**Introduction**

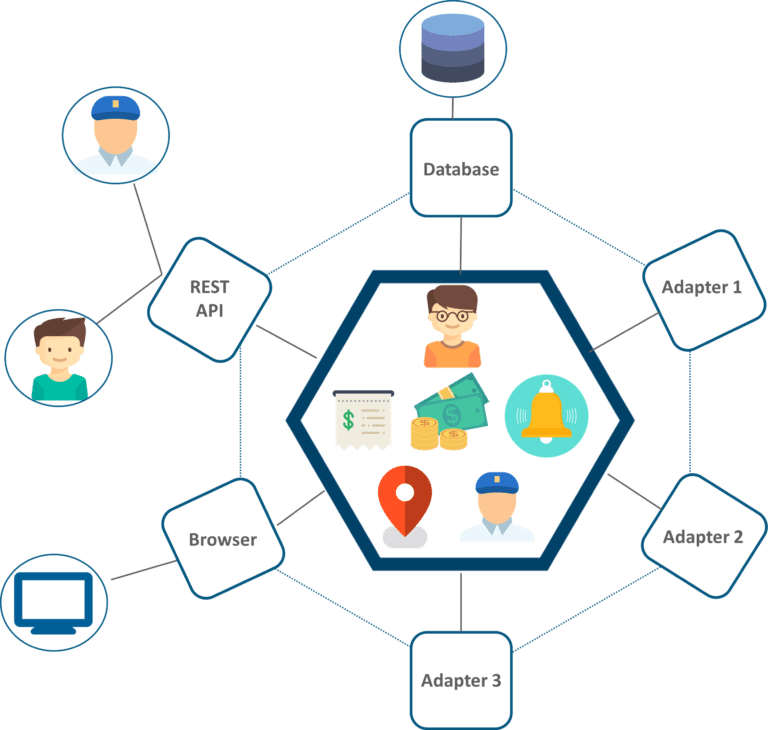
Microservices are individual services that interact with a system through API calls. An architecture built in microservices can allow companies to create individual parts of a system and have them encapsulated so that they can be developed separately from the other parts. This can allow for faster bug fixing, scaling, and code management.

**Uber Technologies Inc.**

Uber is an American ride-hailing company that includes peer-to-peer ridesharing, ride service hailing, food delivery, and a micro mobility system. Uber uses software when ever a user opens their application. This includes passenger management, passenger web UI, driver management, and billing.

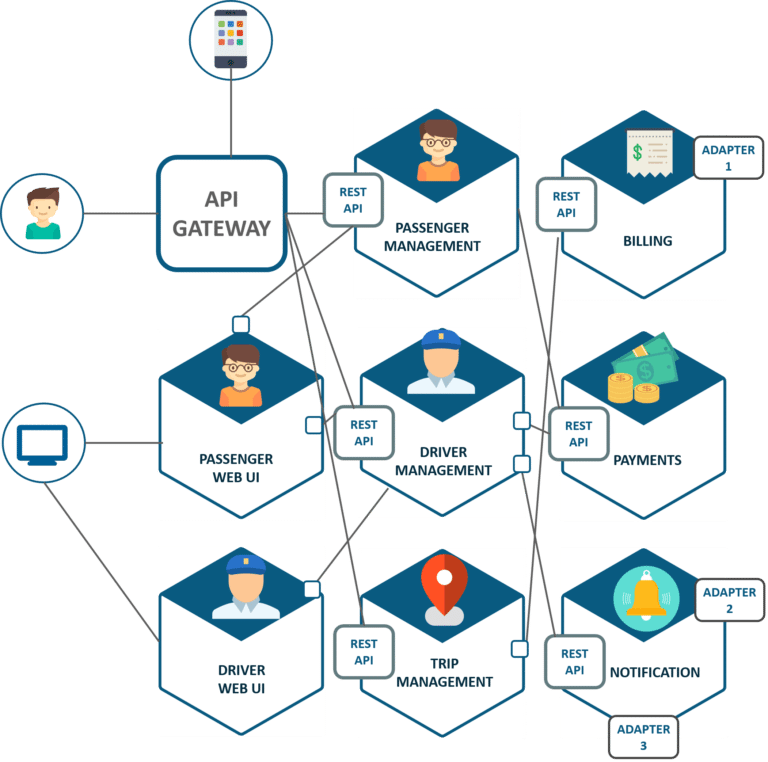
Uber uses microservices to handle most of their operations from passenger management to billing to notifications. This system is triggered when a user sends a request from their frontend UI and notifies drivers of the request through these services.

Uber originally had a monolithic architecture when they first started. This architecture included a single restful API, three adapters to preform billing, payments and notifications, and a MySQL database. This worked for them until they started to rapidly grow and expand which caused a lot of growing pains for the company. The major problems that occurred were that every time a updated or added every other feature had to be rebuilt, deployed and tested again, this was also true for bug fixing as well as working on a single massive code repository made fixing and finding problems harder. The final issue was that the current features were hard to scale with the increasing usership and new additions to the system.



Uber’s monolithic architecture

Uber had many successes with their switch from a monolithic architecture. This change saved them from needing to rebuild their entire system every time an update for a single feature was developed and split their code base from a single repository to many smaller repositories that were more manageable for the developer teams. This change also encapsulated each feature so a bug in a single feature did not bring down the entire system.



