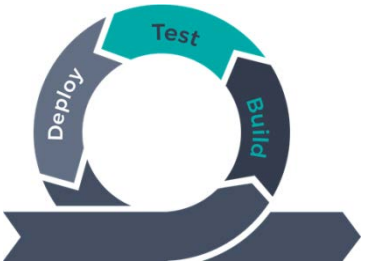
Using NeoLoad for Microservices, API, and Component Testing

A Neotys Whitepaper

TABLE OF CONTENTS

INTRODUCTION



Neotys has always had the approach that automating everything that can be automated, in order to allow faster performance testing cycles; that's why we are constantly improving the NeoLoad Performance Testing platform. The next logical step in the relentless search for application performance leads companies to consider making other strategic shifts as well, such as making a move toward Microservices architectures, API testing and component-level testing.

THE EMERGENCE OF MICROSERVICES ARCHITECTURES

Microservice architecture (also referred to as just 'microservices'), has slowly emerged as a "go-to" way of creating fast and robust enterprise applications by Development teams in recent years. Microservices architectures break down complex tasks into smaller processes that operate independently and communicate through language-agnostic APIs.

Microservices are a suite of independently deployable, modular services where each service runs one, unique process. Microservices communicate through a well-defined method or set of methods to deliver a specific capability or achieve a business goal.

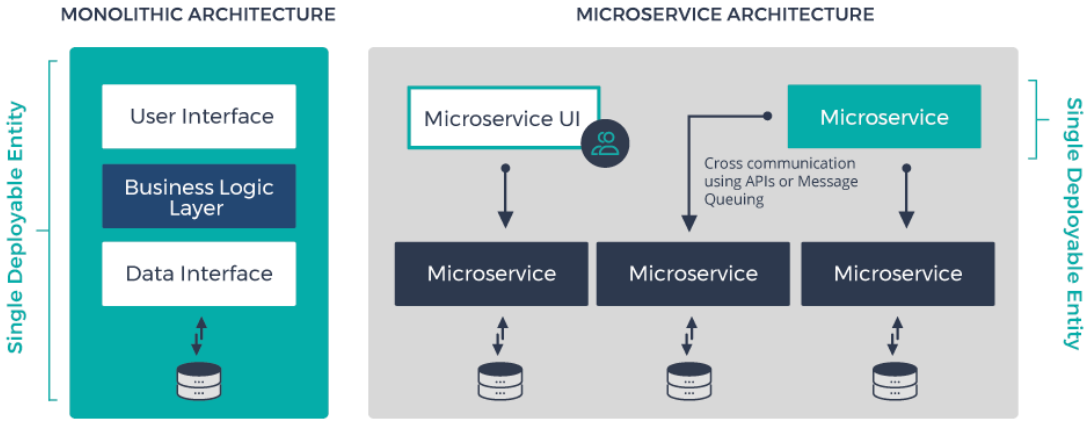


Figure 1 - Monolithic vs Microservices Architectures

- Microservices can be broken down into multiple component services so that each of these services can be built, tested, deployed, and redeployed independently - without compromising application integrity
- Microservices are usually organized around delivering business capabilities and priorities, by using cross-functional teams rather than handing off to specialist teams (e.g. database, network infrastructure, server) for related services
- Microservices architecture is an evolutionary design and is well suited for systems where you can't fully anticipate the types of devices that may one day be accessing your application (e.g. services delivered to IoT devices)
- At their most basic level, microservices receive requests, process them, and generate a response

Why Microservices?

When monolithic legacy applications build on traditional architectures can no longer be scaled to meet heavy loads, or maintained easily, companies are turning increasingly to microservices architectures. Thanks to its inherent scalability, microservices architectures are ideal when deploying across a range of platforms and devices - web, mobile, Internet of Things (IoT), and wearable technology like smart watches. And because they are self-contained in the sense that all related services dependencies (e.g. database) are included in the microservice, it can be easier for DevOps teams to take ownership of meeting performance goals, and deliver against them.

Microservices have definitely entered the mainstream. The list of large Enterprises who have moved to using Microservices is impressive - Netflix, eBay, Amazon, Twitter, PayPal, SoundCloud, and many others have evolved from monolithic applications approaches to Microservices architecture.

