Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Building a Career in Cloud Engineering—From Automation to Architecture

Mahesh Babu Jalukuri
Inceed, USA

ARTICLE INFO

Received: 13 March 2026

Accepted: 31 March 2026

ABSTRACT

Cloud engineering is now a foundational discipline that influences the development of the design, provision, and governance of digital infrastructure in modern organizations. The intersection of scripting and automation, infrastructure-as-code habits, containerization, and observability has offered a technical career ladder that carves a technical career ladder and requires specific tool expertise, as well as a tactical body of thought. Since automation-first scripting frameworks and version-controlled configuration management are initial steps, cloud engineering knowledge accumulates gradually with declarative infrastructure provisioning, container orchestration, CI/CD pipeline architecture, and platform engineering. These technical fields are accompanied by an increased emphasis of the profession on communication clarity, a culture of blameless incidents, organized certification strategies, and mentorship abilities. It is not the knowledge of any single tool that makes a career in cloud engineering; it is the skill to move through a larger ecosystem of mutually reinforcing fields that can be integrated and optimized into coherent and scalable platforms to benefit both the engineering team and organizational goals.

Keywords: Cloud-Native Infrastructure, Infrastructure As Code, Container Orchestration, Devops Automation, Platform Engineering

1. Introduction

Cloud engineering is already regarded as one of the most impactful disciplines of modern technology practice. The architectural change from on-premise infrastructure that was manually delivered to on-premise computing infrastructures to elastic, service-oriented computing models has transformed the manner in which companies construct, provide and manage their digital spaces. Cloud computing, which is generally the provision of hardware, systems and software applications to the internet, provides companies with an effective infrastructure platform at a fraction of the cost compared to a traditional system configuration [1]. It is not technologic change as such, but rather organizational, strategic and profoundly human change in need of professionals who can find their way between the technical and the adaptive business thought.

The magnitude of cloud implementation in industries all over the world is a pointer of the extent to which the field has taken center stage in the operations of the contemporary world. Increased spending by end-users on the public cloud services in the world is projected to increase by 23.1, with worldwide expenditures of USD 332.3 billion in 2021 projected to become USD 397.5 billion in 2022 alone, which contributes to the urgency of organizations investing capital and talent in the cloud-native infrastructure [2]. The resulting rapid growth in demand of resilient and scalable platforms, enhanced by the shift to distributed work models, has resulted in a vast and very competitive labor market for cloud engineering specialists.

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Fig 1: Global Public Cloud End-User Spending [1, 2]

1.1 From Reactive Operations to Proactive Architecture

Historically, infrastructure management was based on the principle of the reactionary model; teams reacted to failures, capacity constraints, and security incidents in retrospect. This pose is reversed in cloud engineering. Based on the principles of declarative configuration, automation-first workflows, and infrastructure-as-code, proactive architecture enables engineering teams to encode resilience, consistency, and scalability into their deployment pipelines long before production workloads actually execute. The factual data of enterprise IaaS adoption also confirms this change: organizations that have shifted their IT infrastructure to the cloud arrangements already report the quantifiable benefits in deployment agility and contracting with vendors, and platform flexibility results directly from the quality of IT capability built around the infrastructure transition [1].

1.2 The Workforce Dimension

To gain insight into the career environment of cloud engineering, one needs to conduct a systematic study of the demand in the industry. A strict semantic content analysis performed among 10,161 cloud computing job postings, through probabilistic topic-modeling processes, found 22 very different competency areas and 46 in-demand skill sets that run the spectrum of the cloud engineering career [2]. The five most common competency areas of the top five, which included, Engineering, Development, Security, Architecture, and Management, were over fifty percent of all cloud job postings, which proves that the domain is not monolithic and has not been a narrowly technical domain. The three most frequently sought-after capabilities in terms of competency categories are communication skills, DevOps tooling, and software development, which demonstrate the interdisciplinary nature of successful cloud practice.

