Performance Monitoring Guidelines

Maria Carla Calzarossa
Department of Electrical, Computer and Biomedical Engineering, University of Pavia
Pavia, Italy
mcc@unipv.it

Luisa Massari
Department of Electrical, Computer and Biomedical Engineering, University of Pavia
Pavia, Italy
luisa.massari@unipv.it

Daniele Tessera
Department of Mathematics and Physics, Catholic University of Sacred Heart
Brescia, Italy
daniele.tessera@unicatt.it

ABSTRACT

Monitoring, that is, the process of collecting measurements on infrastructures and services, is an important subject of performance engineering. Although monitoring is not a new education topic, nowadays its relevance is rapidly increasing and its application is particularly demanding due to the complex distributed architectures of new and emerging technologies. As a consequence, monitoring has become a "must have" skill for students majoring in computer science and in computing-related fields. In this paper, we present a set of guidelines and recommendations to plan, design and setup sound monitoring projects. Moreover, we investigate and discuss the main challenges to be faced to build confidence in the entire monitoring process and ensure measurement quality. Finally, we describe practical applications of these concepts in teaching activities.

CCS CONCEPTS

• **General and reference** → **Measurement**; **Performance**; • **Networks** → **Network performance evaluation**; **Network monitoring**.

KEYWORDS

performance monitoring; active measurements; passive measurements; performance engineering; measurement quality; measurement platforms

ACM Reference Format:
Maria Carla Calzarossa, Luisa Massari, and Daniele Tessera. 2021. Performance Monitoring Guidelines. In *Companion of the 2021 ACM/SPEC International Conference on Performance Engineering (ICPE '21 Companion), April 19–23, 2021, Virtual Event, France.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3447545.3451195

1 INTRODUCTION

Performance engineering has been considered for long time as an insular discipline of computer science widely separated from other disciplines such as computer architecture, system organization, operating systems and software engineering [15]. More recently, as reported by Ferrari [14] and further discussed by Serazzi [29], this insularity has slowly decreased and nowadays performance engineering is permeating to some extent other disciplines both in research and education.

In fact, assessing and predicting the performance of any technological infrastructure and service require specific competencies and practice in the field of performance engineering as well as a deep knowledge of the domain under investigation. Hence, some basic skills in this field should be part of the background of students majoring in computer science and in computing-related fields.

Among these skills, monitoring, that is, the process of collecting measurements on infrastructures and services, is a "must have". In fact, monitoring enables to gain visibility into the behavior and performance of services and infrastructures, judge their health, anticipate potential outages and SLA violations, diagnose performance problems when things go wrong or proactively address and mitigate performance issues (see, e.g., [1, 24]). In addition, monitoring is at the basis of workload characterization [7] and it is one of the stages of the MAPE-K model typically applied in the framework of autonomic computing for developing adaptive systems [19].

It is also worth mentioning that monitoring makes it possible to see in action concepts taught in classes (e.g., Internet protocol stack, software systems), thus bridging the gap between theory and practice.

The complexity of infrastructures and services as well as the potential legal, privacy and ethical issues associated with monitoring open numerous challenges mainly related to the measurement process. For example, in the cloud and multi-cloud worlds, monitoring has to cope with virtualized software-based resources (e.g., virtual machines, containers) sharing physical resources and with potential contentions due to co-located applications. Similarly, monitoring mechanisms for 5G networks should satisfy the requirements introduced by the presence of virtual as well as physical resources. Moreover, any monitoring activity has to comply with current legislation and regulations. Hence, multi-disciplinary competencies are required to address these challenges and design sound monitoring projects.

Although performance monitoring is not a new education topic, nowadays it deserves particular attention because of the pervasiveness of digital technologies often characterized by large-scale distributed architectures. Moreover, monitoring activities are generally of interest for diverse stakeholders, such as users, software developers, service providers, network operators, cloud operators, who often base their decisions on the measurements being collected. It is hence necessary to build confidence in the entire monitoring process.

