Software-Qualitätsmanagement Leitfragen

1. Was ist Qualitätsmanagement?
2. Nennen Sie Software-Qualitätsmerkmale und deren Untermerkmale!
3. Geben Sie Beispiele für konstruktive und analytische Maßnahmen des Qualitätsmanagements an!
4. Stellen Sie konstruktive und analytische Maßnahmen des Qualitätsmanagements gegenüber bezüglich Auswirkungen auf das Aufwand-Nutzen-Verhältnis eines Projekts!
5. Woher stammen die Qualitätsziele eines Projekts?
6. Geben Sie zwei Beispiele für konstruktive Maßnahmen im Komplex Software-Entwurfsmethoden oder –Vorgehen an, die zum Qualitätsziel Zeitverhalten beiträgt!
7. Wie werden beim V-Modell XT Prüfungen (Inspektionen, Tests) mit Entwicklungs- und Projektmanagement-Aktivitäten verknüpft?
8. Wie werden QM-Maßnahmen in SCRUM integriert?

Begriff Qualitätsmanagement

- Qualität
  - Ziele, Eigenschaften

  Definition Softwarequalität:

  Gesamtheit von Funktionen und Merkmalen eines Softwareprodukts, das die Fähigkeit besitzt, angegebene oder implizierte Bedürfnisse zu befriedigen. (ISO9126 / ISO25000)
- Management
  - Führen
  - Kontrollieren
  - Nicht: Verwalten

Software-Qualitätsmerkmale
ISO/IEC 25000: Produktmerkmale

Äußere Qualitätsmerkmale
ISO/IEC 25020, 25023

Funktionalität
Angemessenheit
Richtigkeit
Interoperabilität
Sicherheit
Ordnungsmäßigkeit

Benutzbarkeit
Verständlichkeit
Erlernbarkeit
Bedienbarkeit
Attraktivität
Konformität (Ben.)

Zuverlässigkeit
Reife
Fehlertoleranz
Wiederherstellbarkeit
Konformität (Zuverl)

Effizienz
Zeitverhalten
Verbrauchsverhalten
Konformität (Effiz.)

Innere Qualitätsmerkmale
ISO/IEC 25020, 25022

Wartbarkeit
Analysierbarkeit
Änderbarkeit
Stabilität
Testbarkeit
Konformität (Wartbark.)

Übertragbarkeit
Anpassbarkeit
Installierbarkeit
Koexistenz
Austauschbarkeit
Konformität (Übertragb.)

Funktionalität

Inwieweit besitzt die Software die geforderten Funktionen? – Vorhandensein von Funktionen mit festgelegten Eigenschaften. Diese Funktionen erfüllen die definierten Anforderungen.

- **Angemessenheit**: Eignung von Funktionen für spezifizierte Aufgaben, zum Beispiel aufgabenorientierte Zusammensetzung von Funktionen aus Teilfunktionen.
- **Richtigkeit**: Liefern der richtigen oder vereinbarten Ergebnisse oder Wirkungen, zum Beispiel die benötigte Genauigkeit von berechneten Werten.
- **Interoperabilität**: Fähigkeit, mit vorgegebenen Systemen zusammenzuwirken.
- **Sicherheit**: Fähigkeit, unberechtigten Zugriff, sowohl versehentlich als auch vorsätzlich, auf Programme und Daten zu verhindern.
- **Ordnungsmäßigkeit**: Merkmale von Software, die bewirken, dass die Software anwendungsspezifische Normen oder Vereinbarungen oder gesetzliche Bestimmungen und ähnliche Vorschriften erfüllt.

Zuverlässigkeit

Kann die Software ein bestimmtes Leistungsniveau unter bestimmten Bedingungen über einen bestimmten Zeitraum aufrechterhalten? – Fähigkeit der Software, ihr Leistungsniveau unter festgelegten Bedingungen über einen festgelegten Zeitraum zu bewahren.

- **Reife**: Geringe Versagenshäufigkeit durch Fehlerzustände.
- **Fehlertoleranz**: Fähigkeit, ein spezifiziertes Leistungsniveau bei Software-Fehlern oder Nicht-Einhaltung ihrer spezifizierten Schnittstelle zu bewahren.
- **Wiederherstellbarkeit**: Fähigkeit, bei einem Versagen das Leistungsniveau wiederherzustellen und die direkt betroffenen Daten wiederzugewinnen. Zu berücksichtigen sind die dafür benötigte Zeit und der benötigte Aufwand.
- **Konformität**: Grad, in dem die Software Normen oder Vereinbarungen zur Zuverlässigkeit erfüllt.

Benutzbarkeit

Welchen Aufwand fordert der Einsatz der Software von den Benutzern und wie wird er von diesen beurteilt? – Aufwand, der zur Benutzung erforderlich ist, und individuelle Beurteilung der Benutzung durch eine festgelegte oder vorausgesetzte Benutzergruppe.

- **Verständlichkeit**: Aufwand für den Benutzer, das Konzept und die Anwendung zu verstehen.
- **Erlernbarkeit**: Aufwand für den Benutzer, die Anwendung zu erlernen (zum Beispiel Bedienung, Ein-, Ausgabe).
- **Bedienbarkeit**: Aufwand für den Benutzer, die Anwendung zu bedienen.
- **Attraktivität**: Anziehungskraft der Anwendung gegenüber dem Benutzer.
- **Konformität**: Grad, in dem die Software Normen oder Vereinbarungen zur Benutzbarkeit erfüllt.

Effizienz

Wie liegt das Verhältnis zwischen Leistungsniveau der Software und eingesetzten Betriebsmitteln? – Verhältnis zwischen dem Leistungsniveau der Software und dem Umfang der eingesetzten Betriebsmittel unter festgelegten Bedingungen.

