Deutschlandradio
Deutschlandfunk Kultur | Deutschlandfunk

Alles nur Routinen!

Geschichte(n) der Programmiersprachen in einer Langen Nacht

Autor: **Florian Felix Weyh**

Regie: **Philippe Brühl**

Redaktion: **Dr. Monika Künzel**

SprecherInnen

Erzählerin: **Rebecca Madita Hundt**
Zitatsprecher: **Daniel Berger**
Technische Stimme: **Volker Niederfahrenhorst**
Sprecherin Ellen Ullman: **Sigrid Burkholder**
Sprecherin Grace Hopper: **Susanne Flury**
Sprecherin Ada Lovelace: **Susanne Reuter**
Sprecherin Zitate weiblich: **Edda Fischer**

Sendetermine: **7. September 2019 Deutschlandfunk Kultur**
7./8. September 2019 Deutschlandfunk

1. Stunde

„Die Kunst, mit zwei Fingern zu rechnen“

Gesprächspartner dieser Stunde
Dr. Martin Burckhardt, Philosoph, Programmierer (Berlin)
Peter Fuß, Deutsches Museum (München)
Christoph Kappes, Programmierer, Jurist (München)
Prof. Dr. Jochen Ziegenbalg, Informatiker (Karlsruhe/Berlin)
Prof. Dr. Horst Zuse, Informatiker (Berlin)

Musik 3’44 „Model” Interpret: Balanescu Quartet
Komponist/Texter: Ralf Hütter, Karl Bartos

O-Töne und Technische Stimme darüber:

S1-01 Fuß 0‘41
Ich kann mich erinnern, als ich mit meinem Vater 1962 in einem Großmarkt beim Einkaufen war, wurde das dort so gehandled, dass wenn er einen Artikel aus dem Regal nahm, auf seinen Einkaufswagen, musste er die dazu gehörige Lochkarte mitführen, und dieser Stapel an Karten wurde dann vorm Bezahlen, vor der Kasse von einem Kontrolleur hier noch mal kontrolliert, ob die Karten auch mit dem gewählten Artikeln auf dem Einkaufswagen übereinstimmen und dann ging’s erst mittels einem kleinen Förderband zur Abrechnung. Und war das geschehen, konnte mein Vater seine Rechnung begleichen.

TECHNISCHE STIMME 01
Das folgende Programm wurde geschrieben, um nichtandroide Wesen zu nächtlicher Stunde wach zu halten. Es läuft 180 Minuten lang.

S1-02 Fuß 0‘23
Also Sie sehen: Die Bereiche sind sehr vielfältig! Buchhaltung oder auch Zeiterfassung, bestimmt alle Bereiche, die in so Firmen eben relevant waren, und ohne die ging’s nicht! Datenverarbeitung der Jahre 1950, 60, bis auch noch darüber hinaus, 65, weil man hatte eigentlich nichts anderes.

S1-03 Burckhardt 0‘26
Das Bizarre an der Geschichte des Computers ist, dass sie überhaupt nicht als solche aufgefasst wird! Das heißt, wenn man zurückgeht und solche Momente sich anschaut, die irgendwo in der Geschichte so eine Rolle spielen, zum Beispiel den Übergang von einem erekten Zeichen zum Loch – also eine vollkommene Veränderung des Kommunikationsfeldes – das ist etwas, was quasi explodiert, zu einer bestimmten Zeit. Später wird es gewissermaßen in der „Black Box“ der Maschine einfach genommen, als zweite Natur.

TECHNISCHE STIMME 02
Korrektur: Das Programm enthält ein autonomes Objektmodul „Nachrichten“ und läuft inklusive des Objektmoduls 180 Minuten, exklusive einige Minuten kürzer.

S1-04 Burckhardt 0‘13
Also wenn ich zum Beispiel … ich mach so ein Managerseminar, wo ich Leuten auch wirklich dann, die aus der Industrie kommen, so versuche, diese Gedankenfiguren klar zu machen – für die beginnt der Computer irgendwann in den 80er-Jahren!

TECHNISCHE STIMME 03
Das Objektmodul „Nachrichten“ enthält zeitlich kritische Variablen und ist deswegen nicht exakt kalkulierbar.

S1-05 Ziegenbalg 1‘02
Der Chef von dem Rechenzentrum, mit dem ich’s zu tun hatte, der hatte damals zwei Werkstudenten zu beschäftigen. Einer davon war ich, ein anderer war ein Kommilitone von mir. Und für mich hatte er irgendein Problem, ich weiß gar nicht mehr genau, was es war, und für den Kommilitonen hatte er aber nicht so richtig was Griffiges als Problem! Die Programme waren ja damals noch auf Lochkarten archiviert, und in Lochkarten gestanzt. Ein Stapel von Lochkarten war das Quellenprogramm, und wenn das übersetzt wurde, bekam man einen neuen Stapel von Lochkarten, das war das Maschinenprogramm. Die Programme waren in solchen Schränken dann drin, und der Chef, der uns beide Werkstudenten zu betreuen hatte, der ging da mal an einen Schrank und zog so’n bisschen rum und schaute sich [an], welche Programme da drin sind, und dann sagte er: „Aha! Wir haben hier ein Programm drin, das liegt da schon ewig, und wir wissen gar nicht mehr so richtig, was es eigentlich macht! Schauen Sie sich doch mal das Programm an. Und versuchen Sie zu analysieren, was es tut! Und wenn Sie schon dabei sind, versuchen Sie vielleicht, das Programm ein bisschen effizienter zu machen!“

TECHNISCHE STIMME 04
Warnhinweis: Sie können das Programm „Lange Nacht“ abbrechen, werden dann aber auf einem unbefriedigenden Wissenslevel stagnieren.

S1-06 Ziegenbalg 0‘22
Na gut, dann hat er das gemacht. Und am Ende der Werkstudentenzeit, als nach wasweißich drei Wochen, meldete sich dann der Kommilitone und sagte: Also er hat das Programm analysiert, und er hat’s effektiver gestaltet, das Programm ist jetzt sehr viel kürzer, und es läuft auch sehr, sehr viel schneller – aber was es eigentlich macht, weiß er immer noch nicht! (lacht)

ERZÄHLERIN 01
Programme – so sie nicht Radioprogramme sind – transportieren Botschaften aus einer fremden Welt. Die meisten Menschen verstehen sie nicht, und die, die das tun, sind Spezialisten. Wir nennen sie Programmierer, Softwareentwickler, Informatiker.

TECHNISCHE STIMME 05
Korrektur: Programmiererinnen, Softwareentwicklerinnen, Informatikerinnen.

ERZÄHLERIN 02
Exakt, denn die Geschichte des Programmierens wäre ohne ihren weiblichen Anteil nur halb erzählt. Die zweite Stunde in dieser Langen Nacht wird darauf ihr Augenmerk richten.

TECHNISCHE STIMME 06
„Männer löten, Frauen denken.“

ERZÄHLERIN 03
Zunächst geht es um die Grundlagen. Für alle.

TECHNISCHE STIMME 07
„Die Kunst, mit zwei Fingern zu rechnen“

ERZÄHLERIN 04
… heißt die erste Stunde der Langen Nacht mit Geschichte und Geschichten der Programmiersprachen.

S1-07 Atmo Christoph Kappes zählt mit den Fingern bis 15 und verbiegt sie dazu unter merklicher Anstrengung. Sprecher und technische Stimme darüber

TECHNISCHE STIMME 08
Christoph Kappes, Jurist, Gründer mehrerer IT-Firmen, Programmierer.

ERZÄHLERIN 05
Er verbiegt hörbar die Gelenke, um digital mit seinen Fingern – lateinisch digiti – zu zählen.

S1-08 Kappes 0’29
Als Student – oder war ich sogar noch Schüler in den späten Jahren? – hab ich bei der Volkshochschule, hatte ich PC-Kurse, wo ich dann in 52 Doppelstunden PC-Grundschulung mit den Leuten gemacht hab. Wir haben dann in der Gruppe Computer gespielt, acht Leute waren Speicher, einer war Prozessor, zwei Leute waren der Datenbus. Also die waren gut beschäftigt, die Daten hin- und herzutragen! Eine Multiplikation dauerte fünf Minuten, das war sehr unterhaltsam! Hat mir großen Spaßt gemacht! Und die Leute haben da tatsächlich dann auch begriffen, wie so ein Ding grundlegend funktioniert.

ERZÄHLERIN 06
Genau darum soll es gehen. In dieser Stunde der Langen Nacht helfen uns dabei neben Christoph Kappes auch der emeritierte Informatikprofessor Jochen Ziegenbalg, der eingangs erzählte, dass er schon als Student mit der Unverständlichkeit eines Codes konfrontiert wurde – damals noch in Lochkarten gestanzt. Als Autor des Standardwerks „Algorithmen von Hammurapi bis Gödel“ sieht er algorithmische Muster, wohin er blickt.

S1-09 Ziegenbalg 0‘28
Ich denke ja, dass fast jede menschliche Tätigkeit – oder sehr, sehr viele menschliche Tätigkeiten – algorithmischer Natur sind. Wir denken, wenn wir ein bestimmtes Ziel erreichen wollen, denken wir in Teilschritten: Welche Teilschritte sollte man ansteuern, um dieses globale Ziel zu erreichen? Und schon dieses Zerlegen eines komplexen Problems in Teilprobleme, in Teilschritte, ist eine ganz wichtige algorithmische Tätigkeit.

TECHNISCHE STIMME 09
Radioproblem: Zuweisung von Namen zu Stimmen, ohne den Erzählfluss zu unterbrechen. Lösungsalgorithmus in zwei Teilschritten: a) Stimme einspielen ...

S1-10 Burckhardt 0‘05
Die Programmierung ist letztlich eine Abstraktion. Sie hat einige Implikationen. Zum Beispiel: Lichtgeschwindigkeit.

ERZÄHLERIN 07
… und b) Namen nennen!

TECHNISCHE STIMME 10
Martin Burckhardt. Philosoph und Programmierer. Oder eher: programmierender Philosoph.

S1-11 Burckhardt 0‘21
Die Elektrizität, das ist eine Implikation. Darüber denken Programmierer nie nach! Oder zum Beispiel, dass es eine „creatio ex nihilo“ [ist]. Dass ich zum Beispiel sagen kann: Ich definiere eine Variable, dann ist sie da. Gesagt – getan! Ich war wirklich schon älter, als programmiert habe, saß vor diesem „Gesagt – getan“, das hat mich irritiert! Wo gibt’s das sonst? Das ist Zauber … Magie!

ERZÄHLERIN 08
Und dieser Magie nähern wir uns zunächst mit einem Besuch in der informationstechnischen Sammlung des Deutschen Museums, München. Peter Fuß erläutert sein Konzept:

S1-12 Fuß 0‘54
Meine Führung geht grundsätzlich einem Thema nach. Dieses Thema lautet „Vom Abakus bis zum Supercomputer“. Also ein recht exponiertes Unternehmen! Ich beginne grundsätzlich mit einer Definition „analog/digital“ – denn das ist auch so manchem nicht ganz geläufig –, beginne dann mit dem Abakus, dem ältesten Rechenhilfsgerät der Welt und schaukle mich dann allmählich nach oben. Ja, Rechenstäbchen von Napier, Rechenmaschinen, Schickard, Leibnitz, Lochkarten, Lochkartensysteme, Hollerith, bis dann weitergeht mit Zuses, Zuses erstem Computer. Und dann geht’s in den Anlagen, die ständig größer, volumimöser werden, rauf bis zum Supercomputer, wobei der Supercomputer des Jahres 1976 gemeint ist.

Musik 2’30 „Popcorn” von Hot Butter
Komponist/Texter: Gershon Gary Kingsley

ERZÄHLERIN 09
Strenggenommen ist die Welt ein einfaches Gebilde, lässt man mal ein paar seelische Irritationen weg. Sie besteht aus Entscheidungen: Ich tu’s – ich tu’s nicht. Ich will’s – ich will es nicht. Entweder – oder, schwarz – weiß, ja – nein. Warum bloß gibt es in dieser Welt – zumindest der deutschsprachigen – 26 Buchstaben plus drei Umlaute? Und warum ist der Zahlenraum unendlich? Das macht doch alles nur kompliziert. Besser wäre ein bescheidenes System aus nur zwei Zahlen!

S1-13 Fuß 1‘01
Meine Frage an den Besucher lautet grundsätzlich: „Was sagt Ihnen der Name ‚Binärcode‘?“ Ja, die Antwort, schlichtwegs natürlich durchgängig: „Besteht aus 0 und 1.“ Dann kommt die große Frage: „Ja aber wer hat denn eigentlich den erfunden?“ Ja, da scheiden sich natürlich die Geister! Umso größer ist das Erstaunen, hier Herrn Leibniz ins Spiel zu bringen! Er war ja ein Philosoph – unter anderem –, und er war sehr gläubig. Und er stellte ne These auf, so nach dem Motto, grob überrissen „Gott ist alles“ – sprich 1. Alles andere ist 0, Menschenwerk, sprich 0. Er hat das Ganze aber schon noch fein säuberlich dokumentiert in einem Schriftstück namens „Explication de l’Arithmétique Binaire“. (zeigt es vor) Hier ist also kleinklein aufgegliedert, wie sich jede Ziffer, in Nullen und Einsen zerlegen lässt. Sogar die Grundrechenarten hat er hier aufgeführt.

ZITATSPRECHER 01
„Einer der Haupt-Puncten des christlichen Glaubens, und zwar unter denjenigen, die den Weltweisen am wenigsten eingangen und noch den Heyden nicht wohl beyzubringen, ist die Erschaffung aller Dinge aus nichts durch die Allmacht Gottes. Nun kann man wohl sagen, daß nichts in der Welt sie beßer vorstelle, ja gleichsam demonstrire als der Ursprung der Zahlen wie er alhier vorgestellet, durch deren Ausdrückung bloß und allein mit Eins und Null oder Nichts.“[1]

ERZÄHLERIN 10
… schreibt Gottfried Wilhelm Leibniz bereits am 12. Januar 1697 – Jahre vor seiner Schrift „Explication de l’Arithmétique Binaire“ – in einem Brief an Herzog Rudolf August von Braunschweig-Wolfenbüttel. In diesem allerersten Zeugnis zum Leibniz’schen Dualsystem möchte der Universalgelehrte seinen Gönner dazu bewegen, einen – wie er es nennt – „Denckpfennig“ prägen zu lassen, auf dem der soeben von Leibniz entdeckte binäre Code bildhaft und allgemeinverständlich dargestellt wird.

TECHNISCHE STIMME 11
„Denck-Pfennig“ – Geld als Bildungsmedium. Und zugleich als Möglichkeit, es in der Staatskasse klingeln zu lassen.

ERZÄHLERIN 11
Leibniz legt sogar Skizzen bei, wie er sich diese Münze vorstellt, so wichtig ist ihm das Projekt:

[1] Gottfried Wilhelm Leibniz „Allgemeiner politischer und historischer Briefwechsel“, 13. Band August 1696 – April 1967, Akademie Verlag 2010, S. 117

ZITATSPRECHER 02
„Denn ich sehe, daß sich aus dieser Schreibart der Zahlen wunderliche Vortheil ergeben, die hernach auch in der gemeinen Rechnung zustatten kommen werden.“2

S1-14 Fuß 0‘10
[So] ist also dieser Binärcode nichts, was irgendein Computerbauer der letzten 80 Jahre erfunden hätte, sondern besteht bereits über 300 Jahre.

S1-15 Burckhardt 0‘24
Natürlich Leibniz hat begriffen, dass die beiden Zahlen 0 und 1 fundamental sind! Schrödinger nennt es „die Königszahlen“ der Mathematik. Leibniz hat auch Rechenmaschinen gebaut. Aber ehrlich gesagt, das Dilemma, was Leibniz hatte, und was auch Babbage noch nicht aufgelöst hat, war, dass er letztlich dann doch im Zahlenraum immer noch verblieben ist. Und der hat eigentlich mental diese Umbesetzung in das „Loch“ und „Nichtloch“ nicht vollzogen.

TECHNISCHE STIMME 12
Kommentar außerhalb der Programmzeilen: Charles Babbage wird als männlicher Sparringspartner von Lady Ada Lovelace in Stunde zwei der Langen Nacht thematisiert.

ERZÄHLERIN 12
Was Martin Burckhardt bemängelt, ist die Tatsache, dass Gottfried Wilhelm Leibniz zwar als einer der ersten über den Binärcode nachdachte, ihn aber nicht als etwas gänzlich Neues begriff. Sein Fokus lag auf der Mathematik des herkömmlichen Dezimalsystems. Schon 25 Jahre zuvor hatte er eine analoge Rechenmaschine aus Zahnrädern entworfen, die alle Grundrechenarten beherrschte. Peter Fuß präsentiert im Deutschen Museum einen Nachbau davon:

S1-16 Fuß 0‘35
Ja, hinten haben’s das Konterfei von Herrn Leibniz. Hier an dieser Rechenmaschine, schönes Messinggehäuse, sehen Sie oben acht Federn rausragen. Und überall, wo Sie hier so ne Feder sehen, ist darunter so ne Staffelwalze verbaut. Dann braucht’s eigentlich gar nicht mehr viel Phantasie, um hier eben die verschiedenen Faktoren einzustellen. Und durch Drehen können Sie sozusagen Ihre Summen erzeugen. Ja, und die ersten dieser Maschinen kamen in den Einsatz in Paris bei dortigen Finanzbehörden.

[2] Leibniz a.a.O. S. 121

ERZÄHLERIN 13
… wo sich Rechenfehler und Schlampereien schmerzhaft bemerkbar machen. Eindeutig stand bei mechanischen Rechenmaschinen, die Routineaufgaben im Dezimalsystem erledigten, der Wunsch nach Effizienz und Fehlervermeidung im Vordergrund. Es fehlte jedoch die zum heutigen Computer hinleitende Idee, dass die Maschine programmierbar zu sein habe, man also nicht bloß einige – durch Zahnräder physisch vorbestimmte – Rechenfunktionen ausführen können müsse, sondern schlicht alle mathematischen Operationen. Dazu bedurfte es Anregungen aus einer ganz anderen Richtung, nämlich der Unterhaltungsindustrie … modern ausgedrückt.

Musik 3'10 „Spieluhr-Polka" von Rudolf Buchbinder
Komponist/Texter: Johann Strauß (Sohn), Otto Schulhof

Erzählerin und O-Töne nach einigen Takten darüber:

ERZÄHLERIN 14
Seit der Renaissance faszinierten mechanische Musikautomaten den Menschen.

S1-17 Burckhardt 0'10
Die haben also eine Walze, so eine Musikwalze, und die dreht sich, und dann gibt's einen kleinen erekten Sporn, oder auch viele. Die triggern dann bestimme Metallstücke und erzeugen Töne.

ERZÄHLERIN 15
In ihrem Gefolge kam es zu einem feinmechanischen Boom von Automaten, der immer mehr Kuriositäten hervorbrachte: scheinbar lebende Tiere wie eine Ente, die Eier zu legen vermochte. Oder Puppen, die komplexe Bewegungen ausführen konnten, um Musik zu erzeugen, Worte zu schreiben oder Bilder zu malen. Eine Generation nach Leibniz vermochte ein als „Der Zeichner" bekannter Automat von Henri-Louis Jaquet-Droz bereits vier unterschiedliche Zeichnungen auf ein Stück Papier zu tuschen, das man ihm hinlegte. Seine Mechanik wurde durch austauschbare Nocken gesteuert. Je nachdem, wo diese saßen, bewegte sich der Arm in die eine oder die andere Richtung. Das reichte für komplexe Bilder wie menschliche Gesichter oder einen kleinen Hund.

S1-18 Burckhardt 0'02
Also das ist eine Form der mittelalterlichen Programmierung.

ERZÄHLERIN 16
… sagt Martin Burckhardt über diese Steuerungskunst, die schon Leonardo da Vinci im 15. Jahrhundert bei einem von Sprungfedern angetriebenen Wagen skizziert hatte. Abstrakt betrachtet sind Nocken oder kleine Metallsporne Bits – also Signale. Doch das hatten die Rechenautomatenkonstrukteure in der Nachfolge von Leibnitz erst zu begreifen. Die binäre Theorie der Mathematik musste mit den intuitiv binären Praktiken aus der Uhrmacherzunft zusammentreffen und dazu noch die Idee des austauschbaren Programms entwickeln. Auch das gab es schon, denn Musikpublikum verlangte nach Abwechslung: Bei walzengesteuerten Musikautomaten konnte man bisweilen die Walzen wechseln, sprich unterschiedliche Musikprogramme abfahren. Die wiederum waren aber fest an einen Metallzylinder gebunden. Eine aufwändige Sache.

S1-19 Burckhardt 0‘11
Was wäre, wenn ich quasi diese obere Haut einer solchen Spielwalze abnehmen würde? Und dann hätte ich ja nicht mehr nur ein Gerät, sondern bist wie ein DJ: Ich habe viele Platten!

ERZÄHLERIN 17
Diesen geistigen Sprung vollzog überraschenderweise kein Musikautomatenbauer sondern der Sohn eines französischen Webers, Joseph-Marie Jacquard. Schon als Kind musste er in der väterlichen Werkstatt schuften, was ihm derart missfiel, dass er sich in eine Buchbinderlehre flüchtete. Doch das half nichts, er erbte dennoch die Weberei des Vaters.

