

# Qualitätsmanagement

## Software-Engineering für große Informationssysteme

TU-Wien, Sommersemester 2004

Klaudius Messner

# Überblick

- Was ist Qualität
- Wer / Wann / Was / Wie
- Konstruktive und Analytische QS
  - Beispiele für Konstruktive QS
  - Beispiele für Analytische QS
  - Testen und der Test-First-Ansatz

# Kleine Provokation zum Beginn

## Was ist Qualität?



- Qualität ist das, was der Kunde braucht!
- Die Methoden der Raumfahrttechnik bei einem Kunden anzuwenden, der auf schnelle Marktänderungen reagieren muss, ergibt nicht das, was der Kunde braucht
- Die Methoden von XP auf Software für die Raumfahrt anzuwenden wäre eher unkonventionell und ergibt wahrscheinlich auch nicht das, was der Kunde braucht

# Beispiele Overengineering (1)

- Ein Dieselmotor für einen PKW hält 500.000 km – bis dahin möchten aber die meisten Kunden einen anderen Wagen fahren
  - Maximale Qualität (Haltbarkeit) muss also nicht immer das sein was der Kunde braucht
  - Man sollte sich überlegen, ob das den Motor nicht für viele Kunden unnötig teuer macht



# Beispiel: potentielles Overengineering (2)

- Aus einer (anderen) Vorlesung über QM:

- Welche Ziele wollen wir erreichen?
  - Fachliche Korrektheit
  - Robustheit
  - Wartbarkeit
  - Flexibilität
  - Benutzerfreundlichkeit
  - Durchsatz & Performance

# Beispiel: potentielles Overengineering (2)

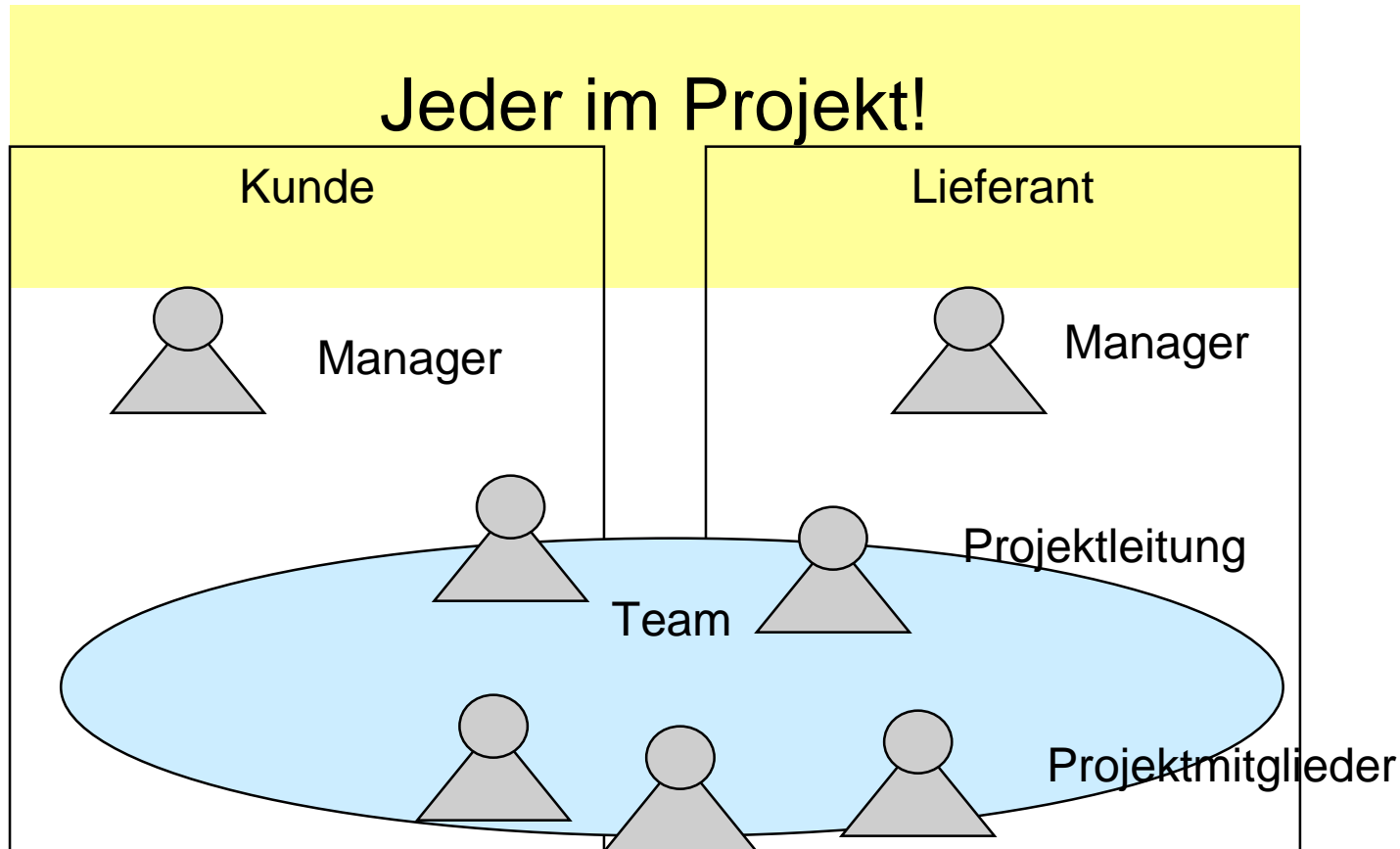
- Und was wollte der Kunde?
- Der Kunde wollte eine WebSite um mit Pilotkunden eine Geschäftsidee auszuprobieren
- Die sollte einigermaßen performant sein – besonders wartbar und flexibel muss sie nicht sein
- Also nicht immer alle nichtfunktionalen Eigenschaften optimieren, sondern jene sicherstellen, die der Kunde benötigt

