Traceon

Intelligente Testautomatisierung und Analyse verteilter, asynchroner Softwaresysteme unter realistischen Bedingungen

Programm / Ausschreibung	IWI 24/26, IWI 24/26, Basisprogramm Ausschreibung 2026	**Status**	laufend
Projektstart	01.02.2026	**Projektende**	31.01.2027
Zeitraum	2026 - 2027	**Projektlaufzeit**	12 Monate
Projektförderung	€ 61.492		
Keywords			

Projektbeschreibung

Ziel dieses Projekts ist die betriebsdatengetriebene Generierung realitätsnaher, relevanter und tiefgehender End-to-End-Testszenarien für Microservice-Systeme. Dafür sollen Erkenntnisse aus dem Betrieb systematisch genutzt, interne Zusammenhänge berücksichtigt und domänenspezifisches Wissen eingebunden werden. Mit QALIPSIS haben wir in den letzten Jahren eine Plattform entwickelt, die Last-, Leistungs- und End-to-End-Tests für verteilte wie auch monolithische Systeme ermöglicht (siehe „Stand der Technik“). Auch die geplante Weiterentwicklung im Rahmen dieses F&E-Projekts wird in QALIPSIS integriert.

Die folgenden drei Entwicklungsziele bilden die Grundlage für den geplanten neuen Ansatz:

1. Generierung realistischer Testszenarien
Klassische Testansätze basieren oft auf künstlichen Szenarien, die reale Nutzung nur unzureichend abbilden. Dadurch bleiben relevante Fehler – insbesondere durch komplexe Service-Interaktionen – unerkannt. Auch die Priorisierung im Testprozess erfolgt meist auf Annahmen statt auf Nutzungsdaten.
Testfälle sollen daher reale Nutzung widerspiegeln. Tracing- und Monitoring-Daten dienen dazu, typische wie auch sicherheitsrelevante Abläufe zu identifizieren und in repräsentative Tests zu überführen.

Durch die gezielte Ableitung realistischer Testszenarien auf Basis von Betriebsdaten (z. B. aus Tracing-Logs) kann sichergestellt werden, dass:
- häufige, sicherheitskritische und geschäftsrelevante Abläufe gezielt geprüft werden,
- Randfälle aus dem Betrieb in die Tests einfließen,
- Tests aus realen Nutzungsprofilen generiert werden und so ihre Aussagekraft steigt,
- sowie Wartung und Pflege effizienter sind, da Tests direkt aus aktuellen Abläufen abgeleitet werden.

Ziel ist es, eine realitätsnahe Testabdeckung zu erreichen, die sich am tatsächlichen Nutzerverhalten orientiert und somit

eine höhere Fehlerentdeckungsrate bei gleichzeitig geringerer Testredundanz ermöglicht.

Um realistische Testszenarien auf Basis von Betriebsdaten ableiten zu können, sind folgende Entwicklungsinhalte erforderlich:
- Erkennung typischer Nutzungsmuster aus echten Betriebsdaten: Entwicklung eines Verfahrens zur Analyse von Tracing- und Monitoring-Daten zur Identifikation typischer Nutzungspfade und Prozessverläufe im Echtbetrieb. Dabei werden auch transportierte und gespeicherte Daten berücksichtigt; sensible Informationen werden verschleiert, um Datenschutz und Vertraulichkeit zu wahren.
- Abstraktion und Generalisierung von Nutzungsszenarien: Entwicklung einer Methodik, um aus beobachteten Nutzungsinstanzen abstrakte, übertragbare Testszenarien zu erzeugen, die wiederverwendbar und systematisch variierbar sind.
- Systematische Überführung in ausführbare Testszenarien: Konzeption eines Mechanismus zur Übersetzung der gewonnenen Szenarien in strukturierte, automatisiert ausführbare Tests, etwa als Sequenzen von API-Aufrufen oder Events.
- Validierung der Testszenarien hinsichtlich ihrer Realitätsnähe: Entwicklung eines Evaluationsverfahrens, das die erzeugten Tests hinsichtlich Nähe zur realen Nutzung und Testwirksamkeit bewertet – ggf. in Rückkopplung mit Domänenexpert:innen.

