LEHRSTUHL 14
SOFTWARE ENGINEERING

Willkommen zur Vorlesung
Methodische Grundlagen des Software-Engineering
im Sommersemester 2011

Prof. Dr. Jan Jürjens

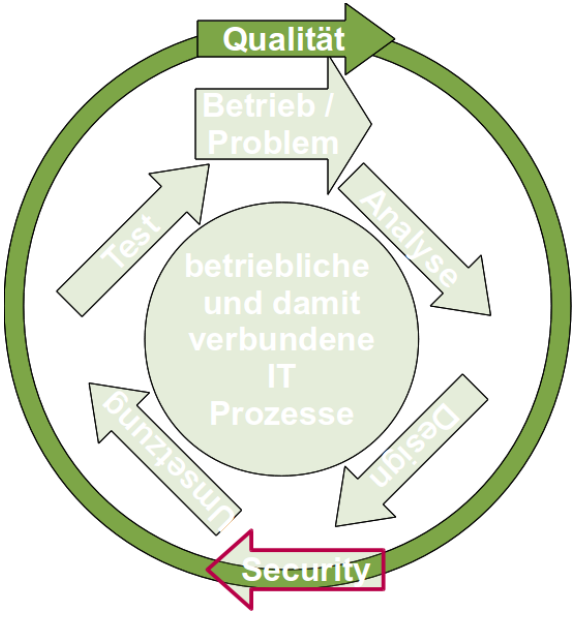
TU Dortmund, Fakultät Informatik, Lehrstuhl XIV

09. Qualitätsmanagement

[inkl. Beiträge von Professor Summerville, St Andrews University]

Einordnung
Grundlagen Qualität

- Was ist Qualität?
- Qualitätsmerkmale
- Qualitätsmanagement
- Qualitätssicherungsprozesse



Einordnung
Grundlagen Qualität

- Business Prozesse
- Qualitätsmanagement
  - Grundlagen
  - Prozessqualität
  - Softwarequalität
- Testen
- Sicherheit
- Sicheres Software Design

Themen

- **Software-Qualität**
- Software-Standards
- Bewertungen und Kontrollen
- Software-Messung und -Metriken

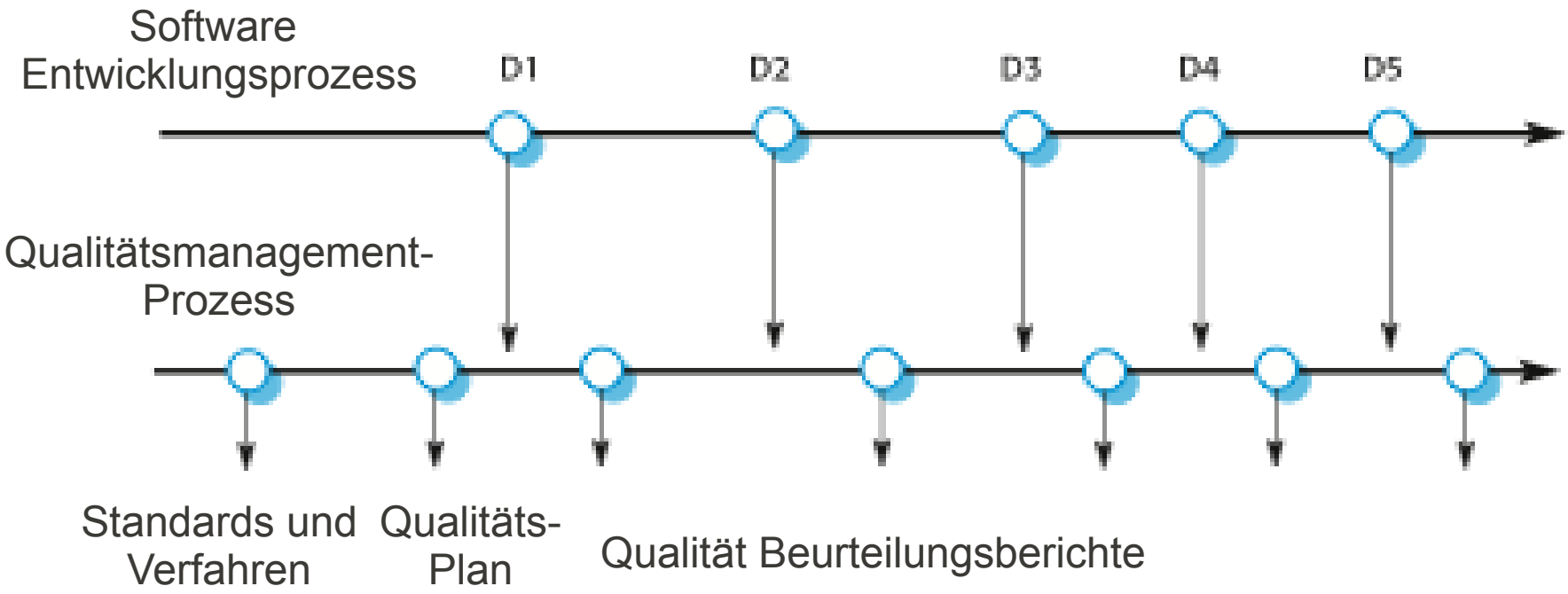
Software-Qualitätsmanagement

- Stellt sicher, dass die geforderte Qualität des Software-Produkts erreicht wird.
- Drei Hauptanliegen:
  - Auf organisatorischer Ebene beschäftigt sich das Qualitätsmanagement mit Schaffung eines Rahmens für die organisatorischen Prozesse und Standards, welche zu qualitativ hochwertiger Software führen.
  - Auf der Projektebene beinhaltet Qualitätsmanagement die Anwendung bestimmter Qualitäts-Prozesse und die Überprüfung, ob die geplanten Prozesse befolgt wurden.
  - Auf Projektebene beinhaltet das Qualitätsmanagement auch die Festlegung eines Qualitätsmanagement-Plans für ein Projekt. Der Qualitätsmanagement-Plan sollte eine Menge von Qualitätszielen für das Projekt definieren und festlegen, welche Prozesse und Standards zu verwenden sind.

Qualitätsmanagement-Aktivitäten

- Qualitätsmanagement bietet eine unabhängige Überprüfung im Software-Entwicklungsprozess.
- Der Qualitätsmanagement-Prozess prüft die Projektergebnisse, um zu gewährleisten, dass sie im Einklang mit den organisatorischen Standards und Zielen sind.
- Das Qualitäts-Team sollte vom Entwicklungsteam unabhängig sein, sodass es ein objektives Bild von der Software erhält. Das erlaubt ihm, ohne Beeinflussung durch die Software-Entwicklung über die Software-Qualität zu berichten.

Qualitätsmanagement und Software-Entwicklung



Qualitätsplanung

- Ein Qualitäts-Plan legt die gewünschten Produktqualitäten und deren Bewertung fest und definiert die wichtigsten Qualitätsmerkmale.
- Der Qualitäts-Plan sollte den Qualitätsbeurteilungsprozess definieren.
- Es sollte festgelegt werden, welche organisatorischen Standards angewendet werden und, wenn nötig, definieren, welche neuen Standards verwendet werden.