# Nichfunktionale Eigenschaften

- Man sollte möglichst versuchen, über die Erfüllung nichtfunktionaler Eigenschaften
  - Nachzudenken
  - Und darüber klare, quantifizierte Vereinbarungen zu treffen

# Wer / Wann / Was / Wie

## Wer ist also für Qualität verantwortlich?





# Wer / Wann / Was / Wie

## Wer ist also für Qualität verantwortlich?



- Problem aber:
  - Wenn jeder für alles verantwortlich ist
  - Passiert am Ende gar nichts
- Daher ist es gut, spezielle Rollen im Projekt zu definieren
  - Qualitätsbeauftragter / -manager
  - Testmanager
- Andere müssen das Thema immer im Kopf haben
  - Projektleiter
  - Architekt

# Wer / Wann / Was / Wie

## Wann wird für Qualität gesorgt?

- Qualitätsmanagement greift von der Projektdefinition
  - Angebotserstellung, Review, Kick-Off, Einbeziehen der Beteiligten
- Über die einzelnen Projektphasen
  - Reviews von Spezifikationen, Prototypen, Test von Code
- Bis zum Projektende
  - Reflexionsworkshops – was können wir für das nächste mal lernen?

# Wer / Wann / Was / Wie Was wird durch QM nachgeprüft?



Die Eigenschaften, die  
für den Erfolg des  
Kunden wichtig sind

# Wer / Wann / Was / Wie

## Was wird durch QM nachgeprüft?

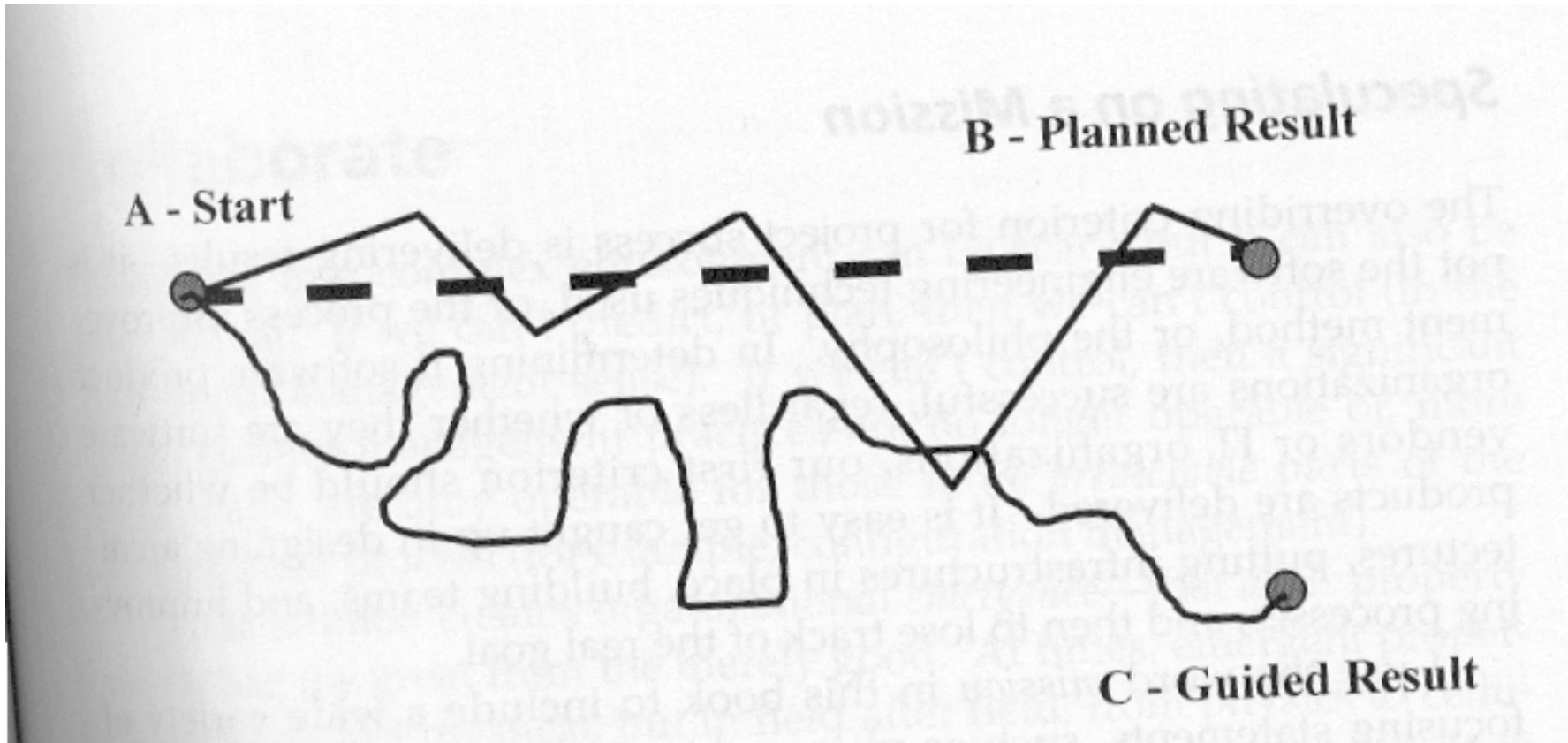
- Problem hier
  - Oft weiß der Kunde nicht einmal, was für ihn wichtig ist und / oder kann es nicht artikulieren
  - Meist gibt es deshalb keine klaren Vereinbarungen
- Damit muß man einbeziehen
  - Erfahrung
  - Stand der Technik
  - Gesetzliche Vorschriften
  - Good Practices aus anderen Projekten ...
- Das ist Beratungspflicht „guter Architekten“

