Congress<sup>4</sup> 1955 on). In the latter half of the 20th century, as computer networks were emerging, teletraffic theory was applied to these systems. In particular queueing theory [2] emerged as a leading tool to attempt to solve some of these complex computer networks performance problems. Many

---

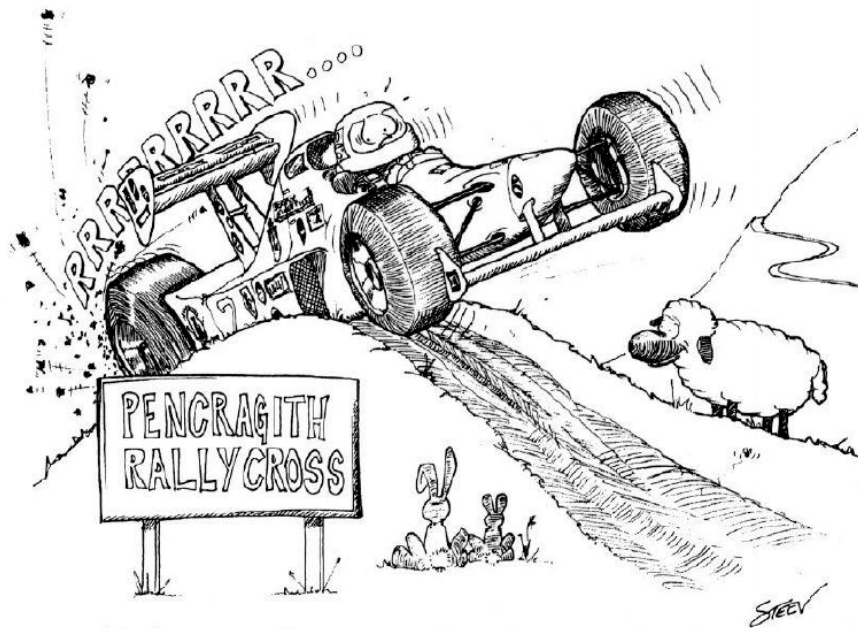
<sup>1</sup> Published by the Institution of Post Office Electrical Engineers

<sup>2</sup> <http://www.alcatel-lucent.com/bstj/>

<sup>3</sup> [http://www.elsevier.com/wps/find/journaldescription.cws\\_home/505618/description](http://www.elsevier.com/wps/find/journaldescription.cws_home/505618/description)

<sup>4</sup> <http://www.i-teletraffic.org/about-itc/>

different issues arose from this translation and attracted considerable attention, such as self-similar traffic [3], effective bandwidths [4] and statistical multiplexing [5] which could be relevant when undertaking performance evaluation of the cloud. As computer systems became more complex, the issues surrounding performance were analyzed with similar tools and techniques [6] [7] and more recently there has been considerable effort to bring some of these theories to bear on component based software [8] and enterprise systems [9].



***Bob soon discovers that having the fastest hardware doesn't guarantee success!***

**Fig. 1<sup>5</sup>.** A major challenge in applying performance engineering is to be aware of all aspects of the system and how it will be used and evaluated.

The key to the success of transforming the tools and theories from one domain to another is in understanding the fundamental differences and limitations of both domains and the relevant theories involved. Cloud computing is a relatively new technology (the term was coined most likely in 2007 [10]) but is built on top of prior research in a number of areas. Cloud computing allows many of the aspects of traditional software systems to be ignored (or abstracted) and allows the scaling and growth of a system to take place automatically. There is a real danger that many of the tried and tested - and successfully implemented - methods and practices will not be put to use in the cloud. This could be either because cloud experts are not aware of them, or more critically that they will not be translated correctly to the cloud as the special characteristics of the cloud will not be taken into account. Success will

<sup>5</sup> Andrew Lee of Whitney Associates <http://www.1202performance.com/>

involve not merely “cloud washing” performance engineering (where everything remains the same except the term “cloud” is added!) but rather to find which of the prior methods should be emphasized and possibly re-evaluated, and which of those methods might not be applicable.

