Experimentelle Methoden in der Informatik

D. Bär, O. Benke, T. Brückner, L. Prechelt, I. Redeke, R. Reussner, M. Schmidt, U. Veigel
Institut für Programmstrukturen und Datenorganisation
Fakultät für Informatik, Universität Karlsruhe

Bericht 38/95, Juli 1995

Zusammenfassung

Dieser Report enthält die Ausarbeitungen von Vorträgen aus einem Seminar gleichen Namens, das am 3./4. Juli 1995 am Institut für Programmstrukturen und Datenorganisation unter Leitung von Walter Tichy, Ernst Heinz, Paul Lukowicz und Lutz Prechelt stattfand.
Die Artikel geben einen Überblick über die mögliche Funktion und den Stellenwert experimentellen Vorgehens in verschiedenen Teilen der Informatik, sowie einerseits deren wissenschaftstheoretische Grundlage und andererseits ihre bisherige praktische Umsetzung.

Inhaltsverzeichnis

Einführung

Zahlreiche Erlebnisse und Beobachtungen im Umfeld der wissenschaftlichen Informatik-Forschung haben in unserer Arbeitsgruppe schon seit längerer Zeit ein unangenehmes Gefühl im Hinblick auf die methodische Korrektheit großer Teile dieser Forschung erzeugt.

Um dieses Gefühl entweder zu bestätigen oder zu widerlegen, entstand vor etwa einem Jahr die Studie [1]. Darin wurden über 400 Artikel aus angesehenen Journalen untersucht. Bei etwa 40 Prozent der Informatikartikel aus dieser Stichprobe, die einen nicht theoretisch-formal validierbaren Entwurfsvorschlag machen (für ein System, eine Sprache oder Formalismus, etc.) und folglich eine experimentelle Untersuchung und Auswertung dieses Vorschlags benötigen, findet sich *keinerlei* solche Auswertung. Bei vielen weiteren ist die Auswertung erheblich zu knapp.

Diese Beobachtungen bestätigen, daß der Einsatz von Experimentation in der wissenschaftlichen Informatik bislang bei Weitem zu kurz kommt. Um diesem Zustand an der Wurzel (sprich: in der Ausbildung des Nachwuchses) abzuhelfen, haben wir das Seminar durchgeführt, dessen Ergebnisse in diesem Bericht präsentiert werden. Das Seminar sollte einen Querschnitt bieten durch viele Aspekte experimentellen Arbeitens, die in der Informatik relevant sind:

1. *Ein beispielhaftes Experiment in der Informatik — und die Folgen.* Das Experiment von Knight und Leveson zur Prüfung der Fehlerunabhängigkeitshypothese bei der N-Versions-Programmierung ist ein Musterbeispiel für ein sauberes Laborexperiment in der Informatik und bildet deshalb den gewissermaßen gloriosen Einstieg in das Seminar. Interessant ist auch, wie die beiden Forscher für ihr Resultat angegriffen wurden.

2. *Die Grundlagen von Wissenschaft und Experimentation.* Diese Exkursion in die wissenschaftstheoretischen Grundlagen dient dazu, den Boden kennenzulernen, auf dem alle Experimentation sich bewegt und ihre Bedeutung und Funktion zu begreifen.

3. *Versuchsentwurf und -durchführung.*
4. *Statistische Techniken zur Experimentauswertung.*
Diese beiden Vorträge hätten einen Überblick geben sollen über mathematische und organisatorische Aspekte des Experimentierens — leider sind sie wegen Krankheit der Seminarteilnehmer ausgefallen und deshalb auch nicht in der Ausarbeitung enthalten. Zum Trost hier Verweise auf entsprechende Literatur:
S.D. Conte, H.E. Dunsmore, and V.Y. Shen: „Soft-

ware Engineering Metrics and Models“, Benjamin/Cummings, Menlo Park, CA, 1985, Kapitel 3.
Larry B. Christensen: „Experimental Methodology“, Allyn and Bacon, Needham Heights, MA, 6th edition, 1994, Kapitel 7-9,11-13.
David S. Moore and George P. McCabe: „Introduction to the Practice of Statistics“, W.H. Freeman and Company, New York, 1993, Kapitel 6 und 8.

5. *Ein Experiment zur Kosteneffektivität von Inspektionsverfahren.* Dieses Experiment repräsentiert ein Mittelding zwischen Laborexperiment (vollständig kontrollierte Bedingungen) und Beobachtung in einer realen Situation (kaum Einflußnahme). Es illustriert zugleich das Spannungsfeld zwischen wissenschaftlicher Genauigkeit und wirtschaftlichen Randbedingungen, in dem Experimentation sich meist bewegen muß.

6. *Die Aussagekraft von Experimenten,* oder: *Sind Flußdiagramme besser als Pseudokode?* In diesem Beitrag geht es um die Frage der Zuverlässigkeit von Aussagen, die mit Experimenten gewonnen werden. Als Beispiel wird ein Paar von zwei Artikeln mit widersprüchlichen Resultaten betrachtet. Einmal werden strukturierte Flußdiagramme für wirksamer als Pseudokode für das Programmverständnis befunden, ein andermal nicht. Und beide Experimente haben so ihre Schwächen... Verdeutlicht die Notwendigkeit der Wiederholung von Experimenten.

7. *Natur-, Ingenieur- und Humanwissenschaft: Unterschiede beim experimentellen Arbeiten.* Die Randbedingungen für das Experimentieren wie auch die Aussagekraft der Ergebnisse unterscheidet sich erheblich zwischen diesen drei Kategorien von Wissenschaften: Von rein theoriebasiertem, exakt reproduzierbarem Arbeiten z.B. in der Physik, über technologie- und apparatabhängiges Arbeiten in Ingenieurwissenschaften wie auch weiten Teilen der Informatik, bis hin zu den extrem schwer reproduzierbaren und verallgemeinerbaren Kontexten, in denen Menschen handeln, wie im Bereich des Software Engineering. Dieser Beitrag klärt die prinzipiellen Unterschiede im Vorgehen durch Vergleich von Beispielen und behandelt als Schwerpunkt die „Benchmark“-Problematik in der Informatik am Beispiel von Sprachverarbeitung: Aufgabe, Vorgehen, Stärken, Probleme.

8. *Herausforderung Software Engineering: Beobachten und Messen außerhalb des Labors.* Das „Experimentieren“ im Sinne kontrollierter Laborversuche ist im Hauptbereich des Software Engineering, dem Programmieren im Großen, fast unmöglich. Eine praktikable Arbeitsweise erzwingt oftmals, kontrollierte Experimente durch Beobachtung einer nur zum Teil entworfenen und nur maßvoll beeinflußbaren Situation zu ersetzen. Wie man dennoch zu Erkenntnissen gelangen kann, ist Gegenstand dieses Beitrags. Die Vorgehensweise von Laborexperimenten zu Fragen des Software Engineering wird ebenfalls an einem Beispiel vorgestellt.

Literatur

[1] Walter F. Tichy, Paul Lukowicz, Lutz Prechelt und Ernst A. Heinz. Experimental Evaluation in Computer Science: A Quantitative Study. Journal of Systems and Software, pp. 9-18, Januar 1995. (Auch als TR 17/94, Fakultät für Informatik, Universität Karlsruhe, August 1994.)

Ein beispielhaftes Experiment in der Informatik — und die Folgen

Thomas Brückner

Zusammenfassung

Im Januar 1986 veröffentlichten John C. Knight und Nancy G. Leveson unter dem Titel *„An Experimental Evaluation of the Assumption of Independence in Multiversion Programming [1]“* einen Bericht über ihr kontrolliertes Experiment zur Fehlerunabhängigkeitshypothese in der N-Versionsprogrammierung (NVP). Dieser Bericht sorgte für einigen Wirbel in der Gemeinde der NVP-Befürworter, weil Knight und Leveson nachweisen konnten, daß die Fehlerunabhängigkeitshypothese der NVP für ihr spezielles Experiment nicht zutreffend war.

Obwohl das Knight/Leveson-Experiment als Musterbeispiel für ein sauberes Laborexperiment in der Informatik betrachtet werden kann, wurden die Autoren in den darauffolgenden Jahren für Ihre Ergebnisse immer wieder und mit zunehmender Schärfe kritisiert bzw. angegriffen. Nachdem sich die Anfeindungen der NVP-Befürworter immer weiter von den in ihrem Bericht gezogenen Schlußfolgerungen entfernten, sahen sich Knight und Leveson gezwungen, im Januar 1990 eine öffentliche Stellungnahme [2] dazu abzugeben.

Die vorliegende Seminararbeit faßt Experimentbericht [1] und Stellungnahme [2] zusammen und dient als Einstieg und Motivation für das Seminar „Experimentelle Methoden in der Informatik“.

1 Einführung

Fehlertoleranz wird in der Informatik vor allem bei jenen Systemen gefordert, die in derart kritischen Anwendungsbereichen zum Einsatz kommen, daß bei einem Versagen erhebliche Gefahren für Menschen und Sachwerte entstehen. Beispiele sind Kernkraftwerke, Flugsicherungssysteme, rechnergesteuerte Röntgengeräte usw. Ein weitverbreiteter Ansatz zur Verbesserung von Zuverlässigkeit und Sicherheit derartiger Systeme ist die Verwendung *replizierter Subsysteme*, d.h. kritische Systemkomponenten werden durch eine bestimmte Anzahl funktionsgleicher Komponenten vervielfältigt. Bezogen auf die Softwareentwicklung bedeutet dies, daß n Versionen ein und derselben Komponente vorliegen, die i.d.R. von n unterschiedlichen Entwicklern entworfen und implementiert werden. Jede dieser Komponenten erhält dieselbe Eingabe (bereitgestellt von einem sogenannten 1:n-Verteiler) und sollte im Idealfall eine Ausgabe liefern, die zu den Ausgaben der übrigen Versionen äquivalent ist. Sollten sich die Ausgaben der Komponenten jedoch unterscheiden, so wählt man per *Mehrheitsentscheidung* die von den meisten Komponenten berechnete Ausgabe als Endergebnis aus. Diese Methode wird auch als *diversitäre Programmierung* oder *N-Versions-Programmierung* (NVP) bezeichnet.

Man geht gemäß der *Fehlerunabhängigkeitshypothese* davon aus, daß das gleichzeitige Auftreten eines Fehlers in mehreren der Komponenten derart unwahrscheinlich ist, daß das Gesamtsystem in der Praxis nahezu keine Ausfälle zeigt. Sei die Wahrscheinlichkeit, daß eine bestimmte Komponente K_i zu einem Zeitpunkt t bei der Eingabe von x ein Fehlverhalten zeigt, mit $P(K_i(x) = \text{Fehler})$ bezeichnet. S sei das aus den Komponenten K_1 bis K_n bestehende Gesamtsystem. Dann ergibt sich bei statistischer Unabhängigkeit der Fehler und gleicher Fehlerwahrscheinlichkeit der Versionen die Fehlerwahrscheinlichkeit des Systems S zum Zeitpunkt t zu

$$\begin{aligned} P(S(x) = \text{Fehler}) &= \prod_{i=1}^{n} P(K_i(x) = \text{Fehler}) \\ &= \Big(P(K(x) = \text{Fehler})\Big)^n \end{aligned}$$

Die Fehlerwahrscheinlichkeit des Systems nimmt demnach unter den genannten Voraussetzungen exponentiell mit der Anzahl der replizierten Komponenten ab. Einige Befürworter der NVP waren deshalb der Meinung, man könne dadurch Softwareentwicklungskosten einsparen, daß man statt einer einzigen, intensiv getesteten Softwarekomponente mehrere billigere und unzuverlässigere Komponenten in NVP-Manier koppelt. Auch die Testphase ließe sich ihrer Ansicht nach erheblich verkürzen, indem sich die diversen Komponenten ohne menschliches Zutun gegenseitig testen.

Die Fehlerunabhängigkeitsvoraussetzung kann aber nur dann gegeben sein, wenn sich die Komponenten in ihrem Inneren (verwendete Algorithmen und Datenstrukturen, Umsetzung in Programmtext etc.) hinreichend voneinander unterscheiden. Idealerweise sollte deshalb bereits die Spezifikation in diversen Formulierungen vorliegen. Aus praktischen und finanziellen Gründen wird allerdings *meist eine gemeinsame Spezifikation* verwendet. Entwurf, Implementation und Test erfolgen dann für jede Komponente getrennt. Es muß deshalb darauf geachtet werden, daß die Spezifikation genügend Spielraum für diversitäre Programmierung bietet. Sie sollte lediglich beschreiben, *was* zu tun ist, aber nicht festlegen, wie es zu tun ist. Die Spielräume für die Verwendung von Datenstrukturen sind jedoch schon dadurch beschränkt, daß die Endergebnisse der einzelnen Komponenten für den Mehrheitsentscheid vergleichbar sein müssen. In vielen

Fällen ist es darüber hinaus empfehlenswert, bereits Zwischenergebnisse miteinander zu vergleichen, um im Fehlerfall so früh wie möglich korrigierend eingreifen zu können[1].

N-Versionsprogrammierung kann den oben angegebenen theoretischen Zuverlässigkeitsgewinn nur dann erbringen, wenn die Fehlerunabhängigkeitsannahme zutrifft. Je wahrscheinlicher das gleichzeitige Fehlverhalten mehrerer Versionen ist, desto unzuverlässiger wird ein N-Versionssystem.

2 Das Knight/Leveson-Experiment

Anlaß für das von John C. Knight[2] und Nancy G. Levenson[3] durchgeführte Experiment war die Vermutung, daß Fehler oftmals weniger von den Programmierern abhängen, sondern eher von der Schwierigkeit der Problemstellung an der jeweiligen kritischen Stelle. Demnach wäre die Fehlerwahrscheinlichkeit gerade bei schwierig umzusetzenden Teilproblemen besonders hoch und damit auch die Wahrscheinlichkeit, daß mehreren Programmierern an diesen Stellen ein Fehler unterläuft.

Für ihr Experiment baten Knight und Leveson Studenten ihrer Vorlesungen, ausgehend von einer gegebenen Spezifikation ein einfaches Anti-Raketen-System zu programmieren. Eingabe für dieses System waren bis zu 100 geordnete Punkte im zweidimensionalen Raum (Radarechos) und eine bestimmte Anzahl Parameter. Die Aufgabe des Systems bestand darin, anhand der Eingabe die Gültigkeit 15 sogenannter *Abfangjäger-Startbedingungen* (Launch Interceptor Conditions, LICs) zu prüfen und gegebenenfalls ein Startsignal für Abfangjäger zu setzen. Die erste Startbedingung lautete beispielsweise:

> *LIC 1: Es gibt mindestens eine Menge zweier konsekutiver Datenpunkte, die eine Distanz größer LENGTH1 auseinanderliegen.*
>
> *(0 <= LENGTH1)*

In einem ersten Verarbeitungsschritt sollten alle 15 Startbedingungen geprüft und die Ergebnisse in den sogenannten *Erfüllbarkeitsvektor* (Conditions Met Vector, CMV) eingetragen werden. Beispiel:

$$\begin{array}{c} LIC1 \\ LIC2 \\ LIC3 \\ LIC4 \\ \vdots \\ LIC15 \end{array} \begin{pmatrix} wahr \\ falsch \\ falsch \\ wahr \\ \vdots \\ wahr \end{pmatrix}$$

Ein weiterer Eingabeparamter, die sogenannte *Logische Verknüpfungsmatrix* (Logical Connector Matrix, LCM) gab an, welche logischen Operationen anschließend auf die Elemente des Erfüllbarkeitsvektors angewendet werden sollten[4]. Beispiel:

$$\begin{array}{c} \\ LIC\ 1 \\ LIC\ 2 \\ LIC\ 3 \\ LIC\ 4 \\ LIC\ 5 \\ \vdots \\ LIC\ 15 \end{array} \begin{array}{c} \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & \cdots & 15 \end{array} \\ \begin{pmatrix} \wedge & \wedge & \vee & \wedge & * & \cdots & * \\ \wedge & \wedge & \vee & \vee & * & \cdots & * \\ \vee & \vee & \wedge & \wedge & * & \cdots & * \\ \wedge & \vee & \wedge & \wedge & * & \cdots & * \\ * & * & * & * & * & \cdots & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ * & * & * & * & * & \cdots & * \end{pmatrix} \end{array}$$

Ergebnis der Verknüpfung der Startbedingungen des Erfüllbarkeitsvektors gemäß Logischer Verknüpfungsmatrix sollte die sogenannte *Vorläufige Entriegelungsmatrix* (Primary Unlocking Matrix, PUM) sein. Beispiel:

$$\begin{array}{c} \\ LIC\ 1 \\ LIC\ 2 \\ LIC\ 3 \\ LIC\ 4 \\ LIC\ 5 \\ \vdots \\ LIC\ 15 \end{array} \begin{array}{c} \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & \cdots & 15 \end{array} \\ \begin{pmatrix} w & f & w & f & w & \cdots & w \\ f & f & w & w & w & \cdots & w \\ w & w & w & w & w & \cdots & w \\ f & w & w & f & w & \cdots & w \\ w & w & w & w & f & \cdots & w \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ w & w & w & w & w & \cdots & f \end{pmatrix} \end{array}$$

Die Diagonalelemente der Vorläufigen Entriegelungsmatrix waren Bestandteil der Eingabe und legten fest, welche der Startbedingungen für die nachfolgenden Berechnungen tatsächlich relevant waren ($\mathrm{PUM}[i,i] = wahr$).

Eine Transformation dieser Matrix sollte den *Endgültigen Entriegelungsvektor* (Final Unlocking Vector, FUV) ergeben, nach der Regel

$$\mathrm{FUV}[i] = wahr \quad \Leftrightarrow \quad \mathrm{PUM}[i,i] = falsch \vee \forall j\ \mathrm{PUM}[i,j] = wahr$$

Die eigentliche Ausgabe des Programms, das Abschußsignal für den Abfangjäger, sollte genau dann auf $wahr$

[1] Die Schnittstellen, an denen die Komponenten Zwischenergebnisse austauschen, werden auch *Cross Check Points* genannt.

[2] University of Virginia, Charlottesville

[3] University of California, Irvine

[4] $\wedge$ = logisches *und*, $\vee$ = logisches *oder*, $*$ = *nicht verwendet*

gesetzt werden, wenn alle Elemente des Endgültigen Entriegelungsvektors *wahr* waren.

Knight und Leveson wählten diese (nicht ganz triviale) Aufgabe hauptsächlich aus zwei Gründen: Zum einen wäre ein solches Anti-Raketen-System in der Realität ein äußerst sicherheitskritisches System und somit ein Kandidat für die Anwendung von NVP-Techniken. Zum anderen wurde die Aufgabe bereits in früheren Experimenten anderer verwendet ([4], [5]). Schwachstellen und Mißverständlichkeiten in der Spezifikation waren nach Meinung von Knight und Leveson dort größtenteils beseitigt worden.

Es wurden keine softwaretechnischen Methoden vorgegeben. Die Studenten durften ihre in entsprechenden Vorlesungen erworbenen Kenntnisse frei nutzen. Knight und Leveson gaben jeweils eine kurze Einführung in das Experiment und ermahnten die Studenten, sich nicht gegenseitig abzusprechen und das Ergebnis auf diese Weise zu verfälschen. Informationen von anderer Seite (Literatur oder klärende Gespräche mit am Experiment nicht direkt beteiligten Personen) durften jedoch beliebig eingeholt werden.

Von den 27 Versuchsteilnehmern strebten 14 den akademischen Grad *Bachelor of Science* an, 8 den Grad *Master of Science* und 4 den Doktorgrad. Die Studenten, die den Grad *Bachelor of Science* bereits erreicht hatten, stammten aus den unterschiedlichsten Fachrichtungen.

Vor Erreichen des *Bachelor of Science* hatten die Studenten Mathematikvorlesungen im Umfang von 12–45 Wochenstunden und Informatikvorlesungen im Umfang von 6–45 Wochenstunden besucht. Studenten höherer Quartale gaben darüber hinaus weitere 0–19 Wochenstunden Mathematik und 0–30 Wochenstunden Informatik an, die sie im Rahmen ihres Studiums gehört hatten. Die Praxiserfahrung im EDV-Bereich reichte von „keine Erfahrung“ bis „zehn Jahre professionelle Programmierpraxis“. Vier Teilnehmer stuften ihre Kenntnisse der im Experiment verwendeten Programmiersprache *Pascal* als „mittelmäßig“ ein, 18 als „umfangreich“ und 4 als „sehr gut“.

Die Studenten erhielten je 15 zufällig ausgewählte Referenzdatensätze zur Kontrolle. Es wurde außerdem erwartet, daß jeder Teilnehmer sein Programm mit weiteren, selbst entworfenen Testfällen intensiv testete. Bestand ein abgegebenes Programm einen Akzeptanztest von 200 zufällig gewählten Testfällen, wurde es in die Reihe der Experimentobjekte aufgenommen. Bis zu diesem Zeitpunkt hatten die Studenten nach eigenen Angaben 1–35 Stunden für das Lesen der Spezifikation benötigt (11,3%), 4–50 Stunden für Entwurf und Implementation (32,9%) sowie weitere 4–70 Stunden für die Testphase (55,8%).

Nachdem die 27 Programme eingereicht worden waren, wurde ihr Praxiseinsatz mittels einer Million weiterer Testfälle simuliert. Aus früheren Experimenten stand ein Referenzprogramm (das sogenannte *Goldprogramm*) zur Verfügung, das sich als äußerst zuverlässig erwiesen hatte. Somit konnte die Simulation vollautomatisch als Vergleich der Ausgaben der 27 Programme mit den Ausgaben des Goldprogramms erfolgen.

Geprüft wurden die 15 Elemente des Erfüllbarkeitsvektors, die 15 × 15 Elemente der Vorläufigen Entriegelungsmatrix und das eigentliche Startsignal, d.h. insgesamt 241 Wahrheitswerte. Ein Fehler wurde registriert, wenn einer der 241 Werte falsch war (keine Übereinstimmung mit der Ausgabe des Goldprogramms) oder das Programm abbrach (z.B. Division durch Null).

3 Ergebnisdaten des Experiments

Die eingereichten Programme besaßen insgesamt hohe Qualität, wie die folgende Tabelle der individuellen Fehlerraten verdeutlicht:

Version	Fehler	$P(Erfolg)$	Version	Fehler	$P(Erfolg)$
1	2	0,999998	15	0	1,000000
2	0	1,000000	16	62	0,999938
3	2297	0,997703	17	269	0,999731
4	0	1,000000	18	115	0,999885
5	0	1,000000	19	264	0,999736
6	1149	0,998851	20	936	0,999064
7	71	0,999929	21	92	0,999908
8	323	0,999677	22	9656	0,990344
9	53	0,999947	23	80	0,999920
10	0	1,000000	24	260	0,999740
11	554	0,999446	25	97	0,999903
12	427	0,999573	26	883	0,999117
13	4	0,999996	27	0	1,000000
14	1368	0,998632			

Bemerkenswert hoch war allerdings die Anzahl der Testfälle, die bei mehr als einem Programm zu Fehlverhalten führten (*korrelierte Fehler*). Die folgende Tabelle zeigt, mit welcher Wahrscheinlichkeit und Häufigkeit Programme beim selben Testfall ein falsches Resultat lieferten:

Anzahl der Programme	Wahrscheinlichkeit	Häufigkeit
2	0,00055100	551
3	0,00034300	343
4	0,00024200	242
5	0,00007300	73
6	0,00003200	32
7	0,00001200	12
8	0,00000200	2

Demnach trat sogar zweimal der Fall auf, daß acht von unterschiedlichen Programmierern entwickelte Programme gleichzeitig versagten. Wären diese Programme Teil eines redundanten Systems gewesen, wären mindestens neun weitere Programme nötig gewesen, um diese Fehler zu kompensieren. Jeder der korrelierten Fehler trat darüberhinaus in Programmen *beider*

beteiligter Universitäten auf. Da diese immerhin 3000 Meilen voneinander entfernt liegen, dürften Absprachen unter den Studenten unwahrscheinlich gewesen sein.

4 Auswertung der Ergebnisdaten

Um zu mathematisch gesicherten Erkenntnissen zu gelangen, untersuchten Knight und Leveson die ermittelten Fehler auf stochastische Unabhängigkeit. Stochastische Unabhängigkeit zweier Zufallsgrößen ist gegeben, wenn gilt:

$$P(A|B) = P(A) \ \textit{und} \ P(B|A) = P(B)$$

Die Wahrscheinlichkeit für das gemeinsame Fehlverhalten zweier Programme muß demnach gleich sein der Wahrscheinlichkeit für das Fehlverhalten nur eines der Programme.

Knight und Leveson hatten sich bei der Generierung der Testfälle an Empfehlungen der Firma Boeing für kritische Testsituationen orientiert, jeder Testfall entsprach einer Ausnahmesituation. Da die Testfälle innerhalb dieses Rahmens zufällig generiert wurden und mögliche Fehlerursachen a priori unbekannt waren, nahmen Knight und Leveson gleiche Fehlerwahrscheinlichkeit für alle Testfälle an.

Im Falle der Gültigkeit der Fehlerunabhängigkeitshypothese gilt für die Wahrscheinlichkeit $P_{keinFehler}$, daß jedes der N Programme ein korrektes Ergebnis liefert ($p_i :=$ Fehlerwahrscheinlichkeit des Programmes i):

$$P_{keinFehler} := P_0 := (1 \Leftrightarrow p_1)(1 \Leftrightarrow p_2) \cdots (1 \Leftrightarrow p_N)$$

Die Wahrscheinlichkeit, daß ein bestimmtes Programm i versagt, die übrigen $N \Leftrightarrow 1$ Programme jedoch korrekt arbeiten, berechnet sich zu:

$$(1 \Leftrightarrow p_1) \cdots (1 \Leftrightarrow p_{i-1}) p_i (1 \Leftrightarrow p_{i+1}) \cdots (1 \Leftrightarrow p_N) = \frac{P_0 p_i}{1 \Leftrightarrow p_i}$$

Demnach gilt für die Wahrscheinlichkeit P_1, daß genau *eines* der N Programme ein falsches Resultat liefert:

$$P_1 = \frac{P_0 p_1}{1 \Leftrightarrow p_1} + \frac{P_0 p_2}{1 \Leftrightarrow p_2} + \cdots + \frac{P_0 p_N}{1 \Leftrightarrow p_N}$$

Die Wahrscheinlichkeit, daß der gegebene Testfall bei *mehr als einem* Programm zu einem Fehler führt, ist:

$$P_{mehr} = 1 \Leftrightarrow P_0 \Leftrightarrow P_1$$

P_{mehr} ist unabhängig vom betrachteten Testfall. Derselbe Testfall kann zu unterschiedlichen Zeitpunkten mehrfach auftreten. Die Abarbeitung der n Testfälle kann demzufolge als „Urnenziehen mit Zurücklegen“ betrachtet werden. Dies führt zu einer Binomialverteilung der Zufallsvariable $K =$ „Anzahl der Testfälle, bei denen mindestens zwei Programme ein falsches Resultat liefern“:

$$P(K = x) = \binom{n}{x} (P_{mehr})^x (1 \Leftrightarrow P_{mehr})^{n-x}$$

Eine Binomialverteilung läßt sich durch die standardisierte Normalverteilung N(0,1) approximieren, wenn man die Zufallsvariable geeignet transformiert [3]. In unserem Fall hat

$$Z = \frac{K \Leftrightarrow n P_{mehr}}{\sqrt{n P_{mehr}(1 \Leftrightarrow P_{mehr})}}$$

eine Verteilung, die durch N(0,1) angenähert werden kann.

Bei Gültigkeit der Fehlerunabhängigkeitsannahme stellen die obigen Formeln ein korrektes Modell für die zu erwartenden Experimentergebnisse dar (Nullhypothese des Experiments). Wäre Z normalverteilt, müßte der Wert dieser Zufallsvariable bei Einsetzen der Experimentdaten mit 99%iger Sicherheit kleiner als 2,33 sein (99% der Standardnormalverteilung).

Knight und Leveson untersuchten in ihrem Experiment $N = 27$ Programme anhand von $n = 1.000.000$ Testfällen und konnten dabei $K = 1255$ Testfälle ermitteln, bei denen mindestens zwei der Programme eine falsche Ausgabe lieferten. Die Wahrscheinlichkeit P_{mehr} konnten sie mit Hilfe der beobachteten Fehlerwahrscheinlichkeiten der Programme abschätzen. Mit diesen Werten erhielten sie $Z = 100,51 > 2,33$, woraus sie folgern konnten, daß die Nullhypothese mit 99%iger Sicherheit zu verwerfen war. Also war die Fehlerunabhängigkeitshypothese mit 99%iger Konfidenz nicht erfüllt.

5 Analyse der Fehler

Knight und Leveson untersuchten anschließend, welche Programmfehler die Ursachen für beobachtetes Fehlverhalten waren. Als Programmfehler definierten sie hierbei einen Programmteil, der bei mindestens einem Testfall eine falsche Ausgabe verursachte. In der folgenden Tabelle ist für jede Programmversion die Anzahl der gefundenen Fehler aufgeführt. Jeder dieser Fehler hatte den Akzeptanztest von 200 zufällig ausgewählten Testfällen ohne Auffälligkeiten überstanden:

Version	Fehler	Version	Fehler
1	1	15	0
2	0	16	2
3	4	17	2
4	0	18	1
5	0	19	1
6	3	20	2
7	1	21	2
8	2	22	3
9	2	23	2
10	0	24	1
11	1	25	3
12	2	26	7
13	1	27	0
14	2	Σ	45

Bei den individuellen (nichtkorrelierten) Fehlern handelte es sich meist um allgemeine Flüchtigkeitsfehler. Die folgende Funktion beispielsweise gibt für den Fall $a = b$ kein gültiges Ergebnis zurück:

```
function max(a,b) : integer) : integer;
begin
if a > b then max := a;
if a < b then max := b;
end;
```

Ein ähnlicher Fehler in einem der 27 Programme führte bei 607 von 1 Million Testfällen zu einer Fehlausgabe. Bemerkenswert ist, daß selbst vermeintlich schwerwiegende Fehler wie die Verwendung falscher Indizes nur geringe Auswirkungen zeigten:

```
bsp3pkte(x[i], y[i], x[j], y[i], x[k], y[k]);
```

Im obigen Prozeduraufruf sollte der Index des vierten Argumentes ‚j' heißen. Dennoch konnte bei einem mit einem ähnlichen Fehler behafteten Programm nur in 1297 Testfällen ein Fehlverhalten beobachtet werden.

Fehler, die von mehreren Programmautoren auf ähnliche Weise begangen wurden (korrelierte Fehler), ließen sich oft auf mangelndes Fachwissen zurückführen und standen meist in engem Zusammenhang zur Semantik der Aufgabenstellung. Mehrere Autoren berücksichtigten beispielsweise nicht, daß die folgende mathematische Äquivalenz bei begrenzter Maschinengenauigkeit keine Gültigkeit besitzt:

$$\alpha, \beta \in [0, 2\pi]: \quad cos(\alpha) = cos(\beta) \Leftrightarrow \alpha = \beta$$

Als Teilproblem zur Überprüfung der Startbedingungen (LICs) war für drei gegebene Punkte A, B und C zu bestimmen, ob diese auf einer Linie liegen. Ausgehend vom allgemeinen Fall bestimmten zahlreiche Programmierer den Winkel φ zwischen $\overline{AB}$ sowie $\overline{AC}$ und bejahten die Kollinearität im Fall $\varphi = 0°$. Liegt jedoch A in der Mitte der drei kollinearen Punkte, so ist $\varphi = 180°$. Dieser Spezialfall wurde zuweilen vergessen, was bei einem der Programme 231 Fehlausgaben verursachte, bei einem anderen lediglich 37.

6 Schlußfolgerungen

Knight und Leveson folgerten aus ihren Ergebnissen, daß die Fehlerunabhängigkeitshypothese *für ihr spezielles Experiment* mit 99%iger Konfidenz abzulehnen sei. Sie verallgemeinerten nicht auf alle Problemstellungen und Entwicklungsmethoden, befürchteten jedoch, daß die von vielen NVP-Befürwortern theoretisch angenommene Zuverlässigkeit replizierter Software in der Praxis unter Umständen nicht erreicht werde. Ihrer Meinung sollte mit dem Einsatz von NVP in sicherheitskritischen Bereichen solange gewartet werden, bis ein schlüssiger Beweis für die Gültigkeit der Fehlerunabhängigkeitsannahme im allgemeinen Fall erbracht worden sei.

Aufbauend auf der bereits erwähnten Analyse der gefundenen Programmfehler stellten Knight und Leveson die Vermutung an, trotz intensivster Testphasen auftretende, korrelierte Fehler seien eher *problem-* als *programmierer*abhängig. Zufällige, individuelle Fehler würden mit hoher Wahrscheinlichkeit bereits vom Übersetzer erkannt, während bei komplexeren Fehlern, die die Semantik der Problemstellung betreffen, die gängigen Prüfmaßnahmen versagten. Außerdem sei nicht auszuschließen, daß eine ähnliche Ausbildung der Programmierer und die Nutzung derselben Sekundärwissensquellen zu einer vergleichbaren Vorgehensweise bei der Lösung eines Problems führen könnten und damit auch zu vergleichbaren Fehlern. In jedem Fall sei weitere Forschung bezüglich des Auftretens korrelierter Fehler sinnvoll und notwendig.

7 Die Nachgeschichte

Die Veröffentlichung des Knight/Leveson-Experiments sorgte für einigen Wirbel in der betroffenen Gemeinde der NVP-Befürworter. Immerhin hatten Knight und Leveson gewagt, an den Grundpfeilern der NVP zu rütteln, wofür sie von deren Verfechtern in der Folgezeit wiederholt und mit zunehmender Schärfe in Wort und Schrift angegriffen wurden.