ICPE '21 Companion, April 19–23, 2021, Virtual Event, France

ACM ISBN 978-1-4503-8331-8/21/04…$15.00
https://doi.org/10.1145/3447545.3451195

For this purpose, in this paper, we address performance monitoring from a methodological perspective and we present a set of guidelines and recommendations for designing sound monitoring projects. We believe that these concepts together with their hands-on application are of paramount importance and should be taught by all courses dealing with technologies, like operating systems, software engineering, computer networks, computer security, cloud computing, Internet of Things, to name a few.

This paper is organized as follows. Section 2 focuses on the plan, design and setup of monitoring projects by presenting the methodological approach and the techniques at the basis of any project as well as examples of measurement platforms. Section 3 investigates and discusses the main challenges to be addressed to ensure measurement quality, while Section 4 describes practical applications of these concepts in teaching activities. Finally, concluding remarks are presented in Section 5.

2 MONITORING PROJECTS

Monitoring is a fundamental field of performance engineering, and it is also very useful in many other application domains, such as security, marketing, healthcare (see Fig. 1). In fact, measurements

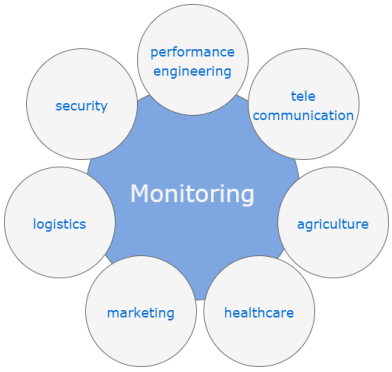

Figure 1: Examples of domains that take advantage of monitoring.

collected by monitoring capture the actual behavior of the phenomenon under investigation and provide its quantitative and qualitative descriptions. Nevertheless, planning a sound monitoring project and carrying out the corresponding experiments are rather challenging. Hence, a solid methodology has to be devised.

In what follows, we present the proposed methodological approach, the measurement process and examples of platforms.

2.1 Methodological approach

The plan, design and setup of any monitoring project have to be formulated as a process consisting of multiple interdependent steps referring to the diverse aspects involved in the project. More precisely, the methodological approach can be summarized by the following questions:

- Why: it refers to the definition of the *objectives* of the monitoring project. Possible examples:
  - status/health assessment;
  - performance assessment/diagnosis;
  - anomaly detection;
  - user profiling;
  - accounting/billing.
- Which/Who: it refers to the definition of the *target(s)* of the monitoring project. Possible examples:
  - technological infrastructures:
    - computation infrastructure (e.g., data centers, servers, virtual machines, microprocessors, GPUs, sensors, actuators);
    - storage infrastructure (e.g., hard disk drives, solid state devices);
    - software infrastructure (e.g., operating systems, DBMS, applications, services);
    - communication infrastructure (e.g., routers, firewalls, access points, base stations);
  - users;
  - workloads;
  - network traffic.
- What: it refers to definition of the *attributes*, i.e., the properties to be measured for the target. Possible examples:
  - number and status of active processes, RAM usage, CPU usage, number and types of instructions, number of cache misses, number of I/O read and write operations;
  - application processing time, number of DNS queries, number of database transactions, number and types of HTTP transactions, user navigation profile, online user interactions;
  - throughput, latency, jitter, error rate, number of dropped packets, protocol types;
  - GPS coordinates, energy consumption, battery usage, temperature.
- Where: it refers to the definition of the *vantage point(s)*, i.e., the location(s) where measurements about the target are collected. Vantage point:
  - any component of the technological infrastructure.
- How: it refers to the definition of the *measurement process*, that is, the techniques to be used to collect measurements. The choice is between two main categories, namely:
  - *active techniques* aimed at collecting measurements on the target components under a controlled artificial workload/traffic;
  - *passive techniques* aimed at collecting measurements on the target components while operating under their real workload/traffic.