Qualitätspläne

- Qualitäts-Plan-Struktur
  - Produkt-Einführung;
  - Produkt-Pläne;
  - Prozessbeschreibungen;
  - Qualitätsziele;
  - Risiken und Risikomanagement.
- Qualitäts-Pläne sollten kurze und prägnante Dokumente sein
  - Wenn sie zu lang sind, wird sie niemand lesen.

Rahmen des Qualitätsmanagements

- Qualitätsmanagement ist besonders für große und komplexe Systeme wichtig. Die Qualitäts-Dokumentation ist eine Aufzeichnung der Fortschritte und unterstützt die Kontinuität der Entwicklung, als auch die Entwicklerteam-Änderungen.
- Für kleinere Systeme muss Qualitätsmanagement weniger Dokumentation enthalten und sollte sich auf die Einführung einer Qualitätsmanagements-Kultur konzentrieren.

Softwarequalität

- Qualität bedeutet vereinfacht, dass ein Produkt seine Spezifikation erfüllen sollte.
- Dies ist problematisch für Software-Systeme
  - Es gibt eine Spannung zwischen Qualitätsanforderungen des Kunden (Effizienz, Zuverlässigkeit, etc.) und Qualitätsanforderungen des Entwicklers (Wartbarkeit, Wiederverwendbarkeit, etc.);
  - Einige Qualitätsanforderungen sind schwierig, in eindeutiger Weise festzulegen;
  - Software-Spezifikationen sind meist unvollständig und oft widersprüchlich.
- Der Fokus könnte "Zwecktauglichkeit" statt Spezifikations-Konformität sein.

Software-Zwecktauglichkeit

- Haben Programmierung und Dokumentation Normen im Entwicklungsprozess eingehalten?
- Wurde die Software ordnungsgemäß geprüft?
- Kann die Software ausreichend zuverlässig in Betrieb genommen werden?
- Ist die Leistungsfähigkeit der Software akzeptabel für den normalen Gebrauch?
- Ist die Software nutzbar?
- Ist die Software gut strukturiert und verständlich?

Software-Qualitätseigenschaften

Sicherheit	Verständlichkeit	Portabilität
Schutz	Testbarkeit	Benutzbarkeit
Verlässlichkeit	Anpassungsfähigkeit	Wiederverwendbarkeit
Widerstandsfähigkeit	Modularität	Effizienz
Robustheit	Komplexität	Lernfähigkeit

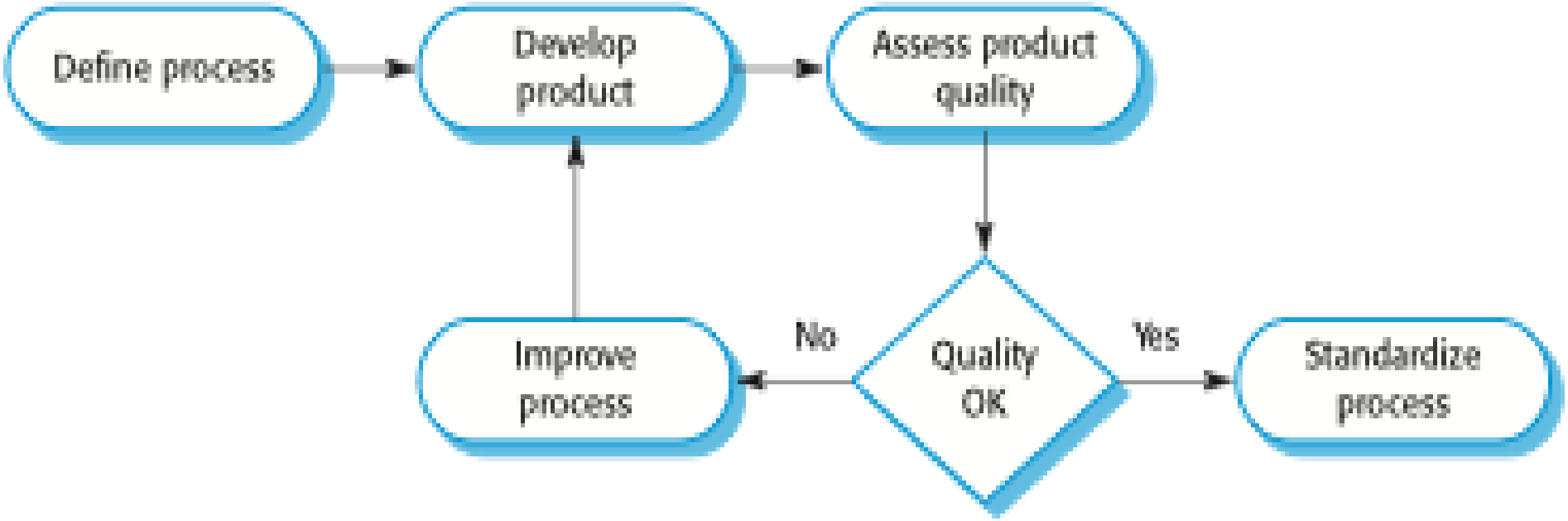
Qualitätskonflikte

- Es ist nicht möglich, ein System in allen Punkten der letzten Folie zu optimieren – zum Beispiel kann eine Verbesserung der Robustheit dazu führen, dass die Performance schlechter wird.
- Ein Qualitätsplan sollte deshalb für die zu entwickelnde Software die wichtigsten Qualitätsmerkmale herausstellen.
- Der Qualitätsplan sollte auch vorgeben, wie die Qualitätsbewertung der Software ablaufen soll. Das beinhaltet u.a. Wartbarkeit und Robustheit.

Prozess- und Produktqualität

- Die Qualität des entwickelten Produkts wird durch die Qualität des Produktionsprozesses beeinflusst.
- Dies ist wichtig in der Softwareentwicklung, da einige der Produktqualitätseigenschaften schwer zu beurteilen sind.
- Es gibt jedoch eine sehr komplexe und wenig verstandene Beziehung zwischen dem Softwareprozess und der Produktqualität.
  - Die Anwendung von individuellen Fähigkeiten und Erfahrung sind besonders wichtig in der Softwareentwicklung.
  - Externe Faktoren wie die Neuartigkeit der Anwendung oder beschleunigte Entwicklungszeitpläne können die Produktqualität beeinträchtigen.

Prozessbasierte Qualität



Frage

- Welche Qualitätsmerkmale können deiner Ansicht nach im Konflikt stehen?

Themen

- Software-Qualität
- **Software-Standards**
- Bewertungen und Kontrollen
- Software-Messung und -Metriken

Softwarestandards

- Standards definieren, welche Eigenschaften eines Produktes oder Prozesses benötigt werden. Sie spielen eine wichtige Rolle im Qualitätsmanagement.
- Standards können internationale oder nationale, Organisations- oder Projektstandards sein.
- Produktstandards definieren Charakteristiken, die alle Softwarekomponenten aufweisen müssen, z.B. ein gemeinsamer Programmierstil.
- Prozessstandards definieren, wie der Softwareprozess beendet werden sollte.

