

Softwaremessung, quantitative Projektsteuerung und Benchmarking

Wie helfen sie dem Software-Projektmanager?

Siegfried Seibert

Zum Benchmarking von Softwareprojekten haben quantitative Prozessmessungen bisher nur eine geringe Verbreitung gefunden; zu Unrecht, wie der Verfasser meint. Der Anwendungsbereich von Softwaremessungen geht dabei weit über das Projektbenchmarking hinaus. Mit ihnen kann ein gravierendes Problem des IT-Projektmanagements bekämpft werden: Studien zeigen regelmäßig, dass die meisten IT-Projekte ihre Termine und ihre Budgets deutlich überziehen. Quantitative Prozessmessungen und Benchmarks können helfen, die Erfolgsbilanz anzuheben sowie Produktivität und Qualität der Entwicklungsprozesse zu verbessern.

Was wollen Kunden und Vorgesetzte von einem Projektleiter gewöhnlich wissen? Sie stellen meist Fragen wie die folgenden:

- „Wann ist das System einsatzbereit? ... Wird das auch klappen?“
- „Wie viele Testzyklen brauchen wir noch? ... Ich meine, wie viele genau?“
- „Werden wir alle Anforderungen erfüllen? ... Wie viele sind schon realisiert?“
- „Wir müssen uns auf die fehlerintensiven Module konzentrieren. Welche sind das?“

Kunden und Vorgesetzte interessiert also weniger, wie das Projektmanagement gemacht wird. Sie interessiert in erster Linie, ob die Ergebnisse stimmen und dass gegebene Zusagen eingehalten werden.

Softwaremessung und PM-Assessmentmodelle

Dazu braucht der Projektleiter geeignete Instrumente, mit denen er die Projektsituation beurteilen und die weitere Entwicklung abschätzen kann. Projektmanagementbezogene Assessmentssysteme wie das „Project-Excellence“-Modell oder PM Delta der GPM sowie das Project Management Maturity Model (PMMM) von Kerzner und andere verwandte Modelle [1] fordern zwar die Anwendung derartiger Instrumente und Methoden, stellen sie aber nicht bereit. Diese Modelle zeigen auf, wie ein Projekt professionell gemanagt werden soll, konzentrieren sich dabei jedoch auf die indirekten Leitungs- und Unterstützungsprozesse. Die wertschöpfenden Arbeitsprozesse und die Projektergebnisse bleiben außer Betracht oder werden lediglich qualitativ beurteilt.

Gerade die wertschöpfenden Prozesse und die Projektergebnisse muss ein Projektleiter aber im Griff haben, wenn er erfolgreich sein will. Dazu ist auf intuitive

Abschätzungen wenig Verlass. Selbst wenn sie von erfahrenen Projektleitern kommen, sind sie bei größeren Projekten äußerst unsicher und manipulationsanfällig. Projekt-Aufwandsschätzungen, Assessments und Benchmarks sollten daher immer auch auf quantitativen Daten (Metriken) beruhen. Im Softwarebereich versteht man unter dem Begriff „Metrik“ die Messung der Eigenschaften eines Softwareprodukts und seines Entwicklungsprozesses. In der Unternehmenspraxis sind Metriken noch wenig verbreitet. In den Vereinigten Staaten wenden Metriken nur etwa 15 bis 20 % der Software-Entwickler an [2]. Deutschland ist im internationalen Bereich noch schwächer vertreten. Die auf diesem Gebiet führende „Deutschsprachige Anwendergruppe für Softwaremetrik und Aufwandsschätzung“ DASMA hat nur etwa 50 Mitglieder [3]. Auch in der Projektmanagementliteratur findet man nur selten Titel zu diesem Themengebiet. Wenn sich damit beschäftigt wird, dann von Softwaretechnikern, beispielsweise der Fachgruppe 2.1.10 „Software-Messung und Bewertung“ der Gesellschaft für Informatik GI [4].

Softwaremessung im Capability Maturity Model CMM

Ziel des vorliegenden Beitrags ist es daher, die Relevanz von Softwaremessungen für das Projektmanagement aufzuzeigen. Dabei wird immer wieder auf das Capability Maturity Model CMM Bezug genommen. Das CMM ist ein Katalog von Best Practices, das in fünf aufeinander aufbauende Reifegradstufen unterteilt ist (Abb. 1). Das Modell dient der Verbesserung von Qualität und Leistungsfähigkeit von Software-Entwicklungen. Seit seiner Entstehung Ende der 80er Jahre hat das CMM weltweit eine immer größere Verbreitung gefunden und ist mittlerweile ein Quasistandard zur Zertifizierung von Software-Organisationen geworden.

In den Best Practices des CMM sind Softwaremessungen ein wichtiger, aber nicht sofort offenkundiger Bestandteil. Auf CMM-Stufe 2 werden im Rahmen der Software-Projektplanung und -verfolgung Produktgrößen-, Aufwands-, Kosten- und Terminschätzungen sowie die Erhebung und Überwachung dieser Messgrößen nach dokumentierten Verfahrensrichtlinien gefordert. Auf Stufe 3 wird gefordert, dass Projektschätzungen auf Basis eigener Erfahrungsdaten vorgenommen werden. Auf Stufe 4 wird ein metrikbasiertes, quantitatives Prozessmanagement gefordert. Spätestens zu der auf Stufe 5 geforderten Prozessverbesserung sind zusätzliche quantitative Benchmarks erforderlich [5]. Im Folgenden sollen Einsatz und Nutzen von SW-Messungen für das Projektmanagement in grober Anlehnung an die Reifegradstufen des CMM erläutert werden. Wichtig dabei ist, dass die metrikbasierten Forderungen auf jeder Stufe in ein aufeinander abgestimmtes Gesamtpaket an Best Practices eingebunden sind.

Um welche Messgrößen geht es?

Um einen Überblick über die wichtigsten Messgrößen zu geben, wird zwischen Produktmetriken und Prozessmetriken unterschieden.

Produktmetriken

Produktmetriken beziehen sich auf das Ergebnis eines Softwareprojekts, insbesondere dessen Größe, Komplexität und Qualität.

Die **Produktgröße** ist eine unverzichtbare Grundgröße für jede Projektmessung, weil nur durch ihre Berücksichtigung größere und kleinere Projekte miteinander vergleichbar gemacht werden können. Zu ihrer Erfassung gibt es zwei verschiedene Ansätze:

- Die technische Produktgröße, gemessen als Anzahl der Befehlszeilen (Einheit KLOC = Kilo Lines of Code).
- Die funktionale Produktgröße, gemessen in **Funktionspunkten** oder damit verwandten Einheiten.

