David Gugerli: Die Welt als Datenbank. zur Relation von Softwareentwicklung, Abfragetechnik und Deutungsautonomie. In: Nach Feierabend 3 2007, p. 11 – 36

David Gugerli

Die Welt als Datenbank
Zur Relation von Softwareentwicklung, Abfragetechnik und Deutungsautonomie[1]

An Leichen fehlt es nie, wenn die Spezialisten der amerikanischen Krimiserie CSI auf allen Fernsehkanälen dieser Welt ihrer Arbeit nachgehen. Meistens treten die Mordopfer ohne erkennbaren Zusammenhang gleich im Doppel auf und bringen dadurch die *ars combinatoria* der technisch hochgerüsteten Laborpolizisten erst so richtig in Fahrt. Seit September 2000 wurden über 160 Episoden allein von CSI Las Vegas produziert. Zusammen mit ihren Filialserien CSI Miami und CSI New York hat die Serie den globalen Markt für TV-Krimis fest im Griff.[2] Ihre Staffage, die Schnittfolgen und die Dramaturgie sind längst zur verbindlichen Norm der Kulturindustrie geworden. Auf dem deutschen Privatsender SAT1 läuft ein Imitat unter dem Titel »R.I.S. Die Sprache der Toten«. Das Kürzel steht zwar nicht für »Crime Scene Investigation«, sondern etwas bieder und bürokratisch für »Rechtsmedizinische Investigative Sonderkommission«. Aber auch die deutschen Science-Cops operieren wie ihre amerikanischen Kollegen mit den letzten Raffinessen der forensischen Spitzentechnik: »Das Team von R.I.S. setzt die modernsten wissenschaftlichen Methoden ein, um Morde aufzuklären. Sie sind die Besten der Besten ihres Fachs, sie verstehen die Sprache der Toten. Und: Sie sind die letzte Chance auf Gerechtigkeit.«[3] Selbst humorvollere Alternativangebote zur technoiden Ernsthaftigkeit von CSI sehen sich gezwungen, mit dem Qualität verbürgenden Label zu spielen, und heißen deshalb beispielsweise Navy CIS.[4]

Der Erfolg von CSI beruht auf einer technowissenschaftlichen Inszenierung effizienter Detektivarbeit, die einer strengen Formel folgt. Steven Bochco, einer der erfolgreichsten Produzenten amerikanischer TV-Serien, bezeichnet CSI als »Formelserie« und stellt einen unübersehbaren Bruch mit bisherigen Erfolgsrezepten fest. »Mord, Puzzle, Überführung. Ein ewiger Dreiklang. Die Charaktere haben kein Privatleben, keine Geschichte und sind teilweise nicht voneinander zu unterscheiden. Aber es ist die erfolgreichste Serie der Welt. ›CSI‹ hat in den letzten sechs Jahren die Regeln des Geschäfts verändert. Bis anhin galten Produktionen mit viel Subtext und starken Charakteren als Messlatte. ›CSI‹ bewies, dass das Gegenteil besser funktioniert: Form statt Inhalt.«[5] Die kulturelle Wirkung, die CSI in 200 Ländern und bei Hunderten von Millionen begeisterter Zuschauer erzeugt, ist schwer zu beurteilen und wird inzwischen in einer Reihe von Untersuchungen

kontrovers diskutiert. Es ist nicht geklärt, ob auch diesseits der Bildschirme immer mehr Verteidiger und Geschworene CSI-Praktiken für Standardverfahren halten. Kriminalistikstudiengänge scheinen sich hingegen tatsächlich einer wachsenden Beliebtheit zu erfreuen, und es mag auch Verbrechen geben, die so geplant und durchgeführt werden, dass ihre Spuren sogar im Fall einer Untersuchung mit raffinierten CSI-Methoden unentdeckt bleiben. All dies gehört zu dem, was man den CSI-Effekt oder gar das CSI-Syndrom genannt hat.[6]

In der Regel wird bei der Beschreibung des Syndroms übersehen, dass sich mit den technowissenschaftlichen TV-Krimis[7] nicht nur die Staffage, sondern auch die Deutungs- und Interpretationsmuster der Krimiwelt grundlegend gewandelt haben.[8] Sowohl Mörder als auch Opfer haben eine neue Rolle zugewiesen erhalten und interessieren in dramaturgischer Hinsicht nur noch als Träger von Spuren, die Evidenz erzeugen können. Hier setzen die Detektive an und interessieren sich dabei erstaunlich wenig für die soziale Dynamik, die sich in einem bestimmten Fall zum Gewaltverbrechen gesteigert hat. Tatmotive sind nachgerade irrelevant, und die Schuldfrage wird kaum je gestellt. Es gibt einen toten und einen lebendigen Körper, deren folgenreiche Begegnung in der jüngsten Vergangenheit forensisch auswertbare Daten erzeugt hat. Zusammen mit dem Tatort werden diese beiden Körper zu einem Reservoir von Spuren, zu einem Pool von Indizien, zu einer Sammlung von tatspezifischen Signifikanten, die sich technisch stabilisieren und dann jederzeit abrufen und zueinander in Beziehung setzen lassen. Die Kunst der Rekombination solcher Daten, deren Herkunft, Qualität und Form einen hohen Grad an Heterogenität aufweisen können, wird dem Publikum als Interpretationsspiel vorgeführt: In optisch stark verwischten Sequenzen werden immer wieder mögliche Narrative simuliert. Jedes dieser provisorischen Auswertungsfragmente zeigt den Detektiven an, wo sich vielleicht noch weitere Spuren suchen und finden lassen. Die Datenbeschaffung kann weitergeführt und verfeinert werden.

Metaphorisch gesprochen wird in CSI die Welt als Datenbank inszeniert, deren Einträge es aufzuspüren und zu kombinieren gilt, um so die alles entscheidenden Einsichten in die Verhältnisse zu gewinnen. Mit den von der Crime Scene »abgerufenen« Daten generieren die Kriminalisten – im technischen wie im cineastischen Sinne – »views«, die Rückschlüsse über die vielfältigen Relationen zwischen den Dingen ermöglichen. Dieser neue Modus der interpretatorischen Praxis ist augenfällig. Er lässt sich dramaturgisch etwa in der extrem collageartigen und übergangsarmen Schnitttechnik beobachten. Erst aus der Kombination kleinster Einheiten wird im Laufe der Sendung überhaupt ein Narrativ entwickelt. Jedes Mitglied des Teams steuert seine Spezialkenntnisse und Spezialfragen zum Gesamtbild des einzelnen Falles bei – fotografische, pathologische, entomologi-

sche, toxikologische, materialwissenschaftliche, informations- und biotechnologische Wissensbestände kommen im arbeitsteiligen Verfahren zur Anwendung und werden in geeigneter Weise miteinander verschränkt. Auch das Arsenal an Apparaturen und Verfahren, wie sie auf der zur Sendung gehörenden Website dargestellt sind, bietet wenig an vorgefertigten Zusammenhängen. Es ist vielmehr eine Sammlung von einzeln abrufbaren Einblicken, etwa in die verschiedenen Folgen, in die Biographien der Figuren und jene ihrer Darsteller und Darstellerinnen. Die Besucher sollen ihre eigenen »views« der Zusammenhänge generieren. Sie können dies etwa mit Hilfe eines CSI-Handbuchs versuchen, das die datenbankgestützten forensischen Geräte mit den dazugehörigen Beschreibungen und Bildern verknüpft. Obwohl das Handbuch in Flash programmiert wurde, imitiert es dennoch eine rudimentäre Datenbank und ist nahtlos in das (datenbankgestützte) »web content management system« der CBS integriert.

Aber nicht nur die Dramaturgie der Serie und der Nutzerkontext ihrer Websites sind von Datenbanktechniken geprägt. Abgesehen davon, dass in den meisten Sendungen Datenbanken nach Personen und Kontrollschildern durchforstet werden (nach dem Motto: »beim FBI oder bei der NSA findet sich eigentlich alles«), nutzen nicht wenige der vorgestellten Technologien ihrerseits ganz unterschiedliche Datenbanken. So hängt zum Beispiel der Video Spectral Comparator VSC2000, eine multifunktionale Workstation zur Dokumentenüberprüfung, an Datenbanken. Er integriert und kombiniert alle verfügbaren Lichtquellen, mit denen die feinsten Veränderungsspuren an Dokumenten aufgespürt werden können. Damit lassen sich dann etwa die Herstellungsmerkmale offizieller Dokumente mit denen eines am Tatort gefundenen Dokuments vergleichen.[9] Und das SLIP oder »Shoe Wear Linking and Identification Program« einer Firma namens Mushroom Software erlaubt es, praktisch jeden Abdruck einer Schuhsohle so zu identifizieren, dass Hersteller, Verkäufer und Kundschaft miteinander verknüpft werden können.[10]

Kein Zweifel, die herkömmliche hermeneutische Prozedur, mit der weiland Inspektor Columbo (gestützt auf seine persönliche Intuition und seinen legendär malträtierten Notizblock) jeden noch so raffinierten Missetäter durch hartnäckiges Interpretieren dingfest gemacht hat, gehört der Vergangenheit an.[11] Im Zeitalter von CSI wirkt sein Verfahren, das vom Tatmotiv ausging und die lückenhaften Erzählungen der Verdächtigen kritisch zu interpretieren suchte, hoffnungslos antiquiert. Der mit allen Wassern gewaschene Interpret Columbo – Tiefenhermeneutiker und Psychologe in Personalunion –, der den Autor des Verbrechens wie kein anderer zu verstehen vermag, der sich einfühlt in das Motiv des Täters und dem längst informierten Zuschauer die Zwangsläufigkeit seiner richtigen Interpretation

vor Augen führt, ist abgelöst worden durch ein Team von wissenschaftlichen Experten, das arbeitsteilig und systematisch vorgeht, eine Plethora von Daten unterschiedlicher Qualität zu isolieren weiß und diese in provisorischen, aber immer definitiver werdenden Simulationen auf originelle Weise rekombiniert. Hermeneutik und Psychologie sind verschwunden.

