COBOL- und PL/I-Anwendungen automatisch wieder verstehen

Daniela Schilling
dschilling@delta-software.com
Delta Software Technology GmbH

Abstract

Je älter eine Anwendung wird, desto schwerer wird ihre Wartung, Weiterentwicklung und Modernisierung, da im Laufe der Zeit oft viel Wissen über die Anwendung verloren geht. Die Größe und Komplexität solcher Anwendungen lässt eine manuelle Wiedergewinnung des Wissens nicht zu. Deshalb gewinnt AMELIO Logic Discovery das Wissen aus COBOL- und PL/I-Anwendungen zu 100% automatisch zurück.

1 Problemstellung

Um (Legacy-)Anwendungen effizient warten zu können, ist ein genaues Wissen über die Anwendung notwendig. Das Gleiche gilt für die Übergabe an Nachfolger, Outsourcen, Modernisieren oder das Neuschreiben in einer anderen Sprache.

Die Kernanwendungen vieler Banken, Versicherungen und großer Unternehmen sind bis heute immer noch COBOL- und PL/I-Anwendungen. Die über Jahrzehnte gewachsenen Anwendungen enthalten das Wissen über die Unternehmensprozesse und bilden diese ab. In dieser Zeit haben die Anwendungen gelebt, sie wurden erweitert, modifiziert, umstrukturiert und an neue Anforderungen angepasst. Dabei ging das Design verloren, wurde die Dokumentation nicht vollständig nachgezogen und oft haben die ursprünglichen Entwickler das Unternehmen verlassen. Kurz: Das Wissen darüber, was in den Anwendungen implementiert ist, welche Sonderfälle und Abhängigkeiten bestehen, ist oft nur noch rudimentär vorhanden und die einzige zuverlässige Quelle für Informationen ist der Source-Code selbst. Aber, je größer und komplexer Anwendungen sind, desto fehleranfälliger, risikoreicher und zeitaufwändiger ist das Wiedergewinnen von Wissen.

2 Automatisch Wissen wiedergewinnen

Automatisch Wissen aus Source-Code wiederzugewinnen und zu redokumentieren, genau diese Herausforderung adressiert AMELIO Logic Discovery. Dabei werden beliebig große und komplexe COBOL- und PL/I-Anwendungen analysiert und Informationen gesammelt. Mittels formaler und logischer Abstraktionen sowie Schlussfolgerungen wird aus den gewonnenen Informationen das wirkliche Wissen wiederhergestellt und die Anwendungslogik von der technischen Infrastruktur getrennt. Die Erkenntnisse werden in verschiedenen Perspektiven übersichtlich und sprachneutral in einem interaktiven Frontend dargestellt. Zusätzlich können sie in Form von kundenspezifischen Reports ausgegeben werden. Auf diese Weise wird gleichzeitig das Wissen über die Anwendung wiedergewonnen und eine Redokumentation vorgenommen.

2.1 Eine Frage der Perspektive

Je nach Ziel und Aufgabenstellung variiert auch das benötigte Wissen. Die Darstellung des gewonnenen Wissens erfolgt thematisch gruppiert in Perspektiven, dabei werden Erkenntnisse zusammen dargestellt, auch wenn sie physikalisch über den Source-Code verteilt sind.

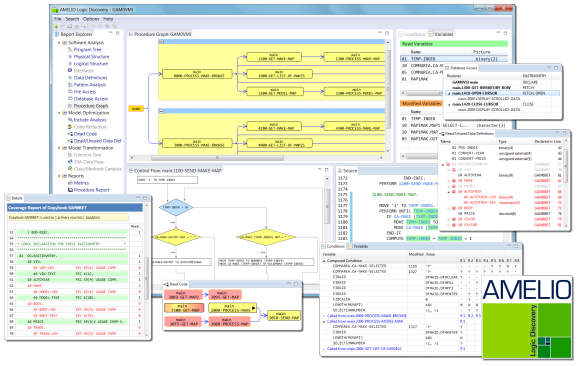

Figure 1: Präsentation von Analyseergebnissen

Der Aufbau der Darstellung ist so gestalltet, dass zunächst über die gesamte Anwendung und zu den einzelnen Themen ein Überblick geboten wird. Detailansichten bieten bei Bedarf zusätzliche Erkenntnisse an. Auf oberster Ebene wird gezeigt, wie die Programme einer Anwendung zusammen gehören, welche Aufrufbeziehungen und Abhängigkeiten zwischen ihnen bestehen und welche Bedingungen erfüllt sein müssen, damit ein Programmaufruf erfolgt. Die weiteren Perspektiven bieten dann Informationen darüber wie die Programme physikalisch und logisch aufgebaut sind, welche Variablen es gibt und wie und wo sie benutzt werden. Variablen und Code-Sequenzen, die nicht verwendet werden oder nur von

anderen nicht verwendeten Abschnitten, werden in der Dead-Code-Perspektive aufgeführt. Die Mustererkennung auf syntaktischer und struktureller Ebene ermöglicht das Aufdecken von Verletzungen von Programmierstandards (Bad Smells), Verwendung von alten Implementierungskonstrukten z.B. der Realisierung von Schleifen mittels Go-To oder aber der Verwendung von plattform- oder systemspezifischen Artefakten. Die Sicht auf Datei- und Datenbankzugriffe komplementiert die Darstellung der Erkenntnisse.

