Systems, Roles, and Development Methodologies

May 7, 2025

1 Cloud Development Life Cycle

1.1 Cloud Computing

Cloud computing[1] is currently the fastest-expanding computing paradigm. Often visualized as a cloud in network diagrams, this term highlights how users tap into web, database, and application services remotely, sidestepping investments in on-site infrastructure. Figure 1 illustrates the bidirectional exchanges between client machines and cloud-hosted services. Clients typically connect through browsers such as Google Chrome or Mozilla Firefox, with the back-end servers maintaining applications and data for them.

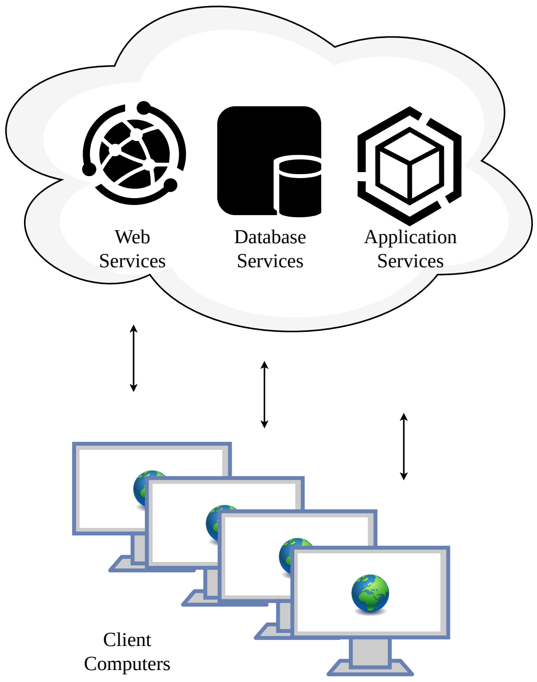

Figure 1: Cloud computing provides a range of services.

Major hardware, software, and consultancy firms, such as Cisco, Dell, IBM, HP, Microsoft, SAP, Amazon Web Services (AWS), and others, are heavily investing in cloud initiatives built on "virtualized resources." The hallmark of these solutions is their elasticity: they can automatically adjust resource usage to match evolving business needs, scaling up or down based on demand. Software as a Service (SaaS) exemplifies one of the primary delivery models within cloud computing.
End users can focus on their core tasks without needing in-depth knowledge of the underlying infrastructure that powers the cloud. Organizations often avoid hiring extra IT staff to manually

[1]A model where organizations or individual users leverage web-hosted services, ranging from databases to applications, via the internet (the "cloud") without needing to purchase or maintain their own hardware, software, or development tools; businesses simply use a web browser to interact with these services, while servers handle software execution and data storage.

scale resources when contractual or budgetary constraints fluctuate, since the cloud provider handles these adjustments seamlessly.
By eliminating hefty upfront investments in IT infrastructure, cloud computing empowers smaller businesses with variable budgets to accelerate their processing capabilities. Likewise, large enterprises can redirect capital toward strategic initiatives instead of data center maintenance. Shared cloud resources mean many organizations co-utilize the same web services, splitting costs, boosting peak-load handling, and preventing underutilization.
For disaster recovery, firms leverage the cloud's geographically distributed sites, which can mitigate downtime; a cloud disruption may still occur, but risk is distributed across numerous servers. Security-focused cloud offerings promise enhanced protection, but consolidating critical data in one place can raise concerns about loss of control. Yet the mobility of web-based access lets users log in from any device or location, liberated from reliance on a single workstation or interface.
Pure-play cloud vendors, companies founded without physical storefronts, provide browser-based applications like Google Apps for spreadsheets and calendars, Akamai's content delivery services, AWS's infrastructure, and Salesforce.com's CRM (now also available on the iPhone). These providers emphasize lower costs and greater flexibility for end users.
Analysts argue that established firms adopt cloud solutions as a strategy to reinforce their core offerings, integrating trends like SaaS, service-oriented architecture (SOA), virtualization, and open-source tools into their internet-based portfolios of software, services, and compute power.