1.3 Scope of This Article

This paper will map out a clear upward trajectory in the level of scripting and automation of infrastructure provisioning, containerization, observability and platform architecture. The sections are based on the

Research Article

previous ones, which represents the compounding and iterative nature of cloud engineering knowledge. The framework discussed is based on the recent peer-reviewed studies and operational trends, which provides the budding and practicing engineer with a sound guideline to career advancement in this fast-emerging field of study.

2. Foundational Tools: Scripting, Automation, and Version Control

Any cloud engineering career cannot be held together without a strict grounding in scripting and automation. The only one that distinguishes between manual infrastructure administrators and cloud-native engineers is the skill to convert operational work into repeatable and testable code. The complexity and distributed nature of deployments in the modern cloud environment mean that more traditional manual processes are always inadequate to meet the requirements of security teams, such as monitoring and defending a highly elastic attack surface that does not have clear boundaries as in traditional IT contexts [3]. Infrastructure Scripting frameworks like PowerShell enable engineers to create deterministic workflows that can communicate with cloud APIs, storage systems, identity management layers, and network configuration interfaces in a single, versionable interface and directly respond to the consistency requirements that a distributed cloud infrastructure places.

An essential aspect of scripting skill is its awareness regarding how it is linked to security and incident response. The current AI-based incident response architectures, which are now the workhorse of enterprise cloud security initiatives, are based on the presence of well-formatted automation pipelines in order to perform containment, remediation, and orchestration activities [3]. Manual investigation processes and systems that rely on fixed rules and rules cannot scale to the amount of events that occur in clouds and the volume, and this constraint has direct relevance to engineers at all levels of the career ladder. The Capital One incident of 2019, where the personal data of more than 100 million clients was compromised because of a misconfigured part of the infrastructure, showed exactly what can go wrong when automation and active verification of the configuration are not incorporated into the operational base of an organization [3]. To aspiring engineers, this points out that scripting is not a productivity aid but a line of operational defense.

2.1 Infrastructure as Code and Version-Controlled Configuration

Infrastructure as Code (IaC) is the methodological development of scripting for full infrastructure lifecycle management. Instead of responding to ad hoc commands, IaC represents the intended state of entire environments in declarative configuration files, which can be reviewed, tested, and versioned. A study exploring the contemporary cloud-native application risk profile has determined that attacker-initiated data attacks make up approximately 55 percent of risk events, with the rest of the 45 percent due to misconfiguration or internal error, a result that makes version-controlled, peer-reviewed IaC one of the most effective risk-reduction techniques in the engineering team [4]. By ensuring that infrastructure definitions are handled as Git-driven workflows with obligatory review gates, the risk of misconfiguration rolling into production is significantly decreased.

2.2 Codebase Composition and the Automation-First Mindset

The insight about what cloud-native codebases consist of further supports the significance of scripted governance. Custom application code is often only 10-20 percent of a total codebase; open-source libraries and third-party dependencies make up 80-90 percent of the codebase [4]. This distribution implies that automated scanning tools, which are provided as part of the CI/CD pipelines as part of scripted workflows, need to scan the vast majority of deployed code during each commit cycle in order to determine known vulnerabilities. Having become fluent with this automation layer early in their careers, such engineers create the scanning, patching, and compliance-checking pipelines that guard whole platforms, a career base that naturally evolves into platform engineering and platform architecture.

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Focus Area	Key Point	Why It Matters
Scripting Foundation	Ops must become code	Avoids manual mistakes
Cloud Complexity	Distributed and elastic infra	Manual work won't scale
Security Operations	Automation supports defense	Handles large attack surface
Incident Response	AI-driven IR needs pipelines	Faster containment/remediation
Capital One (2019)	Misconfiguration exposed 100M+ data	Shows real-world impact
IaC + Git	Infra as versioned configs	Reduces misconfiguration risk
CI/CD Automation	Scan dependencies each commit	Blocks known vulnerabilities

Table 1: Foundational Tools for Cloud Engineering: Scripting, Automation, and Version Control [3, 4]

3. Infrastructure as Code and Containerization

Infrastructure as Code (IaC) is the convergence of software engineering and infrastructure management, both methodologically. Instead of having to configure servers, networks, and storage resources using a graphical interface or manually running commands, IaC stores the desired state of a complete environment in declarative configuration files that can be versioned, reviewed and automatically applied. The cloud's infrastructure landscapes continue to evolve continuously, offering organizations with a multifaceted selection problem with a variety of providers with heterogeneous service configurations a challenge, one distinctive to declarative, query-driven solutions [5]. IaC software like Terraform allows engineers to describe compute, storage, and network resources using a provider-agnostic abstraction layer, which automatically resolves the naming inconsistencies and configuration heterogeneities that various cloud providers impose to describe similar types of services.

