Informatik Spektrum (2020) 43:245–251
https://doi.org/10.1007/s00287-020-01270-7

HAUPTBEITRAG

Informatikpionierleistungen aus dem deutschsprachigen Raum – Eine Schatzsuche!

Axel Lehmann · Peter C. Lockemann[1] · Jürgen Nehmer

Online publiziert: 12. Mai 2020

Zusammenfassung
Die Gesellschaft für Informatik ist 50 Jahre alt geworden – nur wenig jünger als die Informatikentwicklung in Deutschland. Sie hat daher diese Entwicklung in Lehre, Forschung und Praxis begleiten können. Und die konnte durchaus auch im Weltmaßstab mithalten, sie hat, auch wenn dies vielen nicht mehr bewusst ist, eigene bahnbrechende Leistungen hervorgebracht. Der Beitrag soll einige dieser Pionierleistungen ins Gedächtnis zurückrufen. Die Beispiele decken ein breites Spektrum ab: Entwicklung kompletter Rechensysteme mit Hardware und Software, Programmiertechniken, Theorie- und Methodenentwicklung, Informatikanwendungen. Letztlich soll der Beitrag dazu anregen, einer Suche nach weiteren derartigen „Schätzen“ nachzugehen. Dies ist eine Aufgabe, der sich derzeit auch die GI-Fellows widmen.

Einführung

Die Gesellschaft für Informatik ist 50 Jahre alt geworden – das ist immerhin schon mehr als ein halbes Menschenleben. Die Informatik selbst – oder doch das, was man heute darunter versteht – ist aber noch wesentlich älter. Sie geht auch in Deutschland auf die Dreißigerjahre zurück, und das ist nun doch ein ganzes Menschenleben. Für die junge Generation ist dies schon fast eine Ewigkeit. Was Wunder, dass für sie aller Fortschritt, alles was wesentlich unser digitales Leben bestimmt, aus den USA oder demnächst wohl auch aus China zu kommen scheint. In den USA etwa stehen dafür weltmarktbeherrschende Unternehmen wie INTEL, Microsoft, Google, Apple, Amazon u. a. mit den als Quasi-Standards geltenden Hard- und Softwareplattformen wie z. B. INTEL Core, Windows, Android, IOS.

Aber war das schon immer so? Oder stimmt der Eindruck überhaupt? Bei allem Respekt vor den unbestreitbaren wirtschaftlichen Erfolgen der US-amerikanischen IT-Industrie sollte nicht in Vergessenheit geraten, dass diese oft auf bahnbrechenden wissenschaftlichen Informatikpionierleistungen aus dem mitteleuropäischen Raum inklusive des deutschsprachigen Raums basieren. In internationalen, vornehmlich US-amerikanischen Publikationen auf dem Informatiksektor wurde und wird dies wenig zur Kenntnis genommen und droht auch bei uns allmählich in Vergessenheit zu geraten. Das mag auch dem Umstand geschuldet sein, dass Sprachbarrieren Veröffentlichungen europäischer Wissenschaftler in englischsprachigen Fachzeitschriften erschwert haben, die deshalb wenig Verbreitung fanden. Und rührt der Eindruck nicht auch davon her, dass wir im Alltagsleben mit Plattformen aus dem Konsumentenbereich zu tun haben, während Europäer stärker auf Anwendungen im – weniger sichtbaren – industriellen Sektor gesetzt haben? Zweifellos aber hätte es Europa gutgetan, wenn es mehr Wagemut gegeben hätte, den wirtschaftlichen Erfolg von wissenschaftlichen Innovationen durch den unbürokratischen Einsatz von Risikokapital voranzutreiben.

Doch es gibt sie – Pionierleistungen der Informatik aus Europa! Und es gibt sie auch aus dem deutschsprachigen Raum, dem Wirkungsbereich der GI. Sie ausfindig zu machen, gleicht ein wenig einer Schatzsuche. Man muss wissen, wo man suchen muss, man braucht Anhaltspunkte, was man finden will – da haben die GI-Fellows wesentliche Vorarbeiten geleistet –, und schließlich muss man in der Vergangenheit graben und Zeitzeugen befragen. Und Schatzsuche hat es so an sich, dass man nicht alles findet was es wert ist, gefunden zu werden, und dass man schließlich auch nicht alles wegtragen kann.

Im vorliegenden Beitrag soll eine Reihe dieser Pionierleistungen nachgezeichnet werden – solche, die weltweit bekannt sind, solche, die wenig bekannt sind, aber die Informatikentwicklung entscheidend geprägt haben, und auch solche, des es verdient hätten, mehr Wirkung zu entfalten.

