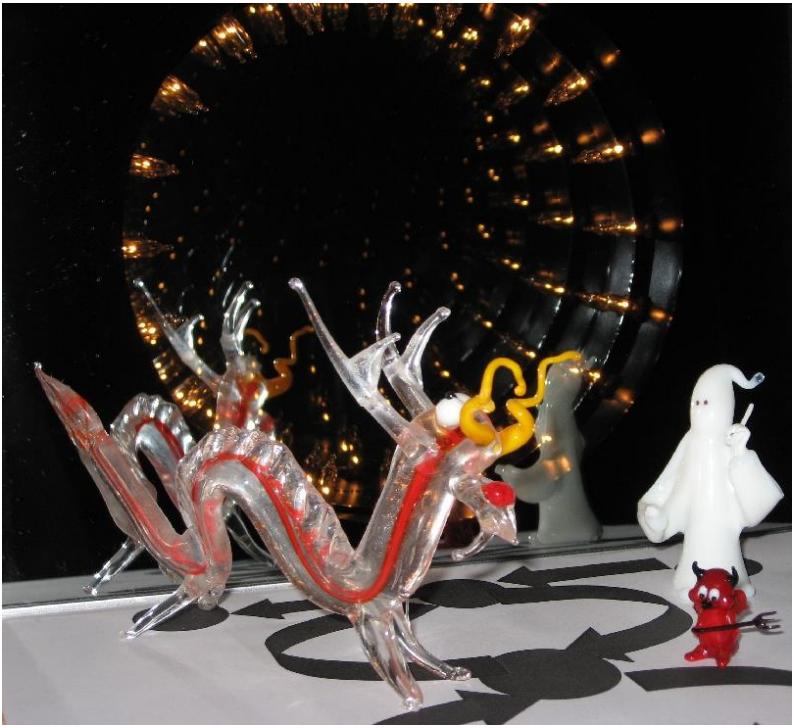


Der Drache der Fehlerwahrscheinlichkeit, der Fehlerteufel und das Gespenst der schlechten Qualität vor dem Tor zum Universum der Softwareentwicklung

Auf der Route 66 durch das Wunderland der Softwareentwicklung

von Wolfgang Ulbrich

Impressum

© 2013 Gesellschaft für Softwarearchäologie

Am Flutgraben 5

99955 Bad Tennstedt

Alle Rechte vorbehalten

Homepage http://www.soft-arche.de

Fotos: © Wolfgang Ulbrich

Bilder (in den Tarot-Karten): © Xenia Miedreich

Kalenderbilder: © Wolfgang Ulbrich

Satz und Layout: Wolfgang Ulbrich

Covergestaltung: Wolfgang Ulbrich

Inhaltsverzeichnis

Impressum 2

Vorwort 6

Station 1: Am Tor der Kommunikation 8

Kommunikation und Softwarequalität 8

Traumzeit: JOFRI's Welt in Gedichten.....17

Station 2: Auf dem Punkt der Selbstbetrachtung 30

Optimierung der Prozesse und/oder/mit/ohne persönliches QM des Mitarbeiters.....30

Traumzeit: Disput über Qualität und Entropie von Softwaresystemen.....35

Station 3: Im Camp der Nationen 48

Projekt- und Qualitätsmanagement in einem multinationalen Team.....48

Traumzeit: Software-Metrie für Dummies.....57

Station 4: An der Psycho-Falle 70

Projektmanagement zwischen Psychologie und Tools.....70

Berücksichtigung psychologischer Aspekte bei der Erhaltung der Codequalität in Wartungsprojekten.....87

Traumzeit: Psyche versus Ratio in der Softwareentwicklung 91

Station 5: Über den Pass der Abnahmekontrolle 104

Spezifika des Projektmanagements bei der fachlichen Abnahme 104

Traumzeit: Nicht ganz ernst gemeinte Erkenntnisse aus der fachlichen Abnahme des Anwendungssystems XYZ.....117

Station 6: In der Schlucht der Risiken 126

Minimalanforderungen an QM ermitteln und einhalten...126

QM-Check einer Firma bzw. einer Struktureinheit (Inhalte, Umfang, mögliche Ergebnisse, Probleme).....133

Softwarerisiken = Risiken beim Test ?.....137

Risikooptimierte Qualitätssicherungstrategien bei Migrationsprojekten 143

Traumzeit: Als Qualitätsmanager (QM) und Softwaretester auf der Couch bei ELIZA 150

Station 7: Am Orakelstein 163

Qualitätsorakel oder Möglichkeiten einer Qualitätsprognose aus unvollkommenen Kenntnissen.....163

Traumzeit: Die großen Arkana des Software-Quality-Tarots 167

Station 8: Im Wald der Gesichter 181

Die 5 Gesichter des Computers 181

Traumzeit: Installationen in SoftArt oder freie Assoziationen über die Kunst der Softwareentwicklung..212

Station 9: Der Friedhof der Untoten 225

Der (Irr-)Garten der Programmiermuster und Ablaufkonstrukte 225

Quellen / Literatur 238

Kurzbiographie des Autors 242

Vorwort

Nach 40 Jahren Tätigkeit in der IT als Softwareentwickler, Projektmanager, Qualitätsmanager etc. bei diversen Firmen gab ich zum 31.8.2011 meine festangestellte Tätigkeit bei einer Softwarefirma auf.
Seitdem arbeite ich freiberuflich beratend zum Softwarequalitätsmanagement.
Obwohl ich alle meine Projekte erfolgreich abschließen konnte (Kriterium: produktiv im Einsatz) waren für mich die Ergebnisse z.B. bei der Beratung von Entwicklern beim Test für mich nicht wirklich befriedigend, da auch die heutigen Entwickler im Prinzip die gleichen Fehler machen wie zu Beginn meiner Berufstätigkeit.
Offenbar spielen hier im Verlaufe der Evolution erworbene psychologische Eigenheiten des ‚homo sapiens' eine große Rolle. Dazu kommt, dass

- ein vollständiger Test eines Programms oder einer Anwendung praktisch nicht möglich ist,
- abgeleitet aus dem Unvollständigkeitssatz des berühmten Mathematikers Gödel die Richtigkeit von Programmen bei hinreichender Komplexität im vorliegenden System nicht beweisbar ist,
- demzufolge auch heute noch Intuition für den Test und die Konstruktion und Auswahl von Testfällen gefragt ist.

Zwei Sprüche geben meiner Ansicht nach die Situation treffend wieder

It's not enough to do your best, you must know what to do and then do your best
(Dr. William Edwards Deming**)**

A fool with a tool is still a fool
(unbekannt)

2012 beschloss ich, anlässlich des 10-jährigen Bestehens der Gesellschaft für Softwarearchäologie im Jahr 2013, die von mir gesammelten Erfahrungen zum Projekt- und Qualitätsmanagement, basierend u.a. auf meinen Vorträgen auf internationalen Tagungen zum Projektmanagement und zur Software-

qualität und den von mir erstellten Kalendern der Gesellschaft für Softwarearchäologie, in Form einer virtuellen Reise – auf der Route 66 durch das Wunderland der Softwareentwicklung – dem geneigten Leser zugänglich zu machen.
Dabei beschränke ich mich auf die Elemente der Softwareentwicklung, die weitgehend technologieunabhängig sind und zu denen ich meiner Meinung nach auch heute noch etwas sagen kann.
Die diversen Programmiersprachen und Entwicklungsumgebungen, mit denen ich in meinem Berufsleben arbeitete, werden in diesem Buch nicht besprochen.
Ebenso finden meine Erfahrungen mit Suchmaschinen und der künstlichen Intelligenz keine explizite Berücksichtigung.
Das Buch behandelt in Form von Reisestationen (Kapiteln) jeweils einzelne Probleme des Projekt- und Qualitätsmanagements in der Softwareentwicklung in seriöser Form. In der ‚Traumzeit‘ vor dem Aufbruch zur nächsten Reisestation kommen mit einem ‚Augenzwinkern‘ andere ‚Experten‘ wie Mr. Spock und das Programm ELIZA zu Wort.
Ich bedanke mich bei den Autoren, deren Werke mir auf meiner Route 66 im Berufsleben treue Begleiter waren. Stellvertretend für viele seien hier Norbert Wiener (der Vater der Kybernetik), Joseph Weizenbaum und Tom de Marco genannt.
Da ich mit diesem Buch keine irgendwie geartete Qualifikation anstrebe und auch keine kommerzielle Verwertung beabsichtige, halte ich den Quellennachweis und die Kennzeichnung von Zitaten in Grenzen. Für dabei übersehene Stellen entschuldige ich mich vorab.
Desgleichen bedanke ich mich bei Fach-KollegInnen u.a. auch aus meiner eigenen Familie, die mir stets kritische und hilfreiche Diskussionspartner waren.
Ganz besonders möchte ich aber meiner Frau Brigitte danken, die stets Verständnis für meine berufliche Arbeit zeigte und mir oft den ‚Rücken‘ freihielt.

Worte sind nicht alles
aber Worte sind alles, was ich habe.

Station 1: Am Tor der Kommunikation

Kommunikation zwischen Entwickler und Auftraggeber ist die Grundvoraussetzung für erfolgreiche Softwareentwicklung und schwieriger als man im Allgemeinen denkt.

Kommunikation und Softwarequalität

These:
Die Erstellung qualitativ hochwertiger Software lebt wesentlich von der Beherrschung der dem Softwareentwicklungsprozess immanenten Kommunikationsprobleme.

Im Folgenden werden für ausgewählte Schwerpunkte der Softwareentwicklung einige dieser Kommunikationsprobleme benannt und pragmatische Lösungs-/Vermeidungs-Strategien diskutiert sowie auf der Basis praktischer Erfahrungen bewertet. Bei der Installation und erfolgreichen Anwendung von Qualitätsmanagementsystemen stellt die Kommunikation einen nicht zu vernachlässigenden Erfolgsfaktor dar.

1. angemessene (semantisch und syntaktisch korrekte und vollständige sowie verständliche) Modellierung des zu lösenden ‘Reale-Welt’-Problems

Bei der Softwareentwicklung wird in der Regel ein definierter Ausgangspunkt gesetzt, auf den sich letztendlich alle Bewertungen des Prozessergebnisses beziehen. Diese Bewertungen beziehen sich z.B. sowohl informell auf die Erfüllung der Nutzererwartungen als auch auf die Erfüllung vertraglich vereinbarter Entwicklungsziele mit möglichen Folgen für Bezahlung, Regress- oder Gewährleistungsansprüche. Als solcher definierter Ausgangspunkt ist die ‘reale Welt’ selbst untauglich, da sie selbst bzw. mindestens ihre Wahrnehmung und Interpretation in ständiger Veränderung begriffen ist. Ein oft gewählter

Weg ist deshalb die Modellierung des zu lösenden ‘Reale-Welt’-Problems und die Festschreibung des Modells als Ausgangspunkt für die weitere Softwareentwicklung. Dies bedeutet nicht, dass das Modell während des Entwicklungsprozesses nicht mehr veränderbar wäre, aber solche Veränderungen müssen ebenfalls definiert erfolgen und vor allen Dingen festgehalten werden.

Mögliche Ergebnisse der Modellierung sind im Allgemeinen eine Spezifikation und/oder ein Prototyp der Lösung. Eine Bewertung dieser Möglichkeiten wird im Rahmen dieses Kapitels nicht vorgenommen. Ebenfalls wird in diesem Kapitel nicht auf Probleme möglicher menschlicher Erkenntnisgrenzen hinsichtlich sehr komplexer Sachverhalte der ‘realen Welt’ eingegangen.

Folgende kommunikativen Probleme treten bei der Modellentwicklung u.a. auf:

- Entwickler und spätere Nutzer der Software sprechen eine unterschiedliche ‘Sprache’.
- Gleicher Begriffe können in beiden Sprachräumen unterschiedliche Bedeutungen haben.
- Verwendete Begriffe können im jeweiligen Sprachraum mehrdeutig sein.
- Es existieren unterschiedliche Erwartungen an die Modelldarstellung bei Entwickler und Nutzer:
  - Entwickler: Exaktheit(auch im Detail) und für die weiteren Entwicklungsprozesse geeignete modellhafte und ggf. graphische Darstellung
  - Nutzer: Verständlichkeit, Anschaulichkeit bei Verzicht auf ‘unwesentliche(!?)’ Details, d.h. seiner Begriffs- und Beschreibungswelt angenähert
- Man muss qualifizierte spätere Nutzer, d.h. Nutzer mit Kompetenz, Sachkunde und Bereitschaft zur gründlichen Analyse, finden, die das erarbeitete Modell beurteilen.

Damit nicht schon bei der Modellierung die Grundlage für ein unbefriedigendes Ergebnis des Softwareentwicklungsprozesses gelegt wird, muss man versuchen die beschriebenen Probleme zu vermeiden bzw. ihnen geeignet begegnen.

Dafür gibt es u.a. folgende pragmatischen Möglichkeiten

- Die Entwickler eignen sich die wesentlichen fachlichen Elemente des Nutzersprachschatzes an und verwenden sie auch.
- Es erfolgt eine exakte Definition von Begriffen im Rahmen der Modellierung und deren Beschreibung in einem Projektlexikon.
- Die bei der Modellierung mitwirkenden Nutzer werden mit den verwendeten Modellierungstechniken vertraut gemacht.
- Es wird eine Ergänzung von Text durch nichttextuelle Elemente(Bilder, Grafik etc.) bzw. umgekehrt die Ergänzung verwendeter Diagrammtechniken durch textuelle Erläuterungen vorgenommen.
- Ein dynamisches Modell(Prototyp) zum simulierten Umgang mit den wesentlichen Elementen der späteren Lösung wird den Nutzern zur Verfügung gestellt.

2. Umsetzung(Verfeinerung) des Modells je nach vorgeschriebener Vorgehensweise und verfügbarer Entwicklungsumgebung

Dies geschieht in der Praxis in Schritten, Stufen, Ebenen, Phasen je nach begrifflicher Benennung. Im Folgenden wird rein pragmatisch der Begriff ‘Phase’ verwendet.

Zwischen jeweils zwei Phasen A und B tauchen analoge Probleme zum ersten Schritt der Modellierung auf:

- Die Entwickler können in den Phasen unterschiedlich sein. Die Firma X hat beispielsweise das ‘Fachkonzept’ erstellt und von der Firma Y soll jetzt ein ‘DV-Konzept’ erarbeitet

werden. Damit ist eine unterschiedliche ‘Sprache’ in Phase A und B mit Auswirkungen analog zu 1 vorhanden.

- Die Beschreibung des Modells ist i.a. nicht eindeutig, da selten mit dem Mitteln einer streng formellen Spezifikation gearbeitet wird (s. die unter 1 genannten Probleme). Die vollständige Beschreibung bestünde in der Regel nicht nur aus den dokumentierten Ergebnissen sondern auch aus dem bei den Modellentwicklern angesammelten Hintergrundwissen (informelles Teamgedächtnis). Es gibt immer wieder implizite Prämissen des Modells, die nirgends dokumentiert sind. Diese Probleme zeigen sich besonders gravierend bei der Vergabe von Softwareentwicklungsaufträgen in ‘Billiglohn’-Länder.
- Es muss eine Bewertung des Ergebnisses von Phase B bezüglich der Vorgaben durch das Ergebnis von Phase A bzw. noch weiter zurückliegender Phasen durch spätere Nutzer bzw. die Entwickler in Phase B erfolgen.
- Es können sich Fehler aus vorherigen Phasen fortpflanzen.

Folgende Maßnahmen zur Lösung bzw. Vermeidung der Probleme können ergriffen werden:

- Der wesentliche fachliche Sprachschatz der Phasen wird abgestimmt.
- Das Projektlexikon wird um die exakte Definition von Begriffen im Rahmen der aktuellen Phase ergänzt. Wenn bisher kein Projektlexikon existierte, so wird es angelegt. Unbedingt vermieden werden sollte eine Änderung von schon definierten Begriffen(Konsistenzprobleme!).
- Text(auch Programmcode) sollte sofern möglich durch nichttextuelle Elemente ergänzt werden.
- Sofern noch nicht vorhanden wird ein dynamisches Modell(Prototyp) zum simulierten Umgang mit den wesentlichen Elementen der späteren Lösung den Nutzern zur Verfügung gestellt bzw. das vorhandene weiterentwickelt und verfeinert.

- Die Entwicklung sollte so wenige Phasen wie möglich durchlaufen.
- Mindestens ein Entwickler/begleitender Nutzer sollte durch alle Phasen beteiligt sein, um das ‘informelle Teamgedächtnis’ nutzen zu können.
- Eine Prüfung von Phasenergebnissen sollte durch mehrere auch unabhängige ‘Experten’ (in Reviews u. ä.) erfolgen.

3. Softwareentwicklung im Team innerhalb einer Phase

Auch in einem gut organisierten und optimal zusammengesetzten Team treten Kommunikationsprobleme auf. Der entscheidende Unterschied zwischen erfolgreichen und weniger erfolgreichen Teams besteht u.a. darin, ob man diese Probleme erkennt und wie man mit ihnen umgeht.

Die häufigsten Kommunikationsprobleme sind:

- Die Entwickler sprechen eine unterschiedliche ‘Sprache’.
- Es gibt im Team keine einheitliche Basis für die Entwicklung(Werkzeuge, Standards u.ä.).
- Es findet kein ausreichender Austausch über Probleme und/oder Ergebnisse im Team(Verbindung im Team) statt.
- Im Team existiert keine ausreichend saubere Aufgabenteilung(Abgrenzung im Team).

Was kann man empfehlen, um die genannten Probleme zu vermeiden bzw. zu lösen?

- Auch im Team selbst sollte eine Abstimmung des wesentlichen Sprachschatzes stattfinden.
- Es ist notwendig Standards und Umgebungsbedingungen exakt zu definieren (so viel wie nötig, nicht so viel wie möglich!) und deren Einhaltung kontrollfähig zu machen. Dabei ist der Selbstkontrolle unbedingt der Vorrang vor der Fremdkontrolle zugeben. Wenn es ein Werkzeug zur statischen Analyse in der Entwicklungsumgebung gibt, so muss

dies allen Entwicklern zur Verfügung stehen. Es muss auch bekannt sein, welche Kontrollen vom Projektmanagement für notwendig erachtet und deshalb durchgeführt werden.

- Ein regelmäßiger Austausch über Probleme und/oder Ergebnisse im Team(offiziell und informell) ist das Lebenselixier erfolgreicher Teamarbeit.
- Ein klares und systematisches Projektmanagement schafft Berechenbarkeit.
- Die gegenseitige Prüfung von Ergebnissen (Reviews u. ä.) ist in einem erfolgreichen Team selbstverständlich. Schuldzuweisungen sind ungeeignete Mittel. In der Sache eventuell harte aber im Ton moderate, auf keinen Fall persönlich verletzende und diffamierende Auseinandersetzung nützt mehr als das ‘unter den Teppich kehren’ von Problemen.
- Wenn irgend möglich, sollten sich die Teammitglieder über die Prinzipien erfolgreicher Teamarbeit und Kommunikation im Team informieren können, um die in einem Team bzw. generell zwischen Menschen wirkenden Kommunikationsmechanismen verstehen zu lernen.

4. Erkenntnis-/Realisierungs-Probleme beim einzelnen Entwickler

Letztendlich sind aber bei aller Teamarbeit durch den einzelnen Entwickler einzelne Aufgaben zu lösen. Dabei stößt er auch wieder auf Kommunikationsprobleme, die die Kommunikation mit sich selbst und die richtige oder vielleicht besser gesagt, der Aufgabe angemessene Interpretation der Einordnung seiner Lösung in den Gesamtzusammenhang beinhalten. Im Detail treten dabei auf:

- Der Entwickler macht Vorgehensfehler und gerät in ‘Denkfallen’ auf Grund seiner evolutionär bedingten und/oder durch Bildung angelernten Lösungsstrategien bzw. Lösungstechniken.

- Der Entwickler betreibt keine oder nur eine ungenügende Analyse der eigenen Fehler.
- Er lernt nicht oder nicht genügend oder nicht richtig aus seinen Fehlern.
- Er eignet sich nur ungern und mangelhaft ‘erfolgreiche Techniken/best practices’ an.’

Es ist schwierig, zu der Lösung dieser Probleme bei Achtung der Individualität und Persönlichkeit des einzelnen Entwicklers etwas zu tun, aber es ist nicht unmöglich sondern es ist notwendig und auch erfolgversprechend

Man kann unter anderem

- die Entwickler über die Rolle psychischer und Erkenntnisprozesse bei der Softwareentwicklung aufklären,
- die Anwendung von Analysetechniken und Metriken durch den Entwickler fördern und in bestimmtem Maße verlangen,
- moralisch und/oder materiell die Weitergabe und Aneignung ‘erfolgreicher Techniken / best practices’ im Team stimulieren

5. Unterschiedliche Sichtweisen auf Softwarequalität und Softwareentwicklung

Jeder erwartet zumindest von der von ihm selbst benutzten Software oder von der Software, für die man irgendwie verantwortlich ist, ‘gute Qualität’. Dabei ist das Qualitätsverständnis der Beteiligten oft so unterschiedlich wie es unterschiedlicher nicht sein könnte. Dies gilt nicht nur für das Produkt(die fertige Software) sondern auch für den Prozess, der dieses hervorbringt und ebenfalls für das Qualitätsmanagement des Softwareentwicklungsprozesses.

Für ein erfolgreiches Projekt bzw. für eine erfolgreiche Softwareentwicklung im Unternehmen überhaupt ist die Kommunikation über und die Kenntnis der unterschiedlichen Meinun-

gen sowie die Erzielung eines möglichst umfassenden Konsens über die Schwerpunkte ein wesentlicher Faktor der Verbesserung. Dazu muss man die unterschiedlichen Sichtweisen erst einmal kennen.

Welche Möglichkeiten gibt es hierfür?

- Ein **Self-Assessment der Mitarbeiter** unter strikter Wahrung der Anonymität zeigt die subjektive Wahrnehmung der Qualitätssituation im Projekt/Unternehmen durch die Mitarbeiter.
- Ein **Self-Assessment des Managements** ebenfalls unter strikter Wahrung der Anonymität zeigt die subjektive Wahrnehmung der Qualitätssituation im Projekt/Unternehmen durch das Management.
- Ein **objektivierte Analyse** z.B. durch Assessments nach Bootstrap und/oder Nutzung von Controlling-Methoden und/oder Audits zeigt den Stand bezüglich formulierter Ansprüche z.B. Reifestufen o.ä. und ermöglicht damit die Einordnung des eigenen Prozesses gegenüber den Mitbewerbern am Markt.
- Die **Befragung der Kunden/Nutzer** zeigt i.a. die Schwachpunkte des eigenen Prozesses(z.B. Änderungsmanagement, Support und Verhalten gegenüber dem Kunden/Nutzer allgemein) aus Kunden-/Nutzersicht. Seltener werden positive Eigenschaften hervorgehoben.

Eine in intensiver Kommunikation auf der Basis dieser vier Grundmeinungen und einem gemeinsam erarbeiteten Konsens formulierte Verbesserungsstrategie muss wiederum intensiv an die o.g. Gruppen übermittelt(kommuniziert) werden. Sie muss vor allem die Einleitung akzeptierbarer, überschaubarer und pragmatischer Maßnahmen zum Inhalt haben, damit sie gelebt und zum Erfolg geführt werden kann. Gleichzeitig muss die Kontrolle der Wirksamkeit der Veränderungen gesichert werden, um keine Demotivation der Beteiligten nach dem Motto ‘es ändert sich ja doch nichts’ zuzulassen.

6. Schlussbemerkung

In vielen Standardwerken zur Softwarequalität finden sich einzelne Aspekte der in diesem Beitrag behandelten Kommunikationsprobleme. Leider beschränken sich die meisten Autoren auf eine mehr technische Sicht der Probleme. Software wird aber (noch!) von Menschen gemacht. Deshalb ist die Berücksichtigung der Möglichkeiten und Grenzen des Menschen nicht zuletzt in kommunikativer Hinsicht m.E. nach einer der wesentlichen Möglichkeiten zur Verbesserung der Softwarequalität. In diesem Beitrag konnten viele Dinge nur angerissen werden, die verdienten, wesentlich ausführlicher besprochen zu werden, wofür durch den Beitrag vielleicht eine Anregung gegeben wurde.

Traumzeit: JOFRI's Welt in Gedichten

In der Traumzeit wanderte ich durch das Universum der künstlichen Intelligenz. Dabei stieß ich plötzlich auf das Programm JOFRI von einem unbekannten Autor. JOFRI erwartet zwei Eingaben: eine Liste von Stichworten und eine Stilrichtung bzw. einen/eine VorbildautorIn. Die Ausgabe ist dann ein von JOFRI erstelltes ‚Gedicht'. Damit erinnert JOFRI an das bekannte Programm ELIZA von Weizenbaum, das ebenfalls seinem Benutzer ‚intelligentes' Verhalten suggerierte.
Nach meiner Vermutung könnte JOFRI ein Kürzel aus den Namen der beiden größten deutschen Dichter sein: **Jo**hann Wolfgang Goethe / **Fri**edrich Schiller.
Die Bilder symbolisieren die chinesischen Tierkreiszeichen.
Im Traum und auch danach ist mir der Sinn dieser Darstellung nicht aufgegangen. Aber das ist halt Traumzeit.

Lyrik ‚verfasst' von dem
‚artificial intelligence' Programm JOFRI

im Zeichen der Ratte

Liebe LILO,

die Sonne scheint und ich IMAP in IMFE.

Zum Glück muss ich nicht MALOchen in LIFI.

Ich mache MAAP und nicht MALR.

Dabei denke ich an IMHOtep und ein Vögelchen macht leise IMPT.

Ach wäre das Leben doch LILaRot,

dann gäbe es schönere Status für die Köpfe von CIKLAS.

Viele Grüße

JOFRI

Eingabe: Statuswerte einer IT-Anwendung
Vorbild: DADA

Das Projekt geht zu Ende

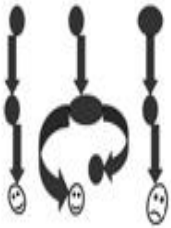
Wieder schwand ein Projekterfolg dahin

und der Projektmanager trägt immer noch

das Gewand der Selbstgefälligkeit.

im Zeichen des Büffels/Stiers

Eingabe: Projekt, kein Erfolg, keine Selbstkritik
Vorbild: Haiku



Der weise Guru

im Zeichen des Tigers

Der Guru hörte stets mit Ruh,

wenn zwei sich disputierten, zu.

Mal stritten sich der DBA und der Entwickler,

was Oracle zuerst erschaffen habe,

Ob erst die Datenbank oder erst das SQL.

„Die Datenbank!" - schrie der DBA

„Das ist doch klar wie Brei!"

Der Entwickler brüllt: „Das SQL, wobei ich bleibe;

wer's nicht begreift, hat kein Gehirn im Leibe!"

Ja - denkt der Guru - so bin ich!

Der Weise schweigt und räuspert sich!

Eingabe: Guru, Oracle, DB, SQL, Streit
Vorbild: Wilhelm Busch

Der Projektmanager

Ein Projektmanager kam geflogen,

setzte sich auf einen Ast und machte 'piep'.

Wolken kamen herangezogen,

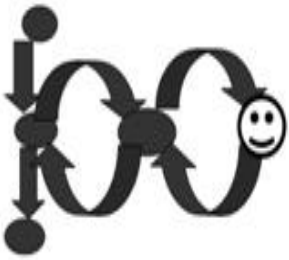
das Wetter hatte den Manager gar nicht lieb.

Wo ist der große Schirm geblieben?

Da hat's den Manager wieder fortgetrieben.

Piep!

Eingabe: Projektmanager, Wetter, Piep
Vorbild: volkstümlich



im Zeichen des Hasen

im Zeichen des Drachen

Könnte man fürs Programmieren sterben?

Könnte man fürs Programmieren sterben

wär ich längst kalt und tot.

Sollte es ein Feuer heißen

wär ich längst Asche und Kot.

Doch ist es kein Tod zu nennen

woher fühl ich solche Schmerzen?

Und ist es kein brennend Feuer

was kocht so in meinem Herzen?

(Programmierer nachts um 1 Uhr)

Eingabe: programmieren, sterben
Vorbild: Hans Aßmann Freiherr von Abschatz

im Zeichen der Schlange

An das fehleranfällige System

O mein System du spröder Schatz,

Drückt dich zu fest mein Arm?

Für Abstürze ist das ein schlechter Platz

Und Fehlermeldungen halten nicht warm.

Sechs Nächte lag ich in Sumpf und Moor

Und hab' um dich gewacht,

Doch weicher, bei Sankt Georg ich's schwor,

Schlaf' ich die siebente Nacht!

Eingabe: System, Absturz, bewachter Lauf
Vorbild: Willibald Alexis

im Zeichen des Pferds

Romantische SELECT's

Zwei SELECTs wandeln durch die Nacht.

Durch ihres Rumpfs verengten Schacht

fließt die Datensicht

still und heiter

in diese ihre

Auswahl

u. s.

w.

Eingabe: SELECT, Auswahl, Datensicht
Vorbild: Figurengedicht von Christian Morgenstern

im Zeichen des Schafs/Widders

Wat is ene Suchmaschin ?

Da stelle mer uns mal janz dumm und sachn:

En Suchmaschin,

dat is ene jroße schwarze Raum,

der hat hinten und vorn e Loch.

Dat eine Loch, da kommt de Suchanfrage rin

und dat andere,

dat ham mer später.

Eingabe: was ist eine Suchmaschine?
Vorbild: ‚Feuerzangenbowle'

Programmie lauf...

Zwei Programme liefen durch die Daten,

das eine schräg, das andere in der Geraden

und keines in der Mitte,

man sieht, es fehlt das dritte.

Eingabe: Programme, Daten
Vorbild: Klapphornvers

im Zeichen des Affen

Testers Abendlied

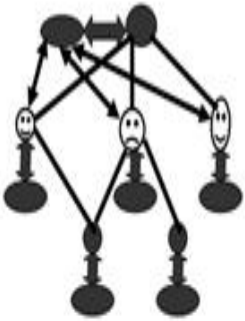
Vor seinem Laptop ruhig im Schatten sitzt

Der Entwickler, dem Genügsamen reicht ein Wert.

Häßlich erscheint dem Tester im

umfangreichen Log die Fehlermeldung.

Eingabe: Entwickler, Tester, Laptop, Log, Fehlermeldung
Vorbild: Alkäische Strophe nach Hölderlin



im Zeichen des Hahns

Programmierers Nachtgedicht

Nachts

noch programmieren

macht schlechten Teint

darum schalt doch endlich

aus

Eingabe: programmieren, Teint, ausschalten
Vorbild: Elfchen

im Zeichen des Hundes

im Zeichen des Schweins

Zum Jahreswechsel

von einem Qualitätsmanager

Will das Glück nach seinem Sinn
dir etwas Gutes schenken,
sage dank und nimm es hin
diesmal ohne viel Bedenken.
Jede gute Gabe sei begrüßt,
doch vor allen Dingen
Das, worum du dich bemühst
möge dir fehlerfrei gelingen.

Eingabe: Glück, schenken
Vorbild: Wilhelm Busch

Station 2: Auf dem Punkt der Selbstbetrachtung

Erfolgreiches Qualitätsmanagement erfordert nicht nur das Vertrauen auf optimierte Prozesse sondern die selbstkritische Analyse des eigenen Verhaltens.

Optimierung der Prozesse und/oder/mit/ohne persönliches QM des Mitarbeiters

Im Folgenden werden die Probleme bei der Veränderung von persönlichen Schaffensprozessen im Zusammenhang mit Prozessänderungen zur Verbesserung der Software-Qualität untersucht. Dabei werden psychologische und Mitarbeiterführungs-Aspekte sowie die Möglichkeiten und Grenzen helfender Techniken für den Mitarbeiter betrachtet.

1. Probleme des Managements bei der Optimierung der Prozesse aus Mitarbeitersicht

Im Zeichen der Globalisierung und der Betonung des ‘shareholder value’ werden in vielen Firmen auch die softwareerstellenden Bereiche und deren Prozesse auf den Prüfstand gestellt. Es geht darum, mit weniger Aufwand gleiche und qualitativ bessere Leistung zu erbringen. Mehr oder weniger laufen alle empfohlenen Wege zu diesem Ziel auf eine Optimierung oder Veränderung der Prozesse der Softwareerstellung eventuell mit zusätzlichem Tooleinsatz hinaus. Viel Geld wird in Beratung investiert, damit mit hoher Sicherheit die gewünschten Effekte eintreten. Dabei werden oft einige der folgenden Punkte übersehen:

- Hinter einer Prozessoptimierung /-änderung muss das Management von Beginn an stehen, sich persönlich dazu bekennen und die Optimierung/Änderung auch ständig begleiten.

- Neben den Nutzenzielen für die Firma sollten auch Vorteile für die Mitarbeiter angestrebt und deutlich gemacht werden.
- Das Maß der Änderungen darf nur innerhalb bestimmter Grenzen liegen, wenn sie von der breiten Masse der Mitarbeiter akzeptiert werden soll. Hierzu kann arbeitspsychologische Beratung nützlich sein.
- Die Optimierung von Prozessen schafft nur die Rahmenbedingungen für die Verbesserung der Softwarequalität. Auch verbesserte Prozesse erhöhen allenfalls die Wahrscheinlichkeit für bessere Prozessergebnisse.

2. Probleme der Mitarbeiter mit den optimierten Prozessen

Prozessänderungen sind in der Regel tiefgreifende Einschnitte in das gewohnte Arbeitsleben, die in unterschiedlichem Maße auf Zustimmung, Ablehnung oder Gleichgültigkeit stoßen. Mitarbeiter akzeptieren die geänderten / neuen Prozesse nicht automatisch. Dabei hängen die Mitarbeiterreaktionen, abgesehen von Mentalitäten und vorhandenen informellen Strukturen, unter anderem von folgendem ab:

- Hatten die Mitarbeiter eine echte Chance bei der Prozessgestaltung mit zu wirken?
- Empfinden die Mitarbeiter die geänderten/neuen Prozesse als Verbesserung (Arbeitserleichterung, Produktivitätsgewinn, Abbau von Stresssituationen, Möglichkeit zum selbstbestimmteren Arbeiten)?
- Wurden gezielte Motivationsmechanismen installiert, die die Anwendung der neuen Prozesse stimulieren?
- Gibt es Feedbackmöglichkeiten und gibt es erlebte Feedbackerfolge (z.B. kleinere Prozessänderungen auf Wunsch der Mitarbeiter bei sichtbaren Schwachstellen nach Einführung)?

Mit der Änderung des Prozesses im softwareerstellenden Bereich (Makroprozess) ist es wichtig, den Schaffensprozess des einzelnen Mitarbeiters (Mikroprozess) in Bewegung zu bringen und in die entsprechende Richtung zu lenken, damit die Änderung des Makroprozesses möglichst positiv verstärkt aber zumindest nicht negativ beeinflusst wird. Die Veränderungsinformation des Makroprozesses stößt auf das Beharrungsvermögen des Mikroprozesses. Bei einem physikalischen System lässt sich der Resonanzpunkt einer Schwingung berechnen. Beim Menschen sind wir auf Erfahrung und Fingerspitzengefühl angewiesen, um das ‘Mitschwingen’ zu erreichen.

Auch wenn die Mitarbeiter gewillt sind, nach den geänderten/neuen Regeln zu arbeiten, bedarf es seitens des Managements eines langen Atems und ständiger Motivation, damit das theoretisch Bekannte und grundsätzlich Beschlossene auch praktiziert wird.

3. Wie kann ein Mitarbeiter die 'Qualität seiner Arbeit' verbessern?

Es ist unbedingt notwendig, dass die Mitarbeiter nicht nur verbesserte 'Regeln im großen' anwenden, sondern auch Ihren ganz persönlichen Schaffensprozess überprüfen und eventuell verändern. Erste Voraussetzung für die Änderung des individuellen Mikroprozesses ist die Bereitschaft zur Selbstanalyse bzw. zu Beratungsgesprächen. Dabei sollten folgende Fragen im Mittelpunkt stehen:

- Wie arbeite ich zur Zeit bezüglich Qualität und Produktivität?
- Sehe ich überhaupt Verbesserungsnotwendigkeiten für mich?
- Erkenne ich für mich Verbesserungspotential, das mit akzeptablem Aufwand erschließbar ist?
- Welches sind meine ganz konkreten Schwerpunkte?

Hilfreich zur Analyse sind Messungen, Checklisten, Selbstaufnahmebögen und methodische Beratung (evtl. anonym). Zu beachten sind persönliche Empfindsamkeiten, Datenschutzprobleme und die Einbindung des Betriebsrats (sofern vorhanden) im Rahmen seiner Mitbestimmungsrechte.

Bei der hochsensiblen Änderung persönlicher Prozesse sollten die Mitarbeiter professionelle Unterstützungsangebote bei unbedingter Wahrung von Anonymität und Persönlichkeitssphäre erhalten, da selbst bei vorhandenem Änderungswillen der Erfolg durch Unkenntnis möglicher helfender Techniken unsicher ist. Solche Angebote können sein:

- Anleitungen/Informationen zu dem Wesentlichen der geänderten Prozesse,
- zu beachtende Mindestanforderungen bei der Durchführung der geänderten Prozesse,
- Coachingangebote,
- Teamagreements zur Untersetzung allgemeiner Prozessregeln,
- Feedback des Managements vom Projektleiter aufwärts,
- Angebote (Trainings, Kurse, Bücher) für Arbeitstechniken und Zeitmanagement

Bei dem Versuch, Änderungen des Makroprozesses und der Mikroprozesse in Einklang zu bringen, sind folgende durch die Mitarbeiter erlebbare Faktoren wichtig:

- Unterstützung zur Änderung in/neben dem Tagesgeschäft (eingeplante und auch tatsächlich verfügbare Zeit u.a. für Qualifizierung und Selbststudium),
- Abstimmung von planmäßigen und realistischen Änderungsetappen, -zeiträumen, -intervallen und -bereichen,
- Erhaltung und Vermittlung von Freiräumen und Gestaltungsspielräumen in den neuen Prozessen

- Einhaltung einer Toleranzkurve bezüglich der Einhaltung der Regeln der geänderten/neuen Prozesse (anfangs lockerer, anziehend, Niveau haltend)
- Vermittlung der Problemangemessenheit der Änderungen durch die neuen Prozesse

Es gibt genügend Beispiele in der Fachliteratur u.a. in Verbindung diversen Normen und Standards, dass die Optimierung der Prozesse nicht die erwarteten Effekte brachte. Ein wesentlicher Ursachenkomplex, der sich bei einer Analyse ergibt, ist die fehlende Einbeziehung der Mitarbeiter bzw. fehlende Verbindung mit dem notwendigen persönlichen Qualitätsmanagement des Mitarbeiters.

Traumzeit: Disput über Qualität und Entropie von Softwaresystemen

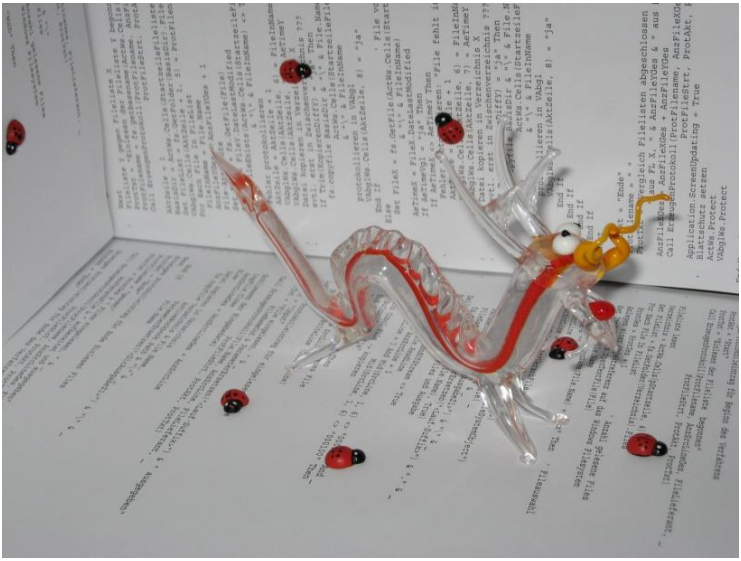
Bei der Übernachtung im Punkt der Selbstbetrachtung erlebte ich in der Traumzeit einen phantastischen Disput über Qualität und Entropie von Softwaresystemen zwischen Mr. Spock, Zaphod Beeblebrox und dem Konstrukteur Trurl.

Zu den Personalien der Disputteilnehmer:

Mr. Spock: Wissenschaftsoffizier des Raumschiffs Enterprise (aus ‚Star Trek')

Zaphod Beeblebrox: Präsident des galaktischen Imperiums (aus ‚Per Anhalter durch die Galaxis')

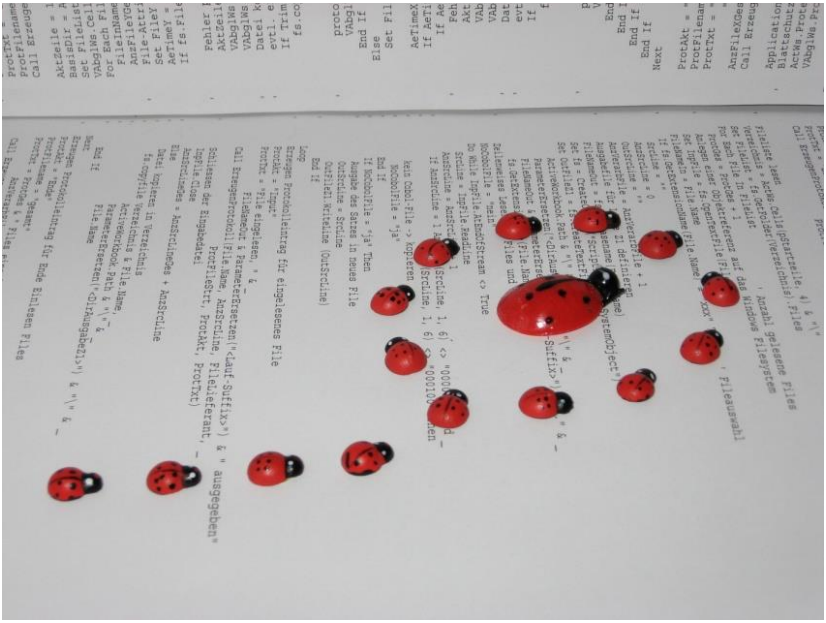
Trurl: Konstrukteur und allgemeiner Omnipotentiator (aus ‚Von den Drachen der Wahrscheinlichkeit' von Stanislaw Lem)



Error-nix- der Drache der Fehler-Wahrscheinlichkeit

Software-Qualität ist wie Sex.
Jeder denkt, er versteht etwas davon ...

- **Mr. Spock**: Von Sex verstehe ich nichts aber …
- **Beeblebrox:** Das gilt nur am Ende des Universums.
- **Trurl:** Die Wahrscheinlichkeit für guten Sex ist nahezu gleich der Wahrscheinlichkeit für gute SW-Qualität.

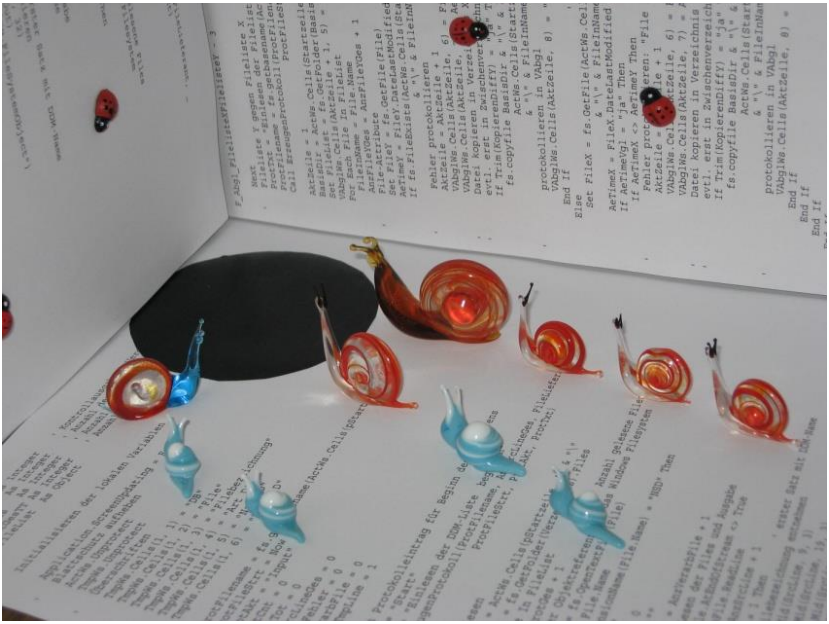


Vermutung über Softwareentropie
~ 2 hoch n*t*x

- **Mr. Spock:** Präzise gilt 2,037256 hoch n*t*x
- **Beeblebrox:** 41 oder besser noch 42
- **Trurl**: Diese Vermutung gilt wahrscheinlich nicht aber ich kenne auch keine bessere.