Beide Größen sind nützliche Vorhersageindikatoren für den Projektaufwand und lassen sich bei Kenntnis der Programmiersprache ineinander umrechnen („Backfiring“). Vorteile der Befehlszeilen sind ihr höherer Verbreitungsgrad und die Möglichkeit, sie automatisiert zu messen. Demgegenüber können Funktionspunkte bereits zu frühen Projektzeitpunkten durch Auszählung des Lastenhefts ermittelt werden. Außerdem sind sie für einen fachkundigen Kunden einfacher nachvollziehbar als Befehlszeilen. Sie eignen sich daher auch als Abrechnungsgrundlage in Kunden-Lieferanten-Projekten. In verschiedenen Ländern (z.B. Australien, Italien) wird ihre Angabe bei Angeboten für öffentliche Ausschreibungen sogar gefordert.

Eine weitere aufwandsbeeinflussende Größenmetrik, die einfach zu bestimmen ist, ist der **Seitenumfang** der System- und Benutzerdokumentation.

Zur Messung der **Produktkomplexität** wird meist die Anzahl der Schnittstellen zwischen den verschiedenen Systemkomponenten (z.B. Modulen, Klassen, Methoden) verwendet. Ein einfaches Komplexitätsmaß ist beispielsweise die zyklomatische Komplexität nach McCabe, die sowohl für Programm-Module und Sys-

| CMM-Stufe | System-erstellung | Qualitäts-sicherung | Projekt-management | Unternehmens-organisation |
|-----------------------|--|--------------------------------|---|--|
| 5 Optimierend | Technologie-innovation | Fehler-vermeidung | | Prozess-verbesserung |
| 4 Gesteuert | | SW-Qualitäts-management | Quantitative Prozess-Steuerung | |
| 3 Definiert | SW-Produkt-Engineering | Peer Reviews | Integriertes SW-Management Intergruppen-koordination | Prozess-Ausrichtung und -Definition Trainings-programme |
| 2 Wiederholbar | Anforderungs-management Konfigurations-management | SW-Qualitäts-sicherungs-system | Projektplanung Projektverfolgung Unterauftrags-management | |
| 1 Initial | Chaotische Ad-hoc-Prozesse | | | |

Abb. 1: Reifegrade und Best Practices im Capability Maturity Model CMM

temmentwürfe als auch für Geschäftsprozessanalysen ermittelt werden kann. Mit Komplexitätsmetriken lassen sich diejenigen Programmteile identifizieren, die besonders fehleranfällig sind. Der Projektleiter gewinnt damit ein Instrument, mit dem er die aufwändigen Code-Inspektionen auf diejenigen Module konzentrieren kann, die davon am meisten profitieren. Weniger komplexe Module können mit einfacheren Mitteln überprüft werden [6].

Als Metrik für die **Produktqualität** wird am häufigsten auf die **Fehleranzahl** verwiesen, unterteilt nach Fehlerursachen und Fehlerschwere. Als Produktmetrik erfordert sie eine nicht unproblematische Abschätzung der in einem System inhärenten Restfehler. Häufiger werden Fehlermetriken daher als Prozessmetriken in der Systemtestphase eingesetzt. Dabei werden die Anzahl der neuen Fehlermeldungen und die Anzahl der als behoben gemeldeten Fehler im Wochenrhythmus gegenübergestellt. Dies ermöglicht relativ verlässliche Prognosen des Freigabezeitpunkts.

Neben Fehlerzählungen und -prognosen werden als Qualitätsmetriken häufig auch **Kundenreklamationen** und **Kundenbefragungen** verwendet. Beispielsweise berichtet Cohn [7] über einen Index, bei dem die schwerwiegenderen A- und B-Reklamationen mit der Zeitdauer gewichtet werden, seit der eine Reklamation eingegangen, aber noch nicht behoben ist. Mit diesem Index konnten Reklamationsbearbeitung und Kundenzufriedenheit innerhalb kurzer Zeit deutlich gesteigert werden.

Prozessmetriken

Prozessmetriken beziehen sich auf Aktivitäten und Instrumente im Software-Entwicklungsprozess. Hierzu zählen zum einen Ressourcenverbrauch, Aufwand, Kosten und Dauer der Projektaktivitäten. Zum anderen zählen hierzu aber auch Metriken zur Verfolgung des Projektfortschritts, der Projektdynamik und des Projektklimas:

- Der Projektfortschritt kann verfolgt werden, indem die Anzahl der definierten, entworfenen, codierten, geprüften und getesteten Module oder die Anzahl der fertiggestellten Code-Zeilen und Dokumentationsseiten gezählt werden. Auch die oben erwähnte Verfolgung von Fehlermeldungen oder von Testaktivitäten und von Reviews liefert Indikatoren für den Projektfortschritt.
- Die Projektdynamik kann anhand der Anzahl beantragter, genehmigter und realisierter Änderungen gemessen werden.
- Zur Verfolgung des Projektklimas kann durch regelmäßige Team- und Stakeholderbefragungen ein Klimaindex ermittelt werden [8].

Software-Metriken zur Aufwandsschätzung

Was kann der Projektleiter nun mit derartigen Softwaremessungen anfangen? Ein erster, wichtiger Anwendungsbereich ist die Verbesserung der Projekt-Aufwandsschätzung. Aufwand, Kosten und Termine von Software-Entwicklungen können auf zwei Wegen geschätzt werden:

- Expertenschätzungen der einzelnen Arbeitspakete des Projektstrukturplans.
- Parametrische Schätzgleichungen, die eine mathematische Beziehung zwischen Projektaufwand, Projektgröße und anderen Einflussfaktoren herstellen.

Während die in Deutschland vorherrschenden Expertenschätzungen bei kleineren Projekten im Vordergrund stehen, dienen parametrische Methoden vornehmlich der Schätzung großer Projekte in frühen Projektphasen. Hier ermöglichen sie bei sorgfältiger Anwendung besse-

re und schnellere Ergebnisse als Expertenschätzungen, da zu diesem Zeitpunkt noch keine ausgereiften Projektstrukturpläne vorliegen. Bekannte parametrische Schätzsysteme sind beispielsweise COCOMO II, PRICE, KnowledgePlan und SLIM. Zur Schätzung von Aufwand und Zeitdauer eines Projekts werden dabei von allen Systemen ähnliche Gleichungen verwendet. Exemplarisch sei dies anhand der COCOMO-Grundgleichung verdeutlicht [9]:

Aufwand = PK · KLOC^{KE} · AM, mit

- Aufwand in Personenmonaten,
- PK = Produktivitätskoeffizient (= 2,94),
- KLOC = 1.000 logische Programmbefehle,
- KE = Komplexitätsexponent (= 1,1),
- AM = Aufwandsmultiplikator (Produkt aus ca. 20 Einzeleinflüssen).