✉ Peter C. Lockemann
lockemann@kit.edu

[1] Karlsruher Institut für Technologie (KIT), 76128 Karlsruhe, Deutschland

Das geht angesichts der Vorgaben nicht ohne Beschränkung der Länge. Und – Schatzsuche! – die Auswahl mag dem einen oder anderen Leser dann auch etwas willkürlich erscheinen.

Rechnerentwicklung

Die Anfänge der deutschen Informatik (damals noch nicht so benannt) werden gemeinhin mit dem Namen Konrad Zuse verbunden. Konrad Zuse war von Hause aus Bauingenieur. Er erkannte, dass die mühseligen und langwierigen statischen Berechnungen im Bauwesen immer den gleichen Rechenschritten folgten, nur mit anderen Daten. Er hatte die Idee, diese Berechnungen zu automatisieren und entwickelte 1936–1938 den elektrisch angetriebenen mechanischen Rechner Z1 mit digitalen mechanischen Schaltungen, der als erster Rechner mit binären Zahlen arbeitete und bereits über ein Ein-/Ausgabewerk, ein Rechnerwerk, ein Speicherwerk und ein Programmwerk verfügte und vor allem mit Gleitkommazahlen auf der Grundlage von Mantisse und Exponent umgehen konnte. Aufgrund der Unzuverlässigkeit der Z1 baute Konrad Zuse 1941 einen in binärer Gleitkommarechnung arbeitenden Rechner mit Speicher und einer Zentralrecheneinheit aus Telefonrelais, die Z3. Berechnungen konnten in Form einer Folge von Befehlen programmiert werden, wenn auch ohne Schleifen. Die Z3 gilt heute als *erster funktionstüchtiger Computer der Welt*. Sie wurde allerdings bei einem Bombenangriff 1943 zerstört. Im Vergleich hierzu rechnete der erste elektronische Computer der Welt, die ENIAC (1946), im Dezimalsystem und die Programmierung erfolgte statisch durch Verkabelung der Komponenten. Noch gegen Kriegsende entstand eine Weiterentwicklung der Z3 auf Relaisbasis, die Z4. Sie war 1950 der einzige funktionierende Rechner in Mitteleuropa und der erste kommerzielle Rechner weltweit. Er war u. a. von 1950 bis 1955 an der ETH Zürich in Betrieb. Mit seiner Zuse KG entwickelte Konrad Zuse weitere Rechner auf modernerer Technologie basierend. Besonders erwähnenswert sind die Rechner Z22 (auf Basis von Radioröhren) und die Z23 (bereits auf Transistorbasis), die eine weite Verbreitung in deutschen Hochschulen fanden (siehe Abb. 1). Generationen von Studierenden in den Natur- und Ingenieurwissenschaften haben in den 1960er-Jahren auf diesen Rechnern ihre ersten Programmierkenntnisse erworben. Die Zuse-Rechner wurden in den 1950er- und 1960er-Jahren auch in den Landvermessungsanstalten und in der Industrie (z. B. Leitz) eingesetzt. Langfristig konnten die Zuse-Rechner aber gegen den steigenden Wettbewerb, vor allem aus den USA, nicht mehr mithalten.

Neben Konrad Zuse verdient ein weiterer deutscher Computerpionier, Heinz Nixdorf, besonders gewürdigt zu werden. Er baute 1954 den ersten Röhrenrechner in Deutschland, der für die Buchhaltung der RWE geliefert wurde. Nixdorf erkannte als erster den großen Markt in der mittelständischen Wirtschaft, der sich durch die Nutzung von Kleinrechnern direkt am Arbeitsplatz eröffnete. Vor allem sah er in kleinen Rechenmaschinen die Möglichkeit, mittelständische Unternehmen mit Buchungs- und Rechenmaschinen zu unterstützen. 1965 stellte er den ersten auf Halbleitern basierenden Tischrechner vor. Er besaß einen Magnetkernspeicher, integrierte Tastatur und eine Schreibmaschine zur Datenausgabe (siehe Abb. 2). Aus ihm ging ein Universalrechner (Nixdorf System) hervor, mit dem die Nixdorf AG in den 1970er-Jahren Marktführer in Deutsch-

Abb. 1 Der Rechner Z22. Ausstellungsstück im Zentrum für Kunst und Medien Karlsruhe. (© Konrad Zuse; Foto © ZKM | Zentrum für Kunst und Medien, Foto: Sónia Alves)

