Integration und Visualisierung von Softwaremetriken für Java in VS Code

Diplomand

Raphael Geiser

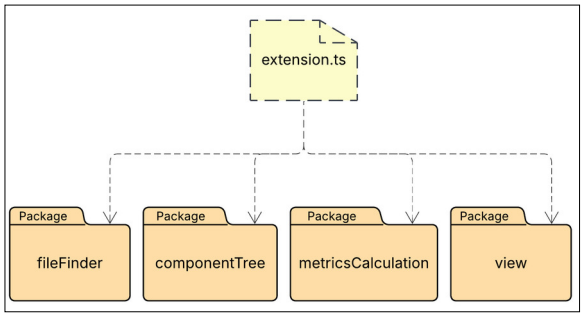
Ausgangslage: In der modernen Softwareentwicklung geht mit wachsender Projektgrösse und verteilten Teams schnell der Überblick über den Quellcode verloren. Zunehmende Komplexität und mangelnde Transparenz führen zu erhöhter Fehleranfälligkeit, steigenden Wartungskosten und erschwerter Qualitätssicherung. Code-Metriken bieten einen objektiven Ansatz, um Software hinsichtlich Struktur, Lesbarkeit und Wartbarkeit messbar zu machen und gezielt Verbesserungen einzuleiten. Sie ermöglichen die frühzeitige Identifikation potenzieller Problemstellen, die Formulierung überprüfbarer Richtlinien und die Unterstützung bei Refactorings. Viele existierende Werkzeuge sind jedoch nur als eigenständige Anwendungen verfügbar und nicht in den Entwicklungsprozess eingebunden, was ihre regelmässige Nutzung erschwert. Visual Studio Code (VSC) ist ein häufig genutzter Editor, bietet bislang jedoch keine integrierte Lösung zur Berechnung von Metriken für Java-Projekte. Eine entsprechende Erweiterung verspricht daher hohen praktischen Nutzen.

Vorgehen / Technologien: Ziel dieser Arbeit war die Entwicklung einer VSC-Erweiterung, die gängige Softwaremetriken automatisch berechnet und direkt im Editor visualisiert. Die Implementierung erfolgte vollständig in TypeScript unter Einsatz der VSC-API sowie des performanten Tree-sitter-Parsers, der Java-Quellcode effizient in Syntaxbäume überführt. Für die interne Datenstruktur wurde das Composite-Pattern genutzt, um die hierarchische Organisation von Paketen, Klassen und Methoden abzubilden. Die modulare, erweiterbare Berechnung der Metriken basiert auf dem Visitor-Pattern, unterstützt durch Generics zur flexiblen Gestaltung der Rückgabetypen. Die Architektur umfasst vier Hauptschritte:

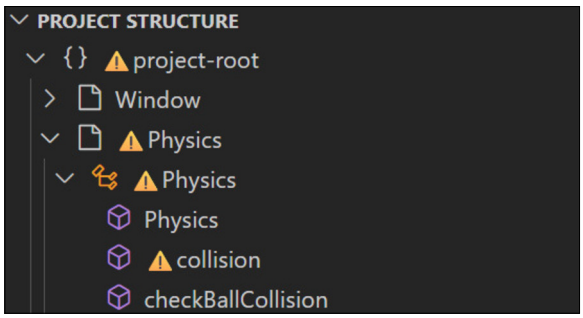
- 1 Auffinden und Parsen der relevanten Dateien
- 2 Aufbau des Komponentenbaums
- 3 Berechnung der Metriken mittels spezialisierter Visitoren
- 4 Darstellung der Resultate über die Tree-View-Schnittstelle von VSC

Ergebnis: Im Rahmen der Arbeit wurden vier zentrale Metriken umgesetzt: Lines of Code (LOC), Cyclomatic Complexity (CC), Lack of Cohesion of Methods (LCOM) sowie ausgewählte Halstead-Metriken. Die Resultate erscheinen in einer interaktiven, baumartigen Ansicht, die sowohl Projektstruktur als auch direkte Navigation zu relevanten Codestellen ermöglicht. Kritische Werte werden visuell hervorgehoben und an übergeordnete Elemente vererbt, sodass sich Problemstellen effizient lokalisieren lassen. Über Konfigurationen können Nutzende bestimmen, welche Metriken berechnet werden sollen, wodurch sich die Lösung flexibel an projektspezifische Anforderungen anpassen lässt. Die entwickelte Erweiterung integriert sich nahtlos in den Entwicklungsprozess, reduziert Analyseaufwand, erhöht Transparenz und unterstützt Entwickler:innen gezielt bei Refactorings. Die implementierte Architektur bildet eine solide Grundlage für Erweiterungen, etwa durch projektspezifische Grenzwerte, zusätzliche Metriken oder erweiterte Visualisierungen einschliesslich zeitlicher Metrikverläufe. Damit trägt die Arbeit zur nachhaltigen Verbesserung der Codequalität in Java-Projekten bei und eröffnet vielfältige Möglichkeiten für Forschung und Weiterentwicklung.

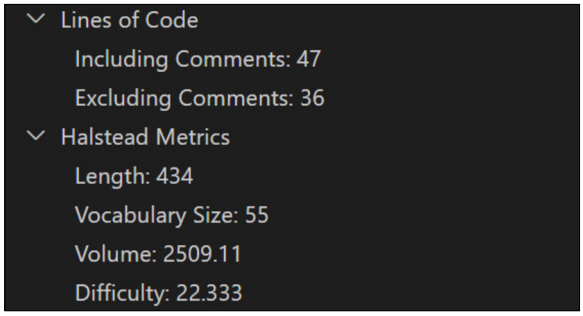
Übersicht über die wichtigsten Komponenten der entwickelten Software.
Eigene Darstellung



Interaktive Darstellung der Projektstruktur. Das Ausrufezeichen weist auf eine mögliche Problemstelle hin.
Eigene Darstellung



Darstellung der Ergebnisse der Metriken zu der angeklickten Komponente in der obigen Ansicht.
Eigene Darstellung



Referent
Prof. Dr. Norbert Frei

Korreferent
Adrian Kretz

Themengebiet
Ingenieurinformatik