Partielle Transformation von C++-Programmcode in einen Java-AST zur Erkennung von Code Smells: Erste Ergebnisse

Marcel Steinbeck, Tobias Nolte
Universität Bremen
marcel, tnolte@informatik.uni-bremen.de

Zusammenfassung

Code Smells sind Strukturschwächen innerhalb von Programmcode, die einen negativen Einfluss auf die Verständlichkeit und Wartbarkeit eines Softwaresystems haben können. Basierend auf den Ergebnissen vergangener Studien, wurden in den letzten Jahren verschiedene Werkzeuge zur automatischen Erkennung von Code Smells entwickelt. Eine beliebte Technik ist dabei die Verwendung abstrakter Syntaxbäume (ASTs) zur Erhebung verschiedener Metriken. Insbesondere für die Programmiersprache Java wurde bereits eine Vielzahl solcher Werkzeuge entwickelt. Tools zur Erkennung von Code Smells in C++-Projekten sind jedoch rar, was nicht zuletzt daran liegt, dass nur wenige geeignete AST-Generatoren für C++ verfügbar sind. Im Folgenden präsentieren wir CPP2Spoon, ein Werkzeug zur partiellen Transformation von C++-Programmcode in einen Java-AST.

1 Einleitung

Um die steigende Komplexität großer Softwaresysteme handhaben zu können, sind in den letzten Jahren verschiedene Designrichtlinien zur Qualitätssicherung von Software entwickelt worden. Code Smells jedoch brechen mit diesen Designrichtlinien und können einen negativen Einfluss auf die Verständlichkeit und Wartbarkeit eines Systems haben. Fowler et al. präsentierten 22 verschiedene Code Smells sowie deren Definition und möglichen Einfluss auf ein Softwaresystem. Ferner wiesen sie darauf hin, Code Smells frühzeitig zu entfernen, um so die innere Qualität zu verbessern [2]. Basierend auf diesem Vorschlag wurden verschiedene Werkzeuge zur Erkennung und Beseitigung von Code Smells entwickelt. Bekannte Vertreter sind unter anderem PMD, SonarQube und JDeodorant. Viele dieser Werkzeuge nutzen einen AST, um bestimmte Metriken automatisch zu erfassen, zu kombinieren und mit geeigneten Schwellwerten zu vergleichen. Werkzeuge zur automatischen Erkennung von Code Smells in C++-Projekten sind jedoch selten, was unter anderem daran liegt, dass nur wenig geeignete AST-Generatoren für C++ zur Verfügung stehen.

Um dieser Problematik entgegenzutreten, haben wir mit der Entwicklung von CPP2Spoon begonnen, einer Programmbibliothek, die unter Verwendung von SrcML C++-Programmcode in einen Spoon-Java-AST übersetzt, um diesen mit dem ebenfalls von uns entwickeltem Tool ISmell auf Code Smells zu untersuchen.

2 Stand der Forschung

Marinescu et al. [3] haben die Plattform iPlasma entwickelt, mit derer sowohl Java als auch C++-Programme auf Code Smells untersucht werden können. Mithilfe von MCC werden die strukturellen Informationen der C++-Quelltexte ausgelesen und in einer Menge von zusammenhängenden ASCII-Tabellen gespeichert. Diese Tabellen werden anschließend in das MEMORIA-Metamodell überführt, welches ebenfalls für die Darstellung von Java-Quelltexten verwendet wird. Um die Wiederverwendbarkeit und Verständlichkeit der Analysen zu gewährleisten, wurde im Rahmen des iPlasma-Projekts die Sprache Sail zur strukturellen Analyse des MEMORIA-Metamodells entwickelt.

Alexandru et al. [5] haben ein Framework für die statische Analyse von C- und C++-Projekten entwickelt, das verschiedene Anfragen zum Erkennen einfacher Code Smells unterstützt. Die Anfragen werden auf einer sogenannten *fact database* ausgeführt, die aus einzelnen Binärdateien für jede zu untersuchende Quelltextdatei besteht. Die Datenbank unterstützt vier Arten von Daten: lexikalische, syntaktische, Typ- und Präprozessor-Daten. Die Daten werden mit einem *heavyweight tolerant analyzer* aus dem Programmcode extrahiert, welcher für das Framework unter Verwendung des Parsers Elsa entwickelt wurde.

3 Von SrcML zum Spoon-AST

Um C++-Programme in ein für uns geeigneten Java-AST zu übersetzen, haben wir uns bei der Entwicklung von CPP2Spoon für das SrcML-Framework [1] entschieden. SrcML selbst ist eine XML-Repräsentation für Programmcode, die den zu analysierenden Quelltext mit einem leichtgewichtigen AST annotiert. Neben C und C++ wird auch C# und Java unterstützt. Das Kommandozeilenprogramm srcml übersetzt alle Eingaben automatisch in eine XML-Datei. Aufgrund des einfachen Aufbaus des

SrcML-ASTs müssen die C++-Quelltextdateien lediglich geparsed und nicht vollständig kompiliert werden, was nicht nur einen hohen Durchsatz beim Verarbeiten der Eingabedaten ermöglicht, sondern auch nicht vollständigen oder kompilierfähigen Programmcode unterstützt. Dieser Vorteil geht jedoch mit der Problematik einher, dass einige, durch zum Beispiel Kompiler, zur Verfügung stehende Informationen in SrcML nicht vorhanden sind. Dies betrifft insbesondere die Typinformation von Variablen und das Auflösen von Typreferenzen. Da diese Informationen jedoch für die Analyse einiger Code Smells erforderlich sind und wir bereits eine Reihe von Code Smell-Erkennern auf Basis des Spoon-Java-ASTs realisiert haben, haben wir uns für die Transformation der SrcML-Ausgabe in diesen AST entschieden, um so bereits vorhandene Tools wiederverwenden zu können.