## 2 Cloud Context

Cloud computing has been used to define applications delivered as services over the Internet (as well as the hardware and middleware that resides in data centers that are used to provide those services) [11]. It encompasses the concepts of Software as-a-service (SAAS), Platform (or Middleware) as-a-service (PAAS) and Infrastructure as-a-service (IAAS) which combine to make up the cloud. Public cloud refers to situations where the cloud, and in particular infrastructure as-a-service, is made available publicly to individuals and organizations and is charged using metered billing (i.e. pay for what you use). Public cloud allows different end users to share hardware resources and network infrastructure and examples include Amazon<sup>6</sup> and Rackspace<sup>7</sup>. The private cloud is targeted at large organizations, and generally provides more flexible billing models as well as the ability for these users to define secure zones within which only their company has access to the hardware and network (e.g. Rackspace private cloud<sup>8</sup>, IBM<sup>9</sup>). Hybrid clouds often refer to situations where organizations are making use of both public and private cloud for their infrastructures. The concept of cloud computing also assumes that infinite resources are theoretically available on demand, whereby a user of the cloud can scale their cloud infrastructure immediately when the need arises, i.e. during a traffic surge.

The metered billing model applied by cloud service providers has significantly changed how organizations need to plan and finance their infrastructures. Infrastructure can be provided as-a-service and no longer requires large capital investments up front. This allows a more flexible and agile approach for many organizations when planning their infrastructure requirements. In particular it has reduced the barrier for startup companies entering the market, as no major capital investment is required to launch new services. Similarly large organizations do not need to make long term bets on their infrastructure and they thus can be more flexible and reactive to unplanned changes in company strategy.

Platform as-a-service is designed to support the entire application development lifecycle (development, testing, deployment, runtime, hosting and delivery). It allows organizations to quickly deploy and deliver live, scalable applications in a fraction of the time this has taken in the past. For example following the Google app engine tutorial<sup>10</sup> will allow a developer to develop, design and deploy a live application with a public facing dynamic web site in a matter of minutes. Traditionally this process may have taken a significantly longer amount of time: organizations would have

---

<sup>6</sup> Amazon EC2 Cloud, <http://www.amazon.com/ec2>

<sup>7</sup> Rackspace Cloud, <http://www.rackspace.com/>

<sup>8</sup> Rackspace Private Cloud, [http://www.rackspace.com/managed\\_hosting/private\\_cloud/](http://www.rackspace.com/managed_hosting/private_cloud/)

<sup>9</sup> IBM Cloud Computing, <http://www.ibm.com/cloud-computing/us/en/private-cloud.html>

<sup>10</sup> Google App Engine, <http://code.google.com/appengine/>.

needed to plan and allocate physical hardware resources, domain name and Internet Protocol (IP) addresses. Capacity planning and high load scalability issues (in terms of available resources) are largely handled by the PAAS provider.

Software as-a-service allows organizations to outsource the development, management and running of services. Increasingly organizations are making use of SAAS solutions for services that are common across their industry and the development of which is not their core competency. Typical examples include CRM systems (e.g. Salesforce<sup>11</sup>), HR systems and accountancy systems (e.g. AccountsIQ<sup>12</sup>). Development environments and test and performance tools are also becoming available and popular as services. Examples include development environments (e.g. CloudBees<sup>13</sup>), cloud based monitoring systems (e.g. Cloudkick<sup>14</sup>), log management as-a-service technologies (e.g. Logentries<sup>15</sup>) and performance monitoring tools (e.g. New Relic<sup>16</sup>). The benefits of SAAS services is mainly in reduced management and running costs (compared to in-house systems), as well as the added benefits of using systems designed by specialists in the domain.

In summary, cloud computing gives the ability to design, develop and deploy large scale applications and it does so by abstracting away many of the complex issues. One major advantage is that it can scale with infinite demand for a particular application and this paradoxically presents new challenges for performance engineering.

### 3 Motivation of Performance Challenges

Performance has always been a major concern for software development and is a critical requirement for IT and software systems. With the immediate availability of theoretically infinite resources on demand, it may be reasonably asked whether performance is still a major concern as systems can "simply scale on demand when load increases". However performance, performance planning, and design are as important as ever and the availability of resources in the cloud has introduced new challenges along with benefits and opportunities. The new challenges are for all providers of as-a-service solutions, whether that is infrastructure, platform or software. The significant challenge is in providing horizontal scaling for their systems such that they can continue to grow and service new customers. Downtime for such providers is generally not acceptable as was recently witnessed with the Amazon outage in April 2011: unplanned downtime resulted in a large number of high profile organizations' systems also being down<sup>17</sup> or in fact a more recent outage which was

---

<sup>11</sup> Salesforce, CRM, <http://www.salesforce.com>

<sup>12</sup> AccountsIQ, Online accountancy platform, <http://www.accountsiq.com/>

