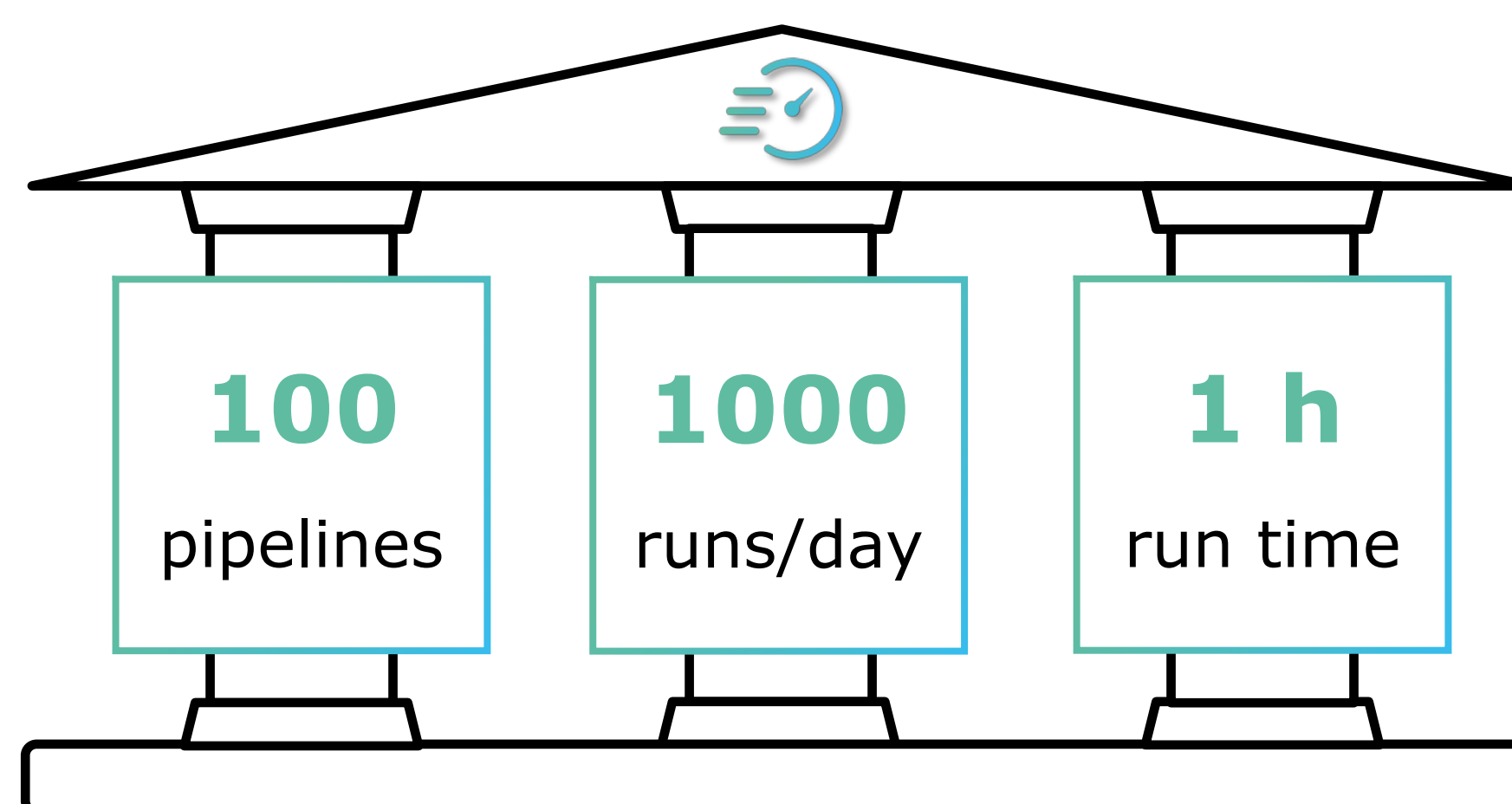
