Kapitel 1

Einleitung

Für die Entwicklung von Softwareprodukten sind entsprechende Verfahren, Methoden und Werkzeuge erforderlich. Die Disziplin Software-Entwicklung umfaßt das Management, die Planung, die Analyse, den Entwurf, die Implementierung, das Testen und die Wartung. In alle Phasen der Software-Entwicklung ist es wichtig, möglichst viele Aktivitäten zu bewerten, um z.B. Fortschritte oder Schwachstellen zu erkennen und gegebenenfalls geeignete Maßnahmen zu unternehmen. Dazu sind entsprechende Verfahren erforderlich. Dies können z.B. Verfahren sein, die die Komplexität von Software-Programmen messen.

Die Diplomarbeit beschäftigt sich mit der Messung der Komplexität von ABAP/4-Programmen. Die Meßverfahren von [McCabe76] und [Halstead77] werden für die Sprache ABAP/4 interpretiert und angewendet.

1.1 Motivation und Ziel der Diplomarbeit

Die Idee zu dieser Diplomarbeit entstammt dem Releaseprojekt „juDit" bei der Bertelsmann Distribution. Die Aussage, die auf den neuen Release umzusetzenden ABAP/4-Programme seien zu kompliziert bzw. zu komplex, umschreibt die Problematik der Umstellung der eigenentwickelten Programme. Diese Aussage ist eine intuitive Einschätzung der Programme aus Sicht des Entwicklers.

Die intuitive Bewertung von Programmen soll in der Diplomarbeit durch geeignete Methoden validiert werden. Dazu müssen Modelle, die die Methoden untermauern, vorhanden sein. In der Literatur sind für die Programmiersprache ABAP/4 keine Software-Messungen beschrieben. Es existieren keine Verfahren zur Messung der Komplexität von Programmen. Um die Komplexität von ABAP/4-Programmen betrachten zu können, werden zwei anerkannte Komplexitätsmessungen verwendet. Die Verfahren stammen von [McCabe76] und [Halstead77]. Die beiden Methoden werden speziell für die Sprache ABAP/4 interpretiert. Die Diplomarbeit ist eine der ersten Arbeiten, die sich mit Verfahren zur Komplexitätsmessung von ABAP/4-Programmen beschäftigt.

Die Messung der Komplexität der beiden Verfahren beruht auf unterschiedlichen Modellen. Bei der zyklomatischen Zahl von McCabe basiert die Komplexität auf der im Programme inliegenden Struktur. Dagegen beschreibt die Programmier-Leistung von Halstead die berechnende Komplexität, d.h. die eingeschlossene Komplexität des Algorithmuses. Zu untersuchen ist, ob die Verfahren zur Komplexitätsmessung von ABAP/4-Programmen eingesetzt werden können. Außerdem wird geprüft, wie stark die zwei Meßverfahren miteinander korrelieren, d.h. ob sie sich ausschließen oder ergänzen.

Als einziges Programm wurde die Lieferscheinerstellung nicht „Eins-zu-Eins“ umgesetzt. Es ist mit demselben Funktionsumfang neu programmiert worden. Die Zielsetzung war möglichst viele SAP-Standard-Anwendungen zu nutzen und den modularen Programmaufbau zu überarbeiten. Der Aspekt, ob die Komplexität der neuen Version der Lieferscheinerstellung geringer geworden ist, konnte nach Beendigung des Projekts nicht zufriedenstellend geklärt werden. Aufbauend auf der Möglichkeit, die Komplexität von ABAP/4-Programmen quantifizieren zu können, soll der Einfluß der Änderungen in der Lieferscheinerstellung bezüglich der Komplexität untersucht werden.

Die Ziele und die sich ergebenen Fragen werden aus der Idee zu dieser Diplomarbeit und der beschrieben Motivation extrahiert:

1. Die Meßverfahren von McCabe und Halstead werden für die Programmiersprache ABAP/4 interpretiert und validiert.

   Kann mittels der beiden Verfahren die Komplexität von ABAP/4-Programme gemessen werden?

2. Es wird eine neue Komplexitätsmessung McHa[1] vorgeschlagen. Sie setzt sich aus den Verfahren von McCabe und Halstead zusammen und wird als 2-Tupel betrachtet.

   Beschreibt der Vorschlag der Kompexlitätsmessung McHa die Komplexität der Programme besser als die jeweiligen Ausgangsmessungen? Wie effizient ist die Messung?

3. Die alte und neue Version der Lieferscheinerstellung sollen im Hinblick auf ihre Komplexitäten untersucht werden.

   Ist bei der neuen Version eine Verringerung der Komplexität im Vergleich zur Alten festzustellen?

