**Micro-services:**

OVERVIEWS

Micro-services - also known as the micro-service architecture - is an architectural style that structures an application as a collection of services that are . Highly maintainable and testable, Loosely coupled, Independently deployable also Organized around business capabilities [1]. The idea is to split the application into a set of smaller interconnected smaller services that has it own business logic, some micro-services will expose their REST, RPC or other API provided by other services [2]. It has a lots of advantage compare to the monolith.

(BAD ABOUT MONOLITHIC)

In Monolithic architecture, it has limited complexity and limitation, it is slow compare to the micro-servers, also expensive to scale and update [2]. Everytime must redeploy the whole project, this also makes continuous delivery hard to achieve. Since it has high coupling, the reliability drop down. One small bug might affect the whole program.

(Q: back to the days that are you are using ASP .NET have any team used Monolithic? Any specific product?)

(Q: please give the biggest drawback why you disgard this method. Or the reason why you use this method)

Evidence: After interviewing a second-level software engineer employee Dylen from Expedia, he mentioned that Microservices was highly used in his online advertising group.

Through out this career in Expedias, he said the company was transferred from

(GOOD ABOUT MISCROSERVICES)

However, in micro-service application is decomposed into sub apps, all of the services has their own database. Every sub app are deploy separately, so these factor ensure the high maintainability and testability. Micro-services enables the continuous delivery/deployment of large, complex applications, which mean using microserves can achieve better scalbililty [2]. This point is mentioned by [Anton Kharenko](https://articles.microservices.com/@antonkharenko)’s article

(Q: please talk about a product that you applied microservices?)

(Q: what is your own feeling pros about microservices that applied to this product?)

Evidence:

(BAD ABOUT MISCROSERVICES)

If you think microservices is ‘the perfect solution’ then you are probability wrong. it also has some drawbacks. Imaging the database is splited, updating database invloving updating different databases across the network. Also the testing is much more complex compare to the monolith architecture.

(Q: what is your own feeling disadvantage about microservices to this product as a software engineer?)

Evidence:

(CHOOSE THE RIGHT ONE)

whether using an Monolithic architecture or a micro-services really depends on the scale of the applications. Monolithic architecture suit the simple lightweight applications better. For more complex system micro services maybe more suitable.

(Q: is there any situation that expedia use any services which is between these two extremes)

(Q: is there any specific software tool that helps you guy develop software using MS or Monolithic)

Evidence: In the article written by [Jake Lumetta](https://medium.freecodecamp.org/@jakelumetta), CEO at ButterCMS. He mentioned that monolithic is ideal in certain circumstances. Imaging a small group of engineering team for example two person engineering team. It will be less work for the engineers which mean they can handle business change more swiftly [3]. This is also proven by former google employee steven in his early stage in company Scalyer. What is even more interesting is when people talking about microservices they are talking about using microservices or a giant monolithic however in real system are many possible services between these two extremes [3].

(Q: do you have any interesting points that you want to address here? )

[1] <https://microservices.io/>

[2]’ Monolithic vs. Microservices Architecture’ <https://articles.microservices.com/monolithic-vs-microservices-architecture-5c4848858f59>

[3]<https://medium.freecodecamp.org/monolith-vs-microservices-which-architecture-is-right-for-your-team-bb840319d531>