S1-20 Burckhardt 0‘22
Jacquard, der ein großer … wie soll ich sagen? … leidenschaftlicher Liebhaber der Arbeitsvermeidung war (lacht), der Haus und Hof verkaufte, bis er gar nichts mehr besaß – und selbst sein Bett hat der verkauft! –, der war wirklich genötigt darüber nachzudenken, wie er mit einem Minimum an Arbeit ein Maximum an Wirkung erzeugen könne. Also ist die Idee: Ich nehme diesen Sporn und ersetze ihn durch ein Loch.

TECHNISCHE STIMME 13
Sprungverweis: goto Martin Burckhardt „Eine kurze Geschichte der Digitalisierung“. read Seite 37.

ZITATSPRECHER 03
„Schlagen bei einer Spieluhr die Sporne auf der Walze kleine Metallplättchen an, müssen bei einem Webstuhl Fäden gehoben werden. Und daraus ergibt sich dann ein charakteristisches Muster. Um beliebig viele Muster erzeugen zu können, hatte

Jacquard nun die Idee, dass man, statt eine ‚fest verdrahtete' Walze zu benutzen, einen Mechanismus entwickeln konnte, der ein Papier daraufhin abtastete, ob an einer bestimmten Stelle ein Loch war oder nicht. Fand sich ein Loch, wurde der Faden gehoben, ansonsten nicht."3

ERZÄHLERIN 18
Und so wurde aus dem Webstuhl eine Tucherzeugungsmaschine mit vorprogrammierbaren Mustern, abertausenden verschiedenen an ein und derselben Anlage! Das brachte zuvor nur der handwerklich arbeitende Mensch selbst zustande, nicht aber ein Automat. Und es bedurfte dazu bloß eines gelochten Stücks fester, strapazierfähiger Pappe: des Lochstreifens oder verkleinert der Lochkarte. So heißt dieser wichtigste Datenträger bis zum Aufkommen von Magnetspeicherbändern allerdings erst seit Hermann Hollerith. Der deutschstämmige amerikanische Erfinder hatte sein Schlüsselerlebnis während einer Eisenbahnfahrt, wie Peter Fuß erläutert:

S1-21 Fuß 0'41
Er sah nämlich, dass der Schaffner das Ticket eines Fahrgastes an verschiedenen Stellen lochte. Er wollte nun wissen, warum er das macht, und bekam dann zur Antwort: „Die geben ansonsten ihr Ticket immer unerlaubterweise weiter!" Um hier gegenzusteuern, wenn ich hier ein Loch reinmache, dann weiß ich, war der Fahrgast männlich/weiblich, ja? Wenn ich hier ein Loch setze, konnte ich wasweißich die Hautfarbe hier hinterlegen, und dort ein Loch, dort ein Loch … also er hat nichts anderes gemacht als Attribute! Das brachte angeblich Herrn Hollerith auf die Idee, so eine Pappkarte auch als Massendatenspeicherungsmittel zu verwenden.

S1-22 Burckhardt 0'10
Seine ersten Lochkarten, die er dann gebaut hat, hatten genau die Maße eines Dollars. Er wusste genau: „Okay, die Cashcow wird mir helfen, richtig richtig reich zu werden!"

ZITATSPRECHER 04
„Die wesentliche Innovation, die seine Apparatur über die ‚händische Lochkartenfotografie' der Schaffner erhob, war, dass er die Information der Lochkarten in elektrische Impulse übersetzte, die wiederum mechanische Zähler aktivierten."

[3] Martin Burckhardt [1] „Eine kurze Geschichte der Digitalisierung", Penguin Verlag 2018, S. 37f.

TECHNISCHE STIMME 14
Registerwechel: Martin Burckhardt „Philosophie der Maschine“. Seite 246. Und weiter darin:

ZITATSPRECHER 05
„Auf diese Weise war der Leseprozess elektrifiziert, waren die 43 Maschinen, die er dem Zensus im Jahr 1890 zur Verfügung stellte, in der Lage, die 62.947.714 Menschen, aus denen die amerikanische Gesamtpopulation bestand, zu einem Bruchteil der Kosten und der Zeit zu erfassen. Weil jeder der aufstrebenden Nationalstaaten an einer Volkszählung interessiert war, gründete Hollerith 1896, kaum 36 Jahre alt, die Tabulating Machine Company. Das Geschäftsmodell der Firma war nicht minder ingeniös als die materielle Apparatur. Denn die Hollerithschen Maschinen waren nicht zu erwerben, sondern wurden lediglich verliehen; zudem benötigten sie als Rohstoff die Hollerithsche Lochkarte.“[4]

Musik 1’12 „Service III” von Orchester Roland Kovac
Komponist/Texter: Roland Kovac 00979 / MPS

ERZÄHLERIN 19
Nun haben wir beides, die Theorie in Gestalt des binären Zahlensystems – Nichts und Gott, Null und Eins, Loch und Sporn, später dann Schalter aus und Schalter an –, und den physischen Datenträger in Gestalt der Lochkarte und des Lochstreifens. Auch die Elektrizität ist schon da und könnte jetzt, zu Beginn des 20. Jahrhunderts, das Computerzeitalter einläuten. Elektrizität ist Lichtgeschwindigkeit, wie Martin Burckhardt schon angemerkt hat. Das macht den Hauptvorteil des elektronischen Rechnens aus. Doch im Grundsatz braucht ein Binärrechner keinen Strom.

ZITATSPRECHER 06
„Mit ein paar Freunden konstruierte ich einmal einen Rechner aus Stöcken und Schnüren. Die Teile stammten aus einem Spielzeug-Konstruktionsbaukasten namens Tinker Toys. Der eine oder andere erinnert sich vielleicht noch an die Holzstäbe, die in die Löcher kleinerer hölzerner Naben passten.“[5]

TECHNISCHE STIMME 15
Source: Daniel Hillis „Computerlogik“. Seite 31.

[4] Martin Burckhardt [2] „Philosophie der Maschine“, Matthes&Seitz 2018, S. 246f.
[5] Daniel Hillis „Computerlogik“, C. Bertelsmann 2001, S. 31

ZITATSPRECHER 07
„Den Computer zu bauen, war nicht einfach: Wir brauchten dazu mehrere zehntausend Teile aus über hundert Tinker-Toys-‚Riesenbaukästen', und das fertige Produkt, das heute in Boston im Computermuseum steht, sieht unglaublich kompliziert aus. Aber das Prinzip, nach dem es funktioniert, ist die einfache Kombination aus Und- und Oder-Funktionen."6

ERZÄHLERIN 20
Das war in den frühen 1980er-Jahren, und der 1956 geborene Daniel Hillis – unter anderem Mitentwickler der kindgerechten Programmiersprache LOGO –, gehört zu einer Generation von amerikanischen Computerpionieren, denen keine Anstrengung zu groß war, um ins binäre Universum vorzudringen. Bei Hillis begann das schon in jungen Jahren:

ZITATSPRECHER 08
„Mein Vater war Epidemiologe, und wir lebten damals in Kalkutta. Bücher in englischer Sprache waren schwer zu beschaffen, aber in der Bibliothek des britischen Konsulats fand ich ein verstaubtes Exemplar eines Buches, das der Logiker George Boole im 19. Jahrhundert geschrieben hatte. Was mich anzog, war der Titel: An Investigation of the Laws of Thought (Eine Untersuchung der Gesetze des Denkens). Das fesselte meine Phantasie. Gab es tatsächlich Gesetze, die das Denken steuerten?"7

ERZÄHLERIN 21
Gleichsam wie ein Lochstreifen? Dazu Martin Burckhardt.

S1-23 Burckhardt 1'18
Boole ist in mancherlei Hinsicht wirklich ein Ausnahmefall! Also ein Wunderkind aus ärmsten Verhältnissen, Schuster, bringt sich als Kind schon Latein und Griechisch bei. Und auch neue Sprachen, lernt Französisch, Italienisch, Deutsch. Übersetzt Ovid als Siebenjähriger! (lacht) Und der Vater ist so stolz und zeigt also herum, was sein Sohn da übersetzt hat. Und die Lehrer in der näheren Umgebung sind irritiert und bezweifeln, ob ein Kind mit sieben Jahren zu einer solchen Gefühlstiefe wie Ovid fähig ist! Und George Boole ist genötigt, weil sein Vater stirbt, schon mit 14, 15 die Familie zu ernähren, wird Lehrer. Er gründet seine eigene Schule mit 15 Jahren, aber hat immer so eine religiöse Ausrichtung. Hat sich überlegt auch Priester zu werden, zeitweilig. Und fragte sich eigentlich die Frage nach der Universalsprache. Ein bisschen wie … da sind wir wieder bei Leibniz! Das sind fast Leibnizsche Fragen! Eines Tages geht er über eine Wiese und fragt ne ganz simple Frage, sagt sich: „Wenn

[6] a.a.O.
[7] Hillis S. 14

ich jetzt zum Beispiel hier so drei Äpfel und drei Birnen habe, dann habe ich offenbar ein gemeinsames Konzept, die Drei, mit dem ich quasi etwas beschreiben könnte.“ Es müsste doch im menschlichen Gehirn ganz viele solcher Konzepte geben, die sozusagen die Vielfalt der Welt auf einen Begriff oder mehrere Begriffe bringen lassen. Er nennt diese Potenz das Unbewusste.

ERZÄHLERIN 22
Was ihn aber zu ganz andere Gedanken führt als später Sigmund Freud mit seinem Konzept des Unterbewussten, nämlich zu einem formal-abstrakten System des menschlichen Denkens.

ZITATSPRECHER 09
„Boole erfand eine Sprache, mit der man logische Aussagen beschreiben und handhaben konnte und mit der sich auch feststellen ließ, ob sie wahr sind oder nicht. Diese Sprache bezeichnet man heute als Boole’sche Algebra.“8

S1-24 Burckhardt 0‘31
Der ist zwar veröffentlich worden, als Mathematiker, seine „Laws of Thought“, und vorher hat er 1848 … das Buch hab ich gar nicht im Kopf. Die Sachen sind alle veröffentlicht worden als Beiträge zu einer neuen Logik. Aber Mathematiker haben das bis auf Charles Saunders Peirce überhaupt nicht aufgenommen. Der lag gewissermaßen wie ein Wackerstein in den mathematischen Bibliotheken, Fremdkörper ohne Ende! Bis Frege ihn entdeckt hat, und Frege hat ihn als Philosoph entdeckt, nicht als Mathematiker. Frege hat sofort begriffen: Damit kann man Wahrheitstabellen anlegen!

ERZÄHLERIN
Erklärt Martin Burckhardt. Und Peter Fuss vom Deutschen Museum in München ergänzt:

S1-25 Fuß 0‘17
Die Eins ist gar keine Eins! Die Eins können Sie sagen „geht“, bei Null „geht nicht“. Ob Sie sagen „geht“ – geht nicht“, „wahr – falsch“, „Strom an – Strom aus“ … bei Eins ist der Strom an. Dann ist er bei Null natürlich aus.

[8] a.a.O.

ERZÄHLERIN 23
George Boole, dieser wissenschaftliche Außenseiter des 19. Jahrhunderts, lieferte die entscheidende Grundlage für die Computerrevolution des zwanzigsten. Die Mathematik des elektronischen Rechners ist keine Mathematik im Sinne der alten Rechenkunst à la „1+1= 2“, sondern eine formallogische Sprache. Sie erlaubt es, logisch formulierbare Probleme auf Entscheidungsheuristiken von zunächst drei Verknüpfungen herunterzubrechen: und, oder, nicht. Je nach eingesetzter Funktion ist erstens etwas wahr, wenn zwei Bedingungen zugleich erfüllt sind („und“); zweitens, wenn nur eine der beiden stimmt („oder“), drittens gar keine („nicht“). Später kam noch die Funktion „nicht/und“ (englische Abkürzung „nand“) sowie „nicht/oder“ (englisch „nor“) hinzu. Mittels dieser theoretischen Konstrukte ließen sich so genannte Logikgatter erzeugen, mit denen einer neuen Weltauffassung und Welterfassung nichts mehr im Wege stand. Und wie Daniel Hillis beschrieb, lassen sich solche Logikgatter auch ohne Strom konstruieren. Entscheidend ist nur, dass man es schafft, viele An/aus-Schalter hintereinander zu montieren. Denkbar wäre wie im Tinker-Toys-Beispiel eine Seilzugmechanik. Oder man nimmt den „Strom“ wörtlich und baut einen Wassercomputer. Daniel Hillis:

ZITATSPRECHER 11
„An Stelle jedes elektronischen Transistors verwendet man ein hydraulisches Ventil. Das Rohr, das der Stromversorgung des Chips entspricht, wäre an eine Hochdruck-Wasserversorgung angeschlossen, und das Rohr, das der Erdleitung entspricht, würde in die Kanalisation führen. (…) Natürlich wäre ein solcher hydraulischer Computer viel langsamer als der neueste Mikroprozessor, (…) denn Wasserdruck pflanzt sich in Rohrleitungen weitaus gemächlicher fort als elektrischer Strom in einem Draht. Und was die Größe angeht: (…) Ein Transistor in einem Chip hat einen Durchmesser von etwa einem millionstel Meter; ein hydraulisches Ventil misst ungefähr zehn Zentimeter. Bei entsprechender Größe der Leitungen würde der hydraulische Computer eine Fläche von rund einem Quadratkilometer voller Leitungen und Ventile beanspruchen.“9

Musikalischer Trenner: „HiFi” von Kodexx Sentimental aus „Das digitale Herz“

TECHNISCHE STIMME 16
Pause. Fahren Sie erst fort, wenn Sie das Gehörte kognitiv verarbeitet haben. Dies beugt einem stack overflow in Ihrem neuronalen System vor.

[9] Hillis S. 29f.

ERZÄHLERIN 24
Pausen sind in diesem Programm nicht vorgesehen. Wir machen mit einem singenden Mathematiker weiter. Rechnen Sie mit!

Musik 3'01 Titel: „New Math" von Tom Lehrer
Komponist/Texter: Tom Lehrer

ERZÄHLERIN 25
Als der Liedermacher – und zeitweilige Mathematikdozent – Tom Lehrer in den frühen 1960er-Jahren die „neue Mathematik" von Boole'scher Algebra bis Mengenlehre besang, konnten seine Studenten – wenn überhaupt! – nur primitive Hilfsmittel benutzen, den Rechenschieber etwa oder gedruckte Logarithmentafeln. In Büros und Werkstätten taten noch millionenfach mechanische Addierwerke ihren Dienst, die im Grunde nur verfeinerte Leibnizmaschinen waren. Genau darüber hatte sich fünfundzwanzig Jahre zuvor ein junger deutscher Ingenieur geärgert.

S1-26 Zuse 0'16
An der Zahnradkonstruktion sind ja die ganzen Tischrechenmaschinenfabrikanten … nicht gescheitert, aber sie kamen an eine Grenze, wo sie zwar Plus und Minus rechnen konnten, aber dann war Schluss! Das war ausgereizt. Und das war dem Vater auch klar.

TECHNISCHE STIMME 17
Horst Zuse. Emeritierter Professor für Informatik, Berlin.

ERZÄHLERIN
Sein Vater Konrad Zuse konstruierte in den 1930er-Jahren einen binären Rechner. Mit über einer Tonne an verbauten mechanischen Schaltgliedern aus Metall, die sich oft verhakten, war das Gerät kaum einsatzfähig – doch unbestritten der erste programmierbare Computer der Welt.

S1-27 Fuß 0'50
Herr Zuse, geboren 1910, war als Jugendlicher schon ein recht flottes Kerlchen im Geiste, würde man heute sagen.

ERZÄHLERIN
Erzählt Peter Fuss.

S1-27 Fuß 0‘50
Man könnte auch salopp sagen: Der hat’s drauf gehabt! Der hat viele Preise gewonnen bei so Wettbewerben. Heute sagt man „Jugend forscht“ dazu und so weiter. Aber Herr Zuse war beruflich Statiker bei den Henschel Flugzeugwerken in Berlin, und da musste er rechnen, rechnen und schlichtwegs noch mal rechnen! Und das war ihm eigenen Angaben nach zu blöd, ja? Das wollte er nicht machen, jeden Tag dieselben stupiden Dinge, zum Teil ganz simple, das ist träge, öde, langweilig, ja! Er baut sich doch lieber eine Maschine, die für ihn das Rechnen übernimmt.

ERZÄHLERIN
Horst Zuse ergänzt:

S1-28 Zuse 0‘39
Er hatte ja den Tischrechenmaschinenfabrikanten Kurt Pannke an der Hand. Er suchte ja Geld, händeringend. Und hat den mal angerufen und gesagt, was er machen will, und der Kurt Pannke, der baute solche mit Zahnrädern. Und der sagte: „Wissen Sie, Herr Zuse, Sie sind ein netter junger Mann, aber auf dem Gebiet der Rechenmaschinen gibt’s überhaupt nichts mehr zu erfinden, gar nichts! Es ist alles erfunden! Es stimmte auch, was diese zahnradgesteuerten … da konnten Sie auch nix mehr machen! Aber er war so großzügig, hat ihm 1.500 Reichsmark gegeben, so großzügig. „Ja, wenn Sie was haben, dann zeigen Sie’s mir mal!“ Naja, da kam die Z1 bei raus.

ZITATSPRECHER 12
„Z1 funktionierte vollständig mechanisch, (…) wies aber schon den grundlegenden Aufbau eines Computers auf: Er bestand aus a) Datenspeicher, b) Recheneinheit und c) Kontrolleinheit mit Daten-Ein/Ausgabegeräten.“10

TECHNISCHE STIMME 18
Source: Francis Hunger „Computer als Männermaschine“. Internet-Manuskript.

ZITATSPRECHER 13
„Der Nachfolger Z2 basierte dann bereits auf elektromechanischen Relais, die im Vorkriegs-Deutschland einfacher verfügbar waren, als die vorgesehenen Vakuumröhren und wurde gemeinsam von Zuse und seinem Freund und Mitarbeiter Helmut Schreyer 1940 fertiggestellt. (…) Der im Dezember 1941 fertiggestellte Z3 wird inzwischen als der erste vollständig funktionierende Rechner mit automatischem Ablauf der einzelnen Operationen und Binärlogik bezeichnet und nimmt damit die

[10] Francis Hunger: http://www.irmielin.org/wp-content/uploads/2012/01/computer_als_maennermaschine.pdf

Stelle ein, die lange Zeit dem Januar 1943 vollendeten Harvard Mark I zugeschrieben wurde.“11

Musikalischer Trenner

ERZÄHLERIN 26
Peter Fuß demonstriert im Deutschen Museum die Wirkungsweise eines Schaltrelais aus der Zuse Z3.

S1-29 Fuß 0’34
Sie haben hier im Inneren des Relais einen durchgängigen Metallkern. Und dieser Metallkern ist umwickelt mit jeder Menge Draht, sprich mit einer Spule! Kommt nun der Strom, wird das Ganze zum Elektromagneten. Das heißt, hier vorne zieht der sogenannte „Anker“ an … ja … und er betätigt die darüber liegenden Kontakte. Sehen Sie’s hier … alles, was offen ist, macht zu, was zu ist, macht auf … und das wechselt hin und her …

S1-30 Zuse 0‘15
Die Z1 war schon ein Digitalrechner, funktionierte schlecht. Aber die Z3 ist ein Digitalrechner, ganz klipp und klar! Nur Nullen und Einsen, alles Nullen und Einsen! Und damit hat er damals schon den ersten Digitalrechner gebaut.

ERZÄHLERIN 27
Der wonach verlangte? Genau: nach einer Programmiersprache! Auch in diesem Punkt erwies sich Konrad Zuse als Pionier, von dem kaum einer etwas ahnte. Denn in der Abgeschiedenheit eines Allgäuer Bergdorfs, wohin es Zuse nach 1945 verschlagen hatte, entstand in der unmittelbaren Nachkriegszeit die allererste Programmiersprache der Welt. Sohn Horst Zuse wiegelt freilich etwas ab:

S1-31 Zuse 0‘07
Also ich würde sagen, es ist ein Entwurf einer Programmiersprache. Faszinierender Entwurf – warum?

ERZÄHLERIN 28
Weil das 300-seitiges Typoscript mit dem Namen „Plankalkül“ bar jeder praktischen Erprobung am Schreibtisch niedergeschrieben wurde, rein aus der Imagination der Maschine heraus. Der Impuls dazu war allerdings noch im Angesicht der Hardware aufgeflammt:

[11] a.a.O.

S1-32 Zuse 0‘45
Da hat er schon 1942 drüber nachgedacht, hier in Berlin: Ich kann doch keinen Mathematiker begeistern, oder salopp gesagt hinterm Ofen vorlocken, wenn der den Lochstreifen nur bestechend aus Löchern per Hand stanzen muss, und die ganzen Instruktionen, das ist dann ne typische Maschinensprache. Da kann ich doch niemanden begeistern mit, dass der meine Maschinen benutzt. Da war ihm 42/43 – da gibt’s auch Unterlagen – schon klar, also das muss anders gehen! Also der Mathematiker muss meine Maschinen über eine Sprache bedienen, und zwar über eine Sprache, die er versteht. Und das ist die Mathematik, wo er dann einfach auch so angeben kann, was gemacht werden soll.