Finally, to put in practice a monitoring project, that is, to carry out monitoring experiments, it is necessary to acquire or implement active or passive tools that devise the selected measurement process according to the intended usage of the measurements, i.e., online or offline. Moreover, it is necessary to define when measurements are to be collected, that is, continuously, periodically or occasionally. In general, it is recommended to consider monitoring as a systematic rather than a sporadic activity.

We outline that a prerequisite of the design of any monitoring project is a solid knowledge of the domain being investigated. In fact, incomplete or inaccurate knowledge of the domain might lead to wrong conclusions. Hence, any project requires multi-disciplinary

competencies and in particular a good understanding of the methodological approach. Moreover, specific data science skills are necessary for the analysis of the measurements collected during a monitoring experiment and the interpretation of the corresponding results.

2.2 Monitoring techniques and tools

As already pointed out, the measurement process is based on active or passive techniques used in isolation or in combination (see, e.g., [9, 13]). In detail, active tools generate artificial workload or inject artificial traffic in a controlled manner on a real infrastructure with the purpose of collecting measurements. On the contrary, passive tools collect measurements on an infrastructure while it is operating, that is, under its actual workload or traffic.

Figure 2 shows a scenario where both active and passive measurements are collected. More precisely, a vantage point, namely, end-system A, collects measurements by generating active traffic that triggers either a router or end-system B. The second vantage point is router R that passively listens to the incoming and outgoing traffic and stores measurements for the offline usage.

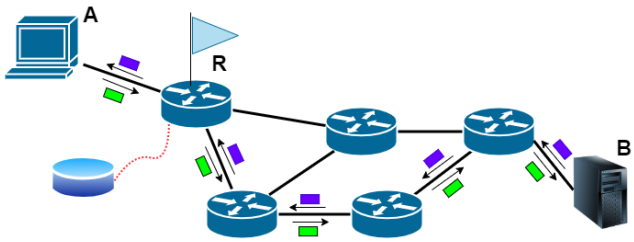

Figure 2: Scenario of active and passive measurements.

Many open-source tools implement active and passive measurement techniques. In addition, proprietary monitoring solutions are offered by technology vendors and service providers.

Let us remark that enterprises typically apply their own principles and practices to build customized monitoring systems deploying sophisticated frameworks where open-source and proprietary monitoring tools are often integrated with analysis, visualization and alerting features [10]. Hence, the various choices involved in the design and setup of monitoring projects, such as attributes to be monitored, vantage points, monitoring techniques (see Sect. 2.1) are directly driven by the functionalities offered by these frameworks.

In what follows, we summarize the main characteristics of the tools implementing active and passive techniques.

2.2.1 Active tools. Active monitoring tools include a wide collection of load and traffic generators. These tools range from simple *commands* and *utilities* that are part of the operating systems (e.g., `ping`[1], `traceroute`[2]), to a large variety of *speed testing* tools [12] to complex *benchmarking* frameworks [22]. While loading the target under investigation, active tools collect measurements, such as latency, upstream and downstream throughput, page load time, elapsed time, response time, that reflect the behavior and performance of the target.

[1]https://linux.die.net/man/8/ping
[2]https://linux.die.net/man/8/traceroute

2.2.2 Passive tools. Passive monitoring tools include network sniffers as well as a large variety of logging facilities, performance monitors and profilers. In particular, *sniffers* collect measurements at a microscopic level by eavesdropping the traffic flowing on wired or wireless networks. The `pcap` library developed within the `tcpdump` project[3] is widely used nowadays to grab packets directly from the network cards. In this framework, Wireshark[4] is the de facto standard used for network sniffing within enterprises and institutions and it is very popular for teaching computer networks. The measurements collected by sniffers refer to the attributes of the packets at each level of the protocol stack, e.g., timestamp, checksum, length, MAC and IP addresses, source and destination port numbers, flags, time to live. Depending on the objective of the monitoring project, these measurements can be used either online or offline.

