Particpiants

Wayne

Sam

Cory

Xiuting

Charith

Mahdi

Cherry

Dianxiang Sun (Mason)

Amrutha

Shujie Chen

Kerong Cai

Zhihao Zheng

Qiaoxiang Peng

Yuejian

Yary Wang

Tony Zhou

Yi Zhou

Hong Liu

Presenter

Goal: speak intelligently for micro-services architecture

Engineering then go coding

Experience needed at Nintendo:

Micro-services Python

AWS

Interviewer are new with Micro-services too

No definition yet

Building Microservices – Sam Newman

Engineers came around microservices at the same time

**Monolithic Architecture**

Pros

* Simple to develop
* Simple to test
* Simple to deploy
* Simple to scale horizontally

Cons

* Quickly becomes too large to fully understand
* Difficult to make fast changes
* Size can affect startup time
* Change impact not fully understood
* Continuous deployment is difficult

“Microservices” are small autonomous services that work together.

Gather together things that change for the same reason, separate those who are different

If can be managed by a small team, it’s probably too big.

AWS did this.

Key benefits

* Flexible tech stack
* Resilient
* Scaling
* Ease of deployment
* Organizational Alignment
* Composability
* Optimizing for Replaceability

Speak only through API

If one component fails, only that component fails. Rest of the services would still be up.

Scaling

* Pay as you go

Organizational alignment

* Smaller teams

Easier to replace code in micro-services

Deployment

* Continuous Integration
* Continuous Delivery

3 questions for continuous integration

* Do you check into mainline
* Do you have a suite of test to validate
* Do you have someone fix it

Netflix is ran on AWS