Der Aufwandsmultiplikator AM liegt bei mittleren Projekten bei eins und spielt für unsere Betrachtung keine wesentliche Rolle. Auf ihn soll daher nicht weiter eingegangen werden.

Der Produktivitätskoeffizient PK und der Komplexitätsexponent KE sind Maße, mit denen die Produktivität einer Softwareorganisation bei zunehmender Projektgröße beziffert werden kann. Die dafür von COCOMO angegebenen mittleren Werte 2,94 und 1,1 wurden durch die statistische Auswertung von rund 160 abgeschlossenen Projekten bestimmt. Ein Anwender der COCOMO-Gleichungen kann zunächst nicht beurteilen, ob in seinem Unternehmen vergleichbare Produktivitätsraten vorliegen. Nur dann wäre die direkte Verwendung dieser Koeffizienten gerechtfertigt. Für eine hohe Schätzgenauigkeit müssen die Koeffizienten der Schätzgleichungen daher an die im Unternehmen vorliegenden Randbedingungen angepasst werden. Dies geschieht durch die als Kalibrierung bezeichnete Auswertung eigener Projektdaten.

Kalibrierung von Schätzgleichungen

Mathematisch gesehen sind bei der Kalibrierung der Produktivitätskoeffizient und der Komplexitätsexponent die zu ermittelnden Größen. Die Anzahl der Programmbefehle und der Aufwand sollten für eine Reihe abgeschlossener Projekte durch die Auswertung von deren Istdaten bekannt sein. Mithilfe eines Tabellenkalkulationsprogramms kann dann eine Kalibrierung auch ohne größere mathematische Kenntnisse vorgenommen werden (Abb. 2): Die erhobenen Datenpunkte werden eingegeben und daraus ein x-y-Diagramm mit einer exponentiellen Trendlinie abgeleitet (bei doppellogarithmischer Skalierung eine Gerade). Für diese Trendlinie werden vom Programm die Gleichungskoeffizienten und das statistische Bestimmtheitsmaß ausgewiesen. Fertig ist die eigene parametrische Schätzgleichung.

Deutlich erhöhte Schätzgenauigkeit

Welchen Gewinn bringt eine solche Kalibrierung? Bei Siemens zeigte eine Analyse von 60 Projekten mit den Original-COCOMO-Koeffizienten nur in 43 % der Fälle eine Schätzgenauigkeit von $\pm 20\%$. Nachdem die COCOMO-Gleichungen mit siemenseigenen Daten

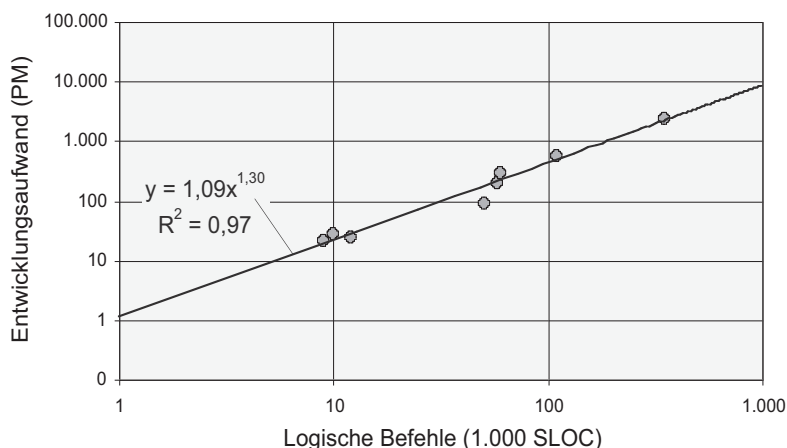


Abb. 2: Kalibrierung von Schätzgleichungen

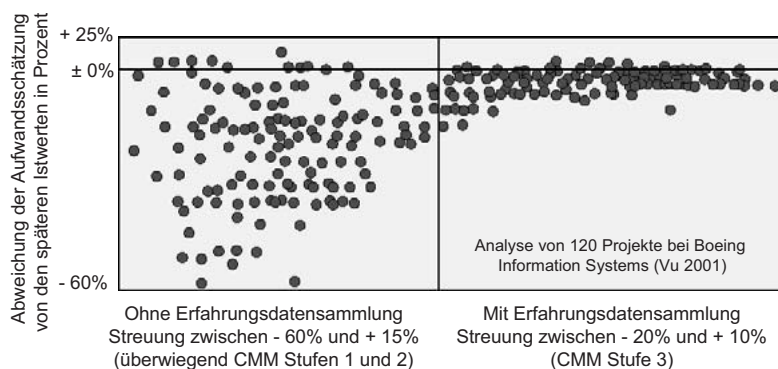


Abb. 3: Verbesserte Schätzungen durch Erfahrungsdatensammlung

kalibriert waren, lagen 95 % der Fälle innerhalb der 20 %-Bandbreite [10, S. 417].

In Untersuchungen von Jones [2, S. 135 f.] und bei Boeing Information Systems [11] führte der Übergang von intuitiven auf erfahrungsdatengestützte Schätzungen zu ähnlichen Verbesserungen der Genauigkeit.

Abb. 3 zeigt die Ergebnisse der Boeing-Studie, bei der 120 Projekte ausgewertet wurden. Die Schätzwerte streuen dabei nicht symmetrisch um die Istwerte. Vielmehr sind die ursprünglichen Schätzwerte niedriger, die späteren Istwerte deutlich höher.

Die Unterschätzung des Projektaufwands ist demnach weit verbreitet. Hierin liegt eine der wichtigsten Ursachen der vielbeklagten Termin- und Kostenüberschreitungen von Softwareprojekten. Eine andere Folgerung: Intuitive Expertenschätzungen liefern regelmäßig niedrigere Werte als erfahrungsdatengestützte Schätzungen. Zum einen liegt dies daran, dass Entwickler bei intuitiven Schätzungen meist ihre Produktivität überschätzen. Zum anderen stoßen hohe Schätzungen häufig auf Widerstände bei Management, Vertrieb und Kunden. „Weiche“ intuitive Schätzungen lassen sich dann von diesen Stakeholdern sehr viel leichter nach unten „korrigieren“ als „harte“ erfahrungsdatengestützte Werte. Wenn anhand eigener Daten belegt wird, dass die Produktivität im Vergleich zu den letzten zehn Projekten verdoppelt werden müsste, kann dies nicht mehr so einfach beiseite geschoben werden.

