Felsen, Steine oder Sand – Messung des optimalen Granularitätsgrades von Software-Services

Moritz Christian Weber
Carola Wondrak

Veröffentlicht in:
Multikonferenz Wirtschaftsinformatik 2012
Tagungsband der MKWI 2012
Hrsg.: Dirk Christian Mattfeld; Susanne Robra-Bissantz

Braunschweig: Institut für Wirtschaftsinformatik, 2012

Felsen, Steine oder Sand – Messung des optimalen Granularitätsgrades von Software-Services

Moritz Christian Weber

Goethe Universität Frankfurt, Lehrstuhl für Betriebswirtschaftslehre insb. e-Finance, 60329 Frankfurt am Main, E-Mail: moweber@wiwi.uni-frankfurt.de

Carola Wondrak

Goethe Universität Frankfurt, Lehrstuhl für Betriebswirtschaftslehre insb. e-Finance, 60329 Frankfurt am Main, E-Mail: cwondrak@wiwi.uni-frankfurt.de

Abstract

Durch den geschickten Entwurf wiederverwendbarer Software können in zukünftigen Entwicklungszyklen Kosten gespart und Mehrwerte generiert werden. Welcher Detailgrad des Entwurfs dabei für die Wiederverwendbarkeit effizient ist, ist bisher nur schwer quantifizierbar. Diese Arbeit nutzt theoretisch fundierte und etablierte Methoden der Portfoliotheorie und wendet sie auf den Bereich der Softwareentwicklung an. Mittels dieser Methoden wird der Restnutzen des wiederverwendbaren Codes bewertet und die Risiken der Reimplementierung verglichen. Durch Anwendung dieser Methoden wird empirisch anhand von drei Softwareprojekten untersucht, wie sich Projektstrukturen auf das Risiko der Reimplementierung auswirken. Das Ziel hierbei ist, den bestmöglichen Granularitätsgrad für Software zu bestimmen.

1 Motivation

Starre, monolithische Informationssysteme prägen den historischen Ursprung betrieblicher IT-Infrastrukturen [16]. Mit zunehmender Vernetzung werden diese unflexiblen Strukturen aufgebrochen, in Teilsysteme untergliedert und in fachspezifische Subsysteme unterteilt [9]. Die Entwicklung hin zu Client-Server- oder Mehrschichtarchitekturen dokumentiert das Bestreben, Geschäftsanwendungen stärker zu modularisieren [16]. Ändern sich betriebliche Anforderungen an das Informationssystem, wird es häufig nicht vollständig ersetzt, sondern lediglich den neuen Gegebenheiten angepasst. Dabei werden Bausteine entfernt, die auf Grund sich wandelnder Anforderungen keinen oder nur geringen Nutzen stiften, und durch neue, adäquate Softwarebausteine ersetzt. Gerade bei prozesskritischen Informationssystemen ist ein vollständiger Austausch einer unternehmensdurchdringenden Anwendung oftmals organisatorisch schwer durchführbar und mit hohen Risiken verbunden.

Ein Paradigma, das diese technischen, wie betriebswirtschaftlichen, Herausforderungen ganzheitlich aufgreift, ist die serviceorientierte Architektur (SOA) [4], die sich zunehmend in Konzepten wie Software-as-a-Service und Cloud-Computing diversifiziert und bereits im betrieblichen Umfeld als Lösung etabliert. „Services sind dabei klar gekapselte und lose gekoppelte Softwarebausteine, die eine definierte Grundfunktionalität über eine standardisierte Schnittstelle bereitstellen“ ([4], S. 187). Erst die Verknüpfung verschiedener Services bildet ein modulares und verteiltes Informationssystem. Im Vergleich zu monolithischen Lösungen ist weder aus technischer, noch aus betriebswirtschaftlicher Sicht geklärt, wie granular die Modularisierung und Verteilung erfolgen soll [12]. Granularität ist im Folgenden definiert als „Anzahl von Untergliederungen eines Elements“ [8]. Bereits frühere Publikationen ([16], S.18) in diesem Bereich hinterfragen die Granularität von Serviceverbünden. Konzepte wie serviceorientierte Analyse und Design [9] prägen Best Practices und Architekturpattern, über die sich mögliche Granularitäten iterativ bestimmen lassen. Um dies empirisch quantifizieren zu können, stellen wir folgende Forschungsfragen:

- Wie granular sollten Software-Services strukturiert sein?
- Wie lässt sich die Güte der Granularität messen bzw. vergleichen?
- Wie kann ein Granularitätsoptimum gegenwärtig bestimmt werden, so dass gute Wiederverwendbarkeit und geringe Änderungsrisiken zu erwarten sind?

Im nachfolgenden Abschnitt wird eine Übersicht über bisherige Untersuchungen in diesem Bereich gegeben. Hieraus werden im dritten Abschnitt quantitative Mess- und Validierungsmethoden abgeleitet. Diese werden auf drei Projekte angewandt und die Ergebnisse im vierten Abschnitt vorgestellt. Abschließend wird die Untersuchung zusammenfassend dargestellt und ein kurzer Forschungsausblick gegeben.

2 Stand der Forschung

Die gegenwärtige Forschung über effiziente Granularität lässt sich in einen gestaltungsorientierten und einen finanzorientierten Bereich unterteilen. In Bezug auf Ersteren wird die gegenwärtige Forschung aus Sicht serviceorientierter Identifikation, Entwurf und Entwicklung dargestellt (Abschnitt 2.1) und mit Untersuchungen bezüglich der Quantifizierbarkeit von Softwaregranularität vertieft (Abschnitt 2.2). Bei Zweiterem wird aus betriebswirtschaftlicher Sicht auf Projektmanagement Bezug genommen (Abschnitt 2.3) und anschließend auf die Portfoliotheorie [21] sowie das Basiskonzept der Risikobewertung (Value-at-Risk) [33] spezialisiert (Abschnitt 2.4).