1.1.1 Cloud Computing Trade-Offs

Privacy and security worries rank among the top obstacles to corporate cloud adoption, but these issues can often be mitigated through technical safeguards (such as encryption or access controls) and organizational measures like specialized training or elevating cloud security to a strategic objective. In some cases, local IT teams resist relinquishing control over corporate data and processes, fueling apprehensions not grounded in formal risk assessments but rather by a preference for maintaining existing workflows.
As with any technology decision, companies must weigh pros and cons when selecting a cloud provider, often with analysts playing a key role in this evaluation by comparing performance, cost, and security features to find the best fit for a project.
A key choice is between a public cloud[2] and a private cloud[3]. Alternatively, a hybrid cloud model can combine both, keeping high-value data under private control while placing less sensitive workloads in a public environment.
Analysts add value by balancing data security risks against the importance of the information. Evaluating the robustness of a provider's disaster recovery strategy is especially crucial for any mission-critical datasets under consideration for cloud storage.

1.1.2 Factors in Choosing a Cloud Provider

Here are some questions you will want to ask when evaluating cloud service providers for your information systems clients:

1. If one tenant in the provider's environment is compromised, what impact might that breach have on the provider's other customers?
2. What assurances exist that data are thoroughly erased from the provider's infrastructure when resources are retired?
3. Does the provider employ open, standardized data formats and architectures that facilitate migration if the client needs to change vendors for cost or security reasons?

[2]A cloud environment managed by an external provider; companies often opt for this when strategic data control is not paramount

[3]A cloud infrastructure built and managed within an organization's own data centers, typically storing critically sensitive or proprietary information

4. Does the provider rely heavily on proprietary APIs or custom configurations that could lock the client into the platform?
5. What controls prevent provider personnel or co-tenants from accessing the client's data?

By reviewing each provider's responses, you and your client can determine which service best aligns with the project's requirements. Additionally, regulatory factors (such as data privacy laws in the European Union versus the United States) may influence vendor selection. In such cases, verifying the geographic location of data centers and ensuring compliance with relevant regulations is essential.

1.1.3 ERP Systems and the Cloud

Researchers suggest that cloud-based ERP solutions can address many traditional implementation challenges at a lower cost[4]. Since the vendor manages software maintenance and upgrades, organizations avoid large capital outlays, and transitioning between modules can be more straightforward than with on-premises ERP.
One example is Workday (www.workday.com), founded by former PeopleSoft executives, which offers subscription-based SaaS for mid- to large-sized organizations. Its platform claims to reduce data storage expenses, produce a smaller environmental footprint, simplify upgrades, and resolve issues common to legacy ERP, such as unpredictable total costs, fragmented IT solutions, and inconsistent service levels.
Another leader in cloud ERP is NetSuite (www.netsuite.com), acquired by Oracle in 2016. It delivers accounting, real-time inventory, CRM, e-commerce, and global business management for SMEs. NetSuite distinguishes itself by highlighting lower IT expenses, real-time insight, and anywhere-access to critical business data.
Like other cloud projects, deploying ERP in the cloud demands assessing strategic value, security risks, user acceptance, and integration feasibility. As an analyst, you can leverage data-gathering and analysis tools to guide organizations toward informed choices on cloud ERP adoption.

1.2 Implementing CDLC

Per National Institute of Standards and Technology (NIST), the cloud paradigm comprises five core characteristics, three service models, and four deployment approaches [1].
The five essential characteristics include:

1. On-demand self service
2. Broad network access
3. Resource pooling
4. Rapid elasticity
5. Measured service

The three service models include:

1. Software as a Service (SaaS)
2. Platform as a Service (PaaS)
3. Infrastructure as a Service (IaaS)

The four deployment models include:

1. Private cloud
2. Community cloud

[4]ERP systems align, simplify, and unify processes across diverse corporate operations.

3. Public cloud
4. Hybrid cloud

Comparing the traditional SDLC with the Cloud Development Life Cycle (CDLC) reveals key differences, particularly concerning the timing of hardware decisions. In the CDLC's predictive model, these choices occur earlier, often steering projects toward cloud adoption sooner.
The CDLC consists of six stages: data gathering, development, launch, ongoing operations, optimization, and improvement. Unlike the SDLC, the CDLC enables cost-effective testing, such as penetration testing ("pentesting"), during development using cloud-based servers. In contrast, SDLC testing typically follows hardware and software deployment, which can be more expensive and time-consuming.
Furthermore, the CDLC fosters closer collaboration with a cloud-based development ecosystem, granting developers easier data access and freeing time for varied system testing. Since the cloud provider maintains the infrastructure, deployment is faster than traditional SDLC approaches. Integration with platforms such as AWS empowers organizations to scale quickly and collaborate seamlessly with external software vendors.
There are five strategic considerations in implementing cloud-based systems (adapted from [2], [3]):