Niedrige Vorgaben sind kein Kavaliersdelikt

In diesem Zusammenhang sollte auch der in Management und Vertrieb weit verbreitete Irrglaube ad acta gelegt werden, mit zu niedrig angesetzten Planvorgaben die betriebliche Produktivität bis zu ihrem Optimum ausreizen zu können. Zu niedrige Planwerte führen dazu, dass wichtige fachliche, soziale und konzeptionelle Aufgaben in der Startphase vernachlässigt werden, was später zu einem erhöhten „Nacharbeitsaufwand“ führt. Wenn dann der zugesagte Termin nicht eingehalten wird, kommt es zu zusätzlichen Rückfragen des Kunden sowie Analyse-, Planungs- und Krisensitzungen im Unternehmen. Ungerechtfertigter Druck des Managements führt zu Verärgerung und erhöhter Personalfuktuation, und zwar immer zuerst bei den Leistungsträgern im Team. Als Microsoft sein Programm Word für Windows, Version 1.0, entwickelte, wurde auf die Entwickler ein immenser Termindruck ausgeübt. Viele stellten daraufhin codierte Module ohne sorgfältige Prüfung als fertig in die Systembibliothek ein. Folge war, dass die anschließenden Systemtests aufgrund der vielen

zu behebenden Fehler viermal so lange dauerten wie vorher bei Microsoft üblich [12, S. 207ff.]. Zu niedrige Projektvorgaben sind also kein „Kavaliersdelikt“, sie sind aufwandserhöhend. Wahrscheinlich hätten viele der auf der linken Hälfte in Abb. 3 gezeigten Projekte mit einem geringeren Aufwand und weniger „atmosphärischen“ Störungen abgeschlossen werden können, wenn sie vorab fundierter geschätzt worden wären.

Metriken zur quantitativen Prozess-Steuerung

Neben der Kalibrierung parametrischer Gleichungen ist das Haupteinsatzfeld von Softwaremetriken die quantitative Prozess-Steuerung von Projekten. Hiermit soll die aus dem Total Quality Management stammende Statistische Prozessregelung (Statistical Process Control SPC) auf das Projektmanagement übertragen werden.

Das Grundprinzip: Bei jedem Arbeitsprozess kann es Abweichungen vom erwarteten Sollwert geben. Als Ursachen dafür kommen die natürliche Streuung einerseits und außerordentliche Einflüsse andererseits in Frage. Die natürliche Streuung entsteht aus einer Vielzahl kleiner, immer wieder vorkommender Einzeleinflüsse. Sie ist relativ gut vorhersehbar. Dagegen treten außerordentliche Einflüsse nur unregelmäßig auf, haben größere Auswirkungen und machen einen Prozess instabil. Bei ihrem Auftreten muss das Management sofort aktiv werden. Bei natürlichen Streuungen sind demgegenüber störende Überreaktionen zu vermeiden.

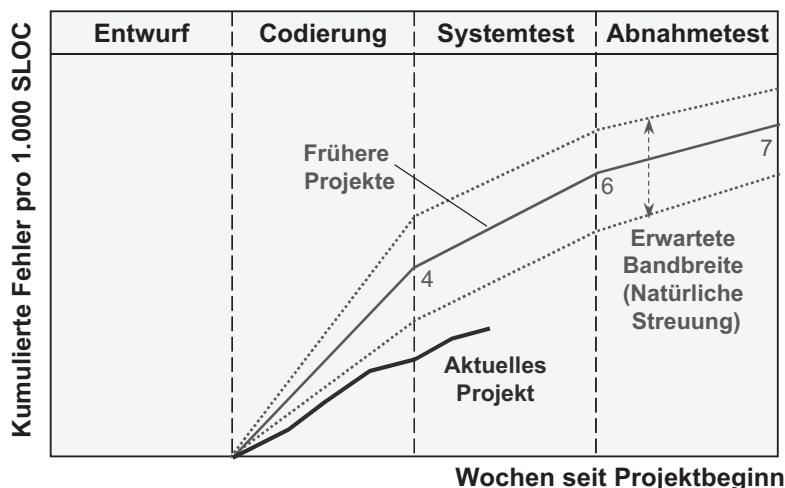


Abb. 4: Kontrolldiagramm zur Verfolgung der Softwarequalität

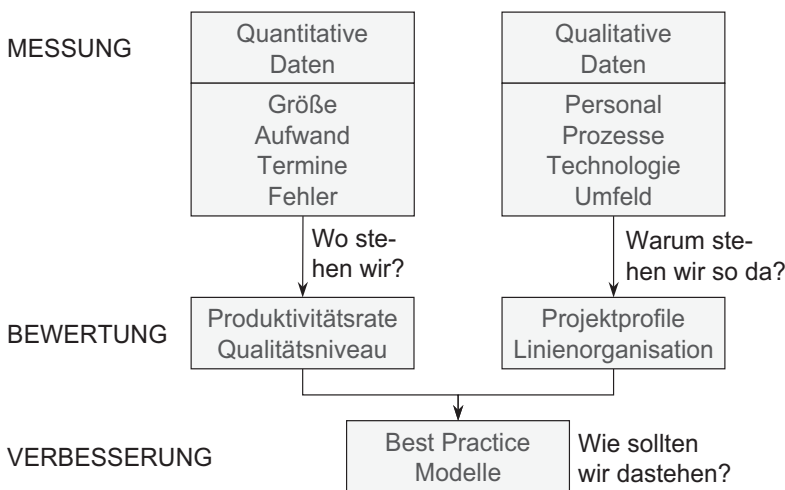


Abb. 5: Struktur von Benchmarkinganalysen [16, S. 56]

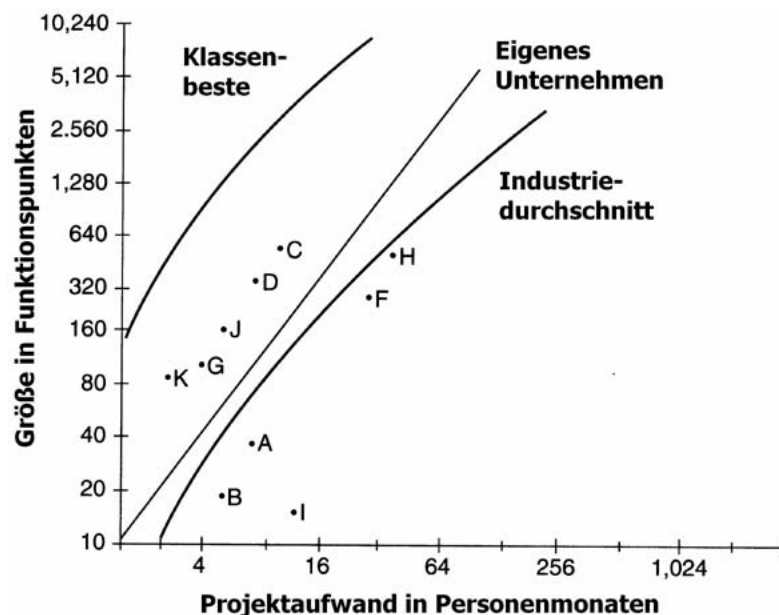


Abb. 6: Beispiel eines Benchmarkingdiagramms [16, S. 83]

Mit Kontrolldiagrammen außerordentliche Einflüsse erkennen

Durch Kontrolldiagramme können – ähnlich einer Fieberkurve im Krankenhaus – natürliche und außerordentliche Faktoren voneinander unterschieden werden. Im Projektmanagement werden die Grenzen dabei meist bei der ein- oder zweifachen Standardabweichung um den Mittelwert gezogen. Innerhalb dieser Grenzen sind dann nach den Regeln der Normalverteilung rund 68 % bzw. 95 % aller Werte zu erwarten.