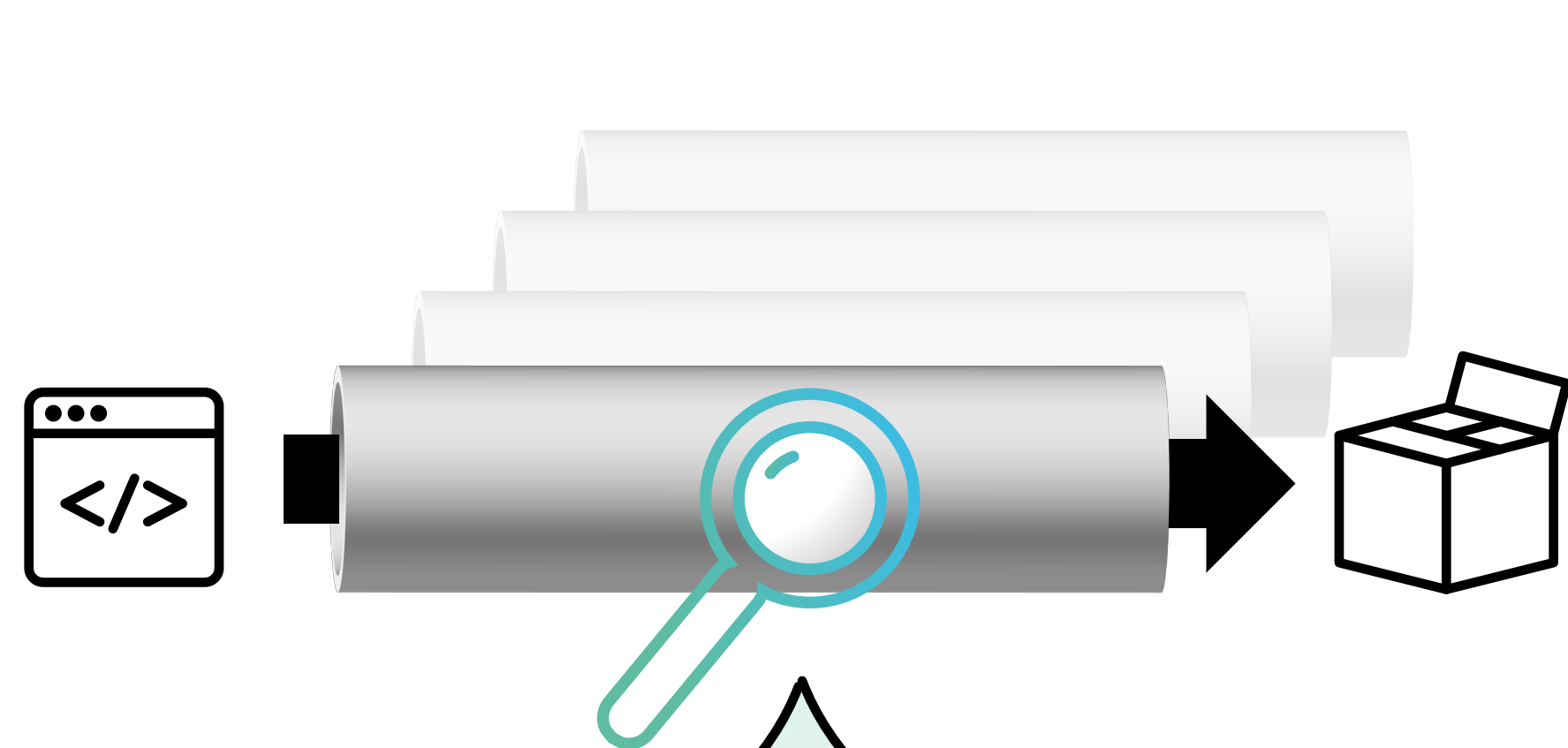