1. Adopt cloud computing primarily when cost savings on physical data centers, upkeep, and support staff drive the project.
2. Pilot with a small-scale deployment, then expand incrementally.
3. Engage multiple cloud vendors to prevent vendor lock-in.
4. Track cloud-driven changes with the same rigor as any IT transformation, ensuring transparent visibility into applications and data across platforms.
5. Formulate an integration plan to leverage hybrid environments, harnessing new synergies, partnerships, and accelerated development paths offered by the cloud.

One should choose The Cloud Development Life Cycle (CDLC) Approach if:

- the primary goal of the project is to cut costs typically spent on on-site data centers
- the effort can begin with a small deployment and expand over time
- it's possible to engage multiple cloud providers
- cloud-driven changes can be tracked to maintain visibility of applications and data across diverse systems
- an integration plan can be devised to leverage emerging synergies

2 Developing Open Source Software

Open source software (OSS) provides a contrast to traditional proprietary development, where source code is kept hidden from users. In OSS projects, anyone (both end users and developers) can inspect, distribute, and alter the code. Community guidelines require that any changes be shared with the entire project team and that all license terms are strictly followed.
Some view OSS not merely as a development method but as a broader philosophy aimed at societal improvement. However, OSS is not a single unified movement; its diversity makes it difficult to generalize about the types of contributors or the motivations behind different projects.
By definition, open source code is openly available for inspection, modification, and redistribution according to each project's license. Development is collaborative: participants contribute with the expectation that everyone benefits. All OSS is governed by licenses that must pass a review process before adoption. For example, Android's operating system is open source. Widely used OSS includes the Apache HTTP Server, the Mozilla Firefox browser, and Linux, a Unix-like free operating system. Emerging open source domains—such as blockchain, cybersecurity, the Internet of Things (IoT), cloud computing, and big data analytics—are explored in later chapters.

2.1 The Open Source Movement

Researchers have classified open source communities[5] into four archetypes: ad hoc, standardized, organized, and commercial, evaluated along six dimensions: structure, environment, objectives, practices, user base, and licensing. Some scholars argue OSS stands at a crossroads, with commercial and community-driven groups needing to recognize both shared goals and potential points of tension.
Prominent nonprofit foundations supporting OSS include The Apache Foundation (established by contributors to the Apache HTTP Server)and The Linux Foundation. The Linux Foundation is dedicated to advancing the technical quality of the Linux kernel[6] and promoting education and collaborative development. Hosting over 500 projects, the foundation assists with project formation, governance frameworks, and community collaboration.
Platforms like GitHub (acquired by Microsoft in 2018) provide repositories where developers store, version, and collaborate on OSS. GitHub also functions as a social network for developers to showcase work, solicit contributions, and even commercialize extensions. Built on Git, the leading open source version control system, it meticulously tracks file histories and code changes.

2.2 The Third Design Space

Collaboration between corporate developers and open source communities represents a relatively new paradigm. Historically, companies guarded proprietary code to maintain competitive advantage, while OSS communities focused on collective improvement. Researchers identified this intersection as the "Third Design Space," a conceptual realm where corporate and communal contributors jointly forge new software and development methodologies [4]. In this evolving environment, participants blend organizational and community resources to produce innovations unattainable within purely commercial or purely volunteer settings.

2.3 Benefits of Corporate Participation in Open Source Software Development

Participation in OSS offers multiple advantages for both you and your employer. Through game-theoretic analysis of interviews with corporate contributors, our team uncovered six rational and six emotional motivations driving for-profit firms to engage in OSS [5]. These motivations are summarized in Table 1.

Table 1: Six rational and six emotional reasons why for-profit corporations choose to develop open source software.

Rational	**Emotional**
Saving money	Accepting responsibility
Performing less maintenance	Improving shared software
Contributing within limits	Gaining community influence
Reducing long-term costs	Relinquishing the gatekeeper role
Increasing marketing benefits	Enhancing developers' skills
Making the first move	Extending project lifespans

[5]Groups of volunteer programmers and supporting corporations collaborating to create, maintain, and enhance open source software.