Die Hauptkritiker waren Professor Algirdas Avizienis von der University of California, Los Angeles (UCLA), der sich durch zahlreiche Veröffentlichungen zur NVP-Methodik einen Namen gemacht hatte, sowie dessen frühere Studenten John Kelly, Michael Lyu und Mark Joseph. Sie führten vor allem zwei Hauptargumente ins Feld, mit denen sie versuchten, die Ergebnisse von

Knight und Levenson zu torpedieren, um deren Experiment als unbrauchbar und unwissenschaftlich abzuqualifizieren. Zum einen behaupteten sie, in eigenen Experimenten zu gänzlich anderen Ergebnissen gekommen zu sein, zum anderen warfen sie Knight und Leveson vor, keine für NVP geeigneten softwaretechnischen Methoden verwendet zu haben, weshalb die Ergebnisse ihres Experiments keine praxisrelevante Aussagekraft besäßen.

Zunächst versuchten Knight und Leveson, Mißverständnisse und Fehlinterpretationen ihrer Arbeit in persönlichen Gesprächen auszuräumen. Weil die Kritik jedoch ungeachtet dessen an Heftigkeit und Unsachlichkeit weiter zunahm, sahen sie sich letztendlich im Jahr 1990 (also vier Jahre nach der Publikation ihres Experiments!) gezwungen, eine Gegendarstellung zu veröffentlichen [2]. Ausgehend von Zitaten aus Veröffentlichungen der Avizienis-Gruppe versuchten sie, Falschaussagen über ihr Experiment zu widerlegen und das in der Fachwelt entstandene Bild der Unwissenschaftlichkeit zurechtzurücken.

Mark Joseph beispielsweise hatte in seiner Dissertation behauptet, mehrere an der UCLA durchgeführte Experimente hätten die von Knight und Leveson berichteten hohen Fehlerraten nicht bestätigt [6]. Knight und Leveson gaben dazu eine Tabelle an, in der sie die wichtigsten Ergebnisse der in Frage kommenden Experimente den Daten ihres eigenen Experimentes gegenüberstellten[5]:

	K+L	Chen	Kelly	NASA	UCLA
Anzahl Versionen	27	7	18	20	6
∅ Fehlerzahl	1.6	k.A.	k.A.	k.A.	1.8
Anzahl Testfälle	1.000.000	32	100	921.000	1000
∅ indiv. Fehlerrate	0,0007	k.A.	0,27	0,006	k.A.
∅ Fehlerrate 3 Vers.	0,00004	0,1	0,2	0,002	k.A.
Stat. Unabhängigkeit	nein	n.g.[6]	n.g.	nein	n.g.

Wie man dieser Tabelle entnehmen kann, liegt sowohl die individuelle als auch die korrelierte Fehlerrate (hier angegeben für 3 Versionen) des Knight/Leveson - Experimentes deutlich unter den Fehlerraten der übrigen Experimente. Darüber hinaus wurde die statistische Unabhängigkeit der Fehler meist gar nicht getestet. Obwohl die Zahlen eindeutig anderes belegen, versuchte die Avizienis-Gruppe, die Experimente in eine von ihnen gewünschte Richtung zu deuten.

Professor Avizienis behauptete in einem anläßlich des 11. Welt-Computer-Kongresses zur Fehlertoleranz veröffentlichten Artikel, daß die Bemühungen von Knight und Leveson die Fallstricke der verfrühten Suche nach numerischen Experimentalergebnissen aufwiesen [7]. Knight und Leveson stellten hierauf in ihrer Stellungnahme die Frage, weshalb Avizienis dann trotz des von ihm angenommenen frühen Entwicklungsstadiums der NVP für einen Einsatz derselben in sicherheitskritischen Systemen plädiere.

Hauptangriffspunkt der NVP-Befürworter waren außerdem die von Knight und Leveson verwendeten softwaretechnischen Methoden. Avizienis, Joseph und andere vertraten die Auffassung, das Knight/Leveson - Experiment hätte ein grundlegend anderes Ergebnis gehabt, wenn nur für die NVP geeignete Softwareentwicklungsmethoden angewandt worden wären. Mit geeigneten Methoden meinten sie hierbei in erster Linie die von ihnen entwickelten und in zahlreichen Publikationen veröffentlichten Paradigmen. Zitat Professor Avizienis: „Die numerischen Ergebnisse von Knight und Leveson messen einzig die Qualität ihrer von Gelegenheitsprogrammierern erstellten Software“ [7].

Interessanterweise läßt sich über die von Avizienis vorgeschlagenen Entwicklungsmethoden sagen, daß sie einen stark restriktiven Entwurfsprozeß vorsehen, der potentiell mögliche Diversität an vielen Stellen einschränkt. Er selbst stellt in seinem zum UCLA-Experiment veröffentlichten Bericht fest: „Zwei Faktoren wurden im Rahmen dieser Untersuchung festgestellt, die die tatsächliche Diversität einschränken [...] Durch Bilder veranschaulichte Algorithmen wurden im allgemeinen implementiert, indem das entsprechende Bild von oben nach unten umgesetzt wurde. Im nachhinein läßt sich sagen, daß ein zweiter Grund für den Mangel an Diversität eine Überspezifikation der logischen Beschreibung [...] war.“ [8]

Dennoch schrieben Professor Avizienis und Michael Lyu in einem Artikel, den sie anläßlich einer Konferenz über digitale Systeme im Luftfahrtbereich veröffentlichten: „Wie man sieht, hatte das Knight/Leveson - Experiment einen eher kleineren Maßstab, da die Spezifikation sechs Seiten lang war und in 327 Zeilen programmiert werden konnte. Der Maßstab des UCLA-Experimentes mit 64 Seiten Spezifikation und mindestens 1250 Quelltextzeilen war bedeutend größer.“[9]

Knight und Leveson zeigten sich besonders betroffen, wenn den in ihrem Experiment verwendeten Programmen mangelnde Qualität unterstellt wurde. In der Tat war die durchschnittliche Fehlerrate von 0,07% sehr gering. Die schlechteste Fehlerrate betrug lediglich 0,9%. Die 27 Programme waren ihrer Meinung nach demnach nicht unzuverlässiger als die meisten von „professionellen“ Programmierern entwickelten, und mindestens ebenso gut wie die in den Experimenten ihrer Kritiker erstellte Software.

Abschließend bemerkten sie in ihrer Stellungnahme, daß bis dato noch nicht eindeutig experimentell nachgewiesen worden sei, daß NVP eine besonders hohe Zuverlässigkeit oder zumindest eine höhere Zuverlässig-

[5] Zwei der angeführten Experimente stammten aus der Zeit vor Knight und Levensons Untersuchung (Chen, 1978, und Kelly, 1983), die anderen beiden Experimente wurden in den Jahren 1987 (UCLA/H) bzw. 1989 (NASA) veröffentlicht.

[6] n.g. = nicht getestet

keit als andere Softwareentwicklungsmethoden garantiert. Solange kein Beweis für diese Hypothese gefunden sei, werfe das Vertrauen auf diese Methode und ihre Anwendung in sicherheitskritischen Systemen grundlegende ethische und moralische Fragen auf. Angriffe auf ihre Person oder ihre Papiere änderten daran nichts.

8 Zusammenfassung

Die Reaktionen auf das Knight / Leveson - Experiment sind symptomatisch für den vergleichsweise niedrigen Stellenwert, den heute ein wissenschaftlich exakt durchgeführtes Experiment in der Informatik hat. Während die experimentelle Vorgehensweise in traditionellen Naturwissenschaften wie Physik, Chemie und den davon abgeleiteten Ingenieurwissenschaften (wie z.B. Maschinenbau, Elektrotechnik) als Weg zur Erlangung wissenschaftlicher Erkenntnisse allgemein anerkannt ist, kann man bei Informatikern eher ein stark polarisiertes Verhalten beobachten: Scheint zum Nachweis einer Vermutung eine Ableitung aus mathematischen Axiomen möglich zu sein, so wird diese mit allen Mitteln der Kunst gesucht. Auf diese Weise ist man auf der sicheren Seite, die konsequent logische Vorgehensweise der Mathematik hat sich seit Jahrhunderten bewährt.

Läßt sich die Fragestellung jedoch nicht in mathematische Formeln gießen (wie es bei eher praktisch orientierten Subdisziplinen wie der Softwaretechnik aufgrund der starken interdisziplinären Verknüpfung in der Regel der Fall ist), wird in Zweifelsfällen auf die eigene Intuition vertraut. Es werden vermeintlich plausible Annahmen getroffen, ein (experimenteller) Nachweis ihrer Gültigkeit wird meist gar nicht erst versucht. Selbst wenn, wie im Beispiel des Knight/Leveson-Experiments, spätere Untersuchungen das Gegenteil beweisen, wird an der Gültigkeit der aufgebauten Theorie eisern festgehalten.

Bemerkenswert ist, daß selbst eindeutig widerlegbare Falschaussagen der Avizienis-Gruppe von den Lektoren wissenschaftlicher Fachverlage und Kongresse nicht zurückgewiesen wurden und ihren Weg in diverse Publikationen finden konnten. Dies mag ein Zeichen dafür sein, daß selbst hier das wissenschaftliche Experiment einen zu geringen Stellenwert besitzt.

Vielleicht können die im Seminar „Experimentelle Methoden in der Informatik“ gesammelten Aufsätze ihren Beitrag zur Bewußtseinsbildung hinsichtlich der Bedeutung experimenteller Verfahren in der Informatik leisten.

Literatur

[1] J. C. Knight und N. G. Leveson, *An Experimental Evaluation of the Assumption of Independence in Multiversion Programming,* IEEE Transactions on Software Engineering 12(1):96-109, Januar 1986

[2] J. C. Knight und N. G. Leveson, *A Reply to the Criticisms of the Knight & Leveson Experiment,* Software Engineering Notes 15(1):24-35, Januar 1990

[3] K. Hinderer, *Stochastik für Informatiker und Ingenieure*, Abschnitt 7.4 (Ausgabe 1991)

[4] P. M. Nagel und J. A. Skrivan, *Software Reliability: Repetitive run experimentation and modeling,* NASA Contractor Rep. CR-165836, Februar 1982

[5] J. R. Dunham, *Experimentations in software reliability: Life-critical applications,* IEEE Transactions on Software Engineering 12(1):110-123, Januar 1986

[6] M. K. Joseph, *Architectural Issues in Fault-Tolerant, Secure Computing Systems,* Ph.D. Dissertation, Dept. of Computer Science, UCLA, 1988

[7] A. Avizienis, *Software Fault Tolerance,* IFIP XI World Computer Congress '89, San Francisco, August 1989

[8] A. Avizienis, M. R. Lyu und W. Schutz, *In Search of Effective Diversity: A Six-Language Study of Fault-Tolerant Control Software,* Tech. Report CSD-870060, UCLA, November 1987

[9] A. Avizienis und M. R. Lyu, *On the Effectiveness of Multiversion Software in Digital Avionics,* AIAA/IEEE 8th Digital Avionics Systems Conference, San Jose, Oktober 1988, Seiten 422-427

Die Grundlagen wissenschaftlicher Arbeit

Michael A. Schmidt

Zusammenfassung

Was ist Wissenschaft? Der Artikel versucht eine Annäherung an die Frage aus zwei Richtungen. Zunächst wird die wissenschaftliche Methode anhand eines Fallbeispiels gegenüber anderen Methoden des Wissenserwerbs abgegrenzt. Der zweite Teil führt in die Grundbegriffe der Wissenschaftstheorie ein. Die Wissenschaftsmodelle Induktivismus und Falsifikationismus werden vorgestellt und die Probleme ihrer Reinformen werden erläutert. Mit den Wissenschaftprogrammen nach Lakatos und der von Kuhn geprägten Vorstellung von Wissenschaft als Paradigma, folgen zwei Modelle, die einen Teil dieser Probleme aufgreifen. Zum Abschluß erfolgt eine Einordnung in die Informatik.

1 Einleitung

Was ist Wissenschaft? Dieser Artikel will das Verständnis für die Frage schärfen. Verständnis setzt aber Wissen voraus. Deshalb beschäftigt sich ein Teil des Artikels mit wissenschaftstheoretischen Modellen. Diese Modelle versuchen, die Rollen von Beobachtungen, Experimenten, Hypothesen und Theorien zu klären. Dabei stehen empirische Wissenschaften im Vordergrund.

Bevor wir uns jedoch mit diesen Theorien beschäftigen, sprechen wir das intuitive Verständnis von Wissenschaft an. Dazu betrachten wir zunächst ein Beispiel:

In einem Artikel in der Los Angeles Times vom 9. Mai, 1982 behaupteten religiöse Gruppen, sie hätten auf Schallplatten von Rock-Bands Botschaften des Teufels gehört als sie diese rückwärts abspielten. Viele Befragte stützten diese Behauptung. Dies waren aufrichtige Einzelpersonen, die glaubten, die Botschaften gehört zu haben. In einigen Staaten der USA wurde daraufhin eine Kennzeichnungspflicht für diese Schallplatten erlassen. Das kam einer Anerkennung der Behauptung gleich.

Obwohl die betroffenen Rock-Bands die Vorwürfe zurückwiesen, blieben die Mitglieder der religiösen Gruppe blieben bei ihrer Behauptung, die Botschaften gehört zu haben.

In der oben beschriebenen Situation, erscheinen die Behauptungen der religiösen Gruppen unwissenschaftlich: Die Begründung dafür läuft unserer Vorstellung von Wissenschaft in dieser Form völlig zuwider, weil sie weder objektive Beobachtungen , Nachprüfbarkeit noch methodisches Vorgehen beinhaltet.

1.1 Nicht wissenschaftliche Methoden des Wissenserwerbs

Wissenschaft ist nicht die einzige Mehode zur Wissensgewinnung. Der Abschnitt stellt fünf dieser Methoden vor und setzt sie in Beziehung zu der wissenschaftlichen Methode.

1. *Beharrlichkeit*: Diese Methode beinhaltet Aberglaube und Gewöhnung. Anschauungen werden nicht hinterfragt, sondern als Tatsachen betrachtet. Gewöhnung bewirkt, daß Dinge und Sachverhalten, die wir oft wahrnehmen auch als wahr angenommen werden.
2. *Intuition*: Erkenntnis ergibt sich spontan und nicht etwa durch Nachdenken oder Ableitung . Intuition gestattet es nicht genaues Wissen von ungenauen Wissen zu trennen.
3. *Autorität*: Die Informationsquelle wird hoch eingeschätzt und respektiert, so daß Kritik gegenüber dem Wissen fehlt. Eine solche Informationsquelle kann beispielsweise ein geistlicher Würdenträger oder der Staat sein. Im oben genannten Artikel spielte Autorität eine Rolle, als einige Staaten eine Kennzeichnungsplicht einführten.
4. *Rationalismus*: Erkenntnis wird nur durch Überlegung gewonnen. So gewonnene Erkenntnis ist mindestens genauso wahr wie durch Wahrnehmung erzielte Erkenntnis.
5. *Empirismus*: Erkenntnis gewinnt man durch Wahrnehmung. Vermutungen können nur durch Wahrnehmungen abgesichert werden.

Auch wenn die beiden letztgenannten Methoden für sich genommen unwissenschaftlich sind, sind sie zusammen wesentliche Bestandteile der Wissenschaft. Hypothesen stellt man nach Überlegung auf, eas ein rationaler Vorgang ist. Grudlage dieser Überlegungen sind meist empirische Beobachtungen. Dies führt uns zum Induktivismus, der im nächsten Abschnitt behandelt wird.

2 Induktivismus

Kommen wie nun zu den wissenschaftstheoretischen Modellen. In der Reinform wie hier dargestellt, ist der *Induktivismus* leicht zu kritisieren. Es kommt aber darauf an Grundkonzepte vorzustellen und nicht vollständige Methoden für die wissenschaftliche Praxis zu liefern.

Der Induktivismus geht davon aus, daß Wissenschaft mit Beobachtung beginnt. Voraussetzungen sind gesunde Sinne und ein unvoreingenommener Beobachter. Unter solchen Voraussetzungen gemachte Beobachtungen sind eine tragfähige, objektive Grundlage für wissenschaftliche Erkenntnis. Idee des Induktivismus ist es, einzelne Beobachtungsaussagen zu sammeln und dann durch Überlegung induktiv auf allgemeine Sätze zu schließen. Es genügt aber nicht, sich nur auf wenige Beobachtungen zu stützen. Deshalb wird gefordert, daß Beobachtungen unter einer Vielfalt von Bedingungen wiederholt werden. So gelangt man zum allgemeinen *Induktionsprinzip*:

> „Wenn eine große Anzahl von As unter einer großen Vielfalt von Bedingungen beobachtet wird und wenn alle diese beobachteten As ohne Ausnahme die Eigenschaft B besitzen, dann besitzen alle As die Eigenschaft B."

Durch experimentieren kann der Induktivist gezielt die Bedingungen der Beobachtung variieren. So bekommt er systematisch eine Vielfalt von Beobachtungsaussagen, wie es das Induktionsprinzip fordert.

2.1 Erklärung und Vorhersage

Die Erklärung und Vorhersage von Phänomenen geschieht deduktiv, nachdem durch Induktion allgemeine Sätze gewonnen wurden. Dabei kommt logisches Schließen zur Anwendung. Als eine Prämisse treten die allgemeinen Sätze und Theorien auf. Als zweite Prämisse nimmt man die Anfangsbedingungen. Durch einen logischen Schluß kommt man zur Vorhersage. Beispiel:

1. Reines Wasser gefriert bei 0 Grad Celsius.
2. Der Kühler meines Autos enthält nahezu reines Wasser.

---

3. Wenn die Temperatur unter 0 Grad Celsius sinkt, dann gefriert das Wasser im Kühler meines Autos.

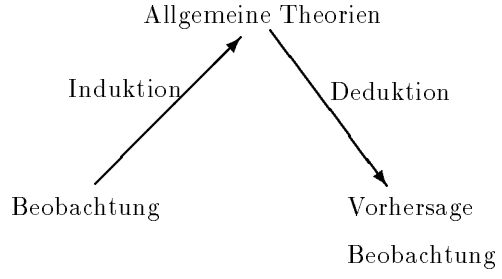


Abbildung 1: Schema des Induktivismus

2.2 Kritik am Induktivismus

Trotz seines formalen Ansatzes weist der Induktivismus einige Mängel auf, die ihn als Wissenschaftsmodell wenig geeignet erscheinen lassen.

Zum einen gibt es ein logisches Problem beim Schluß von einer endlichen Anzahl von Beobachtungen auf allgemeine Sätze, welche sich auf alle Situationen beziehen. Auch wenn n Beobachtungen gemacht werden, daß A die Eigenschaft B besitzt, kann nicht ausgeschlossen werden, daß A in der $n+1$-ten Beobachtung die Eigenschaft C hat. Beispiel: Es ist nicht ausgeschlossen, daß es auch schwarze Schwäne gibt.

Auch ein Rückzug auf Wahrscheinlichkeiten hilft hier nicht weiter. Man kann nicht sagen, eine Theorie wird wahrscheinlicher falls man mehr stützende Beobachtungen macht, also eine breitere Basis für die Induktion hat. Da die Menge der tatsächlich gemachten Beobachtungen endlich ist und einer unendlichen Menge von potentiellen Beobachtungen gegenübersteht, ist die Wahrscheinlichkeit jedes durch Induktion gewonnenen allgemeinen Satzes stets gleich Null.

Ein zweites Problem ergibt sich, falls man versucht das Induktionsprinzip aus der Erfahrung heraus zu beweisen. Dies könnte wie folgt aussehen:

> Das Induktionsprinzip war bei A gültig,
> das Induktionsprinzip war bei B gültig, usw.

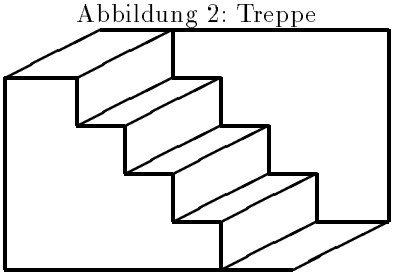
> Daraus folgt, daß das Induktionsprinzip immer gültig ist.

Man würde versuchen, das Induktionsprinzip durch sich selbst zu beweisen. Ein solcher Beweis ist nicht zulässig, da er auf einen Zirkelschluß beruht.

Weitere Argumente gegen den Induktivismus betreffen die Rolle der Beobachtung. Die Grundannahme, daß Beobachtung der Theorie vorausgeht, erweist sich als falsch. Wie ist es einem Beobachter möglich, unwichtige Aspekte wie etwa die Farbe seiner Schnürsenkel aus der Beobachtung auszuklammern? Der Beobachter hat ein Vorwissen, eine Theorie, die ihm bei diesen Entscheidungen hilft. Die Theorie geht der Beobachtung voraus und nicht umgekehrt.

Auch geht der Induktivismus davon aus, daß die Beobachtung eine objektive Grundlage für die Induktion liefert. Bild 2 zeigt für uns eine Projektion einer Treppe. Der Eindruck, ob wir die Treppe von oben oder von unten sehen, wechselt sprunghaft von Zeit zu Zeit. Der entscheidende Punkt ist , daß es Kulturkreise gibt, in denen es nicht üblich ist, zweidimensionale Abbildungen von räumlichen Gegenständen zu machen. Dort sehen die Menschen lediglich Strichmuster. Obwohl sie die gleichen Sinneseindrücke haben wie

Abbildung 2: Treppe



wir, kommen sie zu einer anderen Wahrnehmungserfahrung. Sie hängt vom Vorwissen, den Erwartungen und der Stimmung des einzelnen Beobachters ab. Aus diesen beiden Gründen, der Theoriegeleitetheit und der Subjektivität der Wahrnehmung, folgt, daß Beobachtung keine objektive Grundlage für Wissenschaft sein kann.

3 Falsifikationismus

Im vorangegangen Abschnitt haben wir gesehen, wie die Theorieabhängigkeit der Beobachtung dem Induktivisten ernsthafte Probleme bereitete. Der *Falsifikationismus*, ein anderes Wissenschaftsmodell, das maßgeblich von Karl Popper erarbeitet wurde, nimmt die Probleme des Induktivismus auf:

> „Der Falsifikationismus geht davon aus, daß Beobachtungen generell theoriegeleitet sind und damit Theorie voraussetzen. Gleichermaßen wird auf den Anspruch verzichtet, daß Theorien auf der Basis von Beobachtungen als wahr oder wahrscheinlich betrachtet werden können."

Wie aber soll bei dieser Einschränkung nun Wissenschaft noch möglich sein? Eine kurze Antwort gibt das folgende Zitat: "Es bleibt uns nichts anderes übrig, als Bestehendes dadurch schrittweise zu verbessern, daß wir aufzeigen, was daran falsch ist" [3]. Man will also aus Fehlern lernen. Einen Fehler hat man gemacht, wenn eine Hypothese als falsch nachgewiesen wird. Das kann beispielsweise durch eine Beobachtung geschehen, die der Hypothese widerspricht. Man sagt: „Die Hypothese wurde falsifiziert". Dabei sehen wir schon ein logisches Argument, das den Falsifikationismus unterstützt: Es reicht schon ein Gegenbeispiel aus, um eine Hypothese zu verwerfen. Es gibt also keine logischen Probleme mit der Bergründung der Methodik wie beim Induktivismus.

3.1 Wissenschaftlicher Fortschritt

Der angestrebte Fortschritt ergibt sich hierbei durch den Wettbewerb der Hypothesen untereinander. Falsifizierte Hypothesen scheiden aus. Hypothesen, die sich bewähren sind zwar nicht bewiesen, aber sie werden auf Zeit anerkannt. Werden solche Hypothesen dann doch als falsch nachgewiesen, so hat sich ein neues Problem ergeben, das auf einem höheren Niveau liegen sollte als das Alte. Bemerkenswert ist es, daß Falsifikationen, die Fehlschläge sein können, potentiell wertvolle wissenschaftliche Arbeiten sind.

Die oben vorgestellte Methode ist nur eine erste Annäherung. Tatsächlich ist die Qualität von Falsifikationen unterschiedlich. Falsifiziert man beispielsweise sehr kühne, unwahrscheinliche Hypothese, gewinnt man nicht so viel als würde man eine behutsame Hypothese falsifizieren. Behutsam und und kühn beziehen sich dabei jeweils auf das Hintergrundwissen, was die Gesamtheit aller anerkannten Hypothesen ist. Wenn eine kühne Hypothese sich bewährt, kann dies auch als Fortschritt betrachtet werden. Sie hat dann ein unerwartetes Ergebnis gebracht. Ein Beispiel für eine solche Bewährung war die Entdeckung der Radiowellen durch H. Hertz im Jahre 1886. Durch sie bestätigte Hertz die Maxwallschen Gleichungen, eine damals kühne Hypothese. Allerdings ist es heute keine Leistung mehr, Radio zu hören.

Eine weitere Möglichkeit die Wissenschaft im Sinne der Falsifikation voranzubringen, besteht darin, bekannte Hypothesen durch falsifizierbarere auszutauschen. Meist ist dies dann eine allgemeinere Hypothese. Beispielesweise bietet die Hypothese „Alle Planeten bewegen sich in geschlossenen Bahnen um ihre jeweilige Sonne" mehr Möglichkeiten zur konkreten Falsifikation als die Behauptung „Der Mars bewegt sich auf einer geschlossenen Bahn um die Sonne". Offensichtlich gewinnt man durch die Falsifikation der allgemeinen Hypothese mehr Erkenntnis.

3.2 Praktische Konsequenzen

Der Falsifikationismus hat durchaus praktische Konsequenzen. So folgt aus der Forderung nach falsifizierbaren Hypothesen etwa, daß diese möglichst eindeutig und präzise Formuliert zu formulieren sind. Vorallem die Präzision bezieht sich auf genaue quantitative Angaben. Es ist somit einfacher, Ansatzpunkte für mögliche Falsifikationen zu finden.

Eine weitere Richtlinie gibt es zur Frage der Modifikation von Hypothesen. Da Hypothesen nur durch falsifizierbarere ersetzt werden sollten, ist es im Fall einer Falsifikation nicht erlaubt, ad-hoc Modifikationen vorzunehmen. Solche ad-hoc Modifikationen fügen der

ursprünglichen Hypothese zur Abwehr der Falsifikation eine weitere Bedingung hinzu. Die Falsifizierbarkeit wird dadurch nicht verbessert.

3.3 Kritik am Falsifikationismus

Der Falsifikationismus besitzt ähnliche Probleme wie der Induktivismus. Der erste Kritikpunkt besteht im Vergleich der Falsifizierbarkeit zweier Hypothesen. Es gibt prinzipiell unendlich viele Beobachtungen, die eine Hypothese falsifizieren könnten. Damit läßt sich der Falsifizierbarkeitsgrad als rationales Entscheidungskriterium bei der Wahl von Hypothesen nicht objektiv begründen.

Ein weiteres Problem ergibt sich aus der wissenschaftlichen Praxis. Meistens sind die Hypothesen wesentlich komplexer als hier angedeutet. Außerdem werden sie oft von einer großen Anzahl von Hilfshypothesen begleitet. Diese beziehen sich etwa auf die Anwendbarkeit mathematischer Werkzeuge oder den Gebrauch von Gerätschaften. Im Fall einer Falsifikation, ergibt sich dann die Frage, welche von den vielen Hypothesen falsifiziert wurde. Hier leistet der Falsifikationismus keine Hilfestellung.

Wie wir schon im Induktivismus gesehen haben, ist die Wahrnehmung fehlbar. Dies wirkt sich jedoch nicht nur auf die Unbeweisbarkeit von Theorien aus, sondern auch darauf, daß sie im Grunde nicht endgültig falsifizierbar sind. Die Beobachtungen, die man bei der Falsifikation macht, können falsch sein. Damit wäre die Falsifikation hinfällig.

3.4 Theorienwechsel

Ein Wissenschaftsmodell sollte in der Lage sein, historische Situationen der Wissenschaft zu erklären. Insbesodere der Übergang von einer Theorie zu einer anderen verdient dabei Aufmerksamkeit. Gelingt es nicht einen historischen Theorienwechsel zu erklären, so kann man nicht erwarten, daß das Modell für gegenwärtige Entscheidungen hilfreich ist.

Bei strikter Anwendung eignet sich der falsifikationistische Ansatz nicht zur Erklärung von Theorienwechsel: Kommt eine neue Theorie auf, dann werden manche ihrer Hypothesen zuerst falsifiziert. Würde man nun radikal falsifikationistisch vorgehen, müßte man diese Theorie verwerfen. Betrachten wir als historisches Beispiel die kopernikanische Wende. Kopernikus behauptete unter anderem, daß sich die Erde um die eigen Achse dreht. Das stand im Gegensatz zum ptolemäischen Weltbild, das die Erde als fix annahm. Außerdem war zu jener Zeit noch die aristotelische Mechanik allgemein anerkannt. Nach ihr konnten Körper sich nur bewegen, falls eine Kraft in die entsprechende Richtung auf sie einwirken. So kam es zu folgendem Einwand gegen Kopernikus Theorie: Falls sich die Erde drehe, so müßte sich die Oberfläche in jedem Augenblick um eine beträchtliche Strecke fortbewegen. Nun läßt man einen Stein von der Spitze eines Turms herunterfallen. Da sich die Erde dreht, müßte der Stein in einiger Entfernung vom Fuß des Turmes aufschlagen. Das trifft nicht zu. Damit war die Theorie des Kopernikus falsifiziert und hätte gemaß den Forschriften des Falsifikationismus aufgegeben werden müssen. Ganz im gegensatz dazu brachten die Kopernikanischen Hypothese ganze *Forschungsprogramme* in der Mathematik und Physik in Gang.

4 Forschungsprogramme nach Lakatos

Bisher haben wir Wissenschaftsmodelle vorgestellt, die den lokalen Zusammenhang zwischen Theorien und Hypothesen beim Fortschritt der Wissenschaft erklärt haben. Dabei wurden die Theorien undifferenziert betrachtet. Das hatte unter anderem Probleme bei der Erklärung von Theorienwechseln mit sich gebracht. Die nun folgenden Ansätze stukturieren die Theorien, so daß feinere Mechanismen im wissenschaftlichen Fortschritt erklärt werden können.

4.1 Struktur der Forschungsprogramme

Die Forschungsprogramme von Lakatos könne als eine Fortführung des Falsifikationismus betrachtet werden. Wie bereits erläutert treten in realer Forschung Hypothesen stets zusammen mit einer Reihe von Hilfshypothesen auf. Diese Struktur wird nun präzisiert. Lakatos unterscheidet zwischen dem harten Kern eines Forschungsprogramms und dessen Schutzgürtel. Im harten Kern stehen die Grundannahmen und zentralen Hypothesen des Forschungsprogrammes. Im Schutzgürtel sind die Hilfshypothesen, Anfangsbedingungen, Werkzeuge mathematischer und technischer Natur sowie Techniken zur Anwendung der Werkzeuge enthalten.

Zu dieser Struktur gibt es noch eine positive und negative Heuristik, die die Aktivitäten in Forschungsprogrammen regeln. Die negative Heuristik besagt, daß der harte Kern unveränderbar ist. Das ist eine methodologische Entscheidung der Wissenschaftler des Forschungsprogrammes. Wissenschaftler, die sich nicht an diese Entscheidung halten schließen sich aus dem Forschungsprogramm aus. Die positive Heuristik besagt, daß der Schutzgürtel verändert werden darf. Man kann

Hilfshypothesen hinzufügen oder abändern. Tritt eine Falsifikation einer Hypothese plus Hilfshypothesen auf, so wird diese zunächst in den Schutzgürtel gelenkt. Das bedeutet, daß der Schutzgürtel so modifiziert wird, daß die Falsifikation nicht mehr auftritt. Diese beiden Heuristiken ermöglichen ein gezielteres Vorgehen als der reine Falsifikationismus. In diesem Sinne sind Forschungsprogramme eine Weiterentwicklung des Falsifikationismus.

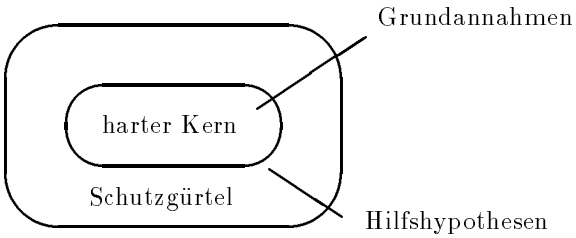


Abbildung 3: Forschungsprogramm-Struktur

4.2 Entwicklung von Forschungsprogrammen

Forschungsprogramme durchlaufen laut LAKATOS im allgemeinen zwei Phasen:

1. In ihrer progressiven Phase führen ihre Forschungsaktivität en zur Entdeckung neuer Phänomene.
2. In ihrer degenerativen Phase hingegen werden keine neuen Phänomene mehr entdeckt.

Die Entwicklung eines Forschungsprogrammes vollzieht sich wie folgt: Zu Beginn stehen die Grundannahmen alleine im Kern, der Schutzgürtel ist fast leer. Es kommt daher zu vielen Falsifikationen. Diese bewirken aber nicht das Verwerfen der Grundannahmen, diese sind durch die negative Heuristik geschützt, sondern sie führen erst zum Aufbau des Schutzgürtels. Es werden mathematische Hilfsmittel und Gerätschaften für Experimente entwickelt. Nun tritt das Forschungsprogramm in seine progressive Phase. Mit Hilfe der Werkzeuge des Schutzgürtels werden neue Phänomene entdeckt. Nach einer gewissen Zeit sinkt die Entdeckungsrate stark ab und das Forschungsprogramm tritt in seine degenerative Phase ein. Es wird dann meist von einem progressiveren Pendant abgelöst.

