THUNDRA

CASE STUDY

Conductor uses Thundra to monitor and troubleshoot its AWS Lambda functions

CONDUCTOR

INDUSTRY
Media & Entertainment

LOCATION
CA, US

USE CASE
Troubleshooting, Debugging

VFX and Animation Industry's Transition to the Cloud, Fueled by Thundra's Automated Observability Platform

The media and entertainment industry consists of film, print, radio, and television segments. The film and television industry alone contributes more than $41 billion each year to the U.S. economy and provides nearly 2 million people with employment. Large studios in the industry, such as Technicolor and Dreamworks, have existed for decades – Technicolor alone has been in business for over 100 years. But due to their size, these companies have smaller subsidiaries that are contracted to aid in the filmmaking process, especially when it comes to creating special effects.

Production studios have created special effects for years, even though visual effects or CGI (computer-generated imagery) are fairly recent developments. Before the 1990s, most visual effects in movies consisted of stop motion and people in suits. But that soon changed. Most notably, Steven Spielberg's 1993 film "Jurassic Park" and the 1995 film "Toy Story" were some of the first few movies that used CGI and pioneered the computer-animated visual effects industry in Hollywood.

Over the last 25–30 years, production studios have been spending more and more money on visual effects, CGI, and computer animations. Studios have their own internal pipelines that work well for creating these effects, as their processes are customized and tailored to their workflows and their own systems.

Helping Offload Workloads to the Cloud

Conductor is a secure, cloud-based platform that enables VFX, VR/AR, and animation studios to seamlessly offload rendering and simulation workloads to the cloud. As the only rendering service that is dynamically scalable to meet the exact needs of even the largest studios, Conductor easily integrates into existing workflows, features an open architecture for customization, provides data insights, and implements controls over usage to ensure budgets and timelines stay on track. Conductor was established in 2015, and has been scheduling workloads on AWS since 2019.

"Conductor empowers its customers with the power of the cloud. Conductor accelerates the filmmaking industry's transition from CAPEX to OPEX, enabling its customers to save time and money." Francois Lebel, the Director of Engineering at Conductor.

The process of contracting other companies tends to be fairly straightforward. Production studios will send film scenes that need to be altered or have CGI added to several smaller studios, and then recollect their assets and images when the other studios have completed their tasks. Those scenes are then added together and post-processing commences, which integrates certain aspects (such as lighting and color) together for a similar look and feel in all the shots. This process ensures the movie progresses seamlessly as though it was created by one studio, despite the fact that it was produced by thousands of people.

Francois Lebel
Director of Engineering

> **Conductor empowers its customers with the power of the cloud. Conductor accelerates the filmmaking industry's transition from CAPEX to OPEX, enabling its customers to save time and money.**

Power of the Cloud

When a studio that had, say, 5K CPU cores on-premises realized that they needed to make changes on a project due to poor planning or other reasons, they came to the conclusion that they needed more compute resources. However, the local render farms (which are essentially regular data centers) that ran visual-effects rendering software did not have the capacity to deliver their shots on time.

That's where Conductor was able to assist. Francois Lebel, the Director of Engineering at Conductor, said, "We've helped countless studios when they had a deadline on Monday, and their local render farm would have taken them two weeks to render the project. We did it in one evening because we used the elasticity of the cloud, and we saved their project. That has happened countless times."

Conductor has taken the benefit of AWS and Google Cloud while helping their customers to render their CGI workloads. Studios of any size now have the ability to render as large of a scale as they want, as long as AWS or Google Cloud has the resources to support them.

> **We've helped countless studios when they had a deadline on Monday, and their local render farm would have taken them two weeks to render the project. We did it in one evening because we used the elasticity of the cloud, and we saved their project. That has happened countless times.**

Francois Lebel
Director of Engineering

Modern Architectures and Challenges

It's always hard to forecast how many hours of work it will take to complete a project and how much rendering will be needed, simply because the rendering process can be unpredictable. A shot may need to be redone, and that could double the spend on the rendering parts.

Prior to 2019, Conductor had workloads on Google Cloud, but they were not serverless-based. Google Cloud wouldn't work on AWS, so Conductor had to engineer something completely new rather than import workloads from Google Cloud.

Serverless on AWS looked like a perfect fit because it provided a pipeline with a number of events they had to react to. Additionally, the pipeline had to have events that would report back to the system so they could track the pipeline's progress through the Web UI without having to directly query a batch.

> **It gets very difficult to search. It's hard to pinpoint issues. It's hard to navigate to a certain time. It's very easy to get drawn among the logs, especially with Lambdas, where you've got different workloads.**

Francois Lebel
Director of Engineering

As a result, in early 2019, Conductor began using AWS serverless services and built event-driven serverless architectures hosted on AWS. They had concluded that the option they had before them was to use serverless or non-serverless workloads. It's not that they were looking to replace an existing tool with serverless; their concern was what technology and stack would most help their customers.

The challenges of monitoring distributed traces started during and after building their AWS Lambda functions. Amazon CloudWatch was helpful with the logs; however, Francois and his team didn't feel like they had a good understanding of what was going on, especially when incidents would occur. Aside from CloudWatch logs, Conductor had nothing else they could utilize to examine their Lambda functions and understand the root cause of problems. This was a crucial issue: CloudWatch made it difficult to consume and search through the logs.

Francois said, "It gets very difficult to search. It's hard to pinpoint issues. It's hard to navigate to a certain time. It's very easy to get drawn among the logs, especially with Lambdas, where you've got different workloads."