ERZÄHLERIN 29
Bei Mathematik – beziehungsweise klassischen Rechenregeln – blieb es nicht, denn Programmieren heißt ja, das Boole‘sche Konzept der formalen Logik auf die Maschine zu übertragen. Rasch stieß Konrad Zuse dabei auf ein komplett nutzloses, die Menschheit jedoch seit Jahrhunderten faszinierendes Regelwerk, das sie selbst geschaffen hat: das Schachspiel. In seiner Autobiografie schreibt er dazu:

ZITATSPRECHER 14
„Ich selbst habe schon während des Krieges und kurze Zeit danach versucht, eine algorithmische Formelsprache zu entwickeln, die die numerischen Rechnungen als Nebenzweck behandelt, in erster Linie aber rein logischen Kombinationen dient. Das Schachspiel diente mir dabei als ein gutes Modell. Nicht, weil ich es als besonders wichtig erachtete, dass man eines Tages einen Schachspieler mit dem Computer besiegt, sondern nur, weil ich ganz bewusst ein Gebiet suchte, auf dem möglichst alle komplizierten und komplexen Kombinationen vorkommen.“12

S1-33 Zuse 0‘34
Er hat auf dem Schachspiel alle Figuren erstmal kontrolliert. Also wie gehen die Springer, Turm, Rochade, König, Dame … wie dürfen die überhaupt ziehen? Welche Spiele beeinflussen sie? Und er hatte auch noch eine Bewertungsfunktion, also um eine Schachsituation zu bewerten. Denn Sie wissen ja, beim Schachspiel sind die Figuren unterschiedlich wichtig! Von den Bauern haben Sie viel, von der Dame haben Sie nur eine, von den Türmen haben Sie zwei, und das hatte auch ne Bewertungsfunktion. Also das ist schon eine erstaunliche Entwicklung, die er da gemacht hat.

[12] zit. nach Horst Zuse „Geschichte der Programmiersprachen“, Typoscript der TU Berlin 1999, S. 62f.

ERZÄHLERIN 30
Sie wurde allerdings nie auf einem Zuse-Computer oder anderer Hardware „implementiert“, wie es im Fachjargon heißt, weil der Plankalkül erst 1972 an die Öffentlichkeit kam. Längst hatten da andere Programmiersprachen wie Algol und Fortran das Rennen gemacht – beides amerikanische Reaktionen auf dasselbe Bedürfnis der Mathematiker, die neuen Maschinen nicht Bit um Bit ansteuern zu müssen, sondern komplette Befehle zu besitzen. Beim Plankalkül waren sie diesem Ziel – theoretisch! – zum ersten Mal nahe gekommen.

S1-34 Zuse 0‘12
Was der Vater auch hatte zum Beispiel: Addition, Multiplikation und so was. Dass man da nicht mehr eine Extrainstruktion auf den Lochstreifen schreiben muss, sondern einfach ein Malzeichen angibt. Das konnte der Plankalkül auch.

Musik 1’30 „That’s Mathematics” von Tom Lehrer

ZITATSPRECHER 15
„Würden Sie bitte die zwei Stücke über Binärarithmetik lesen? Ich meine, ganz durchlesen? Das wird Sie, das gebe ich zu, 15 oder 20 Minuten kosten. (…) Stecken Sie doch einfach das Buch in Ihre Tasche!“[13]

TECHNISCHE STIMME 19
Sprungverweis: goto Frederik Pohl „Signale“. read Vorwort.

ZITATSPRECHER 16
„Wenn dann Ihr Zug das nächste Mal Verspätung hat, (…) dann nehmen Sie (…) die Geschichten in Angriff und lesen Sie sie! Wären Sie nicht gerne der erste aus Ihrem Häuserblock, der wie ein Computer rechnen kann? Ja?“[14]

ERZÄHLERIN 31
Digitale Träume beflügelten nicht nur Ingenieure wie Konrad Zuse, sondern auch Literaten wie den amerikanischen Science-Fiction-Autor Frederik Pohl. Während sich dessen Kollegen Anfang der 60er-Jahre hauptsächlich mit Raumfahrt beschäftigten, wollte Pohl den Umgang mit Null und Eins popularisieren. Kein einfaches Unterfangen, wie er erkennen musste:

[13] Frederik Pohl „Signale“, S. Fischer 1975, S. 8
[14] a.a.O.

ZITATSPRECHER 17
„Die wesentliche Schwierigkeit des Binärsystems liegt im Aussehen der Binärzahlen selbst. Sie wirken unschön, linkisch und merkwürdig. Sie sehen aus wie das Gestotter einer elektrischen Schreibmaschine, bei der eine Type hängt, und sie lassen sich auch nur sehr schwer aussprechen. Eine Zahl wie 11110101000 wird von den wenigsten Menschen schnell erkannt werden, obwohl (…) dies (…) nach der dezimalen Schreibweise 1960 ist.“15

ERZÄHLERIN 32
Um diese Fremdheit des digitalen Abstraktums aufzuheben, demonstrierte Frederik Pohl im Aufsatz „Die Kunst mit den Fingern zu rechnen“, wie einfach sich unsere Grundrechenarten binär vollziehen lassen. Im Radio ist das schwer vermittelbar, während die zweite Pohl’sche Idee fürs Radio nachgerade geschaffen zu sein scheint. Der Autor will nämlich mit dem Computer reden, und zwar auf dem Niveau der Maschine. Dazu nimmt er Anleihen beim Morsecode des Funkverkehrs:

ZITATSPRECHER 18
„Es gibt ein weit verbreitetes Aussprachesystem für die Punkte und Striche des Morsealphabets: dit ist ein Punkt, dah ist ein Strich. (…) Während Jahrzehnten hat es bei unzähligen Funkern in der ganzen Welt funktioniert. Wir wollen 1 als dit und 0 als dah aussprechen. 111.11100.11011 wird dann zu dididit dididididahdah dididahdidit.“16

ERZÄHLERIN 33
Verständlicher wäre allerdings ein System aus Phonemen, das mehr als nur dit und dah kennt, zum Beispiel eigene Laute für die wichtigsten Zahlen besitzt. In Pohls Sprechweise hier nun die Ziffern 1 bis 20. Autor Pohl lässt es sich dabei nicht nehmen, seinen eigenen Namen als Ausgangsphonem zu setzen:

ZITATSPRECHER 19 (englische Intonation beachten!)
pohl / poot / pahtah / pod / too / tot / dye / tee / ooty-pohl / ooty-poot / ooty-pahtah / ooty-pod / ooty-too / ooty-tot / ooty-dye / ooty-tee / ahtah-pohl / ahtah-poot / ahtah-pahtah / ahtah-pod / ahtah-too

ERZÄHLERIN 34
Einhundert wäre demnach “poot one group too-too”, eintausend “ooty-tee one group totter-pohl” und eine Million “pod three group dye-too ooty-poot ohly-pohl”. Man kann davon ausgehen, dass Frederik Pohl Spaß daran hatte, sich mit diesem

[15] Pohl S. 150
[16] Pohl S. 142

dadaistisch klingenden System ein Denkmal zu setzen. Allerdings ist es todernst gemeint, denn der Aufsatz schließt mit den Worten:

ZITATSPRECHER 20
„Tatsächlich scheint es nur noch ein kleiner Schritt zu sein zum gesprochenen Diktat an den Computer über Binärzahlen und Anweisungen und auch, falls dies gewünscht wird, zur gesprochenen Antwort, aber da der Autor nicht seinen vorliegenden Beitrag mit früheren Publikationen auf dem Gebiet der Science Fiction vermischen möchte, wird er solche Vorschläge der Prüfung durch akademischere Geister überlassen.“17

ERZÄHLERIN 35
Sie erfolgte niemals. Was vielleicht auch daran liegt, dass man schon an diesem kurzen Beispiel zu erkennen vermag, wie absurd der Gedanke ist, Menschen könnten mittels binärer Codes kommunizieren. Warum sollten sie das auch tun? Der Code ist schließlich weitaus primitiver und informationsärmer als jede menschliche Sprache. Der Jurist und Programmierer Christoph Kappes bringt es auf den Punkt:

S1-35 Kappes 0‘25
Die Leistung, die Menschen kulturell beim Sprechen und bei dem Gebrauch von Sprache bringen, ist um ein Vielfaches höher, als das Maschinen tun, wenn sie Programmiersprachen ausführen. Und ich traue deswegen auch Maschinen oder Computern eben nur sehr begrenzt zu, das Niveau von Sprachakteuren zu erreichen. Weil das, was man dafür können muss, schon eine ganz gewaltige Leistung ist.

ERZÄHLERIN 36
… die trotz aller Verblüffung über „Sprachassistenten“ wie Siri oder Alexa auch heute noch nicht erbracht wird. Umgekehrt reicht mittlere menschliche Intelligenz, um den Computer zu begreifen – jedenfalls, wenn man von Christoph Kappes unterrichtet wurde. Zu Beginn dieser Langen Nacht hat er geschildert, wie er im Computer-Anfängerkurs eine binäre Multiplikation theatralisch inszenierte. Lassen wir zum Abschluss der ersten Stunde diese Szene noch einmal aufleben:

S1-36 Kappes 0’31
Ich hab denen Filtertüten in die Hand gegeben, für die Speicherplätze, weil ansonsten hätte ja jede Person ein Bit sein müssen und aufstehen oder liegen als Bit quasi. Das war mir zu umständlich, weil dann hätte ich … weiß ich nicht, 256 Personen für eine einfache Addition gebraucht! Ich hab dann mich entschlossen, dass ich das etwas abstrahiere, und jeder hatte eine Filtertüte in der Hand, und in dieser Filtertüte waren Zahlen. Und dadurch waren sie quasi diejenigen, die die Variablen in der Hand hatten.

[17] Pohl S. 158

Da konnte man Zahlen in die Filtertüte tun, und dadurch waren sie ein Speicherplatz mit einer Variablen.

Männerstimme aus dem Musiktitel sagt trocken: „I'm the operator with my pocket calculator."

Musik 3'45 (Blende zu Nachrichten)
Titel: „Pocket Caluclator" Interpret: Balanescu Quartet

TECHNISCHE STIMME 20 über Musik
interrupt Lange Nacht goto Lange Nacht

ERZÄHLERIN 37 über Musik
Mit Geschichte und Geschichten der Programmiersprachen geht es nach den Nachrichten weiter.

2. Stunde

„Männer löten – Frauen denken“

Gesprächspartner dieser Stunde
Dr. Martin Burckhardt, Philosoph, Programmierer (Berlin)
Fiona Krakenbürger, Techniksoziologin (Berlin)
Bernd Leitenberger, Programmierer, Buchautor (Ostfildern)
Kathrin Passig, Programmiererin, Autorin (Berlin)
Prof. Dr. Hans Jürgen Schneider, Informatiker (Erlangen)
Prof. Dr. Jochen Ziegenbalg, Informatiker (Karlsruhe/Berlin)

Musik: „Luminitza“ vom Balanescu Quartet, O-Töne und Technische Stimme darüber:

S2-01 Passig 0‘16
Ich hab auf der „re:publica“ mal mit Anne Schüssler zusammen einen Workshop „Programmieren für Nullcheckerbunnies“ veranstaltet. Und da haben wir am Eingang abgefragt … also es durften nur Leute rein, die definitiv überhaupt keine Ahnung haben, damit alle sich in guter Gesellschaft fühlen und nicht denken, um sie rum wissen alle schon Bescheid.

ERZÄHLERIN 01
„Männer löten, Frauen denken.“ Die zweite Stunde der Langen Nacht der Programmiersprachen.

S2-02 Passig 0‘38
Und da haben wir versucht, ein paar wesentliche Dinge, die beim Programmieren passieren, am Beispiel von Excel-Spredsheets zu erklären. Weil damit arbeiten ja doch die meisten im Beruf. Und letztlich ist der Unterschied gar nicht groß. Also ob man da jetzt da ne Tabellenzelle hat und in die einen Wert reinschreibt oder ob man eine Variable hat und in die den Wert reinschreibt, ist in der Theorie überhaupt kein großer Unterschied. Und trotzdem haben die Leute eben nicht das Gefühl, dass da, was die da machen, Programmierung ist. Obwohl man, wenn man – also vor der Erfindung von Spredsheets – da drauf geguckt hätte, wahrscheinlich gesagt hätte: „Natürlich wird da programmiert! Die Leute werden eine aufwändige Ausbildung brauchen, um diese komplizierte Sache zu beherrschen!“

ERZÄHLERIN
Erklärt Kathrin Passig und Jochen Ziegenbalk ergänzt:

S2-03 Ziegenbalg 0‘45
Wenn man zum Beispiel an die Programmierung von Tabellenkalkulationen denkt, Tabellenkalkulationssystemen. Für viele Leute wäre das gar keine Form der Programmierung. Aber es ist eine der Möglichkeiten, ein algorithmisch gegebenes Problem auf den Computer zu übertragen, durch den Computer realisieren zu lassen. Es ist eine Form der Programmierung! Aber trotzdem, viele Leute denken, wenn sie ein Tabellenkalkulationsblatt erstellen, gar nicht an eine Programmierung. Aber dann haben sie irgendeine Zelle, wo sie eine wenn/ dann-Bedingung brauchen und verwenden müssen, einsetzen, letztlich programmieren müssen. Und dann ist es hilfreich zu wissen, dass eine solche wenn/dann-Beziehung in einem Tabellenkalkulationsblatt, dass das eine Funktion ist, die einen Funktionswert liefert.

S2-04 Passig 0‘21
Ich bin eigentlich sehr spät zum Programmieren gekommen, und das Problem dabei war, dass ich eigentlich schon, seit ich 14 war oder wahrscheinlich noch früher, von Leuten umgeben war, die das schon konnten. Also für mich war das sehr unattraktiv! Ich hatte das Gefühl, die machen das, seit sie acht sind – war wahrscheinlich auch korrekt! – und das ist unaufholbar.

ERZÄHLERIN 02
Kathrin Passig. Programmiererin. Schriftstellerin. Federführende Autorin des Buches „Weniger schlecht programmieren“. Trägerin des Ingeborg-Bachmann-Preises und des Johann-Heinrich-Merck-Preises für literarische Kritik und Essay. Sie kann sich im technischen und im literarischen Universum gleichermaßen versiert bewegen.

S2-05 Passig 0‘32
Ich bin in den 90ern mal mit einem dicken Java-Buch auf Papier in Urlaub gefahren und hab das von vorne bis hinten durchgelesen und dann aber gedacht: „Java, das können jetzt schon alle, jetzt ist es zu spät!“ Das muss 97 gewesen sein (lacht), also die Zeiten, die Sascha Lobo in „Strohfeuer“ beschreibt, wo er – und dieser Teil ist glaub ich autobiographisch – an der TU Berlin rumsteht und jedem, der in der Lage ist, am Montagmorgen in Stralsund Java-Kenntnisse auch nur vorzutäuschen, 30.000 DM auf die Hand verspricht. (lacht)

ERZÄHLERIN 03
Die Erinnerung trügt ein wenig. Es ging nicht um Stralsund, sondern um den Ort Sassnitz auf Rügen.

ZITATSPRECHER 01

„Leider waren die Kandidaten entweder viel zu teuer, ahnungslos, unwillig, in Sassnitz zu arbeiten, oder alles gleichzeitig. (...) Ein Münchner Informatikdoktorand hatte eine Antrittsprämie von vierzigtausend Mark in bar gefordert, ohne Rechnung. Ein Bielefelder, nach eigenen Angaben «ein autodidaktisches Genie», bot an, das gesamte Projekt in einer von ihm selbst entwickelten Sprache zu realisieren, Java sei völlig untauglich. Mit dem letzten Punkt hatte er vermutlich sogar recht. Aber die Unternehmensberatung bestand darauf, das Projekt in Java entwickeln zu lassen. Die Schuld daran trug der Projektleiter, ein hektischer, unseriös wirkender Mann namens Vonnebrink. Er hatte beim Auftraggeber seine Ahnungslosigkeit vertuschen wollen und behauptet, dass es nach eingehender Prüfung der technischen Abteilung eine Realisierungschance für das komplizierte Projekt ausschließlich mit Java gebe – die einzige Programmiersprache, die ihm spontan einfiel.“18

TECHNISCHE STIMME 02

Source: Sascha Lobo „Strohfeuer“. Roman. Seite 96ff.

ZITATSPRECHER 02

„Schon im Lift zur fünften Etage des Universitätsgebäudes, wo die Informatik untergebracht war, schlug mir das Nerdtum entgegen. (…) In der mit alten Sofas vollgestellten Cafeteria des Fachbereichs hockten zehn junge Männer an Laptops, rein visuell alle Volltreffer. So sahen Programmierer aus. In der Tür stehend, räusperte ich mich und begann mit meiner Ansprache. «Hey, Leute, ich weiß, das ist ein bisschen ungewöhnlich, aber ich habe ein Problem und hoffe, dass einer oder zwei von euch es lösen können. Gibt auch ordentlich Geld. (…) Wer von euch kann Java?»
Fast alle Hände gingen hoch. Einer der Jungs sagte, ohne aufzusehen, Java sei doch nicht gleich Java, wie könne man eigentlich eine solche Frage stellen. Darauf ging ich lieber nicht näher ein. «Es ist so, ich hab eine Agentur, und wir haben einen großen Auftrag bekommen. Alles in Java, ist nicht das Sinnvollste für das Projekt, aber der Kunde will es eben so. Leider ist der Auftrag in ...»
«Wenn die Java wollen, ohne zu wissen warum, wird das ein Katastrophenprojekt, das kann ich dir jetzt schon sagen, so was habe ich schon mal gemacht, nie wieder!», sagte ein Junge mit langen, grüngefärbten Haaren. (…) Weil ich den Eindruck hatte, dass die Nerds mich und mein Anliegen nicht besonders ernst nahmen, wedelte ich mit dem Geld herum. «Hier. Zehntausend. Das ist die Antrittsprämie für denjenigen, der am Montag in Sassnitz Java programmiert. Und den Vertrag für sechs Monate unterschreibt, natürlich.».“19

[18] Sascha Lobo „Strohfeuer“, Rowohlt Verlag 2012, S. 96ff.
[19] a.a.O.

TECHNISCHE STIMME 03
Sprungverweis: goto Ellen Ullman „Close to the Machine“. Mein Leben mit dem Computer. read Seite 108.

SPRECHERIN ZITATE ELLEN ULLMAN 01
„»Ich würde gern Java-Programmierungen machen«, sagte Mark.
»Java?« fragte ich. »Wer bezahlt dir, dass du Java lernst?« Zu dem Zeitpunkt war Java eine noch kaum bekannte Computersprache. Es war der letzte Schrei, die Sprache des Internets. Uns war allen klar, dass wir sie lernen mussten, aber keiner wusste genau wie und wann. »Wir haben unser SQL schon vergessen«, sagte ich, »und in C++ sind wir noch nicht sehr weit. Müssen wir uns da schon an Java ranmachen?«
Mark und ich schauten uns an. Beide waren wir mit der Programmiersprache C »erwachsen« geworden, und mit der Nachfolgerin C++ konnte keiner von uns richtig umgehen. Aber Mark wollte nicht, dass ich darüber sprach.
»Randy bringt mir Java bei«, sagte Mark. »Er ist ein alter Java-Hase.«
»Ein alter Java-Hase?« fragte ich. »Wie alt kann der Hase denn sein? Java selbst ist ja erst ein Jahr alt.«
»Randy hat das damals im Mai gelernt.«
Wir führten dieses Gespräch im September. »Ach so, damals im Mai!«
»Genau, das ist heutzutage alt.«
»Richtig«, sagte ich, »uralt.«“[20]

Musikalischer Trenner

S2-06 Passig 0’44
Aber jedenfalls, ich habe so alle diese Züge verpasst!

ERZÄHLERIN
Gesteht die Programmiererin Kathrin Passig

S2-06 Passig 0’44
Und erst als ich dann mit der „Zentralen Intelligenz Agentur“ mit Anfang 30 ungefähr … na, es war ein bisschen vorher, aber so um die 30 plötzlich die Einzige war, die irgendein Interesse für diese Dinge aufgebracht hat, wo die anderen alle noch weniger Ahnung hatten, da hab ich dann auch damit angefangen, da selber was zu machen. Seitdem empfehle ich immer allen, die sagen „Ja ich würde ja gerne, weiß aber nicht wie“ erstmal … also sie sagen dann meistens so was wie „mein Freund ist Softwareentwickler“. Dann sag ich immer: „Du musst als erstes den Freund loswerden!“ (lacht) Das ist so wenig motivierend, wenn man immer genau weiß, man

[20] Ellen Ullman „Close to the Machine”, Suhrkamp Verlag 1999, S. 108-109

sitzt da und stellt sich dumm an und man bräuchte nur eine Frage zu stellen, und dann wüsste der andere sofort die Antwort! Auf die Art kommt man schlecht voran.

S2-07 Krakenbürger 0’12
Ich setz mich total viel ein dafür, dass Frauen auch mehr programmieren lernen und dass wir da mehr Diversität entstehen lassen, aber ich glaube nicht, dass jede und jeder programmieren lernen muss. Aber ich möchte gerne, dass Leute das verstehen und Menschen ihre Angst davor verlieren.

TECHNISCHE STIMME 04
Fiona Krakenbürger. Techniksoziologin. Berlin.

S2-08 Krakenbürger 0‘22
Ich kann für mich sprechen: Ich bin extrem dankbar dafür, dass [sich] diese Tür für mich geöffnet hat, dass ich meine Berührungsängste verloren hab und dadurch überhaupt erst so ne Begeisterung entdecken konnte! Die vielleicht schon vorher dagewesen wäre, wenn ich ebenfalls zum Geburtstag nen Kosmos-Kasten bekommen hätte und nicht die Fisherprice-Küche, die auch super war, aber ne – es gab da halt klare Rollenverteilungen für mich und ein daraus erwachsener Glaube daran, was ich kann und was nicht.