- **Zeitverhalten**: Antwort- und Verarbeitungszeiten sowie Durchsatz bei der Funktionsausführung.
- **Verbrauchsverhalten**: Anzahl und Dauer der benötigten Betriebsmittel bei der Erfüllung der Funktionen. Ressourcenverbrauch, wie CPU-Zeit, Festplattenzugriffe usw.
- **Konformität**: Grad, in dem die Software Normen oder Vereinbarungen zur Effizienz erfüllt.

Wartbarkeit / Änderbarkeit

Welchen Aufwand erfordert die Durchführung vorgegebener Änderungen an der Software? – Aufwand, der zur Durchführung vorgegebener Änderungen notwendig ist. Änderungen können Korrekturen, Verbesserungen oder Anpassungen an Änderungen der Umgebung, der Anforderungen oder der funktionalen Spezifikationen einschließen.

- **Analysierbarkeit**: Aufwand, um Mängel oder Ursachen von Versagen zu diagnostizieren oder um änderungsbedürftige Teile zu bestimmen.
- **Modifizierbarkeit**: Aufwand zur Ausführung von Verbesserungen, zur Fehlerbeseitigung oder Anpassung an Umgebungsänderungen.
- **Stabilität**: Wahrscheinlichkeit des Auftretens unerwarteter Wirkungen von Änderungen.
- **Testbarkeit**: Aufwand, der zur Prüfung der geänderten Software notwendig ist.
- **Konformität**: Grad, in dem die Software Normen oder Vereinbarungen zur Änderbarkeit erfüllt.

Übertragbarkeit (Portabilität)

Wie leicht lässt sich die Software in eine andere Umgebung übertragen? – Eignung der Software, von der Umgebung in eine andere übertragen werden zu können. Umgebung kann organisatorische Umgebung, Hardware- oder Software-Umgebung sein.

- **Anpassbarkeit**: Fähigkeit der Software, diese an verschiedene Umgebungen anzupassen.
- **Installierbarkeit**: Aufwand, der zum Installieren der Software in einer festgelegten Umgebung notwendig ist.
- **Koexistenz**: Fähigkeit der Software neben einer anderen mit ähnlichen oder gleichen Funktionen zu arbeiten.
- **Austauschbarkeit**: Möglichkeit, diese Software anstelle einer spezifizierten anderen in der Umgebung jener Software zu verwenden, sowie der dafür notwendige Aufwand.
- **Konformität**: Grad, in dem die Software Normen oder Vereinbarungen zur Übertragbarkeit erfüllt.

Merkmale der Prozeßqualität

- Prozeß-Reifegrad
  - Reifegradmodell
- Terminplanung
  - Termineinhaltung
  - Güte der Abschätzung
- Budgetplanung
  - Budgeteinhaltung
  - Güte der Abschätzung
- Produktivität
  - Ergebnis ./. Aufwand
- Aufgaben-Koordination
  - Anteil Leerlaufzeiten
  - Grad Paralleltätigkeiten
- Organisation und Kommunikation
  - Regelungsgrad
  - Informationsweg-Länge
  - Spezialisierungsgrad
  - Kommunikationsaufwand
- Erfahrungsmanagement
  - Einarbeitungsaufwand
  - Fehlervermeidung / Anteil Rework
  - Optimierungsgeschwindigkeit
- Kundenzufriedenheit
- Mitarbeiterzufriedenheit

Konstruktive Maßnahmen SWQM

- Technisch-konstruktive Maßnahmen
  - Methoden- und Werkzeuganwendung (Softwareengineering)
  - Programmiersprachen
- Organisatorische Maßnahmen
  - Projektmanagement (z. B. Pläne und Koordinierung)
  - Konfigurationsmanagement (z. B. Schutz vor Veränderungen und Inkonsistenzen)
  - Vorgehensmodell
- Psychologisch orientierte Maßnahmen
  - Schulungen (z. B. zu Qualitätsmaßnahmen und Zielen)
  - Motivationsfördernde Maßnahmen (z. B. Qualitätszirkel)
  - Kommunikationsverbessernde Maßnahmen

Unternehmenskultur, Kommunikation und Motivation

- psychologisch orientierte Maßnahmen mit großem Erfolgspotential
  - SW-Entwicklung basiert auf intellektuellen Fähigkeiten
  - Kommunikation ist wesentlicher Tätigkeitsinhalt
  - Ursache-Wirkungsanalysen: größte Ressource liegt in Freisetzung persönlicher Motivation
- technisch orientierte Maßnahmen in Wirkung begrenzt durch Engagement und Qualifikation

Unternehmenskultur

- Arbeit in Teams: Abteilung, Bereich, Gruppe
- Grundwerte der Gruppe prägen persönliche Einstellung

  z.B. zu Qualität, Kunden, Gewinn, Mitarbeitern
- Unternehmenskultur beeinflußt Handeln
- Mängel wirken auf Arbeitsergebnisse - Qualität und Produktivität

  z.B. Dokumentation

Einfluß der Unternehmenskultur

Grundwerte beeinflussen das Handeln der Mitarbeiter:

- Kommunikation
  (Vorgesetzter-Mitarbeiter, Mitarbeiter untereinander)
- Vorgehen bei Problemlösung (partizipativ, gesteuert)
- Arbeitsmentalität (Gleichgültigkeit, Identifikation)
- textliche Qualität von Dokumenten (oberfächlich, fundiert)
- Ordnung (behindernd, funktional)
- Gestaltung von Gebäuden und Arbeitsplätzen
- Umfang und Qualität der Bildungsmöglichkeiten

Beziehung Vorgesetzter - Mitarbeiter

Besondere Bedeutung für Grundwerte und Motivation

> Führen: zielorientierte Beeinflussung des menschlichen Verhaltens

Möglichkeiten:

– Wecken von Interesse

– Bereitstellung von Identifikationspotentialen

... Erzeugen von Furcht

Motivation und Akzeptanz besonders wichtig

Betrachtung in Hinsicht auf TQM

- Vorgesetzter gibt Ziele vor:
  z.B. Kundenzufriedenheit
- Mitarbeiter wird zu Engagement ermutigt:
  - Anerkennung durch Vorgesetzten
  - Umsetzung von Vorschlägen aus Q-Zirkel
- Vorgesetzter unterstützt Zielerreichung:
  - Bereitstellung von Ressourcen
  - Förderung von positivem Klima
  - Förderung der Miarbeiterentwicklung

Kommunikation

- Anteil von 35 % der Tätigkeiten bei SW-Entw.
  - Kommunikation mit Anwendern
  - Kommunikation im Team
  - Kommunikation mit Vertretern anderer Hierarchiestufen
- Beispiele QM:
  - Optimierung von Regelungen (Codierstil, Designrichtlinien, Kriterien für Testende)
  - Durchführung von Reviews, Inspektionen, QS-Zirkel

Kommunikation - Einflüsse

Sachebene:	Fakten
Beziehungsebene:	Klima

Balance beider Ebenen - Einstellung zum Gesprächspartner wichtig

Bedrohen des Selbstwertgefühls vermeiden:

- niederreden
- Meinung aufzwängen
- wiederholt unterbrechen
- schulmeistern
- direkter Widerspruch
- Schwächen hart aufzeigen
- Ignorieren
- Zynismus
- persönlich verletzen
- Dritte bevorzugen
- sich mit anderem beschäftigen
- Imponiergehabe
- Überheblichkeit

Kommunikation – Maßnahmen des Vorgesetzten zur Verbesserung

- Klima im Team erhalten, verbessern
  - Frustration vermeiden
  - Mobbing unterbinden
  - Stil der Kommunikation in Team erörtern
  - Regeln für Kommunikation aufstellen
  - Vorbildfunktion
- Selbstwertgefühl der Mitarbeiter erhalten, steigern
  - Anerkennung
  - Förderung
- Prinzipien verständlich machen: Schulung, Rollenspiele

Motivation

„Demotivation vermeiden statt Motivation erzeugen“

- Identifikation bieten
  - Anstoß für Veränderungen von innen
  - nachvollziehbare Begründung
  - Gelegenheit zur Identifikation mit Entscheidungen
- Vision vermitteln
  - klare Ziele und Nutzen erleichtern Veränderungen
  - Unternehmensleitung führt an
  - **erfolgreich** wenn: interessant, herausfordernd, klar, realistisch zu erreichen, ernst gemeint

Konstruktive Maßnahmen: Werkzeuge, Entwurfsmethoden, Sprachen

Inhaltliche Behandlung siehe Softwaretechnik

Objektorientierung, Modularisierung, Modellüberprüfung ...

- Werkzeuge zur Unterstützung bei Routine-Arbeiten, umfangreichen Koordinationsaufgaben
- Modelle helfen, Komplexität zu beherrschen
- Generatoren vermeiden (fehleranfällige) Arbeitsschritte
- Verteilte Werkzeuge unterstützen Einhaltung von Abläufen und Kooperation in Teams

CASE-Werkzeuge: Einfluss auf Qualitätsmerkmale

Speichern von Modellen in Repository

Konsistenzerhalt über Modellgrenzen hinweg

Generieren von Dokumentation

Konsistenz, Aktualität, Verständlichkeit der Dokumentation > Klarheit, Korrektheit

Unterstützung bei Navigation und Verfolgen von Zusammenhängen z.B. Anforderungen - Modell

Vollständigkeit, Korrektheit

Wartbarkeit, Portierbarkeit

CASE-Werkzeuge #2: Einfluss auf Qualitätsmerkmale

Unterstützung bei Routinetätigkeiten

Automatisierung von Schritten

Generieren von Dokumentation

Einsparung von Aufwand > Produktivität

Vermeidung von Fehlern > Korrektheit

Durchführung von Prüfungen an Modell und Implementierung

Einsparung von Aufwand > Produktivität

Frühzeitige Vermeidung von Fehlern
> Produktivität, Korrektheit

Entwurfsmethoden: Einfluss auf Qualitätsmerkmale

Abstraktion, Generalisierung

Best-Practice-Lösungen, Patterns

Angemessenheit der Modelle > Klarheit

Strukturierung, Modularisierung, Kapselung

Flexibilität der Architektur > Wartbarkeit, Portabilität

Programmiersprachen: Einfluss auf Qualitätsmerkmale

Symbolische Bezeichner,

Strukturierung von Steuerfluß und Daten

Modulkonzept mit Kapselung

Strenges Typkonzept

Verbesserung von Struktur und Verständlichkeit > Klarheit, Testbarkeit,

Mächtige Sprachelemente, DSLs

Einsparung von Aufwand, Vermeiden von Fehlern > Produktivität

Analytische QM-Maßnahmen

Dynamische Maßnahmen

Test

- Modultest, Integrationstest
- Black-Box-, White-Box-Test
- Lasttest

Statische Maßnahmen

- Audit
- Review / Inspektion
- Walkthrough
- Korrektheitsbeweiser
- Symbolische Ausführung

Gegenüberstellung statische und dynamische Maßnahmen

	Statische Maßn. z.B. Inspektion	Dynamische Maßnahmen: Test
Ablauf	Prüfobjekt liegt vor, wird begutachtet	Prüfobjekt wird ausgeführt, Verhalten beobachtet
Produkte	Alle Arten	Nur ausführbare Produkte (Code)
Phasen	Frühzeitig möglich	Erst nach teilweiser Implementierung
Fehler, Mängel	Auch innere Q-Merkmale: Wartbarkeit etc.	Nur Fehlverhalten (äußere Q-Merkmale)

