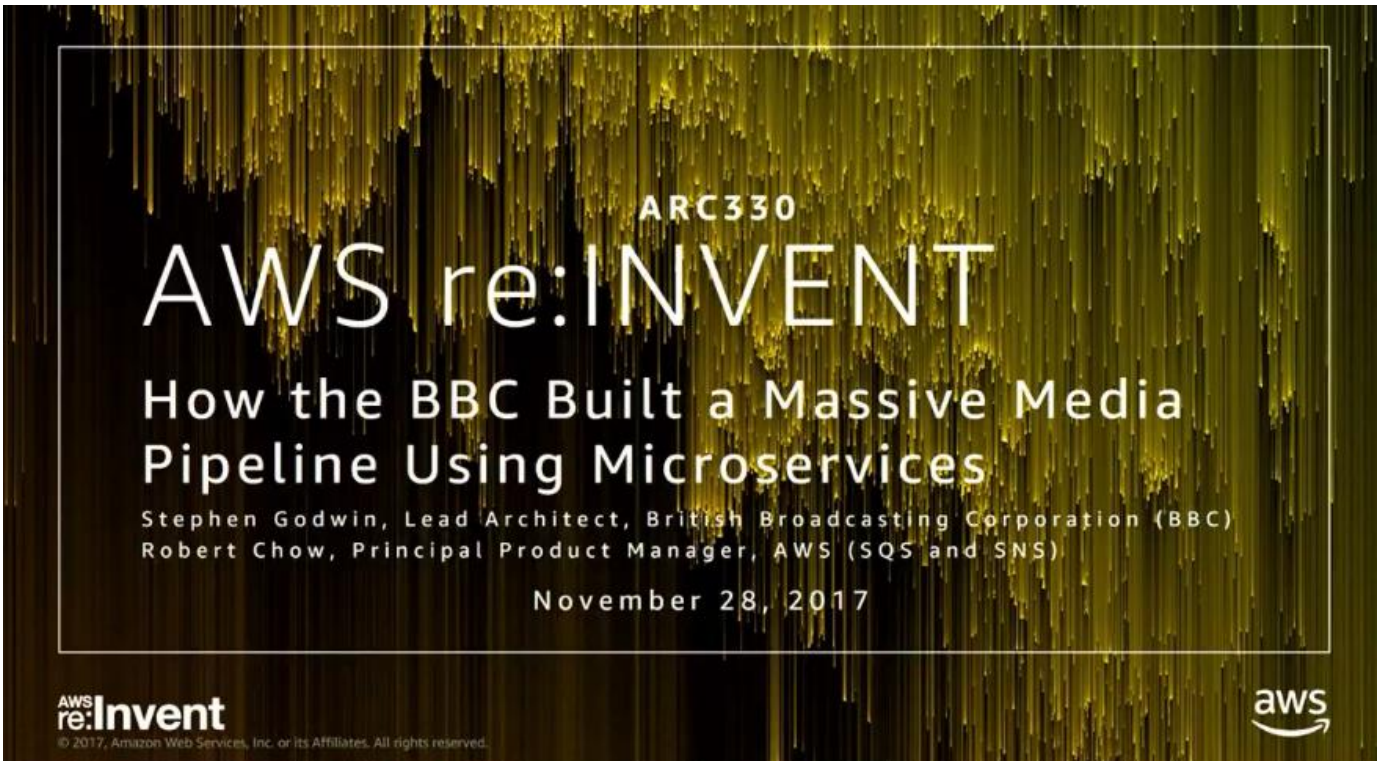
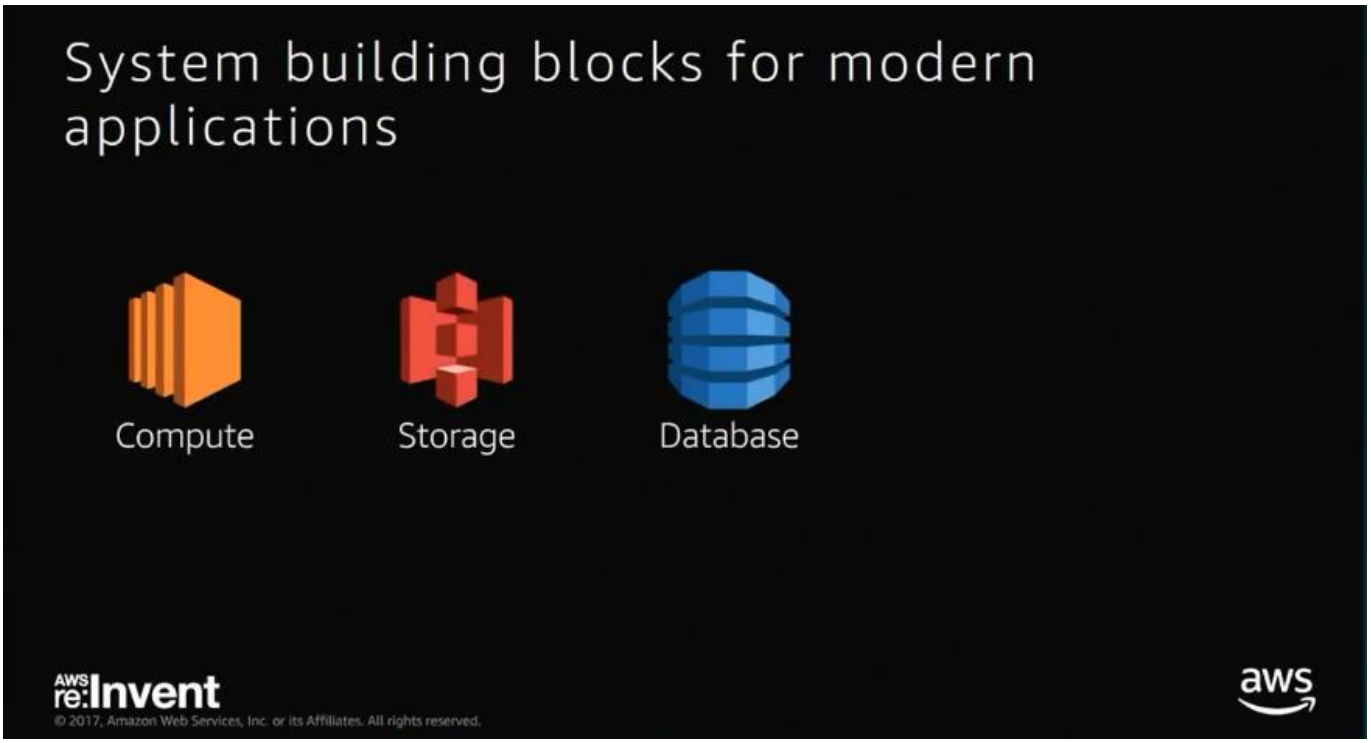
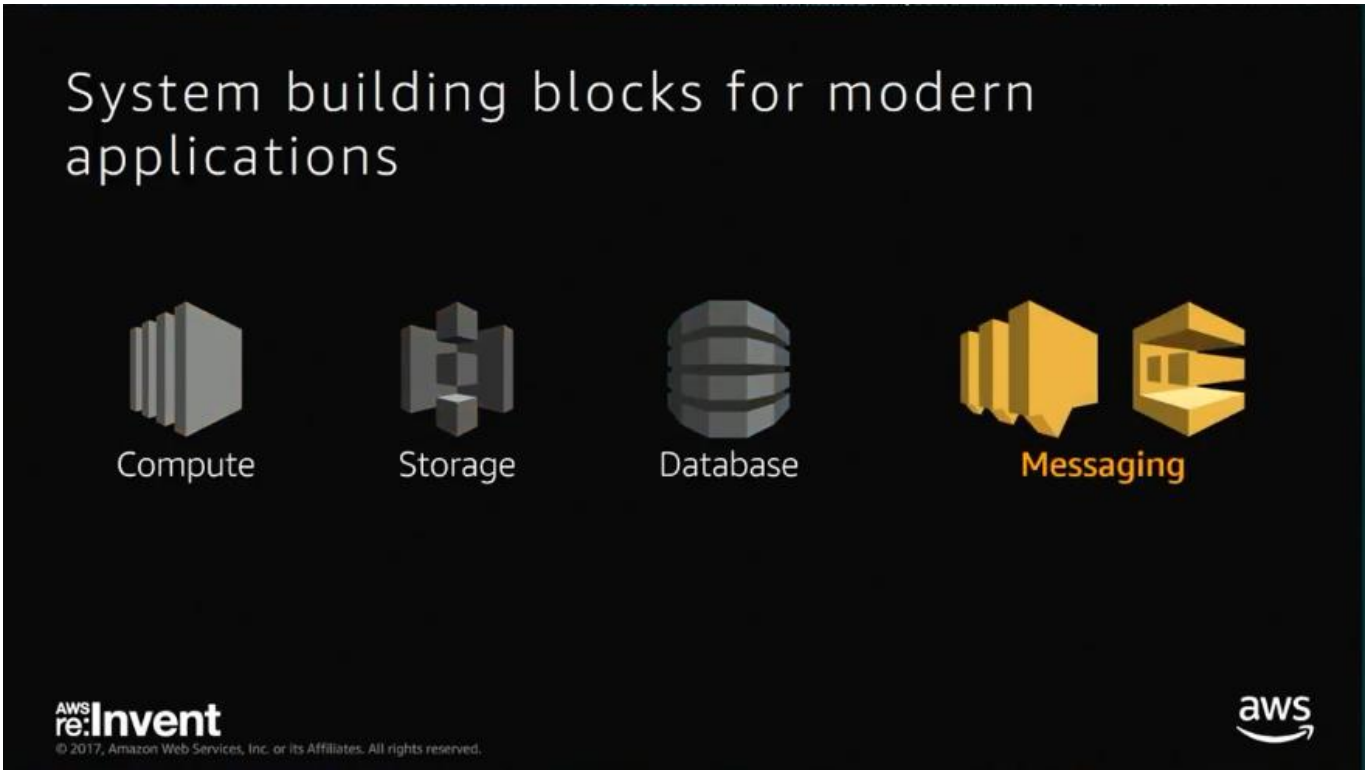


The BBC iPlayer is the biggest audio and video-on-demand service in the UK. Over one-third of the country submits 10 million video playback requests every day, and the service publishes over 10,000 hours of media every week. Moving iPlayer to the cloud has enabled the BBC to shorten the time-to-market of content from 10 hours to 15 minutes. In this session, the BBC's lead architect describes the approach behind creating iPlayer architecture, which uses ***Amazon SQS*** and ***Amazon SNS*** in several ways to improve elasticity, reliability, and maintainability. You see how BBC uses AWS messaging to choreograph the 200 microservices in the iPlayer pipeline, maintain data consistency as media traverses the pipeline, and refresh caches to ensure timely delivery of media to users. This is a rare opportunity to see the internal workings and best practices of one of the largest on-demand content delivery systems operating today.