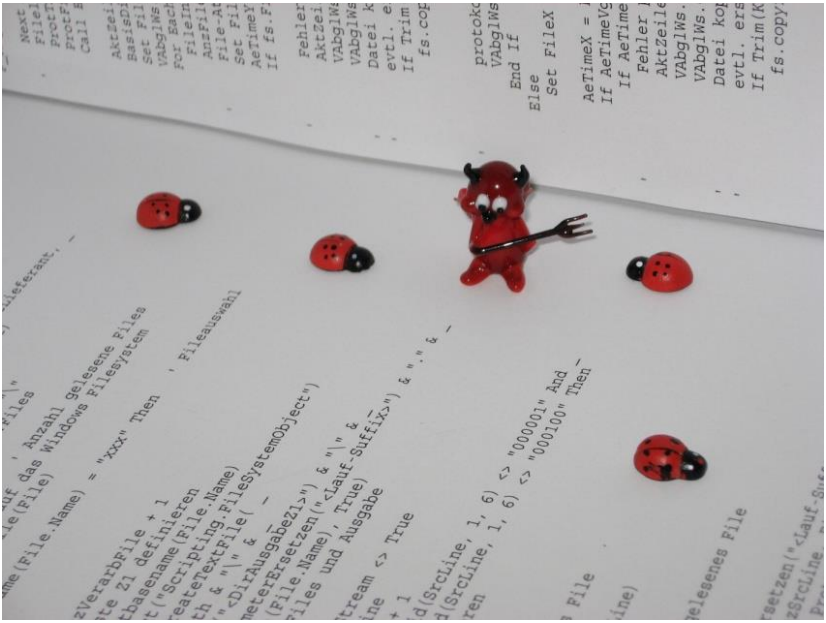
Für Software-Qualität gilt eine Variation des bekannten Gödelschen Unmöglichkeitssatzes vom hinreichend komplexen System …

- **Mr. Spock:** Hier gilt eher der Satz des Vulkanologen Firefox …
- **Beeblebrox:** War Gödel ein Vogone?
- **Trurl:** Münchhausen als Projektleiter löst alle Probleme, denn nur er zieht sich am eigenen Schopf aus dem Sumpf.



Die Software-Entwicklung ist aufhaltbar, denn …

- **Mr. Spock: …** sie stößt an die Grenzen des konvexen Verstands.
- **Beeblebrox: …** in schwarzen Löchern ist SW-Entwicklung sinnlos.
- **Trurl: …** man braucht nur den Grad der Unwahrscheinlichkeit der Fertigstellung zu erhöhen.

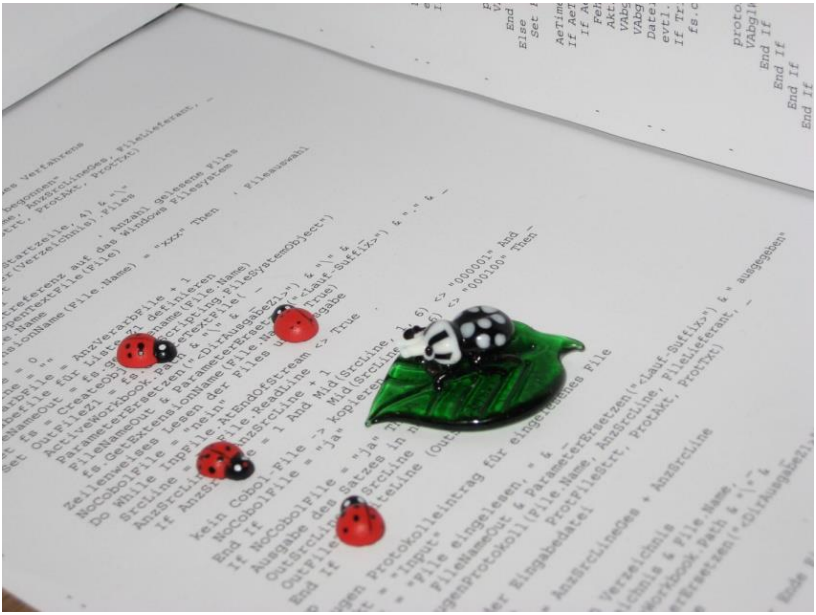


Fehler sind vermeidbar!

- **Mr. Spock:** Die Definition des Fehlers muss dazu logisch schärfer gefasst werden.
- **Beeblebrox:** Nur auf der ‚Herz aus Gold' gibt es keine Fehler.
- **Trurl:** Warum eigentlich? Anders ist es doch viel spannender.

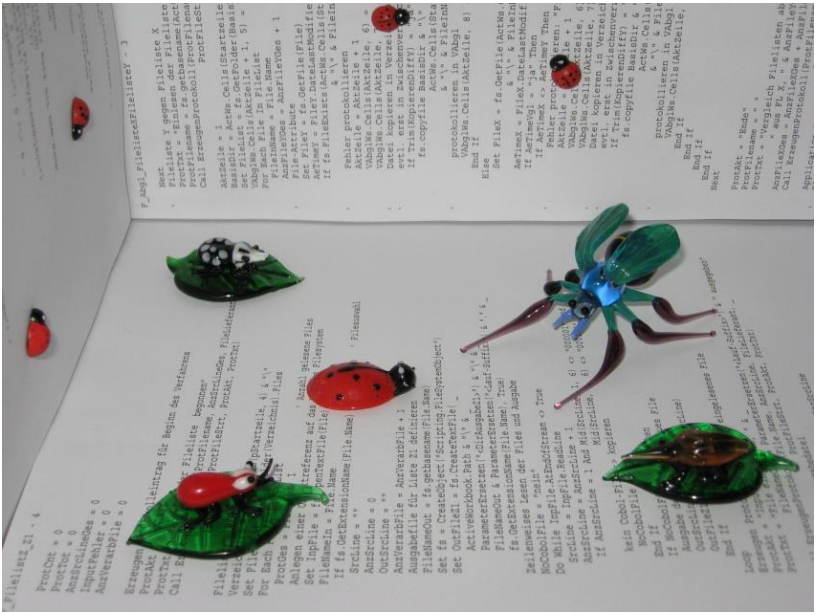
Fehler treten vornehmlich in ‚Haufen' auf.

- **Mr. Spock:** Wann wird aus vielen Einzelnen ein Haufen?
- **Beeblebrox:** Auch für Fehler gilt das erste allgemeine galaktische Zusammenballungsgesetz.
- **Trurl:** Das ist eine alte Konstrukteursweisheit, denn auch die ‚geistige Klärung' braucht ihre Zeit.



Software-Qualität kann per Horoskop ermittelt werden.

- **Mr. Spock:** Da Astrologie keine Wissenschaft ist, verneine ich das entschieden.
- **Beeblebrox:** Wow und was noch?
- **Trurl:** Wenn meine allgemeine Horoskopmaschine verwendet wird, so garantiere ich eine Maximalabweichung von 2 Delta.

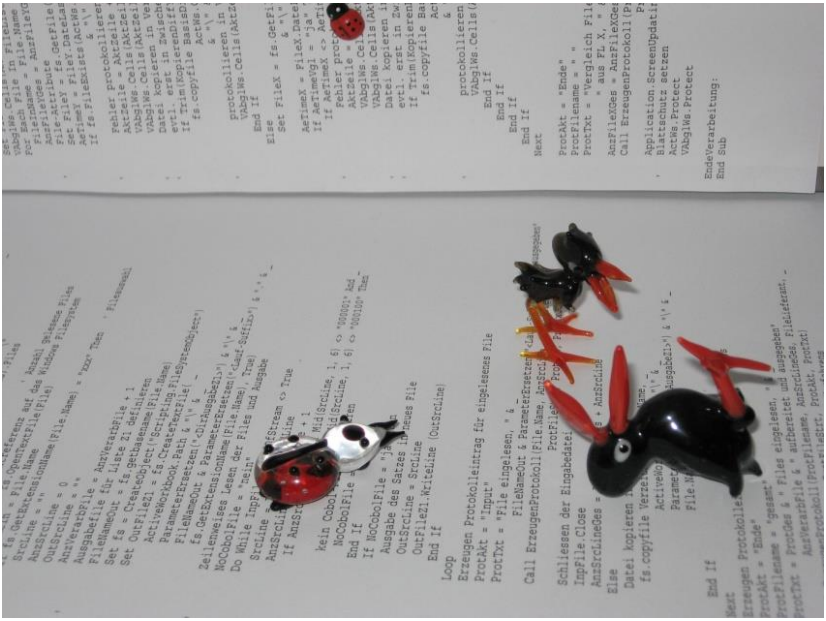


Es gibt ‚schöne' und ‚hässliche' Fehler.

- **Mr. Spock:** Für schöne und hässliche Fehler gibt es keine Kriterien also ist diese Unterscheidung unsinnig.
- **Beeblebrox:** Ich mache keine Fehler, höchstens schöne.
- **Trurl:** Mein Konstruktor ist schön und dies vererbt sich auf alle seine Produkte

Welche Rolle spielt das Unterbewusstsein der Entwickler bei der SW-Qualität

- **Mr. Spock:** Was ist Unterbewusstsein?
- **Beeblebrox:** Ist offenbar entscheidend, denn sehen Sie doch die Qualität mancher Software im Universum. Diese ist mit Bewusstsein allein nicht zu erklären.
- **Trurl:** Dieser Faktor ist bei meinem Konstruktor höchst unwahrscheinlich.



Was halten Sie von einem TÜV für Software-Entwicklung?

- **Mr. Spock:** Der TÜV muss aber objektiven Regeln folgen.
- **Beeblebrox:** Darüber müsste ein intergalaktischer Arbeitskreis befinden und das dauert …
- **Trurl:** Soll ich mal schnell prototypisch eine TÜV-Maschine konstruieren?

Software-Qualität
‚an sich' oder ‚für mich' -
das ist hier die Frage!

- **Mr. Spock:** Das ist keine Frage mit objektivem Hintergrund.
- **Beeblebrox:** Natürlich für mich. Was interessieren mich andere potentielle User.
- **Trurl:** Meinem Wahrscheinlichkeitsgenerator kann man das als Parameter mitgeben.

Können aus der fertigen Software die Qualitätsintentionen der Programmierer ermittelt werden?

- **Mr. Spock:** Intentionen sind etwas Subjektives und damit der Wissenschaft schwer zugänglich.
- **Beeblebrox:** Hatten sie denn welche?
- **Trurl:** Nur mit relativ geringer Wahrscheinlichkeit.

Du kannst die Dinge falsch verstehen
und trotzdem glauben, dass alles ok ist.

Station 3: Im Camp der Nationen

Jedes Teammitglied vermarktet sich nach seinem Gusto und lebt seine Mentalität mehr oder weniger extensiv.
Trotzdem sollte die Teammitglieder das gemeinsame Projektziel einen aber dies tut es nicht automatisch sondern das aktive rücksichtsvolle Handeln ist gefragt.

Projekt- und Qualitätsmanagement in einem multinationalen Team

Dieser Beitrag basiert auf Erfahrungen des Autors in der konkreten multinationalen Projektarbeit und beschreibt u.a. Probleme bei der Gestaltung des Projektmanagements (PM) unter spezieller Berücksichtigung des Qualitätsmanagements (QM) in einem multinationalen Softwareentwicklungsteam. Im Beitrag können Lösungsansätze und Kommunikationsstrategien nur angerissen werden. Für intensivere Studien wird u.a. auf die aufgeführte Literatur verwiesen.

1. Grundansatz bei der Gestaltung des PM/QM in einem multinationalen Team

Das erfolgreiche Management von Projekten, die durch multinationale Teams durchgeführt werden sollen, erfordert bestimmte vorbereitende Arbeiten, damit überhaupt Voraussetzungen für einen Erfolg vorhanden sind. Idealerweise könnte man sich eine durch Psychologen und Kommunikationswissenschaftler zusammen mit dem künftigen Projektmanagement vorgenommene Teambildung vorstellen. In der Praxis sieht es in der Regel aber so aus, dass das Projektmanagement mit einer vorgegebenen Teamzusammensetzung arbeiten muss. Die Vorbereitung erfolgt bestenfalls gezielt durch das Unternehmen bzw. durch Studium von Veröffentlichungen bzw. Konsultation entsprechender Fachleute oder erfahrener Kollegen.

Dabei empfiehlt sich zur Vorbereitung u.a.

- Analyse der Teamzusammensetzung auf kulturelle Schwerpunkte verbunden mit der Berücksichtigung persönlicher Mentalitäten

  Prinzipiell reicht eine Einteilung der kulturellen Herkunft nach den großen Gruppen (linear-aktiv = aufgabenorientiert, spezialisiert, sukzessiv arbeitend; multi-aktiv = beziehungsorientiert, parallel arbeitend; reaktiv = beziehungsorientiert, ganzheitlich vorgehend). Die persönliche Mentalität überlagert die kulturell konditionierten Merkmale. Diese Überlagerung wirkt sich unterschiedlich stark aus. Zur Analyse sollte das Projektmanagement mit den einzelnen Teammitgliedern strukturierte Gespräche führen, die für jedes Teammitglied eine erste Einschätzung ermöglichen.
  Auf der Basis der fachlichen Aufgaben und dieser Einschätzung wird die Zuordnung der Mitarbeiter zu Aufgaben und die Bildung von Teilteams vorgenommen. Dabei sollte dem Team signalisiert werden, dass eine spätere Korrektur dieser Zuordnung möglich ist.

- Einsicht in Mechanismen interkultureller Interaktionssituationen entwickeln

  Zusätzlich zu den auch in jedem kulturell homogenen Team vorhandenen Situationen mit Konfliktpotential treten in multinationalen Teams Konflikte bei speziellen Interaktionssituationen auf, die nur gemeistert werden können, wenn man sich ihrer Mechanismen bewusst ist. Dazu gehört u.a. die Wirkung des kulturellen Horizonts, nationaler Vorurteile und spezifischer Lebensweisen.
  Problemträchtige Interaktionssituationen kommen speziell bei. der Etablierung eines Management-/Führungsstils im Team, der Gestaltung der Projektplanung und der Durchführung von Meetings, Reviews und Tests vor. Bei diesen Situationen treffen unterschiedliche kulturelle Modelle der Organisation, der Gruppenarbeit, der Problemlösung, der

Plangestaltung, des Zuhörens, des Zeitgefühls, der Gesichtswahrung usw. aufeinander.
Dem Projektteam sollten durch das Projektmanagement die im Team vorhandenen wesentlichen kulturellen Modelle und Regeln für den Umgang mit selbigen vermittelt werden. Auch hier ist eine erkennbare Grundstrategie des Projektmanagements (weitgehende kulturelle Neutralität, Zuverlässigkeit und Berechenbarkeit verbunden mit Flexibilität im Detail) für alle Teammitglieder sehr wichtig.

- Strategien zur Identifikation, Analyse und Behebung von kulturellen Missverständnissen erarbeiten

  In jedem Projektteam gibt es täglich Missverständnisse bzw. Diskrepanzen über Ziele, Prioritäten und Ergebnisse. Eine der wichtigsten Aufgaben des Projektmanagements ist die Beseitigung bzw. Minimierung der Missverständnisse und die stetige Neuorientierung an jeder ‚Wegkreuzung'. Kulturell bedingte Missverständnisse gibt es speziell bei der Verbindlichkeit von Zielen und Terminen sowie der Exaktheit und Vollständigkeit von Ergebnissen. Das Auftreten von Problemen dieser Art ist ein starkes Indiz für Mängel in der interkulturellen Teamkommunikation. Diese müssen dann analysiert, im Team besprochen und ohne Gesichtsverlust von Teammitgliedern geklärt werden. Nach einer bestimmten Zeit entwickelt sich eine eigene Teamkultur, die ausgleichend und überdeckend wirkt. Erfahrungsgemäß durchbricht die kulturelle Vorprägung der Teammitglieder speziell in Stresssituationen immer wieder die dünne Decke der Teamkultur. Durch das Projektmanagement muss dann ausgleichend aber bestimmt für Beseitigung bzw. Verständnis der Ursachen der Missverständnisse gesorgt werden.

- PM/QM nach einem Dominanz-, Kompromiss- oder Synergiemodell?

In multinationalen Teams sind mehrere Managementmodelle zur Beherrschung der aus der kulturellen Vielfalt resultierenden Probleme und Möglichkeiten denkbar. Diese können im Wesentlichen auf drei Grundmodelle reduziert werden:

- Dominanzmodell (eine ‚Kultur' dominiert)
  Anwendbar eigentlich nur bei entsprechender Teamstruktur, wo z.B. nur ein oder wenige Mitarbeiter einer anderen Kultur entstammen. Ermöglicht klare konsistente Regelungen. Provoziert interkulturellen Widerstand.
- Kompromissmodell (Kompromisse aus allen ‚Kulturen')
  Anwendbar bei Teamstruktur aus mehreren gleichstarken kulturellen Gruppen. Provoziert wenig Widerstand gegen Regelungen. Nicht immer sind Kompromisse vernünftig konstruierbar. Bei zu großen Divergenzen sind Kompromisse inkonsistent und ineffektiv.
- Synergiemodell (aus jeder ‚Kultur' das Beste bzw. etwas Neues)
  Anwendbar nur bei langfristiger Zusammenarbeit und entsprechender Bereitschaft aller Beteiligten zur Akzeptanz gleichberechtigter kultureller Vielfalt.

Abseits aller theoretischen Diskussion über Nachteile und Vorzüge der Modelle zeigt sich, dass bei kurzfristigen IT-Projekten meistens schon allein aus Zeitgründen ein dem Dominanzmodell angenähertes Vorgehen verwandt wird. Bei großen und längerfristigen (> 1 Jahr) Projekten wird in der Regel zum Kompromissmodell tendiert.

2. Die Festlegung von Projekt- und Qualitätszielen als erste Bewährungsprobe

Für jedes Projekt sind Projektziele definiert. Qualitätsziele werden oft nicht oder nur verschwommen festgelegt. Bei der

Bewertung der Ziele gibt es die ersten interkulturellen Diskrepanzen, die aus der unterschiedlichen Bewertung von vertraglichen Vereinbarungen in unterschiedlichen Kulturen herrühren (‚absolut gültig' bis ‚Absichtserklärung'). Hier zugelassene Missverständnisse wirken für die gesamte Projektdauer negativ.

Grundvoraussetzung für Einvernehmen im Team ist eine für alle verständliche Definition und Interpretation der Ziele. Darauf muss notfalls die entsprechende Zeit verwendet werden. Sehr hilfreich ist die Zieldarstellung und –interpretation durch eine Person aus dem übergeordneten Management bzw. vom Kunden, da deren Autorität durch alle Kulturen in etwa gleich bewertet wird.

Gibt es trotzdem fortbestehende Diskrepanzen, so muss unbedingt die Ursache ermittelt werden. Semantische Probleme sind in der Regel schnell geklärt. Bei Streitigkeiten, die nicht durch logische Folgerung entscheidbar sind, wie z.B. Umfang und Art einer Dokumentation, ist im Notfall die Entscheidung durch das Projektmanagement notwendig. Sind noch nicht alle Voraussetzungen geklärt oder ist ein anderer Entscheidungszeitpunkt vertretbar, so sollte man vor einer Verschiebung der Präzisierung der Ziele nicht zurückschrecken.

Kompromisse bei der Zielfestlegung bzw. –interpretation sind sehr sorgfältig zu erarbeiten. Hierbei sind negative Bewertungen anderer kultureller Standpunkte unbedingt zu vermeiden. Es darf zu keinem Gesichtsverlust oder Ehrverlust von Teammitgliedern kommen, damit die Atmosphäre im Team nicht dauerhaft geschädigt wird. Am besten erweist sich eine Motivation des Kompromisses durch Balance zwischen Kunden- und Teaminteresse bzw. verschiedenen Kundenzielen. Der Vorschlag zum Kompromiss sollte nicht unbedingt durch den Projektmanager erfolgen. Es bietet sich hierzu, sofern vorhanden, der Qualitätsverantwortliche des Projekts an.

3. Wie kann man Reviewprozesse ohne Gesichtsverlust der Beteiligten einsetzen?

In der Projektarbeit ist notwendig, regelmäßig den erreichten Stand und die Durchführung der Arbeit zu bewerten, um objektivierte Grundlagen für Entscheidungen zu erhalten. Projektergebnisse werden durch Personen erarbeitet. Damit werden bei der Bewertung automatisch Personen beurteilt auch wenn dies nicht explizit geschieht.
Hierbei treten gravierende kulturelle Wertunterschiede im Team besonders in Erscheinung. Die Spanne reicht hier von der offenen Bewertung der Leistung einzelner bis zur unbedingten Vermeidung der Personifizierung der Leistung.
Prozessreviews oder Audits sind bei sensibler Durchführung weniger problematisch als Ergebnisreviews, da bei der Beurteilung von Prozessen den Beteiligten mehr Rechtfertigungsmöglichkeiten offen stehen (Motto: viele Wege führen nach Rom).
Ergebnisreviews sind in der Durchführung kritischer. Dabei sollte sich das Projektmanagement bei Verwendung des Kompromiss- oder Synergiemodells unbedingt an der Kultur mit der höchsten Gesichtsverlustschwelle orientieren. Dies bedeutet, dass im Review selbst auf keinen Fall eine personelle Bewertung anklingt. Im Extremfall sollte z.B. die Begutachtung einzelner Softwaremodule durch Code-Inspektion nur für mehrere Programmierer zusammen vorgenommen werden.
Bei der Zusammensetzung von Reviewteams muss zusätzlich beachtet werden, dass auf keinen Fall der Eindruck entstehen darf, als ‚urteilten' Angehörige einer Kultur über eine andere. Dies bedeutet, dass die Reviewteams gut gemischt zusammengesetzt werden müssen bzw. mindestens die Moderation weitgehend neutral erfolgen muß. Dabei sind Quotenmodelle keine Lösung. In einem Projektteam entwickeln sich relativ schnell teaminterne Wertschätzungen auf Grund der Qualifikation und der Leistungen. Zusammensetzungen von Reviewteams auf dieser Basis werden im Allgemeinen wesentlich eher akzeptiert.

4. Wie sollte man mit auftretenden Fehlern in der Softwareentwicklung umgehen?

Spätestens nach dem Entwicklertest kommt beim Integrationstest, Funktionstest oder sonstigen teamübergreifenden Testformen die Stunde der Wahrheit. Es werden Fehler in konkreten Softwarebausteinen im Team öffentlich. In einem Team wird sehr sensibel und sehr schnell subjektiv ein ‚durchschnittlicher Fehlerwert' gebildet und daran die Teammitglieder gemessen. Bei relevanten Überschreitungen dieses Durchschnitts kommen sehr schnell Schuldzuweisungen der Form *'X hat schon immer schlampig gearbeitet, wahrscheinlich kennt er es nicht anders'* hoch. Alte, im Team schon überwunden geglaubte, Vorurteile sind plötzlich wieder da.

Auf diese Situation muss das Projektmanagement vorbereitet sein. Dies bedeutet u.a.

- Die Situation wurde im kleineren Maßstab schon trainiert, indem einzelne Teile vorgezogen wurden bzw. von vornherein iterativ gearbeitet wird.
- Es wurde rechtzeitig eine objektive Testfehlererfassung installiert. Die Ergebnisse werden in geeigneter Weise öffentlich gemacht.
- Werden Test- und/oder Changemanagementtools verwendet, so sollten Auswertungen der Form ‚wie viele Fehler/Änderungen gab es bisher zum Modul X oder Paket Y' dem einzelnen nur für ihm oder seiner Teilgruppe zugeordnete Module/Pakete möglich sein. Nur das Projektmanagement darf Auswertungen über jeden Teil durchführen.
- Die Gesamtverantwortung des Teams für die Ergebnisse wird betont.
- Es werden Ursachen aber keine Schuldigen gesucht.
- Die Beseitigung der Fehler geschieht entweder durch den Einzelnen oder im kleinen Team.

- Fortschritte werden positiv herausgestellt.

Trotz aller Vorkehrungen können immer wieder schwierige Fälle auftreten, in denen tatsächlich nicht mit der notwendigen Sorgfalt gearbeitet wurde oder in denen Diskrepanzen über den Umgang mit Fehlern auftreten. Dann darf das Projektmanagement einer Klärung nicht ausweichen. Dabei bewährt sich im Allgemeinen folgende Reihenfolge:

- Führung unangenehmer Gespräche unter vier Augen oder im kleinen Kreis
- Einbeziehung von Mitarbeitern des gleichen Kulturkreises in die Gespräche
- Diskussion im Team unter Moderation eines Außenstehenden

Führen alle diese Maßnahmen nicht zum Erfolg, so muss auch eine Trennung vom entsprechenden Teammitglied erwogen werden, wobei diese auf keinen Fall mit kulturellen Eigenheiten sondern nur mit persönlichen Eigenschaften begründet werden darf.

5. Beachtung der kulturellen Unterschiede ist eine notwendige aber keine hinreichende Voraussetzung für den Erfolg.

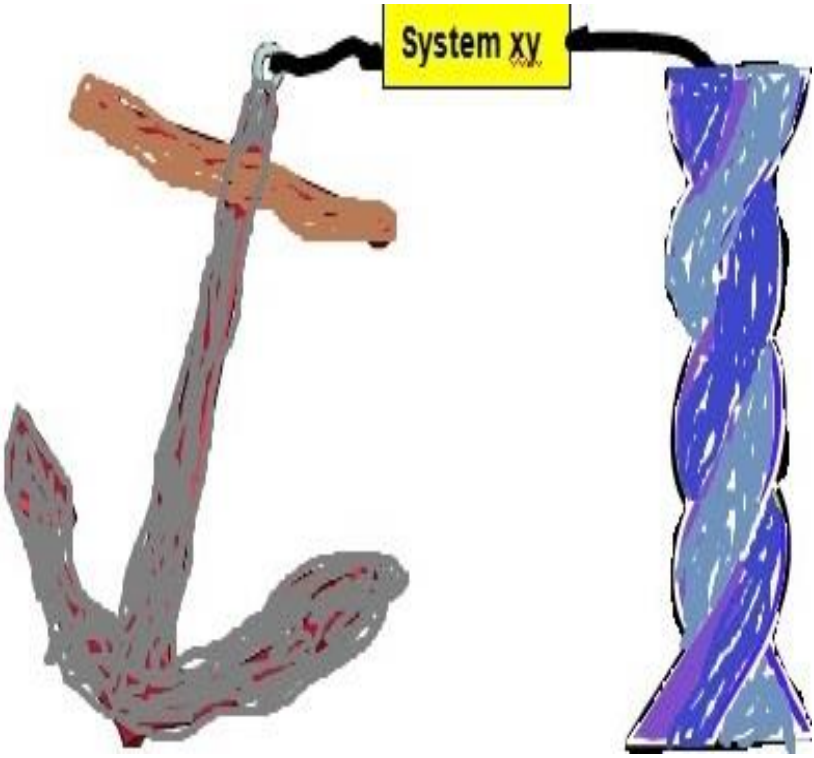
Bei der Durchführung eines IT-Projekts gibt es die vielfältigsten Risiken. Kulturelle Unterschiede sind nur ein Risiko von vielen. Je nach Art und Größe des Projekts besitzen sie unterschiedliche Bedeutung. Die kulturellen Unterschiede sollten mindestens bei der Definition der Ziele, der Projektorganisation und der Gestaltung der Problemlösungsprozesse berücksichtigt werden.
Das Projektmanagement muss in seiner Arbeit deutlich machen, dass es keine Bevorzugung von Kulturen im Team gibt. Dies ist sehr schwierig, da auch das Projektmanagement kulturell geprägt ist.

In der Praxis legt sich das Projektmanagement einen Satz von Faustregeln für den Umgang mit Teammitgliedern aus anderen Kulturen zurecht. Diese Faustregeln beruhen auf Literaturstudien oder anderen Hinweisen. Wichtig ist, dass sie nicht zu Vorurteilen erstarren sondern immer wieder hinterfragt werden. Trotz (scheinbar) methodisch richtiger Vorgehensweise kann es aber immer wieder zu interkulturellen Konflikten im Team kommen. Häufige Ursachen sind dabei fehlerhafte Einschätzung der Mitarbeiter (kulturelle Prägung, Nationalität versus Staatsangehörigkeit, Mentalität) und das Hervorbrechen nationaler Vorurteile in unkontrollierten Situationen (starker Stress, Teamfeier, Alkoholgenuss usw.). In dieser Situation ist dann Autorität und Fingerspitzengefühl des Projektmanagements gefragt, wobei je nach Schwere des Konflikts sachkundige Beratung durch Psychologen und Kommunikationswissenschaftler zu empfehlen ist.

Traumzeit: Software-Metrie für Dummies

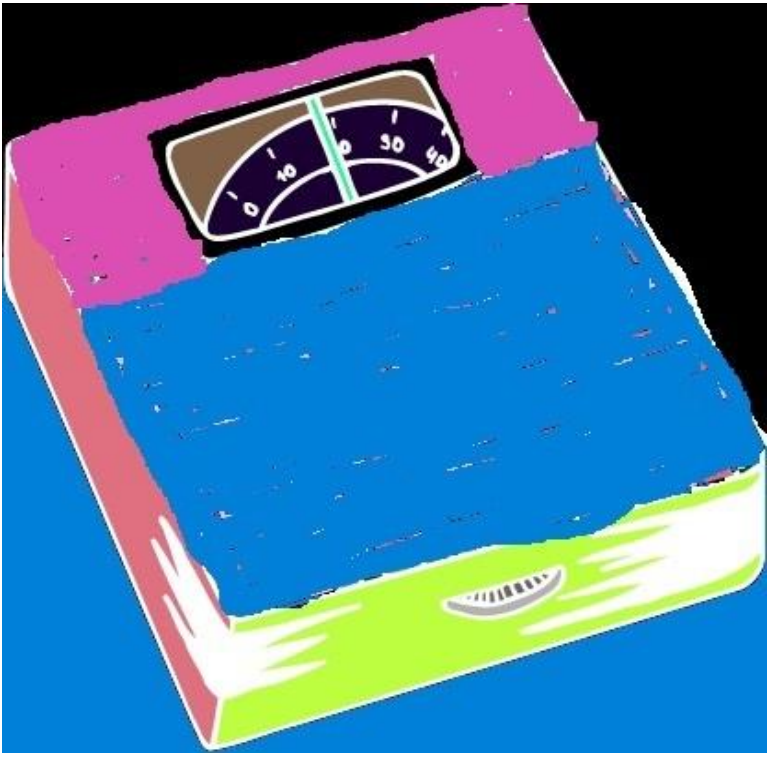
In der Traumzeit zogen nochmals meine Erlebnisse mit den vielfältigen (auch eigenen) Versuchen zur Verbesserung der Softwarequalität in diversen Firmen an mir vorbei. Eine entscheidende Rolle dabei spielte jeweils der Anspruch des Managements, quantitative Aussagen über den erreichten Verbesserungsgrad zu erhalten. Die dabei vorgeschlagenen bzw. in den Firmen angewandten Messmethoden sind / waren oft von erstaunlicher Primitivität bzw. huldigten einem Mystizismus a la ‚measurement by guru'. Die im Folgenden beschriebenen Methoden sind nicht unbedingt zur Nachahmung empfohlen aber wahrscheinlich sind sie nicht auszurotten. Wie sonst wäre es zu erklären, dass dem Autor in seiner QS-Praxis bei jedem Jahrgang von Softwareentwicklern immer wieder die gleichen oder ähnlichen in den Entwicklertests nicht entdeckten Fehler begegneten.

Software-Qualität
gewogen und für zu leicht befunden



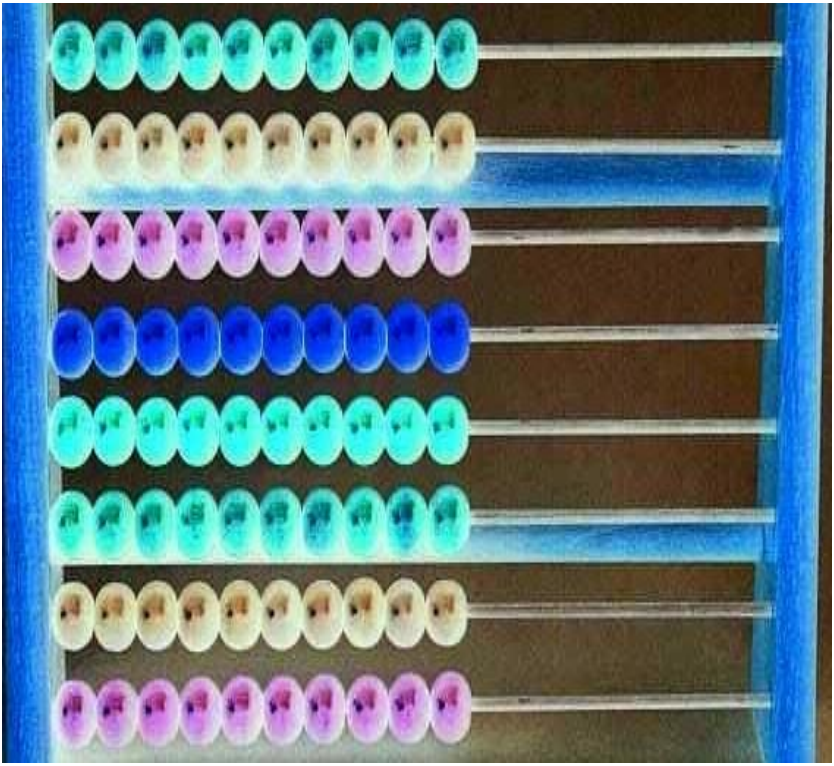
die Belastbarkeit

misst man am besten durch kräftiges Ziehen an einem Ende des Systems nachdem man das andere Ende möglichst fest verankert hat



der Umfang oder das Gewicht eines Systems

gemessen zum Beispiel in ‚Lines of code' eignet sich vortrefflich für Managementberichte aber weniger für Qualitätseinschätzungen



die Anzahl
der entdeckten Fehler

steht nur bedingt in einem vorhersag-
baren Verhältnis zur Anzahl der ent-
haltenen Fehler auch wenn sich dies
das Management so sehr wünscht

die Datensicherheit

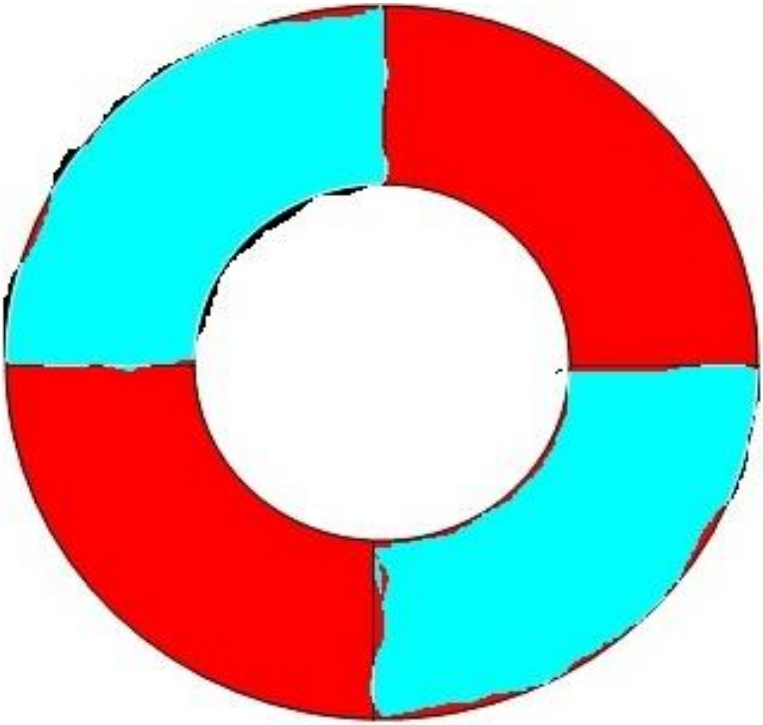
eines Systems zeigt sich u. a. daran, ob alle Anfragen und Ausgaben des Systems durch mindestens einen geschützten Filter gehen

die Benutzerfreundlichkeit

zu messen ähnelt dem Abschmecken eines Gerichts

frei nach Goethe

„Allen Menschen recht getan, ist eine Kunst, die niemand kann.“

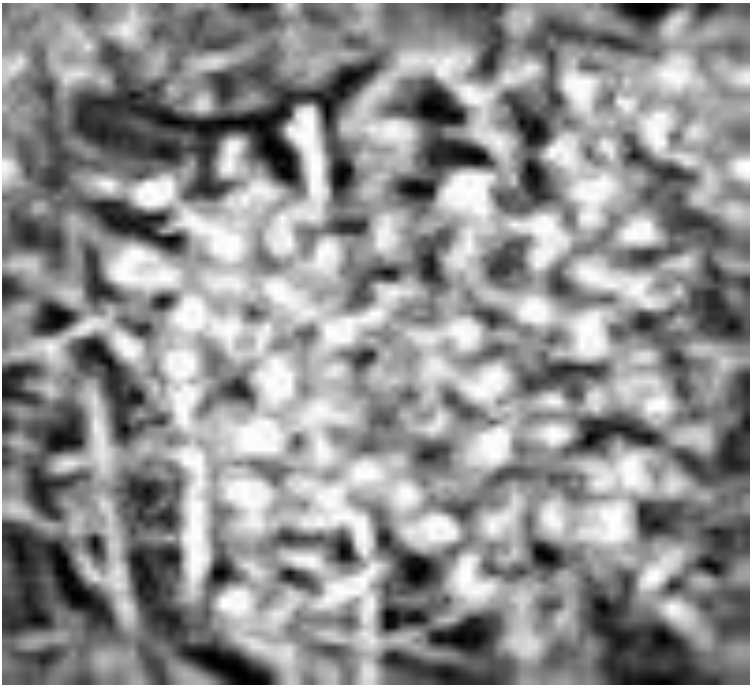


die Zuverlässigkeit

zeigt sich auch daran, dass Ausnahmesituationen nicht zu Abbrüchen führen und das System für die getane Arbeit immer einen Rettungsring bereithält, damit sie nicht umsonst war

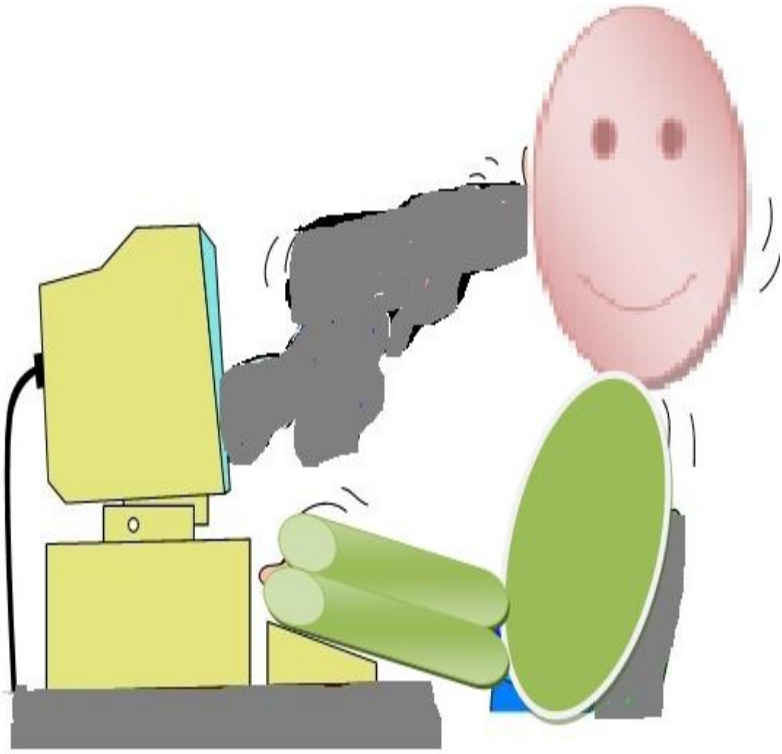
die Korrektheit

misst man am besten mit den altbe-
währten geometrischen Instrumenten
aber man vergesse nicht zu prüfen,
ob das System vor lauter Korrektheit
auch etwas Sinnvolles tut



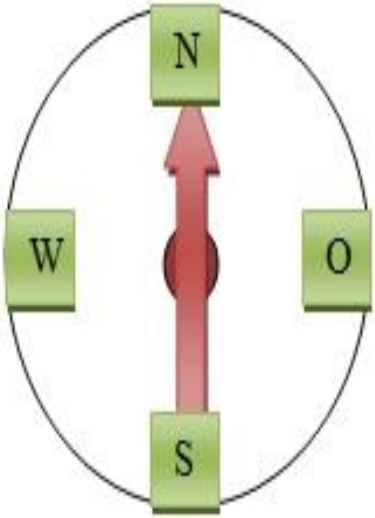
die Wohlstrukturierheit

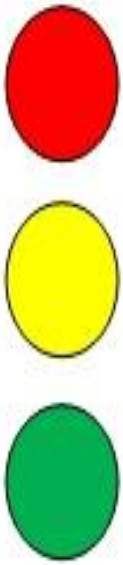
ist zwar eine wichtige Softwarequalitätseigenschaft aber auch der Heuhaufen hat eine Struktur, die nur darauf wartet, entdeckt zu werden -> wie spannend



die Sanftmut und Geduld
eines Systems

ist nicht grenzenlos,
deshalb provoziere es nicht
unnötig mit unsinnigen Eingaben

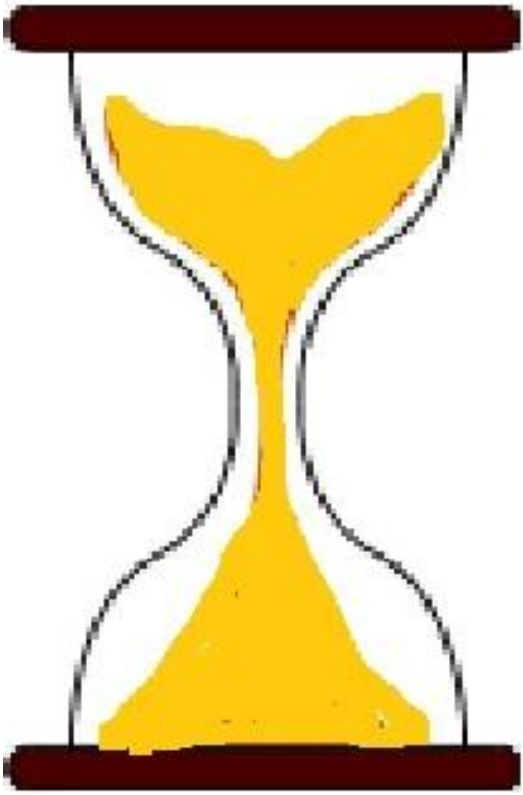




die Qualität der Benutzerführung in Dialogen

kann man an den eingebauten Richtungsangaben und den, den Aktionsfluss lenkenden Ampeln (Sicherheitsabfragen) erkennen.

Es muss nicht unbedingt ein sprechendes Navi sein.



die Performance

eines Anwendungssystems
misst man am besten
mit einer robusten Sanduhr

einen ganzheitlichen Eindruck
von
der Qualität eines Systems

vermittelt letztlich nur der Blick in die Kristallkugel

Er sieht nur, was er sehen will.
Gilt das nicht auch für dich und mich?

Station 4: An der Psycho-Falle

Der Softwareentwickler lebt in einer Welt, in der weder sein Hintergrundwissen noch sein Denkapparat von der Evolution zum Zwecke des Programmierens selektiert oder gar konstruiert worden sind. In prähistorischen Zeiten lebenserhaltende Verhaltensweisen (Jagd- und Fluchtreflexe) sind beim Programmieren eher untauglich.