4.3 Kritik den Forschungsprogramme nach Lakatos

Auch hier stellt sich die Frage der Vergleichbarkeit von Forschungsprogrammen. LAKATOS versucht mit seinem Ansatz, Forschungsprogramme in eine Hierachie bezüglich ihrer Progressivität einzuordnen. Das zentrale Entscheidungsproblem ist die Frage wann ein Programm degenerativ ist. Ein solches Programm kann nämlich durch eine Änderung seines Schutzgürtels durchaus wieder in eine progressive Phase eintreten. Dies ist also kein vollständig objektives Kriterium, um sich zwischen unterschiedlichen Programmen zu entscheiden.

5 Kuhns Paradigmen

Die Forschungsprogramme aus dem vorigen Abschnitt kann man sich gut als kleine Forschungsvorhaben vorstellen, an denen nur wenige Wissenschaftler beteiligt sind. Das nun folgende Wissenschaftsmodell vertritt eine globalere Sicht auf die Wissenschaft und bezieht eine größere wissenschaftliche Gesellschaft mit ein.

5.1 Struktur der Paradigmen

Das *Paradigma* ist bei KUHN der zentrale Begriff. Es steuert die Aktivitäten in der jeweiligen Wissenschaft und strukturiert die Theorien. Den einzelnen Wissenschaftsrichtungen werden jeweils unterschiedliche Paradigmen zugeordnet. Paradigmen umfassen dabei:

1. allgemeine theoretische Annahmen und Gesetze,
2. mathematische und reale Werkzeuge,
3. Techniken zur Anwendung der Gesetze und
4. allgemeine methodologische Vorschriften. „Das Paradigma muß der Realität angepaßt werden“, „Fehlgeschlagene Versuche, ein Paradigma an die Realität anzupassen, müssen als ernsthaftes Problem betrachtet werden“ [1]
5. „stillgeschwiegenes Wissen“. Dies ist meist fachspezifische Methodik, die in der Ausbildung des Wissenschaftlers durch praktische Übung vermittelt wird.

Die ersten Drei Punkte entsprechen etwa dem harten Kern und dem Schutzgürtel der Forschungsprogramme und geben somit die Struktur vor.

5.2 Entwicklung von Paradigmen

Wissenschaft entwickelt sich nach KUHN in etwa wie folgt:

– Vor-Wissenschaft – normale Wissenschaft – Krise – Revolution – Neue Normalwissenschaft – Neue Krise –

Diesen Stadien besitzen jeweils eine Orientierungsfunktion für die Wissenschaftler. In einer vorwissenschaftlichen Situation gibt es meist Debatten über die Grundannahmen des Paradigmas. Die nötigen Werkzeuge stehen noch nicht zur Verfügung. Ein normales, fachwissenschaftliches Arbeiten ist nicht möglich.

In der Normalwissenschaft wird gemäß der allgemeinen methodologischen Vorschriften an der Anpassung des Paradigmas an die Realität und an dessen Konkretisierung gearbeitet. Der Wissenschaftler kann sich auf Details konzentrieren, ohne die Grundannahmen zu hinterfragen. Die Probleme, die dabei entstehen, haben nach KUHN den Charakter des Rätsel Lösens. Diese Rätsel können sowohl theoretischer als auch experimenteller Natur sein. Gelingt es nicht, ein Rätsel zu lösen, so bedeutet das lediglich, daß noch offene Probleme existieren. Die Grundannahmen bleiben vorerst unverändert. Dies ist der Schutzmechanismus des Paradigmas gegenüber Falsifikationen. Erst wenn es sich hartnäckig allen Lösungsversuchen widersetzt, bezeichnet man es als Anomalie des Paradigmas. Treten verstärkt Anomalien auf so gerät das Paradigma in eine Krise. Dann wird der methodologische Zwang gelockert. Es kommt zu immer kühneren Lösungsversuchen, aus denen oft neue, rivalisierende Paradigmen entstehen. Jetzt wird eine Abgrenzung versucht und dazu werden stillgeschwiegenes Wissen und methodologische Grundlagen formuliert.

Schließlich kommt es zu einem sprunghaften Wechsel der Wissenschaftler von einem Paradigma zu einem neuen. Dieser Vorgang ist einer Revolution vergleichbar, weil quasi aus einm alten Paradigma ausgebrochen wird.

5.3 Kritik an Kuhns Paradigmen

Wie auch in vorhergehenden Ansätzen gibt es keine objektiven Kriterien, die dem Wissenschaftler helfen eine Wahl zu treffe. Meist sind bei rivalisierenden Paradigmen ihre Grundannahmen so verschieden, daß sie sich nicht mehr vergleichen lassen.

5.4 Kuhns Paradigmen in der Informatik

Es soll kurz versucht werden, KUHNS Paradigmen auf die Situation in der Informatik anzuwenden. Meiner Ansicht nach stellt sich die Situation recht uneinheitlich dar. Die Normalwissenschaft ist voll im Gang, was die breite Anwendung der Informationstechnik zeigt. Gleichzeitig aber eine Debatte über ihre intellektuelle Substanz geführt [4]. Sie wird vorallem durch den Versuch der Standortbestimmung gegenüber anderen Paradigmen geprägt. Dies sind erstens die Mathematik als Formalwissenschaft, die sich in der Theorie der Informatik wiederfindet. Zweitens die empirischen Wissenschaft, die in der Abstraktion zum Ausdruck kommt. Hier werden Modelle mit Hilfe der formalen Methoden der Theorie entwickelt und auf Tauglichkeit mit empirischen Mitteln überpüft. Drittens die Ingenierswissenschafften, die sich im Design konkreter Systeme ergeben. Die wissenschaftstheoretischen Modelle, die wir vorgestellt haben, besiten voralem in der Abstraktion ihre Gültigkeit. Tabelle 4 zeigt die drei Komponenten mit einigen Beispielen aus der Softwaretechnik.

Theorie	Programmverifikation Wahrscheinlichkeitstheorie Prädikatenlogik Kognitive Psychologie
Abstraktion	Spezifikationsmrthoden N-Versionsprogrammierung Module Lebenszyklusmodelle
Design	Spezifikationssprachen Softwarewerkzeuge Schnittstellenentwurf

Abbildung 4: Komponenten der Informatik am Beispiel der Softwaretechnik

6 Schlußbemerkung

Wir haben gesehen, daß inituitive Vorstellungen von Wissenschaft vor allem in Hinblick auf die Objektivität nicht zu halten sind. Eines der Kernprobleme ist dabei die Erfahrungsabhängigkeit unserer Wahrnehmung. Mit dem Falsifikationismus haben wir eine Methode kennengelernt, die versch Bestehendes dadurch schrittweise zu verbessern, daß sie aufzeigen, was daran falsch ist. Die beiden Ansätze von LAKATOS und KUHN strukturieren die Theorien und versuchen so, die komplexen Zusammenhänge ,die der realen Wissenschaft zugrundeligen, zu erklären. Letzten Abschnitt haben wir gesehen, daß dir Informatik auch einen empirischen Zug trägt. Damit ist es für den Informatiker genauso wichtig wie für den Physiker, die Grundlagen empirischer Wissenschaften und deren Probleme zu verstehen.

Literatur

[1] Chalmers, A. F. *Wege der Wissenschaft.* Springer-Verlag Berlin, 1994.

[2] Christensen *Experimental Methodology*. Allan and Bacon, Needham Heights, MA, 1994.

[3] Anzenbacher, Arno *Einführung in die Philosophie*. Herder, 1981.

[4] Comer, Gries, Mulder, Tucker, Turner, Young *Computing As A Discipline*. Communications of the ACM, Vol. 32 No. 1, Jan. 1989.

Grundlagen der Wissenschaftstheorie

Ralf H. Reussner

Zusammenfassung

Die Aufgaben der Wissenschaftstheorie werden anhand zentraler Fragen erläutert. Diese Fragen geben ein Gerüst zur Einordnung der verschiedenen Ansätze von Lakatos, Kuhn und Feyerabend. Bezüge zwischen den Standpunkten werden erläutert, und es wird versucht, eine einheitliche Sicht aufzuzeigen. Die Informatik soll darin ihren Platz zwischen anderen Wissenschaften finden.

1 Einleitung

Wissenschaftstheorie beschäftigt sich mit den Grundlagen jeder Einzeldisziplin: sie beschreibt die Aufgaben und Ziele der Wissenschaft und die Methoden diese zu erreichen. Gibt es Fragen, die jeden Wissenschaftler, unabhängig vom jeweiligen Fach, betreffen? Warum ist Wissenschaftstheorie interessant? Was ist Wissenschaft überhaupt? Was unterscheidet die Wissenschaftliche Methode eigentlich von anderen Möglichkeiten des Erkenntnisgewinns? Gibt es diese anderen Möglichkeiten überhaupt?
Zumindest eine Frage kann hiermit als positiv beantwortet werden: Diese Fragen sollten wohl für alle Wissenschaftler interessant sein!
Inwieweit die Wissenschaftstheorie die anderen Fragen beantworten kann, und welche Ansätze es dazu gibt soll im Folgenden erläutert werden. Außerdem soll die Informatik als Wissenschaft unter den anderen eingeordnet werden.

2 Zentrale Fragen der Wissenschaftstheorie

Die Darstellung der verschiedenen Positionen folgt [CHA_94], Kap. 9-11,13.

2.1 Die Frage nach der Methodik: Rationalismus versus Relativismus

Diese Frage betrifft die sog. *Wissenschaftliche Methode*, also eine konkrete Anleitung, um wissenschaftliche Erkenntnis zu gewinnen. Hier wird nicht auf verschiedene methodische Ansätze eingegangen, sondern vielmehr sollen hier die beiden Extrempositionen bezüglich der *Beantwortbarkeit* dieser Frage behandelt werden. Die Frage nach der Existenz allgemeingültiger Maßstäbe zur Beurteilung wissenschaftlicher Arbeit beantwortet der *wissenschaftstheoretische Rationalist* im Gegensatz zum *Relativisten* positiv.[7]
Für den Rationalisten sind Maßstäbe des wissenschaftlichen Arbeitens aus verschiedenen Gründen unerläßlich:

- Nur so kann eine Definition und Abgrenzung der Wissenschaft gegeben werden. Nur Erkenntnisgewinn in Übereinstimmung mit diesen Methoden ist Wissenschaft.
- Durch Definition der sog. Wissenschaftlichen Methode werden dem forschenden Wissenschaftler konkrete Handlungsregeln und Modelle vorgegeben.
- Unter anderem sollen dem Wissenschaftler durch die Wissenschaftliche Methode Kriterien an die Hand gegeben werden, die ihm das Auswählen einer Theorie aus mehreren konkurrierenden ermöglichen. (*Theorienauswahl*)
- Dem Wissenschaftshistoriker wird damit ein Modell gegeben, um den Wechsel von Theorien und Paradigmen in der Wissenschaftsgeschichte zu verstehen. Dieser sog. *Theorienwechsel* bezieht sich im Gegensatz zur o.a. Theorienauswahl auf die historische Entwicklung der Wissenschaft.

Die Unterscheidung zwischen Theorienauswahl und -wechsel erscheint vielleicht zunächst wenig sinnvoll. Zu bedenken ist aber, daß die Maßstäbe historischer Theorienwechsel nicht notwendigerweise dieselben sein müssen, nach denen ein heutiger Wissenschaftler seine Theorien auswählt. Schließlich gibt es auch Theorienwechsel in der Wissenschaftstheorie.
Wurde Erkenntnis nicht mit der Wissenschaftlichen Methode gewonnen, so sollte ihr nicht das Vertrauen entgegengebracht werden, welches üblicherweise wissenschaftlicher Erkenntnis zugedacht wird. (Die Frage, ob unwissenschaftlich gewonnene Erkenntnis überhaupt als solche zu bezeichnen ist, hängt natürlich von der Definition des Wortes Erkenntnis ab. Es ist aber denkbar, Erkenntnis (im Gegensatz zu *Wahrheit*) nur auf wissenschaftliche Ergebnisse zu beziehen.)

[7] Die Position des wissenschaftstheoretischen Rationalisten ist nicht zu verwechseln mit der des *erkenntnistheoretischen* Rationalisten. Dem zweiten ist die allgemein unter des Namen Rationalismus bekannte Haltung zu eigen, die ausgehend von Descartes (1596-1650) besagt, daß Erkenntnis nur durch Nachdenken gewonnen werden kann, Sinneseindrücke unterliegen zu sehr Täuschungen. (Ursprünge bei Platons Ideenlehre.) Die Position des wissenschaftstheoretischen Rationalisten hängt zunächst nicht mit dieser erkenntnistheoretischen Position zusammen, und wird im Folgenden einfach nur Rationalismus genannt.

Der Grund für das öffentliche Vertrauen gegenüber wissenschaftlich gewonnener Erkenntnis ist nur durch das Einhalten dieser Methode zu rechtfertigen. Dennoch darf der Wissenschaftler den Ergebnissen bisheriger wissenschaftlicher Arbeit kein absolutes Vertrauen entgegenbringen, sondern muß sie vielmehr kritisch gerade diesen Maßstäben unterziehen. Diese Haltung impliziert allerdings weder bestimmte Ansichten über die Existenz einer objektiven Welt, noch über die Aufgaben von Wissenschaft selbst.
Demgegenüber verneint der Relativist die Existenz solch allgemeiner Maßstäbe. Wissenschaft ist für ihn ein viel zu kompliziertes Netzwerk aus Interessen einzelner Personen und der jeweiligen ihn fördernden Gesellschaft, als daß allgemeingültige Maßstäbe angegeben werden könnten. Diese Einflüsse auf Wissenschaftler sind weltweit und über die Zeit betrachtet zu vielfältig, als jemals in ein einheitliches Modell eingebracht werden zu können. Damit überläßt der Relativist die Frage nach der Art der Wissenschaft weitgehend der subjektiven Empfindung.
Es sind zwei Hauptströmungen zu unterscheiden:

„Individueller Relativismus“
: Zurückgehend auf den Ausspruch des Protagoras (480 - 410 v. Chr.) *„Der Mensch ist das Maß aller Dinge“* nehmen die Vertreter dieser Strömung an, daß es im wesentlichen von den Entscheidungen Einzelner abhängig ist, was Wissenschaft ist, und wie ihre Weiterentwicklung beeinflußt wird.

„Soziologischer Relativismus“ Das Handeln der Wissenschaftler wird in erster Linie von der sie umgebenden Gesellschaft bestimmt, da sie dem modernen Wissenschaftler erst das Arbeiten ermöglicht. Unabhängig davon, inwieweit sich der einzelne Wissenschaftler dieses Einflusses bewußt ist, wird so Zweck und Art der Forschung bestimmt.

Insgesamt kann die Wissenschaftstheorie also nicht alle Ursachen für Theorienwechsel angeben, weil auch psychologische bzw. soziologische Faktoren eine entscheidende Rolle spielen.

2.2 Die Frage nach der Aufgabe der Wissenschaft: Realismus versus Instrumentalismus

Die bisherige Betrachtung beantwortet nicht die Frage nach den Aufgaben der Wissenschaft. Selbst der Erkenntnisgewinn ist als vornehmlicher Sinn der Wissenschaft keineswegs unumstritten. (Es sei beispielsweise auf den Positivismusstreit der 60er Jahre verwiesen (Habermas, Adorno, Albert), zurückgehend auf die Idee der *Kritischen Theorie* von Horkheimer (1937).)
Doch selbst wenn dem Erkenntnisgewinn Priorität eingeräumt wird, gibt es ein breites Spektrum an verschiedenen Positionen über die Aufgaben der Wissenschaft. Die beiden Pole markiert der *Realismus* einerseits und der *Instrumentalismus* andererseits.
Für den Realisten bedeutet wissenschaftliche Arbeit Wahrheitssuche: also die Suche nach der wahren Beschreibung der uns umgebenden Welt. Die in den Theorien vorkommenden Begriffe besitzen hier Pendants in der realen Welt *(Korrespondenztheorie der Wahrheit)*. Dazu muß der Realist zwei Annahmen machen, ohne die sein Standpunkt keinen Sinn hätte:

Voraussetzung des Begriffs der Wahrheit:
: Der Realist definiert Wahrheit als fehlerfreie Charakterisierung der realen Welt, die er als existent annehmen muß.

Voraussetzung einer unabhängigen Welt:
: Für den Realisten gibt es eine unabhängige, *reale* Welt, und diese muß (zumindest teilweise) erkennbar sein. Für ihn existiert Erkenntnis demnach unabhängig vom Menschen.

Diese Annahmen scheinen intuitiv gerechtfertigt zu sein. Um zu erläutern, wieso sie explizit erwähnt werden, soll nun die Korrespondenztheorie der Wahrheit erläutert werden. Auf ihr basieren der Realismus und der später beschriebene Objektivismus.
Die zentrale Aussage der Korrespondenztheorie ist die Definition von Wahrheit als *Übereinstimmung mit der Wirklichkeit*. Dieser Wahrheitsbegriff ist allerdings keineswegs so unproblematisch, wie seine intuitive Definition erwarten läßt. Einige der wichtigsten Kritikpunkte lauten:

- Bei der naiven Definition von Wahrheit in einer nicht formalen Gebrauchssprache treten sofort logische „Katastrophen“ der Bauart: *„Dieser Satz ist nicht wahr.“* auf. Ein Ausweg ist die auf den Logiker *Alfred Tarski* (1936) zurückgehende Trennung zwischen *Objekt-* und *Metasprache*.[TAR_36] Die Metasprache beschreibt Aussagen, die in der Objektsprache abgefaßt sind. Diese Trennung ist uns beispielsweise aus der formalen Logik bekannt, um selbstbezügliche Aussagen wie die erwähnte zu vermeiden. Begriffe zur Beschreibung von objektsprachlichen Aussagen zählen zur Metasprache. Als eingängiges Beispiel mag die Semantik von Formalen Sprachen dienen: Die Bedeutung von reservierten Wörtern oder Kontrollkonstrukten einer Programmiersprache kann nicht allein in derselben Sprache definiert werden, der sie angehören. Die Metasprache ist also in diesem Sinne mächtiger als die Objektsprache. (Die Begriffe *Formale Sprache* und *Semantik* gehen ebenfalls auf Tarski zuruck.)

Demzufolge gehört auch der Begriff der *Wahrheit* der Metasprache an. Weiterhin konnte Tarski zeigen, daß Begriffe der Metasprache nicht sinnvoll in der Objektsprache definiert werden können.
Die gängige Sprache, in der Aussagen abgefaßt werden, ist die Umgangsprache. Es folgt, daß Wahrheit in der Umgangssprache nicht sinnvoll definiert werden kann, dazu benötigten wir eine Metasprache um die Umgangssprache herum, um eben über diese umgangssprachlichen Aussagen zu reden. Diese Metasprache (für eine Umgangssprache) gibt es nicht, und kann auch nicht geschaffen werden, denn in welcher Sprache sollte sie abgefaßt sein? Denn um die Bedeutung umgangssprachlicher Aussagen festzulegen, muß wieder die Umgangssprache bemüht werden. Die Trennung zwischen Objekt- und Metasprache kann also im Alltag nicht durchgeführt werden.
Als Ausweg erscheint die Möglichkeit, Aussagen eben nicht in der Umgangssprache auszudrücken, sondern in einer formalen Sprache, was in der Informatik ja auch geschieht: Algorithmen werden in Formalen Sprachen notiert, damit sie als objektive Problemlösung existieren. Inwieweit diese Formalen Sprachen aber geeignet sind, als Ausdrucksmittel der Mehrzahl der Wissenschaftler und der Vielzahl wissenschaftlicher Betätigungsfelder zu genügen, bleibt hier eine offene Frage.
Allgemeiner mag gelten, daß Aussagen bestimmter Theorien eben nur in der Terminologie dieser Theorien abgefaßt werden können. Das aber entspricht keiner Trennung der Sprachebenen.
Damit bleibt der Begriff der Wahrheit in der naiven Verwendung problematisch.

- Ein ebenfalls auf die Sprache bezogener prominenter Kritikpunkt kommt aus der Richtung des logischen Positivismus: Wittgenstein (1945) zeigt in seinen *Philosophischen Untersuchungen* am Beispiel des Begriffes *Spiel*, daß sich umgangssprachliche Begriffe einer exakten Definition, die hinreichend und notwendig zugleich ist, entziehen. Wittgenstein begründet darauf seine radikale Sprachkritik.[WIT_45]

- Ein anderes Problem der Verwendung der Korrespondenztheorie ergibt sich aus der Anwendung der experimentellen Methode: Experimente sind keine alltäglichen Situationen, sondern im Hinblick auf bestimmte Theorien konstruierte, künstliche Anordnungen. Insofern kann eingewendet werden, daß so gewonnene Erkenntnis auch konstruiert ist und sich nicht unbedingt auf allgemeinere Gegebenheiten beziehen muß. Weiterhin gehen wissenschaftliche Theorien ja gerade über „unmittelbar“ beobachtbare Ereignisse hinaus. (Daß aber der Begriff der *unmittelbaren Beobachtung* ohnehin problematisch ist, zeigt die Existenz diverser Sinnestäuschungen. Zudem ist die Unmittelbarkeit physikalischer Beobachtung bei modernen Meßanlagen ohnehin nicht gegeben.) Wie steht es dann mit der Existenz theoretischer Begriffe?
Daß dieser Einwand keineswegs artifiziell ist, zeigt z.B. die Formulierung eines Sachverhalts durch zwei äquivalente Begriffsbildungen: In der Quantenmechanik erwies sich die Heisenbergsche Matrizenmechanik mit der Schrödingerschen Wellenmechanik als gleichwertig. Aber welche der Begriffsbildungen soll nun Anspruch auf reale Existenz erheben?
In der Geschichte der Erklärungen des Phänomens Licht gibt es wechselnde Positionen bezüglich des Teilchen- oder Wellencharakters des Lichts. Gerade der Welle-Teilchen-Dualismus zeigt in seinem außerhalb der Mathematik schwer vorstellbaren Charakter die ganze Problematik des Korrespondenzbegriffes: hier wird eher die Modellhaftigkeit moderner physikalischer Beschreibungen deutlich.

- Ein darauf basierender Kritikpunkt ist die Veränderlichkeit wissenschaftlicher Theorien. Als menschliches Produkt sind sie immer wieder Veränderungen unterworfen. Die reale Welt, die sie beschreiben sollen, unterliegt zumindest nicht denselben Veränderungen, so daß eine direkte Korrespondenz zwischen realer Welt und Theorie zweifelhaft ist.

Dem Instrumentalisten stellen sich diese Probleme nicht, da er auf die Korrespondenztheorie verzichten kann. Sein Ziel der Wissenschaft ist das Bereitstellen theoretischer Konstrukte, die Verknüpfungen zwischen Beobachtungen zulassen. Diese Konstrukte sollen Anwendungen des Wissens ermöglichen. Eine reale Existenz der theoretisch postulierten Begriffe ist nicht nur unwahrscheinlich, sondern auch sinnlos. Beschreibungen sollen sich nur auf beobachtbare Ereignisse beziehen, alle theoretischen Konstrukte haben nichts mit der Realität zu tun. Es reicht aus, Gesetzmäßigkeiten festzustellen und zu beschreiben. Real existente Ursache zu suchen, wird nicht als sinnvoll erachtet, da die Kluft zwischen Wahrnehmbaren und Nicht-Wahrnehmbaren durch die Wissenschaft ohnehin nicht als überbrückbar angenommen wird.
Die Beschränkung des Existenzgedankens auf beobachtbare Ereignisse erinnert etwas an den Positivismus, und in der Tat kann ein Standardargument gegen den Positivismus auch hier angebracht werden: Alle Beobachtung ist theorieabhängig und damit doch beeinflußt von unseren Gedanken, die ja per definitionem nichts mit der beobachtbaren Realität zu tun haben durften.
Der Instrumentalist benutzt zwar Theorien zur Lösung anwendungsbezogener Probleme und ist damit auf Vor-

hersagen seiner Theorien angewiesen, die reale Existenz der zu diesen Vorhersagen beteiligten theoretischen Konstrukte bezweifelt er; folglich muß er sich doch sehr wundern, wenn z.B. Moleküle „sichtbar" gemacht werden, oder Elektronen in Paul-Fallen direkt nachgewiesen werden.

2.3 Die Frage nach der Erkenntnis: Objektivismus versus Individualismus

Im vorigen Abschnitt ist die Frage nach der Existenzform der Erkenntnis angerissen worden; konkret ausformuliert: Existiert Erkenntnis unabhängig vom (erkennenden) Menschen, oder ist sie ein allein geistiges Produkt in unseren Köpfen? Die erste Position wird vom *Objektivisten* eingenommen der seine Haltung durch folgende Argumente begründet:

- Es gibt objektive Eigenschaften von Dingen unabhängig davon, ob wir uns dieser Qualitäten bewußt sind oder nicht. Selbst wenn alle Menschen blind wären und so direkt keine Farben wahrnähmen, so behielten die Gegenstände doch die Eigenschaft, Licht in verschiedenen Wellenlängen zu reflektieren. Umgekehrt kann auch eine Aussage falsch sein, selbst wenn sie allen als plausibel erscheint.
- In der Wissenschaftsgeschichte gibt es viele Beispiele dafür, daß die „Entdecker" von Theorien sich der Konsequenzen „ihrer" Theorien nicht bewußt waren. Beispielsweise führte Planck 1900 das später nach ihm benannte Wirkungsquantum ein. Dies bildete die Grundlage der von Heisenberg, Schrödinger, Dirac und vielen anderen ausgearbeiteten Quantenmechanik. Ein wesentliches Postulat darin sind die von Heisenberg formulierten Unschärferelationen, die einen vollkommenen Determinismus in der Physik ersetzen. Dennoch hielt Planck an einer streng deterministsichen Sichtweise der Physik fest. (s. [PLA_37])
  Die objektiven Entwicklungsmöglichkeiten einer Theorie werden also oft nicht vom Entdecker selbst ausgeschöpft, sondern von anderen Forschern. Popper vergleicht die Entwicklungsmöglichkeiten einer Theorie mit *Nistkästen* für Vögel. Diese Nistkästen können genutzt werden oder nicht. Sie existieren aber unabhängig von der Nutzung.

Grundsätzlich setzt der hier geschilderte Objektivismus die Existenz einer realen Welt voraus, zumindest sofern die objektive Erkenntnis sich auf die Natur bezieht. Insofern muß sich der Objektivist mit den Einwänden gegen die Korrespondenztheorie auseinandersetzen.

Dem *Individualisten* stellen sich diese Probleme nicht, da er die Korrepondenztheorie nicht voraussetzen muß. Für den wissenschaftstheoretischen Individualisten existiert Erkenntnis nur als Überzeugung einzelner. [8] Eine Überzeugung gilt dann als wahre Erkenntnis, wenn ausreichend Beweismaterial vorliegt. Wobei *ausreichend* und *Beweismaterial* eigentlich erst noch definiert werden müßten, was aber dann leicht wieder einen „realistischen Touch" bekommen könnte. Dennoch ist dieser Ansatz für Informatiker interessant, da er gewisse Übereinstimmung mit der sog. *Konstruktiven Mathematik / Logik* zeigt.[TaD_88] In diesem logischen System sind nur konstruktive Beweise zugelassen. Die Existenz eines Objektes mit bestimmten Eigenschaften kann nur durch Angabe eine Algorithmus' beweisen werden, der die Berechnung des Objektes zuläßt. So entsteht eine Korrespondenz *(Curry-Howard-Isomorphismus)* zwischen der Prädikatenlogik n-ter Stufe (Typentheorie) und dem lambda-Kalkül respektive dem Funktionalen Programmieren.[THO_91] Es ist leicht erkennbar, daß die Idee des ausreichenden Beweismaterials ein gewisses Axiomensystem benötigt, denn in irgendeiner Form muß ja ein Anfangspunkt gegeben sein, von dem aus erst bewiesen werden kann. Allerdings ist die Annahme eines a-priori-Wissens in den Naturwissenschaften nicht unproblematisch. Selbst so plausibles a-priori-Wissen wie die Konstanz von Raum und Zeit ist schon in der Speziellen Relativitätstheorie nicht mehr zu finden. Auch ist das Kausalgesetz seit den Heisenbergschen Unschärferelationen uninteressant geworden, da die Prämissen gewisser Wenn-dann-Aussagen einfach nicht mehr mit ausreichender Exaktheit zu quantifizieren sind.(s. [HEI_28] S. 28, oder auch S. 36 ebenda.)

3 Einordnung bestehender Ansätze

In diesem Abschnitt sollen die als bekannt angenommenen Ansätze von Lakatos und Kuhn in die beschriebenen Positionen eingeordnet werden. Weiterhin wird die Theorie von Feyerabend vorgestellt und ebenfalls klassifiziert.

3.1 Lakatos'sche Forschungsprogramme

Lakatos betont deutlich eine rationalistische Position. Für ihn ist das Hauptproblem der Wissenschaftstheorie *„... das Problem der Aufstellung* allgemeiner *Bedingungen für die Wissenschaftlichkeit einer Theorie."*

[8] Diese Position hat mit der herkömmlichen Bedeutung des Wortes Individualist nur wenig zu tun.

Weiterhin: Daß die *„Lösung uns einen Leitfaden dafür in die Hand geben (sollte), wann die Anerkennung einer wissenschaftlichen Theorie vernünftig ist und wann nicht.“* (vgl. [LAK_82b], S. 182f.) Dementsprechend lehnt er einen relativistischen Standpunkt ab: *„Wenn man ... eine Theorie nur aufgrund der Anzahl, des Glaubens und der Lautstärke ihrer Anhänger beurteilen kann“*, dann läge *“Wahrheit ... in der Macht.“* (vgl. [LAK_74], S. 172)
Lakatos gibt aber nicht dogmatisch ein bestimmtes Kriterium an, dem eine als wissenschaftlich zu bezeichnende Theorie genügen muß, vielmehr stellt er einen Maßstab auf, dem sich konkret ein Maßstab zur Theorienbeurteilung stellen muß. Diesem Maßstab entspricht Lakatos' *Methodologie*. Lakatos fordert, daß eine vorgeschlagene Methodologie mit der Wissenschaftsgeschichte konfrontiert werden soll. Die zur Debatte stehende Methodologie kann nur so gut sein, inwieweit sie in der Lage ist, Theorienwechsel guter Wissenschaft zu erklären. *Gute Wissenschaft* ist für Lakatos in erster Linie die Physik, also muß eine Methodologie Theorienwechsel der Physik erklären. Die Enge dieses Standpunkts kann als Ansatzpunkt der Kritik dienen.
Dementsprechend nennt Lakatos ein gewisses Forschungsprogramm besser als ein dazu konkurriendes, wenn es *progressiver* ist. Dabei erläutert er den Begriff „Progressivität“ eines Programms nur mit dem Maß von neuartigen Voraussagen dieses Programmes. Diese neuartigen Voraussagen sind für Lakatos der Weg des wissenschaftlichen Fortschritts, also das *Ziel* aller Forschungsprogramme. Die *Mittel* dieses Ziel zu erreichen sind die oben erwähnten Methodologien.
Diese konkrete Zielvorgabe für Wissenschaft kennzeichnet Lakatos auch als Realisten: das Ziel von Wissenschaft ist Wahrheit. Und somit ist Lakatos auch als Objektivist zu bezeichnen, denn sein Ziel von Wissenschaft läßt sich nicht ohne die Annahme einer objektiven Welt sinnvoll erreichen. Existenz besteht unabhängig von erkennenden Subjekt: *„Doch der objektive wissenschaftliche Wert einer Theorie ist unabhängig vom menschlichen Bewußtsein ... .“* ([LAK_82a], S. 1)

3.2 Kuhns Paradigmen

Kuhn nimmt im Gegensatz zu Lakatos eine relativistische Position ein. Das heißt nicht, daß er überhaupt keine Kriterien zur Theorienbeurteilung nennt. Die von ihm aufgeführten Kriterien lauten u.a.: Genauigkeit der Voraussage, Anzahl der gelösten Probleme, sowie: Einfachheit, Anwendungsbreite und Verträglichkeit mit anderen Spezialgebieten. Kriterien wie diese bilden die Werte der *„Scientific Community“*. Dadurch sind aber diese Kriterien und ihre Gewichtungen keineswegs unveränderlich, wie das ein Rationalist fordern würde, sondern sie variieren entsprechend dem historischen und kulturellen Hintergrund, in dem die wissenschaftliche Geimeinschaft eingebettet ist. Es gibt für Kuhn keine höhere Norm, als Billigung dieser Werte durch die Gemeinschaft. Insofern kann man Kuhn als “soziologischem Relativisten“ bezeichnen. Dementsprechend ist er auch davon überzeugt, daß die Werte der Gemeinschaft letztlich soziologisch (und natürlich auch psychologisch) festgestellt werden müssen.

3.3 Die anarchistische Erkenntnistheorie von Feyerabend

Eine Zusammenfassung findet sich in [CHA_94], Kap. 12. Feyerabends Ansatz basiert auf seiner Feststellung, daß die bisherigen wissenschaftstheoretischen Modelle nicht in der Lage sind, die historische wissenschaftliche Entwicklung zu erklären. Er zieht daraus den Schluß, daß Wissenschaft zu kompliziert und vielfältig sei, als daß sie jemals durch ein Modell erklärt werden könne. Bis hierher ein typisch relativistischer Ausgangspunkt. Die Eigenwilligkeit des Feyerabendschen Ansatzes liegt in der Radikalität seiner Folgerung: *„Anything goes“* kann als einzige Regel für wissenschaftliches Arbeiten angegeben werden. (Also strenggenommen überhaupt keine Regel.) Demgegenüber ist der rationalistische Standpunkt nicht nur wirklichkeitsfern, sondern nach Feyerabend sogar schädlich, denn:

1. Er engt durch dogmatische Regeln die Kreativität des einzelnen Forschers ein. Egal welche Kriterien für die Theorienwahl gefordert werden, sie hemmen das Potential einiger Theorien und Ansätze.
2. *“... der Versuch, die Regeln durchzusetzen, zur Erhöhung der fachlichen Fähigkeiten auf Kosten der Menschlichkeit führen muß“*.([FEY_83], S. 392)

Feyerabends Kritik betrifft nicht nur die rationalistische Haltung sondern die Wissenschaftstheorie insgesamt. Sie sei in ihren Ausprägungen viel zu kompliziert und entfernt vom Alltag des tätigen Wissenschaftlers, als daß sie von ihm noch beachtet werden könne. Wenn also jemand physikalische Forschung betreiben wolle, so soll er Physik studieren und nicht Wissenschaftstheorie.
Dennoch zieht Feyerabend eine Grenze zwischen „respektablen Denkern“ und sog. *„Cranks“*. Der Begriff des Cranks beinhaltet *“rabiate Weltverbesserer, unbeeinflußbare Verteidiger seltsamer Ideen, fast schon religiös angehauchte Prediger von barem Unsinn, Verrückte, Vernünftige mit großen blinden Flecken, arm im Geiste, einflußreiche Scharlatane.“* ([FEY_78], S. 102, Anm. 49) Den Unterschied macht das Verhalten *nach* Vorstellung einer Theorie aus, nicht so sehr die Aussage der Theorie selbst. Im Gegensatz zum Crank erlaubt der respektable Denker nämlich eine kritische Reflektion seiner Ideen.