ERZÄHLERIN 04
Fiona Krakenbürger kann Assembler. Das ist gewissermaßen die Ursprache des Computers, erfunden im Holozän des Computerzeitalters zwischen 1948 und 50, nur eine Ebene über dem binären 0/1-Code, eine Sammlung von Befehlssätzen, die wenig Raum für gedächtnisfreundliche sprachliche Assoziationen bietet. Bis heute werden vor allem Chips in Maschinen, Haushaltsgeräten und Spielzeugen in Assembler programmiert.

S2-09 Krakenbürger 0‘27
Ein gutes Beispiel dafür ist ein Tamagotchi! Es gibt einen tollen Talk von einer Hackerin, die sehr, sehr viel Zeit damit verbracht hat, ihren Tamagotchi zu reverse-engeneeren. Die hat den aufgemacht und dann versucht, den Programmcode rauszukratzen. Und das lag eben in Assemblercode vor – also in einer Art Assemblercode – und damit hat sie dann versucht rauszufinden: „Okay, wie ist dieser Tamagotchi programmiert“, und hat dann so nach und nach sich rangetastet, den Code entschlüsselt, und kann jetzt ihren Tamagotchi anpassen an ihre eigenen Bedürfnisse!

ERZÄHLERIN 05
Allerdings beherrscht Fiona Krakenbürger Assembler nur rudimentär, wie das eben so ist, wenn man sich nur ein paar Monate damit beschäftigt hat. Aber der Ort, von dem man aufbricht, entscheidet über den inneren Wert der Kenntnisse. Als sie sich vor sieben Jahren entschloss, das Wesen des Computers von Grund auf begreifen zu wollen, studierte sie nichts Technisches, sondern Ethnologie. Und wie eine gute Ethnologin führte sie über ihre Feldforschung am eigenen Objekt Tagebuch. Öffentlich, als Blog:

SPRECHERIN ZITATE WEIBLICH ALLGEMEIN 01
„Und noch eine Sache. Dass ich diesen Artikel verfasst habe, ist auch ein wenig befreiend für mich. Wenn ich offen damit umgehe, dass mir das Programmierenlernen nicht ganz so leichtfällt, heißt das für mich persönlich auch, es zu akzeptieren. Und für die LeserInnen womöglich auch. Manchmal habe ich den Verdacht, dass ein falscher Eindruck in diesem Internet von mir kursiert. Mir fällt das mit dem Programmierenlernen wirklich verdammt schwer. Ich kann ne Menge Dinge. Ich kann wenige Dinge gut und nichts wirklich richtig gut. Außer Tofu braten vielleicht. (…) Klar, ich hab mit Assembler angefangen. Aber ich bin keine Assemblerprogrammiererin. Ich weiß, was Assembler ist und kann kompilierten Code einigermaßen nachvollziehen. Ich wende es aber nicht an, und ich verstehe fremden Assemblercode nicht ohne Weiteres. Ich habe dafür ne Menge über Computerarchitektur und -funktionsweise gelernt. Das heißt aber nicht, dass ich alles verstanden habe, was mir beigebracht wurde. (…) Wenn ich in diesem Blog zum Programmierenlernen motivieren möchte, ist es nicht zielführend wenn ich vorgebe, das wäre eine einfache Angelegenheit. Das kann es für manche sein, für mich gilt das nicht. Ich habe andere Stärken. Ein Musikinstrument lerne ich in Nullkommanichts. Eine neue Programmiersprache – das dauert.“21

S2-10 Krakenbürger 0‘19
Manchmal ist vielleicht so ein komplett nutzloses Herangehen an Technologielernen der richtige Weg! Weil’s nicht darum geht: „Du musst programmieren lernen und dann Entwicklerin werden!“ Sondern: „Ne, schau’s dir doch einfach mal an! Guck mal, wie das funktioniert! Ist es nicht interessant, es rauszufinden?“ Und ich glaube, dass so ne Neugierde eigentlich jedem innewohnt. Man muss halt nur gucken, wie man daran anknüpfen kann.

[21] https://fionalerntprogrammieren.wordpress.com/category/c/

ERZÄHLERIN 06
Die Technik-Neugierde führte Fiona Krakenbürger weg von der Ethnologie, hin zur Techniksoziologie. Mittlerweile arbeitet sie in einem Open-Source-Projekt. Der nichtlineare Weg zum Computer ist dabei mehr als typisch … für Frauen. Ihnen wurde in den vergangenen Jahrzehnten kaum nahegelegt, sich intensiv mit der neuen Weltmaschine zu befassen. Schon die in den 1950er-Jahren geborene amerikanische Software-Ingenieurin Ellen Ullman beschreibt in ihrem vorhin zitierten Erfahrungsbericht „Close to the Machine“ wie sie Ende der 1970er-Jahre als Programmiererin in eine reine Männerwelt eindrang.

TECHNISCHE STIMME 05
Kommentar außerhalb der Programmzeilen: Obwohl 20 Jahre alt, eine unbedingte Lektüreempfehlung!

ERZÄHLERIN 07
Als die Autorin und Programmiererin Kathrin Passig, Jahrgang 1970, dann ins Jugendalter kam, war diese Welt mit dem Homecomputer bereits bis in die Kinderzimmer vorgedrungen – in die Zimmer die Jungen, wohlgemerkt, nicht der Mädchen. Die Jungen waren auf die technische Revolution gut vorbereitet, hatten sie zuvor doch an Fischertechnik- und Kosmos-Elektronikbaukästen geübt. Oder wie es Max Goldt mit satirischer Präzision auf den Punkt bringt:

ZITATSPRECHER 05
„Auch vor dem Einzug der Elektrogehirne in Privathaushalte gab es schon nerds. Sie beschäftigten sich mit CB-Funk und löteten. Was viele heute nicht mehr wissen: Löten war früher eine der wichtigsten Beschäftigungen außerdienstlich unterforderter Männer.“22

TECHNISCHE STIMME 06
Männer löten – Frauen denken.

ERZÄHLERIN 08
Aber es ist auch umgekehrt denkbar. Schließlich gab es in Ost- wie in Westdeutschland den Beruf der Löterin – was zu sozialkritischen Liedern animierte, die man mit respektablem Ernst vortragen sollte.

(8) Musik 1’43 „Du kleine Löterin” Wiglaf Droste und das Spardosen-Terzett

[22] zit. nach Max de Bruijn "Wie werde ich Bill Gates?", S. Fischer Verlag 2000, S. 15

S2-11 David Lettermann (historischer O-Ton)
Please welcome Admiral Grace Hopper!

ERZÄHLERIN 09
Eine alte Dame sitzt 1986 mit schlohweißem Haar neben dem US-Talkmaster David Letterman. Entgegen ihrer sonstigen Gewohnheit, nur in Uniform ihres militärischen Ranges aufzutreten, ist sie hier zivil gekleidet. Zahllose Videos von ihr, diesich mehr als 25 Jahre nach ihrem Tod in YouTube finden lassen, zeigen sie – Admiral Grace Hopper – bei Vorträgen stets korrekt uniformiert. Und Vorträge über die Geschichte der Computerprogrammierung hielt sie bis an ihr Lebensende – weil sich diese Geschichte mit ihrer eigenen Lebensgeschichte deckte. Sie beginnt im 2. Weltkrieg.

S2-12 Grace Hopper (historischer O-Ton) 0'11
I think we've totally forgotten the environment in which MARK 1 appeared. 41 was Pearl Harbour. By 43 we were in thicker things.

Overvoice ad libitum:
Ich denke, wir haben völlig vergessen, unter welchen Umständen die MARK 1 entstand. 1941 war Pearl Harbour. 43 steckten wir tiefer im Schlamassel.

ERZÄHLERIN 10
Geboren 1906, promovierte Grace Hopper in Mathematik und unterrichtete dieses Fach bis 1944 am Vassar College. Als sie nach Pearl Harbour der Familientradition folgend in die US-Navy eintrat, wurde sie statt auf ein Schiff nach Harvard kommandiert. Dort stand die MARK 1, ein elektromechanischer Relaisrechner wie Zuses Z3 und Z4, der ballistische Berechnungen durchführen sollte. Hier beginnt die Geschichte der angewandten Programmiersprachen – und zwar im Zusammenspiel von militärischem Ingenieursgeist bei der Hardware und weiblichem Scharfsinn bei der Software.

TECHNISCHE STIMME 07
Rückverweis: Konrad Zuses zeitgleich erdachtes Konzept des „Plankalkül" blieb bekanntlich ein Papiertiger.

ERZÄHLERIN 11
Als sich Fiona Krakenbürger vor ein paar Jahren – quasi als Urenkelin Grace Hoppers – mit dem Erlernen von Assemblercode herumschlug, beamte sie sich gleichsam auf das Ausgangsterrain der Pionierin zurück. Mehr als alle Männer um sie herum hatte sich Hopper an den kommunikativen Mängeln des binären Codes gestört. Zwar war sie selbst als Mathematikerin mit abstrakten Denkweisen und komplizierte Formeln vertraut, doch:

ZITATSPRECHER 06
„Anders als die meisten anderen Mathematikprofessoren legte sie großen Wert darauf, dass ihre Studenten gut schreiben konnten. Ihren Statistikkurs begann sie mit einer Vorlesung über ihre Lieblingsformel und ließ die Studenten anschließend einen Aufsatz darüber schreiben, den sie nach Klarheit des Ausdrucks und Schreibstil benotete. »Ich bekleckste [die Aufsätze] über und über mit roter Tinte und erntete erboste Proteste, man habe schließlich einen Mathekurs belegt und keinen Englischkurs«, erinnert sie sich. »Dann erklärte ich, dass es sinnlos sei, Mathematik zu lernen, wenn man sein Wissen nicht mit anderen Menschen zu teilen vermöchte.«23

TECHNISCHE STIMME 08
Source: Walter Isaacson „The Innovaters“. Seite 117.

ERZÄHLERIN 12
Dieses Zentralmotiv – nämlich kommunizierbare Formen für schwierige logische Prozeduren zu entwickeln – prägte Grace Hoppers gesamtes Berufsleben, und sie wurde zur ersten Vermittlerin zwischen Mensch und Maschine.

ZITATSPRECHER 07
„Im Jahr 1952 hatte sie den ersten funktionierenden Compiler der Welt entwickelt, das A-0-System, das die Zeichen des mathematischen Codes in eine Maschinensprache übersetzte und damit das Schreiben von Programmen auch für Normalsterbliche möglich machte.“

ERZÄHLERIN
Dazu der Informatiker Jochen Ziegenbalg:

S2-13 Ziegenbalg 0‘18
Wenn man ein bestimmtes Programm vor sich sieht – vielleicht aus 10 Zeilen oder aus 100 Zeilen oder wie auch immer – dann gibt es eine Form der Übersetzung, das ist die sogenannte Compilierung. Die nimmt das Gesamtprogramm und übersetzt es auf einen Schlag in die Maschinensprache. Ausgeführt wird immer nur das Maschinenprogramm!

ERZÄHLERIN 13
Das A-0-System war ein Vorläufer des heute kaum noch als allgemeinverständlich betrachteten Assembler …

[23] Walter Isaacson „The Innovaters“. Bertelsmann Verlag 2018. S. 117

S2-14 Ziegenbalg 0‘06
Es soll früher Mensch gegeben haben, die auch Assemblercode lesen konnten, aber das Normale ist das nicht.

ERZÄHLERIN 14
… doch aus damaliger Sicht schuf es schon einen kommunikativen Fortschritt. Insgesamt krempelte Hopper die zuvor isolationistische Arbeitsweise der Computertüftler um:

ZITATSPRECHER 08
„Wie Seeleute es tun, pflegte Hopper bei der Arbeit ein zupackend-hemdsärmeliges Miteinander und wurde damit zu so etwas wie einer Geburtshelferin der Open-Source-Bewegung: Sie sandte die ersten Versionen des Compilers an ihre Freunde und Bekannten im Programmiererkosmos und bat um Verbesserungsvorschläge. Dasselbe tat sie als technische Leiterin bei der Koordination der COBOL-Entwicklung, der ersten betriebssystemunabhängigen standardisierten Computersprache. Ihr Gefühl, dass das Programmieren hardwareunabhängig gestaltet werden sollte, entsprang ihrer Vorliebe für ein kollegiales Teamwork, selbst Maschinen sollten gut zusammenarbeiten können, fand sie. Es zeigte auch, wie früh sie eine wichtige Tendenz des Computerzeitalters erkannt hatte: dass die Hardware zu einem Alltagsprodukt mutieren würde, während sich das Programmieren als wahrhaft werthaltig erweisen sollte. Bis Bill Gates auftauchte, war dies eine Einsicht, die den meisten Männern verschlossen blieb.“24

TECHNISCHE STIMME 09
Männer löten, Frauen ...

ERZÄHLERIN 15
… strukturieren, organisieren, kommunizieren. Kurzum: Frauen machen aus einem Haufen zusammengebastelter Einzelteile eine brauchbare Maschine.

Musikalischer Trenner

TECHNISCHE STIMME 10
Sprungverweis: goto Martin Burckhardt „Eine kurze Geschichte der Digitalisierung“. read Seite 156.

[24] Isaacson S. 149

ZITATSPRECHER 09

„Um zu demonstrieren, dass die wahre Kunst der Programmierung nicht in der Kenntnis der Hardware, sondern in der Einbildungskraft lag – immer grundiert von der Bereitschaft, lieb gewordene Überzeugungen über Bord zu werfen –, rekrutierte Hopper ihre Programmierer nicht selten unter den Sekretärinnen. Sie hatte beobachtet, dass die jungen Frauen ihre Arbeit höchst gewissenhaft erledigten – und damit eine Tugend an den Tag legten, die im Umgang mit der Maschine unerlässlich war. Hoppers ‚Marilyn-Projekt' – also die Verwandlung des dummen Blondchens zum Programmierer-Genie – wurde belohnt. Sehr bald schon bestand mehr als die Hälfte ihrer Programmiererschar aus Frauen, was bewies, dass man weder höhere Mathematik noch Quantenmechanik beherrschen musste, um einen Computer programmieren zu können."[25]

ERZÄHLERIN 16

Allerdings drehte der Wind rasch. Auf Grace Hoppers Pionierphase folgte die Restauration. Sobald die Männer erkannt hatten, wo die Macht, das Geld und der Einfluss im Computerbusiness liegen würden, verschwanden die Frauen wieder aus der Softwareentwicklung, so dass sich Ellen Ullman in den 1980er-Jahren als Exotin fühlen musste. Unbestreitbar exotisch – zumindest aus Sicht des Mannes – scheint allerdings ihr Verhältnis zur Hardware, quasi zum Faktor Lötkolben zu sein. Während der Lötkolben von Männern konstruktiv verwendet wird, beschreibt Ullman einen destruktiven Prozess – und das voller Lust:

SPRECHERIN ZITATE ELLEN ULLMAN 02

„Ich habe schlechte Laune, also nehme ich meine Computer auseinander. Wenn ich Dichter wäre, würde ich mich betrinken und die Menschen anschreien, die ich liebe. So wie es ist, weide ich meine Maschinen aus. Meine Computer sind nicht kaputt, aber in Augenblicken wie diesem gefällt mir der Anblick der freiliegenden empfindlichen Leiterplatten. Vor mehreren Stunden beschloss ich in einem Anfall von Rastlosigkeit, die Betaversion eines neuen Betriebssystems zu installieren. Dann schienen plötzlich Probleme mit einigen der internen Komponenten aufzutreten. Also baute ich sie eins nach dem anderen aus. Jetzt liegen sie alle um mich herum verteilt – Platinen, Drähte, Speichermodule, Schrauben – alles durcheinander. Zum Testen der Komponenten tue ich etwas, was man niemals tun sollte: Ich schalte die Maschine ohne Gerätedeckel ein. Bevor ich etwas anfasse, müsste ich mich von statischer Elektrizität befreien, aber ich schlurfe auf dem Teppich herum, packe Sachen mit beiden Händen an und bringe Metall mit Metall in Berührung. Ich bin mir der Gefährlichkeit meiner Stimmung bewusst, sie ist rücksichtslos und aufmüpfig, als könnte ich die Gesetze der Physik

[25] Burckhardt [1] S. 156

ungestraft übertreten. Kaputte Maschinen verschaffen ein perverses Wohlbefinden. Selbst als es noch schlimmer wird – eine Zeitlang klappt nicht einmal der Selbsttest beim Einschalten –, freue ich mich über die bösartige Stimmung, die ich bei meinem Gefummel empfinde.“26

Musik 2’18 „Computer No. 9” von Andy Fisher
Komponist/Texter: Carl Birth, Christian Dornaus

S2-15 Grace Hopper (historischer O-Ton) 0’31
I was a college professor teaching mathematics. At that time teaching mathematics was a classified occupation. And then I faced a second obstacle: I was underweight. (Gelächter) The Navy considers that I should weight 140 pounds. (Gelächter) All these years I’ve been trying to explain to the Navy that I’m lean and tough and I don’t need to weight 140 pounds. I weight 105 pounds.

Overvoice ad libitum:
Ich war Lehrerin am College und lehrte Mathematik. Zu dieser Zeit galt man als unabkömmlich, wenn man Mathematik unterrichtete. Dann begegnete mir ein zweites Hindernis: Ich war untergewichtig. Die Navy meinte, dass ich 140 Pfund wiegen müsse. All die Jahre versuchte ich, der Navy zu erklären, das ich dünn und robust sei und keine 140 Pfund wiegen müsse. Ich wog 105 Pfund.

ERZÄHLERIN 17
Als es Grace Hopper 1944 irgendwie doch ohne das vorgeschriebene Mindestgewicht in die US-Navy geschafft hatte – vielleicht weil ihr Großvater schon Admiral gewesen war –, drückte ihr Vorgesetzter in Harvard ihr zur Einstimmung auf das neue Arbeitsfeld ein Buch in die Hand. Es waren die Memoiren von Charles Babbage.

ZITATSPRECHER 11
„Obwohl ein großartiger Mathematiker, hat er einer bescheidenen Abschlussnote wegen keine Aussicht auf eine Professur. Gottlob hat er vermögend geheiratet, ist also frei, seinem Traum nachzugehen: der Konstruktion seiner Rechenmaschine.“27

TECHNISCHE STIMME 11
Source: Martin Burckhardt.

[26] Ullman S. 68
[27] Burckhardt [1] S. 48f.

ZITATSPRECHER 12
„Immerhin gelingt es ihm, im Jahr 1822 einen ersten Prototypen seiner Maschine fertigzustellen. Die Regierungsvertreter, denen er sie präsentiert, sind so beeindruckt, dass sie ihm eine größere Geldsumme zur Verfügung stellen. (...) War die Konstruktion der Maschine ursprünglich auf drei Jahre angesetzt, zieht sich der Bau in die Länge. Da das Projekt gleich mehrere Regierungen überlebt, wird es seiner Kosten wegen angefeindet, mehrfach begutachtet – und am Ende, nach 19 Jahren Bauzeit, eingestellt. Das jedoch ficht Babbage nicht an. In seinem Kopf hat sich die Maschine längst zu einem neuen Projekt verwandelt, zur großen ‚Analytischen Maschine', die viel komplexer und aufregender scheint als die ursprüngliche Rechenmaschine.

ERZÄHLERIN
Oder, um es mit den Worten der jungen Ada Lovelace zu sagen, die Babbage 17-jährig kennenlernt und sogleich zur glühenden Anhängerin seiner Kopfgeburt wird:"28

SPRECHERIN ZITATE ADA LOVELACE 01
„Die Analytische Maschine webt algebraische Muster, wie der Jacquard'sche Webstuhl Blumen und Blätter webt"."29

ERZÄHLERIN
Der Philosoph und Programmierer Martin Burckhardt beschreibt das Verhältnis von Lovelace und Babbage so:

S2-16 Burckhardt 0'34
Im Grunde genommen war es eine kurze Episode, deshalb ist diese Heldenverehrung der Ada gegenüber sagen wir mal einigermaßen sonderbar! Es waren zwei Jahre. Es gab einen italienischen Mathematiker, Menabrea, der hat auch über Maschinen nachgedacht, und sie hat – das ist ihr einziger Beitrag –, sie hat einen Text aus dem Italienischen übersetzt und mit Anmerkungen versehen. Die Anmerkungen waren doppelt so lang wie der ursprüngliche Text. Der hat 15 Seiten, 30 Seiten Anmerkung. Und in diesen 30 Seiten Anmerkung entfaltet sie gewissermaßen so eine kosmologische neue Sicht, was eigentlich so eine Maschine können … sie war wirklich eher so quasi das Public-Relation-Genie!

ERZÄHLERIN 18
Das klingt männlich arrogant. War Ada Lovelace, der sprunghafte Geist an der Seite von Charles Babbage, nicht lange vor Grace Hopper die entscheidende Frau in der Geschichte der Programmiersprachen? Oder war sie doch nur ein PR-Talent, das tat,

[28] a.a.O.
[29] a.a.O.

was Frauen nach Meinung von Männern am häufigsten tun: heiße Luft verbreiten? Walter Isaacson urteilt ähnlich wie Martin Burckhardt:

ZITATSPRECHER 13
„Sie wollte partout für Babbage Öffentlichkeitsarbeit betreiben und als seine Partnerin um Unterstützung für den Bau der Analytischen Maschine werben.“30

SPRECHERIN ZITATE ADA LOVELACE 02
„Ich brenne förmlich darauf, mich mit Ihnen zu unterhalten. Ich werde Ihnen einen Hinweis auf unser Gesprächsthema geben. Haben Sie je daran gedacht, dass mein Verstand Ihren Absichten und Plänen in absehbarer Zukunft zu Diensten sein könnte? Falls dem so ist, falls Sie mich jemals für würdig oder fähig erachten, Ihnen auch nur von geringem Nutzen zu sein, betrachten Sie meinen Verstand getrost als den Ihrigen.“31

ERZÄHLERIN
… schrieb sie Anfang 1841.