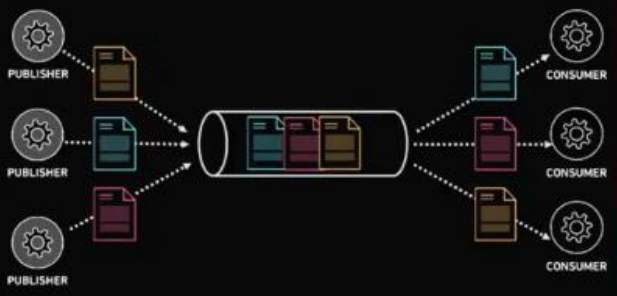


We also think about messaging using 2 specific services ***SQS*** and ***SNS*** that are absolute critical for data management in a scalable system

Amazon SQS & SNS: Decouple applications

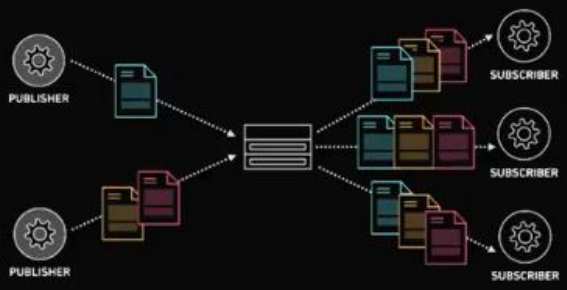
Simple Queue Service (SQS)

- Asynchronous
- Each message processed by one consumer



Simple Notification Service (SNS)

- Pub/sub
- Broadcast or filtered
- High fan-out of messages

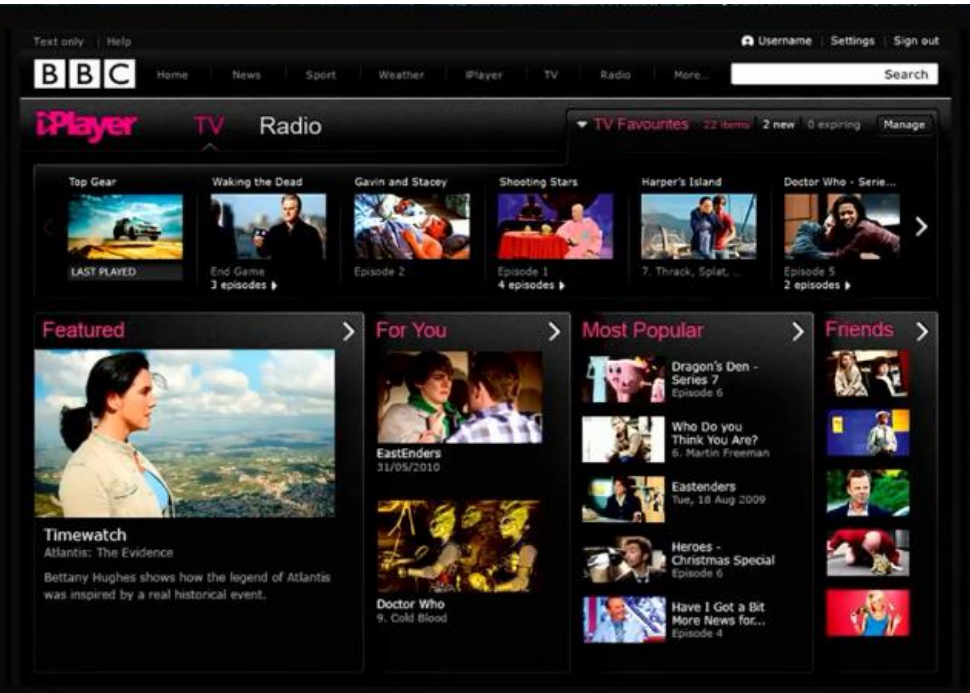


Stephen Godwin BBC

@SteveGodwin

- Lead technical architect at the BBC
- Designed the systems that provides audio and video to BBC iPlayer and iPlayer Radio
- Led the migration of the systems that power iPlayer to a cloud-based microservice architecture
- Worked for nearly 10 years as one of the developers of IBM WebSphere

BBC



BBC iPlayer

WATCHED BY

30 PERCENT
OF THE UK
POPULATION

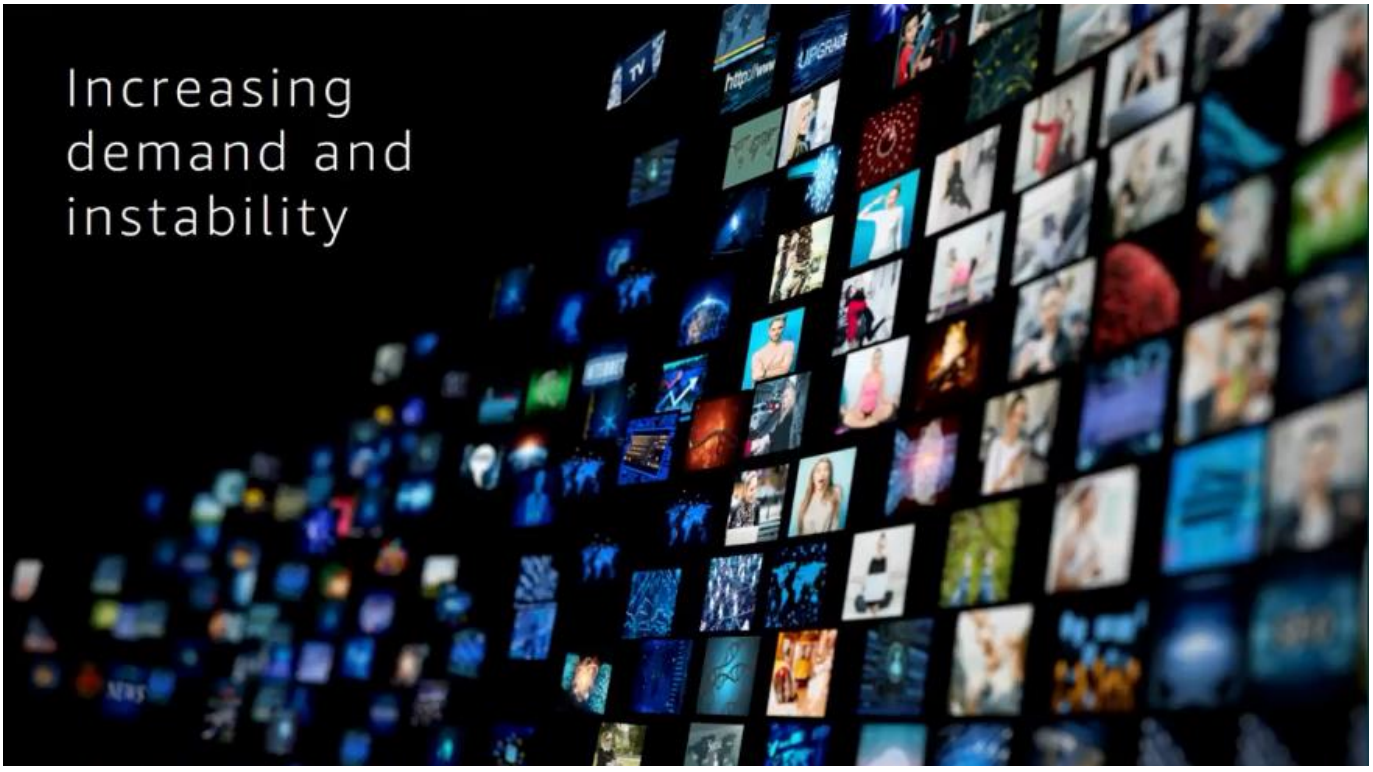
10 THOUSAND HOURS
OF MEDIA PUBLISHED
EVERY WEEK