Allerdings gibt es kein Kriterium zur Auswahl konkurrierender Theorien. Stattdessen wird in der anarchistischen Erkenntnistheorie der Begriff der *Inkommensurabilität* eingeführt. Alle Beobachtung ist theoriegetränkt, also ist auch eine Beobachtungsaussage in den Kontext einer Theorie eingebettet. D.h. sie verwendet Begriffe, die speziell in dieser Theorie eine eigene Bedeutung haben. Diese kann in anderen Theorien wieder ganz anders sein. Als Beispiel können z.B. die Begriffe von Raum und Zeit in ihren Rollen in der Newtonschen Mechanik und der Realativitätstheorie gesehen werden.
Unterscheiden sich nun zwei konkurrierende Theorien fundamental, so gibt es überhaupt keine begriffliche Grundlage, auf der ein Vergleich basieren könnte. Also ist die Theorienwahl nur rein subjektiv zu verstehen.
Desweiteren folgert Feyerabend, daß es keine Überlegenheit der wissenschaftlichen Methode gegenüber anderen Formen des Erkenntnisgewinns existiert. Dies beruht auf den zu unterschiedlichen Begriffswelten verschiedener Erkenntnisformen. Daraus folgt die Inkommensurabilität verschiedener Erkenntnisformen.
Die Radikalität dieser Forderung wird vielleicht am ehesten dadurch verdeutlicht, daß Feyerabend eine Art Trennung der Wissenschaft von der Gesellschaft fordert. Genau wie nach der Französischen Revolution die Einheit von Kirche und Staat aufgehoben wurde, sollte heute die Verquickung von Wissenschaft und Gesellschaft beendet werden. Das hieße z.B., daß Schülern die Möglichkeit gegeben wird, statt Naturwissenschaft Fächer anderer „Erkenntnisformen" zu belegen. (Astrologie, Voodoo u.ä.)
Neuartig an Feyerabends „Theoriekritik" ist vor allem eine ethische Komponente: Feyerabend bekennt sich zu dem Versuch Mills *„die Freiheit aus(zu)weiten, (um) ein erfülltes und befriedigendes Leben"* zu führen und unterstützt sein Eintreten für die *„Förderung der Individualität, die allein wohlentwickelte Menschen erzeugt, erzeugen kann."* (Zitiert nach [FEY_83].) Diese Haltung wird durch die anarchistische Erkenntnistheorie insofern unterstützt, als daß sie versucht die methodischen Zwänge, die den Einzelnen nur einengen, zu beseitigen.
Der Standpunkt Feyerabends ist wohl der relativistischste aller hier vorgestellten. Insofern treffen auf ihn alle Argumente des Rationalismus gegen den Relativismus besonders zu. Allerdings auch ohne eine rationalistische Haltung einzunehmen, können einige Kritikpunkte bemerkt werden.

1. Es muß nicht die dringlichste Aufgabe der Wissenschaftstheorie sein, die Wissenschaftsgeschichte zu erklären. (Auch wenn das ebenfalls die Forderung Lakatos' an eine Methodologie ist.) Also kann bisherigen wissenschaftstheoretischen Modellen nicht zum Vorwurf gemacht werden, Theorienwechsel der Wissenschaftsgeschichte nicht zu erklären. Das ist ja der Grund zur Unterscheidung von Theorienwahl und Theorienwechsel.
2. Es wäre wirklichkeitsfern zu bestreiten, daß bei der Theorienwahl keine subjektiven Elemente eine Rolle spielten. So wird von Feyerabend richtig betont, daß auch z.B. materielle Ausstattung oder Karrierewünsche eines Forschers seine Wahl der zu bearbeitenden Theorie beeinflußen.
   Allerdings heißt das nicht, daß die Theorienwahl keinerlei rationalistischen Argumenten zugänglich wäre.
3. Letztlich ist die Devise *„Anything goes"* nicht unbedingt so kreativitätsfördernd wie behauptet. Denn der Wissenschaftler wird so nicht motiviert, sich mit bestimmten Theorien auseinanderzusetzen. So ist denkbar, das „anything goes" auch *anything stays* bedeutet.

4 Kritik

In diesem Abschnitt sollen die erläuterten Standpunkten kritisch betrachtet werden. Der Darstellung von Verbindungen zwischen den Standpunkten folgt eine darauf basierende einheitlichere Sicht. Darin soll die Informatik als Wissenschaft eingeordnet werden.

4.1 Bezüge zwischen den Standpunkten

Der vorige Abschnitt hat die Ansätze von Lakatos, Kuhn und Feyerabend in ein Koordinatensystem der Achsen *Methodik, Aufgabe* und *Erkenntnis der Wissenschaft* eingeordnet. Nun soll dieses Koordinatensystem als solches untersucht werden. Diese Koordinaten sind nicht unabhängig voneinander variierbar, sondern ergeben nur in bestimmten Kombinationen Sinn, andere sind widersprüchlich. (Die Koordinaten sind also im mathematischen Sinne nicht „linear unabhängig".)
Der Zusammenhang zwischen Objektivismus versus Individualismus und Realismus versus Instrumentalismus ist bestimmt von der Annahme einer objektiven Welt und der Haltung zur Korrespondenztheorie: Der Realist muß, um eine realistische Beschreibung der Natur zu geben, eine Art Korrespondenztheorie annehmen, insofern setzt der Standpunkt des Realisten einen Objetivismus voraus. (Andersherum argumentiert: wenn Erkenntnis nur individuell existiert, so wird eine Beschreibung dessen, was in der Welt unabhängig von uns ist, unmöglich.) Aus der Negation dieser Beziehung erkennt man, daß der Individualismus auch einen Instrumentalismus zur Voraussetzung hat. Es ist ja auch sinnlos von „nur" individuell existenter Erkenntnis eine realistische Weltbeschreibung zu fordern, denn das würde einen objektivistischen Standpunkt benötigen.

Genauer: Wenn „meine“ individuelle Erkenntnis den Anspruch erhebt, die Welt realistisch zu beschreiben, so gibt es keinen Grund, warum nicht auch andere sie teilen sollten. Also ist sie somit doch nicht nur rein individuell.
Ein weniger klarer Zusammenhang besteht im Rahmen der Methodenfrage. Zwar scheint ein Individualist eher eine relativistische Haltung einzunehmen, denn wenn Erkenntnis nur individuell existiert, so machen allgemeinverbindliche Regeln zum Erkenntnisgewinn kaum Sinn. Dennoch kann gefordert werden, daß auch für den individuellen Erkenntnisgewinn Regeln beachtet werden müssen, wie z.B. die erwähnte Bereitstellung “ausreichenden Beweismaterials“.
Die Negation ist ohnehin frei variierbar: Der Realist kann einen relativistischen Standpunkt wählen, Feyerabend würde sich hüten, eine Vorschrift zu erlassen, daß eine realistische Beschreibung der Welt in seiner Theorie unmöglich ist. Ein Realist kann eine Beschreibung dessen, was in der Welt ist, suchen, dies aber ohne festgeschriebene Regeln.
Genau das kann er aber auch tun, wenn er allgemeine Regeln befolgt, er kann also auch ein Rationalist sein.

4.2 Einheitliche Sicht

Meiner Auffassung nach können einige Positionen zusammengefaßt werden, wenn der Begriff der Objektivität nicht mehr naiv verwandt wird. Im wesentlichen verbergen sich hinter dem Begriff Objektivität zwei Eigenschaften, die zu unterscheiden einige Widersprüche aufhebt:

1. Bei der Schilderung des Objektivismus wird *objektiv* im Sinne von *unabhängig* von einzelnen Individuen verstanden. Die gesuchte Erkenntnis ist unabhängig vom Menschen existent, sie bezieht sich also auf eine von sich allein aus gegebene Umwelt. Diese Auffassung von Ojektivität ist also eine *naturbezogene*.
2. Objektiv kann aber auch im Sinne von *Allgemeinverbindlichkeit* verstanden werden. D.h. eine objektive Erkenntnis ist für alle vernünftigen Menschen stringent und nachvollziehbar. (*Vernunft* kann als Denken in Übereinstimmung mit den Axiomen eines logischen Systems definiert werden.)

Bei der naiven Verwendung des Begriffes Objektivität erscheint eine Unterscheidung sinnlos: wenn Erkenntnis sich auf einen Gegenstand bezieht, der außerhalb des Menschen liegt (eben der Natur), so ist doch diese Erkenntnis auch allgemeinverbindlich.
Der Theorienwechsel von der Newtonschen Mechanik zur Modernen Physik mag den Unterschied erläutern: Als Newton 1687 seine *Philosophiae naturalis principia mathematica* veröffentlichte, hatte er religiöse Motive dazu. Das ist als objektive Beschreibung der Natur zu verstehen, denn eine Natur Gottes ist eine nicht von der subjektiven menschlichen Wahrnehmung bestimmte Natur. (vgl. [HEU_92], S. 657f.) Diese Haltung Newtons werden auch Einstein oder Heisenberg nicht in Zweifel gezogen haben. Dennoch war für beide die Theorie Newtons keine objektive Beschreibung der Welt, in dem Sinne als daß Newtons Theorie die Welt fehlerfrei (d.h. realistisch) beschreibt. Sie faßten Newtons Mechanik eher als vorläufig auf, und korrigierten beide diese Theorie (allerdings in unterschiedliche Richtungen), ohne einen Anspruch auf letzte Wahrheit mit ihren Korrekturen zu erheben. Also war Newtons Theorie für beide nicht allgemeinverbindlich, obwohl sie zweifelsohne eine Natur außerhalb des Menschen beschreibt.
Heisenberg ist darüberhinaus der Auffassung, daß durch die Quantenmechanik ein subjektives Element in physikalische Beschreibungen gelangt ist. (z.B. [HEI_59]) Ebenso verhält es sich mit kosmologischen Theorien: Sie alle sollen das beschreiben, was in der Welt existent ist (also ein objektivistischer Ansatz), dennoch kann wohl kaum die Superstringtheorie oder eine Theorie Symmetrischer Gruppen als Einzige Allgemeinverbindlichkeit durch die *Existenz* der in ihr erwähnten Konstrukte behaupten.
Auf der anderen Seite steht die allgemeinverbindliche Seite der Objektivität. Es ist klar geworden, daß weitergehende Theorien, die sich mit der Natur befassen, diesen Anspruch nicht erheben können.
Allerdings kann die Mathematik, als rein geistiges Gebilde (ohne Naturbezug) dies beanspruchen. So ist z.B. der Satz des Pythagoras wahr, unabhängig davon, ob er nun in der realen Welt anwendbar ist oder nicht. (Resultate der Allgemeinen Relativitätstheorie zeigen, daß die Euklidische Geometrie nur eine annähernde Beschreibung an natürliche Gegebenheiten zuläßt.) Diese Wahrheit kann so formuliert werden: *„Falls die Euklidische Geometrie angenommen wird, so gilt der Satz des Pythagoras.“* Sie beruft sich zum einen auf Axiome (Euklidische Geometrie als Axiomensystem), zum anderen auf die logischen Schlußregeln, mit denen ja der Satz des Pythagoras *allgemeinverbindlich* bewiesen ist.[9] Der Raum, in dem mathematische Sätze (und damit auch Algorithmen) liegen, ist also objektiv im Sinne von allgemeinverbindlich vorhanden. (Jeder als totalkorrekt bewiesene Algorithmus impliziert einen mathematischen Satz der Bauart: “Algorithmus x löst Problem y, unter z Rahmenbedingungen.“) Allerdings ist dieser Raum nicht von Natur aus gegeben, die ihn aufspannenden Axiome und Schlußregeln sind von Menschen definiert. Inwieweit diese Axiome eine Nutzung der Mathematik zur Naturbeschreibung zu-

[9] Logische Schlußregeln sollten besser auch axiomatisch begründet werden, als psychologisch. Sonst müßte u.a. untersucht werden, welches System der Logik dem menschlichen Denken am ehesten entspräche. Das aber wäre ein individualistischer Ansatz.

lassen, muß durch Experimente überprüft werden. Insofern besteht in der Mathematik (und ihrer Grundlage, der Logik) ein Objektivitätsbegriff, der keine Naturbezogenheit beinhaltet.
Konkret gibt diese Unterteilung methodische Hinweise: Axiomatik und Beweisbegriff in der Mathematik führt zur Allgemeinverbindlichkeit, Experimente in den Naturwissenschaften zur Naturbezogenheit.
Weder eine rein realistische noch eine rein instrumentalistische Ansicht allein kann beide Arten der Erkenntnis wiedergeben.

4.3 Einordnung der Informatik

Die klassische Unterteilung der Kerninformatik in die Bereiche theoretische, technische und praktische Informatik mag viele der heutigen Forschungsschwerpunkte kaum noch abbilden. (Z.B. der Bau maschinenspezifischer Compiler berührt alle drei Gebiete.) Dennoch soll sie hier benutzt werden, um sie in die oben erwähnten Gebiete naturbezogen und objektiv-allgemeinverbindlich einzuordnen.
Die theoretische Informatik beschäftigt sich im weiteren Sinne mit dem Begriff des Berechnens, wobei Rechnen sich keineswegs nur auf Zahlen bezieht. Insofern kann sie von ihrer Methodik als Teilgebiet der Mathematik gesehen werde, ihre Erkenntnisse sind also bewiesen und allgemeinverbindlich.
Die technische Informatik basiert auf Elektrotechnik und Physik, weshalb sie eher den Ingenieurwissenschaften zuzuordnen ist. Die technische Maxime der Verwertbarkeit des Wissens entspricht auch eher einem instrumantalistischen Standpunkt.
Die praktische Informatik befaßt dich mit dem Bau von Werkzeuge für die Computeranwendung. Die Leistungsfähigkeit dieser Werkzeuge kann nur zum Teil durch technische Daten wiedergegeben werden. Wenn diese Werkzeuge vom Menschen eingesetzt werden, so muß auch die Verwendbarkeit durch entsprechende Experimente geprüft werden. Insofern existiert ein Bezug zum Menschen und damit zur Natur.
Bei der Erstellung informationsverarbeitender Systeme für konkrete Problemlösungen findet ein Modellierungsprozeß statt. Werden natürliche Gegebenheiten z.B. aus der Physik modelliert, so fällt der Informatik eine ähnliche Rolle zu wie z.B. der Integral- und Differentialrechnung im 17. und 18. Jahrhundert. Hier soll die Informatik Methoden der Modellierung bereitstellen. Darüberhinaus prägt die Computersimulation einen neuen Experimentalbegriff. (Neuere Teilgebiete der Physik wie z.B. *Computational Physics* tragen dieser Entwicklung Rechnung.)
Werden Computersysteme in der Administration eingesetzt, so ist eine Einordnung schwieriger, denn inwieweit können verwaltungstechnische Vorgänge in Wirtschaft und Staat als „natürlich“ angesehen werden? Werden in der Praxis Algorithmen entworfen, so sind diese allgemeinverbindlich. Denn was für ein Kriterium könnte stärker die allgemeinverbindliche Objektivität testen, als die Forderung nach Ausführbarkeit von Lösungsmethoden auf unintelligenten Maschinen. Schließlich sind Rechner nicht in der Lage, irgendwelche Zweideutigkeiten einer Lösungsmethode zu interpretieren.

5 Schlussbetrachtung

Die Ansätze des Relativismus und Rationalismus wurden bezüglich der Methodenfrage erklärt, weiterhin die Aufgabe von Wissenschaft durch die Haltungen Realismus versus Instrumentalismus erläutert. Die Positionen des Objektivismus und Individualismus betrafen die Art der Erkenntnis. Kongruenzen und Widersprüche zwischen den Standpunkten wurden dargestellt. Der Begriff der Objektivität zeigte allgemeinverbindliche und naturbezogene Aspekte. Das führte zu einer neuen Sichtweise des Erkenntnisbegriffs mit Auswirkungen auf die Methodenfrage. Die verschiedenen Teilgebiete der Informatik besitzen Beziehungen zu beiden Aspekten der Erkenntnis. Insofern kann Informatik als Wissenschaft nicht durch einen einzelnen Standpunkt erfaßt werden.

Literatur

[CHA_94] Chalmers, A. F. *Wege der Wissenschaft.* Springer, Berlin, 3. Auflage, 1994.

[FEY_78] Feyerabend, P. K. Realismus und Instrumentalismus: Bemerkungen zur Logik der Unterstützung durch Tatsachen. *Der wissenschaftstheoretische Realismus und die Autorität der Wissenschaften. Ausgewählte Schriften, Band 1, pp. 79-112.* Vieweg, Braunschweig, 1983.

[FEY_83] Feyerabend, P. K. *Wider den Methodenzwang: Skizze einer anarchistischen Erkenntnistheorie.* Suhrkamp, Frankfurt/Main 1976, veränderte Ausgabe 1983.

[HEI_28] Heisenberg, W. Erkenntnistheoretische Probleme in der modernen Physik. Unveröffentlichter Vortrag, 1928. *Werner Heisenberg. Gesammelte Werke. Abteilung C I.* pp. 22-28, Piper, München, 1986.

[HEI_59] Heisenberg, W. *Physik und Philosophie.* S. Hirzel, Stuttgart, 1959.

[HEU_92] Heuser, H. *Lehrbuch der Analysis Teil 2.* B.G. Teubner, Stuttgart, 7. Auflage, 1992.

[LAK_74] Lakatos, I. Falsifikation und Methodologie wissenschaftlicher Forschungsprogramme. Pseudo-Wissenschaft. I. Lakatos & A. Musgrave (Hrsg.) *Kritik und Erkenntnisfortschritt.* pp. 89-189, Vieweg, Braunschweig, 1974.

[LAK_82a] Lakatos, I. Wissenschaft und Pseudo-Wissenschaft. J. Worrall & G. Currie (Hrsg.) *I. Lakatos: Philosophische Schriften Band 1.* pp. 1-6, Vieweg, Braunschweig, 1982.

[LAK_82b] Lakatos, I. Warum hat das Kopernikanische Forschungsprogramm das Ptolemäische überrundet? J. Worral & G. Currie (Hrsg.) *I. Lakatos: Philosophische Schriften Band 1.* pp. 182-208, Vieweg, Braunschweig, 1982.

[PLA_37] Planck, M. Determinisums oder Indeterminismus. *Vom Wesen der Willensfreiheit und andere Vorträge.* pp. 192-212, Fischer, Frankfurt/Main, 1991.

[TAR_36] Tarski, A. Der Wahrheitsbegriff in den formalisierten Sprachen. *Studio philosophica.*, 1, 1936.

[TaD_88] Troelstra, A.S., van Dalen, D. Constructivism in Mathematics. Barwise, J. et al. (Hrsg.) *Studies in Logic and the Foundations of Mathematics, Vol. 121.* North-Holland, Amsterdam, 1988.

[THO_91] Thompson, S. *Type Theory and Functional Programming* Addison-Wesley, Wokingham, England, 1991.

[WIT_45] Wittgenstein, L. Philosophische Untersuchungen. 1945, *Werkausgabe: Ludwig Wittgenstein. Band 1.* Suhrkamp, Frankfurt/Main 1984

Ein Experiment zur Kosteneffektivität von Inspektionsverfahren

Ingo Redeke

Zusammenfassung

Ziel dieses Experiments war es, von den Autoren aufgestellte Hypothesen über Variationen von Inspektionsverfahren zu überprüfen und eine Basis zur Festlegung der Kosten für diese zu finden. Man konzentrierte sich daher auf den Vergleich der Zeitdauer der Verfahren und ihrer Effektivität bei der Fehlerdetektion. Variiert wurden die Verfahren in der Anzahl der Sitzungen, der Größe des Teampersonals und dem Korrekturverhalten bei Mehrfach-Sitzungen.

Nach Auswertung der ersten 27% des Datenumfangs wurde festgestellt, daß bei einem durchschnittlichen Zeitaufwand von 10 Tagen pro Inspektion $2s * 2p$-Verfahren die besten Resultate liefern. Dabei lag der Anteil der Erkennung von *echten* Fehlern nach der Vorbereitungsphase bei 17%, nach der Besprechungsphase bei 25%. Im weiteren Verlauf des Experiments (nach 34 Inspektionen - ca. 53%) änderte sich diese Beobachtung nur unwesentlich. Bei einem durchschnittlichen Zeitaufwand von nunmehr 14,5 Tagen pro Inspektion lag die Entdeckungsrate bei 15% nach der Vorbeitungsphase und bei 21% nach der Besprechungsphase, wobei sich $2s * 2pN$ und $2s * 2pK$ in ihrere Effektivität kaum unterschieden.

Es wurde Wert auf eine industrienahe, kontrollierte Durchführung des Experiments geachtet, weil die Resultate vornehmlich in diesem Bereich ihren Nutzen finden.

1 Einleitung

Trotz der weitverbreiteten Übereinstimmung über die Notwendigkeit von Inspektionen, ist sich heute kaum ein Unternehmen im klaren darüber welche Kosten sie verursachen.

Es müssen daher Untersuchungen und Messungen über den Wert einer solchen Inspektion, ihre Dauer und Effektivität bei der Entdeckung von Fehlern gemacht werden. Um dafür einen Maßstab ansetzen zu können, sind im vorliegenden Experiment alle zur Fehlererkennung und Korrektur notwendigen Handlungen seitens des Teams statistisch verteilt worden. Ebenso hat man alle anderen äußeren Bedingungen variiert, um aus einem möglichst breiten Spektrum von Resultaten die entscheidenden Effektivitätsmerkmale herauszufiltern.

Das Projekt selbst bestand aus der Erstellung einer Software zur Steuerung eines Telefonvermittlungssystems bei AT&T.

2 Notation und Begriffe

2.1 Begriffe

- Inspektionssitzung, Sitzung: Ein Gesamtdurchlauf der Inspektion, der in mehrere Inspektionsintervalle unterteilt sein kann.
- Inspektionsintervall, Intervall: Zeitdauer von der Abgabe der Code-Einheit bis zum Ende der Korrekturphase inklusive der Leerlaufzeiten zwischen den Phasen (Man beachte die spätere Änderung dieser Definition in Abschnitt 5). Unterschieden werden Codierungsphase (**C**), Vorbereitungsphase (**V**), Besprechungsphase (**B**) und Korrekturphase (**K**).

2.2 Variablen

Die variierten Bedingungen werden durch drei unabhängige Variablen dargestellt: $S*T'K'$ (z.B. $2s*2pN$ bedeutet 2 Sitzungen mit je 2 Personen ohne Korrektur). Es gibt insgesamt $3 * 2^2$ Kombinationen, von denen jedoch $2s*4p$-Verfahren aus Kostengründen ausgelassen wurden. Zusätzlich gibt es noch vier abhängige Variablen.

unabhängige Variable

- T: Teamgröße - eine, zwei oder vier Personen (jeweils zusätzlich zum Autor). Dabei ist $T \geq T'$.
- S: Anzahl der Sitzungen - eine oder zwei Sitzungen.
- K': Boolscher Wert, der angibt ob bei S=2 eine Fehlerkorrektur zwischen den Sitzungen stattfindet (Wert K) oder nicht (Wert N).

abhängige Variable

- I_n: Länge des Inspektionsintervalls.
- R_g: Geschätzte Fehlerentdeckungsrate.
- R_b: Entdeckungsrate der Besprechungen.
- R_f: Prozentualer Anteil der potentiellen Fehler, die bei Besprechungen als irrelevant aussortiert werden.

- A: Inspektionsaufwand, gemessen als Summe der entscheidenden Unterintervalle. (Wurde erst im zweiten Teil des Experiments hinzugefügt. Die dafür relevanten Daten sind noch nicht vollständig analysiert worden.)

2.3 Hypothesen

Von den Autoren werden folgende Hypothesen aufgestellt, die durch das Experiment überprüft werden sollen.

- Inspektionen mit großen Teams benötigen größere Intervalle, finden aber dafür mehr Fehler als kleine Teams.
- Besprechungen erhöhen die Effektivität der Fehlerentdeckung nicht sonderlich.
- Inspektionen aus Mehrfach-Sitzungen sind effektiver als Einzel-Sitzungen, erhöhen allerdings die Dauer der Inspektionsintervalle.

Zur Überprüfung der Hypothesen müssen die signifikanten Variablen gefunden und ihr Einfluß analysiert werden.

3 Aufbau und Voraussetzungen

Zu erstellen sind ein Compiler und eine Entwicklungsumgebung für ein Telefonvermittlungssystem mit einer geschätzten Größe von 30K Codezeilen.

Das Team besteht aus zunächst 6, später 11 erfahrenen Softwareentwicklern sowie einem IQE (Inspection Quality Engineer), der die Rolle eines Koordinators und Beobachters einnimmt. Der Code-Autor selbst ist kein Inspekteur.

Das Gesamtexperiment geht über 64 Inspektionsverfahren (reduziert von zuvor auf 100 angesetzten Inspektionen), die jeweils in ein oder zwei Sitzungen unterteilt werden. Für die Vorbeitungs-, Besprechungs-, und Korrekturphasen sind jeweils spezielle Formblätter (im folgenden V-, B-, K-Formblätter, oder -Berichte genannt) vorbereitet worden, die die Teilnehmer entsprechend ausfüllen müssen. Sie enthalten u.a. Einträge für Zeitdauer der Phase sowie Lokalität der Fehler.

Es gibt kein bestimmtes Verfahren, mit dem die Inspekteure in der Vorbereitungsphase arbeiten müssen. Um nach Möglichkeit eine weite statistische Streuung zu erreichen, wird den Inspekteuren freie Hand bei ihren individuellen Verfahren gelassen.

3.1 Schätzmethoden für die Fehlerrate

Die Fehlerentdeckungsrate $R = \frac{F_g}{N}$ ist ein wichtiges Maß zur Bestimmung der Effektivität. Sie ist definiert als das Verhältnis der Anzahl aller während der Inspektion gefundenen Fehler F_g zur Gesamtzahl aller Fehler N.

Da N i.d.R. unbekannt ist, muß es geschätzt werden. Diese Schätzung soll dabei so präzise wie möglich und zu jedem Zeitpunkt des Experiments durchführbar sein, um ineffektive Methoden frühzeitig zu erkennen. Um das zu erreichen, entschieden sich die Autoren für die Verwendung von sogenannten „capture-recapture"-Verfahren (im weiteren kurz CRC). Den Hintergrund für diese Verfahren liefert die Vorstellung, daß nur noch wenige unentdeckte Fehler erwarten werden können, wenn mehrere der (voneinander unabhängigen) Inspekteure eine hohe Zahl von Fehlern übereinstimmend finden. Auf der anderen Seite spricht eine große Zahl verschiedener gefundener Fehler dafür, daß noch etliche Fehler unerkannt geblieben sind. (Entwickelt wurden diese Verfahren von Burnham und Overton, um die Anzahl von Wildbeständen zu schätzen. Daher auch die Bezeichnung „capture-recapture". Siehe auch [4].)

Es stehen zwei CRC-Verfahren zur Verfügung, die in Abhängigkeit von drei Voraussetzungen angewandt werden:

A Die Tätigkeiten der Inspekteure sind statistisch unabhängig.

B Alle Inspekteure sind gleich effektiv bei der Enteckung von Fehlern.

C Alle Fehler haben die gleiche Entdeckungswahrscheinlichkeit.

Wird Bedingung A verletzt (z.B., wenn Inspekteure zusammenarbeiten, oder sich absprechen, gezielt nach disjunkten Fehlermengen zu suchen), kann kein verläßlicher Schätzer gefunden werden.

Sind die Bedingungen A und B erfüllt, so kann eine Daumenregel verwendet werden.

Daumenregel:

$$N = max\{0, a_1 f_1 + \cdots + a_k f_k\} + n \qquad (1)$$

mit n als der Gesamtzahl aller in der Vorbereitungsphase gefundenen Fehlern, m die Anzahl der Inspekteure, $k \leq m$, f_j die Anzahl Fehler, die von genau j Inspekteuren gefunden wurden. Die $a_1 ... a_k$ sind von k und m abhängige Konstanten.

Sind dagegen Bedingung A und C erfüllt, B jedoch nicht, ist ein Maximum-Likelihood-Schätzverfahren zu verwenden.

Maximum-Likelihood-Schätzung:

$$L(X) = \ln\binom{X}{n} + \sum_{j=1}^{m}(X \Leftrightarrow n_j)\ln(X \Leftrightarrow n_j) \Leftrightarrow Xm\ln(X) \quad (2)$$

Gesucht ist das Maximum bezüglich X. n_j ist die Anzahl der Fehler, die vom Inspekteur j gefunden werden.

Im Verlauf des Experiments wurde deutlich, daß weder **B** noch **C** erfüllt werden. Dennoch können nach Wiel [3] die Fehler in eine kleine Anzahl von Klassen ähnlich großer Entdeckungswahrscheinlichkeiten eingeordnet werden. An Hand der Gleichung (2) ist damit eine Schätzung der Größe jeder dieser Unterpopulationen möglich. Die Kombination der Einzelschätzungen liefert den gesuchten Schätzer für die Gesamtfehlerzahl.

3.2 Sensitivitätsanalyse

Die *Sensitivität* des Experiments, also der minimal erkennbare Unterschied zwischen den Verfahren, wird durch eine Monte-Carlo Simulation[10] bestimmt. Dazu stehen zwei Verfahren T_a und T_b mit ihren Fehlererkennungswahrscheinlichkeiten p_a und p_b zur Verfügung.

Zur wird zunächst eine Anzahl Code-Einheiten mit bekannter Größe g und Fehlerdichte f_f (Normalverteilung mit μ und σ) erzeugt. Es ist dann $N = f_f * g$.

Danach werden T_a und T_b auf unterschiedliche Code-Gruppen angewendet, die aus 5, 10 oder 15 Einheiten bestehen. Eine Binomialverteilung mit den Parametern $p_{a,b}$ und N (einer Einheit) liefert die Zahlen $n_{a,b}$ der mit Hilfe von $T_{a,b}$ gefundenen Fehler.

Der Wilcoxon-Vorzeichenrang-Test[11] ermittelt, ob n_a und n_b aus der gleichen Population stammen (gleichverteilt sind).

Diese Prozedur wird 100 mal für jedes Experiment wiederholt.

Die Autoren führten 600 derartige Einzelexperimente durch. Diese wurden gebildet aus 25 Kombinationen der Mittelwerte (53, 67, 80, 93, 107) und den Standardabweichungen (3, 7, 13, 27, 40), sowie 24 Paaren von Wahrscheinlichkeiten $p_a \in \{0.2, 0.4, 0.6, 0.8\}$ und $p_b = p_a + d$ mit $d \in \{0, 0.025, 0.05, 0.075, 0.1, 0.15\}$.

Die Sensitivität des Experiments ist als die Differenz d_i zwischen p_a und p_b definiert, für die gilt:

- bei mehr als 50% der 100 Einzelresultate wird die Nullhypothese zu mehr als 90% zurückgewiesen.
- bei d_{i-1} wird für weniger als 50% die Nullhypothese zu mehr als 90% zurückgewiesen.
- Die Nullhypothese ist definiert als $p_b = p_a$.

Die Simulation für verschiedene Werte ergab i.d.R. eine Sensibilität von $d = 0.075$, wobei der stärkste Einfluß bei die Standardabweichungen lag: je kleiner σ, desto feiner die Sensitivität.

3.3 Interne und Externe Gültigkeit

Als Einschränkung für die externe Gültigkeit wurden alle Einflüsse analysiert, die eine Generalisierbarkeit der Ergebnisse des Experiments für industrielle Nutzung beeinträchtigen.

- Generalisierung des Personals: Durch das ausschließliche Verwenden von Mitarbeitern aus dem Industriebereich kann das Personal generalisiert werden.
- Äußere Situation: Das Ausführen des Experiments in einer industriellen Umgebung macht das Experiment repräsentativ für diesen Bereich.
- Representativität des Personals: Da sich das Personal nicht aus allen Bereichen der industriellen Entwicklung rekrutiert, besteht hier eine leichte Gefahr für die externe Gültigkeit des Experiments.

Um interne Gültigkeit gewährleisten zu können sucht man zu verhindern, daß die abhängigen Variablen nicht ohne Kenntnis der Wissenschaftler beeinflußt werden. So sind Maßnahmen zu ergreifen bei:

- Auswahleffekten: Einzelne Personen können z.B. durch bessere Fähigkeiten als andere die Einzelsitzungs-Methoden als besonders effektiv erscheinen lassen, obgleich dies nur von den Fähigkeiten abhängt. Vermeiden läßt sich dieser Effekt durch zufälliges Auswählen und zuweisen des Teampersonals.
- Reifungseffekten: Die Fähigkeiten eines Entwicklers können sich durch die reinen Inspektionstätigkeit während des Experiments verbessern. Das zufällige Auswählen des Personals soll hier Inhomogenitätseffekte verhindern.
- Materialeffekte: Das verwendete Experiment-Material kann interne Gültigkeit verletzen, wenn Größe oder Qualität des Codes variieren oder etwa Unterschiede in der Datensammlung auftreten.