2.2 Prozeduren und Composites

Eine besondere Analyse ist die Erkennung von Prozeduren und Composites. In COBOL gibt es, zumindest in älteren Versionen, auf physikalischer Ebene nur Divisions, Sections und Parapgraphen, aber keine Prozeduren. Anhand von Aufrufbeziehungen und Verwendungen kann AMELIO Logic Discovery dennoch logische Prozeduren und deren Schnittstelle ermitteln.

Um die Übersicht über große COBOL- und PL/I-Anwendung zurück zu gewinnen, reicht es jedoch nicht aus, die Prozeduren und ihre Aufrufbeziehung zu ermitteln. Deshalb werden die Prozeduren zusätzlich in Composites oder technischen Gruppen zusammengefasst. Diese Gruppierung erfolgt zum einen auf Grund der ermittelten Aufrufbeziehungen zwischen den Prozeduren und zum anderen anhand von technischen Eigenschaften der Prozeduren (technische Infrastruktur). Für Composites kann zudem deren Schnittstelle ermittelt werden.

Enthält eine Prozedur nur Datenbank- oder File-Zugriffe, aber keine Business-Logik, so wird sie als CRUD-Prozedur markiert.

2.3 Mehr als nur Standard

Es besteht die Möglichkeit, die Analysen und Reports von AMELIO Logic Discovery um projekt- oder kundenspezifische Anforderungen zu erweitern. Es ist möglich, neue Analysen hinzuzufügen oder bestehende anzupassen, dies gilt besonders für den Bereich der Mustererkennung, so können z.B. firmeninterne Richtlinien in die Bad Smell-Erkennung aufgenommen werden. Sowohl der Inhalt als auch die Aufbereitung der Reports kann so aufgebaut werden, dass sie den jeweiligen Firmenstandards entsprechen.

3 Alles neu macht der Mai

AMELIO Logic Discovery hilft jedoch nicht nur dabei das Wissen über die Anwendung zurück zu gewinnen, sondern auch bei der Vorbereitung, Planung und Durchführung von Modernisierung- und Migrationsprojekten. Die für die Aufgabenstellung relevanten Analysen werden ggf. projektspezifisch konfiguriert. Zu den möglichen unterstützten Einsatzgebieten gehören einerseits Clean Up- und Refactoring-Projekte und andererseits Migrations- und Modernisierungsprojekte.

3.1 Clean Up und Refactoring

Zur Verbesserung der Wartbarkeit einer Anwendung liefert AMELIO Logic Discovery automatisch Vorschläge für eine Bereinigung oder ein Refactoring.

Für die Bereinigung des Codes wird z.B. ermittelt, welche Code-Sequenzen oder Datendefinitionen tot sind bzw. nur aus anderen toten Code-Stellen heraus verwendet werden. Für Copybooks (COBOL) und Includes (PL/I) wird für jedes enthaltene Statement analysiert, wie oft es verwendet wird bzw. wie oft es toten Code erzeugt, so dass auch Vorschläge zur Bereinigung und Refaktorisierung der generischen Module erstellt werden. Desweiteren werden alle Stellen im Code markiert, in denen Coding-Standards verletzt, unterschiedliche Schreibweisen genutzt oder eine veraltete Syntax verwendet wird.

Vorschläge für ein Refactoring werden unter anderem durch die Analyse von Prozeduren und Composites ermittelt. Große Composites sprechen für die Auslagerung in ein eigenständiges Unterprogramm. Enthält ein Composite dagegen viele kleine Paragraphen bzw. Prozeduren, so spricht dies dafür, dass diese zu sinvollen größeren Einheiten zusammengefasst werden. Es besteht auch die Möglichkeit, Paragraphen, Prozeduren, Sections oder Datenfelder zu ermitteln, die umbenannt werden sollten.

3.2 Migration und Modernisierung

Soll eine Migration oder Modernisierung, die über ein Refactoring hinaus gehen, der Anwendung vorgenommen werden, so liefert AMELIO Logic Discovery das für Entscheidungen und die Planung relevante Wissen, sowohl zur Vorbereitung des Projekts als auch zur Projektbegleitung.

Für Migrationsprojekte werden z.B. mittels der Musteranalyse alle plattformspezifischen Code-Stellen ermittelt. Ist die Zielplattform bereits bekannt, so kann automatisch analysiert werden, ob eine Modifikation erforderlich ist oder ob das Artefakt auch von der neuen Plattform unterstützt wird und für die Migration nicht relevant ist.

Bei einem Austausch des Datenbanksystems oder der Einführung einer Service-Orientierten-Architektur werden alle umszutellenden Code-Stellen ermittelt.

Wird die Anwendung oder Teile daraus in einer anderen Sprache neuentwickelt, so geschieht dies zumeist durch Entwickler, die die neue Sprache sehr gut kennen, jedoch mit COBOL oder PL/I nur wenig vertraut sind. In solchen Projekten hilft die modellorientierte, sprachneutrale Darstellung der Analyseergebnisse, das Wissen über die Anwendung zu transferieren.

Auf diese Weise hilft AMELIO Logic Discovery die Komplexität von Migrations- und Modernisierungsprojekten genauer abzuschätzen und das Risiko bei der Umsetzung zu minimieren.