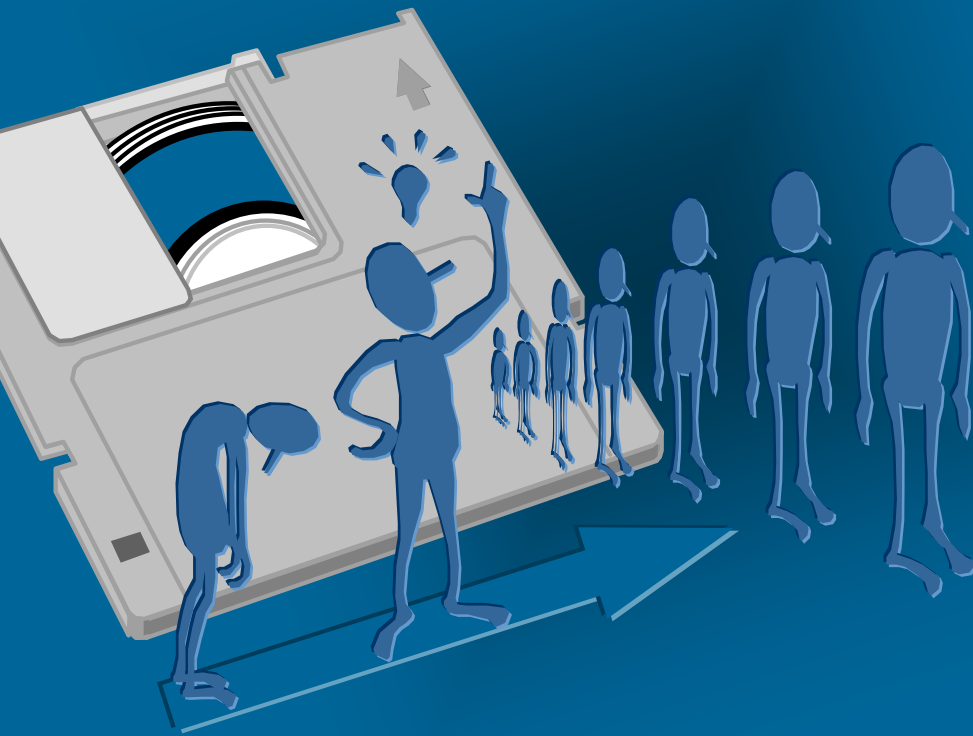


Softwaremanagement



Der Weg vom Einzelkämpfer zum Entwicklungsteam

Prof. Dr.-Ing. Dagmar Meyer



Fachhochschule Braunschweig/Wolfenbüttel
Fachbereich Elektrotechnik

Übersicht

- Motivation
- Einführung
- Das Software-Entwicklungsteam
- Der Software-Entwicklungsprozess ...
- ... und wie er mit Leben erfüllt wird
- Dokumente - nicht nur für Bürokraten
- CMM - was steckt dahinter?
- Zusammenfassung



Übersicht

- Motivation
- Einführung
- Das Software-Entwicklungsteam
- Der Software-Entwicklungsprozess ...
- ... und wie er mit Leben erfüllt wird
- Dokumente - nicht nur für Bürokraten
- CMM - was steckt dahinter?
- Zusammenfassung



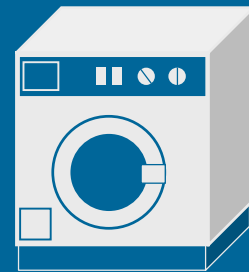
Software ist (fast) überall ...



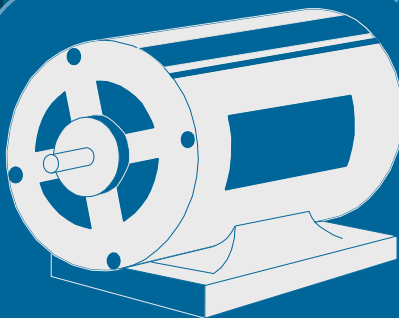
Automatisierungstechnik



Automobil



Hausgeräte



Antriebstechnik



Telekommunikation



Verkehrstechnik

Software

Software als Wettbewerbsfaktor

[...] Mit der Studie "Analyse und Evaluation der Softwareentwicklung in Deutschland" liegt jetzt eine umfangreiche Untersuchung zur wirtschaftlichen und technologischen Bedeutung von Software und von Softwareentwicklung in Deutschland vor.

[...]

Die Studie zeigt, dass Software und Softwareentwicklung für nahezu alle Produkte und Prozesse quer durch alle Branchen der deutschen Volkswirtschaft zunehmend zum wettbewerbsbestimmenden Faktor wird. [...]

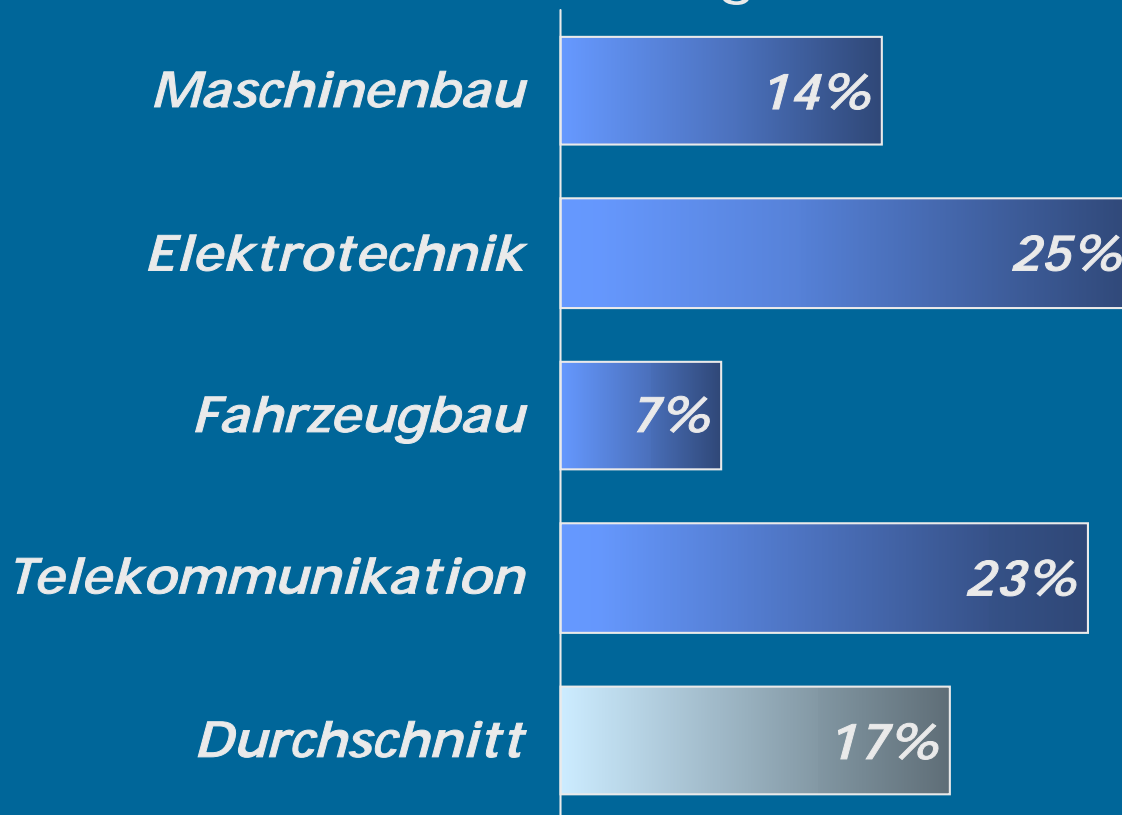
Zuverlässigkeit und Funktionalität sind aus Sicht der Unternehmen die wichtigsten Qualitätseigenschaften von Software. Bisher ist aber die ingenieurmäßige Entwicklung qualitativ hochwertiger Software in Deutschland noch zu gering verbreitet. [...]