Another category of passive monitoring tools is represented by *logging* facilities. This category includes many diverse facilities typically integrated into operating systems and applications/services (e.g., web, mail, access control, auditing). These facilities collect measurements of the activities and events generated by applications and services into log or trace files for their offline usage. Obviously, measurements are specific of the target being monitored. For example, web servers provide logging facilities that collect details about the web transactions being processed [5]. Similarly, media servers log information about their requests and responses [8], whereas all flavors of Linux operating system provide the syslog service that collect details about kernel and user activities [16].

Performance monitors refer to the category of passive monitoring tools aimed at measuring the status and activities of technological infrastructures and services. To monitor hardware and software events (e.g., retired instructions, processor clock cycles, floating point operations, cache references, branch misses, page faults, context-switches), these tools rely on the performance counters made available by Performance Monitoring Units or by operating systems [17]. For example, the Performance Counters for Linux[5] is a kernel-based subsystem used to collect performance data. Similarly, the Microsoft Windows Performance Counters[6] provide a high-level abstraction layer used to monitor system performance.

Event-based sampling is typically used in the framework of performance monitors. In particular, overflows of the performance counters associated with individual events trigger interrupts whose handlers are responsible for collecting measurement samples.

These samples can be used for performance profiling of applications and services [32]. For this purpose, it is necessary to apply call path profiling techniques to attribute the measurements to the calling context seen at the time of the event [18].

An alternative profiling solution is based on *performance profilers* that are part of the software development ecosystems. Measurements collected by these tools usually rely on code instrumentation, that is, monitoring probes inserted into the source or binary code of

[3]https://www.tcpdump.org
[4]https://www.wireshark.org/
[5]https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/developer_guide/perf
[6]https://docs.microsoft.com/en-us/windows/win32/perfctrs/performance-counters-portal

the application. To provide more details about the application performance, profilers usually implement a hybrid solution exploiting probes as well as performance counters.

We outline that monitoring is supported by many diverse open-source and proprietary tools (see, e.g., [11, 30] for detailed surveys in the cloud computing domain). These tools can be used for different purposes. For example, in the HPC domain tools are often used for performance profiling of parallel applications and for driving their optimization (see, e.g., [6, 21]).

Monitoring is also important for modern DevOps pipelines where automated runtime monitoring tools collect measurements related to resource usage as well as business operations and help to quickly identify bottlenecks in the process (see, e.g., [4, 25]).

2.3 Measurement platforms

Another important aspect to be considered in the design of monitoring projects deals with the availability of measurement platforms which could be exploited to collect performance measurements. These platforms often exploit probes at strategic locations within access and backbone networks and behind residential gateways.

As discussed in [2], many diverse platforms have been deployed for network diagnosis and troubleshooting. For example, the Measurement Lab M-Lab[7] is a large open source project supported by organizations, educational institutions and private companies aimed at collecting information about Internet performance. For this purpose, the platform provides measurement tools and widely distributed servers to support active measurements and facilitate exchange of large-scale measurements.

RIPE Atlas[8] is another example of popular measurement platform. This infrastructure is deployed by the RIPE Network Coordination Centre and aims at collecting and sharing measurements about Internet connectivity. More precisely, it provides the ability to perform active measurements from thousands of vantage points distributed across the Internet.

In the framework of Internet platforms, it is also worth mentioning the active and passive measurement distributed infrastructures operated by CAIDA[9] with the aim of collecting interconnectivity measurements and sharing the resulting datasets.

In summary, the proposed methodology suggests the importance of a precise definition of the monitoring objectives since the entire monitoring project will be driven by these objectives. Moreover, suitable measurement techniques and tools, used in isolation or in combination, have to the devised.

3 DISCUSSION

As already outlined, a sound monitoring project poses many diverse challenges requiring both theoretical competencies and practical skills. Hence, to ensure the quality of the data being measured, it is recommended to properly address these challenges.

In what follows, we present and discuss the main monitoring challenges as well as the main challenges related to measurement quality.