Die Bedeutung von Standards

- Erschließung der besten Verfahren – Vermeidung der Wiederholung vergangener Fehler.
- Standards sind ein Rahmen, um zu definieren, was Qualität in einem bestimmten Umfeld ausmacht, z.B. die Meinung eines bestimmten Unternehmens bzgl. Qualität.
- Sie bieten Kontinuität – neue Mitarbeiter verstehen das Unternehmen, indem sie die genutzten Standards verstehen.

Produkt- und Prozessstandards

Produktstandards	Prozessstandards
Form der Entwurfsprüfung	Leitung der Designprüfung
Struktur der Anforderungsdokumente	Vorlagen für neuen Systemcode
Formatierung der Methodenköpfe	Versionsrelease-Prozess
Java Programmierstil	Genehmigung des Projektplans
Formatierung des Projektplans	Kontrolle der Änderungen
Form einer Änderungsanfrage	Prozess der Testdurchläufe

Probleme mit Standards

- Sie könnten von Softwareentwicklern als nicht relevant oder up-to-date angesehen werden.
- Wenn sie nicht von den Softwaretools unterstützt werden, ist das Aufrechterhalten der Dokumentation (bzgl der Standards) oft mit langwierigem Formular-Ausfüllen verbunden.

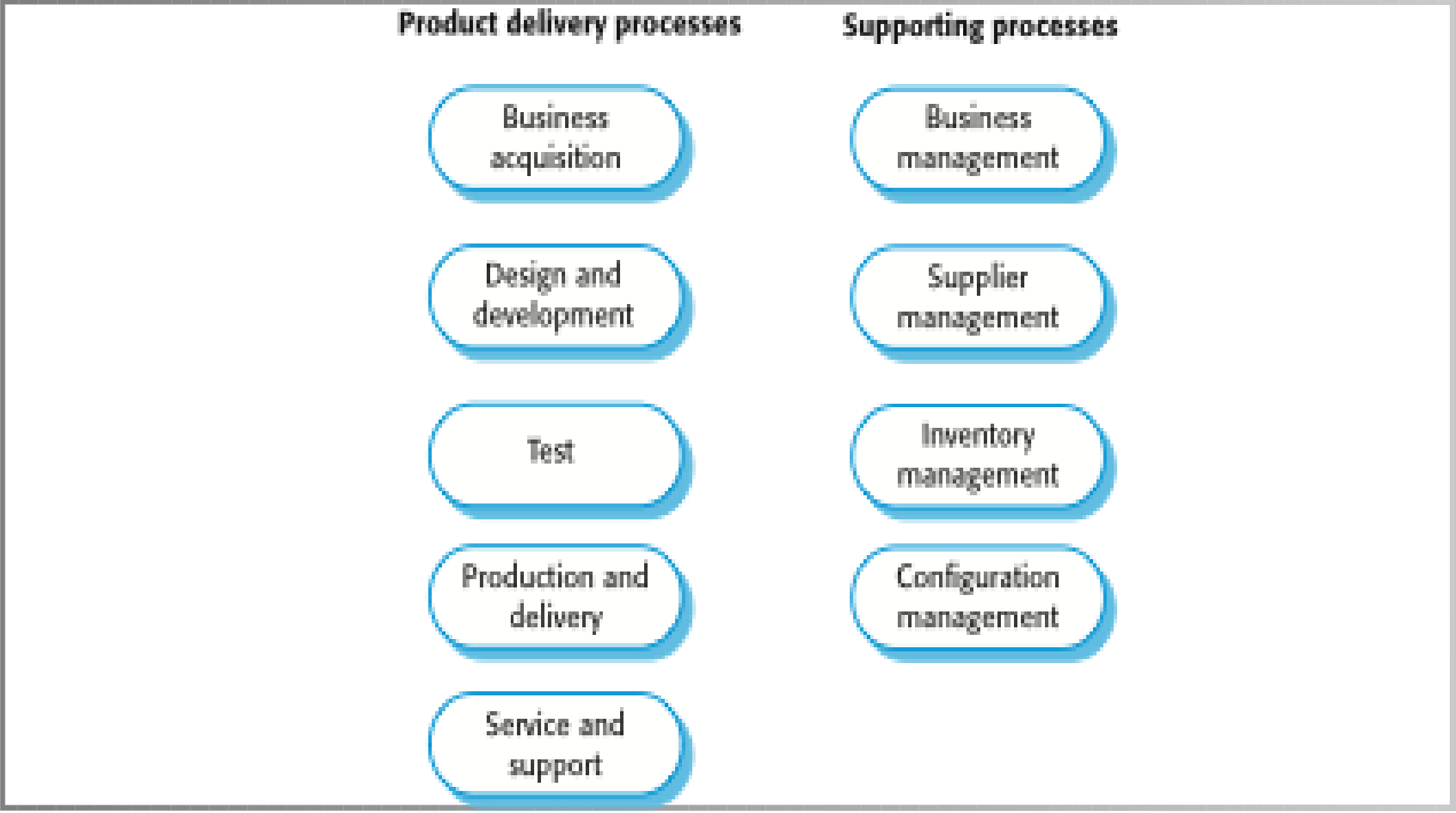
Entwicklung von Standards

- Involviert Fachleute in die Entwicklung. Die Entwickler sollten das Grundprinzip hinter dem Standard verstehen.
- Regelmäßige Überprüfung der Standards und deren Nutzen. Sie können schnell veralten, was ihrem Ansehen unter den Entwicklern schädigt.
- Detaillierte Standards sollten spezialisierte Programmunterstützung erhalten. Übermäßige Büroarbeit ist das größte Argument gegen Standards.
  - Web-basierte Formulare sind nicht gut genug!