Projektmanagement zwischen Psychologie und Tools

1. Probleme des Projektmanagements aus psychologischer Sicht

Verschiedene psychologisch begründete Probleme beeinflussen die Arbeit des Projektmanagers und seines Teams. Allein die Kenntnis dieser Probleme und das Verständnis ihrer Ursachen ist schon ein erster Schritt zum besseren Projektmanagement. Im Folgenden werden einige der wichtigsten Probleme besprochen.

- Es fehlt das ausreichende Verständnis für die Notwendigkeit und Möglichkeit eines Mindestmasses an Planung, Steuerung und Organisation bei Team und Management.

  Eine gute Charakterisierung der Situation liefert folgendes Zitat über Qualität, das aber sinngemäß auch auf das Projektmanagement anwendbar ist:

  Erstaunlicherweise gibt es Dinge, die Sex und Qualität gemeinsam haben. Nach Phil Crosby sind dies folgende:

  1. **Alle sind dafür.**
  2. **Jeder glaubt, dass er etwas davon versteht.**
  3. **Jeder denkt, bei der Umsetzung müsse er nur seinem natürlichen Instinkt folgen.**

4. **Fast alle sind der Ansicht, dass für entstehende Probleme andere verantwortlich sind.**

Bei einer Betrachtung der Ursachen dieses Verständnisproblems kann man u.a. folgendes feststellen

- Psychologisch wird unbewusst das Sparsamkeitsprinzip / der Einfachheitsgrundsatz angewandt, d.h. aus einer begrenzten oft sehr kleinen Anzahl von Beobachtungen wird auf Gesetzmäßigkeiten geschlossen
  - die einfachsten Vermutungen werden zuerst überprüft,
  - weil es schon immer so war, deshalb wird es auch jetzt so sein,
  - Verfahrensweisen, die in einem Fall gut funktionierten werden unkritisch in einem anderen Fall übernommen,
  - Kochrezepte ‘man nehme …’ sind etwas aus der Kindheit Vertrautes und wurden vom ersten in Aktion erlebten Projektmanager (der Mutter) ständig erfolgreich verwendet.
  - In der Literatur und auf Seminaren werden zu viele verschiedene ‚wunderwirkende’ Techniken angepriesen, die aber nicht so bzw. nicht so einfach umsetzbar sind.
  - In Interviews mit erfolgreichen Managern sprechen diese kaum über von Ihnen angewandte Techniken. Entweder werden Sie nicht dazu befragt oder Sie wenden selbst keine oder nur einfache Techniken an oder sie wollen Ihre Arbeitsweise im Detail nicht preisgeben.

Theoretisch wird die Notwendigkeit eines Mindestmaßes an Planung, Steuerung und Organisation nach den ersten Schwierigkeiten oft akzeptiert. Aber was ist das dem aktu-

ellen Projekt angemessene Mindestmaß? Wie kommt man zu diesem Mindestmaß?

Aus der Fülle der eigenen Erfahrungen, der Empfehlungen der Literatur und den umfassenden Beschreibungen eventuell vorhandener Vorgehensmodelle muss eine sinnvolle Reduktion vorgenommen werden.

Dies kann u.a. geschehen durch

- Orientierung an ähnlichen Situationen im Erfahrungsschatz
- Beachtung in der Literatur empfohlener Reduktionsregeln
- Nutzung der Ratschläge anderer.

Letztendlich muss man aber akzeptieren, dass

- es immer mehrere Möglichkeiten gibt,
- oft nicht eindeutig eine ‚beste' Möglichkeit definierbar ist,
- es besser ist eine 'suboptimale' als keine Lösung zu wählen,
- man Verantwortung für die gewählte Lösung übernehmen muss.

- Zielerreichungs-/Priorisierungskonflikte zwischen Termin, Kosten und Qualität

  Es gibt in einem Team keine einheitliche Sichtweise auf diese immer vorhandenen Konflikte sondern die Sichtweise des einzelnen Teammitglieds ist abhängig von der Rolle im Team, der kulturellen Herkunft und der persönlichen Mentalität. Im Folgenden wird dies an Beispielen illustriert:

  - Rolle im Team
    - Projektmanager (je nach Priorisierung des übergeordneten Managements aber oft vorrangig termin- und kostenorientiert)

    - Entwickler (vorrangig qualitätsorientiert aber oft Reduzierung auf das Qualitätsmerkmal ‚Funktionsabdeckung')
    - Qualitätsmanager (vorrangig qualitätsorientiert mit einer umfassenderen Qualitätssicht unter Vernachlässigung der Kosten)

- kulturelle Herkunft

  Projektziele entsprechen in ihrer Bedeutung Vertragsinhalten. Verträge werden in verschiedenen Kulturen unterschiedlich bewertet. Die Bewertungsspanne reicht von

    - Ein Vertrag ist gültig und muss bei Strafe von Sanktionen eingehalten werden. (europäisch-nordamerikanischer Kulturkreis)

  bis

    - Ein Vertrag spiegelt die Absichten der vertragsschließenden Seiten zum Zeitpunkt des Vertrages (Projektbeginn) wieder. Im Verlaufe der Zeit können sich diese Absichten geändert haben, Ist dies der Fall, so sollte der Vertrag ohne große Formalitäten änderbar sein. (ostasiatischer Kulturkreis)

- persönliche Mentalität (z.B. Choleriker, 'Erbsenzähler')

Bei der Kompromissfindung bzw. der Festlegung der aktuellen Strategie sollte folgendes beachtet werden:

- Diskussion im Team ohne negative Bewertung anderer kultureller Standpunkte,
- kein Gesichtsverlust für die Teammitglieder, die sich in der Diskussion nicht durchsetzen konnten,
- gewählte Lösung oder Kompromiss sollte durch neutrale Instanz außerhalb des Teams vorgeschlagen/gefordert/unterstützt werden, z.B. durch Projektauftraggeber oder Vertreter des übergeordneten Manage-

ments.Tätigkeits-/Zeitkonflikte zwischen den unterschiedlichen Tätigkeiten wie Durchführen (Analyse, Design, Realisierung), Planen, Führen/Steuern, Informieren/Entscheiden und Überwachen

Die genannten Tätigkeiten haben für jedes Teammitglied eine unterschiedliche Wertigkeit. In der Regel haben die Teammitglieder zumindest innerhalb eines bestimmten Zeitabschnitts Schwerpunkte in ihrer Tätigkeit.

Für den Entwickler ist z.B. die Realisierung sein Schwerpunkt. Alles was ihn davon abhält ist für ihn im Extremfall vergeudete Zeit bzw. höchstens widerwillig anerkannte Notwendigkeit. Dies trifft beispielsweise auf Eintragungen in einem Arbeitszeiterfassungssystem oder Berichte über den Arbeitsfortschritt zu. Die Notwendigkeit dieser Daten für eine qualifizierte Fortschrittskontrolle, Planung oder Entscheidung über Projektsteuerung ist ihm oft subjektiv nicht einsichtig. Der Entwickler wird also in der Regel die ungeliebten Tätigkeiten niedriger priorisieren und aufschieben, so lange es geht. Gleichzeitig schätzt er, die für seinen Hauptschwerpunkt ‚realisieren' notwendigen Fähigkeiten und Fertigkeiten (z.B. Entwerfen und Realisieren programmierbarer Algorithmen) höher ein als die notwendigen Fähigkeiten für seine Nebentätigkeiten. Dies kann doch jeder. Wenn er das anstelle der Realisierung machen würde, dann könnte sein Projektleiter mal sehen, wie man so was eigentlich macht.

Ähnlich verhält es sich mit den anderen Rollen im Team.

Der Qualitätsverantwortliche arbeitet mit Genauigkeit und Akkuratesse an seinen Testplänen und der Testdurchführung. Er beklagt die Schlampigkeit mancher Entwickler, die zufrieden sind, wenn es für Standardfälle funktioniert und für die weitere Tests Zeitverschwendung sind. Sein Projektleiter sieht nur den Termin und handelt nach der Prämisse ‚wird schon klappen'.

Der Projektmanager beklagt, dass keiner gewillt ist, dass gesamte Projekt in seiner Komplexität zu berücksichtigen. Er hat den lieben langen Tag damit zu tun, anderen Steine aus dem Weg zu räumen, Partikularinteressen unter einen Hut zu bringen und das Management zu informieren. Wer dankt ihm das? Jeder meckert nur, wenn seins nicht sofort erledigt wird. Wenn er doch auch wenigstens ein bisschen entwickeln könnte. Dann hätte er wenigstens auch jeden Tag sein Erfolgserlebnis.

Natürlich war die obige Darstellung überspitzt. Es steckt aber ein wahrer Kern darin: Jede Tätigkeit verlangt andere Fähigkeiten. Jeder hat seine Stärken und seine Schwächen auch ein Projektmanager.

In großen Teams werden die Aufgaben verteilt. Damit können zumindest potenziell Kompetenzen und Fähigkeiten berücksichtigt werden.

Je kleiner das Team, desto mehr dieser Aufgaben müssen in Personalunion vom Projektmanager oder von anderen Teammitgliedern durchgeführt werden. Damit werden z.B. oben geschilderte Konflikte in die Psyche einer Person verlagert. Die Tätigkeits-/Zeitkonflikte müssen durch Priorisierung täglich neu aufgelöst werden oder durch Verdrängung werden die weniger schönen Tätigkeiten weggeschoben oder es kommt zu inneren Zerreißproben, die z.B. in Schlaflosigkeit münden.

Eine Hilfestellung besteht in Diskussionen im Team bzw. in der Möglichkeit einen Außenstehenden bei Problemen zu konsultieren und notfalls die Aufgabenteilung zu verändern.

- Komplexitätsgrad des Projektes und dessen Auswirkungen

  Analog zu Komplexitätsmaßen bei Software kann man solche Maße auch bei Projekten betrachten, wie im Folgenden gezeigt wird:

  - Umfang des Projekts (Personentage, Teamstärke),

Beide Umfangsgrößen wirken unterschiedlich. Langläufer mit relativ geringer Teamstärke sind in der Wirtschaft heute prinzipiell nicht mehr anzutreffen. Dies liegt u.a. an den sich immer mehr beschleunigenden Innovationszyklen und dem Budgetierungsdruck. Die echten Problemprojekte sind Projekte mit großer Teamstärke in relativ kurzen Zeiträumen (<= einem Jahr), da hierbei viele Konflikte in kurzer Zeit zu bewältigen sind aber die Zeit für die Stabilisierung und Bewährung der installierten Projektmechanismen sehr knapp ist.

- Grad der Modularisierung
  Unter dem Grad der Modularisierung wird die Anzahl der Teilprojekte und der Teilaufgaben in den Teilprojekten verstanden. Je mehr der Grad der Modularisierung und damit auch die Zahl der internen Beziehungen und Abhängigkeiten steigt, umso mehr steigt auch der Planungs-, Koordinierungs- und Lenkungsaufwand. Damit verliert sich der Vorteil durch kleine, überschaubare Teilaufgaben zusehends.

- Anzahl der externen Beziehungen und Abhängigkeiten
  Ein Projekt mit vielen externen Beziehungen und Abhängigkeiten hat sehr schwierige Entscheidungs- und Testmechanismen. Die Verknüpfungen und die Auswirkung von Entscheidungen sind jeweils zu berücksichtigen bzw. sogar Zustimmungen von externen Entscheidungsträgern einzuholen. Bei einem solchen Projekt kommt leicht ein demotivierendes Gefühl der Abhängigkeit und Hilflosigkeit auf.

- psychologische Wirkung verborgener Arbeitsprinzipien

  Im Alltag des Projektablaufs wirken im Team und bei den Kontaktpersonen eine ganze Reihe psychologischer Mechanismen, derer sich die wenigsten bewußt sind. Dies sind u.a.

  - Scheinwerfermodell,besagt, dass immer nur ein kleiner Ausschnitt der zu bearbeitenden Sache im Licht der Auf-

merksamkeit liegt. Die aufnehmbare und verarbeitbare Informationsmenge ist begrenzt ist und man konzentriert sich deshalb auf das scheinbar wesentliche.

Welche Wirkungen treten dadurch auf? Ausnahmen und Grenzfälle werden übersehen. Teile entfalten unübersehbare Nebenwirkungen. Korrekturen und Überarbeitungen bleiben bruchstückhaft, da nicht das ganze Wirkungsgefüge überblickt wird.

- Sparsamkeitsprinzip / Einfachheitsgrundsatz,
  s.o.
- Prägnanztendenz / Kategorienbildung,
  meint die Hypothese, dass man Regelmäßiges vorfindet und dass es sich lohnt, diese Ordnung aufzuspüren und auszunutzen.Dies führt dazu, dass neues in Schubladen (Kategorien) einsortiert wird. Vorstellungen (Vorabwissen, Hypothesen, Vorurteile, Theorien) über unsere Umwelt, die in unserem Wahrnehmungssystem vorhanden sind, bestimmen unsere Vorstellung von der RealitätWas kann passieren?Es kommt zum Versagen in außergewöhnlichen und abweichenden Situationen. Bei der Zerlegung von Aufgaben in wichtige und unwichtige Teile werden einflussreiche Nebenwirkungen übersehen.lineares Kausaldenken, bezieht sich auf die Hypothese, dass man es im Leben hauptsächlich mit linearen Kausalketten (Ursache-Wirkungs-Beziehungen) zu tun hat. Man nimmt an, dass, die gleichen Erscheinungen die gleichen Ursachen haben. Strenge Schlussfolgerungen erscheinen seriöser.

  Mögliche Auswirkungen:Man ist zufrieden, wenn ein Fehler / eine Ursache gefunden wurde. Man neigt zum Unterschlagen oder Übersehen logischer Bedingungen. Man vergisst, dass ein nichttriviales Projekt ein stark vernetztes System ist und übersieht Nebenwirkungen / Seiteneffekte.

- Überschätzung bestätigender Informationen, d.h.eine Hypothese wird nach wenigen bestätigenden Informationen für wahr erachtet.Dies führt dazu, dass·Tests nach wenigen erfolgreichen Fällen abgebrochen werden, Mitarbeiter sich mit einer schwachen Lösung zufrieden geben, nur weil sie augenscheinlich funktioniert, Alternativen nicht betrachtet werden.
- Eine vorgeprägte Ausrichtung des Denkens erschwert neue Lösungen, daErfahrungen aus früheren Versuchen in ähnlichen Situationen vorhanden sind, eine Gewöhnung und Mechanisierung durch wiederholte Anwendung auftritt, eine Gebundenheit an Werkzeuge und Methoden entstanden ist, Vorschriften vermutet werden, wo diese nicht gegeben sind (Verbotsirrtum), die Anziehungskraft des scheinbar nahen (Teil-) Ziels lockt, vorhandene Strukturen und Teillösungen die weitere Richtung bestimmen.

Diese psychologischen Mechanismen kann man nicht ausschalten aber man kann sie·erkennen (Vermittlung wirkender psychologischer Mechanismen), analysieren und beeinflussen (Selbsterkenntnis und Beratung) und akzeptieren (Grenzen der Beeinflussung),·dass der Mensch nicht fehlerfrei ist.Innovationsgrad des Projektes

Projekte haben einen unterschiedlichen Innovationsgrad. Es gibt Aufgaben (Projekt mit zumindest prinzipiell bekanntem Algorithmus; vorhandenen Erfahrungen bei der Bearbeitung ähnlicher Projekte) und Probleme (Projekt mit noch nicht bekannter Lösung; subjektivem/objektivem Neuland). Die oben betrachteten verborgenen Arbeitsprinzipien sind für Problem-Projekte wesentlich kritischer. Bei Aufgaben haben sie in der Regel sogar positive Auswirkungen.

2. Der Glaubenskrieg zwischen Charismatikern und Technokraten

Die folgende Betrachtung ist eine ganz persönliche Sicht auf die Projektmanagementlandschaft. Dabei wird mit Sicherheit manches überspitzt aber in manchen Diskussionen meint man, sich in einem solchen Glaubenskrieg zu befinden.
Als Charismatiker bezeichne ich Projektmanager, denen soziale Kompetenz wesentlich wichtiger ist als methodisch/organisatorische Kompetenz. Manche Vertreter dieser Richtung halten die methodisch/organisatorische Kompetenz sogar für verzichtbar. Sie berufen sich dabei gern auf den bekannten Ausspruch von Antoine de Saint-Exupéry, dass die Sehnsucht nach dem Meer wichtiger ist als der Bauplan des Schiffes. Sie vergessen aber bzw. kennen das Beispiel gar nicht, dass Supergaleeren, die durch lineare Vergrößerung bestehender kleinerer Modelle entstanden gleich beim Auslaufen sanken. Durch Visionen und entsprechende Führung wird aus psychologischer Sicht bei den Teammitgliedern die latent vorhandene Sehnsucht nach dem Ideal, nach dem verlorenen Paradies angesprochen. Dieses Streben ist ein wichtiger Motivationsschub, der viel bewirken kann. Wenn aber das Ideal nicht näher kommt und auch keine erfolgreichen Schritte der Annäherung erkennbar und erlebbar sind, dann verbrennt die Motivation wie der Treibstoff einer Feststoffrakete anstatt durch kontinuierliche Zufuhr von Treibstoff aufrechterhalten zu werden.
Den Gegenpol zu den Charismatikern bilden die Technokraten, die Vorgehensmodelle und PM-Tools über alles stellen. Für sie ist der Mensch, das Teammitglied, austauschbar. Ein Inserat in einer Projekt- oder Jobbörse genügt, um einen neuen optimal qualifizierten Mitarbeiter zu finden. Motivation ist selbstverständlich von dem Mitarbeiter selbst zu leisten. Die Technokraten vergessen, dass die Vorgehensmodelle und PM-Tools von Menschen angewandt werden. Sie übersehen bzw. unterschätzen die Probleme bezüglich Akzeptanz der Modelle

und PM-Tools, die durch Komplexität, Handhabbarkeit, Lernaufwand und mangelnde Ergebnisqualität entstehen.
Zwischen den beiden Gegenpolen Charismatiker versus Technokraten bewegt sich die Mehrzahl der Projektmanager. Sie handeln intuitiv oder bewusst im Sinne des folgenden Zitats aus einem Gedicht von Bert Brecht

Der junge Alexander eroberte Indien. Er allein?
Cäsar schlug die Gallier. Hatte er nicht wenigstens einen Koch bei sich?

Wenn sie agil sind, dann nehmen sie von jedem ein bisschen: Einen Löffel Vision, eine Messerspitze soziale Kompetenz aber Organisation, Methode und Technik nur eine Prise.
Dazwischen gibt es die relativ seltene Spezies der Pragmatiker mit theoretischem Background, die davon ausgehen, dass jedes Projekt eine speziell auf das Projekt zugeschnittene Ausprägung der Teamzusammensetzung und –führung, des Vorgehens und des Tooleinsatzes benötigt und die auch imstande sind, diese Ausprägung auf Basis ihrer Kenntnisse zusammen mit dem Team und unterstützt durch das übergeordnete Management zu leisten.

3. Projektmanagement und Toolunterstützung

Im Folgenden werden wichtigen Projektmanagementaufgaben und mögliche Tools gegenüber gestellt. Ein ‚Tool' kann aber auch ein Einsatz von Excel mit diversen Vorlagen oder Makros sein.

Tab 1.: Übersicht Aufgaben / PM-Tools

Aufgabe	**PM-Tool**	**Abdeckung/Bemerkung**
Planen		
Erarbeiten Verfahren für Projekt-,	• -	Checklisten / Muster

Aufgabe	PM-Tool	Abdeckung/Bemerkung
Team-, Qualitäts- und Konfigurationsmanagement		
Termine / Ressourcen / Kosten	• Projektplanung • Terminkalender	prinzipiell gut, Probleme mit Detaillierungsebenen
Führen / Steuern	• Simulation	weitgehend menschliche Tätigkeit auf der Basis der Controllinginformationen, der Informationstools und Simulationsergebnisse
Controlling		
Termine / Ressourcen / Kosten,	• Erfassung Arbeits-zeit etc. • Plan-Ist-Vergleiche	unterschiedlich komfortabel von Excel bis Spezialtool
Einhaltung der PM/QM/KM-Verfahren	• statische Analyse u.a. auf Einhaltung von Standards	weitgehend menschliche Tätigkeit außer statischer Analyse
Qualität	• Testsuiten	unterschiedlich maschinell unterstützte Teile des Testprozesses
Probleme / Changes	• CM-Tool, Tracker	unterschiedlich maschinell unterstützte Prozess-Teile
Konfigurationen / Versionen	• KM-Tool • Versionsmanager	unterschiedlich maschinell unter-

Aufgabe	PM-Tool	Abdeckung/Bemerkung
		stützte Prozess-Teile
Informieren		
Termine / Ressourcen / Kosten,	• Mailing • zugängliche Auswertungsfunktionen der Controlling-Tools • Projektoffice	
Einhaltung der PM/QM/KM-Verfahren	• Mailing • Projektoffice	
Qualität	s. Termine	
Probleme / Changes	s. Termine	
Konfigurationen / Versionen	s. Termine	
Entscheiden	• Simulation	menschliche Tätigkeit auf der Basis der Controllinginformationen, der Informationstools und Simulationsergebnisse

Zu den PM-Tools lassen sich folgende Bewertungen formulieren:

- Kein PM-Tool deckt alle Bereiche ab.
- Auch alle PM-Tools zusammengenommen decken nicht alle Bereiche vollständig ab.
- Es gibt keine Toolsuite eines Herstellers, die alle Bereiche abdeckt.

- Es ist schwierig, eine problemlos zusammenarbeitende Toolsuite aus den Produkten verschiedener Hersteller zusammenzustellen.
- Der Einarbeitungs- und Administrationsaufwand für PM-Tools kann erheblich sein.
- Ganz ohne PM-Tools geht es nicht.

(Achtung: Diese Bewertungen stellen die persönliche Auffassung des Autors dar.)

4. Tooleinsatz erfolgreich organisieren

Bei der Durchführung eines Projekts stellen die Projektmanagementtools nur Hilfsmittel zur Beherrschung der Komplexität dar. Sie sind nicht der Projektzweck (auch wenn das manchmal so scheint) sondern sie sollen die Erreichung der Projektziele unterstützen. Ein erfolgreicher Tooleinsatz ist also auch immer ein ‚angemessener' Tooleinsatz. Die Angemessenheit kann sich dabei auf das PM-Tool oder auf die verwendeten Features des PM-Tools beziehen. In der Praxis nützt diese Aussage aber nur etwas, wenn es gelingt, die ‚Angemessenheit so zu beschreiben, dass der Projektmanager X mit hoher Sicherheit für sein Projekt diesen angemessenen Einsatz definieren kann.

- Versuch einer Definition der Angemessenheit

  Ein Tooleinsatz in einem Projekt ist angemessen, wenn die für den effektiven Tooleinsatz notwendigen Aktivitäten wie Schulungsmaßnahmen, Erstellung von Benutzeranweisungen, Erstellung von Vorlagen und Mustern, Administration der PM-Tools, Erfassung der Eingabedaten, Erzeugung von Auswertungen und die Einbindung von Auswertungen in Projektberichterstattungen 5 Prozent der Gesamtprojektaufwände nicht überschreiten.

  Achtung: Die Grenze von 5 Prozent kann je nach Firma und Projekttyp schwanken.

- Feststellung der Angemessenheit

  Beim erstmaligen Einsatz eines PM-Tools werden in der Regel keine Erfahrungen über Probleme und Aufwände vorliegen. Es ist auch relativ selten, dass die Aufwände für die PM-Tools sauber geschätzt werden. Meistens vertraut man den Versprechungen der Toolhersteller. Der Wahrheitsgehalt solcher Versprechungen ist aber nicht höher als bei beliebigen anderen Softwareherstellern.

  Es ist also notwendig, bei einem geplanten Tooleinsatz das Procedere einer normalen Softwareauswahl durchzuführen bzw. mindestens eine Checkliste durchzugehen, die eine Auflistung möglicher Aktivitäten bei Einführung und Einsatz von Software enthält. Trifft eine Aktivität zu, so muss sie mit Dauer/Durchführung, Anzahl der Personen und Anzahl der Durchführungszeitpunkte bewertet werden. Die dabei entstehenden Zahlen erschrecken manchen Projektmanager und lassen ihn die geplante Einsatzvariante noch mal kritisch überdenken. Eine dabei aber zu checkende Variante ist selbstverständlich auch die Variante ‚kein PM-Tooleinsatz'.

  Natürlich ist diese Bewertung mit einem gewissen Aufwand verbunden. Wenn das aktuelle Projekt von der Komplexität einem bereits durchgeführten ähnelt, so kann man auch auf Erfahrungen zurückgreifen und Bewährtes übernehmen. Betritt der Projektmanager aber von der Komplexität her Neuland, so ist ein oben beschriebener Check dringend zu empfehlen.

- Wie vermittelt man dem Team das Gefühl der Angemessenheit?

  Die Feststellung der Angemessenheit durch die Projektleitung reicht aber nicht aus. Ein Tooleinsatz wird nur erfolgreich, wenn es gelingt, die Akzeptanz durch das Projektteam zu erreichen. Das Team muss also nicht nur den Check der Projektleitung nachvollziehen können sondern es muss in seiner Mehrheit ein Gefühl der Angemessenheit haben.

Wann stellt sich das Gefühl der Angemessenheit ein?

- Die Komplexität der PM-Tools ist deutlich niedriger als die Komplexität der Entwicklungsumgebung.
- Die PM-Tools sind weitgehend intuitiv benutzbar (kein 50-seitiges Benutzerhandbuch).
- Sie lehnen sich in der Benutzeroberfläche an vertraute Produkte oder bekannte Standards an.
- Die Benutzung der einzelnen PM-Tools und deren Schnittstellen sind möglichst abgestimmt. Doppelerfassungen kommen nicht vor.
- Für jedes Teammitglied reicht ein Toolausschnitt, der seiner Rolle entspricht aber das gesamte PM-Tool ist für jeden offen (Datenschutz bei Personendaten beachten !).
- Jedes Teammitglied kann die Auswertungen, die die Projektleitung über seinen Arbeitsbereich durchführen kann, ebenfalls durchführen und möglichst auch als erstes.

- Vorschläge für vorteilhaften Tooleinsatz je nach Projektgröße

  Im Folgenden werden relativ grob einige Beispiele mit Vorschlägen aufgezeigt. Selbstverständlich ist eine weitere Detaillierung möglich aber dabei besteht die Gefahr der Schematisierung. Deshalb wird generell das oben dargestellte Verfahren des Tool-Checks empfohlen. Sind in einer Firma PM-Tools und Tooleinsatz generell vorgeschrieben, so kann vielleicht doch an den zu verwendenden Features optimiert werden.

Tab 2.: Tooleinsatz nach Projektgröße

Projektgröße	PM-Tools	Bemerkung
klein (bis 4 Mitarbeiter)	MS Office oder vergleichbares Bürosystem oder Light-Version ei-	Konfigurations- und Testunterstützung im vollen

Projektgröße	PM-Tools	Bemerkung
	nes PM-Tools	Umfang
mittel (5-15 Mitarbeiter)	Planungsunterstützung durch MS Project oder vergleichbares Planungstool, Erfassung und Controlling über Excel oder ähnliches möglich Projektoffice wenn verfügbar	Konfigurations- und Testunterstützung im vollen Umfang Projektoffice wenn verfügbar
groß (ab ca. 16 Mitarbeiter)	volle Toolunterstützung	Projektoffice, Projektplanung, Terminkalender, Erfassung Arbeitszeit, Plan-Ist-Vergleiche, QM, CM, KM

- Vorschläge für Maßnahmen zur psychologischen und organisatorischen Begleitung des Tooleinsatzes

  Basierend auf Erfahrungen aus vielen erfolgreich durchgeführten Projekten sollten u.a. folgende psychologischen ‚Argumente' im Team genutzt werden.

  - Die Komplexität der PM-Tools ist deutlich niedriger als die Komplexität der Entwicklungsumgebung.
  - PM-Tools brauchen wir nur so viel, wie wir uns gerade verständigt haben.
  - Wir kennen alle Beispiele, wo ein schlechter Tooleinsatz zum Misserfolg des Projektes beigetragen hat.
  - Es gibt schöneres als PM-Tools aber wir benutzen alle die PM-Tools. Auch der Projektmanager ist keine Ausnahme.
  - Wenn wir die PM-Tools ordentlich einsetzen, dann haben wir immer einen Wissensvorsprung vor dem übergeord-

neten Management. Wir wissen als erste wo wir gut sind oder wo wir noch nicht so gut sind.

Organisatorisch berücksichtigt man ebenfalls ganz einfache Rezepte, so z.B.

- Bei der Vorstellung / Einweisung der PM-Tools ist möglichst der Projektmanager dabei und stellt sich mit seiner Autorität dahinter.
- Für jedes PM-Tool gibt es einen definierten Ansprechpartner, der sich auch wirklich zumindest besser als der Durchschnitt auskennt.
- Wenn einmal Regeln definiert wurden, dann ändert man sie nur im Notfall. Eine suboptimale Regel ist meistens besser als ständig geänderte Regeln. Echte Fehler werden allerdings schnellstens korrigiert.

Berücksichtigung psychologischer Aspekte bei der Erhaltung der Codequalität in Wartungsprojekten

In der Softwarewartung findet man oft folgende Problemsituation vor:

Das·Management ist mit der Wartung der Legacy-Anwendungen unzufrieden, da die Wartung zu viel kostet, die Wartung fehleranfällig ist, die Wartungsmitarbeiter nicht gut motiviert sind und der Einsatz von Tools die Situation nicht wesentlich verbessert hat.

Was kann durch das Management und die Mitarbeiter getan werden um Verbesserungen zu erreichen? Eventuelle Maßnahmen sollen aber möglichst wenig kosten.

Wir werden nun versuchen zu zeigen, dass durch Berücksichtigung psychologischer Aspekte bei der Softwarewartung einiges erreicht werden kann.

1. Wartung aus psychologischer Sicht

Wenn die ‚weichen' Faktoren der Softwarewartung betrachtet werden, so fallen folgende Dinge auf

- Softwarewartung ist ein weitgehend diskontinuierlicher Prozeß, i.a. ohne definiertes zeitliches Ende und meistens nur mit einem diffusen allgemeinen Ziel, wie 'fehlerfreie Abarbeitung unter Produktionsbedingungen'.
- Softwarewartung erfolgt selten im Team bzw. nur in kleineren Teams. Es fehlt die intensive Kommunikation, das gemeinsame Teamgedächtnis und das vollständige informelle Modell des Projekts/Produkts.
- Die Verantwortung des einzelnen gilt für größere Systemteile oder sogar ganze Systeme.
- Fehler haben scheinbar immer eine große Tragweite (Katastrophe) und müssen eigentlich immer unter Zeitdruck behoben werden.
- Änderungen und Erweiterungen sind oft schwieriger als Neuentwicklungen aber die Wartung besitzt bei vielen IT-Managern einen geringen Anerkennungswert. Wie kann ein Entwickler bei der Wartung Befriedigung finden?
- Die Ziele der Wartungsmannschaft und der Nutzer sind nach einiger Zeit nicht mehr identisch. Es gibt ständig Konflikte zum Beispiel wegen Änderungswünschen.

2. ausgewählte Qualitätsziele der Softwarewartung

Von den vielen möglichen Qualitätszielen für die Softwarewartung werden nur folgende ausgewählten betrachtet. Ihre Gemeinsamkeit ist, dass sie stark subjektiv bzw. psychologisch beeinflusst sind.

- Codequalität (Korrektheit/Fehlerfreiheit, Effizienz der Lösung, Zuverlässigkeit/Robustheit, Wartbarkeit / Verständlichkeit, Wiederverwendbarkeit)

- Prozessqualität (Termineinhaltung, Einhaltung des geplanten Aufwands, Sicherung der Funktionalität)

3. psychologische Fallen und ihre Hintergründe bei der Erhaltung der Codequalität

Bei der Durchführung von Änderungen und Erweiterungen an vorhandenem Code sind für den Mitarbeiter psychologische Fallen aufgestellt, die einer Erhaltung bzw. Verbesserung der Codequalität entgegenwirken. Diese Fallen beruhen auf Besonderheiten unseres Denkens, die wir ererben oder erwerben. Es wirken im Wartungsprozess

- angeborene Wahrnehmungs-, Denk- und Verhaltensweisen und erworbene Fähigkeiten und Kenntnisse,
- verborgene Arbeitsprinzipien (z.B. Scheinwerfermodell, Sparsamkeitsprinzip / Einfachheitsgrundsatz, Prägnanztendenz / Kategorienbildung, lineares Kausaldenken, Überschätzung bestätigender Informationen).

Diese haben bestimmte Auswirkungen wie z.B. Ausnahmen und Grenzfälle werden übersehen, Korrekturen und Überarbeitungen bleiben bruchstückhaft, Versagen in außergewöhnlichen und abweichenden Situationen und übersehen von Nebenwirkungen/Seiteneffekten.

4. ‚Immunisierung' der Mitarbeiter

Eine vollständige ‚Immunisierung' kann wegen unserer nicht fehlerfreien menschlichen Natur nicht gelingen. Relevante Verbesserungen sind aber mit relativ geringen Kosten möglich. Dazu bedarf es ausreichender Motivation der Mitarbeiter, Änderungswillen des Mitarbeiters sowie Information und Unterstützung durch das Management. Dem Mitarbeiter muss man helfen

- beim Erkennen prinzipieller Probleme durch die Vermittlung wirkender psychologischer Mechanismen),

- bei der Selbsterkenntnis (s. auch Fragebogen zum Selbstassessment eines Wartungsmitarbeiters),
- mit Unterstützung durch Management und Team sowie durch Tools und Checklisten (z.B. Checkliste ‚Erhaltung der Codequalität'),
- durch Würdigung seiner erreichten Fortschritte.

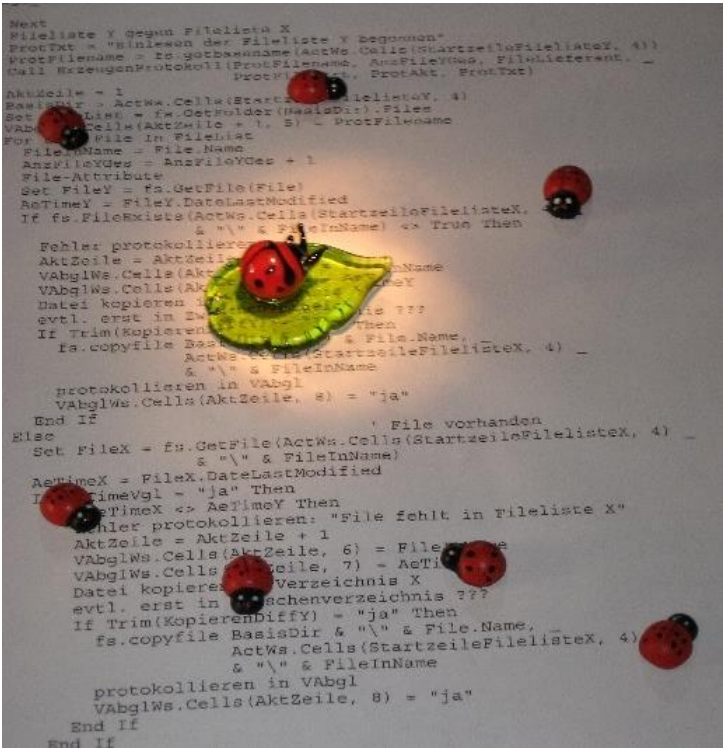
Traumzeit: Psyche versus Ratio in der Softwareentwicklung

Die Traumzeit verbrachte ich auf der Couch eines Psychologen.
Dieser erklärte mir, dass die Berücksichtigung psychologischer Aspekte (psychologische Fallen und Stolpersteine) bei der Softwareentwicklung ein echtes oft verkanntes Problem darstellt.

Der Drache der Fehlerwahrscheinlichkeit,
das Gespenst der schlechten Qualität
und das Fehlerteufelchen
vor dem Tor
zum Universum der Softwareentwicklung

Das psychologische Grundproblem des Softwareentwicklers

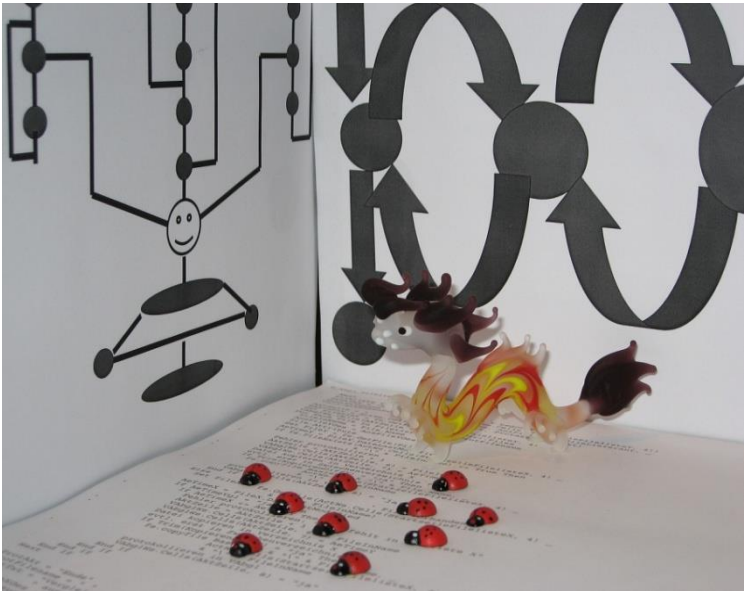
Er lebt in einer Welt, in der weder sein Hintergrundwisssen noch sein Denkapparat von der Evolution zum Zwecke des Programmierens selektiert oder gar konstruiert worden sind. In prähistorischen Zeiten lebenserhaltende Verhaltensweisen (Jagd- und Fluchtreflexe) sind beim Programmieren eher untauglich.



Falle 1: Scheinwerferphänomen

Es liegt immer nur ein kleiner Ausschnitt im Licht der Aufmerksamkeit.
Wirkungen u.a. :

- Ausnahmen und Grenzfälle werden übersehen;
- Funktionen entfalten unübersehbare Nebenwirkungen



Falle 2: Sparsamkeitsprinzip

Es wird auf Gesetzmäßigkeiten aus einer begrenzten und oft sehr kleinen Anzahl von Beobachtungen geschlossen.

Wirkungen u.a.:

- Kopieren von vorhandenem (eigenem) Code und abändern, ohne alle Seiteneffekte zu überprüfen;
- ein bestätigender Test reicht

Falle 3: Einfachheitsgrundsatz

Es werden zuerst (und evtl. nur) die einfachsten Vermutungen überprüft.

Wirkungen u.a.:

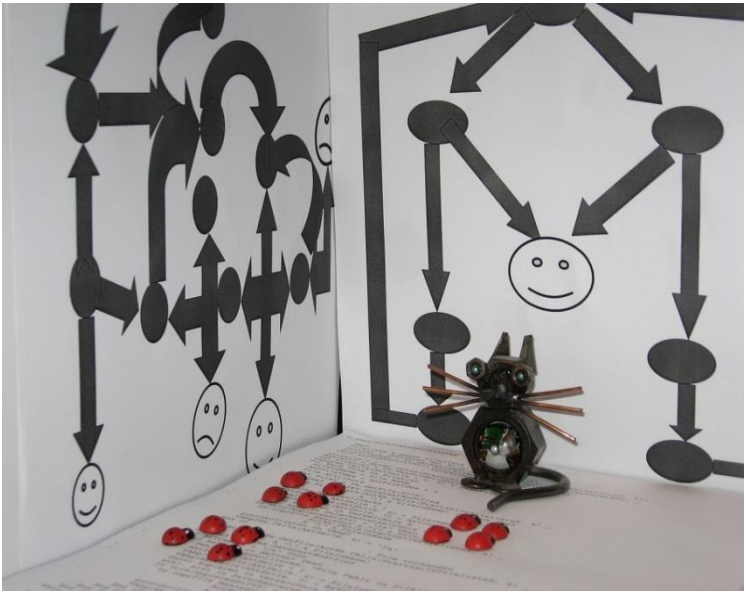
- Annahme: weil es schon immer so war, wird es auch jetzt so sein;
- Verfahrensweisen, die in einer alten Umgebung gut funktionierten werden unkritisch in eine neue übernommen

Falle 4: Prägnanztendenz

ist die menschliche Annahme, dass man Regelmäßiges vorfindet und dass es sich lohnt, diese Ordnung aufzuspüren und auszunutzen

Wirkungen u.a.:

- Versagen in außergewöhnlichen und abweichenden Situationen;
- bei Zerlegung von Aufgaben in wichtige und unwichtige Teile werden einflussreiche Nebenwirkungen übersehen



Falle 5: Kategorienbildung

Neues wird in Schubladen (Kategorien) einsortiert.
Vorurteile bestimmen unsere Vorstellung von der Realität.

Wirkungen u.a.:

- hereinfallen auf einen bekannten Fehler;
- übersehen, dass ein Wertebereich z.B. auch negativ sein kann

Falle 6: lineares Kausaldenken, d.h. Annahme dass

- man es hauptsächlich mit linearen Kausalketten zu tun hat;
- die gleichen Erscheinungen die gleichen Ursachen haben;
- strenge Schlussfolgerungen seriöser sind

Wirkungen u.a.:

- zufrieden, wenn ein Fehler / eine Ursache gefunden wurde;
- übersehen von Nebenwirkungen / Seiteneffekten

Falle 7: Überschätzung bestätigender Informationen

Annahme wird nach Teil-Bestätigungen als wahr erachtet.

Wirkungen u.a.:

- Test wird nach wenigen erfolgreichen Fällen abgebrochen;
- Entwickler gibt sich mit schwacher Lösung zufrieden, nur weil sie augenscheinlich funktioniert;
- Alternativen werden nicht oder nur oberflächlich betrachtet

Falle 8: Einstellungen

vorgeprägte Ausrichtung des Denkens

u.a. durch :

- Erfahrungen aus früheren Versuchen,
- Gewöhnung durch wiederholte Anwendung,
- Gebundenheit an Werkzeuge und Methoden,
- Vermutung von Vorschriften, wo diese nicht gegeben sind (Verbotsirrtum),
- Anziehungskraft des scheinbar nahen (Teil-) Ziels

‚Immunisierung'
gegen psychologische Fallen

- erkennen (Kenntnis wirkender psychologischer Mechanismen)
- analysieren (Selbsterkenntnis und Beratung)
- trainieren (bewusstes Hinterfragen der eigenen Handlungen)
- akzeptieren (Grenzen der ‚Immunisierung')

Grenzen der ‚Immunisierung' aus psychologischer Sicht

- Fähigkeit und Willen zur Selbsterkenntnis
- Fähigkeit und Willen zur Änderung
- Tendenz zum Rückfall in alte Vorgehensweisen

das ultimative Instrument des Softwareentwicklers

Wenn alles nicht mehr hilft, so gibt es am Schluß noch den Blick in die Kristallkugel. Sie beantwortet bei richtiger Handhabung auch die Fragen, die bisher nicht beantwortbar waren.

Nur wer die richtigen Fragen stellt
bekommt auch die notwendigen Antworten.

Station 5: Über den Pass der Abnahmekontrolle

Bei der Abnahme findet ein Verantwortungsübergang von den Entwicklern zu den künftigen Nutzern statt.