Abb. 2 Nixdorf-Magnetkontencomputer mit Lochkartenleser 820/35, 1970. Die 820-Systeme mit ihren verschiedenen Peripheriegeräten waren echte Arbeitsplatzcomputer, die Rechenkapazitäten am Arbeitsplatz des Sachbearbeiters zur Verfügung stellten. Der Drucker mit Kontenblatteinzug und Endlosformular ermöglichte Fakturieren und Buchen in einem Arbeitsgang. Mit Magnetkonten als externen Datenspeichern wurden die Ausgangsdaten automatisch und fehlerfrei vom Kontenblatt in den Computer eingelesen. Der Rechner ist ausgestellt im Heinz Nixdorf MuseumsForum, Paderborn. (Inv.-Nr.: E-1994-1645. Foto: Sergej Magel, Heinz Nixdorf MuseumsForum)

land im Bereich der *Mittleren Datentechnik* wurde. Es folgten elektronische Kassensysteme und Bankenterminals, die man heute noch antreffen kann. Für Heinz Nixdorf besonders wichtig war die große Zahl der von ihm geschaffenen Arbeitsplätze und die qualifizierte Weiterbildung, die er seinen Angestellten und Auszubildenden ermöglichte. Ursache für den Erfolg war neben der Fokussierung auf die Mittlere Datentechnik ein zusammen mit den Rechnern ausgeliefertes komplettes Dienstleistungspaket und die einfache, klar festgelegte Struktur der Nixdorf-Hardware und -Software. Mit seinen Kleinrechnern war Nixdorf fast 10 Jahre den Arbeitsplatzrechnern voraus, die ab 1980 den Siegeszug antraten.

Man kann sich heute kaum mehr vorstellen, mit welch geringen Hauptspeichergrößen Rechneranwendungen, gemessen am tatsächlichen Bedarf, über Jahrzehnte auskommen mussten. Dass Programmentwickler mit Beginn der 1970er-Jahre von der aufwendigen, fehlerträchtigen Hauptspeicherverwaltung befreit wurden, lag an der Technik des *Virtuellen Speichers*. Erstmals wurde die Idee in einer Dissertation von F. R. Güntsch 1956 vorgestellt, der sich ja später als maßgeblicher Förderer der deutschen Informatik einen Namen machte. Ein virtueller Speicher ist ein imaginärer, d. h. real nicht vorhandener Speicherraum sehr großer Kapazität, der einem einzelnen Anwendungsprogramm mit all seinen Daten bei dessen Ausführung zugeordnet wird. Die Zuordnung von Bereichen des virtuellen Speichers zu Bereichen im viel kleineren physischen Hauptspeicher bzw. in einem externen Massenspeicher (z. B. einer Festplatte) erfolgt blockweise automatisch, d. h. unsichtbar für Anwendungsprogramme. Diese Aufgabe erledigt in heutigen Rechnern mit virtueller Adressierung eine in Hardware realisierte Memory Management Unit (MMU). Ihre Kernaufgabe besteht in der Ersetzung von virtuellen Adressen durch reale Hauptspeicheradressen. Dies geschieht über Seitentabellen, die für alle Blöcke des virtuellen Speichers (den Seiten„pages“) potenziell die korrespondierenden Adressen der Blöcke im Hauptspeicher (den Kacheln, „frames“) enthalten. Wird durch den Prozessor auf eine Seite zugegriffen, für die in der Seitentabelle keine Kachel eingetragen ist, dann stößt die MMU einen per Software implementierten Seitenaustausch mit dem Hintergrundspeicher an, durch den die fehlende Seite im Hauptspeicher bereitgestellt wird.

In welchen Dimensionen F. R. Güntsch denken musste, kann man daran ermessen, dass die gängige Massenspeichertechnologie damals der Trommelspeicher war. In dem Entwurf von Güntschs Rechner bildeten 10 Trommeln mit jeweils 100 Blöcken zu je 100 Speicherworten den Massenspeicher zur Aufnahme des virtuellen Speichers mit einer Gesamtkapazität von 100.000 Speicherworten. Sein Hauptspeicher bestand gerade mal aus 600 Speicherworten. Deshalb war es noch ein großer Schritt von der ursprünglichen Idee des virtuellen Speichers in seiner Dissertation bis zur MMU in unseren heutigen modernen Rechnersystemen. Das ändert aber nichts daran, dass das Prinzip eines virtuellen Speichers in einer zwar rudimentären, aber dennoch vollständigen Version von ihm erstmals detailliert zu Papier gebracht wurde (siehe Abb. 3).