A NEW PROGRAM
IS AVAILABLE
ON iPLAYER

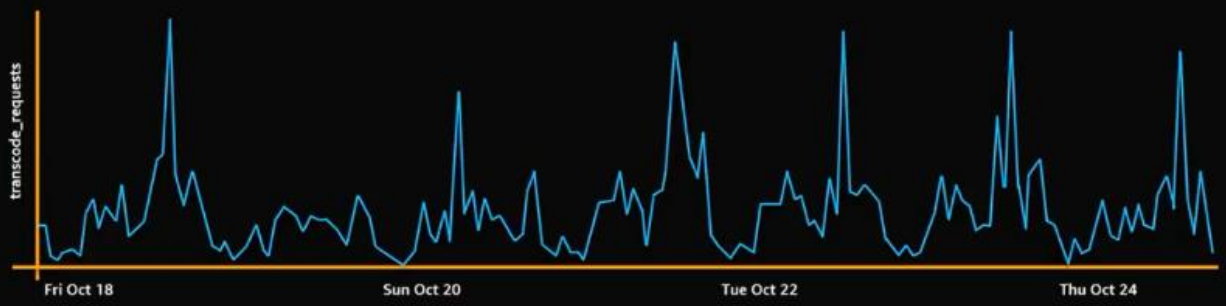
EVERY

15

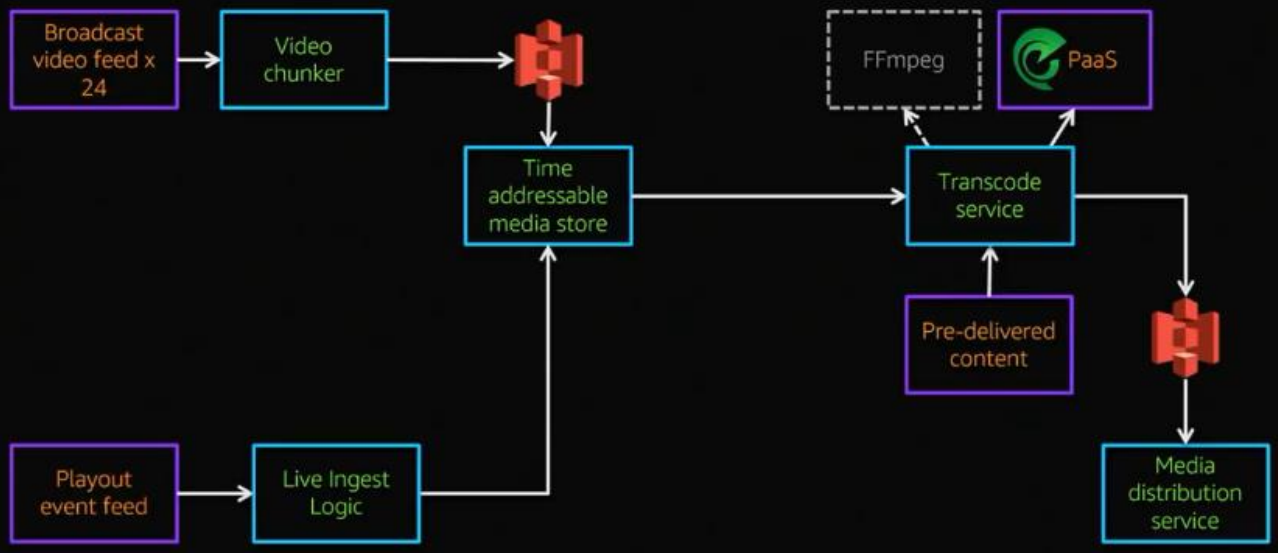
MINUTES

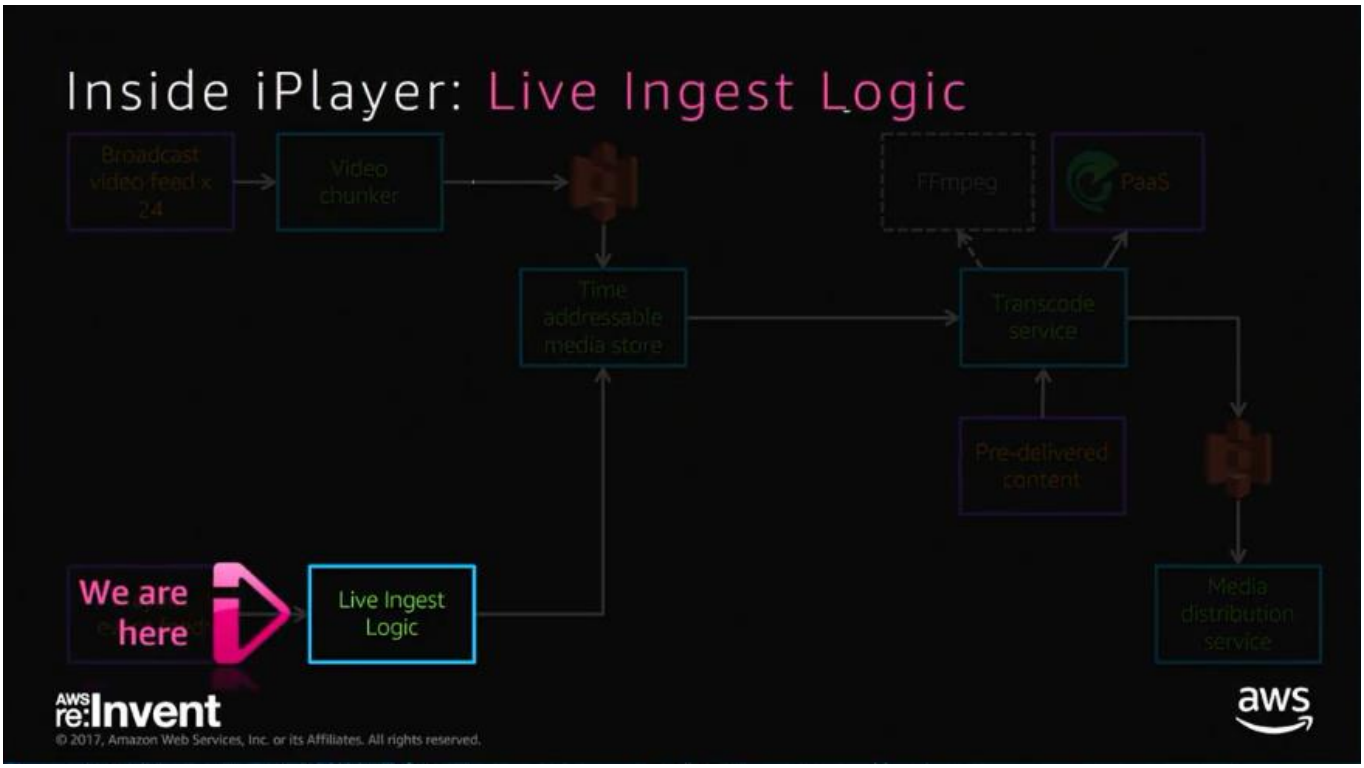


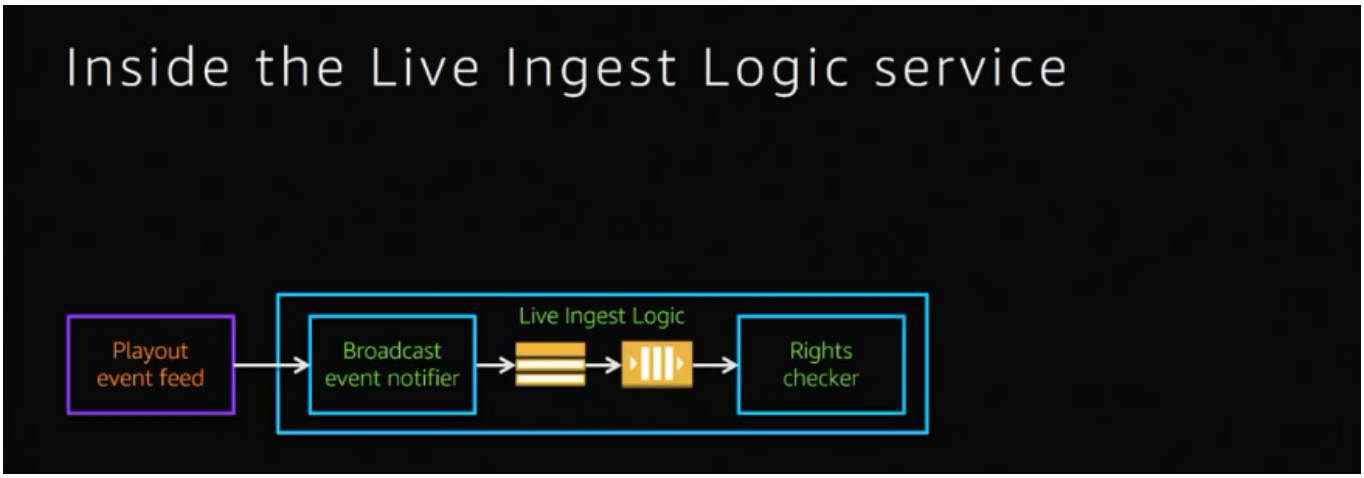
A perfect pattern for an elastic architecture



