MESR
منار الشرق للدراسات والأبحاث
Manar Elsharq for Studies & Research
Manar Alsharq Journal
for Education and Instructional Technology
Homepage:
http://meijournals.com/ojs/index.php/majeit/index
ISSN: 2790-6698
مجلة منار الشرق
للتربية و تكنولوجيا
التعليم

دراسة تحليلية عن الحوسبة بدون خوادم

An Analytical Study of Serverless Computing

محمد علي بدوي العوض كنين

Dr. Mohammed Ali Badawi Al-Awad Kunin

أستاذ مساعد – جامعة وادي النيل – كلية علوم الحاسوب وتقانة المعلومات

Assistant Professor — Nile Valley University

mohalimoh2020@gmail.com

استلام البحث: 15/06/2026 مراجعة البحث: 10/07/2026 قبول البحث:13/08/2026

الملخص:

تُعدّ الحوسبة بلا خوادم من أبرز النماذج التقنية الحديثة في الحوسبة السحابية، إذ تُمكّن المطورين من تشغيل التطبيقات دون الحاجة إلى إدارة البنية التحتية التقليدية. تُقدّم هذه الورقة دراسة تحليلية معمقة لهذا النموذج، تشمل تعريف المفهوم وتطوره التاريخي، والمنهجيات الأساسية المُستخدمة، والمنصات الرائدة مثل AWS Lambda وGoogle Cloud Functions و Azure Functions، بالإضافة إلى مقارنة شاملة مع النماذج التقليدية كالأجهزة الافتراضية والحاويات.

الكلمات المفتاحية: الحوسبة السحابية، الحوسبة بدون خوادم، خدمات FaaS، AWS Lambda، التوسع التلقائي، البنية التحتية، الحوسبة القائمة على الوظائف.

Abstract

Serverless computing is one of the most prominent recent technical paradigms in cloud computing, allowing developers to run applications without managing traditional infrastructure. This paper presents an in-depth analytical study of this model, covering the definition of the concept and its historical evolution, the core methodologies used, and the leading platforms such as AWS Lambda, Google Cloud Functions, and Azure Functions, as well as a comprehensive comparison with traditional models such as virtual machines and containers.

Keywords: Risk Perception, Algorithmic Digital Environment, Digital Framing, Algorithmic Exposure, Cognitive Processing, Perceptual Stabilization, Cognitive Reconstruction.

1. Introduction

Recent decades have witnessed rapid development in cloud computing, beginning with physical servers tied to specific locations, moving through virtual machines (VMs), then containers, and arriving at the newest and most abstract model: serverless computing.

In this model, developers are freed from the burdens of managing servers, configuring environments, and updating operating systems, and can focus entirely on writing code and business logic. The cloud

computing platform handles everything: allocating resources, scaling automatically, and billing precisely according to actual execution time.

This paper aims to provide an integrated understanding of this model by reviewing its core concepts, analyzing its technical architecture, comparing it with other alternatives, and highlighting the most significant challenges and future opportunities.

2. Theoretical Background and Historical Evolution

2.1 The Concept of Serverless Computing

Serverless computing is an execution model in which the cloud computing platform manages servers entirely, in a way that is transparent to the developer. Code runs in small, independent units called "functions," which is the origin of the term Functions as a Service (FaaS), the most common form of this model.

Despite the name, servers do exist; however, they are completely hidden from the developer. They are created and destroyed dynamically on demand, which means the developer never "sees" or manages a server directly.

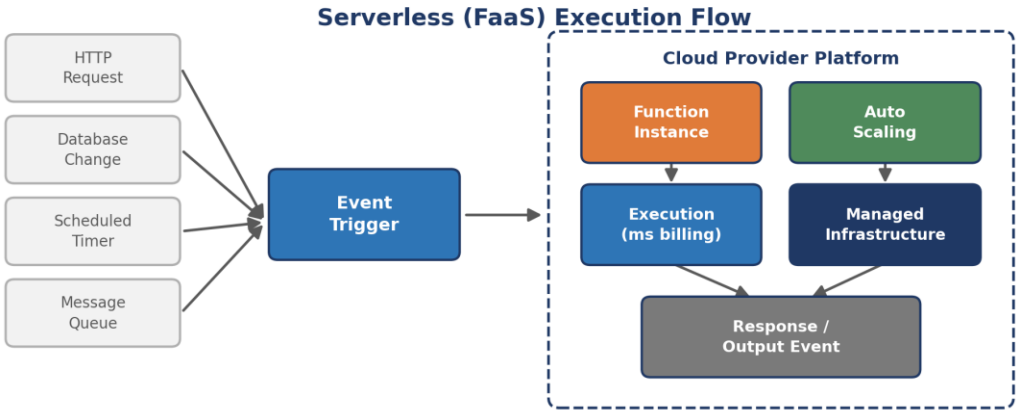

Figure 1: Serverless (FaaS) execution flow — from event trigger to managed cloud infrastructure

2.2 Historical Evolution

1. 2006: Amazon launched the EC2 service, the first successful model of virtual machines as a service.
2. 2013: Docker and container technology spread, marking a revolution in application deployment.
3. 2014: Amazon launched AWS Lambda, the first major commercial FaaS platform — a true turning point in the history of cloud computing.