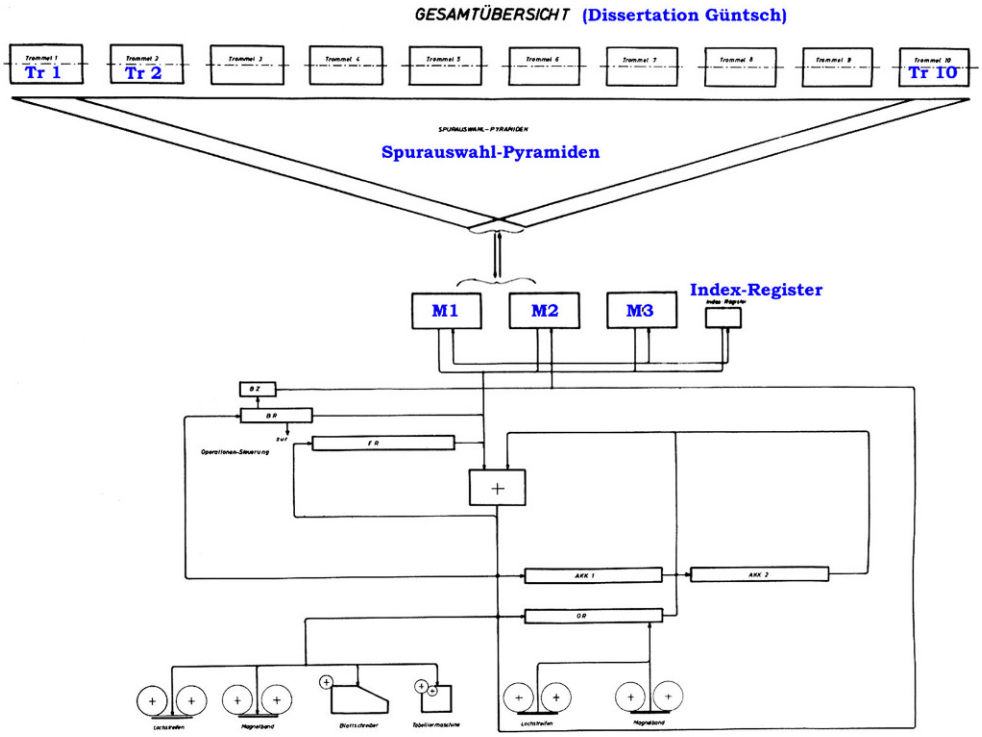

Abb. 3 Grundprinzip eines digitalen Rechnergerätes mit mehreren asynchron laufenden Trommeln und automatischem Schnellspeicherbetrieb. (Abbildung aus der Dissertation von Fritz Rudolf Güntsch. TU Berlin 1957)

Programmierung

Wurden mit den ersten käuflichen Rechnern ausschließlich einfache numerische Probleme gelöst, so begann Anfang der 1960er-Jahre eine stürmische Phase der Erschließung neuer Anwendungsfelder für Computer. Die Entwicklung geeigneter Programmsysteme, die immer umfangreicher und komplexer wurden, wäre ohne die Erfindung höherer Programmiersprachen (Higher Level Languages, HHL) wohl gescheitert. HLL befreiten Programmierer von der Notwendigkeit, Maschinensprachen zu beherrschen; sie erlaubten stattdessen, komplexe Sachverhalte kompakt und übersichtlich in strukturierter Form auszudrücken, reduzierten die Fehlerhäufigkeit in Programmen und ebneten den Weg für die Portierung von Programmen auf Rechnern unterschiedlicher Architektur.

Eine der frühen HLL, Algol 60, kam zwar diesen Idealvorstellungen bereits nahe, wies aber noch einige entscheidende Schwächen auf, die auf die mangelnde Unterstützung zur Lösung nicht-numerischer Aufgabenstellungen zurückzuführen war. Die HLL *PASCAL* des Schweizer Informatikers Niklaus Wirth vom Anfang der 1970er-Jahre beseitigte einige dieser Schwächen. Neu eingeführt wurden: Datentypen zur Zeichen- und Stringverarbeitung, Zeigertypen, zusammengesetzte Datentypen („records“). Zusammen mit den aus Algol 60 übernommenen Möglichkeiten der Schachtelung von Prozeduren und Funktionen war ein großer Schritt zur strukturierten Programmierung getan. PASCAL war die erste höhere Programmiersprache, die starke Typenbindung konsequent umsetzte und damit Programmierfehler bereits zur Compilezeit erkannte, die durch Wertezuweisung zwischen typinkompatiblen Variablen verursacht werden. Mit Fug und Recht kann man PASCAL als Meilenstein bei der Entwicklung moderner höherer Programmiersprachen einstufen.

Mit *Modula-2* als Weiterentwicklung von PASCAL führte Niklaus Wirth ein schlüssiges Modulkonzept in die Sprache ein, dessen übersichtliche, klare Ausdrucksform auch in nachfolgenden Programmiersprachen nicht mehr erreicht wurde. Module kapseln Daten ein und machen sie über eine entsprechende Schnittstelle verfügbar. Jedes Modul wird durch ein Definitionsmodul und ein Implementierungsmodul beschrieben. Im Definitionsmodul werden nur Konstanten, Typen, Variablen und Prozeduren aufgeführt, die es exportiert, d. h. für andere Module zur Verfügung stellt. Das Implementierungsmodul kapselt die nach außen abgeschottete Implementierung ein.

