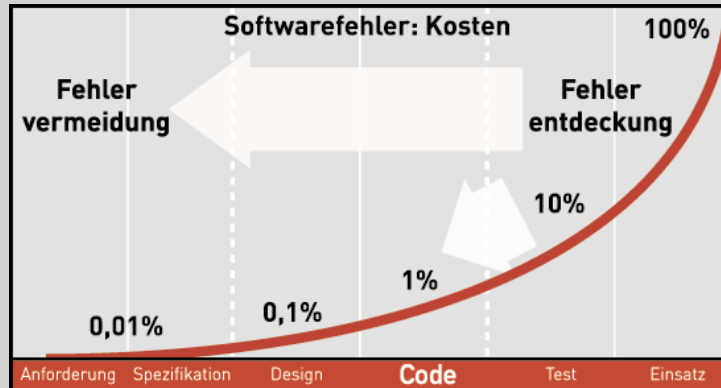


„Qualität von innen schaffen“



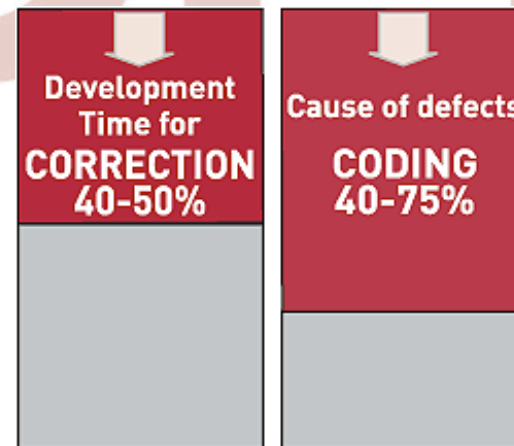
Der Trend zu Reviews  
Reinhard Haberfellner QiDO

# Erinnern Sie sich noch?



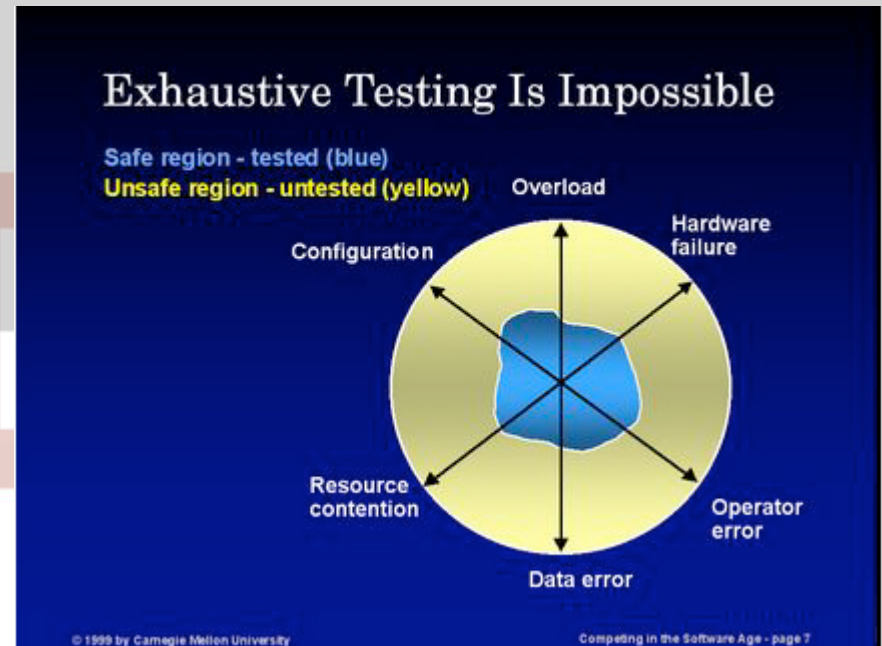
- Vermeiden statt ausbessern
- Testen ist viel teurer als Fehler im Keim zu ersticken
- Beim Kunden wird der Schaden unbegrenzt hoch

- Nicht die Erstellung sondern das Ausbessern braucht den Hauptteil der Zeit
- Von Anfang an da, wann wird es merkbar?



# Schon vergessen? Sicherheit kann man nicht „ertesten“

- Testen ist eine Auswahl an Prüfungen der unbedingt nötigen Voraussetzungen
- Sind die Anforderungen überhaupt eindeutig?
- Qualität entsteht ab dem ersten Tag, besonders wenn sie leistbar bleiben soll
- Abschlussprüfung bedeutet nicht erlernt
- Was macht der Nächste mit dem Projekt? Suchen?