Inside iPlayer: Video factory

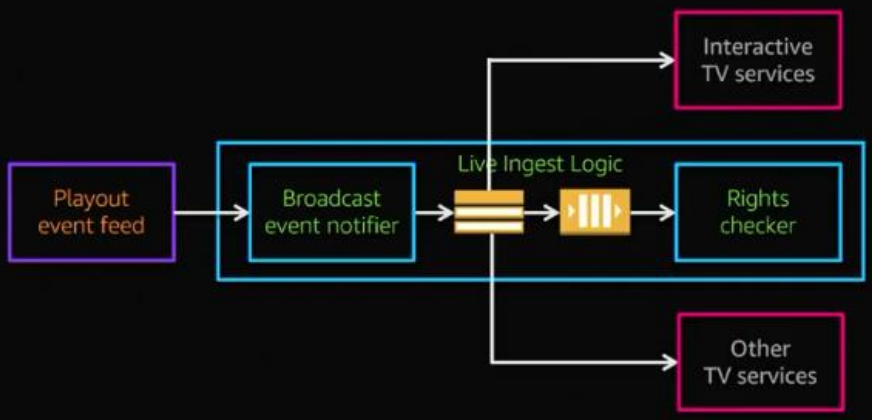




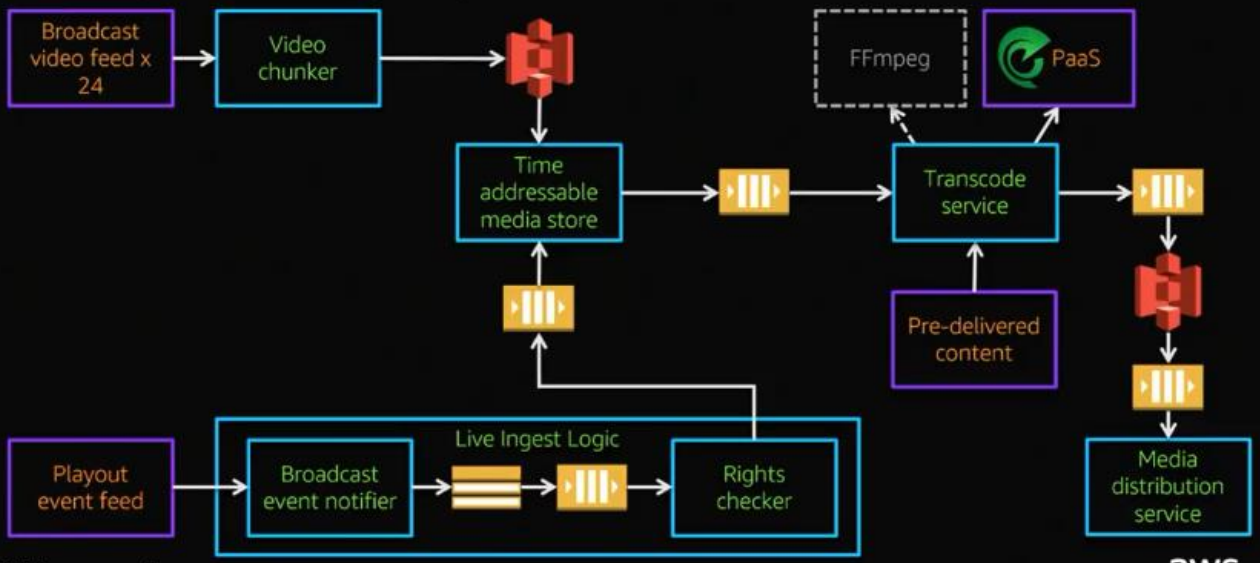


This is a pattern where an SQS queue subscribes to an SNS topic like startEvent or stopEvent