Mit der Einführung eines übersichtlichen Thread-Konzeptes in Modula-2 gelang Wirth der Durchbruch zur Entwicklung nebenläufiger Programmsysteme auf HHL-Sprachniveau. Dank eines übersichtlichen POINTER-Konzepts hat sich Modula-2 auch als höhere Sprache für die Entwicklung von Betriebssystemen bewährt.

Dank ihrer Klarheit, Kompaktheit und Einfachheit fanden beide Sprachen weltweite Verbreitung als Lehrsprachen in Hochschulen. Ihre innovativen Konzepte standen aber auch Pate für HHL in der industriellen Softwareentwicklung. Dass es dort zu eigenen Hochsprachen kam, lag an den zusätzlichen Anforderungen der Industrie nach Robustheit, Portabilität, Plattformunabhängigkeit, professionellem Versionsmanagement, Anwendungsunterstützung durch umfangreiche Klassenbibliotheken etc. Die enge Verwandtschaft sieht man noch dem Industriestandard Java an, der bei SUN Microsystems entstanden ist und hinter dem heute der Softwareriese ORACLE steht.

Höhere Programmiersprachen setzen allerdings effiziente Übersetzer voraus. 1959 erschien der Artikel „Sequentielle Formelübersetzung“ von Klaus Samelson und Friedrich L. Bauer, der den Übersetzungsvorgang mittels des Prinzips des „Kellerns“ revolutionierte – einer Ausprägung des Teile-und-Herrsche-Prinzips, das immer anwendbar ist, wenn die Struktur der Symbolfolge geschachtelten Charakter hat. *Sequenzielle Formelübersetzung* ist ein Verfahren zur Analyse und Transformation blockstrukturierter Programmiersprachen – damals Algol 58 bzw. dann Algol 60 – nach dem Kellerprinzip und stellt ein effizientes, fundamentales Konzept zur Übersetzung von Programmen einer höheren Programmiersprache in Maschinenprogrammen dar. Samelson und Bauer sahen die Probleme vor allem in der syntaktischen Analyse und gingen davon aus, dass die Übersetzungsprobleme nach dem Zerkleinern der Symbolfolge elementar seien, d. h. dass elementare Teilausdrücke unmittelbar infolge von Maschinenbefehlen umgesetzt werden könnten. Die Arbeit ist auch heute noch einer der internationalen Eckpfeiler des modernen Übersetzerbaus. David Gries (ein Doktorand von Herrn Bauer) konnte übrigens später zeigen, dass in der Erzeugung der Maschinenbefehle noch schwierige Probleme stecken, von denen wir heute wissen, dass sie der Komplexitätsklasse NP angehören.

Theoretische und methodische Grundlagen

Wie findet man die sprichwörtliche Nadel im Heuhaufen – oder in die Informatik übersetzt: Wie findet man ein Stück Information in Terabyte-Datenbasen? Neben den heute häufig anzutreffenden Hash-Techniken kommen zur Lösung dieses Problems seit jeher verschiedene Indexierungstechniken zum Einsatz. Eine Indexierungstechnik mit überlegenen Merkmalen wurde Anfang der 1970er-Jahre von Rudolf Bayer und Kollegen unter dem Begriff *B-Baum* entwickelt. Den Erfolg verdankt dieser Baum drei Eigenschaften: Er ist selbstorganisierend durch genau kontrolliertes Wachstum mittels Spaltung der Endpunkte und Höhenwachstum ausschließlich am Wurzelknoten, sodass immer perfekt balanzierte Bäume entstehen, die nie eine Reorganisation er-

fordern; die Zahl der Nachfolger liegt in einem festgelegten Intervall, die Balanzierung wird also durch nur partiell gefüllte Blöcke erkauft; und die Blöcke lassen sich direkt auf Blöcke auf dem Hintergrundspeicher abbilden. Der großzügige Umgang mit Speicherplatz war spätestens dann kein Problem mehr, als die Speicherkapazität immer weiter anwuchs. Seinen Durchbruch erreichte der B-Baum mit dem Forschungsprototyp System R der IBM. Der B-Baum hat inzwischen viele Varianten gefunden, aber die Ubiquität des Prinzips ist heute fast 50 Jahre später noch ungebrochen. So basiert auch das File-System von Google auf B-Bäumen.

