Einstiegspunkte für Design und Codierung bei einer Wartungsaufgabe oder Fehlermeldung

Felix Fröhlich und Thorben Wiese

Universität Hamburg
Fakultät für Mathematik,
Informatik und Naturwissenschaften
Department Informatik

Abstract. Bei der Entwicklung und Wartung von Software spielt die Identifikation des Nutzen von Programmabschnitten eine große Rolle, um entsprechende Funktionen einer Software zu ändern oder zu reparieren. Diese *Feature Locations* stellen einen Einstigespunkt in den Quelltext für Design- oder Code-Änderungen dar und können mithilfe verschiedener Technologien ermittelt werden. In dieser Seminararbeit stellen wir unterschiedliche Technologien und deren Verfahren vor und geben einen Überblick über geeignete Nutzungsfelder.

1 Einleitung

Die erste Quelle [1].
...
Ziel dieser Seminararbeit ist es, die verschiedenen Analyseverfahren und Technologien zur Erkennung von Features im Code zu beschreiben und zu vergleichen.

2 Begriffe

Für die Vorstellung der Analyseverfahren sollen zunächst einige Begriffe definiert und erklärt werden.

Feature

Ein Feature ist ein Software Artefakt, das eine spezifische Funktionalität implementiert [2]. Diese Funktionalität wird in natürlicher Sprache beschrieben und wird von einem Programmabschnitt wiedergespiegelt. Ein Feature besteht üblicherweise aus einen Namen, einer Bedeutung (Intension) und einer Erweiterung (Extension) [3].

Feature Location

Der Prozess der Feature Location beschreibt die Identifikation der Beziehung zwischen Features und deren Implementierung. Dabei liegt die Beschreibung des Features in natürlicher Sprache vor, die dann einem entsprechenden Codeabschnitt zugeordnet werden soll (Mapping) [1].

3 Analyseverfahren

In diesem Abschnitt sollen verschiedene Analyseverfahren vorgestellt werden, die das Finden von Feature Locations ermöglichen.

Program Dependence Analysis (PDA)

Die Analyse eines Programms auf interne Abhängigkeiten wird Abhängigkeitsanalyse (engl. Dependence Analysis) genannt. Sie umfasst in der Regel Kontrollflussabhängigkeiten und Datenabhängigkeiten innerhalb eines Programmes. Diese werden zur Übersetzungszeit mithilfe des Compilers festgestellt. Ziel dieser Analyse ist es zum Beispiel, zu überprüfen, ob ein Programm parallelisiert ausgeführt werden kann. Für die Analyse von Features im Quelltext ist diese Methode hilfreich, da durch sowohl Kontrollflussabhängigkeiten, als auch Datenabhängigkeiten auch semantische Verknüpfungen verschiedener Abschnitte des Codes hergestellt werden können [4].

Trace Analysis

Latent Semantic Indexing (LSI)

Das Verfahren des Latent Semantix Indexing wird zum Indexieren von Abschnitten und Mustern eines Textes mithilfe von Singulärwertzerlegung der Ausdruck-Dokument-Matrix verwendet. Die Ausdruck-Dokument-Matrix (engl. Document-Term-Matrix) stellt die Frequenz der Ausdrücke innerhalb eines Dokumentes oder Textes dar. Die Singulärwertzerlegung dieser Matrix ist die Darstellung der ursprünglichen Matrix als Produkt dreier spezieller Matrizen, deren Singulärwerte auf bestimmte Eigenschaften der Matrix schließen lassen. LSI geht davon aus, dass Wörter, die in einem bestimmten Kontext verwendet werden, eine ähnliche Bedeutung haben. Dieses Verfahren lässt sich von natürlicher Sprache auch auf Quellcode übertragen, insbesondere auf Kommentare innerhalb des Codes [5].

3.1 Statische Feature Location Technologien

In diesem Abschnitt sollen statische Technologien zum Finden von Features im Quelltext vorgestellt werden.

Statische Analyse

Die Analyse von Quelltext zu einem Zeitpunkt, bei dem das Programm kompiliert wird und noch nicht ausgeführt wurde, wird statische Analyse genannt. Hierbei werden mithilfe von zum Beispeil Datenfluss-Analyse und Kontrollgraphen alle Abhängigkeiten und Funktionsaufrufe innerhalb des Codes analysiert und es können unter anderem Fehler wie zum Beispiel Race-Conditions oder Buffer-Overflows identifiziert werden. Dieser Prozess wird häufig von automatisierten Tools durchgeführt [6].

Technologie Beispiele

Robillard et al. [35]
Shao et al. [40]

3.2 Dynamische Feature Location Technologien

Dynamische Analyse

Allgemein dynamische Analyse

Technologie Beispiele

Wong et al. [49]
Liu et al. [25] (SITIR)

3.3 Textuelle Feature Location Technologien

Allgemein textuelle Analyse, Tools beschreiben, Beispiele

4 Vergleich

5 Fazit

Für welchen Zweck welches Analyseverfahren und welche Technologie Ergebnis wahrscheinlich: Alles gar nicht so schlecht, je nach Bedarf muss eine Technologie ausgewählt werden oder eventuell mit einer anderen kombiniert werden.

Literaturverzeichnis

[1] Rubin J, Chechik M. A Survey of Feature Location Techniques. University of Toronto, Department of Computer Science; 2013.

[2] IEEE. Std. 829;. Available from: https://standards.ieee.org/findstds/standard/829-2008.html.

[3] Chen K, Rajlich V. Case Study of Feature Location Using Dependence Graph. Wayne State University Department of Computer Science;. Available from: http://www.cs.wayne.edu/~severe/publications/Chen.IWPC.2000.FeatureLocation.pdf.

[4] Stafford J. A Formal, Language-Independent, and Compositional Approach to Interprocedural Control Dependence Analysis. University of Colorado;. Available from: ftp://ftp.sei.cmu.edu/pub/jas/Stafford-thesis.pdf.

[5] Deerwester S, Dumais S, Furnas G, Landauer T, Harshman R. Indexing by Latent Semantic Analysis. University of Colorado;. Available from: http://lsa.colorado.edu/papers/JASIS.lsi.90.pdf.

[6] Open Web Application Security Project (OWASP). Static Code Analysis. Open Web Application Security Project (OWASP);. Available from: https://www.owasp.org/index.php/Static_Code_Analysis.