The issue of declarative infrastructure selection is not that simple. In choosing across providers, a provider recommender system should be able to take into account the entire cross-join of combinations of compute, storage and network services; the computational cost is unbounded with respect to the number of regions, price levels, and types of service a provider provides [5]. This mathematical fact is what directly guides the design of modular, provider-scoped configuration files by IaC practitioners who explicitly set boundaries on criteria instead of using monolithic templates. An internalization of this complexity by engineers allows them to construct systematic, criterion-based provisioning workflows that can be computed at scale and audited operationally.

3.1 Declarative Configuration and Service Selection

Declarative IaC has the strength of dividing the intent and execution. The selection criteria are defined by engineers, including RAM capacity, storage size, volume of data transfer, and cost specifications, and the tooling is used to select the service configurations that fulfill the selection criteria using the available providers [5]. This model can be compared to the SQL based declarative paradigm in which the engineer specifies what is required and the engine is left to figure out how to get it instead of writing step-by-step procedural instructions. To cloud engineers, it implies that structured Terraform setups are effectively runnable infrastructure contracts, which can be reviewed by technical and non-technical stakeholders.

3.2 Containerization and the Software Container Lifecycle

Containerization has become one of the core fields of study in cloud engineering, allowing portable, isolated packages of applications that do not tie the operating environment to underlying hardware. Systematic review of 145 primary studies with research on software container maintenance established that container scheduling is the most studied area, with 42 percent of the studies describing container scheduling techniques, 24 percent describing performance evaluation, 22 percent describing security measures, and 12 percent describing container image detection [6]. Such allocation of the research focus

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

indicates the operational focus of the production cloud environments, where efficient scheduling of resources and security posture directly impact platform reliability and cost efficiency.

3.3 Container Orchestration and Scheduling

Orchestration systems automate container deployment, scaling and lifecycle management of containers in clusters of host machines that are distributed. Regarding the algorithm of scheduling research, it is found that in 33 percent of the studies related to scheduling containers, the authors implement a heuristic algorithm, 32 percent of them a metaheuristic algorithm, 18 percent of them a mathematical programming algorithm, and 17 percent of them a machine learning algorithm, resource utilization is found to be the most frequently prioritized performance goal in all four categories [6]. Fluent cloud engineers are able to make sensible choices concerning the configuration of orchestration platforms, the design of their own custom schedulers, and the policy on resource allocation that optimizes performance and operational cost on production workloads.