Quelle: Pressemitteilung Nr. 206/2000 des bmb+f vom 29.12.2000



Software gewinnt an Bedeutung ...

Anteil der Softwareentwicklung an Neuentwicklungskosten



Quelle: Studie "Analyse und Evaluation der Softwareentwicklung in Deutschland"

Übersicht

- Motivation
- **Einführung**
- Das Software-Entwicklungsteam
- Der Software-Entwicklungsprozess ...
- ... und wie er mit Leben erfüllt wird
- Dokumente - nicht nur für Bürokraten
- CMM - was steckt dahinter?
- Zusammenfassung



Softwaremanagement – warum?

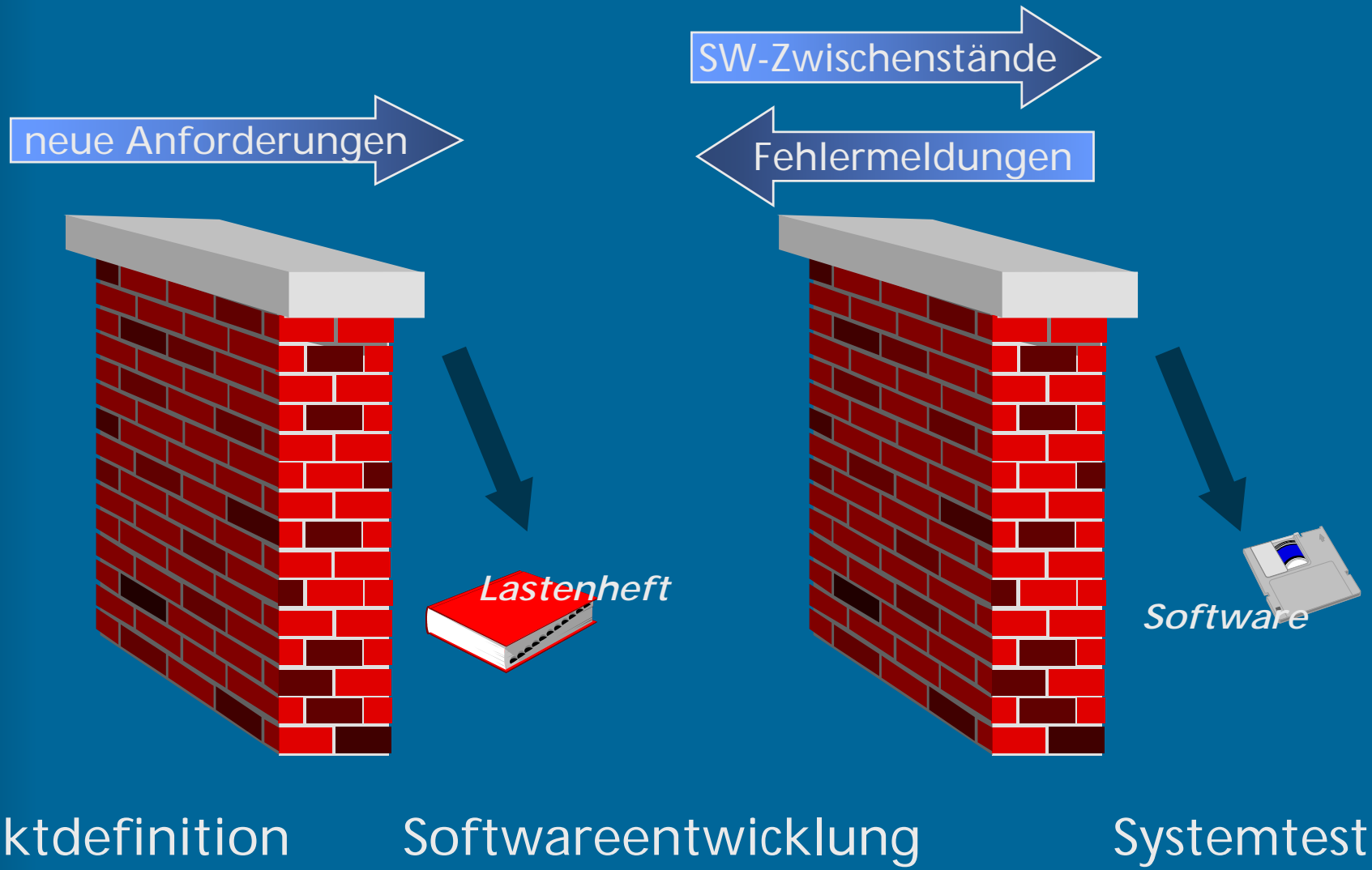
- Software wird heute auch in Branchen entwickelt, die (noch) über geringe oder keine Erfahrung auf diesem Gebiet verfügen
- Software wird immer komplexer
 - Zunahme der Leistungsfähigkeit der Hardware (schneller, mehr Speicher)
 - Neuere Methoden im Bereich der Softwaretechnik finden Verbreitung (z.B. Objektorientierung)
- Kürzere Entwicklungszyklen
 - Time to market von 1-2 Jahren keine Seltenheit
- Bildung großer Entwicklungsteams ist erforderlich



Worum geht es eigentlich?

- Software-Entwicklungsprozess ...
 - Wie definiere ich ihn?
 - Wie kann er umgesetzt werden?
- Organisation ...
 - Welche Aufgaben (Rollen) gibt es im Projekt?
 - Welche Qualifikationen sind notwendig?
 - Wie organisiere ich ein Entwicklungsteam?
- Planung ...
 - Was plane ich?
 - Wie plane ich?