[10] Monte-Carlo-Methoden ermitteln das wahrscheinliche Verhalten eines Systems, dessen Komponenten den Wahrscheinlichkeitsgesetzen unterliegen durch häufiges wiederholen des simulierten Vorgangs.

[11] siehe dazu [6]

Um das zu verhindern, haben die Wissenschaftler einheitliche Formblätter für jede Phase kreiert und lassen die Anwendungsmethode nach zufälligem Muster auswählen.

4 Durchführung

Steht eine Codeeinheit zur Inspektion bereit, informiert der Autor den IQE, der seinerseits nach dem Zufallsprinzip ein Anwendungsverfahren sowie das Teampersonal auswählt. Dabei darf kein Inspekteur bei $S = 2$ in beiden Teams vertreten sein.

In der Vorbereitungsphase erhält jeder Inspekteur eine Kopie der Code-Einheit. Er trägt die Anfangs- und Endzeit seiner Kontrolltätigkeit sowie Seite und Zeile jedes vermuteten Fehlers in sein V-Formblatt ein. Gesucht wird dabei nach Kommentarfehlern ebenso wie nach Verletzungen von Standards und nach echten Implementierungsfehlern. Die Wahl der dabei verwendeten Technik bleibt dem einzelnen Inspekteur überlassen. (In den meisten Fällen handelte es sich um intuitive ad hoc-Verfahren oder Suche nach Checklisten).

Ist der Besprechungstermin festgelegt, sammelt der IQE alle Formblätter und bearbeitete Code-Einheiten ein und ruft das Team zur Besprechung zusammen. Dort werden unwichtige oder unkorrekte Fehlerbeschreibungen aussortiert und eventuell neue Fehler entdeckt. Allen als *echt* eingestuften Fehlern wird eine eigene ID zugewiesen (zur Unterscheidung zwischen *echten* Fehlern und anderen, siehe Abschnitt „Analyse und Auswertung“).

Zeitdauer der Besprechung, Seite, Zeile und ID der echten Fehler werden im Besprechungs-Formblatt eingetragen. Die mit den neuen IDs versehenen V-Berichte gehen an den Code-Autor, der dann mit der Korrekturphase beginnt.

Abschließend übergibt der Autor alle Berichte einschließlich seines Korrektur-Formblatts dem IQE, der den Autor zur Datenvalidation befragt (z.B. müssen bei Mehrfach-Sitzungen Fehler, die eventuell doppelt gefunden wurden aussortiert werden). Im K-Bericht wird eingetragen, ob der Fehler korrigiert werden mußte oder etwa zurückgestellt wurde, ob er andere Fehler ausgelöst hat und ähnliche Charakteristika.

5 Auswertung

Die Daten aus allen drei Klassen von Berichten bilden die Grundlage für die Auswertung.

Dabei wird die Effektivität der $S * T'K'$- Methode durch das Zusammenfassen aller vom Team gefundenen Fehler bestimmt. Durch Vergleichen der B-Berichte beider Sitzungsteile in $2sN$-Verfahren, erhält man ein Maß für den zusätzlichen Gewinn durch 2-fach Sitzungen.

Den V-Formblättern kann entnommen werden, welche Fehler der jeweilige Inspekteur erkannt und welche er übersehen hat. Zusammen mit der oben genannten Team-Fehlerrate liefert diese Information die Basis für den CRC-Schätzer und ein Maß für den Gewinn durch Besprechungen R_b.

Außerdem liefern die Berichte eine Übersicht über die verstrichene Zeit, so daß Vergleiche zwischen den durchschnittlichen Inspektionsintervallen und der Verteilung der Intervalle für jede Methode gezogen werden können.

Die Aufbereitung der Daten findet nach zwei Gesichtspunkten statt:

- Fehlerdaten: Abhängig von den K-Berichten, wurden drei Fehlerklassen erzeugt. Die Einteilung fand deswegen erst nach der Korrekturphase statt, weil sich gezeigt hat, daß die Code-Autoren verläßlichere Urteile über die Fehler abgeben können, nachdem sie diese bearbeitet haben.

  - Falschentscheidungen (false positives): Fehler, für die keine Änderung notwendig ist.
  - Leichte Fehler: Fehler, deren Korrektur lediglich die Lesbarkeit des Codes erhöhen.
  - Echte Fehler: Fehler, die den korrekten Ablauf verhindern, Anforderungen unerfüllbar machen oder die Effizienz beeinträchtigen.

  Nach 17 der insgesamt 64 Inspektionen (ca. 27%) konnten im Durchschnitt aller betrachteten Verfahren 18% der während der Besprechungsphase vermuteten Fehler als *Falschentscheidungen*, 57% als *leichte Fehler* und nur 25% als *echte Fehler* eingestuft werden. Zu diesem Zeitpunkt lieferten die V-Berichte im Durchschnitt 17% *echte Fehler*.

  Nach 34 Inspektionen (ca. 53%), lag der durchschnittliche Anteil der *Falschentscheidungen* bei den Verfahren bei 24%. 55% waren *leichte Fehler* und 21% schließlich stellten sich als *echte Fehler* heraus. Hier lag die Entdeckungsrate der *echten Fehler* im Durchschnitt aller V-Berichte bei 15%.

- Intervalldaten: Es hat sich gezeigt, daß die Zeit zwischen den tatsächlichen Inspektionsaktivitäten für die Analyse nicht verwertet werden kann. Daher werden an zwei Stellen diese Zeiten neu bewertet.

  - Einige Autoren haben nicht sofort im Anschluß an die Besprechungen mit der Fehlerkorrektur begonnen, sondern sich zunächst

auf andere Arbeiten konzentriert. Das Inspektionsintervall wird daher neu definiert als der Zeitraum ab Beginn der Vorberitungsphase bis Ende der Besprechungsphase. Bei 2s-Methoden wird das längere der so entstandenen Unterintervalle betrachtet.

- Alle Tage an denen keine Inspektionsaktivität stattfindet werden aus der Auswertung ausgenommen.

Alle weiteren Analysen finden aufgrund dieser reduzierten Zeiten statt.

5.1 Intervalldaten

Da die Intervalldauer ein sehr wichtiges Kostenmaß darstellt, werden die Intervalle der verschiedenen Methoden miteinander verglichen:

- Der Vergleich von $1s*1p$-, $1s*2p$-, und $1s*4p$-Methoden zeigt keinen Unterschied hinsichtlich der Kosten. Man kann daher schließen, daß die Teamgröße T hier keinen Einfluß auf die Intervallänge ausübt.
- Die zusätzlichen Kosten durch Mehrfach-Sitzungsverfahren lassen sich beim Vergleich von $1s$-, und $2sN$-Methoden zeigen. Dabei stellt man fest, daß $I_{2sN} \leq I_{1s}$ ist. Allerdings variiert I_{2sN} stärker als I_{1s}.
- Die Kosten der Serialität von Mehrfach-Sitzungen werden durch Vergleich von $2sK$-, mit $2sN$-Verfahren ermittelt: bei $1p$ macht sich kein Unterschied für die Intervallzeiten $I_{2sK,2sN}$ bemerkbar, wohl aber bei $2p$-Verfahren. Dieses Verhalten deutet darauf hin, daß die Intervallänge nur durch größeres T beeinflußt wird.
- Die mittlere Intervalldauer liegt für fast alle Verfahren bei etwa 7,5 Tagen. Lediglich $2s*2pK$ dauert länger (20 Tage).

5.2 Effektivität der Fehlererkennung

Zunächst einige Vergleiche bezüglich der Effektivität durch Fehlererkennung:

- Alle Verfahren mit Einzelsitzungen unterscheiden sich untereinander nicht in ihrer Effektivität, obwohl mit der Zahl der Personen auch Mittelwert und Varianz ansteigen.
- Die $2s*1p$-Methoden liegen beide etwa gleich zu den $1s$-Methoden.
- $2s*2p$-Methoden sind effektiver als jede Einzel-Sitzungsmethode.
- $2s*2pK$-, und $2s*2pN$-Methoden unterscheiden sich nur sehr unwesentlich in ihrer Effizienz.

Auffallend ist der Unterschied zwischen $2s*2p$ und $1s*4p$. Damit wären 4 Personen effektiver bei der Fehlererkennung, wenn sie in 2 kleinen Gruppen arbeiten, als in einer Einzelgruppe. Andererseits ist der Unterschied zwischen $2s*1p$ und $1s*2p$ nur sehr gering.

Der Vergleich der pro-Kopf-Raten beim Erkennen von echten Fehlern zeigt einen deutlichen Vorteil der $2p$-Verfahren gegenüber allen anderen, sogar denen mit $T' = 4$. Am besten schneidet dabei $2s*2pN$ ab, daß mit 20% über dem Durchschnitt von 15% liegt.

5.3 Besprechung

Der Anteil der bei den Besprechungen aussortierten irrelevanten oder unwichtigen Fehler R_f (also Einträge in den V-Berichten, die weder echte, noch leichte Fehler, noch Falschentscheidungen sind), liegt im Duchschnitt aller Verfahren bei 25%.. Dabei unterscheiden sich die R_f der einzelnen Methoden kaum voneinander.

Während der Besprechungsphase werden im Durchschnitt aller betrachteten Verfahren $R_b = 33\%$ aller Fehler gefunden, mit einem maximalen Mittelwert bei den $2s*1pK$-Verfahren (ca. 50%).

6 Ergebnisse

Wie man den Ergebnissen entnehmen kann, werden nicht alle anfänglich aufgestellten Hypothesen der Autoren bestätigt.

Es hat sich gezeigt, daß tatsächlich $R_b = 33\%$ ist, ganz im Gegensatz zur Vermutung, daß Besprechungen nicht sonderlich zur Fehlerentdeckung beitragen. D.h. im Durchschnitt aller Anwendungsmethoden wurden 33% der entdeckten Fehler bei Besprechungen gefunden. Zu der Hypothese kam es, weil einer der Autoren (Votta) in einer ähnlichen Studie gezeigt hat, daß nur 5% aller gefundenen Fehler bei Besprechungen entdeckt werden. Allerdings handelte es sich dabei um *Design-Inspektionen* und nicht um Code-Inspektionen.

Beim derzeitigen Stand des Experiments (53% des Umfangs) kann noch keine Aussage über die dritte Hypothese, die Kosteneffektivität von Mehrfach-Sitzungen im Ggs. zu Einzel-Sitzungen, gemacht werden. Allerdings scheint die erste Hypothese, die aussagt, daß größere Teams mehr Zeit benötigen als kleine, durchaus Bestand zu haben.

Die Autoren geben an, daß noch weitere Beobachtungen gemacht werden müssen, sowie Fragen noch zu beantworten sind, die erst im Verlauf des Experiments auftraten. Fragen, die beispielsweise die Optimalität von Verfahren oder Abstufungen zwischen der Leistungsfähigkeit dieser Verfahren betreffen.

Im Vergleich schneiden $2sK$ und $2sN$ etwa gleich ab, wobei die $2sK$-Verfahren allerdings mehr als doppelt soviel Zeit wie $2s * 2pN$ benötigen. Das deutet darauf hin, daß Korrekturphasen im Vergleich zu Besprechungen und Vorbereitungen sehr zeitaufwendig sind. Aufgrund des geringen Unterschieds zwischen den Effektivitäten beider Verfahren werden $2s * 2pK$-, und $2s * 1pK$-Methoden in Zukunft nicht weiter verfolgt.

Die noch zu bestimmende Größe A, des Aufwands für eine Inspektion läßt sich offenbar durch Vermindern der Größe T' von 4 auf 2 Personen ohne gravierende Effektivitätsverluste erhöhen.

7 Einordnung und Beurteilung

Das vorliegende Experiment erfüllt formal die Kriterien eines klassischen Versuchs in den Naturwissenschaften: die drei Teile Aufbau, Durchführung und Bewertung finden auch hier ihre Entsprechung.

Das betrachtete Versuchsobjekt ist nun allerdings kein physikalischer oder medizinischer Vorgang mehr sondern ein Arbeitsprozess, der Bewertungen über bestimmte Methoden in der Softwareentwicklung ermöglichen soll. Dazu wurde eine eindeutig bestimmte Menge von Variablen variiert, während die sich bei menschlichen Arbeitsweisen unterscheidenden Merkmale als möglichst statistisch gleichverteilt angenommen wurden.

Eine Übertragung des Versuchs auf den Bereich der Beriebs-, oder Sozialwissenschaften sollte daher keine Schwierigkeit bereiten.

Problematisch ist allerdings die Durchführung des Experiments. So wird im weiteren Verlauf etwa ganz auf 2-fach Sitzungen *mit* Korrekturphase wie auch auf $1s*1p$-Verfahren verzichtet, obgleich erst ca. die Hälfte der Inspektionsserie bearbeitet wurde. Als Begründung wird angeführt, daß diese Verfahren zu kostenintensiv seien. Ein solches Vorgehen ist inkonsequent bezüglich des angestrebten Versuchsziels ein objektives Maß für die Kosten der Verfahren erst noch zu ermitteln.

Die Gleichverteilung der Verfahren wird beeinflußt durch das anfängliche Auslassen von $1s$-Verfahren, die erst im nachhinein bearbeitet wurden. Eine fest vorgegebene Anzahl von Inspektionen pro Verfahren könnte die gewünschte Gleichverteilung gewährleisten und darüberhinaus noch die im letzten Absatz angesprochene Inkonsequenz bereinigen. Das erforderte natürlich, daß *alle* Einzelinspektionen durchgeführt werden müßten, was von vornherein als zu kostengefährdend eingeschätzt wurde.

Literatur

[1] A. Porter, H. Siy, C.A. Toman, L.G. Votta. An Experiment to Assess the Cost-Benefits of Code Inspections in Large Scale Software Development. *Research Paper, 17th International Conference on Software Engineering , Seattle, April 1995*, 6. September 1994. [ca. 27% des Experimentumfangs]

[2] A. Porter, H. Siy, C.A. Toman, L.G. Votta. An Experiment to Assess the Cost-Benefits of Code Inspections in Large Scale Software Development. *3rd Symposium on the Foundations of Software Engineering, Washington, Oktober 1995*, 6. März 1995. [ca. 53% des Experimentumfangs]

[3] L.G. Votta, S.A.V. Wiel. Does every Inspection need a Meeting? *Proceedings of ACM SIGSOFT '93 Symposium on Foundations of Software Engineering*, S. 107-114. Association for Computing Machinery, Dez. 1993.

[4] K.P. Burnham, W.S. Overton. Estimation of the Size of a closed Population when Capture Probabilities vary among Animals. *Biometrika*, Nr. 65, 1978, S. 625-633.

[5] S.A.V. Wiel, L.G. Votta. Assessing Software Design using Capture-Recapture Methods. *IEEE Trans. Software Eng.*. SE-19, S. 1045-1054, November 1993.

[6] S. Siegel, N.J. Castellan. *Nonparametric Statistics For the Behavioral Sciences*. McGraw-Hill Inc., New York, 2.Ausgabe 1988.

Die Aussagekraft von Experimenten oder "Sind Flußdiagramme besser als Pseudokode ?"

Ulrich Veigel

Zusammenfassung

Flußdiagramme dienten seit Erfindung des Computers lange Zeit als unumstrittene Mittel bei der Programmerstellung und Dokumentation. In zwei Untersuchungen in den 70er und 80er Jahren sollte ihr tatsächlicher Nutzen experimentell analysiert werden. In dem Vortrag wird deutlich, daß die Ergebnisse dieser Experimente für Softwareprojekte realer Größe kaum Relevanz besitzen. Zur Verdeutlichung wird das Kriterium der 'externen Gültigkeit' vorgestellt, das meist der für ein Experiment noch wichtigeren internen Gültigkeit widerspricht.

1 Einleitung

1.1 Einordnung des Vortrags in das Thema des Seminars

In den Vorträgen zuvor wurde uns die wissenschaftliche Herangehensweise an die Gestaltung und Auswertung von Experimenten vorgestellt.
In diesem Vortrag wird gezeigt, wie eine Lehrmeinung, nämlich, daß Flußdiagramme wichtige Hilfsmittel bei der Programmerstellung und beim Verstehen fertiger Programme sind, in Experimenten überprüft wird. Die beiden Untersuchungen, die hier betrachtet werden, führen zu gegensätzlichen Ergebnissen. Die erste Untersuchung unter der Leitung von Ben Shneiderman aus dem Jahre 1977 führt zur grundsätzlichen Ablehnung von Flußdiagrammen, während David A. Scanlan zwölf Jahre später einen signifikanten Nutzen feststellt. Doch beide Experimente besitzen keine externe Gültigkeit, da nur kleine Programme verwendet wurden. Auf größere Softwareprojekte von mehreren tausend Zeilen lassen sich die Ergebnisse daher nicht übertragen.

1.2 Gliederung der Vortrags

1. Untersuchung von Ben Shneiderman mit dem Ergebnis: Flußdiagramme sind nutzlos
2. Vorstellung der externen Gültigkeit und Würdigung des Shneiderman-Experiments
3. Untersuchung von David Scanlan mit dem Ergebnis: Flußdiagramme sind durchaus sinnvoll

2 Untersuchung von Ben Shneiderman 1977

2.1 Vorbemerkungen

Im Jahre 1977 veröffentlichten die vier amerikanischen Wissenschaftler Shneiderman, Mayer, McKay und Heller die Ergebnisse von fünf Experimenten, die sie durchführten zur Frage, wie nützlich detaillierte Flußdiagramme als Hilfsmittel in der Programmierung sind. Aus heutiger Sicht fällt es schwer, sich die Bedeutung vorzustellen, die Flußdiagramme in der Anfangszeit des Programmierens hatten. Seit der Einführung von Computern wurden sie genutzt. 'Coding begins with the drawing of a flow diagram', eine Aussage von Goldstein und von Neumann (1947), den Erfindern des Computers, war eine gängige Lehrmeinung. Ende der siebziger Jahre wurden Flußdiagramme für etwa drei Viertel aller Fortran-Softwareprojekte eingesetzt, in einem Drittel sogar sehr intensiv. Doch Flußdiagramme wurden heiß diskutiert. Arom findet sie nutzlos zum Fehlerfinden. Weinberg meint, Flußdiagramme nützten höchstens dem Programmierer, anderen jedoch nicht. Ledgard und Chmura gehen noch einen Schritt weiter, sie stellen gar die These auf, daß Flußdiagramme vom funktionalen Verständnis ablenkten. Und Brooks nennt sie einen völlig überbezahlten Teil der Programmdokumentation.

2.2 Zu allen Experimenten

Shneiderman führt fünf verschiedene Experimente durch: Eines zur Programmerstellung, zwei zum Verstehen eines Programms, eines zum Verstehen und Debugging und ein letztes zur gezielten Änderung in einem Programm. Jedes dieser Experimente wird einmal durchgeführt. Die Testpersonen sind Studenten, die in einem Experiment vergleichbare Vorkenntnisse besitzen. Als Programmiersprache wird Fortran verwendet, eine in den siebziger Jahren sehr gebräuchliche Sprache. Die Experimente werden in der Form von Klausuren bzw. Tests geschrieben.

2.3 Experiment I: Schreiben eines Programms

Die teilnehmenden Studenten nehmen an einem Fortran-Einführungskurs teil. Im Experiment erhalten sie eine Aufgabe gegeben, ein bestimmtes Programm zu schreiben. Die Studenten werden in zwei Gruppen aufgeteilt: Die erste besteht aus 45 Personen, die neben dem Programm auch ein Flußdiagramm schreiben müssen. Das Programm wird mit 25 Verrechnungspunkten berücksichtigt, das Flußdiagramm mit 15. Die zweite Gruppe (28 Studenten) muß nur ein Programm abliefern. Allen Teilnehmern steht unbegrenzt viel Zeit zur Verfügung.
Ergebnis: Die Gruppe, die auch ein Flußdiagramm abliefern muß, erreicht im Durchschnitt 94 Prozent der möglichen Punkte (dabei 13,1 der 15 Punkte für das Flußdiagramm). Die zweite Gruppe erreicht durchschnittlich 95 Prozent der möglichen Verrechnungspunkte. Ein Unterschied zwischen beiden Gruppen sei somit nicht erkennbar, schließen die Autoren. Allerdings kann man sich aufgrund dieser Angaben ausrechnen, daß die erste Gruppe 98 Prozent der möglichen Punkte für das Programm erhält.

2.4 Experiment II: Verstehen eines Programms

Wieder
bilden Studenten aus einem Fortran-Einführungskurs die Testpersonen. Die beiden Gruppen, in die die Personen aufgeteilt werden, erhalten je zwei Programme im Fortran-Kode. Das eine besteht aus 27 Befehlen, das zweite aus 24. Die erste Gruppe, bestehend aus 25 Personen, erhält zusätzlich zum ersten Programm ein Flußdiagramm, die zweite Gruppe (28 Personen) zum zweiten Programm. Allen Teilnehmern werden dieselben Verständnisfragen gestellt. Aufgabe ist es, zu gegebenen Inputs die Ausgaben der Programme zu bestimmen, wobei die Antworten entweder richtig oder falsch sind. Wieder steht den Testpersonen unbegrenzt viel Zeit zur Abgabe der Lösungsblätter zur Verfügung.
Ergebnis: Die Gruppe, die das Flußdiagramm für das zweite Programm erhält, bekommt mehr Punkte bezüglich beider Programme: 97,0 Prozent der Punkte für das erste und 94,4 Prozent der Punkte für das zweite Programm. Die andere Gruppe erhält 94,4 Prozent der Punkte für das erste und 89,6 Prozent der Punkte für das zweite Programm. Wir können dem Ergebnis entnehmen, daß das zweite Programm wohl etwas schwerer zu verstehen war als das erste. Die zweite Gruppe scheint, wohl zufällig, etwas bessere Leistungen abzuliefern. Die Autoren sind über das Ergebnis etwas überrascht, da sie extra viele Sprünge in die Programme eingebaut hätten, um so dem Flußdiagramm einen Vorteil zu verschaffen.

2.5 Experiment III: Verstehen eines Programms und Debuggen

An diesem Experiment nehmen Studenten aus einem Fortran-Fortgeschrittenen-Kurs teil. Dabei werden zwei Gruppen unterschieden, die unterschiedliche Übung mit Flußdiagrammen besitzen. Die erste Gruppe (Non-Flow-Chart=NFC, 43 Personen) nutzte normalerweise keine Flußdiagramme, die Mitglieder der zweiten Gruppe (Flow-Chart=FC, 27 Personen) sind es gewohnt, mit Flußdiagrammen zu arbeiten. Beide Gruppen werden in drei Untergruppen aufgeteilt. Da die beiden Gruppen FC und NFC nicht miteinander vergleichbar sind, können wir dieses Experiment auch als zwei Experimente mit unterschiedlichen Testpersonen betrachten. Alle Teilnehmer erhalten ein Fortran-Listing mit einem Umfang von 81 Zeilen, davon 43 für das Hauptprogramm, 23 für ein Unterprogramm, die restlichen Zeilen sind Kommentar im Hauptprogramm. Die erste Untergruppe erhält ein Mikro-Flußdiagramm auf vier Seiten, die zweite ein gröberes Makro-Flußdiagram (eine Seite). Die dritte Untergruppe muß die Aufgabe ohne Flußdiagramm lösen.
Im Programm wurden drei Fehler versteckt, von denen einer zu einer falschen Ausgabe führt. Der Test gliedert sich in zwei Teile: Im ersten Teil müssen alle Fehler gefunden werden, wobei den Testpersonen die Zahl der Fehler nicht genannt wird. Gefundene Fehler müssen behoben werden. Nachdem die Lösungsblätter eingesammelt sind, werden die Fehler genannt und müssen jetzt gezielt beseitigt werden. Außerdem müssen noch elf Verständnisfragen im Multiple-Choice-Verfahren beantwortet werden. Die Gruppe NFC hat für den ersten Teil 40 Minuten, für den zweiten 20 Minuten Zeit. Die Gruppe FC 50 Minuten für den ersten und 30 Minuten für den zweiten Teil.
Ergebnis: Es überrascht nicht, daß die Gruppe ohne Erfahrung mit Flußdiagrammen sowohl beim Verstehen als auch Debuggen die besten Ergebnisse erzielt, wenn sie kein Flußdiagramm benutzt. Die Teilnehmer der Untergruppe ohne Flußdiagramm erhalten durchschnittlich 52 bzw. 12 Verrechnungspunkte für die beiden Teile, die Untergruppe mit Mikroflußdiagramm 46 bzw 4 Punkte und die mit Makro-Flußdiagramm nur 34 Punkte im ersten Teil, dagegen 11 im zweiten. Die zweite Gruppe erhält die besten Werte, die mit der ersten Gruppe nicht vergleichbar sind, in der Untergruppe, die das Mikroflußdiagramm einsetzen konnte: 76 Punkte für den ersten Teil, 62 für den zweiten. Die Untergruppe mit Makro-Flußdiagramm (55 bzw. 42 Punkte) und ohne Flußdiagramm (53 bzw. 42 Punkte) schließen nahezu gleich ab. Die Autoren betonen, daß beide Gruppen ihren Vorkenntnissen gemäße Ergebnisse erzielen. Der geringe Nutzen von Flußdiagrammen wird auch

durch eine Zusatzfrage an die Teilnehmer ausgedrückt, in der diese aussagen, Flußdiagramme seien nicht sehr hilfreich. Wir können jedoch erkennen, daß in der zweiten Gruppe, die wohl eher geeignet ist, die Nützlichkeit von Flußdiagrammen zu untersuchen, ein eindeutiger Vorteil für die Verwendung der detaillierten Flußdiagramme zu erkennen ist.

2.6 Experiment IV: Programmänderung

Es nehmen Studenten von zwei Universitäten an diesem Experiment teil. Die erste Gruppe (33 Personen) hat lediglich eine Einführung in Flußdiagramme genossen (NFC), die zweite (37 Studenten) ist in deren Verwendung geübt (FC). Alle Teilnehmer erhalten ein 48-Zeilen-Fortran-Programm, das zusätzlich 27 Kommentarzeilen besitzt, wovon 23 als Block am Anfang stehen. Außerdem erhalten die Teilnehmer einen Beispiel-Output und drei Beschreibungen von Veränderungen, die durchgeführt werden sollen. Nachdem es Schwierigkeiten mit einer der Aufgaben gibt, werden allerdings schließlich nur zwei davon bewertet. In beiden Gruppen werden drei Testgruppen gebildet: Eine erhält ein einseitiges Makro-Flußdiagramm, die zweite ein dreiseitiges Mikro-Flußdiagramm und die dritte kann lediglich mit dem Programm arbeiten. Nach einer kurzen Einführung steht eine Bearbeitungszeit von 45 Minuten zur Verfügung. Dabei wird die Zeit pro Modifikation von den Teilnehmern selbst erfaßt.
Ergebnis: In der Gruppe FC erweist sich für die erste Modifikation das Makro-Flußdiagramm (88 Verrechnungspunkte) wenig besser als das Mikro-Flußdiagramm (87), während die Gruppe ohne Flußdiagramm mit 73 Verrechnungspunkten deutlich abfällt. Anders sieht es für diese Gruppe bei der zweiten Änderung aus: Dort liegt die Gruppe ohne Flußdiagramm (85 Punkte) vor der mit Mikro- und Makro (81 bzw. 77 Punkte).
Die drei Testgruppen in der Gruppe NFC schließen für die erste Änderung identisch ab (je 77 Punkte), während für die zweite Änderung die Gruppe mit dem Mikro-Flußdiagramm (71 Punkte) deutlich vor der Gruppe ohne Diagramm (64 Punkte) und der mit Makro-Flußdiagramm (59 Punkte) liegt.
Bei der Betrachtung der Bearbeitungsdauer werden keine bedeutenden Unterschiede sichtbar. Die Autoren betonen, daß die Unterschiede innerhalb der Gruppen anscheinend sehr groß waren. Das insgesamt gute Abschneiden erklären sie mit dem ausführlichen Kommentarblock.

2.7 Experiment V: Programmverstehen

An diesem Experiment nehmen 58 Studenten aus einem Fortran-Ferienkurs teil, in dem auch Flußdiagramme eingeführt wurden. Als Programm wird ein Array-Misch-Algorithmus vorgelegt. Eine erste Testgruppe erhält lediglich den Fortran-Kode (23 Zeilen), eine zweite lediglich ein detailliertes Flußdiagramm (eine Seite) - die dritte Gruppe erhält sowohl Programmlisting als auch Flußdiagramm. Die Aufgabe ist es, Fragen zu beantworten zum Bereich Handsimulation, also zu gegebenem Input den Output zu berechnen, und zur Interpretation. 25 Minuten steht als Bearbeitungszeit zur Verfügung.
Ergebnis: Die Gruppe, die lediglich das Programm zur Verfügung hat, schließt sowohl bei der Handsimulation (57,8 Punkte) als auch bei den Verständnisfragen (62,4 Punkte) am besten ab. Die Gruppe mit beiden Darstellungsformen liegt knapp dahinter auf Platz zwei bei der Handsimulation (56,9 Punkte), während dort die Gruppe, die nur das Flußdiagramm hat, lediglich auf 48,5 Punkte kommt. Bei der Interpretation liegen die beiden letztgenannten Gruppen etwa gleich: 50,0 Punkte für die mit beiden Darstellungsformen, 51,2 Punkte für die Gruppe nur mit Flußdiagramm. Die Autoren erklären das Resultat damit, daß das detaillierte Flußdiagramm etwa die gleiche Information wie das Listing enthält. Anscheinende störe die redundante Information das Verständnis. Zur Handsimulation sei das Listing in jedem Fall hilfreich.

2.8 Ergebnis der ersten Untersuchung

Nach Meinung der Autoren zeigen ihre Experimente für Flußdiagramme keine sichtbaren Vorteile. Teilweise wurde sogar das Gegenteil deutlich, da diese gegenüber einem Listing wohl etwas zu wenig Details enthalten. Die Autoren äußern die Vermutung, daß moderne Programmierkonzepte keine Flußdiagramme benötigen. Sie hoffen auf weitere Experimente, in denen diese These anhand komplexerer Probleme bewiesen wird.

3 Problematik externer Gültigkeit

3.1 Übersicht

Die Notwendigkeit interner Gültigkeit (internal validity) eines Experiments haben wir in vorangegangenen Vorträgen kennengelernt. Interne Gültigkeit ist ein Kriterium, wie ein Experiment aufgebaut ist. Man kann interne Gültigkeit gut kontrollieren und sicherstellen.
Externe Gültigkeit (external validity) dagegen ist das Maß, inwiefern die Ergebnisse eines Experiments auf andere Personen, Umgebungen und in andere zeitliche Zusammenhänge übertragen werden können. Ziel ist es, aus der begrenzten Information, die ein Experiment liefert, eine umfassende Aussage auf die reale Welt zu machen.
Es gibt drei große Gefahren für die externe Gültigkeit:

1. Population validity:
   Die Auswahl der Testpersonen ist nicht repräsentativ.

2. Ecological validity:
   Die Testbedingungen bzw. die Aufgabenstellung können nicht auf die reale Welt übertragen werden.

3. Temporal validity:
   Die Ergebnisse sind auf verschiedene Weise zeitabhängig und können nicht in die Zukunft übertragen werden.

3.2 Erste Gefahr: Population validity

Kann das Ergebnis aus einem Experiment mit 20 Psychologie-Studenten auf alle Psychologie-Studenten übertragen werden oder gar auf alle Studenten oder alle Menschen ?
Untersuchungen zeigen, daß Psychologie-Studenten bzw. Albino-Ratten die am häufigsten verwendeten Versuchsobjekte sind. Etwa 80 Prozent aller Tests zum menschlichen Verhalten verwenden College Studenten.
Die richtige Auswahl der Testpersonen findet in zwei Schritten statt:
Eine bestimmte Gruppe steht für das Experiment zur Verfügung. Dies können Mitarbeiter eines Unternehmens sein oder Studenten in einem bestimmten Seminar. Aus dieser experimentell verfügbaren Bevölkerung werden die Testpersonen ausgewählt. Diese erste Generalisierung ist vergleichsweise unproblematisch, wenn die Auswahl zufällig erfolgt und ausreichend viele Testpersonen zum Experiment herangezogen werden. Wählt man von allen Studenten einer Universität 50 zufällig aus, die dann an einem Experiment teilnehmen, so kann das Ergebnis auf alle Studenten dieser Universität übertragen werden. Die entscheidende Frage für diese erste Generalisierung ist, ob alle theoretisch verfügbaren Personen auch beim Experiment mitmachen können oder wollen. Scheidet eine bestimmte Gruppe von vorneherein aus, weil z.B. nur die Mitarbeiter der Tagesschicht zur Verfügung stehen, so kann das Ergebnis ggf. nicht auf die Mitarbeiter der Nachtschicht übertragen werden.
Die zweite Generalisierung ist kritisch: Von der Gruppe der experimentell verfügbaren Personen wird jetzt auf alle Zielpersonen geschlossen. Ist eine Aussage, die für alle Angestellten eines Unternehmens zutrifft, auch für alle Arbeitnehmer dieser Branche oder gar für alle arbeitenden Menschen gültig ? Für unsere Fragestellung heißt das: Muß ein Ergebnis, das möglicherweise auf alle Informatik-Studenten übertragen werden kann, auch für alle, die professionell Software entwickeln Gültigkeit besitzen, die aber vielleicht nicht den Hintergrund eines Informatik-Studiums besitzen ? Es kann sein, daß deren andere Denkweise zu einem deutlich abweichenden Ergebnis bei derselben Fragestellung geführt hätte. Ein anderes Beispiel ist, ob man Ergebnisse, die aus Experimenten mit Pascal-Programmierern gewonnen werden, auch auf C-Programmierer übertragen kann: Vielleicht sind die einen größere Ästheten und etwas gemütlicher, während die anderen verschlungene For-Schleifen bevorzugen.
Augenfällig wird die Problematik, wenn man sich vorstellt, man wollte ein Experiment bzgl. Gewaltbereitschaft, das mit Skinheads durchgeführt wurde, auf die Gesamtbevölkerung übertragen.