Abb. 4 zeigt als Beispiel ein Kontrolldiagramm zur Prognose und Verfolgung von Fehlermeldungen. Die historische Erfahrungslinie und deren Bandbreite wurden vorab aus der Auswertung von etwa 25 früheren Projekten gewonnen. Solange bei einem neuen Projekt die Fehlerzahlen innerhalb dieser Bandbreiten liegen, befindet sich der Fehlerverlauf innerhalb der erwarteten natürlichen Streuung. Er ist unter „statistischer Kontrolle“.

Außerhalb der Bandbreiten liegen außergewöhnliche Einflüsse vor, deren Ursachen ermittelt werden müssen. Beispielsweise kann die Abweichung des in Abb. 4 dargestellten Projekts viele Ursachen haben. Dazu gehört auch die Möglichkeit, dass das Team keine Fehlermeldungen abgegeben hat oder dass zu wenig getestet wurde. Falls diese Ursachen ausgeschlossen werden können, handelt es sich wahrscheinlich um ein besonders zuverlässiges Softwaresystem. Teilweise können Abweichungen durch Cross-Checks mit anderen Kontrolldiagrammen analysiert werden, andernfalls sind spezielle Ursachenanalysen erforderlich.

Das „Manager’s Handbook“ des Software Engineering Laboratory SEL der US-Raumfahrtbehörde NASA enthält rund ein Dutzend Beispiele empfehlenswerter Kontrolldiagramme für Qualitäts- und Projektfortschrittsmetriken [13]. Kontrolldiagramme können aber auch für die klassischen Projektsteuerungsgrößen Termine, Aufwände und Kosten geführt werden. Beispielsweise wird bei Telcordia, einem Abkömmling der Bell Laboratories, auch für die Termineinhaltung ein Kontrolldiagramm geführt. Telcordia ist mit mehr als 3.500 SW-Entwicklern die bisher größte nach CMM Stufe 5 zertifizierte Softwareorganisation weltweit. Das Terminkontrolldiagramm wird aus Microsoft-Project-Daten abgeleitet und dient dazu, Abweichungen von der normalen Termintreue frühzeitig zu erkennen [14]. Lipke [15] berichtet über die Anwendung der statistischen Prozessregelung bei der Arbeitswertanalyse. Daraus lassen sich automatisierte Warnmeldungen erzeugen, wenn Kosten- und Leistungsabweichung aus dem bisher üblichen Rahmen ausbrechen.

Metriken und Benchmarking

Mit Kontrolldiagrammen können Projektprozesse zwar innerhalb ihrer natürlichen Streuung vorhergesagt und auftretende negative Abweichungen erkannt und beseitigt werden. Zur substanziellen Verbesserung des Produktivitäts- und Qualitätsniveaus leisten sie jedoch nur einen begrenzten Beitrag. Hierzu besser geeignet sind metrikgestützte Benchmarkinganalysen. Durch Benchmarkinganalysen kann ein Unternehmen erkennen, wo es im Vergleich zu seinen Wettbewerbern

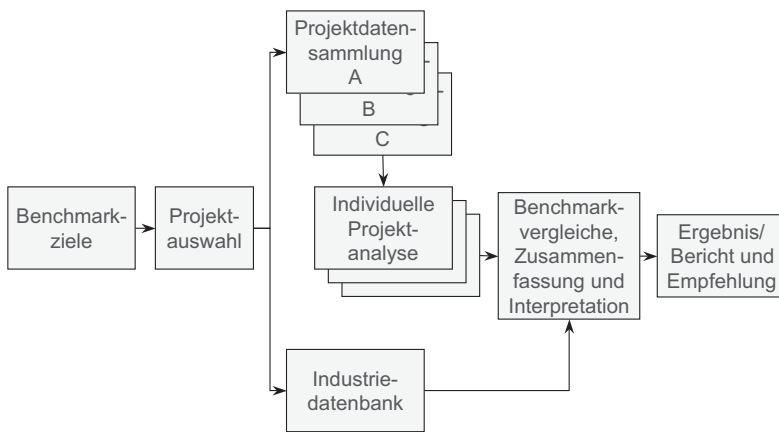


Abb. 7: Ablauf von Benchmarkingstudien [16, S. 56]

und anderen ähnlichen Unternehmen steht. Neben quantitativen Daten werden dabei auch qualitative Informationen über die Ursachen von Leistungsunterschieden ausgewertet (Abb. 5).

Beim Benchmarking vergleicht sich ein Unternehmen sowohl mit dem Durchschnitt als auch mit den Besten seiner Klasse (Abb. 6). Aus der Auswertung sieht das Unternehmen, wie stark und in welchen Bereichen es sich verbessern muss, wenn es zur Spitzenklasse gehören will. Um Unternehmen zu Verbesserungsprogrammen zu veranlassen, sind quantitative Benchmarks wie in Abb. 6 dem Top-Management leichter vermittelbar als abstrakte CMM-, PMMM- oder PM-Delta-Einstufungen. Wenn ein Unternehmen von fünf Funktionspunkten pro Personenmonat auf zehn kommen muss, um mit dem Industriedurchschnitt gleichzuziehen, versteht ein Top-Manager das ohne lange Erklärung.

Sammlung von Benchmarkingdaten

Projektbezogene Benchmarkingansätze gibt es im Softwarebereich bereits seit den 70er Jahren. Seither wurden für Vergleiche mit anderen Unternehmen mehrere firmenübergreifende Industrie-Datenbasen aufgebaut, insbesondere für Daten zum Projektaufwand und zur Projektgröße. Am bekanntesten sind

- die International Software Benchmarking Standards

Group (ISBSG, gesprochen „Ice-Bags“) [17],

- das Data & Analysis Center for Software (DACS) des amerikanischen Department of Defense [18] und
- Datenbasen von Anbietern kommerzieller Tools zur Software-Aufwandsschätzung, insb. KnowledgePlan und SLIM [19].