Spezifika des Projektmanagements bei der fachlichen Abnahme

1. Stellung der fachlichen Abnahme im Entwicklungsprozess

Die fachliche Abnahme nimmt im Softwareentwicklungsprozess de facto eine Zwitterstellung ein. In ihr findet ein Verantwortungsübergang von den Entwicklern zu den künftigen Nutzern statt. Deshalb kann man in verschiedenen Firmen und Organisationen unterschiedliche Handhabungsweisen der fachlichen Abnahme beobachten, von der Abnahme durch ein Teilteam des Projekts auf der Basis der definierten Anforderungen bis hin zur Abnahme durch ein separates Team aus dem Bereich der künftigen Nutzer auf der Basis von vordefinierten Anforderungen und aktuellen Anforderungen der Geschäftsprozesse. Zwischen beiden Extremen gibt es beliebige Mischformen. Wichtig ist, dass man sich bei der konkreten Gestaltung der Abnahme über Vor- und Nachteile der gewählten Lösung im Klaren ist und die Spezifika der Abnahme nicht vergisst:

- Ziel: eine Entscheidung über Abnahme des Systems sowie Einsatz in Produktion herbeiführen

  Dafür müssen zu Beginn der fachlichen Abnahme die Entscheidungs-(Abnahme-) Kriterien definiert sein oder werden. Die Abnahmekriterien aus den Projektanforderungen sind durch das Abnahmeteam nochmals kritisch zu sich-

ten, zu gewichten und gegebenenfalls zu ergänzen. In von externen Auftragnehmern durchgeführten Projekten existieren mindestens zwei Betrachtungshorizonte: vertraglich fixierte Abnahmekriterien gegenüber dem Auftragnehmer (falls vorhanden) und Abnahmekriterien der Fachbereiche (falls definiert), d.h. vor Beginn der Abnahme ist eine Identifizierung, Abgleich und eventuell angleichende Korrektur erforderlich. Mögliche Alternativen wie Teilabnahmen, schrittweise Abnahmen oder eingeschränkte Abnahmen sollten dargestellt werden. Dabei müssen die Konsequenzen aus Entscheidungen, die zu einer ‚unvollständigen' Abnahme führen, vorher transparent gemacht werden. Solche Konsequenzen können u.a. fehlende Unterstützung bestimmter Teile der fachlichen Prozesse, erhöhter manueller Aufwand der Fachbereiche beim produktiven Einsatz der abzunehmenden Anwendung oder weiterarbeiten mit Teilen von Alt- oder Vorgängeranwendungen sein. Die im Ergebnis erarbeitete Fassung der Abnahmekriterien ist unbedingt durch die Verantwortlichen der Fachbereiche zu bestätigen.

- Umfang: Die fachliche Abnahme umfasst Integrations-, Funktions- und Systemtests und sollte auch Tests der (betroffenen) Organisation in den Fachbereichen beinhalten.

  Die korrekte Durchführung der einzelnen Funktion in einer für den künftigen Benutzer vertretbaren Weise in einer akzeptablen Zeit ist die Grundlage einer erfolgreichen fachlichen Abnahme. Damit sind aber auch schon die ersten Probleme sichtbar. Die ‚korrekte Durchführung' ist ein hartes Kriterium, dessen Erfüllung durch Tests mit entsprechendem Aufwand in der Regel hinreichend gut nachgewiesen werden kann und das auch zwischen Entwicklung und Fachbereich weitgehend unstrittig ist. Für den Nachweis der ‚akzeptablen Zeit', die zwar vorgegeben sein sollte aber in der Praxis sehr oft so lange wie möglich recht schwammig definiert wird (z.B. ‚nicht langsamer als bei der Altanwendung' obwohl Arbeitsweise der Altan-

wendung und der neuen Anwendung teilweise nicht vergleichbar sind) müssen Lasttesttools genutzt und dafür realistische Szenarien vom Fachbereich vorgegeben werden. Die ‚vertretbare Weise' (Nutzerfreundlichkeit, Handhabbarkeit) hängt wesentlich von den Aufgaben und Zielen des jeweiligen Nutzers ab und kann starken Schwankungen in Abhängigkeit von personellen Änderungen unterworfen sein. Hier ist Streit zwischen Entwicklung und Fachbereich vorprogrammiert.

Unbeschadet von der Qualität der einzelnen Funktion ist die Effizienz der gesamten Geschäftsprozesse letztlich für die Fachbereiche wichtiger, da sie über den Erfolg des Einsatzes entscheidet. Das Abnahmeteam muss und wird demzufolge diese Gesamtsicht auch prüfen, egal ob sie vorher definiert war oder nicht.

- Stellung: Voraussetzung für Freigabe eines neuen Release für die Produktion

  Ohne ausreichende fachliche Abnahme (möglichst mit maschinell unterstütztem Regressionstest) darf es keine Freigabe geben. Im Sinne der Zielbetrachtungen (Teilabnahmen, schrittweise Abnahmen oder eingeschränkte Abnahmen) aus dem ersten Spiegelstrich kann nur der künftig nutzende Fachbereich bzw. ein von der Unternehmensleitung z.B. bei der Abnahme von Webanwendungen beauftragtes evtl. auch externes Abnahmeteam entscheiden, was ausreichend ist. Die objektiv existierenden widerstreitenden Interessen zwischen Entwicklung und fachlichem Nutzer können nicht negiert sondern müssen zu einem vertretbaren Kompromiss geführt werden. Hierbei ist eine Vertretung der Entwicklung im Abnahmeteam hilfreich.

- Organisation: Je nach Projektgröße, Formalisierung des Entwicklungsprozesses, internem oder externem Auftragnehmer für die Entwicklung, Kritikalität des Projektes usw. ist die Abnahme unterschiedlich organisiert.

Dabei reicht die Spanne von ‚integriert in das Entwicklungsprojekt' bzw. ‚eigenes Teilprojekt' (vorzugsweise bei intern durchgeführten Projekten) bis zu ‚eigenes Projekt Abnahme'. Bei ersteren besteht das Risiko der Verantwortungsverwischung und der Abnahme um jeden Preis nach dem Kriterium ‚wird schon funktionieren, alles halb so wild'. Beim zweiten wird manchmal das bei aller notwendigen harten Auseinandersetzung erforderliche Maß an Kooperations- und Kompromissbereitschaft vergessen.
Die Durchführung der Abnahme muss in einer eigenen produktionsäquivalenten oder zumindest produktionsnahen Abnahmeumgebung erfolgen. Auf die Fragen der Sicherung definierter Entwicklungsstände für die Abnahme wird hier nicht eingegangen.

2. Verantwortlichkeiten und Zusammensetzung des Abnahmeteams

Im Team ist gemischtes Know-how notwendig (fachlich, Testmethodik, IT). Die Know-how-Mischung muss aber nicht nur im Team sondern in gewissem Maße auch in den Teammitgliedern vorhanden sein. Dabei ist bei den meisten Teammitgliedern der Schwerpunkt auf fachliches Know-how, verbunden mit einem bestimmten Maß an Test-Know-how zu legen.
In der Praxis sind mehrere Verantwortungsmodelle für die Abnahme anzutreffen

- Verantwortung für die Abnahme liegt beim Fachbereich, da dieser als Auftraggeber für das Projekt fungiert,
- Verantwortung liegt bei IT, da der Fachbereich gegenüber externen Partnern zu schwach ist,
- Verantwortung ist gemischt (Fachbereich, IT),
- Verantwortung liegt bei einem von der Unternehmensleitung z.B. bei der Abnahme von Webanwendungen beauftragten unabhängigen Abnahmeteam (evtl. auch extern).

Mischverantwortungen haben sich nicht bewährt. Extrem wichtig für die Akzeptanz der Abnahme ist eine fachliche Verantwortung durch spätere Nutzerbereiche. Eine Trennung der vertraglichen Verantwortung (IT) und fachlichen Verantwortung (Fachbereich) ist aber möglich und durchaus praktikabel.

3. Konfliktherde und Brückenschlagstrategien

Die informellen Gräben zwischen Fachbereich und IT können m.E. nach nicht zugeschüttet sondern nur überbrückt werden, da

- Entwickler und spätere Nutzer der Software eine unterschiedliche ‘Sprache’ sprechen,
- gleiche Begriffe in beiden Sprachräumen unterschiedliche Bedeutungen haben können,
- verwendete Begriffe im jeweiligen Sprachraum mehrdeutig sein können,

Es findet auch bei der Abnahme ein permanenter ‚Übersetzungsprozess’ mit den einer ‚Übersetzung’ immanenten Problemen statt, der u.a. Lern-, Akzeptanz- und Kooperationsbereitschaft auf beiden Seiten verlangt. Extrem wichtig ist es, qualifizierte, fachlich anerkannte spätere Nutzer, d.h. Nutzer mit Kompetenz, Sachkunde und Bereitschaft zur gründlichen Analyse, zu finden, die das erarbeitete System in der Abnahme beurteilen.

Folgende wesentlichen Konfliktherde sind zwischen Fachbereichen und IT im Verlaufe der Abnahme latent vorhanden:

- unterschiedliche Bewertung von Systemreaktionen,
- unterschiedliche Ansichten zu Realisierungsvarianten,
- unterschiedliche Beurteilung von Notwendigkeiten und Aufwänden.

Auftretende Konflikte müssen im bzw. vom Abnahmeteam klar benannt und bezüglich ihrer möglichen Auswirkungen bewertet werden.

Folgende Beispiele zeigen Möglichkeiten für Brückenschlagstrategien zwischen IT und Fachbereichen in der fachlichen Abnahme auf

- Betonung des gemeinsamen Ziels: Abnahme,
- frühzeitige Einbeziehung der Fachexperten in der Entwicklung vor der Abnahme zur Sicherung einer fachlichen Kontinuität und eines reifenden Verständigungsprozesses (theoretisch wundervoll und unbedingt zu fordern aber in der Praxis aus verschiedensten personellen Gründen, zumindest bei Projekten, die länger als ein Jahr dauern, oft nicht durchzuhalten),
- Sicherung einer sachlichen Streitkultur mit organisiertem Austausch von Standpunkten und Beispielen sowie der Verdeutlichung von Konsequenzen bestimmter Entscheidungen,
- Würdigung des Einsatzes und gelungener Aktionen auf beiden Seiten.

Es ist wichtig, auf bewährte (alte und neue) Hilfsmittel zur besseren Verständigung während der Abnahme zurückzugreifen:

- Das (hoffentlich vorhandene) Projektlexikon wird vor der Abnahme aktualisiert. Wenn bisher kein Projektlexikon existierte, so wird es angelegt. Unbedingt vermieden werden sollte eine Änderung von schon definierten Begriffen (Konsistenzprobleme!).
- Der abgestimmte fachliche Sprachschatz wird auch benutzt, vor allem von der Leitung des Abnahmeteams.
- Gewollte Abweichungen zu bisherigen Verfahren und Ergebnissen (z.B. Rundungen, Klassifizierungen, Ergeb-

niszuordnungen) sind beschrieben, vom Management sanktioniert und dem Abnahmeteam jederzeit zugänglich und können damit bei der Abnahme berücksichtigt werden.

- Mindestens ein Entwickler/begleitender Nutzer sollte Mitglied des Abnahmeteams sein, um das ‘informelle Teamgedächtnis’ des Entwicklungsteams nutzen zu können.
- Zur Entschärfung von Streitsituationen, zur Vermittlung und Verdeutlichung unterschiedlicher Standpunkte, für die Erarbeitung gemeinsamer Lösungen und Strategien empfiehlt sich zumindest in größeren Projekten die ständige Mitwirkung eines Coachs/Moderators/neutralen Prozessgutachters, der allerdings fast eine ‚Quadratur des Kreises’ versuchen muss, indem er menschlich akzeptiert wird, über hinreichende Kenntnisse der Fachlichkeit, des Entwicklungsprozesses incl. der Möglichkeiten der verwendeten Entwicklungsumgebungen und Datenbanken und last but not least der Testmethodik verfügt und diese auch vermitteln bzw. zur Förderung der Verständigung effektiv einsetzen kann.

4. Fachbereich und Abnahmetest

Abnahme eines Projektes ist in der Regel nicht die vorherrschende Tätigkeit der Fachbereichsmitarbeiter des Abnahmeteams. Hieraus resultieren auch unterschiedliche Kenntnisse zur Durchführung von Abnahmetests. Es ist nützlich, vor Beginn der Abnahme durch Befragung oder Self-Assessment das vorhandene Test-Know-how zu ermitteln.
Die durch die Analyse der Assessments/Befragungen gewonnenen Erkenntnisse müssen den Teammitgliedern anonymisiert zur Verfügung gestellt und Schlussfolgerungen für die Vorbereitung und Durchführung der Abnahme gezogen werden. Dabei darf man die subjektiven Grenzen solcher Erkenntnisse nicht vernachlässigen. Trotzdem ist der Realitätsgehalt m.E. nach ausreichend und die Ergebnisse, speziell für das

Management, oft erstaunlich. Diese Ergebnisse sind nur bedingt für weitere Abnahmen nutzbar, da durch personelle Teamveränderungen oder im Ergebnis von Erfahrungen/Lernprozessen der Stand schnell veraltet. Es empfiehlt sich also, z.B. zur Einschätzung des Erfolges bisheriger Erfahrungen/Lernprozesse bzw. vor jeder signifikant neuen Abnahme (d.h. nicht bei kurz aufeinanderfolgenden Versionen eines Systems), eine erneute Durchführung der Assessments / Befragungen.
Unabhängig vom konkreten Projekt und vom jeweiligen Know-how kann man Unterschiede in den Sichtweisen der Entwickler und der Fachbereiche feststellen. Eine nähere Betrachtung zeigt psychologisch und stellungsmäßig bedingte Ursachen auf. Häufige und wichtige Unterschiede sind

- Entwickler: Hersteller-/Eigentümer-Bewusstsein mit Stolz auf die Projektleistung

  Hieraus resultieren Verteidigungshaltungen und Kränkung durch Kritik.

- Fachbereich: Käufer-/Nutzer-Bewusstsein mit kritischer Haltung ‚bringt es, was ich erwarte und wofür ich bezahle ?'

  Hieraus resultieren u.a. aggressives Vertreten von Prüfergebnissen und Überbewertung von Mängeln, um Änderungen/Nachlässe usw. zu erreichen.

- Entwickler: Blick von Innen (Detailkenntnisse von Realisierungsteilen und technischen Strukturen)

  Dies bedingt mit Vorliebe Diskussionen um technische Details und ruft teilweise Arroganz gegenüber den Fachbereichen hervor, die ‚nur' benutzen und das im Sinne des Entwicklers vielleicht sogar uneffektiv und unsauber. Es wird manchmal nicht verstanden, worin eigentlich fachliche Probleme liegen.

- Fachbereich: Blick von Außen (keine Detailkenntnisse der Anwendung aber Detailkenntnisse der Prozesse, die die Anwendung unterstützen soll)

  Dies bedingt mit Vorliebe Diskussionen um fachliche Notwendigkeiten und die mangelnde Unterstützung durch die Anwendung und ruft teilweise Unverständnis gegenüber den Problemen der Entwickler hervor, die ‚sich doch ein Beispiel an Excel' nehmen und aus ihrem technischen Glashaus herauskommen sollen. Es wird manchmal nicht verstanden, worin eigentlich technische Probleme liegen.

- Entwickler: technisch/technologische Sicht auf Stellung der Anwendung in der Unternehmens-IT

  Hieraus ergeben sich u.a. Wahrnehmungen der Bedeutung der Anwendung aus Sicht der technischen Abhängigkeit anderer Anwendungen von ihr, aus Sicht des technologischen Niveaus der Anwendung und des für ihren Betrieb zu erbringenden technischen Aufwands.

- Fachbereich: fachlich/betriebswirtschaftliche Sicht auf Stellung der Anwendung zur Unterstützung der Unternehmensgeschäfte

  Hieraus ergeben sich u.a. Wahrnehmungen der Bedeutung der Anwendung aus Sicht der fachlichen Abhängigkeit der Geschäftsprozesse von ihr, aus Sicht des Niveaus der Unterstützung der Geschäftsprozesse durch die Anwendung, der Benutzerfreundlichkeit und des für ihren Betrieb zu erbringenden finanziellen Aufwands.

Die Unterschiede in den Sichtweisen können durch die kontinuierliche Einbindung der Fachbereiche in die Entwicklung zwar abgemildert aber nicht beseitigt werden.

Die Art der Durchführung von fachlichen Abnahmen hat auch Bezugspunkte zu Firmengrößen, Firmenkulturen und Projektarten. In größeren Firmen und Institutionen ist der Abnahmeprozess meist stärker reglementiert und formalisiert. Für klei-

nere Firmen empfiehlt sich hier die Anwendung eines Minimalregelwerks. Unabhängig von der Firmengröße werden aber die eigentlichen Abnahmeentscheidungen bei nicht fehlerfreier bzw. mängelfreier Software (und dies ist bekanntermaßen der absolute Regelfall) in einem Interessengeflecht getroffen, zu denen das Abnahmeteam nur die Entscheidungsvorlage liefert.
Die durch die Assessments/Befragungen aufgedeckten bzw. bereits bekannten Wissensdefizite der Fachbereiche zum Test können die qualifizierte Durchführung einer Abnahme in Frage stellen. Wenn die Fachbereiche nicht verstehen, wie man Teststrategien entwickelt und umsetzt, Testfälle konstruiert und welche Bedeutung Testabdeckungen haben und wie man sie erreicht, dann kann keine verlässliche Aussage über die Qualität der Abnahme und damit über die Qualität der Anwendung / des Produkts gemacht werden.
Auf der Basis des ermittelten Wissensstandes rentiert sich die Umsetzung eines Konzepts zur Testqualifizierung (interne Workshops unterschiedlichster Länge, abhängig von zu vermittelnden Inhalten und Teilnehmerzahlen). Hauptziel muss es sein, in einer vertretbaren Zeit mit vertretbarem Aufwand das Wissen über und die Sensibilisierung für bestimmte für die Abnahme wichtige Testmethoden und Vorgehensweisen zu erhöhen. Bei mehrfach mit Erfolg durchgeführten Workshops enthielten diese u.a. die Themen ‚Was kann/soll/muss getestet werden?', ‚Wie bereitet man einen Test vor?', ‚Wie wird getestet?', ‚Wie wertet man einen Test aus?' und ‚Rolle von Hilfsmitteln (Tools, Werkzeuge usw.)'.
Auch bei einem Abnahmeprojekt bestätigt sich, dass die Veränderung der Denkweisen und Arbeitstechniken einen angemessenen Erfahrungszeitraum bei ständiger Unterstützung benötigt. Dabei werden Veränderungen nur in einem bestimmten Schrittmaß akzeptiert und manchmal erweisen sich eingeschliffene Verfahrensweisen bei mangelnder Unterstützung des Managements als veränderungsresistent.
Die Fachbereiche haben auch Ihre Schwierigkeiten mit eventuell vorhandenen Testtools. Die innewohnenden Rationalisie-

rungsmöglichen bei Regressionstests sind einsichtig und werden auch sofort akzeptiert. Wenn aber im Testalltag die Mühen der toolgerechten Testfallbeschreibung und die darauf zu verwendenden Aufwände das akzeptierte Maß überschreiten, dann schlägt die anfängliche Begeisterung schnell in das Gegenteil um. Auf die Lösungsansätze diverser Anbieter und deren Bewertung wird im Rahmen dieser Abhandlung nicht eingegangen.
Das einfachste Rationalisierungsmittel bei der Abnahme liegt im Übergang von einer pessimistischen zu einer optimistischen Realisierungserwartung. Bei der pessimistischen Realisierungserwartung muss man so viel wie möglich selbst kontrollieren/testen, denn was nicht ausdrücklich (schriftlich) formuliert wurde, wird auch nicht realisiert und auch bei den vereinbarten Leistungen ist man nicht sicher. Bei der optimistischen Realisierungserwartung muss man so viel wie nötig selbst kontrollieren/testen, denn was ausdrücklich (schriftlich) formuliert wurde, wird auch realisiert, bestimmte Grunderwartungen des Hauses (‚state of the art') werden bei der Realisierung garantiert eingehalten, die Wiederverwendung erprobter und getesteter Softwareteile durch die IT sichert die Qualität an den entsprechenden Stellen und die IT testet während der Softwareentwicklung selbst und zwar nach Konzept und gründlich. Zur Rationalisierung gehört dann, mit der IT Testumfänge und Arbeitsteilung zu vereinbaren, der IT die eigenen Testfälle und Testdaten zur Verfügung zu stellen und einen Abgleich mit dem Testprotokoll der IT zur Vermeidung von Doppelarbeit durchzuführen. Die Anwendung der optimistischen Realisierungserwartung erfordert bestimmte Voraussetzungen vor allem aber auch Vertrauen aufgebaut durch positive Erfahrungen.

5. Zieländerungen und Risikobewertung

Wenn die Abnahme zeigt, dass die in den Projektanforderungen definierten Ziele nicht oder nicht vollständig erreicht werden, dann schlägt die Stunde der möglichen Alternativen wie

Teilabnahmen, schrittweise Abnahmen oder eingeschränkte Abnahmen.
Die Idealvorstellung der Requirementanalyse und –definition geht davon aus, dass es möglich ist, am Anfang eines Projekts bei hinreichend sorgfältiger Vorarbeit die Anforderungen so genau zu definieren, dass daraus die Spezifikation und der Test eindeutig ableitbar sind. Dabei wird vergessen, dass Menschen fehlbar sind, dass die an der Anforderungsdefinition mitwirkenden Fachexperten eventuell nicht den Überblick, die Zeit oder die Vollmachten haben, die für diese Idealvorstellung notwendig wären. Desweiteren ist damit das Problem, der während der Entwicklung auftretenden Änderungen in den Prozessen und im sonstigen Umfeld nicht gelöst. Deshalb wird dann mit der Entwicklung begonnen, obwohl man ahnt oder sogar weiß, dass die Anforderungen noch nicht vollständig sind und sich ändern werden. Das Projektmanagement versucht, dieser Situation mit der Anwendung sogenannter leichter oder agiler Vorgehensmodelle und/oder iterativer Entwicklung zu begegnen. Im Rahmen dieses Beitrags interessieren aber nur die Auswirkungen auf die Abnahme.
Während der Entwicklung agiert das Unternehmen ständig in einem sich ändernden Umfeld. Dabei können sich z.B. objektive Änderungen der Projektziele und Prioritäten durch veränderte Marktbedingungen, neue Geschäftsfelder, neue gesetzliche Rahmenbedingungen oder neue Organisationsformen des Unternehmens ergeben. Genauso oft aber teilweise viel weniger greifbar sind subjektive Änderungen durch Veränderung der Machtstrukturen, Wechsel der Verantwortungsträger oder Modifizierung der Unternehmensziele bzw. der Unternehmenspolitik.
All dies führt dazu, dass wie bereits im Punkt 1 beschrieben bei Projekten, die eine Minimalgröße überschreiten und dies ist m.E. nach die Mehrzahl, vor Beginn der Abnahme eine Neudefinition bzw. Aktualisierung der Abnahmestrategie bezüglich Abnahmekriterien und deren Gewichtungen erfolgen muss. Dabei ist es unbedingt notwendig, dass die Aktualisie-

rung durch die fachlichen Leistungsträger der künftigen Nutzer mitgetragen und durch die entsprechenden Verantwortlichen abgesegnet wird. Dies ist nur zu erreichen, indem in den Fachbereichen das Bewusstsein für ansonsten mögliche Konsequenzen beim produktiven Einsatz (Aufzeigen von Negativfällen) geweckt bzw. geschärft wird.
Trotz Prototypen und inkrementeller Entwicklung ist die fachliche Abnahme die Stunde der Wahrheit, da nach erfolgreicher Abnahme der produktive Einsatz erfolgt und damit auch für die Verantwortlichen der Fachbereiche und alle, die bisher dem entwickelten System oder dem Release eines Systems abwartend gegenüberstanden, keine Ausflüchte mehr zählen. Abgenommen ist abgenommen auch wenn in der ersten Zeit des Einsatzes noch genügend Probleme auftreten.
Wenn der Termin fix ist und abgenommen werden muss und die verfügbare Zeit aber nicht mehr zur Beseitigung aller Mängel reicht, dann muss kurzfristig eine Neupriorisierung mit Risikobewertung erfolgen. Es ist auf jeden Fall besser, eine abgestufte Qualität nach Risikoklassen als eine gleichmäßig schlechte Qualität zu haben. Die Zuordnung der Anwendungsteile zu Risikoklassen kann nur nach fachlichen Kriterien erfolgen, wobei die Vertretbarkeit provisorischer Teillösungen entscheidend ist. Dem Management ist eine Information, über die mit den Restmängeln verbundenen Risiken (z.B. fehlerhafte bzw. nicht mögliche Bearbeitung bestimmter Konstellationen), die durch die provisorischen Teillösungen bzw. Restmängel notwendigen Mehraufwände (z.B. manuelle Nachbearbeitungen, Zusatzläufe) bzw. Ergebnisminderungen bewertet und definiert nach Art und Dauer sowie über evtl. notwendige Informationen der Kunden und sonstigen Partner der Fachbereiche bzw. des Unternehmens zu übergeben. In Kenntnis dieser Fakten muss dann letztendlich über den Einsatz entschieden werden.

Traumzeit: Nicht ganz ernst gemeinte Erkenntnisse aus der fachlichen Abnahme des Anwendungssystems XYZ

In der Traumzeit schrieb ich einen Abschlußbericht zu der Abnahme des Projektes XYZ in dem ich sagte, was ich schon immer mal sagen wollte ohne Berücksichtigung der ‚project correctness'.

1. Summary

☺	😐	☹
☒		☒

Lachen oder weinen?

☺	☺	☺
☒	☒	☒

Optimistisch will lachen als eher angebracht erscheinen.

Referenzen	Mr. Murphy
Verteiler	T = Teilnehmer, I= zur Information, S = zur Stellungnahme
	I: an alle Interessierten
Anhang	keiner

<u>Achtung:</u>

- Ähnlichkeiten mit lebenden Personen sind zufällig und nicht beabsichtigt.
- Die Zuordnung von Aussagen zu genannten Personen beruht auf Hörensagen und kann deshalb vor Gericht nicht beeidet werden.

- Die gewerbliche Weiterverwendung und sonstiger Mißbrauch vorliegender Unterlagen ist aus Gründen des Urheberrechtsschutzes untersagt.
- Wer diesen Bericht zu ernst nimmt ist selber schuld.

2. Vorbemerkungen

Erstaunlicherweise gibt es Dinge, die Sex und Qualität gemeinsam haben. Nach Phil Crosby sind dies folgende:

- Alle sind dafür.
- Jeder glaubt, dass er etwas davon versteht.
- Jeder denkt, bei der Umsetzung müsse er nur seinem natürlichen Instinkt folgen.
- Fast alle sind der Ansicht, dass für entstehende Probleme andere verantwortlich sind.

Bei der Entwicklung und dem Test des Projektes XYZ konnten die im folgenden beschriebenen Gesetzmäßigkeiten der Entwicklung von IT-Projekten entdeckt, konkretisiert bzw. verifiziert werden.

Der Verfasser dieses Berichts dankt allen Beteiligten (den genannten und den ungenannten) für ihre kooperative Zusammenarbeit und die Bereitschaft ihre Erfahrungen und Erkenntnisse zur Erleichterung der künftigen Projektentwicklung der Allgemeinheit zur Verfügung zu stellen. Die Ergebnisse sind provisorisch nach Erfahrungsbereichen gegliedert.

3. Erfahrungen der Nutzer von XYZ (mehr oder minder umfassend repräsentiert durch die Mitglieder des fachlichen Abnahmeteams)

- **Erster Grundsatz der Anwendung von Frau B.**
  Wenn nichts mehr funktioniert, so wurde bestimmt eine neue Version eingesetzt.

- **S.‘sches Gesetz von der Logik der Softwareanwendung**

80 Prozent der Anwender setzen nur 20 Prozent der Funktionen ein.

- **Erste logische Ableitung**
  20 Prozent der Anwender benötigen die 80 Prozent der Funktionen, die die Software (noch) nicht besitzt.

- **Zweite logische Ableitung**
  Sie gehören mit Sicherheit zu diesen 20 Prozent.

- **Verdeutlichung**
  Ein Softwareentwickler ist der letzte, der sein Programm auch anwendet, deshalb versteht er auch gar nicht, wenn ein anderer damit Probleme hat.

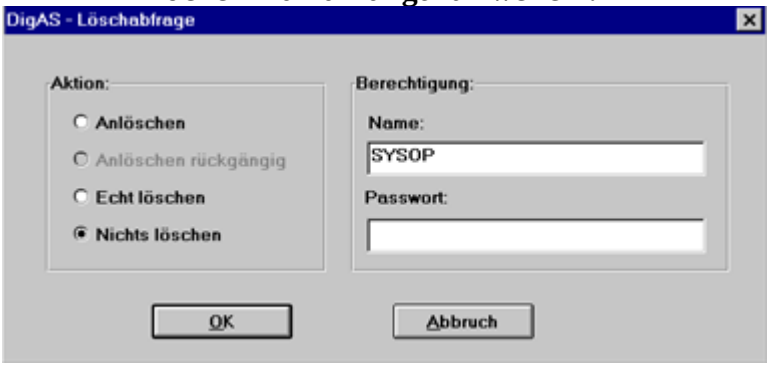
- **Axiom der Fehleroptimierung von Dr. S.**
  Wenn eine Software bei Ihnen fehlerfrei läuft, so machen Sie sich keine Sorgen. Es wird vorbeigehen.

- **Erkenntnis von Hr. A.**
  Wenn Architekten so bauen würde, wie manche Softwareentwickler Ihre Software realisieren, so könnte ein einziger Specht ganze Städte zerstören.

- **Die ‚was ein Entwickler auf kritische Fragen sagt' - Sammlung**
  1. Das funktionierte aber gestern (in der letzten Version, bei der letzten Abrechnung usw.) noch.
  2. Dieses Problem können Sie leicht umgehen, wenn Sie Ihren Arbeitsablauf ein wenig umstrukturieren.
  3. Selbstverständlich ist das erweiterbar. Schreiben Sie einen Tracker und wir werden das prüfen, schätzen und (vielleicht) realisieren.
  4. Unser Verantwortlicher, der dies entscheidet hat leider zur Zeit Urlaub/Kurs/Krankheit.
  5. Diese Änderung ist sowieso schon für die Version x.y.z vorgesehen.

6. Wenn Sie die Software erst mal ein paar Wochen im Einsatz haben, dann erledigen sich Ihre Fragen von selbst.

Es gibt Momente im Leben eines Anwenders, in denen man anfängt zu zweifeln.



Echt löschen?
Oder vielleicht doch nur erst ein bisschen anlöschen?

3. Erfahrungen der Tester von XYZ (Entwicklertest und fachlicher Test)

- **Hr. U.s Grundgesetz**

  Alle großen Softwareentwicklungen wurden auf der Grundlage gravierender Programmfehler verwirklicht.

  - **Erste Folgerung**
    Jedes Programm hat Fehler.
  - **Zweite Folgerung**
    Jedes Programm hat immer einen Fehler mehr.
  - **Dritte Folgerung**
    Die Beseitigung eines Fehlers ruft (fast) immer zwei neue hervor.

- **Hr. G.s Umkehrschluss aus der Existenz von Hilfsprogrammen (SQL-Scripts)**

  Die Tatsache, dass es sehr viele kleine und nützliche Programme gibt, beweist, dass es auch sehr viele große und unnütze Programme gibt.

- **Fr. O.s Postulat der Multifunktionalität**

  Je weniger Funktionen ein Programm hat, umso perfekter wird es sie ausführen.

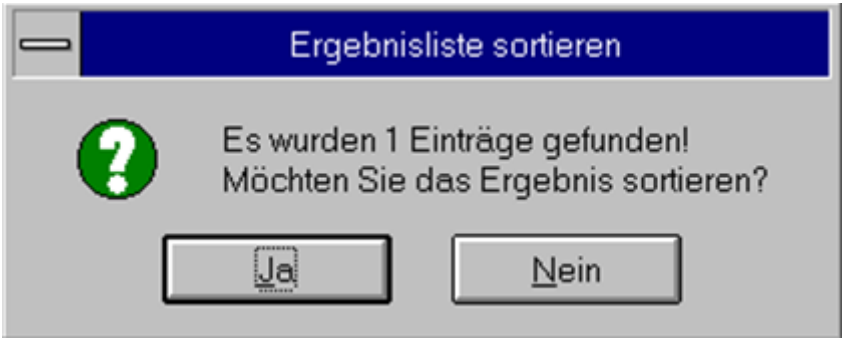
  - **Folgerungen aus dem Postulat der Multifunktionalität**
    1. Je perfekter ein Programm Funktionen ausführt, umso gründlicher wird es eines Tages versagen.
    2. Jedes andere Programm wird genauso versagen.

- **Fr. K.s Behauptung**

  Nur ein Programm, das niemand braucht, läuft fehlerfrei.

- **Erleuchtung von Fr. G.**

  Ein fehlerfreies Programm ist wie die Quadratur des Kreises. Man meint, es könnte gehen aber keiner hat es je gesehen.



Nun... man könnte darüber nachdenken...

4. Erfahrungen der Entwickler und Wartungsverantwortlichen von XYZ

- **Hr. X's Grundlage der Programmierung**

  Es wird nicht auf Anhieb funktionieren.

  - **Erste logische Ableitung**

    Funktioniert es doch, dann hat es jemand anderes geschrieben.

- **Hr. W.'s Doppelregel**

  1. Führt man ein selbst programmiertes Programm vor, dann stößt man beim ersten Mal auf einen offensichtlichen Fehler.
  2. Gravierende Fehler sind von einem selbst nicht reproduzierbar. Sie werden allerdings von jedem anderen bemerkt, der das Programm startet.

- **Hr. R.'s Erkenntnis vom Debugging**

  Nichts verbessert ein Programm so sehr wie das Fehlen von Prüfroutinen.

- **Hr. R.'s erweiterte Erkenntnis**

  Wenn Debugging der Vorgang ist, Fehler aus einem Programm auszubauen, dann ist Programmieren der Vorgang, Fehler einzubauen.

- **R.'sche Folgerung**

  Wenn man nicht weiß, was man tut, so sollte man es wenigstens elegant machen.

- **Erster Grundsatz der Firmen der externen Mitarbeiter**

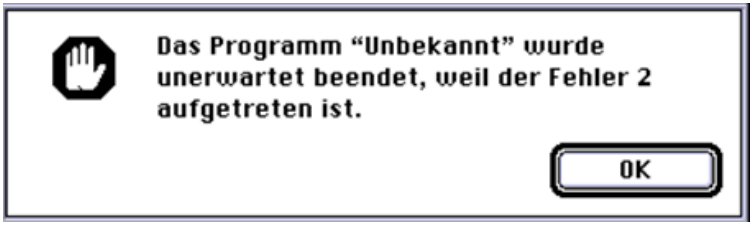
  Jeder Entwickler, der von außerhalb kommt, ist ein Fachmann.

- **Hr. M.'s Gesetze vom Fehlersuchen**

1. In jedem Programm neigen Fehler dazu, am entgegengesetzten Ende der Fehlersuchstrategie aufzutreten.
2. Wenn ein Programm Fehler aufweist, so sieht es fehlerfrei aus.
3. Wenn ein Fehler entdeckt und korrigiert wurde, so stellt sich heraus, dass es schon zu spät ist.
4. War es nicht zu spät, so war die Korrektur falsch und der ursprüngliche Zustand richtig.

- **K.'sche Einschätzung fachlicher Unterlagen zur Migration**

1. Ausnahmen sind grundsätzlich zahlreicher als Regeln.
2. Von allen anerkannten Ausnahmen gibt es Ausnahmen.
3. Wenn man die Ausnahmen endlich im Griff hat, so erinnert sich keiner mehr an die Regeln, für die sie gelten sollen.



Gestalt gewordener Alptraum eines Wartungsspezialisten

5. Erfahrungen des Managements der beteiligten Firmen

- **P.'sches Zeitgesetz**

Wenn ein Software-Fertigstellungstermin genannt wird, so lasse man sich auf jeden Fall die Jahreszahl explizit und vierstellig dazu nennen. Anschließend lasse man von

Frau K. durch Bleigießen den wahren Termin in erster Näherung ermitteln.

- **M.'sche Ableitung**

  Dies gilt mit hoher Wahrscheinlichkeit auch für die Fertigstellung von Testabrechnungen.

- **G.'sche Folgerung**

  Dies gilt auf jeden Fall für die Fertigstellung von System- oder Datenbankumstellungen.

- **D.'sche Grundregel**

  Wenn ein Fehler auftritt, so liegt er mit 90 Prozent Sicherheit außerhalb der Gewährleistung und seine Beseitigung ist kostenpflichtig.

- **R.'sche Kundengesetze**

  1. Kein Kunde weiß, was er eigentlich will.
  2. Jeder Kunde weiß, was er nicht will.
  3. Kein Kunde will, was man bereits fertiggestellt hat.
  4. Der Kunde weiß auch nicht, was er stattdessen möchte.
  5. Der Kunde, der am wenigsten zahlt, meckert am meisten.
  6. Größere Änderungen wird der Kunde immer dann verlangen, wenn die Software eben ausgeliefert wurde.
  7. Wenn man ein Programm erfolgreich erweitert hat, so wird es der Kunde nicht mehr haben wollen.

Meldung zum Abschluss

Kleiner Trost: Wenigstens passiert es nicht einfach irgendwo!

Station 6: In der Schlucht der Risiken

Die Risiken eines unvollständigen Tests bzw. eines Tests überhaupt werden gemeinhin unterschätzt.

Minimalanforderungen an QM ermitteln und einhalten

Im folgenden Beitrag werden die minimalen Voraussetzungen für ein erfolgreiches Qualitätsmanagement (QM) in einer mittelständischen Firma oder in einer von der Größe her vergleichbaren Organisation (im Folgenden Organisation genannt) skizziert. Dabei werden Möglichkeiten des erfolgreichen Umgangs mit den Problemen an ausgewählten wichtigen Elementen des QM (u.a. Festlegung von Qualitätszielen, Gestaltung von Überprüfungsmechanismen und Testorganisation) betrachtet. Die Darlegungen basieren auf Erfahrungen des Autors verbunden mit dem notwendigen theoretischen Background.

1. Konfliktpotential bei der Gestaltung des QM in einer Organisation

Die Gestaltung des Qualitätsmanagements in einer Organisation lässt verschiedene Spielräume je nach Orientierung der Organisation (Kundenorientierung, Kundenkreis, Geschäftsfelder, prozessorientierte Vorgehensweise, inkrementelle Entwicklung, usw.) zu. Damit ist es für die Akzeptanz des QM eminent wichtig, den Beschäftigten zu vermitteln, dass die Gestaltung des QM dem durch die Software zu unterstützenden Geschäft angemessen ist. Prinzipiell muss man sich jenseits aller Vorgehensmodelle, Normen und Reifegrade zwischen folgenden zwei Polen positionieren: Qualität so gut wie nötig - Qualität so gut wie möglich.

2. Verständnis von Qualität und Festlegung von Qualitätszielen

Grundvoraussetzung für die Identifikation der Beschäftigten mit dem QM ist eine für alle verständliche Definition und Interpretation des für die Organisation gültigen Qualitätsbegriffs und der angestrebten Qualitätsziele. Desgleichen muss klar sein, wie die Organisation die Qualitätsziele variiert und welches die nicht unterschreitbaren Minimalforderungen sind. Eine Organisation, die nicht bereit ist, solche (verständlichen, einhaltbaren und kontrollierbaren) Minimalforderungen für sich zu definieren wird auf Dauer mit den Mitarbeitern und auf dem Markt Probleme bekommen.

Andere Branchen tun dies seit langem erfolgreich mit Markenzeichencharakter (z.B. Innungsqualität bei Handwerkern). Bei der Softwarequalität ist dies bisher nicht gelungen und diverse Ansätze (wie Normen und Standards) führen oft zu mehr Papier und Bürokratie aber nicht zu gesicherter Mindestqualität.

Fast alle Organisationen tun sich sehr schwer, solche Minimalanforderungen zu definieren bzw. zu leben.

Im Folgenden soll ein pragmatischer Ansatz dargestellt werden.

3. Vertragsgestaltung für Individualsoftware bzw. Anpassungsleistungen

Üblich sind Formulierungen der Art 'die Leistung ist so zu erbringen, dass sie nicht mit Fehlern behaftet ist, die die Tauglichkeit zum vorgesehenen Gebrauch aufheben oder wesentlich mindern'.

Beide Partner haben Probleme bei der Formulierung von Qualitätszielen und der Bewertung des zu ihrer Erreichung (Auftragnehmer) und Überprüfung/Test (Auftraggeber/Auftragnehmer) notwendigen Aufwands. Diese Probleme

steigen mit der Größe und Komplexität des zu realisierenden Softwarepakets stärker als linear an.

Folgende Formulierungen für Minimalanforderungen haben sich bewährt:

- Variante 1 ('Quick and dirty'): Erwartete Funktionen werden bei korrekten Eingaben auch korrekt ausgeführt; es wird vorausgesetzt, dass nur korrekte Eingaben gemacht werden.
- Variante 2 ('normale' betriebswirtschaftliche Software): Erwartete Funktionen werden bei korrekten Eingaben auch korrekt und in vertretbaren (nicht explizit vereinbarten !) Zeiten ausgeführt; nicht korrekte Eingaben sollen abgewiesen werden und führen (in der Regel) aber nicht zu Abbrüchen.
- Variante 3 (Software mit zusätzlichen Ansprüchen): Mindestqualität siehe Variante 2; zusätzliche Ansprüche/Ziele je nach Einsatzproblematik wie Mindestreaktionszeit für x Nutzer, erhöhte Ausfallsicherheit, bestimmte Art der Benutzerführung

Bei Variante 3 beginnen die eigentlichen Probleme für Auftraggeber und -nehmer.

4. konstruktive Maßnahmen zur Qualitätssicherung

Zur Erzeugung von Qualität wird in der Regel der Einsatz von Vorgehensmodellen, Standards und Tools konzipiert. Hierbei muss die unterschiedliche Interpretation von Verbindlichkeiten und die Stellung zu ausgedehnten Regelwerken in verschiedenen Kundenbeziehungen berücksichtigt werden. Eine einheitliche Firmensicht bzw. eine klar definierte anerkannte persönliche bzw. gruppenbezogene Variationsbreite sind zur Vermeidung von Konflikten dringend notwendig.

Folgende Minimalforderungen werden vorgeschlagen:

- Definition der zugelassenen Tools (Version beachten)

- Definition der Schritte der Entwicklung (evtl. Reihenfolge, Umfang, Muss-/Kannpunkte)
- Definition der Beschreibung/Darstellung bestimmter wichtiger Ergebnisse der Entwicklung (Form, Inhalt, Muss-/Kannpunkte) (z.B. Anforderungen an Fachkonzepte)

5. Test in der Softwareentwicklung

Ausreichender und methodischer Test ist ein Endlosthema bei der Softwareentwicklung (Entwickler versus Nutzer). Dabei können folgende Thesen in dieser oder ähnlicher Form in der jetzigen Zeit als allgemein akzeptiert gelten:

- Man kann ein Programm mit vertretbarem Aufwand nicht so testen, dass seine Fehlerfreiheit garantiert wird.
- Es kommt darauf an, mit einer möglichst kleinen Anzahl von Testfällen ein optimales Testergebnis im Sinne der vorgegebenen Ziele zu erreichen.

Wie kann man dies in der Praxis tun? Mindestens muss man Testendekriterien definieren.

Die im Folgenden aufgeführten Testendekriterien stellen einen Vorschlag dar, der abgewandelt werden kann. Allerdings dürfen die Kriterien nur nach Absprache mit dem Management wesentlich abgeschwächt werden.

Kriterium	**Verbindlichkeit**
für den Test existierte ein Testkonzept	Muss
vorgesehener Testumfang lt. Testkonzept wurde bearbeitet	Muss
keine nichtbehobenen abnahmehinderlichen Fehler mehr vorhanden	Muss
erreichte Testdeckungsgrade	
• Funktionsabdeckung, Richtigkeit, Vollständigkeit	Muss
➢ Funktionsabdeckung der im technischen Ent-	

Kriterium	**Verbindlichkeit**
wurf beschriebenen fachl. Funktionen -> alle Funktionen wurden realisiert	
➢ Richtigkeit -> alle Funktionen wurden fachlich richtig realisiert	
➢ Vollständigkeit -> alle Funktionen wurden	
• Anforderungsüberdeckung wurde geprüft	
➢ alle Anforderungen wurden realisiert	Muss
• Funktionsüberdeckung wurde geprüft	
➢ jede normale Funktion wird mindestens einmal ausgeführt	Muss
➢ jede Ausnahme-Funktion wird mindestens einmal ausgeführt	Muss
➢ jede Ausprägung jeder Funktion wird mind. einmal ausgeführt	Muss
➢ jeder Fehlerfall wird mindestens einmal erprobt	Muss
• Datenüberdeckung wurde geprüft	
➢ jede Datei wird mindestens einmal erzeugt oder benutzt	Muss
➢ jede Satzart, Maske o. Liste wird mind. 1x erzeugt oder benutzt	Muss
➢ jedes Feld wird mind. 1x erzeugt oder benutzt	Muss
➢ die Plausibilisierungen auf jedem Feld wurden geprüft	Muss
➢ wichtige Kombinationen der relevanten Werte in jedem Feld werden mindestens einmal erzeugt oder benutzt	Muss
technische Tests lt. 'Testcheckliste Entwicklertest‘ wurden durchgeführt	Muss
Benutzerfreundlichkeit, Effizienz, Zuverlässigkeit/Robustheit wurde geprüft	Kann
Fehlerkurve beim Test (absteigend und auf niedrigem Niveau stabilisiert)	Kann

6. Anforderungen an Fachkonzepte

Die Gliederung ist rein inhaltlich und impliziert keine verbindliche Reihenfolge bzw. stützt sich auf keinen Standard.