The Performance Testing Challenges Inherent in Microservices Architectures

Microservices architectures come with their own set of complexities and concerns, particularly around performance testing. Because of the nature of these architectures, a solid testing plan requires new approaches to confirm proper operation and continued availability under heavy load or in the face of resource failure. While it can be a multi-step process to characterize a scenario of what "real world" usage of an entire microservice will look like in terms of performance under load, DevOps organizations can develop and implement a comprehensive load testing plan to make certain their microservices applications perform well under "real-world" load and failure scenarios.

UNDERSTAND THE LOAD PERFORMANCE AND SCALABILITY OF APIS

Application Programming Interfaces (APIs) are central to modern data-driven applications, and are becoming even more so. At the same time, Representational State Transfer (REST) has become more popular than ever, especially for Web-based applications. Most leading websites (Twitter, Google, Flickr, and many others) use REST APIs. And with the emergence of RESTful API frameworks like Swagger, and of a robust load testing platform like NeoLoad, API testing has become a lot more straightforward.

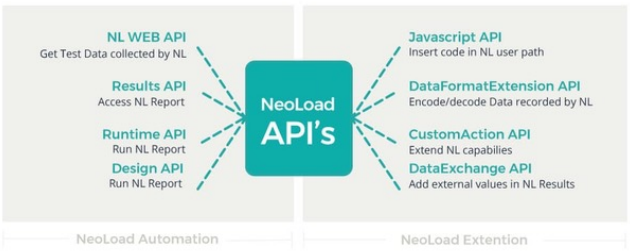


Figure 2 - NeoLoad Provides a Rich Set of RESTful APIs for Integration

APIs are often overlooked when designing performance testing approaches, sometimes because users don't interact directly with them. However, testing web services and APIs is critical to producing quality software that is robust and performs well under real-world load scenarios. Many aspects of API testing are similar to traditional end-to-end load testing, but since there is no GUI involved there are some special considerations.

API Testing - More Important than Ever to Your Business

Including API tests in your app development process provides several benefits the businesses that get passed down to customers in the form of better quality applications with higher performance. Including API tests in your overall SDLC improves test quality, test coverage, and enables test reuse. From a performance point of view, a slow API can decimate perceived user experience when an application is deployed into production. But, these issues may only emerge under heavy loads. That's why it's very important to be able to characterize the API's load behavior early in the SDLC, when performance issues are far cheaper and easier to address. In fact, you probably want to consider testing your APIs right alongside the application itself.

Because of the importance of APIs in providing integration and openness for your applications, API test automation has the potential of significantly accelerating the testing and development process. It ensures consistency in testing and enables continuous improvement in application quality. Increasingly, testing and development teams are moving towards API test automation and integrating their testing tools with CI frameworks like Jenkins.

The Business Benefits of Load Testing APIs

- Gain insight into user experience early in the SDLC, even before GUI is developed
- Simulate traffic from the cloud or on-premise and develop realistic load scenarios
- Visualize the impact of "at scale" load on your servers

THE ROLE OF COMPONENT TESTING

At Neotys we say, "Test early...and often", meaning that load testing should ideally be done both earlier in the SDLC, more often, and at different levels of granularity. Component testing is similar in approach to API testing, but specific methods are often required to ensure full test coverage of all components and application layers.