Als Maschine und Denkmodell verändert die Datenbank jedoch nicht nur die Prozeduren kriminalistischer Arbeit, um damit beispielsweise zu einem höheren Output an gelösten Fällen pro Sendung zu führen. Als Maschine und Denkmodell steht die Datenbank für die Versicherung, dass diesseits und jenseits der Bildschirme der kombinatorische Freiheitsgrad jeder »signifying practice« erweitert werden kann. Der am CSI-Beispiel festgemachte kulturelle Wandel findet jedoch keineswegs im luftleeren Raum der Signifikantenspiele statt. Wenn die erfolgreichsten kulturindustriellen Produkte einer Zeit so grundlegende kommunikative Verfahren wie Suchen, Deuten und Verstehen im Modus der Datenbankabfrage präsentieren, dann stellt sich die Frage nach dem Zusammenhang zwischen Datenbankentwicklung und dem Wandel der »signifying practice« einer Epoche wie von selbst. Sie ist Gegenstand der folgenden Überlegungen.

Historiographische Zugänge

Die Behauptung, dass sich die Such- und Deutungskultur des ausgehenden 20. und des beginnenden 21. Jahrhunderts an den Rekombinationsofferten der rechnergestützten Datenbanktechnik orientiert, muss an der Geschichte der Datenbanktechnik überprüft werden. Dabei wird es hilfreich sein, im Hinblick auf die offenbar irgendwo »zwischen Columbo und CSI«, also seit den 1970er Jahren erfolgte Veränderung sowohl die Zirkulationsbedingungen des Wissens als auch die *dramatis personae* des Wissens im Auge zu behalten. Zu ersteren gehört die gesteigerte Verfügbarkeit und Kombinierbarkeit von Daten sowie die größere Freiheit, an vorhandene Daten neue, unerwartete Fragen zu richten. Zu letzteren zählt die funktionale Ausdifferenzierung der beteiligten Akteure des Wissens bzw. die Darstellung des Deutungsprozesses als arbeitsteiliges Verfahren. Diese großen Veränderungen lassen sich, so die These, als Konsequenz der Entwicklung der rechnergestützten Datenbanktechnik beobachten. Sie fand in den 1970er und frühen 1980er Jahren statt und fiel damit in eine Zeit, in der sich »Flexibilisierung« in den unterschiedlichsten Feldern gesellschaftlicher Praxis einer besonderen Attraktivität erfreut hat – vom Detailhandel bis zum Militär, vom Transportwesen bis zur Hochschule wurde die Allokation verfügbarer Ressourcen flexibilisiert.[12] Um schnell auf veränderte Marktbedingungen reagieren zu können, um Ausbildungsgänge den

erwarteten Chancen anzupassen, um knappe Mittel optimal einzusetzen und um Güter nach dem neuen Prinzip der »just in time«-Produktion sowohl herstellen als auch vertreiben zu können, kam dem Wissen um Relationen und Interdependenzen eine steigende Bedeutung zu. Operative Konzepte, Informationssysteme und Analysetechniken veränderten sich in atemberaubender Geschwindigkeit. »Il n'y a pas de phénomènes fondamentaux. Il n'y a que des relations réciproques et des décalages perpétuels entre elles«, lautet das in seiner Grundsätzlichkeit fast schon paradoxe Diktum von Michel Foucault aus dem Jahre 1982.[13]

Im Unterschied zur Tatortuntersuchung im Fernsehen ist beim Versuch, die interdependente Entwicklung von Datenbankkultur und Deutungstechnik nachzuzeichnen, nicht einfach evident, wo man die zur Beurteilung des interessierenden Zusammenhangs notwendigen Spuren und Daten finden kann. Gewiss, auch die Computergeschichte ist in den letzten Jahren auf ein respektables Niveau gekommen. Dennoch dominieren nach wie vor jene Studien, die einen bald sechs Jahrzehnte dauernden, höchst verwickelten soziotechnischen Wandel auf das Thema einer Computerrevolution verkürzen wollen.[14] Sie teilen mit James Cortada die Einschätzung eines »immer schneller, immer mehr« als wichtigstem Merkmal der Veränderung.[15] Der Unterschied liegt einzig darin, ob die vielen exponentiellen Wachstumskurven als Zeichen des Fortschritts oder als Zeichen zunehmender Bedrohung gelesen werden.[16] Drei in den letzten Jahren entstandene Forschungsrichtungen bieten hingegen nützliche Hinweise, um den Zusammenhang zwischen Datenbankentwicklung und kulturellem Wandel zu verstehen. Dazu gehört erstens die genauere Untersuchung der Nutzerkontexte, wie sie etwa von Paul Edwards vorgeschlagen wurde,[17] zweitens eine Abkehr vom Primat der Hardwaregeschichte[18] zugunsten einer Softwaregeschichte, wie sie Martin Campbell-Kelly vorgelegt hat,[19] und drittens eine Computergeschichte, welche mit sozialhistorischen Ansätzen Strategien der Professionalisierung und der Governance von Unternehmen und Verwaltungen im Kontext von Rechner- und Software-Entwicklung untersucht. Dazu gehören die Arbeiten von Thomas Haigh über die Frühzeit von »Management Information Systems«, Jon Agars »Government Machine« und JoAnne Yates' Studie über den Einsatz von Rechnern in der Lebensversicherungsbranche.[20] Die Genese und die Verbreitung von Maschinen sind, darin sind sich die letztgenannten Untersuchungen einig, nur als Wirkungszusammenhang von verschiedenen Akteuren, Artefakten, Institutionen und diskursiven Strategien zu verstehen.

Für die bislang unbearbeitet gebliebene Geschichte der relationalen Datenbank sind die Voraussetzungen für eine kritische Differenzierung relativ günstig, da bereits das populäre Standardnarrativ mit einem einigermaßen komplexen Verlauf

der Entwicklung rechnet. Es modelliert die Figur eines unverstandenen »key theorist« Edgar F. Codd, dessen bahnbrechende Ideen von IBM kaum gewürdigt worden seien und dessen Vorschläge nur dank des Drucks eines universitären Konkurrenzprojekts das Licht des Marktes erblickt hätten.[21] Im Nachruf der New York Times auf Codd wurde diese Geschichte auf den Punkt gebracht: »While working as a researcher at the I.B.M. San Jose Research Laboratory in the 1960's and 70's, Dr. Codd wrote several papers outlining his ideas. To his frustration, I.B.M. largely ignored his work, as the company was investing heavily at the time in commercializing a different type of database system.« Seine Ideen seien für IBM zu »revolutionär« gewesen und hätten ihn – im Unterschied zu anderen – auch nie reich gemacht. »It was not until 1978 that Frank T. Cary, then chairman and chief executive of I.B.M., ordered the company to build a product based on Dr. Codd's ideas. But I.B.M. was beaten to the market by Lawrence J. Ellison, a Silicon Valley entrepreneur, who used Dr. Codd's papers as the basis of a product around which he built a start-up company that has since become the Oracle Corporation.«[22] Die Erzählung nutzt zwei beliebte Motive der Erfinder- und Innovationsgeschichte – das Versagen des mächtigen Marktführers und das Leiden des sympathischen Protagonisten[23] – und rechnet bereits in dieser handlichen Form mit Irrungen und Wirrungen, die über das Normalmaß einer an der rasanten Entwicklung orientierten Computerhistoriografie hinausgehen. Ausgeblendet wird dabei allerdings die Tatsache, dass die Schwierigkeiten bei der Entwicklung eines implementierbaren relationalen Datenbankmodells in direktem Zusammenhang standen mit einem großen informationspragmatischen Umbau. Die »Geschichte vom armen Codd« überschätzt jedoch nicht nur die Wirkung der Befehle eines IBM-CEO, es unterschätzt vor allem die enormen Veränderungen der Zirkulationsbedingungen von Wissen und den gewaltigen Umbau in der Landschaft der Akteure des Wissens, welche die Entwicklung der relationalen Datenbank gleichzeitig voraussetzte und bewirkte.

Die neue Gewaltenteilung

1970 erschien in den *Communications* der *Association for Computing Machinery* (ACM) ein Artikel, der in kürzester Zeit zum wichtigsten Referenzpunkt für eine ganze Entwicklergemeinschaft aufstieg. Unter dem Titel »A Relational Model of Data for Large Shared Data Banks« stellte der am IBM Research Laboratory in San José arbeitende Edgar F. Codd seine Überlegungen zu einer neuen Datenbankarchitektur vor.[24] Codds Artikel wird bis heute immer wieder als Meilenstein zitiert – vielleicht gerade wegen seiner abstrakten Argumentationsweise.[25]

Programmatisch war bereits der erste Satz des Artikels. »Future users of large data banks must be protected from having to know how the data is organized in the machine (the internal representation).«[26] Die bevorstehenden Veränderungen der Zirkulationsbedingungen des Wissens, davon ging Codd aus, machten es notwendig, die Kompetenzen der involvierten Akteure des Wissens neu festzulegen. So werde der zukünftige Betrieb von großen Datenbanken neue Anwendergruppen hervorbringen, die man davor »zu schützen« habe, die interne Organisation und Repräsentation der sie interessierenden Wissensbestände kennen zu müssen. Codds Forderung nach einem selektiven Wissensschutz war dem Umstand geschuldet, dass Abfragen komplexer Datenbanken sich sehr schnell zu anforderungsreichen Aufgaben entwickelten, die nur mit den ausgefeilten Kenntnissen eines professionellen Programmierers zu lösen waren. Denn große Datenbanken unterstanden nicht bloß einem ständigen Wandel der Datenrepräsentation, »as a result of changes in query, update, and report traffic and natural growth in the types of stored information.«[27] Sowohl hierarchisch aufgebaute Datenbanken als auch Datenbanken mit Netzwerkstruktur hatten den Nachteil, dass die Zugangswege zur Information in ihren Speicherstrukturen vordefiniert waren und neue Abfragen sich gezwungenermaßen entlang dieser Pfade bewegen mussten. Ohne eine genaue Kenntnis über den Verlauf dieser Pfade ließen sich neue Abfragen gar nicht durchführen. Codd versprach, mit einer relationalen Datenbankstruktur einen stark erweiterten, informationstechnisch inkompetenten aber abfragetechnisch urteilssicheren Kreis von zukünftigen Nutzern zu bedienen. »Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed and even when some aspects of the external representation are changed.«[28] Dadurch, dass sie sich nicht mehr um die Speicherungsform von Daten kümmern müssten, würden sich die Nutzer umso besser auf die immer flexiblere Abfrage von Daten konzentrieren können. Fachleute bräuchten sie dafür nicht mehr. Im Gegenzug würden die Programmierer für eine verlässliche Ablage und Verfügbarkeit, d.h. für die Organisation und Repräsentation der Daten zu sorgen haben – und wären damit ihrerseits vor den immer spezielleren Abfragewünschen der Datenbanknutzer geschützt. Dass sich damit – lange vor Larry Ellison – auf der User-Seite der Datenbank ein Übergang von der gezielten Suche nach abgespeicherten Daten hin zur generalisierten Abfrage von Datenbankbeständen, also ein Übergang von der Recherche zur Befragung des Orakels stattfand, fiel 1970 niemandem auf.[29]