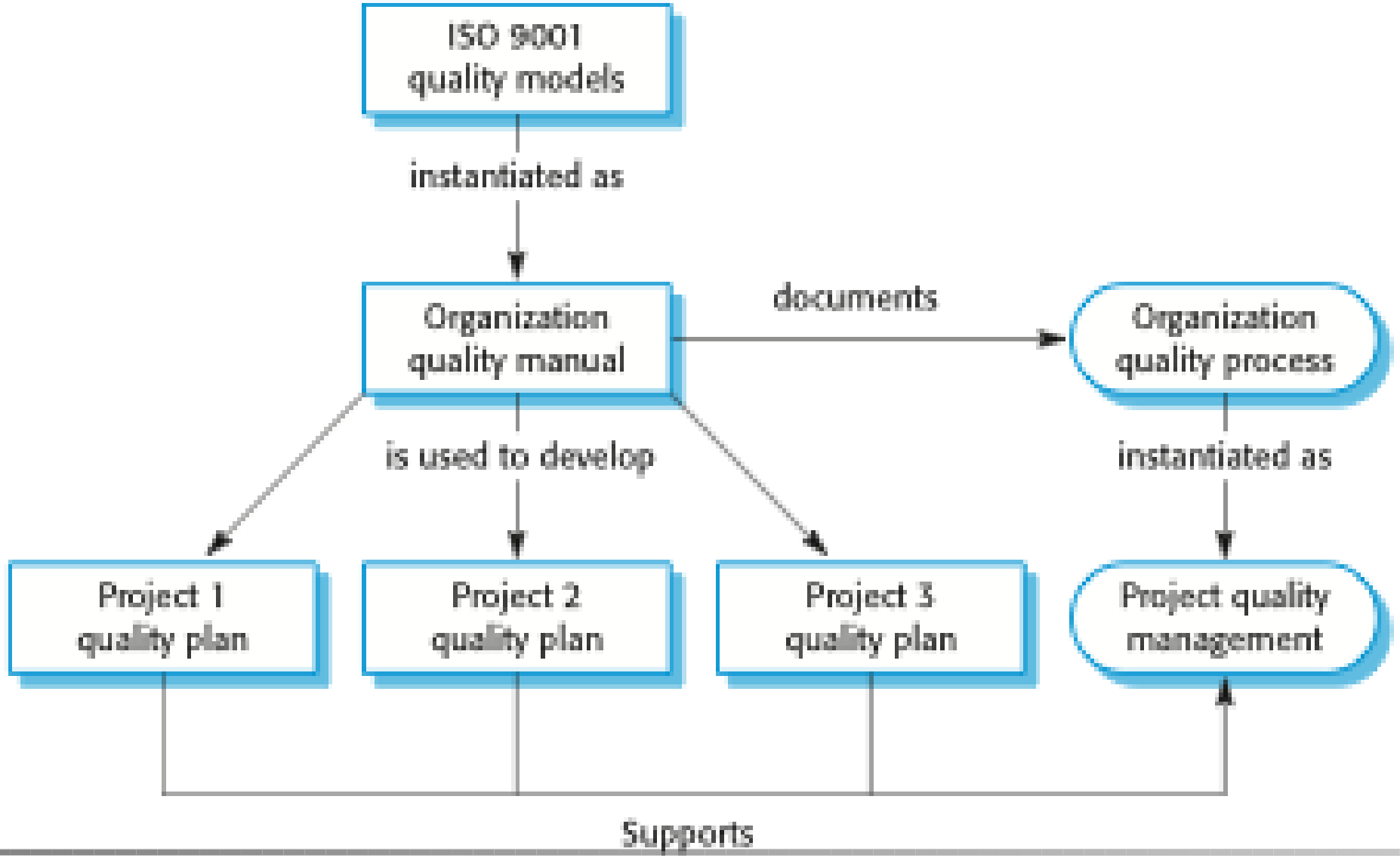
ISO 9001 Standardisierungswerk

- Eine internationale Sammlung von Standards, die als Basis des Qualitätsmanagements verwendet werden kann.
- ISO 9001, der allgemeinste dieser Standards, wird für Unternehmen herangezogen, die Softwareprodukte entwerfen, entwickeln und pflegen.
- Der ISO 9001 Standard ist ein Rahmenwerk, um Softwarestandards zu entwickeln.
  - Darin sind allgemeine Qualitätsmerkmale und die Qualität von Abläufen beschrieben. Außerdem werden organisatorische und prozedurale Normen dargelegt, die definiert und in einem Qualitätshandbuch niedergeschrieben werden sollten.

ISO 9001 Kernprozesse



ISO 9001 - Qualitätsmanagement



ISO 9001 Zertifikation

- Qualitätsstandards und Abläufe sollten in einem organisatorischen Qualitätshandbuch niedergeschrieben werden.
- Eine externe Stelle muss die Konformität des Qualitätshandbuchs mit der ISO 9000 bestätigen.
- Manche Kunden verlangen ISO 9000 Konformität von den Anbietern.

Die wichtigsten Punkte

- Software-Qualitätsmanagement soll sicherstellen, dass das Produkt wenige Fehler hat und die Standards bei Wartbarkeit, Zuverlässigkeit, Portabilität und weiteren Faktoren einhält.
- Software-Qualitätsmanagement beinhaltet, dass für Abläufe und Produkte Standards erstellt werden, sowie dass bei diesen Abläufen auch die Standards eingehalten werden.
- Software-Standards sind wichtig für die Qualitätssicherung, weil sie „den besten Weg“ anzeigen.
- Qualitätsmanagements-Abläufe müssen in einem organisatorischen Qualitätshandbuch, auf Basis des generischen Modells der ISO 9001, dokumentiert werden.

Frage

- Kann man mit der ISO 9001 die Qualität eines Softwareprodukts zertifizieren?

Themen

- Software-Qualität
- Software-Standards
- **Bewertungen und Kontrollen**
- Software-Messung und -Metriken

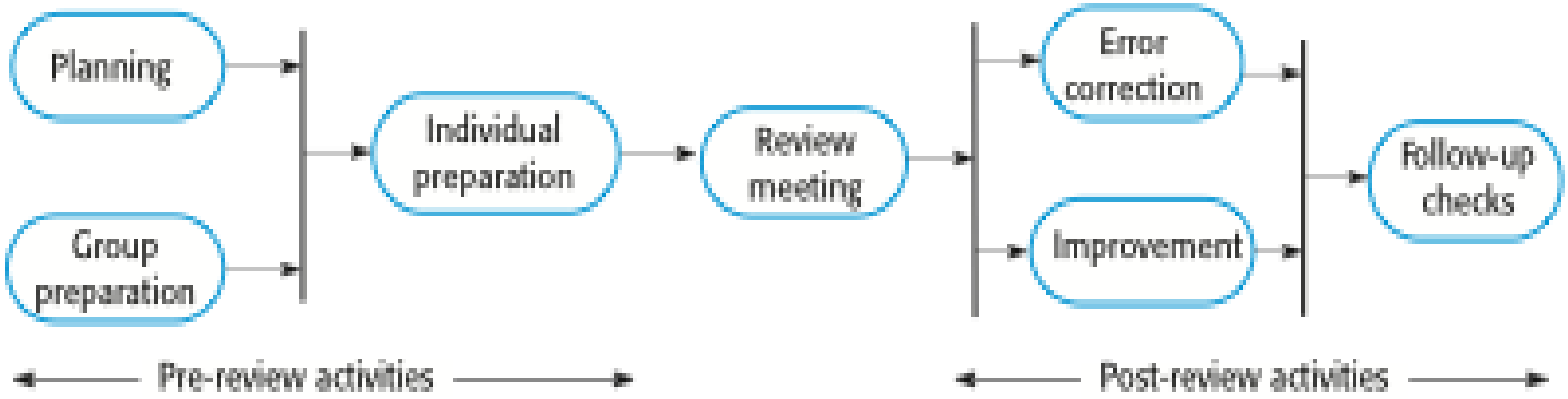
Bewertung und Kontrolle

- Eine Gruppe befasst sich mit Teilen oder dem ganzen System und seiner Dokumentation, um potentielle Probleme zu finden.
- Die Software oder Dokumente sollten als freigegeben markiert werden, damit das Fortfahren zum nächsten Schritt durch das Management abgesegnet ist.
- Es gibt drei verschiedene Prüfungen mit verschiedenen Zielen:
  - Inspektion zur Mängelbeseitigung (Produkt)
  - Prüfung der Fortschrittsbewertung (Produkt / Abläufe)
  - Prüfung der Qualität (Produkt / Standards)

Qualitätsprüfung

- Eine Gruppe prüft teilweise oder ganz das Softwaresystem und die dazugehörige Dokumentation.
- Code, Entwurf, Spezifikationen, Testentwürfe, Standards, etc. können alle überprüft werden.
- Die Software oder Dokumente sollten als freigegeben markiert werden, damit das Fortfahren zum nächsten Schritt durch das Management abgesegnet ist.