Scope



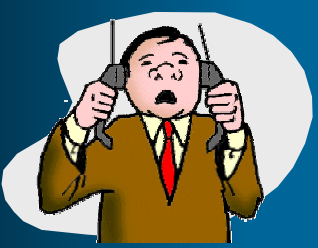
Übersicht

- Motivation
- Einführung
- **Das Software-Entwicklungsteam**
- Der Software-Entwicklungsprozess ...
- ... und wie er mit Leben erfüllt wird
- Dokumente - nicht nur für Bürokraten
- CMM - was steckt dahinter?
- Zusammenfassung



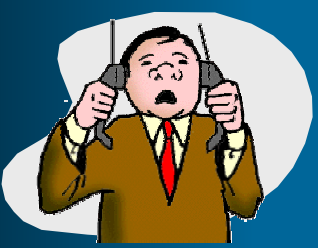
Rollen im Entwicklungsteam





Der Softwaremanager ...

- Leitet das Entwicklungsteam
- Ist verantwortlich für ...
 - die Planung und die Einhaltung von Terminen
 - das Budget
 - den Aufbau des Entwicklungsteams
- Erstattet dem Management Bericht über den Projektfortschritt
- Bildet die Schnittstelle zu
 - Hardwareentwicklung
 - Systemtest
 - Marketing / Vertrieb



Der Softwaremanager

– Wie soll er sein? –

- Der Softwaremanager sollte *nie* gleichzeitig auch Architekt sein!
- Wissen über SW ist gut und wichtig – aber die Architektur bestimmt der Architekt!
- Wichtig sind ...
 - Kommunikationsfähigkeit
 - Durchsetzungsvermögen
 - Planungsvermögen und
 - Eine Menge Verständnis für die (gar nicht so seltene) Spezies der Software-Entwickler und deren Bedürfnisse und Eigenheiten!

Der Softwarearchitekt ...



- Erstellt das Design (die Architektur) der SW
- Ist verantwortlich für ...
 - Die „Lebensfähigkeit“ des Designs (Wartbarkeit, Erweiterbarkeit, Realisierbarkeit, ...)
 - Auswahl der Programmiersprache, Toolkette usw.
 - Coding-Standards und deren Einhaltung
 - Die Durchführung von Codeinspektionen / Reviews
- Berät und unterstützt den Softwaremanager bei der Planung
- Ist *immer* gleichzeitig auch Entwickler

Der Softwarearchitekt

– Was zeichnet ihn aus? –



- “ An architect always implements! “
- Entwicklungserfahrung
- Sehr gutes Systemwissen und –verständnis
- Fundiertes Wissen im Bereich Softwaretechnik
- Überzeugungskraft, Durchsetzungsvermögen
- Gute pädagogische Fähigkeiten
- Soziale Kompetenz und Teamfähigkeit
- „Menschliche Kompatibilität“ zum Softwaremanager ist ausgesprochen hilfreich!

Der Konfigurationsmanager ...



- Überwacht die gesamte Versionskontrolle
- Erlässt projektspezifische KM-Richtlinien
- Ist verantwortlich für ...
 - Auswahl und Bereitstellung geeigneter Werkzeuge (Versionsverwaltungssystem)
 - Backup-Strategien, Datensicherung usw.
 - Die Erzeugung und Verwaltung der unterschiedlichen Softwarestände
 - Die Versionskontrolle von
 - Software
 - Tools
 - Dokumenten

Der Konfigurationsmanager

– Was muss er können? –



- Gute Kenntnis der Entwicklungsumgebung
- Guter Überblick über das Gesamtprojekt
- Der KM *muss* selbst absolut von der Wichtigkeit seiner Aufgabe überzeugt sein!
- Überzeugen, überzeugen, überzeugen!
- Wichtige Eigenschaften sind:
 - Organisationstalent
 - Konsequenz
 - Belastbarkeit
 - Ruhe bewahren in Stresssituationen

Der Entwickler ...



- Entwickelt Softwarekomponenten und erstellt deren Design (unter Beachtung der Gesamtarchitektur des Systems)
- Ist verantwortlich für ...
 - Die Funktionalität
 - Die Qualität
 - Die termingerechte Fertigstellung „seiner“ Komponenten bzw. Funktionen
- Führt Codeinspektionen / Reviews durch
- Beseitigt Fehler in der Software



Der Entwickler ...

... und seine Eigenschaften

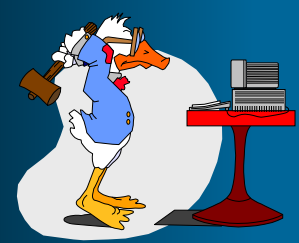
- Gute Kenntnisse in Softwaretechnik allgemein
- Sehr gute Beherrschung der verwendeten Sprache und Entwicklungsumgebung
- Gute sprachliche Ausdrucksfähigkeit
- Belastbarkeit
- Hohe Teamfähigkeit
- Verantwortungsbewusstsein
- Selbständigkeit (!)
- Lebenslanges Lernen ist absolut notwendig

Der Toolmaker ...

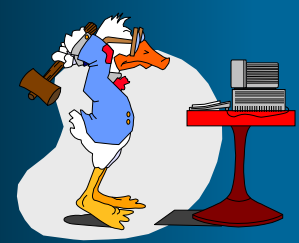


- Erstellt
 - Hilfsmittel für die Entwickler
 - Werkzeuge, die neben der Toolkette für die Generierung der Software erforderlich sind
- Wartet und pflegt Compiler, Editor usw.
- Ist Ansprechpartner bei Problemen
- Organisiert Updates und hält Kontakt zu den Herstellern der verwendeten Tools
- Konfiguriert diese, soweit erforderlich, entsprechend den Bedürfnissen der Entwickler

Der Tester ...



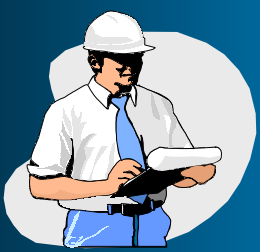
- Testet die Software vor der Übergabe an den Systemtest
- Die Tests können
 - „von Hand“ oder
 - Automatisch (Testautomatisierung) durchgeführt werden.
- Erstellt Testpläne und Testprotokolle
- Meldet gefundene Fehler an die Entwickler und / oder trägt sie in die Fehlerdatenbank ein
- Hält die Testaufbauten auf aktuellem Stand



Der Tester

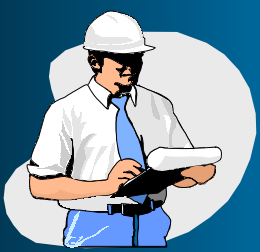
– Woran erkennt man ihn? –

- Der Tester braucht destruktive Energie!
- Der Tester verfügt über ein gutes Technikverständnis.
- Systemverständnis erleichtert den Entwicklern die Arbeit!
- In der Praxis gilt oft Entwickler = Tester (aber bitte nicht für die eigene Software ;-)
- Wichtig sind
 - Geduld und Ausdauer
 - Verantwortungsbewusstsein
 - Arbeitsorganisation



Der Technologiepate ...