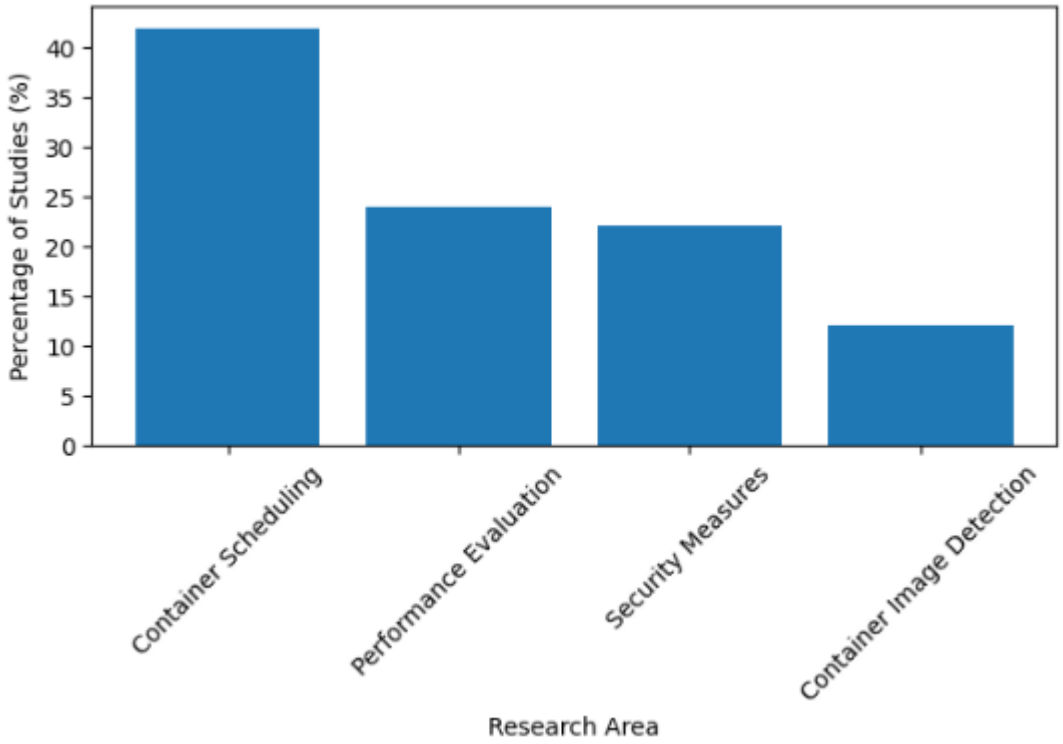

Fig 2: Distribution of Research Focus in Software Container Maintenance (145 Studies) [5, 6]

4. Observability, CI/CD Workflows, and Platform Engineering

Observability is the art of knowing the inner workings of a distributed system based on its outer outputs. With a cloud where applications are implemented in dozens of microservices that are implemented in dynamically provided infrastructure using ephemeral containers, a traditional monitoring strategy based on configured thresholds and known failure modes is fundamentally inadequate. Full observability as a shift of basic monitoring requires a paradigm shift in organizational thinking about system reliability that includes distributed tracing, structured logging, and extensive metrics collection and must work together to provide actionable information [7]. An observable maturity model in stages (with Basic, Advanced and Predictive stages) offers engineering teams a logical roadmap on how to transition away from reactive dashboards and siloed telemetry to AI/ML-based anomaly detection and automated remediation that directly reduces the mean time to detection and the mean time to recovery across production pipelines [7].

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

4.1 Observability Tooling and Scalability Challenges

Popular use of surveillance instruments like Prometheus and Grafana has been evident among cloud environments, but organizations have always faced difficulties in effective scaling of these applications. Studies of Prometheus implementations underscore that, although this tool is efficient at metrics collection and time-series data management, there are issues with flexibility to high-cardinality metrics, efficiency of long-term storage, and query performance as data volumes grow [7]. Such gaps are usually only revealed during incidents, which makes the team enter into the reactive troubleshooting process and makes the recovery process take a lot longer, something that the Predictive maturity stage of observability directly solves by having automated recovery features that allow self-healing systems to identify, diagnose and fix errors on their own.

4.2 CI/CD Pipeline Architecture and DevOps Dimensions

Continuous Integration pipelines and Continuous Delivery pipelines automate the process of deploying the code to a running system. Empirical studies on the effect of the dimensions of DevOps on the results of pipelines revealed that 44 percent of the polled companies already reported their intentions to adopt the DevOps practices, which is an indication of the industry-wide awareness of the pipeline automation as a competitive condition [8]. Nevertheless, the study has made a counterintuitive conclusion that automation is not the factor that creates the most significant effect on continuous delivery and deployment successes. The visibility practices have the greatest explanatory power when the three DevOps dimensions, namely automation, visibility and control, are assessed in unison, with the magnitude of visibility affecting continuous deployment at 49.8% [8].