Der Ablauf einer Softwareprüfung



Bewertungen und agile Methoden

- Der Bewertungsprozess bei agiler Softwareentwicklung ist in der Regel informell.
  - Zum Beispiel in Scrum gibt es nach jeder Iteration, die die Software abschließt, ein Bewertungstreffen, in dem Qualität und Probleme besprochen werden.
- Beim „extreme programming (XP)“ wird durch Programmierung in Zweierteams sichergestellt, dass der Code ständig von einem Teammitglied überprüft und resümiert wird.
- XP basiert auf der Initiative der Einzelnen, den Code zu verbesseren und zu „refactorn“. Agile Ansätze bauen normalerweise nicht auf Standards auf, sodass Fragen, die durch Standards hervorgerufen werden, wegfallen.

Programm-Inspektionen

- Das sind Bewertungen von Fachkollegen, in denen die Quellen des Systems überprüft werden, um Anomalien und Fehler zu entdecken.
- Inspektionen benötigen keine Ausführung des Codes und können deshalb schon vor der Implementation verwendet werden.
- Sie können auf jede Art von Artefakten des System angewendet werden (z.B. Anforderungen, Design, Datenkonfiguration, Datentest, etc.).
- Es hat sich gezeigt, dass sie eine effektive Methode sind, um Programmfehler zu entdecken.

Prüfliste Inspektion

- Ein Prüfliste mit häufigen Fehlern sollte als Grundlage für die Inspektion dienen.
- Fehler-Prüflisten sind Programmiersprachen-abhängig und reflektieren die charakteristischen Fehler, die in dieser Sprache häufig auftreten.
- Generell gilt: Je weniger die Eingabe überprüft wird, desto länger ist die Prüfliste.
- Beispiele: Initialisierung, konstante Namensgebung, Schleifenabbruchsbedingungen, Arraygrenzen, etc..

Prüfliste einer Inspektion (1)

Fehlerklasse	Inspektionsaktionen
Datenfehler	• Wurden alle Variablen initialisiert bevor ihre Werte benutzt werden? • Wurden alle Konstanten benannt? • Sollte die obere Grenze des Arrays gleich der Größe des Arrays sein oder um einen kleiner? • Wenn 'char strings' benutzt werden, wurde ein Trennsymbol explizit angegeben? • Gibt es die Möglichkeit eines 'buffer overflows'?
Kontrollfehler	• Für jede bedingte Anweisung – ist die Bedingung korrekt? • Ist bei jeder Schleife gewiss, dass sie terminiert? • Ist bei geklammerten Bedingungen die Klammerung korrekt? • Bei Fallunterscheidungen – ist jeder Fall abgedeckt? • Wenn 'breaks' bei den Fallunterscheidungen benötigt werden – wurde sie alle gesetzt?
Ein-/Ausgabe Fehler	• Wurden alle Eingabevariablen benutzt? • Werden allen Ausgabevariablen ein Wert zugewiesen, bevor sie ausgegeben werden? • Können unerwartete Eingaben Verfälschungen herbeiführen?

Prüfliste einer Inspektion (2)

Fehlerklasse	Inspektionsaktionen
Interfacefehler	• Haben alle Funktions- und Methodenaufrufe die korrekte Anzahl an Parametern? • Sind die Parametertypen formal und implementiert die gleichen? • Ist die Reihenfolge der Parameter richtig? • Wenn Komponenten gemeinsamen Speicher benutzen, verwenden sie dann die gleiche Speicherstruktur?
Speicher-Verwaltungs-fehler	• Wenn eine verkettete Struktur modifiziert wird, wurden alle Verkettungen korrekt neu gesetzt? • Wenn dynamischer Speicher benutzt wird, wurde der Speicher korrekt reserviert? • Wird der Speicher wieder korrekt freigegeben, wenn er längere Zeit nicht benutzt wird?
Ausnahme-behandlungs-fehler	• Wurden alle möglichen Fehlerbedingungen in Betracht gezogen und behandelt?

Agile Methoden und Inspektionen

- Agile Prozesse verwenden selten formale Inspektionen oder paarweises Überprüfen.
- Vielmehr bauen sie auf Teammitglieder auf, die kooperativ gegenseitig ihren Code überprüfen und informelle Richtlinien wie „check before check-in“. Die Programmierer sollen also auch ihren eigenen Code überprüfen.
- Extreme-Programming-Verfechter argumentieren, dass paarweises Programmieren ein effektiver Ersatz für Inspektionen sind, da im Endeffekt eine ständige Überprüfung stattfindet.
- Zwei Menschen überprüfen jede Zeile Code, bevor sie akzeptiert wird.

Themen

- Software-Qualität
- Software-Standards
- Bewertungen und Kontrollen
- **Software-Messung und -Metriken**