- Berät das Entwicklungsteam in technischen Fragen
- Kennt oder erfragt spezielle Wünsche und Anforderungen der zukünftigen Kunden
- Berät den Softwaremanager bei der Planung der Produktstufen (Prioritäten beim Kunden!)
- Beurteilt ggf. die Relevanz und Dringlichkeit geforderter Fehlerbehebungen
- Kommt evtl. aus Marketing oder Service



Der Technologiepate

– Experte und Ratgeber –

- ... sollte die Sprache der Kunden *und* die Sprache der Entwickler sprechen.
- ... muss sich durch hohe Fachkompetenz auszeichnen.
- ... muss dem Entwicklungsteam jederzeit für Fragen zur Verfügung stehen (kurze Reaktionszeiten!)
- ... sollte das Projekt bis zur Markteinführung begleiten.



Der Qualitätsmanager ...

- Überwacht die Einhaltung des Software-Entwicklungsprozesses und der projektspezifischen Richtlinien
- Erstellt Statistiken rund um das Thema Qualität
- Hat ausschließlich eine beratende Funktion, d.h. keine direkte Weisungsbefugnis gegenüber dem Entwicklungsteam
- Berichtet direkt an das Management



Qualitätsmanager ... werden selten geliebt!

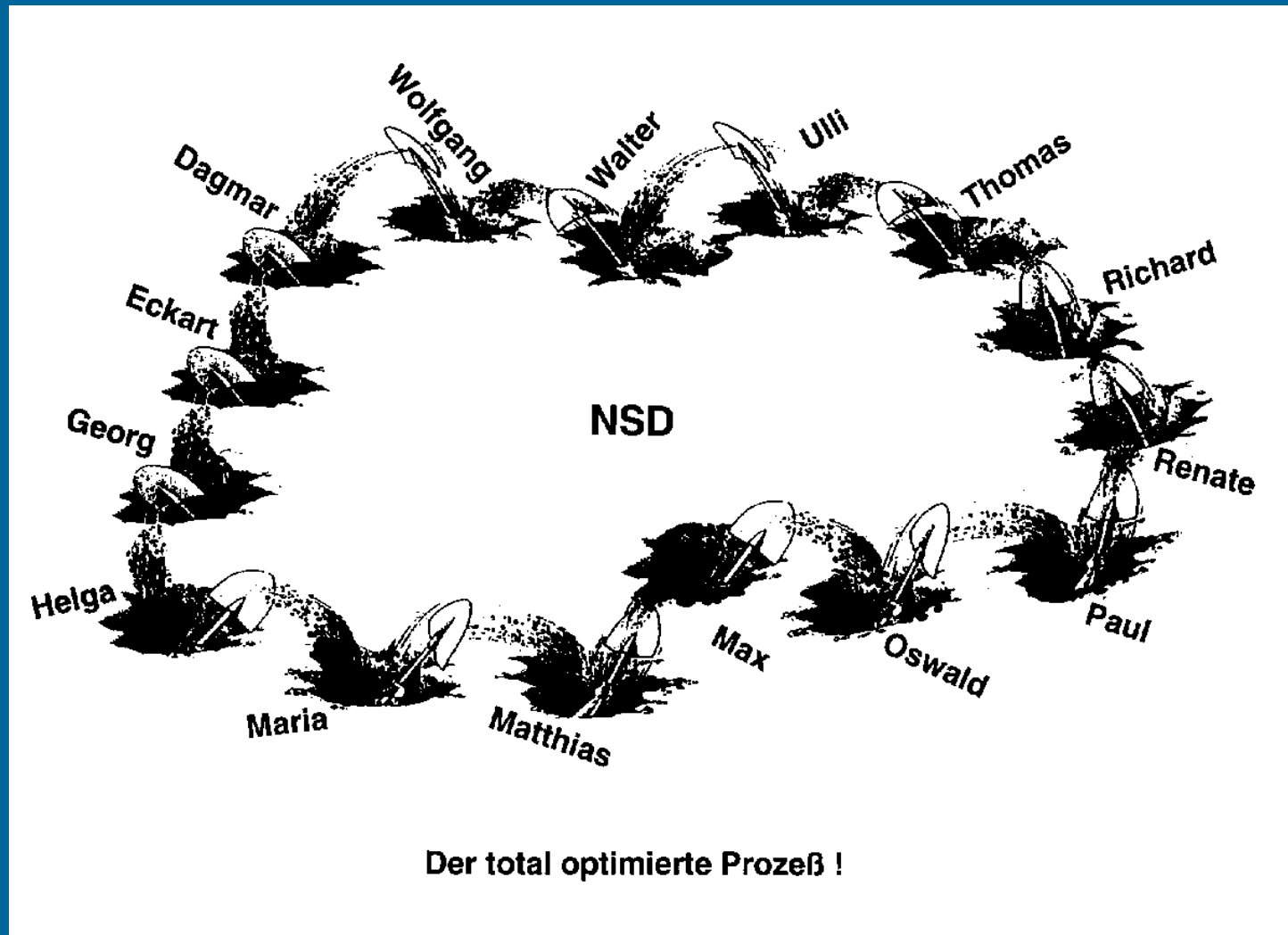
- Der QM ist ein Experte in allen Fragen des Entwicklungsprozesses.
- Er muss sich durch Überzeugung durchsetzen.
- Er muss „einen guten Draht“ zu den Entwicklern und zum Softwaremanager haben.
- Nützliche Eigenschaften:
 - Gute Kommunikationsfähigkeit
 - Ausstrahlung von Autorität
 - Eine gehörige Portion Idealismus schadet nicht

Übersicht

- Motivation
- Einführung
- Das Software-Entwicklungsteam
- Der Software-Entwicklungsprozess ...
- ... und wie er mit Leben erfüllt wird
- Dokumente - nicht nur für Bürokraten
- CMM - was steckt dahinter?
- Zusammenfassung



Der Software-Entwicklungsprozess ???



Der Software-Entwicklungsprozess

– Wasserfall vs. Inkrementell –

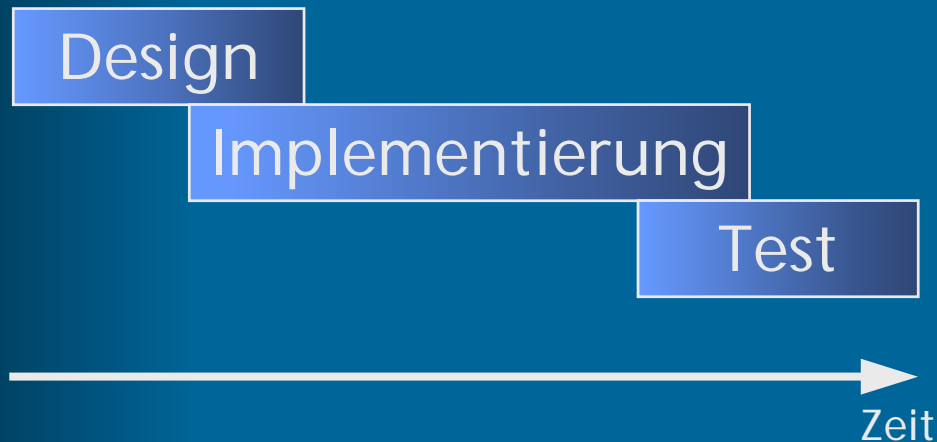


- klassisches Prozessmodell
- für kleine Projekte sinnvoll
- 3 Phasen:
 - Design
 - Implementierung
 - Test