4.3 Visibility, Control, and Pipeline Maturity

The priority of visibility over automation directly reflects on the design and priorities of pipeline instrumentation by cloud engineers. Organizations that automated and standardized software packaging and deployment into a controlled pipeline environment reported a 50% decrease in potential customer-impacting problems after a release [8]. This is not just because of the automation of build and test processes but because of the systematic visibility that cross-functional teams have due to dashboards, release metrics, and deployment status reporting. Moreover, of the software practitioners surveyed, two out of five claimed that the simplest automation functionality to adopt is software builds, then packaging and deployment, indicating that pipeline visibility infrastructure ought to be created alongside, rather than following, foundational automation work [8]. Platform engineering is the natural intersection of these two, which formalizes the observability, control and automation practices that facilitate reliable, self-service deployment by distributed engineering organizations.

Section / Focus	Key Concept	Tools/Evidence	Outcome / Benefit
Observability (Core)	Understand system internals from outputs	Tracing, logs, metrics	Faster detection & recovery
Observability Maturity	Basic → Advanced → Predictive	AI/ML anomaly detection	Self-healing + reduced MTTR
Tooling Challenges	Scaling metrics is difficult	Prometheus, Grafana	Cardinality and storage issues
CI/CD Pipeline Architecture	CI/CD automates delivery	DevOps adoption: 44%	Industry-wide standard
Visibility + Control	Visibility has highest impact	Visibility effect: 49.8%	Fewer release failures (~50%)

Table 2: Observability, CI/CD Workflows, and Platform Engineering Summary [7, 8]

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

5. Soft Skills, Certifications, and Long-Term Career Growth

The only way a cloud engineering career can be maintained over its lifespan is through technical proficiency. The field remains in continuous growth with a rate that requires emerging experts not only to excel in the use of tools but also to be organizationally flexible in order to move the technical complexity into strategic choices. The size of cloud computing market spending is expected to increase to 1.3 trillion in 2025 [9], a trend that indicates not only the magnitude of infrastructure spending but also the demand for engineers who will lead, communicate, and mentor in more complex platform organizations. The engineers who go beyond the individual contribution level at this scale are the ones that use technical depth in conjunction with the communication and documentation disciplines that render complex systems understandable to cross-functional stakeholders.

5.1 Communication, Documentation, and Organizational Influence

Documentation is not an administrative burden in cloud engineering, and it is one of the main artifacts of engineering. Architecture decision logs, runbook writing, and incident retrospective writing all encapsulate institutional knowledge, on which teams rely to become dependable and integrate new engineers successfully. The importance of organized experimentation and constant feedback on cloud organizations is cultural, as the volume of work produced by technically mature platforms that run more than 700K experiments in production that led to more than 4K changes in search products reflects a culture of validation that cannot work without engineering teams capable of documenting, analyzing, and communicating the results [9]. These documentation and communication practices are built early enough to make engineers recognized as organizational knowledge nodes, professionals whose sphere of influence covers far beyond the bounds of any given technical project.

5.2 Incident Management and Blameless Culture

Failure of cloud infrastructure is not an exception; it is a fact of operations in a distributed and dynamic environment. Post-incident learning determines the differences between the organizations that become stronger and those that deteriorate because of the constant downfalls. Security-related incidents are widespread in the cloud-native environment, and 84 percent of organizations report security incidents directly associated with cloud-native infrastructure [10]. Blameless postmortem culture that investigates systemic contributory factors as opposed to personal mistakes yields more valuable analyses and more lasting remedial measures. Once engineers acquire the facilitation and communication skills to make a postmortem effective (getting lessons applicable across the company and converting results into platform improvements), they become key contributors to the development of engineering culture, whether they are formally in the role or not.

5.3 Certifications and Structured Career Development

Professional certifications are validated structured programs that are fixed to the competency areas that are most desired in cloud engineering positions. The cloud-native security certifications have witnessed an 83 percent annual growth on the number of candidates registered, as the industry acknowledges that specialized credentials fill the skills gap in containerized and distributed environments that are critical [10]. In addition to platform-specific credentials, Zero Trust architecture and identity management certifications indicate that the individual has acquired control over security frameworks that can be shown to minimize the exposure of organizations to risk. Organizations implementing the principles of Zero Trust observe successful breaches that are 57 times less than those that have traditional perimeter-based models [10], which explains why engineers with proven security architecture expertise enjoy high career differentiation. Platform credentials are supplemented by vendor-neutral certifications in IaC, container orchestration and cloud networking providing transferable and framework-agnostic knowledge that remains useful across provider ecosystems.