Competing in software age  
*SEI Carnegie Mellon University*

# Inspektionen und Reviews gehören schon lange selbstverständlich zur Entwicklung

Welche Methode oder Prozess auch immer, agile oder generiert, um Inspektionen kommt man nie herum

- RUP Aktivität „Review Code“
- XP Pair Programming
- CMM Level 3 (Peer Reviews)
- ISO 9001 4.10 Inspection & Testing
- SCRUM Inspect & adapt



# Was soll reviewt werden?

- Die Spezifikationen auf Vollständigkeit , Eindeutigkeit, Sinnhaftigkeit im Zusammenhang mit bestehenden Systemen
- Guidelines
  - Die Einhaltung genereller Regeln, vereinbarter Konventionen (z.B. Namen) und „Hausordnungen“
- Standards
  - Sprachspezifische Regeln ( e.g. Java Standards @ Sun, Kompatibilitäten, Sicherheitsauflagen)
- Best Practices
  - Was sich bereits vielfach bewährt hat obwohl man es anders auch zum Laufen bringt

# Warum sind Reviews wichtig?



- Sie erhöhen das Vertrauen auf Richtigkeit
- Sie sorgen für Konsistenz
- Sie machen Abläufe konsistent und transparent
- Sie verbessern die Lesbarkeit, in Specs und Code
- Sie vermindern Unsicherheiten und zeigen frühe Abweichungen von einem gewählten Pfad
- Wenn sie fehlen, führt das zu vermehrter Wiederholung , Neuprogrammierung oder unnötiger Suchzeit

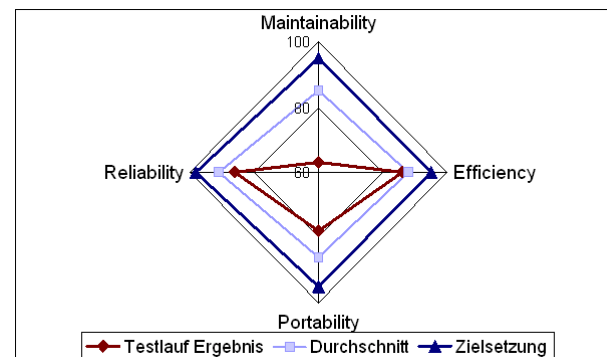
# Anforderungen an Standards

- Vereinbarungen sind von Anfang bekannt und abgestimmt
- Können überprüft werden und werden es auch
- Nehmen Bezug auf die Situation, Realität und den Zweck
- Sind dokumentiert und jederzeit nachvollziehbar
- Verursachen keinen Mehraufwand sondern vermeiden Folgeaufwand
- Sind anpassbar im Verlauf des Projektes



# Woher kommen die folgenden Erfahrungen?

- QiDO analysiert und reviewed seit vielen Jahren
- QiDO hat ungezählte Standards überprüft und mitentwickelt
- QiDO führt laufend Entwickler-Workshops durch
  - Mit Vertretern aus Teams von 3 bis über 100 Entwicklern
  - Teams aus Österreich, den CEE Ländern, Russland, China ...
- Ergebnisse von weit über 100 Projekten als Referenz
- Vom einfachen Interface über Web-Applikationen bis zur IN Plattform



# Die Unterstützung durch Tools

- Viele Werkzeuge auf dem Markt funktionieren wenn:
  - Sie umfassend installiert, adaptiert und dokumentiert sind
  - Sie einfach genug sind in Hochdruckphasen bedient zu werden
  - Sie konsistent über das ganze Team wirken und brauchbare Outputs liefern
  - Der nötige Eingabe-Aufwand realistisch ist
- Komplexe Tools liefern wertvolle Ergebnisse ..
  - Nach Wochen intensiver Vorbereitung
  - Solange genügend Ressourcen abgestellt sind dafür
  - Solange das Budget reicht, ab dann bleiben die Defekte drinnen bis zum Tag der Wahrheit

# Warum gibt es Zeitüberschreitungen?

- Wenn während der Realisierung entdeckt wird, dass diverse Anforderungen einander widersprechen oder fehlen
- Weil unerwartete Troubles mehr Zeit zur Beseitigung brauchen als erwartet
- Weil die Komplexität Testen dramatisch verlängert
- Weil das neue Team die alten Teile des Codes nicht versteht
- Vielleicht auch weil es 10 verschiedene Module gibt die die gleiche Aufgabe erledigen
- Weil sich schon wieder die Anforderungen geändert haben

# Warum verlangt Sicherheit nach innerer Qualität

- Testen dient vorwiegend dazu erwartete Funktionsabläufe zu verifizieren
- Hacken dient dazu das System in unerwartete Konditionen zu bringen
- Was darf das System tun und was nicht? (Damit Buffer Overflow möglich ist, müssen bestimmte Bedingungen erfüllt werden)
- Sicherheit erfordert grundsätzliche Aufmerksamkeit auf potentielle Fallen (z.B. richtiges "Error handling")
- ALLE Pfade sollten so weit als möglich überprüft sein, nicht nur die getesteten
- Menschliches Übersehen ersetzen durch automatisierte Analyse, das ist billiger und besser

# Warum ist die Autoindustrie Vorbild und Albtraum zugleich?

- In den 70ern wurde in Japan durch TQM eine Umwälzung der Vorgangsweise gestartet, statt Fehler finden und ausmerzen wird durch Prozessverbesserung die Möglichkeit Fehler zu machen minimiert.
- Durch den Anstieg des Anteils elektronischer Lösungen am Gesamtkunstwerk Auto löst jeder Softwarefehler fatale Auswirkungen aus , daher ein sprunghafter Anstieg von Rückholaktionen.
- Trotzdem wird die Integration vieler Funktionen in ein System vorangetrieben aus Kostengründen - und Autohersteller verweigern aus guten Gründen Remote Access per Funk.