4. 2016: Google and Microsoft entered the market with Google Cloud Functions and Azure Functions, respectively.
5. 2019 – present: The model matured and its use expanded to cover wide-ranging sectors, while its monitoring and debugging tools advanced considerably.

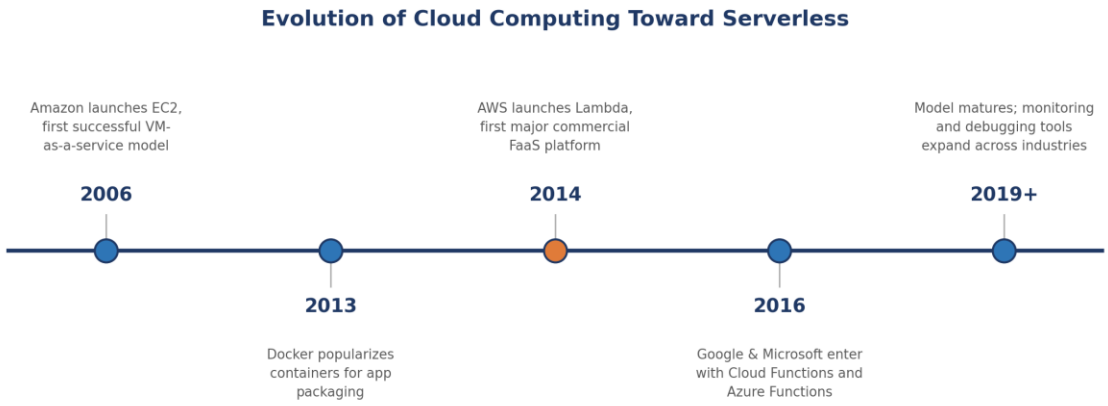

Figure 2: Evolution of cloud computing toward the serverless model

3. Methodology

3.1 Research Approach

This research relies on a descriptive, analytical, and comparative approach, combining a review of specialized scientific literature with an analysis of official technical documentation issued by major service providers (AWS, Google Cloud, Microsoft Azure), alongside the study of documented application cases in real production environments.

3.2 Data Sources

The study's data sources included research papers published in peer-reviewed scientific conferences and journals (IEEE, ACM), the official technical documentation of major cloud computing platforms, and field studies published on the performance of production applications built on this model.

3.3 Comparison Criteria

To compare the different models, the following criteria were adopted: startup time (cold/warm start), the pricing model, automatic scaling mechanisms, the allowed execution duration, and the administrative complexity required from the development team.

4. Results and Analysis

4.1 Leading Platforms and Their Comparison

The following is an objective comparison between the different models, based on the criteria defined above:

Criterion	Virtual Machines (VMs)	Containers	Serverless
Startup time	Minutes	Seconds	**Milliseconds ✓**
Pricing model	Fixed	Fixed + variable	**Based on actual usage ✓**
Automatic scaling	Manual	Semi-automatic	**Fully automatic ✓**
Execution duration	Long ✓	Long ✓	Limited (minutes)
Administrative complexity	High	Medium	**Very low ✓**

Table 1: Comparison between virtual machines, containers, and the serverless model

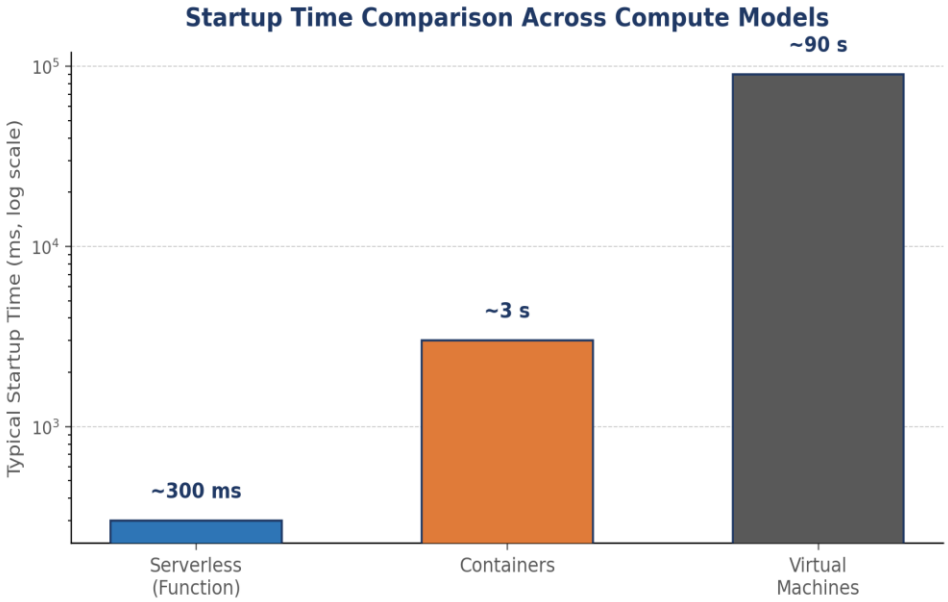

Figure 3: Comparison of typical startup times across compute models (logarithmic scale)