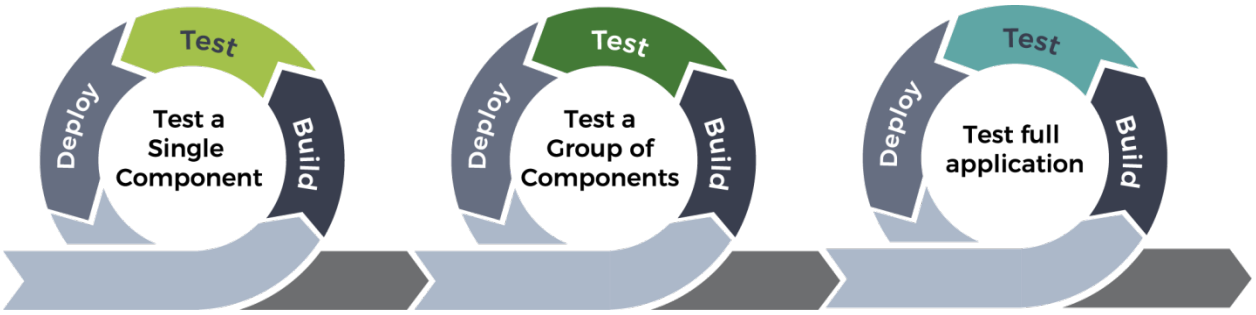


Figure 3 - Component Testing in a Shift Left Approach

Testing a single component , then testing integrated groups of components together (e.g. for a UI) ensures that these components can still function properly when you get to more-complex test and load scenarios later in the SDLC. Then, you can transition with confidence to doing more traditional system-wide performance testing in a pre-production environment.

In order to test as soon as possible in a "shift left" testing approach, load testing tools must be able to generate traffic with specific protocols. Web APIs are not always available for internal microservices, specific protocols may be mandatory to tests some services, messaging protocols (e.g. JMS, MQTT, KAFKA, MQ Series, ProtoBuf, etc.) or to validate the behavior of Databases (SQL load generation).

Component testing is also increasingly achieved in a Continuous Integrat ion (CI) and Continuous Delivery (CD) approach. The objective of CI is essentially defensive; performance testers and developers test on a very regular and automated basis and are intent on having trends and comparing their results to a reference test, typically to know if their current modifications on the code is having a deteriorating or improving effect on performance.

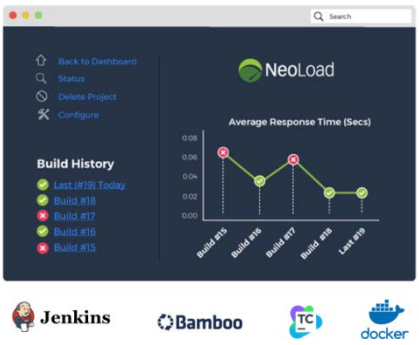


Figure 4 - NeoLoad has several out-of-the-box integrations with popular CI frameworks

Focus on Application Performance

Take advantage of today's best practices by "Shift Left" testing (testing early and often in your SDLC). As more DevOps and Agile-focused teams adopt this approach, what they are really doing is changing the granularity of what is tested at different stages of the SDLC, moving toward deploying the application into a pre-production environment. Additionally, you should consider how and when testing APIs, Microservices and components should be a part of your overall performance testing plan.

The actual users of your applications are likely to interact with the application through many services, components, databases, networks and maybe even third-party connected apps, you need to make sure their actual end-user experience will be satisfactory without frustration or failure. Using load testing to make sure that a particular API, component or microservice performs well is, after all, just a means to achieving this goal, not the ultimate goal itself.

MAKE PERFORMANCE TESTING A COMPETITIVE EDGE

To achieve both speed and quality for performance testing in a DevOps environment, businesses will need to empower testers and developers to take ownership of application performance, build more-engaging applications for end-users, automate the testing process wherever possible, and transform the way applications are designed and how their performance is tested.

Beating the competition to the market will depend on the developers and testers working as a team, as key players in transforming the business, or in creating new opportunities, by creating faster and better developer and tester practice. DevOps teams will lead the way by delivering better and faster insights through applications built and tested to outperform the competition.

- **Be confident**: Cover more test cases, ensure professional grade quality, make testers trusted partners
- **Support Agile:** Make performance testing a key element of your Agile software delivery
- **Achieve DevOps:** Make performance testing a continuous process across Dev, QA and Ops

ABOUT NEOTYS

The success of your digital strategy relies on your ability to deliver fast and reliable software, regularly. Creating great software quickly, using an optimized performance testing process is your competitive advantage – Agile and DevOps are part of the solution.

Neotys has over 12 years of development investment into NeoLoad – the performance testing platform designed to accelerate Agile and DevOps processes. It's built by engineers who recognized that in order to achieve their own Agile adoption objective, they needed to create a product that could facilitate superior load and performance testing continuously. The end result – up to 10x faster test creation and maintenance with NeoLoad.

We truly believe that the Performance Engineer can become the critical application performance partner providing the best testing coverage while respecting the cadence of the Continuous Delivery process. As performance becomes the responsibility of the wider team, continued delivery of an optimized performance testing platform is what drives our work every day.