- neueres Prozessmodell
- für fast alle Projekte sinnvoll
- Aufteilung des Entwicklungszeitraums in N Inkremente
- innerhalb der Inkremente: Wasserfallmodell



Das Wasserfallmodell



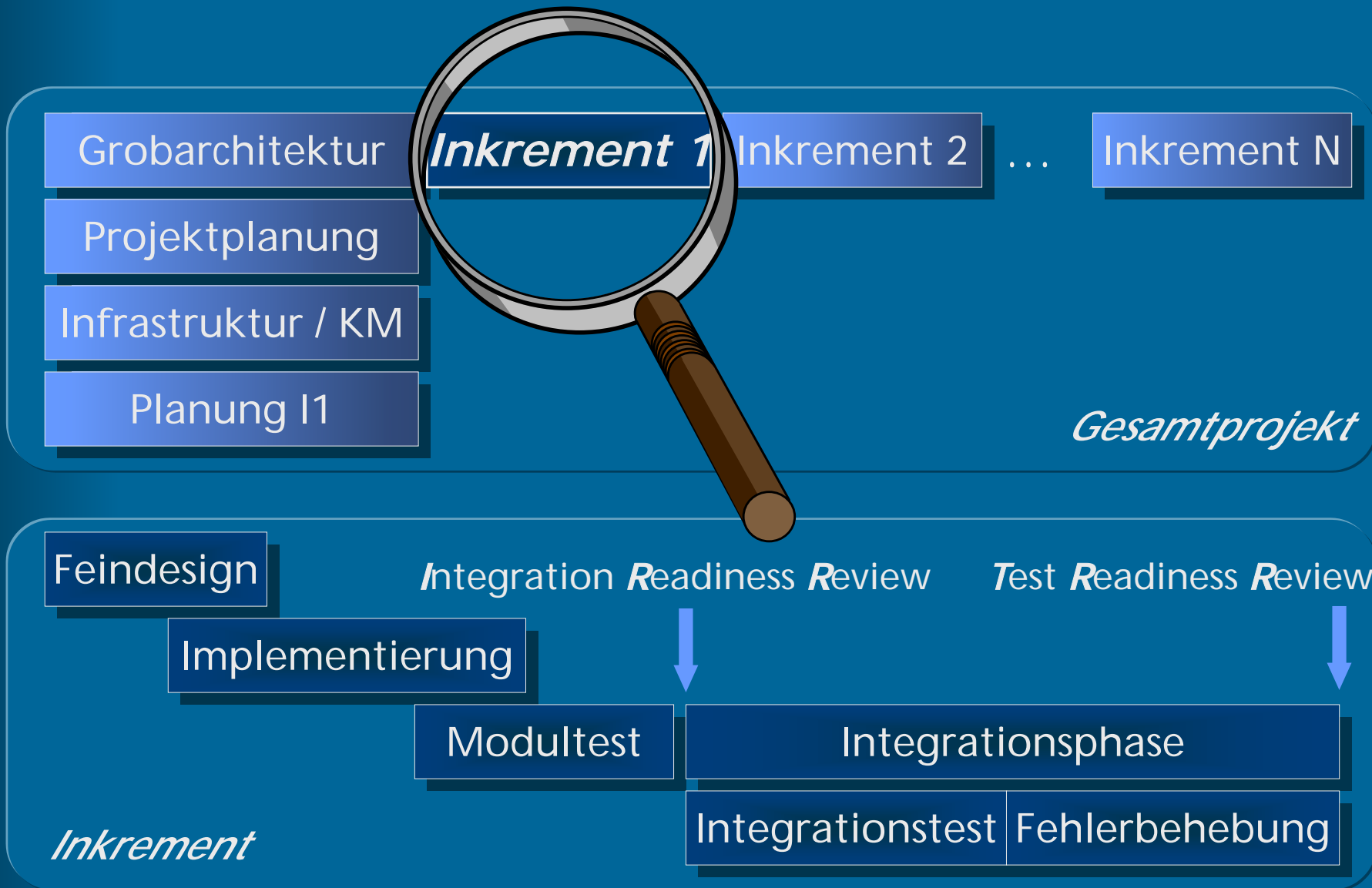
☺ *Vorteile*

- Prozess ist sehr einfach zu implementieren
- läuft quasi von selbst
- geringer Management-Aufwand

☹ *Nachteile*

- Fehler werden spät erkannt
- Reaktion auf geänderte Anforderungen schwierig
- Verfolgung des Projektfortschritts schlecht möglich
- Entwickler geraten zum Zieltermin hin stark unter Zeitdruck

Der inkrementelle Entwicklungsprozess



Der inkrementelle Entwicklungsprozess

– Bewertung –

😊 *Vorteile*

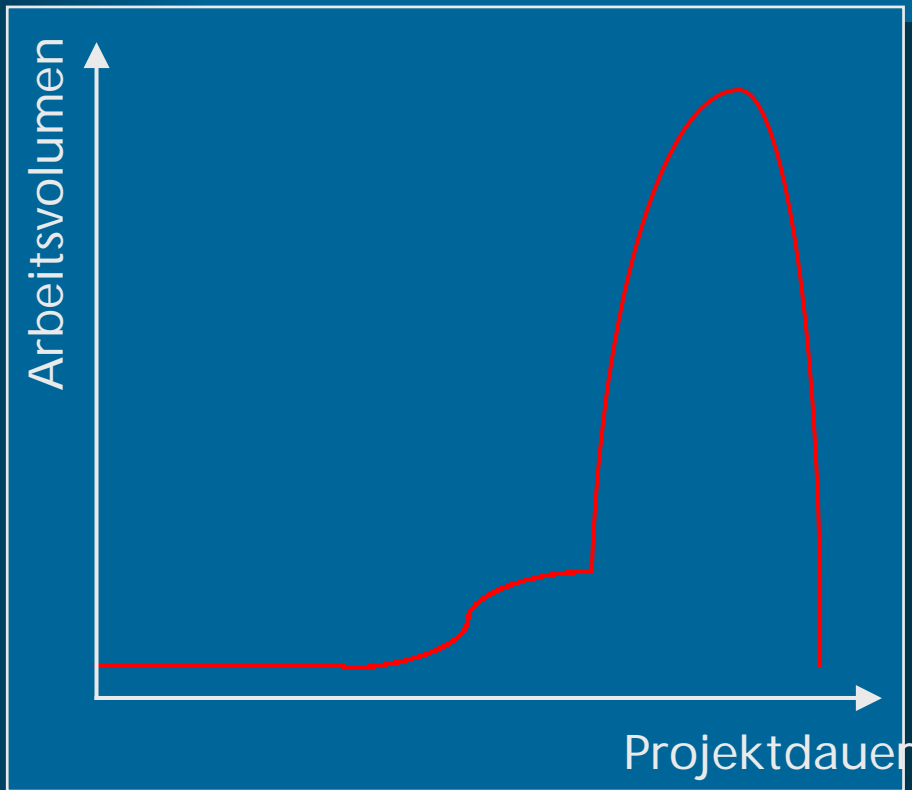
- Flexible Reaktion auf geänderte Anforderungen
- Reduktion der Personenabhängigkeit (!)
- Detailliertes Tracking des Projektfortschritts möglich
- Frühes Erkennen von Designfehlern möglich
- Fehlerbehebung schon während der Entwicklung
- gleichmäßigere Belastung der Entwickler