1.2 Theoretischer Hintergrund

Die Meßverfahren von [McCabe76] und [Halstead77] werden für die Komplexitätsmessung von ABAP/4-Programmen aus den folgenden Gründen eingesetzt: Es

[1] Der Name leitet sich aus den Namen der Autoren <u>Mc</u>Cabe und <u>Ha</u>lstead der verwendeten Komplexitätsmessungen ab.

handelt sich bei den Verfahren um eine der ersten und vollständigen Software-Messungen. Sie zählen zu den be- und anerkanntesten Meßverfahren und werden heutzutage noch in vielen Untersuchungen verwendet.

Die zyklomatische Zahl von McCabe mißt die strukturelle Komplexität von Programmen. Das Komplexitätsmaß kann direkt aus dem Programmtext ermittelt werden. Die Grundlage für die Berechnung der zyklomatischen Zahl bildet der Kontroll-Fluß-Graph G. Die zyklomatische Zahl $v(G)$ eines Graphen G, bestehend aus e Kanten, n Knoten und p miteinander verbundenen Modulen ist: $v(G) = e - n + 2p$. Bei einem Programm, das sich aus mehreren Modulen zusammensetzt, wird jedes Modul durch einen Kontroll-Fluß-Graphen dargestellt. Die Gesamt-Komplexität eines modularisierten Programms ist die Summe der zyklomatischen Zahlen der einzelnen Module. Bei einem geschlossenen Graph G ist die zyklomatische Zahl gleich der maximalen Anzahl linear unabhängiger Regelkreise. Ein Regelkreis ist eine abwechselnde Folge von Knoten und Kanten, wobei der erste und letzte Knoten identisch ist. Jeder Knoten korrespondiert mit einem Programmfragment. Die Komplexitätsmessung definiert die Basispfade, die in Kombination alle möglichen Pfade im Programm generieren, d.h. sie beinhaltet die Summe aller linear unabhängigen Pfade. Unter der Voraussetzung, das ein Programm aus keinen Modulen besteht, schlägt McCabe eine vereinfachte Komplexitätsmessung vor: $v(G) = \pi + 1$. π ist die Anzahl der Verzweigungen im Programm.

Im Gegensatz zu McCabe hat Halstead ein Meßverfahren für die berechnende Komplexität vorgeschlagen. Die Messungen basieren auf der Anzahl verwendeter eindeutiger Operatoren η_1 und Operanden η_2 sowie der Gesamtzahl der Operatoren N_1 und Operanden N_2 im Programm. Diese vier Basisgrößen können direkt aus dem Programmtext ermittelt werden. Schwierig ist die Definition der Regeln zur Identifizierung von Operatoren und Operanden. Alle weiteren Messungen werden aus den vier Basisgrößen abgeleitet. Die Programm-Länge N setzt sich wie folgt zusammen: $N = N_1 + N_2$. Die Länge kann auch durch $\widehat{N} = \eta_1 \log_2 \eta_1 + \eta_2 \log_2 \eta_2$ berechnet werden. $V = N \log_2 \eta$ ist die Größe des Programms in Bits. Dies gilt unter der Voraussetzung, daß eine Binärkodierung mit gleicher Wortlänge des Vokabulars $\eta = \eta_1 + \eta_2$ vorliegt. η unterschiedliche Schlüsselworte können durch $\log_2 \eta$ Bits dargestellt werden. Das gesamte Programm mit N Operatoren und Operanden erfordert V Bits zur Kodierung. Die Größe V ist von der Implementierung des Algorithmuses abhängig und kann stark variieren. Die Programmier-Leistung E ist der Aufwand zur Kodierung eines Algorithmus: $E = \frac{V}{L}$. L wird als Programm-Level bezeichnet und zeigt an, wie geeignet eine Programmiersprache zur Implementierung eines Algorithmus ist. L nähert sich dann dem Wert 1 an. Halstead leitet den Programmieraufwand eines Programms aus einer Betrachtung der zur Programmerstellung notwendigen Anzahl mentaler Vergleiche her. Der Aufwand ensteht aufgrund der Auswahl von Operatoren und Operanden aus einem Vokabular und die Auswahl wird durch binäres Suchen vorgenommen. Somit sind für die Erstellung eines Programms V Vergleich erforderlich [Liggesmeyer90].