3.3 Zweite Gefahr: Ecological validity

Die ökologische Gültigkeit ist das Maß, auf das die Ergebnisse einer Untersuchung generalisiert werden können über die Testfragestellungen und Umweltbedingungen hinaus. Ist eine bestimmte Anordnung der Hilfsmittel, ein bestimmter Ort oder Experimentator wichtig für das Eintreten der Ergebnisse ? Kann von Laborexperimenten auf die reale Welt geschlossen werden ?

3.3.1 Multiple-Treatment-Interference

Der mehrfache Behandlungs-Effekt drückt aus, daß es Auswirkungen einer vorherigen Teilnahme einer Person an einem Experiment geben kann auf die Teilnahme derselben Person in einem zweiten Experiment. In jedem Falle muß man berücksichtigen, ob die Ergebnisse des zweiten Experiments nur unter Berücksichtigung

des ersten Gültigkeit besitzen. Dabei sind zwei Problemsituationen denkbar:
Zum einen kann ein Teilnehmer innerhalb eines Experiments mit zwei verschiedenen Testbedingungen konfrontiert werden, die sich gegenseitig beeinflussen. So wurde 1963 von Fox ein Experiment bzgl. der Lesegeschwindigkeit in Abhängigkeit der Schriftart durchgeführt. Eine Gruppe erhielt einen Text in Standard Elite und anschließend in Gothic Elite, eine zweite Gruppe bekam zunächst einen Text in Gothic, dann Standard Elite. Erstaunlicherweise war die Lesegeschwindigkeit in beiden Gruppen höher in der Schriftart, mit der der Test begonnen wurde. Hätte man also keine Kontrollgruppe getestet, so wäre das Ergebnis willkürlich gewesen.
Eine Multiple-Treatment-Interference kann auch zwischen verschiedenen ähnlichen Experimenten auftreten. Die vorherige Teilnahme kann dabei sowohl zu einem Lerneffekt als auch dazu führen, daß man bei einem zweiten Test schlechtere Ergebnisse erzielt als jemand, der noch an keinem derartigen Experiment teilgenommen hat. So wurde einmal ein Experiment durchgeführt, bei dem unzusammenhängende Adjektive einer Testperson vorgetragen wurden. Diese mußte möglichst viele wiederholen. Dabei schlossen Testpersonen deutlich besser ab, wenn sie zum ersten Mal an diesem Experiment teilnahmen.
Auch Versuchstiere werden i.d.R. nur für ein Experiment verwendet.
Man umgeht die Multiple-Treatment-Interference, indem man möglichst Testpersonen findet, die zuvor an keinem Experiment teilgenommen haben.

3.3.2 Hawthorne Effekt

Der Hawthorne Effekt drückt die Tatsache aus, daß eine Testperson weiß, daß sie an einem Experiment teilnimmt. Die Teilnahme an einem Experiment ist vergleichbar mit dem Auftreten vor einer Fernseh- oder Videokamera: Viele Menschen ändern dort ihr Verhalten, werden aufmerksamer und ehrgeiziger. Auch konnte nachgewiesen werden, daß viele Leute höhere Unterwürfigkeit zeigen, wenn sie wissen, daß sie an einem wissenschaftlichen Experiment teilnehmen. Würde man zum Beispiel jemanden auf der Straße ansprechen und auffordern, einige Kniebeugen zu machen, so würde er wohl verständnislos mit dem Kopf schütteln. Fügte man dann jedoch hinzu, dies sei eine große Untersuchung zur Volksgesundheit, so wäre mancher zu einer kurzen Übung bereit.
Für unsere Fragestellung heißt das: Wenn beispielsweise getestet wird, wie gut ein Programm dokumentiert wird, und man sagt dies der Testperson, so wird diese vermutlich besonders großen Wert auf die Qualität der Dokumentation legen. Das Ergebnis läßt sich dann nicht auf Programme der realen Welt übertragen.
Es ist nahezu unmöglich, den Hawthorne Effekt zu umgehen, zumindest mit Experimenten, an denen Menschen teilnehmen. Die wissenschaftliche Ethik gebietet es nämlich, die Testpersonen weitgehend über Ziele und möglichen Rückschlüsse eines Experimentes aufzuklären.

3.3.3 Novelty bzw. Disruption Effekt

Dieser Überraschungs- bzw. Störungseffekt drückt aus, daß es zu bemerkenswerten Ergebnissen kommen kann, wenn die Experimentalbedingungen etwas Neues oder Ungewöhnliches enthalten. So ist es einsichtig, daß aus einer Liste unzusammenhängender Wörter dasjenige am besten gemerkt werden kann, das in einer anderen Farbe geschrieben ist. Es wäre in diesem Falle völlig falsch, das Ergebnis auf das Wort selbst zurückzuführen. Andererseits haben Experimente mit der Einführung neuer Lernkonzepte an Schulen gezeigt, daß diese dort wesentlich besser angenommen wurden, wo man gewohnt war, mit neuen Konzepten zu arbeiten.
Zur Verhinderung dieses Effekt sollte ein Experiment keine außergewöhnlichen Elemente enthalten, es sei denn, genau diese Elemente sind Gegenstand des Experiments.

3.3.4 Experimenter Effekt

Dieser Effekt tritt ein, wenn der Veranstalter eines Experiments bestimmte Erwartungen an die Ergebnisse hat und - freiwillig oder unfreiwillig - auf bestimmte Ziele hinarbeitet. So könnte ein Arzt, der ein neues Medikament an einer Patientengruppe ausprobiert, während eine Kontrollgruppe nur ein Placebo-Präparat erhält, unterschwellig, z.B. durch freundliches Zureden, die Patienten mit dem neuen Medikament fördern und so zu einem verfälschten Ergebnis beitragen.
Abhilfe schafft man hier, indem nicht nur der Patient nicht davon in Kenntnis gesetzt wird, zu welcher Testgruppe er gehört, sondern indem auch der Experimentator nicht weiß, welches Medikament er gerade verabreicht. Für unsere Fragestellung erscheint dieser Effekt vermeidbar, wenn Aufgabenstellungen und Bewertungen sauber und korrekt sind und alle Testpersonen gleich behandelt werden.

3.3.5 Pretesting Effekt

Dieses Problem drückt die Tatsache aus, daß Personen, die in einem vorherigen Experiment schon mit einer bestimmten Frage konfrontiert wurden, möglicherweise beim zweiten Mal eine andere Antwort geben. Der Pretesting Effekt soll nicht unbedingt vermieden werden, da es ja auch interessant sein kann zu erfahren,

welche Antwort eine Person bei einem zweiten Experiment gibt. Doch er muß in jedem Fall berücksichtigt werden.
Ein noch wichtigerer Unterschied besteht zwischen freiwilligen und unfreiwilligen Teilnehmern. Menschen, die bei einem Experiment freiwillig mitmachen, sind bei weitem motivierter als solche, die dazu gezwungen werden.

3.4 Dritte Gefahr: Temporal Validity

Die zeitliche Gültigkeit eines Experiments ist das Maß, inwieweit die Ergebnisse bzgl. der Zeit verallgemeinert werden können.
So gab es in den siebziger Jahren zwei Studien, die besagten, daß zwischen 75 und 90 Prozent aller Untersuchungen in bestimmten Psychologie-Zeitschriften aufgrund nur einer Sitzung stattfanden. Da aufgrund dieser Untersuchungen allgemeine Aussagen getroffen werden, scheinen die Untersucher anzunehmen, daß die Ergebnisse über die Zeit unverändert bleiben.
Dies trifft nicht notwendiger Weise zu: In einem Experiment 1964 wurden Personen gefragt, welche der zehn vorgeschlagenen Jobs ihnen am besten gefallen würden. Zwei, die eine Testperson etwa gleich einschätzte, wurden näher untersucht. Die Testperson sollte sich zwischen diesen beiden entscheiden. Diese Frage mußte 4, 15 bzw. 90 Minuten später noch einmal beantwortet werden. Dabei kam es im Durchschnitt zu völlig unterschiedlichen Ergebnissen. Nach vier Minuten war der gewählte Job unbeliebter, nach 15 deutlich beliebter, und nach 90 Minuten wurde wieder das Anfangsniveau erreicht.
Man unterscheidet verschiedene Gefahren für die zeitliche Gültigkeit:

3.4.1 Saisonale Schwankungen

Es gibt Veränderungen, die regelmäßig in Teilen der Bevölkerung auftreten. Wenn man z.B. das Konsumverhalten analyiseren würde, so dürfte man nicht das Einkaufsverhalten im Weihnachtsgeschäft auf das ganze Jahr übertragen. Die Zahl der Verkehrsunfälle steigt während der Hauptreisezeit bedeutend an.
Saisonale Schwankungen können zu festen Zeiten eintreten oder aber ergeignisabhängig. Der Tod eines Angehörigen kann nicht vorhergesagt werden, wenn er jedoch eingetreten ist, so laufen immer ähnliche psychologische Prozesse ab.

3.4.2 Zyklische Schwankungen

Zyklische Schwankungen finden in einzelnen Testpersonen statt, nicht in ganzen Bevölkerungsgruppen. Menschen haben einen etwa 24 stündigen Rhythmus mit unterschiedlichen Pulsraten, Temperatur und Arbeit innerer Organe. Der Grad an Aufmerksamkeit ändert sich im Laufe eines Tages.
Bei Tests anspruchsvoller gegen weniger anspruchsvolle Programmier- oder Spezifikationsverfahren sollte darauf geachtet werden, daß Testpersonen so frisch und ausgeschlafen sind, wie sie es bei der regulären Anwendung sind.

3.4.3 Personologische Variation

Unter dem Stichwort Personologische Variation versteht man die Veränderung im Charakter und Geschmack eines Menschen. Politische Ansichten haben sich zum Beispiel mit den Erfahrungen diverser Kriege und Affären über Jahre hinweg verändert. Der Geschmack, welche Kleidung als modern gilt, ändert sich. Im Bereich der Informatik wandeln sich die Programmierstile: Vor Jahren noch war strukturierte Programmeriung in Mode, heute ist objektorientiertes Programmieren das große Schlagwort und in einigen Jahren wird dieses durch eine andere Philosophie abgelöst werden.
Gegen Modeänderungen kann man ein Experiment wohl kaum absichern. Es ist daher wohl unumgänglich, ein Experiment einige Jahre später unter den neuen Fragestellungen der Zeit zu wiederholen.

3.4.4 Beziehung zwischen interner und externer Gültigkeit

Man könnte nun anhand obiger Auflistung ein Experiment so planen, daß es externe Gültigkeit besitzt. Verschiedene Gruppen von Personen werden zu verschiedenen Zeiten unter verschiedenen Bedingungen getestet. Doch leider scheinen sich interne und externe Gültigkeit gegenläufig zu verhalten. Die interne Gültigkeit wird dadurch erhöht, daß die Testpersonen nur aus einer bestimmten Personengruppe stammen. Ein Experiment wird am besten in einem geschlossenen Laboratorium durchgeführt, so daß immer gleiche Testbedingungen garantiert sind. Die Fragen sind standardisiert und werden am besten durch einen Computer gestellt. Auch fordert die interne Gültigkeit, daß das Experiment zu möglichst einem Zeitpunkt stattfindet.
Bei der Abschätzung, ob ein Experiment externe Gültigkeit besitzt, ist es wichtig zu berücksichtigen, welche Aussagen von dem Experiment erwartet werden. So wurde einmal ein Experiment durchgeführt, bei denen die Testpersonen ihre Abneigung gegen einen bestimmten Künstler zum Ausdruck bringen sollten, indem sie, während sie an ihn dachten, mit Hilfe eines Schalters Schocks auslösen sollten. Bei manchen Testpersonen lagen scheinbar zufällig Waffen auf dem

Tisch. Diese Testpersonen reagierten deutlich agressiver als andere, bei denen die Waffen fehlten. Dieses Experiment besitzt strenggenommen keine externe Gültigkeit, da solche Situationen in der Realität kaum vorkommen. Doch kann man immerhin schließen, daß, unter bestimmten Umständen, der Anblick einer Waffe zu einer Verstärkung der Abneigung gegen bestimmte Personen führt.
Bei einem Experiment jedoch, in dem neue Lernverfahren, die an Schulen eingeführt werden sollen, getestet werden, ist die externe Gültigkeit von großer Bedeutung.
Für den Bereich der Informatik gilt dies ebenfalls: Die Ergebnisse sollen auf die reale Welt übertragen werden.

3.4.5 Zusammenfassung externe Gültigkeit

Die Übertragbarkeit des Ergebnisses eines Experiments ist durch vielfältige Faktoren gefährdet: Durch die falsche Auswahl der Testpersonen, durch falsche Testumstände und Aufgabenstellungen und dadurch, daß man die zeitliche Gültigkeit nicht berücksichtigt. Obwohl es sehr wichtig ist, ein extern gültiges Experiment durchzuführen, muß doch der erste Schwerpunkt darauf gelegt werden, das Experiment intern gültig zu machen. Unglücklicherweise können sich die Kriterien für beide Ziele widersprechen. Daher sollte man sich erst dann um externe Gültigkeit bemühen, nachdem man einen Effekt mit einer intern gültigen Studie ermittelt hat.

3.4.6 Externe Gültigkeit des Shneiderman-Experiments

Nach dieser allgemeinen Darstellung der externen Gültigkeit ist es interessant, die obigen von Ben Shneiderman durchgeführten Experimente unter dieser Fragestellung zu betrachten:
Die Population Validity ist nur eingeschränkt sichergestellt: Es werden lediglich College-Studenten getestet. Diese werden nicht zufällig ausgewählt. Als Programmiersprache wird nur Fortran verwendet. Die Ergebnisse sind also bestenfalls auf alle College-Studenten, die kleine Programme in Fortran programmieren, zu übertragen.
Die Ecological Validity sieht besser aus: Multiple Treatment, Novelty und Pretesting Effekt scheinen keinen Einfluß zu besitzen. Der Hawthorne Effekt läßt sich nie vermeiden. Da die Studenten jedoch mehrere Klausuren schrieben und das Experiment auch nur in dieser Form geschrieben wurde, scheint er zu keinen Verfälschungen geführt zu haben.
Von zeitlicher Gültigkeit kann kaum die Rede sein: Der Test fand Mitte der siebziger Jahre statt. Alle Veränderungen hinsichtlich strukturierter und objektorientierter Programmierung wurden noch nicht berücksichtigt. Die Studenten waren Anfänger bzw. Fortgeschrittene. Die Erfahrung, die ein Programmierer im Laufe der Jahre sammelt, wurde überhaupt nicht berücksichtigt. Die größten Probleme der ersten Untersuchungen sind, daß die Beispielprogramme sehr klein und damit praxisirrelevant waren und daß meist unbegrenzt viel Zeit zur Verfügung stand, was zu sehr guten Ergebnissen in beiden Testgruppen führte.

4 Untersuchung von David A. Scanlan 1989

4.1 Vorbemerkungen

Während er Studenten ausbildet, stellt Scanlan fest, daß diese Flußdiagramme Pseudokode vorziehen. Er kennt vorherige Untersuchungen, u.a. auch die von Ben Shneiderman. Doch er kritisiert diese hart. Ein Hauptangriffspunkt ist die Tatsache, daß Shneiderman den Zeitfaktor völlig vernachlässigt. 'Zeit, die gebraucht wird, um einen Algorithmus zu verstehen, ist sensitivste und wichtigste Maßzahl', gibt Scanlan als seinen Leitsatz an. Auch kritisiert er an Shneiderman, daß dort teilweise exakter Input bzw. Output zum Verständnis nötig waren und diese Information ganz einfach nicht aus Flußdiagrammen heraus gewonnen werden konnte.
Scanlan geht daran, ein intern voll gültiges Experiment durchzuführen. Seine Hypothesen sind, daß Flußdiagramme weniger Zeit zum Verstehen benötigen und zu weniger Fehler führen. Auch gäben sie mehr Sicherheit beim Verständnis, reduzieren die Zeit, um Fragen zu beantworten, und die Zahl, wie oft ein Algorithmus angeschaut werden muß.
Wie wir sehen werden, ist die externe Gültigkeit des Scanlan-Experiments nicht sichtbar höher als die von Shneidermans.

4.2 Das Experiment

Scanlan läßt 82 Personen an dem Experiment teilnehmen. Diese erhalten einige Tage vor Durchführung ausführliche Instruktionen. Auch können vor und während des Experimentes alle Fragen gestellt werden, da Scanlan das nicht völlige Verständnis des Ablaufs eines Experiments für ein großes Problem hält.
Die Testteilnehmer besitzen Kenntnisse in Pascal, Cobol und Softwaretechnik. Sie sind meistens Fortgeschrittene, auch einige Profis sind darunter. Vor Teilnahme an dem Experiment müssen sie einige einfache Fragen richtig beantworten, um eine gewisse Grundqualifikation nachzuweisen.
Als Testmaterial werden drei Algorithmen verwendet.

Diese sind verschachtelte if..then..else-Ausdrücke. Die Semantik ist nicht sehr sinnvoll. In Abhängigkeit bestimmter Prädikate ('crispy', 'hard', 'green',...) werden bestimmte Tätigkeiten durchgeführt ('steam', 'fry', 'bake',...). Für jeden Algorithmus werden die Prädikate und Tätigkeiten mit Hilfe eines Zufallsgenerators mit zwei verschiedenen Belegungen gefüllt. Diese werden Muster A und Muster B genannt.
Jede Testperson erhält alle drei Algorithmen als Flußdiagramm in einem der beiden Muster und im jeweils anderen Muster alle drei Algorithmen als formatierter Pseudokode. Die Algorithmen heißen 'einfach' (14 Zeilen Pseudokode, zwei if-Ausdrücke), 'medium' (24 Zeilen Pseudokode, vier if-Ausdrücke) und 'komplex' (34 Zeilen Pseudokode, sechs if-Ausdrücke).
Als Testgerät verwendet Scanlan einen IBM PC. Die Eingaben erfolgen über Tastatur, Ausgaben auf den Bildschirm und ein spezielles Display, auf dem die Algorithmen angezeigt werden. Ein spezielles Sprachausgabeteil führt den Benutzer durch das Experiment.
Der PC registriert für jeden Anwender die Zeit (in Sekunden), die jede Testperson einen jeden Algorithmus anschaut, außerdem die Dauer für die Beantwortung einer jeden Frage. Desweitern wird gespeichert, wie oft ein bestimmter Algorithmus angeschaut wird, alle Antworten und die Sicherheit, mit denen die Antworten eingegeben werden.
Das Experiment findet in einem schalldichten Raum im psychologischen Institut statt. Die Teilnehmer haben unbegrenzt viel Zeit, so daß die Richtigkeit der Antworten eine untergeordnete Rolle spielt. Im Durchschnitt benötigt eine Testperson etwa zwei einhalb Stunden.
Das Ergebnis zeigt signifikante Vorteile für die Verwendung von Flußdiagrammen. Die durchschnittliche Zeit zur Beantwortung einer Frage bzgl. des einfachen Algorithmus beträgt 7,8 Sekunden, wenn der Algorithmus als Flußdiagramm dargestellt wird, 13,4 Sekunden, wenn er als Pseudokode vorliegt. Beim Medium-Algorithmus 6,1 Sekunden bzw. 11,7 Sekunden und beim komplexen Algorithmus 6,3 Sekunden bzw. 15,8 Sekunden. Scanlan meint daraus zu erkennen, daß Flußdiagramme mit zunehmender Komplexität der Algorithmen nützlicher werden. Dabei nimmt auch die Fehlerrate in der Pseudokode-Gruppe deutlich zu.

4.3 Würdigung des Experiments

Vergleichen wir das Experiment mit Ben Shneidermans, so fällt der Aufwand auf, den Scanlan betreibt, um die interne Gültigkeit sicherzustellen. Hier kann man ihm wohl nichts vorwerfen. Eingeschränkt hat er die Fragestellung, da lediglich fertige Algorithmen verstanden werden müssen, während Shneiderman auch Schreiben und Änderung berücksichtigte.
Doch welche externe Gültigkeit hat Scanlans Experiment ? Wir müssen feststellen, daß diese kaum besser ist als bei Shneiderman. Der Hawthorne-Effekt mag etwas verstärkt worden sein, da sich das Experiment deutlicher von Klausuren unterscheidet als Shneidermans Tests. Die Berücksichtigung der Zeiten ist sicher ein Vorteil von Scanlan. Doch der Hauptkritikpunkt bleibt: Die Algorithmen haben mit der Praxis fast nichts zu tun. Zum einen ist selbst der sogenannte komplexe Algorithmus viel zu klein und zum anderen fordert Scanlan von seinen Testpersonen ein fast perfektes Verständnis der Algorithmen, um die Fragen zu beantworten. Ein solches Verständnis ist in der Realität meist nicht nötig, um Änderungen in fertigen Algorithmen durchzuführen.
Selbstverständlich müssen wir desweiteren die zeitliche Gültigkeit kritisieren, daß Flußdiagramme heute aufgrund prozeduraler und objektorientierter Programmierung nicht mehr sinnvoll verwendet werden können.

5 Abschließende Bemerkungen

Wir haben anhand praktischer Beispiele gesehen, daß das Fehlen externer Gültigkeit Experimente praktisch wertlos macht. Interne Gültigkeit ist für Experimente notwendig, doch ohne, daß die Fragestellungen Entsprechungen in der Realität haben, nützt sie nichts. Ein praktisch relevanter Algorithmus müßte eine Länge von 500 bis 1000 Zeilen besitzen. Dieser dürfte dann nicht bis ins kleinste Detail verstanden werden müssen, um die Fragen zu beantworten.

6 Literaturhinweise

Ben Shneiderman, Richard Mayer, Don McKay und Peter Heller: 'Experimental Investigations of the Utility of Detailed Flowcharts in Programming', CACM 20(6), Juni 1977.
David A. Scanlan: 'Structured Flowcharts Outperform Pseudocode: An Experimental Comparison', IEEE Software, pp. 28-36, September 1989.
Larry B. Christensen: 'Experimental Methodology', Allyn and Bacon, Needham Heights, MA, 6th edition, 1994, Kapitel 14.

Natur-, Ingenieur- und Humanwissenschaften: Unterschiede beim experimentellen Arbeiten

Oliver Benke

Zusammenfassung

Im Folgenden soll die experimentelle Vorgehensweise der verschiedenen Wissenschaftsbereiche miteinander verglichen werden, im Wesentlichen anhand eines Beispiels aus der Physik (Millikan-Versuch) und der Betrachtung von Evaluationsverfahren in der Sprachverarbeitung als typischem Teilgebiet der Informatik. Quantitative Verfahren sind in der Informatik noch nicht sehr weit entwickelt, es besteht die Hoffnung, daß die Informatik von im empirischen Bereich weiter entwickelten Wissenschaften lernen kann.

1 Naturwissenschaften: Physik

Bei der Physik handelt es sich um eine der am weitesten entwickelten exakten Wissenschaften. Insofern könnte es lohnend sein, diese in vielen Punkten vorbildliche Wissenschaft genauer zu betrachten.

Die Vorgehensweise in der Physik soll anhand des Beispieles des Millikan-Versuches kritisch beleuchtet werden.

Der auch als Öltröpfchen-Versuch bekannt gewordene Millikan-Versuch wurde 1916 von Robert Andrews Millikan durchgeführt. Dieser Versuch war der klassische Versuch zur Beantwortung der Frage, ob die elektrische Ladung von diskreter oder von kontinuierlicher Natur ist, im Zusammenhang Welle-Teilchen-Dualismus und Teilchenmodell des Elektrons ist er von fundamentaler Bedeutung. Nebenbei wurde mit Hilfe des Millikan-Versuches der Wert der elektrischen Elementarladung verhältnismäßig genau bestimmt.

1.1 Der Millikan-Versuch

Kleine Öltröpfchen aus einem Zerstäuber werden zwischen die Platten eines Kondensators gebracht und durch ein Mikroskop beobachtet (siehe [9]). Auf die Öltröpfchen wirken nun folgende Kräfte:

- Die *Erdanziehungskraft*,
- bedingt durch den Plattenkondensator die *elektrische Kraft*, wahlweise nach oben oder nach unten gerichtet und
- die *Luftreibungskraft*, sofern die Öltröpfchen sich bewegen.

Durch den Austritt aus dem Zerstäuber tragen die Öltröpfchen bereits eine gewisse Ladung, mit Hilfe von zum Beispiel Röntgenstrahlung ist eine Umladung möglich.

Werden in dem Kondensator Fäden gespannt oder vergleichbare Markierungen angebracht, so ist die pro Zeiteinheit zurückgelegte Wegstrecke der Öltröpfchen – in Worten: die Geschwindigkeit v – gut meßbar.

Durch Veränderung einiger Parameter (elektrisches Feld des Plattenkondensators, Ladung der Öltröpfchen) ist es möglich, die Ladung eines einzelnen Öltröpfchens zu berechnen.

Im luftleeren Raum würden die Öltröpfchen gemäß $v = gt$ ständig beschleunigt[12]. Aufgrund der Luftreibungskraft herrscht irgendwann ein Kräftegleichgewicht, d.h. die Beschleunigung, die die Öltröpfchen durch die Erdanziehungskraft erfahren, ist ebenso groß wie die in entgegengesetzter Richtung wirkende Luftreibungskraft. Folglich ist die Beschleunigung Null, die Geschwindigkeit bleibt konstant. Dies wird in der Physik auch als *gleichförmige Bewegung* bezeichnet wird[13].

1.1.1 Das Stokessche Gesetz

Die Luftreibungskraft F_R kann mit Hilfe des Stokesschen Gesetzes

$$F_R = 6\pi\eta r v \tag{3}$$

näherungsweise bestimmt werden. Die obige Formel bedeutet: die Luftreibungskraft ist proportional zum Tröpfchenradius r und zur Geschwindigkeit v. Die Zähigkeit η ist für Luft eine Konstante.

Umso schneller sich ein Tröpfchen bewegt, umso größer wird die entgegengesetzt gerichtete Luftreibungskraft.

Bei der Herleitung des Stokesschen Gesetzes werden folgende Annahmen getroffen:

- Der sich relativ zum Medium mit Geschwindigkeit v und Radius r bewegende Körper ist eine *Kugel*. Dies kann bei – nicht zu kleinen – Öltröpfchen näherungsweise als gegeben angenommen werden, bei sehr kleinen Öltröpfchen hat Millikan merkwürdige Ergebnisse erhalten und auch richtig interpretiert (siehe 1.1.6).

[12] $g = 9,81 ms^{-2}$ Erdbeschleunigung, v Geschwindigkeit in ms^{-1} und t Zeit in s

[13] $F = ma$ Grundgesetz der Mechanik, F Kraft, m Masse, a Beschleunigung

- Das Medium, hier die Luft, ist *homogen*. Bei hinreichend großen Öltröpfchen fällt die Inhomogenität der Luft nicht zu stark ins Gewicht.
- Die Zähigkeit η ist hinreichend groß, der Radius der Kugel hinreichend klein[14]. Dies ist beim Millikan-Versuch uneingeschränkt gegeben.

1.1.2 Berechnung der Ladung eines Öltröpfchens

Wenn Elektrische Kraft F_E und Erdanziehungskraft F_G gleichgerichtet sind, so kompensiert die Luftreibungskraft F_R die Summe aus Erdanziehungskraft F_G und elektrischer Kraft F_E, also:

$$F_R = 6\pi\eta r v_1 = qE + mg \tag{4}$$

mit η Zähigkeit der Luft, r und v_1 Radius und Geschwindigkeit des Öltröpfchens, g Erdbeschleunigung ($g \approx 9,81 ms^{-2}$), E elektrische Feldstärke (abhängig vom verwendeten Plattenkondensator, aber konstant bei idealem Feld), q Ladung des Öltröpfchens.

Bewegt sich das Tröpfchen nach oben – ist also F_E entgegengesetzt zu F_G gerichtet –, so nenne ich die Geschwindigkeit v_2, sonst v_1.

(4) läßt sich umformen zu

$$v_1 = \frac{qE + mg}{6\pi\eta r} \tag{5}$$

Wird der Plattenkondensator umgepolt – also die Richtung des elektrischen Feldes umgedreht –, so wirken Erdanziehungskraft und elektrische Kraft in entgegengesetzter Richtung, es folgt also analog

$$F_R = qE \Leftrightarrow mg \tag{6}$$

und damit

$$v_2 = \frac{qE \Leftrightarrow mg}{6\pi\eta r} \tag{7}$$

1.1.3 Elimination schwer meßbarer Größen

Der Radius r und die Masse m des Öltröpfchens lassen sich nur sehr schwer messen, weshalb sie eliminiert werden sollen.

Das Umpolen des Kondensators in 1.1.2 hatte lediglich den Sinn, eine rechnerische Elimination von r zu ermöglich. Die Masse m kann mit Hilfe des hier nicht weiter behandelten Gesetzes von Archimedes aus der Formel entfernt werden. Das *Gesetz von Archimedes* lautet:

$$m = \frac{4}{3}\pi r^3 \sigma \tag{8}$$

, wobei σ die Dichte des Öls ist. Die Dichte des Öls sei bekannt, also mit Hilfe anderer Versuche bereits bestimmt.

Zunächst sollen nun $v_1 + v_2$ und $v_1 \Leftrightarrow v_2$ gebildet werden:

$$v_1 + v_2 = \frac{2qE}{6\pi\eta r}$$

$$\Rightarrow r = \frac{qE}{3\pi\eta(v_1 + v_2)} \tag{9}$$

und analog

$$v_1 \Leftrightarrow v_2 = \frac{mg}{3\pi\eta r}$$

$$\Rightarrow r = \frac{mg}{3\pi\eta(v_1 \Leftrightarrow v_2)} \tag{10}$$

.

Das Gesetz von Archimedes (8) in die Gleichung (10) eingesetzt ergibt

$$r = \frac{\frac{4}{3}\pi r^3 \sigma g}{3\pi\eta(v_1 \Leftrightarrow v_2)}$$

$$\Rightarrow 1 = \frac{4r^2\sigma g}{9\eta(v_1 \Leftrightarrow v_2)}$$

$$\Rightarrow r^2 = \frac{9\eta(v_1 \Leftrightarrow v_2)}{4\sigma g} \tag{11}$$

Durch Gleichsetzen von (9) und (11) ergibt sich über r^2 und unter Verwendung von $E = U/d$: [15]

$$\frac{9\eta(v_1 \Leftrightarrow v_2)}{4\sigma g} = \left(\frac{qU}{3\pi\eta d(v_1 + v_2)}\right)^2$$

$$\boxed{\Rightarrow q = \frac{9\pi d}{2U}\sqrt{\frac{\eta^3}{\sigma g}}(v_1 + v_2)\sqrt{v_1 \Leftrightarrow v_2}} \tag{12}$$

In (12) sind v_1 und v_2 meßbar Größen, wobei darauf geachtet werden muß, daß sowohl bei der Bestimmung von v_1 als auch bei der Bestimmung von v_2 das gleiche Öltröpfchen verwendet wird. Die an den Kondensator angelegte Spannung U und der Abstand der beiden Kondensatorplatten d sind leicht einstellbar, bei der Zähigkeit der Luft η und der Dichte des verwendeten Öls σ handelt es sich um anderweitig meßbare Konstanten.

[14] Reynoldssche Zahl $Re = \frac{r\sigma v}{\eta} < 0,3$, Kriterium für mehr oder minder wirbelfreie Strömungen

[15] Dies ist eine Gleichung, die bei Plattenkondensatoren allgemein gilt, allerdings bei nicht idealen Kondensatoren und besonders im Randbereich eines realen Kondensators nur in Näherung. Debei ist d der Abstand der beiden Platten und U die Spannung, beides also leicht einstellbare Größen.

1.1.4 Ergebnis

Sei $e = 1,603 \cdot 10^{-19} C$. Dann sind die Ladungen, die beim Millikan-Versuch gemessen werden können, allesamt Vielfache von e. Aus diesem Grund wird e auch als *Elementarladung* bezeichnet, eine kleinere Ladung als e scheint nicht erzeugbar zu sein, die elektrische Ladung ist von diskreter und nicht von kontinuierlicher Natur.

Millikan hat es seinerzeit geschafft, den Wert der Elementarladung mit einer Genauigkeit von 10^{-5} zu bestimmen [12].

1.1.5 Zugeben von Meßfehlern

Feynman schreibt in [6]:

> Millikan [...] erhielt ein Resultat, von dem wir heute wissen, daß es nicht ganz richtig ist. Es liegt ein bißchen daneben, denn es benutzt einen unzutreffenden Wert für die Viskosität der Luft. Es ist interessant, sich die Geschichte der Messungen der Elektronenladung nach Millikan anzusehen. Wenn man sie als Funktion der Zeit darstellt, stellt man fest, daß die nächste ein bißchen höher liegt als die von Millikan, und die darauf folgende liegt noch ein wenig höher als jene und die nächste wiederum etwas höher, bis sie sich schließlich bei einer Zahl einpendeln, die eben höher liegt.

Auch Physiker sind offenbar nicht immer unvoreingenommen und ehrlich, sondern sie schielen auf die Ergebnisse von Autoritäten – wie alle Menschen. Im Übrigen hat wohl jeder in ein Resultat, das dem von anderen Forschern ermittelten exakt entspricht, mehr Vertrauen als in eines, das daneben liegt – bei Abweichungen wird eher nach möglichen Fehlern gesucht, bei mit anderen übereinstimmenden Werten der Wert leichter kritiklos übernommen.