[7]https://www.measurementlab.net/
[8]https://atlas.ripe.net/
[9]https://www.caida.org/data/

3.1 Monitoring challenges

The main challenges of a monitoring experiment deal with the choices associated with the plan, design and setup of the project as well as with the use policies of technological infrastructures being monitored and possible legal constraints of the measurements to be collected.

Monitoring choices affect the resources consumed by monitoring activities (e.g., processing, memory, bandwidth). Therefore, it is recommended to lower as much as possible this overhead since it might affect the operating conditions and the performance of technological infrastructures and services. Additional challenges refer to monitoring intrusiveness and perturbations due to resources being shared between the monitoring process and the corresponding target. This phenomenon might alter the behavior of the target in an arbitrary manner and lead to unexpected consequences, including performance degradation.

To address these challenges, a monitoring project should be designed as to avoid collecting too many measurements.

Thereby, as a general rule, it is advisable to avoid aggressive monitoring activities and to apply sampling techniques whenever deemed necessary. In addition, conservative monitoring that collects few measurements should be avoided as this could make monitoring ineffective. Hence, an appropriate tradeoff between the amount of measurements to be collected and the cost associated with the measurement process has to be achieved [27].

Unexpected effects are also experienced by monitoring experiments whenever they cannot exploit a dedicated environment, instead they run simultaneously with real workload/traffic. For example, the measurements collected when running speed tests usually vary from time to time due to the background traffic flowing on the Internet. Similarly, measurements collected by performance profilers running in virtualized environments might be affected by workload/traffic being co-located [31].

As a consequence, a single monitoring experiment is often not sufficient to distill these effects. Therefore, it is recommended to repeat the monitoring experiment multiple times.

The design of a monitoring project should also take into account technical issues related to the tools to be used for collecting measurements as well as legal, privacy and ethical issues associated with measurements. For example, it might be impossible to monitor a technological component because of the lack of suitable tools. Hence, their availability and cost should be assessed during the planning of a monitoring project.

Other aspects hindering a project refer to the permissions and authorizations to be acquired before monitoring infrastructures and services and in particular for choosing vantage points. For example, in cloud environments providers have access to the lower layers of the infrastructure, whereas users can generally monitor the upper layer, that is, the virtualized resources being allocated. Similarly, network operators and Internet service providers can monitor their entire infrastructure, while users perceive the infrastructure as a sort of "black box" and they can only collect measurements at the edge of the network.

It is also worth mentioning that the monitoring activities performed by users have to comply with the acceptable use policies and restrictions imposed by providers.

An additional challenge of the measurement process is imposed by the existing data protection laws and regulations issued by various countries, such as the European General Data Protection Regulation[10]. In particular, to comply with legal obligations, passive measurements should avoid exposing personally identifiable information and obtain explicit user consent whenever required.

3.2 Measurement quality challenges

Measurement quality is of paramount important since measurements are at the basis of many decision processes. First of all, as outlined by Paxson [28], to ensure measurement quality, it is necessary to assess the accuracy and precision of the selected monitoring tools. For example, a "slow" packet sniffer might fail to keep up with the rate at which the network tap sends out the raw packet streams, thus leading to inaccurate and unreliable measurements.

Moreover, measurements could be biased. As pointed out in [26], changing insignificant and seemingly irrelevant aspects of the infrastructure used in the monitoring experiments can introduce a significant measurements bias, that might over-state effects or even lead to incorrect or misleading conclusions. For example, special attention should be placed in the selection and use of active measurement platforms to avoid measurement biases due to various effects, such as geographic coverage, demographics.

It is also very important to ensure the representativeness and validity of the measurements and avoid the risk of ignoring significant although rare phenomena. Hence, sampling techniques should be applied with particular care to cope with the so-called mice and elephants phenomenon [3]. In addition, the choice of the number of samples plays a key role. For example, components with more samples tend to have more reliable results, while components with fewer samples might include some noise in their measurements.