[6]A monolithic kernel that mediates between user-mode applications (e.g., CPU, memory, and interprocess communication) and hardware; one of the largest open source projects globally.

2.4 Licensing and Compliance

License selection is fundamental in OSS development and distribution. Common licenses include the Apache License 2.0, GNU General Public License (GPL), Common Development and Distribution License (CDDL), and Mozilla Public License 2.0. As licenses emerge from community consensus, different projects adopt terms that best suit their goals. Users and organizations must identify the applicable license for reporting and ensure compliance.
Tools and standards exist to streamline license management. The SPDX Working Group (Software Package Data Exchange) develops a shared specification for reporting license information. FOSSology, a Linux Foundation project, provides an open source toolkit for scanning license, copyright, and export control data. According to its documentation, "you can run license, copyright and export control scans from the command line." FOSSology also offers a compliance workflow, enabling automatic SPDX or README generation with copyright details. It can scan entire distributions, rescan only modified files, and greatly reduce manual effort on large projects.
On an individual level, contributing to OSS bolsters your résumé. Recent reports indicate that over 90 percent of hiring managers face shortages of open source expertise, and a similar proportion are willing to support employee certification in relevant skills. You can join OSS initiatives by registering on GitHub, enrolling in projects via the Linux Foundation, or seeing if your employer has an internal OSS group.

2.5 The Role of the Analyst in Open Source Software

As an analyst, your organization may task you with participating in OSS communities. Over 75 percent of organizations surveyed in 2022 reported increased reliance on open source software compared to the previous year [6]. This growing adoption has led to a shortage of skilled OSS developers. The Linux kernel community exemplifies a large, mostly virtual ecosystem with contributors offering varying levels of engagement. Other notable OSS projects include Mozilla Firefox, Android, and numerous Apache initiatives. Even NASA maintains an active OSS community [7].
Companies may involve analysts in OSS for several reasons: to explore potential software benefits or respond to a bandwagon effect as competitors engage in OSS. Another driver is the concept of "responsive design." *Responsive design* refers to leveraging your OSS contributions within your organization to integrate community-developed features into proprietary products, processes, and IT artifacts. Through this approach, the resulting solutions embody both communal and corporate innovations.

References

[1] P. Mell and T. Grance, “The NIST Definition of Cloud Computing,” National Institute of Standards and Technology, Gaithersburg, MD, Tech. Rep. Special Publication 800-145, 2011. DOI: 10.6028/NIST.SP.800-145. [Online]. Available: https://csrc.nist.gov/pubs/sp/800/145/final (cit. on p. 3).

[2] D. Elman, “The Unexpected Challenges of Cloud Transformation,” Nucleus Research, Boston, MA, Tech. Rep., 2020. [Online]. Available: https://www.govinfosecurity.com/whitepapers/nucleus-research-unexpected-challenges-cloud-transformation-w-6885 (cit. on p. 4).

[3] R. Hill. “12 Steps to a Successful Cloud Implementation.” Published by Route Fifty. (2019), [Online]. Available: https://www.route-fifty.com/infrastructure/2019/05/12-steps-to-a-successful-cloud-implementation/297931/ (cit. on p. 4).

[4] K. E. Kendall, J. E. Kendall, M. Germonprez, and L. Mathiassen, “The Third Design Space: A postcolonial perspective on corporate engagement with open source software communities,” *Information Systems Journal*, vol. 30, no. 2, pp. 369–402, 2020. DOI: 10.1111/isj.12270. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/isj.12270 (cit. on p. 5).

[5] J. E. Kendall, K. E. Kendall, and M. Germonprez, “Game theory and open source contribution: Rationale behind corporate participation in open source software development,” *Journal of Organizational Computing and Electronic Commerce*, vol. 26, no. 4, pp. 323–343, 2016. DOI: 10.1080/10919392.2016.1228360. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/10919392.2016.1228360 (cit. on p. 5).

[6] The Linux Foundation Research Team, “The 10th Annual Open Source Jobs Report,” The Linux Foundation Research Team, Tech. Rep. 10, 2022. DOI: 10.70828/RZRE1873. [Online]. Available: https://www.linuxfoundation.org/research/the-10th-annual-open-source-jobs-report (cit. on p. 6).

[7] NASA, *NASA Open Source Software*, https://code.nasa.gov/, A catalog of open source software released by NASA, 2012 (cit. on p. 6).