S2-17 Burckhardt 0‘24
Sie hat diesen Genieverdacht, weil sie mit 25 dann doch mitbekommen hat, dass sie die Tochter von Byron ist, und das Leben war irgendwie zu langweilig. Auch die Kinder waren zu langweilig, der Ehemann war zu langweilig. Und was hat sie gemacht? Hunderennen. Ja? Und dann hat sie Gebärmutterhalskrebs bekommen relativ früh, ist dann gestorben. Aber immer mit dem Bewusstsein eigentlich, mein Gehirn wäre sozusagen die Zukunft. Das war das Leitmotiv.

ERZÄHLERIN 19
Mit diesem Selbstbewusstsein kam Charles Babbage zunächst gut zurecht, er besaß ja selbst nicht wenig davon. Die klassische Arbeitsteilung – er Bastler, sie Algorithmentüftlerin –, konnte ihm nur nutzen, denn sein gigantischer Maschinentraum forderte den ganzen Mann und war seiner Zeit weit voraus, wie wir bei Martin Burckhardt lesen:

ZITATSPRECHER 14
„Strenggenommen war Babbages Maschine ein Verbund aus einer zentralen Recheneinheit und einer angeschlossenen Geräteperipherie. Dabei waren als Ausgabegeräte ein Drucker, ein Kurvenplotter und eine Glocke eingeplant, die, wie der Beep unserer heutigen Computer, den Nutzer in Kenntnis setzen sollte, wenn ein

[30] Isaacson S. 43
[31] a.a.O.

Prozess beendet sei. Die Ergebnisse wiederum sollten in Lochkarten geschrieben oder in Metallplatten gestanzt werden. Die Rechnereinheit enthielt eine Mühle genannte Prozessoreinheit sowie einen Speicher, der etwa 10.000 Wörter enthalten sollte – in unserer heutigen Metrik 12 Kilobyte an Daten. Abgesehen davon, dass diese Mühle die vier Grundrechenarten beherrschen sollte, sollte sie Programmanweisungen folgen, die auf Lochkarten verzeichnet waren.«32

S2-18 Burckhardt 0‘23
Die erste Maschine hat er gebaut, hat 20.000 Astronomen eingespart … also das hat er schon vor Augen geführt! Hat Geld vom Parlament bekommen, hat dann seine nächste Maschine gebaut, und ist dann in einen lebenslangen Kampf mit Werkzeugmachern eingetreten, weil es zu der Zeit noch keine normierte Schraube gab! Man kann sich vorstellen, 50.000 Einzelteile, ja? (lacht) Ohne Normierung, ohne DIN A „sowieso Schrauben“ – absoluter Horror!

ERZÄHLERIN 20
Daraus folgten Verzögerungen, Rückschläge, Geldnot. Babbage begann, sich mit der Politik über die weitere Finanzierung zu streiten und erwartete, dass sich Lady Ada Lovelace als Angehörige der Oberschicht – und gemäß ihren eigenen Versprechungen – für seine Sache einsetzen werde. Vor diesen Karren wollte sie sich nun doch nicht spannen lassen. Der Ton zwischen beiden wurde rauer. In einem Brief an ihre Mutter berichtet Lovelace:

SPRECHERIN ZITATE ADA LOVELACE 04
„Ich wurde auf überaus verblüffende Weise durch das ungebührliche Betragen eines Mr. Babbage überrumpelt & genötigt ... Ich bedaure es sehr, dass ich zu dem Schluss gelangen muss, dass er eine der unpraktischsten, selbstsüchtigsten und maßlosesten Personen ist, mit denen man nur zu tun haben kann ... Ich gab Babbage umgehend zu verstehen, dass keine Macht der Welt mich dazu bewegen könne, mich in den Dienst seiner Streitigkeiten zu stellen oder mich in irgendeiner Weise zu seinem Sprachrohr zu machen ... Er schnaubte vor Wut; ich gab mich unerschütterlich und ungerührt.“33

ERZÄHLERIN 21
Ob die Parteinahme von Lady Lovelace etwas bewirkt hätte, sei dahingestellt. Babbages Analytical Engine blieb ein großartiges Manifest des Geistes, der an der Materie scheitert. Das Ende lesen wir bei Walter Isaacson:

[32] Burckhardt [2], S. 239
[33] zit. nach Isaacson S. 50

ZITATSPRECHER 16
„Babbage (…) starb verarmt. Was Lady Lovelace betrifft, so veröffentlichte sie keine weiteren wissenschaftlichen Arbeiten. Ihr Leben geriet zu einer Abwärtsspirale, sie wurde spielsüchtig und cannabisabhängig. Irgendwann hatte sie eine Affäre mit einem ihrer Spielpartner, der sie anschließend erpresste und dazu zwang, den Familienschmuck zu verpfänden. (…) Als sie 1852 im Alter von 36 Jahren starb, wurde sie getreu ihrem letzten Willen auf dem Land beigesetzt: in Nottinghamshire in der Gruft ihres Vaters, den sie nie gekannt hatte und der im selben Alter gestorben war wie sie. (…) Im Laufe der Zeit wurde Ada Lovelace zur Computerpionierin und Ikone des Feminismus. Das amerikanische Verteidigungsministerium hat beispielsweise eine Programmierhochsprache ihr zu Ehren Ada genannt. (…) Fakt ist, dass Adas Beitrag fundiert und inspirierend ist.“34

Musik 1’01 „Character IV” von Interpret: Zakarya / Yves Weyh

S2-19 Passig 1‘36
Ich hab vor ner Weile mal … ich weiß gar nicht mehr, wie’s dazu gekommen ist. Ich glaube, ich hab mich über COBOL lustig gemacht!

ERZÄHLERIN
Kathrin Passig – Programmiererin und Autorin:

S2-19 Passig 1‘36
Und dann meinte jemand, dass COBOL-Programmierer händeringend gesucht werden. Und zwar, weil es immer noch Software gibt aus den 60er-Jahren, die ganz unten in Banken und Flugbuchungssystemen – also den Systemen, die als erste auf Computer umgestellt haben – auf der alleruntersten Ebene noch läuft. Oder auf der sagen wir mal zweituntersten vielleicht. Und die hat man einfach nicht ausgetauscht. Und dann hab ich gelacht und mir gedacht: „Ach, diese konservativen Leute, die sich einfach nicht durchringen konnten, diese Software auszutauschen!“ Aber dann ist mir klar geworden, das hat natürlich gute Gründe, diese Software läuft … also wenn man was seit 50 Jahren oder so halbwegs reibungslos läuft, dann ist es sehr schlau, es nicht auszutauschen gegen irgendwas, wo überhaupt nicht gesagt ist, dass es auch nur bis nächsten Montag funktionieren wird. Und die ganzen ursprünglichen Programmierer sind natürlich jetzt in Ruhestand, und man kann die zwar noch überreden, ab und zu mal aus ihrem Ruhestand zurückzukommen und irgendwelche Änderungen vorzunehmen. Aber man kann sie nicht rumkommandieren. Die machen das nur zum Spaß. Und wenn sie keine Lust haben oder man ihnen dumm kommt oder zu wenig bezahlt, dann gehen sie wieder. Deshalb sind COBOL-Programmierer händeringend

[34] Isaacson S. 52f.

gesucht. Und ich hab mir dann Stellenanzeigen angeguckt und festgestellt: „Hallo, das ist wirklich so!“ (lacht) Alle wollen welche, es gibt keine. Und sah mich schon in einer lukrativen Zukunft den Rest meines Lebens Geld mit COBOL verdienen. Und dann hab ich mal Code ergoogelt, wie der so aussieht. Dann hab ich den mir ein paar Minuten angeguckt, und dann hab ich die Seiten alle wieder zugemacht und andere Zukunftspläne gemacht.

S2-20 Leitenberger 0‘16
Wenn man so ne Hierarchie mal baut: Assembler, sagt man meistens, ist die Programmiersprache der ersten Generation. FORTRAN und COBOL als spezialisierte Programmiersprachen für einen bestimmten Verwendungszweck sind die zweite Generation. Und was danach kommt – das sind die meisten Programmiersprachen, die es heute gibt – ist die dritte Generation.

TECHNISCHE STIMME 12
Bernd Leitenberger. Programmierer, Fachbuchautor. Ostfildern.

S2-21 Schneider 0‘36
Man hat Anfangs mit diesen Assemblersprachen gearbeitet und dann gemerkt, das ist zu niedriges Niveau, da muss ich viel zu viel Details machen. Und wenn man von einem Anwendergebiet herkommt, etwa von der numerischen Mathematik oder von den kaufmännischen Anwendungen oder Werkzeugmaschinensteuerung! War ganz früh schon dabei! Dann hat man einen Vorrat von Denkmustern, was dort üblich ist, wie man das dort beschreibt. Und wenn man die Programmiersprache so macht, dass sie möglichst nahe an dieser Schreibweise dranbleibt, dann ist das für den Anwender sehr viel einfacher, den Rechner zu programmieren.

TECHNISCHE STIMME 13
Hans Jürgen Schneider, emeritierter Professor für Informatik an der Friedrich-Alexander-Universität Erlangen-Nürnberg.

S2-22 Leitenberger 0‘05
Also ich sag immer: COBOL ist eine Programmiersprache, wo für Betriebswirtschaftler entworfen worden ist.

ERZÄHLERIN 22
COBOL ist Grace Hoppers große Hinterlassenschaft. Unter der Federführung von „amazing Grace“ – so ihr Spitzname – entwickelten Teams in den 1950er-Jahren die „Common Business Oriented Language“, die den Übergang der Rechnertechnologie vom militärischen und wissenschaftlichen Bereich in die Wirtschaft ermöglichte.

Komplizierte mathematische Operationen beherrschten die zeitgleich entwickelten Programmiersprachen der zweiten Generation wie ALGOL und FORTRAN besser …

S2-23 Historischer O-Ton 0’05
FORTRAN represents the most advanced coding systems available today.

ERZÄHLERIN 23
… doch COBOL nahm Rücksicht auf die spezifischen Bedürfnisse von Banken, Versicherungen, Fluggesellschaften und anderen Organisationen, die ihre Datenmengen bis dato mit Karteikarten zu bewältigen suchten.

S2-24 Leitenberger 0’23
Damit kann man sehr gut Daten verarbeiten! Also zum Beispiel wird bei COBOL definiert, wie Daten formatiert werden, damit kann man dann relativ gut Konten führen. Man kann mit dem aber fast gar nicht rechnen, weil zum Beispiel selbst Rechenoperationen als Worte geschrieben werden! Also anstatt „1+2“ schreiben Sie: „1 ad 2“. Oder „five multiply by three“.

ERZÄHLERIN 24
Darin spiegelt sich wohl bis heute Grace Hoppers Mission, den Umgang mit der Rechenmaschine sprachnäher, menschlicher, allgemeinverständlicher zu machen. Womit sie allerdings nicht gerechnet hatte – niemand aus ihrer Generation tat das –, war das hohe Alter, das Programmiersprachen einst erreichen würden. Zehn Jahre, zwanzig Jahre allenfalls vermochte man sich vorzustellen – dann spätestens würde sich die Hardware derart radikal verändert haben, dass alte Programmiersprachen darauf nicht mehr laufen könnten. Die Softwareentwickler der ersten Generation irrten allerdings gründlich, wie Ellen Ullman schreibt:

SPRECHERIN ZITATE ELLEN ULLMAN 02
„Ich arbeitete einmal an einem Mainframe-Computersystem, wo die Auflistung meines COBOL-Programms auf Endlos-Papier so hoch angesehen wurde wie ein Mensch. Mein Programm war sechzehn Jahre alt, als ich es erbte. Nach den Bibliothekseinträgen hatten vor mir sechsundneunzig Programmierer daran gearbeitet. Ich verbrachte ein Jahr damit, alle Unterroutinen und Servicemodule zu durchwandern, aber da waren immer noch geheimnisvolle Orte, an die ich nicht zu rühren wagte. Dieses System hatte Fehler, die seit zehn Jahren niemand in Ordnung bringen konnte. Es gab Bereiche, wo das Hinzufügen einer einzigen Code-Zeile merkwürdige Resultate erzeugte, von Programmierern ‚Nebenwirkungen‘ genannt, Fehler, die nicht direkt aus dem hinzugefügten Code entstehen, sondern aus einer späteren, unbekannten Permutation weiter unten im Prozess. Mein Programm näherte sich dem Ende seines ‚Lebenszyklus‘. Es war dem Tod sehr nahe. Dennoch konnte das System

nicht weggeworfen werden. Wenn ein Computersystem einmal alt geworden ist, versteht es keiner mehr vollständig. Ein System, das aus alter, schrottreifer Technik besteht, wird paradoxerweise wertvoll. Man lässt es weiterlaufen, jedoch wie in einer Samtschatulle: Öffne es vorsichtig und schau es dir an, aber berühren verboten.“35

ERZÄHLERIN 25
Als Ellen Ullman dies Ende der 1990er-Jahre schrieb, war COBOL noch die in der Wirtschaft am weitesten verbreitete Programmiersprache, obwohl sie nicht nur viele Fehler und Undurchsichtigkeiten enthielt, sondern auch einen fatalen Mangel aufwies: Sie konnte das Jahr 2000 nicht als Datensatz verarbeiten, da Jahreszahlen aus Speicherplatzgründen grundsätzlich nur zweistellig abgespeichert wurden. Bekannt wurde dies – mit einem gewaltigen hysterischen Rauschen in der Öffentlichkeit – als „Jahr-2000-Problem“ oder „Millenium-Bug. Man erwartete gravierende Probleme in allen möglichen elektronischen Systemen. Tatsächlich passierte relativ wenig. In einer Wikipedia-Aufzählung kann man von nicht funktionierenden Fahrkartenentwertern in zwei australischen Bundesstaaten lesen, von rund 150 ausgefallenen Spielautomaten an einer US-Pferderennbahn und 170 fälschlich auf den 4. Januar 1900 datierte Gerichtsvorladungen in Südkorea. Dass der Weltuntergang ausblieb, hatte allerdings zuvor Software-Wartungskosten von mehreren hundert Milliarden US-Dollar erzeugt und – nach glimpflicher Bewältigung des Jahrtausendwechsels – COBOL damit für weitere Jahrhunderte vor dem Untergang bewahrt. Denn der Code erlaubt nun Datumsangaben bis zum 31.12.9999. Auch andere Altsprachen sind durchaus noch am Leben, wie Bernd Leitenberger berichtet:

S2-25 Leitenberger 0‘28
Also so Klimamodelle oder die Wettervorhersage, aber auch zum Beispiel eine Carcrashsimulation, das alles wird mit FORTRAN immer noch gemacht. Die grundlegenden Elementarroutinen dafür, die wurden alle irgendwann mal in den 50er-Jahren bis 60er-Jahren geschrieben und die werden bis heute benutzt. Und was man dann draufbaut, ist eben, dass die Simulation immer feiner wird oder dass die grafische Ausgabe noch besser wird. Aber das Basismodell schmeißt man nicht weg. Und dann wechselt man auch nicht die Programmiersprache. Weil sonst müsste man alles neu schreiben.

ERZÄHLERIN
Der Informatiker Hans Jürgen Schneider ergänzt:

[35] Ullman S. 116

S2-26 Schneider 0‘32
Da gibt es übrigens ein sehr interessantes Zitat, das leider nirgends aufgeschrieben ist. Das war eine Konferenz in München über Programmiersprachen. In der Diskussion – also die Vorträge sind aufgeschrieben – in der Diskussion ist der William Wulf gefragt worden nach seinem Vortrag, wie denn seiner Meinung nach die Programmiersprache aussähe im Jahr 2000? Das war also in den 70er … Mitte der 70er-Jahre war das. Da sagte er, er weiß nicht, wie die aussehen wird, aber er weiß, dass sie FORTRAN heißt!

ERZÄHLERIN 26
Ein Paradox … oder das Eingeständnis eigenen Scheiterns. Denn zu diesem Zeitpunkt hatte der amerikanische Informatiker William Wulff selbst eine neue Programmiersprache entworfen, BLISS, die heute keiner mehr kennt. Genauso wie die Kinderprogrammiersprache LOGO von Daniel Hillis längst aus allen Curricula wieder entschwunden ist. Dagegen besitzen etliche Urprogrammiersprachen ein scheinbar ewiges Leben, obwohl ihre Schöpfer bei der Arbeit keinerlei Gedanken an die Zukunft verschwendeten.

S2-27 Bernd Leitenberger 0’13
Sie können ja nicht irgendwo ein System entwerfen, das für die Zukunft ist, weil Sie erstens nicht wissen, wie die Zukunft wird. Und zweitens ein zweitens ein solches System, das auf Zukunft entworfen ist, wird viel komplizierter sein als das, was Sie derzeit brauchen. Das bezahlt Ihnen ja auch keiner.

ERZÄHLERIN 27
Und das Paradox wird laut Ellen Ullman noch größer: Je älter fundamentale und in großen Firmen implementierte Software ist, desto mehr trifft die Kategorie der Zukunftsfähigkeit auf sie gar nicht mehr zu, weil sie zum unhintergehbaren Tatbestand geworden ist.

SPRECHERIN ZITATE ELLEN ULLMAN 03
„Der Wert eines alten Systems ist axiomatisch. (…) Im Lauf der Jahre und mit dem Kommen und Gehen von ungezählten Programmierern und Analytikern nimmt das System ein Eigenleben an. Es läuft. Das ist seine Daseinsberechtigung, es leistet nützliche Arbeit. Egal wie schlecht, egal wie fehlerhaft, egal wie überflüssig – es läuft.“36

Musik 1’05 : „Ballet of the chicks in their shells” von Isao Tomita
Komponist/Texter: Modest Mussorgskij, Isao Tomita

[36] ebd. S. 117

S2-28 Schneider 0‘12
Es gab mal vor vielen Jahren im Internet, als ich das angefangen habe mit der Vorlesung, eine Internetseite, auf der waren 8.000 Sprachen! Aber die ist nicht mehr weitergepflegt worden, und inzwischen ist sie verschwunden.

ERZÄHLERIN 28
Seit mehr als 30 Jahren hält Hans Jürgen Schneider in Erlangen jedes Semester eine Vorlesung zur Geschichte der Programmiersprachen. In einer geschichtsvergessenen Disziplin wie der Informatik stellt das den Lehrenden vor ein Ausdauerproblem ganz eigener Art:

S2-29 Schneider 0‘13
Interessant ist, dass in der ersten Woche – oder vielleicht auch noch in der zweiten – eine größere Zahl Studenten kommt, so um die 30, 35. Und nach drei Wochen bleiben dann fünf oder sechs übrig.

ERZÄHLERIN 29
Dass bei solch überbordendem Interesse an den historischen Grundlagen des eigenen Faches nicht mal genau bekannt ist, wie weit sich das Fach eigentlich ausdehnt, verwundert da kaum noch. Exakte Daten über die tatsächliche Anzahl aller weltweit je erfundenen Programmiersprachen fehlen.

SPRECHERIN ZITATE WEIBLICH ALLGEMEIN 02
„Es gibt, je nach Zählweise, zwischen 500 und über 8.000 Programmiersprachen.“[37]

ERZÄHLERIN 30
… sagt auch Kathrin Passig zusammen mit ihrem Co-Autor Johannes Jander im Handbuch „Weniger schlecht programmieren“ – vermutlich in Bezug auf dieselbe Quelle im Internet. Für Hans Jürgen Schneider ist die Unterscheidung zwischen Alt, Neu und Abspaltung nicht neu:

S2-30 Schneider 0‘13
Es gibt sehr viele Programmiersprachen, die sich als neue Programmiersprache empfinden oder von ihren Erfindern entsprechend propagiert werden. Aber im Grunde genommen sind das Dialekte von solchen, die schon waren!

[37] Johannes Jander / Kathrin Passig: „Weniger schlecht programmieren“. O'Reilly 2013. (eBook ohne Seitenzahlen)

ERZÄHLERIN 31
Somit dürften sich die tatsächlich eigenständigen Programmiersprachen im niedrigen dreistelligen Bereich bewegen. Zwischen ihnen – der Tür zur Maschine – und Anwenderprogramm – dem Fenster zur Welt – sitzt noch das Betriebssystem. Jedes Mal beim Hochfahren des Computers macht es sich zeitraubend bemerkbar.

S2-31 Schneider 0‘34
Das Betriebssystem ist eigentlich ne Schnittstelle zwischen der Hardware und den ganzen Programmiersprachen und Anwendungssystemen. Ohne Betriebssystem können Sie einen Rechner überhaupt nicht benutzen! Selbst diese alte Zuse-22, mit der ich angefangen habe zu arbeiten, die hatte so etwas wie ein rudimentäres Betriebssystem. Das waren so ungefähr 2.000 Speicherplätze mit Befehlen. Man hat dann also eine Möglichkeit zu sagen: „Lies mir mal das Programm ein!“ Oder: „Lies Daten ein!“ Die ganze Verwaltung der externen Speicher wie Platte oder so Sachen, das regelt alles das Betriebssystem.