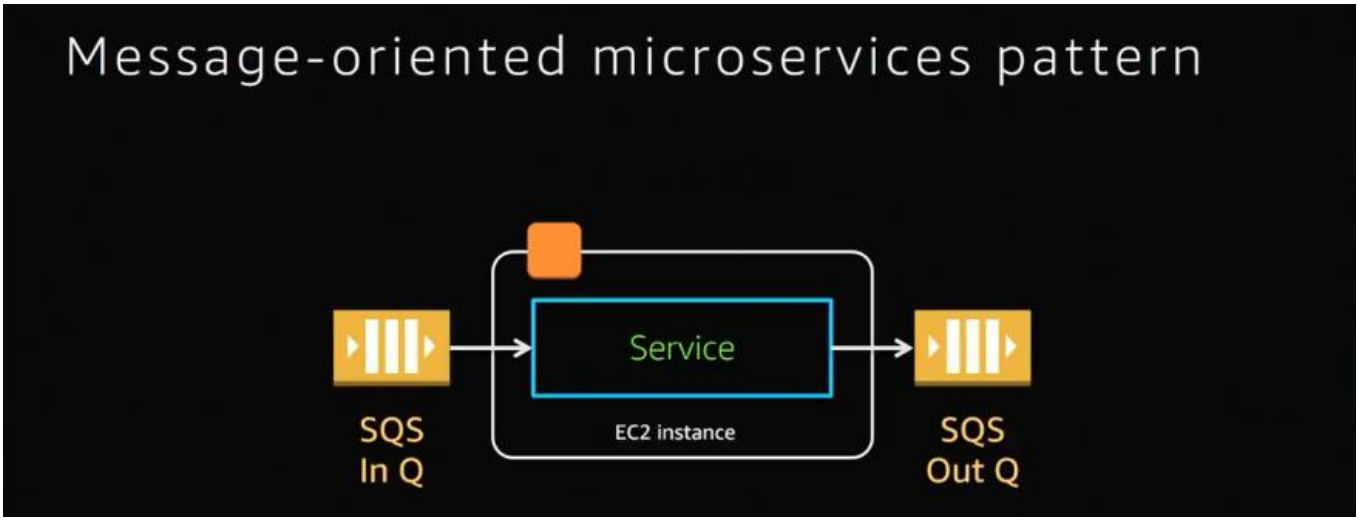
Inside the Live Ingest Logic service



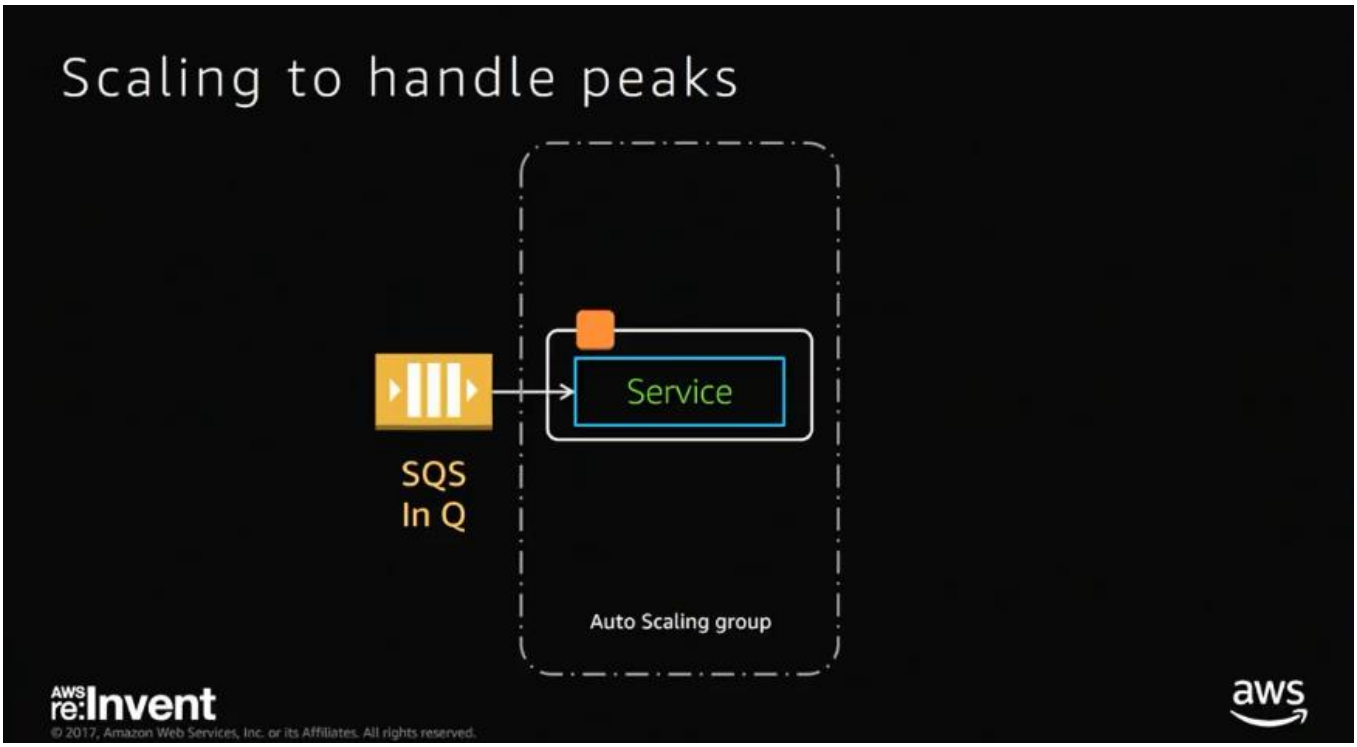
Using messaging to wire the system

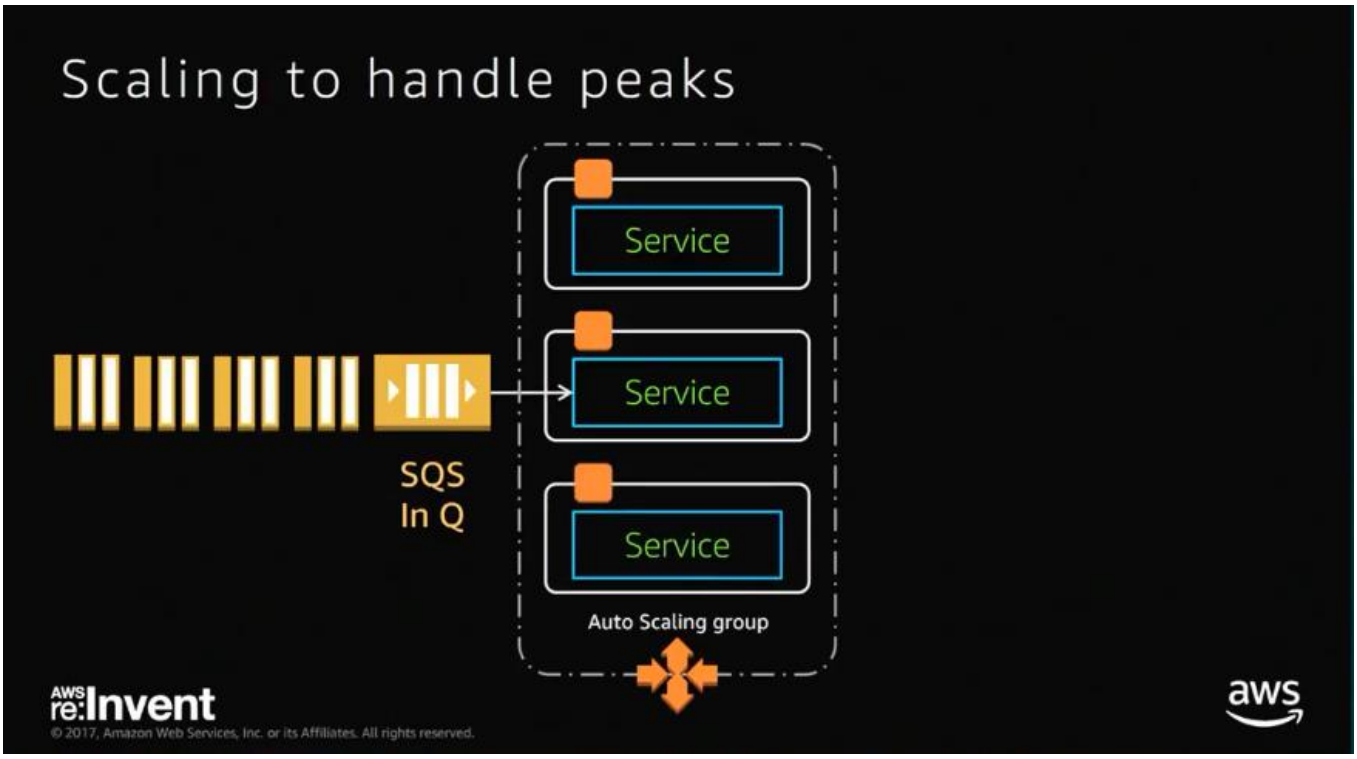


Message-oriented microservices pattern

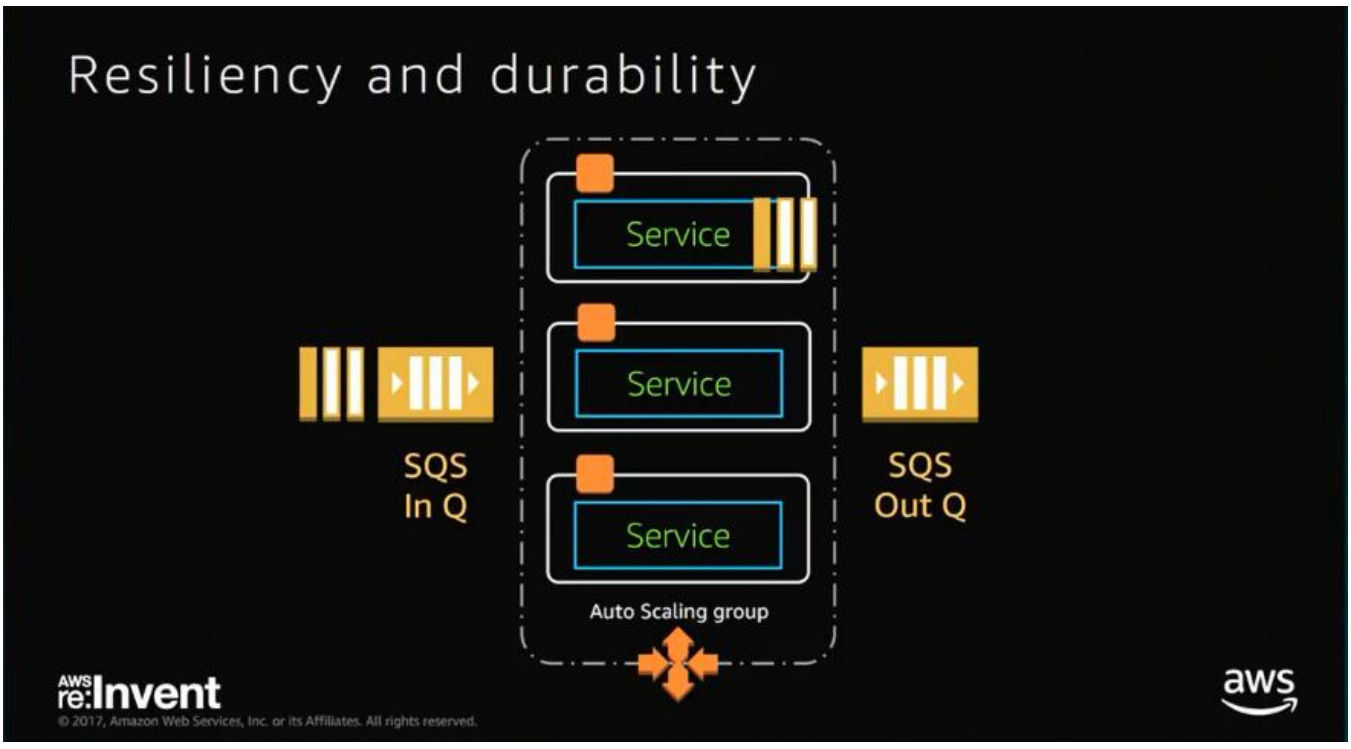


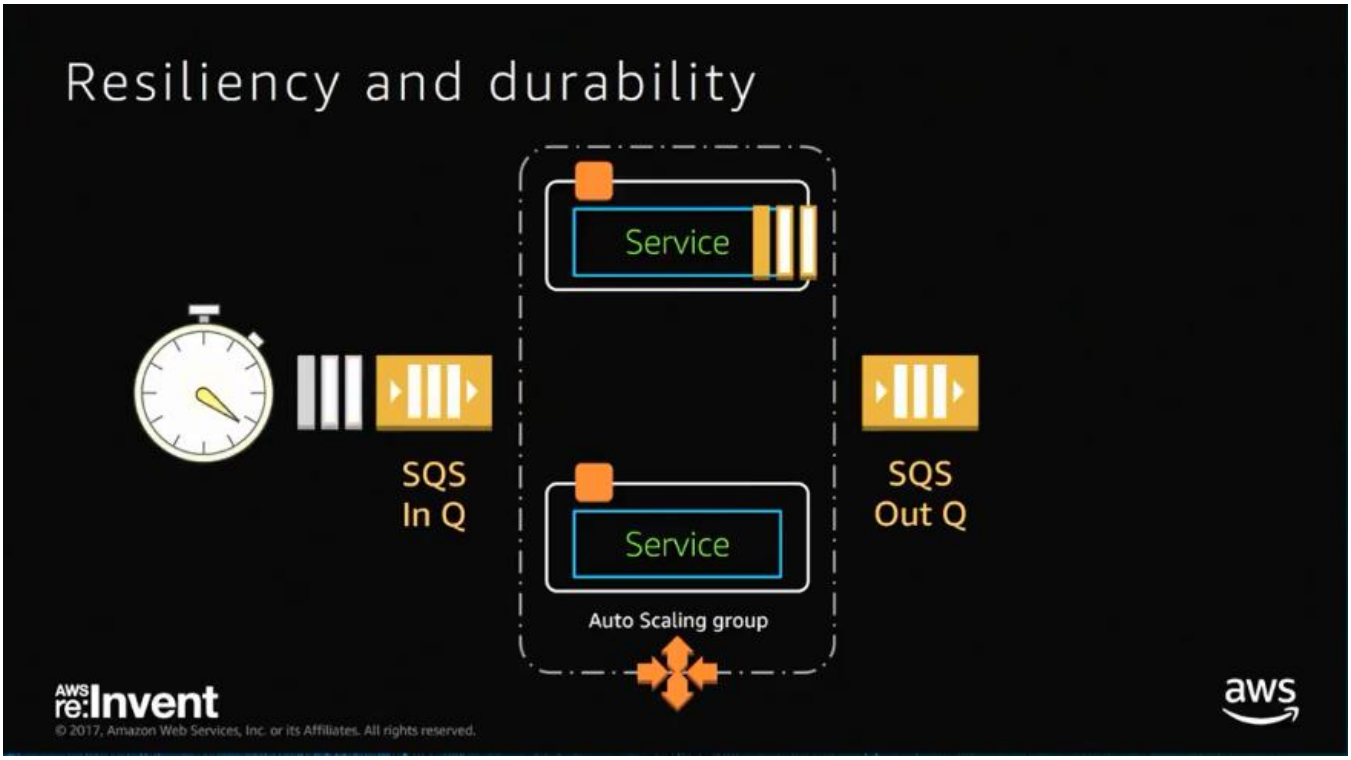
Running on each Service is a Java application running on an EC2 instance that doesn't know much about the queues when doing its job.