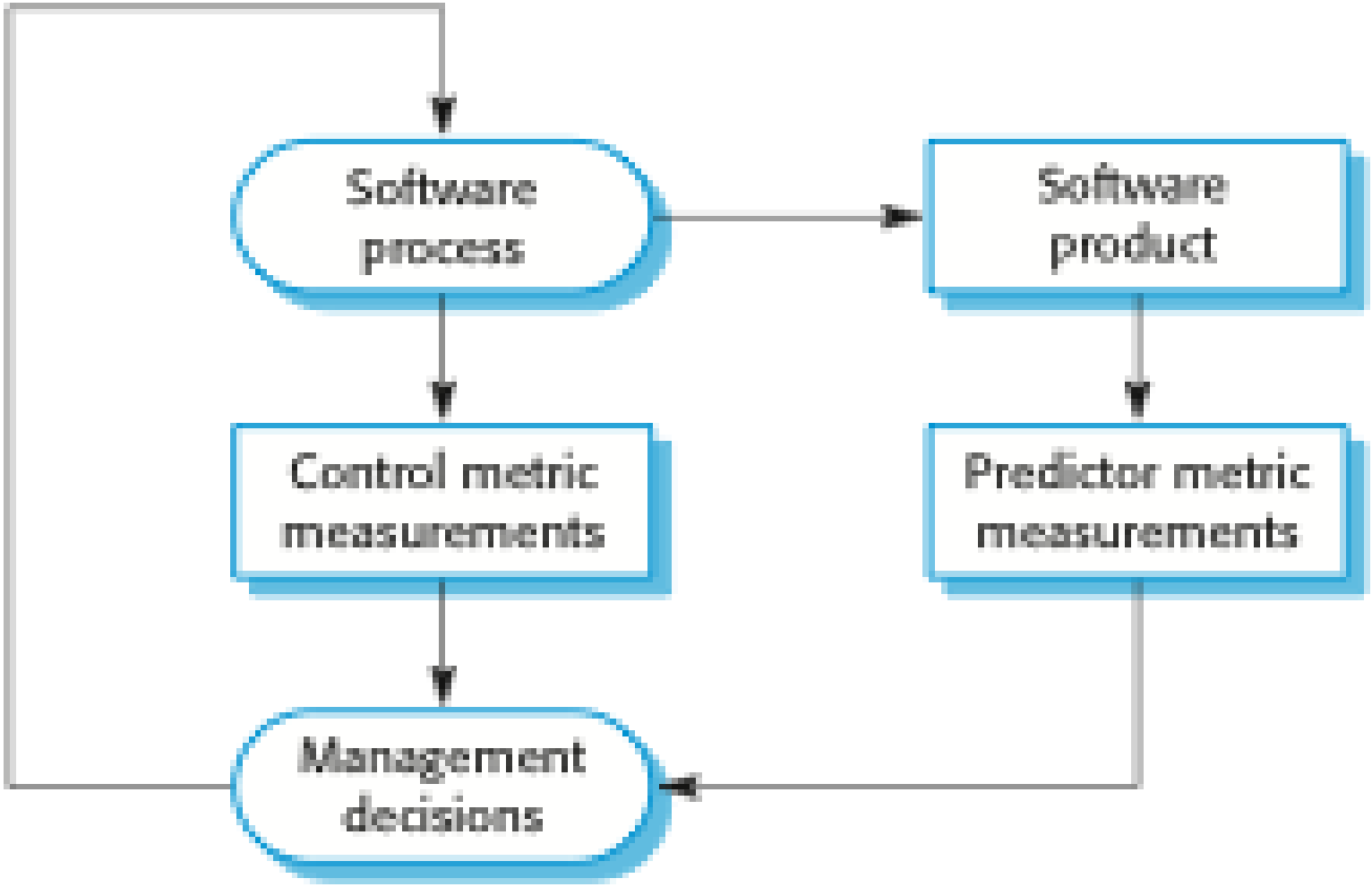
Softwaremessung und -metrik

- Softwaremessung meint, dass einem Attribut des Softwareproduktes oder -prozesses ein numerischer Wert zugewiesen wird.
- Das erlaubt das objektive Vergleichen zwischen Techniken und Prozessen.
- Obwohl einige Unternehmen Messungen eingeführt haben, arbeiten die meisten Unternehmen noch ohne systematische Messung von Software.
- Es gibt nur wenige etablierte Standards in diesem Bereich.

Softwaremetrik

- Zur Softwaremetrik gehört jede Art von Messung, welche ein Softwaresystem, -prozess oder zugehörige Dokumentation betrifft.
  - Z.B. die Anzahl der Codezeilen; der Fog-Index (vgl. Folie 48); die Anzahl der Personen-Tage, die für die Entwicklung einer Komponente benötigt wird.
- Das erlaubt, die Software und ihren Prozess zu quantifizieren.
- Kann verwendet werden, um Produktattribute vorherzusagen und den Softwareprozess zu steuern.
- Produktmetriken.