2.1 SOA Design

Bezüglich SOA Design und Entwicklung existieren sowohl wirtschaftsinformatikorientierte [16], als auch stark technikorientierte [9] Forschungsstränge. Beide Bereiche sind stark auf die Frage der Analyse, Identifikation und des Zuschnitts von Services ([35], S.1, [3], S.1) ausgerichtet. Insbesondere das Design von Schnittstellen [23], Modellierungssprachen, Design Methoden und deren technologischer Unterstützung [25] ist Kern der Untersuchungen. Somit konzentriert sich dieser Forschungsbereich anwendungsbezogen sehr auf die Untersuchung von Entwurfsmustern [3], Best Practices und Lessons Learned [2]. Ansätze zur Bestimmung oder Überprüfung des optimalen Servicezuschnitts sind oftmals deskriptiv [23] und folgen einem langwierigen, iterativen bzw. inkrementellen Ansatz ([32], S.4, [2][9]). Hierdurch kann es zu unerwünschter

Überspezifikation [22] sowie Doppelungen und Inkonsistenzen ([32], S.4) kommen. Zhang et al. verwenden Clustering-Techniken zur Service Identifikation, um architektonische Informationen zu extrahieren und zusammenhängende Module zu identifizieren ([35], S.1). Von Winkler werden Prozessaktivitäten aufgespalten und redundante Logik zu Services gröberer Granularität zusammengefasst, um durch den Entwurf von Software mit ungünstiger Granularität die Gefahr von Überspezifikationen zu vermeiden [32]. So versprechen granularere Services einen höheren Grad an Wiederverwendbarkeit, was gleichzeitig auch zu engeren Kopplungen beim Datenmodell und der Komplexität der Service-Interaktionen führen kann [22]. Entwurfsprinzipien lassen sich aber nicht generell auf alle Formen von Services übertragen. Während dies bei Querschnitts-Enterprise-Services und kanalorientierten Enterprise Services effektiv funktioniert, ist eine entsprechende Umsetzung bezüglich Prozess-Enterprise-Services schwierig, da deren Granularität in der Regel kleiner geschnitten wird, als in klassischen Anwendungssystemen [27].

2.2 Granularitätsanalyse

Andere Ansätze versuchen, die optimale Granularität ([16], S.18) nicht nur aus Design- und Entwicklungssicht zu beurteilen, sondern auch die Güte der Granularität nutzen- bzw. wertmäßig zu quantifizieren: Erradi et al. schlägt eine Quantifizierung mittels einer 10er-Abstufung vor, die unter anderem Abhängigkeiten zwischen den Komponenten berücksichtigt. Außerdem kritisieren sie das Fehlen einer theoretisch fundierten Methode, um die korrekte Granularität für maximale Wiederverwendung zu finden. Alternative Vorgehensweisen gehen auf die selektive Identifikation von Services aus betriebswirtschaftlicher Sicht ein, indem sie Funktionen entsprechend ihres Outsourcing- und Sichtbarkeits-Potenzials bewerten [15]. Dies wird in [6] anhand der Evaluation und Simulation von internen Leistungsverrechnungen empirisch untersucht. Darüber hinaus gibt Friedl darauf aufbauend eine abstrakte Granularitätsempfehlung/-klassifikation entsprechend typischer Einflussfaktoren eines Entwicklungsprojekts [12]. Katzmarik unterstützt diese Granularitätsentscheidung durch mikroökonomische Modelle der Struktur von Software-as-a-Sevice-Produkten [14].

2.3 Projektmanagement

Durch das geschickte Management der Projekte lässt sich die Organisation der Entwicklung von Softwareportfolios verbessern. Lichter und Mandl-Striegnitz zeigen, dass die meisten Projekte ursächlich an organisatorischen Gegebenheiten und weniger an der technischen Umsetzung scheitern [19]. Zusätzlich entstehen ineffiziente Prozesse, die Planungsfehler und unerwartete Kosten verursachen. Ein Abschätzungsmerkmal, wie sich die Kosten eines Planungsfehlers gegenwärtig auf den Entwicklungs- bzw. Anpassungsaufwand zukünftig auswirken, gibt die sogenannte Zehnerregel [24]. Diese schätzt ab, dass ein unentdeckter Planungsfehler in jeder Folgeperiode die notwendigen Kosten verzehnfacht, um diesen Fehler zu bereinigen. Mittels effizientem Projektmanagement und validierbaren Projektkennzahlen lassen sich Fehlentwicklungen frühzeitig erkennen [19].

Insbesondere die Wiederverwendbarkeit bestehender Softwareteile gilt als Mittel zur Reduktion von Kosten und Arbeitseinsatz in mehrperiodigen Softwareentwicklungsprojekten. Notwendigerweise sollten die Software-Artefakte nicht zu kompliziert gestaltet sein und Probleme mit „komplexen Interdependenzen“ [28] berücksichtigt werden. Dies bezieht sowohl Abhängigkeiten zwischen den einzelnen Teilen der Software als auch eine gute Granularisierung mit ein. Insbesondere muss der Struktur des Projektportfolios und der Auswahl von Teilprojekten besondere Aufmerksamkeit gewidmet werden [28].

2.4 Portfoliotheorie und Value-at-Risk

Ein Softwareprojekt besteht, ähnlich wie ein Finanzportfolio, aus mehreren Komponenten, die miteinander verknüpft sind. Jede Teilkomponente stiftet einen Wertbeitrag zum Verbund. Aufgrund des, zum Teil sehr langlebigen, Einsatzes von Software im Unternehmen soll die Auswahl der Teilkomponenten möglichst effizient erfolgen, so dass ein hohes Maß an Wiederverwendung und geringere Wartungskosten zu erwarten sind.