Dennoch war die von Codd in die Zukunft projizierte Arbeits- und Gewaltenteilung an softwarearchitektonische Veränderungen gebunden, die sich zunächst nur theoretisch abschätzen und mathematisch formulieren ließen. Jedenfalls musste,

soviel war klar, die Datenbank der Zukunft so eingerichtet sein, dass ihre Bestände auch mit Fragen konfrontiert werden konnten, die sich bei der Planung der Datenbank noch gar nicht hatten stellen lassen. Codd machte es sich darum erstens zur Aufgabe, eine einfache und allgemeine Organisation der Daten in Tabellen zu entwickeln, die sich über Schlüssel untereinander verbinden ließen. Zweitens war ein Verwaltungsinstrument für die konsistente Veränderung von Einträgen und die Erweiterung der Datenbank zu bauen. Und drittens galt es, eine Abfragesprache zu schaffen, die mathematischen Ansprüchen genügte und dennoch möglichst nahe an der natürlichen Sprache jener Nutzer lag, die keine Kenntnisse im Programmieren hatten. Dass für diese Liste von Aufgaben mehr als ein Artikel Codds notwendig war, versteht sich von selbst. Allein die Klärung des Modells, seiner Vor- und Nachteile sowie der Differenzen zu konkurrierenden Entwürfen nahm mehrere Jahre intensiver Diskussionen in Anspruch.

Je konkreter Codds Modell beschrieben werden konnte, desto größer wurde die Verunsicherung unter jenen Datenbankspezialisten, die bis in alle Details mit herkömmlichen Modellen vertraut waren. Die Konfusion, die sich aus der Gleichzeitigkeit verschiedener Systementwürfe und aus den damit verbundenen unterschiedlichen Zukunftsvorstellungen ergab, war beachtlich.[30] Verwirrt und beunruhigt waren insbesondere jene, die an der privilegierten Rolle des Programmierers festhalten wollten und ihn weiterhin als professionellen Navigator durch die Gewässer komplexer Datenbestände verstanden.[31] Einer der Beunruhigten, Edgar F. Sibley, versuchte die Welt der Navigatoren dadurch wieder in Ordnung zu bringen, dass er Unterschiede und Gemeinsamkeiten der verschiedenen Entwürfe notierte und sortierte. »Practitioners of data base technology have been somewhat confused by the many different systems for describing and manipulating data. The two major approaches that have emerged may be termed the relational or set theoretic, and the data structured or procedural. There are obviously differences in these, but there are also similarities.«[32]

Die Sortierarbeit des gemäßigten »Prozeduralen« Sibley hielt die »Relationalen« nicht davon ab, weiter Öl ins Feuer zu gießen. Edgar F. Codd und Christopher J. Date taten dies etwa in einem Beitrag über den »Interactive Support for Nonprogrammers: The Relational and Network Approaches«[33] oder mit einer anderen gemeinsam verfassten Arbeit, die zwar vorgab, ebenfalls einen bloßen Vergleich der konkurrierenden Konzepte vorzustellen, in Tat und Wahrheit aber vor allem die spezifischen Vorteile und die radikale Konsequenz des eigenen Modells betonte: »It may be argued, in fact, that the relational model is a representation of the data in terms of its natural structure only – it contains absolutely no consideration of storage/access details (pointers, physical ordering, indexing or similar

access techniques…); in a word, no ›representation clutter‹.«[34] Mit solchen Sätzen wollten Date und Codd offensichtlich nicht nur die mathematischen und softwarehygienischen Vorzüge ihres relationalen Modells belegen, sondern nebenher auch jene selbstsicheren Datenbankspezialisten in Rage bringen, für die Pointers, Indizes und feste Pfade alles andere als »verwirrende Stördaten«, sondern gerade die Garanten einer effizienten Datenbankabfrage darstellten. Über eine zweite rhetorische Angriffslinie hoben Date und Codd die Vorteile für den technisch ungebildeten Nutzer, also für die Hausfrau und den Manager, hervor und behaupteten damit indirekt, die erweiterten Nutzermärkte der Zukunft könnten nur von ihrem relationalen Datenbankkonzept erobert und bedient werden. »The relational model is a particularly suitable structure for the truly casual user (i.e., a non-technical person who merely wishes to interrogate the database, for example a housewife who wants to make enquiries about this week's best buys at the supermarket). In the not too distant future the majority of computer users will probably be at this level.« Für jeden Datenbankspezialisten der frühen 1970er Jahre, der bislang all seine Programmierkünste in den Dienst einer Ressourcen schonenden, effizienten Abfragetechnik gestellt hatte, war das ein Horrorszenario. Doch Codd und Date fuhren fort, die Figur des zukünftigen Programmierers auf den »casual user« auszurichten, ihn also zu zwingen, die Folgen einer strikten »data independence«[35] und der damit verbundenen »user independence« zu akzeptieren. »The system must therefore be capable of supporting such a view at the casual user level – and this in itself is a strong argument for providing the same view at the programmer level, since any operation at the casual user level must be implementable at the programmer level too.«[36]

Die Debatte erreichte ihren Höhepunkt im Anschluss an das Panel »Data Models: Data-Structure Set vs. Relational«, zu welchem die Kontrahenten im Rahmen einer von der *Association for Computing Machinery* 1974 organisierten Konferenz aufgeboten und für die einige der bereits zitierten Artikel vorbereitet worden waren.[37] Codds prominentester Opponent war der ein Jahr zuvor mit dem Turing Award ausgezeichnete Charles W. Bachman. Das Protokoll ihrer großen Auseinandersetzung spiegelt eine angestrengt kontrollierte Diskussion, die ständig außer Kontrolle zu geraten drohte. Bereits in seiner zweiten Wortmeldung machte Codd klar, dass die Differenzen enorm waren: »With the relational model we have well-known high-level logics to act as the target temporarily to get the query accurately and precisely defined – whereas with networks, as far as I am aware, there are no comparable logics available at that level to pin down the semantics of the casual user's transactions. I would like to be corrected if I am wrong.«[38] Korrigiert wurde er daraufhin so oft, dass er sich gegen Ende der Debatte sogar gegen die Unterstel-

lung glaubte wehren zu müssen, sein relationaler Zugang hätte religiöse Züge.[39] Tatsächlich offenbarte die Debatte zwischen Codd und Bachman über weite Strecken so grundsätzliche Differenzen, dass der wechselseitige Fundamentalismus-Verdacht nicht von der Hand zu weisen war. Die Diskussionen blieben von Misstrauen und gegenseitiger Skepsis geprägt. Selbst dort, wo Codd den Versuch unternahm, die Gegensätze etwas sorgfältiger zu sortieren und die Polemik auf ein erträgliches Maß zu reduzieren, glaubte Bachman noch immer mit Sicherheit eine Finte erkennen zu können – »I think we have just seen the red herring that we want to avoid«, war seine Antwort auf Codds scheinbar sachliche Frage, ob Daten irgendeine Form von Speicher- und Zugangsinformationen enthalten sollten oder nicht. »There are all kinds of logical and arithmetic semantic constraints that you may want to impose on a data base«, hatte Codd eingeräumt. »I think it is a debatable point as to whether some of these should be in the data structure and some in separate declarations.« Die für ihn entscheidende Frage sei nur, wo dies zu geschehen habe. »You cannot devise in advance a set of data types that will give you all the possible semantic checks that any data base could require. The issue is whether you should put some of the semantic constraints in the data structure of your principal schema and some elsewhere, or all elsewhere.«[40]

Dass Codds Polemik bisweilen solch subtile Formen annahm, verbesserte das Diskussionsklima nicht, zumal er sonst nur wenige Gelegenheiten ungenutzt ließ, im offenen Schlagabtausch Punkte zu machen. Seitenhiebe wie »I am rather surprised by your military, that is, hierarchical approach to security«[41] waren keine Seltenheit. Immer wieder wurde das Recht des Nutzers gegen die Dominanz des Programmierers ausgespielt. Dazu gehörte insbesondere Codds Attacke auf die betonte Programmiererfixierung der Database Task Group der Conference on Data Systems Languages (CODASYL) – »I am at loss to understand why there has not been more emphasis during the past six years by CODASYL on users other than the application programmer.«[42] Als dieser Angriff Codds von Edgar Sibley etwas schwach mit dem Hinweis pariert wurde, es gebe doch bei CODASYL eine *End Users Facilities Task Group*, die schon seit vier Monaten am Arbeiten sei, setzte Codd noch einen Schlag nach: »The fact that this group has been established so recently supports what I just said.«[43]

Unter diesen Bedingungen musste die Debatte ohne erkennbare Lösung enden. Klar war nur, dass es bei Bachman et al. um den zielorientierten, gut kontrollierten Einsatz technischer Ressourcen, bei Codd et al. um ein größtmögliches Angebot an Deutungsautonomie für eine zunehmend heterogene Nutzergemeinschaft ging. Die Frage nach Einschränkung und Erweiterung des Datenbankzugriffs, das Verhältnis zwischen Kontrollmacht der Programmierer und Abfrageflexibilität der