☹️ *Nachteile*

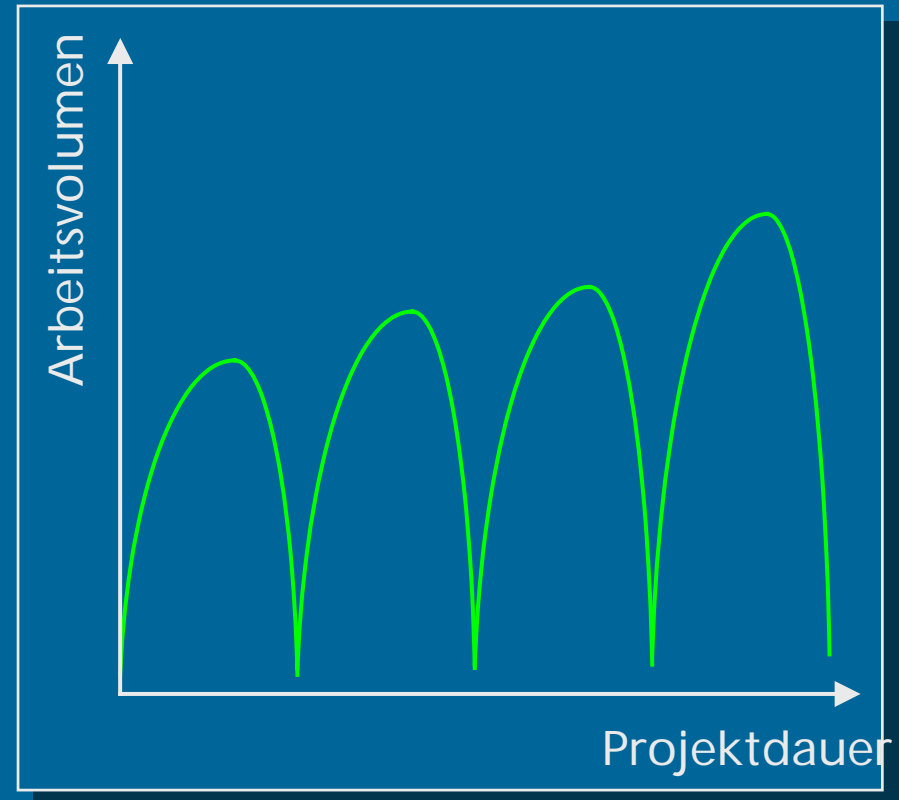
- Leicht erhöhter Managementaufwand
- Umgang mit dem Prozess muss „geübt“ werden
- Kaum Erholungsphasen für die Entwickler
- Detaillierte Planung erforderlich

Arbeitseinsatz der Entwickler

Wasserfallmodell



Inkrementelles Modell



Projekttablauf in der Software-Entwicklung

- Startphase -

<i>wer ...</i>	<i>macht was ...</i>
Softwarearchitekt	Grobdesign Toolkette auswählen (wenn HW bekannt!)
Softwaremanager	Projektplanung Planung des 1. Inkrements Aufbau des Entwicklungsteams
Konfigurationsmanager	Planung des Konfigurationsmanagements Planung der notwendigen Infrastruktur
Qualitätsmanager	Planung des Qualitätsmanagements
Wenige (!) Entwickler / Tester	Coding-Standards erstellen Entwicklungsumgebung bereitstellen Testumgebung / Testautomatisierung

Projekttablauf in der Software-Entwicklung

– Integrationsphasen –

<i>wer ...</i>	<i>macht was ...</i>
Konfigurationsmanager	Hat während der Integration die Kontrolle über das Projekt Bestimmt Zeitpunkte für die Integrationen
Softwaremanager	Entscheidung über das Inkrementende Zuweisung von Fehlern an die Entwickler Planung des folgenden Inkrements
Softwarearchitekt	Unterstützung des Softwaremanagers Integrationstest und Fehlerbehebung
Qualitätsmanager	Überwachung des Integrationstests Fehlerstatistik Durchführung von IRR und TRR
Tester / Entwickler	Integrationstest und Fehlerbehebung

Übersicht

- Motivation
- Einführung
- Das Software-Entwicklungsteam
- Der Software-Entwicklungsprozess ...
- ... und wie er mit Leben erfüllt wird
- Dokumente - nicht nur für Bürokraten
- CMM - was steckt dahinter?
- Zusammenfassung



Wie alles beginnt ...

Wir brauchen ein
Produkt, das ...

Unser Konkurrent
XY hat das ja auch
schon ...

Es muss auf jeden
Fall dies und das
und
noch etwas können
...

Und billiger sein
als ...



Auftakt



Softwaremanager

- Erste Budgetplanung
- Entwicklungsteam aufbauen
- Koordination mit Hardware, Systemtest usw.
- Anforderungen sammeln



Softwarearchitekt

- Grobdesign erstellen
- Programmiersprache, Toolkette auswählen
- Coding-Standards



Entwickler



Entwickler

Planungsphase

*Inkrement:
Entwicklungsabschnitt,
Ziel: nach außen sichtbare
Teilfunktionalität*

Inhaltliche Definition
des ersten Inkrements

*Schätzklausur mit
verdeckter
Abgabe
der Schätzungen*

Grobe Festlegung der
Inkreme (Anzahl und Ziele)

Aufwandsabschätzung
für die einzelnen WPs

Definition von
Workpackages (WPs)

*logische Softwarekomponente,
z.B. „externe Kommunikation“*

Grobdesign



Inkrementtablauf

*Geplante Zeit abgelaufen
Integration sinnvoll möglich?
Wenn ja: Beginn der Integrationsphase*

Feindesign

Integration *Readiness Review*

Test *Readiness Review*

Implementierung

Modultest

Integrationsphase

Integrationstest

Fehlerbehebung

*Konfigurationsmanager übernimmt
die Kontrolle*

⇒ *Sperrung des Archivs*

⇒ *erste Integration*

Übersicht

- Motivation
- Einführung
- Das Software-Entwicklungsteam
- Der Software-Entwicklungsprozess ...
- ... und wie er mit Leben erfüllt wird
- **Dokumente - nicht nur für Bürokraten**
- CMM - was steckt dahinter?
- Zusammenfassung und Ausblick



Allgemeine Dokumente



Übergeordnete Dokumente

Gültigkeit für alle SW-Projekte in der Organisation, dienen der Prozessdefinition

- Softwaremanagementplan
- Konfigurationsmanagement-Plan
- Qualitätsmanagement-Plan
- ...