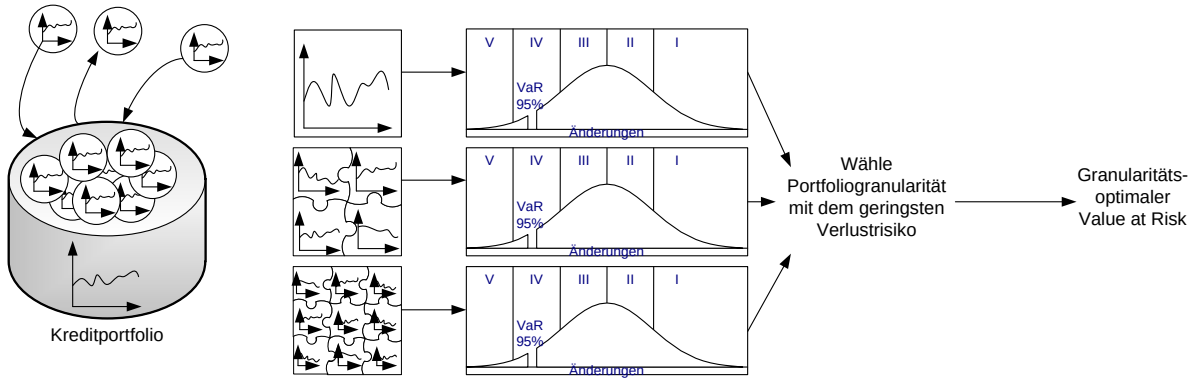

Bild 1: Portfolio- und Risikodiversifikation über Korrelationseffekte

Ein theoretisch fundierter Ansatz zur Auswahl von Teilkomponenten eines Finanzportfolios, liefert die Portfoliotheorie nach Markowitz [21]. Diese erklärt, wie ein optimales Portfolio risikominimal strukturiert werden kann, indem man es effizient in Teilinvestitionen zerlegt (sog. Diversifikation). Dabei werden wechselseitige Einflüsse zwischen den Komponenten berücksichtigt und so zusammengestellt, dass sich das Gesamtrisiko von Wertverlusten minimiert (sog. Korrelation). Diese Wertverluste von Investitionen lassen sich mittels des Value-at-Risk-Ansatzes[1] (VaR) abschätzen (vgl. Bild 1), der auf jedes Risikofaktormodell angewandt werden kann und ein sehr universelles Framework ist [20]. Hierbei wird der maxi-male Wertverlust berechnet, der in einer gegebenen Zeitperiode mit einer bestimmten Wahrscheinlichkeit nicht überschritten wird [31][33]. In Erweiterung des “Expected Loss”-Konzepts, das den Erwartungswert der Verluste ausdrückt, misst der VaR explizit die unerwarteten Verluste [7] und schließt so auch Extremfälle mit ein.

Das Konzept lässt sich, ausgehend von der Einzelbewertung, auf mehrere Investitionstitel bis hin zu ganzen Portfolios anwenden. Liegen mehrere risikobehaftete Anlagen in einem Portfolio vor, so müssen zusätzlich die Synergieeffekte berücksichtigt werden [30]. Schwankungen dieser Komponenten können gleichzeitig Chancen und Risiken für den Gesamtverbund bedeuten. Die Risikostreuung kann aber durch sinnvolle Kombination von Einzelkomponenten erfolgen. Ziel ist es, Synergieeffekte zu finden, die die Risiken untereinander möglichst gut kompensieren [26].

Durch die Selektion verschiedener Portfoliostrukturierungen kann mittels des VaR-Ansatzes die Höhe der Verlustrisiken systematisch abgeschätzt werden. Werden ineffiziente Kombinationen ausgewählt, so entstehen sogenannte Klumpenrisiken [17]. Durch diese unzureichende Granularisierung droht die Gefahr eines höheren diversifizierten VaR [1]. Die Portfoliotheorie hilft, eine granularitätsoptimale Auswahl von Portfoliokomponenten zu finden, so dass Risikokosten von Wertänderungen in der zukünftigen Periode minimiert werden. Von Rockafellar und Uryasev wird ein Methodentransfer des VaR-Konzepts aus dem Kontext der Finanzdomäne in andere Anwendungsfelder explizit gefordert [26].

3 Methodik

Im Folgenden wird, in Anlehnung an die Portfoliotheorie, das Konzept des VaR auf Softwareentwicklungsprojekte übertragen. Mittels der Änderungsdaten aus einem Software-Versionskontrollsystem werden die historischen Varianzen und Kovarianzen der Projektquellcodes gemessen und mittels dreier Methoden alternative Granularitäten simuliert, um den Grad des geringsten Änderungsaufwandsrisikos zu bestimmen.

3.1 Annahmen

Entsprechend der Portfoliotheorie [21] wird angenommen, dass jedes Portfolio, dem man einen impliziten Wert zuordnen kann, so diversifizierbar ist, dass eine anforderungsoptimale oder risikominimale Allokation angenommen werden kann. In der modernen Softwareentwicklung gilt es als guter Stil, jeweils nur eine Klasse, einen Service oder eine Entität in eine Datei zu kapseln [9]. Deshalb wird im Folgenden angenommen, dass jede Änderung einer Datei implizit der Änderung eines Portfolioassets entspricht. Des Weiteren sehen agile Entwicklungspraktiken vor, dass jede Änderung einer Software eine neue fachliche Anforderung oder einen Bugfix umsetzt und diese somit im Versionskontrollsystem dokumentiert. Vereinfachend wird angenommen, dass der Änderungsaufwand an einer Datei stark mit der Anzahl der veränderten und weiterbestehenden Zeilen im Code (Lines of Code (LOC)) korreliert ist. Es wird angenommen, dass Art und Weise der Entwicklung und Überarbeitung im Zeitverlauf ähnlich sind. Historische Änderungsstrukturen sind somit ein guter Schätzwert für das zukünftige Änderungsrisiko. Als heutiger Wert (vergleichbar mit einem Barwert) wird der Umfang der jeweiligen Quellcodedatei angesetzt. Bei historisch bekannten Entwicklungskosten lässt sich die LOC-Rechnung via COCOMO-Methodik in einen konkreten Geldwert umrechnen ([14], S.23).