Die Programmiersprache ABAP/4 wird zur Anwendungsentwicklung im SAP System R/2 eingesetzt. Sie ermöglicht eine Programmierung im betriebswirtschaftlichen Umfeld und dient im R/2-System zur Entwicklung von Dialoganwendungen und zur Auswertung von Datenbanken. ABAP/4 erlaubt eine strukturierte Programmierung. Übliche Kontrollstrukturen und Konzepte zur Modularisierung sind vorhanden. Der Sprachumfang ist auf den Einsatz im Rahmen betrieblicher Informationssysteme zugeschnitten. ABAP/4 ist eine formatfreie Sprache und ABAP/4-Programme bestehen aus Operatoren und Operanden. Die Sprache ist ereignisorientiert. Die Anweisungen im Programm werden nicht sequentiell in der Reihenfolge ihres Auftretens durchgeführt. Mittels Ereignisschlüsselworten werden Anweisungen zu Verarbeitungsblöcken zusammengefaßt. Das Eintreten bestimmter Ereignisse ist abhängig von anderen Programmen oder von interaktiven Anwendereingaben. Die Ausführung innerhalb eines Verarbeitungsblockes beruht auf seiner inliegenden Struktur. Die ABAP/4-Schlüsselworte werden in verschiedene Kategorien unterschieden: Ereignisschlüsselworte geben die Verarbeitungszeitpunkte an, bei denen die entsprechenden Verarbeitungsblöcke abgearbeitet werden. In den Blöcken formen Steuerungsschlüsselworte Kontrollstrukturen und operationale Schlüsselworte führen bestimmte Verarbeitungen aus.

1.3 Komplexitätsmessungen und ihre Ergebnisse

Um die Meßverfahren von McCabe und Halstead auf ABAP/4-Programme anwenden zu können, wird die durch die Ereignisse hervorgerufene Ablaufsteuerung im Programm vereinfacht. Die Ereignissschlüsselworte werden als sequentielle Anweisungen aufgefaßt. Die durch die Ereignisse hervorgerufen Struktur wird vernachlässigt. Die inliegende Struktur eines Verarbeiungsblocks bleibt dabei unberücksichtigt. Mit dieser vereinfachten Betrachtung der ABAP/4-Ablaufsteuerung können die Kontroll-Fluß-Graphen von McCabe für die Beispiel-Programme erstellt werden, ohne zusätzlich die Struktur der Ereignisse abbilden zu müssen. Bei der Aufstellung der Regeln zur Identifizierung von Operatoren und Operanden für die Messungen nach Halstead, werden die Ereignisschlüsselworte als gewöhnliche Operatoren betrachtet.

Zur Validation der Verfahren für ABAP/4-Programme wurden 62 Schulungsprogramme ausgewählt. Der Vorteil dieser Programme ist, daß sie überschaubar und leicht einzuordnen sind. Das erleichert die Messung und die anschließend Überprüfung gegenüber der intuitiven Einschätzung.

Die erste Einschätzung der Beispiel-Programme erfolgt intuitiv. Die Programme werden durch die Relationen "komlexer als" und "gleich komplex" relativ zueinander eingeordnet. Diese Bewertung wird als intuitive Komplexität bezeichnet. Die intuitive Komplexität teilt die Beispiel-Programme in Klassen ein. Innerhalb einer Klasse sind die Programme intuitiv als gleich komplex anzusehen. Die Ergebnisse der Komplexitätsmessung nach McCabe und Halstead werden gegenüber der intuitiven Komplexität geprüft. Es wird jeweils die Korrelation zwischen der intuitiven Komplexität und den Meßergebnissen der beiden Verfahren betrachtet. Die Korrelation beschreibt die statistisch gesicherte Steigerung der Ergebnisse der Komplexitätsmessungen zu der intuitiven Einschätzung. Die Irrtumswahrscheinlichkeit liegt bei beiden Verfahren unter 1%. Diese Resultat besagt, das unter Voraussetzung der Einschränkung in der ABAP/4-Ablaufsteuerung und der Interpretation der Verfahren für die Programmiersprache ABAP/4 die Messungen als Komplexitätsmessungen eingesetzt werden können. Eine weiterführende Untersuchung muß an großen ABAP/4-Programmen erfolgen, um zu überprüfen, ob die dort die Ergebnisse bestätigen lassen. In der Diplomarbeit wurde diese Validation nur am Beispiel der alten und neuen Version der Lieferscheinerstellung durchgeführt. Anhand dieses einen Beispiels liefern die Komplexitätsmessungen die erwarteten Ergebnisse.

Unter Aussschaltung der intuitiven Komplexität ist ein Zusammenhang zwischen den Meßverfahren von McCabe und Halstead zu mindestens 95% gesichert. Dieses Resultat besagt, daß die beiden Verfahren jeweils die Beispiel-Programme ähnlich relativ zueinander einordnen. Die Abweichung kann durch Schwächen der Methoden oder die Interpretation für die Programmiersprache ABAP/4 hervorgerufen werden.