Projektspezifische Dokumente

Gültigkeit über die gesamte Dauer eines Software-Projektes

- Grobdesign
- Software-Entwicklungsplan
- KM-Plan (projektspezifisch)
- QM-Plan (projektspezifisch)
- Coding-Standards
- ...



Gültigkeit für jeweils ein Inkrement

- Inkrementplan
- Testplan
- Testprotokolle
- Releasenotes
- Feindesigndokumente
- ...

Übersicht

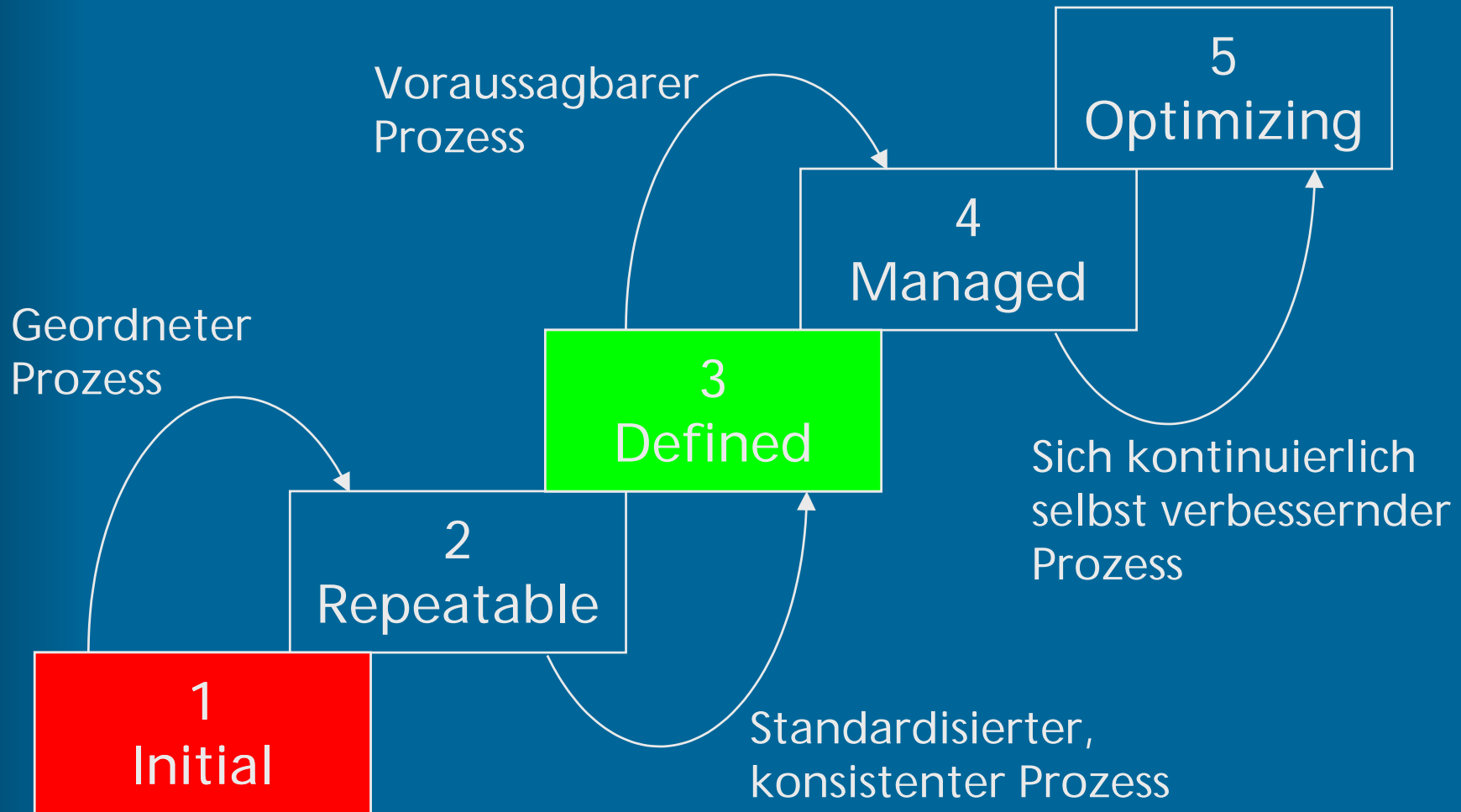
- Motivation
- Einführung
- Das Software-Entwicklungsteam
- Der Software-Entwicklungsprozess ...
- ... und wie er mit Leben erfüllt wird
- Dokumente - nicht nur für Bürokraten
- CMM - was steckt dahinter?
- Zusammenfassung