5.4 Mentorship and Knowledge Multiplication

Professional outputs that a top cloud engineer leaves behind are often not technical deliverables but the skills that the junior members develop. Engineering capacity increases at the team level when structured mentorship is used, where learning goals, frequent feedback loops, and gradually increasing independent ownership of projects are in place. The majority of the cloud-competent engineers develop their skill sets via on-the-job training and participation in professional communities [9], and practiced mentorship and exchange of peer knowledge become the key processes through which cloud engineering expertise spreads across the industry. The engineers who invest in developing people create robust teams that can maintain the quality of the platform without a heroic effort on behalf of an individual, the organizational base that scalable, long-lived infrastructure programs are built upon.

Conclusion

The careers of cloud engineering are developed gradually one technology supporting and increasing the previous one. The foundation of operationalization is scripting and automation; IaC and containerization expand the operationalization base to reproducible, scalable infrastructure; observability and CI/CD pipelines change the deployment into a standardized capability, not a risk event. The difference between engineers who grow in engineering, architecture and positions of influence is not the extent of tools that they have mastered but the combination of technical perfection and communication training, security sophistication and knowledge transfer commitment. Professional learning is not an episodic event but an inherent professional practice, which is maintained by certifications, community participation and mentoring. With the cloud infrastructure currently growing to the edge computing, AI-enhanced operations, and multi-clouds, the most suitable group of people to lead it are those who view the profession as a system aimed at resiliency, scaled, and continuously enhanced through feedback and iteration.

References

[1] Rajesri Govindaraju et al., "IT Infrastructure Transformation and its Impact on IT Capabilities in the Cloud Computing Context," International Journal on Electrical Engineering and Informatics, 2018. [Online]. Available: https://www.ijeei.org/docs-1950592575b714f21ed906.pdf

[2] Ozcan Ozyurt et al., "Career in Cloud Computing: Exploratory Analysis of In-Demand Competency Areas and Skill Sets," MDPI, 2022. [Online]. Available: https://doi.org/10.3390/app12199787

[3] Devashish Ghanshyambhai Patel and Sudha Rani Pujari, "AI-driven incident response in cloud security," International Journal of Science and Research Archive, 2025. [Online]. Available: https://journalijsra.com/sites/default/files/fulltext_pdf/IJSRA-2025-1742.pdf

[4] Theodoros Theodoropoulos et al., "Security in cloud-native services: A survey," MDPI, 2023. [Online]. Available: https://doi.org/10.3390/jcp3040034

[5] Miranda Zhang et al., "A Declarative Recommender System for Cloud Infrastructure Services Selection," Arxiv. [Online]. Available: https://arxiv.org/pdf/1210.2047

[6] Ruchika Malhotra et al., "A systematic literature review on maintenance of software containers," ACM, 2024. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/3645092

[7] Hardik R Patel, "A maturity model for observability in big data pipelines: From reactive monitoring to predictive resilience," Journal of Computer Science and Technology Studies, 2025. [Online]. Available: https://al-kindipublishers.org/index.php/jcsts/article/view/11343/10099

[8] Dr. Hanadi Salameh, "The Impact of DevOps Automation, Controls, and Visibility Practices on Software Continuous Deployment and Delivery," IMECONF, 2019. [Online]. Available: https://www.dpublication.com/wp-content/uploads/2019/09/IME-F793.pdf

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

[9] Brian S. Mitchell, "Cloud native software engineering," arXiv:2307.01045v1, 2023. [Online]. Available: https://arxiv.org/pdf/2307.01045

[10] Arun Ganapathi, "Building a career in cloud-native IAM: Essential skills and strategies," Cognizance J. Multidiscip. Stud., 2025. [Online]. Available: https://cognizancejournal.com/vol5issue9/V5I917.pdf