<sup>13</sup> Cloudbees, The Cloudbees platform, <http://cloudbees.com/>

<sup>14</sup> Cloudkick, Cloud based monitoring, <http://cloudkick.com>

<sup>15</sup> Logentries, Log management as-a-service, <https://logentries.com>

<sup>16</sup> New Relic, Web app performance monitoring, <http://newrelic.com>

<sup>17</sup> Pepitone, J.: CNN, Amazon EC2 outage downs Reddit, Quora (2011) [http://money.cnn.com/2011/04/21/technology/amazon\\_server\\_outage/index.htm](http://money.cnn.com/2011/04/21/technology/amazon_server_outage/index.htm)

ongoing at the time of writing this paper<sup>18</sup>. As-a-service providers largely sell to other business users where downtime or poor performance is not acceptable as it can have a knock-on effect to their customers' business. Therefore performance and scalability is an important requirement for as-a-service providers.

Furthermore as-a-service systems tend to be larger in size than traditional in-house enterprise systems. This is due to the fact that they are often providing the in-house service on a mass scale to large numbers of enterprise customers. Thus as-a-service solutions are following new architectures and making use of new technologies to handle the massive volumes of data and user load. Examples include technologies associated with "Big Data" systems such as NOSQL data bases (e.g. Apache Cassandra<sup>19</sup>, MongoDB<sup>20</sup>, Big Table [12]) or distributed file systems (such as Apache Hadoop [13]). The scale of as-a-service systems, in particular IAAS deployments, introduces a range of new problems for the performance community. IAAS organizations for example can manage tens of thousands of servers<sup>21</sup>.

New technologies and architectures require new performance monitoring and analysis techniques, algorithms and tools, to gather the required data for performance and system test teams, such that they can effectively assess the performance of systems during development and production. Skills are mainly lacking in these areas due to the fact that the technologies are at the cutting edge. There is an onus on educators and organizations to develop appropriate training schemes in the relevant areas and technologies, to cater for these new systems.

## 4 Classical Performance Engineering

The bulk of the research in the telecommunications domain is mathematical in nature and a considerable amount of it is based on queueing theory [1], [2]. However as the evolution of the domains changed to computer communications there was increased interest in the traffic profiles and the distributions associated with them. This included a re-examination of one of the main theories that traffic would aggregate when combined, and it was shown that for some traffic the opposite occurred. This effect, known as self-similar traffic, disrupted many previous assumptions and led to considerable scrutiny [3]. This area could be of particularly interest to cloud computing as one of the basic economic assumptions is that by combining many companies usage together, savings will follow.

Another breakthrough in performance engineering emerged when users were allowed to multiplex, or combine, their traffic together in a statistical manner (statistical multiplexing). This occurred in broadband networks and the theory of "effective bandwidths" was put forward as a way to deal with this new type of traffic planning [4], [5]. This allowed some of the older theories and methods to continue

---

<sup>18</sup> Wainwright, P.: ZDNet.com, Lightning Strike Zaps EC2 Ireland. 8th August 2011, <http://www.zdnet.com/blog/saas/lightning-strike-zaps-ec2-ireland/1382>

<sup>19</sup> Apache Cassandra. <http://incubator.apache.org/cassandra/>.

<sup>20</sup> mongodb. <http://www.mongodb.org/>.

<sup>21</sup> Rich Miller, "Who Has the Most Web Servers?" Datacenterknowledge.com, May 2009, <http://www.datacenterknowledge.com/archives/2009/05/14/whos-got-the-most-web-servers/>

working, but changing the manner in which the resources were accounted for. Similarly this technique could have relevance in cloud computing, as one set of users could potentially affect other users, where they are sharing resources (in a statistical sense).

Performance engineering typically has to monitor (to collect the data), has to build models (to experiment with the system), and then has to be able to extract analysis from these models (to explore what-if type questions). The monitoring for pre-cloud enterprise systems is difficult with many layers of complexity; in cloud systems this becomes an increasingly more complex challenge. The modeling (either mathematical or simulation) has been extensively researched in software systems [6], [7], [8], [9] and many techniques can be employed to undertake to build useful models. However while there were many issues for enterprise systems, these will be exacerbated for cloud computing systems due to the scale and additional layers involved.

## **5 Cloud Specific Challenges**

There are a number of areas where results from performance engineering of software systems could benefit the area of cloud computing. Examples include SAAS performance design; autonomies; performance monitoring; resource utilization; and data analysis.