Uber's microservice architecture

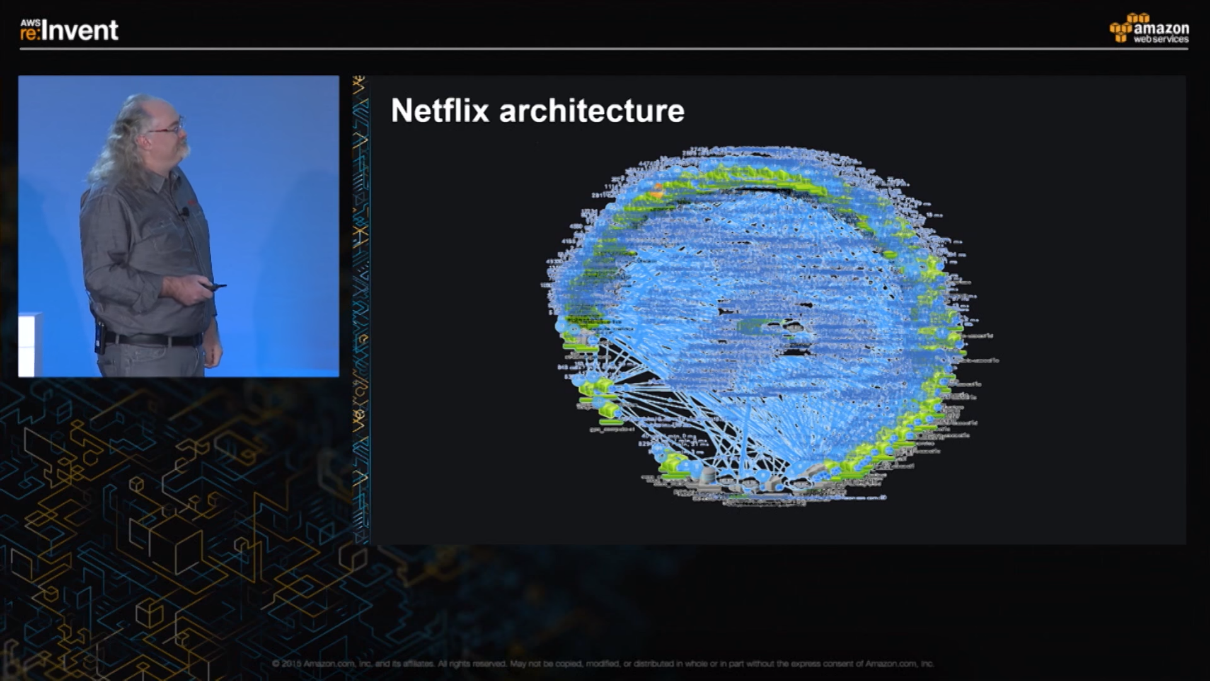
I believe that Uber made the correct choice in changing to a microservices architecture because if they hadn’t then their growing pains would have just gotten worse and this would have slowed down their progress for the long run.

**Netflix Inc**

Netflix Inc. is an American media service provider and production company that was founded in 1997. Netflix is known as the worlds leading streaming entertaining service and has over 167 million paid memberships in over 190 countries. Initially founded in 1997 and opening in 1998 as a DVD rental and sales site “netfilx.com”. In 2007 Netflix introduced their streaming platform and began to rapidly grow. In 2013 Netflix became a producer of original content with the launch of the series House of Cards, Hemlock Grove, Arrested Development, And Orange is the New Black.

Netflix claims to use around 700 microservices that do everything that makes the service. One service stores what the user has watched while another handles subscription fees while another provides the user’s device the files to play the program.

When Netflix created their streaming platform they had decided to switch from their monolithic architecture to a microservices based architecture. This was to accommodate the new streaming platform that was in development and so that if any part of the service was to be changed the entire system would not have to be rebuild and tested for each iteration.



Dave Hahn, a senior engineer from Netflix’s Performance and Reliability Department, shows off the entire architecture of Netflix in one flow diagram. (Amazon Web Services/YouTube)

Netflix’s architecture is extremely advanced and they have claimed to have around 700 individual services. Netflix attributes a lot of its success to it’s microservice architecture that they developed.

I believe that Netflix made the right decision as well using the microservice architecture to handle their systems because if they hadn’t used this architecture then their service would have become a monumental mess to maintain with all of the services in a single repository.

**Comparison**

|  |  |  |
| --- | --- | --- |
| Company | Uber Technologies Inc. | Netflix Inc. |
| Hosting | Self Hosted | Amazon Web Services |
| Number of Microservices | 8 | About 700 |
| Reasoning | Not needing to rebuild entire system upon update, growing pains | Build alongside new service system. |
|  |  |  |

**Conclusion**

In conclusion I think that both companies were successful in their use of MSA. Netflix I believe used MSA more extremely then Uber did but both companies used the system to benefit what they were trying to do and succeeded in their actions. Both Uber and Netflix had similar issues with rapid expansion and code management and used microservice architecture to solve their problems.

**References**

<https://medium.com/edureka/microservice-architecture-5e7f056b90f1>

<https://www.uber.com/ca/en/about/>

<https://medium.com/refraction-tech-everything/how-netflix-works-the-hugely-simplified-complex-stuff-that-happens-every-time-you-hit-play-3a40c9be254b>

https://media.netflix.com/en/about-netflix