ERZÄHLERIN 32
Am weitesten verbreitet ist UNIX, auf dem zum Beispiel auch die Apple-Betriebssysteme für Computer, Handys und Tablets basieren. Die UNIX zugrundeliegende Programmiersprache C++ weist dabei einen Abstraktionsgrad auf, der selbst Profis wie Ellen Ullman verstört:

SPRECHERIN ZITATE ELLEN ULLMAN 04
„Ich schielte auf den Code. Es war die merkwürdige, hässliche Syntax von C++. Mit ihrer Vorgängerin C bin ich technisch aufgewachsen, konnte sie schreiben und lesen wie Englisch, sie korrigieren, wenn man sie mir nur am Telefon vorlas, aber hier war es, als wäre ich ein Spanier, der versucht Portugiesisch zu lesen. Ja, die Wurzeln waren dieselben, ja, das hier sind dieselben Wörter, doch woher sind all die komischen Buchstaben gekommen? Wie ist aus dem Wort das hier geworden? Diese Sprache kam eindeutig von der anderen Seite einer politischen und kulturellen Grenze, aus einem neuen und unbekannten Land, in dem ich mich gerade eben verständlich machen konnte.“[38]

ZITATSPRECHER 21
„Befehlszeilen in UNIX lesen sich manchmal, als habe jemand ein eingerolltes Gürteltier über die Tastatur gewälzt.“[39]

[38] Ullman S. 111f.
[39] in „Technology Review“ 6/2019, S. 60

ERZÄHLERIN 33
… spottet der Schriftsteller Peter Glaser, der wie Kathrin Passig ebenfalls literarische mit digitalen Welten zu verbinden vermag. Dass sich solches Grenzgängertum zwischen Sprache und Programmiersprache überhaupt bilden konnte, liegt am Urknall unserer heutigen digitalen Welt: der Entwicklung des Mikroprozessors Anfang der 1970er-Jahre – und der Erfindung einer primitiven, von 14-Jährigen beherrschbaren Programmiersprache namens …

TECHNISCHE STIMME 14
Beginner's All purpose Symbolic Instruction Code.

ERZÄHLERIN 34
… kurz BASIC, entwickelt am Dartmouth College in New Hampshire, USA. Hans Jürgen Schneider erläutert:

S2-32 Schneider 0'37
Das ist historisch sehr interessant, da staunen auch meine Studenten immer! Die Idee hinter dieser BASIC-Entwicklung war, dieses College hatte damals höchstens 20 Prozent Studenten, die was mit Technik zu tun hatten und Computern. Der ganze Rest kaufmännisch, Manager, zukünftige Manager! Und die beiden Autoren Kurtz und Kemeny hatten also die Idee, ein System auf'm Rechner zu installieren, das diesen zukünftigen Managern, die über Investitionen von Computern entscheiden müssen, mal klar zu machen, wie so was überhaupt funktioniert!

ERZÄHLERIN 35
Strenggenommen entstand BASIC also als PR-Maßnahme von Informatikern, um ihren Arbeitsbereich langfristig ökonomisch abzusichern.

S2-33 Schneider 0'26
Auch die beiden, die das Basic entwickelt haben, konnten sich nicht vorstellen, dass auf Dauer Manager mit dem Computer arbeiten! Aber drüber entscheiden, was gekauft wird! Und deshalb sollten denen ein sehr einfaches System, das sie also nicht abschreckt, geschaffen werden, mit dem sie einmal einen Kurs machen können. Der wurde sogar integriert damals in die normalen Vorlesungen. Und naja, weil sie's so einfach gemacht haben, hat das sich natürlich sehr weit verbreitet.

Musik „Hanging upside down" von Balanescu Quartet
Komponist/Texter: David Byrne, Angel Fernandez

TECHNISCHE STIMME 15 über Musik
interrupt Lange Nacht goto Lange Nacht

ERZÄHLERIN 36 über Musik
Über BASIC, den goto-Befehl und dem daraus resultierenden Spaghetticode reden wir in der dritten Stunde der langen Nacht, in der wir auch etwas über ternäre Logik erfahren werden. Denn 0 und 1 sind der Weisheit nur vorletzter Schluss.

Musik

3. Stunde

„Von der binären Verschwendung des Westens und der ternären Askese des Ostens“

Gesprächspartner dieser Stunde
Dr. Martin Burckhardt, Philosoph, Programmierer (Berlin)
Peter Fuß, Deutsches Museum (München)
Francis Hunger, Medienkünstler, Programmierer (Leipzig)
Christoph Kappes, Programmierer, Jurist (München)
Fiona Krakenbürger, Techniksoziologin (Berlin)
Bernd Leitenberger, Programmierer, Buchautor (Ostfildern)
Kathrin Passig, Programmiererin, Autorin (Berlin)
Prof. Dr. Hans Jürgen Schneider, Informatiker (Erlangen)
Prof. Dr. Jochen Ziegenbalg, Informatiker (Karlsruhe/Berlin)

Musik 3’26 „Biking is better” von Wintergatan
Komponist/Texter: Martin Molin

S3-01 Schneider 0‘16
In den 60er-Jahren gab’s die berühmte goto-Kontroverse. Dann hieß es, die Qualität eines Programms ist umgekehrt proportional zur Anzahl der Goto’s. Und dann gab es Sprachentwicklungen, die gesagt haben, wir machen jetzt Programmiersprachen, in denen gibt’s kein goto mehr.

ERZÄHLERIN 01
Erklärt der Informatiker Hans Jürgen Schneider. Wir sind zurück mit der Langen Nacht der Programmiersprachen. Die dritte Stunde bricht an: „Von der binären Verschwendung des Westens und der ternären Askese des Ostens“
Schneiders Kollege Bernd Leitenberger erklärt weiter:

S3-02 Leitenberger 0‘37
Goto sprang einfach ohne irgendeine Prüfung an eine andere Zeile. Diese Zeile hat ne Zeilennummer bekommen. Und dann mussten Sie eben im Quelltext gucken: Stand da eben „goto 100“? Dann mussten Sie gucken, was steht in der Zeile 100? Der springende Punkt war: Mit dieser Anweisung goto konnte man eigentlich überall rumspringen, vor und zurück! Man konnte aber auch vom Hauptprogramm mitten in ein Unterprogramm reinspringen oder aus dem Unterprogramm wieder ins Hauptprogramm rein, was normalerweise bei höheren Programmiersprachen völlig

verboten ist. Deshalb hieß das Ganze „Spaghetticode“, weil im Prinzip musste man dann den Code entwirren, wie wenn man anfangen würde, dann jetzt Spaghettis da auf nem Teller da hinten auszutrennen, dass man also nicht mehr alle verschlungen übereinander hat.

TECHNISCHE STIMME 01
goto „Weniger schlecht programmieren“ von Johannes Jander und Kathrin Passig.

ZITATSPRECHER 01
„Als ich ungefähr acht war, brachte mein Vater eines Abends einen der ersten Homecomputer, einen ZX81, mit nach Hause.“40

TECHNISCHE STIMME 02
read subroutine „Englisch lernen mit dem ZX81“ von Jan Bölsche.

ZITATSPRECHER 02
„Ein schwarzes, türstopperartiges Ding mit einer kaum benutzbaren Folientastatur, die aber den für mich unschätzbaren Vorteil besaß, dass auf ihr sämtliche BASIC-Befehle aufgedruckt waren. So konnte ich trotz kompletter Unkenntnis der Bedeutung von Wörtern wie print, goto und next einfach ausprobieren, was passiert, wenn man diese Befehle in ein Programm schreibt und es dann startet. Während mein Vater bald das Interesse am Programmieren verlor, verbrachte ich jede freie Minute mit dieser vermutlich ineffizientesten Methode des Programmierenlernens durch Stochern im Dunkeln. Nach ein paar Wochen zog ich dann endlich das englischsprachige Anleitungsbuch als Informationsquelle hinzu. (…) Als es in der sechsten Klasse dann endlich losging mit dem Englischunterricht, war mein Wortschatz bereits um die Schlüsselwörter der Programmiersprache C angewachsen. Kurze Zeit später konnte ich dann endlich auch Sätze sagen wie: ‚My name is Jan. My pen is in my pencil-case. My pencil-case is black.‘”41

S3-03 Leitenberger 0‘15
Wobei man sagen muss: Diese ganzen Heimcomputerprogramme, die ja immer mit einem BASIC-Interpreter gekommen sind – das Erste, wenn Sie den eingeschaltet haben, war so ein BASIC-Interpreter und dann „ready“ mit nem Curser –, die wurden ja von den meisten Leuten nicht gekauft, damit die BASIC programmieren! Die haben die meistens gekauft um Spiele zu spielen.

---

[40] zit. nach Jander/Passig o.S.

[41] a.a.O.

ERZÄHLERIN 02
Die Vermutung von Bernd Leitenberger stimmt sicherlich, dennoch blieben ab den späten 1970er-Jahren genügend Leute übrig, die sich durch die gerade noch aushaltbare Komplexität von BASIC dazu verleiten ließen, als Laien – und das oft schon als Jugendliche – außerhalb eines Informatikstudiums eine Programmiersprache zu lernen. Das hatte es zuvor nie gegeben: Algorithmisches Denken wurde massentauglich. Dank des genialischen kommerziellen Spürsinns eines jungen Studenten namens Bill Gates ging die nun auch für Spiel- und Hobbyzwecke erschwingliche Hardware eine fast automatische Verbindung mit BASIC als Basissoftware ein.

ZITATSPRECHER 03
„Aufgrund der Minimalität der Sprache ließen sich leicht BASIC-Interpreter für die mit sehr kleinen Speichern ausgestatteten Mikrocomputer der ersten Generationen entwickeln. In der Folgezeit wurde praktisch jeder Mikrocomputer mit einem BASIC-Dialekt ausgestattet, welcher in etwa der von der Firma Microsoft entwickelten BASIC-Version entsprach. BASIC wurde so fast zur ‚kanonischen' Programmiersprache der Mikrocomputer."[42]

ERZÄHLERIN 03
Das schreibt Jochen Ziegenbalg in seinem Buch „Algorithmen von Hammurapi bis Gödel". Entscheidend für den Erfolg war auch die Art und Weise, wie die Programmiersprache ihre Anweisungen in den 0/1-Code der Maschinensprache übersetzt, nämlich Schritt um Schritt abarbeitend, Jochen Ziegenbalg:

S3-04 Ziegenbalg 0'15
Und die Abarbeitung besteht darin – wenn ein solches BASIC-Programm aus 10 Zeilen besteht – dass jede Zeile genommen wird, übersetzt wird und ausgeführt wird. Dann kommt die nächste Zeile, wird genommen, übersetzt, ausgeführt. Es gibt kein Gesamtmaschinenprogramm.

ERZÄHLERIN 04
Damit wurden die Programme zwar langsam, benötigten aber wenig Speicherplatz – das Hauptkriterium für Massentauglichkeit. Im Gegensatz zu Grace Hoppers „Compiler"-Konzept für die Großrechner, bei dem das gesamte Programm am Stück übertragen wird, ermöglichte dieser „Interpreter"-Ansatz auch bei bescheidener Hardware ausführbare Routinen. Von echten Programmierern bis heute nicht für voll genommen, trat BASIC in der Microsoft-Variante von Bill Gates einen Siegeszug

[42] Ziegenbalg S. 269

sondergleichen an, der allerdings John G. Kemeny erschütterte, einen der Väter des Ur-BASIC. 1987 klagte Kemeny in einem Interview:

ZITATSPRECHER 04
„Sie bekommen zum Beispiel von Microsoft für den IBM-PC nur diese lausige BASIC-Version. Das ärgert mich. (…) Es gibt ein berühmtes Zitat von einem Computerwissenschaftler: Wer seine ersten Programmiererfahrungen mit BASIC macht, handelt sich eine Krankheit ein, von der sich einige Leute nie erholen. Ich muss Ihnen sagen, in Bezug auf Microsoft-BASIC (…) stimme ich dieser Aussage zu. Ich möchte nicht, dass mein Kind in dieser Sprache das Programmieren lernt.“43

S3-05 Passig 0‘09
Also ich hab leider auch viele Jahre lang – und auch öffentlich – immer gesagt „Hausfrauenprogrammiersprache“. Ich möchte mich dafür jetzt genauso öffentlich entschuldigen, wenn’s geht.

ERZÄHLERIN 05
… betont Kathrin Passig, meint aber nicht das mittlerweile in die Jahre gekommene BASIC, sondern die erste Scriptsprache fürs Internet „Personal Home Page“, kurz PHP, mit der ab Mitte der 1990er-Jahre Hunderttausende ihre Webseiten zusammenbauten … auch das eine Form von Programmierung. Das miserable Standing von BASIC – weil laientauglich – übertrug sich nahtlos auf das ebenso laientaugliche PHP. Zu Unrecht:

S3-06 Passig 1‘05
Das war in vieler Hinsicht falsch! Das war zum einen eine komplett unnötige Hausfrauenbeleidigung und zum anderen auch eine unnötige Beleidigung von PHP! Eine Sprache, mit der man natürlich genauso gut wie mit jeder anderen alles machen kann. Man schießt sich ein bisschen leichter ins Knie, aber die Einstiegshürde ist eben auch extrem gering! Das war eben auch der Weg, auf dem ich dahin gekommen bin: Wenn ich will, dass sich auf einer Webseite irgendwas ändert, und ich mach das mit PHP, dann ist das wirklich ein extrem winziger Aufwand, den ich dafür treiben muss. Als ich das das erste Mal gemacht hab, hab ich einfach irgendwo ne Zeile herkopiert und in mein HTML reinkopiert, und zack die Sache lief! Die Einstiegsschwelle ist einfach sehr niedrig und es ist zum einen ungerecht der Sprache das vorzuwerfen, und zum anderen hat es natürlich auch Methode! Also gerade dass die Einstiegshürde so niedrig ist, führt eben dazu, dass Leute, die ihre gutbezahlte und hoch angesehene

[43] zit. nach Bernd Leitenberger „Computergeschichte(n)“ Edition Computer bei BoD 2014 (eBook ohne Seitenzahl)

Programmierdomäne gegen irgendwelche niederen Neulinge verteidigen wolle – gegen Konkurrenz! –, diese Sprache schlechtreden!

ERZÄHLERIN 06
Eine plausible Vermutung für die Mäkelei von Programmierern an bestimmten, als unterkomplex oder „primitiv" verachteten Sprachen. Denn auch in der Welt der Programmierung existieren Adel und Fußvolk.

S3-07 Passig 0'12
Da ganz oben, da stehen Programmiersprachen, mit denen hab ich so wenig zu tun, dass ich glaub ich noch nicht mal mit den Leuten rede, die sich damit auskennen. Ich vermute, es ist irgendsowas wie LISP, also so was, wo Leute dann anerkennend nicken und „Oho, LISP!" sagen.

ERZÄHLERIN 07
„Oho, BASIC!" hat wohl nie einer gesagt – es sei denn, er lobte einen Vorschüler. Im Falle dieser populären Basisprogrammiersprache trat zur internen Verachtung aus der IT-Zunft auch noch externe Kritik hinzu – und zwar aus dem Feuilleton: Ging es nicht um Sprache? Und war Sprache nicht Sache der Philologen, der Hochkultur?

S3-08 Hanno Murena (historischer O-Ton) 0'36
„Ich kann all dat printen, was ich getten kann." Dieser Satz in deutsch-englischem Kauderwelsch ist durchaus logisch, aber dennoch kaum verständlich. Er stammt von einem Lehrer, der in einem 90-Mark-Sprachkursus die vor 25 Jahren die in Amerika entwickelte, vielseitig anwendbare Programmiersprache für Anfänger, BASIC, vermitteln möchte. Und so spricht dieser Lehrer zu seinen Schüler, die allesamt erwachsen sind, in Banken, Büros oder Sekretariaten arbeiten, wie mit dummen Kindern oder mit exotischen Ausländern, denen er sich mühsam verständlich zu machen versucht.

ERZÄHLERIN 08
Hanno Murena, damals Kulturchef der Deutschen Welle, berichtete 1986 im Deutschlandfunk über seine offenkundig griesgrämig stimmenden Erfahrungen mit einem BASIC-Kurs. Schon der Lehrer aus der IT-Branche findet sein Gefallen nicht:

S3-09 Hanno Murena (historischer O-Ton) 0'42
Wenn er von RAM und ROM redet, von print und input, von goto, run und list, so lehrt er nicht etwa Englisch, sondern versucht die Grundbegriffe des BASIC zu erläutern. Das Ganze ist ein Reinfall. Der sicher gut ausgewiesene Computerfreak ist der deutschen Sprache kaum noch mächtig. Seine verstümmelten Sprachreste sind didaktisch nicht mehr verwendbar. Auch wird der Verdacht erhärtet, dass der fast

ausschließliche Umgang mit Computern, der sich ja über formelhafte Programmiersprachen vollzieht, dann zu einer Sprachverarmung führt, wenn die Kommunikation Mensch-Computer auf inhaltsleeren Formelsalat beschränkt bleibt.

TECHNISCHE STIMME 03
goto Johannes Jander und Kathrin Passig. print Grundsatzbemerkung.

SPRECHERIN ZITATE WEIBLICH ALLGEMEIN 01
„Eine Programmiersprache ist in erster Linie eine künstliche Sprache zur Kommunikation von Menschen mit Maschinen. Im Unterschied zu natürlichen Sprachen, die der Kommunikation zwischen Menschen dienen, sind Programmiersprachen vollkommen eindeutig definiert: Es gibt keinerlei Interpretationsspielraum, was ein bestimmtes Sprachkonstrukt bedeutet.“[44]

ERZÄHLERIN 09
Genau das macht Programmiersprachen untauglich für feuilletonistische Betrachtungen wie für ästhetische Sprachkritik. Die Begriffe darin sind allenfalls zu Scheinworten geronnene Zahnräder oder Hebel oder Schalter: Sie lösen nur genau das aus, wofür sie ihrerseits vorgesehen sind. Sie können nicht mehrdeutig, ironisch oder anspielungsreich sein. Aus alltagssprachlicher Sicht ist die verbale Welt der Programmiersprachen öde und unattraktiv, weil sich in ihrem verknappten Englisch weder Poesie noch Witz findet. Meistens jedenfalls.

SPRECHERIN ZITATE WEIBLICH ALLGEMEIN 02
„PHP-Nutzern kann es passieren, dass sie aus heiterem Himmel mit der Nachricht Parse error: syntax error, unexpected T_PAAMAYIM_NEKUDOTAYIM konfrontiert werden.“

TECHNISCHE STIMME 04
source Johannes Jander und Kathrin Passig.

SPRECHERIN ZITATE WEIBLICH ALLGEMEIN 03
„Was aussieht wie eine Warnung vor der unmittelbar bevorstehenden Machtübernahme durch Außerirdische, ist eine hebräische Fehlermeldung, die ursprünglich aus der in Israel entwickelten Zend Engine 0.5 stammt. Paamayim Nekudotayim sind zwei aufeinanderfolgende Doppelpunkte. Die Fehlermeldung hat immerhin den Vorteil großer Eindeutigkeit, ihre Eingabe in Suchmaschinen führt viel

[44] Jander/Passig o.S.

schneller zur Erleuchtung als eine Suche nach ‚double colon'. Machen Sie es den Zend-Entwicklern trotzdem nicht nach."45

Musik 7'50 „Nieder mit IT!" von Thomas Pigor, Benedikt Eichhorn
Komponist/Texter: Thomas Pigor, Benedikt Eichhorn, Ulf Henrich

TECHNISCHE STIMME 05
Es folgt ein kurzes Zitat aus dem Roman „Dunkle Zahlen" von Matthias Sengel.

ZITATSPRECHER 08
„Mit ausgebreiteten Armen verkündet Ptuschkow: ‚Bei diesem Prachtstück handelt es sich um den allerersten elektrischen Ternärrechner der Welt. Und zugleich um den einzigen Ternärrechner, der es je zur Serienreife gebracht hat. Ein Ternärrechner, was heißt das?'"46

S3-10 Francis Hunger 1'07
Die Fragen, die in der Bearbeitung von mathematischen Problemen – und aber auch informatorischen Problemen – auftauchen, sind häufig nicht nur binär, sondern können auch ternär abgebildet werden.

ERZÄHLERIN
So der Leipziger Medienkünstler, Programmierer und Medienwissenschaftler Francis Hunger.

S3-10 Francis Hunger 1'07
Wenn ich klären möchte, ob ein Apfel grün oder rot ist, dann brauche ich im Binären zwei Schritte. Ich muss zuerst feststellen: Ist der grün, ja oder nein? Wenn nicht, nächster Schritt, muss ich feststellen: Ist der rot, ja oder nein? Und dann weiß ich: Okay, er ist entweder keins von beiden, er ist weder grün, noch rot. Oder er ist rot oder er ist grün. Und das mach ich natürlich im Ternären auf ne sehr angenehme Art und Weise, indem klar ist, dass Rot -1 ist und Grün ist +1 und keins von beiden ist 0. In diesem Sinne – jetzt wirklich sehr als Analogie gesprochen, ließen sich da die Programmierstrukturen vereinfachen und der Code sozusagen straffen. Und offensichtlich hat das als Argument ausgereicht, um zu sagen: „Okay, wir machen das mit dem ternären Computer, wir probieren das aus!"

[45] Jander/Passig o.S.
[46] Matthias Senkel „Dunkle Zahlen". Matthes&Seitz 2018, S. 463

ERZÄHLERIN 10
Bei den Vorarbeiten zu einer Perfomance stieß Francis Hunger auf den einzigen je gebauten Ternärrechner der Welt, den Setun.