Effizienz = Nutzen / Aufwand

Nutzen analytische Maßnahmen:

- Risiko geringer, Fehler und Mängel früher erkannt → Fehlerfortpflanzung verhindert, Rework verringert

Nutzen konstruktive Maßnahmen:

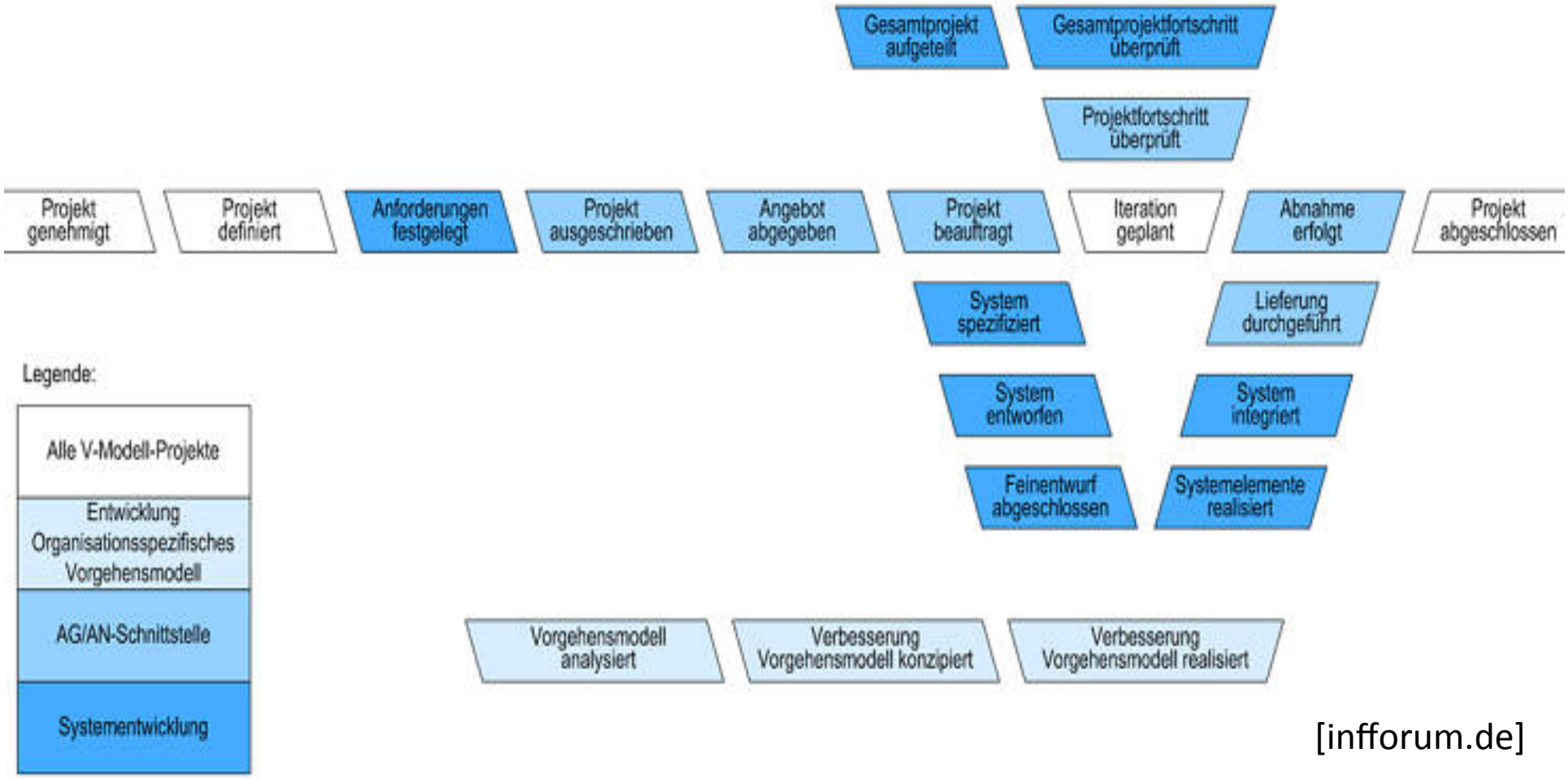
- Fehler und Mängel vermieden, Rework vermieden

deutlich besseres Aufwand-Nutzen-Verhältnis

Aufwand:

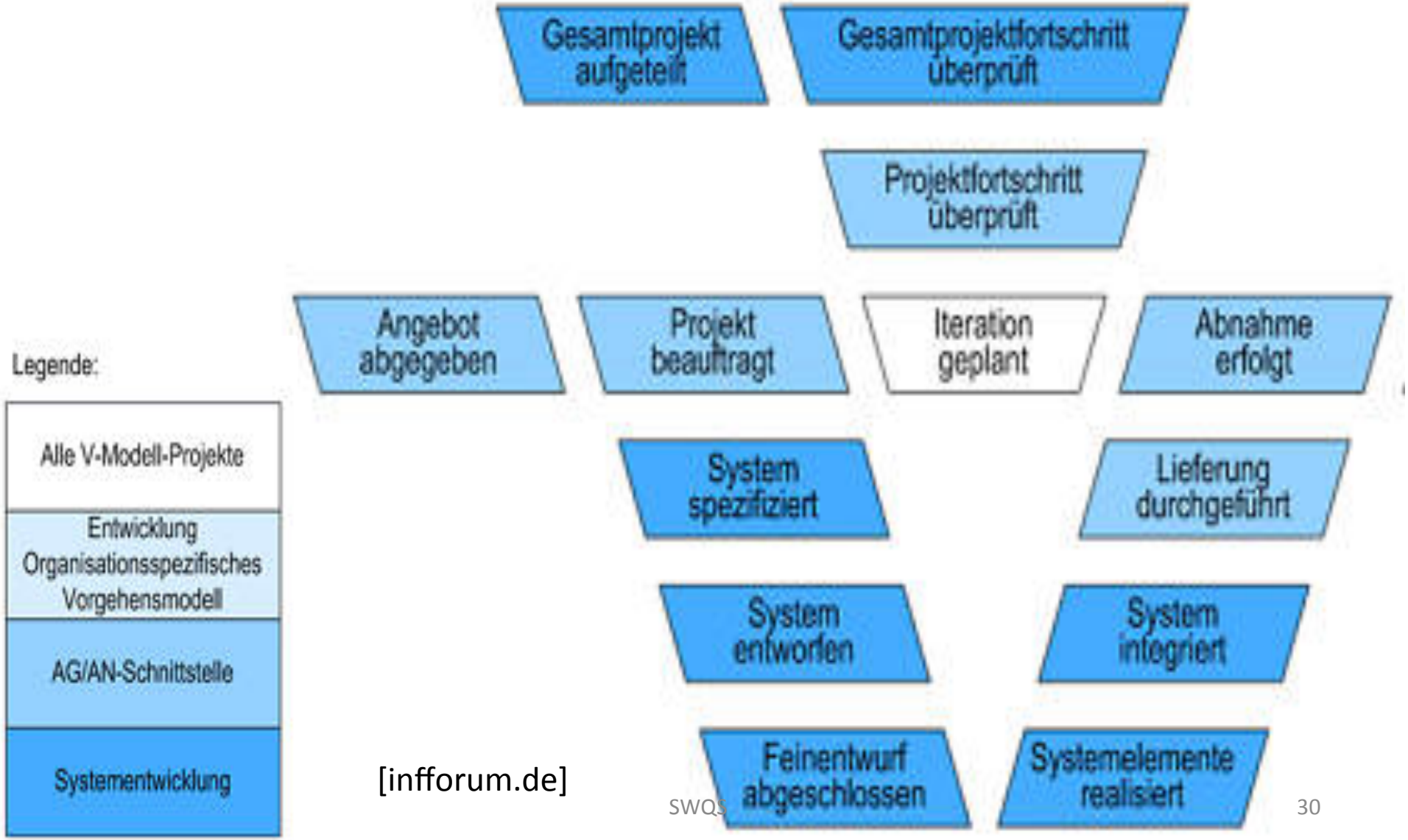
- QM-Mitarbeiter, Werkzeuge, Materialen, unproduktive Zeit der Entwickler

Entwicklungsprozess V-Modell XT



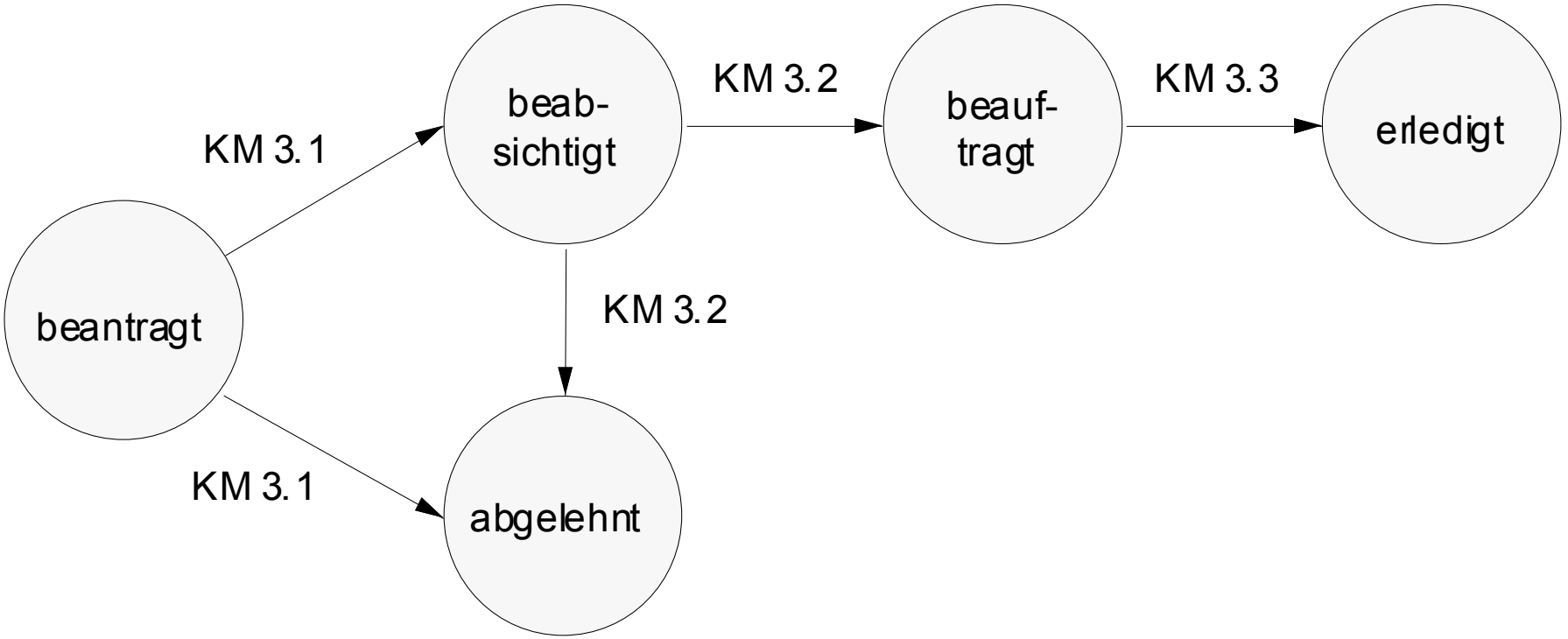
[infforum.de]