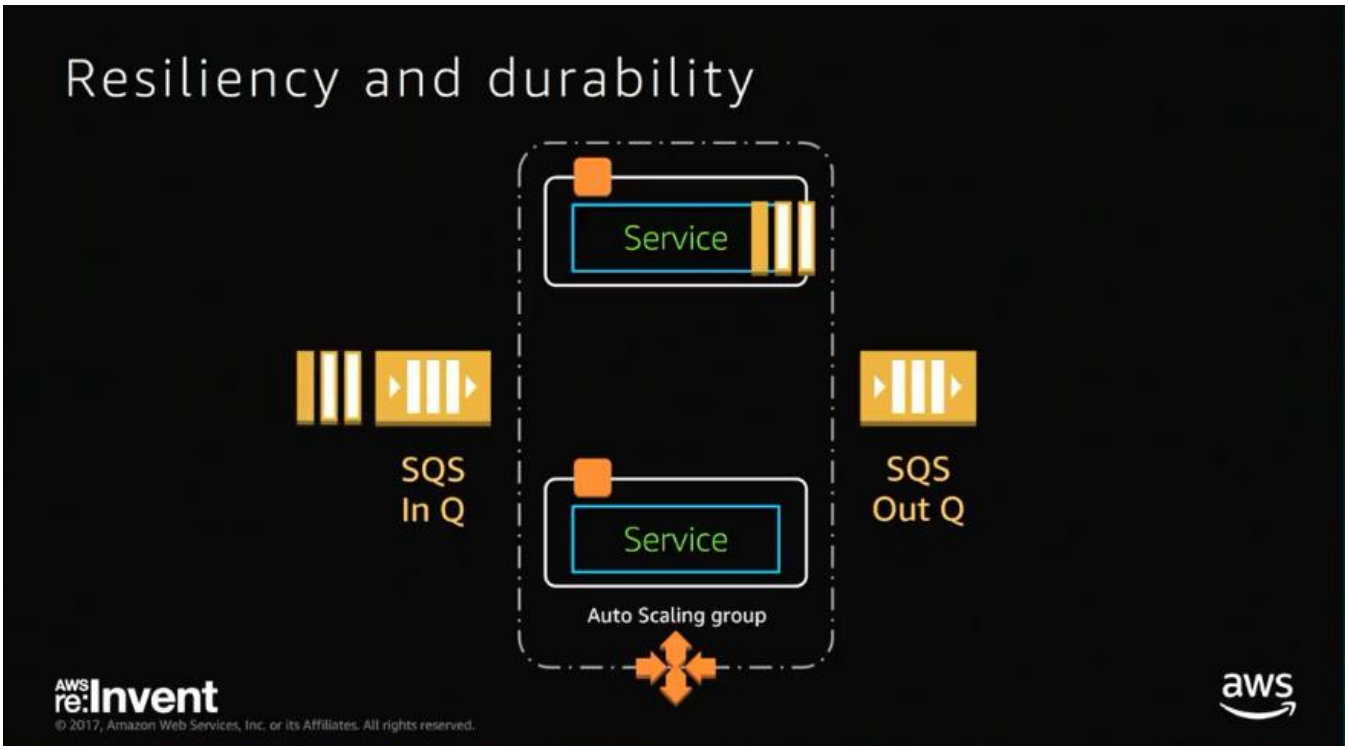


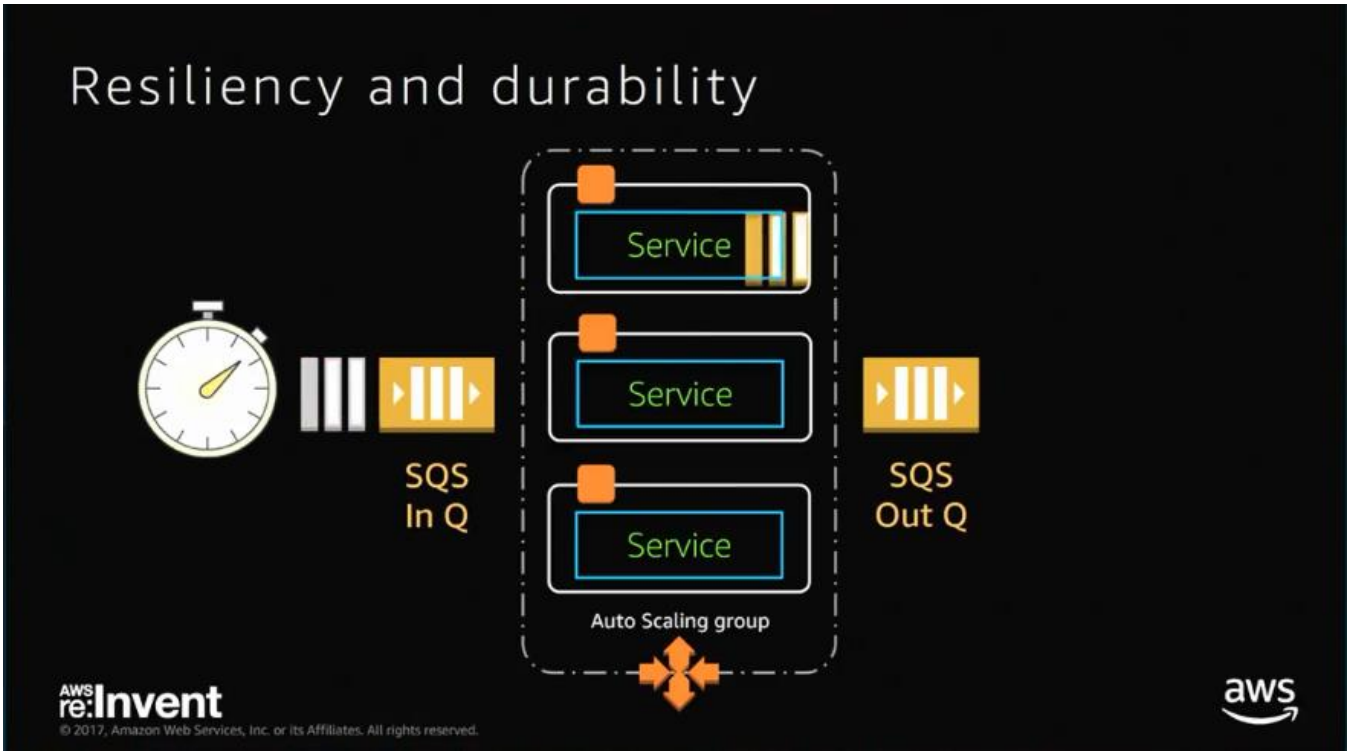
We normally run at least 3 instances in a different auto-scaling group as above so as to scale up or down when needed.



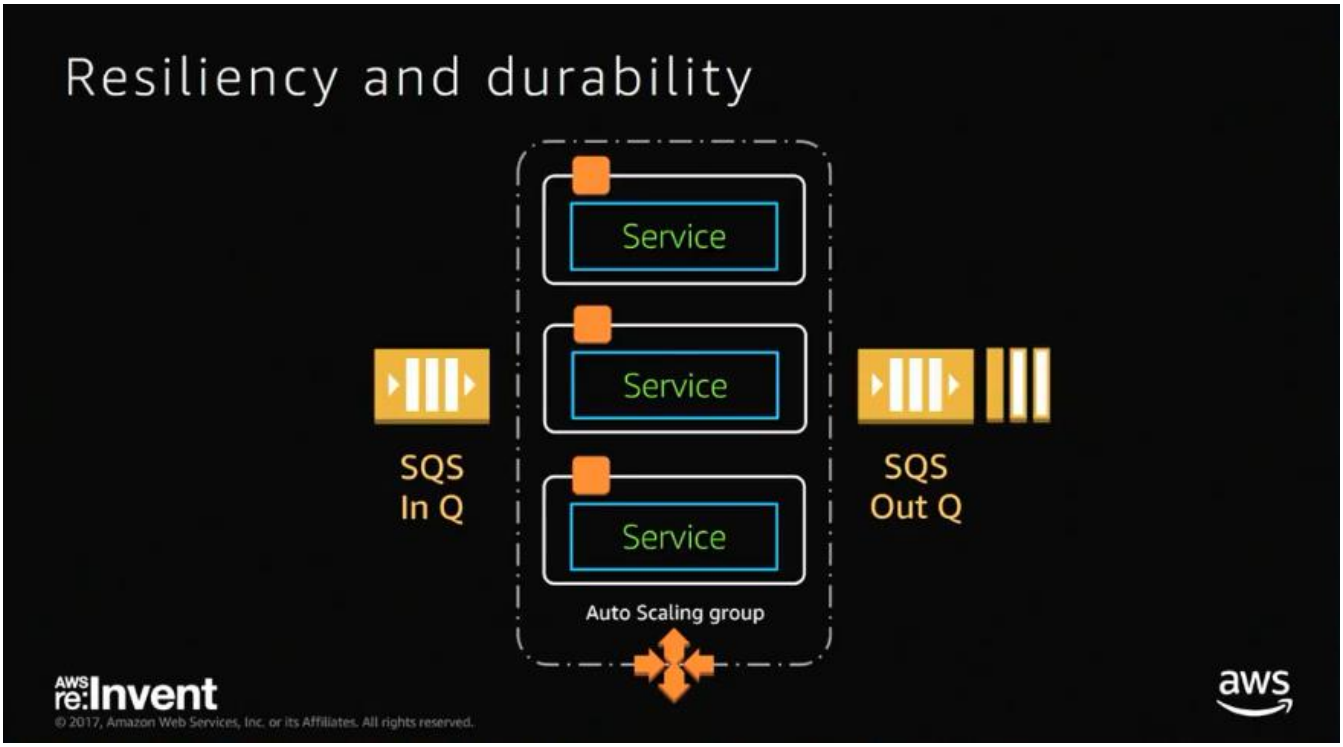


If a message processing fails due to an EC2 instance getting terminated of faulty, then the message will reappear back in the SQS In Q after a certain period of time (Visibility Timeout) if it hasn't generated a message onto the SQS Out Q queue



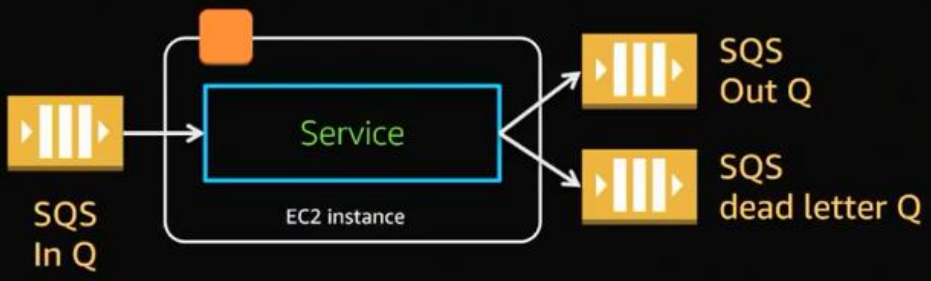


This is what happens in normal processing, an EC2 instance takes a message off the queue for processing and passing the result as a message in the Out Q queue as below