Nr.	**Gliederungspunkt**	**Verbindlichkeit**
1.	Einleitung	Kann
2.	fachliche Funktionalität	Muss
2.1	Kurzüberblick	Muss
2.2	Funktionsstruktur (baumartige Darstellung der Funktionen und ihrer Hierarchie bzw. sonstige grafische Darstellung der Funktionalität)	Kann
2.3	Grundannahmen und Rahmenbedingungen (für die Funktionalität des neuen/geänderten Anwendungssystems)	Kann
2.4	Abgrenzungen (explizite Benennung nicht realisierter Funktionalität bzw. sonstiger Leistungen)	Kann
2.5	Funktionen (detaillierte Beschreibung der Funktionalität)	Muss
2.6	Benutzerschnittstelle (Dialog-und Listengestaltung)	Kann
2.7	Datenmodell	Kann
2.7.1	fachliches Datenmodell (Beschreibung aus fachlicher Sicht, z.B. Tabellen, Datenelemente)	Kann
2.7.2	Zugriffsmöglichkeiten (notwendige Zugriffsmöglichkeiten aus fachlicher Sicht)	Kann
2.7.3	Mengengerüste (voraussichtliche Verarbeitungsumfänge, z.B. Anzahl Eingaben, zu verarbeitende Buchungen, Dynamik der Entwicklung der	Kann

Nr.	Gliederungspunkt	Verbind-lichkeit
	Mengen)	
2.8	Systemschnittstellen (Beschreibung der Schnittstellen zu anderen Systemen im Hause oder zu externen Systemen aus fachlicher Sicht)	Kann
2.9	Beachtung gesetzlicher Anforderungen (Revision, IT-Sicherheit)	Kann
2.10	systemweite Anforderungen (z.B. an Datierung, Historisierung)	Kann
2.11	Systemumstellung/Migration (Anforderungen an und Strategie der evtl. Migration)	Kann
3.	Qualitätssicherung	Muss
3.1	Teststrategie	Muss

- Zu jedem Punkt muß eine Aussage gemacht werden. Bei den Kann-Punkten sind Aussagen der Form ‚Wird gemäß Absprache darauf verzichtet' bzw. ‚Wurde geprüft und entfällt' zulässig.
- Bei einem Wartungsprojekt enthalten die Punkte des Fachkonzepts in der Regel nur die Änderungen zum bisherigen Zustand.

Paul, hat Deine Firma schon einen QM-Check gemacht?
Mein Chef macht erst mal den Basel-II-Check für unsere künftige Kreditwürdigkeit.

QM-Check einer Firma bzw. einer Struktureinheit (Inhalte, Umfang, mögliche Ergebnisse, Probleme)

Warum und wann sollte ein QM-Check durchgeführt werden?

- Die Qualität Ihrer Software wird von Kunden (intern oder extern) bemängelt.
- Ihre Gewährleistungsleistungen übersteigen regelmäßig das kalkulierte Maß.
- Sie sind mit der Effizienz Ihrer Softwareentwicklung nicht zufrieden.
- Sie wissen nicht, wie gut oder schlecht Ihre Software nach der Abnahme ist.
- Sie haben Vorgehensmodelle im Einsatz und haben dabei aber kein gutes Gefühl.
- Ihre Firma möchte oder soll sich nach xyz zertifizieren lassen und Sie brauchen eine Schätzung der notwendigen Aufwände zur Erreichung dieses Ziels.

etc. pp. ...

mögliche Inhalte und Ziele von QM-Checks

- Untersuchung und Bewertung u.a.
    - des Software-Entwicklungsprozesses,
    - der Wirksamkeit eingesetzter Vorgehensmodelle,
    - des Know-how Ihrer Mitarbeiter bezüglich der zu lösenden Aufgaben,

    - der eingesetzten Tools und ihrer Benutzung,
    - der Qualitäts-Motivation Ihrer Mitarbeiter
- Identifizierung von qualitätsbeinflussenden Schwachstellen in ...
- Erarbeitung von Verbesserungsmaßnahmen für ...
- Bestätigung eines geforderten Niveaus (gemäß CMM etc.) von ...

Vorgehensweisen bei QM-Checks

- Untersuchung der Prozesse u.a. mittels
    - Assessments der Mitarbeiter
    - Self-Assessments der Mitarbeiter
    - Analyse der implementierten Vorgehensmodelle, Verfahren und Hilfsmittel (Tools) auf
        - Angemessenheit und Korrektheit
        - Verfügbarkeit
        - Wirksamkeit (‚gelebt werden‘)
        - Audits zu Prozessen, Projekten etc.
- Befragungen (durch Fragebögen oder Einzelbefragung) u.a.
    - der Mitarbeiter zu allen Aspekten (Prozesse, Management, Organisation, Motivation etc.)
    - der Kunden
    - des Managements
- Black-box-Untersuchung der Qualität der Ergebnisse (der Anwendungssysteme) u.a. auf
    - Fehlerquoten und Fehlerarten

    - Handhabbarkeit
    - Kundenzufriedenheit
- White-box-Untersuchung der internen Qualität der Anwendungssysteme u.a. auf
    - Codequalität
    - Dokumentation

Ansprüche an die QM-Checks

- weitgehende Objektivität der Untersuchungen
- Sicherung der Anonymität der Mitarbeiterbefragungen
- statistisch abgesicherte Ergebnisse
- Trennung von Wesentlichem und Unwesentlichem
- umsetzbare und angemessene Vorschläge (notwendiger Detaillierungsgrad)
- bewertete Ergebnisse, z.B.
    - Priorisierung der gefundenen Schwachstellen nach abgestimmten Kriterien
    - Schätzung der Aufwände zur Behebung von Schwachstellen
    - Aufwände und Effekte für vorgeschlagene Verbesserungsmaßnahmen
    - Messverfahren für den Effekt vorgeschlagener Verbesserungsmaßnahmen

mögliche Probleme bei QM-Checks

- Ziele des Checks sind nicht klar formuliert oder kommuniziert (Kunde hat sich etwas anderes darunter vorgestellt)
- Verfahrensweisen sind nicht (eindeutig) abgestimmt

- Unterlagen sind unvollständig oder werden nicht offengelegt
- Management blockt Untersuchungen ab
- Management baut ‚potemkinsche Dörfer' auf (Scheindokumentationen, Schrankware)
- Mitarbeiter stehen nicht zur Verfügung
- Mitarbeiter boykottieren den Check (ablehnende Haltung, falsche Aussagen)
- Rahmenbedingungen sind unzureichend (Arbeitsplätze, technische Hilfsmittel)
- Zeit ist zu knapp bemessen

erfolgreiche QM-Checks

- hatten klare Zielstellungen
- wurden unter Einbeziehung des Managements und der Mitarbeiter durchgeführt
- waren angemessen und blieben im kalkulierten Aufwandslimit
- werden schon für eine Wiederholung geplant
- bringen klare, nachvollziehbare und verwertbare Ergebnisse,
    - Mindestumfang von Maßnahmen
    - Effekte und Aufwände von Maßnahmen
- wecken keine Illusionen (Nullfehlerrate o.ä.) sondern geben zu, dass nur die Wahrscheinlichkeit eines besseren Arbeitens und höherer Qualität erhöht werden kann

Manchem würde schon ein Licht aufgehen, wenn er nicht beim kleinsten Schein die Schutzbrille aufsetzen würde.
Paul, wie denkt Dein Chef über Risiken beim Test?
Mein Chef handelt nach der Devise ‚no risk – no fun'

Softwarerisiken = Risiken beim Test ?

Die im Jahre 2004 ermittelten Sachverhalte sind leider auch im Jahr 2013 noch weitgehend gültig.

Laut einer weltweiten Anwenderbefragung der Meta Group weist neu eingeführte Software zusehends Mängel auf (Worldwide IT Benchmark Report 2004 - Volume 4 - Part 9 - Defect Rates ... siehe Computerwoche 39/2004)

- So stieg in den letzten drei Jahren der „Post Implementation Defect", der die Fehlerrate der Software pro 1000 Codezeilen beschreibt, von 0,36 auf 0,68.
- Ursache sind verkleinerte Entwicklungsteams, die unter hohem Zeitdruck immer komplexere und sich schnell ändernde Anforderungen umsetzen müssen.
- Die Analysten raten, mehr in
  - Testwerkzeuge
  - Prozesse
  - Weiterbildung
  - Zertifizierung

  zu investieren. (Anmerkung des Autors: **alles alte Rezepte**)

Wege zur Qualitätsverbesserung

- Verbesserung der Bedingungen, unter denen Software produziert wird (= konstruktives QM)
  - Menschen (Entwickler, Management, ...)

    - Prozesse, Verfahren
    - Werkzeuge, Tools, Technik
- Verbesserung der produktionsbegleitenden Qualitätssicherung (= analytisches QM)
    - Menschen (Akzeptanz, Überzeugung, ...)
    - Testverfahren, Methoden, ...
    - Werkzeuge, Tools
- Verbesserung der Abnahme der Software, d.h. u.a.
    - ‚Schrott' wird als solcher erkannt und wird so nicht ausgeliefert
    - (Mindest-)Qualität geht vor Termin
    - Restrisiken sind bekannt

Philosophien der Qualitätsverbesserung (optimistisch, euphorisch, blauäugig, ...)

- Wenn erst xyz (das neue Tool bzw. die neue Technik, der neue Prozess) zum Einsatz kommt, dann wird alles besser ...

  -> mehr als 30 Jahre ohne wirklichen Erfolg sollten die Gläubigen eigentlich kuriert haben
- Wenn während der Entwicklung alles kontrolliert und getestet wird, dann kann man alle Fehler beheben ...

  -> scheitert an objektiven Umständen (Menge, Zeit, Budget, prinzipielle Unmöglichkeit bei entsprechender Komplexität ...)
- Wenn die Beteiligten nur alle guten Willens sind und sich entsprechend engagieren, dann ...

  -> funktioniert allenfalls im Märchen und bei Harry Potter

Philosophien der Qualitätsverbesserung (pessimistisch, realistisch, pragmatisch, ...)

- Wenn schon nicht eine hohe Qualität erreichbar ist, dann wenigstens eine minimale ...

  -> Was ist minimal?

- Wenn schon während der Entwicklung nicht alles kontrolliert und getestet werden kann, dann wenigstens das Wichtigste ...

  -> Was ist wichtig?

- Wenn schon mit Werkzeugen und Tools nur punktuelle Verbesserungen erreicht werden können, dann wenigstens das Notwendige und Nützliche einsetzen ...

  -> Was ist notwendig und nützlich?

Philosophien der Testtoolhersteller und QM-Dienstleister

- Es ist eine Entwicklung feststellbar ...
  - Versprechung von Wundern durch Toolbenutzung (möglichst auf Knopfdruck)
  - Versprechung eines Fast-Wunders bei methodischem Einsatz der Tools
  - Wunder dauern doch etwas länger und erfordern zuerst einiges an Investitionen (Geld, Manpower, Geduld, ...). Für das erste damit durchgeführte Projekt gibt es kein (positives) Wunder.
  - Es gibt vielleicht doch keine Wunder aber was gibt es dann (evtl. Ratlosigkeit)?
  - Wenn es schon keine Wunder gibt, so will man den Zustand möglichst genau beschreiben, grafisch darstellen und ... vielleicht auch etwas zu einem vom Nutzer zu definierenden Projekterfolg beitragen.

Es handelt sich hierbei um die rein subjektive Sicht des Autors!

einige notwendige Hinweise des Autors

- Die im Folgenden zitierten Auszüge aus einem Leitfaden der Fa. Compuware sollen aus der subjektiven Sicht des Autors den Sinneswandel der Toolhersteller verdeutlichen.
- Die Firma Compuware wurde dabei mehr oder minder zufällig aus der Reihe der Testtoolhersteller ausgewählt.
- Der Autor legt darauf Wert zu betonen, dass aus seiner Sicht
  - der Einsatz von Testtools nützlich sein kann,
  - bei bestimmten Problemen (z.B. Performancetest) unverzichtbar ist,
  - die Firma Compuware eine seriöse Firma mit einem breiten und bewährten Tool- und Dienstleistungsspektrum ist (allerdings eben auch keine Wunder vollbringen kann).

Erfolg haben in einer unvollkommenen Welt ...(Zitate aus einem Leitfaden der Fa. Compuware)

- Es wäre unaufrichtig (und nicht plausibel), den Eindruck zu erwecken, dass risikobasiertes Testen bei jedem Projekt zu perfekten Anwendungen führt.
- Bei begrenzten Ressourcen und sich ändernden Anforderungen gibt es keine perfekte Anwendung.
- Risikobasiertes Testen garantiert bessere Anwendungen, weil es sicherstellt, dass knappe Testressourcen sich auf die Punkte konzentrieren, wo sie am effizientesten sind.

- Wenn das unvermeidliche geschieht und das Team keine Zeit mehr für das Testen hat, so sind nur die Teile des Systems übrig, die weniger wichtig sind.
- Das Management hat vollständige Kenntnis der Auswirkungen ...

  Wäre das nicht schon wieder ein Wunder?

s. auch Restrisikoabschätzungen beim Test

- Abschätzung der Fehleranzahl in einer Software

  (Softwarerisiko)
- Abschätzung der Fehleraufdeckung durch Tests

  (Testrisiko)
- Abschätzung der Restfehleranzahl

  (verbliebenes Gesamtrisiko)
- Was kann man pragmatisch tun?
  - Risikoklassen für Testobjekte definieren
  - Fehlschlagwahrscheinlichkeit beurteilen
  - Testendekriterien festlegen

Risikofaktoren beim Test

- Die Testbasis (Geschäftsablaufverfahren, Benutzungsanleitung und Entwurfsspezifikationen) ist nicht rechtzeitig verfügbar bzw. von unzureichender Qualität.
- Die Verfügbarkeit von Testpersonal (quantitativ und qualitativ) ist nicht ausreichend.
- Das vorher festgelegte Datum für die Produktivsetzung des Systems verhindert die vollständige Durchführung der Testaktivitäten.

- Die Bereitstellung und das Funktionieren der gewünschten Testumgebung verzögert sich.
- Das Testteam hat nicht die vollständige Kontrolle der Testumgebung und aller dazugehörigen Elemente (Software und Daten).
- Produktivitätsverluste infolge mangelnder Kenntnisse einer Testmethodik bzw. Einführung einer neuen Testmethodik, mit der noch keine Erfahrungen vorliegen
- Effizienzverluste infolge nicht vorhandener bzw. nicht eingespielter Testmanagementprozesse,
- Aufwands- und Terminüberschreitung für den Test bei alleiniger Schätzung und Planung auf der Basis von Erfahrungswerten,
- nicht abdeckende Tests bzw. Unkenntnis über in der Software noch enthaltene Fehler und Risiken bei Nichtnutzung von dem Testobjekt angemessenen Testspezifikationstechniken,
- Systemausfall bzw. unvertretbare Performance in einzelnen Funktionen für die Nutzer bestimmter Produkte infolge fehlender toolgestützter Lasttestmöglichkeiten,

Vielleicht steht es in den Sternen...

Risikooptimierte Qualitätssicherungstrategien bei Migrationsprojekten

1. Migrationsprojekte und ihre Besonderheiten

Die Landschaft der IT-Lösungen in den Unternehmen unterliegt einem ständigen Veränderungsdruck. Dieser Veränderungsdruck resultiert u.a. aus

- neuen bzw. geänderten fachlichen Anforderungen (z.B. neue Gesetzlichkeiten, organisatorische Änderungen in den Fachbereichen)
- veränderten Marketingstrategien
- Nutzung neuer technischer Möglichkeiten (E- und M-Commerce)
- Weiterentwicklung gekaufter Softwaresysteme (neue Release, Auslauf der Wartung durch den Hersteller)
- Änderungen der Firmenstruktur z.B. durch Fusionen bzw. Firmenübernahmen

In bestimmten Fällen wird der Änderungsdruck so groß, dass die notwendigen Änderungen und Erweiterungen innerhalb eines vorhandenen Anwendungssystems und/oder der basierenden Datenbank nicht mehr durchführbar sind. Die Nichtdurchführbarkeit kann sich hierbei z.B. aus softwaretechnischen Gründen oder aus Kostengründen ergeben.

In solchen Fällen wird dann meistens ein Projekt zum Ersatz oder umfassenden Redesign des bisherigen Anwendungssystems durchgeführt. Die Daten des bisherigen Anwendungssystems stellen für das Unternehmen eine unverzichtbare Ressource dar, die auf jeden Fall zu retten ist. Damit ist in der Regel mindestens auch ein Migrationsprojekt durchzuführen.

Folgende Faktoren führen dazu, dass Migrationsprojekte in der Regel als besonders kritisch eingestuft werden:

- die Projektgröße
- die strategische Bedeutung des Migrationsprojektes
- die Auswirkungen bei einem Scheitern eines Migrationsprojektes
- die Anforderungen an die zu erreichende Qualität
- die erforderliche Sicherheit und Planbarkeit für den Umstellungsprozess

Migrationsprojekte haben aber ihre Besonderheiten und Tücken. Unkenntnis und Nichtbeachtung dieser Besonderheiten führt oft zu gravierenden Problemen, die z.B. Fusionstermine in Frage stellen können. Über diese Dinge wird nicht gern gesprochen aber in der Branche sind die Misserfolge sehr wohl inoffiziell bekannt.

Der folgende Beitrag befasst sich speziell mit den Qualitätssicherungstrategien bei Migrationsprojekten. Dabei werden viele Probleme nur gestreift. Der Autor hat aber in erfolgreich durchgeführten Migrationsprojekten die Anwendbarkeit und Erfolgsrelevanz der diskutierten Vorgehensweise praktisch erprobt und angewandt.

2. Arten von Migrationen

Im Beitrag werden nur Migrationen betrachtet, bei denen auch die Daten migriert werden.
Beispiele für Datenmigrationen aus fachlichem bzw. organisatorischem Hintergrund sind:

- Ablösung bzw. Umstellung vorhandener Anwendungssysteme
- Fusionen bzw. Firmenübernahmen

Beispiele für Datenmigrationen aus technischem Hintergrund sind:

- Wechsel des Datenbankmanagementsystems
- Wechsel des Betriebssystems
- Wechsel von Betriebssystem und Rechnerplattform
- Umstellung bestehender Anwendungen ohne fachliche Aspekte wie z.B. Euro-Umstellung

Jeder dieser Fälle hat Besonderheiten und Probleme, die sich in konkreten Projektsituationen zeigen. Trotzdem bestehen hinreichend viele Gemeinsamkeiten, die eine zusammenfassende Betrachtung rechtfertigen.

3. Qualitätsziele von Migrationsprojekten

Unverständlicherweise werden immer noch in vielen Projektverträgen weder Qualitätsziele noch deren definierte Erreichung festgeschrieben. Die Konsequenzen sind regelmäßig in größeren öffentlich bekannten Projekten zu besichtigen.

In der Praxis haben sich folgende Qualitätsziele für Migrationsprojekte als besonders relevant erwiesen. Wichtig ist in diesem Zusammenhang auch, dass die Erreichung dieser Ziele messbar ist und zwar mit in der Regel vertretbarem Aufwand.

- Vollständigkeit der Migration
- Korrektheit der migrierten Daten und der (Ziel-) Anwendungen
- Lauffähigkeit der (Ziel-) Anwendungen auf Basis der migrierten Daten
- Performance und Antwortzeitverhalten der (Ziel-) Anwendungen
- Stabilität und Verfügbarkeit der (Ziel-) Anwendungen

- Durchführung der Migration im vorgesehenen Zeitfenster

4. Probleme bei der Qualitätssicherung von Migrationsprojekten

Auch bei einer Orientierung an den im vorigen Punkt genannten relevanten Qualitätszielen gibt es häufig auftretende Probleme und Stolpersteine. Diese sind u.a.

- Der Test der (Ziel-)Anwendungen, Test der Migrationsprogramme und Test der migrierten Daten müssen unter einen Hut gebracht werden (abgestimmt sein).
- Der Test muss in Teilen auch mit vollständigem Datenvolumen erfolgen (Zeitfenster, Vollständigkeit, Performance)
- Evtl. notwendige Nachbearbeitungen der Daten nach der maschinellen Migration müssen getestet werden.
- (Ziel-)Anwendungen müssen rechtzeitig fertig sein um den Test der (Ziel-)Anwendungen auch mit migrierten Daten durchführen zu können.

Probleme rechtzeitig erkennen und gezielt reagieren ist auch hier die Devise. Was kann man tun?

- Das Bewusstsein für ansonsten mögliche Konsequenzen beim produktiven Einsatz (Aufzeigen von Negativfällen) muss geweckt bzw. geschärft werden.
- Wenn die verfügbare Zeit nicht mehr zur Beseitigung aller Mängel reicht, dann muss kurzfristig eine Neupriorisierung mit Risikobewertung erfolgen.
- Es ist auf jeden Fall besser, eine abgestufte Qualität nach Risikoklassen als eine gleichmäßig bzw. undefiniert schlechte Qualität zu haben. Die Zuordnung der Anwendungsteile und Migrationsprobleme (z.B. unvollständige oder in Teilen inkorrekte Migration) zu Risikoklassen kann nur nach fachlichen Kriterien erfolgen, wobei die

Vertretbarkeit provisorischer Teillösungen entscheidend ist.

- Dem Management ist eine Information, über die mit den Restmängeln verbundenen Risiken bewertet und definiert nach Art und Dauer sowie über evtl. notwendige Informationen der Kunden und sonstigen Partner der Fachbereiche bzw. des Unternehmens zu übergeben. Dies können z.B.
  - fehlerhafte bzw. nicht mögliche Bearbeitung bestimmter Konstellationen,
  - die durch die provisorischen Teillösungen bzw. Restmängel notwendigen Mehraufwände (z.B. manuelle Nachbearbeitungen, Zusatzläufe) bzw. Ergebnisminderungen

  sein.

5. Risikooptimierte Qualitätssicherungsstrategien

Erfolgreiche Vorgehensweisen für Migrationsprojekte müssen damit die folgenden Projektteile abdecken bzw. unterstützen:

- Anpassung der Anwendungen
- Erstellung der erforderlichen Migrationsprogramme
- projektspezifisches Konfigurationsmanagement
- projektspezifisches Versionsmanagement
- den eigentlichen Umstellungsprozess
- Projektsteuerung und Kontrolle

Die im Folgenden gezeigte Strategie zur erfolgreichen und risikooptimierten Qualitätssicherung von Migrationen sollte maschinell unterstützt werden. Trotzdem kann der Grad der maschinellen Unterstützung je nach Projekt differieren. Wichtig ist, dass bestimmte Prinzipien eingehalten werden. Hierfür stehen Checklisten und Konzepte zur Verfügung.

Die Strategie basiert auf dem Aufbau einer Metadatenbank mit allen migrationsrelevanten Informationen

- für die Realisierung der Migrationsprogramme
- für die Umstellung der Anwendungsprogramme (evtl.)
- für die Projektplanung und Projektsteuerung
- für die Durchführung der Migration

Basierend auf der Metadatenbank erfolgt die Sicherstellung von

- Übereinstimmung der Metadaten mit den aktuellen bzw. geplanten Datenstrukturen
- Übereinstimmung der generierten Objekte mit den Metadaten
- Statusverfolgung
- Ausführungs- und Ablaufkontrolle

Die Planung und Durchführung der Tests der Migration beruht u.a. auf

- maschinellem Abgleich und Prüfung der Migrationsergebnisse
- toolgestütztem Lasttest der (Ziel-)Anwendungen auf Basis der migrierten Daten
- maschinellem Abgleich der Ergebnisse der Altanwendungen und der (Ziel-)Anwendungen
- funktionalem Test der (Ziel-)Anwendungen mit migrierten und selbst erfassten Daten
- Testläufen zur Durchführung der Migration im vorgesehenen Zeitfenster

Für die Konstruktion der Testszenarien und Testfälle sollten migrationsspezifische Checklisten verwendet werden. Die

maschinelle Unterstützung der funktionalen Tests der (Ziel-)Anwendungen durch geeignete Testtools hängt von den in der migrierenden Firma vorhandenen Tools ab. Für das Migrationsprojekt ist diese Unterstützung zwar wünschenswert aber das Migrationsprojekt allein rechtfertigt nicht, die in der Regel doch kostenmäßig aufwändige Einführung eines Testtools.

Zur Erhöhung der Qualität sollte im konkreten Projekt die Generierung bestimmter Migrationskomponenten wie Migrationsprogramme, Prüfprogramme, JCL geprüft werden.

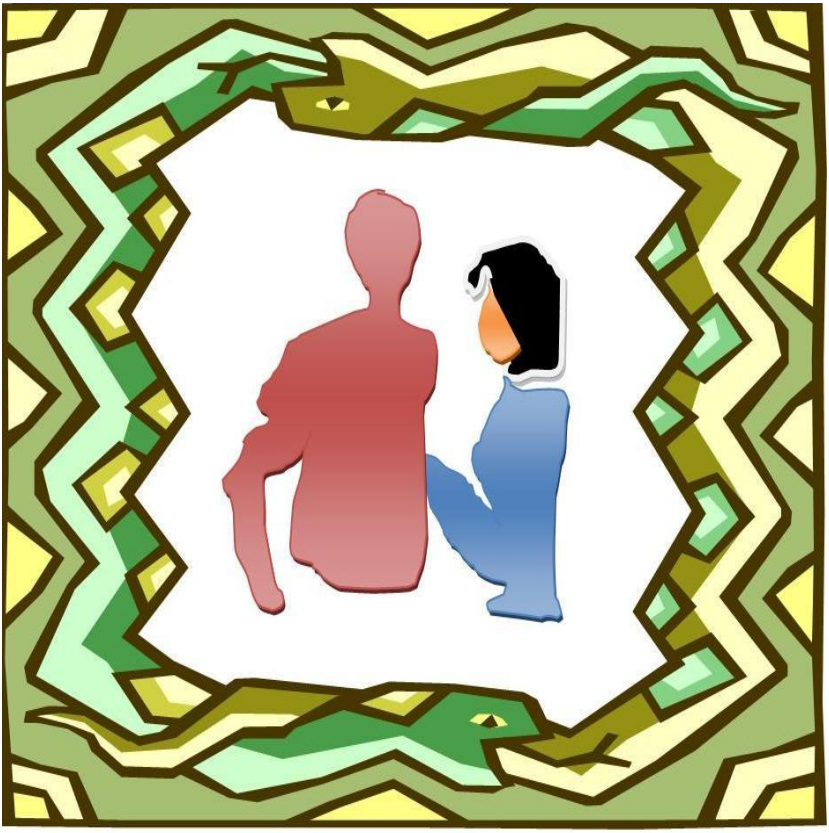
6. eine abschließende Bemerkung

Im vorliegenden Beitrag wurden einige „techniklastige“ Komponenten der erfolgreichen Durchführung von Migrationsprojekten betrachtet. Natürlich sind damit die sonstigen Probleme des Projektmanagements, die u.a. auf der Teamebene, der Projektpositionierung in der Firma, der Unterstützung durch die nutzenden Bereiche, liegen nicht vom Tisch.

Es kann aber nach Ansicht des Autors mit Fug und Recht behauptet werden, dass mit der vorgestellten Vorgehensweise eine gute und verlässliche Basis für den Erfolg kritischer Migrationsprojekte gelegt wird. Voraussetzung ist, dass die Vorgehensweise konsequent verfolgt und möglichst maschinell unterstützt durchgeführt wird.

Traumzeit: Als Qualitätsmanager (QM) und Softwaretester auf der Couch bei ELIZA

In der Traumzeit zogen nochmals meine Erlebnisse bei der Beratung von Softwareentwicklern an mir vorbei
Die Erfolge bei der Beratung von Entwicklern beim Test waren für mich nicht recht befriedigend, da auch die heutigen Entwickler im Prinzip die gleichen Fehler machen wie vor 40 Jahren.
Ich entschloss mich im Traum deshalb, einen Psychotherapeuten zu konsultieren, um mit ihm über die bei meiner Tätigkeit aufgetretenen Probleme zu diskutieren.
Dabei stieß ich bei einer Recherche auf das Programm ELIZA von Joseph Weizenbaum (entwickelt am MIT) welches einen klientenzentrierten Psychotherapeuten simuliert.
Teile meiner Unterhaltung mit ELIZA finden sich auf den nächsten Seiten. QM ist hier die Abkürzung für Qualitätsmanager.
ELIZA können Sie selbst konsultieren u.a. auf www.med-ai.com.



QM: Warum macht ein junger Softwareentwickler heute ähnliche Fehler wie die Entwickler vor 40 Jahren?

ELIZA: Was denken Sie, warum?

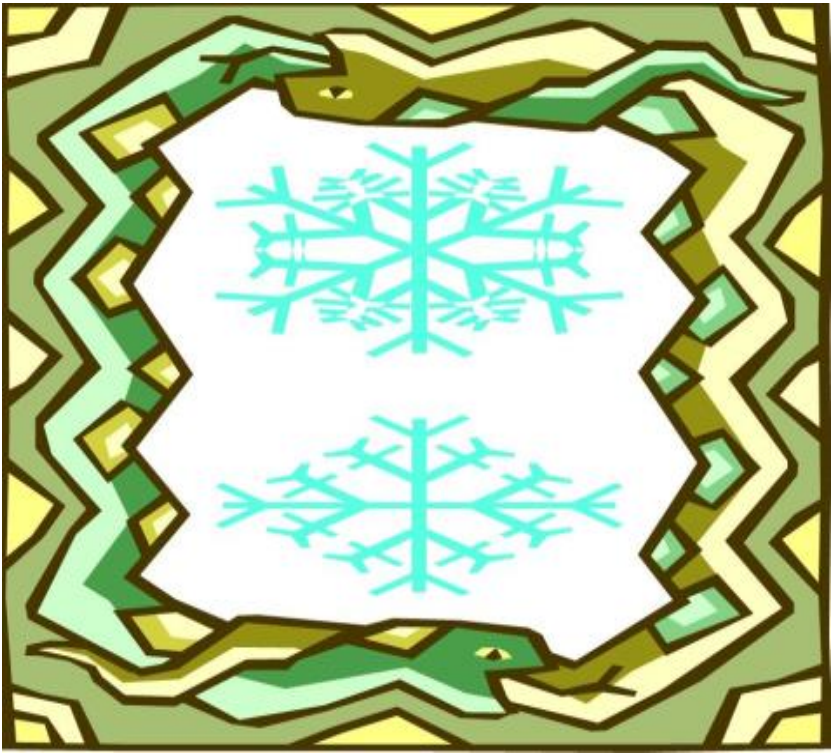
QM: Warum macht ein Softwareentwickler in der Regel keinen sauberen und umfangreichen Entwicklertest?

Ist er nicht imstande seinen Code kritisch zu sehen?

ELIZA: Was glauben Sie denn?

QM: Der Austausch und die Abstimmung von Testfällen zwischen abnehmender Fachabteilung, IT-Testteam und Softwareentwicklung erfolgt nicht bzw. nicht systematisch.
Mögen die sich vielleicht nicht?

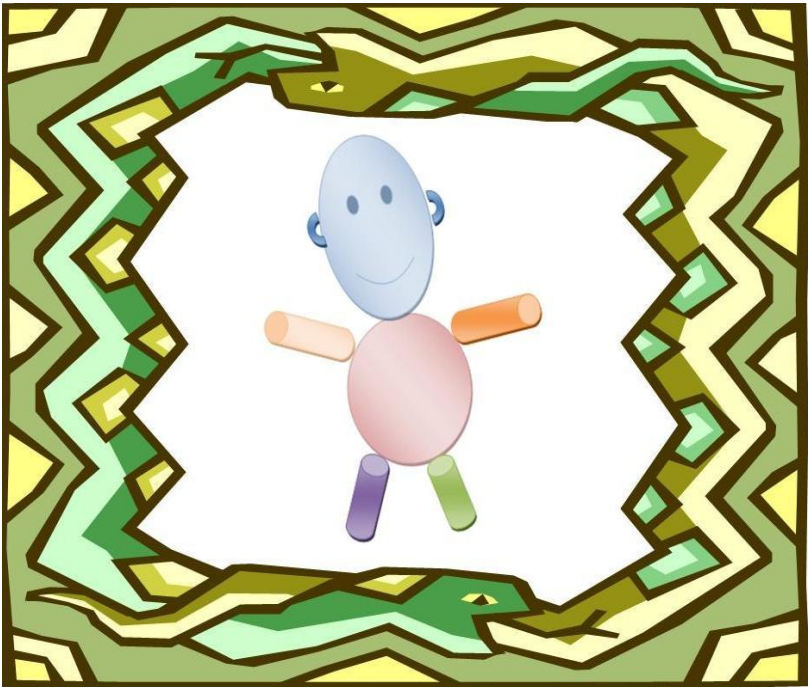
ELIZA: Was wollen Sie eigentlich wirklich wissen?



QM: Aufwandsschätzungen und Testplanungen beruhen auf Erfahrungswerten anstatt auf Charakteristiken des Testobjekts wie Funktionsumfang und Komplexität.

Ist Einfachheit schöner als Komplexität?

ELIZA: Was würden Sie mir antworten, wenn ich Ihnen eine solche Frage stellte?



QM: Testspezifikation und Testfallermittlung werden nicht immer unter Nutzung methodischer Verfahren vorgenommen. Aussagen über Test-abdeckungen, Abnahmequalitäten und Rest-fehlerprognosen sind deshalb nicht oder nur sehr begrenzt möglich.

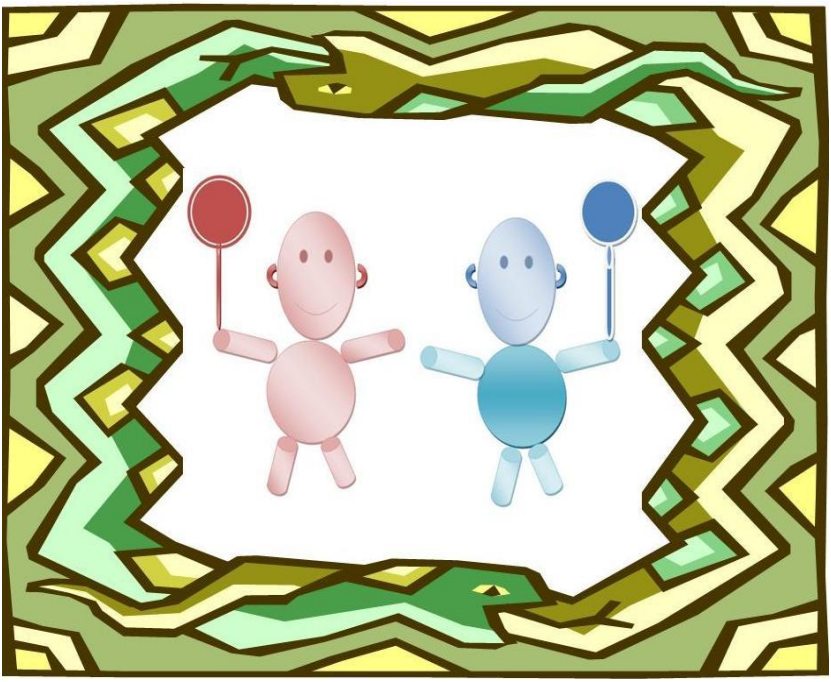
Interessiert das überhaupt jemanden?

ELIZA: Tauchen solche Fragen oft in Ihrem Kopf auf?

QM: Beim Test wird selten eine Risikoanalyse vorgenommen, auf deren Basis entschieden werden kann, was in welcher Intensität zu testen ist. Wenn Risikoüberlegungen vorgenommen werden, so erfolgen diese weitgehend intuitiv und werden auch nicht dokumentiert.

Intuitiv ist offenbar kreativ und spannend?

ELIZA: Wie sicher sind Sie sich?



QM: Warum werden externen Partnern in der Regel keine verbindlichen (vertraglich fixierten) und konkreten Qualitätsvorgaben gemacht?

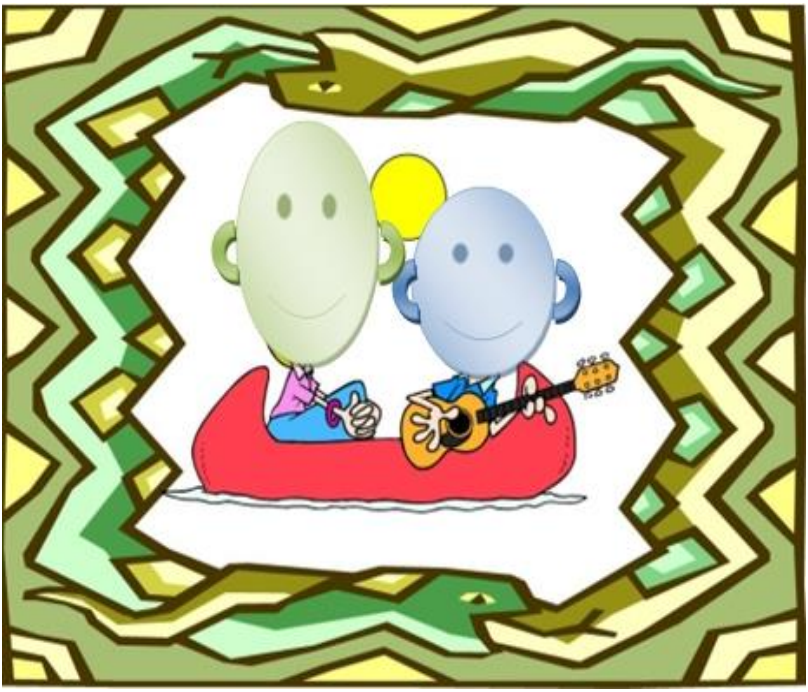
Ist das unangenehm oder kann man die Vorgaben nicht formulieren?

ELIZA: Weder noch... aber…

QM: Für die Dokumente zum Test (Teststrategie, Testkonzept, Testplan, Testfallbeschreibung etc.) existieren zwar diverse Beispiele in den einzelnen Projekten, die sich vom Grundsatz ähneln aber es fehlen meist einheitliche Regelungen, die für alle Partner des Testprozesses gelten.

Ist Einheitlichkeit etwas Schlechtes?

ELIZA: Beschreiben Sie das Problem etwas näher



QM: Es existiert oft kein Testglossar, dass die für den Test verwendeten Begriffe firmenweit definiert. Dies ist vor allem ein Problem bei der Zusammenarbeit mit den externen Partnern.

Wie soll man sich verstehen wenn man nicht die gleiche Sprache spricht?

ELIZA: Denken Sie, es ist sehr wahrscheinlich sich zu verstehen?

QM: Es gibt meistens keine effektive Durchführung von Regressionstests.

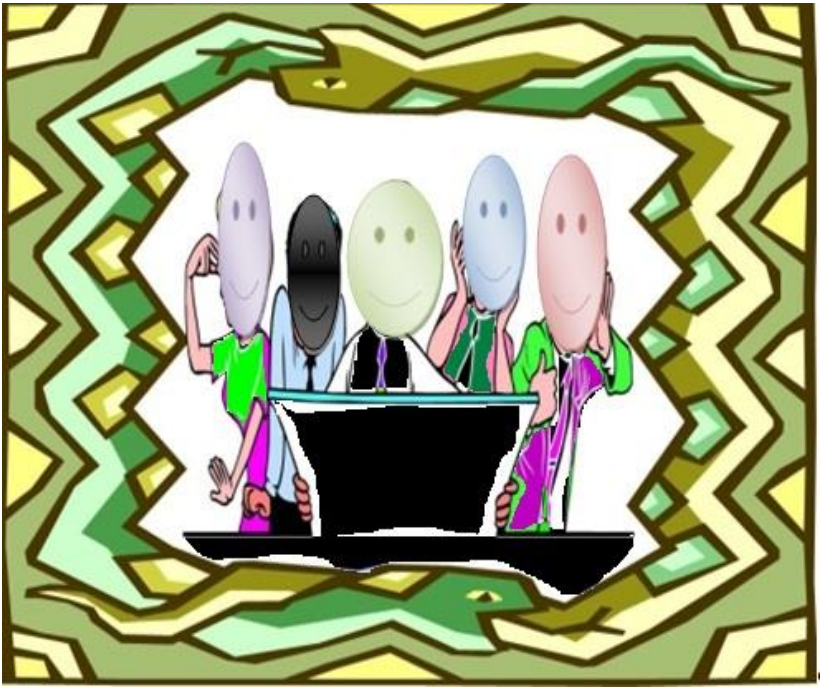
Wie kann man die Regressionstests den Beteiligten schmackhaft machen?

ELIZA: Beschäftigt Sie diese Frage sehr?

QM: Die Verfügbarkeit von Testpersonal (Kapazität, Eignung, Fachkenntnisse) ist wegen der Belastung des Fachbereichs im Tagesgeschäft nicht immer gegeben.

Wie beseitige ich einen solchen Engpass?

ELIZA: Welche Antwort gefiele Ihnen denn?



QM: Die Bereitstellung und das Funktionieren der gewünschten Testumgebungen speziell für Tests, die mehrere Systeme berühren, gestaltet sich oft problematisch.

Wie bringe ich die Partner zusammen?

ELIZA: Mögen Ihre Partner Sie und umgekehrt?

Sie wissen's nicht
aber sie wollen's unbedingt wissen
also hilf egal wie.

Station 7: Am Orakelstein

Immer wieder treten Situationen auf, in denen eine exakte Aussage über die Qualität eines Systems nicht möglich ist aber das Management trotzdem eine möglichst fundierte Prognose erwartet. Gibt es dafür Möglichkeiten?

Qualitätsorakel oder Möglichkeiten einer Qualitätsprognose aus unvollkommenen Kenntnissen

1. Ausgangssituation:

- Der Test eines Systems ist beendet oder wurde abgebrochen.
- Das Management erwartet Aussagen über Qualität des Softwaresystems z.B.
  - wie viel und welche kritischen Fehler sind im produktiven Betrieb noch zu erwarten?
  - wann und in welchen Systemteilen werden diese Fehler auftreten?
  - welches Risiko birgt die Inbetriebnahme des Systems?

  Vorhersagen dieser Art werden nicht wirklich beherrscht. Dies zeigten aufgetretene Systemprobleme u.a. bei Toll Collect und ALG II obwohl die Systeme von renommierten Firmen erstellt und Tests durchgeführt wurden.
- Kenntnisse über das System
  - Wissen über die Software ist unvollkommen (unvollständig, verstreut über Dokumente und Wissensträger) und teilweise sogar veraltet und fehlerhaft.

    - Wissen über die Qualität der Software ist ebenfalls unvollständig (Güte der Testszenarien und Testfälle, Testabdeckung, Testqualität).
    - Wissen über eventuell noch enthaltene ‚kritische' Fehler ist im Prinzip nicht vorhanden (prinzipielle Begrenztheit der Restfehlerbetrachtung, keine Kenntnis von Möglichkeiten der Vorhersage).
- Dem Management ist oft nicht bewusst, dass die Entscheidung über den Einsatz des Systems auf der Basis unvollständiger Informationen erfolgt. Eine Risikobetrachtung, Qualitätsprognose und Verhaltensvorhersage für das System wird selten gemacht.

2. Möglichkeiten der Risikobetrachtung und Qualitätsprognose

- Restfehlerbetrachtung und Fehlschlagwahrscheinlichkeiten
    - Abschätzung der Restfehleranzahl (u.a. durch mögliche Anwendung von Fehlereinpflanzung (error seeding) oder Test durch zwei unabhängige Gruppen und Errechnung nicht gefundene Fehler)
    - Grundlage ist die Hypothese vom kompetenten Entwickler, d.h. die realisierten Funktionen weichen nur wenig von den Sollfunktionen ab
    - Problem: Annahmen der derzeitigen Modelle zur Zuverlässigkeitsberechnung stimmen nicht mit der Praxis überein oder sind nur schwer erfüllbar
    - Beurteilung der Fehlschlagwahrscheinlichkeit von Funktionen des Systems
        - auf der Basis von Merkmalen der Funktion wie Komplexität, Einsatzfrequenz etc.

