AN INTRODUCTION TO SCALABILITY

Most businesses aim to grow over time, but sometimes it's easier said than done. Without scalability, it can be difficult or even impossible to grow beyond a certain point. This challenge applies both to businesses in general and to the software on which many of them depend.

Scalability prepares software systems to increase their capacity when demand calls for it. Prioritizing scalability in software can be the difference between a successful company and a burning-out startup. If you anticipate growth down the road, then making software scalable from the very beginning allows it to grow alongside the company, ultimately leading to cheaper, easier, and more efficient changes over time.

Types of Scalability

Economic vs Computational

In an economic sense, scalability implies that a company's business model can handle increasing sales as long as it receives increasing resources. The foundations of some business models prevent this growth, even when the extra resources are available. After all, if your restaurant only has five tables, then buying more hamburger patties and buns won't increase sales when the tables are full. Adding a drive through or expanding the building might remedy the situation, but it would constitute a more expensive redesign rather than inherent scalability planned from the start.

The same ideas apply to the scalability of computational systems and IT departments. All

computers, applications, programs, and algorithms can have the capacity to scale to support increasing users or workloads, but only if they are set up to do so. Databases need to be able to increase work if they receive more hardware resources like servers and processors. Algorithms can benefit from architectural innovations to help manage multi-server configurations while cloud computing has wide applications for enabling the scalability of many parts of the IT infrastructure. But if these possibilities are not part of the original design of the system, then it is not considered scalable, and it would require redesign to permit scalability.

Out vs Up

Much like water pouring into an ice cube tray, additional workloads in a scaled-out system will spread among more and more pieces of hardware. Each ice cube slot, or piece of hardware, then only needs to manage a small portion—and it's easy to add more ice cube trays. In contrast, a scaled-up system focuses on using stronger and more advanced hardware, akin to addressing the pouring of water with a taller and taller cup.

Scaling out, or horizontally, is the most common enterprise scalability solution because the costs are comparatively low for practically unlimited growth. Adding pieces of hardware that are similar to or compatible with what the system already uses makes the process cheap and easy. However, the speed of systems relying on horizontal scalability might be lacking because it is limited by the server's communication speed.

When focusing more on performance, vertical scalability offers better results. Rather than adding more hardware, this method adds network interfaces, memory, or storage to handle growing workloads. The smaller quantity of pieces in the system also decreases vulnerability while maintaining higher speeds. The main problem that will arise when scaling up is an eventual reduction in the extent to which hardware can be upgraded before it's no longer worth the money. Imagine commissioning someone to build a ten-foot-tall water cup!

Benefits of Prioritizing Scalability

Even if your company isn't immediately growing, preparing for future scaling can provide both short- and long-term benefits. At the start, it can lower costs by allowing you to buy only what you need for your current workload, knowing that the system will be capable of growing in the future if necessary. In comparison, a system unable to scale might require the purchase of a large bundle of software tools upfront in an attempt to prepare for any unforeseeable future changes in workload.

Starting with a simpler, cheaper system also allows greater flexibility. As the system begins to grow and scale in small steps, it will grant the ability to change priorities if it seems necessary. Today's market evolves more rapidly than ever before, meaning that what you originally planned at the outset might not apply within a short time. You may have intended to buy a certain package once growth reached a higher level, but changes in the market might make you change your mind and opt for something else. In a scalable system, that's a perfectly valid option. And even with changing market demands, the same software used in the very beginning will still perform its purpose, because its purpose was to grow and change with the company.

When making changes to the system, scalability also provides the benefit of smaller learning curves for staff. Implementing new software in less agile systems would require significant effort for employees to learn to use, whereas adding new features to scalable systems should be fairly easy,

if not actively beneficial to their workflow.

On the other hand, not prioritizing scalability can lead to some negative consequences. They won't be visible at first when the workload is light, but as soon as growth puts more strain on the system, limitations to productivity will arise. The only remedy will be patches that add complexity to the system, compounding the problems over time. Users will suffer as loading times become slower due to an overloaded server. Employees will also struggle, potentially leading them to avoid using the software in favor of something they see as more reliable. These problems can all lead to risk for breaches and data loss.

Implementing Scalability

When planning a scalable system, there are several factors for an IT team to take into consideration. The first is costs, which can easily be higher than the business side that the company would prefer. The price tag can be particularly high if the company is already on its feet using a non-scalable model and making a switch to a scalable one. But the costs of implementing scalability are an investment that eventually prove their value.

The next consideration is the most obvious: usage. As more people use the system, it needs to be able to handle the traffic. Enabling this scaling-up should be as easy as granting the software access to more resources. Keep an eye out for and avoid any artificial limits on possible connections and simultaneous users.

Similarly, sites with user-generated content or other large amounts of unstructured data need to ensure scalability in the maximum stored data limits. The two factors influencing this area of scalability are the database style and indexing. For instance, NoSQL databases tend to pair better with the goal of scalability than SQL. Choose carefully to ensure that the system can scale quickly to handle sudden dramatic rises in stored data.

In some infrastructures, prioritizing scalability can mean sacrificing control and visibility. Switching to a distributed model on the cloud is what allows for horizontal scalability, but cloud providers then own the devices and links. To make up for this, IT teams in cloud and hybrid infrastructures need to closely monitor the network and its applications as they build scalability.

Applications also pose new problems in scaled-up software. Rather than going down entirely, as they may have done in the past, applications are more likely to become slower and slower. If you don't keep an eye on them, then the lagging speed can cause repercussions for the employees who use them. Workers might even resort to alternate applications without IT's involvement, making it difficult to keep track of what's on the network.

Additional considerations to help smoothly implement scalability include load balancing software, edge computing, asynchronous processing, platform-as-a-service, function-as-a-service, and software-as-a-service. The methods and tools that you take advantage of should ultimately depend on the size and needs of your company and its IT systems.