3.2 Datenerhebung

Zur direkten und empirischen Messung der Struktur- und Entwicklungskultur von Softwareprojekten werden die Daten eines Software-Versionskontrollsystems genutzt. Über dieses lässt sich historisch die Größe jeder Datei, sowie die erfolgten Softwareänderungen exakt messen und somit der Umfang je Anforderungsumsetzung oder Bugfix genau bestimmen. Dabei kann eine Anforderungsumsetzung mehrere Dateiänderungen umfassen. Vergleichbar zu [13], [29] und [34] messen wir in dieser Ausarbeitung den Restnutzen wiederverwendbaren Codes als die Inhalte eines Dokuments, die von einem Autor erstellt wurden und nach Abzug aller Änderungen eines Überarbeiters (Hinzufügungen/Löschungen) noch im Dokument erhalten bleiben. Dieser Restnutzen des wiederverwendbaren Codes wird für jede Datei jeder Anforderungsumsetzung (Patch) ermittelt und gemeinsam mit den Restnutzen der zugehörigen Dateien je Änderung gespeichert (vgl. Bild 2 „Datenerhebung“).

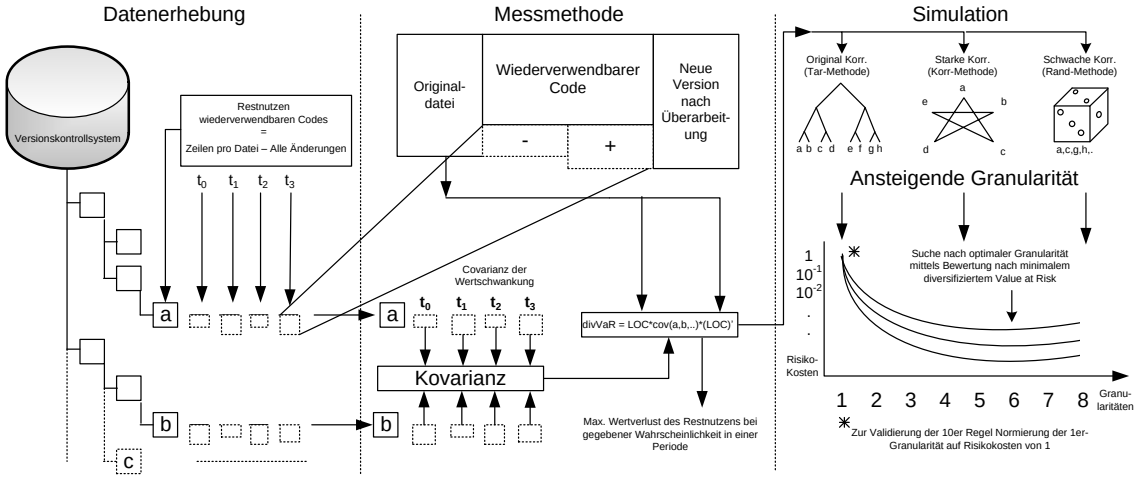

Bild 2: Datenerhebung, Messmethode und Simulation der Untersuchung

3.3 Messmethode

Um zu bestimmen, in welcher Strukturierung der Restnutzen des wiederverwendbaren Codes möglichst groß bleibt, wird das Konzept des diversifizierten VaR angewandt [30]. Dieses bestimmt den Verlust, der in einer prognostizierten Zeitperiode mit einer gegeben Wahrscheinlichkeit nicht unterschritten wird. Dabei fließen historische Schwankungen und korrelative Abhängigkeiten zwischen den Elementen eines Portfolios in die Berechnung ein. Hierzu werden die Schwankungen des Restnutzens wiederverwendbaren Codes genutzt, die in der Datenerhebung bestimmt wurden. Für jeweils paarweise geänderte Dateien wird ein Kovarianz-Wert berechnet und in einer Kovarianzmatrix zusammengefasst [30]. Um den relativen Restnutzen des wiederverwendbaren Portfolio-Codes zu bestimmen, werden mittels Matrizenmultiplikation das Produkt aus Dateiumfängen zum aktuellen Zeitpunkt und der Varianz-Kovarianz-Matrix des relativen Restnutzens des wiederverwendbaren Codes gebildet und das Risikoquantil entsprechend der gegebenen Wahrscheinlichkeit bestimmt [31][33] (vgl. Bild 2 „Messmethode“).

3.4 Simulation

Durch die Projektstruktur ist die Granularität jedes historisch gewachsenen Softwareprojekts implizit gegeben. Für diese lässt sich ein diversifizierter VaR-Wert für den Restnutzen des wiederverwendbaren Codes berechnen. Um die Güte der Granularität zu bewerten, sind vergleichbare Varianten der gegebenen Projektstruktur notwendig, die aus dieser abgeleitet werden müssen. Eine hinreichend effektive Methode (nachfolgend „Tar-Methode“), um Varianten gröberer Granularität gleicher Grundstruktur zu generieren, ist, beginnend mit der tiefsten Ordnerebene, alle Dateien je Ordner zu einer einzigen Datei zusammenzufassen und in die nächsthöhere Ordnerebene zu verschieben [18]. Vereinfachend wird hierzu der Tar-Algorithmus rekursiv angewendet [11]. Paketstrukturen, sowie funktionale Gliederung von Ordnern, begünstigen, dass auch bei umfangreicheren Zusammenfassungen die Abhängigkeiten und zusammenhängenden Änderungen in selber Struktur erhalten bleiben. Die gröbste, und mutmaßlich schlechteste, Granularität wäre nach dieser Methode den gesamten Quelltext in eine Datei zu schreiben. Je Zusammenfassung werden der Umfang der Dateien, sowie die Kovarianzen neu berechnet und mittels VaR-Methode ein Risikowert bestimmt.