# Warum mit Outsourcing der Anteil von Reviews steigt

- Eine laufende Kontrolle wird gerade bei auswärts vergebenen Aufgaben sehr geschätzt.
- Besonders fremde Dienstleister sind interessiert an aktiver Fehlervermeidung - der Reputation wegen.
  - Die meisten CMMI Level 4 und 5 Unternehmen sind heute .... in Indien.
- Man muss intern und extern beweisen, dass die neue Outsourcing Strategie gute Ergebnisse liefert, daher investiert man im voraus in den Erfolg und nicht das vielfache im Nachhinein in den Fix.



DIE RISKANTE ABKÜRZUNG ÜBER DIE "MAMMUTWIESE".

# Ein Beispiel aus der Praxis



- Deutsches High Tech Unternehmen führt vor zwei Jahren gemeinsam mit einer Universität ein intensives Review-Programm ein und analysiert seither permanent beim Check-In.
- Erweiterungen eines Programmteiles die bisher für 5 Tage geschätzt wurde ( von guten Schätzern!) dauern plötzlich nur mehr 2 Tage
- „Wir konnten alles diesmal schneller finden, weil der Code viel lesbarer ist als früher“.
- „Die Tests waren plötzlich viel schneller erledigt, als bei der letzten Änderung“

# Die 7 Best Practices für sicherer Software

von Gary McGraw, CEO Cigital

Gereiht nach Effizienz und Wichtigkeit:

- Platz 1: Static Analysis
  - Code Review ist absolut notwendige Praxis , wenn gleich per se nicht ausreichend sichere Applikationen zu garantieren
- Bonuspunkt: Für externe Analyse, also durchgeführt von Leuten die nicht zum Kernteam der Entwicklung oder Architektur gehören.



*Code Reviews are important but underutilised*

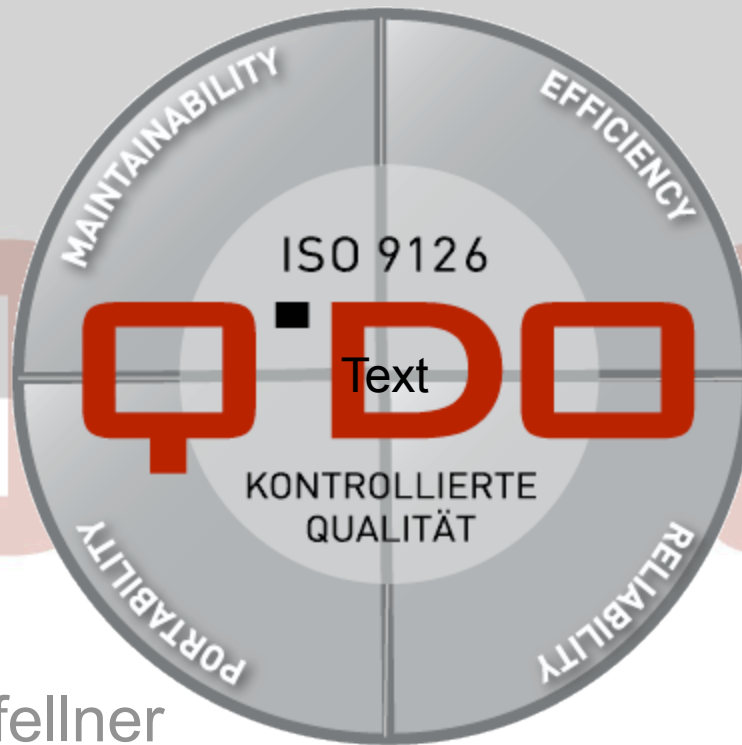
Paul Herzlich , Ovum UK

# Automatisierte Reviews



- Softwarequalität wird quantifizierbar
- Zeitersparnis durch Abläufe im Hintergrund
- „Training on the job“ durch regelmässige Anwendung
- Konsistente, objektive und nachvollziehbare Prüfung
- Trends fallen frühzeitig auf ( zB Komplexität steigt über gewisse Ausmasse an = Testen unmöglich)
- Benchmarking mit anderen Abteilungen, Firmen, Branchen

# Weitere Fragen ?



Vielen Dank  
Reinhard Haberfellner

Kostenloser Test gefällig?

Mail to: [office@qido.at](mailto:office@qido.at)

Tel

+43 (1) 968 2459

Web

[www.qido.at](http://www.qido.at)



QUALITY AT THE SOURCE