Die genannten Datenbasen wurden primär zur Kalibrierung von Aufwandsschätzsystemen eingerichtet. Zur fundierten Festlegung von Verbesserungsmaßnahmen sind ergänzende projektspezifische Datenerhebungen und qualitative Analysen erforderlich. Hierzu wird von ver-

schiedenen Consulting-Unternehmen die Durchführung kundenindividueller Benchmarkingstudien angeboten. Bekannt sind hier beispielsweise Software Productivity Research, die Standish Group, die Gartner Group und Howard Rubin Associates. Abb. 7 zeigt den typischen Ablauf einer solchen Studie.

Benchmarkingstudien bedürfen eines sorgfältigen Vorgehens und kommen normalerweise nicht ohne Vor-Ort-Besuche zur Datenerhebung und Prozessanalyse aus. Hauptproblem ist die Konsistenz der erhobenen Benchmarking-Daten: In verschiedenen Unternehmen werden Größen wie Lines of Code, Personenmonate oder Fehlerzahl häufig ganz unterschiedlich ermittelt. Beispielsweise erfassen Unternehmen den Projektaufwand oft nur lückenhaft und ungleichmäßig. Dies betrifft u.a. die Fragen,

- welche Projektaktivitäten im Aufwand enthalten sind und welche nicht,
- wie viele Stunden pro Monat normalerweise gearbeitet wird,
- ob die Stunden von Supportpersonal und oberem Management in die Projektkosten eingerechnet wurden oder nicht und
- wie unbezahlte Überstunden behandelt werden.

Zusätzlich sind nicht alle Arten von Softwareprojekten direkt miteinander vergleichbar. Die Produktivität in einem technisch geprägten Projekt für öffentliche

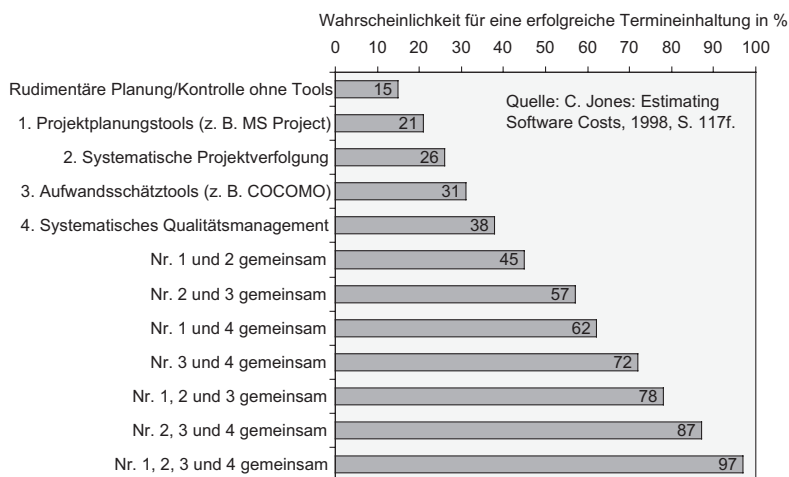


Abb. 8: SW-Projektplanungstools und Projekterfolg

Auftraggeber ist aufgrund der umfangreichen Dokumentationsanforderungen eine ganz andere als bei einem Web-Projekt für einen mittelständischen Kunden. Auch zwischen kleinen und großen Projekten gibt es erhebliche Produktivitätsunterschiede. Projekte müssen somit zumindest hinsichtlich der Art und der Größe des Projekts klassifiziert werden, um sinnvolle Vergleiche anstellen zu können.

Nutzen von Softwaremessungen

Wie ist nun der Einsatz von SW-Metriken insgesamt zu beurteilen? Empirische Erkenntnisse können aus der großen Zahl von Untersuchungen zu CMM-Verbesserungsprogrammen gewonnen werden [20]. Aus Platzgründen sei hier lediglich auf zwei Untersuchungen näher eingegangen: Eine Auswertung der Function-Point-Datenbank von Jones und eine Untersuchung auf Basis der COCOMO-Datenbank.

Zum Einsatz von Projektmanagementmethoden in der Softwareentwicklung wertete Jones die Daten von rund 600 Unternehmen aus, die Mehrzahl davon Großunternehmen mit mehr als 1.000 Software-Mitarbeitern [2, S. 93]. Abb. 8 zeigt zusammengefasst die Auswirkungen unterschiedlicher Management-Instrumente auf die Termineinhaltungswahrscheinlichkeit.

Aus dieser Grafik ergeben sich folgende Erkenntnisse:

- Durch metrikbasierte Aufwandsschätztools und eine systematische Projektverfolgung steigt die Termineinhaltungswahrscheinlichkeit stärker an als durch die Verwendung von Planungstools wie MS Project.
- Einen noch stärkeren positiven Einfluss hat allerdings ein systematisches Qualitätsmanagement. Darunter fallen intensive Design- und Code-Inspektionen, umfangreiche Tests und automatische Fehlerverfolgungssysteme.

Das Diagramm bezieht sich auf mittelgroße industrielle Projekte im Umfang von 80 bis 100 Personenjahren. Bei kleinen Projekten von einem Personenjahr ist die Erfolgswahrscheinlichkeit um 35 Prozentpunkte höher als in Abb. 8. Diese Projekte können damit auch ohne Metriken mit traditionellen PM-Methoden (im Diagramm: „Nr. 1 und 2 gemeinsam“) erfolgreich gemanagt werden.

Auch die geringe Schwankungsbreite auf CMM-Stufe 3 bei der Boeing-Untersuchung (Abb. 3) ist nicht allein auf die erfahrungsdatengestützten Aufwandsschätzungen zurückzuführen. Mindestens ebenso wichtig ist, dass auf dieser Stufe durch unternehmensweit definierte Prozesse eine hohe Konstanz der Abläufe realisiert wird.