Spoon ist eine Programmbibliothek zur Analyse und Transformation von Java-Programmcode [4]. Der bereitgestellte AST umfasst viele nützliche Eigenschaften, wie beispielsweise das Auflösen von Typreferenzen und Traversieren ganzer Programmpakete. Spoon wird seit 2006 von mehreren Entwicklern aktiv gepflegt und bietet neben der reinen Erzeugung von ASTs auch einen umfangreichen Mechanismus zum Filtern einzelner AST-Knoten an. Dieser ist besonders gut zur Erhebung diverser Metriken geeignet. Der AST ist aufgeteilt in eine Schnittstelle – eine Sammlung von Interfaces – und eine Implementierung. Die Schnittstelle ist in sich geschlossen, was bedeutet, dass jeder durch ein Interface repräsentierte AST-Knoten nur Abhängigkeiten zu anderen Interfaces der Schnittstelle hat. Somit können bereits modellierte AST-Knoten ohne Weiteres erweitert, kombiniert und in das Modell eingetragen werden. Diese Eigenschaft macht sich CPP2Spoon zunutze, um C++-Sprachkonstrukte ohne entsprechendes Java-Pendant durch eigene AST-Knoten zu repräsentieren.

4 Code Smell-Erkennung mit ISmell

ISmell (Incremental Smell Detection) ist ein ebenfalls von uns entwickeltes Tool zur automatischen Erkennung verschiedener Code Smells, unter anderem: *Gottklassen*, *lange Methoden*, *komplexe Methoden*, *lange Parameterlisten*, *Datenklassen*, *ungenutzter Programmcode*, *Zyklen* und *Feature-Neid*. Die zur Erkennung von Code Smells notwendigen Metriken werden mit Hilfe von Spoon erfasst. Folgende Sprachkonstrukte müssen dabei durch den AST bereitgestellt werden:

- Namensräume
- Funktionen, Variablen und Parameter
- Kontrollstrukturen
- Klassen, Attribute und Methoden
- Typinformationen und Typreferenzen

Obwohl CPP2Spoon sich noch in der Entwicklung befindet, werden bereits viele dieser Konstrukte unterstützt. Dies ist unter anderem der Tatsache geschuldet, dass sich C++ und Java in vielen Punkten strukturell ähneln. So lassen sich nahezu alle Kontrollstrukturen von C++ ohne Weiteres in Java abbilden. Eine der wenigen Ausnahmen stellt die *goto*-Anweisung dar, für die es in Java keine direkte Abbildung gibt. Andere Konzepte wie Namensräume sind in C++ flexibler als in Java, bedürfen für den Spoon-AST jedoch nur geringe Anpassungen. So wurde der AST-Knoten für Java-Pakete um die Fähigkeit erweitert Funktionen und globale Variablen enthalten zu können. Die Traversierung des AST-Knotens wurde ebenfalls entsprechend angepasst, was eine nahtlose Integration in Spoon ermöglicht. Da Typinformationen und -referenzen in SrcML nicht enthalten sind, wurde CPP2Spoon um eine semantische Analyse erweitert, die jedoch derzeit noch nicht vollständig zur Verfügung steht. Auch Klassen und die damit einhergehenden Mechanismen wie Attribute, Methoden und Vererbung werden noch nicht unterstützt. Die zur Verfügung stehenden Transformationen sind jedoch bereits ausreichend, um einen Teil der von ISmell unterstützten Code Smells korrekt erkennen zu können, ohne ISmell selber anpassen zu müssen.

5 Konklusion und Ausblick

Schon jetzt sind wir mit CPP2Spoon in der Lage, einen Teil unserer in ISmell implementierten Mechanismen zur Erkennung von Code Smells für C++ wiederzuverwenden. Eine Transformation von SrcML in einen anderen Java-AST ist ebenfalls denkbar, was eine Wiederverwendung weiterer Tools wie *JDeodorant* erlauben würde. Da SrcML neben C++ auch C# unterstützt, ist eine Transformation von C#-Programmcode in den Spoon-AST geplant, um die Unterstützung von ISmell auf weitere Programmiersprachen auszuweiten.

Literatur

[1] M. L. Collard, H. H. Kagdi, and J. I. Maletic. An xml-based lightweight c++ fact extractor. In *11th IEEE International Workshop on Program Comprehension, 2003.*, pages 134–143, May 2003.

[2] M. Fowler, K. Beck, J. Brant, W. Opdyke, and D. Roberts. *Refactoring: Improving the Design of Existing Code.* Addison-Wesley, 1999.

[3] C. Marinescu, R. Marinescu, P. F. Mihancea, and R. Wettel. iplasma: An integrated platform for quality assessment of object-oriented design. In *In ICSM (Industrial and Tool Volume*, pages 77–80. Society Press, 2005.

[4] R. Pawlak, M. Monperrus, N. Petitprez, C. Noguera, and L. Seinturier. Spoon: A Library for Implementing Analyses and Transformations of Java Source Code. *Software: Practice and Experience*, 46:1155–1179, 2015.

[5] A. Telea and H. Byelas. Querying large c and c++ code bases: the open approach. 2009.