Another critical aspect related to measurement quality refers to the choice of the vantage points. In fact, these locations provide different perspectives that could significantly skew the interpretation of the measurements being collected. This is particularly serious for sniffers since vantage points determine the portion of monitored network. As suggested in [20], multi-vantage points might help to improve accuracy although this choice is not always feasible.

Finally, all known deficiencies associated with the measurement process and the measurements collected should be clearly identified [23]. This could be addressed by associating comprehensive meta-data with the measurements.

In summary, all these monitoring and measurement quality challenges suggest that their careful evaluation is a key aspect of the design of sound monitoring projects. In addition, the choices made at the various steps should be assessed to check whether they are actually necessary and possible.

4 MONITORING FOR EDUCATION

The proposed methodological approach provides principles and practices to design and implement effective monitoring projects. We believe that teaching these concepts is of paramount important to better engage students in the learning process and give them a sense of how notions explained in theory actually work.

[10]https://eur-lex.europa.eu/eli/reg/2016/679/oj

For example, this is the case of computer networking classes where the lectures on the Internet protocol stack are nicely complemented by hands-on lab sessions. In particular, to show how the various protocols work, it is necessary to plan, design and setup suitable monitoring projects by addressing the steps of the proposed methodological approach (see Sect. 2.1). The diagnosis of the network health could be chosen as specific objective of a project. For this purpose, the communication infrastructure and end-system B are the targets (see Fig. 2), while end-system A is the vantage point. The properties to be measured for the target refer to the network topology as well as to intermediate and end-to-end latency. The measurement process is based on the `traceroute` command used to generate sequences of multiple packets towards end-system B and collect measurements. To directly observe the traffic, this process is coupled with passive measurements collected by sniffing the network traffic using the Wireshark tool running on end-system A.

Once the project is defined, a large variety of experiments can be carried out by students from different vantage points (e.g., on campus, at home) towards different end-systems (e.g., local servers, remote servers), thus allowing them to diagnose different network infrastructures. Students will also be able to exploit the ICMP, TCP and UDP protocols and observe the details of packets being sent and received. In addition, they will understand how the IP protocol's time to live works by directly observing the packets being sent and the packets being triggered.

This practical learning style will also make students aware of the limitations and pitfalls inherent in a network monitoring project and especially in the corresponding measurement process. For example, the `traceroute` command cannot identify asymmetric paths due to network congestion or unstable paths due to temporary failures. Moreover, as discussed in Section 3.1, the experiments are not usually performed in a dedicated environment. Hence, to ensure measurement quality, students should be advised to repeat them multiple times. Finally, in case of passive measurements, it is recommended to make students aware of the potential privacy issues associated with the measurements being collected.

Similar approaches can be adopted to design monitoring projects that allow students to explore in depth and experiment the concepts taught in classes, such as operating systems, computer architecture, high performance computing. For example, to observe the behavior of the various operating system mechanisms, profiling techniques based on performance counters are particularly useful. In fact, as discussed in Sect. 2.2.2, counters monitor the hardware and software events occurred on the target system while executing a given workload, e.g., commands, applications. These measurements provide snapshots of the activities performed by the operating system, thus allowing students to investigate in detail 'where the time goes'.

Finally, students should be warned of the unintended alteration in system behavior that might arise while collecting measurements. In particular, sampling intervals should be chosen carefully to avoid sampling in lockstep with regular activities.

In summary, we believe that a methodological approach is fundamental for the design of effective monitoring projects and these concepts should be taught and applied more extensively to enhance the learning process.

5 CONCLUSIONS

Performance monitoring is a significant field of performance engineering that enables the assessment of the status and conditions of technological infrastructures and services.

Despite the availability of many proprietary and open-source monitoring tools, the plan, design and setup of sound monitoring projects are not straightforward. In this paper, we addressed performance monitoring from a methodological perspective by presenting guidelines and recommendations for effective monitoring projects and by discussing their practical applications in teaching activities. In fact, we believe it is of paramount importance to build durable competencies and confidence in the entire monitoring process.