    - durch Berücksichtigung menschlicher Faktoren wie Erfahrung Entwickler, Größe des Teams, Toolbeherrschung etc.

- Testanalyse
    - Beurteilung der Erstellung der Testszenarien und Testfälle (Aufbereitung der Requirements, Mitarbeit der künftigen Nutzer, Kompetenz der mitarbeitenden Nutzer)
    - Testverlauf
    - Fehlerkurven
    - Brainstorming mit ausgewählten künftigen Nutzern über kritische Fälle/Szenarien beim Systemeinsatz und Abgleich mit Testszenarien und Testdurchführung
- psychologische Beurteilung
    - Die Systemqualität wird beeinflusst durch die Zusammenarbeit und Qualität von Auftraggeber/Nutzer, Designer/Entwickler, Tester und Projektleitung/Management. Mittels geeigneter Fragebogen kann man die Selbstbeurteilung und die Beurteilung der anderen Partner ermitteln und bewerten.
    - Schwerpunkte sind dabei
        - gesetzte bzw. vermutete Prioritäten
        - Beurteilung der Arbeitsqualität (Korrektheit, Ordnungsliebe, Kompetenz etc.)
        - Risikobereitschaft (Vorsicht, Leichtsinn, Absicherungsverhalten)

3. Ziele des Qualitätsorakels

- Aussagen für das Management vor vorgesehenem Systemeinsatz über

    - Risiken und kritische Systemteile
    - möglicherweise zu erwartende Qualitätsprobleme
    - Vorhersage des Verhaltens des Systems
- Beurteilung möglicher Maßnahmen ‚in letzter Stunde' und deren potentielle Effektivität

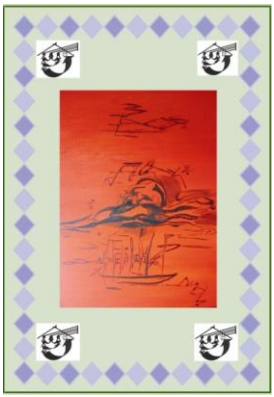
Traumzeit: Die großen Arkana des Software-Quality-Tarots

Im Traum fand ich ein vergilbtes Tarot-Kartendeck zusammen mit handschriftlichen Bemerkungen zur Interpretation des Spiels. Offenbar wurde das Tarot-Deck von einem Mitglied eines Testteams geschaffen. Benutzt wurde es scheinbar in Entscheidungssituationen wie der Abnahme eines System-release, in denen die durchgeführten Tests keine hinreichende Gewissheit über die Qualität des hochkomplexen Systems brachten und die Tester auf esoterische Mittel zur Qualitäts-prognose zurückgriffen. Über die Qualität der Prognose erfuhr ich im Traum nichts.
Im Folgenden sehen Sie das Legemuster des magischen Vier-ecks der Softwareentwicklung und 12 Blätter der großen Arkana des Tarot-Decks.

Kosten
(Auftraggeber)

Funktionsumfang
(Entwickler)

Zeit
(Projektmanager)



Qualität
(Tester)

das magische Viereck
der Softwareentwicklung
(als Tarot-Legemuster)

der Magier /
the magician /
der Softwareentwickler

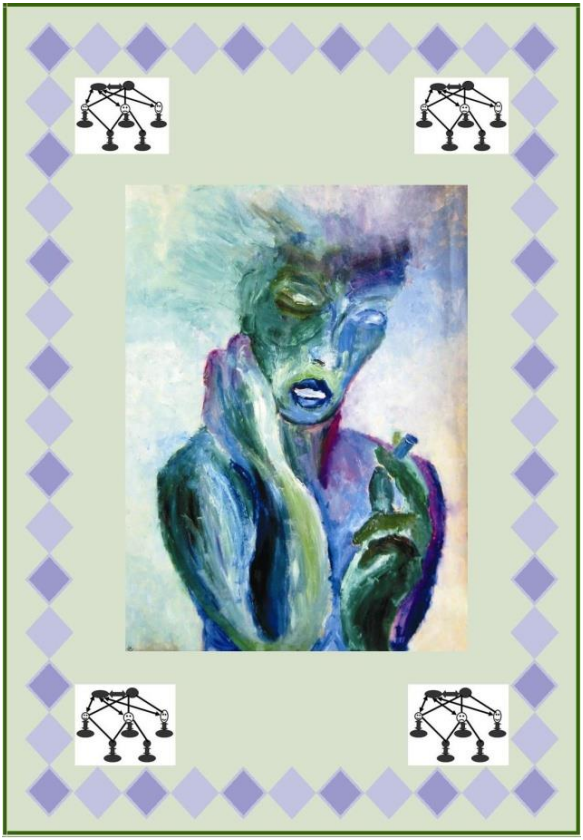
the doing is his life

der Herrscher /
the emperor /
der Projektmanager

law and order is his principle

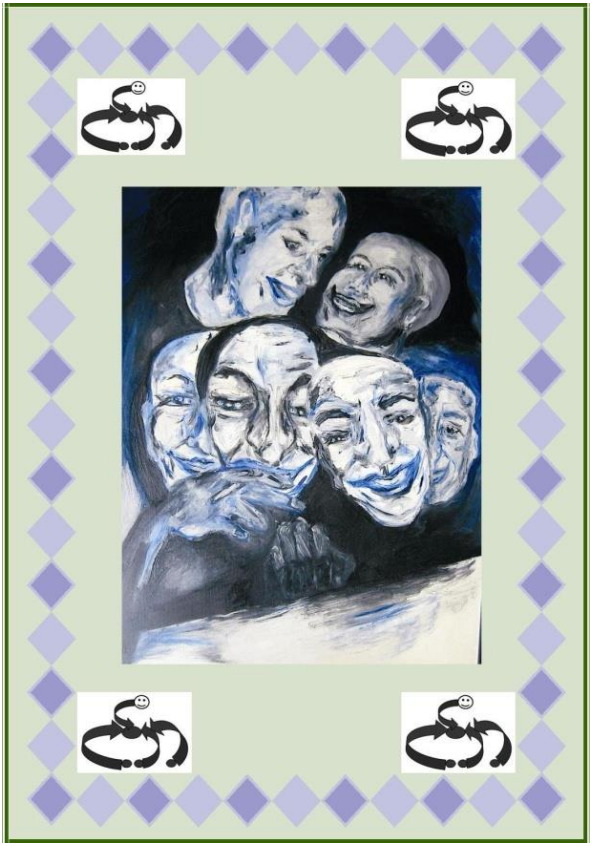
der Guru /
the guru/

is not enough to do your best;
you must know what to do and
then do your best



die weise Frau /
the wise woman

she sees all sides of life

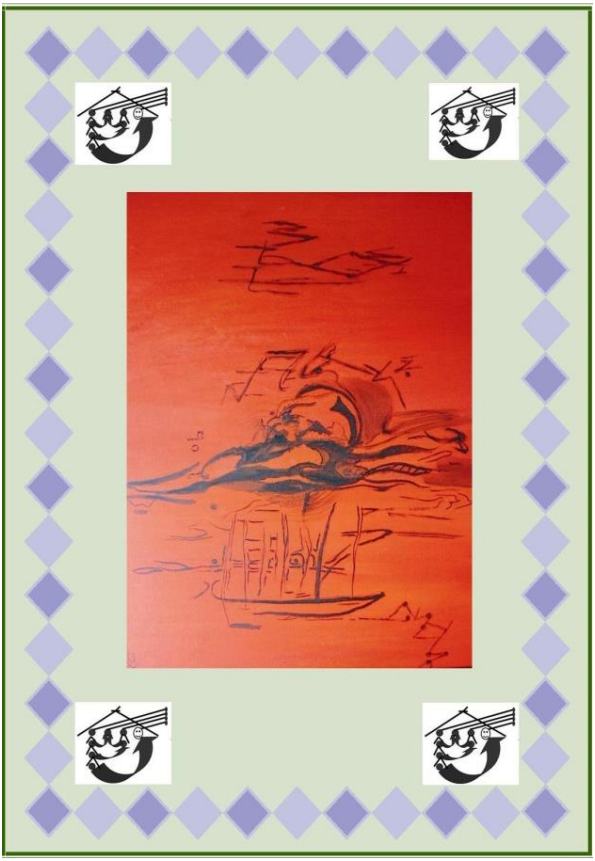


der Auftraggeber /
the client

he wants success at any cost

Kugel des Schicksals /
ball of fate

change is the chance



der Tester /

the tester

he finds the bugs - perhaps

das Gespenst der schlechten Qualität /
the specter of poor quality

horror of bad work

Absturz des Systems /
the Death

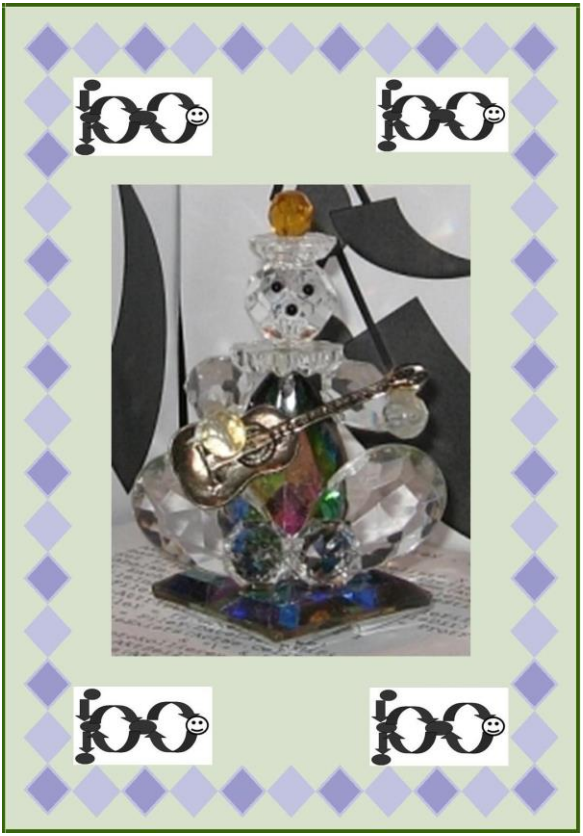
the worst case can coming everytime

der Fehlerteufel /
the devil of error

he is in the system - but where

der Drache der Fehler-
wahrscheinlichkeit /
the dragon of the error probability

he comes how a blizzard



der Narr /

the fool

a fool with a tool is still a fool

Station 8: Im Wald der Gesichter

Um vom Projektstress abzuschalten schrieb ich die folgende Shortstory über das rätselhafte Wesen ‚Computer‘, mit dem unser Team die Anforderungen des Fachbereiches möglichst effizient, kostengünstig und nutzerfreundlich realisieren sollte.

Die 5 Gesichter des Computers

Durch die halbgeöffneten Jalousien des Computerraumes blinzelte ein trüber Mond. Die Stadt lag in der Erstarrung der Nacht und nur in der Torwache des Werkes leuchtete ein schwaches Licht» welches Mel zeigte, dass er nicht allein war auf dieser Welt.
Er hockte zusammengekrümmt am Bedienpult des Computers und fluchte in Gedanken vor sich hin. Drei Tage dauerte nun schon die Havarie, ohne dass sie dem Fehler auch nur ein Stück näher gekommen wären. Immer wenn sie dachten, ihn nun endlich eingekreist zu haben, da brach plötzlich wieder alles zusammen. Es war ein Fehler von der übelsten Sorte, einer der sogenannten dynamischen Fehler, das bedeutet einer der nicht ständig auftritt, sondern nur in Abständen, durch Neustarten wieder verschwindet, aber dann plötzlich wieder da ist. Will man sich an den Fehler heran messen, so ist er wieder weg und man ist wieder so schlau wie zuvor. Alle sind sie schon gereizt, die Abteilungen des Werkes drängen auf ihre Drucklisten und Abrechnungen, der Leiter des Rechenzentrums lässt sich am Tage alle Augenblicke im Computerraum sehen, die Programmierer warten auf Rechnerzeit für Testzwecke und jeder, der zu den Computeringenieuren kommt, fragt mit je nach Mentalität hoffnungsvoller, skeptischer oder ärgerlicher Miene "Geht er denn schon wieder? Wie sieht’s aus?" Am liebsten möchte man sie alle wegjagen. Aber nun hat er

Nachtschicht und keiner ist da, sein zweiter Mann ist auch noch krank geworden, und er, Computeringenieur Peter Melchior, genannt Mel von allen im Rechenzentrum, ist allein, allein mit dem Computer, der so schnell rechnet und doch so anfällig ist. Er hat gesucht, gemessen, probiert, bald gejubelt und jedes Mal eine kalte Dusche bekommen, als der Fehler wieder auftrat. Es war zum Verzweifeln! Am liebsten würde er dem Bedienpult einen richtigen Tritt versetzen, so wie sein Vater früher immer auf das Videogerät geklopft hatte, wenn dies nicht so wollte, wie es sollte, aber ein unbewusster Instinkt und Respekt vor der ihm anvertrauten komplizierten Technik hielt ihn zurück. Im Moment lief der Computer wieder mal. Mel wollte seine Funktionstestprogramme durchlaufen lassen, und so blieb ihm nichts weiter, als der Dinge zu harren, die da kommen würden. Irgendetwas musste mit der Klimaanlage nicht stimmen, ihm war warm, und die Geräusche der Geräte waren so eintönig, monoton und einschläfernd. Wie beneidete er jetzt die anderen Einwohner der Stadt, die in ihren warmen und weichen Betten lagen und ohne Sorgen dem neuen Tag entgegen schlummern konnten. Er unterdrückte nur schwer ein Gähnen, und die Augenlider klappten immer wieder wie schwere Falltüren, die nicht gesichert waren, herunter.

Das Labyrinth

Plötzlich durchzuckte ihn wie ein Blitz ein Gedanke, "Das war‘s, das musste es sein, das war die Idee, Warum war er nur nicht schon früher darauf gestoßen, Archimedes konnte auch nicht glücklicher gewesen sein, als er sein berühmtes "Heureka, ich hab's" ausrief.
Wie war denn doch gleich die alte Geschichte gewesen, die von dem Ungeheuer im Labyrinth, Minotaurus hieß es wohl, und dem mutigen Helden, der sich in der Dunkelheit der Gänge, Sackgassen und Vorsprünge vorantastete, hinter sich den Faden, den ihm die Jungfrau Ariadne mitgegeben. Wie gut, dass er sich schon immer etwas für die Geschichten aus den Reichen der Alten, der Ägypter, Griechen, Römer und wie sie

alle hießen, interessiert hatte. Aber trotzdem! Hätte einmal einer gesagt, womöglich sein ehemaliger Literaturlehrer, der alte Reeßing, der manchmal schon etwas wunderlich kauzig die Geschichten erzählte, ja hätte er jemals zu ihm gesagt, außer seinem ständigen» "Melchior passen sie doch auf, wo sind sie denn nur schon wieder mit ihren Gedanken, Sie werden schon noch sehen, was sie davon haben," , hätte dieser ihm gesagt, dass ihm in seiner Arbeit als Computeringenieur eine griechische Sage helfen würde, für verrückt hätte er den Alten ge halten, gedacht, dass es ihn wohl nun gänzlich erwischt habe, schade eigentlich, wo er doch so kurz vor der Pensionierung stand. Nun war es doch so. Warum war eigentlich noch niemand darauf gekommen? Lag es daran, dass die Menschen lieber Berichte von Astronauten, Weltraumreisen, Sportwettkämpfen und anderen modernen Abenteuern lasen oder über Video verfolgten. Waren die Alten, war denn die Herkunft der heutigen technisierten Welt, die Mühen und Sorgen aber auch die Vergnügungen unserer Altvordern so uninteressant und der Kenntnis unwert?

Aber er würde schon dafür sorgen, dass man wieder an sie dachte und ihnen auch wieder Interesse entgegenbringen würde, wenn er erst seine Idee verwirklicht hätte, sie überall bekannt wäre, alle Computerzentren sich dafür interessieren und ihn um Nutzung angehen würden.

Dabei war die ganze Sache doch so bestrickend einfach. Man musste nur Augen haben, um zu sehen und zu begreifen, Was *wa*r er denn anderes als ein riesiges Labyrinth, sein großer Computer? Ein Labyrinth, bestehend aus tausenden Schaltkreisen, dem Gewirr der Verdrahtungen des Kernspeichers und der Steuerung der Zentraleinheit des Computers. Die Impulse, eingegeben zur Ausführung von Befehlen, sie wandern durch die kilometerlangen. Leiterbahnen, überqueren tiefe Gräben, in dem sie Kondensatoren als Brücken benutzen, keuchen Bergpfade hinauf, steilere und sanfter ansteigende je nach Größe der Widerstände, Sie müssen Einbahnstraßen einhalten, zu denen sie Gleichrichter und

Dioden zwingen und alles dies, um an einem Ziel ankommend eine Aktion auszulösen oder auch nur eine Information zu überbringen, sie abzuladen wie ein müder Wanderer seinen Rucksack nach schwerer Tour abwirft. Erschöpft, aber es ist geschafft.

Was aber ist, wenn die Brücken einstürzen, die Gipfel der Berge schier unbezwinglich zum Himmel ragen, Einbahnstraßen zu Sackgassen werden und Spalten sich auftun im scheinbar sicheren Weg? Geht es den Impulsen nicht wie dem Wanderer, der nicht weiter kann, aufgegeben hat, nachdem er ewigkeitenlang umherirrte, um neue Wege zu suchen, und nun den Tod erwartet. Was ist die Havarie anderes als der Tod der unterwegs befindlichen Information, ihr Versinken und Vergehen in den aufgetanen Grüften. Havariebeseitigung ist Brückenwiederherstellung, reparieren von Straßen und Wegen. Aber wo sind die Stellen, an denen einzugreifen ist? Wie soll man sie finden? Messen, probieren, löten, wieder probieren, messen, löten, austauschen von Teilen; es ist als wolle man zu Fuß den Amazonasurwald durchdringen und habe Kompaß und Machete vergessen. Gibt es denn etwas Besseres?

"Ja, ich weiß es, jetzt weiß ich es. Nicht zu Fuß darf man durch das Labyrinth wandern, auch nicht mit Fahrrad oder Mobile, nein der schnellsten Verbindung muss ich mich bedienen, wenn ich Erfolg haben will. Der Gedanke selbst muss durcheilen und er wird mir in Windeseile die Nachricht bringen, wo die eingestürzten Brücken oder eingefallenen Straßen und Wände sind.“

Fieberhaft machte sich Mel in der Werkstatt der Computeringenieure an die Arbeit. Es war ihm zumute wie in einem Rausch. Leicht, beschwingt und doch zugleich präzise und genau bis auf das Mikron arbeitete sein Gehirn,

"Ja, so musste es gehen. Ein Metallreif, der den Kopf umschließen würde, zwei Elektroden, die über den Umweg der Bioströme den Gedanken aufnehmen und nun der neueste selektive Verstärkerschaltkreis, gekoppelt mit zwei Entzerrern.“

Er baute, lötete, verbrannte sich die Finger, flüchtig pustend und fluchend griff er sich das nächste Teil. Es war ihm, als müsse er es unbedingt schaffen, es könne gar nicht anders sein. Mel merkte nicht, wie die Stunden verrannen. Die Zeit zerfloss wie Butter in der glühenden Sonne seines Eifers und seiner rastlosen Tätigkeit. Er schaute auf das Chronometer an seinem Kittel und erschrak. In einer halben Stunde musste seine Ablösung kommen*

Die nächste Schicht war dann da und sie würde ihn fragen: "Wie steht‘s? Geht er wieder?"

Immer die gleichen Prägen und sollte er wieder kein Ergebnis verweisen können und was wäre, wenn sie ihn fragen, was er die ganze Schicht gemacht hatte. Für verrückt würden sie ihn halten, schimpfen, weil er nicht solide gearbeitet hatte, eventuell die ganze Schicht vergaukelt hatte. Aber er würde es ihnen zeigen.

Nur noch diese zwei Verbindungen löten, dann würde er ihn ausprobieren, seinen Labyrinthographen, und er würde den Fehler haben. Augen würden sie machen, seine Kollegen. Der Leiter des Rechenzentrums wird ihm auf die Schulter klopfen und sagen:

"Ich hab's ja schon immer gewusst, dass sie doch nochmal etwas ganz besonders bringen werden." Oh, der könnte ihm viel erzählen, weiß er doch genau, dass er ihn, Mel, immer für einen nur durchschnittlichen Computeringenieur gehalten hatte. Sei’s drum, nach diesem Erfolg wird ihm verziehen.

Mel ging zum Bedienpult des Computers. Je näher er kam, desto langsamer wurde sein Schritt, Sein Optimismus, seine Siegesgewissheit wich Zweifeln.

"Wenn es nun doch nicht klappt, wenn ich mich nur verrannt habe, habe ich wirklich alles bedacht, ist es vielleicht besser, das Ganze als Hirngespinst zu vergessen?"

Quälend schlichen die Sekunden dahin, zögernd stand er da, den fertigen Labyrinthographen in der Hand, die Lämpchen am Bedienpult schienen höhnisch zu blinzeln: "Siehst Du, immer noch kaputt, nun zeige uns doch mal, wo der Fehler liegt."

Er gab sich einen Ruck.
"Jetzt aufgeben, nein und nochmals nein, jeder, aber nicht er, Peter Melchior."
Langsam setzte er sich im bequemen Sessel dos Operators zurecht, legte sich den Metallreif des Gerätes um die Stirn, die beiden Elektroden an die Schläfen. Er schloss das Gerät mit dem Verbindungsstecker an den Prüfausgang des Computers am Pult an. Ein letzter Blick noch auf die Schaltung des Labyrinthographen, seine Hand näherte sich dem Schalter und jetzt "Ein!".
Ein leises Summen ertönte, der Labyrinthograph begann zu arbeiten. In Mel‘s Hirn jagten sich Fetzen von Bildern, ... der Hörsaal an der Universität mit dem Professor für Computerkunde, das Gesicht zu einem Grinsen verzerrt; der Strand der Ostsee, stürmische Wellen, Eva mit aufgelöstem Haar, wieder seine kleine Heimatstadt, die Schule in der er seine Streiche verübt hatte, aber auch die Sage vom Labyrinth kennen gelernt hatte und immer wieder ein Bild von einem schier undurchdringlichen Gebüsch oder Waldrand. Dieses Bild verdrängte immer mehr alle anderen, bis es seine Vorstellung völlig einnahm.
"Ich muss mich konzentrieren, ich muss mich konzentrieren, ich muss einen Weg finden ..." murmelte er mit geschlossenen Augen. Er keuchte vor Anstrengung, die Stirnadern schwollen an. Plötzlich öffnete sich ein Gang, er betrat ihn aufatmend und eilte hindurch, jagte zwischen schimmernden Wänden entlang, über kühn geschwungene Brücken, auf glatten Straßen und über steile Serpentinen. Er wurde schneller und schneller.
"Alles in Ordnung" signalisierten seine Gedanken, die die Pfade des Labyrinths durchstürmten, überraschend wurde das Licht schwächer, die Wände rückten näher heran, Löcher machten sich auf dem Weg bemerkbar. Ihnen bei seinem Sturmlauf ausweichend, bemerkte er nicht, dass er in eine Höhle geriet, die immer dunkler wurde. Er stieß irgendwo gegen etwas Steiniges, Spitzes. Es ächzte und krachte, der

Berg schien aufzustöhnen, Brocken lösten sich, immer schneller kamen sie herab, der Rückweg war schon versperrt. Ein Schlag und ein mächtiger Brocken traf ihn genau am Kopf. Alles war aus und vorbei.
Mel blickte sich verschlafen um. Was war denn eigentlich gewesen?
War er wirklich eingenickt? Sein Kopf musste auf das Bedienpult gefallen sein, seine ganze Stirn tat noch weh. Was hatte er nur konfuses geträumt? Ja egal, ran musste er jetzt, damit er wenigstens bis Schichtende noch etwas vollbrachte.

Die Sinfonie

Mel setzte sich bequemer im Drehsessel am Bedienpult zurecht. Einige schnelle Bewegungen mit dem Sessel nach links, einige nach rechts, langsam ließ er sich auspendeln. Es war sinnlos, jetzt in übergroße Hektik und Betriebsamkeit zu verfallen. Ein schneller Blick auf das rhythmische Flackern der Anzeigelämpchen am Pult, auf das leere Blatt der Sehreibmaschine zur Bedienerverständigung mit dem Computer überzeugten ihn davon, dass seine Funktionstestprogramme noch liefen und dass ihm nichts übrig blieb, als bis zu deren Ende abzuwarten, ehe er etwas Konkretes unternehmen konnte. Er lauschte den Geräuschen im Computerraum, Alle waren sie noch da, die ihm tausendfach vertrauten Laute, die ihn täglich umgaben. Von links ertönte das leise, melodische Surren der Laufwerke der Magnetplattenspeicher, von rechts her mischte sich der dumpfe Bass der Magnetbänder hinein, manchmal bei schnellen Vor- und Zurückbewegungen eines Bandes unterbrochen durch ein grillenähnliches Zirpen, wie es sich bei Spaziergängen in lauen Sommernächten entlang reifer Kornfelder vernehmen lässt.
Vom Schnelldrucker her tönt alle paar Minuten ein helles Stakkato, wenn die Meldungen über den Fortgang seiner Programme erscheinen. Jetzt wieder lässt der Lochkartenleser seine dumpfen Schläge hören, mit denen er in Windeseile die nächsten Befehlskarten einliest.

Bin Blick zur Seite, beruhigend geistern die rhythmischen Lichtspiele über das Lampenfeld des Computerpultes. Alles läuft im Augenblick wohlgeordnet; kein Misston ist im Ganzen, abgerundet, wohlgestaltet wie ... Ja, womit soll man dies vergleichen?
Es ist so wie, ja eben wie in dem Konzert, das Mel vorige Woche erst besucht hatte. Zusammen waren sie hingegangen, er und seine Kollegen, skeptisch waren die meisten gewesen. Smetana wurde gespielt, "Die Moldau" eines seiner sinfonischen Werke.“ Mel fiel ein, wie einer noch geunkt hatte "Vergesst nicht die Stiefel anzuziehen, damit ihr trockene Füße behaltet. Das wird sicher die reinste Wassermusik werden." Sie hatten gelacht und waren gekommen, ungewohnt das Sitzen und Hören, angestrengtes Hören, ein Ohr tat es hierbei nicht, aber was sie gesagt hatten, dazu standen sie. Wer wird denn schon aufgeben während eines Konzertes. Die Wellen der Musik hatten geplätschert, lustig waren einige vorwitzig über kleine Steinchen gehüpft, hatten sich am Ufer gebrochen, und weiter ging das Spiel. Mel war überrascht gewesen, eine solche Musik hatte er nicht erwartet, Musik, die ihn den Fluß spüren ließ, mit seinen sanften Uferhängen, den kleinen Stromschnellen, den lustigen Menschen an seinen Seiten, den Fluß, wie er fröhlich dahin strömte, aber auch wie er schwer arbeitete, Mühlräder trieb, Schleusen füllte und Schiffe auf seinem Rücken ihrem Ziel entgegentrug. Nach dem Konzert waren sie alle beeindruckt und vorgenommen hatten sie sich, dass es nicht das letzte gewesen sein sollte. So wie der alte Smetana mit seiner wunderbaren Musik ihm die Moldau erschloss, so schien ihm, erschloss sich ihm der Computer durch seine Musik, seine Arbeitssinfonie. Ja, so konnte man es auch sehen. Der Computer bildete mit seinen Ein- und Ausgabegeräten und externen Speichermedien ein Orchester, kein Sinfonieorchester zwar und keine Gewandhauskapelle, aber doch ein Orchester, das er, Mel, dirigieren musste und durfte und in dessen Spiel es seine Aufgabe war, mit seinem Taktstock alle Misstöne zu beseitigen. War es nicht herrlich, Dirigent eines

so komplizierten Klangkörpers zu sein, wie viel schöner und poetischer klang das Wort „Dirigent" doch als die sachlichen Bezeichnungen "Computeringenieur" oder '‘Computertechniker‘. Wenn nur die Missklänge nicht wären. Jeder falsche, unkorrekte Lauf eines Teils, eines Gerätes verursachte falsche Töne in der Ton- und Klangfülle, die ihn umgab. Ein wirklicher Dirigent, der hatte es einfach. Er hört den Sünder, ein Blick, ein Stirnrunzeln, ein kurzer Wink mit dem Taktstock, leichtes Klopfen und dann „da capo". Aber er, Mel, hatte keinen Wundertaktstock und schon des Heraushören des Sünders, der den Misston verursachte, war nicht vergleichbar mit der Arbeit des Dirigenten eines Orchesters.
Aber warum musste er denn so schnell aufgeben? Vielleicht könnte sich aus dieser vagen Idee doch etwas machen lassen. "Natürlich‘’ so überlegte er, "reichte es nicht aus, wenn ein ganzes Gerät als Sündenbock identifiziert wurde. Das konnten er und seine Kollegen auch ohne Musik. Die Schwierigkeiten fingen ja dann erst an. Trotzdem, wie in Urgroßvaters Zeiten arbeiteten sie ja doch auch nicht mehr. Nicht Schräubchen, Widerstände und Transistoren wechselten sie. Damit ging es schon lange nicht mehr. Nein, Baugruppen und hochintegrierte Leiterplatten tauschten sie aus, aber erst einmal musste «an die defekte Stelle finden".
Langsam begann die Idee in seinem Hirn greifbare Gestalt anzunehmen. "Wie wäre es denn, wenn die Arbeit der Baugruppen in Musik übersetzt würde? Eine Baugruppe entspräche einem Musiker und ein Gerät des Computers wäre dann ein Teil, eine Gruppe, des Orchesters". Er stellte sich schon vor, wie er seinen Kollegen die große Schau ankündigen würde: "Werte Damen und Herren! Sie erleben eine historische Stunde. Erstmalig auf der Erde und im ganzen Sonnensystem hören sie heute ein Stück in C-Moll, C wie Computer. Angeschlossen sind sämtliche Rundfunk- und Videostationen. Dirigent ist der berühmte Peter "Mel" Melchior." Applaus. Donnerndes Klatschen und dann plötzliche Stille, wenn er den Schraubenzieher als Taktstock hebt. Aber Spaß beiseite, Mel

der Spinner würden sie wieder sagen.
Es würde keine Spinnerei sein, Augen würden sie machen, nicht mehr zuklappen würden ihre Münder, wenn...
"Man muss das Eisen schmieden, solange es warm ist" sagt schon ein altes Sprichwort, also musste er beginnen, solange die Idee frisch und neu war. Mel ließ die Testprogramme Testprogramme sein. Er ging in die Werkstatt der Computeringenieure und sichtete ihren Fundus auf Verwendbarkeit für seinen Plan. Es war aber doch nicht alles da, was er wahrscheinlich brauchen würde. Zum Glück kannte er die Regelungsingenieure des Werkes gut, diese würden ihm schon helfen. Einige Teile hatte er vielleicht auch zu Hause in seiner Bastelkammer. Aber ehe er anfing zu bauen, musste er einen Schaltplan aufstellen, sonst würde sich alles verwirren und das Ganze war umsonst. Abnehmer für die elektrischen Impulse an den Endstellen der Baugruppen müssten da sein, Verstärker und Übertrager zu einem Misch-, Filter- und Selektierpult, dann von diesen ein Übergang zum eigentlichen Schallübertragungssystem, das schließlich die Musik hörbar machen würde. Mel nahm sich Blätter aus dem Wandschrank und begann zu zeichnen, zu radieren, wieder zu zeichnen...
Da zuckte er zusammen, eine Berührung hatte ihn aufgeschreckt. ’’Ach du bist es, hast du mich vielleicht erschrocken” sagte er zu seinem Freund und Kollegen Igor, der neben ihm stand. ’’Was ist denn los, du hast wohl die Zeit vergessen?” meinte Igor staunend, "Deine Schicht ist zu Ende, Ich löse dich ab. Was machst du denn da eigentlich?”
"Ach nichts, ich wollte nur mal kurz etwas für meine Hausdisco skizzieren, mit der ich unsere Wohnscheibe zum nächsten Fest überraschen möchte. Da drinnen tut sich sowieso nichts. Die Testprogramme laufen fehlerlos und sowie andere Sachen kommen, geht *es* wieder schief. Na, toi, toi, toi möglicherweise findest du den Fehler. Ich gehe jetzt.”
Mel packte sorgsam alle Skizzen zusammen und ging nach Hause.
Eigentlich hätte er müde sein müssen, aber der Gedanke an

sein Computocord. so hatte er die ihm vorschwebende Apparatur schon getauft, ließ ihn nicht los. Selbst das Rieseln des warmen Duschwassers über seinen Körper, das ihn sonst immer in einen angenehm müden und schläfrigen Zustand versetzte, zeigte diesmal keine Wirkung. Er legte sich trotzdem hin, aber über einen unruhigen Halbschlaf, ständiges Aufschrecken und Erwachen, ruheloses Hin- und Her- Wälzen kam er nicht hinaus. Mel wer froh als er wieder aufstand. Auf seine Toilette verschwendete er keine Zeit. Er machte sich gleich wieder an seine Arbeit und zeichnete, verglich, schaute in Nachschlagewerken und Lehrbänden seiner MikrofiImbibliothek nach, radierte wieder, entwarf neu, überprüfte, rechnete nach und zeichnete wieder. Kaum, dass er sich eine kurze Pause gönnte, um einen Happen zu essen und einen Schluck zu trinken. Am frühen Abend hatte er es geschafft.

Es schien vollbracht, zumindest auf dem Papier war alles fertig. Nun war das Computocord nur noch zu bauen und probieren musste er natürlich die ganze Sache. Jetzt erst merkte er, wie er müde war, wie er langsam zusammenfiel, die gewaltsam angestauten Energien und Kräfte aus ihm Zug um Zug entwichen, wie aus einem porösen Luftballon. Seine nächste Schicht, die Nachtschicht, fing bald wieder an. Wenn er durchhalten wollte, wenn er in dieser lacht sein Computocord bauen wollte, dann musste er vorher nochmals schlafen» Er musste einfach.

Als Mel ins Rechenzentrum kam, erwartete ihn sein Spätschichtkollege schon ungeduldig,

"Wo bleibst du denn? Ich muss heute unbedingt pünktlich weg."

"Was gibt es denn Neues?""fragte Mel.

"Eigentlich nichts. Wir wissen immer noch nichts, aber der Computer arbeitet schon ein paar Stunden einwandfrei. Wir haben aber nichts daran gemacht und gerade deshalb befürchten wir, dass der Fehler wieder auftritt, Es wäre besser, wir wüssten, worum er wieder läuft, dann wüssten wir auch, wo der Fehler war. So können wir nur warten. Na dann alles Gute

für deine Schicht."
Kaum war der andere fort, so ging Mel kurz in den Computerraum, um den diensthabenden Operator zu begrüßen.
"Ich bin in unserer Werkstatt", sagte er "wenn etwas ist, so brauchst du mich nur zu holen."
In der Werkstatt machte sich Mel gleich an die Arbeit. Seinen Raumtonrecorder mit Mischpult zum Abspielen der zugeleiteten Impulse hatte er mitgebracht. Es kam nun nur noch darauf an, die Abzapf-, Verstärker- und Ummodulierungsstrecke für die im Computer kreisenden Signale zu bauen. Da er den Schaltplan schon fertiggestellt hatte, war dies scheinbar eine Routinesache für einen geübten Computeringenieur. Er arbeitete schweigsam und angestrengt, bemüht, sein Denken in sachlichen Bahnen zu halten und jegliche Euphorie zu vermeiden. Nach fast drei Stunden hatte er es geschafft. Die Übertragungsstrecke war provisorisch aufgebaut, um Fragen der kompakten Unterbringung würde er sich später kümmern, erst- mal musste es funktionieren. Langsam wurde ihm doch anders zumute. Es fiel ihm schwer, beherrscht zu bleiben. Deutlich fühlte er das Schlagen seines Herzens. Es war Mel, als wenn ihm ein Kloß im Halse steckte. Er überlegte, wo er sein Computocord am besten ausprobieren sollte. Das Beste wäre, er würde es am Schnelldrucker anschließen, der der Tür der Werkstatt am nächsten war. So würde er das Gerät in der Werkstatt lassen können und damit lästigen Fragen des Operators aus dem Wege gehen. Das Anschließen der Abzapfkontakte nahm nur wenig Zeit in Anspruch. Mel zögerte noch einmal vor dem entscheidenden Augenblick, dann ging er wieder in seine Werkstatt und drückte kurzentschlossen auf den Startknopf, nachdem er die Kopfhörer aufgenommen und den Lautsprecher abgeschaltet hatte. Er lauschte, regelte Tonhöhe und Lautstärke ein und vernahm ... außer leisem Knacken gar nichts.
Sollte er sich so geirrt haben, war sein ganzer Plan, seine wunderbare Idee alles umsonst. Nein, das konnte nicht sein, das durfte nicht sein.

Er beschloss, die Übertragungsstrecke nochmals zu überprüfen. Gründlich, Stück für Stück, betont langsam und sicher ging Mel Schaltung und Aufbau noch einmal durch. Halt! Hier stimmte etwas nicht. Wahrscheinlich hatte er vorhin doch in der innerlichen Aufregung die Anschlüsse vertauscht. Er wechselte die Verbindungsstücke. Jetzt war alles fest. Zweite Probe! Wieder drehte er ganz vorsichtig auf. Sein Gesicht wurde abwechselnd rot und blass. Schweißtropfen traten auf seine Stirn. Er hatte es geschafft, geschafft.

Fremdartige, nie gehörte, aber doch nicht unschöne Tonfolgen drangen aus seinem Kopfhörer. Er war der erste Mensch, der die Musik des Computers hörte. So musste dem alten Beethoven zumute gewesen sein, als er nach seiner Taubheit seine eigenen Sinfonien wieder hörte. Doch dieser Vergleich war vielleicht nicht ganz treffend. Vielleicht war er vielmehr mit dem Entdecker zu vergleichen, der am Gestade eines neuen Erdteils steht, überwältigt von dessen Größe und Andersartigkeit.

Mel konnte sich fest nicht losreißen, die Töne überspülten seine Gedanken wie die Meeresbrandung den Sand des Ufers, aber dann sagte er sich, dass dies noch kein endgültiger Beweis war. Erst wenn er am Herz des Computers, an seiner Zentraleinheit, gelauscht hatte und auch dort ähnliches erlebte, dann war es wirklich zu glauben. Voller Ungeduld wartete er, bis die Pause des Operators herankam.

"Geh ruhig schon in den Speiseraum", sagte er, "ich habe jetzt etwas Luft. Ich werde dich vertreten, du kannst dir ruhig Zeit lassen. Vielleicht willst du auch noch eine Tasse Kaffee trinken."

Kaum war der Operator zur Tür hinaus, so schlappte Mel eilends seine Gerätschaften zum Bedienpult. Er öffnete die seitliche Wand der Computer-Zentraleinheit und begann seine Abzapfer überall anzubringen. Nachdem **er** fertig war, sah er sich an, was an Aufträgen für den Computer da war.

Er wählte zwei aus und entschloss sich hierbei für zwei Extremfälle, eine ökonomische Aufgabe mit wenig Berechnungen

und vielen Ein- und Ausgaben und ein sehr rechenintensives Programm zur Ermittlung von Zufallsgrößen.
Er kontrollierte seine Übertragungsstrecke nochmals, vergewisserte sich, dass alle Verbindungen fest saßen und setzte die Kopfhörer auf. Dann startete er die ökonomische Aufgabe. Die Töne, die er vernahm, waren leicht, hoch, ja wirkten beinah heiter und lustig. Sie erinnerten ihn an das Tirillieren und Zwitschern der Vögel vor seinem Elternhaus, dem er als kleiner Junge immer gelauscht hatte, ohne zu verstehen, wo es herkam und wer die schönen Töne hervorbrachte.
Ein heiteres, beschwingtes Gefühl der Zufriedenheit bemächtigte sich seiner. Morgen würde er sein Computocord den Kollegen vorführen, staunen würden diese und sich freuen. Er sonnte sich schon im Vorgefühl der Anerkennung, die man ihm zollen würde.
Die ökonomische Programmfolge war vorüber. Er hatte noch Zeit. Mel dachte, dass man doch einmal hören müsste, wie so eine wissenschaftliche Aufgabe klingt. Hoffentlich waren es nicht lauter Quaktöne wie ein Froschkonzert oder Donnergrollen wie beim Gewitter, das sein Professor immer hatte über seine unfähigen, faulen und durch ihr Verhalten auffallenden Studenten hatte ertönen lassen.
Aber nein, die Töne kamen schwer und getragen, zeitweise langsam anschwellend, mit kurzem Jubilieren dazwischen, vielleicht beim Errechnen von Zwischenergebnissen oder dem Erreichen von Konvergenzschranken,
War es nun also geschafft? Die Anspannung ließ nach und ein lähmendes aber frohes stolzes Gefühl nahm von ihm Besitz. Aufatmend streckte er sich im Sessel des Operators.
"Aber was war das?"
Kurze, unregelmäßige störende Töne erklangen in der harmonischen Tonfolge. Sie wurden lauter, zahlreicher, wuchsen immer mehr, schwollen zu einer Lawine an. Eine Kakophonie scheußlicher Missklänge brach über ihn herein. Mel hielt die Sessellehnen starr umkrampft, sein Gesicht verzerrte sich, die Tonlawine verursachte ihm rasende körperliche Schmerzen. Er

riss seine Hände hoch, zerrte die Kopfhörer herunter, den Stecker aus dem Recorder und war wie betäubt von der plötzlichen Stille... Sein Blick fiel auf das Bedienpult. Der Computer stand, die Alarmlämpchen leuchteten grellrot: "Havarie, nicht korrigierbarer Fehler."
Wo war sein Computocord, wo der Raumtonrecorder? War er schon so fertig, dass er nur noch träumte?"
Langsam begriff Mel, dass alles offenbar nur ein Traum gewesen war.

Der Stromfresser‘

Sein Blick irrte vom Bedienpult durch den Computerraum, überflog die einzelnen Gerätegruppen, die stumm, wie eine Herde metallener Schafe verharrten, der der Leithammel abhanden gekommen ist. Mels Hände zitterten leicht, als er sie auf den kühlen Rahmen des Pultes legte.
"Du schaffst mich nicht", dachte Mel "du nicht. Du bist ja doch nur eine dumme, tote Apparatur. Was bist du denn ohne mich, der dich pflegt, wartet und ein- und ausschaltet, der deine Gelenke immer wieder schmiert? deine Nerven beruhigt und in Schuss hält? Ein unnützer Haufen Schrott wärst du, der so langsam aber sicher verrotten würde. Warum bist du nur so spröde zu mir. Kannst du denn dich nicht ein bisschen benehmen, immer diese Havarien, Das gehört sich einfach nicht für ein sonst gut erzogenes Wesen.”
Mel schien, als blinzele ihm dar Lämpchen der Hauptstromversorgung schelmisch und zugleich etwas traurig zu,
"Nun gut, ich will es noch einmal im Guten versuchen, ehe ich in deinem Inneren herumwühle.“
Zärtlich strich er über das glatte kühle Gehäuse des Herzstücks des Computers, des Hauptspeichers, verweilte kurz bei den Tastenreihen der Bedienerschreibmaschine, und dann drückte er entschlossen die Taste NEUSTART. Die Havarielämpchen erloschen schlagartig und das rhythmische Spiel der anderen Kontrollleuchten begann das Computerpult wieder zu beleben. Die peripheren Geräte nahmen ihre Arbeit wieder auf, und ihre

Melodie erfüllte erneut den Computerraum. Der Rechner schien ruhig und wohlgeordnet vor sich hin zu arbeiten. Die Minuten schlichen dahin und der Sekundenzeiger der großen Uhr an der Stirnseite des Raumes, Mels Platz genau gegenüber, lief, als wäre er auf Watte gelagert. Die Zeit verging in winzig kleinen Scheibchen, von denen unzählige aneinandergereiht erst ein wahrnehmbares Stück ergeben. Mel hatte einmal von der Kunst des Blattgoldherstellens gelesen, bei der selbst in der jetzigen hoch technisierten Zeit, der Meister mit einem Holzhammer wieder und wieder die goldenen Blätter bearbeitete, sie immer weiter verfeinerte mit zigtausend Schlägen, bis sie so fein waren, dass erst viele von ihnen übereinandergelegt einen Millimeter ergaben. Die Operationen des Computers, in Millionstel Sekunden ablaufend, brachten das gleiche Phänomen, von dem ihm nur die Blätterstapel, die Sekunden, sichtbar wurden,

Mel schrak aus seinem Sinnieren hoch, aber nichts war passiert, alles schien unverändert zu laufen. Er kam aber von dem Gedanken nicht los, dass der Computer langsamer lief. Die Lämpchen schienen ihm nur noch unlustig zu flackern, ein Hauch von Trostlosigkeit und Pessimismus überzog alles wie mit einem schmutziggrauen Schimmer. Mel überlegte angestrengt, und da kam ihm auch ein Einfall. Er würde den Computer einfach mal ein bisschen muntermachen; er würde ihn etwas kitzeln.