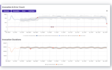
> **We once had an issue that Thundra helped us debug, right where it was. We were keeping a state across in the execution of the Lambda function. So basically, different invocations would use the same cache data, and that wasn't meant to happen. Thundra allowed us to troubleshoot and pinpoint the issue in two minutes instead of using more primitive methods of debugging to figure out what was going on.**

Troubleshooting and Debugging with Thundra

"We once had an issue that Thundra helped us debug, right where it was. We were keeping a state across in the execution of the Lambda function. So basically, different invocations would use the same cache data, and that wasn't meant to happen. Thundra allowed us to troubleshoot and pinpoint the issue in two minutes instead of using more primitive methods of debugging to figure out what was going on."

Conductor's backend team had some limited experience with maintaining Lambda functions in production, but not enough to feel confident that what they built would be stable. Additionally, when there were bugs, they couldn't pinpoint issues quickly if they were on the serverless stack.

Thundra's AI-powered anomaly detection dashboard enabled Conductor's software teams to understand the issues, errors, and timeouts in their Lambda functions with a single glance. The automated invocation, error, and cold start charts were very useful in helping them to quickly see the outliers in their data trends.



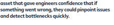
Thundra's AI-powered anomaly detection charts

In an ideal world, engineers could simply look at daily reports and be assured that everything was running smoothly. But it's not just about monitoring; the health of the software stack is crucial as well. Thundra gave confidence to Conductor's software teams with unique features, such as the "offline debugger." Francois said that the offline debugger has been a powerful asset that gave engineers confidence that if something went wrong, they could pinpoint issues and detect bottlenecks quickly.

> **The offline debugger has been a powerful asset that gave engineers confidence that if something went wrong, they could pinpoint issues and detect bottlenecks quickly.**

Heal by the First Intention

Initially, Francois's team didn't know exactly how much memory their Lambda functions would consume. Some of their Lambda functions were easy to estimate because of the work they did, but others proved more difficult. Thanks to Thundra, Francois's team were able to easily monitor the memory consumption of those Lambda functions, which before would often run out of memory.



Thundra's count and duration metric charts for automated distributed traces

After implementing Thundra, Francois's team realized that some Lambda functions always failed due to insufficient memory, so they would restart and process the record. The problem was that these functions would inevitably run out of memory again. Thundra was helpful in letting the team see and understand that those Lambda functions were growing in memory until they ran out of available memory, and then would crash as a result. So, Thundra's services allowed the team to prioritize rewriting the Lambda functions in Golang; as a result, the functions now don't even use 100 MBs of memory.

This proved to be much more efficient and faster than Python implementation. Porting the Lambda function from Python to Go was made easier by Thundra because the team knew that they had this tool behind them. If there was an error, they could quickly review their functions and, with the help of the offline debugger, detect the problem quickly.

> **It gets very difficult to search. It's hard to pinpoint issues. It's hard to navigate to a certain time. It's very easy to get drawn among the logs, especially with Lambdas, where you've got different workloads.**

Francois Lebel
Director of Engineering

Benefits of Thundra

Thundra simplified the life of the software teams in Conductor with its out-of-the-box reports, anomaly detection charts, alerts and insights, distributed traces, and our offline debugger. As a result, Conductor's engineers felt very confident in performing their workloads with their serverless stack. Thundra assured them that if anything broke or looked like it was going to break, they would be able to immediately understand the root cause.

"Thundra gives us the confidence and peace of mind that if something goes wrong, we'll know. So we no longer need to stress about it."

Thundra's trace chart of an invocation

Francois believes the Go Lambda function they deployed would have required many more code reviews if they didn't have Thundra in place, as they had previously run vanilla Lambda functions without any tools aside from CloudWatch. He also thinks that without Thundra, the approach to things like code reviews or deploying code would have been much more conservative.

Initially, saving on cost hadn't been the focus for Conductor's engineering team. But after implementing Thundra, they noticed a considerable decrease in their AWS bills. They were then able to indirectly save on cost because they were able to better provision Lambda functions since they knew exactly how much memory they would use. When the functions used more memory than expected, Thundra's offline debugger was there to help them pinpoint the reason why.

Sometimes a Lambda function would consume five times more memory than usual. Thanks to Thundra, Francois's team was able to devise a way to rewrite the function in a more efficient way. So, as a result, Conductor was able to save money with Thundra because they were able to better optimize their serverless stack overall.

Conductor has been investing more in AWS and Thundra because with their system, their new features require Lambda functions. They felt they could commit to AWS more and more due to the visibility and tools they had to troubleshoot Lambda functions.

Detect, Debug, Troubleshoot

Investing in observability early brings a couple of advantages to development teams as well as to operations or infrastructure teams. Thundra allowed engineers at Conductor to confidently invest in AWS Lambda functions. Conductor didn't have to hire engineers, as they had a proven track record of not only shipping Lambda functions to production, but also of maintaining and optimizing them over time.

> **Serverless is great on paper, but in practice, it is a rocky road unless you've got the right tools. I think Thundra has uniquely positioned itself to shine among those tools in a sea of many other competitors for observability and monitoring. Thundra has diversified itself into offering a real developer tool, the offline debugger, that can be used by the actual engineers. Now, it isn't just the DevOps or Infrastructure teams that monitor the software stack; the actual engineers that build them can do this as well.**

Overall, Thundra offered an entirely new approach to development across teams thanks to our specific expertise, and Conductor's engineers greatly benefited from the capability of Thundra's services, such as our offline debugger. Thundra was very helpful in scaling how effective the backend engineering team could be by leveraging Lambda functions with a high degree of confidence.

Lastly, Francois said, "Serverless is great on paper, but in practice, it is a rocky road unless you've got the right tools. I think Thundra has uniquely positioned itself to shine among those tools in a sea of many other competitors for observability and monitoring. Thundra has diversified itself into offering a real developer tool, the offline debugger, that can be used by the actual engineers. Now, it isn't just the DevOps or Infrastructure teams that monitor the software stack; the actual engineers that build them can do this as well."