Anwender sowie der Trade-off zwischen technischer Performanz und mathematischer Eleganz waren weitere Themen, mit deren Hilfe sich die Fronten zwischen den beiden Lagern stabilisieren ließen. Bachman selber hat in einem längeren Interview, das er Thomas Haigh 2004 gab, weder seine Skepsis gegenüber relationalen Datenbanken revidiert (»I'm still skeptical«)[44] noch Codd angesichts des ökonomischen Erfolgs seines Konzepts *ex post* Recht gegeben. Vielmehr hat Bachman darauf hingewiesen, dass hinter den beiden zerstrittenen Parteien zwei mächtige Konkurrenten gestanden hätten – IBM auf Codds, Honeywell auf Bachmans Seite.[45]

Bachmans Argument zu folgen hieße, dass das relationale Modell von einem seit jeher auf den Dienstleistungssektor ausgerichteten Konzern gestützt wurde, während das prozedural orientierte Modell in einem von Chemie, Rüstung und Prozessautomation geprägten industriellen Mischkonzern gefördert wurde. Diese unternehmenshistorische Erklärung ist jedoch nicht überzeugend. Erstens boten sowohl IBM als auch Honeywell ihren Kunden erfolgreiche prozedurale Datenbanksysteme an: Honeywell eine Weiterentwicklung des CODASYL-basierten Netzwerkdatenbanksystem IDS, und IBM das 1968 eingeführte hierarchisch strukturierte IMS.[46] Honeywell und IBM kämpften zwar auf demselben Markt um Kunden; sie hatten zur Idee eines relationalen Datenbankmodells jedoch die gleiche Distanz. Zweitens hat ausgerechnet Bachmans Arbeitgeber Honeywell bereits 1976 mit Multics Relational Data Store (MRDS) das erste relationale *database management system* angeboten, allerdings nur im Paket mit Multics Integrated Data Store, welches auf dem CODASYL-Modell beruhte.[47] Drittens gab es sowohl für Bachman als auch für Codd Gründe, sich nicht nur bei Fragen zur zukünftigen Gestaltung von Datenbanken von den gegenwärtigen Produkten ihrer Arbeitgeber zu distanzieren. So hat Bachman seine wichtigste berufliche Sozialisation gar nicht bei Honeywell, sondern in den 1950er Jahren bei Dow Chemical und in den 1960er Jahren bei General Electrics durchlaufen (1961 bis 1970). Zur Zeit der großen Debatte war er erst vier Jahre bei Honeywell tätig und verließ die Firma 1980 wieder. Codd wiederum war zwar durchaus ein IBM-Mann, aber die Distanz von Mitgliedern der Research Labs zum Konzern und die von Codd und seinem Umfeld verbreitete Geschichte, IBM habe seine Vorschläge zu wenig unterstützt, passen nicht zu einer bedingungslosen Identifizierung mit den Anliegen des Konzerns.

Die Gegensätze zwischen den beiden Lagern spiegeln sich jedoch in den völlig unterschiedlichen beruflichen Laufbahnen ihrer beiden Protagonisten. Da stand auf der einen Seite der Ingenieur »Charlie« Bachman, der als Macher den praktischen Übergang eines Industriebetriebs ins Computerzeitalter schon bei General Electrics mitgestaltet hatte. Sein Interesse galt funktionierenden Systemen und

pragmatischen Problemlösungen. Kosteneffizienz musste auch beim Betrieb eines Rechenzentrums das oberste Ziel bleiben. Und das wiederum hieß nichts anderes, als Datenbanken so zu betreiben, dass sie möglichst nahtlos mit den Anwendungsprogrammen interagieren konnten. Dafür musste man als Programmierer sowohl die Daten als auch die Anwendungen möglichst genau kennen und unzählige Tricks anwenden, die das präzise aufeinander abgestimmte Zusammenspiel vor dem Zusammenbruch bewahrten.[48]

Auf der anderen Seite agierte der britische Mathematiker Ted Codd, der nach dem Kriegsdienst für die Royal Air Force bei IBM in New York gearbeitet hatte. Während der McCarthy-Ära wanderte er nach Kanada aus. Nach seiner Rückkehr studierte er an der University of Michigan, erwarb dort einen PhD und wechselte anschließend 1967 in die geschützte Werkstatt des IBM Forschungslabors im kalifornischen San José. Für Codd mussten Datenbanken mathematisch beschreibbar sein, die dazugehörige Theorie schön und einfach bleiben.[49]

Bachman und Codd stehen also nicht für unterschiedliche Corporate Identities, sie verkörpern vielmehr den kulturellen Gegensatz zwischen dem Techniker und dem Akademiker bzw. zwischen dem Pragmatiker und dem Theoretiker. Ihr Selbstverständnis hätte unterschiedlicher nicht sein können. Entsprechend unterschiedlich waren ihre Vorstellungen darüber, wer in Zukunft mit Datenbanken hantieren sollte, wer sie zu konstruieren, zu unterhalten, zu überwachen und zu nutzen hatte. Dieser Unterschied reflektiert das Verhältnis des Nutzers zur Maschine. Während technisch versierte Nutzer ihre Maschinen selber revidieren können, sind technisch unerfahrene Anwender darauf angewiesen, dass sich ihnen die Maschine als Blackbox präsentiert. Von Technikern bediente Maschinen können jederzeit verbessert und ergänzt werden; Systeme, die von Managern und anderen »casual users« bedient werden, müssen hingegen stabil laufen. Während sich Bachman für die mit entsprechendem Fachwissen jederzeit auf spezifische Anwendungen optimierbare Datenbankmaschine stark machte, setzte sich Codd für ein aus der Sicht zukünftiger Nutzer solid gebautes, aber multifunktionales Datenbanksystem ein.

Relationalität in der Bay Area

Um die Mitte der 1970er Jahre waren die zentralen Begriffe für Codds Modell geklärt und stabil verfügbar geworden. Die Charakteristika einer relationalen Datenbank stellten Astrahan und Chamberlin – selbstredend mit einer großen Referenz an den Gründungsvater der Relationalitätsgemeinschaften – wie folgt zusammen: »In a series of papers, Codd[50] has introduced the relational model of

data and discussed its advantages in terms of simplicity, symmetry, data independence, and semantic completeness.«[51] Alle Daten eines relationalen Datenbanksystems müssten durch ein zusammengehörendes Set von klar bezeichneten Tabellen, sogenannten Relationen dargestellt werden können. Innerhalb jeder Relation gebe es eindeutig bezeichnete Spalten. Die Ordnung der Reihen spiele keine Rolle, aber jede Reihe stelle ein adressierbares Element der von der Relation beschriebenen Entität dar. Sie müsse von anderen unterscheidbar sein und dürfe nur einmal vorkommen. Zusätzlich habe jede Relation eine Spalte, die als Primärschlüssel bezeichnet werde.[52] Was »data independence« theoretisch hieß, war in den Diskussionen der vergangenen Jahre, gerade in der Auseinandersetzung mit den Anhängern der prozeduralen Datenbankarchitektur, klar geworden. Was das von ihr verfolgte Ziel einer erhöhten »user independence« alles erforderlich machte, zeigte jedoch erst die konkrete Entwicklungsarbeit.

Zwischen 1974 und 1979 wurde am IBM Forschungslaboratorium im kalifornischen San José an einem Projekt gearbeitet, das später als »System R« bekannt geworden ist.[53] In einer ersten Phase wurde in den Jahren 1974 und 1975 eine »Structured English Query Language« (SEQUEL, später SQL) entwickelt, mit der ein zukünftiger Nutzer seine Abfragen an einem interaktiven Terminal formulieren konnte.[54] Große Bedeutung wurde dabei dem »human factor« beigemessen, und es wurden verschiedene experimentelle Studien zur Lernbarkeit und zur Nutzbarkeit des Systems durchgeführt. In der zweiten Projektphase (1976 und 1977) ging es darum, das System für mehrere, gleichzeitig arbeitende Nutzer umzubauen und die vorhandene SQL so anzupassen, dass sie auf verschiedenen Systemen angewendet werden konnte. Nutzer, die mit den Programmiersprachen PL/I und Cobol vertraut waren, sollten die gleichen Möglichkeiten haben und die gleiche Syntax verwenden, wie die so genannten »ad hoc query users«.[55] 1978 und 1979 wurden dann im Konzern und bei drei Kunden Tests im Betrieb durchgeführt und die Erfahrungen der Anwender evaluiert.

»System R« war zweifellos ein großes Experiment der IBM, das eine demonstrative Nutzerorientierung an den Tag legte. Der Bau von »user friendly interfaces«[56] sowie eine konsequente Modularisierung des Systems waren die wichtigsten Strategien, mit denen man die Projektziele zu erreichen suchte.[57] Die Arbeit an der Unabhängigkeit von Daten und Nutzern bereitete den Entwicklern aber viel Kopfzerbrechen. Während die Abfragesprache in ihrer zweiten Version recht gut zu funktionieren schien, kämpften die Entwickler von »System R« nach wie vor mit der Frage, wie sie einen funktionalen Ersatz für die in prozeduralen Datenbanksystemen verwendeten Zusatzinformationen über die Strukturierung der Daten bereitstellen konnten, ohne die Anwender in irgendeiner Weise damit zu belasten.