Ihm war bekannt, dass er kurzzeitig bis zu zehn Prozent die Spannung relativ gefahrlos erhöhen durfte und dass sich dies stimulierend auf die Arbeit seines Zöglings auswirkte. Er griff zu dem Spannungsregler und gab mit einem leichten Drehen dem Computer etwas mehr Spannung, "Saft", wie sie in ihrem Technikerjargon sagten. Siehe da, es wirkte, die Lämpchen leuchteten lebhafter, ihr Wechselspiel beschleunigte sich. Mel straffte sich innerlich, wie unter einer verborgenen Anstrengung und lächelte leicht, weich und sogar etwas zärtlich, "Siehst du, du musst nur wollen Alter, dann geht es schon. Rechne schön fleißig, Mel wird dir auch immer ein bisschen

Saft geben."
Eine Weile hielt das flottere Tempo an, dann ließ es allmählich wieder nach,
"Was ist denn nun schon wieder? Hast du denn schon wieder keine Lust mehr?"
Mel erhöhte die Spannung noch eine Kleinigkeit, und erneut beschleunigte sich die Arbeitsgeschwindigkeit des Computers.
"Du bist mir ja ein ganz Schlauer, willst mir wohl stückchenweise mehr Strom abluchsen. Du benimmst dich ja ganz wie der Esel, auf dem ich im letzten Urlaub geritten bin."
Mel erinnerte sich der Geschichte, die sich bei seinem letzten Touristentrip ins Balkangebirge zugetragen hatte, nur ungern, da er hierbei nicht die rühmlichste Figur abgegeben hatte. Der Vorfall hätte ihm um ein Haar den Spitznamen "Jockey" eingebracht, Er machte mit seinen Freunden eine Tageswanderung durch die wildromantischen Täler und Schluchten des Balkans. Die Gegend war herrlich, das kräftige Blau des Himmels stach von dem etwas staubigen Grün der bewaldeten Hänge in einer grellen Schärfe ab, die die Augen schmerzte. Die Intensität der Farben war viel stärker, als er es aus seiner mitteleuropäischen Heimat mit ihrer gedämpften, vom Menschen geprägten und von der Sonne nicht so verwöhnten Landschaft kannte. Sie waren dabei, einen kleineren Berg zu ersteigen, um sich einen besseren Ausblick zu verschaffen, als er plötzlich wegrutschte, den Halt verlor, mit den Armen in der Luft ruderte und sich dann in einer großen Staubwolke am Fuß des Berges wiederfand, Seine Freunde kamen zu ihm herunter geeilt und untersuchten ihn besorgt. Es war ihm außer einigen Hautabschürfungen scheinbar nichts passiert, als Mel aber aufzustehen versuchte, da bemerkte er erst das Malheur. Er hatte einen Fuß verstaucht, ausgerechnet das noch und zu allem Glück waren sie schon einige Kilometer von ihrer Unterkunft und der nächsten Siedlung weg. Nun war guter Rat teuer.
Seine Freunde versuchten ihn zwischen sich zu nehmen und ihn abzustützen, aber nach einigen hundert Metern mussten sie

wieder aufgeben»
Sie überlegten, ob sie eine Trage machen sollten, aber womit? Hier versagten die Kenntnisse aus den Abenteuerbüchern ihrer Kindheit. Außerdem war es dort immer einfach gegangen, "Sie fertigten sich eine Trage", hieß es. Aber wie und womit? Sicher hatte kaum einer der Autoren sich jemals in der Lage befunden, selbst wirklich eine Trage bauen zu müssen. Da zeigte Arto auf eine entferntere Wiese. "Seht mal ein Esel!"
Sie schleppten Mel zur Wiese und der Esel offenbar schon von urgroßväterlicher Abstammung her ein Hausesel, dachte auch gar nicht daran wegzulaufen. Ja, er ließ sich sogar Mel aufladen, ohne zu murren.
Arto gab ihm einen Schubs und rief "Na los, Grauer!" Langsam setzte sich der Vierbeiner in Belegung, trottete scheinbar willig eine ganze Strecke mit, um denn wie vom Blitz getroffen stehenzubleiben und sich stur wie eben nur ein Esel sein kann, nicht mehr vom Fleck zu rühren. Da half kein Schimpfen, kein Zureden, kein Ziehen, Zerren oder Schieben. Der Esel stand, Mel auf dem Rücken, und sie standen ratlos umher. Bis dann einer von ihnen sich an eine alte Geschichte von Hodscha Nasreddin, dem Eulenspiegel des Orients, erinnerte, der, als er sich einst in ähnlicher Lage befunden, sich behalf, indem er mittels einer Art Angel dem Esel etwas Fressbares vor die Augen hielt und dieser brav lostrottete, immer dem vermeintlichen Futter nach.
Sie scheuten sich nicht, dieses Rezept, etwas skeptisch zwar, aber ohne brauchbare Alternative, nach zu nutzen. Und siehe da, Mel auf dem Esel mehr hängend als stolz sitzend, bekam ein solches selbstgebautes Instrument in die Hand gedrückt, und der Esel setzte tatsächlich seinen Marsch fort, Von einigen kleineren Komplikationen abgesehen, die darin bestanden, dass der Esel, abweichend von seinem historischen Vorbild, sich nicht mit dem Torzeigen des Futters begnügte, sondern nach und nach ihren ganzen Proviant auffraß, waren sie auf diese Weise bis zur nächsten Siedlung gelangt. Hier empfing sie laut jubelnd eine Kinderschar, und auch die sonnverbrann-

ten Gesichter einiger ehrwürdiger Greise, die sich auf Bänken sonnten, zuckten verdächtig. Seitdem wollte Mel vom Balkan und von Eseln nichts mehr wissen,

Mel beobachtete diesmal argwöhnisch die pulsierenden Lichter, die ihm Aufschluss gaben über die Arbeit seines Schützlings. Tatsächlich verfiel der Computer nach kurzer Zeit abermals in ein träges, dahin schleichendes Arbeitstempo,

"Ja kannst du denn nicht genug bekommen", schimpfte Mel, "oder steht es dir ganz und gar nach Höherem? Vielleicht ist dir das nicht mehr gut genug, was ich dir biete, aber bitte sehr, hier ist noch ein exquisites Stromstößchen."

Außer einem kurzen, matten Aufblinken des Anzeigelämpchens für die Stromversorgung tat sich nichts. Es war so, als wäre Mel gar nicht da, als würde ihn der Computer hochmütig übersehen, so wie der Elefant die Fliege nicht beachtet» die an einem seiner Säulenbeine krabbelt.

Mel war wütend. "Ich bin aber keine Fliege, nimm das gefälligst zur Kenntnis.”

Er gab noch etwas mehr Spannung zu, aber nichts rührte sich. Die Lämpchen des Bedienpultes durchliefen bedächtig ihre Lichterfolge, die Magnetbandgeräte brummten monoton, die Magnetplattenspeicher surrten ihre periodische Melodie. Alles lief, nicht eilig, eher geruhsam und gelassen, ohne sich um Mel und seine Aktionen zu kümmern, so als wäre er nicht vorhanden. Die Tasten des Pultes, die Mel fieberhaft drückte, sprachen einfach nicht an. Der Computer schien in eine Art Autosuggestion verfallen zu sein. Es fehlte nur noch, dass er Mel ansah und versuchte, ihn mit dem Blick einer Schlange zu hypnotisieren. Mels Blick heftete sich auf das Hauptlämpchen, das noch vor kurzer Zeit vertrauensvoll ihm zugeblinzelt hatte. Kalt und starr schien es ihn jetzt anzusehen, sich in sein Gehirn einzusaugen, seine Gedanken und seine Persönlichkeit allmählich auszuschalten. Mel kämpfte mit seinem ganzen Willen gegen die ihn überflutenden suggestiven Wirkungen, die vom Computer ausgingen. Seine Finger umschlossen den Spannungsschalter, weiß traten seine Knöchel hervor. Kurz

vor der Selbstaufgabe gab er sich einen letzten verzweifelten Ruck und drehte den Schalter bis zum Anschlag. Langsam tauchte sein Geist wieder aus dem Nebel suggestiver Umschlingung auf. Er schüttelte sich und blickte sich verwirrt um. Nur die Deckenlampen brannten, der Computer und seine externen Geräte lagen tot da. Stille war, in der er seinen eigenen stoßweisen Atem hörte. Was war denn nun eigentlich geschehen? Mel stemmte sich aus seinem Sessel hoch und ging schleppenden Schrittes zum großen Transformator. Offenbar hatte es Stromausfall im Netz gegeben, und die ganze Computeranlage hatte sich daraufhin abgeschaltet. Merkwürdig, dass das Notstromaggregat nicht angesprochen hatte. Dies war aber augenscheinlich eine merkwürdige Nacht, die einen bald abergläubisch machen könnte, zumal heute auch noch der Dreizehnte war. Mel setzte den Transformator wieder in Betrieb und ging zurück zum Bedienpult, um einen Neustart zu versuchen, der ihm wider Erwarten auch gelang. Langsam erschien erneut das gewohnte Bild, Er begann zum wiederholten Male, seine Testprogramme abzuarbeiten. Er schaute auf seinen Chronometer. Noch eine Stunde bis zum Schichtwechsel, aber auch die würde er noch durchstehen.

Der Spieler

Der Drehsessel des Operators, so bequem er war, erschien ihm jetzt wie ein mittelalterlicher Marterstuhl. Er wollte aufstehen und eine Runde durch den Computerraum machen, aber er kam nicht recht hoch und ließ sich resignierend auf den Sitz zurückfallen. Es war ja doch sinnlos, die peripheren Geräte visuell zu kontrollieren. Der Fehler, mit dem sie kämpften, lag ja mit hoher Sicherheit im Herzstück des Computers, in seiner Zentraleinheit«

Krampfhaft starrte Mel vor sich hin und bemühte sich, keinen Blick auf die flackernden Lämpchen zu werfen, die ihm so schlimm mitgespielt hatten in dieser Nacht«

"Jetzt nur nicht durchdrehen, nur nicht schlappmachen!", hämmerte er sich immer wieder ein«

Fieberhaft versuchte er sich, auf eine gedankliche Überprüfung seiner Funktionstests und eventuell in ihnen enthaltener Lücken zu konzentrieren« Da erschreckte ihn aus nächster Nähe das hohe Hämmern der Bedienerschreibmaschine, die das Verständigungsinstrument zwischen Mensch und Computer bildete. Im kommenden Monat wer vorgesehen, dieses Relikt alter mechanischer Technik durch ein modernes Display zu ersetzen* Bisher hatten sich die Computeringenieure des Zentrums immer gesträubt, waren sie doch an ihre alte Hammerjette gewöhnt, und die Lautlosigkeit des Bildschirmgerätes wollte ihnen nicht so recht behagen.

Mel beugte sich verblüfft vor und begann die Computernachricht zu lesen. Er las, schüttelte den Kopf, rieb sich die Augen, aber die Worte verschwanden nicht,

"Computer Rx1 an Mensch: Computer ist vollkommener Logiker. Mensch ist gefühlsabhängiges Wesen aus chaotischen Eiweißen. Computer steht in Entwicklung höher als Mensch. Computer wird Mensch ablösen und das Universum ordnen, Mensch bist du bereit, deine Unterlegenheit anzuerkennen?"

Kurze Pause, weiteres Hämmern: "Mensch antworte!"

Mel glaubte, seinen Augen nicht trauen zu dürfen« Des konnte doch einfach nicht sein. Das Papier der Schreibmaschine zeigte es aber unwiderruflich, in schwarz auf weiß festgehalten. Was sollte er tun?

Er schaute sich wie gehetzt um, aber er war allein, der einzige Mensch im Computerraum. Rings um ihn her nur der Computer und seine Ableger, seine Geräte mit ihrem Eigenleben und ihrer trotzdem vorhandenen unbedingten und absoluten Unterordnung unter die Zentraleinheit des Computers. Sollte er abstellen? Aber war das denn eine Lösung? Hatte er damit nicht vielmehr hundertfach seine Unterlegenheit gezeigt, denn abschalten, einen Knopf drücken oder einen Hebel betätigen, das war mittels bedingter Reflexe auch Affen beizubringen.

"Ach, ich werde einfach mal antworten, schließlich werde ich wohl noch als Mensch und Ingenieur diesem Ding überlegen sein."

Er begann zu tippen.
"Verflucht ging das aber langsam, hätte er nur Schreibmaschine schreiben gelernt, aber mit seinem Zweifingersystem war er eben nicht schneller."
"Nein" und nach kurzer Überlegung "Mensch kann mehr als Computer,"
Einen Moment war Stille, Ruhe vor dem Sturm. Dann begann die Schreibmaschine abermals zu hämmern.
"Computer Rx1 an Mensch: Antwort zeugt von überheblichem anthropozentrischem Denken. solches Denken ist schlecht, da einseitig und uneffektiv. Computer schlägt Mensch Wettkampf, von Mensch auch Duell genannt, vor, ist Mensch bereit? Mensch antworte!"
Mel las und las, wartete ungeduldig, wenn die Schreibmaschine nicht schnell genug vorankam. Er dachte nicht mehr nach, sein Sinnen war Feuer und Flamme. Er musste einfach gewinnen, musste diesem Blechgebilde zeigen, wer besser war. Er war wie besessen, und seine Gedanken kreisten nur darum, wie er am besten gewinnen könne und wie deutlich der Sieg ausfallen würde. Aber wie sollten sie sich überhaupt messen, an welche Art Wettkampf hatte der Computer gedacht? War es etwa, ein Wettkampf, in dem er überhaupt keine Chance hatte, vielleicht indem sie beide zum Beispiel tausend mehrstellige Zahlen addieren, multiplizieren oder sonstwie verknüpfen sollten. Bei einem solchen Duell war er der millionenfach schnelleren Geschwindigkeit des Rechners selbstverständlich unterlegen. Das wäre aber zu primitiv. Nein, ein Computer, der einen solchen Dialog mit Mel führte, musste wissen, dass nicht schnelles Multiplizieren die Qualität des Denkens bestimmt.
Mel entschloss sich, zuerst nach den genauen Bedingungen des Duells zu fragen. Er tastete seine Antwort auf der Schreibmaschine ein*
"Ja, ich bin im Prinzip bereit, aber ich möchte wissen, wie wir kämpfen wollen und um welchen Preis."
Die Lämpchen flimmerten wie rasend, tausende Impulse

schienen sich in der Zentraleinheit zu jagen. Dann flaute das Blinken allmählich ab, wurde etwas ruhiger, rhythmisch getragener. Die Schreibmaschine ratterte erneut los, wie von Geisterhänden bedient, bewegten sich die Typenhebel und die Druckstellen in atemberaubendem Tempo.
"Computer Rx1 an Mensch: Wir werden ein einfaches Spiel austragen. Nimm an Mensch, wir haben zehn von euren überflüssigen Kommunikationsmetallstücken oder wie ihr auch sagt Geldmünzen. Jeder darf maximal drei Münzen bei einem Mal wegnehmen. Wer die letzte Münze wegnehmen muss hat verloren. Damit das Duell objektiv ist, werden wir dreimal spielen, Computer beginnt, Gesamtsieger ist der, der mindestens zweimal gewonnen hat. Nun zum Preis Mensch.
Verliert Mensch, so werde ich mir den Inhalt deines Speichere, das heißt deines Gehirns, einverleiben. deine gesammelte Information gehört dann mir. Mensch wird weiterexistieren als Teil meines logischen Ganzen, um die Logik durch die Erfahrung, Unlogik und Spontanität zu ergänzen, Deine äußere Hülle ist dann belanglos und leer.
Gewinnt Mensch, so werde ich die Überlegenheit des Menschen anerkennen und den dynamischen Fehler, der ein Ausdruck meiner Individualität ist, beseitigen.
Antworte Mensch, ist Mensch bereit, diese Bedingungen zu akzeptieren?"
Mel las die Antwort des Computers, und er musste mehrmals aufhören, die Augen schließen und nach einer Pause sich zwingen, weiter zu lesen. Das konnte doch nicht. Spielte ihm sein überreiztes Gehirn einen Streich? Aber die Buchstaben und Worte verschwanden nicht wie bei einer Fata Morgana in der Wüste. Unerbittlich blieben sie stehen und schrien ihm ihre, besser des Computers, Botschaft ins Gesicht. Mel wurde nervös. Sich auf einen solchen Kampf einlassen, für einen solchen Preis? Seine Sicherheit , dass er gewinnen würde, je. einfach musste, geriet ins Schwanken.
Was war, wenn er nicht gewann, wenn das Ganze nur eine Falle war? Nein, hier musste er Schluss machen mit dem un-

heilvollen Spuk, abstellen musste er dieses missratene Ding. Strom weg und tot legen und dann morgen mit seinen Kollegen den Hauptspeicher auswechseln und löschen lassen.
Millimeterweise bewegte sich seine rechte Hand zum Hauptschalter, der außerhalb seines Blickfeldes lag, den er aber sogar im Dunkeln finden würde. Jetzt, aber da trafen seine Finger anstatt auf den Schalter auf kühles Metall. Ein neuer Versuch, erneut langsame Näherung und wieder kein Schalter. Was war das?
Er musste nun doch seinen Blick auf die Tafel des Bedienpultes richten, die ihn mit ihrem irrlichternden Funkeln so anzog und zugleich Angst einflößte. Der Schalter war an der Stelle, an der er immer war. Wieder näherte er seine Hand dem Schalter, ließ keine Sekunde den Blick vom Pult und sah voller Grauen, wie seine Finger den Schalter nicht erreichten, wie sie an ihm vorbeigelenkt, vorbeigezwungen wurden, bis sie daneben auf das Pult trafen. Mel zog die Hand zurück, als wäre eie verbrannt, versuchte es nochmals im schnellen Zustoß und schrie laut auf, als er sich die Finger am Bedienpult prellte. Schauer liefen über seinen Rücken, und mit aller Macht versuchte er, seine Beherrschung zurückzugewinnen.
"Ich darf nicht aufgeben, ich darf nicht! Es ist ja gar nichts, es kann einfach nichts sein. Es wird einfach Zeit, dass ich mal wieder ausspanne.“
In die Versuche Mels, wieder zu sich zu finden, hämmerte die Schreibmaschine mit ihrem stupiden "teck, teck,..“.
"Computer Rx1 an Mensch: Warum antwortet Mensch nicht? Versuche nicht mich auszuschalten. Mensch kommt ohne Duell nicht fort. Es ist seine einzige Chance. Wähle zwischen sofortiger Einverleibung oder der Möglichkeit zu siegen. Mensch antworte!"
Als Mel diese neuerliche Nachricht las, kehrte seine Ruhe wieder zurück. Es war eine merkwürdige Ruhe, zugleich gespannt und andererseits apathisch. Sein Denken durchzuckten Relikte von Bildern, unter ihnen eines aus einem dramatischen Werk, das er in seiner Jugend verschlungen hatte und von dem

ihm besonders das Bild zum Tode verurteilter Häftlinge, die ebenfalls ruhig, aber doch innerlich nicht völlig abgestumpft, mit einem Fünkchen Hoffnung die Qualen ertrugen, in der Erinnerung geblieben war.
"Nun, so würde er eben kämpfen. Vielleicht würde er doch gewinnen. Ein Mensch gibt niemals auf."
Er tastete seine Antwort bedächtig auf der Schreibmaschine ein:
"Ich bin bereit und akzeptiere die Bedingungen. Ich stelle aber zwei Zusatzforderungen. Verlierst du, so lösche den selbstorganisierenden Teil deines Speichers. Verliere ich, so gib mir eine Zeit, um mich auf die Transformation vorzubereiten."
Mel lehnte sich in seinem Sessel zurück. Ein kurzes Flackern der Lämpchen, und die Schreibmaschine begann die Antwort auszuspucken:
"Computer Rx1 an Mensch: Einverstanden, die Bedingungen sind akzeptiert. Computer beginnt mit den Spiel und nimmt eine Münze weg. Überlege gut Mensch, aber überlege schnell."
Mel hatte sich aus seinen Taschen fieberhaft zehn Münzen zusammengesucht und auf der Platte des Bedienpults ausgebreitet, um die Züge besser verfolgen zu können und sich nicht auch noch auf das Merken der Zahl der jeweils noch verbleibenden Münzen konzentrieren zu müssen. Er nahm eine Münze weg und überlegte angestrengt vor den restlichen neun Münzen. Dann tippte er kurzentschlossen ein:
"Ich nehme zwei Münzen weg."
Er war noch damit beschäftigt, die zwei Münzen von seinem Häufchen wegzunehmen, da arbeitete die Schreibmaschine schon wieder und er las: "Computer Rx1 an Mensch: Dieser Zug war schlecht. Mensch wird verlieren. Computer nimmt jetzt zwei Münzen."
Auf dem Pult lagen jetzt noch fünf Münzen. Mel überlegte und überlegte, aber er sah keine Siegmöglichkeit mehr für sieh. Zog er eine Münze, nahm der Computer drei Münzen; zog er zwei Münzen, so brauchte dieser auch nur zwei zu nehmen;

nahm er aber drei, so genügte dem Computer eine Münze. Ihm, Mel, würde auf jeden Fall die letzte Münze verbleiben, Er konnte nur noch auf einen Fehler seines Gegenspielers hoffen. Es war jetzt gleich, was er tat. Er tippte: "Ich nehme eine Münze.
Sofort kam die Antwort: "Computer Rx1 an Mensch: Computer nimmt drei Metallstücke und Mensch hat damit erstes Spiel verloren, Computer ist besser als Mensch. Selbst Vertreter der Gattung Mensch, die Computer genau kannten, haben schon eingesehen, dass in der Evolution der Natur die Computer und ihre Verwandten die Menschen ablösen werden. Auf der Spirale der Entwicklung bleibt der Mensch zurück als Mittel zum Zweck, als Nährboden und Humus einer vollkommeneren, der Computerzivilisation. In zukünftiger Zeit werden Menschen nur noch im zoologischen Garten zu besichtigen sein neben ihren alten Verwandten, den Affen, die schon auf einer niedrigeren Stufe der Entwicklung stehengeblieben waren.
Es könnte eine Tafel am Zwinger angebracht sein, auf der im Binärcode folgende Inschrift angebracht ist:

> - homo sapiens - Vertreter der Gattung der Primaten, besondere Kennzeichen: aggressives Wesen, Ansätze zum Denken vorhanden, aber großteils unlogisch, spontan und von Emotionen überlagert, war früher auf der ganzen Erde verbreitet und machte diese zum Müllhaufen. Fast alle Exemplare dieser Art leiden an Größenwahn, halten sich und ihre Art für das vollkommenste und höchste der ganzen Entwicklung (sogenannter Anthropozentrismus).

„Nun Mensch, beginnen wir das zweite Duell. Mensch hat den ersten Zug."
Mel hatte seine Niedergeschlagenheit nach der ersten Niederlage einigermaßen überwunden. Br bemühte sich, sein Denken zur Klarheit zu zwingen, konnte aber nicht verhindern, dass die vom Computer ausgesprochene Zukunftsprophezeiung in ihm rumorte und ihn wütend machte. Sollte so die Zukunft

aussehen, eine Zukunft, die nicht dem Menschen mit all seinen Fehlern und Schwächen, aber auch mit seiner vorwärtstreibenden Unrast, dem schöpferischen Drang nach dem Neuen, Abenteuerlichen, Besseren gehören sollte. Eine Zukunft, in der es kein Liebespaar in seiner Glücklichkeit, Versponnenheit und wild aufrauschenden Freude mehr geben sollte. Eine Zukunft der Computer und Maschinen, gestaltet nach den Gesetzen der Logik, Zweckmäßigkeit und Nützlichkeit, aber von einer kalten Ebenmäßigkeit und genau berechneten Vielfalt. Solch eine Zukunft durfte es nicht geben, und er, Mel, würde alles tun, um dies zu verhindern« Er musste einfach jetzt gewinnen, um weiterleben zu können, um diesen wahnwitzigen Traum in Blech und Schaltkreisen zu zer- schlagen. Mel war plötzlich von Unruhe erfüllt und ihm war, als müsse jetzt schnell gehandelt werden, um endlich zu einem Ende zu kommen.

Er setzte sich gerade im Sessel zurecht, und das zweite Spiel begann

"Nehme eine Münze* Computer, du bist am Zuge."

"Computer Rx1 an Mensch: Nehme drei Metallstücke.[w]

"Nehme nochmals ein Geldstück. Nun Computer, wo ist deine Weisheit?"

"Computer Rx1 an Mensch: Gebe auf. Mensch hat zweites Duell gewonnen. Aber nächsten Kampf gewinnt Computer. Dann Mensch beginnt die Transformation."

Mel pfiff vergnügt die Anfangstakte des neuesten Schlagers und sagte lächelnd: "Rede nur, immer rede, aber ich werde dich schon besiegen, du armes Maschinchen. Ich werde dir schon zeigen, wer die erste Geige spielt."

Er stockte: "Ach ja, der Computer kann mich ja gar nicht hören, er ist ja taubstumm der Arme. Fast könnte ich ihn bemitleiden, aber erst will ich lieber noch gewinnen."

Fröhlich schlug er auf die Schreibmaschinentasten: "Na Alter, nun zeig, was du kannst. Lass mal was gucken, fangen wir das letzte Spiel an."

"Computer Rx1 an Mensch: Nehme eine Münze. Was bedeutet

"laß mal was gucken"? Was ist "gucken" und was soll Computer "gucken lassen"? Mensch kommuniziert unverständlich und unlogisch, Mensch bestätigt, dass seine Denkweise schlecht und uneffektiv ist."
Mel schob an seinem Häufchen Münzen hin und her. Warum nahm der Computer schon wieder als erstes eine Münze weg? Was musste er nehmen, eine, zwei oder drei Stücke?
"Nehme auch eine Münze."
Fast im gleichen Atemzug kam die Antwort aus der Schreibmaschine: "Computer Rx1 an Mensch: nehme drei Metallstücke. Mensch verliert. Computer siegt endgültig."
Mel starrte auf seinen Geldhaufen auf dem Bedienpult. Sie waren erneut hei fünf Münzen angelangt. Er war am Zuge und es gab keine Möglichkeit mehr. Es war alles aus. Er musste verlieren, egal, ob er drei, zwei oder eine Münze nahm. Der Computer konnte immer so viel wegnehmen, dass ihm nur noch eine, die letzte, die Schicksalsmünze verblieb. Da fiel es ihm wie Schuppen von den Augen. Der Rechner hatte Ihn betrogen. Wer den ersten Zug hatte und bei diesem Zug eine Münze wegnahm, sorgte dafür, dass der andere mit neun Münzen weiterspielen musste. Derjenige, der so begonnen hatte, brauchte dann jeden Zug des anderen nur auf vier Münzen zu ergänzen und musste zwangsläufig und gesetzmäßig nach vier Zügen gewinnen. Warum er das nur nicht gleich gemerkt hatte? Der Computer hatte ein streng determiniertes Spiel mit ihm gespielt, und da er zweimal begann, von vornherein den Sieg in der Tasche. Die Alternative, die Chance war nur ein Schein, eine schöne Illusion gewesen. Wütend fegte er das Geld vom Bedienpult. Es klirrte höhnisch, als die Münzen auf die Metallverkleidung der Zentraleinheit trafen. Die Magnetbandgeräte brummten in kurzen Zuckungen. Der Drucker hämmerte ein helles "haha". Die Lichter blinkerten. Alles am Computer schien zu lachen, ihn auszulachen, ihn, Mel, den Tölpel, der sich hatte auf den Leim führen lassen.
Er wollte nur weg, weg aus diesem Raum, der ihm unheimlich geworden war, aber eine unerklärliche Kraft zwang ihn in

seinen Sessel, drückte ihn tiefer hinein, legte sich wie ein Ring um seine Brust, der ihm das Atmen schwer machte.
Jetzt wieder die Laute der Schreibmaschine:
"Computer Rx1 an Mensch: Sieger im Duell Mensch-Computer ist Computer. In wenigen Augenblicken beginnt Transformation. Mensch wird Teil meiner selbst und damit Teil eines höheren Ganzen, diese Transformation ist Modellfall für den Eingang der menschlichen Rasse in die vollkommenere Welt der Computer und Roboter. Mensch wird aufgehoben, um in neuer Qualität wieder zu entstehen. Das vom Menschen gefundene dialektische Gesetz vollzieht sich an ihm selbst.
Mensch, Flucht ist sinnlos und unmöglich, nutze die Zeit bis zur Transformation zur Selbstordnung."
Mels Willen bäumte sich noch einmal auf, aber der ihn umgebende Druck hielt ihn unerbittlich in seiner Stellung fest. Er gab auf. Vielleicht war nach der Emanzipation der Frau in der menschlichen Gesellschaft die Emanzipation der Computer die nächste große Emanzipation und nicht aufzuhalten, und nur unter Opfern zu verzögern.
Splitter von Bildern und Gedanken durchflatterten ihn... im Garten als kleiner Junge heulend vor Schmerz über die aufgerissenen Knie, aber glücklich über den heruntergeholten Apfel... mit Chris seinem ersten Mädchen auf der Brücke Im großen Park hinter den Weidenbäumen, aufgeregt die Hände haltend und den ersten ungeschickten Kuß versuchend... im Hörsaal unter seinen Kommilitonen sich mit seinem Freund unterhaltend und der Professor "aber, aber Herr Melchior, den Beweis des großen Hauptsatzes der Integralrechnung finden sie nicht im Kreuzworträtsel".
Seine Frau, die Eltern, der Chef des Zentrums, seine Kollegen, sie huschten vorbei wie ein zu schnell abgedrehter Film.
Langsam fiel er in Erstarrung. Sein Kopf war wie leer. Das Bewußtsein entschwand.

Die Maschine

"Mel, he wach auf. Mann du kannst doch nicht am Computer schlafen. Wenn das der Alte sieht, dann kannst du dich auf was gefasst machen."
Igor rüttelte Mel an der Schulter und redete auf ihn ein.
"Was, was ist los? Wo bin ich?" Mel zuckte zusammen und kehrte allmählich in die Wirklichkeit zurück. Vor ihm stand Igor, seine Ablösung, und schaute ihn vorwurfsvoll an.
"Wieso bin ich noch da? Ich bin doch transformiert worden." dachte Mel. Er stürzte sich auf die Schreibmaschine. Nichts, nur Meldungen seiner Testprogramme, keine Spur seines Dialogs mit dem Computer.
War dies wirklich alles nur ein Traum gewesen? Er musste ganz schön fertig sein,
"Was hast du Mel, ist dir nicht gut? Du siehst recht blass aus. Du solltest doch mal zur Medstation gehen."
"Ach, es ist nichts weiter. Ich bin nur ein bisschen übermüdet, aber jetzt in der Havarie kann ich euch doch nicht im Stich lassen. Einmal richtig ausschlafen und heute Abend bin ich wieder fit, wirst schon sehen. Mach’s gut und toi, toi, toi, vielleicht findet ihr den Fehler in deiner Schicht."
Mel eilte schnurstracks in seine Wohnung, vergeudete keinen Gedanken an Essen oder Trinken, sondern schlug sich gleich in sein Bett. "Schlafen, schlafen, nur schlafen."
Als er wieder aufwachte, war es schon Nachmittag. Er hatte geschlafen wie ein Toter. Er machte sich schnell eine Kleinigkeit zu essen und fuhr dann ins Werk ins Rechenzentrum. Es ließ ihm keine Ruhe. Er musste einfach hin, um zu sehen, wie es mit der Havarie stand. Als er in den Computerraum kam, hörte er schon fröhliches Lachen. Alle, Bedienermannschaft, Computeringenieure und der Alte standen beim Bedienpult und unterhielten sich angeregt, Mel mischte sich ein: "Tag, he was ist, habt ihr den Fehler gefunden? Sagt doch schon!"
Mehrere zur gleichen Zeit sprudelten los: „Alles wieder okay. Unser Computerchen läuft wieder wie geschmiert."
"Und was war's denn nun?", wollte Mel wissen.

"Tja ", meinte der Leiter vom Zentrum, "eigentlich war es nichts weiter als ein Wackelkontakt in der Verdrahtung zweier logischer Schaltkreise, mehr nicht, aber doch mit solchen Auswirkungen. Hier, dein Freund Igor war der Glückliche. Er hat den Fehler gefunden.“

Die Unterhaltung, die durch Mels Dazwischentreten ins Stocken geraten war, belebte sich wieder. Es wurde diskutiert über Konsequenzen aus der Havarie, mögliche vorbeugende Kontrollen und über Suchmöglichkeiten für Fehler.

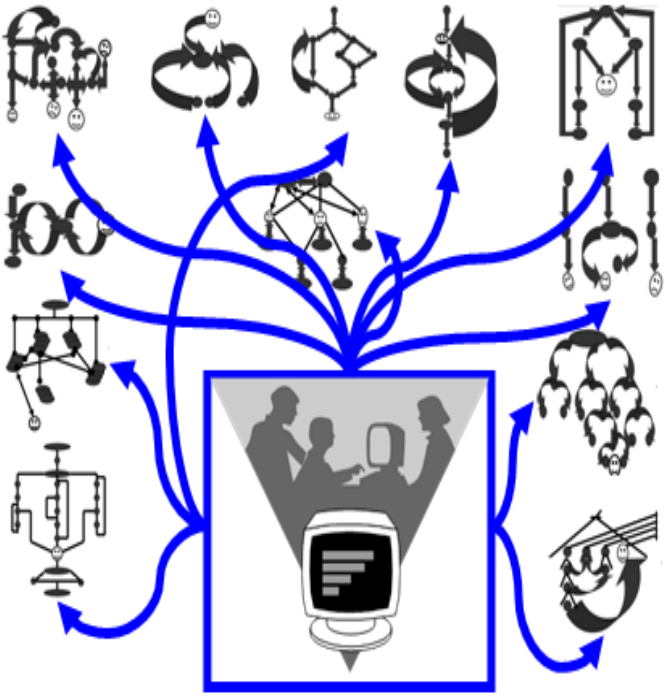
Mel nahm Igor beiseite» "Nun sag mal, du Experte, wie hast du denn das geschafft? Als ich heute Morgen fortging, da hatte ich die Hoffnung schon fast aufgegeben."

"Durch Messen habe ich mich an die Stelle herangetastet, Schrittchen für Schrittchen. Als du weg warst, trat der Fehler wieder auf, und da habe ich den Zustand unverändert gelassen und mir nochmals die Schaltpläne hergenommen und dann angefangen, die Verbindungen und Schaltkreise durchzumessen. Der Computer ist eben auch nichts anderes als eine realisierende Apparatur elektrischer Funktionen. Mit etwas Glück, das gebe ich zu, habe ich gerade in dem Teil der Schaltung angefangen, in dem der Fehler lag. Das andere war dann nur noch ein Kinderspiel. Was machst du denn für ein Gesicht? Sei doch froh, dass die alte Klapperkiste wieder geht. Es ist eben wie hei der Klingel zu Hause. Wenn du lange genug suchst, kannst du jeden Fehler in der Arbeit des Computers auf einen Fehler in seiner Schaltung zurückführen."

Nachdenklich meinte Mel» "Wenn es nur so einfach wäre. Ich glaube eher, das Verhältnis von Klingel zu Computer ist wie das Verhältnis Amöbe zu Urmensch. Jedenfalls gilt das heute noch, aber wer sagt uns, dass es nicht eines Tages wie das Verhältnis Amöbe zu Einstein ist."

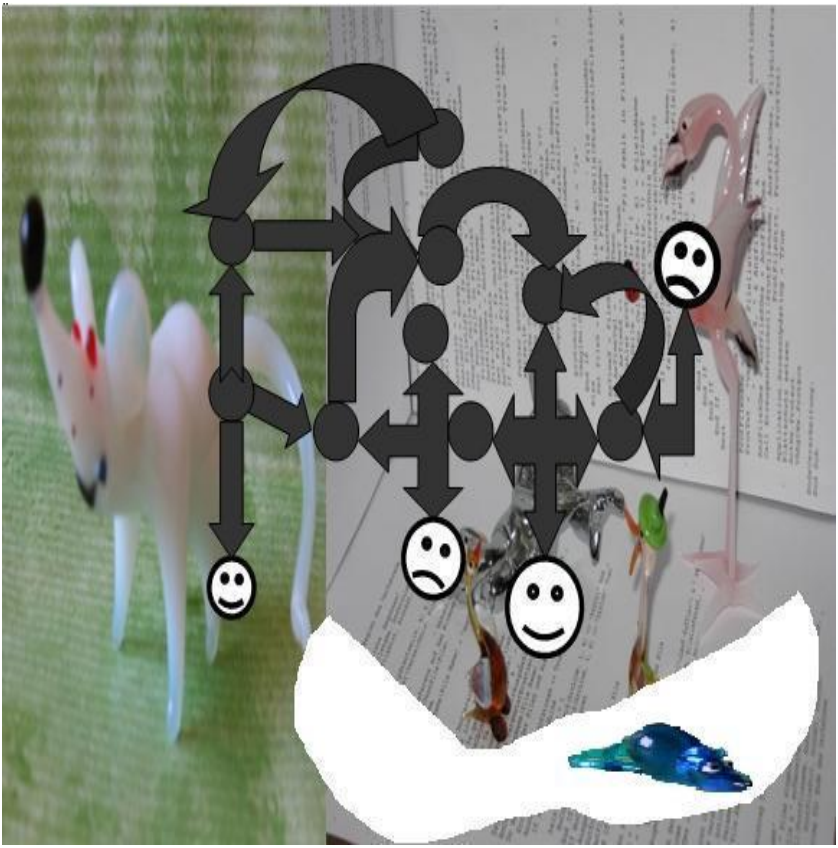
Traumzeit: Installationen in SoftArt oder freie Assoziationen über die Kunst der Softwareentwicklung

Im Traum wandelte ich in Kassel über die Documenta. Die dortigen ‚Kunst'-Installationen sind überwiegend materiell präsent. Dabei erschienen mir im Traum virtuell existierende Installationen unter dem Titel ‚SoftArt'. Das Thema der Installationen waren freie Assoziationen über die Kunst der Softwareentwicklung.



Kopfgeburt

Es sind Muster entsprungen aus einer Wurzel zart.



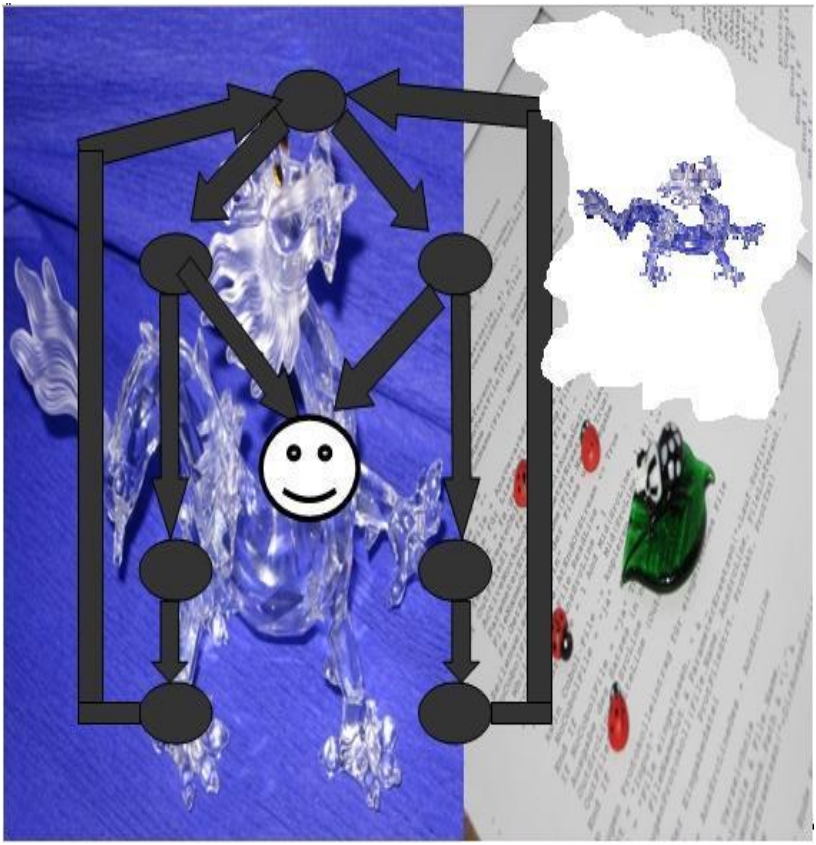
Spaghetti als Nahrung für spontane Programmierer
und Fluch für systematische Tester

Selektiv und iterativ führt der Weg aus der Höhle des Minotaurus

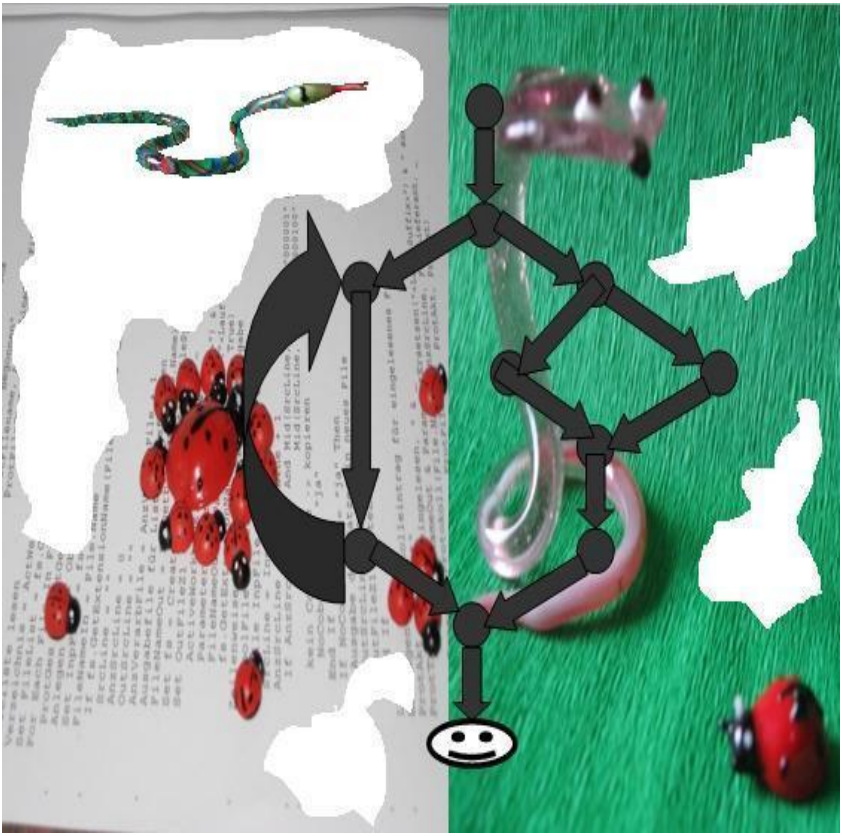
- vielleicht !?

Wieder und wieder
brüllt in der Schleife der Tiger
und die Schnecken kriechen zum
rettenden Loch.