1.4 Vorschlag einer Komplexitätsmessung

Das Meßverfahren von McCabe weist eine Reihe von Schwächen auf. Zwei konnten bei der Messung der Beispiel-Programme nachvollzogen werden. Die Mängel sind, daß das Verfahren nicht sensitiv auf sequentielle Anweisungen ist und das Mehrfachaufrufe von Modulen nicht abgebildet werden. Der Nachteil, das sequentielle Programmfragmente nicht in die zyklomatische Zahl einfließen, da sie keine Struktur besitzen, wird, durch die Messung von Halstead auszugleichen, versucht. Die Programmier-Leistung von Halstead bildet berechnende Komplexität ab. Der Vorschlag der Komplexitätsmessung McHa ist ein 2-Tupel. Durch den Vorschlag wird der Mangel der zyklomatischen Zahl, die Programme in wenige Komplexitätsklassen zu unterteilen, ebenfalls behoben.

Die Validation der Komplexitätsmessung McHa zeigt, daß sequentielle Programme abgebildet werden können. Das ist mit einer Irrtumswahrscheinlichkeit von unter 1% statistisch gesichert. Die detailliertere Klasseneinteilung kann nur innerhalb einer Komplexitätsklasse nach McCabe bestätigt werden. Bei einem Vergleich bei nicht-sequentiellen Programmen zwischen der zyklomatischen Zahl und der Komplexitätsmessung McHa zeigt sich, daß durch McHa keine nennenswerte Verbesserung festzustellen ist.

Die Ermittlung der Anzahl der Operatoren und Operanden bei dem Verfahren von Halstead erwies sich als aufwendig. Mit Hilfe der Faktorenanalyse werden die vier Basisgrößen die eindeutigen Operatoren η_1 und Operanden η_2 und die Gesamtzahl der Operatoren N_1 und Operanden N_2 auf einen Faktor reduziert. Diese Vereinfachung gilt nur für die Beispiel-Programme. Der Faktor beschreibt die Korrelationen zwischen den Basisgrösen. Zusammen mit den Faktorenwerten lassen sich die Basisgrößen extrahieren und die Programmier-Leistung kann bestimmt werden. Die

Komplexitätsmessung McHa unter Verwendung des extrahierten Faktors ist effizienter als die beiden Ursprungsverfahren.

1.5 Darstellung der Gesamt-Ergebnisse

Die durchgeführten Untersuchungen in der Diplomarbeit zeigen, daß die Meßverfahren von McCabe und Halstead zur Komplexitätsmessung von ABAP/4-Programmen einsetzbar sind. Voraussetzung ist die vereinfachte ABAP/4-Ablaufsteuerung und die angefertigten Interpretationen. Diese umfassen die ABAP/4-Kontrollgraphen und die Regeln zur Identifizierung von Operatoren und Operanden. Damit sind die Methoden eine der ersten Verfahren zur Messung der Komplexität von ABAP/4-Programmen.

Die Korrelationsrechnung bestätigt einen hohen Zusammenhang zwischen der zyklomatischen Zahl und der Programmier-Leistung. Die statistische Sicherheit beträgt 95%. Die vorgeschlagene Komplexitätsmessung McHa baut auf diese starke Beziehung auf. Es gibt ABAP/4-Programme, die von den Verfahren von McCabe und Halstead unterschiedlich eingeordnet werden. Eine genauere Untersuchung zeigt, daß dies sequentielle Programme sind. Die Schwäche der zyklomatischen Zahl, sequentielle Programme nicht abbilden zu können, wird durch die Kombination mit der Programmier-Leistung behoben. Innerhalb einer zyklomatischen Komplexitätsklasse erfolgt außerdem durch die Komplexitätsmessung McHa eine differenzierte Betrachtung der ABAP/4-Programme. Die Möglichkeit der Extraktion eines Faktors aus den vier Basisgrößen von Halstead schafft die Grundlage für eine effiziente Komplexitätsmessung McHa. Die Operatoren und Operanden brauchen nicht mehr gemessen werden, was aufwendig ist, sondern können berechnet werden.

Die drei in der Diplomarbeit behandelten Meßverfahren wurden an ABAP/4-Schulungsprogrammen validiert. Diese sind im Gegensatz zu Programmen in Software-Systemen klein und besitzen eine geringe Komplexität. Die Verfahren konnten nur an einem großen Beispiel untersucht werden. Das Beispiel ist die alte und neue Version der Lieferscheinerstellung. Die Messungen bestätigen die intuitive Einschätzung. Weitere Überprüfungen an anderen ABAP/4-Progammen müßten folgen.