Auch das für Bachman zentrale Problem der Performance – für den Betrieb einer relationalen Datenbank musste eine gewaltige Speicher- und Verarbeitungskapazität vorausgesetzt werden können – war kaum aus der Welt zu schaffen. Ein wichtiger Schritt zur Lösung war die Verwendung des von Rudolf Bayer und Edward M. McCreight bereits 1972 vorgestellten Konzepts der B-Bäume, mit deren Hilfe man die gespeicherten Daten wesentlich effizienter indexieren konnte. Dadurch ließen sich die bei einer Abfrage zu erwartenden Schreib- und Lesevorgänge um eine Größenordnung reduzieren, ohne deshalb die Datenstruktur an die Abfragemöglichkeit koppeln zu müssen.[58]

Die Geschichte vom armen Ted Codd will es, dass die IBM viel zu spät oder überhaupt nicht auf ihn gehört habe und die Entwicklung eines marktfähigen relationalen Datenbanksystems deshalb um Jahre verzögert worden sei. Eine kritische Lektüre der Geschichte von »System R« erlaubt eine andere Interpretation. Zwar wurde während der Entwicklung von »System R« stets der experimentelle Status des Projekts als »feasability study« betont: »At each user site, System R was installed for experimental purposes only, and not as a supported commercial product.«[59] Dennoch hatte das Projekt eine stattliche Zahl von Mitarbeitern und wurde in einer großen, funktional ausdifferenzierten Gruppe bearbeitet.[60] Innerhalb von IBM war »System R« also deutlich mehr als das Hobby einiger weniger Entwickler. Besonders auffällig ist, dass IBM sehr eingehende Beschreibungen der Entwicklungsarbeiten an »System R« zuhanden der**#für die (Duden: Helvetismus)#** Technikercommunity publizierte.[61] Das konnte nur heißen, dass der Status von »System R« innerhalb von IBM besser war, als es Codd gerne dargestellt haben wollte. Weil dem Projekt noch keine strategische, sprich: kommerzielle und damit geheime Bedeutung zukam,[62] durfte es mit der Unterstützung von außen rechnen – die »System R«-Publikationen waren Einladungen zum Mitdenken. Die beliebte These, wonach es vor allem der Erfolg des an der Universität Berkeley laufenden Projekts INGRES war, welcher IBM »endlich« zum Handeln zwang, muss umformuliert werden.[63] Ganz offensichtlich suchten die »System R«-Entwickler Gesprächspartner in anderen Entwicklungszentren außerhalb von IBM, und fanden sie ab 1975 auch tatsächlich in der näheren Umgebung, nämlich an der University of California in Berkeley, wo unter der Leitung von Michael Stonebraker und Eugene Wong seit 1973 an einem Prototypen für ein relationales Datenbanksystem gearbeitet wurde. Es ist nicht weiter erstaunlich, dass Informatiker an einer Universität die Anregungen Codds aufgenommen haben, waren seine Ideen doch in den frühen 1970er Jahren genau in jenem mathematisch interessanten, vorkommerziellen Bereich angesiedelt, der eine akademische Beschäftigung rechtfertigte.

Die Bedingungen der beiden Projekte in Berkeley und in San José hätten unterschiedlicher nicht sein können: Akademische Beweglichkeit auf der einen Seite, professionelle Projektmanagementkultur auf der anderen Seite. Hier ein schlanker PDP-11 Rechner auf noch wackliger UNIX Basis, dort die mächtigen IBM 370 Mainframes mit verschiedenen hochstabilen Betriebssystemen.[64] Eine ständig wechselnde studentische Mitarbeiterschaft in Berkeley – »a collection of goofy academicians«[65] – stand den geregelten Anstellungsverhältnissen erfahrener Programmierer und Projektmanager in San José gegenüber. Stonebrakers auffällig ironische Selbstdarstellung des INGRES-Projekts in Berkeley steht in scharfem Kontrast zum betont seriösen Publikationsstil jener IBM-Ingenieure, die sich mit »System R« beschäftigten. Liest man die Berichte von Stonebraker über den Verlauf seines Projekts, dann wird klar, dass INGRES mit wesentlich größeren Schwierigkeiten zu kämpfen hatte als das bei IBM entwickelte »System R«. Allein die Beschaffung eines Rechners und der Projektmittel, aber auch der schnelle Wechsel der am Projekt beteiligten Studenten, bereiteten Stonebraker und Wong große Schwierigkeiten. Wenn es aber einen Konkurrenzdruck zwischen IBM und dem universitären Projekt INGRES gab, dann dürfte dieser in jenem Moment zu spüren gewesen sein, als Edgar F. Codd nach Berkeley fuhr, um zu schauen, woran Stonebraker und sein Team gerade arbeiteten. Stonebraker erinnerte sich später mit Schrecken an den Besuch des IBM-Gurus: »In January 1975 we invited Ted Codd to come to Berkeley in early March to see a demonstration of INGRES. The final two weeks before his visit everyone worked night and day so that we would have something to show him. What we demonstrated was a very ›buggy‹ system«.[66]

Obwohl die Teams durchaus dieselben Themen diskutierten, kamen sie sehr oft zu unterschiedlichen Schlussfolgerungen. So wurde die Frage, ob statische oder dynamische Ablageverfahren angewandt werden sollten, paradoxerweise vom personell dynamischen Team in Berkeley mit der Wahl einer statischen Lösung beantwortet (was später als Fehler eingestuft wurde), während die im notorisch unbeweglichen IBM-Konzern arbeitenden Entwickler von »System R« ausgerechnet die dynamische Datenstruktur der B-Bäume bevorzugten. Weder die eine noch die andere Seite hatte sich die Entscheidung darüber leicht gemacht und die Überlegungen, die sie für bzw. gegen die Verwendung von B-Bäumen angestellt hatten, in spezifischen Papers publiziert.[67] Die Experimente und Versuchsanordnungen beider Teams erhöhten aber die insgesamt verfügbare Erfahrung der virtuellen Entwicklergemeinschaft im Lager der »Relationalen«.[68]

Der genauere Vergleich zwischen den beiden Projekten bringt jedoch auch eine ganze Reihe von Gemeinsamkeiten hervor. Abgesehen davon, dass beide Teams angetrieben wurden von der theoretischen Faszination des relationalen Modells –

der dogmatische Referenzpunkt Ted Codd war an beiden Orten unbestritten[69] –, investierten sowohl INGRES als auch »System R« enorme intellektuelle Ressourcen in ihr Projekt. So stark sich ihre Arbeitsweise und Arbeitsbedingungen auch unterschieden, die Arbeiten haben sich immer wieder ergänzt und wechselseitig befruchtet. In beiden Projekten wurden schwerwiegende strategische Fehler gemacht, abenteuerliche Abkürzungen gewählt, die man später bereute und dann auch in entsprechend selbstkritischen Berichten öffentlich darstellte. Dass INGRES und »System R« mehr versprachen, als sie je einlösen konnten und dass ihre Leiter hinterher behaupteten, mehr gemacht zu haben, als es tatsächlich der Fall gewesen sein dürfte, gehört nur zum Wesen jeder Projekt- und Berichtprosa, nicht zu einem harten Konkurrenzkampf. Was Stonebraker für INGRES festhielt, galt in abgewandelter Form auch für »System R«: »Our goals expanded several times (always when we were in danger of achieving the previous collection).«[70]

Beide Teams betrieben einen großen Aufwand bei der Entwicklung einer »high-level search and query language«. Dies war für die Zukunft des relationalen Modells von entscheidender Bedeutung, denn von der Nutzerfreundlichkeit dieser Abfragesprache hing die Erweiterung der technisch nichtversierten Nutzergemeinde ab. Mit der in Berkeley entwickelten Abfragesprache QUEL und mit den aus San José stammenden Abfragesprachen ALPHA, SQUARE, SEQUEL und SQL verfolgte man das große Ziel der »user« und »data independence« und gab einem neuen Typ von Datenbankbenutzer ein mächtiges Instrument in die Hand. Das weitaus überzeugendste Beispiel für die präzedenzlose Macht sowohl einer »high-level search and query language« im Speziellen wie dem relationalen Datenbankmodell im Allgemeinen findet sich in einem Artikel von Stonebraker aus dem Jahre 1976, aus einer Zeit also, in der die Arbeiten an Relationalität quer zur Bay Area auf Hochtouren liefen. Stonebrakers Beispiel war folgendes: Man stelle sich ein Unternehmen vor, das mit zwei Relationen beschrieben werden kann. In einer ersten Relation oder Tabelle »EMP«, wie sie typischerweise in der Personalabteilung einer Firma geführt wird, finden sich Angaben zum Namen, zur Abteilung, zum Gehalt, zum Vorgesetzten und zum Alter der Angestellten. In einer zweiten, davon unabhängigen Relation oder Tabelle »DEPT« der Raumverwaltungsabteilung wird die Zuweisung der Abteilungen auf die verschiedenen Geschosse im Gebäude des Unternehmens festgehalten. Die beiden Relationen lauten also:

EMP (NAME, DEPT, SALARY, MANAGER, AGE)
DEPT (DEPT, FLOOR#)

Eine dreizeilige QUEL-Abfrage eines mit der Restrukturierung des Unternehmens beauftragten Managers könnte nun wie folgt lauten:

```
RANGE OF E IS EMP
RANGE OF D IS DEPT
DELETE E WHERE E.DEPT = D.DEPT AND D.FLOOR# = 1
```

Geändert würde dabei die Relation EMP und damit ein Anstellungsverhältnis. Als betriebswirtschaftliche Handlungsanweisung hätte diese Operation folgendem Klartext entsprochen: »Fire everybody on the first floor.«[71] Die zynische Empathie, welche das INGRES-Team mit diesem Beispiel für die Rolle eines Managers an den Tag legte, ist wohl nicht zu übertreffen. Etwas feinfühliger lautete die Abfrage, welche das »System R«-Team bereits ein Jahr zuvor in SEQUEL aufgeschrieben und publiziert hatte:

```
SELECTNAME
FROM E IN EMP
WHERE SAL >
SELECT SAL
FROM EMP
WHERE MNO = E.MGR;;
```

Die dazugehörige Frage in der real existierenden Managerwelt lautete: »Find names of employees who earn more than their manager.«[72]

Beide Fragen denken sich in einen neuen Datenbanknutzer hinein, versuchen ihn durch die Attraktivität der leistungsfähigen Abfragemöglichkeit zu gewinnen und illustrieren die Art und Weise, wie relationale Datenbanken auch in kürzester Zeit Fragen liefern können, für deren Beantwortung Relationen genutzt werden müssen, die nicht einmal die Frage hätten antizipieren können. Dass solche, quer zu den bereits erfassten Daten eines Unternehmens gestellte Fragen insbesondere Manager ansprachen, versteht sich von selbst. Denn Manager mussten seit den guten alten Zeiten des Scientific Management mit möglichst großer informationstechnischer Raffinesse Zusammenhänge aufdecken bzw. konstruieren, die quer zur organisatorischen Struktur eines Unternehmens verliefen und deshalb nirgends in speziell angefertigten Tabellen erfasst sein konnten.