Produkt in zentraler Position



[infforum.de]

Änderungs(Auftrags-)zustände

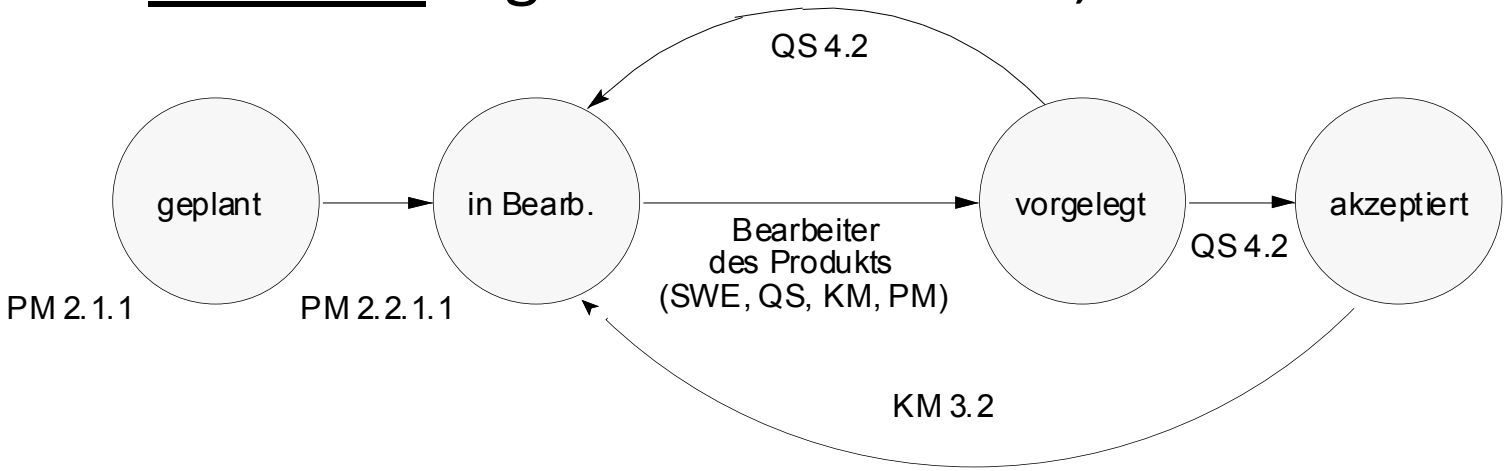


Status einer Änderung als Steuerungsmittel
[V-Modell]

Produktzustände im V-Modell

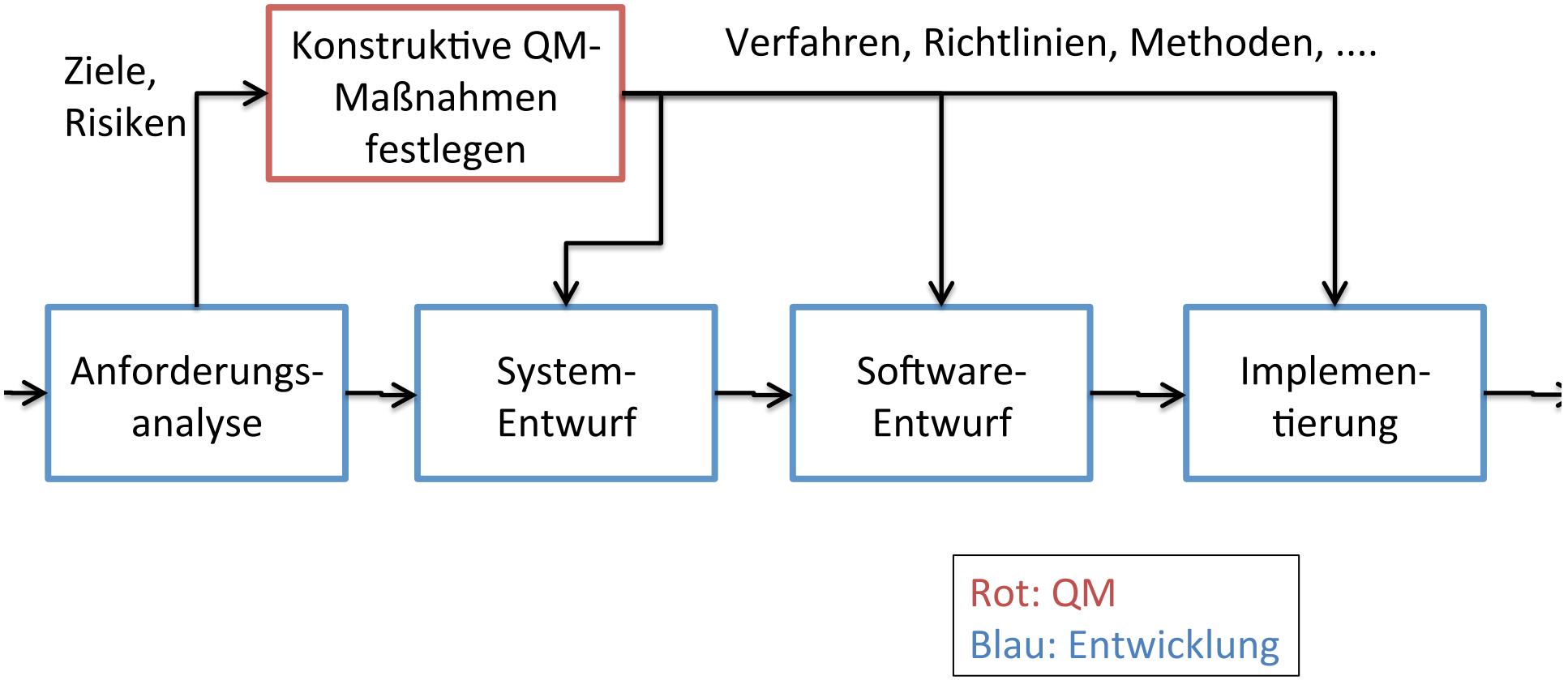
Freigabestatus von Produkten für Planverfolgung, vollständige Prüfungen und konfliktfreie Bearbeitung

