**ARTICLE ON MICROSERCVICES**

The Microservices architecture is getting a lot of attention these days and it is trending. If it is hard to believe, then just Google it and check what trend says. You can see a big spike of interest over this term, starting in 2014, and the trend keeps on increasing as time passes.

It is worth studying the microservices architectural style when it is at the peak. I believe there is a moderate risk associated with it and it would be good to spend time to understand it. As of now, it seems very promising.

There are some strong points which favor this hype. Several big companies like Netflix, Amazon, and others have talked about the way they scaled and ease out the continuous delivery of their services using a microservices architecture.

Microservices architecture design doesn’t seem to be a buzz to ignore. This architectural framework is the core selling point for emerging startups like Dockers, CoreOS, Cloud Computing, and others. These new products are easing out the development and deployment effort for microservices architecture based applications.

Docker is an [open-source container](https://dzone.com/articles/14-more-open-source-tools-for-taming-kubernetes) technology, which enables us to deploy several self-contained isolated applications on single Linux OS as they are running in their OS environment. It has seen a huge growth in the past year and its major sponsor Docker Inc. has been valued more than $1 billion while getting funding.

I believe there are enough reasons to have a comprehensive analysis of this architectural framework.

**Now question arises what is Microservices architecture and how it works?**

A microservices architecture makes more sense when we compare it with monolithic application design.

In monolithic architectural design, we create a big cumbersome application with all modules tightly coupled inside a single executable, which is typically deployed on a web or application server. But there are some disadvantages to this architectural design. These disadvantages or drawbacks have become the strengths of microservices architecture.

* **The problem in continuous delivery.**
* **Difficult to manage team and project.**
* **Expensive**[**scalability and performance**](https://dzone.com/articles/component-load-testing)**.**
* **Lack of technology diversity.**
* **Not easy to replace components.**

In short, Microservices is an architectural style, an approach to [software development](https://apiumhub.com/web-development-barcelona/) in which a large application is built as a suite of modular services; small, independently versioned, and scalable customer-focused services with specific business goals, which communicate with each other over standard protocols with well-defined interfaces. As they are independently deployable and scalable, each service also provides a firm module boundary, even allowing for different services to be written in [different programming languages](https://apiumhub.com/tech-blog-barcelona/new-programming-languages/) and can also be managed by different teams.

Microservices architecture is a way of breaking large software projects into smaller, independent, and loosely coupled modules. Individual modules are responsible for highly defined and discrete tasks and communicate with other modules through simple, universally accessible APIs.