Perspektiven der Relationalität

Die Arbeiten an der relationalen Datenbank erweckte alte Hoffnungen zu neuem Leben. Bereits in den 1960er Jahre hatten Unternehmensberater damit gerechnet, in absehbarer Zeit über ein leistungsfähiges »Management Information System« verfügen zu können. In einem einzigen Topf sollten all jene Informationen gesammelt werden, die bisher aus einer Vielzahl verstreuter Berichte, Formulare und Memoranden mühsam zusammengesucht werden mussten. Manche träumten sogar von einer zukünftigen »electronic data bank, or pool of information, from which reports of many types can be drawn.«[73] Dafür sollten die Manager nicht Computer- oder gar Programmierkurse belegen müssen. »Der Computer kann nützlich sein, aber nur, wenn wir ihm die Fragen stellen, auf die es ankommt. Das ist die Aufgabe des Managements, und die Wissenschaft hilft uns dabei, diese Fragen zu präzisieren«, hatte Alfred Eisenpreis 1966 an einer Tagung über Wissenschaft und Handel festgehalten. »Das Management soll nur wissen, welche Fragen zu stellen und wie die Antworten zu bewerten sind«, sagte der prominente Betriebswissenschaftler.[74] Der Traum vom Computer als einem mächtigen Instrument des Managements, das nur noch kluge Fragen an ein großes Reservoir von Daten zu stellen hatte, schien Mitte der 1970er Jahre dank der Entwicklungsarbeit an relationalen Datenbanken Wirklichkeit werden zu können. Ein Unternehmen sollte in absehbarer Zeit wie ein offenes Buch vor den Augen seiner Manager liegen.

Der Computer war von einem schnell rechnenden Rationalisierungsinstrument zu einem betriebswirtschaftlich nutzbaren Restrukturierungswerkzeug mutiert. Dank der Vielfalt der mittels rechnergestützter Datenbanksysteme herstellbaren Zusammenhänge steigerte er nicht mehr bloß die Leistungsfähigkeit der Datenverarbeitungskapazität eines Unternehmens. Durch die Anwendung der neuen *ars combinatoria,* die von relationalen Datenbanken offeriert wurden, versprach der Computer nun auch die unternehmensinternen Transaktionskosten zu senken. Selbst dem mittleren Management konnten in absehbarer Zeit zu tiefen Abfragekosten Quervergleiche über Tabellen- und Abteilungsgrenzen hinweg möglich gemacht werden. Dass dieses Angebot in eine Zeit fiel, in der die Restrukturierung von ganzen Unternehmungen zum alltäglichen Problem geworden war, erhöhte die Attraktivität relationaler Datenbanktechnik in den späten 1970er und den frühen 1980er Jahren dramatisch, insbesondere als mit Oracle auch auf kleineren Rechnern relationale Datenbanksysteme implementierbar geworden waren.

Die Steigerung der betriebswirtschaftlichen Deutungsautonomie, oder allgemeiner: die erhöhte interpretatorische Freiheit des Nutzers von Datenbanken fiel aber auch in eine Zeit, in der die Vorstellungen darüber, was Deutung überhaupt ist

und nach welchen Regeln sie erfolgen kann, in grundsätzlicher Weise verändert worden sind. Am Beispiel der Interpretation von Texten, deren »user« man bekanntlich auch als Leser bezeichnen kann, ist diese Frage in den 1970er Jahren *in extenso* diskutiert worden. Wie schrieb Roland Barthes 1970 in *S/Z*? »Einen Text interpretieren heißt nicht, ihm einen (mehr oder weniger begründeten, mehr oder weniger freien) Sinn geben, heißt vielmehr abschätzen, aus welchem Pluralem er gebildet ist.«[75] Der Text ist bei Barthes eine »Galaxie von Signifikanten«, die in seinem Gewebe unendlich komplex und vielfältig zueinander in Beziehung treten. Interpretation heißt also nicht, mit hermeneutisch geschulten Abfragetechniken jenen ursprünglichen Sinn zu eruieren, den ihm ein Autor möglicherweise gegeben haben wollte. Vielmehr ist der Text eine Maschine zur Produktion von Interpretationen, wie Umberto Eco einmal gesagt hat.[76] Die Trennung von Autor und Leser, die sich aus dieser Vorstellung von Text ergibt, ist so strikt wie die Trennung von Programmierer und Nutzer. Auch der Text wird, um nochmals Barthes zu zitieren, stets »durch mehrere Zugänge« erschlossen, »von denen keiner mit Sicherheit zum Hauptzugang gemacht werden könnte.«[77]

Bereits Umberto Ecos Untersuchung »Das offene Kunstwerk« von 1962 und Susan Sontags Essay »Against Interpretation« von 1964 waren erste Schritte zur Kritik des hermeneutischen Verfahrens gewesen. Um 1970 ließ sich diese Bewegung nicht einmal mehr mit der Rückversicherung auf Gadamers »Wahrheit und Methode« aufhalten – die gemeinsame, sinnstiftende kulturelle Einheit von Autor und Leser löste sich auf.[78] Während Michel Foucault die Frage nach dem Autor[79] zum Anlass nahm, über diskursanalytische Bedingungen und die Ordnung des Sagbaren nachzudenken, und Roland Barthes schlicht den Tod des Autors proklamierte,[80] sprach Wolfgang Iser von der »Unbestimmtheit als Wirkungsbedingung literarischer Prosa«.[81] Die Figur des emanzipierten und selbständigen Lesers, der sich seinem Text gegenüber freier als je zuvor bewegen konnte, hatte Konjunktur. Eine flankierende Maßnahme dafür war, die Möglichkeitsbedingungen jeder interpretatorischen Arbeit, nämlich die Präsentation des Textes, zu verändern. Dieter E. Sattler stellte deshalb die Forderung auf, dass selbst klassische Texte so abgedruckt werden müssten, dass die Deutungsautonomie ihrer Leser nicht eingeschränkt, sondern unterstützt würde.[82] Metaphorisch gesprochen veränderte er damit die Struktur der Datenbank, auf die sich jede Deutung zu stützen hatte. Auf den (hochpolitischen) Streit um das »richtige« Verständnis der Texte Hölderlins antwortete Sattler in seiner Frankfurter Ausgabe mit einer philologischen und drucktechnischen Innovation. Die von ihm präsentierte Ausgabe sollte die flexible Kombination von Deutungselementen ermöglichen, weil das »Lesebedürfnis kritischer, selbständiger und anspruchsvoller geworden« sei.[83] Nur so würden sich auch jene

Textschichten und Beziehungen zwischen Textteilen freilegen lassen, die dem Leser eine hinreichende Flexibilität in der *ars combinatoria* seiner Lektüre bzw. Textabfrage ermöglichte.

»Wenn die zeitgenössischen Poetiken mit Strukturen des Kunstwerks arbeiten, die eine besondere selbständige Mitwirkung des Rezipierenden, oft eine stets variable Rekonstruktion des angebotenen Materials verlangen, so spiegeln sie eine allgemeine Tendenz unserer Kultur in Richtung auf jene Prozesse, bei denen sich, statt einer eindeutigen und notwendigen Folge von Ereignissen, ein Möglichkeitsfeld, eine ›Ambiguität‹ der Situation ausbildet, so dass von Mal zu Mal verschiedene operative oder interpretative Entscheidungen ausgelöst werden«, hatte Umberto Eco in seinem »Offenen Kunstwerk« geschrieben.[84]

Mit Texten, davon war man in den 1970er Jahren überzeugt, ist es wie mit Datenbanken. Beide hatten sowohl theoretisch als auch pragmatisch eine Rekonfiguration durchlaufen und verlangten nach variablen Rekonstruktionen des angebotenen Materials, ließen sich als mehrdeutiges Möglichkeitsfeld verstehen, das variable operative und interpretative Prozeduren und Entscheidungen zulässt. Weder sollte ihre Präsentation so beschaffen sein, dass ihre Deutung nur in eingeschränkter, vorgespurter Weise möglich bleibt, noch können sie für sich selber sprechen. Das wieder aber haben sie mit jenen elektronischen, biologischen und materiellen Datenbanken gemeinsam, welche die Welt der forensischen Spezialisten in CSI ausmachen. Nur über die Abfrage dieses informationellen Möglichkeitsfeldes lassen sich Zusammenhänge simulieren, überprüfen und erkennen. Dafür braucht es spezielle Technologien, Verfahren und Sprachen, welche aus vorhandenen Daten neuen Sinn generierten. Daten sprechen nie für sich selber. Darum antwortet der Laborleiter von CSI Las Vegas auf die Frage, warum er als forensischer Spurensucher arbeite: »Because the dead can't speak for themselves.«[85]

Anmerkungen

1 Ich danke Daniela Zetti und Lea Haller für logistische Unterstützung und kritische Hinweise, welche mir die Abfassung des vorliegenden Aufsatzes ermöglicht haben.

2 Adams, Guy: »CSI: The Cop Show that Conquered the World«, in: *The Independent*, 19. Dezember 2006.

3 http://www.sat1.de/filme_serien/ris/serie/, 1. Mai 2007.

4 http://www.sat1.de/filme_serien/ncis/serie/, 1. Mai 2007.

5 Krogerus, Mikael: »Der Serientäter. Interview mit Steven Bochco«, in: *NZZ Folio*, Nr. 10, 2006. Vgl. auch Kramp, Leif und Stephan Alexander Weichert: »Im Schatten des CSI-Effekts. TV-Serien verdrängen den Kinofilm«, in: *Medienheft*, 22. Januar 2007.

6 Schweitzer, Nicholas J. und Michael J. Saks: »The CSI Effect. Popular Fiction About Forensic Science Affects Public Expectations About Real Forensic Science«, in: *Jurimetrics* 47, 2007, S. 1–8; Mann, Michael D.: »The ›CSI Effect‹. Better Jurors through Television and Science?«, in: *Buffalo Public Interest Law Journal* 24, 2006, S. 157–183; Shelton, Donald E. et al.: »A Study of Juror Expectations and Demands Concerning Scientific Evidence. Does the ›CSI Effect‹ Exist?«, in: *Vanderbilt Journal of Entertainment & Technology Law* 9, 2006, S. 331–368; Tyler, Tom R.: »Viewing CSI and the Threshold of Guilt. Managing Truth and Justice in Reality and Fiction«, in: *Yale Law Journal* 115, 2006, S. 1050–1085; Podlas, Kimberlianne: »The C.S.I. Effect. Exposing the Media Myth«, in: *Media and Entertainment Law Journal* 16, 2006, S. 429–465.