Henning Schulz, André van Hoorn, Dušan Okanović, Stefan Siegl, Christoph Heger, Alexander Wert, Tobias Angerstein, Alper Hidiroglu, Manuel Palenga, Christoph Zorn, Vincenzo Ferme, Alberto Avritzer

# ContinuITy: Automated Load Testing in DevOps

## Load Testing vs. DevOps



Typical **CI/CD pipelines** in DevOps have huge dimensions, short execution times, and are **automated**. However, load tests need much time, resources, and expertise.

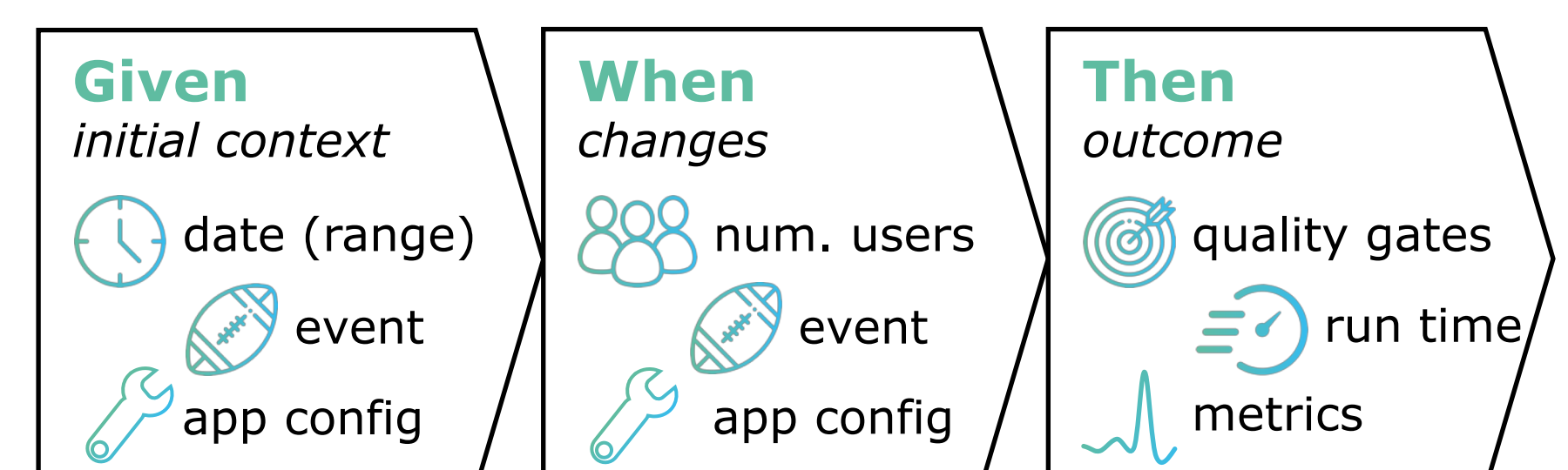
With ContinuITy, we make load testing **easy to use** and automatically generate **time-efficient** and **resource-efficient** load tests, which fit into CI/CD pipelines.

## Describing Tests in Natural Language

**Given** *Black Friday* and service carts  
**when** varying the *CPU cores*  
**then** ensure *response time < 1 s*.

A user creates a load test description in a template-based natural language – the **behavior-driven load testing** language. In this way, the level of expertise required to define a load test is reduced.