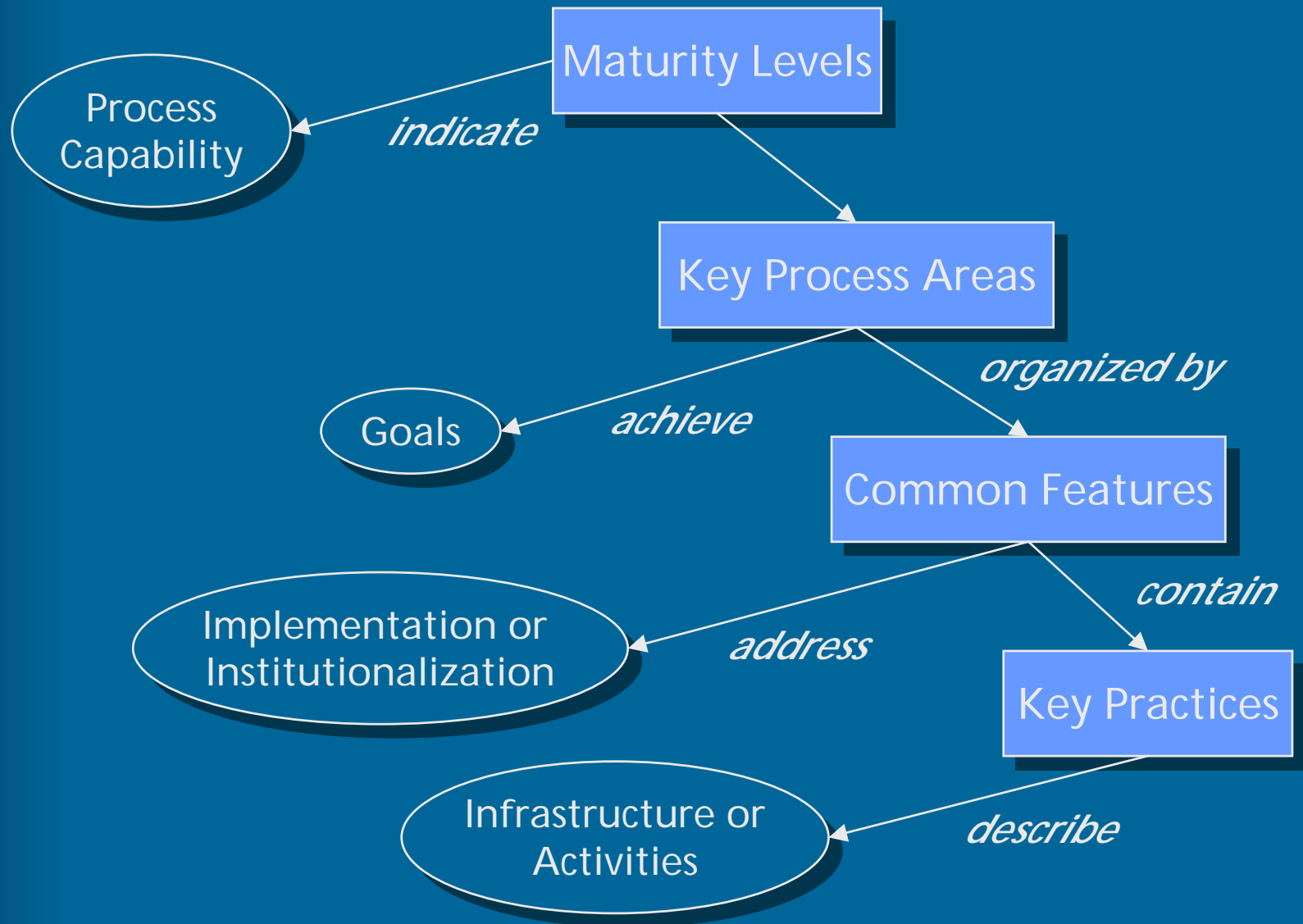
Capability Maturity Model[®] (SW-CMM[®]) for Software

- Entwickelt vom **Software Engineering Institute (SEI)** der Carnegie Mellon Universität
- CMM umfasst:
 - Kriterien zur Bewertung des Istzustands
 - Fragenkataloge zur Durchführung von Audits
 - Aufzeigen von Wegen, wie der Software-Entwicklungsprozess schrittweise verbessert werden kann
- Unterscheidung von 5 „Reifegraden“ des Prozesses

CMM – die 5 Bewertungsstufen



Struktur des Capability Maturity Models



Key Process Areas

– Level 2 (Repeatable) –

- Anforderungsmanagement
 - Kontrolle der Beziehung zum „Kunden“ (⇒ Change Control)
- Software Projektplanung
 - Realistische Planung des Ablaufs der Softwareentwicklung
- Software Projektverfolgung und Aufsicht
 - Fortschrittskontrolle
- Software Unterauftragsmanagement
 - Selektion qualifizierter Unterauftragnehmer
- Software Qualitätssicherung
 - Management erhält Einblick in den Entwicklungsprozess
- Software Konfigurationsmanagement
 - Vollständigkeit und Reproduzierbarkeit sicherstellen



Key Process Areas

– Level 3: Defined –

- Prozessaktivitäten in Organisationsstruktur verankern
 - Organisatorische Verantwortung für das Thema Entwicklungsprozess verankern
- Organisationsweite Prozessdefinition
- Trainingsprogramm
 - Training als organisatorische Verantwortung
- Integriertes Softwaremanagement
 - Engineering und Managementaktivitäten bilden einen Prozess
- Software Produkt-Engineering
 - Engineering Prozess (Ablauf technischer Aktivitäten im Projekt)
- Koordination zwischen Gruppen
- Peer Reviews
 - Fehler erkennen, Code besser verstehen (!)



Key Process Areas

– Level 4: Managed –

- Quantitatives Prozessmanagement
 - Quantitative Kontrolle der Erfüllung Prozesses des Projekts
 - Ermittlung der speziellen Ursachen für Abweichungen innerhalb eines messbar stabilen Prozesses
 - ggf. Korrektur der Umstände, die zu der vorübergehenden Abweichung geführt haben
- Software Qualitätsmanagement
 - Ziel: quantitative Bewertung der Qualität der Software eines Projekts
 - Erreichen spezifischer Qualitätsziele
 - Erfordert umfassende statistische Erhebungen

Key Process Areas

– Level 5: Optimizing –

- „Fehlerverhütung“
 - Ursachen für Fehler identifizieren, erneutes Auftreten verhindern
 - Fehleranalyse im Projekt, Ursachenforschung
 - Änderung des Software Prozesses
- Technology Change Management
 - Identifizierung neuer Technologien (Tools, Methoden, Prozesse)
 - Geordnete Einführung in der Organisation
- Process Change Management
 - Kontinuierliche Verbesserung des Software Prozesses
 - Ziele: Verbesserung der Softwarequalität, Produktivitätssteigerung, Verkürzung der Entwicklungszeiten

Übersicht

- Motivation
- Einführung
- Das Software-Entwicklungsteam
- Der Software-Entwicklungsprozess ...
- ... und wie er mit Leben erfüllt wird
- Dokumente - nicht nur für Bürokraten
- CMM - was steckt dahinter?
- Zusammenfassung



Zusammenfassung

- Softwaremanagement wird immer wichtiger
- Ein stabiler, reproduzierbarer Software-Entwicklungsprozess hilft ...
 - Termine einzuhalten
 - Produktentwicklungszyklen zu verkürzen
 - Weitgehende Unabhängigkeit von einzelnen Personen zu schaffen
 - Qualität zu sichern
- Jeder, der sich mit Software beschäftigt, muss sich auch mit diesem Thema auseinandersetzen!



Wer mehr zu dem Thema wissen möchte ...

- <http://www.sei.cmu.edu/cmm/cmm.html>
- Humphrey, Watts S.
Managing the Software Process
SEI Series in Software-Engineering
Addison Wesley Longman Inc., 1990
- Kruchten, Philippe
*From Waterfall to Iterative Lifecycle –
A tough transition for project managers*
Rational Software White Paper
- Brooks, Frederick P.
*The mythical man-month – essays on software engineering
Anniversary edition*
Addison Wesley Longman Inc., 1995



Wer mehr zu dem Thema wissen möchte ...

- Zachary, G. Pascal
Show-Stopper!: The Breakneck Race to Create Windows NT and the next Generation at Microsoft
Free Press, 1994
- Yourdon, Edward
Death March - The Complete Software Developer's Guide to Surviving „Mission Impossible“ Projects
Prentice Hall, 1997