Zur Validierung der Tar-Methode werden die Ergebnisse mittels methodischer Triangulation durch zwei weitere Validierungsmethoden überprüft. Bei der ersten Validierungsmethode („Korr-Methode“) wird in wiederholten Simulationsläufen [5] initial eine Datei gewählt. Ausgehend von dieser Datei werden rekursiv in Abhängigkeit geänderte Dateien hinzugefügt. Dies wird wiederholt bis die Anzahl der Dateien, denen der entsprechenden Granularitätsstufen der Tar-Methode entspricht. Bei Zweiterer („Rand-Methode“) werden zufällig Dateien aus dem Projekt gewählt und die Zusammenhänge dabei nicht berücksichtigt. Die erste Methode simuliert hierdurch Teilprojekte des Gesamtprojektes mit einem starken Zusammenhang, die zweite Methode simuliert den Wegfall dieser Zusammenhänge. [17] zeigt, dass bei Veränderung der Korrelationsstrukturen die Bewertung der Granularität mittels VaR kaum beeinflusst wird. Durch Mittelung der Teilergebnisse der Validierungsmethoden wird die Struktur des Gesamtprojekts angenähert [5].

Auch wenn die Validierungsmethoden gute Vergleichswerte zwischen den einzelnen Granularitätsstufen erwarten lassen, so ist zu beachten, dass diese zu aggressiv, zu klumpig und zu schlecht diversifiziert simulieren/abschätzen sowie extreme Projektstrukturen generieren können. Deshalb werden die zwei Validierungs-Methoden ausschließlich zur Triangulierung und Validierung der ersten Methode verwendet, nicht aber für eine Aussage bezüglich der absoluten Höhe des Risikos des relativen Restnutzens des wiederverwendbaren Codes. Zum besseren Vergleich, auch mit Aussagen der Zehnerregel [24], wird die schlechteste Granularisierung je Methode auf eins normiert. Dies ist üblicherweise der Fall, wenn nur eine Datei den gesamten Quellcode beinhaltet ([14], S.29). Die anderen VaR-Werte werden zu diesem ins Verhältnis gesetzt (vgl. Bild 2 „Simulation“).

4 Ergebnisse

4.1 Deskriptive Ergebnisse

Im Folgenden wird die vorgestellte Methodik zur Bestimmung der optimalen Granularität exemplarisch auf drei Softwareentwicklungsprojekte angewandt. Bei der Auswahl dieser Projekte, die allesamt fachlich unterschiedliche Teile einer serviceorientierten Infrastruktur sind, wurde berücksichtigt, dass es jeweils wechselseitige Gemeinsamkeiten (vgl. Hervorhebungen in Tabelle 1) und Unterschiede in der Struktur dieser Projekte gibt. Die Projekte sind durch folgende Eckdaten in Tabelle 1 charakterisiert:

Projekt	Files	LOC	Ordner	Level	Patches	Dateiänderungen	LOC/File
A	1.171	221.446	**213**	**8**	7.200	24.552	**189**
B	3.000	1.208.428	**185**	**8**	**22.169**	**80.127**	403
C	1.974	525.930	708	13	**22.970**	**68.298**	**266**

Tabelle 1: Deskriptive Statistik

Während Projekt A sich durch eine geringere Anzahl an Dateien (1171), LOC (221446) und Ordnerleveln auszeichnet, finden sich in Projekt C und B fast doppelt bzw. dreimal so viele Dateien und mehr als dreifach so viele Patches und Dateiänderungen. Projekt B hat im Vergleich zu den anderen zwei Projekten etwa doppelt solange Quellcodedateien (403 LOC/File) und mehr als doppelt so viele LOC wie Projekt C und fünf Mal so viele LOC wie Projekt A.

Dafür verfügt es über gleiche Tiefe der Ordnerhierarchie wie Projekt A und eine ähnliche Anzahl an Dateien, Patches und Dateiänderungen wie Projekt C. Dieses hat aber mit 13 Levels eine deutlich granularere Ordnerstruktur, als die beiden anderen Projekte. Projekt C hat allerdings mit durchschnittlich 266 LOC/File tendenziell Dateilängen wie Projekt A. Über diese Mischung aus Hetero- und Homogenitäten soll eine breite Menge an Strukturcharakteristika von Softwareprojekten abgedeckt werden.

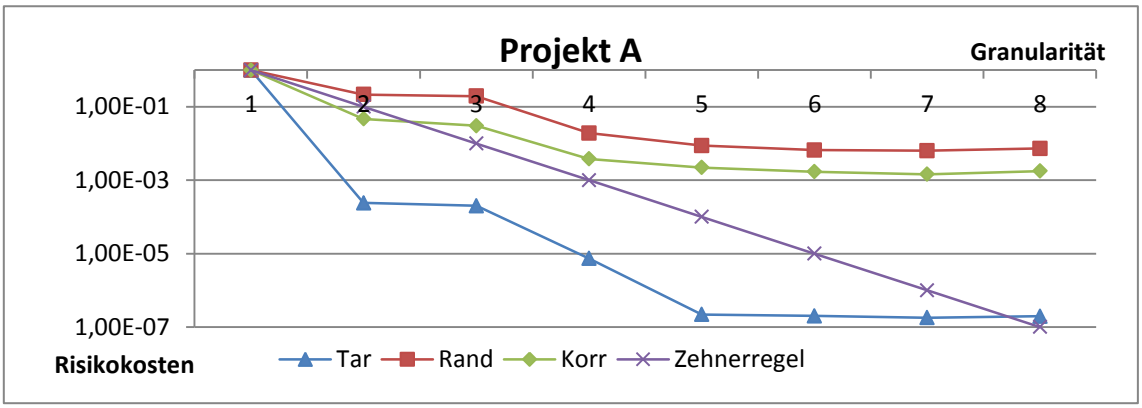

Bild 3: Untersuchungsergebnisse Projekt A

Projekt A (Bild 3) umfasst insgesamt 1171 Files in acht Ordnerstufen und somit acht, in allen Methoden simulierbare, Granularitätsgrade. Nach einem degressiven Abfall der Risikokosten zwischen der ersten und fünften Granularitätsstufe, stabilisieren sich die Ergebnisse von der fünften bis zur achten Granularitätsstufe bei allen drei Methoden. Das lokale Optimum findet sich bei allen drei Methoden bei Granularitätsstufe sieben, was 1163 Dateien entspricht.