1.1.6 Gültigkeit des Stokesschen Gesetzes, Interpretation des Versuchsergebnisses

Bei der Verwendung von Öltröpfchen mit sehr kleinen Radien liefert der Millikan-Versuch scheinbar das Ergebnis, daß die Größe der Elementarladung e mit kleiner werdendem Radius stark zunimmt ([12]); dies könnte naiv so interpretiert werden, daß die Ladung eines Elektrons von der Größe des Öltröpfchens abhängt.

Millikan hat dieses Resultat dadurch erklärt, daß das Gesetz von Stokes für sehr kleine Öltröpfchen nicht anwendbar ist (siehe 1.1.1). Damit das Gesetz gilt, müssen sich Körper von Kugelgestalt in einem homogenen Medium bewegen – dies ist bei sehr kleinen Öltröpfchen nicht mehr gegeben, die Tröpfchen haben nur noch mit relativ schlechter Näherung Kugelgestalt, und die nun verhältnismäßig großen Gasmoleküle der Luft bilden nun ein eher inhomogenes Medium.

Dies zeigt: die Ergebnisse von Experimenten müßen auch in der Physik mit Intuition bewertet und richtig interpretiert werden, die für das verwendete physikalische Modell getroffenen Annahmen – wie Vernachläßigung der Gravitationskraft des Experimentators oder der durch ein Auto hervorgerufenen Vibrationen des Gebäudes – müssen im Einzelfalle kritisch überprüft werden.

2 Humanwissenschaften

Was Experimente anbelangt, so gehe ich davon aus, daß die Probleme umso größer werden, je stärker Menschen als Objekte der Forschung auftreten – „Laborbedingungen" sind selten herrstellbar, die Reproduzierbarkeit meist kaum gegeben, die Ergebnisse können nur schwer verallgemeinert werden. Insofern stehen die „weicheren" Humanwissenschaften deutlich schlechter da als die Physik mit ihren eindeutigen Anforderungen an Experimente und wissenschaftliche Arbeit.

Trotzdem könnten die Humanwissenschaften der Informatik in vielen Bereichen als Vorbild dienen; die statistischen Methoden sind hier häufig gut entwickelt, und auch die Informatik hat häufig direkt mit Menschen als Forschungsobjekten zu tun.

2.1 Anwendung statistischer Methoden

Es gibt wenige Bereiche von Interesse, wo alle Menschen wirklich gleich sind. Aus diesem Grund müssen sich die Humanwissenschaften – anders als Teile der Physik – häufig mit statistischen Aussagen begnügen; z.B. im Marketing ist es mehr als ausreichend, wenn etwa 80% der Zielgruppe erreicht werden können – in der Physik würde eine Aussage, die in 20% der Fälle falsch ist, kaum als wertvoll betrachtet.

Statistische Verfahren stellen einige schwer erfüllbare Anforderungen:

- Die *Stichprobe* sollte *möglichst groß* sein – eine Vergrößerung der Stichprobe ist aber häufig mit imensen Kosten verbunden.
- Die *verwendeten Verfahren* müssen sehr sorgfältig ausgewählt werden. In der Statistik können –

aus mathematischer Sicht – zur Beantwortung einer Frage häufig mehrere alternative Verfahren verwendet werden. Dabei führen die unterschiedlichen Verfahren allerdings auch zu verschiedenen Ergebnissen, es ist eine schwere Aufgabe, das „richtige" Verfahren auszuwählen und zu begründen. Die Natur der Problemstellung muß bewertet und eingeschätzt werden, es gehen häufig nicht quantifizierbare Zusatzinformationen und Einschätzungen ein ([7]).

- Ein statistisches Verfahren ist in der Regel nur unter genau definierten *Voraussetzungen* anwendbar. Ob diese Voraussetzungen erfüllt sind, ist häufig nicht oder nur mit extrem hohem Aufwand zu ermitteln. Folglich wird beispielsweise fast immer davon ausgegangen, daß eine Zufallsgröße stetig, linear und normalverteilt ist. Die Annahme, daß das untersuchte Merkmal normalverteilt sei, kann zu einer erheblichen Verfälschung des Ergebnisses führen, wenn das Merkmal auch in Näherung eben nicht normalverteilt ist.[16]

2.2 Labor- und Feldversuche

Auch in den Humanwissenschaften existieren Laborversuche, also Experimente, bei denen künstliche Bedingungen geschaffen und alle Einflüsse sehr gut unter Kontrolle gehalten werden können – zum Beispiel die Messung des Hautwiderstandes bei der Darbietung erotischer Anzeigen im Marketing.

In der Praxis vorherrschend sind in den Humanwissenschaften allerdings Feldexperimente, also Versuche, bei denen die normalen Umweltbedingungen erhalten bleiben. Bei Feldexperimenten ist eine Kontrolle aller Störfaktoren nur sehr selten möglich.

2.3 Datenschutz und Ethik

Menschen haben ein Interesse an einer Privatsphäre, möchten nicht „meßbar" sein. Wegen des Datenschutzes ist es unmöglich, sich bestimmte Daten zu beschaffen.[17]

[16] Natürlich gibt es Verfahren, mit denen untersucht werden kann, ob die Annahme einer bestimmten Verteilung sinnvoll ist – ob diese Verfahren in der Praxis auch verwendet werden, ist eine andere Frage.

[17] Dies ist nur ein Beispiel für Bereiche, wo die Beschaffung verläßlicher Daten von Menschen gezielt behindert wird – Unternehmen werden kaum bereit sein, interne Daten zur Verfügung stellen, da die Konkurrenz beispielsweise über die eigene Marktsituation möglichst wenig informiert werden soll. Viele Daten sind auch für Wissenschaftler nicht erhältlich, und wenn die Unternehmen zur Zusammenarbeit mit einer Forschungseinrichtung bereit sind, so müssen die Unternehmensdaten in wissenschaftlichen Publikationen häufig nachträglich verfälscht werden

Ohne derlei nicht zu überschreitende Grenzen, wäre es insbesondere für die Wirtschaftswissenschaften sehr viel einfacher, verläßliche Prognosen abzugeben, auch und besonders die persönlichen Daten von Einzelpersonen wären – technisch gesehen – hervorragend auswertbar, wenn es darum geht, Werbung wirksamer zu gestalten, Arbeitnehmer einzustellen, das eigene Risiko als Bank oder Versicherung zu minimieren, etc.

3 Informatik

Bei *vollkommen kontrollierten Bedingungen* spricht man auch von Laborbedingungen.

Derartige Laborbedingungen sind in der Informatik im Allgemeinen und in der Sprachverarbeitung im Besonderen sehr viel eher zu erreichen als in großen Teilen der Humanwissenschaften.

Bei der Sprachverarbeitung handelt sich im Übrigen um ein Teilgebiet der Informatik, welches schon verhältnismäßig weit entwickelt ist – die hier entwickelte Methodik müßte auf andere Gebiete wie Neuronale Netze oder Robotik übertragbar sein.

3.1 Aufgaben und Probleme der Sprachverarbeitung

Die am natürlichsten erscheinende Form der Kommunikation ist die menschliche Sprache, so daß es wünschenswert ist, auch mit einem Rechner in normaler, geschriebener oder gesprochener Sprache kommunizieren zu können.

Desweiteren wäre es nützlich, wenn Computer in der Lage wären, Texte nach gewissen Gesichtspunkten zu analysieren, beispielsweise das Finden von Texten, die sich mit experimentellen Methoden in der Informatik befassen, in einem Datenbestand wie dem Internet – eine Anwendung des Information Retrieval.

Die Sprachverarbeitung wird mit einer Reihe von Problemen konfrontiert: ([2])

- Jeder Mensch spricht anders, hinzu kommen Dialekte, undeutliche Aussprache, Nebengeräusche, etc.
- Wie soll sich das System verhalten, wenn es ein „unbekanntes" Wort hört? Wie ist überhaupt entscheidbar, daß ein Wort dem System unbekannt ist – irgend ein Wort, das „am Besten" paßt, findet sich immer ...
- Häufig muß die Verarbeitung in Echtzeit verarbeitet werden.

- Die Anforderungen an die Rechenleistung der Computer sind extrem hoch.

Bei dem Versuch, natürliche Sprache zu *verstehen*, kommen weitere Probleme hinzu.

3.2 Spracherkennungs-Benchmarks

Monika Woszczyna schreibt in [14]:

> Die Fortschritte auf das Evaluierungsziel sind jedesmal gewaltig, zum einen, weil nur die Besten mit ordentlicher Förderung rechnen können, zum anderen, weil man als guter Teilnehmer nach der Evaluation alle Karten auf den Tisch legen muß und sich verdächtig macht, wenn im Laufe des nächsten Jahres niemand in der Lage ist, die guten Ergebnisse mit ähnlichen Methoden zu reproduzieren.

Problematisch wird es, wenn Wissenschaftler ihre Algorithmen nur auf das Evaluierungsziel hin optimieren. Wenn in der Evaluation nur die Wortfehlerrate gemessen wird, so ist die Rechenzeit von geringer Bedeutung, in praktischen Anwendungen ist es jedoch nicht hinnehmbar, daß eine leistungsfähige Workstation für die Auswertung eines einfachen Satzes mehr als eine Stunde Rechenzeit benötigt ([14]).

3.2.1 Benchmarking: Methodik

Um Evaluationen effizient durchführen zu können, müssen eine Reihe von Voraussetzungen erfüllt sein ([2]):

- Es müssen klare und objektive Kriterien zur Evaluation definiert werden. In dem Zusammenhang: auch und gerade in der Physik nimmt die Meßtechnik eine Schlüsselstellung ein, das Entwickeln eines neuen und guten Meßverfahrens stellt dort eine hoch anerkannte wissenschaftliche Leistung dar.
- Die Werkzeuge zur Durchführung der Evaluation – also beispielsweise CDs mit Sprachdaten – sollten allen Wissenschaftlern zur Verfügung stehen, damit sie schon vor der offiziellen Evaluation ihre Arbeiten optimieren können. Eine Alternative hierzu wäre es, die Testdaten nicht vorher auszuhändigen, um zu vermeiden, daß jemand sein System genau auf die bei der Evaluation verwendeten Daten (mit beschränkten Wortschatz, wenigen Sprechern oder dergleichen) zu optimiert.
- Während der Evaluation selbst müssen die Testverfahren und Testdaten selbstverständlich für alle Gruppen gleich sein, damit die Daten überhaupt miteinander verglichen werden können.

Im Folgenden sollen einige bekannte Kongresse, bei denen unterschiedliche Sprachverarbeitungsverfahren verglichen werden, kurz vorgestellt werden.

3.2.2 NIST Evaluationen

Die meisten Evaluationen werden in den USA vom NIST (National Institute of Standards and Technology) in Zusammenarbeit mit Förderern wie ARPA (Advanced Research Project Agency) organisiert und durchgeführt.

Ressource Management (RM) : Hier kommen relativ kleine Vokabulare zur Anwendung, also etwa 1000 Worte, die Aufgabe besteht in der abgelesenen Befehehlseingabe. Die Worterkennungsrate liegt hier mittlerweile bei 98%, die letzte Evaluation war 1992 ([14]).

WSJ-CSR Bei dem *Wall Street Journal-based Continous Speech Recognition* Test (WSJ-CSR) kommt es im Wesentlichen auf große Vokabulare und gelesene Sprache an. Als Basis dient Material aus dem *Wall Street Journal* und – seit 1994 zusätzlich – den *North American Business (NAB) news*. Die Evaluationen, die unter WSJ-CSR laufen, lassen sich wie folgt klassifizieren (siehe [3]):

- Sprecher-Abhängig oder Sprecher-Unabhängig (speaker dependent SD/ speaker independant SI)
- Größe des verwendeten Vokabulars (5K/ 20K/ 64K verwendete Worte)
- explizite oder implizite Punktuation (varbalized/ non-verbalized punctuation VP/NVP)

Bei der letzten Evaluierung ([4]) wurden erstmals auch Worte in den Tests verwendet, die nicht zum offiziellen Wortschatz gehörten (OOV, „Out-Of-Vocabulary"). Hierdurch traten charakteristische Fehler auf: aus „flywheels" wurde „fly we'll" oder „flight wheel", aus „powerbooks" machten einige Systeme „our books", und „centrifuges" wurde in „sent refuses" verwandelt.

Laut M. Woszczyna ([14]) handelt es sich hier mittlerweile nur noch um Materialschlacht, so daß zweifelhaft ist, ob auf diesem Gebiet auch 1996 wieder evaluiert wird.

Airline Traffic Information System (ATIS) : Bei diesen Tests werden Datenbankanfrage mit spontaner Sprache durchgeführt. Die Vokabulare sind typischerweise kleiner als 2000 Worte ([3]), es kommt im Wesentlichen auf präzise Worterkennung an. Gezählt wird die Anzahl der richtig generierten Antworten. Die Systeme sollten auch

dann noch richtige Antworten generieren, wenn der Anwender sehr undeutlich spricht, Silben einfach wegläßt, Nebengeräusche stören, etc. Laut [14] wurde bisher ohne Benutzer-Interaktion evaluiert, die Benutzerfreundlichkeit und damit praktische Anwendbarkeit spielte somit keine Rolle. In [3] wird auf erste Ansätze hingewiesen, die Zufriedenheit der Benutzer mit dem System zu berücksichtigen und die Systeme in einer realistischen interaktiven Umgebung zu bewerten.

Die ATIS-Tests werden in drei Kategorien eingeteilt:

- Erkennen spontaner Sprache (spontaneous speech recognition SPREC)
- Verstehen natürlicher Sprache (natural language understanding NL)
- Erkennen gesprochener (gelesener) Sprache (spoken language understanding SLS)

Beim ATIS Test zählt eine falsche Antwort genauso wie keine Antwort; es wäre denkbar, eine falsche Antwort stärker zu gewichten. ([3], [4], [14])

VERBMOBIL Evaluation Die Aufgabe bei dieser Evaluation besteht in der Terminabsprache ([14]).

SWB Switchboard Hier werden die Systeme mit spontaner Telephonabsprache über unterschiedliche Themengebiete konfrontiert, wobei bei der Aufnahme keine besonderen Regeln zur Anwendung kommen ([14]).

3.2.3 Klassen von Evaluationskriterien

In dem Text von Sundheim und Chinchor ([13]) werden Evaluationskriterien in drei Klassen eingeteilt:

progress evaluation : Vergleich des Systems mit dem, was bereits vorher vorgestellt wurde.

adequacy evaluation : Frage, ob das System die Anforderungen eines potentiellen Kunden adequat erfüllen kann.

diagnostic evaluation : Frage, wo das System Fehler macht und warum die Fehler auftreten. Derlei Fragestellungen werden natürlich intensiv von den Entwicklern selbst bearbeitet.

3.2.4 Beispiele von Evaluationskriterien

Eine Schlüsselrolle kommt den Evaluationskriterien zu, hier zwei Beispiele, entnommen aus dem Text von Doddington [2]:

- *Ziel*: Entwicklung von Spracherkennungsverfahren, die robust sind gegen Störungen des akustischen Signals, zum Beispiel durch Störungen am Mikrofon oder durch Nebengeräusche.
  Evaluationskriterium: Wordfehlerrate minimieren.
- *Kontextabhängiges Sprachverstehen*: Entwickeln von Technologien, die es erlauben, interaktiv Datenbanken abzufragen.
  Evaluationskriterium: Anzahl der „richtigen" Antworten.

3.2.5 Stärken

- Die Forscher erhalten durch die Evaluationskriterien eine klar definierte Aufgabenstellung, wissen, welche Leistungskriterien ihr System möglichst gut zu erfüllen hat ([2]).
- Evaluationen unterstützen den „Fortschritt" in der Sprachverarbeitung. Wenn es Standard ist, daß ein neues System gewisse Qualitätskriterien erfüllen muß – z.B. eine geringere Wordfehlerrate als alle bisher verfügbaren Ansätze –, so kann der Effekt, daß immer wieder neue, aber im Endeffekt gleichwertige Verfahren publiziert werden, vermieden werden.
- Es ist für Außenstehende einfacher möglich, das für sie beste Verfahren zu finden.
- Überhaupt erkennen, wo technischer Fortschritt stattfindet und möglich ist – wo es also eher lohnt, Forschungsgelder zu investieren. Direkt im Zusammenhang damit steht der Nachteil, daß Forschungsgruppen versucht sein könnten, über gezielte Manipulationen bei Evaluationen ihre eigene finanzielle Lage zu verbessern.

3.2.6 Probleme

- Testdaten in ausreichender Menge und mit hinreichendem Realitätsbezug sind nicht immer leicht zu erhalten. In der Physik können sehr viel „härtere" Daten gesammelt werden, natürliche Sprache ist immer sprecherabhängig, die Methoden können kaum Allgemeingültigkeit erreichen.
- In der Vergangenheit haben diverse Forschungsgruppe ihre Forschungsergebnisse häufig nicht miteinander verglichen (siehe 3.2.8); mittlerweile ist es auf den entsprechenden Konferenzen allerdings Standard, daß die Forschungsgruppen ihre Ergebnisse miteinander vergleichen.
- Optimierung auf das Evaluationsziel – dabei z.B. Vernachlaessigung des Rechenzeitbedarfs.

- Problematik, sinnvolle Evaluierungskriterien zu definieren.

- Wirklich innovative Ideen können durch Evaluationen blockiert werden ([2]): Systeme, die auf lange bekannten Verfahren basieren, sind in der Regel besser ausgereift als vollkommen neue Ideen, so daß wirklich neue Ansätze nur noch dann eine faire Chance erhalten, wenn sie wirklich revolutionär sind, die Leistung der alternativen Verfahren schon im Anfangsstadium der Entwicklung übersteigen. Verfahren, die zwar ein sehr gutes Entwicklungspotential hätten, aber nicht sofort mit den „Klassikern" Schritt halten können, werden so deutlich benachteiligt.[18]

- Wenn die Evaluationen zu viel Zeit in Anspruch nehmen, so dürfen sie wohl als Zeitverschwendung betrachtet werden ([2]). Kosten und Nutzen sind natürlich auch hier gegeneinander abzuwägen.

In der Physik muß vom Experimentator im Wesentlich gefordert werden, daß er ehrlich ist, seine Vorgehensweise für andere nachvollziehbar und reproduzierbar dokumentiert und bei der Auswertung seiner Ergebnisse nicht voreingenommen ist.

Zusätzlich besteht in der Sprachverarbeitung und weiten Gebieten der Informatik die große Gefahr, daß „Experimente" gezielt so angelegt werden, daß eine vorher gefaßte Vermutung untermauert wird oder auch, daß ein Programm zum Beispiel auf ein Evaluierungsziel hin optimiert wird – unter Vernachlässigung der praktischen Brauchbarkeit.

Allerdings hat die Sprachverarbeitung bei Experimenten noch nicht die Probleme, die Auftreten, wenn Menschen die Objekte der Untersuchung sind – im Software Engineering oder in der Psychologie beispielsweise ist die Reproduzierbarkeit der Versuche häufig kaum gegeben, eine Adaption der harten Kriterien der Physik damit weniger leicht. Die unterschiedlichsten Umwelteinflüsse und die verschiedensten das Ergebnis verfälschenden Faktoren können in solchen Bereichen vom Experimentator kaum noch im Überblick behalten werden.

[18] Dies ist allerdings ein Problem, welches nicht primär von den Evaluationen hervorgerufen wird – neue, wenig entwickelte Ansätze sind meistens erst einmal schlechter, Gallium Arsenid wird nicht problemlos das lange etablierte Sillicium verdrängen können. Dadurch, daß die neuen Verfahren in der Regel definitiv schlechter *sind*, können sie sich gegen die klassischen Verfahren nur sehr schwer durchsetzen – die Evaluationen sorgen lediglich dafür, daß leicht erkennbar wird, welches Verfahren überlegen ist. Das neue Verfahren bietet vielleicht eine gute Grundlage für weitere Forschungsanstrengungen, aber in der Praxis sollte es nicht sofort umgesetzt werden.

3.2.7 Gesamtsystem im Auge behalten

Allgemein sollte niemals das Gesamtsystem aus den Augen verloren werden – etwas, wozu Benchmarks leicht verleiten. So betrachtet Fellbaum in [5] die seinerzeit existenten Textverarbeitungs- und Tabellenkalkulationsprogramme mit natürlichsprachlicher Eingabemöglichkeit als nutzlos – und zwar nicht, weil die Spracherkennungsalgorithmen zu langsam oder zu fehlerhaft arbeiten, sondern weil die Schnittstelle zur Mensch-Maschine-Kommunikation nicht ordentlich genug entworfen wurde. Eine auf Maus und Tastatur beruhende Benutzerführung ist nicht ohne grundlegende Änderungen auf natürlichsprachliche Schnittstellen übertragbar, das Problem der Ergonomie darf auch bei etwas so natürlich erscheinendem wie gesprochener Sprache nicht vernachläßigt werden.

Benchmarks lenken die Aufmerksamkeit der Entwickler auf einige kritische Punkte – sofern es problematische Bereiche gibt, die von den verwendeten Tests nicht abgedeckt werden, ist dies ein schwerwiegender Nachteil. Objektive Kriterien zum Vergleich diverser Systeme werden in diesem Fall vorgegaukelt, trotzdem sie de facto nicht existieren

3.2.8 Historische Bemerkungen

Ein relativ früh entwickeltes Teilgebiet war das Information Retrieval, welches seit etwa 30 Jahren bearbeitet wird, auch quantitativ ([8]). Allerdings

- haben die Gruppen nicht mit den gleichen Daten gearbeitet,

- wurden unterschiedliche Evaluierungs-Werkzeuge verwendet,

- wurden kaum Versuche unternommen, verschiedene Systeme miteinander zu vergleichen.

Ein weiteres Problem bestand darin, Mengen von Testdaten realistischer Größe zu erhalten.

4 Vergleichende Betrachtung der Wissenschaften

4.1 Definition Experimente

In dem physikalischen Lehrbuch von Finn/Alonso wird definiert:

> Ein Versuch besteht darin, ein Phänomen zu beobachten, das unter vorher festgelegten und

sorgfältig kontrollierten Bedingungen abläuft. Der Wissenschaftler kann also die Bedingungen willkürlich verändern, wodurch es leichter wird, ihren Einfluß auf die Vorgänge zu erkennen. ([11], S.3)

Die Physiker können die Bedingungen und Umwelteinflüsse in der Regel kontrollieren, in den Humanwissenschaften ist dies nur selten der Fall. Demzufolge haben die Geisteswissenschaftler die harte Definition etwas abgewandelt, wie anhand einer Definition aus einem Marketing-Lehrbuch ersichtlich wird:

> Als Experiment bezeichnet man die Überprüfung eines vermuteten (Ursache-Wirkungs-)Zusammenhanges unter kontrollierten, vorher festgelegten Bedingungen. Das Wesen eines Experimentes besteht darin, daß eine unabhängige Variable (z.B. der Preis) verändert wird und die Auswirkung dieser Veränderung auf eine abhängige Variable (z.B. Absatzmenge) gemessen wird.
>
> ...
>
> Als Störvariablen werden jene Einflußgrößen der abhängigen Variablen bezeichnet, die vorab nicht festgelegt werden können, *nicht kontrollierbar sind* (z.B. Konkurrenzmaßnahmen). ([1], S. 142 ff.)

In der Physik wird die vollständige Kontrolle der Bedingungen als wesentlich bezeichnet, im Marketing wird die Überprüfung von Kausalzusammenhängen besonders hervorgehoben.

Im Marketing sind Techniken entwickelt worden, um die Störfaktoren halbwegs in den Griff zu bekommen – zum Beispiel die Verwendung von Referenzgruppen. Voraussetzung für den Erfolg dieses Ansatzes ist es, daß die Störfaktoren auf die Versuchsgruppe ebenso wirken wie auf die Referenzgruppe ([1], S. 144 ff.).

4.2 Unterschiede zwischen den Wissenschaften

- *Exaktheit*: in der Physik können die Konfidenzintervalle häufig kleiner sein als im Marketing, in den Geisteswissenschaften sind häufig nur verbalqualitative Aussagen möglich, teilweise ist eine Beschränkung auf den statistischen Mittelwert notwendig, es kann sein, daß nur Aussagen getroffen werden können, die mit einer Wahrscheinlichkeit kleiner Eins zutreffend sind, etc.
- *Reproduzierbarkeit*
- Einbeziehung von *Menschen* als Objekte
- *Kontrolle von Umwelteinflüssen*, die das Ergebnis in einer Größenordnung verfälschen, die nicht zu vernachlässigen ist.

4.3 Gemeinsame Probleme der Wissenschaften

- Konkurrenzdruck
- Möglichkeit, bei der Darstellung der eigenen Ergebnisse und bei der Bewertung der eigenen Arbeit nicht ganz ehrlich zu sein, zum Beispiel der Versuch, den eigenen Algorithmus besser dastehen zu lassen
- Hemmungen, eigene Fehler zuzugeben
- Quantitative Forschung ist häufig sehr aufwendig, kostenintensiv und – da es sich meist nur um Routinearbeiten handelt – wenig interessant
- Es müssen passende *Modelle* gefunden und in ihrer Anwendbarkeit begründet werden (Satz von Stokes in der Physik, Normalverteilungsannahme in den Wirtschaftswissenschaften, Evaluationskriterien in der Informatik)
- Schwierigkeiten bei der *Modellbildung* und Interpretation der Ergebnisse

Jede Wissenschaft muß sich einzuhaltende wissenschaftliche Mindeststandards geben. Zum Beispiel könnte von einer Arbeit über einen neuen Algorithmus verlangt werden, daß er quantitativ mit bereits existierenden Verfahren verglichen wird.

5 Schlußbemerkungen

Die Physik ist eine der ersten Wissenschaften, die eine fundierte experimentelle Methodik entwickelt hat und (nicht nur) in dieser Beziehung nach wie vor vorbildlich für andere Wissenschaften. Die schwere Kontrollierbarkeit von Umwelteinflüssen und insbesondere die durch die Einbeziehung von Menschen in den Objektbereich auftretenden Probleme sind der Physik allerdings auch fremd.

Die Humanwissenschaften haben durch die Anwendung stochastischer Verfahren gezeigt, daß es auch in einem für quantitative Forschung nicht eben optimalen Umfeld möglich ist, experimentelle Methoden anzuwenden und methodisch zu fundieren. Teilgebiete der Informatik bewegen sich, was Reproduzierbarkeit und Nähe zu Menschen betrifft, auf einem ähnlichen Niveau

wie Teile der Humanwissenschaften, andere Teilbereiche können relativ leicht Bedingungen herstellen, die sich mit den in den Naturwissenschaften üblichen Laborbedingungen messen lassen.

In der Sprachverarbeitung wurden mit Evaluationen erste Erfolge erzielt, die Forschungsgruppen bauen häufig auf den Arbeiten derjenigen auf, die im Vorjahr am Besten abgeschnitten haben. In anderen Gebieten wie dem Optical Character Recognition existieren vergleichbare Ansätze.

Quantitative Forschung ist in der Informatik insgesamt allerdings noch wenig verbreitet, wissenschaftliche Richtlinen sind noch wenig entwickelt. Da die Informatik sich irgendwo zwischen Natur- und Geisteswissenschaften bewegt, kann sie von beiden lernen.

Literatur

[1] Ralph Berndt. *Marketing.* Springer, 1990. 1. Käuferverhalten, Marktforschung und Marketing-Prognosen.

[2] G. Doddington. Program overview/ technology overview, 1993. Vortragsfolien.

[3] D.S. Pallet et al. Benchmark tests for the DARPA spoken language program, 1993.

[4] D.S. Pallet et al. Benchmark test for the ARPA spoken language program, 1995.

[5] Fellbaum. *Elektronische Sprachverarbeitung.* Franzis, 1991. S. 256-261.

[6] R.P. Feynman. *Sie belieben wohl zu scherzen, Mr. Feynman !* Serie Pieper, 1990.

[7] W. Gaul. Ausgewählte statistische Probleme im Marketing. Folienkopien, 1995.

[8] Donna Harman. Overview of TREC-1, 1993.

[9] Joachim Grehn (Hg.). *Metzler Physik.* J.B. Metzlersche Verlagsbuchhandlung Stuttgart, 2 edition, 1988.

[10] John C. Knight and Nancy G. Leveson. An experimental evaluation of the assumption of independence in multiversion programming. *IEEE Transaction on Software Engineering*, 12(1):96–109, Januar 1986.

[11] E.J. Finn M. Alonso. *Physik.* Addison-Wesley, 3 edition, 1988.

[12] Schpolski. *Atomphysik, Teil 1.* VEB Deutscher Verlag der Wissenschaften, Berlin, 1979. Seiten 14-19.

[13] B.M. Sundheim and N.A. Chinchor. Survey of the message understanding conferences, 1993.

[14] M. Woszczyna. Kurzübersicht Spracherkennungs-Benchmarks. Email, April 1995.

Herausforderung Software Engineering: Beobachten und Messen außerhalb des Labors

Dieter Bär

Zusammenfassung

Das „Experimentieren“ im Sinne kontrollierter Laborversuche ist im Hauptbereich des Software Engineering, dem Programmieren im Großen, fast unmöglich. Gegenstand dieser Ausarbeitung ist die Beschreibung einer praktikablen, teils automatisierten Arbeitsweise anhand ausgewählter Beispiele.

1 Einleitung

Im Zuge der Entwicklung neuer Modelle, Konzepte, Umgebungen und Sprachen kommt es immer wieder zu „Glaubenskriegen“ über den Vorteil des einen oder den Nachteil des anderen. Hierbei spielen weniger Fakten eine Rolle, denn dazu müßten welche vorhanden sein, als vielmehr persönliche, subjektive Ansichten. Eine objektiven Kritik hingegen bedarf neben der Theorie gesicherter Daten. Diese erhalten wir durch Experimente.

Nun können wir aber in der Softwareentwicklung nicht einfach den Anspruch physikalischer Versuche übernehmen, denn wir stoßen auf folgende Probleme:

- Physikalische Versuche finden i.a. in einer wohldefinierten Umgebung, sprich Labor, statt. Laborversuche in der Softwareentwicklung können aufgrund des finanziellen und zeitlichen Umfangs nur begrenzt durchgeführt werden.
- Die Wiederholbarkeit der Experimente und die Nachvollziehbarkeit der Ergebnisse können außerhalb eines Labors nicht sichergestellt werden. Die Entwicklungsumgebungen und Projekte sind zu verschieden.

Wir wollen nun anhand verschiedener Beispiele Wege aufzeigen, wie dennoch - zumindest ansatzweise - eine Lösung aussehen kann.

2 Softwarefehler und Komplexität

2.1 Einleitung

Das Ziel des Experimentes von Basili und Perricone aus [1] liegt in der Analyse der Beziehung zwischen der Häufigkeit und der Verteilung von Fehlern während der Softwareentwicklung und -wartung. In diesem Fall wurden speziell die Komplexität des Projektes, die Erfahrung der Entwickler und die Rolle der Wiederverwendung existierender Software unter dem Aspekt der Qualität und Zuverlässigkeit betrachtet. Dazu wurde eine Analyse der Änderungen an einem mittelgroßen Softwareprojekt - außerhalb eines Labors - durchgeführt. Anhand dieses Experiments werden wir Möglichkeiten der systematischen Datenerfassung und Auswertung aufzeigen.

2.2 Das Experiment

Aufgabe des Projektes war die Entwicklung eines Programms zur Durchführung von Studien in der Planung von Satelliten. Das Programm umfaßte 90.000 Zeilen an Quellcode in Fortran. Dabei wurde eine große Anzahl vorhandener Module wiederverwendet und teilweise den Gegebenheiten angepaßt. Das Verhältnis zwischen der Anzahl neuer Module und der Wiederverwendung (modifizierten) Codes betrug 1:1.

2.3 Durchführung

Für das Protokoll eines Versuches benötigen wir neben obiger **Beschreibung des Experimentes** noch eine **Beschreibung des Umgebung.** Dazu gehört u.a. das Feststellen des Kenntnisstandes der Mitarbeiter. Ebenso müssen wir den Zeitraum der Datenerfassung angeben, d.h. im wesentlichen eine genaue Beschreibung der Entwicklungsstufe zum Zeitpunkt der Datenerfassung.

Zur Erfassung experimenteller Daten müssen wir verschiedene **Festlegungen** treffen. Dazu wurden hier folgende Definitionen getroffen:

1. *Module:* Innerhalb der Module wurden nur die Teile erfaßt, die in Fortran geschrieben wurden. Dazu unterschied man zwischen *modifizierten* Module, d.h. Module, die während anderer Projekte implementiert wurden, und *neuen,* die speziell für dieses Projekt entwickelt wurden.
2. *Anzahl der Zeilen an Quellcode und der ausführbaren Zeilen:* Die Anzahl der Quellcodezeilen umfaßte Fortran-Anweisungen sowie Kommentare. Als ausführbare Zeilen wurden alle Zeilen mit ausführbaren Befehlen gezählt.

3. *Fehler:* Als Fehler wurden alles betrachtet, was ein nicht spezifikationsgerechtes Verhalten zur Folge hatte. Weiter wurde zwischen *textuellen* Fehlern, d.h. Schreibfehlern, und *konzeptionellen* Fehlern, d.h. Fehlern in der Implementierung der Spezifikation, unterschieden.

Als nächstes müssen wir **Formblätter** nach Aussehen und Inhalt entwerfen. In diesem Projekt wurden folgende Inhalte besonders hervorgehoben:

- Das Datum einer Änderung/ Fehlerentdeckung.
- Die Beschreibung der Änderung/ des Fehlers.
- Die Anzahl der geänderten Komponenten.
- Die Art der Änderung/ der Typ des Fehlers.
- Der Korrekturaufwand.

Die Berichte der Entwickler müssen natürlich einer ständigen **Kontrolle** unterliegen. Das Verfahren dazu müssen wir festlegen. In diesem Fall wurden die ausgefüllten Formulare vom Projektleiter kontrolliert und unterzeichnet. In einem weiteren Schritt wurden die Reporte auf Widersprüche untersucht. Auf eine Rücksprache mit dem betreffenden Entwickler wurde hier verzichtet.