Prädiktor und Kontrollmessungen



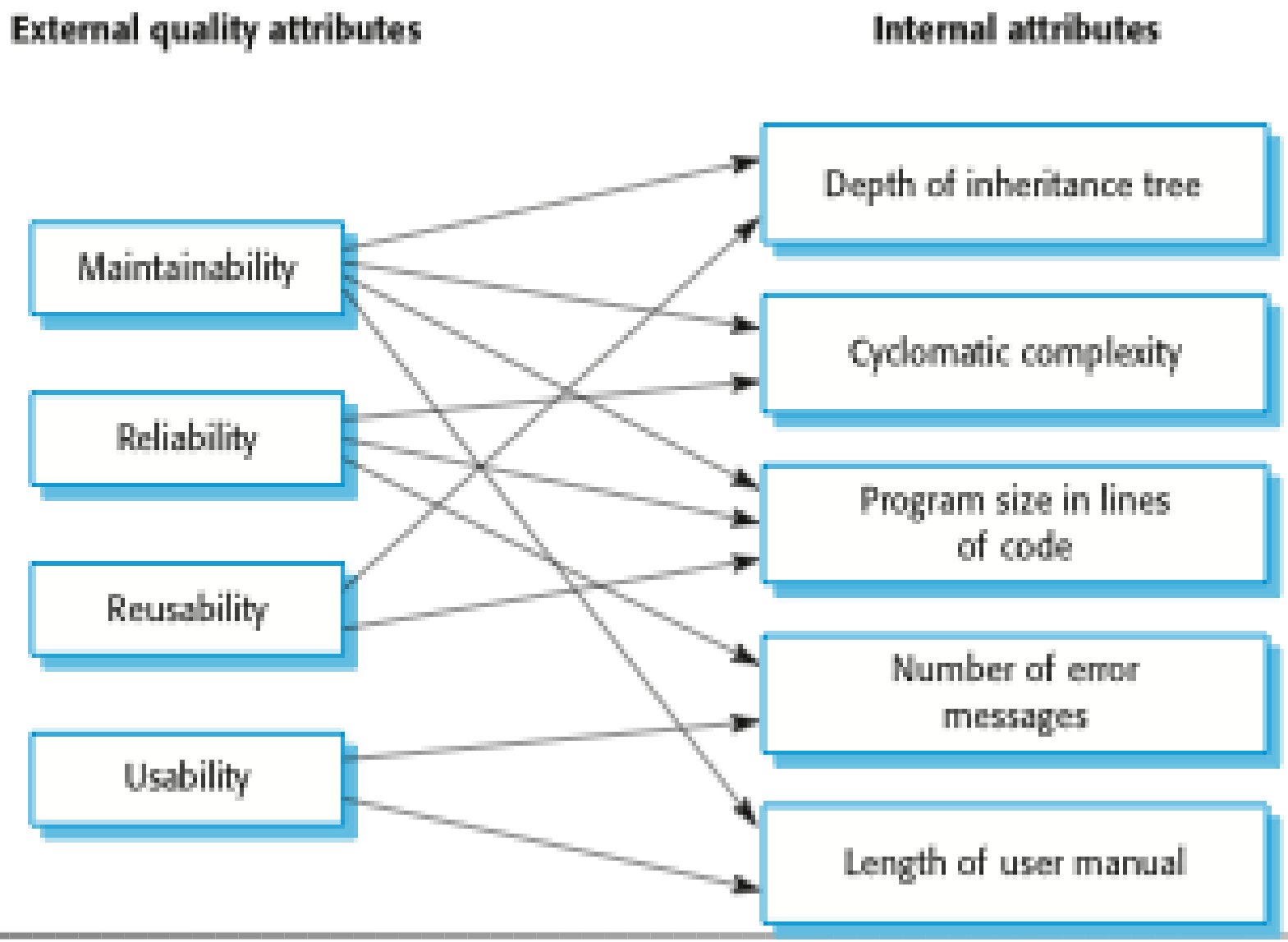
Verwendung von Messungen

- Um Systemqualitätsattributen einen Wert zuzuweisen.
  - Durch die Messung der Eigenschaften einer Systemkomponente, (wie ihre zyklomatische Komplexität, vgl. Folie 48) und deren Zusammenfassung, ist es möglich, Systemqualitätsattribute (wie Wartbarkeit) zu beurteilen.
- Um die Systemkomponenten zu identifizieren, deren Qualität unter dem Standard liegen.
  - Messungen können individuelle Komponenten, deren Charakteristiken von der Norm abweichen, identifizieren. Zum Beispiel kann man die Komponenten messen, um die mit der höchsten Komplexität zu finden. Diese beinhalten am ehesten Fehler, da sie am schwersten zu verstehen sind.

Messungsannahmen

- Eine Softwareeigenschaft kann gemessen werden.
- Es existiert eine Verbindung zwischen den Dingen, die man messen kann, und den Dingen, die man wissen will. Man kann nur die internen Attribute messen, ist aber öfter an den externen Softwareattributen interessiert.
- Diese Verbindung wurde formalisiert und validiert.
- Es kann schwierig sein, herauszufinden, was gemessen werden muss, um wünschenswerte externe Qualitätsattribute zu bekommen.

Beziehung zwischen externen und internen Attributen



Probleme mit Messung in der Industrie

- Es ist unmöglich, die Rendite zu berechnen, die eine Einführung eines organisatorischen Metrikprogramms mit sich bringt.
- Es gibt keine Standards für Softwaremetrik oder standardisierte Prozesse für Messung und Analyse.
- In viele Firmen sind die Softwareprozesse nicht standardisiert, wenig definiert und kontrolliert.
- Die meiste Forschung über Softwaremessung befasst sich mit Code-basierter Metrik und plangetriebenen Entwicklungsprozessen. Allerdings wird mittlerweile mehr und mehr Software durch Konfiguration von ERP Systemen oder Components-off-the-Shelf (COTS) entwickelt.
- Messungen einzuführen, verlangt zusätzlichen Aufwand bei den Arbeitsprozessen.

Produktmetriken

- Eine Qualitätsmetrik soll eine Vorhersage für die Produktqualität sein.
- Klassen der Produktmetrik:
  - Dynamische Metriken, die durch Messungen des Programms während der Laufzeit gesammelt werden.
    - Dynamische Metriken helfen, Effizienz und Zuverlässigkeit abzuschätzen.
  - Statische Metriken, die durch Messung einer Darstellung des Programms gesammelt werden.
    - Statische Metriken helfen, Komplexität, Verständlichkeit und Wartbarkeit abzuschätzen.

Dynamische und statische Metriken

- Dynamische Metriken sind eng mit den Software-Qualitätsattributen verbunden.
  - Es ist relativ einfach, die Antwortzeit (Performance) oder die Anzahl der Fehler (Funktionsfähigkeit) eines Systems zu messen.
- Statische Metriken haben eine indirekte Verbindung zu den Qualitätsattributen.
  - Man muss versuchen, eine Beziehung zwischen diesen Metriken und den Eigenschaften, wie Komplexität, Verständlichkeit und Wartbarkeit, herzustellen.

Frage

- Was ist ein grundlegendes Problem an Metriken?

Statische Metrik eines Softwareprodukts

Softwaremetrik	Beschreibung
Fan-in/Fan-out	Der Fan-in einer Funktion oder Methode X ist die Anzahl der Funktionen oder Methoden, die X aufrufen. Fan-out ist die Anzahl der Funktionen oder Methoden, die von X aufgerufen werden. Ein hoher Fan-in Wert zeigt, dass X eng mit dem Rest des System verbunden ist. Änderungen an X können weitreichende Folgewirkungen haben. Ein hoher Fan-out Wert legt nahe, dass die gesamte Komplexität von X hoch sein muss, da die Komplexität der Steuerungslogik von X die aufgerufenen Komponenten koordinieren muss.
Länge des Codes	Das ist eine Messung der Größe des Programms. Generell kann man sagen: Je größer der Code einer Komponente, desto komplexer und fehleranfälliger wird sie sein. Die Länge des Codes zu messen hat sich als zuverlässige Metrik zum Vorhersagen von Fehleranfälligkeit herausgestellt.

Statische Metrik eines Softwareprodukts

Softwaremetrik	Beschreibung
Zyklomatische Komplexität	Das ist ein Maß für die Kontrollkomplexität des Programms und direkt mit der Verständlichkeit verbunden.
Länge der Bezeichner	Das ist ein Maß der durchschnittlichen Länge von Bezeichnern (Namen von Variablen, Klassen, Methoden, etc.) in einem Programm. Je länger sie sind, desto wahrscheinlicher ist es, dass sie aussagekräftig sind. Folglich ist auch das Programm dadurch verständlicher.
Tiefe der Verschachtelung	Das ist ein Maß der Tiefe der Verschachtelung von if-Bedingungen in einem Programm. Tief verschachtelte if-Bedingungen sind schwer zu verstehen und potentiell fehleranfälliger.
Fog Index	Das ist ein Maß der durchschnittlichen Länge von Wörtern und Sätzen in einem Dokument. Je höher der Fog Index eines Dokuments ist, desto schwerer ist es zu verstehen.

Die objektorientierte Metriken-Suite „CK“

Softwaremetrik	Beschreibung
Gewichtete Methoden pro Klasse (WMC)	Das ist die Anzahl der Methoden in jeder Klasse, gewichtet durch die Komplexität. Eine einfache Methode kann die Komplexität von 1 haben, eine größere und komplexere einen sehr viel höheren Wert. Je größer dieser Wert ist, desto komplexer ist die Objektklasse. Komplexe Objekte sind meistens schwerer zu verstehen. Sie dürfen nicht logisch zusammenhängen, sonst kann man sie nicht effektiv als Superklassen in einem Vererbungsbaum wiederverwenden.
Tiefe des Vererbungs-baums (DIT)	Dieser Wert repräsentiert die Anzahl der Ebenen in einem Vererbungsbaum, in dem Unterklassen Attribute und Operationen (Methoden) von einer Superklasse erben. Je tiefer der Baum, desto komplexer ist das Design. Man muss viele Klassen verstehen, um das unterste Blatt des Vererbungsbaums nachvollziehen zu können.
Zahl der Kinder (NOC)	Das ist ein Maß für die unmittelbaren Unterklassen einer Klasse. Es misst die Breite einer Klassenhierarchie, während DIT die Tiefe repräsentiert. Ein hoher NOC-Wert kann eine hohe Wiederverwendung anzeigen. Es sollte mehr Anstrengung in die Validierung der Basisklassen fließen, da die Zahl der Unterklassen, die von ihnen abhängen, groß ist.