### **5.1 SAAS Performance Design**

Software as-a-service systems are centralized services typically designed to cater for large numbers of end users. Consumer services include social platforms (e.g. Facebook) or online email services (e.g. gmail). There are also increasing numbers of business services being delivered as-a-service. The nature in which these services are being implemented requires horizontal scalability and the ability to quickly scale up and down during times of different workloads [14]. Performance design for scalability and reliability is an important area for these systems. Performance design can be challenging on large scale systems with large numbers of components. These as-a-service systems will more than likely be even larger in scale than traditional in-house enterprise systems and thus performance design will be a major challenge, as it was for enterprise software [15], [16]. Furthermore, while hardware resources may be immediately available in abundance as part of using the cloud, there is an associated cost that is clearly measurable due to inefficient design. Software that inefficiently makes use of cloud resources (without due regard to the associated cost) may well result in a high financial cost. In the past the cost of running inefficient hardware was capped by the available hardware resources in-house (which was typically paid for through capital expenditure). In the cloud this is no longer the case and developers and designers are now closer to the financial costs associated with running their software. Thus responsible design of software with respect to performance is required such that efficient usage of the cloud is attained.

## 5.2 Autonomics

As systems grow in size, management from a performance perspective on a manual basis becomes more difficult. Autonomic management of systems has been a growing area of research over the past decade [17]. Automatic scaling based on alerting and user defined thresholds is something available today from as-a-service providers so that system will scale on demand. Automated automation and integration frameworks (e.g. Chef<sup>22</sup>) are allowing this to happen currently for industry. Further advances in this area will be required such that performance monitoring can integrate with these frameworks for better autonomic performance management. In particular performance monitoring methodologies, real time analytics and decision making research will be required to drive the autonomic management process.

## 5.3 Performance Monitoring

With the abundance of new technologies and middleware (Distributed file systems, NoSQL databases, Search platforms<sup>23</sup>) for large scale systems processing "Big Data" there is a need for performance monitoring and analysis techniques to be developed such that performance metrics can be obtained, analyzed and understood in the context of these new technologies. Traditional monitoring methods for enterprise systems may be applicable in certain cases, however new techniques will most likely be required specifically for these new platforms and architectures. Monitoring and management of the cloud are also starting to be delivered as-a-service<sup>14, 15, 16</sup> which means that tool providers will centrally store monitoring data from large numbers of customers systems. This in itself will provide opportunities in terms of data analytics.

## 5.4 Resource Utilization

An emerging requirement exists in the area of measuring the utilization of large cloud deployments in an automated manner such that utilization metrics can be efficiently collected and properly understood. A view of how well hardware is being utilized in the context of different workloads is currently a major challenge for cloud providers. This understanding is required to maximize the efficiency of cloud infrastructures. Research is required in the area utilization analysis in the context of different software workloads. Such analysis can be applied, for example, to maximize the system utilization, to relocate workloads, to increase energy efficiency, or indeed to reduce costs.

## 5.5 Data Analysis

While techniques and approaches for gathering monitoring data have become better appreciated through the development of performance tools for in-house enterprise

---

<sup>22</sup> Chef, Systems integration framework, <http://www.opscode.com/chef/>

<sup>23</sup> Apache Lucene, <http://lucene.apache.org/java/docs/index.html>

systems [18], [19] the analysis of the large volume of data collected has been a major challenge. Work has been performed in this area to allow for domain knowledge to be applied for enterprise systems [16]. New challenges exist for the cloud however due to the larger scale of systems and the much larger volumes of data produced by these systems. Real time analytics is a growing area and provides challenges in the analysis of upwards of millions of events per second with real time constraints. An example of such systems today can be seen with the emergence of new technologies taking on these challenges such as log management as-a-service. For example, an individual enterprise may produce terabytes of log data per month<sup>24</sup> which can equate to 100,000s of events per second<sup>25</sup>. A log management as-a-service technology handling log analysis for large numbers of enterprises must be able to manage millions of events per second, performing visualization, analysis and alerting in real time to allow for autonomic management of the system. Other similar technologies that are emerging include real time analytics for gaming platforms<sup>26</sup> and real time analytics for performance monitoring<sup>16</sup>. Further research in the area of data analytics with time constraints in the coming years will enhance the performance management of cloud based systems and this is probably going to be a rich area of research. Data mining, anomaly detection and machine learning techniques are possibly going to be applicable for as-a-service performance monitoring tools. Vendors of such technologies will be in control of large volumes of customer data compared to traditional performance monitoring tools (deployed in-house). As-a-service tools will therefore be exposed to this customer data and as such will provide opportunities for data mining techniques to be applied and patterns and trends identified that may prove beneficial to all customers.