We offered a methodology that covers the diverse aspects to be considered in a project. Particular attention was dedicated to the measurement process and to the main challenges affecting this process. We also investigated and discussed the challenges to be faced for ensuring measurement quality and for complying with the limitations imposed by current legislation and regulations.

The analysis of these issues suggested the importance of a proper choice of the vantage points to avoid the risk of monitoring under distorted perspectives, that might lead to measurement bias and wrong conclusions.

Moreover, it is suggested to avoid aggressive monitoring activities since they might introduce perturbations in the behavior of the target and lead to unexpected consequences. Similarly, conservative monitoring activities are not recommended as they might end up collecting few ineffective measurement samples. Therefore, an appropriate tradeoff between the amount of measurements to be collected and the cost associated with the measurement process should be achieved.

In conclusion, we believe that principles and practices of performance monitoring should be mastered to gain insights into the behavior and performance of technological infrastructures and services and ensure their proper functioning.

REFERENCES

[1] G. Aceto, A. Botta, W. De Donato, and A. Pescapè. 2013. Cloud monitoring: A survey. *Computer Networks* 57, 9 (2013), 2093 – 2115.

[2] V. Bajpai and J. Schonwalder. 2015. A Survey on Internet Performance Measurement Platforms and Related Standardization Efforts. *IEEE Communications Surveys Tutorials* 17, 3 (2015), 1313–1341.

[3] N. Basher, A. Mahanti, A. Mahanti, C. Williamson, and M. Arlitt. 2008. A Comparative Analysis of Web and Peer-to-Peer Traffic. In *Proceedings of the 17th International Conference on World Wide Web (WWW '08)*. Association for Computing Machinery, 287–296.

[4] C. Bezemer, S. Eismann, V. Ferme, J. Grohmann, R. Heinrich, P. Jamshidi, W. Shang, A. van Hoorn, M. Villavicencio, J. Walter, and F. Willnecker. 2019. How is Performance Addressed in DevOps?. In *Proceedings of the ACM/SPEC International Conference on Performance Engineering (ICPE '19)*. Association for Computing Machinery, 45–50.

[5] M.C. Calzarossa and L. Massari. 2011. Analysis of Web Logs: Challenges and Findings. In *Performance Evaluation of Computer and Communication Systems. Milestones and Future Challenges (Lecture Notes in Computer Science, Vol. 6821)*, K.A. Hummel, H. Hlavacs, and W. Gansterer (Eds.). Springer, 227–239.

[6] M.C. Calzarossa, L. Massari, and D. Tessera. 2003. Load Imbalance in Parallel Programs. In *Parallel Computing Technologies (Lecture Notes in Computer Science, Vol. 2763)*, V.E. Malyshkin (Ed.). Springer, 197–206.

[7] M.C. Calzarossa, L. Massari, and D. Tessera. 2016. Workload Characterization: A Survey Revisited. *ACM Comput. Surv.* 48, 3 (2016), 48.1–48.43.

[8] L. Cherkasova and M. Gupta. 2004. Analysis of enterprise media server workloads: access patterns, locality, content evolution, and rates of change. *IEEE/ACM Transactions on Networking* 12, 5 (2004), 781–794.

[9] M. Crovella and B. Krishnamurthy. 2006. *"Internet Measurement: Infrastructure, Traffic & Applications"*. Wiley.

[10] R. Ewaschuk. 2016. Monitoring Distributed Systems. In *Site reliability engineering: How Google runs production systems*, B. Beyer, C. Jones, J. Petoff, and N. R. Murphy (Eds.). O'Reilly Media.

[11] K. Fatema, V.C. Emeakaroha, P. Healy, J. Morrison, and T. Lynn. 2014. A survey of Cloud monitoring tools: Taxonomy, capabilities and objectives. *Journal of Parallel Distributed Computing* 74 (2014), 2918–2933.

[12] N. Feamster and J. Livingood. 2020. Measuring Internet Speed: Current Challenges and Future Recommendations. *Commun. ACM* 63, 12 (2020), 72–80.

