Vom Java-Quellcode zum Sequenzdiagramm durch statische Codeanalyse und Large Language Models

Studiengang: MAS-IT Software Engineering

Ein hybrides Verfahren aus statischer Analyse und Large Language Models (LLMs) erzeugt entlang der im Quellcode abgebildeten Aufrufketten ein kontextangereichertes Modell und leitet daraus kompakte Sequenzdiagramme ab. Der Mehrwert liegt nicht nur im Diagramm, sondern auch im generierten Modell, das prüfbar, rekonstruierbar und wiederverwendbar ist.

Softwaredokumentation altert schnell, und in vielen Legacy-Systemen stellt der Quellcode die einzige verlässliche Wissensquelle dar. Reverse Engineering mittels automatisch erzeugter Diagramme (z.B. Sequenzdiagramme) bietet hier Unterstützung, doch bestehende Tools liefern meist rein syntaktische Darstellungen, die semantisch arm und ohne Bezug zur fachlichen Logik bleiben.

Vor diesem Hintergrund untersucht diese Masterthesis, inwieweit ein hybrides Verfahren aus statischer Analyse und Large Language Models (LLMs) eingesetzt werden kann, um diese Limitationen zu lösen. Entlang der im Quellcode vorhandenen Aufrufketten soll die statische Analyse, deterministisch ein Modell erzeugen, welches anschliessend durch LLM-Inferenzen mit Kontext angereichert und transformiert wird.

Leitfrage: «Inwieweit kann ein hybrides Verfahren aus statischer Analyse und LLM eingesetzt werden, um aus Java-Quellcode entlang der im Quellcode abgebildeter Aufrufketten ein Modell zu generieren, das anschliessend genutzt wird, um mittels LLM kompakte und verständliche Sequenzdiagramme zu erzeugen?»

Zunächst entsteht durch statische Analyse ein Aufrufbaum, der die im Quellcode abgebildeten Aufrufketten strukturell präzise erfasst. Aufgrund begrenzter Kontextfenster wird der Aufrufbaum methodenweise/knotenweise durch LLM-Inferenzen zu listenartigen, semantisch angereicherten NL-Zusammenfassungen verarbeitet. Dabei setzt das LLM, Marker für weiterführende Methodenaufrufe (Kinder im Baum). Diese Marker ermöglichen schlussendlich die verlustfreie Komposition der Knoten zu einem konsistenten Modell, welches Struktur und Bedeutung der Aufrufketten explizit macht. Das resultierende Modell dient als belastbare Grundlage für die automatische Generierung kompakter und verständlicher PlantUML-Sequenzdiagramme. Zugleich ist es prüfbar, rekonstruierbar und wiederverwendbar.

Die Evaluation zeigt eine vollständige Abdeckung bei der Komposition von Aufrufbeziehungen (Recall = 1,0; TP = 107, FN = 0), was auf eine stabile Funktionsweise der Marker hindeutet. Die generierten Knotenzusammenfassungen erreichen in der Expert:innenbewertung rund 92,7%, während die generierten Sequenzdiagramme durchschnittlich 88,9% erreichen. Besonders kompakte Diagramme (kurze Aufrufkette) erhielten Spitzenwerte bis 100%, während grössere und komplexere an Übersichtlichkeit verlieren.

Limitationen bestehen in der Konsistenz der statischen Analyse, zwischen SootUp und JavaParser und in systematischen Bewertungsverzerrungen des LLM-Judges bei kurzen Methoden. Zudem zeigt sich möglicherweise eine eingeschränkte Skalierbarkeit bei sehr grosse Szenarien sowie eine hohe Prozessdauer von etwa 30 Sekunden je Knoten. Handlungsempfehlungen umfassen u.a. Caching-Mechanismen, selektive Komposition des Modells und die Verarbeitung von Jimple als alternative Quelle. Die Befunde sind vielversprechend, müssen aber durch eine grössere empirische Erhebung validiert werden.

Zentrale Erkenntnis: Nicht das Sequenzdiagramm selbst, sondern das markerbasierte, kontextangereicherte Modell ist das entscheidende Artefakt. Es verbindet Faktentreue der statischen Analyse mit der semantischen Verdichtung durch das LLM, bleibt deterministisch rekonstruierbar und ist über die Diagrammgenerierung hinaus vielseitig einsetzbar.

Christophe Viette