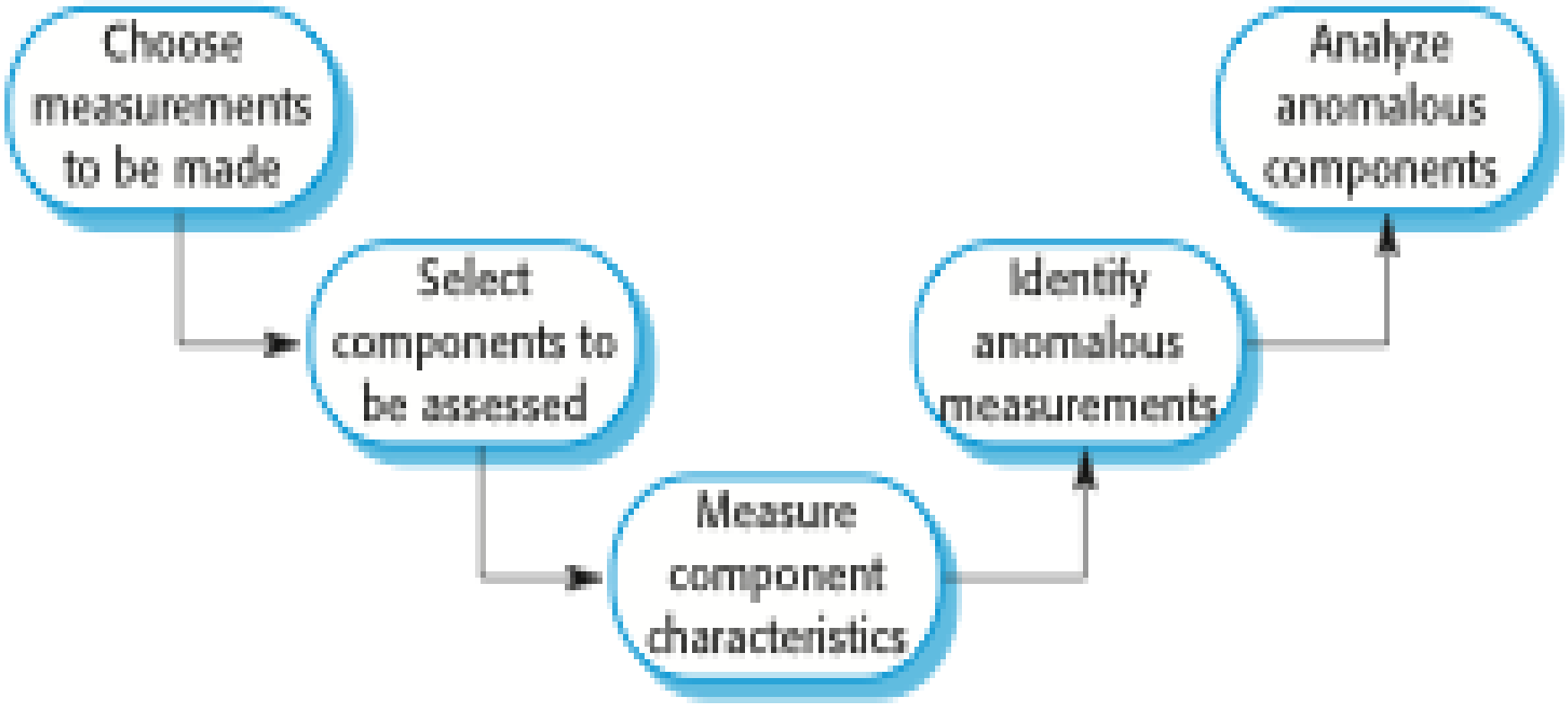
Die objektorientierte Metriken-Suite „CK“

Softwaremetrik	Beschreibung
Die Kopplung von Klassen (CBO)	Klassen sind gekoppelt, wenn Methoden einer Klasse Methoden oder Variablen einer anderen Klassen benutzen. CBO ist ein Maß dafür, wie viele Kopplungen existieren. Ein hoher CBO-Wert zeigt, dass die Klassen in einem hohen Maße voneinander abhängig sind. Damit ist es wahrscheinlicher, dass die Änderung einer Klasse viele Klassen im Programm betrifft.
Reaktion einer Klasse (RFC)	RFC ist ein Maß für die Anzahl der Methoden, die potentiell als Antwort auf eine Nachricht an die umgebene Klasse ausgeführt werden können. RFC ist mit der Komplexität verbunden. Je höher der RFC-Wert, desto komplexer und wahrscheinlich fehleranfälliger ist eine Klasse.
Mangel an Zusammenhalt in Methoden (LCOM)	LCOM wird durch die Betrachtung von Methodenpaaren berechnet. LCOM ist Differenz der Anzahl von Methodenpaaren mit geteilten Attributen und der Anzahl ohne diese. Der Wert dieser Metrik wurde ausführlich diskutiert und existiert in verschiedenen Variationen. Es ist nicht klar, ob dieses Maß irgendeine zusätzliche Information zu den anderen Metriken liefert.

Analyse der Softwarekomponenten

- Softwarekomponenten können separat mit Hilfe einer Reihe von anderen Metriken analysiert werden.
- Die Werte dieser Metriken können dann für verschiedene Komponenten und mit historisch gesammelten Daten, die von früheren Objekten stammen, verglichen werden.
- Anomale Messungen, die erheblich von der Norm abweichen, lassen vermuten, dass es Probleme mit der Qualität dieser Komponenten gibt.

Der Prozess einer Produktmessung



Überraschungen bei Messungen

- Die Reduzierung der Anzahl der Fehler in einem Programm führt zu einer höheren Aufrufrate der Hilfe.
  - Das Programm wird nun als zuverlässiger betrachtet und hat dadurch einen breiteren, vielfältigeren Markt. Der prozentuale Anteil an Benutzern, die die Hilfe aufrufen, mag sinken, der reale Anteil aber wird steigen.
  - Ein zuverlässiges System wird anders benutzt, als ein System, in dem die Benutzer um die Fehler herum arbeiten. Das führt zu einer höheren Aufrufrate der Hilfe.

Die wichtigsten Punkte

- Um Bewertungen des Softwareprozesses zu erstellen, wird eine Gruppe von Mitarbeitern benötigt, die überprüfen ob Qualitätsstandards eingehalten werden.
- Bei einer Programminspektion oder dem paarweisen Überprüfen testet eine kleine Gruppe systematisch den Code. Sie lesen den Code im Detail und halten nach möglichen Fehlern und Versäumnissen Ausschau.
- Softwaremessungen können verwendet werden, um Daten über die Software und ihren Prozess zu sammeln.
- Metriken der Produktqualität können hilfreich sein, um anomale Komponenten zu identifizieren, die womöglich Qualitätsprobleme haben.

Zusammenfassung

- Lesson Learned
  - Grundlagen Quälitatsmanagement
  - Qualitätseigenschaften
  - Grundlagen Standards
  - Softwareprüfung
  - Grundlagen Metriken