Projekt B (Bild 4) umfasst ebenfalls acht simulierbare Granularitätsstufen. Auch dort stabilisieren sich die Risikokosten nach degressivem Abfall und finden bei der siebten Granularitätsstufe ein lokales Minimum (2775 Dateien). Projekte C (Bild 5) umfasst mit dreizehn deutlich mehr simulierbare Stufen. Hier stabilisieren sich die Granularitäten ab der sechsten Granularitätsstufe und das Optimum liegt hier ebenfalls auf der vorletzen Granularitätsstufen (1888 Dateien). Anhand der Anzahl der Änderungen lässt sich nachvollziehen, dass Projekt B und C deutlich länger softwaretechnisch gereift sind und somit länger iterativ einem Optimum annähern konnten.

4.2 Zehnerregel als Maß für die Kosten falscher Granularität

Wie erwartet, sinken die diversifizierten Risikokosten mit steigender Granularität. Die Ergebnisse der drei Messmethoden sind in Bild 3-5 mit den Vergleichswerten der theoretischen Zehnerregel dargestellt. Dabei wird die Granularität vom Grad eins als schlechtester Granularitätsgrad angesetzt und die Schätzung der Kosten der Zehnerregel mit den nachfolgenden Granularitätswerten auf 10% des Vorwerts gesenkt. Generell ist zu erkennen, dass die theoretische Zehnerregel die Risikokosten abhängig von der Granularität systematisch über- oder unterschätzt. Auch lässt sich erkennen, dass die Zehnerregel die Höhe der Risikokosten dem Wert Tar-Methode am besten im Optimum approximiert. Gleichzeitig zeigen die drei Projekte, dass diese Daumenregel die Höhe der Risikokosten zwischen schlechtester und optimaler Granularität aber auch deutlich unterbewerten (Bild 3) oder überbewerten (Bild 4) kann. Dies ist insbesondere dann der Fall, wenn mehrere Granularitätsstufen messbar ähnliche Risikokosten haben.

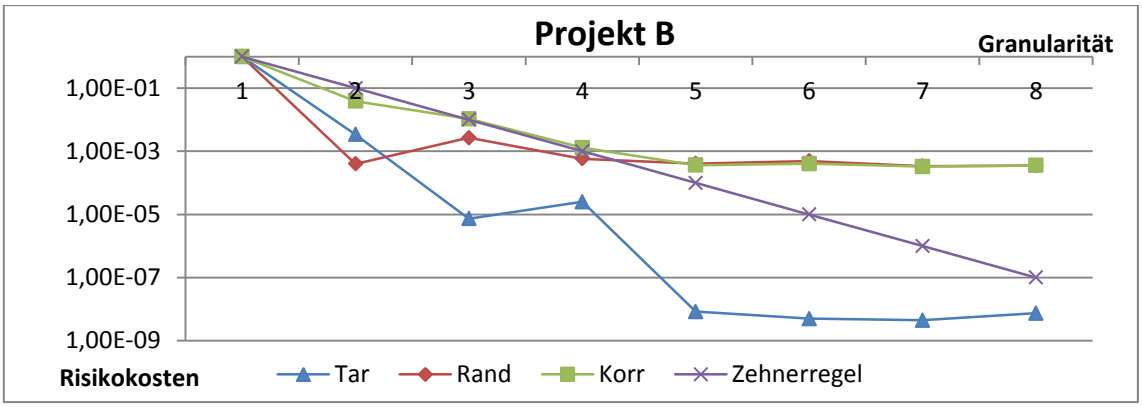

Bild 4: Untersuchungsergebnisse Projekt B

Ebenfalls lässt sich zeigen, dass die Validierungsmethoden, auf Grund der Randomisierung oftmals aggressiver, klumpiger und schlechter diversifiziert simulieren als die Zehnerregel oder die Tar-Methode. Dass die Validierungsmethoden trotz Randomisierung das Minimum der Tar-Methode bestätigen können, deckt sich mit anderen empirischen Untersuchungen [17]. Im Abgleich mit der Zehnerregel kann somit bestätigt werden, dass die Tar-Methode die Höhe von Granularitätsrisiken besser abschätzt, aber auch die Validierungsmethoden, die den gleichen Granularitätsgrad als optimal bestimmen, soweit die Höhe der Granularitätsrisiken zweitrangig ist.

