# Dockers and Containers: Do They Make Sense for Your Enterprise?

By Bala Pedapalli

Application Development

## Introduction

Containers are like an undiscovered band that's played the same venue and the same old songs for many years but suddenly finds a stunning vocalist and vaults to the top of the charts. Their exciting new lead vocalist is Docker, and they're embarking on a worldwide tour.

Containers are virtual encapsulations, or images, of an application with its own operating environment, for example a single instance of MySQL. Enterprises find containers intriguing for many reasons, primarily because they vastly simplify application deployment, they require low overhead, and once you create a container you can use it in an unlimited number of places.

Created as a restricted modified runtime environment (called Jail) in UNIX for programs to run with less access to protected resources, containers got their name after the Sun Solaris 10 release in 2005. Along the way, they were enhanced to isolate processes from all resources except where they were explicitly allowed, which isolated them more and made them more secure. Running applications in a container was considered good practice, but dependencies like libraries and special network access made the building, maintaining and sharing of containers very difficult to manage. So the band played, but only in small, underpopulated venues for a curious but ultimately disinterested audience.

Enter the new lead singer. Docker, an open-source project since 2013, made creating, managing and sharing containers very easy through the use of images. Docker Inc., is currently working with Microsoft, Amazon and Google in development in an effort to grow this technology through their cloud offerings. In doing so, Docker has taken the main stage.

So what exactly is Docker? Think of Docker as having two main components: a daemon, which is a program running as a background process that's responsible for starting and stopping containers and creating images; and a command-line interface (CLI) that interacts with the daemon by issuing commands. The image contains all the software to run the containers and can either be created locally, or more often, pulled from a public or private repository.

Here are the main benefits causing all the buzz around Docker:

- *Docker helps reduce development costs.* Developer machine setup and creating an environment to reproduce the same errors as in production are challenging tasks that require a lot of a developer's time. But by using the same image as in production, a developer can easily reproduce the environment to troubleshoot. One developer can create an image and share it with the team to reduce development setup time. For example, if you're using a web server with PHP and a MySQL database for development, all you need to do is go to Docker and pull this image into your repository. This is principally what makes containers so great.
- *Docker helps reduce IT administration overhead.* An IT team has considerable work in maintaining various machines (webserver, database, etc.) for each dev, test and production environment, and the sharing of these machines between different applications only complicates the matter. Using the same image to create containers across different environments considerably reduces this workload. Also, because each application gets its own container, application upgrades can be performed without affecting other applications.
- *Docker provides better security and protection.* By running an application in a container, the application has fewer privileges and therefore cannot affect other applications, their data or the operating system. Considering that many library dependencies are downloaded from the Internet, having this restricted environment to run applications protects the system as a whole.

## The Relationship between Docker, Microservices and Virtualization

Are you able to leverage Docker and containers in your business? For enterprises, the transition may be quite complicated. Taking a look at some other technologies like virtualization and microservices will help answer the question.

Virtualization is the division of a server into separate environments, each with its own operating system and applications running within it. It's still widely in use at enterprises today but is beginning to be regarded as a legacy technology because it's resource intensive. Only a limited number of virtual machines can be run on physical hardware, but you can run hundreds or even thousands of containers on a server.

Microservices are individual services running within an application, for example shipping and billing services running in a retail app. Microservices and containers complement each other, providing agility, scalability, stability and cost reduction to application delivery.

Here's how to tell if you're ready for Docker and containers:

If you're running virtualization in your enterprise, a simple way to discern whether you can leverage Docker and containers is to look at the architecture of your legacy enterprise applications. Are your applications written in such a way that will allow them to be separated into microservices? If so, then each microservice can be made into its own container, and the benefits of Docker and containerization are on your doorstop. If you're not entirely sure whether your apps qualify in this fashion, you can bring in a consultant to do an assessment to help determine your readiness for containerization, explain best practices, and go over a migration path.

Docker is a mature technology with lots of benefits to offer enterprise IT. There's a cost to learning new technology and implementing it as part of your IT solution, but steep upfront costs result in cost savings and other benefits down the line. It also pays to be at the forefront of innovation rather than behind the curve and virtualization will eventually give way to microservices and containerization technologies like Docker.

So if your enterprise applications have been playing the same old tune, it might time for a new lead singer. Maybe Docker is for you.

## About the Author:

Bala Pedapalli has over 10 years of experience developing enterprise solutions utilizing technologies like .NET, JavaScript, as well as a variety of content management systems like SharePoint and WordPress. He graduated with degrees in Electrical and Electronic Engineering from the PSG College of Technology.

## About AIM Consulting

AIM Consulting is a rapidly growing, nationally recognized leader in technology solutions and services. We have the people, processes, and tools to provide companies with strategic guidance on business-critical initiatives and deliver end-to-end solutions. We meet the highest standard of excellence in technology, for better value than other consulting companies, because we are 100% focused on forging long-term relationships with deeply experienced consultants and building high-performance, service-oriented teams that produce results.

## Ready for a Solution?

- Digital Experience and Mobile
- Application Development
- Delivery Leadership
- Business Systems and Strategy
- IT Infrastructure, Cloud, and ITSM
- Data and Analytics

Tell Us About Your Challenge

## Offices

- Denver
- Minneapolis
- Seattle