2. Validierung interner Systemwirkungen
Klassische Tests auf Ebene der Schnittstellen erfassen oft nur das sichtbare Verhalten eines Microservice-Systems, nicht aber die internen Abläufe. Gerade in verteilten Systemen mit vielen Services treten Fehler auf, die äußerlich unauffällig bleiben – etwa durch falsche Datenspeicherung, nicht ausgelöste asynchrone Prozesse oder fehlerhafte Validierungen. Solche Probleme zeigen sich oft erst verzögert oder in Kombination mit anderen Ereignissen und sind mit einfachen Blackbox-Tests schwer zu erkennen.

Ohne tiefere Prüfungen werden interne Inkonsistenzen, Seiteneffekte oder fehlerhafte Datenweitergaben leicht übersehen – kritisch besonders in sicherheits- oder geschäftsrelevanten Anwendungen, wo falsche Datenverarbeitung gravierende Folgen hat.

Eine automatisierte, systematische Validierung der Wirkung von Eingaben im Gesamtsystem – auch über mehrere Services hinweg – ist daher entscheidend für Qualität und Zuverlässigkeit moderner Microservice-Architekturen.

Um sinnvolle und korrekte Überprüfungen zu generieren sind folgende Entwicklungsinhalte erforderlich:
- Erkennung systeminterner Abhängigkeiten: Zusammenhänge im System müssen automatisch erkannt werden – etwa Aufrufketten, Servicekommunikation oder Datenflüsse über Systemgrenzen. Auch indirekte Abhängigkeiten wie persistente Zustände sind einzubeziehen.
Ableitung überprüfbarer Wirkungszusammenhänge: Aus diesen Abhängigkeiten sind überprüfbare Erwartungen abzuleiten, z. B. „Tritt Ereignis A auf, muss Zustand B folgen“ oder „Ein POST erzeugt einen DB-Eintrag, der später per GET abrufbar ist“.
- Erweiterung der Testauswertung um Systemverhalten: Die Auswertung darf nicht nur Schnittstellenantworten prüfen, sondern auch interne Wirkungen und Seiteneffekte wie Statusänderungen, DB-Operationen oder Nachrichten auf asynchronen Kanälen.
- Nutzung bestehender Analysewerkzeuge und Frameworks: Bestehende Infrastrukturen zur Laufzeitanalyse (z. B. Tracing, Observability, Logs) sollen genutzt und erweitert werden, um die nötigen Informationen zur Wirkungsanalyse zu gewinnen.

- Einbettung in das bestehende Testframework: Die erkannten Überprüfungen sind so ins QALIPSIS-Testframework einzubetten, dass sie automatisch mit generierten Tests verknüpft und kontinuierlich ausgeführt werden.

3. Domänenwissen gezielt nutzen

Automatisch abgeleitete Tests und Zusammenhänge basieren meist auf statistischer oder heuristischer Analyse von Betriebsdaten. Solche Verfahren erkennen Muster, garantieren aber weder semantische Bedeutung noch domänenspezifische Korrektheit. In der Praxis entstehen daraus zwei Probleme:
- Fehlinterpretationen: Korrelationen werden fälschlich als Kausalitäten interpretiert. So entstehen Tests mit falschen Annahmen über das Systemverhalten, die bei legitimen Änderungen fehlschlagen (fragile Tests).
- Mangel an Relevanz: Automatisch generierte Tests können aus Domänensicht belanglos oder redundant sein – sie prüfen Szenarien, die zwar auftreten, aber weder sicherheitsrelevant noch geschäftskritisch sind.

Domänenexpert:innen – z.B. Entwickler:innen oder Tester:innen – verfügen über implizites Wissen über das konkrete System, seine Funktionsweise, Anforderungen und Randbedingungen. Dieses Wissen ist oft nicht dokumentiert, aber entscheidend für die sinnvolle Bewertung und Steuerung von Tests.
Durch die gezielte Einbindung solcher Expert:innen können:
- falsche Zusammenhänge erkannt und aussortiert,
- wichtige Szenarien hervorgehoben oder ergänzt und
- die Verständlichkeit und Akzeptanz automatisch generierter Tests erhöht werden.

Dies verbessert nicht nur die Qualität der Tests, sondern erhöht auch das Vertrauen der Anwender:innen in den Testprozess insgesamt.

