Scale-up or scale-out: Is there one solution that fits all problem sizes?

Marcel Stolin
marcelstolin@gmail.com

Abstract

The scalability of a computing environment impacts the performance and the ability to serve business needs. Two approaches to scale a computing environment are scale-up and scale-out. The objective of this essay is to describe the scale-up and scale-out approaches while mentioning their limitations. In addition, this essay describes why a computing environment should be designed to scale-out in the first place to safe costs.

1 Introduction

With the increase of interest in Artificial Intelligence and the growing complexity of modern computing architectures, more powerful resources are needed to adapt to business needs. If an IT environment reaches the limit of its computational power and can't serve incoming requests efficiently, it has reached the limit of its scalability [4]. The limit of scalability can be extended by providing more resources. Although, increasing the resources also increases the cost of running the environment. The two primary approaches to increase the capacity of a computing environment are vertical scaling-up and horizontal scaling-out [2]. Both approaches are not exclusive and a computing environment can be designed to scale vertically, horizontally or both [4]. Although both strategies can be used to increase the capacity of a computing environment, each strategy follows a different approach.

The contribution of this essay is as follows. First, the approach of each scaling strategy is introduced. Second, based on a use case, the limitations of both scaling approaches are described, while mentioning the best approach to solve the problem. Third, a summary of this essay is given with a logical conclusion.

2 Background

This section introduces the concept of the scaling-up and scaling-out approaches.

2.1 Scale-up

Scale-up refers to improving the hardware of individual nodes [4]. By adding more powerful resources to a node, a node can take more throughput and perform more specialized tasks [2]. An increase of the computing capacity of a node can include adding memory or adding more CPU cores [4]. Due to the low complexity of scaling-up, it is the most applied scaling approach [4].

2.2 Scale-out

Scaling-out is achieved by adding more nodes to the computing environment [4]. Adding more nodes, increases the overall computing capacity of the environment and in addition, the workload can be distributed across all nodes to handle and increasing number of requests [4, 2]. To scale efficiently, nodes should be homogeneous. Homogeneous nodes are able to perform the same work and will respond as other nodes [2].

A typical example for using a scale-out strategy is, to create new instances of a database to serve read-operations for rapidly increasing requests.

3 Limitations of scaling

This section describes the limitations of both scaling approaches.

Overall, scalability is limited by the resource capacity it is able to provide to each active user [4]. If a computing environment reaches the limitations of its hardware capacity, options are to scale-up by adding more powerful resources or to scale-out to distribute the work across multiple nodes. If a computing environment is not designed to scale-out, the only option remaining is to add more powerful hardware [1].

3.1 Limitations of scaling-up

Scaling-up by adding more powerful hardware to the computing environment, is limited by maximum hardware capacity and causes downtimes due to IT resource replacements [5]. An economical threshold can be reached, where buying more powerful hardware is not affordable. Another limitation can be that no more powerful hardware is available by the provider [1, 4].

3.2 Limitations of scaling-out

If the environment is designed to scale-out, buying more powerful resources is not needed in the first place. The computing capacity can be increased by adding more nodes which are instantly available [1].

Designing an environment to scale-out, shifts the focus from infrastructure to development. The goal of scaling-out is to increase the computing capacity by combining the computing power of several nodes, which makes a scale-out design

more complex compared to a scale-up architecture [4]. The computing capacity of a scale-out architecture is limited by the computing power of each additional node. Therefore, scaling-out is more efficient with homogeneous nodes, where each node adds the same amount of computing power to the system. With non-homogeneous nodes, the complexity of capacity planning and distributing work across nodes grows [4].

4 Recommended scaling-strategy

In this section, a reasonable recommendation of how to scale a computing environment is given.

Although, scaling-up is the simpler scaling approach due to its low complexity, it has significant drawbacks compared to scaling-out. In general, computing environments should be designed to scale-out in the first place. If a computing environment is designed to scale-up, it is possible that no more powerful hardware is available. In addition, more powerful can become unaffordable. While the cost of buying more powerful hardware grows proportionally, the cost of investing engineering power to design an environment to scale-out grows linear [1]. With a scaling-out architecture, the environment can adapt to business needs without interruptions. If the limitation of a scaling-out architecture is reached, scaling-up is still a remaining option.

5 Conclusion

The ability to scale is a critical characteristic of modern and complex computing environments. With the ability to scale, a computing environment is able to adapt to business needs. Scaling-up and scaling-out are two scaling approaches to increase the computing capacity of a computing environment. Scaling-up is the simpler approach, but it is limited and comes with significant disadvantages according to cost and flexibility. On the other hand, scaling-out adds more complexity to the environment but is able to scale the environment without increasing the cost proportionally. A modern computing environment should be designed to scale-out in the first place to handle performance spikes and keeping the cost low.

References

[1] Abbott Martin L. and Fisher Michael T. *Scalability Rules: 50 Principles for Scaling Web Sites*. Addison-Wesley Professional, May 2011.

[2] Abbott Martin L. and Fisher Michael T. *The Art of Scalability: Scalable Web Architecture, Processes, and Organizations for the Modern Enterprise, Second Edition*. Addison-Wesley Professional, June 2015.

[3] F. Rossi, M. Nardelli, and V. Cardellini. Horizontal and vertical scaling of container-based applications using reinforcement learning. In *2019 IEEE 12th International Conference on Cloud Computing (CLOUD)*, pages 329–338, 2019.

[4] Bill Wilder. *Cloud Architecture Patterns*. O'Reilly Media, Inc., September 2012.

[5] Mahmood Zaigham, Puttini Ricardo, and Erl Thomas. *Cloud Computing: Concepts, Technology & Architecture*. Pearson, May 2013.