

Entwicklung eines Werkzeugs zur Messung von Softwarequalität sicherheitsrelevanter Industriesoftware durch statische Codeanalyse

N. Jung, M. Krell, V. Maurer, Prof. Dr. rer. nat. D. Reinert, C. Staron

Fachbereich Informatik
Fachhochschule Bonn-Rhein-Sieg
Grantham-Allee 20
53757 St. Augustin
VitaliMaurer@arcor.de
Christian.Staron@kayser-staron.de

Abstract: Im Rahmen eines Softwareentwicklungsprojekts wurde im Berufsgenossenschaftlichen Institut für Arbeitsschutz (BGIA) in enger Zusammenarbeit mit dem Fachbereich Informatik der Fachhochschule Bonn-Rhein-Sieg eine Anwendung zur Unterstützung von Softwareprüfungen implementiert. Basierend auf Softwaremetriken dient die Anwendung dem BGIA zur statischen Quellcodeanalyse sicherheitsrelevanter Software und unterstützt auf diese Weise Maschinenprüfer bei ihrer Arbeit. In diesem Projekt konnten bekannte Metriken, wie Halstead und McCabe, sowie neu entwickelte Kennzahlen und Qualitätskriterien an die Softwareanforderungen von Industriemaschinen angepasst und in einem Prototypen integriert werden.

1 Motivation zur Softwarequalität und Methoden zu deren Prüfung

Die zunehmende Bedeutung von Software für Anwendungen des täglichen Bedarfs und der wachsende Anspruch an ihre Zuverlässigkeit führen zu einem steigenden Qualitätsanspruch an Software. Dabei verdienen sicherheitskritische Anwendungen eine besondere Aufmerksamkeit. Aufgrund ihrer Flexibilität, dem daraus resultierenden wirtschaftlichen Nutzen und ihrer permanenten Fortentwicklung, bilden Mikrocontroller eine wichtige Komponente der heutigen Sicherheitstechnik. Zur Prüfung von Softwarequalität existieren in der Literatur verschiedene Methoden. Informelle Qualitätssicherungsmaßnahmen wie Review, Inspektion und Walkthrough, können während des Entwurfs eingesetzt werden, liefern jedoch häufig nicht direkt quantifizierbare Ergebnisse. Dagegen ermöglichen analytische Verfahren Messergebnisse und schaffen somit die Basis für eine Interpretation und Bewertung der Resultate. Das analytische Verfahren der Codeanalyse bildet die Basis für unsere Entwicklung, weswegen bezüglich der anderen Ansätze an dieser Stelle auf die Literatur verwiesen wird. [BA00; BA98; BB98; DU92; EH02; FP97; KS97; SO03]

2 Entwicklung und Einsatz eines Softwareanalysewerkzeugs

Ein prominenter Vertreter der analytischen Verfahren ist die statische Codeanalyse, bei der zu untersuchende Softwarequellcodes nach verschiedenen Metriken untersucht werden. Die Resultate aus diesen Untersuchungen können dann wiederum als Basis für Qualitätskriterien der Software dienen. Im Berufsgenossenschaftlichen Institut für Arbeitsschutz (BGIA) wird seit über 10 Jahren das auf einer UNIX-Workstation laufende Werkzeug Logiscope der Fa. Verilog zur Codeanalyse maschinennaher sicherheitsrelevanter Industriesoftware für Echtzeitanwendungen eingesetzt. Mit dem Einsatz konnten zahlreiche Erfahrungen gesammelt werden, aus denen sich neue Anforderungen entwickelten. In Zusammenarbeit mit dem Fachbereich Informatik der Fachhochschule Bonn-Rhein-Sieg wurden Entwicklungen bezüglich Metriken und Qualitätskriterien insbesondere für Assembler, und deren Analyseunterstützung initiiert.