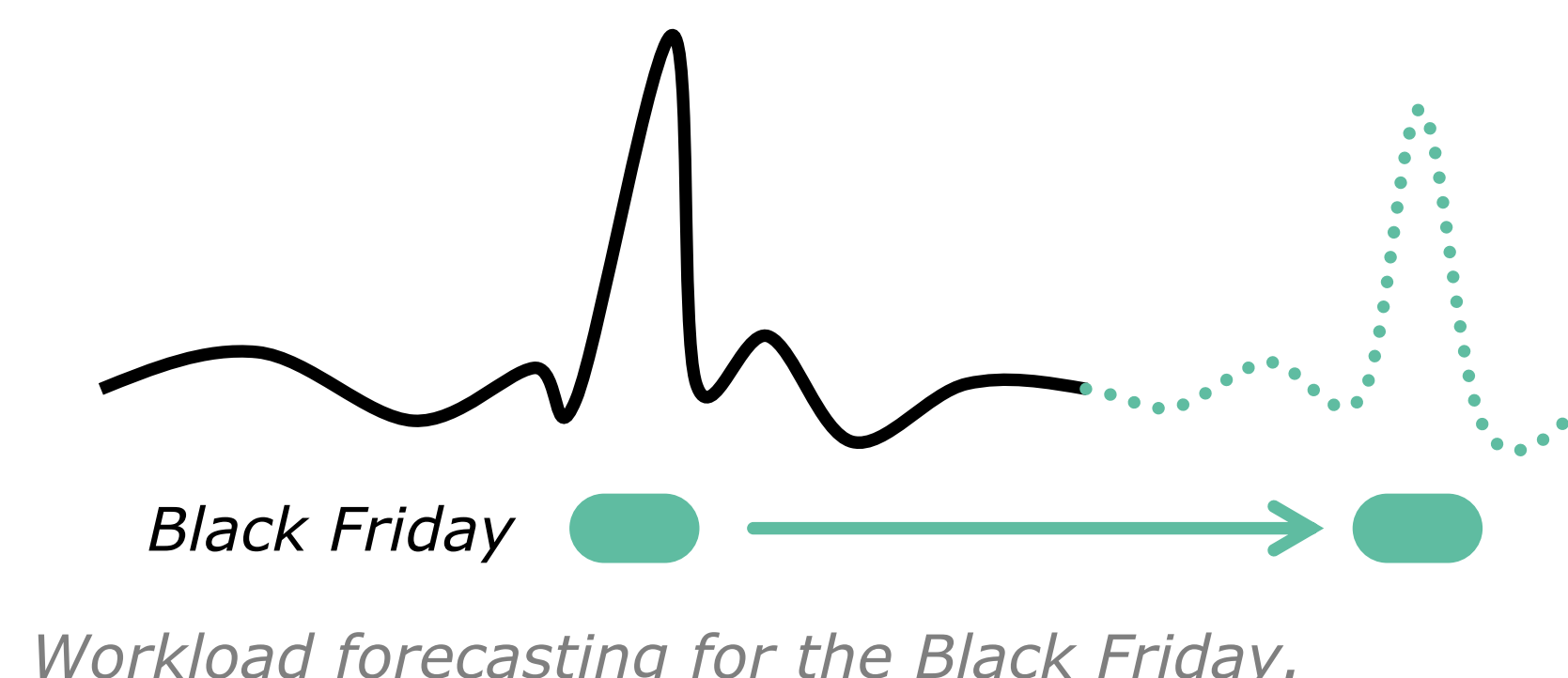
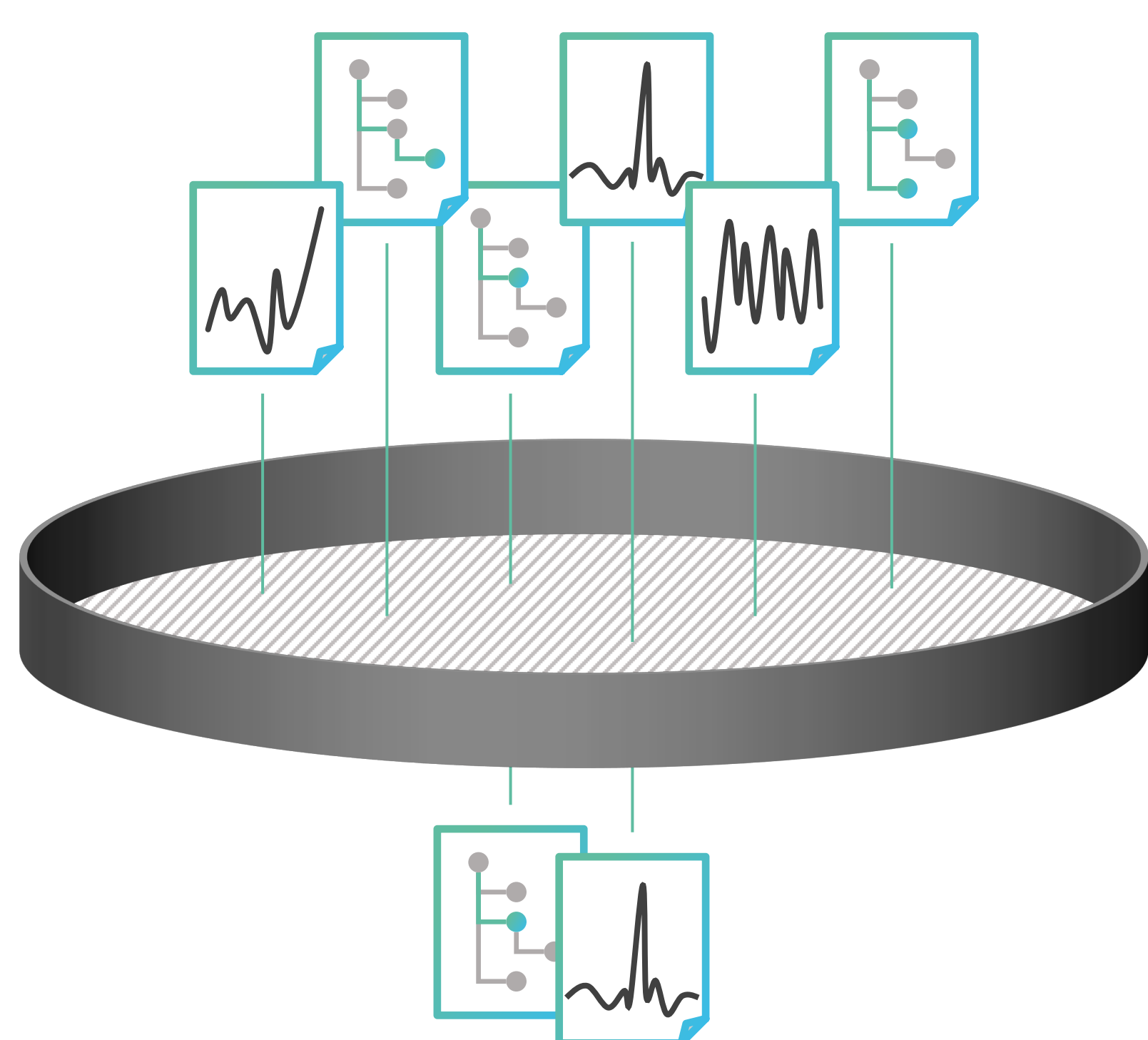
A test consists of *given*, *when*, and *then* clauses, defining the initial test and workload context, the changes made to the initial context, and the expected outcome. This is based on behavior-driven (functional) testing.