[13] D.G. Feitelson. 2015. *Workload Modeling for Computer System Performance Evaluation.* Cambridge University Press.

[14] D. Ferrari. 1986. Considerations on the insularity of performance evaluation. *IEEE Transactions on Software Engineering* 12, 06 (1986), 678–683.

[15] D. Ferrari. 2003. Insularity revisited. In *Performance Evaluation – Stories and Perspectives*, G. Kotsis (Ed.). Austrian Computer Society, 1–9.

[16] R. Gerhards. 2009. *The Syslog Protocol.* RFC 5424. RFC Editor.

[17] B. Gregg. 2020. *Systems Performance: Enterprise and the Cloud.* Addison-Wesley.

[18] R.J. Hall. 1992. Call path profiling. In *Proceedings of the 14th International Conference on Software Engineering*. 296–306.

[19] B. Jacob, R. Lanyon-Hogg, D.K. Nadgir, and A.F. Yassin. 2005. *A Practical Guide to the IBM Autonomic Computing Toolkit.* IBM Redbooks.

[20] J. Jueckstock, S. Sarker, P. Snyder, P. Papadopoulos, M. Varvello, B. Livshits, and A. Kapravelos. 2019. The Blind Men and the Internet: Multi-Vantage Point Web Measurements. arXiv:1905.08767.

[21] M. Knobloch and B. Mohr. 2020. Tools for GPU computing–debugging and performance analysis of heterogenous HPC applications. *Supercomputing Frontiers and Innovations* 7, 1 (2020), 91–111.

[22] S. Kounev, K.-D. Lange, and J. von Kistowski. 2020. *Systems Benchmarking - For Scientists and Engineers.* Springer.

[23] B. Krishnamurthy, W. Willinger, P. Gill, and M. Arlitt. 2011. A Socratic method for validation of measurement-based networking research. *Computer Communications* 34, 1 (2011), 43 – 53.

[24] S. Lee, K. Levanti, and H.S. Kim. 2014. Network monitoring: Present and future. *Computer Networks* 65 (2014), 84 – 98.

[25] L. Leite, C. Rocha, F. Kon, D. Milojicic, and P. Meirelles. 2019. A Survey of DevOps Concepts and Challenges. *ACM Comput. Surv.* 52, 6 (2019), 127.1–127.35.

[26] T. Mytkowicz, A. Diwan, M. Hauswirth, and P. F. Sweeney. 2009. Producing Wrong Data without Doing Anything Obviously Wrong! *SIGARCH Comput. Archit. News* 37, 1 (2009), 265–276.

[27] M. Natu, R.K. Ghosh, R.K. Shyamsundar, and R. Ranjan. 2016. Holistic Performance Monitoring of Hybrid Clouds: Complexities and Future Directions. *IEEE Cloud Computing* 3, 1 (2016), 72–81.

[28] V. Paxson. 2004. Strategies for Sound Internet Measurement. In *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement (IMC '04)*. Association for Computing Machinery, 263–271.

[29] G. Serazzi. 2017. Learning Performance Engineering through Applications. In *Proceedings of the 8th ACM/SPEC International Conference on Performance Engineering Companion (ICPE '17 Companion)*. Association for Computing Machinery, 193–194.

[30] J. Ward and A. Barker. 2014. Observing the clouds: a survey and taxonomy of cloud monitoring. *Journal of Cloud Computing* 3, 1 (2014).

[31] R. Weingärtner, G.B. Bräscher, and C.B. Westphall. 2015. Cloud resource management: A survey on forecasting and profiling models. *Journal of Network and Computer Applications* 47 (2015), 99–106.

[32] H. Xu, Q. Wang, S. Song, L.K. John, and X. Liu. 2019. Can We Trust Profiling Results? Understanding and Fixing the Inaccuracy in Modern Profilers. In *Proceedings of the ACM International Conference on Supercomputing (ICS '19)*. 284–295.