7 Dazu gehören neben CSI und seinen bereits genannten Derivaten etwa *Without a Trace* (2002), *Cold Case* (2003), *NCIS* (2003), *Criminal Minds* (2005), *Shark* (2006).

8 Eine Ausnahme bildet eine Besprechung von Norbert Schneider in der FAZ: Schneider, Norbert: »Fernsehserie ›CSI‹. Der Mensch als toter Körper«, in: *Frankfurter Allgemeine Zeitung*, 21. Februar 2006, S. 48.

9 Mokrzycki, Gregg M.: »Advances in Document Examination. The Video Spectral Comparator 2000«, in: *Forensic Science Communications*, 22. Juni 2007. http://www.fbi.gov/hq/lab/fsc/backissu/oct1999/mokrzyck.htm, 1. Mai 2007.

10 http://www.mushroomsoftwareonline.com/Slip1.htm, 1. Mai 2007. SLIP ist seit 1995 erhältlich. »Mushroom Software was started in 1994. In 1994 we created a Modus Operandi Database and a Report Writing System to test the acceptance and feasibility of mobile/Laptop police operations.« http://www.mushroomsoftwareonline.com/, 1. Mai 2007.

11 Dass Columbo noch immer eine Fangemeinde hat, ist damit natürlich nicht bestritten. Vgl. Block, Armin und Stefan Fuchs: *Columbo. Das große Buch für Fans*, Berlin 1998.

12 Gugerli, David et al.: *Die Zukunftsmaschine. Konjunkturen der Eidgenössischen Technischen Hochschule Zürich 1855–2005*, Zürich 2005, S. 297–304; Piore, Michael J. und Charles F. Sabel: *Das Ende der Massenproduktion. Studie über die Requalifizierung der Arbeit und die Rückkehr der Ökonomie in die Gesellschaft*, Frankfurt a.M. 1989; Sennett, Richard: *Der flexible Mensch. Die Kultur des neuen Kapitalismus*, Berlin 1998; Gugerli, David: »Die Entwicklung der digitalen Telefonie (1960–1985). Die Kosten soziotechnischer Flexibilisierungen«, in: Kurt Stadelmann, Thomas Hengartner und Museum für Kommunikation Bern (Hg.): *Telemagie. 150 Jahre Telekommunikation in der Schweiz*, Zürich 2002, S. 154–167; Girschik, Katja: »Machine-Readable Codes. The Swiss Retailer Migros and the Quest for Flow Velocity since the mid-1960s«, in: *Entreprises et histoire* 44, 2006, S. 55–65; Hürlimann, Gisela: *Die Eisenbahn der Zukunft. Automatisierung, Schnellverkehr und Modernisierung bei den SBB, 1955–2005*, Zürich 2007.

13 Foucault, Michel: »Espace, savoir et pouvoir«, in: Defert, Daniel und François Ewald (Hg.): *Dits et écrits 1954–1988 par Michel Foucault, Bd. IV, 1980–1988*, Paris 1994 (1982), S. 270–285, hier S. 277.

14 Castells, Manuel: *Das Informationszeitalter I. Der Aufstieg der Netzwerkgesellschaft*, Opladen 2001.

15 Cortada, James W.: *The Digital Hand. How Computers Changed the Work of American Manufacturing, Transportation, and Retail Industries*, Oxford 2004; Cortada, James W.: *The Digital*

Hand. How Computers Changed the Work of American Financial, Telecommunications, Media, and Entertainment Industries, Oxford 2006.

16 Rochlin, Gene I.: *Trapped in the Net. The Unanticipated Consequences of Computerization*, Princeton 1997.

17 Edwards, Paul N.: »From ›Impact‹ to Social Process. Computers in Society and Culture«, in: Jasanoff, Sheila et al. (Hg.): *Handbook of Science and Technology Studies*, Thousand Oaks u.a. 1994, S. 257–285; Edwards, Paul N.: *The Closed World. Computers and the Politics of Discourse in Cold War America*, Cambridge und London 1996.

18 Ceruzzi, Paul: *A History of Modern Computing*, Cambridge/Mass. 1998.

19 Campbell-Kelly, Martin: *From Airline Reservations to Sonic the Hedgehog. A History of the Software Industry*, Cambridge/Mass. 2003; siehe auch Hashagen, Ulf et al. (Hg.): *History of Computing. Software Issues*, Berlin 2002.

20 Haigh, Thomas: »Inventing Information Systems. The Systems Men and the Computer, 1950–1968«, in: *Business History Review* 75, 2001, S. 15–61; Agar, Jon: *The Government Machine. A Revolutionary History of the Computer*, Cambridge/Mass. 2003; Yates, JoAnne: *Structuring the Information Age. Life Insurance and Technology in the Twentieth Century*, Baltimore 2005.

21 United States National Research Council (Hg.): *Funding a Revolution. Government Support for Computing Research*, Washington D.C. 1999, S. 159–168.

22 Hafner, Katie: »Edgar Codd, Key Theorist of Databases, Dies at 79«, in: *The New York Times*, 23. April 2003.

23 Vgl. zum Beispiel die Präsentation der Geschichte der Web Browser-Technologie und der Rolle von Marc Andreesen in Reid, Robert H.: *Architects of the Web. 1000 Days that Built the Future of Business*, New York 1997.

24 Codd, Edgar F.: »A Relational Model of Data for Large Shared Data Banks«, in: *Communications of the ACM* 13, 1970, S. 377–387.

25 Die Digital Library der ACM führt 799 Artikel auf, die Codd zitieren, vgl. http://portal.acm.org/.

26 Codd, *A Relational Model of Data for Large Shared Data Banks*, a.a.O., S. 377.

27 Ebd., S. 377.

28 Ebd., S. 377.

29 Zu Larry Ellison und zur Geschichte von *Oracle* siehe Symonds, Matthew und Larry Ellison: *Softwar. An Intimate Portrait of Larry Ellison and Oracle*, New York 2003.

30 Everest, Gordon C.: »The Futures of Database Management«, in: *Proceedings of the 1974 ACM SIGFIDET (Now SIGMOD) Workshop on Data Description, Access and Control*, Ann Arbor, Michigan, May 01–03, 1974, S. 445–462.

31 Bachman, Charles W.: »The Programmer as Navigator«, in: *Communications of the ACM* 16, 1973, S. 653–658.

32 Sibley, Edgar H.: »On the Equivalences of Data Based Systems«, in: *Proceedings of the 1974 ACM SIGFIDET (now SIGMOD) Workshop on Data Description, Access and Control. Data Models:*

Data-Structure-Set versus Relational, Ann Arbor, Michigan, May 01–03, 1975, S. 43–76.

33 Codd, Edgar F. und Christopher J. Date: »Interactive Support for Non-Programmers. The Relational and Network Approaches«, in: *Proceedings of the 1974 ACM SIGFIDET (now SIGMOD) Workshop on Data Description, Access and Control. Data Models: Data-Structure-Set versus Relational*, Ann Arbor, Michigan, May 01–03, 1975, S. 11–41.

34 Date, Christopher J. und Edgar. F. Codd: »The Relational and Network Approaches. Comparison of the Application Programming Interfaces«, in: *Proceedings of the 1974 ACM SIGFIDET (now SIGMOD) Workshop on Data Description, Access and Control. Data Models: Data-Structure-Set versus Relational,* Ann Arbor, Michigan, May 01–03, 1975, S. 83–113, hier S. 95.

35 »Data Indepencence« wurde 1977 von Date als »immunity of applications to change in storage structure and access strategy« definiert. Date, Christopher J.: An Introduction to Database Systems, New York 1977, zit. nach Chamberlin, Donald D. et al.: »A History and Evaluation of System R«, in: *Communications of the ACM* 24, 1981, S. 632–646, hier S. 632.

36 Date/Codd, *The Relational and Network Approaches*, a.a.O., hier S. 95.

37 Everest, *The Futures of Database Management*, a.a.O.; Codd/Date, *Interactive Support for Non-Programmers*, a.a.O.; Date/Codd, *The Relational and Network Approaches*, a.a.O.; Sibley, *On the Equivalences of Data Based Systems*, a.a.O.

38 Panel and Audience: »Discussion«, in: *Proceedings of the 1974 ACM SIGFIDET (now SIGMOD) Workshop on Data Description, Access and Control. Data Models: Data-Structure-Set versus Relational,* Ann Arbor, Michigan, May 01–03, 1975, S. 123–144, hier S. 123.

39 Ebd., S. 134.

40 Ebd., S. 128.

41 Ebd., S. 137.

42 Ebd., S. 132. Im Rahmen von CODASYL wurde seit 1959 über Programmiersprachen verhandelt, die auf Rechnern unterschiedlicher Hersteller verwendet werden konnten. 1969 veröffentlichte die Data Base Task Group eine erste Beschreibung einer Sprache für das Netzwerk-Datenbank-Modell. Zur Arbeit von CODASYL zur Zeit der Debatte zwischen Codd und Bachman vgl. CODASYL Data Description Language Committee: *CODASYL Data Description Language. Journal of Development*, Washington 1973. Siehe auch Olle, T. William: *The Codasyl Approach to Data Base Management*, Hoboken NJ 1978 sowie Knowles, J. S. und D. M. R. Bell: »The Codasyl Model«, in: Stocker, Peter M. et al. (Hg.): *Databases – Role and Structure*, Cambridge/UK 1984.

43 Panel and Audience, *Discussion*, a.a.O., S. 132.

44 Haigh, Thomas: »Charles W. Bachman Interview, September 25–26, 2004, Tucson, Arizona. Interview conducted for the Special Interest Group on the Management of Data (SIGMOD) of the Association for Computing Machinery (ACM). Transcript and original tapes donated to the Charles Babbage Institute«, in: *ACM Oral History interviews*, 2006, S. 1–106, hier S. 104.