Folgende Aspekte stellen zu erreichende Zielfaktoren der Neuentwicklung dar:

- Verwendung der Kennzahlen im Kontext sicherheitsrelevanter Assemblersoftware
- Effiziente Softwareanalyse durch weitestgehende Automatisierung des Prozesses
- Analyse verschiedener Assemblerdialekte, beziehungsweise weiterer, in der Industrietechnik eingesetzter Programmiersprachkonzepte, ähnlich den Assemblercodes, beispielsweise zur Programmierung speicherprogrammierbarer Steuerungen
- Tiefe Integration der Analyse in den Entwicklungsprozess, durch konsequente Unterstützung einer engen Kooperation zwischen Prüfinstanz und Produkthersteller, auf Basis der Analyseergebnisse

Ausgehend von den über 40 in Logiscope verwendeten Metriken, die auf den veröffentlichten Metriken nach Halstead und McCabe aufbauen, konnten diese Kennzahlen angepasst, weiterentwickelt und an aktuellen sicherheitsrelevanten Assemblercodes für Industriesoftware geprüft und validiert werden. Darüber hinaus wurde ein Softwareentwicklungsprojekt durchgeführt, welches die entwickelten Kennzahlen in ein PC-gestütztes Softwareanalysewerkzeug überführt, die eine teilautomatisierte Analyse von Assemblerprogrammen ermöglicht. [HA77; KR03; ST04]

2.1 Entwicklungsstand des Softwareanalysewerkzeugs „mEtRIKA“

In einem ersten Schritt wurden zusätzliche Metriken gebildet, die es erlaubten die Komplexität der Software ohne CASE-Strukturen und Aussprünge in ein Fehlermodul, die bei sicherheitsrelevanter Software notwendig sind, zu bewerten. Auf diese Weise kann die Codestruktur vereinfacht werden, sodass vermeidliche Strukturschwächen ausgeblendet und die Gutachter auf tatsächliche Schwächen klarer hingewiesen werden. Speziell für Assemblerprogramme betrifft dies beispielsweise die Anzahl: toter Sprungmarken, überflüssiger Sprünge, einzeliger Schleifen und Schleifen in einem Modul. Es wurden auch zwei neue hybride Metriken eingeführt, die Module mit großem Umfang aber geringer Komplexität finden und die Kommentarrate mit dem verwendeten Wortschatz und der Modulkomplexität verbinden. Mit Hilfe der neuen Metriken wurden die Qualitätskriterien Testbarkeit, Einfachheit und insbesondere Lesbarkeit und Selbstbeschreibung zur deutlicheren Identifikation tatsächlicher Schwächen angepasst. [KR03]

Nach dieser Vorarbeit konnte eine Softwareanwendung konzipiert und prototypisch implementiert werden, welche sicherheitsrelevante Assemblercodes analysiert. Der Anwender wird dabei anhand einer Benutzeroberfläche, ähnlich eines Assistenten, durch die Konfiguration des Analysewerkzeugs geführt. Die Analyse verschiedener Assemblerdialekte wird teilautomatisiert durch eine Klassifizierung der Befehle des zu prüfenden Quellcodes unterstützt. Mit Start der Analyse werden die zu analysierenden Quellcodes sukzessive überprüft und eine Reihe zentraler Metriken ermittelt, auf deren Basis die o.g. Qualitätskennzahlen berechnet werden. Die Resultate werden anschließend rein numerisch, sowie grafisch in Form eines Kontrollflussgraphen repräsentiert. Darüber hinaus stehen sie innerhalb der Anwendung und in Form elektronischer „Berichtsmappen“ anwendungsunabhängig zur Verfügung. [ST04] In einem Folgeprojekt soll das Analysewerkzeug auch für die Hochsprache C nutzbar gemacht werden.

2.2 Analyse sicherheitsrelevanter Software und Einbettung des Analysewerkzeugs in den Prüfungsprozess

Im Rahmen einer Softwareprüfung kann ein Analysewerkzeug die Software-Begutachtung wesentlich unterstützen. Das Werkzeug ermöglicht einen schnellen Einstieg in komplexe Softwarearchitekturen und weist auf gegebenenfalls kritische Softwarekomponenten hin. Die Kooperation mit dem BGIA ermöglichte während der Neuentwicklung die Integration und Einbindung von Softwareprüfern in den Entwurfs- und Entwicklungsprozess, und dadurch eine starke Praxisorientierung des Werkzeuges. Einen besonderen Stellenwert erhielt dabei die weitestgehende Automatisierung des Analyseprozesses, um die Arbeit der Softwareprüfer durch die IT maßgeblich zu unterstützen. Eine unter Logiscope ausschließlich manuelle Konfiguration auf neue Prozessorfamilien konnte auf diese Weise teilautomatisiert werden, und stellt sich nach der Implementierung als sehr unterstützende Funktion heraus. Darüber hinaus konnten weitere Aspekte in der Entwicklung berücksichtigt werden, die sich beispielsweise in der komfortablen Projektverwaltung einer Softwareanalyse oder in dem darauf aufbauenden werkzeugunabhängigen elektronischen „Berichtswesen“ widerspiegeln.