Die bisher fundierteste, offen gelegte Untersuchung zu den Produktivitätsauswirkungen von CMM-Programmen wurde von Clark anhand einer Stichprobe aus 161 Projekten der COCOMO-II-Datenbasis vorgenommen [21]. Bei der Stichprobe nahmen die SW-Projektkosten auf den einzelnen CMM-Stufen um folgende Prozentsätze ab:

- Stufe 2 (n = 108 Projekte): 2 % (bei kleinen Projekten) bis 5 % (bei großen Projekten) niedriger als Stufe 1 (n = 6),
 - Stufe 3 (n = 23): 2 bis 5 % niedriger als Stufe 2,
 - Stufe 4 (n = 18): 4 bis 15 % niedriger als Stufe 3 und
 - Stufe 5 (n = 6): 7 bis 25 % niedriger als Stufe 4.
- Hieraus können folgende Schlussfolgerungen abgeleitet werden:

- Bei großen Projekten (größer als 500 KLOC) ist der Kostensenkungseffekt wesentlich höher als bei Kleinprojekten (10 KLOC). Dieses Ergebnis bestätigt nochmals die Aussage, dass SW-Metriken weniger ein Instrument für Kleinprojekte als vielmehr für mittlere und große Projekte sind.
- Auf den Stufen 2 und 3 sind die beobachteten Kostensenkungen geringer als auf den Stufen 4 und 5. Für die quantitative Prozess-Steuerung und quantitative Benchmarkings kann damit ein wichtiger Produktivitätssteigerungseffekt belegt werden.

Andere Studien und Fallstudien [20] zeigten durchweg noch wesentlich höhere Produktivitätseffekte, sind allerdings methodisch und empirisch weniger fundiert.

Hauptprobleme und Ausblick

Natürlich sind mit der Anwendung von SW-Metriken auch Probleme verbunden, die vermieden werden sollten. Das gravierendste Problem liegt darin, dass Mitarbeiter ihr Verhalten auf gewünschte Messergebnisse hin ausrichten. Wenn beispielsweise festgelegt wird, dass alle codierten Prozeduren ab einer bestimmten zyklomatischen Komplexität einem Review unterzogen werden müssen, könnten viele Programmierer versuchen, unter der festgelegten Grenze zu bleiben. Vielleicht wird dadurch die Code-Komplexität in Einzelfällen auch verringert. Häufig wird von den Betroffenen jedoch nur manipuliert, um das als lästig empfundene Review zu vermeiden. Um solche kontraproduktiven Effekte zu vermeiden, sind zwei Punkte zu empfehlen:

- Unter allen Umständen verhindern, dass Metriken zur Mitarbeiterbeurteilung verwendet werden und persönliche Konsequenzen für die Betroffenen nach sich ziehen, egal ob bewusst oder unbewusst.
- Innerhalb eines Metriksystems eine zweite Metrik verfolgen, die das durch die erste Metrik geförderte Verhalten nivelliert.

Aus dem letzten Punkt darf sich allerdings keine „Metrik-Inflation“ entwickeln. Vielmehr sollten Metriksysteme immer nur wenige, sorgfältig ausgewählte

| Bereich | Messgrößen | Quellen | Häufigkeit |
|---------------------|--|--|---|
| Projektschätzungen | <ul style="list-style-type: none"> Schätzung der erwarteten Gesamtzahl der Befehlszeilen und/oder Funktionspunkte (neu, modifiziert, wiederverwendet) Schätzung der erwarteten Gesamtzahl Module Schätzung von erwartetem Aufwand/Kosten Schätzung der erwarteten Meilensteintermine | Manager | Monatlich |
| Ressourcenverbrauch | <ul style="list-style-type: none"> Personenstunden (gesamt, nach Aktivitäten) Rechnernutzung Kosten (gesamt, nach Aktivität, disponiert) | Entwickler Automatisiert Controller | Wöchentlich Wöchentlich Monatlich |
| Projektfortschritt | <ul style="list-style-type: none"> Anzahl Anforderungen (spezifiziert, noch offen, Änderungen, Rückfragen) Module (entworfen, codiert, geprüft, getestet) Anzahl Befehlszeilen (kumuliert) Tests (durchgeführt, bestanden) | Manager Entwickler Automatisiert Entwickler | 14-tägig 14-tägig Wöchentlich 14-tägig |
| Fehler/Änderungen | <ul style="list-style-type: none"> Anzahl Fehler (nach Kategorie und Schwere) Anzahl Änderungen (nach Kategorie) Quellcode-Änderungen | Entwickler Entwickler Automatisiert | Fallweise Fallweise Wöchentlich |

Abb. 9: Metrik-System des Software Engineering Laboratory der NASA [22, S. 17 ff.]

Messgrößen enthalten, mit denen die Informationsbedürfnisse der Stakeholder am besten abgedeckt werden. Um einen Eindruck zu geben, wie dies aussehen könnte, zeigt Abb. 9 das seit Jahren mit Erfolg laufende Metrikprogramm des NASA-SEL. Die Messgrößenerfassung erfolgt dabei in wöchentlichen, 14-tägigen und monatlichen Intervallen. Nur wenige Größen werden dabei automatisiert erfasst.

Software-Prozess-Steuerung in Echtzeit

Das NASA-System entstand Anfang der 90er Jahre. Inzwischen ist die Entwicklung jedoch wesentlich weiter. Auf CMM-Stufe 4 oder 5 zertifizierte indische Unternehmen haben Prozess-Steuerungssysteme aufgebaut, die Metriken in Echtzeit verfolgen. Die Teams führen dabei alle Projektaktivitäten ohne Ausnahme mit automatisierten, webgestützten Workflowsystemen durch. Diese Workflowsysteme bereiten unmittelbar nach den Aktionen der Benutzer die daraus resultierenden Metriken auf. Manager, Kunden und benachbarte Teams können den Projektfortschritt dann ohne Zeitverzug „Realtime“ in ihrem Webbrowser verfolgen. Sobald eine Anforderung kreiert, geändert oder genehmigt ist, können die anderen Benutzer dies sehen. Sobald ein Review abgeschlossen ist, werden die gefundenen Fehler in die Fehlerverfolgung und Fehlerstatistik aufgenommen und die Kontrolldiagramme zur Prozess-Steuerung aktualisiert [23].

Warum sind diese fortschrittlichen Prozess-Steuerungssysteme gerade in Indien entstanden? Die meisten der großen indischen Softwarefirmen machen ihr Geschäft mit Auftragsentwicklungen für europäische und amerikanische Kunden. Hierbei müssen sie eine immense Vertrauensbarriere überwinden: Ihre Auftraggeber wissen nicht, ob sie sich bei termin- oder qualitätskritischen Aufträgen wirklich auf die 10.000 km entfernten Projektteams verlassen können. Für den Auftraggeber zugängliche, webgestützte Steuerungssysteme haben wesentlich geholfen, diese Akzeptanzbarriere zu überwinden und gleichzeitig das Projektmanagement zu verbessern. IT-Organisationen, die keine ähnlichen Systeme aufbauen, könnten

bald ernsthafte Wettbewerbsnachteile erleiden.