Elements of behavior-driven load tests.

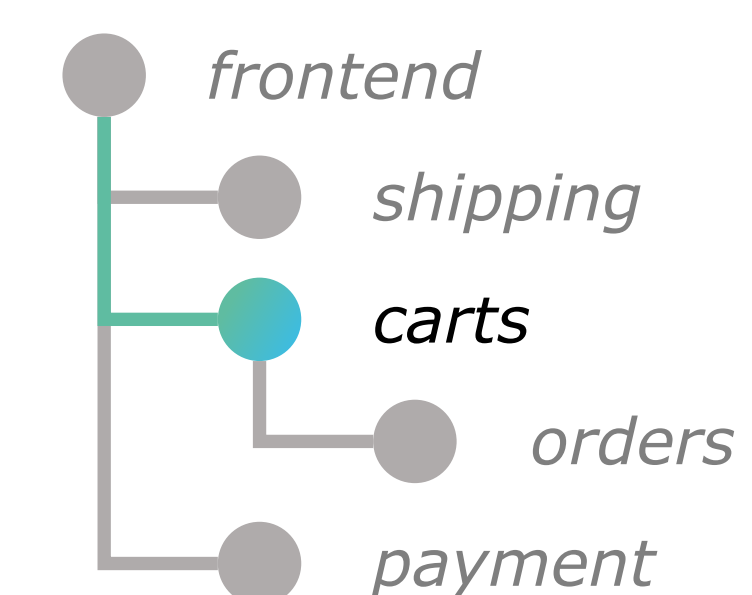
We add a concept of **events**, which influence the workload, e.g., *Black Friday*.

## Tailoring to Relevant Contexts & Services



Workload forecasting for the Black Friday.