Produkt: Begriff des V-Modells, besser: Artefakt



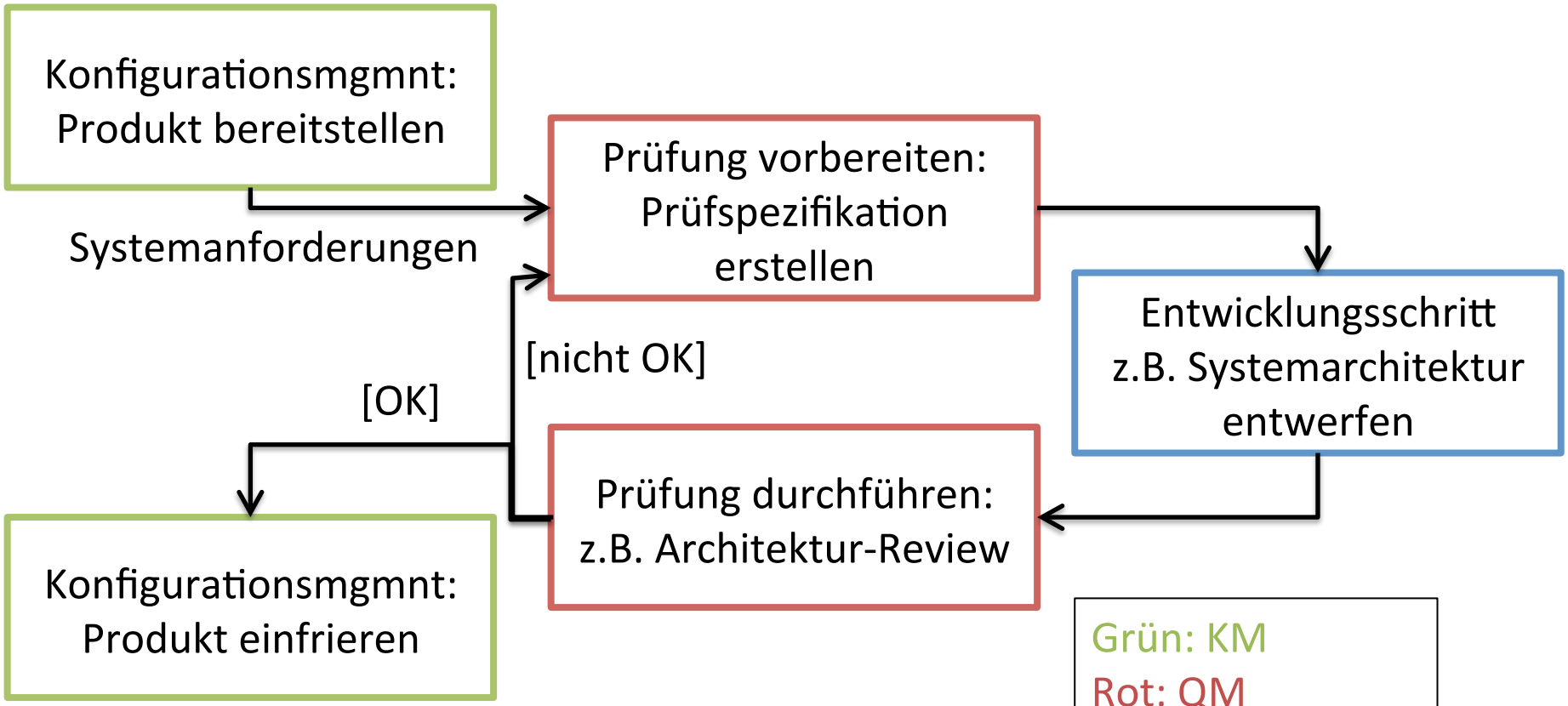
Auswirkung z.B. Zeitpunkt der 100% Fertigstellung eines Moduls anhand erfolgreicher Prüfung erfaßbar

QM im Entwicklungsprozess: Konstruktive Maßnahmen



QM im Entwicklungsprozess: Analytische Maßnahmen

- Prüfung vorbereiten vor jedem Entwicklungsschritt
- Prüfung durchführen während/nach jedem Entwicklungsschritt
- Rückmeldung an Projektmanagement



Konstruktive Maßnahmen: Abhängig von Risiken

	Art der Vorgaben für den Entwickler	**Verpflichtungen seitens des Entwicklers**
Stufe "niedrig"	keine Vorgaben	keine Verpflichtungen
Stufe "mittel"	statistische Vorgaben (z.B. Mindestabdeckung) zur Durchführung der Prüfung	Dokumentation muß den Vorgaben genügen
Stufe "hoch"	genaue Spezifikation der Vorbereitung, Durchführung und Auswertung der Prüfung	Prüfprotokoll gemäß Vorgaben QM