Fazit

Insgesamt sollte mit diesem Artikel der Nutzen von Metrikprogrammen herausgearbeitet und ein Überblick zu deren Anwendung und Auswertung gegeben werden. Es sollte nicht aufgezeigt werden, wie Metriksysteme im Detail funktionieren oder wie bei deren Einführung am besten vorzugehen ist. Hierzu sei dem Leser die weiterführende Literatur zu diesem Thema empfohlen, beispielsweise das Software Measurement Guidebook des Software Engineering Laboratory der NASA [24]. ■

Literatur

- [1] Schelle, Heinz: Das aktuelle Stichwort: Benchmarking. In: *Projektmanagement aktuell* 14, 2003, Heft 2, S. 29–32
- [2] Jones, T. C.: *Estimating Software Costs*. New York, McGraw-Hill, 1998
- [3] Bundschuh, M.: *Die Function Point Methode im praktischen Einsatz bei Softwareprojekten*. In: Schelle, H.; Reschke, H.; Schnopp, R.; Schub, A. (Hrsg.): *Loseblattsammlung „Projekte erfolgreich managen“*, 13. Aktualisierung, Kapitel 4.6.9, Köln 1994 ff. sowie www.dasma.de, 12. 7. 2003
- [4] ivs.cs.uni-magdeburg.de/sw-eng/us/giak/, 12. 7. 2003
- [5] Paulk, Mark C.; Curtis, Bill; Chrissis, Mary Beth; Weber, Charles V.: *Capability Maturity Model for Software, Version 1.1*. Software Engineering Institute, CMU/SEI-93-TR-24, DTIC Number ADA263403, February 1993, www.sei.cmu.edu/publications/documents/93.reports/93.tr.024.html, 15. 7. 2003
- [6] Steinmetz, A. H.: *Management von mittleren Softwareprojekten*. In: Ottmann, R.; Grau, N. (Hrsg.): *Projektmanagement – Strategien und Lösungen für die Zukunft*. Tagungsband zum 17. Deutschen Projektmanagement-Forum im Oktober 2000 in Frankfurt, S. 139–154
- [7] Cohn, M.: *A Measured Response: Useful Metrics to give managers the numbers to back up project hunches*. In: *The Software Testing and Quality Engineering Magazine*, September/October 2002, www.stqemagazine.com/featured.asp?id=24, 1. 7. 2003
- [8] Raeder, A.: *Projekt „Euro Basis Standard“*. In: Ottmann, R.; Grau, N. (Hrsg.): *Projektmanagement – Strategien und Lösungen für die Zukunft*. Tagungsband zum 17. Deutschen Projektmanagement-Forum im Oktober 2000 in Frankfurt, S. 59–70
- [9] Seibert, S.: *Aufwandsschätzung von Softwareprojekten*. Vieweg, Wiesbaden, erscheint Anfang 2004
- [10] Burghardt, M.: *Projektmanagement: Leitfaden für die Planung, Überwachung und Steuerung von Entwicklungsprojekten*. 3. Auflage, Publicidid MCD, München 1995
- [11] Vu, J. D.: *Process Improvement Journey (From level 1 to Level 5)*. Keynote-Presentation at SEPG 2001, www.processgroup.com/john-vu-keynote2001.pdf, 15. 7. 2003
- [12] McConnell, S.: *Rapid Development: Taming Wild Software Schedules*. Microsoft Press, Redmond, Wash. 1996
- [13] NASA Software Engineering Laboratory (Hrsg.): *Manager's Handbook for Software Development. Revision 1, Document SEL-84-101*, Greenbelt, NASA Goddard Space Flight Center, NASA, Maryland November 1990, sel.gsfc.nasa.gov/website/documents/online-doc/84-101.pdf, 12. 7. 2003
- [14] Pitterman, B.: *Telcordia Technologies: The Journey to High Maturity*. In: *IEEE Software*, July/August 2000, S. 89–96
- [15] Lipke, W.: *Statistical Process Control of Project Performance*.

- In: *CrossTalk*, March 2002, S. 15–18, www.stsc.hill.af.mil/crosstalk/2002/03/lipke.pdf, 12. 7. 2003
- [16] Jones, T. C.: *Software Assessments, Benchmarks and Best Practices*. Addison-Wesley 2000
- [17] www.isbsg.org.au, 12. 7. 2003
- [18] www.dacs.dtic.mil/databases/sled.html, 12. 7. 2003
- [19] www.spr.com und www.qsm.com, 15. 7. 2003
- [20] Vgl. hierzu das Verzeichnis entsprechender Studien auf der Website des Software Productivity Consortium, <http://www.software.org/Library/ROIofPI.asp>, 15. 7. 2003
- [21] Clark, B. K.: *Effects of Process Maturity on Development Effort*. Working Paper, Center for Software Engineering CSE, University of Southern California USC, Los Angeles 1999, sunset.usc.edu/~bkclark/Research, 12. 7. 2003
- [22] NASA Software Engineering Laboratory (Hrsg.): *Recommended Approach to Software Development*. Revision 3, Document SEL-81-305, Greenbelt, NASA Goddard Space Flight Center, Maryland June 1992, sel.gsfc.nasa.gov/website/documents/online-doc/81-305new.pdf, 12. 7. 2003
- [23] Yourdon, E.: *Managing Projects With Visible Processes*. In: *Computerworld*, June 12, 2000, www.yourdon.com/articles/0006cw.html, 15. 7. 2003. Ähnliche Funktionalitäten bietet auch das Projekt-Workflowtool Livelink, www.opentext.com, 15. 7. 2003
- [24] NASA Software Engineering Laboratory (Hrsg.): *Software Measurement Guidebook*, Revision 1, Document NASA-GB-001-94, Greenbelt, NASA Goddard Space Flight Center, NASA, Maryland June 1995, sel.gsfc.nasa.gov/website/documents/online-doc/94-102-N.pdf, 12. 7. 2003

Schlagwörter

Aufwandsschätzung, Metrikprogramme, Quantitative Prozess-Steuerung, Quantitatives Benchmarking, Software-Messung



Autor

Dr. Siegfried Seibert, Diplom-Wirtschaftsingenieur, ist Professor für Betriebswirtschaftslehre, insbesondere IT-Projektmanagement und Unternehmensführung, am Standort Dieburg der Fachhochschule Darmstadt. Schwerpunkt seiner externen Trainings- und Beratungstätigkeit ist

die Software-Kostenschätzung. Darüber hinaus verfügt er über eine langjährige, leitende Industriepraxis in der Automobil-Zulieferindustrie. Seit mehreren Jahre arbeitet er im Programmkomitee der GPM-Jahreskongresse mit. Im Vorstand der GPM ist Siegfried Seibert für das Ressort Publikationen zuständig.

Anschrift

Fachhochschule Darmstadt
 Fachbereich Wirtschaft (Campus Dieburg)
 Max-Planck-Str. 2
 D-64807 Dieburg
 E-Mail: s.seibert@fh-darmstadt.de