Orchestration, do you think of the orchestra? Well, let us move into how DevOps see orchestration and see if you still assume the same thing when you hear Orchestration.

Orchestration is an automation process that supports workflows, such as the provisioning of resources. With orchestration, managing a complex infrastructure is less like being a builder and more like conducting an orchestra. If you think about what a builder does, they build a building piece by piece by piece. They're responsible for every part of that building. But a conductor is not responsible for every detail of what the musicians are doing, they are simply leading the musicians, and the musicians are responsible for the details.

But let us put this into the software development context for orchestration. Instead of going out and manually creating a piece of infrastructure chunk at a time, we are simply signalling what we need to be done at the high-level design, and orchestration tools take care of the details.

We will go over a few examples to see what orchestration looks like. The first example involves a customer requesting more resources for a particular web service. This customer is leading a vital marketing effort, and they're anticipating a significantly increased load on this web service, and they're requesting more resources to support that load.

Instead of going out and manually standing up nodes with orchestration, engineers will go to their orchestration tool and tell the orchestration tool, "we need an additional six more nodes for this particular service", estimating the number of additional nodes required for the customer. A few minutes later, the tool has these new nodes up and running.

The engineers did not need to go out and manually handle all the details of getting each one of those nodes set up. They simply told the orchestration tool, "we need six more". The orchestration tool took care of the rest.

In the second example, we have a monitoring tool detecting in real-time an increased load on the service. In the first example, the customer pre-planned and requested extra resources for the web service because of the foreseen new load beforehand. However, in this case, the monitoring tool detects it in real-time and communicates to the orchestration tool.

The orchestration tool responds to that event by automatically spinning up additional resources to handle the load. Then when the load decreases again later, the monitoring tool communicates to the orchestration tool to spin the extra resources back down, freeing them up to be used by something else. This creates a system that automatically detects changes in load and automatically provisions additional resources when needed and de-provisioning those resources when they're no longer required.

Keep in mind that all of this is happening while the engineer is working on another task. You will find that most video web services like Netflix, Stan, Dinsey+ all use orchestration to grow and shrink their resource pool due to the demand throughout the day.

Having the ability to spin up resources automatically can be a powerful tool to allow us to have environments capable of handling significant changes in load. This ensures there is no wasted resources by having online servers with no demand spun down.

What does all this mean? A positive outcome for why you should use orchestration is its scalability. With orchestration, resources can quickly and efficiently be increased or decreased to meet rapidly changing needs. Orchestration can also offer us additional stability by adding these additional resources to ensure there is no lag time in a product to the customers.

Automation tools and orchestration tools can automatically respond to fix problems before users see them. They're capable of noticing, for example, that a particular node has become unhealthy by its performance is degrading or it is broken in some way. An orchestration tool can remove that node from the resource pool and replace it with one or more healthy nodes to prevent that unstable node from actually creating downtime or instability for the user. Orchestration tools can also save time because engineers don't have to worry about every single detail of managing resources. Their time is freed up to work on more valuable things.

An orchestration can allow us to offer resources to customers in a self-service and automated fashion. One example we use at WYWM is on Azure to stand up VM's for students. When we request additional resources, our engineers don't do anything, as it's purely automated. An instructor can ask the instance to be added, and the Azure orchestration tools stand it up for us without any intervention by a human other than changing the number of resources we wish to stand up.

Orchestration tools also offer granular insight into resource usage, and this is because these tools control the provisioning of resources. They're able to give you data and insight into how and where those resources are being used. They can tell you what software, which services, and which customers use most of the resources.

A quick summary would be, orchestration is a powerful practice. It can allow you to scale up and scale down in response to rapidly changing business needs. It can allow you to create self-adapting environments that can automatically recover from faults and failures and generally give you more significant stability.