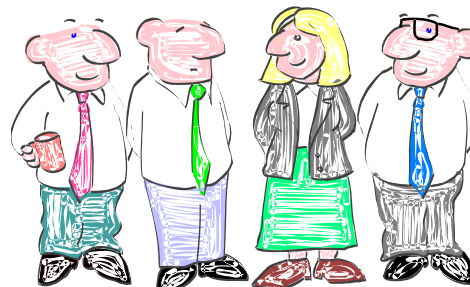


Quality Point München

Messen von Qualität

Probleme und pragmatische Ansätze



Probleme bei der Softwareentwicklung

- Anwendungsproblem
 - Auswirkungen von Softwarefehlern
- Managementproblem
 - Beherrschung der Softwarekomponenten in den Entwicklungsstufen und deren Übergänge
- Technologieproblem
 - sich ändernde Soft- und Hardwareplattformen, Entwicklungsparadigmen usw.

Lösungsansätze

- **genialer Wurf** basierend auf dem Orakel von Delphi oder der Empfehlung eines Gurus
- **schrittweise Verbesserung** basierend auf
 - Setzen von Zielen und suchen von Wegen zur Zielerreichung
 - Verständnis der Prozess- bzw. Produktcharakteristika (qualitativ)
 - Ermittlung/Messung wichtiger Parameter der Prozesse bzw. Produkte (quantitativ)
 - Änderungen und Messen der Auswirkungen

klassische Ansätze der Messung

- Bestimmung des Softwareumfangs (LOC o.Ä)
- Ermittlung des Leistungsverhaltens von Software-/Hardwaresystemen (Laufzeiten, Ausfallsicherheit usw.)
- Analyse und Test von Programmen (z.B. Fehleranzahl, ‚saubere‘ Programmierung)

Arten von Metriken (1)

- **Metrik** beschreibt einen allgemeinen funktionalen Zusammenhang
- **Maß** ist eine bewertete Metrik

- **Prozeßmetriken**
 - Reifegradmetriken (Organisationsniveau, Ressourcenniveau, Technologieniveau, Datenmanagement, Prozeßsteuerung, Dokumentationsstandards, Prozeßniveau usw.)
 - Managementmetriken (Meilenstein-, Review-, Risiko-Produktivitätsmetriken, Nutzerzufriedenheit, Effizienz, Versionsänderung usw.)
 - Life-Cycle-Metriken (Entwurfs-, Implementations-, Wartungsmetriken)

Arten von Metriken (2)

- **Produktmetriken**

- Umfangsmetriken (LOC, Anz. Dokumentationsseiten usw.)
- Architekturmetriken (Anzahl Klassen, Schichtenanzahl usw.)
- Strukturmetriken (Tiefe, Breite, Kopplung)
- Qualitätsmetriken (Performance, Wartbarkeits-, Funktionalitäts-, Zuverlässigkeits-, Portabilitätsmetriken usw. analog zu Qualitätsmerkmalen)
- Komplexitätsmetriken (algorithmische Komplexitätsmetriken, psychologische Komplexitätsmetriken wie Topologie, Steuerfluß, Entropie)

Arten von Metriken (3)

- **Ressourcenmetriken**

- Personalmetriken (Programmiererfahrung, Kommunikationsniveau, Produktivität, Teamstruktur usw.)
- Softwaremetriken (Leistungsmerkmale, Paradigmenabdeckung, Erneuerungsrate usw.)
- Hardwaremetriken (Leistungsmerkmale, Verfügbarkeit, Zuverlässigkeit usw.)

Situation in der Softwaremessung

- **Wissenschaftler**
 - haben hohes theoretisches Niveau und in der Regel sind die Resultate in der Praxis noch nicht brauchbar bzw. nicht in der Praxis überprüft
- **Praktiker**
 - wollen schnell brauchbare Resultate für aktuelle Probleme
- **Kunden/Anwender**
 - sind nicht imstande zu formulieren, was sie brauchen und können nur hoffen, daß sie das bekommen, was sie sich wünschen

Grad der Messgenauigkeit

- **Längenmessung** durch Körpermaße (mehrfaches Anlegen eines Ellenbogens, Abschreiten einer Länge)
- **Qualitätsmessung** mittels ‚Stubenfliege‘ (gute Luftqualität solange Stubenfliege lebt)
- **Anlegen eines ‚Guru‘-Maßes** (der Guru prognostiziert Produktivität, Qualität u.ä.)

spezielle Ziele der Softwaremessung

- Entwickler
 - Hilfe bei der Beurteilung der Zielerreichung in großen Projekten
- Manager
 - Mittel, um z.B. mittels Meilensteinen ein Gefühl für den Projektzustand und die Risiken bis zum Ziel zu erhalten
- Kunde/Anwender
 - Hilfe bei der Definition der Qualität und der Funktionalität
- Wartung
 - Hilfe bei der Entscheidung über Wiederverwendbarkeit, Reengineering usw.

