APPDYNAMICS

Managing microservices complexity with AppDynamics

What is microservice architecture?

Microservices is a growing trend where complex applications are being broken into smaller services (hence microservices). While this is not unlike service-oriented architecture, it's being done for a different reason. The movement away from waterfall development and ITIL operations towards agile development and DevOps are causing siloed teams to be broken apart.

These new smaller teams, usually organized around business capability, take complete responsibility for software development and work together on an entire service lifecycle (dev, test, ops). In line with Conway's law*, this results in developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms. They release independently from one another, yet these microservices must work together to deliver the application functionality. This adds a lot of complexity, especially as teams use various technologies to deliver their independent services.

Microservices speed up the delivery of software features. They allow for a rapid iteration cycle, allowing the software to flex to business changes and experimentation. This ultimately yields a product which incrementally changes and is improved over time, versus monolithic software which focuses on large and heavy software releases.

Ideally this splitting up into services is organized around business capability. Such services take a broad-stack implementation of software for that business area, including user-interface, persistent storage, and any external collaborations. Consequently, the teams are cross-functional, including the full range of skills required for development: user experience, database, and project management.

Challenges in managing microservices-based applications

The days of Java- and Oracle-based applications are no longer; microservices are built with a large degree of variability. Today's microservices are built with multiple languages, many backends, and a large number of Web service-based API calls, each of which must perform well for the service to deliver the required data. Monitoring the end-to-end performance and latency of business transactions made out of microservices, usually communicating asynchronously, becomes a difficult task. This complexity and diversity creates unique challenges with these new architectures.

"Microservices are simpler, developers get more productive, and systems can be scaled quickly and precisely."

Gartner

KEY FEATURES

– Core APM: Transaction-tracing across multiple languages and Web services calls (which are the glue of the microservices architecture)

– EUM: Measuring the end-user experience from the user through components

– Database: Monitoring the large degree of variable backends

– Analytics: To handle custom logs and transactional data

* Any organization that designs a system (defined broadly) will produce a design whose structure is a copy of the organization's communication structure. – Melvyn Conway, 1967

Business impact of suboptimal microservice management

Microservices are being implemented for agility between the business, IT, and other groups like marketing to more quickly drive feedback into the software lifecycle. Each service typically chooses its own language, data store, and associated technology to meet the demands of the service that must be delivered. The services are all API-driven, allowing for reuse of the services, which can be constructed into many different applications. This enables the business to better leverage its data and capabilities easily into more distinct products and offerings. This level of complexity is far beyond what occurs in today's already complex applications. Isolating issues and enabling teams to be responsible for their specific services is impossible without the right level of visibility and consistency across the various implemented technologies.

AppDynamics provides comprehensive support for microservices

All of these services are owned by different teams, and are built on different languages with different backends, yet they must all work together properly to deliver a seamless application to the user. AppDynamics' breadth and depth of technology support allows us to measure the business transaction, made of multiple microservices usually communicating asynchronously, from the user through each service, and baseline each metric collected. This ensures that each service and team can independently become operationally proficient, along with allowing their business to measure and understand each application and each user interaction within the application.

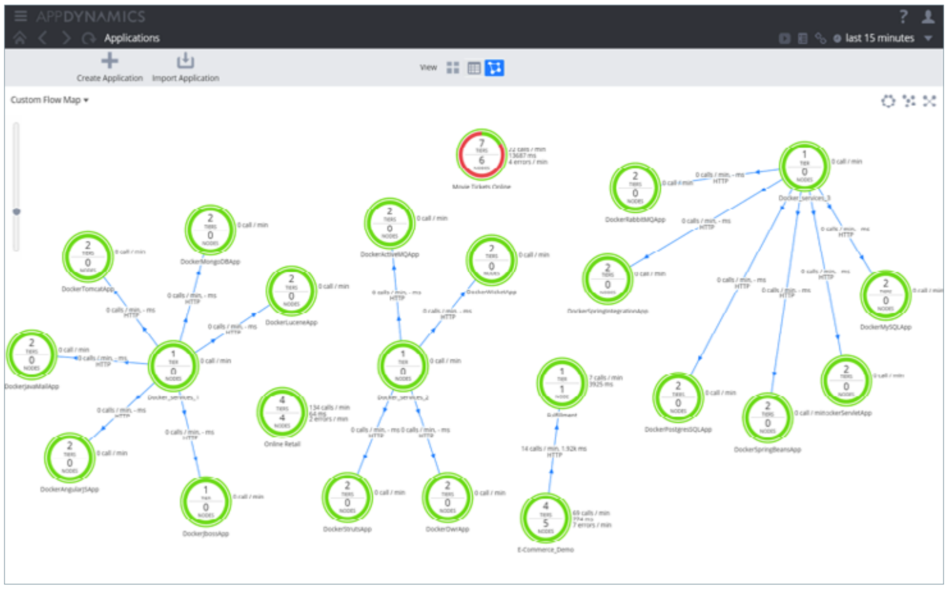

Figure: Visualizing microservices with AppDynamics

KEY BENEFITS

– Service endpoint views allow teams to measure specific microservices performance

– Transaction tracing and topology allow teams to work together to resolve issues

– Support of modern languages and an open-ended platform allow for easier metric and log ingestion across any technology selected by each team

Try it FREE at appdynamics.com