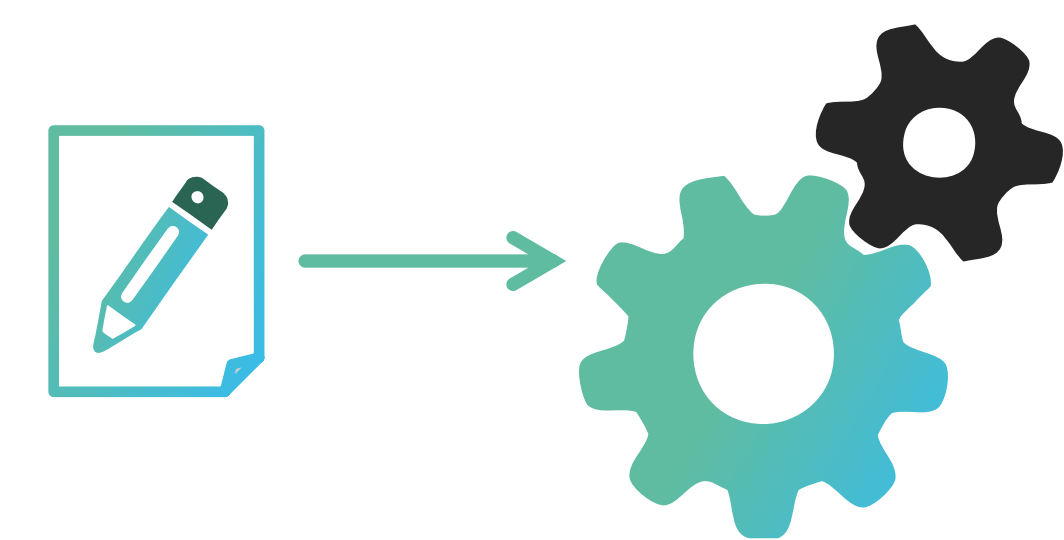
Based on the test description, we generate a load test that is tailored to the specified test and workload context. We use **time series forecasting** to predict the future workload, respecting influencing events such as *Black Fridays*.



Tracing a request to the service carts.

Also, we restrict the test's workload to the relevant **microservices** (e.g., *carts*) by tracing the individual requests through the application. Hence, the required test infrastructure is minimized.

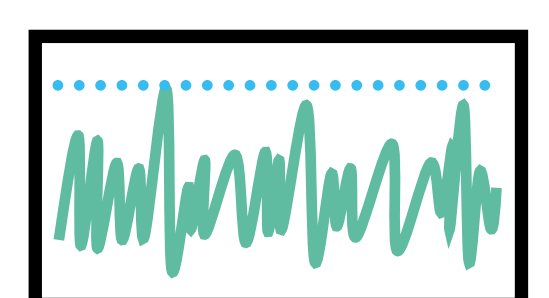
## Automated Execution & Analysis



We **parameterize** the generated load test automatically by using pre-defined annotations. It is then automatically executed by **BenchFlow** for all explored configurations, e.g., *CPU cores*.

We apply **regression analysis** on the results for detecting and locating regressions in multiple application versions.

The maximum response time was 942 ms.



Natural-language report with a chart.

Finally, we provide a **report** about the test results based on the user concern defined in the behavior-driven load test.

## References

**Towards Automating Representative Load Testing in Continuous Software Engineering**  
 Henning Schulz, Tobias Angerstein, and André van Hoorn  
 Companion Proceedings ICPE 2018

**Behavior-driven Load Testing Using Contextual Knowledge – Approach and Experiences**  
 Henning Schulz, Dušan Okanović, André van Hoorn, Vincenzo Ferme, Cesare Pautasso  
 Proceedings ICPE 2019

**A Declarative Approach for Performance Tests Execution in Continuous Software Development Environments**  
 Vincenzo Ferme and Cesare Pautasso  
 Proceedings ICPE 2018

**Concern-driven Reporting of Software Performance Analysis Results**  
 Dušan Okanović, André van Hoorn, Christoph Zorn, Fabian Beck, Vincenzo Ferme and Jürgen Walter  
 Proceedings ICPE 2019