Abbildung 1 zeigt das sukzessive Vorgehen bei einer Softwareanalyse. Nach jedem Schritt werden Datenobjekte erzeugt, die zentral in einer Projektdatei, in Form einer relationalen Datenbank, verwaltet werden. Beispiele für diese Daten sind allgemeine Projektdaten, wie Projektname, Datum der Analyse und verantwortlicher Gutachter. Die Referenzdaten spiegeln die Konfigurationseinstellungen des Analysewerkzeugs wider. Sie müssen vorgenommen werden, da durch verschiedene und nicht standardisierte Befehlssätze, wie beispielsweise Assemblercodes bei Mikrocontrollern, die Befehle syntaktisch aber auch semantisch stark voneinander abweichen können. Um für diesen Fall Metriken, wie beispielsweise bedingte und unbedingte Sprünge im Programmablauf, richtig berechnen zu können, sind diese Einstellungen notwendig. Mit dem Abschließen der Konfiguration kann der eigentliche Analyseprozess angestoßen werden. Dabei werden Ergebnisdaten erzeugt, welche wiederum in der Projektdatei gespeichert werden. Die Ergebnisse des Werkzeugs umfassen die Metriken nach Halstead und McCabe, sowie die o.g. Anpassungen [KR03; ST04].

Mit diesen Kennzahlen werden Qualitätskriterien errechnet, die die Programmgüte hinsichtlich Testbarkeit, Selbstbeschreibung, Einfachheit, und Lesbarkeit bewerten. Gemeinsam dienen die Kennzahlen und Qualitätsmerkmale der Analyse der Programmkomplexität und Programmstruktur.

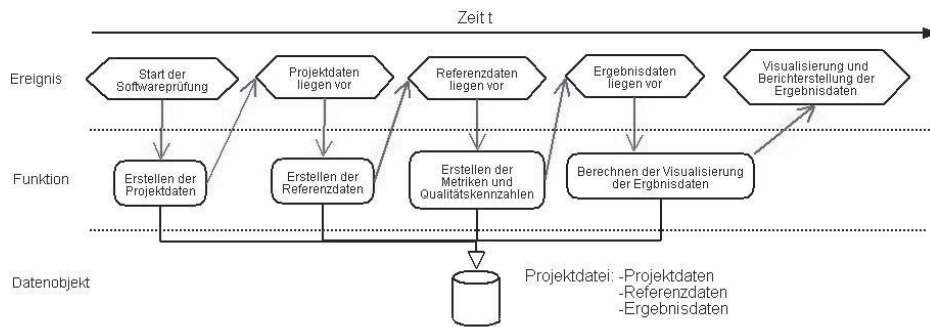


Abbildung 1: Vorgehen bei der Softwareanalyse anhand „mEtRIKA“

Nach Gesprächen mit Softwaregutachtern haben sich insbesondere visuelle Darstellungen des Kontrollflusses als sehr nützlich zur Prüfung erwiesen. Demnach wurde ein Konzept zur Darstellung und Speicherung der Softwareeigenschaften unabhängig vom Analysewerkzeug entwickelt. Auf Basis der berechneten Ergebnisse können elektronische Berichte aus dem Analysewerkzeug erzeugt werden, die in Form einer „Berichtsmappe“ grafische Darstellungen der Analyseergebnisse enthalten. Zur Entwicklung des elektronischen Berichts wurde auf Microsoft Excel zurückgegriffen, sodass die Visualisierungen in Arbeitsblättern dokumentiert, und innerhalb einer Exceldatei gespeichert werden. Auf diese Weise können die Analyseergebnisse in Dateiformat unabhängig vom Analysewerkzeug elektronisch verteilt, weiterverarbeitet, und als Prüfdokument genutzt werden. Diese Funktionalität kann insbesondere zur Unterstützung des Entwicklungsprozesses verwendet werden, da Softwareprüfungen in der Regel begleitend zum Entwicklungsprozess durchgeführt werden.