4.2 Advantages of Serverless Computing

The study revealed a number of fundamental advantages of this model, most notably:

- Precise billing: Costs are calculated accurately based on actual execution time (in milliseconds), significantly reducing overall cost for applications with intermittent workloads.
- Instant automatic scaling: The platform automatically handles millions of concurrent requests without any intervention from the development team.
- No server management burden: Developers can fully focus on business logic, away from infrastructure concerns and updates.

- Faster time to market: These features accelerate the development cycle, lower operating costs, and enable small teams to build highly scalable services efficiently.

4.3 Challenges and Limitations

On the other hand, the study identified fundamental technical challenges that must be considered when deciding on adoption:

- Cold start problem: When a function is not used for a period, reinitializing it takes a noticeable amount of time, ranging between 100 and 1000 milliseconds, which negatively affects user experience in time-sensitive applications.
- Vendor lock-in: Each platform relies on its own proprietary APIs and tools, making migration or multi-cloud operation more difficult.
- Limited execution duration: Major platforms impose a maximum runtime ceiling for a single function (15 minutes on AWS Lambda), restricting their use for prolonged computational tasks.
- Monitoring and debugging challenges: The distributed nature and ephemeral states of functions make diagnosing errors and tracking performance more complex compared to traditional models.

4.4 Key Use Cases

The study indicated that this model excels in specific scenarios, most notably: event-driven processing such as image analysis and format conversion, building APIs with variable load, running scheduled tasks and ETL operations, Internet of Things (IoT) applications, and real-time data stream processing.

5. Discussion and Future Trends

The technical landscape is moving toward more mature and flexible uses of this model. There are notable research and practical trends worth monitoring and following:

First — Edge Serverless

Integrating the serverless model with edge computing allows functions to run closer to the end user, radically reducing response time and partially addressing the cold start problem.

Second — Standardization and Addressing Lock-in

Open-source initiatives such as Knative and OpenFaaS seek to build a unified abstraction layer that allows functions to be migrated between different platforms with ease, which will reduce dependence on a single vendor.

Third — Artificial Intelligence Integration

Major platforms have begun integrating artificial intelligence capabilities with serverless computing, enabling the creation of highly flexible, intelligent processing pipelines that operate on demand.

Fourth — Improved Monitoring Tools

The field is witnessing growth in specialized monitoring and debugging tools (such as Lumigo and Dashbird), addressing one of the most prominent weaknesses of the current model.

6. Conclusion and Recommendations

This study concludes that serverless cloud computing represents a genuine qualitative shift in the world of modern application development, offering a unique blend of operational efficiency, cost reduction, and superior scalability. However, it is not a magic solution for every problem; it is best suited to event-driven applications of an intermittent nature, while it may not be the optimal choice for continuous computational operations or applications that are highly sensitive to response time.

Based on the research findings, the researcher recommends the following:

1. Adopting a hybrid approach that combines serverless and containers according to the nature of each component in the application.
2. Investing in training development teams on specialized monitoring tools for this environment.
3. Designing functions to be small in size and with a single responsibility, to achieve maximum benefit from the model.
4. Carefully studying costs before full adoption, especially for applications with fixed and high workloads.

References

1. Jonas, E., et al. (2019). Cloud Programming Simplified: A Berkeley View on Serverless Computing. UC Berkeley Technical Report.
2. AWS Documentation. (2024). AWS Lambda Developer Guide. Amazon Web Services, Inc.
3. Google Cloud. (2024). Cloud Functions Documentation. Google LLC.
4. Microsoft Azure. (2024). Azure Functions Overview. Microsoft Corporation.
5. Lloyd, W., et al. (2018). Serverless Computing: An Investigation of Factors Influencing Microservice Performance. IEEE International Conference on Cloud Engineering.
6. Baldini, I., et al. (2017). Serverless Computing: Current Trends and Open Problems. Research Advances in Cloud Computing, Springer.
7. Knative Community. (2024). Knative: Kubernetes-based platform for serverless workloads. knative.dev.

8. Castro, P., et al. (2019). The Rise of Serverless Computing. Communications of the ACM, 62(12), 44–54.