Softwaremess- und -bewertungsstrategien

- **Evaluierungen**
 - z.B. gewichtete Fragebogenbewertung zu Merkmalen
- **Merkmalwertabschätzungen mit Formel**
 - z.B. Produktivität = Produktumfang / (Anforderungen * Zeit)
- **modellbezogene Softwaremessung**
 - z.B. Komplexität, Anzahl Klassen, Hierarchieebenen
- **direkte Messung**
 - LOC, Abarbeitungszeit

potentieller Nutzen durch Anwendung von Softwaremetriken

- beim Entwurf
 - Aufdeckung fehlender Prozeß- bzw. Produktbestandteile
 - Entdeckung von Komplexitätsschwerpunkten
 - Größenverständnis
- durch die Messung
 - quantifizierbare Veränderungen bzw. Verbesserungen
 - Vergleichbarkeit von Projekten, Software usw.
- durch die Auswertung und Bewertung
 - Möglichkeiten und Grenzen aufzeigen
 - zielorientiertes Vorgehen



Grundsätze eines Metrikprogramms

- schrittweise Einführung von Metriken für die Verbesserung der Softwarequalität auf der Basis eines verbesserten Entwicklungsprozesses
- systematische Anwendung der Methoden und Techniken der Softwaremetrie
- anfängliche Ausrichtung der Softwaremessung auf Schwerpunkte mit schnell realisierbaren Erfolgen (Goal-Question-Metric-Methode)
- Einbeziehung von Management und Mitarbeitern

- *Metriken sind ein Mittel zum Zweck !*
- *Die Nützlichkeit von Metriken muss periodisch überprüft werden !*

allgemeine Erfahrungen mit Softwaremetriken

- Die Einführung von Metriken verursacht am Anfang nur zusätzliche Kosten.
- fehlende Ansatzpunkte zum Messen bringen Verzögerungen
- unscharfe Zielsetzungen verursachen Irritationen
- hoher Aufwand infolge fehlender Toolunterstützung verursacht Akzeptanzprobleme
- Analyse und Interpretation der Ergebnisse bedarf einer fortlaufenden mühevollen Kleinarbeit
- Softwaremessung ist erst der zweite Schritt nach Anwendung eines Vorgehensmodells
- ***Softwaremessung bedarf eines ‚langen Atems‘ beim Management***

pragmatische Beispiele für Softwaremetriken (1)

- **Qualität der fachlichen Vorgaben**
- Messung nach erfolgter Realisierung und Einführung in die Produktion bezüglich Vollständigkeit, Richtigkeit, Verständlichkeit
- Basis: erstelltes und abgenommenes Fachkonzept bzw. die fachliche Beschreibung incl. die beigefügten Anlagen
- Beurteilung nach der Realisierung nach einer festgelegten Karenzzeit, Beurteilung durch den Realisierungsverantwortlichen im Projekt und den Verantwortlichen des Fachbereichs
- Maßstab zur Beurteilung: während der Realisierung aufgetretene Änderungswünsche und im produktiven Betrieb auftretende Fehler bzw. notwendige Nachbesserungen infolge fachlicher Mängel.
- Die Beurteilung obiger Kriterien erfolgt nach einer Skalierung.

pragmatische Beispiele für Softwaremetriken (2)

- **Anwenderzufriedenheit**
- Messung nach einer festgelegten Karenzzeit (damit überhaupt eine entsprechende angemessene Nutzung im produktiven Betrieb erfolgen konnte)
- Ermittlung durch Befragung der (ausgewählten) Anwender
- Die Beurteilung der Anwenderzufriedenheit erfolgt nach einer Skalierung .
- Beurteilt werden folgende Unterpunkte
 - Benutzerfreundlichkeit
 - Stabilität/Zuverlässigkeit
 - Schnelligkeit/Performance
 - Qualität der Einführung

pragmatische Beispiele für Softwaremetriken (3)

- **Zuverlässigkeit der Produktion**
- Messung (Zählung) der Ausfälle nach folgenden Kriterien (Ursachen)
 - Softwarefehler
 - Hardware- und Systemsoftwareprobleme (incl. Netz)
 - Bedienungsfehler
- Dabei gilt
 - evtl. zusätzlich eine Gruppierung der Ausfälle nach Wichtigkeit (Konsequenzen für wen, welches Ausmaß usw.) vornehmen
 - Eingruppierung der Ausfälle in o.g. Gruppen muß möglichst objektiv vorgenommen werden, d.h. weder die Verursacher noch die davon Betroffenen dürfen die Eingruppierung vornehmen.
 - Bei der statistischen Auswertung müssen vergleichbare Zeiträume gewählt werden (z.B. Monat).

abschließende Hinweise

- Ein System, das gemessen wird, paßt sich an (oft auch vordergründig, übertrieben und trivial). Dies gilt auch für die Mitarbeiter.
- Konfliktpotential mit Mitarbeitervertretungen (Betriebsrat u.ä.) beachten
- Vorsicht mit Vergleichen z.B. von ganzen Projekten, da die Rahmenbedingungen nicht immer vergleichbar sind
- Lieber wenige Daten konsequent für Metriken nutzen als einen weiteren Datenfriedhof schaffen.
- Metrikeinführung mit sonstigen Veränderungen verknüpfen
- Aussagekraft von maschinell erhobenen Komplexitätsmetriken ist begrenzt

Quellenhinweise

- Softwaremetrie (R.Dumke, O.-v.-Guericke Universität Magdeburg)
- diverse Literaturstellen in Büchern zum Software-Qualitätsmanagement
- Eigene Untersuchungen, Beratungs- und Einführungsergebnisse bei verschiedenen Kunden