Modellierung war schon immer und bleibt eine Schlüsseltechnologie in den Natur- und Ingenieurwissenschaften, ohne die eine mathematisch fundierte Analyse eines Problems und die Erarbeitung von Lösungskonzeptionen kaum möglich ist. Unter den zahlreichen existierenden Modellierungsmethoden ragen *Petri-Netze* durch ihre verblüffende Einfachheit heraus. Zentral ist die Idee *lokaler Zustandskomponenten* statt – wie üblich – *globaler* Zustände. Im einfachsten Fall modelliert eine Zustandskomponente eine Bedingung mit dem aktuellen Status „erfüllt“ oder „unerfüllt“. Dynamisches Verhalten entsteht, indem in einem *Schritt* einige Bedingungen zugleich ihren Status wechseln. Mit solchen – in Ursache und Wirkung begrenzten – Schritten kann man die lokale, zeitliche Unabhängigkeit von Übergängen in verteilten Systemen explizit modellieren. Das Ganze wird intuitiv überaus einfach grafisch dargestellt: Wie bei einem Brettspiel verschiebt ein Schritt einige „Token“ in einem zugrunde liegenden Graphen.

Diese Grundidee wurde vielfach verallgemeinert: An die Stelle binärer Bedingungen und einzelner, einfacher Token treten Prädikate und beliebige Datenstrukturen; alternative Schritte werden nach stochastischen Prinzipien ausgewählt; zeitliche Aspekte werden einbezogen, etc.

Für Petri-Netze gibt es zahlreiche mathematisch fundierte Techniken zum Nachweis wichtiger Eigenschaften; insbesondere gibt es einen ausdrucksstarken Kalkül zur Berechnung von Invarianten. Alle diese Techniken sind in zahlreichen Softwarewerkzeugen implementiert.

Carl Adam Petri entwickelte in den 1960er-Jahren den Formalismus ursprünglich, weil er erkannte, dass viele (insbesondere große) der in den Natur- und Ingenieurwissenschaften untersuchten/konstruierten Systeme die Eigenschaften der Nebenläufigkeit und Asynchronität aufweisen. Asynchronität bedeutet, dass ein Schritt weder durch einen Takt noch durch Zeitbedingungen eintritt, sondern ausschließlich durch Vorliegen lokaler Gegebenheiten. Nebenläufigkeit beschreibt die Eigenschaft, dass unabhängige Aktivitäten in einem System zeitlich überlappen können. Es war die große Leistung von Petri, so früh für diesen Systemtyp eine mathematisch fundierte, einfache Modellierungsmethode zu entwickeln, die sich auf das Wesentliche konzentriert.

Heute werden mit Petri-Netzen in der Informatik insbesondere verteilte Systeme, Kommunikationsprotokolle etc. modelliert und analysiert. Petri-Netz-Modelle komplexer Geschäftsprozesse und systembiologischer Prozesse sind ebenfalls wegen ihrer Analysierbarkeit beliebt.

Informatikanwendungen

Bereits Anfang der 1970er-Jahre konnten Unternehmen ihre Daten elektronisch verarbeiten. Individuelle Programmentwicklung für die Kunden war die Regel, die verfügbare Hardware waren Großrechner (Mainframes). Fünf IBM-Mitarbeiter setzten dem die Vision einer Standardanwendungssoftware für die Echtzeitverarbeitung entgegen, und weil sie bei der IBM kein Gehör fanden, gründeten sie 1972 das Unternehmen „SAP Systemanalyse und Programmentwicklung“. Ihr Produkt? Standardisierte und durch konsequente Parametrisierung erweiterbare Programme für betriebswirtschaftliche Anwendungen. Der Durchbruch gelang Anfang der 1990er-Jahre durch neue technologische Möglichkeiten als Client-Server-System mit einem zentralen Datenbankserver. 1991 präsentierte die SAP erste Anwendungen des *Systems R/3* auf der neuen Grundlage. Anstatt für Großrechner wurde die Software für kleinere Spezialrechner, Arbeitsplatzcomputer und Standardpersonalcomputer entwickelt. Damit schien das Produkt auf die Bedürfnisse des Mittelstands zugeschnitten, es interessierte aber von Anfang an auch große Konzerne und wurde zum Weltschlager. Es umfasst drei Anwendungsbereiche: Rechnungswesen, Logistik und Personalwirtschaft, erweitert um zahlreiche Branchenlösungen. Um R/3 bildete sich allmählich ein umfangreiches Ökosystem von Partnern, Beratern und Dienstleistern. Das System hat bislang jeden Wandel – technologisch etwa zur In-Memory-Technologie und zu cloudbasierten Technologien und wirtschaftlich vom Kauf- zum Mietmodell – gemeistert. SAP R/3 (aktueller Nachfolger: SAP S/4HANA) ist eines der wenigen und zudem größten Softwaresysteme aus Deutschland, die es zu einer dominierenden Rolle weltweit gebracht haben.