## 6 Conclusions

Performance engineering has been applied to many domains over a long time period and with each new domain there is a translation of the most appropriate methods to successfully manage the system's performance. The traditional techniques can be reapplied, or methods and techniques that are most appropriate for the new systems may well undergo significant development. Typically there is a renewed focus on the most appropriate tools, and for the emerging cloud computing area this will most likely follow a similar pattern to previous transitions. Cloud washing will not work for performance engineering, but performance engineering will play a crucial role in the success of cloud computing.

**Acknowledgments.** The author appreciates the permission to use the cartoon (in Fig. 1) from Andrew Lee, and the many discussions about this paper (and the research) with Trevor Parsons and Liam Murphy. This research is supported, in part, by Science Foundation Ireland grant 10/CE/I1855.

---

<sup>24</sup> Oltsik, J.: "The invisible log data explosion" Cnet.com (2007)

<sup>25</sup> Log Math, <http://chuvakin.blogspot.com/2010/08/log-math.html>

<sup>26</sup> Swrve, Real time gaming analytics, <http://swrve.com/>

## References

1. Erlang, A.K.: Solution of some problems in the theory of probabilities of significance in automatic telephone exchanges. *Elektroteknikeren* Vol 13 (1917)
2. Kleinrock, L.: *Queueing Systems*. Wiley (1975)
3. Leland, W. E., Taqqu, M.S., Willinger, W., Wilson D.V.: On the self-similar nature of Ethernet traffic. In *Communications architectures, protocols and applications*, pp. 183-193 ACM, New York (1993)
4. Kelly, F.P.: Notes on effective bandwidths. *Stochastic Networks: Theory and Applications*. Oxford University Press pp. 141-168 (1996)
5. Blondia, C., Casals O.: Statistical multiplexing of VBR sources: A matrix-analytic approach. *Performance Evaluation* 16 (1-3) pp. 5-20 (1992)
6. Jain, R.: *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Wiley- Interscience, New York (1991)
7. Smith, C.: *Performance Engineering of Software Systems*. Addison-Wesley Longman Publishing, Boston (1990)
8. Franks, G., Majumdar, S., Neilson, J., Petriu, D., Rolia, J., Woodside, M.: Performance analysis of distributed server systems. In *6th International Conference on Software Quality*, pp. 15-26 (1996)
9. Balsamo, S., Di Marco, A., Inverardi, P., Simeoni, M.: Model-based performance prediction in software development: a survey. *IEEE Transactions on Software Engineering*, 30 (5) pp. 295-310 (2004)
10. Boss, G., Malladi, P., Quan, D., Legregni, L., Hall, H.: *Cloud Computing*. IBM (2007)
11. Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I, Zaharia M.: A View of Cloud Computing. *Communications of the ACM* 53 (4) pp. 50-58 (2010)
12. Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A., Burrows, M., Chandra, T., Fikes, A., Gruber, R. E.: Bigtable: A Distributed Storage System for Structured Data. In *Conference on Usenix Symposium on Operating Systems Design and Implementation* pp. 205-218 (2006)
13. White, T.: *Hadoop: The Definitive Guide*. O'Reilly Media (2010)
14. Borthakur, D. et. Al.: Apache Hadoop Goes Realtime at Facebook. In *Proceedings of the International Conference on Management of Data* (2011)
15. Tate, B., Clarke, M., Lee, B., Linskey, P.: *Bitter EJB*. Manning (2003)
16. Parsons, T. and Murphy, J.: Detecting Performance Antipatterns in Component Based Enterprise Systems, *Journal of Object Technology*, 7 (3) pp. 55-90 (2008)
17. Dobson, S., Sterritt, R., Nixon, P., Hinchey, M.: Fulfilling the Vision of Autonomic Computing. *Computer* 43 (1) pp. 35-41 (2010)
18. Parsons, T., Mos, A., Trofin, M., Gschwind, T. and Murphy, J.: Extracting Interactions in Component Based Systems. *IEEE Transactions on Software Engineering*, 34 (6) pp. 783-799 (2008)
19. Kozioloka, H.: Performance evaluation of component-based software systems: A survey. *Performance Evaluation* 67 (8) pp. 634-658 (2010)