# Wer / Wann / Was / Wie Wie erreicht man Qualität?



Durch ständiges  
Nachsteuern,  
um die Straße  
nicht zu verlassen

# Wir erinnern uns ... Das Ziel ändert sich ...



# QS – aber WIE?

## Man unterscheidet



- Konstruktive QS
  - Wie baue ich die Sachen so, dass gar nicht erst gravierende Probleme auftauchen?
  - Erhöhen der Wahrscheinlichkeit, dass man auf der Straße bleibt - Fahrertraining
  - Während der SW-Entwicklung qualitätssichernde Arbeitsschritte durchführen.
- Analytische QS
  - Wie stelle ich fest, ob ich noch auf der richtigen Straße bin, ob meine Software funktional richtig ist und ob sie die „nicht funktionalen“ Eigenschaften hat, die ich zugesagt habe?
  - Untersuchungen von fertigen Produkten oder Teilprodukten.

# Konstruktive QS Ausbildung



- Es ist klar, dass wer das Konzept „nichtfunktionaler Eigenschaften“ nicht kennt, diese schlecht tracken kann
- Gute Ausbildung der Teammitglieder ist daher eine der wirkungsvollsten QS Maßnahmen überhaupt



# Konstruktive QS – Ausbildung: Daher gut zu wissen

- Architekturen sowie Design und sonstige Patterns
  - Wenn etwas ganz neu ist, ist das immer verdächtig
- Regeln guten Software Designs
  - Design by Contract
  - Typprüfungen (statisch oder dynamisch)
  - Abhängigkeitsmanagement
- Codereviews
  - Punktuell (das wäre schon analytisch)
  - Oder permanent in der Form von Pair Programming
- Configuration- und Build Management

# Konstruktive QS – Ausbildung: Daher gut zu wissen



- Refactoring
  - Was ist mieser Code und was riecht schlecht
  - Und wie sorgt man dafür, dass es wieder besser riecht
- Defensive Programmierung
  - Wie programmiere ich gleich so, dass sich das System redundant auch selbst überprüft
    - Checks an Schnittstellen
    - Assertions
    - Contracts

# Konstruktive QS

## Prozeßbezogene Maßnahmen

- Termin- und Budgetplanung, Projekt-Controlling
  - Dienen vor allem der Überprüfung, ob man noch auf der Straße ist und genug Meilen gemacht hat ..
  - Wenn man abweicht ist das keine Tragödie – man muß nur wissen, wo man steht



# Konstruktive QS – Prozeßbezogenes Risikomanagement

- Projekte sind meist inhärent neu und damit auch inhärent riskant



- Man kann nicht immer sicher ausschließen, dass man von Risiken getroffen wird
- Aber man kann sich vorher schon mit seinem Auftraggeber darüber unterhalten und das permanent monitoren

# Konstruktive QS – Prozeßbezogenen Kommunikation



- Viele (wenn nicht die meisten) Projektprobleme werden verursacht durch
  - Aneinander Vorbeireden
  - Nicht miteinander reden
  - Nicht oft genug miteinander reden
- Kommunikationstraining ist also eine der effektivsten konstruktiven QS-Maßnahmen für Informatiker

# Analytische QS

- Untersuchungen der Vertragsgrundlagen
  - Angebotsreviews
- Untersuchungen der Spezifikation
  - Reviews der Spezifikationen, von Dokumenten
- Untersuchungen des fertigen Produktes
  - Testen