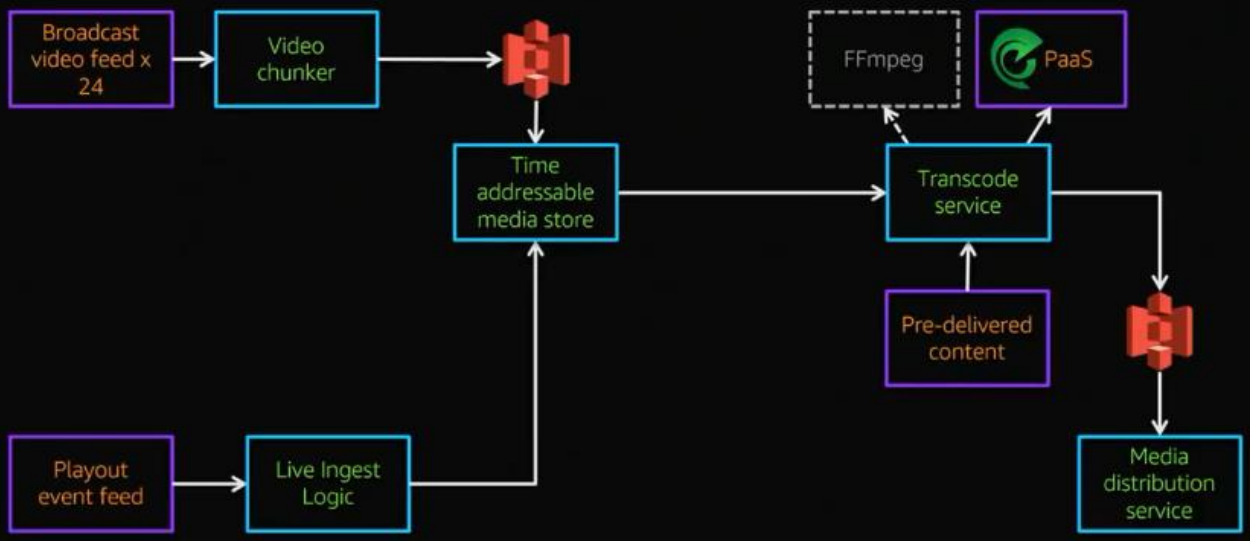


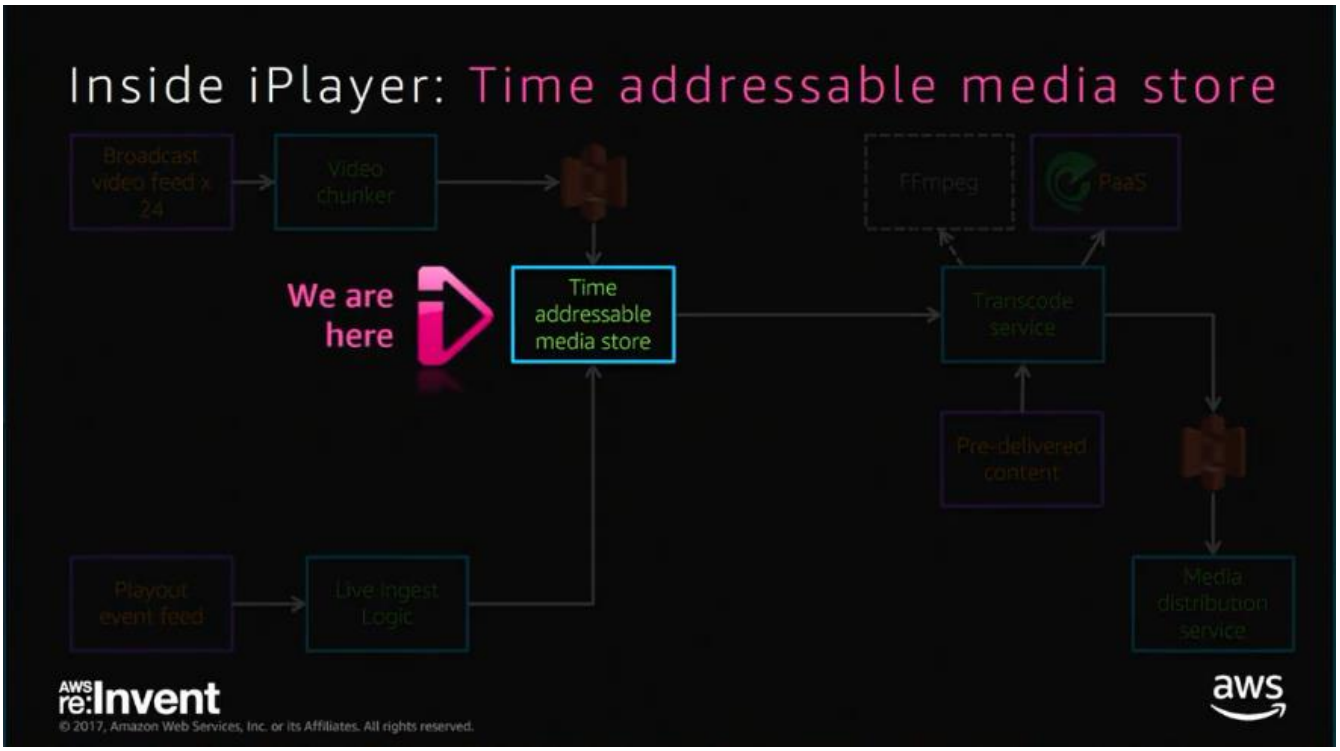
Only after the result message has been placed in the SQS Out Q do we then delete the equivalent message from the SQS In Q queue