4.3 Grenznutzen der Granularisierung

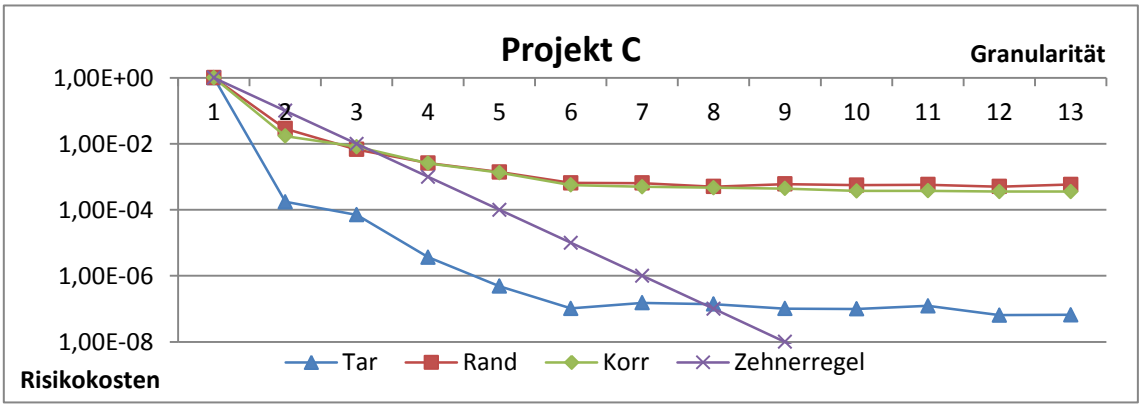

Bild 5: Untersuchungsergebnisse Projekt C

Die Daten zeigen, dass mit steigender Granularität die Risikokosten erst sinken und sich dann auf eher stationärem Niveau mit lokalem Minimum stabilisieren. Es lässt sich intuitiv nachvollziehen, dass Projekte solange mit Best Practice Methoden und iterativen Ansätzen untergliedert werden, bis eine handhabbare Softwareprojektbasis entsteht. Gleichzeitig zeigt das stationäre Niveau aller drei Projekte, dass eine annähernd gute Granularität oftmals auch schon auf niedrigeren Granularitätstufen erreicht werden kann. Der Grenznutzen weiterer Granularisierung in Richtung des Optimums nimmt aufgrund steigender Koordinationskomplexität ab. Allen drei Projekten ist gemein, dass die, in den Projekten konkret vorliegende und entsprechend der Bewertung maximale Granularität zwar hinreichend gut, aber im Vergleich zum Optimum etwas zu feingliedrig ist. Dies ist ein möglicher Effekt der, in der Literatur diskutierten, Überspezifikation [22]. Softwarestrukturen, die ursprünglich für Wiederverwendung angelegt wurden, aber dann nie wiederverwendet werden, erzeugen einen ineffizienten Overhead. Die Vermessung der drei Projekte zeigt, dass eine leichte Über- oder Unterspezifikation der optimalen Granularität

vertretbar sein kann. Zwar erhöhen sich die Risikokosten rund um den optimalen Granularitätspunkt leicht, diese Erhöhung ist aber weit geringer, als bei niedriger oder gar monolithischer Granularität.

5 Zusammenfassung

Bezüglich der Strukturierung von Softwareprojekten wurden drei Forschungsfragen motiviert. Diese betreffen die Granularität der Struktur von Software-Services, die Mess- und Vergleichbarkeit der Güte von Granularität sowie das Granularitätsoptimum, welches gute Wiederverwendbarkeit und geringe Änderungsrisiken verspricht. Dazu wurden in Abschnitt 2 aktuelle Untersuchungsansätze dargestellt. Einerseits führen gestaltungsorientierte Ansätze und Best-Practices-Erkenntnisse zu guten Services- und Architektur-Designs. Andererseits motivieren erste quantitative Beiträge zu Granularitätsmetriken eine tiefgreifendere quantitative Untersuchung. In dem vorliegenden Beitrag wurden diese Ansätze mittels Erkenntnissen des Projektportfoliomanagements zu Projektrisikokosten, sowie der Portfoliotheorie und der Methode des VaR erweitert. Hierbei wurde, basierend auf der etablierten Portfoliotheorie [21], das Risiko der Reimplementierung anhand des relativen Restnutzens wiederverwendbaren Codes bewertet. Effiziente Granularität durch die Wahl der richtigen Softwareprojektstruktur wird mittels Bewertung der Risikokosten messbar.

Zur empirischen Quantifizierbarkeit wurde der Umfang von Software-Komponenten und der historischen Software-Änderungen mit Hilfe von Software-Versionskontrollsystemen gemessen. Drei Softwareprojekte wurden mittels drei Methoden in alternativen Granularitäten simuliert und entsprechend des Risikos der Reimplementierung bewertet.

Die Untersuchung zeigt, dass, trotz der Unterschiedlichkeit der Methoden, jeweils in allen drei Projekten ein Grad der Granularisierung als optimal bewertet wird, auch wenn nicht alle Methoden die Höhe der Risikokosten exakt quantifizieren. Gleichzeitig kann die Zehnerregel des Softwareprojektmanagements empirisch zu Teilen bestätigt werden, auch wenn die logarithmische Struktur dieser Regel nur zu Teilen die Charakteristika von Softwareprojekten korrekt widerspiegelt. Zusätzlich zeigt sich, dass es einen Grenznutzen der Granularisierung gibt und dass der Nutzen rund um das Optimum nur sehr schwach abfällt.

Die vorgestellte Methode ist ein Ansatz zur Quantifizierung der Granulariätsgüte. Weitere Untersuchungen müssen zeigen, wie die Spezifika aus Projektstruktur und Methode variieren. Es ist kritisch anzumerken, dass dieser Ansatz nur Projektrisiken betrachtet, mögliche Renditen aber nicht, da diese nicht in den Softwareversionsdaten messbar sind. Die Messmethoden können auch nur Granularitätsgrade bewerten, die zwischen 1 und der gegebenen Projektgranularität der historischen Daten liegen.

Desweiteren soll die Untersuchung auf eine größere Anzahl an Projekten ausgeweitet werden. Ziel ist es, empirisch zu untersuchen, in welcher Art und Weise Projekt-Charakteristika das Risiko der Reimplementierung und somit den Grad der optimalen, sowie den Bereich der guten Granularität beeinflussen. Ziel ist eine Daumenregel, die auch ohne historisches Software-Repository eine empirische, belegte Abschätzung der Güte der Projektgranularität liefert. Bezüglich der gestellten Forschungsfragen konnte gezeigt werden, dass sich ein Granularitätsoptimum gegenwärtig bestimmen lässt, sodass gute Wiederverwendbarkeit und geringe Änderungsrisiken zu erwarten sind. Es wurde eine Methode vorgestellt, welche die Güte der Granularität messbar macht und somit eine quantitative Aussage erlaubt, wie granular Software-Services strukturiert sein sollen.