Auch die **Auswertung** der Testdaten müssen wir zu Beginn festlegen. Weiter haben wir die **Auswirkungen** der Datenerfassung auf den Entwicklungsprozeß zu erfassen. Zu einem vollständigem Experiment gehört weiter eine **Interpretation** der Auswertung und die Bestimmung der **Konsequenzen** für zukünftige Projekte. Auf die Ergebnisse dieses Experimentes gehen wir im nächsten Abschnitt ein.

2.4 Ergebnisse

Als erstes Ergebnis stellte man schon während der Kontrolle der Formblätter fest, daß die Anzahl an Korrekturen von Verbesserungen dreimal so groß war, wie vor der Einführung der Formblätter. Weiter wurde festgestellt, daß 89 Prozent der Fehler verbessert werden konnten, indem lediglich ein Modul geändert wurde - ein Argument für die Modularisierung von Projekten. Die Fehler waren fast gleichverteilt auf beide Modultypen. Im folgenden beschreiben wir die Auswertung, wie sie in diesem Experiment durchgeführt wurde.

Zur Aufwandsabschätzung für die Korrekturen wurden folgende Einheiten gewählt:

1. eine Stunde oder weniger
2. eine Stunde bis ein Tag
3. ein Tag bis drei Tage
4. mehr als drei Tage.

Bevor wir auf die Fehlerverteilung nach Typ und dem zugehörigen Korrekturaufwand eingehen, hier eine feinere Klassifikation der Fehlertypen:

1. Voraussetzungen falsch oder falsch verstanden
2. funktionale Spezifikation nicht korrekt oder falsch umgesetzt
3. Entwurfsfehler, die mehrere Komponenten betreffen
   - falsche Vorausetzung bzgl. Werte oder der Struktur von Daten
   - Ausdruck falsch berechnet oder nicht korrekte logische Kontrollstruktur
4. Entwurfsfehler, die eine Komponente betreffen (Klassifikation s.o.)
5. externe Umgebung falsch verstanden
6. Fehler im Gebrauch der Programmiersprache/ des Kompilers
7. Schreibfehler
8. Fehler aufgrund einer fehlerhaften Korrektur eines früheren Fehlers

In diesem Experiment waren die meisten Fehler das Ergebnis einer falschen oder falsch verstandenen Spezifikation. Bei anderen Projekten stellte man jedoch fest, daß die größte Anzahl an Fehlern Entwurfsfehler einzelner Komponenten sind. Weiter wurde erkannt, daß die Wiederverwendung vorhandener Module nicht notwendigerweise die Implementierungskosten senkt: In den Kategorien 1 und 2 war der Korrekturaufwand für neue und modifizierte Module nahezu gleich. In den Klassen 3 und 4 jedoch, dominierte die Zeit zur Verbesserung modifizierter Module. Der Schluß liegt nahe, daß Fehler in wiederverwendeten Modulen weit schwerer zu verbessern sind, als in neuen, selbst oder kürzlich entwickelten und verstandenen Modulen.

Zur Untersuchung der einzelnen Module hinsichtlich Implementierungsfehler wurde eine weitere Einteilung vorgenommen:

1. Schnittstellenfehler: fehlerhafte Modulschnittstellen/ fehlerhafter Gebrauch externer Modulschnittstellen
2. Initialisierungsfehler: Fehler in der Initialisierung von Datenstrukturen bei Ein- bzw. Austritt aus dem Modul
3. Datenfehler: falscher Zugriff auf Datenstrukturen
4. Berechnungsfehler: falsche Berechnung einer Funktion, d.h. eines Wertes einer Variablen
5. Fehler in Kontrollstrukturen: bei gegebenen Eingabedaten wird nicht der spezifizierte Pfad durchlaufen.

Des weiteren wird jede dieser Klassen in zwei weitere Unterklassen aufgeteilt, die unterscheiden zwischen einer unterlassenen Anweisung und einer falschen Anweisung. Ein Parameter der zum Aufruf eines Unterprogramms benötigt wird aber fehlt, ist z.B. ein Schnittstellenfehler vom Typ Unterlassung.

Man stellte fest, daß Schnittstellenfehler unabhängig vom Modultyp am häufigsten vorkommen. Fehler in den Kontrollstrukturen kamen häufiger in neuen Module vor, während Initialisierungs- und Datenfehler öfter in modifizierten Module registriert wurden. Das Verhältnis bzgl der Modultypen in den Fehlerklassen unterlassene Anweisung und falsche Anweisung hielt sich die Waage. Allerdings bestanden zweidrittel aller Fehler aus falschen Anweisungen.

Das Verhältnis von Modulgröße zur Fehlerhäufigkeit wurde mit Fehlern pro 1000 Zeilen ausführbarem Quellcode festgelegt. Das folgende erschien zunächst überraschend: je kleiner die Module, desto größer war die Fehlerrate. Für dieses Ergebnis gibt es aber zwei einfache Gründe: erstens werden größere Module mit mehr Sorgfalt implementiert und zweitens wird es mit zunehmender Größe der Module offensichtlich schwieriger den Kontrollfluß zu testen, womit aber zahlreiche Fehler unentdeckt bleiben.

Eine andere Erklärung für diese Tatsache wurde zunächst auch in der Komplexität der einzelnen Module vermutet. Mit der Einführung einer Größe für die Modulkomplexität, hier definiert durch die Anzahl an Entscheidungen + 1, konnte dies jedoch widerlegt werden: größere Module sind komplexer als kleinere. Dabei war die durchschnittliche Komplexität fehlerbehaftete Module gleich derer aller Module.

Aufgrund der Auswertung der Daten dieser empirischen Untersuchung aus der Softwareentwicklung im Großen können wir zusammenfassend folgende Schlüsse ziehen: Da wiederverwendete, modifizierte Module fehleranfälliger waren als neue, muß mehr Aufwand auf die Problemdefinition sowie die Spezifikation bzgl. der Anwendung in anderen Projekten getrieben werden. Außerdem sind die Vor- und Nachteile einer Wiederverwendung abzuwägen: auf der einen Seite können die Entwicklungskosten für ein Modul gesenkt werden, dem gegenüber steht der größere Aufwand beim Korrigieren eines Fehlers in einem (modifizierten) Modul. Aus den gegenläufigen Daten von Modulgröße und Fehlerhäufigkeit folgt, daß es zu diesem Zeitpunkt keinen Sinn macht, Modulgrößen und deren Komplexität künstlich zu begrenzen.

2.5 Zusammenfassung

An diesem Experiment wurde exemplarisch die Durchführung eines Experiments parallel zur Entwicklung außerhalb eines Labors aufgezeigt: das Vorgehen bei der Datenerfassung, Auswertungsstrategien und Interpretation der Ergebnisse.

Sicherlich führt eine Datenerfassung zur Erhöhung der Entwicklungskosten. Auf der anderen Seite können Probleme und Problembereiche früher erkannt und in der Entwicklung zukünftiger Projekte berücksichtigt werden. Dennoch läßt sich ein solches Experiment nur bedingt mit einem physikalischen vergleichen, da das Umfeld durch das Projekt bestimmt wird, und eine Bestätigung der Ergebnisse oft nur durch eine ähnliche Datenerfassung und Auswertungsstrategie erreicht werden kann, nicht jedoch durch eine Wiederholung des Versuchs unter denselben Voraussetzungen. Es handelt sich insoweit also lediglich um eine empirische Untersuchung. Ein Versuch unter quasi Laborbedingungen besprechen wir nun im folgendem Abschnitt.

3 Zwei Entwicklungsmodelle im Experiment

3.1 Hintergründe des Experimentes

Die Verfasser von [2] beschreiben folgendes Laborexperiment: Um die Vor- und Nachteile des Prototypmodells bzw. des Spezifikationsmodells zu ermitteln, wurde ein Projekt (Entwurf und Implementierung eines COCOMO User Interfaces) gleichzeitig an sieben Teams vergeben. Vier Teams entwickelten mit dem Spezifikationsmodell, drei Teams mit dem Prototypmodell. Das typische Merkmal eines Laborversuchs, mehrere Gruppen die gleichzeitig eine identische Aufgabenstellung erhalten und bearbeiten, war damit gegeben. Was ausßerdem in einer Laborumgebung zu berücksichtigen ist und welche Ergebnisse erwartet werden können, zeigen wir durch diesen Versuch auf.

Hintergrund waren die folgenden Entwicklungsmodelle und die dazugehörigen Auswahlkriterien:

1. **Spezifaktionsmodell:** Problemlösung spezifizieren (abstrakte Spezifikation), daraus Entwurf für Implementierung spezifizieren (konkrete Spezifikation)
2. **Prototypmodell:** Prototyp entwickeln, evtl. Prototypen der Module erstellen und bzgl. Problemdefinition testen

Die theoretischen Vorteile des Spezifikationsmodells liegen in der Durchschaubarkeit und in den Steuerungsmöglichkeiten während des Entwicklungsprozesses. Weiterhin lassen sich weitere Module leicht integrieren. Das Spezifikationsmodell wird oft als Teil des Wasserfallmodells gesehen. Das Prototypmodell

ermöglicht das frühzeitige Erkennen von Risikobereichen und dient gleichzeitig dem Dialog mit dem Auftraggeber. Durch Rückkopplung können die Wünsche des Benutzers und das entstehende Produkt angeglichen werden. Die Nachteile liegen auf der einen Seite in der Schwierigkeit weniger mathematisch funktionellen Einheiten, wie der Mensch-Maschine-Schnittstelle, (formal) zu spezifizieren und auf der anderen Seite in der konzeptionellen Nähe zum Modell des Codierens und Verbesserns.

Das Experiment sollte dazu beitragen, folgende praxisorientierte Fragen zu lösen:

- Welche Anwendungsbereiche eignen sich besonders für das eine bzw. das andere Modell?
- Welche Auswirkungen hat das Entwurfsmodell auf die Planung, den Aufwand und die Produktivität, sowie auf die Größe, die Qualität und die Wartbarkeit des Produktes?
- Gibt es Gründe, die beiden Modelle zu vermischen? Wie sieht dann ein neues Modell aus?

Fragen also, die nicht ohne weiteres aus der theoretischen Formulierung der Entwurfsverfahren geklärt werden können.

3.2 Das Umfeld des Versuchs

Es ging darum ein Programm zur Unterstützung einer Kostenabschätzung für Softwareprojekte zu erstellen. Die Erstellung einer Benutzeroberfläche nimmt offensichtlich unter dieser Zielsetzung weit mehr Raum ein als die algorithmische Verarbeitung der durch diese erfaßten Daten. Da wären neben den Eingabemöglichkeiten, die Abfragemodi, die Formatierung und Darstellung der Ausgabe, Hinweise auf syntaktisch fehlerhafte Eingabe und eine on-line Hilfe. Dazu ist eine umfangreiche Menüstruktur und eine Darstellung von Tabellen notwendig.

Den Teams wurden modellspezifisch Meilensteine vorgegeben. Weiterhin wurden von den Versuchsleitern Reporte bzgl. dem Stand der Entwicklung an den Meilensteinen erstellt. Dabei wurde zum einen die Spezifikation in Bezug auf die Produkanforderungen untersucht, zum anderen die Prototypen einem Test aus Sicht des Auftraggebers unterzogen. Jedes Programm wurde dann nach folgenden vier Kriterien in einer Skala zwischen 0 und 10 bewertet:

1. **Funktionalität:** Nutzen der Berechnungen, der Benutzeroberfläche, der Ausgabe und der Funktionen für den Umgang mit Dateien
2. **Robustheit:** Schutz vor Programmabbruch, Programmabsturz und Datenverlust durch äußere Einflüsse
3. **Benutzerfreundlichkeit:** Vorhandensein und Gebrauch der geforderten Funktionen, sowie die Nachvollziehbarkeit des Programmablaufs
4. **Erlernbarkeit:** Schnelligkeit mit welcher der Benutzer mit dem Programm zurechtkommt, d.h. die gewünschten Ergebnisse erhält, dazu Bewertung der Programm-, Hilfe- und Fehlermeldungen sowie der Dokumentation.

Außerdem wurde von jedem Student eine Ausarbeitung zu folgender Frage verlangt: „Was würden wir besser machen, wenn wir erneut dieses Projekt durchzuführen hätten?“.

Das Programm selbst wurde in UCB Pascal unter UNIX auf einer VAX 11/780 implementiert. Die Dokumentation wurde ebenfalls unter UNIX erstellt. Dafür mußten die Erfahrungswerte der Teilnehmer im Umgang mit obigen Werkzeugen anhand einer Selbsteinschätzung ermittelt werden. Nach der Einteilung der Teams und der Festlegung der Modelle wurde die Organisation innerhalb der Teams freigestellt. Die Mehrheit bevorzugte eine demokratische Organisationsstruktur, in der alle an jedem Entwicklungsschritt teilnahmen.

Da das Experiment im Rahmen einer Lehrveranstaltung durchgeführt wurde, mußten einige Einschränkungen, wie z.B. nicht isolierte Teams, nicht repräsentative Tests, Berücksichtigung der Neigung der Teilnehmer bzgl. der Modell und der manchmal ungenauen Datenerfassung in Kauf genommen werden.

Mit dem Setzen von unverrückbaren Meilensteinen, einer Festlegung der Datenerfassung, dem Erfassen von gruppenspezifischen Merkmalen und dem Einholen von Meinungen der Entwickler, unter Berücksichtigung der Gruppenzugehörigkeit, wurden weitere Methoden eines Laborexperiments realisiert.

3.3 Ergebnisse

Die Programmgröße und der Entwicklungsaufwand unterschied sich erheblich: Im Durchschnitt benötigten die Teams, die mit dem Prototypmodell entwickelten, 45 Prozent weniger Aufwand bei gleichzeitig 40 Prozent kleineren Programmen. Im wesentlichen läßt sich dies auf folgende zwei Gründe zurückführen: Im Spezifikationsmodell versuchten die Teams möglichst viel durch die Spezifikation zu erfassen, wohingegen im Prototypmodell lediglich das Nötigste implementiert wurde und weitere Funktionen bei verbleibender Zeit integriert werden sollten. Im allgemeinen war der Entwicklungsaufwand proportional zur Programmgröße.

Durch die unterschiedliche Teamgröße und dem konzeptionellem Mehraufwand des Prototypmodells

bzgl. des Quellcodes läßt sich keine Aussage über die Produktivität der Teammitglieder treffen. Die Produktqualität war im Durchschnitt gleich: In der Funktionalität und der Fehlertoleranz hatten Spezifikations-Teams leichte Vorteile, wohingegen die Benutzerfreundlichkeit und die Erlernbarkeit durch die Prototyp-Teams besser realisiert wurde.

Paradox im Vergleich zum ersten Experiment erscheint, daß die Studenten bei einer Umfrage nach Abschluß der Entwicklung es bevorzugten, die Programme, die mit dem Prototypmodell entstanden sind, zu warten. Der eher subjektive Eindruck der Teilnehmer hängt aber wohl wesentlich mit der unterschiedlichen Größe der Programme zusammen.

Wie oben beschrieben, mußten die Teams je nach Modell zu verschiedenen Zeitpunkten eine bestimmte Entwicklungsstufe erreicht haben. Dabei stieg der Aufwand kurz vor diesen „deadlines“ speziell in den Spezifikations-Teams stark an. Erwartungsgemäß verwendeten die Prototyp-Teams weniger Zeit auf den Entwurf und die Programmierung als auf das Testen, das Besprechen und Verbessern der Programme. Dadurch wurde zum einen im Spezifikationsmodell die Aussage, daß man zuerst nachdenkt und dann codiert, und zum anderen im Prototypmodell die Ansicht immer etwas funktionsfähiges vorweisen zu können, bestätigt.

Signifikant erschienen zunächst die Unterschiede im Dokumentationsumfang: In den Spezifikations-Teams wurde mehr als doppelt soviel Aufwand benötigt. Betrachtet man jedoch den Aufwand im Verhältnis zur Größe des Quellcodes und berücksichtigt, daß schon während des Entwurf große Teile der Dokumentation entstehen, so gleichen sich die Werte an.

Kurzgesagt wurden folgende Ergebnisse deutlich:

- signifikante Unterschiede im Dokumentationsumfang, den absoluten Mannstunden und der Wartbarkeitsbewertung
- kleinere Unterschiede in Funktionalität und Robustheit auf der einen Seite, und Benutzerfreundlichkeit und Erlernbarkeit auf der anderen Seite
- keine Unterschiede in der Produktivität.

Ein anderes wichtiges Ergebnis konnte in Zusammenhang mit der Teamgöße festgestellt werden: kleinere Teams benötigten 41 Prozent weniger Aufwand bei gleichzeitig nur 8 Prozent kleinerem Quellcode. Dies trägt der Tatsache Rechnung, daß in größeren (demokratischen) Teams mehr Kommunikationsaufwand nötig ist. Daneben scheinen sich kleinere Teams mehr auf das Wesentliche zu konzentrieren, als eine Unzahl möglicherweise nützlicher Funktionen zu integrieren. Die Folge ist natürlich, daß dadurch weit weniger Code und Debugging notwendig wird. Dies wurde auch durch die abschließende Befragung der Teilnehmer bestätigt.

Über den Entwicklungsprozeß im allgemeinen läßt sich festhalten, daß hier zwar der Programmierungsaufwand dominierte, aber die Dokumentation hinsichtlich des Auswands mehr und mehr gleichzieht. Auch hatte die Wettkampfsituation einen positiven Effekt auf die Produktqualität. Hierbei spielte die Benutzeroberfläche die entscheidende Rolle.

Zusammenfassend die wichtigsten Ergebnisse im Vergleich der beiden Modelle:

- Das Prototypmodell impliziert ein kleineres Produkt mit nahezu gleichen Eigenschaften aber weniger Entwicklungsaufwand. Der Entwickler sieht eher was gebraucht wird und was nicht.
- Mißt man die Produktivität in DSI/ MH (*delivered source instructions per man-hour*) liegt das Spezifikationsmodell vorne. Setzt man jedoch die Zufriedenheit des Anwenders in Zusammenhang mit den Mannstunden als Maß für die Produktivität ein, hat das Prototypmodell klare Vorteile.
- Das Prototypmodell bevorzugt konzeptionell die Entwicklung einer Benutzeroberfläche, den Dialog mit dem Auftraggeber, die Erkennung von Risikobereichen und die Reduzierung des Deadline-Effekt.
- Das Spezifikationsmodell unterstützt die Planung und den Entwurf. Es liegt also nahe, einem Prototypen eine Spezifikation der Module folgen zu lassen, was in einem neuen Prototypmodell heute auch realisiert wird.
- Kleinere Teams haben eine größere Produktivität.

3.4 Zusammenfassung

In diesem Beispiel haben wir dargelegt, welche Bedingungen und Ergebnisse in einer Laborumgebung im Vergleich zum ersten Experiment von Bedeutung sind. Dennoch fehlten auch hier einige Voraussetzungen:

1. gleiche Teamgrößen, nicht die Organisationsstruktur sondern die Entwicklungsmodelle sollten untersucht werden
2. genauere Definition der Benutzerschnittstellen
3. verbesserte Datenerfassung, zum einem bzgl. der schon erfaßten Werte, aber auch bzgl. der Entwicklung und den Entscheidungen innerhalb der Gruppen
4. Untersuchung des Einflusses der Teamgröße, der Programmiersprache, der vorhandenen Werkzeuge und des Einflusses des Auftraggebers

5. Betrachtung des einzelnen Entwicklers.

Wir sehen an diesem Beispiel, daß ein großer Aufwand betrieben werden muß, um annähernd Laborverhältnisse zu simulieren. Dabei unterliegen wir der Gefahr, das eigentliche, hier die beiden Modelle, durch zu viele Aspekte, wie Teamgröße und Organisation, aus den Augen zu verlieren. Weiterhin wurden in diesem Experiment die wesentlichen Merkmale eines Versuchs, wie Begriffdefinitionen, Formblätter, Auswertung (vgl. Produktivitätsmessung) und Interpretation deutlich. Typisch für eine Laborumgebung war auch die Form der Ergebnisse durch Vergleich der Gruppen untereinander.

In beiden Experimentumgebungen bilden Datenerfassung und Auswertung zentrale Aufgaben. Was liegt also näher, um kostengünstig vorzugehen, als zu automatisieren, und parallel zur Entwicklung eine Datenbasis zu schaffen, auf die in zukünftigen Projekten zurückgegriffen werden kann. Dadurch auch außerhalb des Labors unter realistischen Bedingungen durch Versuche Ergebnisse zu erzielen. Einem solchem Projekt widmet sich der nächste Abschnitt.

4 Das TAME-Projekt

4.1 Einleitung

Entwicklungsmodelle führen zu vernünftiger Planung und Analyse eines Projekts. Die zugrundeliegenden Prinzipien haben sich über Jahre hinweg bewährt. Die Modelle sind im wesentlichen verbesserungsorientiert, d.h. sie beinhalten einen Rückkopplungsmechanismus, um Fehler zu verbessern oder Änderungen einzubringen. Ein Modell muß dazu anpaßbar und steuerbar sein. Mit der **Anpassungsfähigkeit** *(tailorability)* wird die Möglichkeit, das Modell für einen bestimmten Zweck nutzbar zu machen, bezeichnet. Ein Entwicklungsprozeß ist **steuerbar** *(tractable)*, wenn er leicht zu planen, zu verwalten und auszuführen ist. Eine Softwareentwicklungsumgebung *(software engineering environment)*, kurz SEE, muß diese Fähigkeiten unterstützen und soweit wie technisch möglich automatisieren. Im TAME-Projekt *(Tailoring A Measurement Environment)* der University of Maryland wurde ein solches verbesserungsorientiertes Entwicklungsmodell entworfen und im TAME-System als integrierte SEE, kurz ISEE, implementiert.

4.2 Der Softwareentwicklungsprozeß

Aus den bekannten Modellen (z.B. dem Wasserfall- oder Spiralmodell) sowie den dazugehörigen Techniken (Methoden und Werkzeuge) lassen sich folgenden Prinzipien zur Entwicklung von Software ableiten:

1. unterscheide zwischen Konstruktions- und Analysephasen
2. plane den Konstruktionsprozeß
3. formalisiere Analyse und Verbesserung des Entwurfs
4. analysiere die Entwurfmethoden
5. sehe Rückkopplungsmechanismen innerhalb eines Projektes und unter abgeschlossenen und zukünftigen Projekten vor
6. bedenke, daß sich alle Projekte unterscheiden
7. passe das gewählte Entwicklungsmodell einem Projekt und seiner Umgebung an
8. formalisiere diese Anpassung bzgl. der Qualität, der Produktivität und der Projektumgebung, falls möglich
9. verwende Erfahrungen aus anderen Projekten
10. erlaube eine flexible Steuerung des Prozesses

Zur Analyse und Anpassung benötigen wir Daten. Dazu sind Meßmethoden notwendig. Im TAME-Projekt wurden folgende Punkte berücksichtigt:

1. Messung ermöglicht Charakterisierung, Berechnung und Vorhersagen der verschiedenen Stufen eines Softwareentwicklungsprozesse
2. sowohl Entwicklung als auch das Produkt selbst sind zu betrachten
3. durch Messung können Kosten, Effektivität, Zuverlässigkeit, Korrektheit, Leistung, Wartbarkeit und Benutzerfreundlichkeit untersucht werden
4. Messung benötigt einen Bezugspunkt, sie muß die unterschiedlichen Sichten der am Projekt Beteiligten berücksichtigen
5. eher subjektive Eigenschaften, wie z.B. Erfahrungen der Entwickler, sollten erfaßt werden können
6. komplexe Eigenschaften können zerlegt und dann vermessen werden
7. eine Messung muß Teil der Entwicklungs- und Wartungsumgebung sein
8. der Meßprozeß ist von Anfang an Teil der Entwicklung
9. jedes Projekt benötigt eigene Maße
10. auch die Umgebung läßt sich durch Maße charakterisieren
11. Datenerfassung- und Auswertungsstrategien sind zu bestimmen
12. die Interpretation der Maße muß angegeben werden
13. anhand der gewonnenen Daten läßt sich eine Erfahrungsbasis aufbauen
14. die Erfahrungsbasis bildet die Grundlage für ein Expertensystem

4.3 Das TAME-Projekt

Das TAME-Projekt, d.h. das Entwicklungsmodell, basiert auf folgenden Paradigmen, die aus obigen Prinzipien hervorgegangen sind (**GQM-Modell** für *Goal/Question/Metric*).

Verbesserungsorientiert (strukturelle Grundlage)

1. Bestimmung der Projektumgebung
2. Zielbestimmung unter Berücksichtigung meßbarer Qualitätsmerkmale
3. Wahl des Entwicklungsmodells und der dazugehörigen Werkzeuge
4. Ausführen des Modells, dabei Entwicklung, Datenerfassung und Rückkopplung
5. Auswertung der Daten: Bestimmung von Risiken, Aufzeichnen der Erkenntnisse und Vorschläge zur Verbesserung
6. Ausführen des nächsten Projekts auf der Grundlage des gewonnenen Wissens

Modelldefinition (Formalisierung)

1. Zielbestimmung anhand des Zwecks und der Umgebung
2. produktbezogene Fragen bzgl. Problemdefinition (Größe, Komplexität, Kosten, Zeitaufwand, Fehlerklassen, Anwendungskategorien), Qualitätsfestlegung (Zuverlässigkeit, Benutzerfreundlichkeit, Datenerfassung) und der Realisierung des Rückkopplungsmechanismus
3. modellbezogene Fragen bzgl. der Modelldefinition (Anwendungscharakterisierung und Nutzen), der Qualität und wieder des Rückkopplungsverfahrens
4. Festlegung der Maße, der Datenerfassung und Durchführung der Interpretation

Diese Gesichtspunkte sollten im TAME-System realisiert werden. Wir erkennen, daß zur automatischen Modellgenerierung Datenmaterial aus früheren Projekten oder eben Experimenten unerläßlich ist. Außerdem sehen wir, daß in den klassischen Modellen meist nur eine Teilmenge der im TAME-Projekt geforderten Komponenten vorgesehen ist. Weiterhin muß eine umfangreiche Datenbasis aus Laborexperimenten und Softwareentwicklungen aus der Industrie gewonnen werden.

4.4 Das TAME-System

Im TAME-System wurde und wird versucht, möglichst viele
Komponenten aus dem TAME-Entwicklungsmodell zu unterstützen. Die Entwickler unterscheiden dabei zwischen externen Anforderungen, d.h. die Schnittstellen zu den Anwendern, und internen Zielsetzungen, d.h. der Funktionalität, die zur Realisierung des TAME-Projekts vorgeschlagen wurden. Als Grundlage für die Schnittstellen nach außen dienen die Prinzipien (oder formalisiert Schablonen) des TAME-Prozeßmodells.

Ziel ist

1. einen Mechanismus zur konstruktiven und quantifizierbaren Projektzielbestimmung (bzgl. Entwurf und Analyse) zu finden. Dazu soll der Benutzer ein bekanntes, Teile eines bekannten oder ein neues Entwicklungsmodell auswählen bzw. generieren. Grundlage hierfür ist die Erfahrungsbasis.
2. die Datenerfassung zu automatisieren, sowie manuell gesammelte Daten zu erfassen und zu prüfen. Dies beinhaltet z.B. die Erfassung der Anzahl an Quellcodezeilen, der Programmkomplexität, der Kompilierungsvorgänge und der Anzahl der Testläufe.
3. eine Steuerung der Messung und Analyse zu integrieren. Dazu werden Werkzeuge zur Messung und Datenerfassung, eine Bestimmung der Maße und die Anwendung statistischer Methoden benötigt.
4. eine Interpretation der Analyseergebnisse bzgl. der Umgebung zu erstellen und den Rückkopplungsmechanismus für Verbesserungen am Modell selbst, sowie den Werkzeugen und Methoden einzuleiten, d.h. die Erfahrungsbasis zu erweitern.
5. aufgrund der Erfahrungsbasis für das ganze Unternehmen einen Lernprozeß einzuleiten.

Um diese Funktionalität dem Anwender verfügbar zu machen, werden folgende Ziele angestrebt:

1. Implementierung einer Benutzeroberfläche. Dazu wird zwischen physikalischen Schnittstellen, wie z.B. einem Fenstersystem und einer logischen Oberfläche unterschieden, d.h. der Zugang zu Daten und Meßvorgängen soll dem Benutzer nur über das GQM-Modell möglich sein.
2. Darstellung der Daten, Ergebnisse und Erfahrungen.
3. Bereitstellung einer Datenbank zur Sicherung der neugewonnenen Erkenntnisse.
4. Zugangskontrolle zu den verschiedenen Komponenten, insbesondere der Erfahrungsbasis (Konsistenz).
5. Konfigurations- und Steuerungsmöglichkeiten des Sytems.
6. Schnittstelle zu einer entwurfsorientierten SEE zwecks Rückkopplung mit dem aktuellen Entwicklungsprozeß und dem damit verbundenen Austausch von Informationen mit der Erfahrungsbasis.

7. Implementierung als verteiltes System.

Das TAME-Systems wurde entspechend den Vorgaben in einer Schichtenarchitektur entworfen: am oberen Ende befindet sich die (physikalische) Benutzerschnittstelle, als Fundament dient die Erfahrungsbasis und im Zentrum steht der Rückkopplungsmechanismus. Wir sehen, daß Methoden, die zur Durchführung von Experimenten angewandt werden, im TAME-System integriert werden: Datenerfassung durch Messung, Auswertung und Interpretation.

4.5 Die erste Implementierung

Der erste Prototyp ließ erwartungsgemäß viele Wünsche offen. So wurde zunächst nur ein Teil der Funktionen implementiert. Schwierigkeiten traten bei der Formalisierung des GQM-Paradigmas, der Integrierung der einzelnen Komponenten in das Gesamtsystem und bei der Realisierung des Expertensystems (der Wissensrepräsentation) auf. Mit den Prototypen der einzelnen Komponenten, soweit vorhanden, konnten aber bereits Erfahrungen gesammelt werden. Auf die einzelnen Ansätze, den Entwicklungsstand und die Erfahrungen im Einzelnen werden wir hier nicht weiter eingehen.

4.6 Zusammenfassung

Das Ziel der Entwicklung eines Systems zur Modellgenerierung und Projektsteuerung erforderte zunächst eine genaue Untersuchung des Softwareentwicklungsprozesses. Dieses beinhaltete die Beschreibung der Entwicklungsumgebung, die Planung zur Integration von Verbesserungen in das laufenden Projekt, sowie die Ausführung des Modells unter Einhaltung des Plans. Grundlage zur Modellauswahl bildet eine Datenmenge, die auf wohldefinierten Messungen beruht. Eingeleitet wird weiterhin ein Lernprozeß, der die Entwicklung von Software via Rückkopplung optimiert. Die vollständige Realisierung des TAME-Systems kann nur mit Fortschritt der Forschung vorangetrieben werden. Die implementierten Prototypen liefern heute schon Werkzeuge, z.B. zur Datenerfassung, Reporterstellung und Analyse.

In Softwarelabors wie z.B dem Software Engineering Laboratory (SEL) der Experimental Software Engineering Group (ESEG) [19] wird die Grundlage für eine automatische und projektspezifische Modellgenerierung durch Datenerfassung und Werkzeugentwicklung gelegt. Durch Anwendung der Werkzeuge im Großen (hier z.B. bei der NASA) wird die Erfahrungsbasis schnell vergrößert. Das Experimentieren im Großen wird durch Automatisierung möglich.

[19] WWW: http://www.cs.umd.edu/projects/SoftEng/tame/

5 Fazit

Zu einer vollständigen Realisierung des TAME-Systems sind noch umfangreiche Forschungen notwendig. Darum muß heute noch mit Teillösungen gearbeitet werden. Zu der auf Experimenten basierenden Datenerfassung ist eine Standardisierung von Maßen unabdingbar. Dazu sind Laborversuche notwendig. Da aber außerhalb eines Labors kaum ein Projekt parallel mehrmals entwickelt werden kann [20], müssen die Methoden, die im ersten Experiment aufgezeigt wurden, automatisiert werden. Wo dieses nicht möglich ist, muß eine Kosten-Nutzen-Relation aufstellt werden. Bei einer Durchführung müssen dann folgende Punkte realisiert werden:

- Projektbeschreibung: Problemdefinition, Ziele und Umgebung
- Begriffsdefinitionen: Festlegung der zu untersuchenden Größen und deren Maße
- Formblätter: Gestaltung und Kontrollmechanismen
- Auswertungsstrategien: Methoden und Werkzeuge
- Interpretation der Ergebnisse
- Auswirkungen auf den laufenden Entwicklungsprozeß
- Anwendung des gewonnenen Wissens in zukünftigen Projekten

Mit dem starken Anstieg der Kosten für Softwareentwicklung und -wartung in den Unternehmen, muß der Entwicklungsprozeß durch Einfluß von vorhandenen Erfahrungen optimiert werden. Eine Möglichkeit dazu ist das Experimentieren, verstanden als Datenerfassung und Auswertung innerhalb einer Organisation über deren Projekte.

Literatur

[1] Victor R. Basili, Barry T. Perricone: *Software Errors and Complexity: An Empirical Investigation,* Communication of the ACM, vol. 27, no. 1, Januar 1984, pp. 42-52

[2] Barry W. Boehm, Terence E. Gray, Thomas Seewaldt: *Prototyping versus Specifying: A Multiproject Experiment,* IEEE Trans. on Software Engineering, vol. 10, no. 3, Mai 1984, pp. 290-303

[20] Ausnahme: bei hohen Sicherheitsanforderungen

[3] Victor R. Basili, H. Dieter Rombach: *The TAME Project: Towards Improvement-Oriented Software Environments,* IEEE Trans. on Software Engineering, vol. 14, no. 6, Mai 1988, pp. 758-773