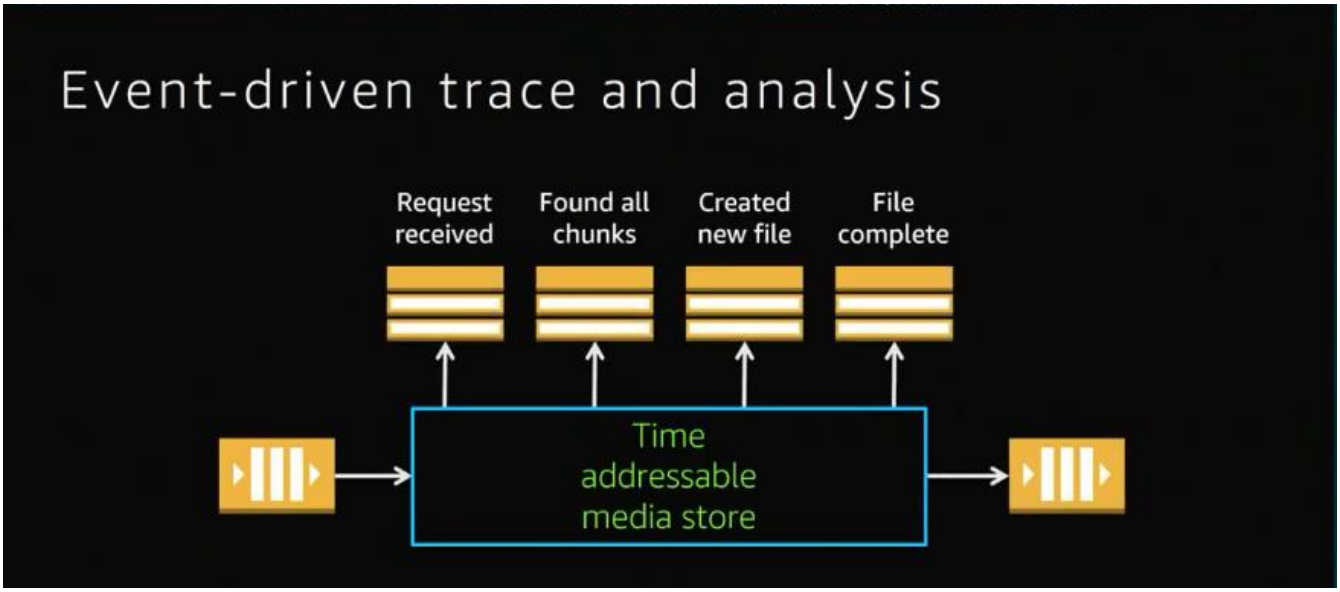
Error handling



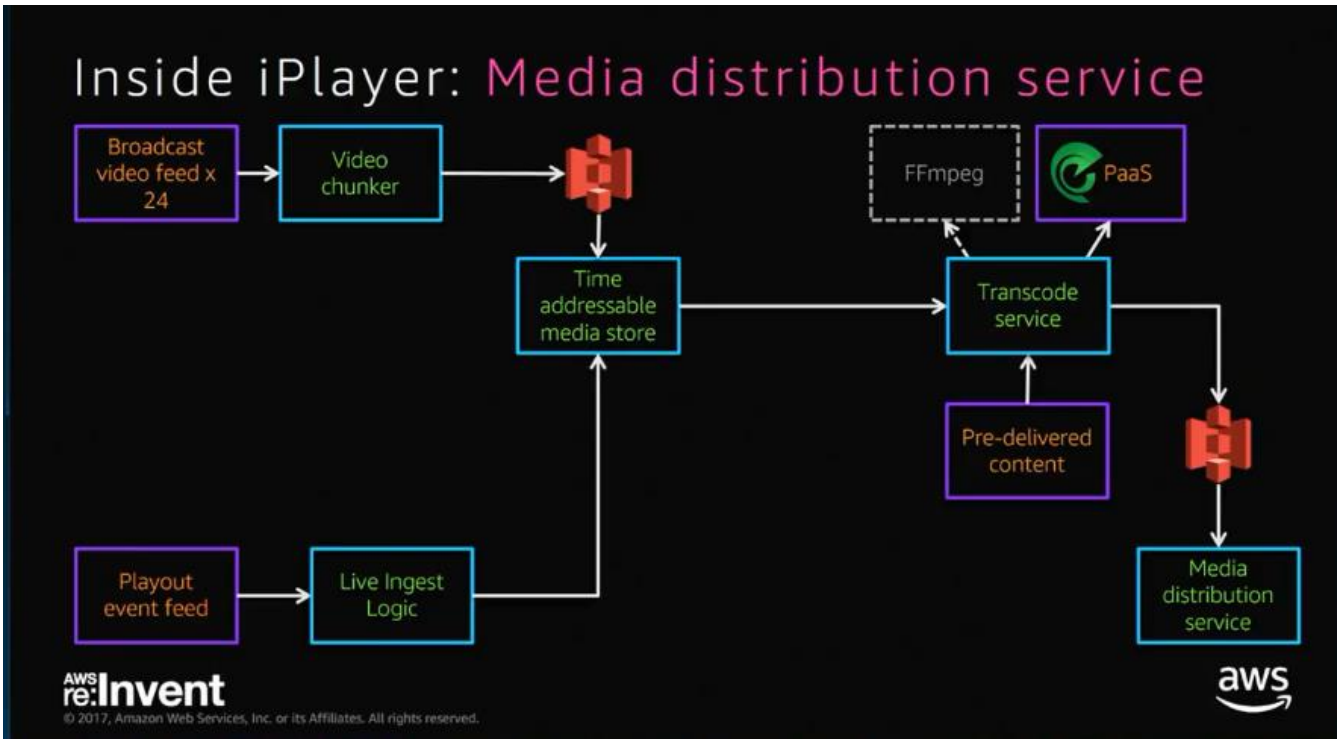
Inside iPlayer: Time addressable media store



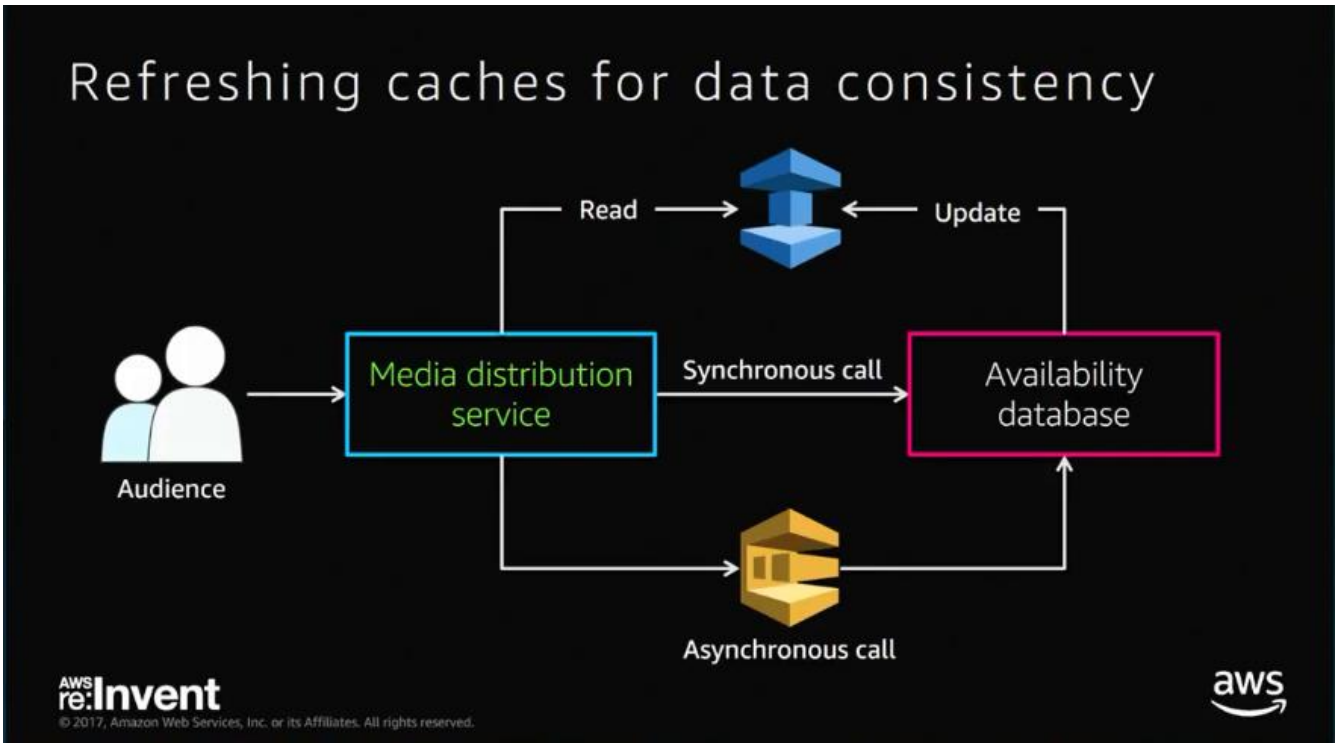




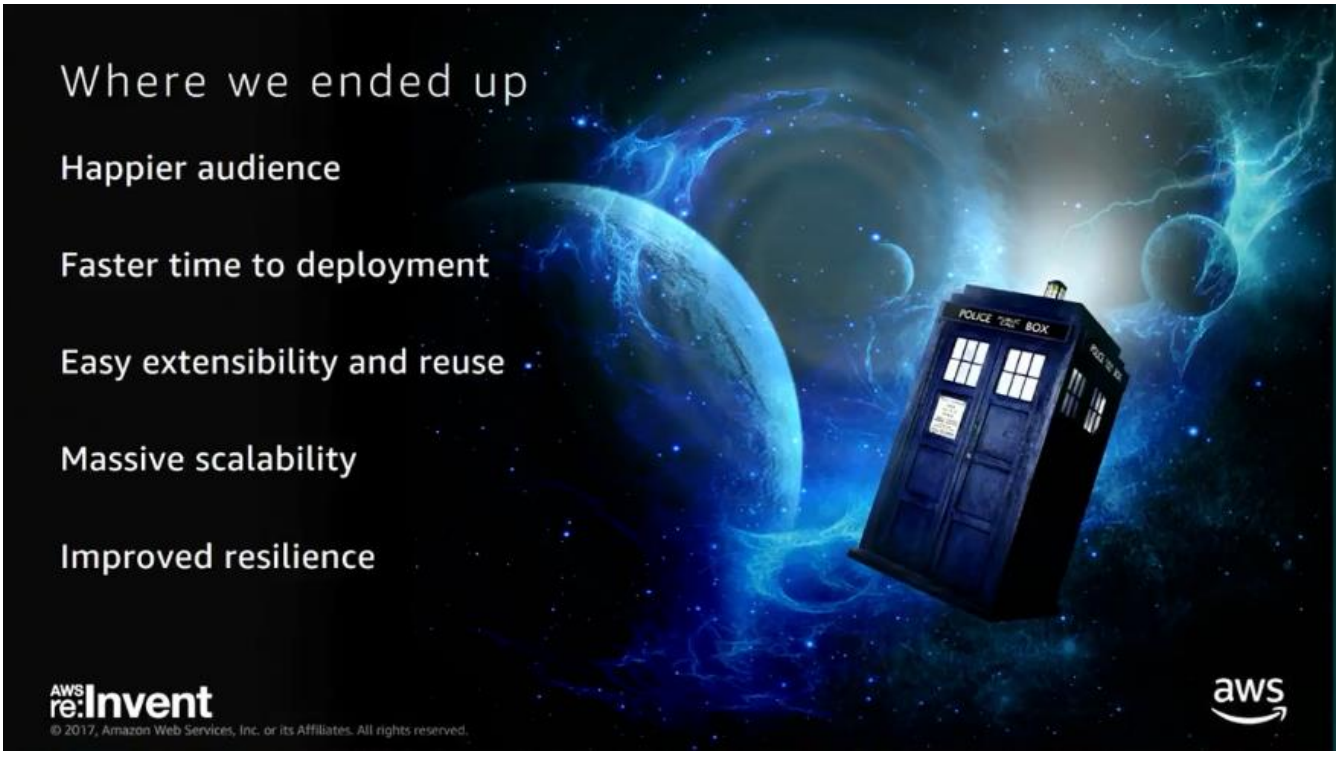
Each message entering the system gets given ***a unique correlation identifier*** that gets copied from service to service as it moves through the processing pipeline, this allows us to know what event logs are from what message. We can use a particular correlation ID to see all events or messages have been triggered from that initial message


Inside iPlayer: Media distribution service
Broadcast video feed x 24
Video chunker
FFmpeg
PaaS
Time addressable media store
Transcode service
Pre-delivered content
Playout event feed
Live Ingest Logic
Media distribution service
aws re:Invent
© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.
aws


Inside iPlayer: Media distribution service
We are all the way over here
Media distribution service
aws re:Invent
© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.
aws



Every time a user tries to view a video content, this service gets called about what ***CDN*** should be used to get the nearest copy of the video or file by querying the ***Availability database***. We also cache the result in ***AWS ElasticCache*** so that we don't need to query again for this particular user and content. If a user makes a video request and we query the cache and get a hit, if the cache period for that content is going to be stale in the next 1 minute, we will put in an async request into the queue to refresh that content when possible and reset the TTL in the cache for continuous content freshness.




LESSONS
LEARNED
Designing for idempotency
Elastic scaling is great—linear scaling is better
Reliable pipelines of microservices
Think about how you will debug your system


what's next
BBC TS&A
TECHNOLOGY STRATEGY AND ARCHITECTURE


LEARN MORE
aws.amazon.com/sqs/getting-started
aws.amazon.com/sns/getting-started
AWS re:Invent
© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.
aws