Um Expertenwissen nutzbar zu machen sind folgende Entwicklungsinhalte erforderlich:
- Verständliche Darstellung automatisch abgeleiteter Zusammenhänge: Die vom Test-Tool erkannten Zusammenhänge (z. B. Kausalitäten zwischen Service Aufrufen oder Constraints) müssen so aufbereitet sein, dass Domänenexpert:innen sie nachvollziehen können. Dazu zählen die Reduktion auf relevante Informationen und eine benutzerfreundliche Visualisierung, etwa als Sequenzdiagramme, Graphen oder Regeln in natürlicher Sprache.
- Erklärbare Testfälle und Constraints: Auch die generierten Testfälle und Prüfregeln müssen so dargestellt sein, dass ihre Intention klar erkennbar ist. Nur so können Expert:innen fundiert bewerten, ob ein Test korrekt, relevant oder irreführend ist.
- Interaktive Feedbackmechanismen: Es braucht intuitive Mechanismen, mit denen Expert:innen Bewertungen und Korrekturen vornehmen können – etwa durch Ablehnen, Anpassen oder Bestätigen von Vorschlägen. Rückmeldungen sollten mit minimalem Aufwand möglich sein (z. B. Klick, Kommentar, Bewertungsskala).

Das geplante FFG-Projekt erweitert den Stand der Technik grundlegend, da es erstmals eine automatisierte, trace-basierte Generierung von End-to-End-Tests für komplexe verteilte Systeme ermöglicht. Traceon nutzt Monitoring- und Tracing-Daten aus Produktions- oder Staging-Umgebungen, um Nutzungspfade, Aufrufsequenzen und Systemverhalten automatisch zu extrahieren. Daraus werden semantisch angereicherte Testmodelle abgeleitet, die nicht nur Einzelaufrufe prüfen, sondern auch komplexe, kausale Zusammenhänge abbilden. Auf dieser Basis entstehen vollständige QALIPSIS-Testfälle, die realistische Szenarien über mehrere Services, Datenbanken und Kommunikationskanäle hinweg validieren.

Diese Kombination aus automatisierter Analyse, Constraint-Ableitung und Testfallgenerierung ist neuartig und in bestehenden Tools nicht verfügbar.

4. Geplante, messbare Ergebnisse nach Projektabschluss

- Erstmalige Kombination aus automatisierter Analyse, Constraint-Ableitung und Testfallgenerierung.
- Experimentelles System zur automatisierten Generierung realistischer End-to-End-Tests aus Tracedaten.
- Interaktives Feedbackmodul zur Einbindung von Expert:innen in den Generierungsprozess.
- Innovative Werkzeuge zur Darstellung, Bewertung und Korrektur erkannter Systemzusammenhänge.
- Validierte Testfälle, die reale Nutzungsszenarien abdecken und interne Verarbeitung gezielt prüfen.
- Dokumentierte Methodik zur Integration in bestehende Qualitätssicherungsprozesse mit QALIPSIS.
- Automatisierte Erkennung relevanter Testpfade aus realer Nutzung → realistische Tests ohne manuelles Mapping von Businesslogik.
- Validierung von Systemverhalten über Servicegrenzen hinweg, auch in heterogenen Architekturen (Microservices, Legacy, IoT).
- Integration in CI/CD-Pipelines durch automatische Generierung ausführbarer QALIPSIS-Testfälle.
- Nachvollziehbarkeit durch visuelle Darstellung abgeleiteter Aufrufmuster und zugehöriger Tests.
- Skalierbare Ausführung mit QALIPSIS, auch unter hoher Last mit realistischen Nutzerprofilen.

5. Erwartete Effekte

- Reduktion des manuellen Testaufwands um bis zu 70 %: Testfälle entstehen automatisch aus Nutzungspatterns und Tracedaten.
- Erhöhung der realitätsnahen Testabdeckung um 50–80 %: Testfälle spiegeln tatsächliche Nutzungspfade und Systemverhalten wider.
- Schnellere Fehlererkennung und -behebung (Faktor 2–3), da reale Pfade automatisch geprüft und typische Fehler früh erkannt werden.
- Kürzere Time-to-Market durch automatisierte Testgenerierung im Rahmen kontinuierlicher Entwicklungsprozesse.

Ergebnis des FFG-Projektes ist ein ausführlich getesteter, evaluierter Prototyp. Optimierungen und die Serienreifentwicklung erfolgen erst nach Abschluss des FFG-Projektes.

Projektkoordinator

- AERIS IT Solutions GmbH

Projektpartner

- Software Competence Center Hagenberg GmbH