Weitgehend unbekannt ist dagegen, dass auch die erste Office-Suite aus Deutschland kam. Marco Börries gründete 1984 die Softwarefirma Star Division, in der die Entwicklung von StarWriter erfolgte, zunächst für ein Z80-Heimcomputersystem und dann auf MS-DOS. Später folgte die Integration der diversen Einzelprogramme zu einer Office-Suite mit allen im Büro relevanten Programmmodulen für DOS und nachfolgend bis 1995 für Windows. Sie firmierte nun unter dem Namen *StarOffice*. Sun Microsystems (später in Oracle aufgegangen) erwarb StarOffice 1999 und übernahm fortan Weiterentwicklung und Vertrieb.

Bildete SAP R/3 den größten deutschen Erfolg im kommerziellen Bereich, stellt der ab 1982 von einer Grup-

pe um Karlheinz Brandenburg entwickelte Standard MP3 einen deutschen Erfolg im Weltmaßstab im Konsumentenbereich dar. *MP3* (geläufiger Kurzname für den Standard MPEG-Audio Layer 3) ist ein Verfahren zur verlustbehafteten Kompression digital gespeicherter Audiodaten. MP3 bedient sich dabei der Psychoakustik mit dem Ziel, nur für den Menschen wahrnehmbare Signalanteile zu speichern. Dadurch wird, bei nicht (oder nur kaum) verringert wahrgenommener Audioqualität eine starke Reduktion der Datenmenge möglich. Bei einer Beispieldatenrate von 192 kbit/s, die bereits eine hohe Qualität ermöglicht, beträgt die Kompressionsrate einer MP3-Audiodatei etwa 85 % gegenüber einer unkomprimierten Audio-CD. MP3 ist das dominierende Audiocodierverfahren zur Speicherung und Übertragung von Musik auf Computern, Smartphones und im Internet und war das erste Verfahren, mit dem weitverbreitet Musik von Rechnern über Netzwerke und Internet geladen werden konnte. Die geschätzte Zahl der Geräte, die MP3 verstehen, liegt bei 8 Mrd. Spätere, auch unter wesentlicher Beteiligung der Erlanger Forschungsgruppe entwickelte Verfahren wie z. B. AAC (Advanced Audio Coding) verbessern sowohl die Tonqualität als auch die Codiereffizienz.

GI-Fellow-Projekt

Die hier aufgeführten Beiträge – so hoffen wir – überzeugen unsere Leser, dass es im mitteleuropäischen Raum wissenschaftliche und technische Leistungen der Informatik gab, die für Entwicklung alltäglich genutzter Produkte grundlegend waren und teilweise auch heute noch maßgebend sind. Es gibt aber durchaus noch mehr. Vor ca. 3 Jahren haben die GI-Fellows beschlossen, diese Schatzsuche im Rahmen eines GI-Projektes unter dem Titel: „Geniale Ideen – Meilensteine der Informatik aus dem deutschsprachigen Raum“ systematisch zu betreiben, um dem Vergessen entgegenzuwirken. Im Fokus steht aus Aufwandsgründen der deutschsprachige Raum (kurz D-A-CH), der durch die noch lebenden Zeitzeugen und deren Vernetzung in der GI leichter für die GI-Fellows aufzuarbeiten ist.

Großen Raum nahm die Debatte unter den Fellows ein, in welcher Form die Pionierleistungen aufbereitet und publiziert werden sollten; sie bestimmt maßgeblich die Zielgruppe, die mit den Publikationen erreicht werden sollen. Es kristallisierte sich schnell heraus, dass wir vornehmlich an Technik und Wissenschaft interessierte Laien, insbesondere junge Leute vor ihrer Berufswahl, ansprechen wollen. Daraus ergab sich zwangsläufig eine Publikationsform durch moderne Medien, z. B. journalistisch ansprechend aufbereitete Podcasts und Videocasts, die breit über das Internet zugänglich sind.

Zur Finanzierung der Journalisten und der medialen Aufbereitung der Pionierleistungen, vorzugsweise in Interviewform, haben die GI-Fellows Fördermittel von der Klaus-Tschira-Stiftung eingeworben. Die ersten Aufbereitungen sind inzwischen unter dem Webportal www.geniale-ideen.de verfügbar. Je nach Resonanz der Leserschaft und unter der Voraussetzung, dass es gelingt, weitere Fördermittel einzuwerben, planen die GI-Fellows weitere Informatikpionierleistungen von der Geburtsstunde der Informatik bis zum heutigen Zeitpunkt in dieser Art zu veröffentlichen.

Nachwort

Man braucht nicht unbedingt so weit zurückzugehen wie wir das taten, um Pionierleistungen aus Mitteleuropa zu identifizieren. Aber es fehlt oftmals an Risikofreude und Geschäftssinn und noch mehr an Investoren, um diesen Leistungen auch kommerziell Weltgeltung zu verschaffen. Wer erinnert sich noch, dass ebay auch Wurzeln in Deutschland hatte? Oder aktueller: Warum finden sich Implementierungen hocheffizienter Graphalgorithmen der beiden Karlsruher Forscher Dorothea Wagner und Peter Sanders bereits seit Jahren in Navigationsgeräten ausländischer Anbieter, während sie von deutschen Firmen lange nicht zur Kenntnis genommen wurden?