S3-11 Francis Hunger 0‘39
Am Anfang, als ich das recherchiert habe, dachte ich – und denke, dass denken viele! – ganz amateurhaft: „Wow, drei Zustände: ja, nein, vielleicht!“ Und das wär natürlich ein großartiger Computer gewesen, der in den 60er-Jahren in der Sowjetunion entwickelt worden wäre, ein Computer der „vielleicht“ ausdrücken kann! Es stellte sich dann irgendwann die Ernüchterung ein, dass es eben ein weiterer exakter, logischer Zustand ist. Also wenn ich ne Entscheidungsfrage habe, dann kann ich nicht nur sagen „das oder das“, sondern ich kann eben sagen: „Dieses oder das oder jenes.“

ERZÄHLERIN 11
Da es im Westen kaum etwas darüber zu lesen gab, übersetzte Francis Hunger russische Originaltexte ins Deutsche und Englische. Dieses zweisprachige, 2007 veröffentlichte Buch „Setun – Eine Recherche über den sowjetischen Ternärcomputer“ diente wiederum dem Leipziger Schriftsteller Matthias Senkel als eine der Inspirationsquellen für seinen Computerroman „Dunkle Zahlen“ von 2018. Ganz zum Schluss führt einer von Senkels Protagonisten das rare Stück in einem Computermuseum der Zukunft vor:

ZITATSPRECHER 09
„Ptuschkow lässt der Gruppe genügend Zeit, den Rechner von allen Seiten zu betrachten. Unverkennbar Neunzehnhundertfünfzigerjahre: Der Setun ähnelt einer wuchtigen Schrankwand mit eingebauten Vitrinen. Hinter den Scheiben sind Schaltelemente wie Schmuckgeschirr aufgereiht. Dank seiner kompakten Ausführung hätte der Setun beinah in ein Standardwohnzimmer jener Jahre gepasst, auch wenn das Stahlblechgehäuse mit ausgestanzten Lüftungsschlitzen wenig heimelig wirkt. Die silbrige Konsole vor den Schrankmodulen bringt einige Besucher auf die falsche Fährte, es handle sich um einen frühen Synthesizer; eine Vermutung, zu der insbesondere die handlichen Kippschalter und das Steckpult mit Klinkenkabel verführen. ‚Nein‘, sagt Ptuschkow, ‚der Setun ist weder ein Synthesizer, noch ein digitales Mischpult. Die Sache ist die, meine Damen und Herren: Rein äußerlich kann man ihm seine Einzigartigkeit überhaupt nicht ansehen. Was aber macht diese Einzigartigkeit aus?‘“[47]

[47] Senkel S. 462

S3-12 Francis Hunger 1‘00
Es gibt eine Reihe von 18 Schaltern und diese 18 Schalter belegen 18 Stellen im Speicher. Und jeder der Schalter hat drei Zustände, nämlich Mitte, oben, unten, und das heißt, jeder einzelne Schalter wird umgelegt – meinetwegen wenn er oben steht, ist es eben -1, auf der Mitte 0 und unten +1 –, jeder Schalter wird umgelegt. Jetzt ist für diesen Speicherbereich eine ternäre Zahl zusammengestellt. Die Zahl wiederum besteht eben aus lauter -1, 0 und 1. Die Zahl würde sich dann, wenn man die jetzt ausspricht, meinetwegen eben so lesen: -1, 0, -1, -1, 0, 0, 1 … und so weiter, bis man 18 Stellen hat. Und daneben gibt’s einen Knopf. Und wenn man den Knopf drückt, dann wird das in den Speicher reingeschrieben. Und dann kann man die nächste Zeile programmieren, und so entsteht das Programm.

ZITATSPRECHER 10
„‘Um etwa die Jahreszahl 2021 binär darzustellen, benötigt man elf Zeichen. Um 2021 ternär darzustellen, benötigt man, wie Sie hier sehen können, lediglich acht‘, erklärt Ptuschkow, der mit seinem Handprojektor eine Tabelle an die Wand wirft. (…) ‚Es lassen sich folglich Speicher- und Schaltelemente einsparen, in diesem Fall Dioden und Ferritkerne. Auch die Handhabung negativer Zahlen ist mit dem Ternärsystem viel einfacher ...‘
An dieser Stelle hält Ptuschkow sich zurück, weil er weiß, wie schnell er seine Zuhörer verlieren kann, wenn er ein klein wenig tiefer in die Mathematik oder in die Begriffswelt der elektronischen Gerätetechnik eindringt. (…) Und so fährt der Alte fort: ‚Diese und weitere Eigenschaften machten den Setun zu einem platzsparenden, kostengünstigen und verlässlichen Rechner, der sich zudem leicht bedienen ließ. Mit nur vierundzwanzig Programmierbefehlen konnte er all seine Aufgaben in Wissenschaft und Industrie erfüllen. Doch nun, nachdem ich Ihnen so viele Vorzüge aufgezählt habe, fragen Sie sich gewiss, warum das Ternärsystem nicht längst den Binärstandard abgelöst hat?‘“[48]

S3-13 Francis Hunger 1‘00
Mitte der 50er-Jahre – in der Sowjetunion geben wir mal noch zwei, drei Jahre Verzögerung hinzu, vielleicht in den USA ein bisschen eher –, da ist noch ganz viel offen! Auch für Experimentieren: Wo kann denn die Reise hingehen mit dieser Automatisierung des Rechnens? Deswegen gab’s da auch ne große Offenheit, einfach zu sagen: „Okay, gut, machen wir halt mal ternär!“ Das Bild wandelt sich dann sehr stark in den 60er- und 70er-Jahren, nämlich ab dem Moment, wo eine Rechnergeneration auf die nächste Generation aufbaut, und quasi die funktionierende Technologie fortgesetzt, erweitert, ergänzt wird. Dann hat man immer schon bestehende Infrastrukturen, der Produktion zum Beispiel. Also die Bauelemente, die

[48] Senkel a.a.O.

hergestellt werden, aber auch die bestehenden Infrastrukturen des Rechnens und des Denkens in und mit Computern. Und in dem Fall eben auch erklärt sich darüber, warum das binäre System so stark gewesen ist im Vergleich zu dem ternären.

ERZÄHLERIN 12
Die Kippschalter-Architektur des Setun mag heute vorsintflutlich anmuten und ins Bild einer – freundlich gesagt – robusten Ostblock-Technik passen, doch war sie keineswegs eine Spezialität der Sowjetunion. Auch der wichtigste Mikrocomputer-Bausatz im Westen, mit dem der Siegeszug von Homecomputer und PC im Jahre 1975 begann, wurde noch mit Kippschaltern programmiert. In seinem profunden historischen Abriss „Computergeschichte(n)“ beschreibt Bernd Leitenberger diese legendäre Maschine, mit der die Karriere von Bill Gates und all den anderen Softwarepionieren seiner Generation begann:

ZITATSPRECHER 11
„Der Altair 8800 war in seiner Grundausführung kein Gerät, welches auf Anhieb begeistern konnte. Um Kosten zu sparen (…) und um schnell ein Gerät auf den Markt zu bringen, beschränkte sich Ed Roberts auf das Notwendigste: Der Altair 8800 steckte in einem Klappgehäuse. An der Frontseite konnte er den Status mit LEDs ausgeben. Über diese wurden auch Ergebnisse ‚visualisiert‘. Eingaben machte man durch Kippschalter. Jeder Schalter stand für ein Bit und so musste man acht Stück umlegen, um ein einziges Byte einzugeben. (…) Innerhalb eines Monats erhielt Roberts 4.000 Vorbestellungen. (…) Jeder, der sich für Computer interessierte, wollte einen solchen Rechner haben. (…) Wir wissen heute, dass der Altair nicht der erste PC war. Es gab mindestens zwei Maschinen vor ihm. Aber es war der erste kommerziell erfolgreiche PC.“[49]

ERZÄHLERIN 13
Der erwähnte Ed Roberts, ein ehemaliger US-Airforce-Elektroniker, betrieb eigentlich ein Unternehmen, das Raketenmodell-Bausätze für Bastler herstellte – also Männerspielzeug! Als der 8080-Chip von Intel auf den Markt kam – mit rund 6000 Transistoren der erste vollwertige Mikroprozessor der Welt –, flossen bei Roberts die Kernkompetenzen „Elektronik“ und „Männerspielzeugerfahrung“ zur folgenschweren Idee des Altair 8800 zusammen. Er kaufte leicht fehlerhafte Zweite-Wahl-Chips zu einem stark reduzierten Preis und führte sie dem Sekundärmarkt „Bastler“ zu, der binnen kurzem zu einem Primärmarkt werden sollte. Das allerdings schien Ed Roberts dann nicht mehr zu interessieren. Für sich selbst setzte dieser Ur-Vater des PC ganz andere biographische Prioritäten, fernab der Maschine:

[49] Leitenberger o.S.

ZITATSPRECHER 12
„Von 1977 bis 1984 züchtete er Schweine auf einer Ranch in Georgia. Danach studierte er von 1984 bis 1988 Medizin, als eine neu eingerichtete Universität auch ältere Studenten akzeptierte. Bis zu seinem Tode praktizierte er als Landarzt in einem kleinen Dorf nahe Cochran, Georgia.“50

Musik 1’37 „Colossus” Orchester Roland Kovac
Komponist/Texter: Roland Kovac

S3-14 Francis Hunger 0‘35
Der Nikolaj Petrowitsch Brusencov, der der Hauptentwickler des ternären Setun-Computers war, der hatte Funkmechanik und Elektronik studiert, und in dem Zuge hatte er die Aufgabe, bestimmte Eigenschaften eines Gerätes – ich kann nicht mehr sagen, welches – auszurechnen. Und er war davon derart enerviert, dass er dann, als er von den ersten elektronischen Computern in der Sowjetunion hörte, sehr aufmerksam wurde und angefangen hat, sich genau dafür zu interessieren.

TECHNISCHE STIMME 06
goto Francis Hunger „Setun“. read Aussage von Nikolaj Petrowitsch Brusencov:

ZITATSPRECHER 13
„Wir sollten für die Moskauer Staatliche Universität den Rechner M-2 erhalten, der in Isaak Semenovich Bruks Laboratorium hergestellt worden war. Doch es passierte ein Missgeschick. Bei der Wahl in die Akademie der Wissenschaften der UdSSR sprach sich Sergej L’vovič Sobolev – unser Leiter – nicht für Bruk, sondern für Sergej Alekseevič Lebedev aus. Bruk war verletzt und gab den Rechner nicht heraus. Ich ging zu Sobolev und fragte, womit ich mich denn nun beschäftigen solle. Er antwortete mir: ‚Ach komm, wir bauen unseren eigenen Computer.‘ Das war Ende 1955. Zu dieser Zeit waren Transistoren noch nicht zugänglich, doch es war klar, dass der Rechner nicht auf Vakuumröhren basieren sollte. Röhren haben eine kurze Lebensdauer, und auf Röhren basierte Rechner standen den größten Teil der Zeit still, weil sie ständig repariert werden mussten. (…) Julij Izrailevič Gutenmacher baute den Rechner LEM-1 aus Ferritdioden-Einzelteilen. Mir kam der Gedanke, dass man es versuchen könne, einen Rechner aus diesen Einzelteilen herzustellen, solange keine Transistoren zur Verfügung standen.“51

[50] Leitenberger o.S.
[51] Francis Hunger „Setun“, Institut für Buchkunst Leipzig 2007, S. 152f.

ERZÄHLERIN 14
Der mangelnde Zugang zu amerikanischer Transistor-Hochtechnologie war damals durchaus normal, auch im Westen. Heinz Zemanek von der TU Wien baute zwar schon 1958 den ersten rein auf Halbeitertechnik basierenden Computer in Kontinentaleuropa, musste dafür aber auf eher unpassende Transistoren aus der Hörgeräteproduktion zurückgreifen. Weil diese nur sehr langsam schalteten, hinkte sein Rechner der amerikanischen Konkurrenz mit martialischen Namen wie „Taifun" oder „Wirbelwind" merklich hinterher. Mit Wiener Schmäh gab Zemanek zu Protokoll, für ein „Mailüfterl" werde es bei seinem Rechner schon reichen.

Musikalischer Trenner WH: „HiFi" von Kodexx Sentimental aus „Das digitale Herz"

ERZÄHLERIN 15
Im Westen herrschte nicht lange Mangel, und durch die Weltmarkt-Dominanz der USA wirkte sich das auch auf den ternären Setun des Ostens aus. Nach gut fünf Dutzend Geräten wurde dessen Produktion eingestellt. Einen kurzen Moment lang hatte man allerdings auf internationale Absatzmärkte geschielt:

ZITATSPRECHER 14
„Der Setun kostete 27.500 Rubel mit der gesamten Peripherie. Die Tschechen errechneten, dass sie, wenn sie den Setun zu Weltmarktpreisen verkaufen würden, pro Rechner einen Gewinn von etwa einer halben Million Dollar erwirtschaften könnten."[52]

ERZÄHLERIN 16
Schreibt Francis Hunger in seinem Buch über Setun. Doch der größte Vorteil des ternären Rechners – sich asketisch im Umgang mit der Hardware zu verhalten – zählte angesichts der rasanten weltweiten Fortschritte bei der Transistoren- und später der Chip-Produktion von Woche zu Woche weniger. Askese gilt nur dort als Tugend, wo Knappheit herrscht, und die Low-Tech-Bauart mit Ferritspulen, die elektromagnetische Schaltzustände einnehmen konnten, besaß nur in den 1950er-Jahren den Charme der quasi handwerklichen Rechenmaschinenherstellung:

ZITATSPRECHER 15
„Der Arbeitstag begann mit einer ‚Frühgymnastik': Jeder Mitarbeiter des Laboratoriums, inklusive des Projektleiters, erhielt fünf Ferritkerne mit einem Durchmesser von drei Millimetern. (…) Mithilfe einer einfachen Nadel wurden auf jeden Ferritkern 52 Windungen aufgerollt. Danach wurden sie von Laboranten und

[52] Hunger S. 156

Technikern genutzt, die darauf die Windung für die Stromzuführung (fünf Windungen) und Steuerwicklungen (zwölf Windungen) aufbrachten. Dann montierten sie die Ferritdioden-Zellen auf eine Platine, löteten die Dioden an, überprüften die Einhaltung der Parameter, brachten eine Markierung und das persönliche Zeichen des Kontrolleurs an.“53

TECHNISCHE STIMME 07
Männer löten – Frauen denken.

ERZÄHLERIN 17
Vice versa: Frauen wickelten, während Männer grübelten. Nämlich über die auf 18 Kippschaltern basierende – also 18stellige – dreiwertige Programmierlogik des Setun. Diese unterschied sich zwar vom seit Leibniz etablierten binären Code, eröffnete aber letztlich keinen neuen geistigen Raum etwaiger Unwägbarkeiten wie „unentschieden“, „vielleicht“ oder „zufällig“ – das wäre die Computerrevolution gewesen. In seiner Logik folgte auch der Setun den Regeln der Bool‘schen Algebra. Francis Hunger blickt auf die russische Originalanleitung und erläutert:

S3-15 Francis Hunger 0‘40
Für jede dieser Ziffernfolgen – also das sind jetzt drei Ziffern –, die stehen für einen Befehl. Zum Beispiel „Übertrage den Wert, der sich in diesen 18 Stellen jetzt hier befindet in das Register S“. Und Register S war sozusagen das Hauptrechenregister, so ne Art kleiner Zwischenspeicher. Oder „Gib diesen Wert jetzt auf dem Drucker aus!“ Oder Ähnliches. Das war alles codiert eben in diesem System -1, 0 und 1, und deswegen ist es dann eben entgegen unserer naiven Hoffnung eben doch ein sehr exaktes System, das einfach nur anders rechnet als nur mit 0 und Einsen.

ZITATSPRECHER 16
„Denen, die in der zweiwertigen Logik gedrillt wurden, ist es nicht gegeben, in die dreiwertige einzudringen.“54

ERZÄHLERIN 18
… gab in hohem Alter Nikolaj Petrowitsch Brusencov zu Protokoll und zeigte damit die leichte Verbitterung eines unverstandenen Genies. Als kreative Tätigkeit bringt Programmieren eben eine entsprechende Empfindsamkeit mit sich. Der Mensch an der Maschine ist nicht die Maschine, auch wenn er mit ihr maschinisch spricht. Dazu der Programmierer Bernd Leitenberger:

[53] Hunger S. 109
[54] Hunger S. 159

S3-16 Leitenberger 0‘18
Ein Programm ist ja im Prinzip geistige Arbeit! Manche Leute schreiben gern Gedichte, andere Leute malen gerne, und da wollen sie auch ein schönes Gedicht haben oder ein schönes Bild! Und man kann ein Programm machen, dass es schnell hingerotzt ist. Oder man kann ein Programm machen, das schön ist. Also schön zum Beispiel, dass man es nach einem Monat noch verstehen kann, wenn man’s anguckt.

ZITATSPRECHER 17
„Hab in uraltem Code einen Kommentar von mir an mich gefunden: ‚Wenn du das hier liest, wirst du glauben, es wäre falsch und die gesamte Funktion überarbeiten. Mach das nicht! VERTRAU MIR!‘"[55]

ERZÄHLERIN 19
… spießt ein Programmierer auf Twitter ein Grundproblem aller Software auf: Obwohl sie funktioniert, kann sie selbst für den eigenen Schöpfer nach einiger Zeit unlesbar geworden sein.

S3-17 Kathrin Passig 0‘33
Die einen sagen, wenn man seinen Code vernünftig schreibt, dann ist er auch ohne Kommentar gut verständlich. Weil er dann gleich so angelegt ist, dass man auf mehr oder weniger den ersten Blick sieht, was er tut. Weil die Variablen vernünftige Namen haben, die auch beschreiben, was da jetzt drinsteckt. Dann gibt es Leute, die sagen: „Jede glaubt von sich, dass der eigene Code intuitiv verständlich und selbsterklärend ist, aber es stimmt fast nie.“ Und man sollte dann eher mit der Annahme arbeiten, dass man zu den Leuten gehört, die das nur glauben und trotzdem erklärende Kommentare an den Rand schreiben.

ERZÄHLERIN 20
Eine nützliche Einsicht, die Kathrin Passig bei der eigenen Arbeit einerseits befolgt – andererseits nicht gut genug befolgt:

S3-18 Kathrin Passig 0‘36
Der Code von mir, den ich zu lesen, zu verstehen versucht hab, der war nicht älter als vier Wochen! Und da stand irgendwas drin wie „Man könnte es auch soundso machen, aber das gibt Probleme!“ Und ich hab diese Zeile angeschaut und mir gedacht: „Welche Probleme, welche? Warum hat sie nicht hingeschrieben, welche Probleme man kriegt, wenn man es so macht? Jetzt muss ich das alles noch mal ausprobieren um rauszufinden, was diese Person, die ich vor vier Wochen war, damit gemeint hat!“ Dann hab ich eine zweite Kommentarzeile reingeschrieben, in der ich diese andere

[55] https://twitter.com/markmueller1979

Person verflucht habe, damit ich wenigstens, wenn ich das dritte Mal draufschaue dran denke, dass ich sowas nicht mehr mache. Sondern wenn ich drankommentiere „Verursacht Probleme“ auch reinschreiben muss welche.

S3-19 Martin Burckhardt 0‘25
Es gibt ja die berüchtigten Spaghetti-Artists, ja? Also die ihren Code schreiben, der funktioniert irgendwie auch, der ist nur vollkommen hirnlos, weil der nicht intersubjektiv ist! Der kann niemandem mehr vermittelt werden. Und die wirkliche Kunst der Programmierung besteht eigentlich darin, so weit von dir selbst abzusehen, dich quasi systematisch zu enteignen – als nicht sich auszudrücken, sondern sich zu enteignen – dass ein anderer das, was du tust, benutzen kann.

ERZÄHLERIN 21
Intersubjektivität, die der Philosoph und Programmierer Martin Burckhardt hier ins Spiel bringt, ist das eine – ein Programm schön zu schreiben das andere. Dahinter steht die Frage nach Eleganz im Code.

SPRECHERIN ZITATE ELLEN ULLMAN 01
„Das Projekt beginnt im Kopf des Programmierers mit der Schönheit eines Kristalls.“

TECHNISCHE STIMME 08
goto Ellen Ullman.

SPRECHERIN ZITATE ELLEN ULLMAN 02
„Eine Zeitlang ist die Welt ein ruhiger, mathematischer Ort. Mensch und Maschine vereinen sich in präziser Anmut. Ich habe einmal in meinem Leben Methamphetamine genommen, dieses Speed-High ist der einzige Zustand, der mit dem Gefühl zu Beginn eines Projekts vergleichbar ist. Ja, ich verstehe. Ja, das geht. Ja, wie eindeutig. Oh, ja. Ich verstehe. Dann geschieht etwas. Nach Monaten des Codierens treten die ersten Unregelmäßigkeiten menschlichen Denkens auf. Man schreibt Code, und plötzlich tun sich dunkle, nicht-spezifizierte Zonen auf. All die Seiten sorgsam zusammengestellter Dokumente, und dennoch, zwischen den Sätzen fehlt etwas. (…) Jetzt beginnt eine Phase der Frustration. Der Programmierer wendet sich mit Fragen an den Analytiker, der Analytiker an die Anwender, die Anwender an ihre Manager, die Manager ihrerseits an die Analytiker, der Analytiker an die Programmierer. Es stellt sich heraus, dass einige Dinge einfach noch nicht verstanden sind. (…) Eine lange Liste von Ausnahmesituationen kommt zum Vorschein, Dinge, die, wenn auch sehr selten, dennoch auftreten. Ob man sie programmieren sollte? Ja, natürlich. (…) Details und Ausnahmen häufen sich. Schon bald muss der wunderschöne Kristall neu geschnitten werden. Die eine und die andere hübsche Kante verschwinden. Die gesamte elegante

Struktur verliert den Zusammenhalt. Was anmutig begann, stellt sich schon bald als Durcheinander heraus. Der menschliche Verstand ist schlampig.“56

ERZÄHLERIN
Der Programmierer Bernd Leitenberger kennt das Phänomen:

S3-20 Leitenberger 0‘18
Es gibt sogar solche Wettbewerbe, wo man unlesbare Programme schreibt. Also ich hab immer Informatikunterricht. Ich hab da immer dann mal so ein Programm den Studenten als Quelltext gezeigt und hab sie mal aufgerufen zu raten, was da drauf ist! Also selbst ich komm da nicht drauf! Das ist ein C-Programm, da sehen Sie, wenn Sie den Quelltext sehen, eigentlich bloß wirre Zeichenfolgen, die keinen Sinn machen! Und wenn’s Programm läuft, dann tut es ein Gedicht ausgeben.