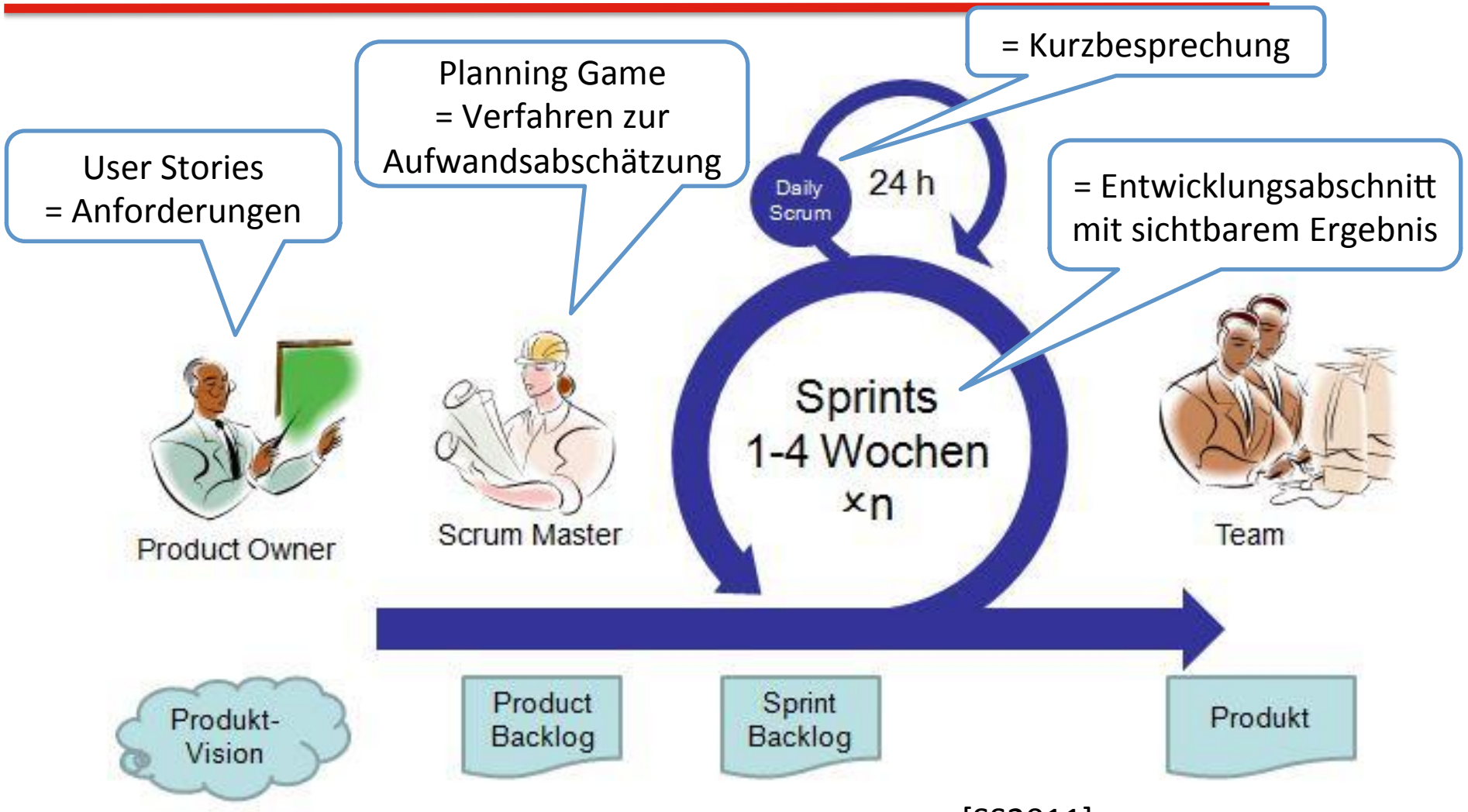
Analytische Maßnahmen: Verzahnen mit Entwicklung und KM

- Bild auschecken, Prüfungvorgaben. Entwickeln, Prüfung, einchecken
- Gegenüberstellen zu Testgetriebener Entwicklung

Agile Prinzipien

- Kleine Schritte statt Big Bang
- Alles verifizieren
- Eigenem Können immer misstrauen
- Nächste Schritte ehrlich besprechen
- Konsequent vereinfachen
- Überarbeiten und Refactoring
- Pareto: das Wichtigste (Ergebniswirksamste, Riskanteste, Schwierigste) zuerst
- Erfahrungen gemeinsam machen und teilen

Agiles Vorgehen: Scrum



[SS2011]

Grundlegende Rollen in Scrum

Product Owner
stellt fachliche Anforderungen und priorisiert sie

Stakeholder
Beobachter und Ratgeber

Team
entwickelt das Produkt

ScrumMaster
managt den Prozeß und beseitigt Hindernisse

Scrum und Testen?

Pro:

- Nutzerzufriedenheit mit großer Bedeutung
- Unit Tests, frühzeitiges Testen und testgetriebene Entwicklung vorgesehen

Kontra:

- Keine festen Phasen, sondern fast kontinuierliche Freigaben
- keine formalen Prozesse (außer Sprint)
- Wenige formal definierte Artefakte
- Langfristige Ziele im Hintergrund

Maßnahmen für Steigerung des Pro:

- Akzeptanztests mit Mehr-Augen-Prinzip
- Externes QM-Team für Systemtests, Regressionstests, Testautomation sowie Defect Tracking
- Testen innerhalb des gleichen Sprint
- Gemeinsame „Definition of Done“: Aktivitäten, Ergebnisse und Qualitäten
- Pair Testing analog zum Pair Programming
- User Stories für Qualitätsziele

Scrum und konstruktive QM?

Pro:

- Nutzerzufriedenheit mit großer Bedeutung
- Hohe Motivation des Teams
- Gute Qualifikation des Teams
- Lernen, Fehlervermeidung, Effizienz des Entwicklungsprozesses im Fokus
- Einfachheit, Wirksamkeit als Ziele

Kontra:

- Methoden nicht formal festgehalten, keine formal definierte Artefakte
- Langfristige und Qualitätsziele eher im Hintergrund

Maßnahmen für Steigerung des Pro:

- Bewusste Reflektion von Best Practices
- Bewusste Auswertung von Fehlern und Mängeln
- User Stories für Qualitätsziele
- Checklisten als vorgefertigte Arbeitsmittel als Speicher für Best Practice einsetzen