Und – Mitteleuropa sollte sich auf seine Stärken besinnen. Wir waren immer stark, wenn es um die Verbindung von Informatik und ihren Anwendungen ging, wenn sich interdisziplinäre Teams zusammenfinden mussten, wenn Denken in ganzen Systemen gefordert war. SAP und Nixdorf sind gute Beispiele. Da liegen gerade für die jüngere Generation die Chancen für eine erfolgreiche Informatikzukunft.

Man muss also alles tun, damit Europa, und mit ihm Deutschland, in Zukunft ein Hort bleibt, in dem Ideen gefördert und Innovationen zur Erhaltung unseres Wohlstands konsequent vorangetrieben werden. Wenn uns das gelingt, werden wir auch zukünftig erfolgreich Schatzsuche nach großen Informatikpionierleistungen betreiben können.

Danksagung Die Autoren danken Rudolf Bayer, Gerhard Goos (†), Wolfgang Karl, Klaus-Peter Löhr, Burkhard Monien, Wolfgang Reisig, Christine Regitz, Reinhard Wilhelm für ihre Anmerkungen zu den einzelnen Leistungen.

Weiterführende Literatur

Erster funktionstüchtiger Computer

1. Rojas R (Hrsg) (2010) Die Rechenmaschinen von Konrad Zuse, 2. Aufl. Springer, Heidelberg, Berlin, New York
2. Rojas R (1996) Die Architektur der Rechenmaschinen Z1 und Z3 von Konrad Zuse. Informatik-Spektrum 19(6):303–315
3. Bruderer HE (2018) Erfindung des Computers, Elektronenrechner, Entwicklungen in Deutschland, England und der Schweiz Bd. 2. De Gruyter, Berlin

Mittlere Datentechnik

4. Wikipedia (2020) Nixdorf Computer. https://de.wikipedia.org/wiki/Nixdorf_Computer. Zugegriffen: 10.05.2020
5. Kemper K (1993) Heinz Nixdorf – Eine deutsche Karriere. Verlag Moderne Industrie, Landsberg am Lech
6. Berg C (2016) Heinz Nixdorf – Eine Biographie. Schöningh, Paderborn

Virtueller Speicher

7. Güntsch F R (1957) Logischer Entwurf eines digitalen Rechengerätes mit mehreren asynchron laufenden Trommeln und automatischem Schnellspeicherbetrieb. Dissertation D83, TU Berlin
8. Kilburn T, Edwards DBG, Lanigan MJ, Sumner FH (1962) One level storage systems. IRE Trans Electron Comput 11:223–245
9. Jessen E (2004) Origin of the virtual memory concept. IEEE Ann Hist Comput 26(4):71–72
10. Denning PJ (1970) Virtual memory. ACM Comput Surv 2:153–189

Pascal und Modula-2

11. Wirth N (1971) The programming language Pascal. Acta Inform 1:35–63
12. Wirth N (1977) Modula: a language for modular multiprogramming. Softw: Pract Exper 7:3–35
13. Wirth N (1982) Programming in modula-2. Springer, Heidelberg, Berlin, New York

Sequentielle Formelübersetzung

14. Samelson K, Bauer FL (1960) Sequential Formula Translation. Commun ACM 3(2):76–83
15. Waite WM, Goos G (1984) Compiler construction. Monographs in computer science. Springer, Heidelberg, Berlin, New York

B-Baum

16. Bayer R, McCreight EM (1972) Organization and maintenance of large ordered indexes. Acta Inform 1:173–189
17. Comer D (1979) The ubiquitous B-tree. ACM Comput Surv 11(2):121–137
18. Bayer R, Schkolnick M (1977) Concurency of operations on B-trees. Acta Inform 9:1–21

Petri Netze

19. Reisig W (1982) Petrinetze – Eine Einführung. Springer, Heidelberg, Berlin, New York
20. Reisig W (1985) Systementwurf mit Netzen. Springer, Heidelberg, Berlin, New York

System R/3

21. SAP (2020) Geschichte der SAP. https://www.sap.com/corporate/de/company/history.2001-2010.html. Zugegriffen: 10.05.2020
22. Meissner G (1999) SAP – die heimliche Software-Macht. Heyne Business, München

Star Office

23. Wikipedia (2020) StarOffice. https://de.wikipedia.org/wiki/StarOffice. Zugegriffen: 10.05.2020

mp3

24. Miller F (2015) Die mp3-Story: Eine deutsche Erfolgsgeschichte. Carl Hanser, München

Axel Lehmann

Peter C. Lockemann

Jürgen Nehmer