TECHNISCHE STIMME 09
“BSG [Buchstabensalatgenerator]”

ERZÄHLERIN 22
Eine solche Programmierung parodiert Matthias Senkel in seinem Roman „Dunkle Zahlen“. Offenbar in BASIC ausgeführt, lässt sich der Code mit etwas Anstrengung sogar nachvollziehen.

TECHNISCHE STIMME 10 nach ca. 10 Zeilen unter Musik ausfaden

```
10 ON i GOTO 20, 30, 40, 50, 60, 70
20 ON j GOTO 21, 22, 23, 24
21 PRINT "A"; : GOTO 80
22 PRINT "B"; : GOTO 80
23 PRINT "C"; : GOTO 80
24 PRINT "D"; : GOTO 80
30 ON j GOTO 31, 32, 33, 34
31 PRINT "E"; : GOTO 80
32 PRINT "F"; : GOTO 80
33 PRINT "G"; : GOTO 80
34 PRINT "H"; : GOTO 80
40 ON j GOTO 41, 42, 43, 44
41 PRINT "I"; : GOTO 78
42 PRINT "J"; : GOTO 80
43 PRINT "K"; : GOTO 80
44 PRINT "L"; : GOTO 80
```

[56] Ullman S. 27f.

```
50 ON j GOTO 51, 52, 53, 54, 55
51 PRINT "M"; : GOTO 80
52 PRINT "N"; : GOTO 80
53 PRINT "O"; : GOTO 80
54 PRINT "P"; : GOTO 80
55 PRINT "Q"; : GOTO 80
60 ON j GOTO 61, 62, 63, 64, 65
61 PRINT "R"; : GOTO 80
62 PRINT "S"; : GOTO 80
63 PRINT "T"; : GOTO 80
64 PRINT "U"; : GOTO 80
65 PRINT "V"; : GOTO 80
70 ON j GOTO 71, 72, 73, 74
71 PRINT "W"; : GOTO 80
72 PRINT "X"; : GOTO 80
73 PRINT "Y"; : GOTO 80
74 PRINT "Z"; : GOTO 78
78 if k = 8 then 79 ELSE 80
79 PRINT " ";
80 i = INT(RND(1) * 6) + 1
81 k = k + 1
82 j = INT(INT(k * RND(1)) / 2) + 1
83 IF k > 9 THEN 84 ELSE 85
84 k = 0
85 GOTO 10
99 END57
```

Musik 2'21 „Honey dripper" von Oscar Peterson Trio
Komponist/Texter: Joseph Christopher Liggins

TECHNISCHE STIMME 11
set Variable „Digitalisierungsangst", define „Digitalisierungsangst"

S3-21 Krakenbürger 0'42
Es ist halt irgendwie wie der Weltuntergang!

ERZÄHLERIN
Die Techniksoziologin Fiona Krakenbürger:

[57] Senkel S. 372f.

S3-21 Krakenbürger 0‘42
So: „Das kommt! Und das wird richtig schlimm! Wir müssen jetzt Lösungen finden!“ Ne – es geht darum, wer macht denn diese Technologien? Das sind Instrumente, die verwendet werden, leider oftmals im kapitalistischen System, um bestimmte Kapitalanreicherungen noch größer zu machen. Und es ist halt die Frage: Wer gestaltet das denn? Und kann man da nicht dafür sorgen, dass mehr verschiedene Menschen mit unterschiedlicheren Vorstellungen und Wertvorstellungen vor allem auch diese Technologien prägen? Und ich glaub, wir müssen da halt auch wieder so’n bisschen wegkommen von diesen „die Allmacht der Algorithmen“ und so weiter und so fort. Hin zu: „Technologien sind Werkzeuge, die genutzt werden, für bestimmte Zwecke. Punkt.“

S3-22 Kappes 0’54
Ich unterscheide immer – sage ich mal – Handhabungs- und Anwendungswissen von Einordnungswissen, und dann tatsächlich ner eher politisch-philosophischen Dimension.

ERZÄHLERIN
Christoph Kappes, Programmierer und Jurist.

S3-22 Kappes 0’54
Politisch-Philosophisch ist das schon hilfreich zu wissen oder sogar nötig zu wissen, wie der Rechner im Detail funktioniert, weil erst dadurch verstehe ich, was eigentlich Determinismus ist, beispielsweise. Also ich traue der Maschine eigentlich auch nicht so viel zu, weil ich sie noch gelernt hab als eine, die nur Nullen und Einsen hin- und herschiebt, und ich das auch verstehe. Also das hat nichts Mystisches! Das hat nichts Geheimnisvolles, aber auch nicht, was mich gefährdet. Also es macht mir auch keine Angst, ja? Für mich ist das wie die Bedienung von nem … weiß ich nicht, nem Schachspiel, wo ich einmal lernen muss: Welche Figuren gibt es, welche Farben hat das Brett? Und dann muss ich noch wissen, was die Figuren dürfen (lacht), sonst wird das nichts! Aber mit den drei einfachen Bausteinen kann ich Schachspielen, ohne dass ich da Phantasien über mystische Geschichten bekomme,

ERZÄHLERIN 23
Was bei Fiona Krakenbürger wie bei Christoph Kappes durchklingt, ist die Beherrschbarkeit des digitalen Universums, das uns keine Angst machen sollte. Längst hat es zwar unser reales Universum durchdrungen – manche sagen: unterjocht –, doch stehen wir dem nicht hilflos gegenüber. Einer, der diese Position früh anzweifelte, war der Literatur- und Medienwissenschaftler Friedrich Kittler. Er sah paradoxerweise die größte Gefahr des Computers in der Entwicklung zu immer mehr Bedienungskomfort hin, bei der Hardware wie bei Programmiersprachen: Je umgänglicher die Maschinen,

desto unzugänglicher würden sie zugleich.

ZITATSPRECHER 20
„In der guten alten Zeit, als Mikroprozessorpins noch groß genug für schlichte Lötkolben waren, konnten auch Literaturwissenschaftler mit dem Intel-Prozessor 8086 anstellen, was sie wollten.“58

TECHNISCHE STIMME 12
Source: Friedrich Kittler: „Protected Mode“, Aufsatz von 1991.

ZITATSPRECHER 21
„Diese guten alten Zeiten sind unwiderruflich vergangen. Unter Stichwörtern wie Benutzeroberfläche, Anwenderfreundlichkeit oder auch Datenschutz hat die Industrie den Menschen mittlerweile dazu verdammt, Mensch zu bleiben. (…) Je höher und komfortabler die Hochsprachen, desto unüberbrückbarer ihr Abstand zu einer Hardware, die nach wie vor alle Arbeit tut. (…) Während es auf der einen Seite, in Kenntnis von Codes oder Algorithmen, prinzipiell machbar bleibt, Anwendersoftware (…) zu schreiben, wird es auf der anderen und benutzerfreundlich kaschierten Seite nachgerade unmöglich, vom Fertigprodukt auf seine Produktionsbedingungen zurückzuschließen oder diese Bedingungen gar zu verändern.59

ERZÄHLERIN 24
Was Friedrich Kittler – der mit Begeisterung an Hardware herumlötete und sich selbst die Programmiersprache Assembler beigebracht hatte –, was der Literaturwissenschaftler schon damals aufziehen sah, war die Spaltung der Welt in jene elitäre Kaste, die Code versteht, und jene Menschenmasse, die ihn ausbaden muss. Dabei prognostizierte er eine Untertanenmentalität der Konsumenten, die umso stärker ausfalle, je weniger Kenntnisse IT-Geräte zu ihrer Benutzung verlangten. Tatsächlich ist dieser Zustand heute längst erreicht. Man könnte von einer „programmierten Gesellschaft“ sprechen, weil so viel subtile Verhaltenssteuerung irgendwo in einem Code niedergelegt ist. Martin Burckhardt sieht das ähnlich wie Kittler:

S3-23 Martin Burckhardt 0‘24
Eine programmierte Gesellschaft ist plötzlich wirklich die schriftgewordene Gesellschaft! Und deshalb ist es wichtig, sich selbst darüber klar zu werden: Was ist die Ethik? Was sind die Gedanken, die in diese Form der Schriftproduktion eingehen?

---

58 Friedrich A. Kittler: „Protected Mode“ in: „Draculas Vermächtnis“. Reclam Leipzig 1993. S. 209f.
59 a.a.O.

Das ist im Grunde genommen, wenn man so will, so was wie eine Art von eingestandenes Analphabetenwesen. Also ich fühle mich gar nicht verantwortlich für das Alphabet! Das ist wirklich so eigentlich Defätismus, das ist Preisgabe der Gestaltung der Gesellschaft.

ERZÄHLERIN 25
Umgekehrt gestalten jene die Gesellschaft, die das ihr zugrundeliegende Alphabet beherrschen. Das ist beileibe keine neue Entwicklung, sondern beginnt schon im Vor-Computer-Zeitalter mit der Lochkarte, womit wir zum Ende dieser Langen Nacht an ihren Ausgangspunkt zurückkehren.

TECHNISCHE STIMME 13
goto Startpunkt „Lange Nacht der Programmiersprachen“, repeat Peter Fuß.

S3-24 Fuß 0‘20
… als ich mit meinem Vater 1962 in einem Großmarkt beim Einkaufen war, wurde das dort so gehandled, dass wenn er einen Artikel aus dem Regal nahm, auf seinen Einkaufswagen, musste er die dazu gehörige Lochkarte mitführen, und dieser Stapel an Karten wurde dann vor …

ERZÄHLERIN 26 darüber
Doch was dort eine harmlose Technik zur Verbesserung des Warenwirtschaftssystems gewesen war, hatte nur wenige Jahre zuvor seine rabenschwarze Nachtseite bewiesen. Am Fall der von IBM aufgekauften, weltweit agierenden Hollerith-Gesellschaft lässt sich beispielhaft sehen, dass Technik Moral nicht kennt – weswegen wir sie im Umgang mit Technik beweisen müssen. Das Quartett aus Tabelliermaschine, Lochkartensortierer, -locher und -leser bleibt ein Mahnmal problematischer Programmierbarkeit, wie Peter Fuss erklärt.

S3-25 Fuß 0‘16
Also man spricht dieser Maschine auch zu, dass sie ein sehr schwarzes Kapitel auch der deutschen Geschichte hier hilfreich bedient hat, nämlich in … ja in Konzentrationslagern, um mit Häftlingen hier Auswertungen und so weiter …

ERZÄHLERIN
Und Martin Burkhardt schließt:

S3-26 Martin Burckhardt 0‘49
Die Hollerithsche Maschinengesellschaft wurde zu eigentlich fast einer Horrorunternehmung. In den 30er… in den 20er-Jahren in Deutschland etabliert, waren die ersten Aficionados und begeisterten Lochkartenfans die Nazis natürlich. Die

haben sofort begriffen, dass man auf diese Art und Weise Gesellschaftskontrolle machen könnte und haben schon mit der ersten Volkszählung 1933/34 – ersten preußischen Volkszählung – haben begonnen, wirklich ganz Deutschland zu kartieren. Also wusste man zum Beispiel, dass in Berlin-Wilmersdorf der Bezirk mit der höchsten „Judendichte“ ist. Das waren die Szenarien, mit denen man begonnen hat, dann auch tatsächlich den Holocaust zu organisieren: Jedes Konzentrationslager hatte eine Hollerith-Maschine, und selbst der Code, der auf dem Unterarm des Konzentrationslagerhäftlings war, das war die Nummer seiner Hollerith-Karte!

Musik 3’34 (blendbar)
Titel: „Papa J.” Zakarya / Yves Weyh
Komponist/Texter: Yves Weyh

Absage

Musik

Musikliste

1. Stunde

Titel: Konlied MX
Länge: 06:45
Interpret: Autechre
Komponist: Rob Brown, Sean Booth
Label: WARP Best.-Nr: WARPCD82
Plattentitel: CD: Routine

Titel: Model
Länge: 03:42
Interpret: Balenescu Quartet
Komponist: Ralf Hütter, Karl Bartos
Label: Mute Records Best.-Nr: INT846.887
Plattentitel: Possessed

Titel: Popcorn
Länge: 01:34
Interpret: Hot Butter
Komponist: Gershon Kingsley
Label: SONY MUSIC MEDIA Best.-Nr: 88697942132

Titel: Louis XIV´s
Länge: 00:56
Interpret: Murcof
Komponist: Fernando Corona
Label: Leaf Label
Plattentitel: The Versailles Sessions

Titel: Spieluhr Polka Bearbeitung für Klavier, op. 9 Nr. 3
Länge: 03:12
Solist: Rudolf Buchbinder (Klavier)
Komponist: Otto Schulhof
Label: TELDEC CLASSICS Best.-Nr: 3984-26865-2

Titel: Move 4 (Chair)
Länge: 01:36
Interpret und Komponist: Achim Wollscheid
Label: FORCE INC MUSIC WORKS Best.-Nr: MP3CD 61
Plattentitel: Modulations & Transformations 4

Titel: Dreamhorse
Länge: 03:49
Interpret und Komponist: Arild Andersen
Label: ECM-Records Best.-Nr: 1774448
Plattentitel: Live at Belleville

Titel: Service III
Länge: 01:13
Interpret: Kovac, Roland Orchester
Komponist: Roland Kovac
Label: edel records Best.-Nr: 0209741MSW
Plattentitel: Trip to the Mars

Titel: Folia Part I
Länge: 01:32
Interpret und Komponist: Andy Moor, Yannis Kyriakides
Label: Unsounds
Plattentitel: Folia

Titel: Pneuma II
Länge: 00:52
Interpret: Biosphere
Komponist: Geir Jenssen
Label: TOUCH UK Best.-Nr: Tone 38
Plattentitel: Wireless - Live at the Arnolfini, Bristol

Titel: New Math
Länge: 03:00
Interpret und Komponist: Tom Lehrer
Label: RHINO Best.-Nr: R279831
Plattentitel: The Remains of Tom Lehrer

Titel: Hifi
Länge: 00:17
Interpret: Kodexx Sentimental
Komponist: Jörg Ritzenhoff
Label: POISE Best.-Nr: MPO02
Plattentitel: Das digitale Herz

Titel: Berlin
Länge: 00:35
Interpret und Komponist: Alva Nato, Ryuichi Sakamoto
Label: raster-noton Best.-Nr: r-n65
Plattentitel: Insen

Titel: That's mathematics
Länge: 01:33
Interpret: Tom Lehrer
Komponist: Thomas Andrew "Tom" Lehrer
Label: RHINO/WARNER BROS. Best.-Nr: R 2 79831

Titel: Netlon Sentinel
Länge: 02:25
Interpret: Autechre
Komponist: Rob Brown, Sean Booth
Label: WARP Best.-Nr: wapep7cd
Plattentitel: CD: ep7

Titel: Cheeta2
Länge: 02:46
Interpret: Aphex Twin
Komponist: Richard David James
Label: WARP Best.-Nr: WAP391CD
Plattentitel: Cheetah

2. Stunde

Titel: Sound Shadows
Länge: 00:51
Interpret und Komponist: Sorrel Hays
Label: Wergo Best.-Nr: WER6307-2
Plattentitel: Riverrun

Titel: Mi corazón
Länge: 02:30
Interpret: The Brandt Brauer Frick Ensemble
Komponist: Daniel Brandt, Jan Brauer, Paul Friedrich Frick
Label: Studio K7 Best.-Nr: !K7286CD
Plattentitel: Mr. Machine

Titel: Glimpse
Länge: 07:31
Interpret: Coil / ELpH
Komponist: Steven Stapleton, John Balance
Label: Eskaton Best.-Nr: ESKATON 002
Plattentitel: Born again Pagans

Titel: Second bad vilbel
Länge: 01:45
Interpret: Autechre
Komponist: Rob Brown, Sean Booth
Label: WARP Best.-Nr: wap64cd
Plattentitel: CD: Anvil Vapre

Titel: Sim Gishel
Länge: 02:40
Interpret: Autechre
Komponist: Rob Brown, Sean Booth
Label: WARP Best.-Nr: warpcd128
Plattentitel: CD: Confield

Titel: Modifié
Länge: 00:18
Interpret: Biosphere
Komponist: Geir Jenssen
Label: TOUCH UK Best.-Nr: Tone 38
Plattentitel: Autour de la Lune

Titel: Du kleine Löterin
Länge: 01:12
Interpret: Wiglaf Droste
Komponist: Peter Janssens
Label: HÖRKUNST BEI KUNSTMANN Best.-Nr: 5238-2
Plattentitel: Das Konzert

Titel: Computer No. 9
Länge: 00:18
Interpret: Andy Fisher
Komponist: Carl Birth, Christian Dornaus
Label: Bear Family Records Best.-Nr: BCD 16163

Titel: Lullys Turquerie as Interpreted by an advanced Script
Länge: 02:34
Interpret: Murcof
Komponist: Fernando Corona
Label: Leaf Label
Plattentitel: The Versailles Sessions

Titel: Erweiterter Schwitters
Länge: 00:10
Interpret und Komponist: Stephan von Huene
Label: Wergo Best.-Nr: WER6307-2
Plattentitel: Riverrun

Titel: Hanging upside down
Länge: 02:55
Interpret: Angel Fernandez
Komponist: David Byrne, Angel Fernandez
Label: Mute Records Best.-Nr: INT846.887
Plattentitel: Possessed

Titel: Where are you?
Länge: 02:53
Interpret: Coil
Komponist: John Balance, Peter Christopherson, Thighpaulsandra
Label: Chalice Best.-Nr: Graal CD 005

3. Stunde

Titel: Character IV
Länge: 00:44
Interpret: Zakarya / Yves Weyh
Komponist: Yves Weyh
Label: Tzadik Best.-Nr: TZ 8110
Plattentitel: 413 A

Titel: Ballet of the chicks in their shells (9)
Länge: 01:05
Interpret: Isao Tomita
Komponist: Modest Mussorgskij
Label: RCA Records Label Best.-Nr: APL1-0838
Plattentitel: Pictures at an Exhibition

Titel: Biking is better
Länge: 01:32
Interpret: Wintergatan
Komponist: Martin Molin
Label: Sommarfagel Records Best.-Nr: SOM105
Plattentitel: Wintergatan

Titel: Live Acid Jam
Länge: 01:16
Interpret: Kid 606
Komponist: Miguel Manuel De Pedro
Label: VALVE RECORDS Best.-Nr: V66
Plattentitel: Who still kill sound?

Titel: Total recovery is possible
Länge: 00:29
Interpret: Kid 606
Komponist: Miguel Manuel De Pedro
Label: Ipecac Recordings Best.-Nr: IPC-46
Plattentitel: Kill sound before sound kills you

Titel: Morning
Länge: 00:35
Interpret und Komponist: Alva Nato, Ryuichi Sakamoto
Label: raster-noton Best.-Nr: r-n65
Plattentitel: Insen

Titel: Rosace 3
Länge: 03:19
Interpret und Komponist: François Bayle
Label: ELLIPSIS ARTS Best.-Nr: 3670

Titel: En Phase / Hors Phase from Dedans Dehors
Länge: 01:21
Interpret und Komponist: Bernard Permegiani
Label: ELLIPSIS ARTS Best.-Nr: CD3670
Plattentitel: OHM - The early Gurus of Electronic

Titel: Colossus
Länge: 01:39
Interpret: Orchester Roland Kovac
Komponist: Roland Kovac
Label: MPS Best.-Nr: 0209741MSW
Plattentitel: Trip to the Mars

Titel: Tewe
Länge: 00:14
Interpret: Autechre
Komponist: Rob Brown, Sean Booth
Label: ROUGH TRADE Best.-Nr: RTD 126.3256.2
Plattentitel: Ciastic Slide

Titel: Honey dripper
Länge: 02:24
Interpret: Oscar Peterson Trio
Komponist: Joseph Christopher Liggins
Label: Verve Best.-Nr: 521 440-2
Plattentitel: Night Train

Titel: Appalachian grove I
Länge: 00:55
Interpret und Komponist: Laurie Spiegel
Label: ELLIPSIS ARTS Best.-Nr: 3670

Titel: Nieder mit IT
Länge: 06:19
Interpret: Pigor, Eichorn & der Ulf
Komponist: Thomas Pigor
Plattentitel: Update. Zungenschlag
Mit Axel Naumer, Pigor, Eichhorn & Ulf, Thomas C. Breuer, Rosemie Warth und der Band Schlag auf Schlag mit Matthias Dörsam, Wolf Mayer, Stephan Schmolck, Dirik Schilgen