**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?) -- Jia, mingo~~

~~(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 advertisement group.

Through out this career in Expedias, he said the Expedia web was monolithic based back to the second verion. It is a single java based application. It contains a all their function in all such as ‘search API’, search information. The biggest issue mention by him is that when loading the whole code base in to IDE such as intellj or eclipse it takes a long time due to it huge size. Indexing takes hours. Autocomplete also takes a while to load out.

(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?) hotel team/ jsp front end.~~

Evidence:

As Dylen mentioned, all their teams will will transfer their own server storage to amazon s3. All the team need to separate their part in the code base and make their own cloud application, from implementation to deployment. After that all the code base for each team are separated it make the whole develpemt lose coupling.

The advertisement team were using jsp back to the old day. They develop their own API.

Add get info, get slot info, get ad info services, due to the reason that there are too many ad on the web, Expedia advertising group wants to manage these advertisement in all, for example how big , the URL of the ad, what kind of the ad is that. Using a internal API it can be hooked up to different app (web app, mobile app)

By simply us mongoDB to access or even a json file.

(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? ) single dependency. Amzon s3.~~

Evidence:

As mention by Dylen, miscroservse might lead to a situation that different team are using different framework and techmology. For example it happened that one team is using backbone js whileas another team is using react js. However when using monolithic methodiogy this issue will not happeded since all the the developer are using one single code base. All the backend was achieved by java and front-end using freemarker template (FTL). Frontend is very accessable, In one word, the technology everyone using is tht same.

Bask to few years, while the advertment is using jsp in microservices, the technology hotel team using does not support jsp any more. This raise the problems that the advertisement team need to develop a unique custom solution to the hotel team. Is such a conflict happeded which team should take responsibility? Moreover hotel team and hotel infosite team may have the same issue, should infrosite abanded jsp as well?

Another issue that raised by him is that the time talking to backend becomes longer.

Addig one single point dependency is also a bad practise to system design. For example, if the one point in microservice break the whole thing talk to this services will fail.

(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)~~

~~Case by case.~~

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].

Metioned by him that this situation always happed, sometime they will use some ugly solution to solve this problem, thir monolithic will still exist. Sometime they will write some code the get some information form the old version of the code and add into the could. Such as using iframe to get some info and abanded the frame afterwords.

they always deveope a custom solution to solve this kind of problem this makes it harder.

Such as loadlegacyversion( ) else( ).

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

the web application is quite completed that, for example the version of the different web page from the same site might be different, in expedia the after customer pay the fee they,

it also depepent on teams, some team are lazy so that using miscroservies might not correperate. It actually need a big guy who have a big insight of the technology.

The interesting point that mention by him is that microserver is the trend the development of software. However this could gone even smaller one servers only responces for one thing. Nevertheless we should not using the monolithic for good practice. Using microvice, It can save tech develpent opertraiotn time since the server is on cloud.

Another interesting point that mention is that who ever make a build fail, another person need to sit their and fix the issue, this makes the person hate the one who make the make build fail.

[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>

**Micro-services:**

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[1]. The idea is to split the application into a set of smaller interconnected smaller services that has its own business logic, some micro-services will expose their REST or other API provided by other services [2].

In Monolithic architecture, it has limited complexity, it is slow compared to the micro-servers, also expensive to scale and update [2]. Redeployment makes continuous delivery hard to achieve. Since it has a high coupling, the reliability drops down. One small bug might affect the whole program. After interviewing a second-level software engineer from Expedia, he mentioned that microservices was highly used in his online advertisement group. Throughout his career in Expedia, he said the Expedia web was monolithic based back to the second version. It is a single java based application. It contains all their functions in all, such as ‘search API’, ‘search information’. The biggest issue using monolithic mention by him is that when loading the whole code base into IDE such as IntelliJ or eclipse takes a long time due to its huge size. Indexing takes hours. Autocomplete also takes a while to load out.

In micro-service, applications are decomposed into sub apps, all of the services have their own database. Every sub-app is deployed separately. Microservices enables the continuous delivery/deployment of large, complex applications, which mean using microservers can achieve better scalability [2]. This point is also mentioned by Anton Kharenko’s article. As mentioned by the engineer from Expedia, all their teams will transfer their own server storage to amazon s3. All the team needs to separate their part in the code base and make their own cloud application from implementation to deployment. After that, all the code base for each team are separated it makes the whole development lose coupled. The advertisement team in Expedia were using JSP back to the old day. They develop their own API such as “add get info”, “get slot info”, “get ad info services”. Due to the reason that there are too many ads on the web, Expedia advertising group wants to manage these advertisements in all, for example, how big is the add, the URL of the ad, what kind of the ad is that. Using an internal API that can be hooked up to a different app (web app, mobile app) and simply us MongoDB to access the database make life easier.

If you think microservices is ‘the perfect solution’ then you are probably wrong. it also has some drawbacks. Imaging the database is split, updating database involving updating different databases across the network. Also, the testing is much more complex compared to the monolithic architecture. As mention by this engineer, microservices might lead to a situation that different teams are using different framework and technology. For example, it happened that one team is using backbone js while another team is using react js. However, when using monolithic methodology this issue will not happen since all the developer is using one single code base. All the backend was achieved by java and front-end using freemarker template (FTL). The frontend is very accessible, In one sentence, the technology everyone using is the same. Back to few years, while the advertisement in Expedia is using JSP in microservices, the technology hotel team using does not support JSP any more. This raises the problems that the advertisement team need to develop a unique custom solution to the hotel team. If such conflict happened which team should take responsibility? Moreover, the hotel team and hotel info site team may have the same issue, should info site abandon JSP as well? Another issue that raised is that the time talking to backend becomes longer. Adding one single point dependency is also bad practice to system design.

Whether using a monolithic architecture or a micro-services really depends on the scale of the applications. Monolithic architecture suits the simple lightweight applications better. For more complex system microservices maybe more suitable. 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 a real system there are many possible services between these two extremes [3]. Mentioned by the engineer in Expedia that this situation always happens, sometime they will use some ugly solution to handle this situation, while the monolithic code base will still exist. Sometime they will write some code to get some information from the old version of the code and add into the could such as load\_legacy\_version( ) or using iframe to get some information and abandon the frame afterward. Expedia always develop a custom solution to solve this kind of problem this makes it harder. Whether to use microservices also dependent on teams, some team are lazy so that they might not cooperate. It actually needs someone who has a deep insight view of the technology to talk to different teams. The interesting point that mentions by him is that microserver is the trend the software development. However, this could have gone even smaller one servers only responses for one task. Nevertheless, we should not use monolithic for good practice. Using microservers can save tech development operation time since the server is on the cloud. Another interesting point that mention by him is in Expedia whoever make a build fail, another person needs to fix the issue, this makes the person hate the one who makes the make build fail.

[1] Kong,2018 ‘What are microservices?’ access date: 2019 [Access Date]:8th of Mar <https://microservices.io/>

[2] [Anton Kharenko](https://articles.microservices.com/@antonkharenko) Oct 9, 2015 ’Monolithic vs. Microservices Architecture’ access date: 2019 [Access Date] 8th of Mar <https://articles.microservices.com/monolithic-vs-microservices-architecture-5c4848858f59>

[3] [Jake Lumetta](https://medium.freecodecamp.org/@jakelumetta) Jan 17, 2018 ‘Monolith vs microservices: which architecture is right for your team?’ [Access Date] 8th of Mar

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