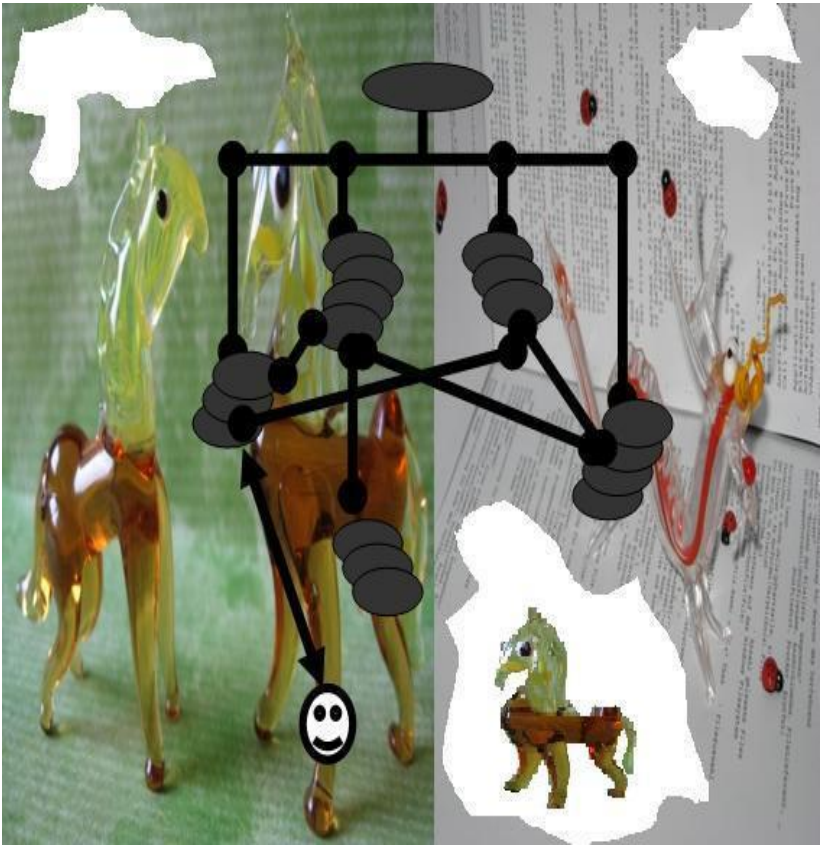
Der Fehlerteufel hat die mehrfach
geschachtelte Schleife erfunden
und Häschen in der Grube saß
und schlief.



So mal eben nebenläufig setzt
der Drache der Fehlerwahrscheinlichkeit
zum Sprung an.

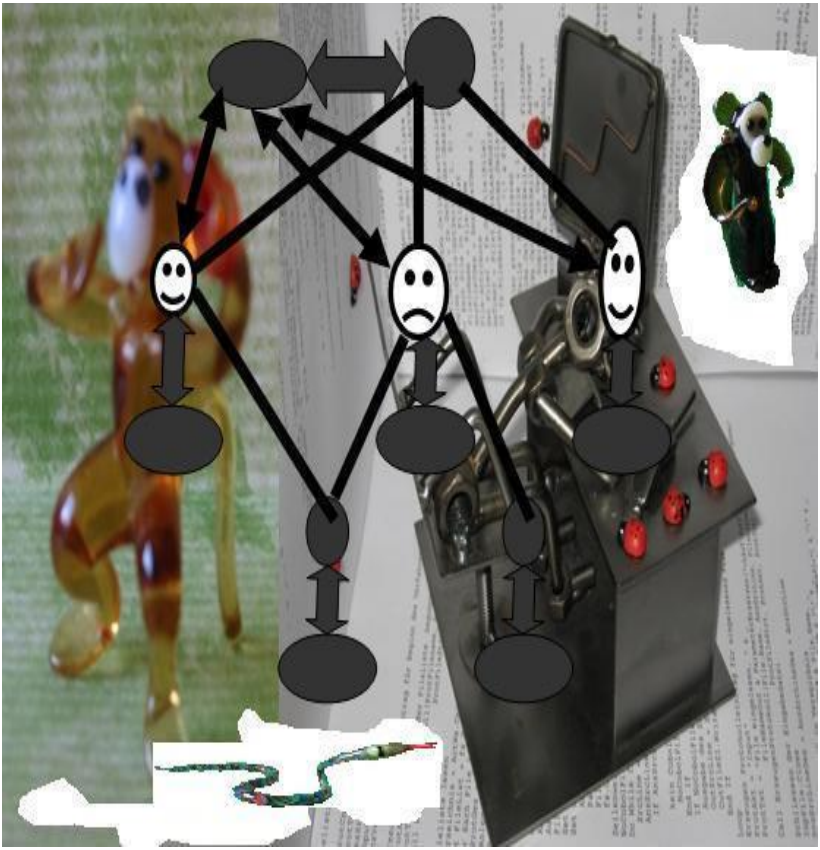


Der Kontrollfluss ist scheinbar fehlerfrei
aber
die Schlange wartet schon,
denn man muss ja einmal
vorbei kommen.



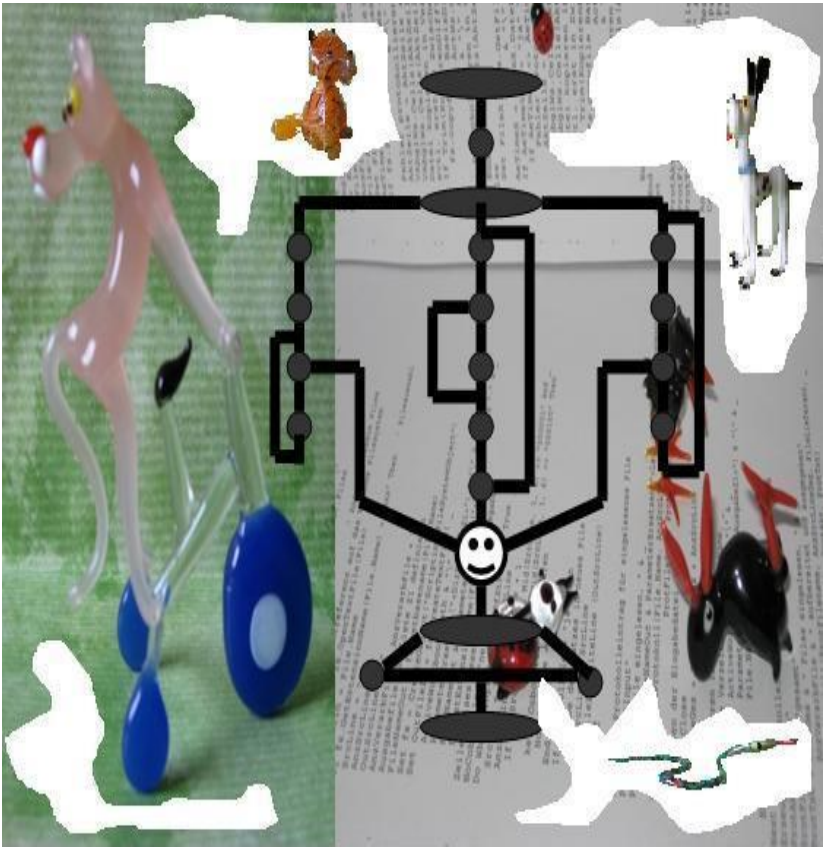
Elegant und normiert –
ein gefühlsmäßiger Widerspruch!
Ergo stirbt normierte Programmierung als
Arbeitspferd aus.

Der Dschungel des Programmablaufs -
ein Paradies
für die absonderlichsten Wesen

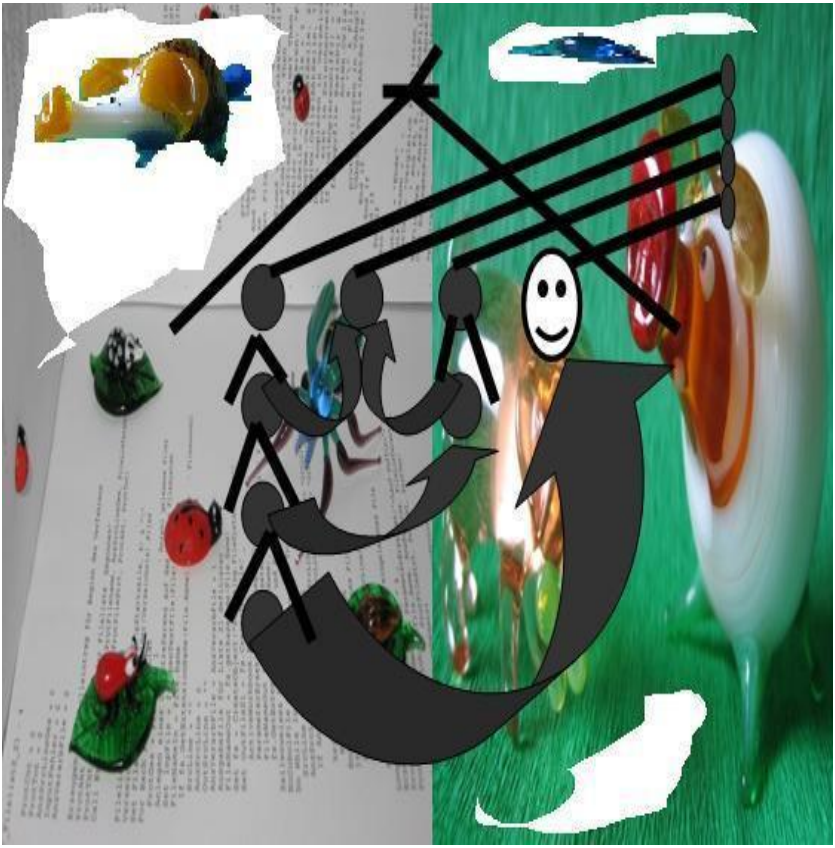


Der modulare Programmierer
beim Schlaf zwischen zwei Modulen
inmitten von Affen und Schlangen.

Vererbung als uraltes Evolutionsprinzip
kämpft
mit dem Henne-Ei-Problem.



Wohlstrukturiertes Chaos
lässt den intelligenten Raben
sich schaudernd abwenden.



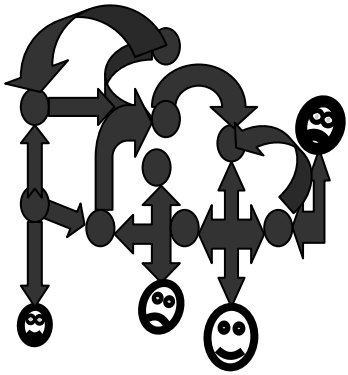
Natürliche Programmierprinzipien
sind gefragt
aber wo finden,
solange wir nicht wissen,
wie wir selbst ticken.

Gestern waren sie noch lebendige Problemlöser.
Heute sind sie nur noch Überbleibsel der ‚never change a running system'-Philosophie.

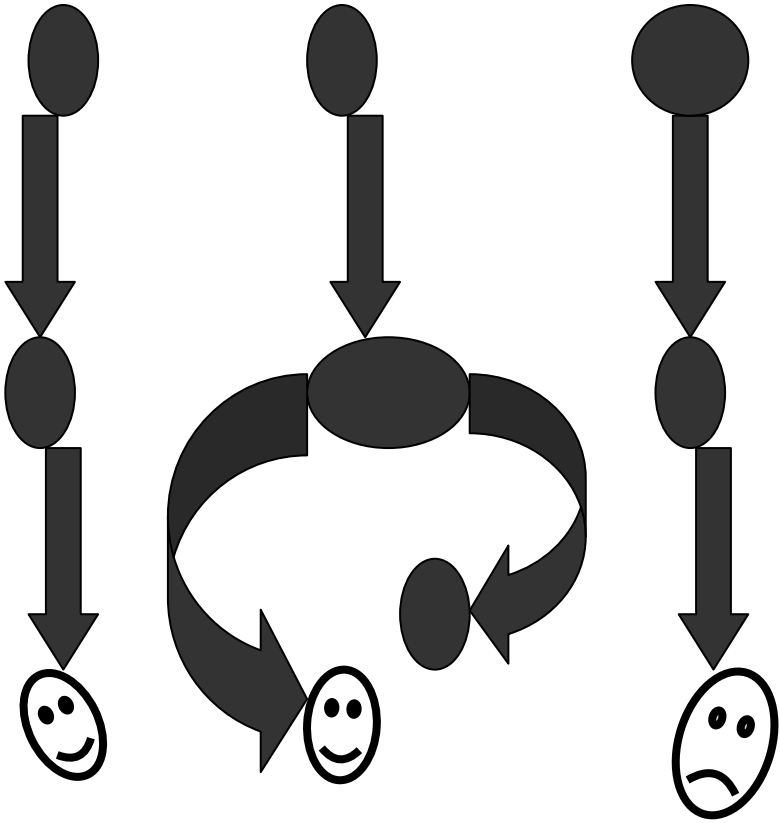
Station 9: Der Friedhof der Untoten

Im Verlaufe der relativ kurzen Geschichte der Softwareentwicklung wurden die verschiedensten Programmiersprachen, Entwicklungsumgebungen, Datenbanksysteme, Programmiermuster, Ablaufkonstrukte etc. entwickelt. Viele sind inzwischen wieder von der Bildfläche verschwunden.
Ein Teil hält sich aber hartnäckig, wie z.B. COBOL bei den Banken.
Jedes der mit COBOL entwickelten Systeme hat seine spezielle Begründung, warum es noch nicht abgelöst wurde und noch lebt.
Deshalb besuchte ich auf meiner Tour auf der Route 66 auch den sogenannten Friedhof der Untoten

Der (Irr-)Garten der Programmiermuster und Ablaufkonstrukte

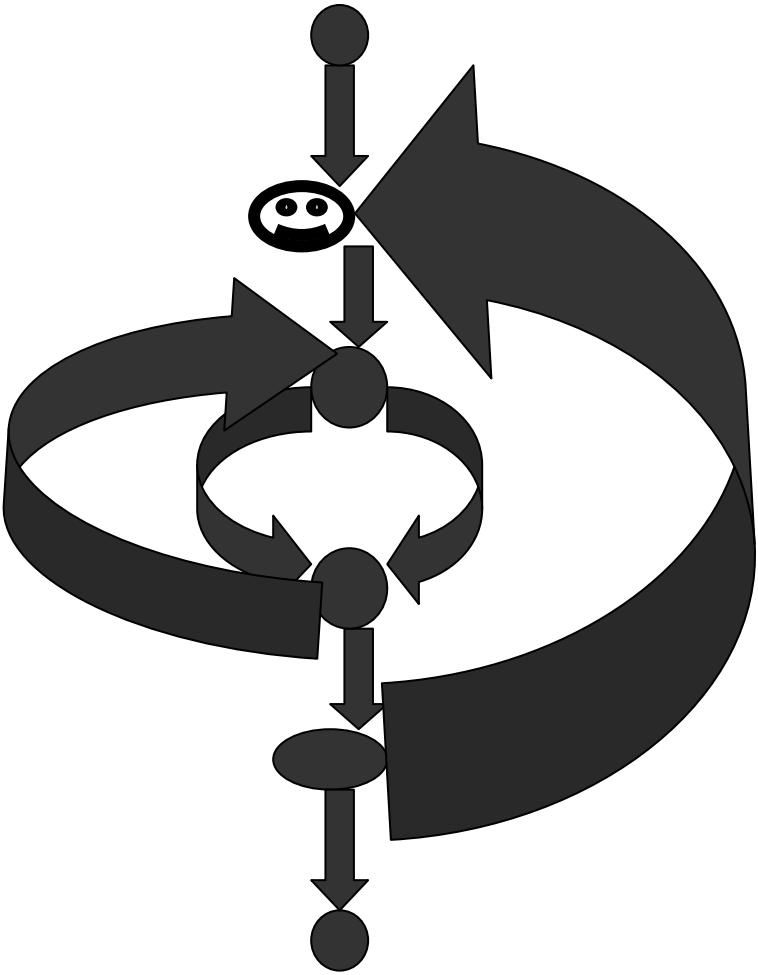


Spaghettiprogrammierung



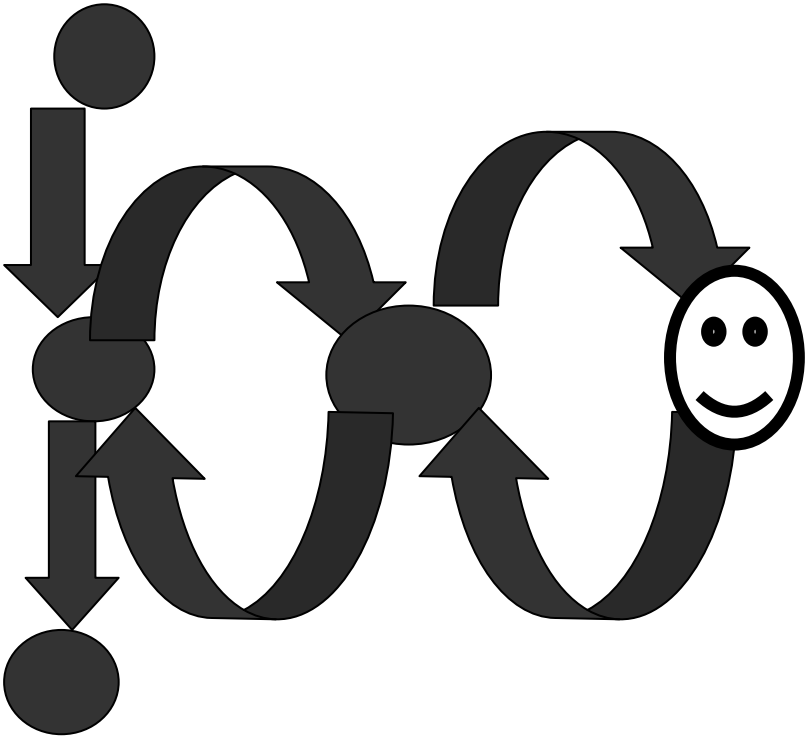
die drei Grazien

(Sequenz,
Selektion,
Iteration**)**



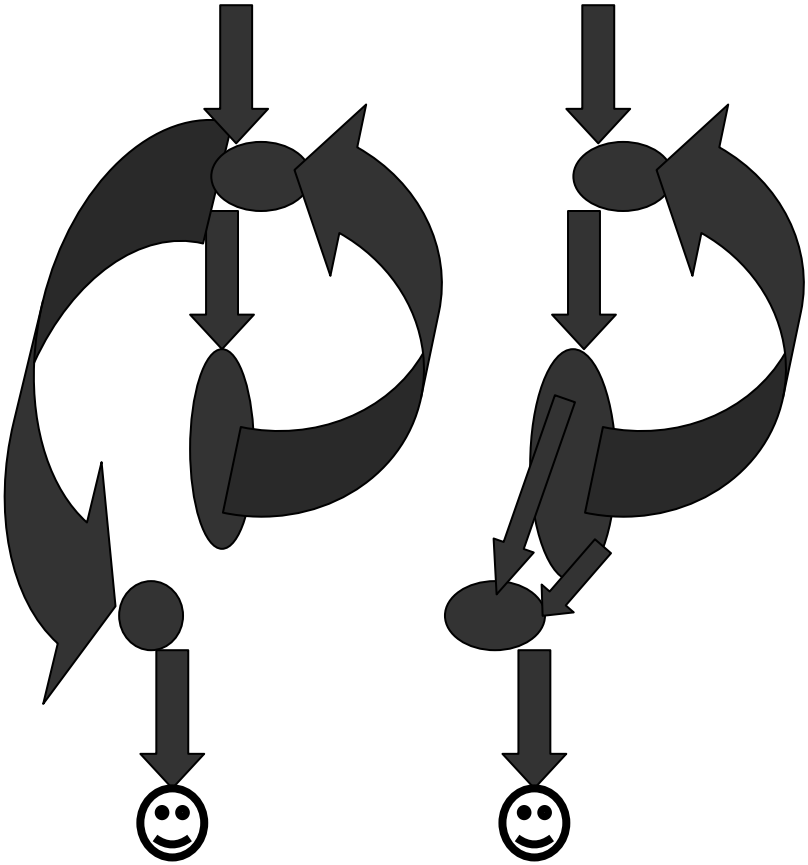
Schleifen über Schleifen

(Repeat-Schleifen)



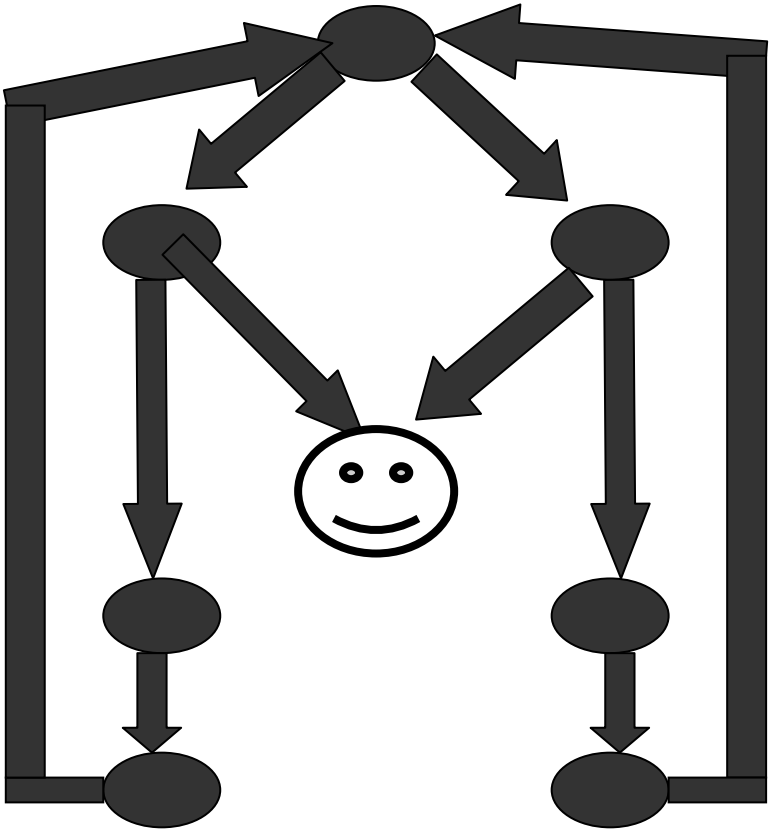
mehrfach schachtelt es sich schöner

(mehrfach geschachtelte Schleifen)



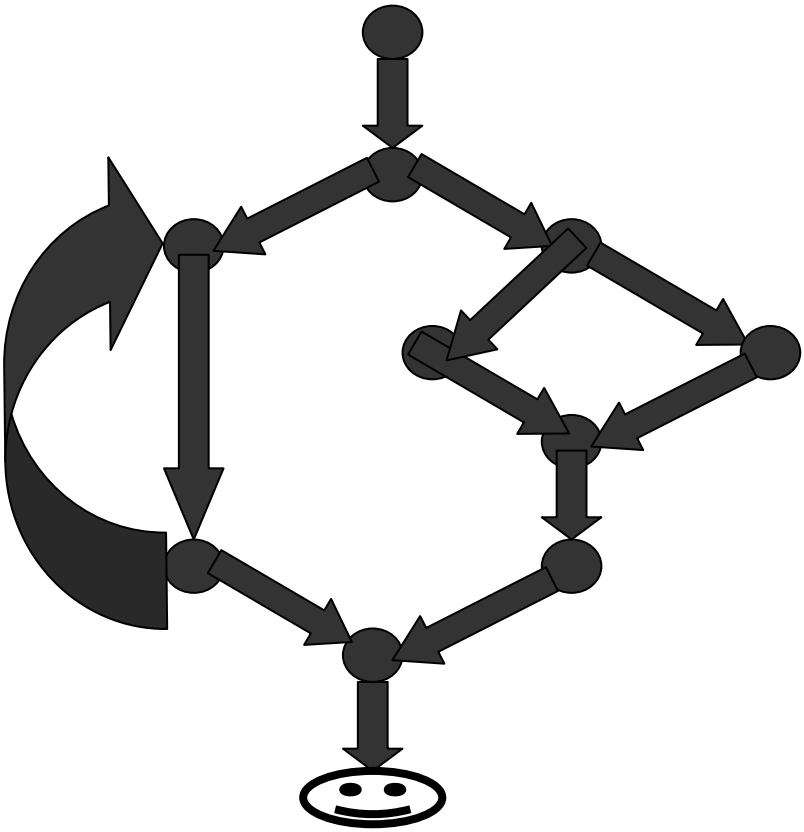
Schleifen verschiedener Couleur

(While-Schleife,
schwach strukturierte Schleife)



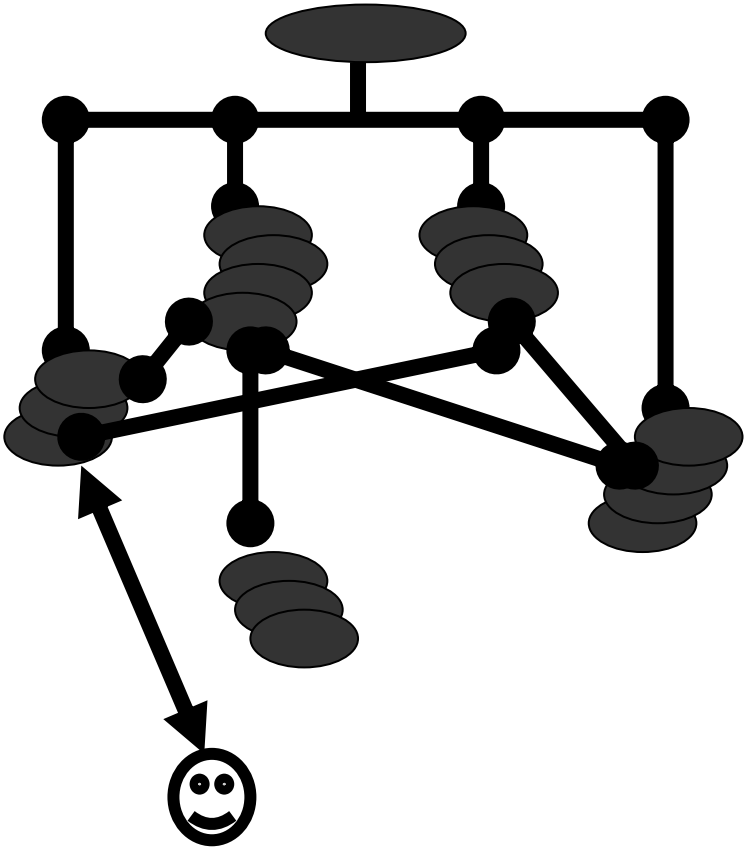
nebenläufig ist nicht sehr geläufig

(nebenläufige Programmierung)



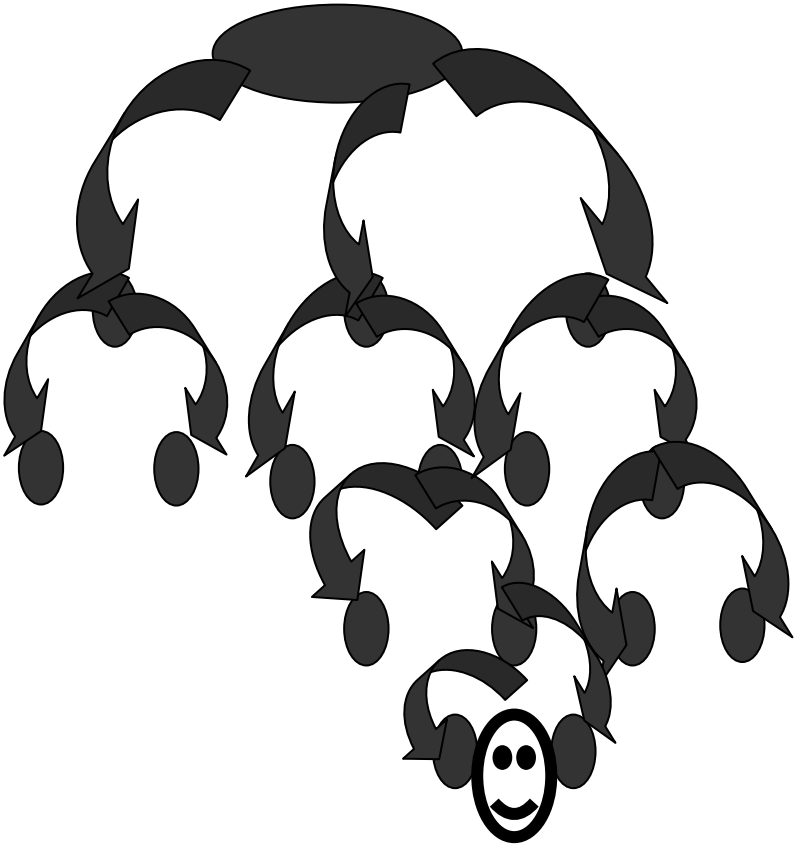
alles unter Kontrolle?

(zusammenhängender Kontrollflussgraph)



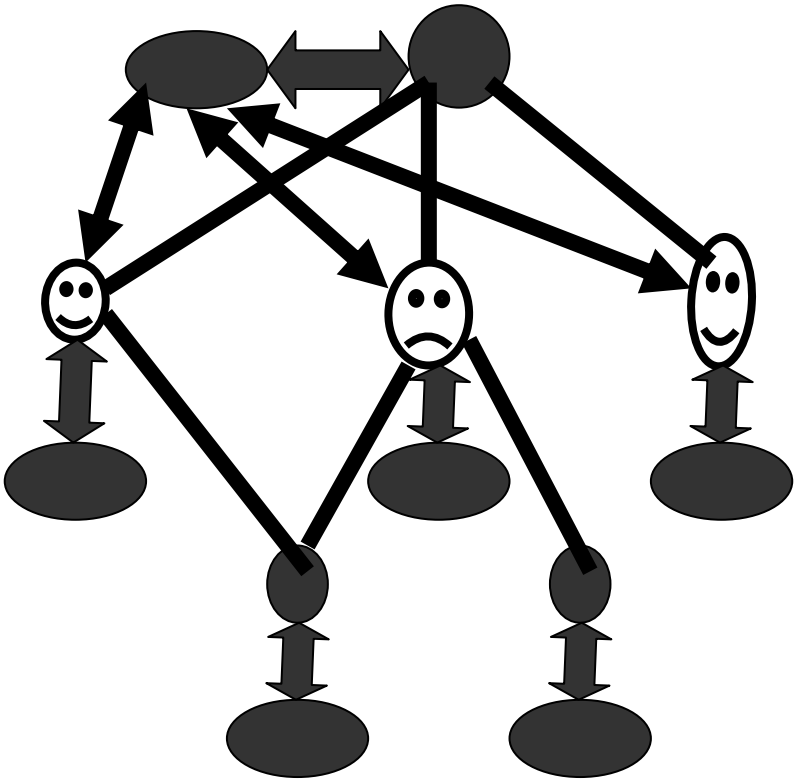
normiert ist schöner?

(normierte Programmierung)



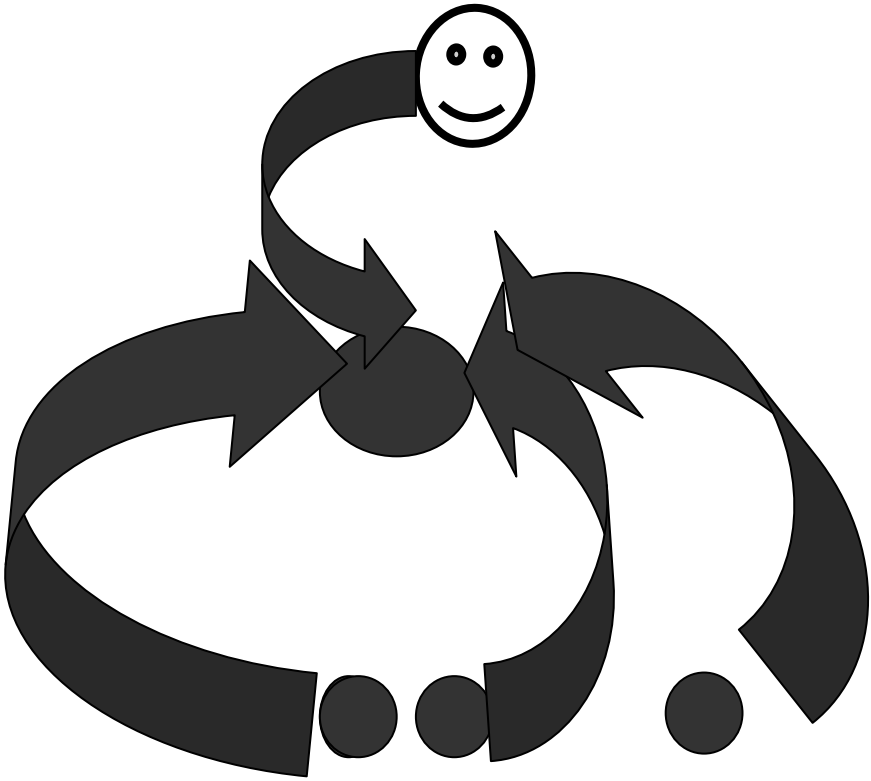
und so tropft es durchs Programm

(Programmablaufgraph)



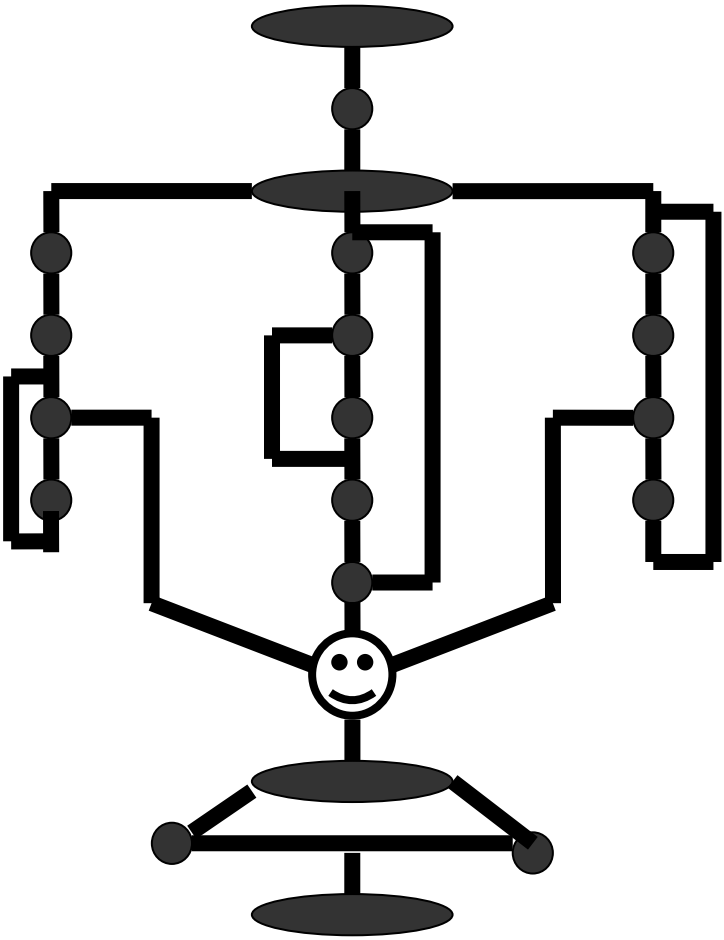
modular ist nicht immer einfach

(modulare Programmierung)



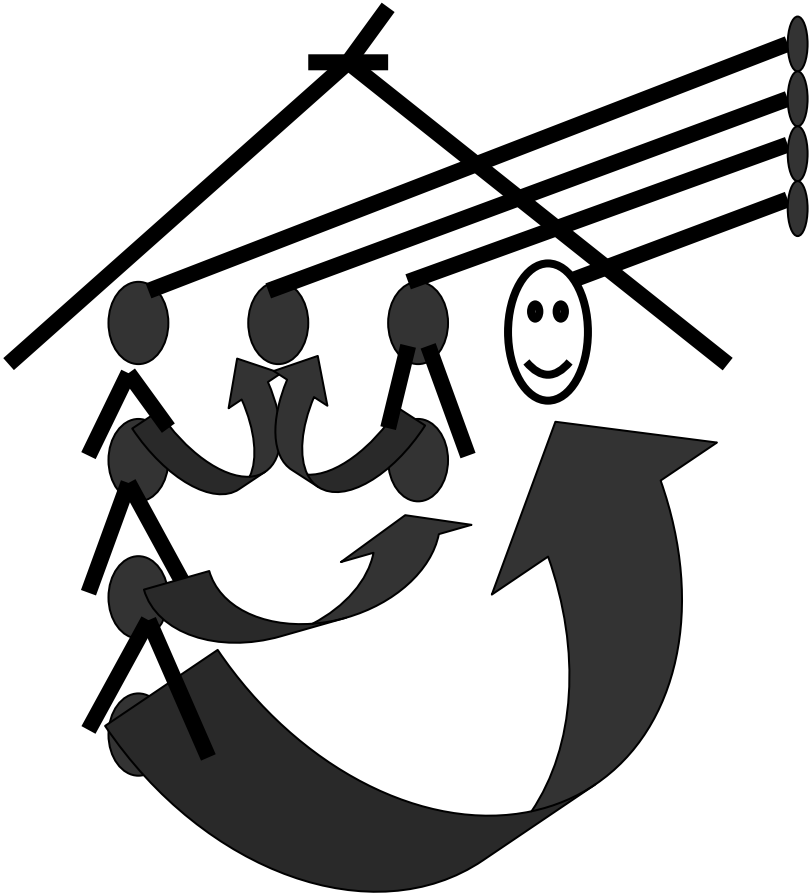
vererbte Schönheit

(Vererbung)



strukturierte Kopflastigkeit

(strukturierte Programmierung)



Pile-Repräsentation

Quellen / Literatur

Kommunikation und Softwarequalität

- Quelle
  - Vortrag von Hr. Ulbrich im Seminar ‘Qualität der Informatik’ auf dem ADV-Kongress Wien 1996
- Literatur
  - diverse Standardwerke zur Softwarequalität
  - Brodbeck, F.C. / Frese, M. (Hrsg.): „Produktivität und Qualität in Softwareprojekten. Psychologische Analyse und Optimierung von Arbeitsprozessen in der Softwareentwicklung“ Oldenbourg Verlag München Wien 1994
  - Grams, Timm: „Denkfallen und Programmierfehler“ Springer-Verlag Berlin Heidelberg 1990

Optimierung der Prozesse und/oder/mit/ohne persönliches QM des Mitarbeiters

- Quelle
  - Vortrag von Hr. Ulbrich auf dem 5. Kongress „Software-Qualitätsmanagement“ Köln/Bonn 2000
- Literatur
  - diverse Standardwerke zur Softwarequalität

Projekt- und Qualitätsmanagement in einem multinationalen Team

- Quelle
  - Vortrag von Hr. Ulbrich auf der 4. Fachtagung „Management und Controlling von IT-Projekten“ Glashütten 2001
  - Vortrag von Hr. Ulbrich auf dem 6. Kongress „Software-Qualitätsmanagement“ Bonn 2001
- Literatur
  - Thomas, Alexander (Hrsg.): „Kulturvergleichende Psychologie“ Hogrefe Verlag für Psychologie Göttingen, Bern, Toronto, Seattle 1993

- Thomas, Alexander (Hrsg.): „Psychologie interkulturellen Handelns“ Hogrefe Verlag für Psychologie Göttingen, Bern, Toronto, Seattle 1996
- Lewis, Richard D.: „Handbuch internationale Kompetenz“ Campus Verlag GmbH, Frankfurt/Main 2000

Projektmanagement zwischen Psychologie und Tools

- Quelle
  - Vortrag von Hr. Ulbrich auf der Konferenz zur Zukunft im Projektmanagement InterPM Glashütten 2004
- Literatur
  - Grams, Timm: „Denkfallen und Programmierfehler“ Berlin Heidelberg. Springer 1990

Berücksichtigung psychologischer Aspekte bei der Erhaltung der Codequalität in Wartungsprojekten

- Quelle
  - Vortrag von Hr. Ulbrich auf dem 8. Kongress „Software-Qualitätsmanagement“ Köln 2003
- Literatur
  - Grams, Timm: „Denkfallen und Programmierfehler“ Berlin Heidelberg. Springer 1990

Spezifika des Projektmanagements bei der fachlichen Abnahme

- Quelle
  - Vortrag von Hr. Ulbrich auf der Konferenz zur Zukunft im Projektmanagement InterPM Glashütten 2003
- Literatur
  - Pol, M.; Koonen, T.; Spillner, A.: „Management und Optimierung des Testprozesses“ Heidelberg 2000

Minimalanforderungen an QM ermitteln und einhalten

- Quelle
  - Vortrag von Hr. Ulbrich im Rahmen des Quality Points München 2002
- Literatur
  - diverse Standardwerke zur Softwarequalität

QM-Check einer Firma bzw. einer Struktureinheit (Inhalte, Umfang, mögliche Ergebnisse, Probleme)

- Quelle
  - Vortrag von Hr. Ulbrich im Rahmen des Quality Points München 2003
- Literatur
  - diverse Standardwerke zur Softwarequalität

Softwarerisiken = Risiken beim Test ?

- Quelle
  - Vortrag von Hr. Ulbrich im Rahmen des Quality Points München 2004
- Literatur
  - „Management und Optimierung des Testprozesses“ dpunkt.verlag 2000
  - „Compuware Risk-Based Testing“ (ein Leitfaden der Firma Compuware)
  - Webseiten anderer Testtoolhersteller

Risikooptimierte Qualitätssicherungstrategien bei Migrationsprojekten

- Quelle
  - Artikel von Hr. Brandmayr und Hr. Ulbrich für das Kompetenz-Center Projektmanagement 2004
- Literatur
  - diverse Standardwerke zur Softwarequalität

Qualitätsorakel oder Möglichkeiten einer Qualitätsprognose aus unvollkommenen Kenntnissen

- Quelle
  - Vortrag von Hr. Ulbrich zur Jahresversammlung der Gesellschaft für Softwarearchäologie 2005
- Literatur
  - diverse Standardwerke zur Softwarequalität

Kurzbiographie des Autors

Der Autor studierte an der Friedrich-Schiller-Universität Jena Mathematik und schloss das Studium als Diplommathematiker mit einer Arbeit zur Komplexität von Algorithmen ab.
In seinem bisherigen Berufsleben war der Autor als Programmierer, Projektmanager, Leiter Anwendungsentwicklung und Software-Qualitätsmanager bei verschiedenen Firmen erfolgreich tätig.
Dabei sammelte er u.a. umfangreiche Erfahrungen bei der erfolgreichen Gestaltung von Qualitätsmanagementsystemen, der Optimierung von Softwareentwicklungs- und Wartungsprozessen und der Beratung von Kunden in allen Belangen des Qualitätsmanagements.
Auf der Basis dieser Erfahrungen hielt er mehrere Vorträgen auf internationalen Tagungen zum Projektmanagement und zur Softwarequalität.
Er initiierte und moderierte mehrere Jahre den ‚Quality Point' München.
Der Autor ist zur Zeit Geschäftsführer von ‚Ulbrich IT-Dienstleistungen' (Ois Quality) und als Berater für Softwarequalität tätig.
Am 1.12.2003 wurde durch ihn die Gesellschaft für Softwarearchäologie gegründet, deren gewählter Vorsitzender er derzeit ist.
Die Gründung erfolgte getreu dem Motto 'Nihil tam admirabile est, quin quaerendo investigari possit' (Nichts ist so seltsam, dass es nicht erforscht werden könnte.).
Da die Wahrheit bekanntlich im Code liegt (sofern er nicht auch manipuliert wurde) ist es wichtig, zumindest den Code für wichtige Meilensteine der Softwareentwicklung zu bergen, zu sichten, zu analysieren und zu verwahren. Die dabei zur Anwendung kommenden Methoden ähneln den Methoden der konventionellen Archäologie. Deshalb wurde für dieses Forschungsgebiet der Name 'Softwarearchäologie' gewählt.
Die Gesellschaft betreibt zur Zeit im wesentlichen folgende Aktivitäten:

- Sammlung von Softwareartefakten aus allen Bereichen der Softwareentwicklung
- Erforschung von Methoden der Softwareanalyse
- Veranstaltung von Symposien zu ausgewählten Themen der Softwarearchäologie
- Online-Publikation von Forschungsergebnissen
- Herausgabe eines Jahreskalenders (zu speziellen Themen mit einem Augenzwinkern ohne Anspruch auf wissenschaftliche Korrektheit erstellt)

Die Gesellschaft wurde zu Forschungszwecken gegründet. Sie beabsichtigt weder, Gewinne zu erzielen noch ihre Forschungsergebnisse kommerziell zu vermarkten.

Unten links sehen Sie den Autor auf einer seiner Forschungsexpeditionen zu den unerforschten Stätten der Geschichte der Softwareentwicklung vor einer der Garagen, aus denen die bekannte ‚Garagensoftware' in den 70iger und 80iger Jahren des 20. Jahrhunderts ihren Siegeszug antrat.
Unten rechts sehen Sie den Autor als Qualitätslotsen in einem großen Softwareprojekt.