45 Ebd.

46 United States National Research Council, *Funding a Revolution*, a.a.O., S. 161f.

47 Weeldreyer, James A. und Oris D. Friesen: »Multics Relational Data Store. An Implementa-

tion of a Relational Data Base Manager«, in: *The Eleventh Hawaii International Conference on Systems Sciences*, Honolulu, January 5–6, 1978, S. 52–66.

48 Vgl. Haigh, *Charles W. Bachman Interview*, a.a.O., S. 19, sowie Haigh, *Inventing Information Systems*, a.a.O.

49 Vgl. Date, Christopher J.: »Edgar F. Codd. August 23rd, 1923 - April 18th, 2003«, in: *SIGMOD Record* 32, 2003, S. 4–13.

50 Codd, *A Relational Model of Data for Large Shared Data Banks*, a.a.O.; Codd, Edgar F.: »Relational Completeness of Data Base Sublanguages«, in: Courant Computer Science Symposia (Hg.): *Data Base Systems*, Engelwood Cliffs, N.J. 1971, S. 65–98; Codd, Edgar F.: »A Data Base Sublanguage Founded on the Relational Calculus«, in: *The 1971 ACM SIGFIDET Workshop*, San Diego, California, 1971, S. 35–68.

51 Astrahan, Morton M. und Donald D. Chamberlin: »Implementation of a Structured English Query Language«, in: *Communications of the ACM* 18 1975, S. 580–588, hier S. 580.

52 Ebd., S. 580.

53 Die Darstellung stützt sich auf Chamberlin et al., *A History and Evaluation of System R*, a.a.O., S. 632–646.

54 Astrahan/Chamberlin, *Implementation of a Structured English Query Language*, a.a.O.

55 Chamberlin et al., *A History and Evaluation of System R*, a.a.O., S. 636.

56 Ebd., S. 636. Zur IBM-internen Definition der »user orientation« vgl. den entsprechenden Eintrag im IBM Jargon and General Computing Dictionary: »user orientation *n.* A term for the kind of manual now described as *user friendly*. This was first used in 1968 at a meeting of hardware publications managers, when it was stated that the greatest need was for publications developers to understand who the readers were and why they were readers. It took fifteen years for this truth to be widely appreciated and applied.« Cowlishaw, Mike: *IBM Jargon and General Computing Dictionary*, 10. Aufl., Winchester UK 1990, S. 57.

57 Die (ex post) formulierten Projektziele lauteten u.a. wie folgt: »(1) To provide a high-level, nonnavigational user interface for maximum user productivity and data independence. […] (3) To support a rapidly changing database environment, in which tables, indexes, views, transactions, and other objects could easily be added to and removed from the database without stopping the system. […] (6) To provide a flexible mechanism whereby different views of stored data can be defined and various users can be authorized to query and update these views. (7) To support all of the above functions with a level of performance comparable to existing lower-function database systems.« Chamberlin et al., *A History and Evaluation of System R*, a.a.O., S. 633.

58 Bayer, Rudolf und Edward McCreight: »Organization and Maintenance of Large Ordered Indexes«, in: *Acta Informatica* 1, 1972, S. 173–189; Bayer, Rudolf und Edward McCreight: »Symmetric Binary B-Trees. Data Structure and Maintenance Algorithms«, in: *Acta Informatica* 1, 1972, S. 290–306.

59 Chamberlin et al., *A History and Evaluation of System R*, a.a.O., S. 634.

60 Eine unvollständige Liste der prominentesten Entwickler hat Paul McJones auf http://www.mcjones.org/System_R/people.html zusammengestellt.

61 Paul McJones führt auf http://www.mcjones.org/System_R/bib.html insgesamt 45 Aufsätze auf (4. Dezember 2000).

62 Die bei IBM laufenden Entwicklungsprojekte wiesen markante Unterschiede hinsichtlich Prestige, Ressourcen und Dringlichkeit auf. Mögliche Bezeichnungen waren *Hobby, Adtech* und *Strategic Project*. »System R« dürfte Mitte der 1970er Jahre in die Kategorie *Adtech* gefallen sein. Der *IBM Jargon and General Computing Dictionary* vermerkt unter *adtech n*: »Advanced Technology. Time put aside for a risky project, not necessarily directly related to a product. May mean: a) Play time (when someone else is doing it), or b) Exciting, innovative system design with no product deadlines (when speaker is doing it).« Cowlishaw, *IBM Jargon and General Computing Dictionary*, a.a.O., S. 3.

63 United States National Research Council, *Funding a Revolution*, a.a.O., S. 162–168. INGRES steht für Interactive Graphics and Retrieval System. Stonebraker, Michael et al.: »The Design and Implementation of INGRES«, in: *ACM Transactions on Database Systems (TODS)* 1, 1976, S. 189–222.

64 »In order to provide a useful host-language capability, it was decided that System R should support both PL/I and Cobol application programs as well as a standalone query interface, and that the system should run under either the VM/CMS or MVS/TSO operating system environment.« Chamberlin et al., *A History and Evaluation of System R*, a.a.O., S. 636. Zur Hardware und zur UNIX-Umgebung des INGRES-Projekts siehe Stonebraker, Michael: »Retrospection on a Database System«, in: *ACM Transactions on Database Systems* 5, 1980, S. 225–240.

65 Stonebraker, *Retrospection on a Database System*, a.a.O. , S. 232.

66 Ebd., S. 228.

67 Ebd., S. 228. Die Gründe für die Entscheidung des INGRES-Teams gegen B-Bäume sind in Stonebraker, Michael und Gerald Held: »B-trees Re-examined«, in: *Communications of the ACM* 21, 1978, S. 139–143, diskutiert. Das INGRES-Team hat diese Entscheidung später revidiert und ebenfalls B-Bäume verwendet, vgl. Stonebraker, Michael und Lawrence A. Rowe: »The Commercial INGRES Epilogue«, in: Michael Stonebraker (Hg.): *The INGRES Papers. Anatomy of a Relational Database System*, Reading/Mass. 1986, S. 63–82, S. 76.

68 Dies zeigt auch die unerwartet große Zahl an eingereichten Beiträgen für die *International Conference on Very Large Data Bases*, die im September 1975 in Framingham, Massachusetts stattgefunden hat. Vgl. Kerr, Douglas S. (Hg.): *Proceedings of the International Conference on Very Large Data Bases*, September 22–24, 1975.

69 Nicht zuletzt deshalb musste Codd ständig neue Versionen seines »ursprünglichen« Konzepts veröffentlichen. Vgl. Gerald Held im Vorwort zum Reader Stonebraker, Michael: *The INGRES Papers. Anatomy of a Relational Database System*, Reading/Mass. 1986. Zudem scheint Codd wenig in die Entwicklungsarbeit von »System R« involviert gewesen zu sein. Die ständigen Referenzen der IBM-Entwickler auf Codd als Gründervater der Community haben auch mit dessen Marginalisierung in der konkreten Entwicklungsarbeit zu tun.

70 Stonebraker, *Retrospection on a Database System*, a.a.O., S. 230.

71 Stonebraker et al., *The Design and Implementation of INGRES*, a.a.O., S. 191.

72 Astrahan/Chamberlin, *Implementation of a Structured English Query Language*, a.a.O., S. 582.

73 Keller, Arnold E.: »The Man behind Systems at Shell Oil«, in: *Business Automation* 7, 1962, S. 20–24, zit. nach Haigh, Thomas: »›A Veritable Bucket of Facts‹. Origins of the Data Base Management System«, in: *SIGMOD Record* 35, 2006, S. 33–49, hier S. 34.

74 Eisenpreis, Alfred: »Beispiele der Zusammenarbeit von Theorie und Praxis in einem Handelsunternehmen«, in: Applebaum, William (Hg.): *Wissenschaft und Handel. Der Brückenschlag zwischen Theorie und Praxis,* 4.-7. Juli 1966 in Rüschlikon-Zürich, Bern 1967, S. 131–141, hier S. 133.

75 Barthes, Roland: *S/Z*, Frankfurt a.M. 1987 (1970), S. 9f.

76 »Ein Erzähler darf das eigene Werk nicht interpretieren, andernfalls hätte er keinen Roman geschrieben, denn ein Roman ist eine Maschine zur Erzeugung von Interpretationen.« Eco, Umberto: *Nachschrift zum ›Namen der Rose‹*, München 1986 (1983), S. 9–10.

77 Barthes, *S/Z*, a.a.O., S. 9f.

78 Eco, Umberto: *Das offene Kunstwerk*, Frankfurt a.M. 2002 (1962); Sontag, Susan: »Against Interpretation«, in: Dies. (Hg.): *Against Interpretation and Other Essays*, New York 1996 (1964), S. 3–14; Gadamer, Hans-Georg: *Wahrheit und Methode. Grundzüge einer philosophischen Hermeneutik*, Tübingen 1975 (1960).

79 Foucault, Michel: »Was ist ein Autor?«, in: Ders. (Hg.): *Schriften zur Literatur*, Frankfurt a.M. 1988 (1969), S. 7–31.

80 Barthes, Roland: »Der Tod des Autors«, in: Fotis, Jannidis (Hg.): *Texte zur Theorie der Autorschaft*, Stuttgart 2000 (1968), S. 185–193.

81 Iser, Wolfgang: *Die Appellstruktur der Texte. Unbestimmtheit als Wirkungsbedingung literarischer Prosa*, Konstanz 1970. Vgl. auch Weinrich, Harald: »Für eine Literaturgeschichte des Lesers«, in: Ders. (Hg.): *Literatur für Leser,* Stuttgart 1971, S. 23-34.

82 Sattler, Dieter E. (Hg.): *Friedrich Hölderlin. Sämtliche Werke, »Frankfurter Ausgabe«.* Historisch-kritische Ausgabe, Frankfurt a.M. 1975, S. 16–20.

83 http://www.hoelderlin.de/materialien/html/marginalien.html.

84 Eco, *Das offene Kunstwerk*, a.a.O., S. 90.

85 »This is one of his favorite quotes and he uses it frequently (episode 703, ›Toe Tags‹).« http://en.wikipedia.org/wiki/Gil_Grissom.