2.3 Prüfung verschiedener Quellcodearten und erste Erfahrungen

Inzwischen konnten Projekte initiiert werden, die sich mit der Weiterentwicklung des Analysewerkzeugs hinsichtlich der Softwareprüfung von verschiedenen Quellcodetypen beschäftigen. Im Rahmen der Erstentwicklung stand die Prüfung von Assemblercodes, wegen des praktischen Einsatzes im BGIA, im Vordergrund.

In den „mEtRIKA“ – Entwicklungsschritten, die bisher Inhalte für Abschlussarbeiten bildeten, konnten im Werkzeug die Kennzahlen, Kriterien und Analysefunktionen implementiert, und an beispielhaften Softwarekomponenten mit einigen hundert Quellcodezeilen getestet und validiert werden. Im Vergleich zum bisher eingesetzten kommerziellen Analyseprodukt liefert „mEtRIKA“ bereits jetzt vergleichbare Resultate.

3 Fazit und Ausblick

Anhand dieses Projekts konnte aufgezeigt werden, wie effizient und kostengünstig sich wissenschaftliche Methoden zur Qualitätssicherung von Software in einem „relativ einfachen“ Werkzeug und somit in den gesamten Entwicklungsprozess einbetten lassen. Durch Verwendung einer Programmiersprache, gemäß dem „Rapid Application Development“ – Ansatz, können Fortentwicklungen zeitnah erzielt werden. Dadurch bleibt ebenso Raum für individuelle Anpassungen des Werkzeugs. Bislang weist die Gesamtanwendung zwar einen prototypischen Charakter auf, erlangt jedoch in ersten Gegenüberstellungen mit einer gewerblichen Implementierung analoge Resultate.

Literaturverzeichnis

- [BA00] Balzert, H.: Lehrbuch der Software – Technik: Software-Entwicklung. 2. Auflage. Spektrum, Akademischer Verlag, Berlin, Heidelberg, 2000.
- [BA98] Balzert, H.: Lehrbuch der Software – Technik: Software-Management, Software-Qualitätssicherung, Unternehmensmodellierung. Spektrum, Akademischer Verlag, Berlin, Heidelberg, 1998.
- [BB98] Bömer, T. et.al.: Programmierregeln für die Erstellung von Software für Steuerungen mit Sicherheitsaufgaben. Wirtschaftsverlag NM-Verlag für neue Wissenschaften GmbH, Berlin, Dortmund, 1998.
- [DU92] Dumke, R.: Softwareentwicklung nach Maß: Schätzen – Messen – Bewerten. Friedrich Vieweg & Sohn Verlagsgesellschaft mbH, Braunschweig, Wiesbaden, 1992.
- [EH02] Ehrenberger, W.: Software-Verifikation – Verfahren für den Zuverlässigkeitsnachweis von Software. Carl Hanser Verlag, München, Wien, 2002.
- [FP97] Fenton, N. E.; Pfleeger, S. L.: Software Metrics – A Rigorous and Practical Approach. Second Edition. PWS Publishing Company, Boston, 1997.
- [HA77] Halstead, M. H.: Elements of Software Science. Elsevier North-Holland, Inc.; New York, 1977.
- [JU03] Jung, N.; Krell, M.; Reinert, D.; Schaefer, M.: Development for Metrics for Safety Related Software in Machinery. Publication on the 3rd International Conference „Safety of Industrial Automated Systems (SIAS)“, Nancy, 2003.
- [KR03] Krell, M.: Bestimmung von Qualitätskriterien für sicherheitsrelevante Software im Maschinenschutz auf Basis von zertifizierten Industrieanwendungen. Diplomabschlussarbeit – Fachhochschule Bonn-Rhein-Sieg / Fachbereich Informatik, Sankt Augustin, 2003.
- [KS97] Klug, J.; Schaefer, M.: Fehlererkennende Maßnahmen in Mikroprozessoren / Sicherheitstechnisches Informations- und Arbeitsblatt. Berufsgenossenschaftliches Institut für Arbeitsschutz (BIA) – Handbuch 29. Lfg. VII/97, Sankt Augustin, 1997.
- [SO03] Sommerville, I.: Software Engineering. 6. Auflage. Pearson Studium, München, 2003.
- [ST04] Staron, C.: Entwicklung eines Analysewerkzeugs zur Ermittlung von Metriken und Qualitätskriterien sicherheitsrelevanter Software im Maschinenschutz. Bachelor Thesis – Fachhochschule Bonn-Rhein-Sieg / Fachbereich Informatik, Sankt Augustin, 2004.