6 Literatur

[1] Albrecht, P.; Maurer, R. (2005): Investment- und Risikomanagement: Modelle, Methoden, Anwendungen: Schäffer-Poeschel; Auflage: 2. ,2005.

[2] Arsanjani, A. et.al. (2008): SOMA: A method for developing service-oriented solutions, IBM Systems Journal, Vol 47, Nr. 3, 2008.

[3] Arsanjani, A. (1998): Service-oriented modeling and architecture, IBM, 2005.

[4] Becker, A.; Widjaja, T.; Buxmann, P. (2011): Nutzenpotenziale und Herausforderungen des Einsatzes von Serviceorientierten Architekturen, WIRTSCHAFTSINFORMATIK, 2011/04.

[5] Blobel, V.; Lohrmann, E. (1998): Statistische und numerische Methoden der Datenanalyse; Teubner, 1998.

[6] Braunwarth, K.; Friedl, B. (2010): Towards a financially optimal design of IT services, ,ICIS, 2010, Saint Louis.

[7] Campbell,R.;Huisman, R..; Koedijik,K. (2000): Optimal portfolio selection in a Value-at-Risk framework. Journal of Banking& Finance 25(2001).

[8] Duden (2007), Stichwort “Granularität”, Brockhaus, 2007.

[9] Erl,T. (2005): Service-Oriented Architecture: Concepts, Technology & Design, Prentice Hall,2005.

[10] Erradi, A.; Anand, S.; Kulkarni, N. (2006): SOAF: An Architectural Framework for Service Definition and Realization.

[11] FSF (2006): TAR; http://www.gnu.org/software/tar/tar.html; Abruf: 06.08.2011.

[12] Friedl, B. (2011): Zur optimalen Granularität von IT-Services - Eine Analyse relevanter ökonomischer Einflussfaktoren, 10. Int. Tagung Wirtschaftsinformatik, Zürich, 2011.

[13] Grigore, M.; Rosenkranz, C. (2011): Increasing the Willingness to Collaborate Online: Analysis of Sentiment-Driven Interactions in Peer Content Production. ICIS 2011.

[14] Katzmarzik (2011): Product differentiation for Software-as-a-Service Providers, BISE, 01/2011, S. 19-31.

[15] Klose, K.; Knackstedt, R.; Beverungen, D. (2007): Identification of Services - A Stakeholder-based approach to SOA development and its application in the area of production planning, ECIS, 2007.

[16] Krafzig, D.; Banke, K.; Slama, D. (2004): Enterprise SOA: Service Oriented Architecture Best Practices, Prentice Hall International, 2004.

[17] Huschens, S.; Stahl, G.(2004): Granularität dominiert Korrelation, Risknews 06/04.

[18] Lee, J.; Seo, Y.; Lee, J. (2007): Method and appartus for merging data objects, European Patent EP181840A2, 2007.

[19] Lichter, H.; Mandl-Striegnitz, P. (1999): Defizite im Software- Projektmanagement - Erfahrungen aus einer industriellen Studie, Informatique 5/1999.

[20] Lütkebohmert, E.; Gordy, M. (2007). Granularity adjustment for Basel II. No 2007,01, Discussion Paper Series 2: Banking and Financial Studies, Deutsche Bundesbank, Research Centre.

[21] Markowitz, H. (1952): Portfolio Selection: Journal of Finance,1952, S. 77-91.

[22] Millard, D. E. et. al. (2009): Pragmatic web service design: An agile approach with the service responsibility and interaction design method. Computer Science, 2009.

[23] Offermann, P. (2008): SOAM - Eine Methode zur Konzeption betrieblicher Software mit einer Serviceorientierten Architektur, WIRTSCHAFTSINFORMATIK, 2008/06.

[24] Pfeifer, T. (2001): Qualitätsmanagement: Strategien, Methoden, Techniken, Hanser Fachbuch, 3. Auflage,2001.

[25] Quartel, D.; Dijkman, R.; van Sinderen, M. (2004): Methodological support for service-oriented design in ISDL, ICSOC, 2004, New York.

[26] Rockafellar, R.; Uryasev,S. (1999): Optimization of Cond. Value-at-Risk, USA, 1999.

[27] Schelp, J.; Winter, R. (2007): Entwurf von Anwendungssystemen und Entwurf von Enterprise Services - Ähnlichkeiten und Unterschiede. WIRTSCHAFTSINFORMATIK Ausgabe Nr.: 2008-01.

[28] Schirmer, I.; Zimmermann, K. (2008):Visualisierung von Projektportfolios zur Unterstützung des Architekturmanagements, Universität Hamburg.

[29] Turek, P.; Wierzbicki, A.; Nielek, R. (2010): WikiTeams: Evaluating Teamwork in Wikipedia. In: Proceedings of the WebSci10, 2010.

[30] Wiedemann, A. (2002): Messung von Zinsrisiken mit dem Value-at-Risk-Konzept, WISU 12/2002, S. 1548-1553.

[31] Wilkens, M. et. al. (2001): Basel II- Berücksichtigung von Diversifikationseffekten im Kredit-portfolio durch das Granularity Adjustment, Zeitschrift für das gesamte Kreditwesen, 12/2001 ; S. 670-676.

[32] Winkler, V. (2007): Identifikation und Gestaltung von Services. Vorgehen und beispielhafte Anwendung im Finanzdienstleistungsbereich. Wirtschaftsinformatik 49, 4, 257-266.

[33] Wolke, T. (2008): Risikomanagement; Oldenbourg Wissenschaftsverlag, 07/2008.

[34] Wierzbicki, A.; Turek, P.; Nielek, R.. (2010): Learning about team collaboration from Wikipedia edit history; International Symposium on Wikis and Open Collaboration, 2010.

[35] Zhang, Z.; Lui, R.; Yanz, Z. (2005): Service Identification and Packaging in Service Oriented Reengineering. Proceedings of the 17th International Conference on Software Engineering und Knowledge Engineering, SEKE'05.